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

02/22/2021

Contents

Developer Tools

Introduction

Adobe Experience Manager (AEM) has common developer tools that are used to develop your Java Content Repository (JCR), Apache Sling, or AEM applications.

Objectives

After completing this course, you will be able to:

- Explain the features and UI elements of the AEM developer tools
- Install a package in AEM
- Create, build, and download packages
- Describe a content package structure
- Create an immutable package
- Create a mutable package

Developer Tools

The three most common developer tools available in AEM for developers and administrators are:

- CRXDE Lite
- Web Console
- Package Manager

Local Development

- For local development, Developers have full access to CRXDE Lite (/crx/de) and the AEM Web Console (/system/console).

Note: On local development (using the cloud-ready quickstart), /apps and /libs can be written to directly, which is different from Cloud environments where those top-level folders are immutable.

AEM as a Cloud Service Development tools

- Customers can access CRXDE lite on the development environment but not stage or production. The immutable repository (/libs, /apps) cannot be written to at runtime, so attempting to do so will result in errors.
- Customers have access to package manager for author instances (and not publish) in Cloud Service. They can only upload packages containing mutable content.
- A set of tools for debugging AEM as a Cloud Service developer environments are available in the Developer Console for dev, stage, and production environments. The URL can be determined by adjusting the author or publish service URLs as follows:
  `https://dev-console>-<namespace>.<cluster>.dev.adobeaemcloud.com`
- As a shortcut, you can use the following Cloud Manager CLI command to launch the developer console based on an environment parameter described below:
  `aio cloudmanager:open-developer-console <ENVIRONMENTID> --programId <PROGRAMID>`

CRXDE Lite

CRXDE Lite is embedded into AEM, which allows you to perform common development and administration tasks within the browser. It is a light Integrated Development Environment (IDE) for quick access to the JCR. Because it is embedded in the server and is always available, CRXDE Lite is often the preferred tool for administrators and developers for working with nodes and properties in the JCR. It provides you with quick and direct access to the repository for monitoring, configuration, and development.

You can access CRXDE Lite directly by navigating to http://localhost:4502/crx/de/index.jsp or by navigating to **Tools** > **CRXDE Lite** within AEM.
For local development, Developers have full access to CRXDE Lite (/crx/de) and the AEM Web Console (/system/console). Note that on local development (using the cloud-ready quickstart), /apps and /libs can be written to directly, which is different from Cloud environments where those top-level folders are immutable.
Customers can access CRX/DE lite on the development environment (but not stage or production). Note that the immutable repository (/libs, /apps) cannot be written to at runtime, so attempting to do so will result in errors.
Developer Tools under /system/console will not be available in Skyline. Instead, a raw data dump tool will be exposed at a dedicated URL that outputs information to either the screen or as a file download.

With CRXDE Lite, you can access the Java Content Repository (JCR) of the AEM service and validate any value in the repository. CRXDE Lite can be used in development in combination with an IDE such as Eclipse, Visual Studio, or similar IDEs. You can also use it for quick developmental tests and training purposes.
CRXDE Lite is useful for:

- Code/content validation
- Product training
- Operations debugging
- Overlays from /libs

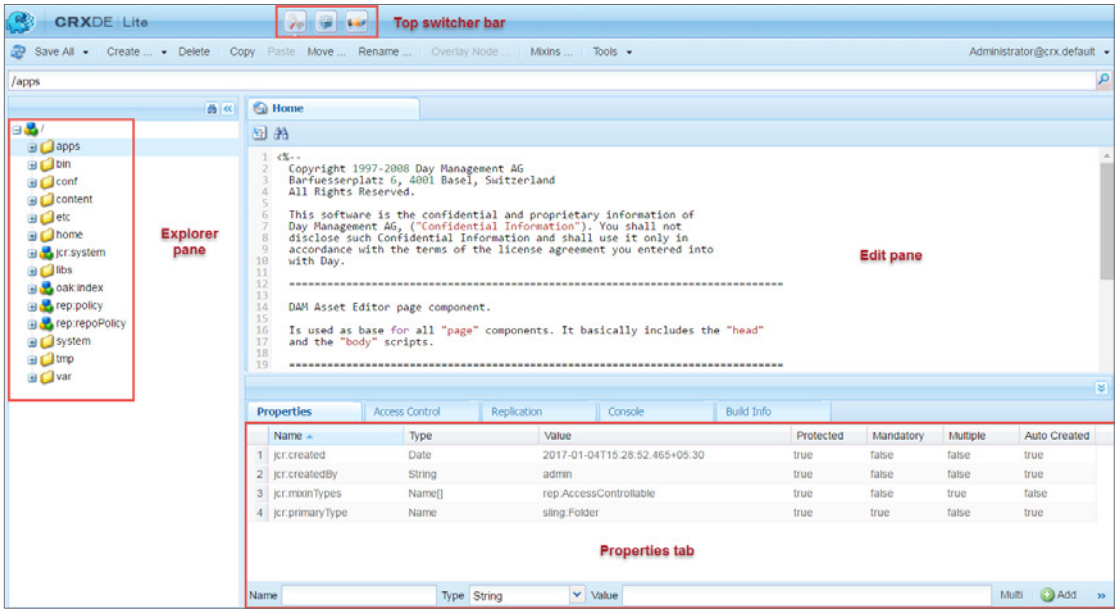
The CRXDE Lite UI contains:

- **Top switcher bar**: Enables you to quickly switch between CRXDE Lite, Package Manager, and Package Share.
- **Explorer pane**: Displays a folder tree structure of all the nodes in the repository.

You can perform the following actions on a node in the tree:

- Select the node and view its properties in the **Properties** tab. Examine all the JCR properties of different nodes.

- Right-click the node and perform an action on it, such as renaming the node, creating a new node, creating a folder, and creating a file.
- Edit the code. The **Edit** pane allows you to double-click a file, such as a .jsp or a .html file, in the **Explorer** pane to display its content. You can then modify the code and save the changes.
- View node properties. On the **Properties** tab, you can display and view the properties of the node that you selected. You can also add new properties or delete existing ones.



Tip: Bookmark the **CRXDE** Lite URL (http://localhost:4502/crx/de/index.jsp) in your browser to access this tool, as you will use it often in your training as well as in your role as an AEM developer or administrator.

Web Console

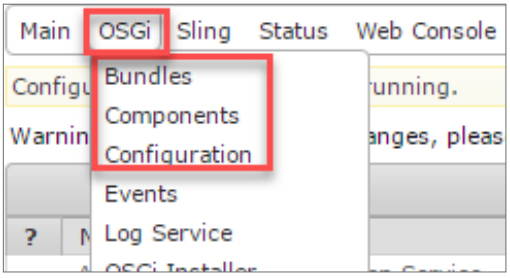
The Web Console in AEM is based on the Apache Felix Web Management Console, and is used to manage OSGi bundles and configurations. Any changes made through this console are automatically applied to the running system, without the need to restart the service. OSGi is where you manage your Java classes in the form of OSGi components and services.

You can access the console at http://localhost:4502/system/console.

Note: The Web console can only be accessed for local development.

The Web Console contains a selection of tabs for maintaining menu selections under the OSGi tab. These tabs include:

- **Bundles**: Used for installing and managing bundles.
- **Components**: Used for managing and controlling the status of components required for AEM.
- **Configuration**: Used for configuring an OSGi bundle and is the underlying mechanism for configuring AEM parameters.



Package Management in AEM

A package is a *.zip file that holds AEM repository content in the form of a file-system serialization called Vault serialization. Vault is provided by Apache Jackrabbit.

Packages provide an easy-to-use-and-edit representation of files, such as pages, assets, and folders. They also enable you to import and export repository content from one service or environment to another.

A package can contain:

- Page-related content
- Project-related content
- Assets-related content
- Vault meta information, such as filter definitions and import configuration information
- Other package information such as package settings, package filters, package screenshots, and package icons

You can use packages to perform any of the following tasks:

- Install new functionality
- Transfer content between services
- Export content to the local file system

You can work with packages by using Package Manager or Package Share. You can access this through the following link: http://localhost:4502/crx/packmgr/index.jsp.

Package Manager

Package Manager is used to import or export content on your service, transfer content between services, and back up repository content. With Package Manager, you can perform the following common tasks:

- Create, build, and download content packages
- Upload, validate, and install packages
- Modify existing packages
- View package information

You can also use filters to create a package containing page content or project-related content.

Creating and Building New Packages

When creating packages using Package Manager, you can apply rules and filters to determine the content a package should extract from the repository. After you define the content, you can build it. A package is created in a .zip file, which you can download to your local file system. You can also test the contents of the package *before* building it.

Package Manager provides you with many options to work with packages:

- **Rebuild**: Helps rebuild the package if there is a change in the repository content.
- **Edit**: Helps edit filters or rules applied to the package.
- **Test**: Helps perform a dry run of the installation.
- **Rewrap**: Helps recreate the package with additional information such as thumbnails and icons.

Downloading Packages to the File System

You can download a package by clicking the **Download** link. This link is displayed when the package details are expanded. After downloading the package, you can unzip the contents of the package to your local system.

Typically, an unzipped (extracted) content package contains the following folders:

- **jcr_root**: Contains files and folders that are serialized nodes and properties from the JCR.
- **META-INF**: Contains metadata regarding node definitions and the filter.xml file that gives directions to Vault about the paths to include.

Package Share

Package Share is a centralized server where public packages are made available. These packages may include hotfixes, new functionality, updates, or documentation. You can search, download, and install any package either to your service or your local file system.

Within the Package Share, you have access to the following:

- AEM packages provided by Adobe (for example, new functionalities such as updated core components, hotfixes, and service packs)
- Shared packages provided by other organizations and made public by Adobe

You can access this console through the following link: http://localhost:4502/crx/packageshare/index.html

You can also directly access Package Share through the following link (after signing in using your Adobe ID): https://experience.adobe.com/#/downloads/content/software-distribution/en/aem.html

Deploying content packages via Cloud Manager and Package Manager

Customers deploy custom code to cloud environments through Cloud Manager. It should be noted that Cloud Manager transforms locally assembled content packages into an artifact conforming to the Sling Feature Model, which is how an AEM service application is described when running in a cloud environment.

Content packages written for AEM Cloud applications must have a clean separation between immutable and mutable content. If this separation does not exist, Cloud Manager will enforce it by failing the build.

Immutable content packages

All code persisted in the immutable repository (/apps) must be checked into Git and deployed through Cloud Manager. In other words, unlike current AEM solutions, code is never deployed directly to a running AEM service. This ensures that the code running for a given release in any Cloud environment is identical, which eliminates the risk of unintentional code variation on production. As an example, OSGI configuration should be committed to source control rather than managed at runtime via the AEM Web Console's configuration manager.

Mutable content packages

Content such as folder path hierarchies, service users, and ACLs are typically committed into a Maven archetype-based AEM project. Techniques include exporting from AEM or writing directly as XML. During the build and deployment process, Cloud Manager generates the resulting mutable content package and installs it, writing out all the previous content nodes.

The full list of mutable content includes:

- service users (add, modify, remove)
- service user ACLs (add, modify, remove)
- folders (add, modify, remove)
- node types (add, modify, remove)
- index definitions (add, modify, remove)
- editable templates (add, modify, remove)
- script (packages can trigger Install hooks at various stages of the install process of package installation)

Content packages are deployed to all environment types (dev, stage, prod); however, it is not possible to limit deployment to a specific environment.

Also, there is no mechanism to rollback the mutable content package changes after they have been applied. If customers detect a problem, they can choose to fix it in their next code release or, as a last resort, restore the entire system to a point in time *before* the deployment.

Exercise 1: Install a package

Scenario: As an AEM developer or administrator/development operations, you need to have sample content to reference and base your code off of. In this task, you will install the WKND content package so you can reference the WKND implementation during development. The WKND content package is a container content package. This means that the only thing that is contained in this package is other content packages. This allows for a single content package for a project but still maintains the separation of mutable and immutable content.

This exercise includes the following tasks:

1. Install a container package
2. Verify the newly installed site

Task 1: Install a container package

In this task, you will validate and install a container package using the Package Manager and then validate that the mutable and immutable content packages extracted were successfully installed.

Download the container package:

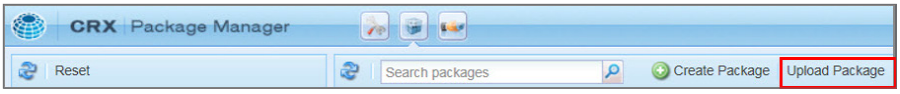
1. If your AEM instance is not running, start it.
2. Download the aem-guides-wknd.all-#.#.#.zip content package using the URL: https://github.com/adobe/aem-guides-wknd/releases

Note: The #.## indicates the version of the WKND site you will use. The instructor will provide this information to you. The version in the screenshots might be different than the version you are given to use.

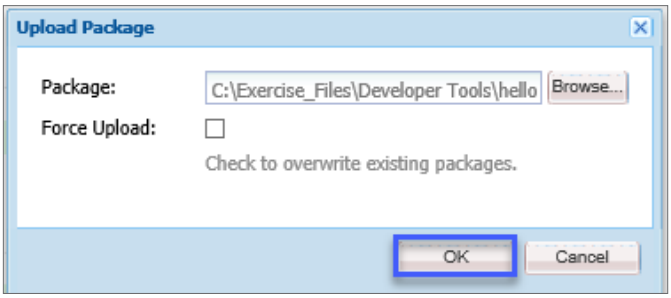
Note: If you are using ReadyTech, **aem-guides-wknd.all-#.#.#.zip** can be found on the desktop. If you are working locally and cannot download from the link above, your instructor can provide this zip for you.

To validate a package:

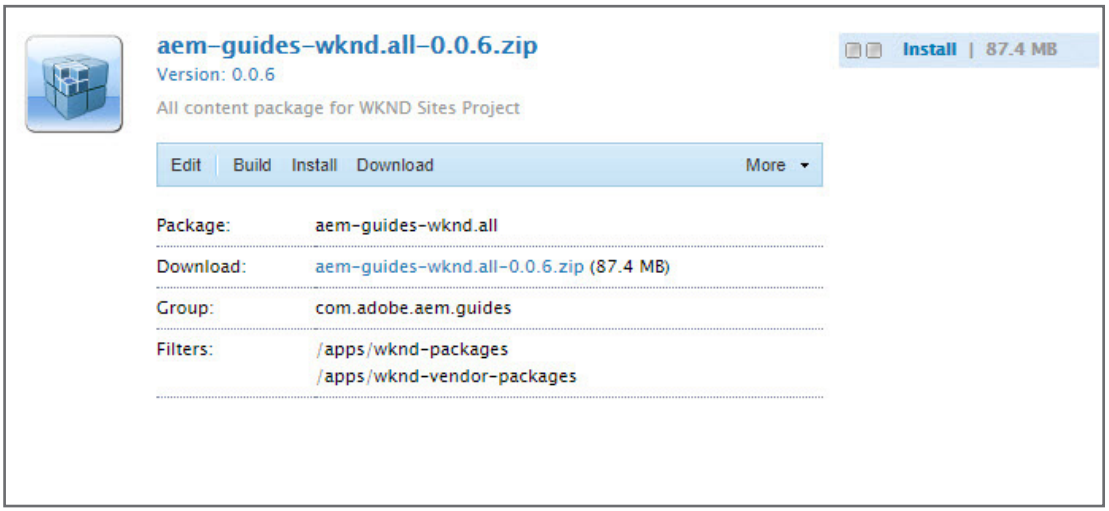
3. Verify you are logged on to your AEM author service on port 4502 (http://localhost:4502).
4. Navigate to http://localhost:4502/crx/packmgr/index.jsp. This will take you to the AEM Package Manager tool.
5. Click **Upload Package.**



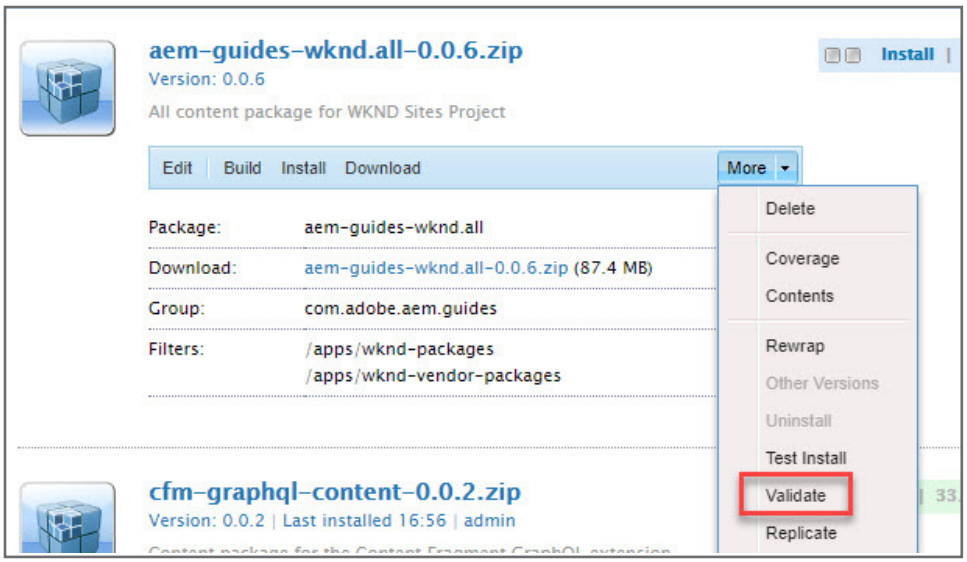
6. In the **Upload Package** dialog box, click **Browse** and select the **aem-guides-wknd.all-#.#.#.zip** package that you downloaded in Step 1.
7. Click **Open** and then click **OK** as shown:



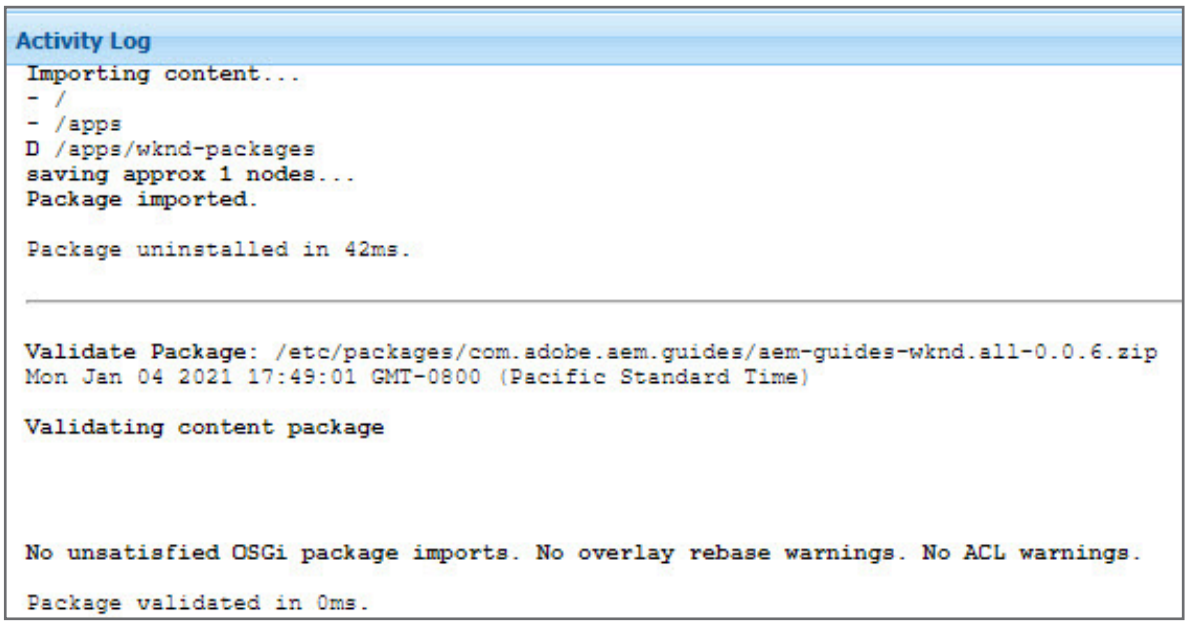
Your uploaded package is now available in AEM Package Manager, as shown.



8. Click **More** > **Validate** as shown:



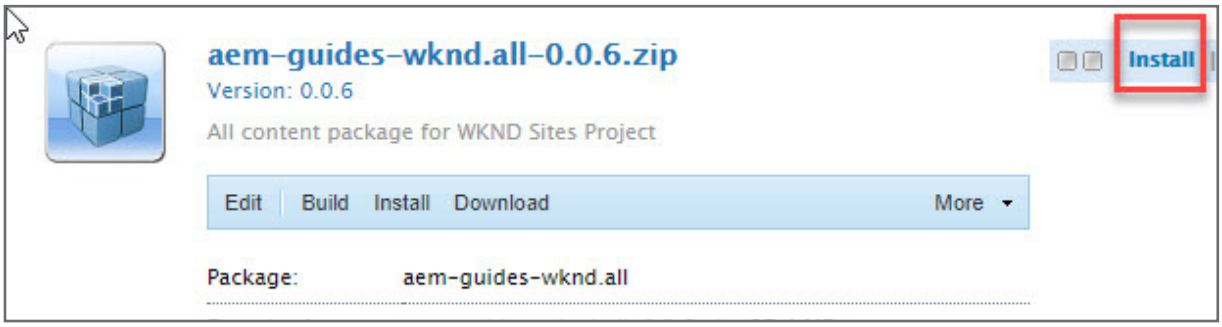
9. Leave all options selected as-is and click **Validate** in the dialog box. In the **Activity Log** below the **aem-guides-wknd.all-#.#.#.zip**, notice there are no issues with your package:



Note: The validation utility will notify you of any issues that impact overlaid JCR resources in /apps, any unsatisfied bundles, and any Access Control Lists (ACL) conflicts. In other words, this will prevent any problems with OSGi bundles that are unable to start after installing a package, any impacts on permissions (ACLs), and files in /libs that impact overlaid files in /apps.

To install your package:

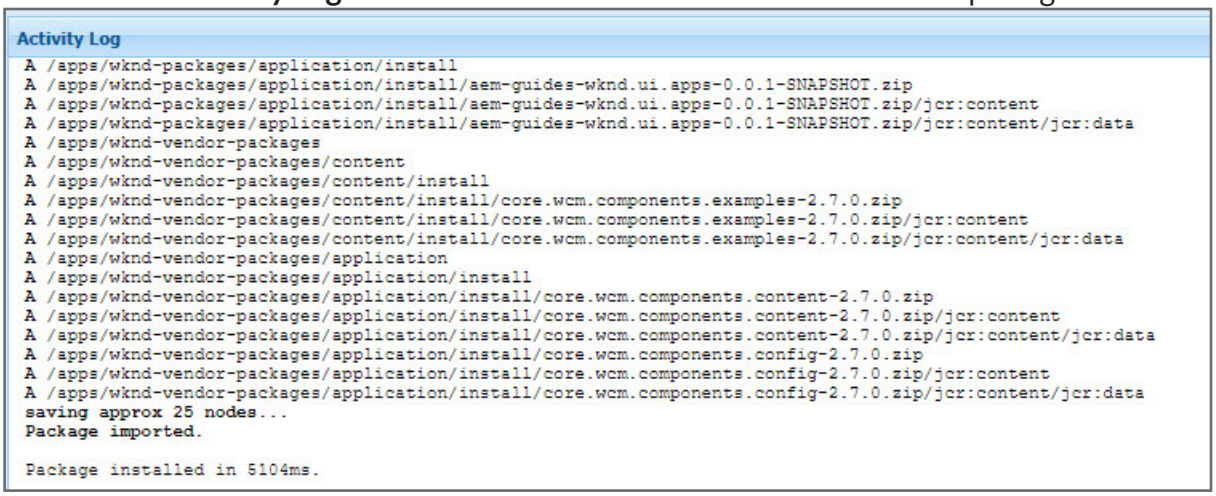
10. Click **Install**, as shown.



The **Install Package** dialog box appears.

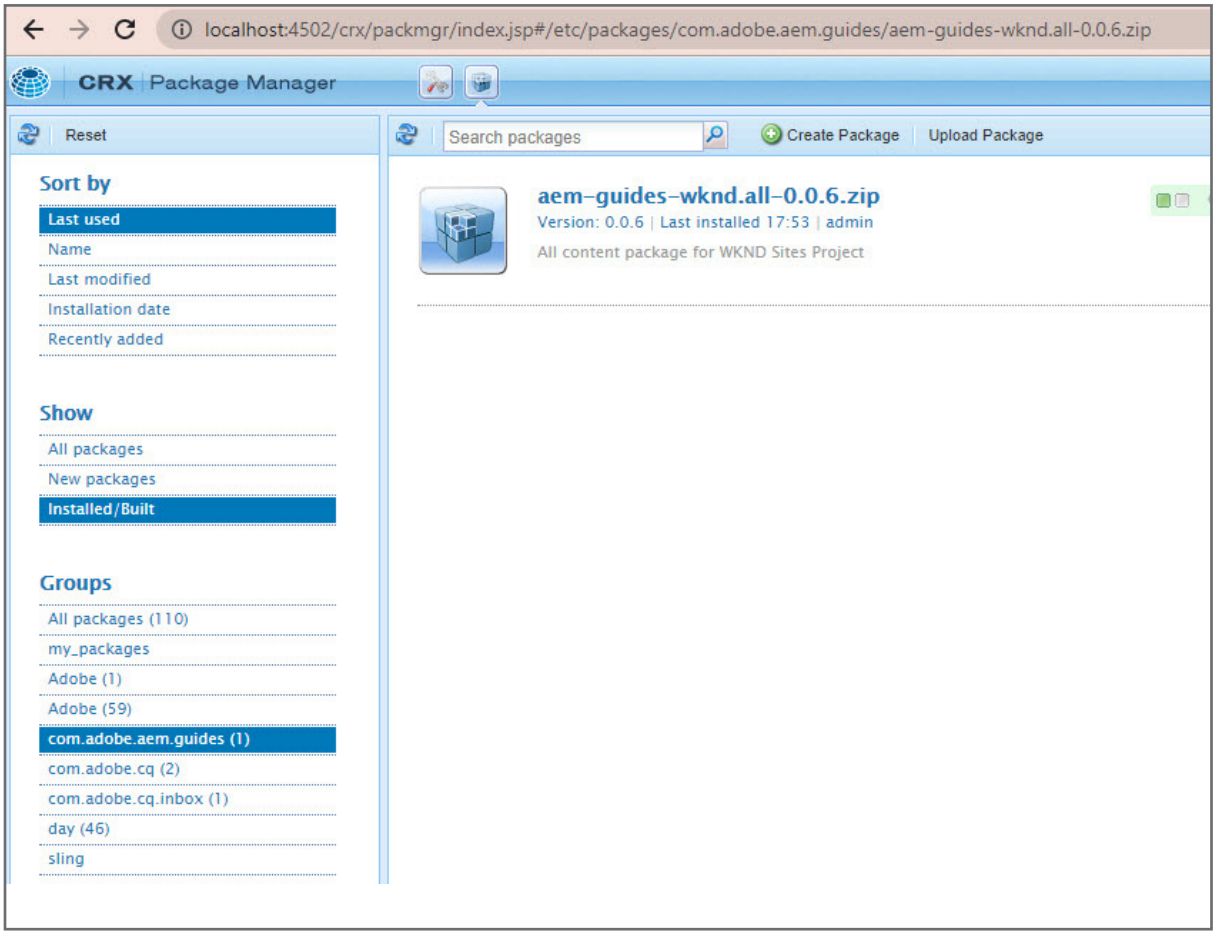
11. Ignore the **Advanced Settings** area and click **Install**.

12. Check the **Activity Log**. You can see that the content was added from the package.



Note: This container content package that you installed contains mutable and immutable content packages.

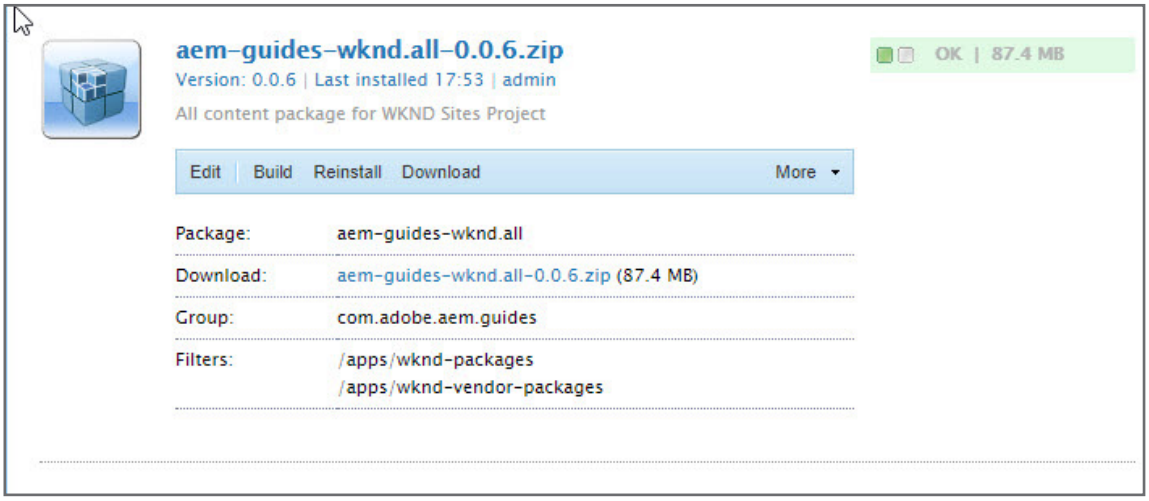
13. In the left navigation pane, under **Groups**, click on **com.adobe.aem.guides** and verify the container package, as shown:



Note: If you do not see **com.adobe.aem.guides (1)** in the left navigation pane (under **Groups**) OR, after clicking the **com.adobe.aem.guides (1)** link, you do not see the two packages (referenced in Step 11) in the main Packages area, refresh the Package Manager. Sometimes, it takes time for the screen to refresh in order for you to be able to see updated links (on the left side) as well as see the corresponding packages associated with the links on the left side.

Note: A container package can ONLY include other content packages (.zip) and bundles (.jars).

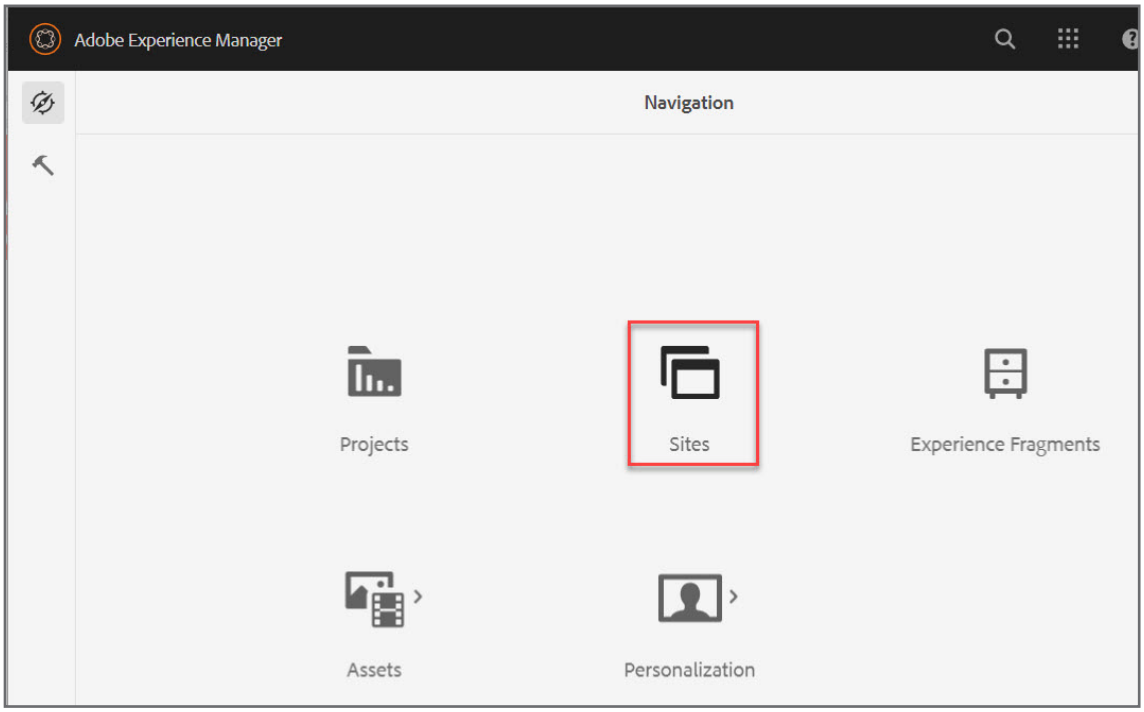
14. The **aem-guides-wknd.all-#.#.#.zip** contains all of the mutable content for the WKND project. Click the **aem-guides-wknd.all-#.#.#.zip** link and scroll down to the **Filters** section:



Task 2: Verify the newly installed site

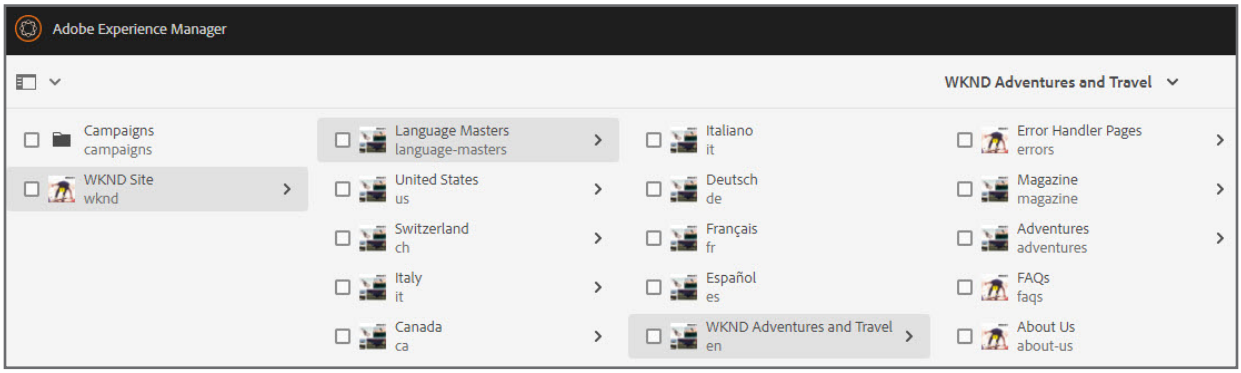
In this task, you will navigate to Sites and verify the newly installed site.

1. Click the browser tab with the Adobe author service.
2. Click **Adobe Experience Manager** in the upper left.
3. In the Navigation page area, click **Sites**, as shown.

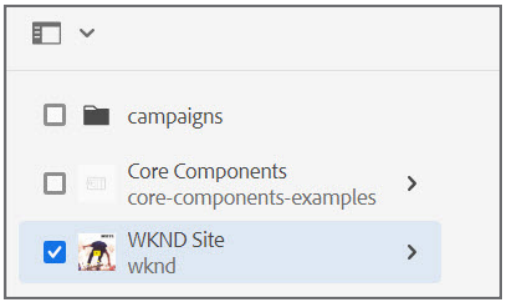


Note: You may see the **Product Navigation** tutorial dialog guiding you through the navigation. You may click **Next** to proceed through the tutorial and learn the basic AEM UI elements and navigation, or you may click **Close** to hide the tutorial.

4. In the column view, verify the new **WKND Site**. Navigate to **WKND Site** > **Language Masters** > **English**, as shown.



TIP: If you see a checkmark on the WKND site thumbnail, as shown below, it means you have *selected* WKND site for editing and/or managing the page. Click the thumbnail again to clear the selection and click the right-pointing arrow instead.



You have successfully installed the WKND container package, which contains both mutable and immutable content packages.

Exercise 2: Create, build, and download packages

In this exercise, you will learn how to create a content package. Locally, you can create content packages for mutable and immutable content. However, on an AEM service, only immutable content packages can be created and managed since mutable code (/apps) can only be installed to an AEM service via Cloud manager. Your instructor may provide you with a course-filter.txt file; you can use those filters rather than the example filters in this exercise..

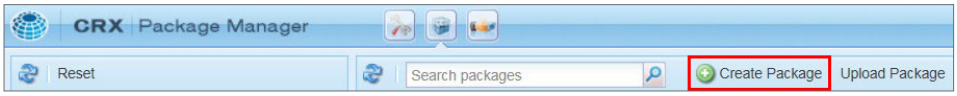
This exercise includes the following tasks:

1. Create an immutable package
2. Create a mutable package

Task 1: Create an immutable package

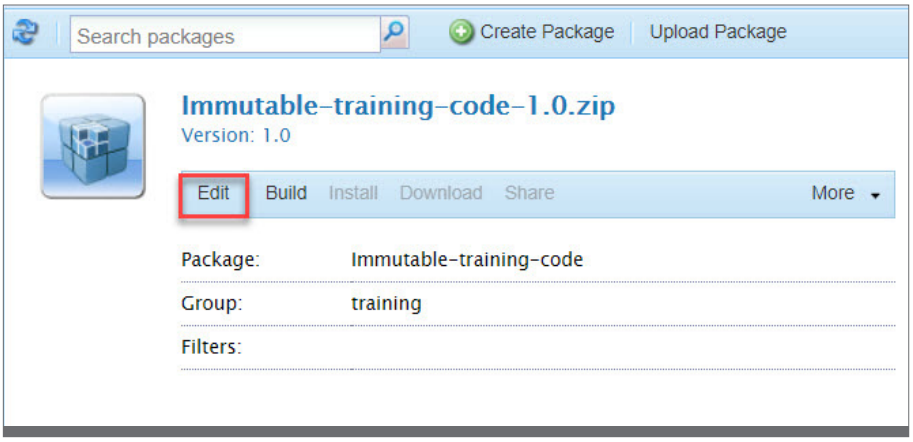
In this task, you will validate and install a package using the Package Manager.

1. In your AEM **Package Manager** tool, click **Create Package** as shown:



2. In the **New Package** dialog box, type the following details:
   a. Package Name: **Immutable-training-code**
   b. Version: **1.0**
   c. Group: **training**
3. Click **OK**. The **Immutable-training-code** package is created.

4. Click **Edit** on the newly created package, as shown.



5. To add filters to the package, click the **Filters** tab and then click **Add Filter**.

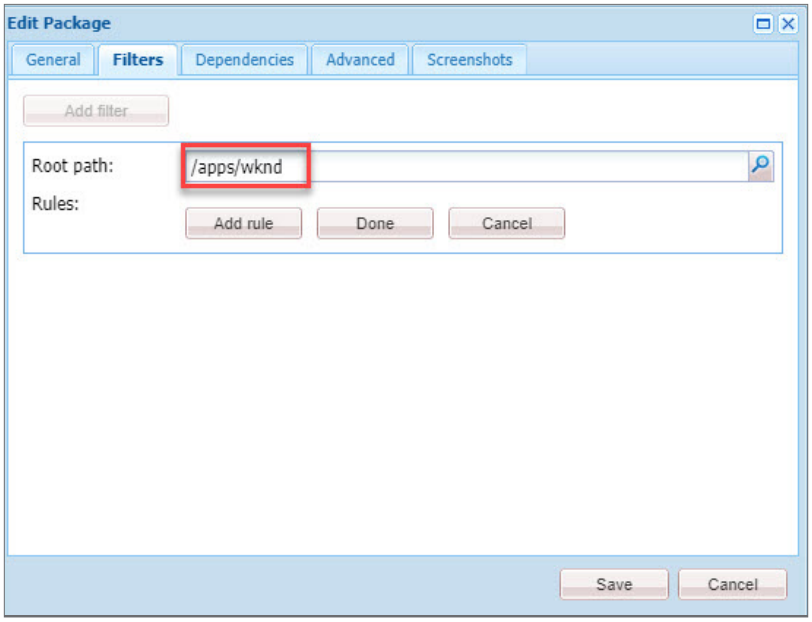
Note: Filters are the mechanism used to add content to packages. Here, you specify the paths that contain the content from the JCR you want to include in a package. Before adding filters, your package is completely empty. You may also restrict file types added to a package using filter rules, such as excluding all *.txt files.

6. For the **Root** path, type: **/apps/wknd**

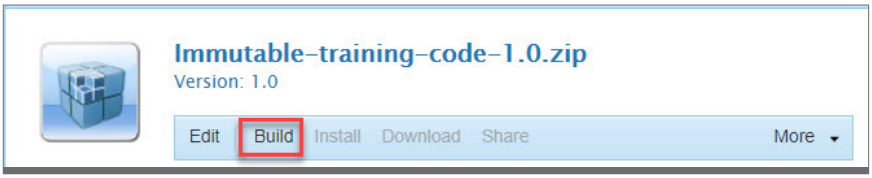
TIP: As you type in the Root path field, the field will auto-complete.

Note: If your instructor has provided you with a **course-filter.txt** file, use the immutable paths noted in this file.

7. Click **Done**, as shown.



8. Click **Save**.
9. Click **Build** to build the package, as shown.



10. Click **Build** again in the confirmation dialog box. The package is now ready for download.

Note: As the package is being built, the activity log is running at the bottom of the screen. This area is known as the Activity Log. The log shows a series of "A" actions. "A" denotes a node is being added to the content package.

11. Click the **Download** link to download a copy of the package to your computer. The package is downloaded to your computer's default **Downloads** folder for the browser.

Note: You should see a list of all packages in your service organized by group on the left side menu. Note that there is now one new package, indicated by **(1)**, in the **training** group that you just created for **Immutable-training-code**. Therefore, you can notice that you may use the group name to categorize and organize your packages in AEM.

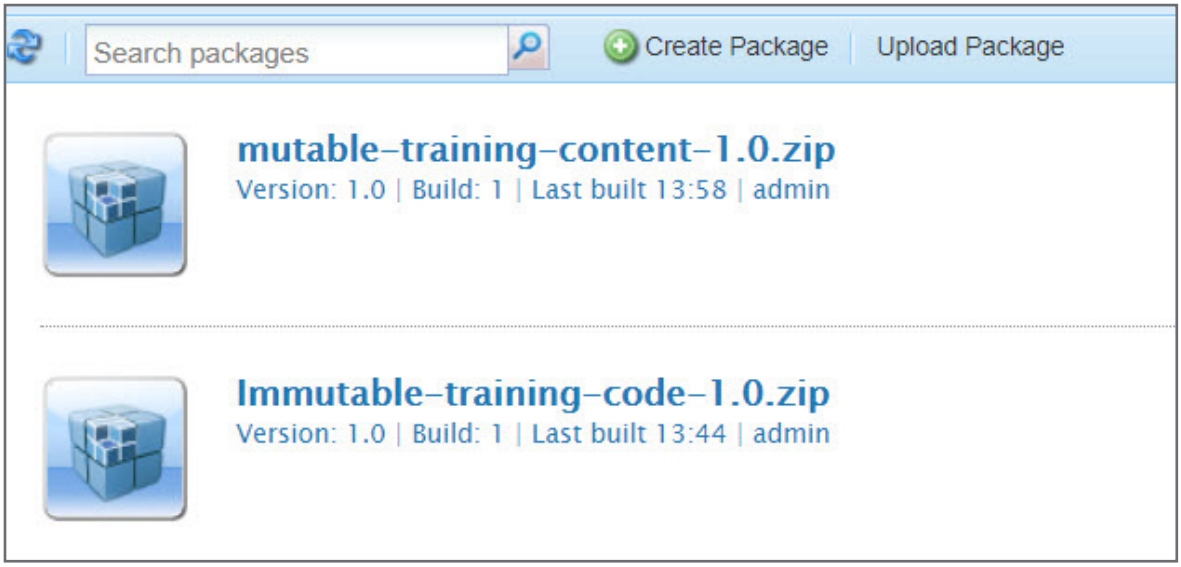
You have successfully created an immutable package.

Task 2: Create a mutable package

1. In your **CRXDE Lite** browser tab, click the **Package Manager** icon. The **Package Manager** screen appears.
2. Click **Create Package**.
3. In the **New Package** dialog box, type the following details:
   a. Package Name: **mutable-training-content**
   b. Version: **1.0**
   c. Group: **training**
4. Click **OK**. The **mutable-training-content** package is created.
5. Click **Edit** on the newly created package.
6. To add filters to the package, click the **Filters** tab and then click **Add Filter**.
7. For the **Root** path, enter **/conf/wknd**.

Note: If your instructor has provided you with a **course-filter.txt** file, use the mutable paths noted in this file.

8. Click **Done** and click **Save**.
9. Click **Build** to build the package. Click **Build** again in the confirmation dialog box. The package is now ready for download.
10. Click the **Download** link to download a copy of the package to your computer. The package is downloaded to your computer's default **Downloads** folder for the browser.
11. On the left side menu, click the **training** group link and notice the packages under the group.



You have successfully created a mutable package.

References

Bookmark/"favorite" these sites for a local author installation (development environment):

- CRXDE Lite: http://localhost:4502/crx/de/index.jsp
- Web Console: http://localhost:4502/system/console
- Package Manager: http://localhost:4502/crx/packmgr/index.jsp