**Daily - AEM Task**

**1. Editable/Dynamic Template:**

There are two types of content, template level content and page level content.

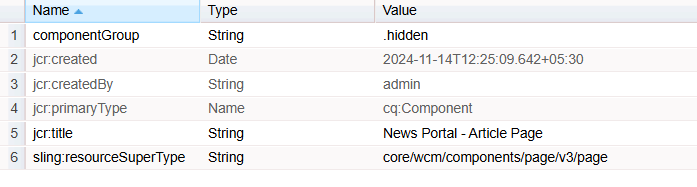
Global/common level content will be declared in Template.

Local/page level content will be declared in Page.

Page Component > Template Types > Template > Webpages

**Step 1: Page component creation**

In the newsportal website, already page component available under apps/component, just copy paste and rename.

  
**Step 2: Template Type creation**

In the newsportal website, already page template available under **/conf/newsportal/settings/wcm/template-types/page**, Just copy paste the node.

A screenshot of a computer

Description automatically generated

3 things need to be done:

Change the jcr:title of the article-page > jcr:content

A screenshot of a computer

Description automatically generated

Change the **sling:resourceType** to article-page component path on both Initial and Structure node.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**EVERY PAGE SHOULD have jcr:content and root mandatory node.**

**EVERY PAGE SHOULD CONTAIN ATLEAST ONE CONTAINER COMPONENT THAT IS ROOT. Root is pointing to container component.**

**Remove all the nodes under root on initial and structure node.**

A screenshot of a computer program

Description automatically generated

**Step 3: Template Creation**

Go to AEM Start > Tools > General > Templates > Newsportal (click on Create button at top right corner) > Select Article Template Type > Give name “Article Template”

A screenshot of a web page

Description automatically generated

A screenshot of a computer

Description automatically generated

Then Create button.

Template created under > /conf/newsportal/settings/wcm/templates/article-template

A screenshot of a computer

Description automatically generated

Whatever nodes are available under template type will be available under article template as well. So, template type is a blue print of template with predefined structure.

**Step 4: Content creation in template**

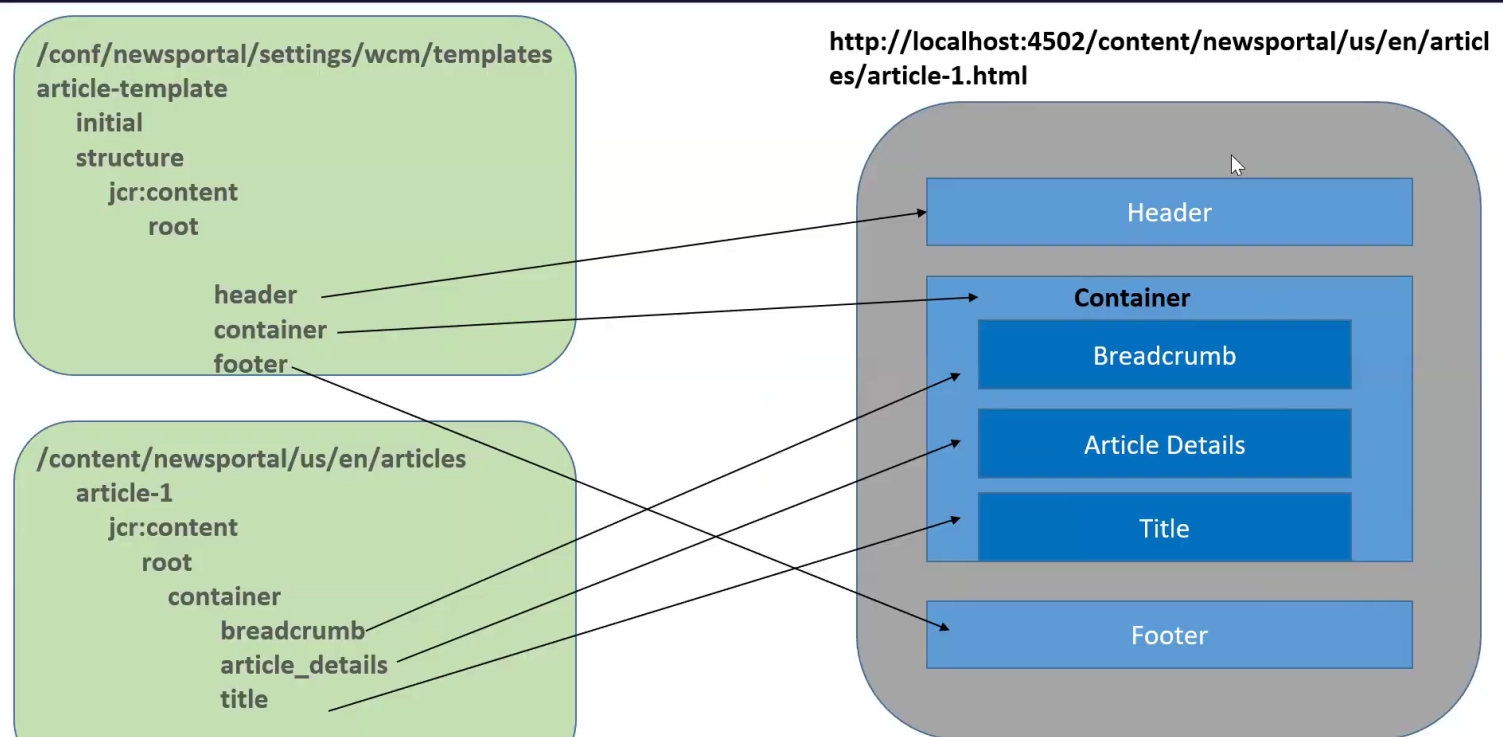
Without Enable the template, we can’t use this for page creation.

Page execution always starts from template. And structure node will be executed

Select the Article Template > Edit

A close-up of a computer screen

Description automatically generated



Default component will be authored in the Initial mode/Default mode. The connection will be executed only at the time of page creation. After that, no relationship between the initial content and template.

**Initial content is like a DISCOUNT. Discounts are avail only for new customers.**

If you made any changes in initial mode component/content, I will impact only newly created pages not existing pages.

A screenshot of a computer

Description automatically generated

Change the componentGroup in the component and make the policy update on the template.

**2. Experience Fragment (XF): Similar to Dynamic/Editable template creation.**

If there is re-usable content, In page level or template level, We can add it as a XF.

**We can use the XF for third party applications using the “plain” sector.**

destaco.com/content/experience-fragments/newsportal/us/en/site/header/product-header.plain.html

A screenshot of a computer

Description automatically generated

**3. Content Fragment (CF):**

**Step 1: Create a Content Fragment Model**

**Tools > General > Content Fragment Model > Go to Newsportal folder > Click on “Create” and provide the model name and click on “Create” button**

A screenshot of a computer

Description automatically generated

**Step 2:**

**Go to Assets console > Files > Newsportal > Create a folder “Content Fragment”**

**Go to “content-fragment” folder > click on “create” button > Contact Fragment**

A screenshot of a computer

Description automatically generated

**Select the new created content fragment model & click Next > provide a name of the content fragment > “Blog Author” > then click Create button.**

A screenshot of a computer

Description automatically generated

**Select the content fragment > click on Edit**

A screenshot of a computer

Description automatically generated

**Archiving Headless, using below 2 implementations in AEM:**

1. Asset HTTP API

localhost:4502/content/dam/newsportal/content-fragments/author-profiles

http://localhost:4502/api/assets/newsportal/content-fragments/author-profiles.json

2. GraphQL

Tools > General > GraphQL > Click on create and select newsportal.  
 **CF vs XF:**

1. We can use CF & XF for global content (common)

2. CF & XF are having variations

3. CF & XF support to export data (cf- .**json**, xf - .**html**)

**Difference:**

CF – Render structured/plain content without design

XF – Render content with design (styles, ex: header,footer)

CF – used to headless approach

XF – user to personalization (target), marketing (mobile, email social,web) global content (ex: header and footer)

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**4. Project Set-up**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Newsportal**

-header

- top-bar

- brand-logo

- navigation

- footer

-footer

-footer-list

**Web-page**

**AEM Components**

1. index.html (divided into multiple components)

**AEM Clientlibs**

2. css

3. js

4. fonts and icons

**AEM DAM**

5. assets

6. content (text+asset reference) 🡪Content Nodes (pages, template)

**Three Important Steps:**

1. upload all assets into AEM DAM (localhost:4502/assets)

2. create clientlib and upload all required js, css and fonts, resources

3. convert index.html into multiple components (.html, dialog)

**<field>** is a mandatory one child node for Multifield.

Multifield node shouldn’t have “**name**” property. It should be in “**field**” node.

**5. OSGI Components & Services:**

----------------------------------------------------------------------

OSGI Component:

- It is a Java Class

- OSGI Component can listen lifecycle events (activate, deactivate, modified)

OSGI Annotations:

-----------------

**Class Level:**

1. @Component  🡪 It will register a normal java class into OSGi component.

**Method Level:**

2) @Activate

3) @Deactivate

4) @Modified

A screen shot of a computer program

Description automatically generated

5) @Reference – We call one component into another

@Reference will assign a dynamic object that is created by osgi container to

ArticleService articleObj;

A screen shot of a computer program

Description automatically generated

If you want to use once OSGi component into another OSGi component, we have to convert that component into OSGi service component.

Eg.

@Component(service=ArticleService.class)

We can’t define OSGI service, without OSGI component.

OSGi service is an OSGi component. But OSGi component may or may not be an OSGi service.

ArticleService

@Activate

- activate()

@Deactivate

- deactivate()

@Modified

- update()

PressReleaseService

- activate

- deactivate

- update

newsportal.core.vid.jar

- activate

- Container Identify all OSGI Component java classes based on XML files inside jar file

- Register OSGI component java classes with OSGI container

- Creates Object for all OSGI Component Java classes

- Triggers activate event and execute activate method.

**IMPORTANT:**

1. All OSGI components are singleton () java classes.

2. We can't create object for OSGI components.

Browser - Client

  Request - <https://gorest.co.in/public/v2/posts>

Java

- Client

- Request

- Response

- convert response into string

**OSGi Dynamic Configuration:**

OSGi annotation:

**Class level:**

@Designate > Link the configuration class file with osgi component “ArticleService.class” file

@ObjectClassDefinition > used to create a dynamic osgi configuration class

**Property level:**

@AttributeDefinition

**Step 1:** Create a configuration class and define 3 properties that are make it dynamic.

A screenshot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated

Step 2: Link the configuration class file with osgi ArticleService.class file using @Designate

A screen shot of a computer

Description automatically generated

Step 3: We have to pass the dynamic configuration class as an argument to all lifecycle method (Activate, Deactivate, Modified)

Using the dynamic object **config**, call the method from ArticleConfiguration class. And assign it to instance variable.

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer screen

Description automatically generated

Step 4: We have to create a dynamic configuration file (JSON) under ui.config module.

ui.config

/apps

/newsportal

/osgiconfig

/config

com.bhasaka.newsportal.core.services.ArticleService.cfg.json (//cfg is a selector and json is an extension)

{ // These key : value pair taken from ArticleConfiguration.class file. Key is a dynamic property, go configuration class file take it from there. Value we can define.

“articleRestApi”:”https://www.google.com”,

“enable”: “true”,

“clientid”:”989898”

}

Terminal:

>cd ui.config

ui.config> cd mvn clean install -PautoInstallPackage

cd ../core

core> mvn clean install -PautoInstallBundle

Update Dynamic Configuratin:

----------------------------

1. CRX De

2. System Console Configuration

ui.config

/apps

/newsportal

/osgicongis

/config

com.bhasaka.newsportal.core.services.ArticleService.cfg.json

{

"articleRestAPI":"https://gorest.co.in/public/v2/posts",

"enable":"true",

"clientId":"8908978"

}