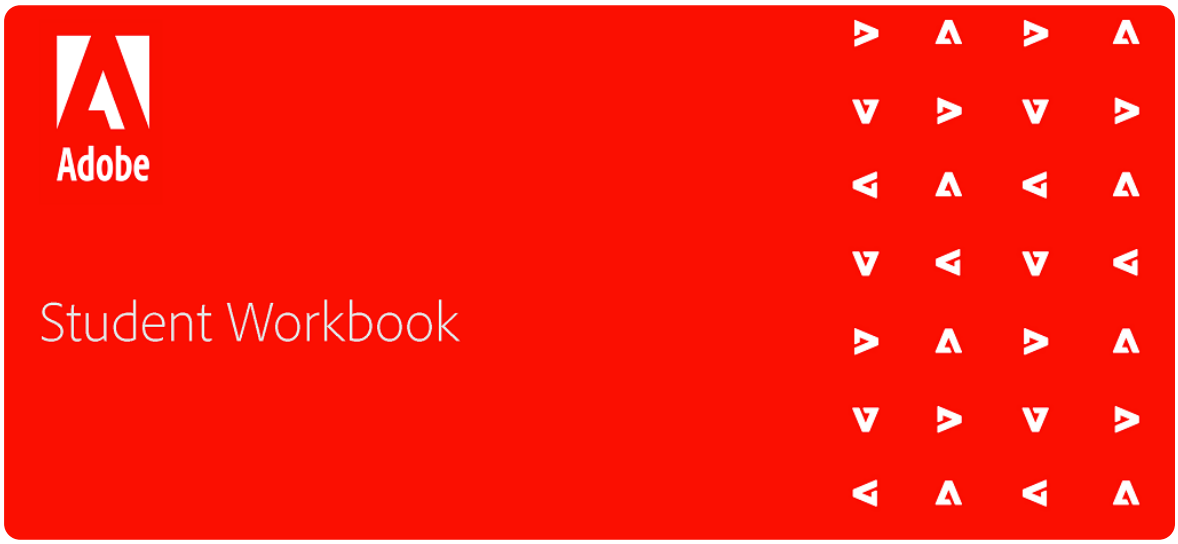
DevOps for AEM as a Cloud Service

Administer Context-aware Configurations



Find your journey at learning.adobe.com >

Contents

Administer Context-aware Configurations

- Introduction
- Context-Aware Configurations
- Exercise 1: Create and use a configuration folder
- Exercise 2: Sync the configuration folder with your Maven project
- Global Configurations
  - Assets Platform Configurations
  - Platform Search Filters
  - Workflow Configurations
  - Maintenance Configurations
  - Translation Rules in Configurations

DevOps for AEM as a Cloud Service

Adobe, 345 Park Avenue, San Jose, California 95110, USA.

10-14-2020

Introduction

Your application may need different configurations for sites, regions, and tenants for different contexts. The context - aware configurations support inheritance for the nested contexts and the global fallback values that are shared among different content resources. Adobe Experience Manager (AEM) provides global and tenant configuration options for a wide variety of features. Each context folder offers security and the ability to enable new features per context. AEM has a wide range of context configurations including Cloud Services, Editable Templates, Content Fragment Models, ContextHub segments, Workflows, Search Facets, and Metadata Schemas.

Objectives

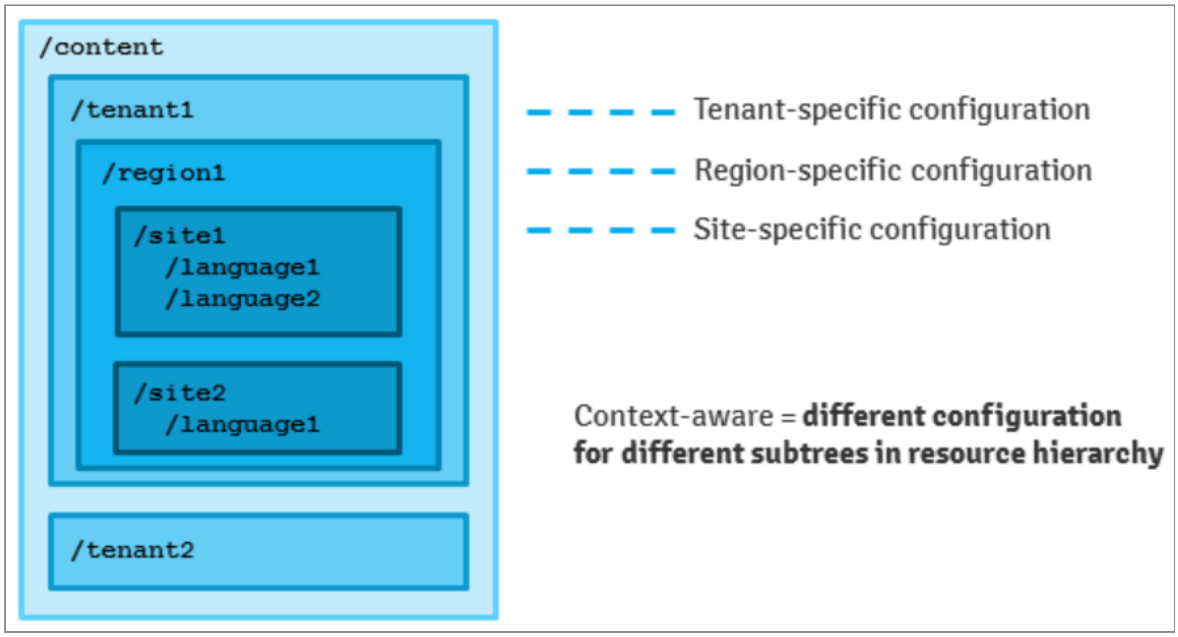
After completing this module, you will be able to:

- Describe context-aware configurations
- Create and use a configuration folder
- Sync the configuration folder with your Maven project
- Explain global configurations

Context-Aware Configurations[1]

Configurations are designed to provide the logic and structure for storing service configurations. They are related to a content resource or a resource tree such as, a website or a tenant site. These configurations are not system configuration. You can control and define the content subtrees to be the contexts in your application.

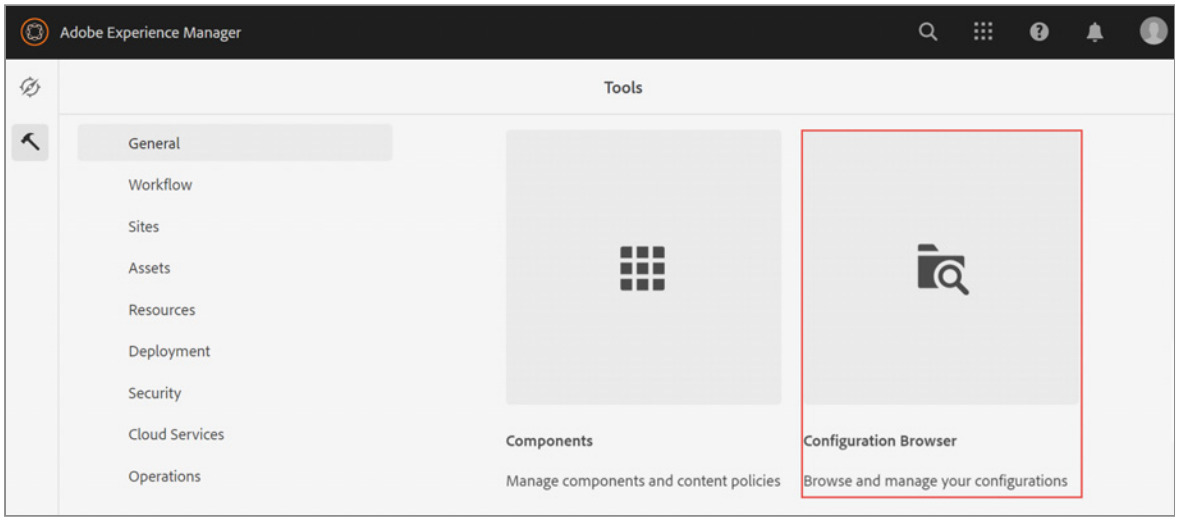
The content structure of your website may look similar to the one shown in the below screenshot:



You can get the matching configuration for each content resource without caring about it's storage or the inheritance process by using the Context-Aware Configuration Java Application Programming Interface (API).

Configuration Browser

You can access the context-aware configurations from the **Configuration Browser** console. This console is available in the **Tools** > **General** panel, as shown:



The **Configuration Browser** console enables you to browse the **/conf** folders and not the configurations within them. You can create configurations for:

- Editable Templates
- Content Fragment Models
- Cloud Configurations
- ContextHub segments

In the **Configuration Browser** console, you can view the **Effective Permissions** of a configuration and assign the following new permissions to users or groups associated with the configuration:

- Browse configurations
- Modify configurations
- Delete configurations

Implementing Context-Aware Configurations

Context-aware configurations enable you to layer a configuration across different folders, including **/libs , /apps , /conf** and subfolders under **/conf** . It supports inheritance so a customer can configure global configuration while making specific changes for each microsite.

The context-aware configuration implementation provides a set of Service Provider Interfaces (SPIs) that help overlay, enhance, or replace the default implementation and adapt to your needs.

It is recommended that you should store the context-aware configurations under **/conf**.

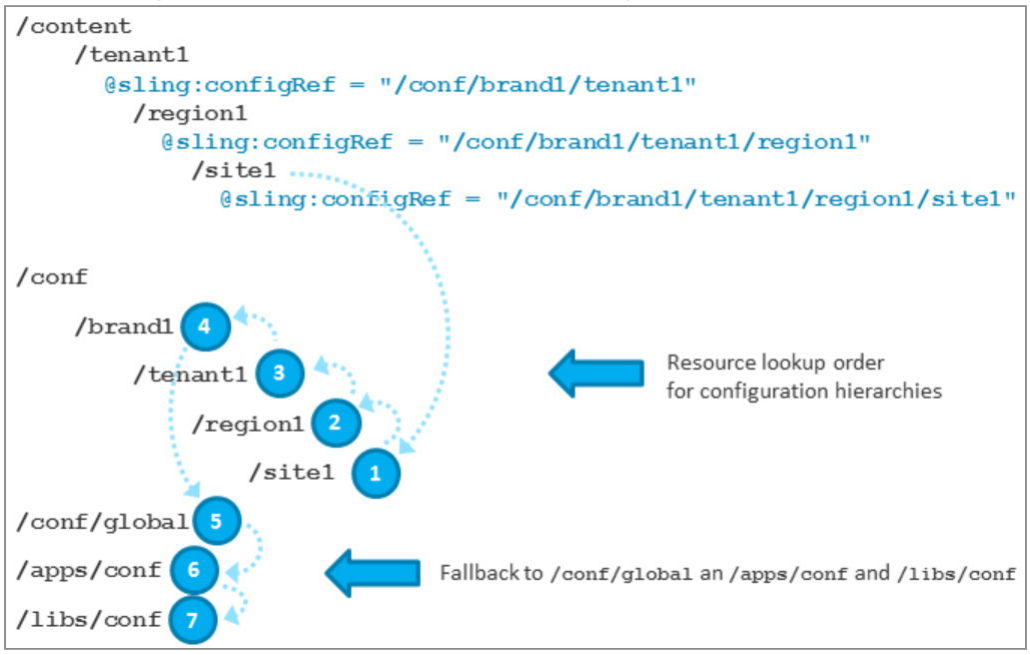
The resolution order considers the following locations:

1. **/libs/settings**
2. **/apps/settings**
3. **/conf/global/settings**
4. **/conf/tenant/settings**

By default, all configuration data is stored in **/conf**. The fallback paths are **/conf/global, /apps/conf, and /libs/conf**. These paths are configurable in the service configuration.

Configuration Lookup Order

The following illustration shows an example of configuration resource lookup:



If you get the context-aware configuration through the API for any resource below **/content/tenant1/region1/site1** is resolved in the following order:

1. **/conf/brand1/tenant1/region1/site1** - because it is referenced by **/content/tenant1/region1/site1**
2. **/conf/brand1/tenant1/region1** - because it is referenced by **/content/tenant1/region1** (parent context)
3. **/conf/brand1/tenant1** - because it is referenced by **/content/tenant1** (parent context)
4. **/conf/brand1** - because it is a parent of **/conf/brand1/tenant1**
5. **/conf/global** - because it is configured as a fallback path
6. **/apps/conf** - because it is configured as a fallback path
7. **/libs/conf** - because it is configured as a fallback path

When multiple development teams work on the same AEM environment, there is likely to be some degree of multi-tenancy.

The challenges yo face when implementing a multi-tenant environment are:

- Additional technical complexity
- Increased development overhead
- Cross-organization dependencies on shared resource
- Increased operational complexity

A multi-tenant application provides the following benefits:

- Reduced hardware costs
- Reduced time to market for the future sites
- Lower implementation costs for the future tenants
- Standard architecture and development practices across the business
- A common codebase

The following functionalities of AEM use configuration:

- Editable Templates
- Content Fragment Models
- Translation Cloud Services
- ContextHub Segments

Exercise 1: Create and use a configuration folder

Scenario: As an administrator, you want to set up folders for editable templates for template authors in AEM without using the code. You need to quickly create context-aware configurations to organize your AEM implementation in a site-specific folder structure. In this exercise, you will perform the following tasks:

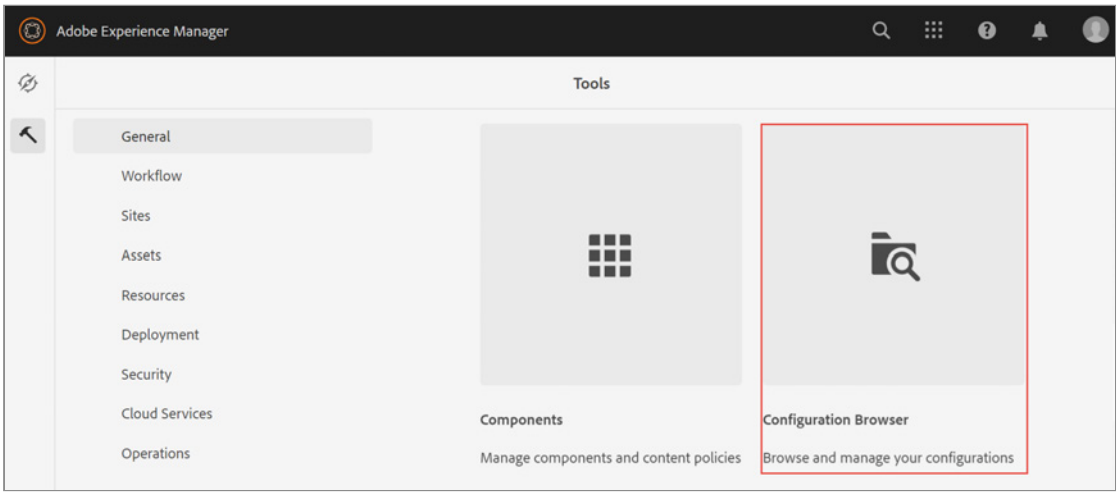
1. Create a configuration folder
2. Enable the functionality by using the Configuration Browser

Prerequisites:

- A running AEM author service

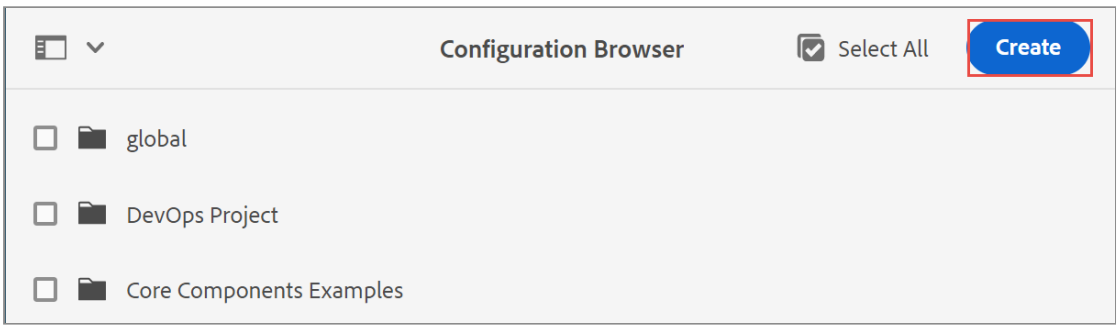
Task 1: Create a configuration folder

1. On your AEM author service, navigate to **Tools** > **General** > **Configuration Browser**, as shown:



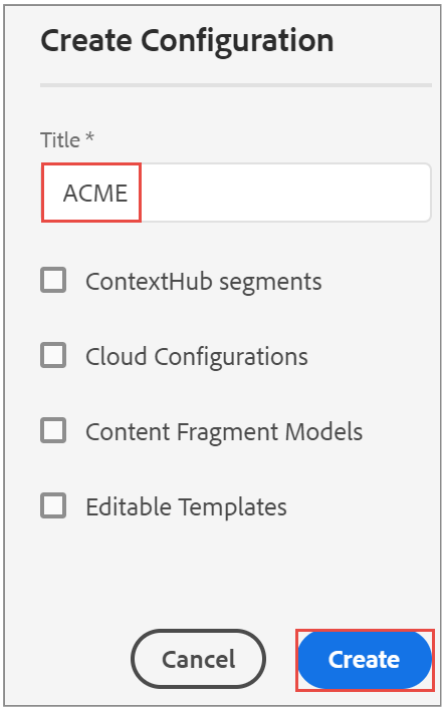
The **Configuration Browser** console opens with the current configurations that are available in AEM.

2. Click **Create** on the actions bar, as shown:



The **Create Configuration** dialog box opens.

3. Type **ACME** in the **Title** box and click **Create**, as shown:

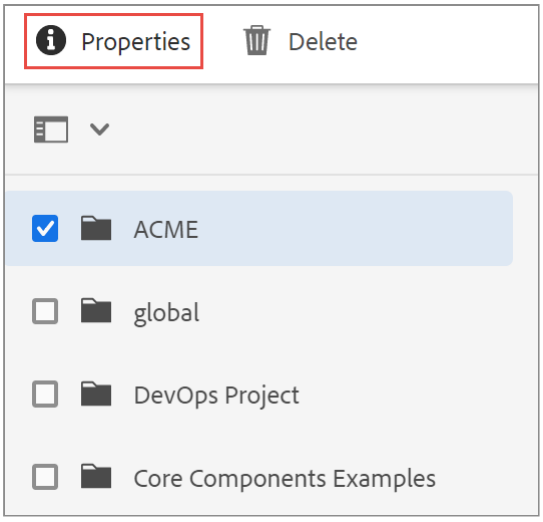


The form has been submitted successfully message appears at the bottom of the **Configuration Browser** console. This confirms that the configuration is created.

Task 2: Enable functionalities by using the Configuration Browser

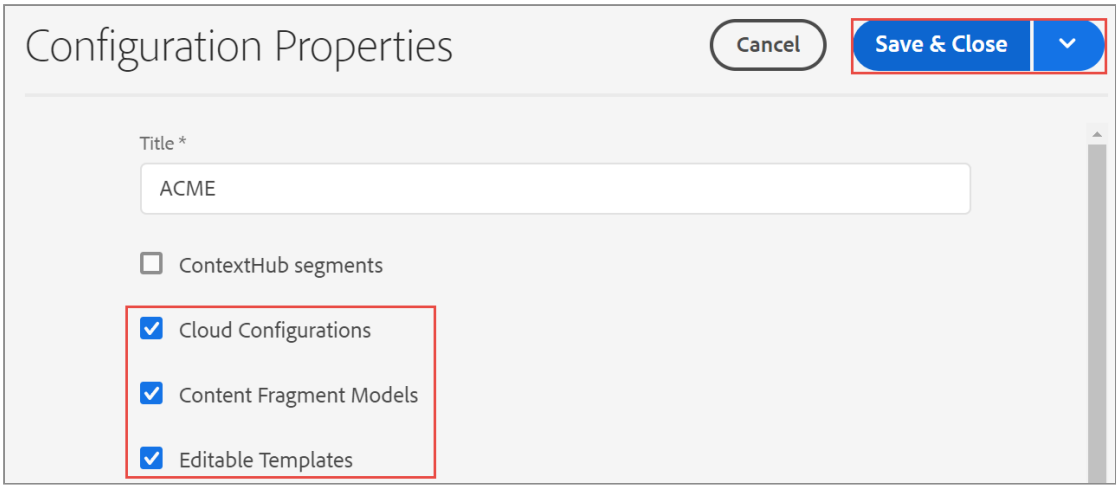
1. Ensure you are in the **ACME** folder of the **Configuration Browser** console.

2. Select the **ACME** folder and click **Properties** on the actions bar, as shown:



The **Configuration Properties** page opens.

3. Select the **Cloud Configurations**, **Content Fragment Models**, and **Editable Templates** check boxes and click **Save & Close**, as shown:

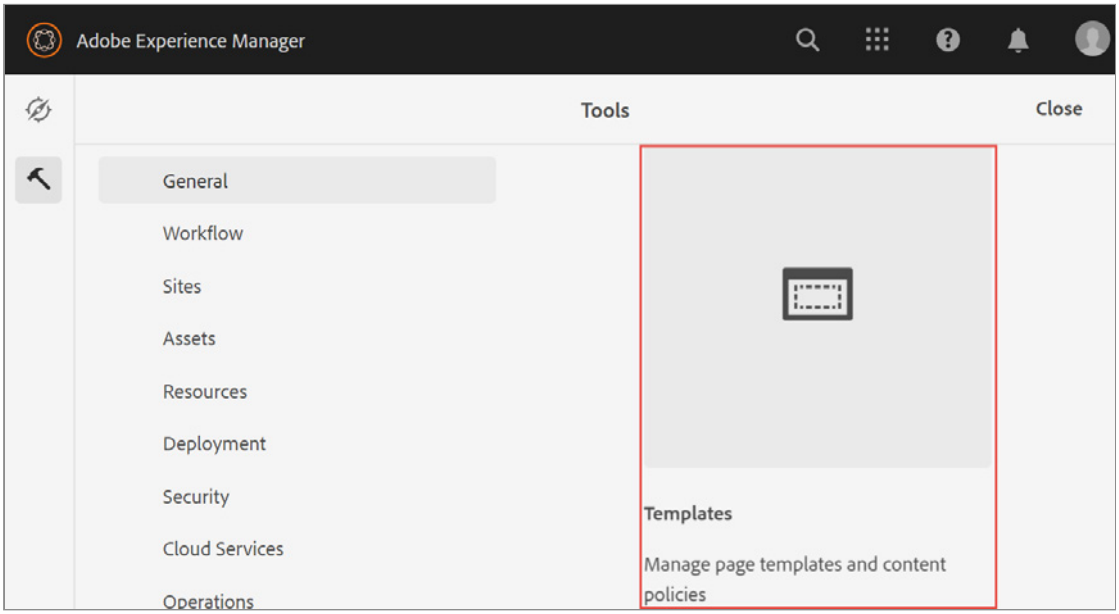


The form has been submitted successfully message appears at the bottom of the **Configuration Browser** console.

To view the configuration folder for different AEM functionalities:

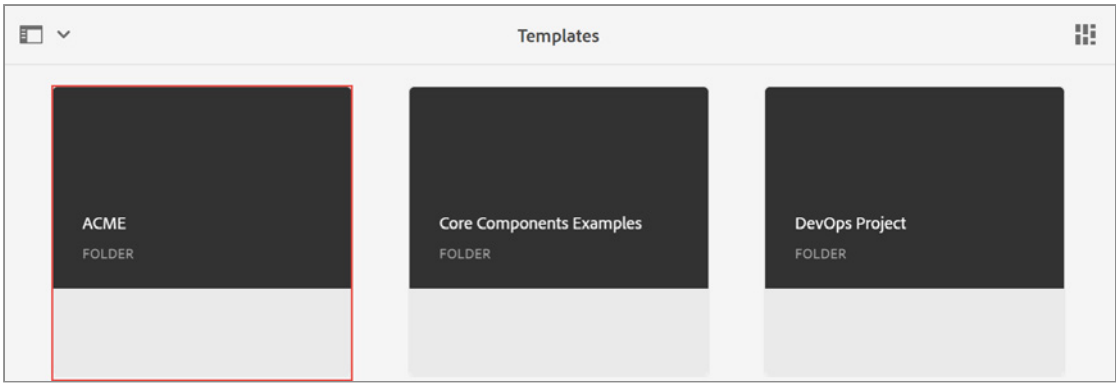
4. Click **Adobe Experience Manager** from the header bar.

5. Navigate to **Tools** > **General** > **Templates**, as shown:



The **Templates** console opens.

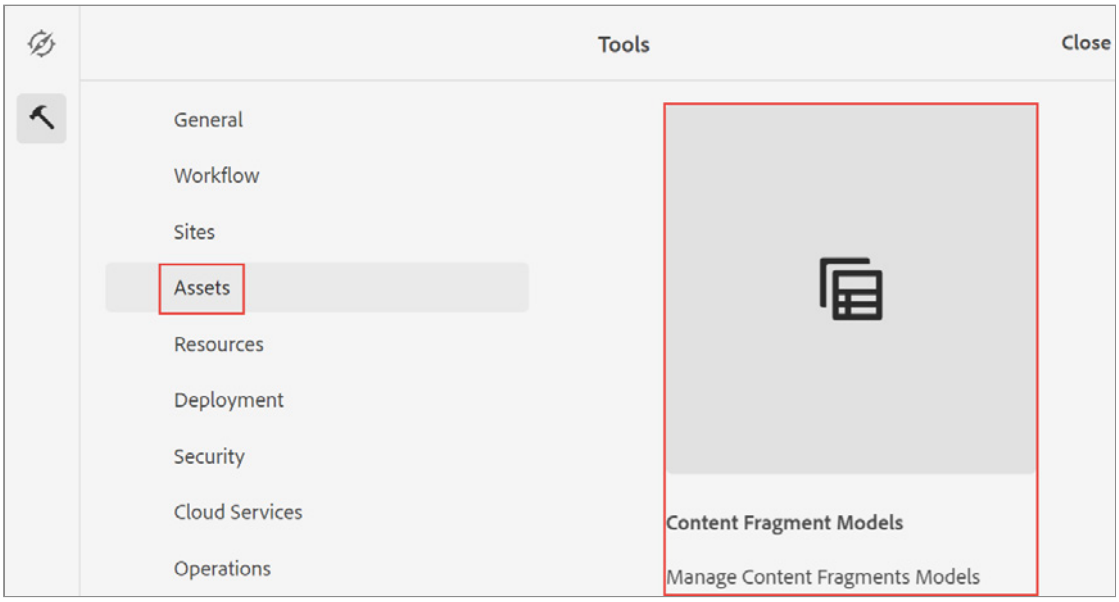
6. Notice that the **ACME** folder is created, as shown:



This folder helps create editable templates for your site implementation.

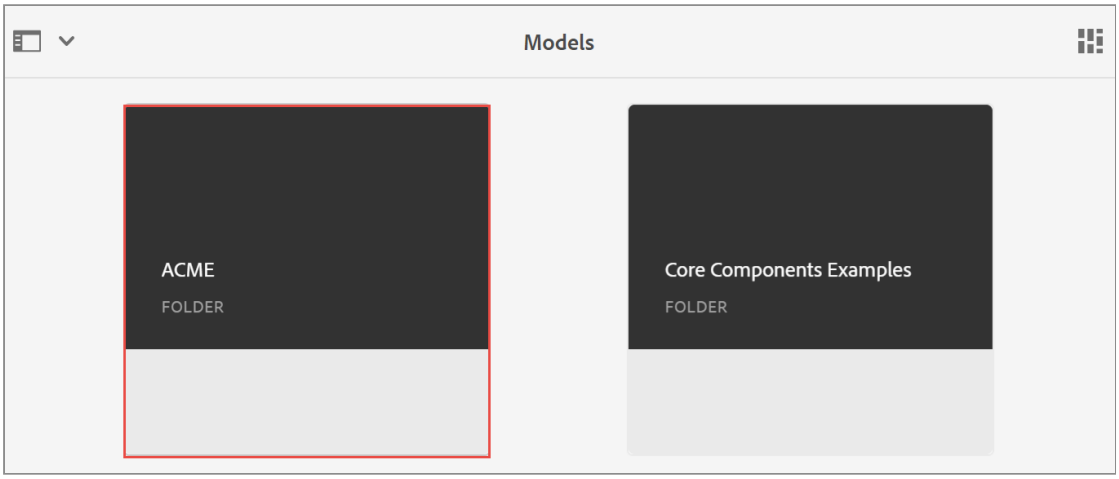
7. Click **Adobe Experience Manager** from the header bar.

8. Click **Tools** and click **Assets** > **Content Fragment Models**, as shown:



The **Models** console opens.

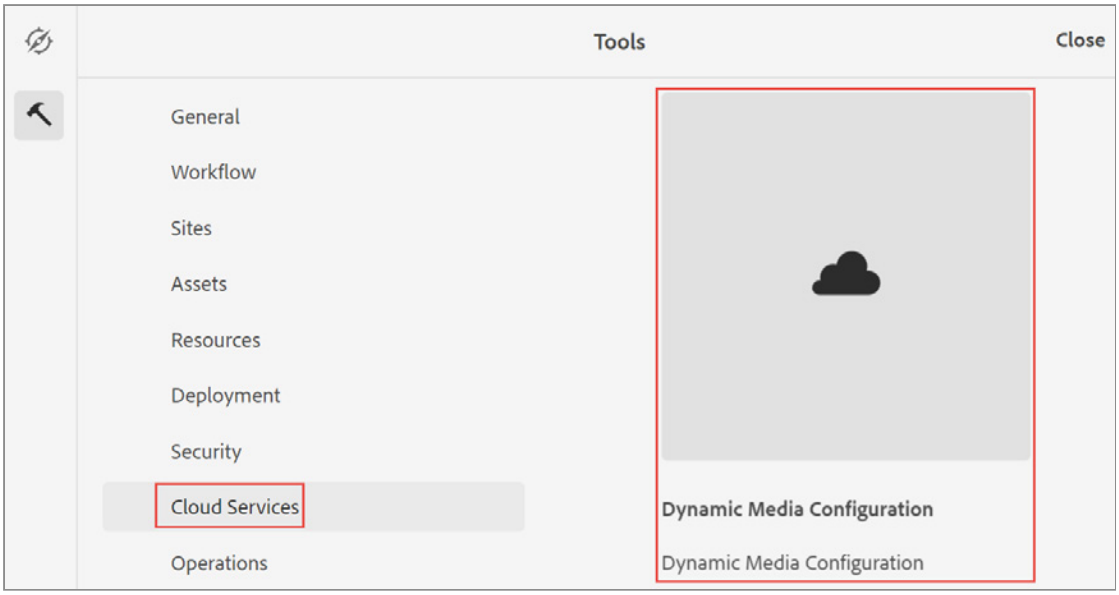
9. Notice the new **ACME** folder is created, as shown:



This folder helps create Content Fragment Models for your site implementation.

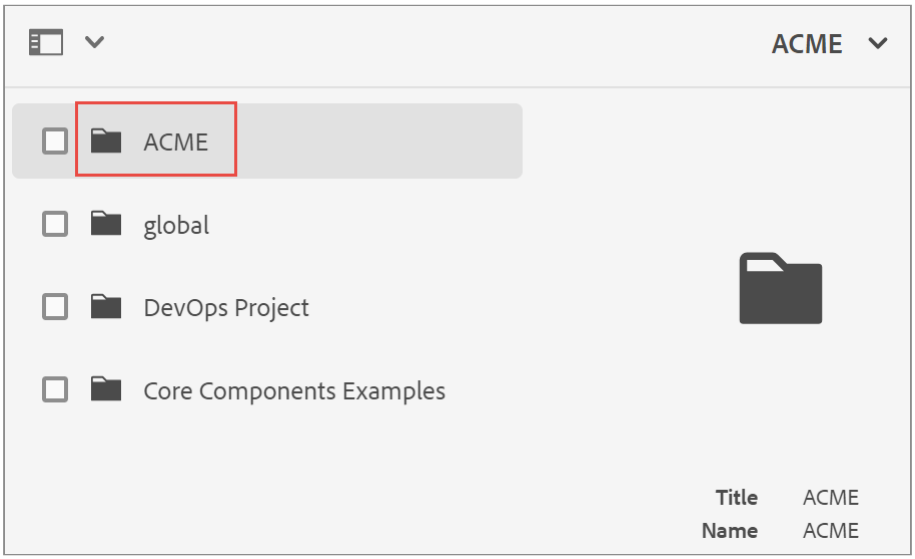
10. Click **Adobe Experience Manager** from the header bar.

11. Click **Tools** and click **Cloud Services** > **Dynamic Media Configuration**, as shown:



The **Dynamic Media Configuration Browser** console opens.

12. Notice the new **ACME** folder is created, as shown:



This folder helps configure Dynamic Media for your Assets implementation.

To view the ACME config folder in repository:

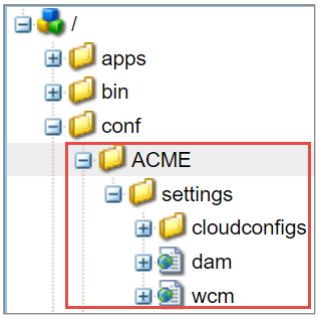
13. Click **Adobe Experience Manager** from the header bar.

14. Click **Tools** and click **General** > **CRXDE Lite**. The **CRXDE Lite** page opens.

15. Click the + icon beside the **conf** folder to expand and view its structure.

16. Notice that the **ACME** folder is added to the **conf** folder.

17. Click the + icon beside the **ACME** folder and notice that the functionalities that you enabled in step 3 are saved under **settings**, as shown:

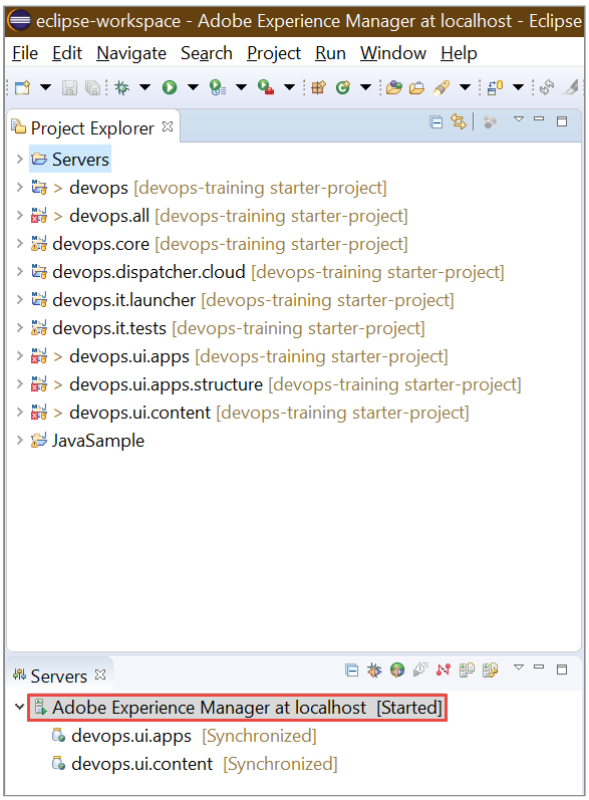


Exercise 2: Sync the configuration folder with your Maven project

Scenario: You have created a new configuration folder within your local AEM service. To use the configuration in your project, you need to synchronize it back to the Maven project. This will ensure that a configuration will be created whenever your install the code.

1. Open the Eclipse application, on the left-hand side of the workspace (below the **Project Explorer** tab), on the **Servers** tab, check if the **Adobe Experience Manager at localhost** is started.

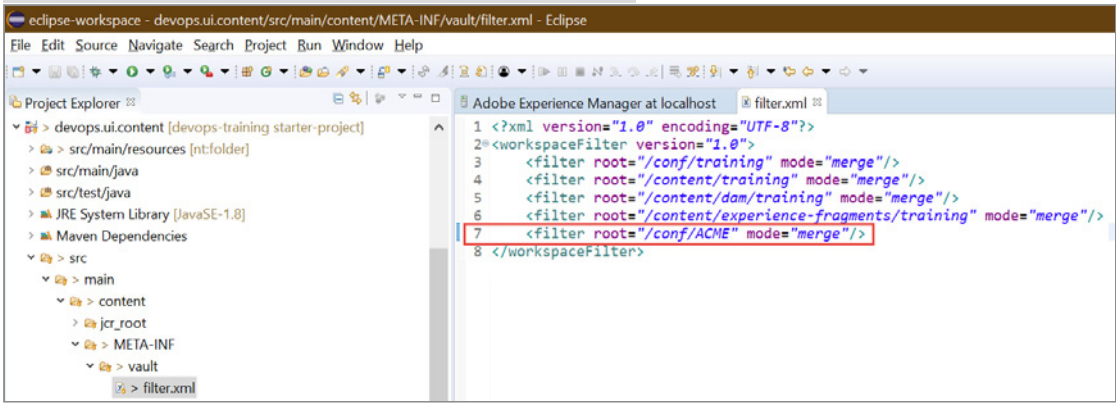
2. If the server is not started, right-click and select **Start**, the status changes to **Adobe Experience Manager at localhost [Started]**, which confirms the server is running, as shown:



> **Note**: Refer to the Technical Basics module on Eclipse to recreate or troubleshoot the server if you do not have a server present or if it is not started.
> The context-aware configurations (/conf) are considered as mutable content and, therefore, you should add it to the ui.content maven module.

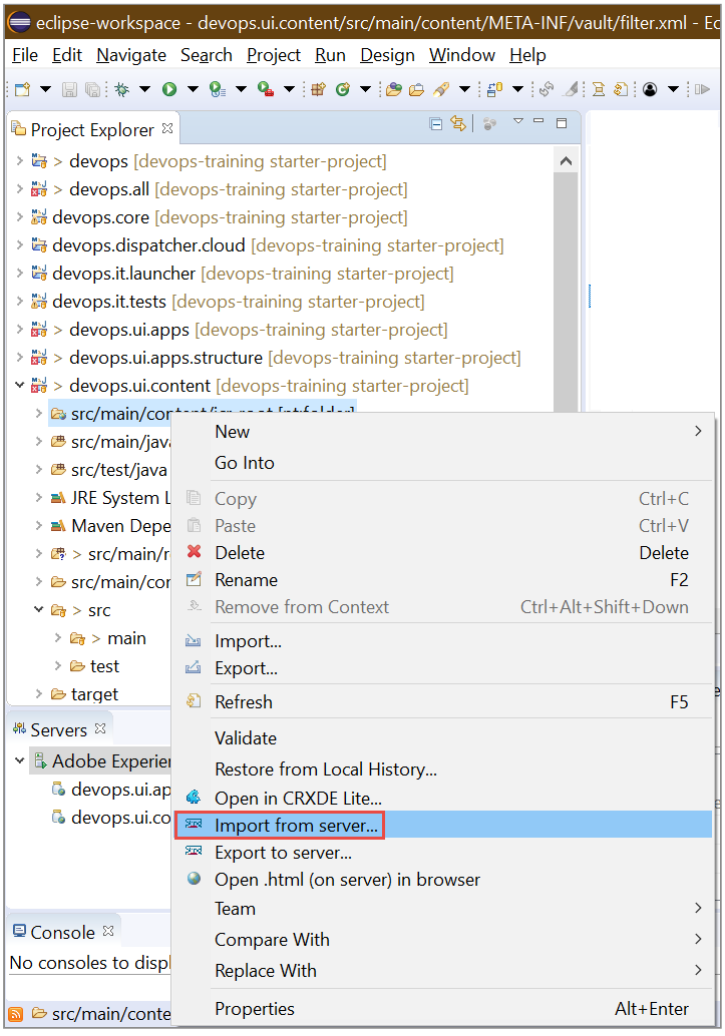
3. Go to **devops.ui.content/src/main/content/META-INF/vault**, open **filter.xml**, and add the below line of code, as shown:

```
<filter root="/conf/ACME" mode="merge"/>
```



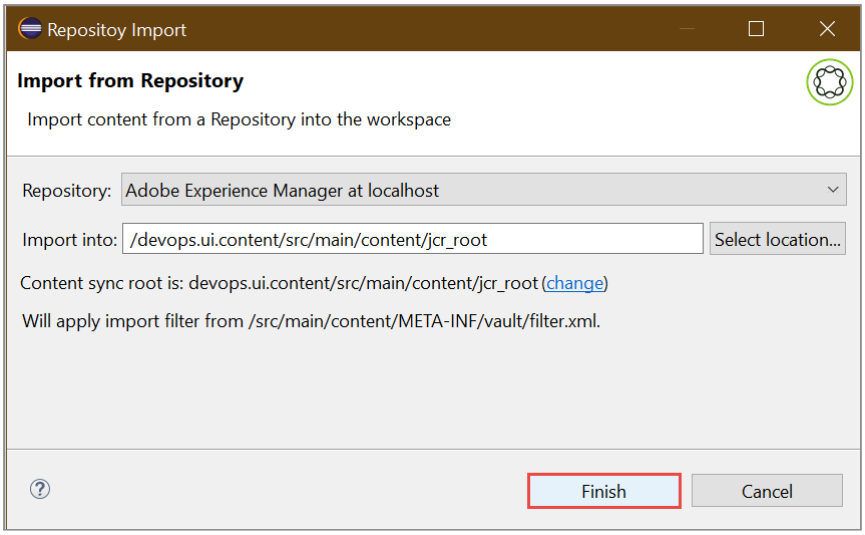
Note: The filter.xml file determines the nodes that you need to import into the project and the nodes that you need to install in AEM.

4. Under **devops.ui.content**, right-click the **src/main/content/jcr_root** and select **Import from server**, as shown:

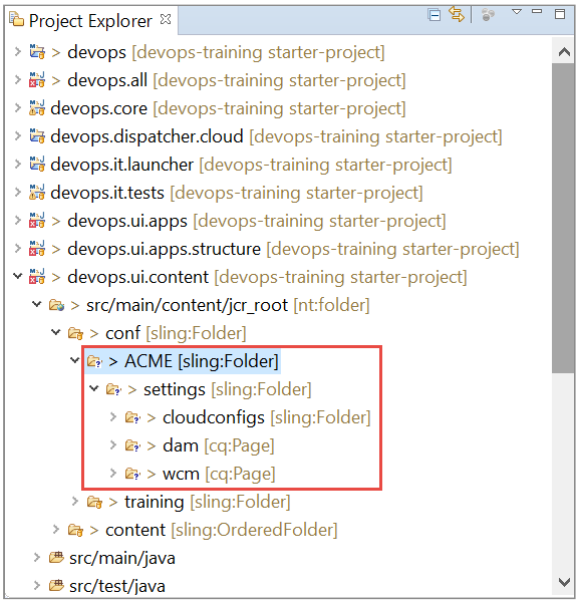


The **Repository Import** dialog box opens.

5. Click **Finish**, as shown:



The contents are now in sync with the AEM server, as shown:



Note: Any changes you sync back to your maven project should be committed into source control through the following command:

```
git commit -a -m "added new files"
```

Global Configurations

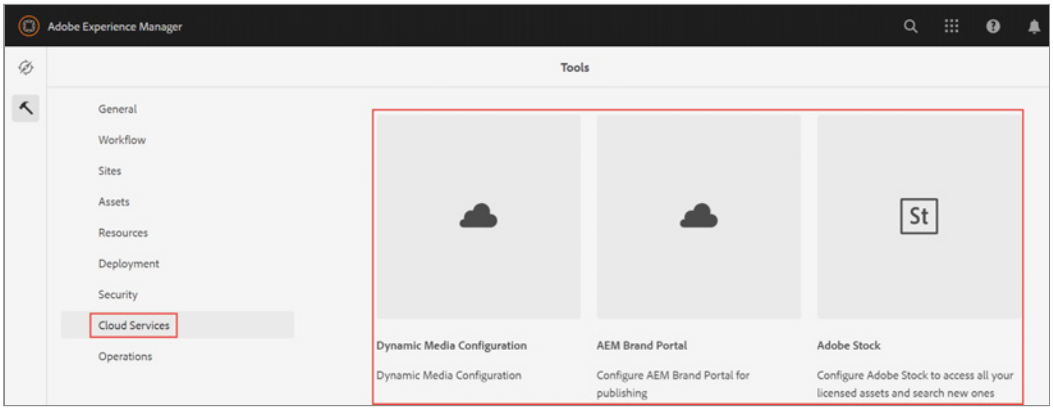
Global configurations are:

- Configurations that apply to the entire AEM platform
- Used by the following functionalities of AEM:
    - Assets
    - Search filters
    - Workflows
    - Maintenance
    - Translation rules

Assets Platform Configurations

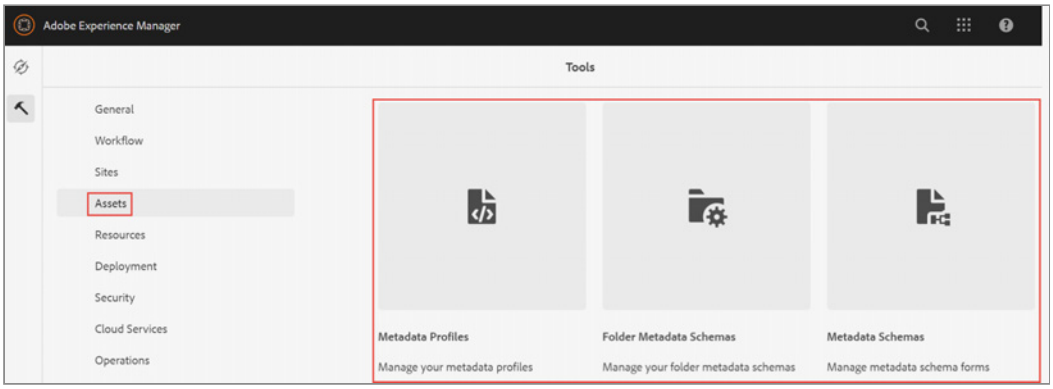
You can integrate AEM Assets with the following solutions through configurations from **Tools** > **Cloud Services** panel, as shown:

- Brand Portal [2]
- Dynamic Media [3]
- Adobe Stock [4]



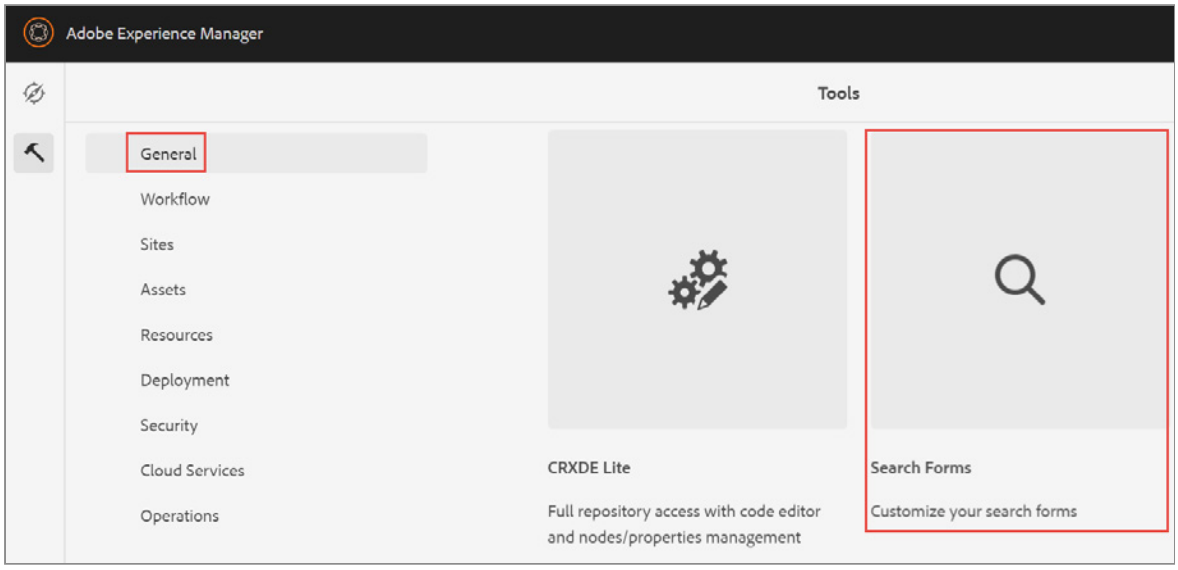
You can further customize AEM Assets with the below global configurations from **Tools** > **Assets** panel, as shown:

- Asset Schemas [5]
- Asset Folder Schemas [6]
- Asset Profiles [7]



Platform Search Filters [8]

You can use **Search Forms** from the **Tools** console to customize the selection of search predicates used in the search panels available in various AEM consoles, as shown:



Customizing these panels makes the search functionality versatile according to your specific needs.

A range of out-of-the-box predicates are available. You can add multiple predicates, including (amongst others) the Fulltext predicate for full-text searches, the Property predicate to search for assets that match a single property specified by you, or the Options predicate to search assets that match one or more values that you specify for a particular property.

Common Search Filters

The below table describes the commonly used configurations of AEM:

Configuration		Description
Page Editor	Documents search	Defines document predicates in the authoring UI search rail
	Image Search	Defines image predicates in the authoring UI search rail
	Page search	Defines page predicates in the authoring UI search rail
Sites Admin Search Rail		Defines site and page predicates of the Sites console search rail
Assets Admin Search Rail		Defines asset and folder predicates of the Assets console search rail
Project Admin Search Rail		Defines project predicates of the Project console search rail

Note: The common search predicates are defined in:
/libs/cq/gui/components/common/admin/customsearch/searchpredicates

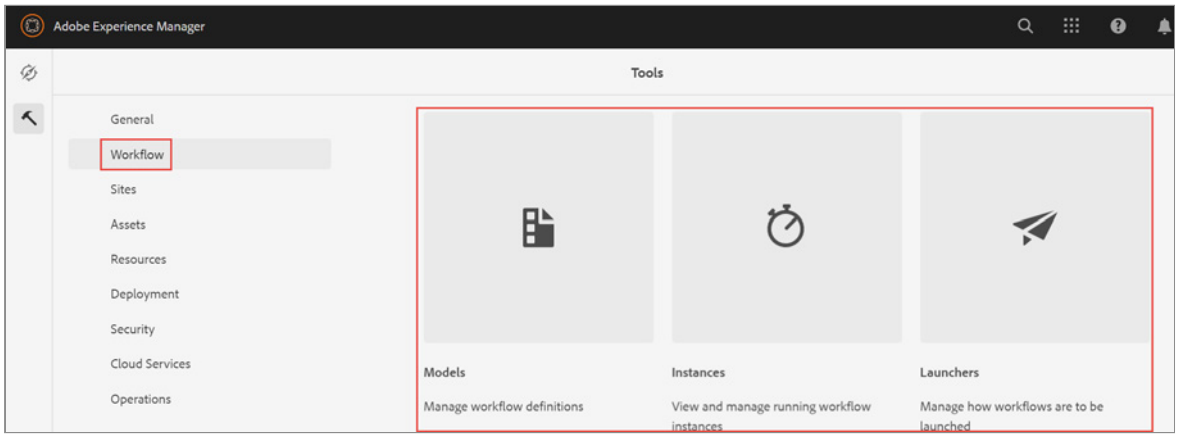
This information is for reference only, you should not make changes to /libs .

The customized configurations are stored (as appropriate) under:

- **/apps/cq/gui/content/facets/**
- **/apps/commerce/gui/content/facets/**

Workflow Configurations [9]

Workflows enable you to automate AEM activities. It consists of a series of steps that are executed in a specific order. Each step performs a distinct activity such as activating a page or sending an email message. Workflows can interact with assets in the repository, user accounts, and AEM services. Therefore, workflows can coordinate complicated activities that involve any aspect of AEM. You can access the **Workflow** panel from the **Tools** console, as shown:



The new and modified workflow models and launchers are stored as global configurations. The definitions of workflow models, launchers, scripts, and notifications are stored in the repository according to their types such as out-of-the-box, custom, and others. For example, the:

- Out-of-the-box workflow designs are held under: **/libs/settings/workflow/models/**
- Custom workflow designs are held under: **/conf/global/settings/workflow/models/...**
- Runtime workflow designs (both out-of-the-box and custom) are held under: **/var/workflow/models/**
- Legacy workflow designs (both design-time and runtime) are held under: **/etc/workflow/models/**

Maintenance Configurations [10]

For AEM as a Cloud Service, the configurations should be committed to source control and deployed by using the Cloud Manager. Adobe will manage maintenance tasks that do not require customer decisions such as, Datastore Garbage Collection while customers will configure other maintenance tasks, such as Ad-hoc Task Purge, Workflow Purge, and Project Purge.

Translation Rules in Configurations [11]

Translation rules identify content in AEM to be extracted for translation. Out of the box translation rules cover common use cases such as Text components and alt text for Image components. Depending on a projects translation requirements additional rules may be needed. The Translation Configuration UI enables a user to manage rules for translating content in AEM Sites.

Translation rules [12] are expressed in XML format and stored in these possible locations:

- **/conf/global/settings/translation/rules/translation_rules.xml**
- **/apps/settings/translation/rules/translation_rules.xml**
- **/libs/settings/translation/rules/translation_rules.xml**

The file applies to all translation projects.

References

1. Context-aware Configurations https://sling.apache.org/documentation/bundles/context-aware-configuration/context-aware-configuration.html ↩

2. Brand Portal https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/brand-portal/configure-aem-assets-with-brand-portal.html ↩

3. Dynamic Media https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/dynamicmedia/administering-dynamic-media.html ↩

4. Adobe Stock https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/manage/aem-assets-adobe-stock.html ↩

5. Metadata Schemas https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/manage/metadata-schemas.html ↩

6. Asset Folder schemas https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/admin/folder-metadata-schema.html ↩

7. Metadata profiles https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/manage/metadata-profiles.html ↩

8. Search Facets: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/assets/admin/search-facets.html ↩

9. Workflows: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/sites/authoring/workflows/overview.html ↩

10. Maintenance: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/operations/maintenance.html ↩

11. Identifying Content to Translate: https://docs.adobe.com/content/help/en/experience-manager-65/administering/introduction/tc-rules.html ↩

12. Translation Rules https://docs.adobe.com/content/help/en/experience-manager-learn/sites/translation/translation-rules-editor-technical-video-setup.html ↩