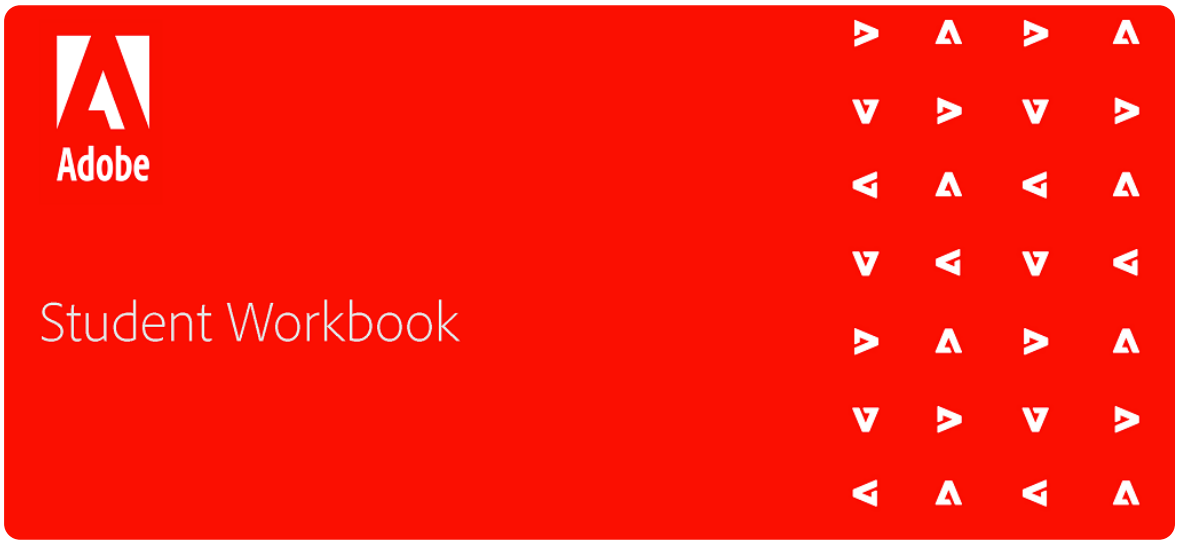
DevOps for AEM as a Cloud Service

Manage Publishing and Content Delivery



Find your journey at learning.adobe.com >

Contents

DevOps for AEM as a Cloud Service

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

10-14-2020

Introduction

Adobe Experience Manager (AEM) as a Cloud Service uses the Sling Content Distribution capability to move the appropriate content. The Publish Tier setup is automated, including self-configuration when publish nodes are added, removed, or recycled during runtime. A single publish or unpublish request can include multiple resources but will return a single status applied to all. The request succeeds for all resources in the AEM Publish Service or fails for all. This ensures the resources within the AEM Publish Service are never in an inconsistent state.
AEM's Dispatcher helps protect your AEM server from attack. Therefore, you can increase the security of your AEM instance by using the Dispatcher in conjunction with an enterprise-class web server.

Objectives

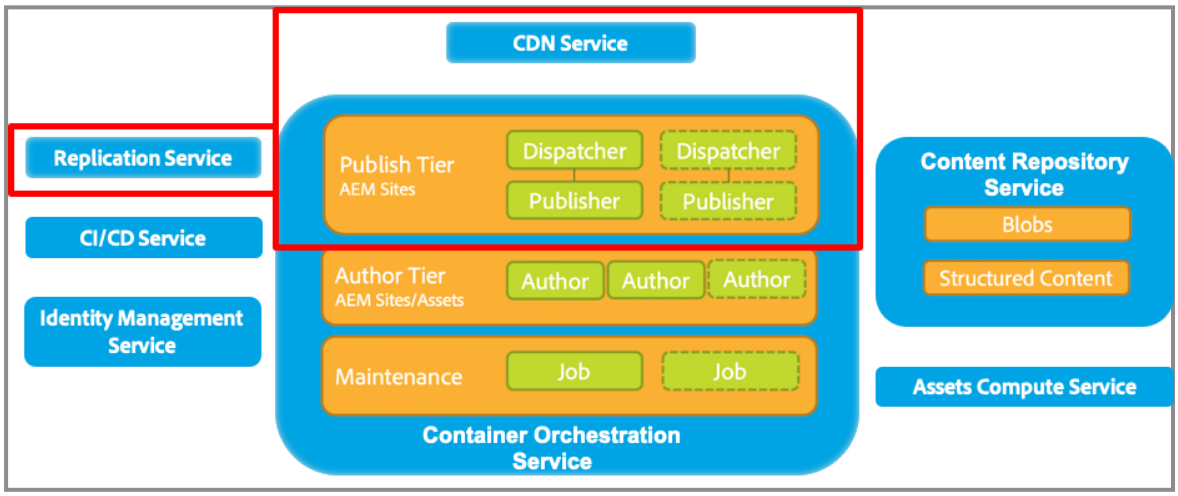
After completing this module, you will be able to:

- Describe the AEMaaCS Publish Service
- Describe Apache Sling Content Distribution and the Golden Master
- Publish a page
- Publish a tree of nodes
- Explain Content Delivery Network (CDN)
- Configure and test the Dispatcher

AEM as a Cloud Service Publish Service

The AEM as a Cloud Service publish service includes:

- Content Delivery Network (CDN) Service
- Publish Tier
- Replication Service



The AEM publish tier contains two or more publisher nodes, within a single publish farm. As a farm, the publisher nodes can operate independent of each other. Each node consists of an AEM publisher and a web server equipped with the AEM Dispatcher module. The publish tier scales automatically with site traffic needs. End users, or site visitors, visit the website through the AEM Publish Service.

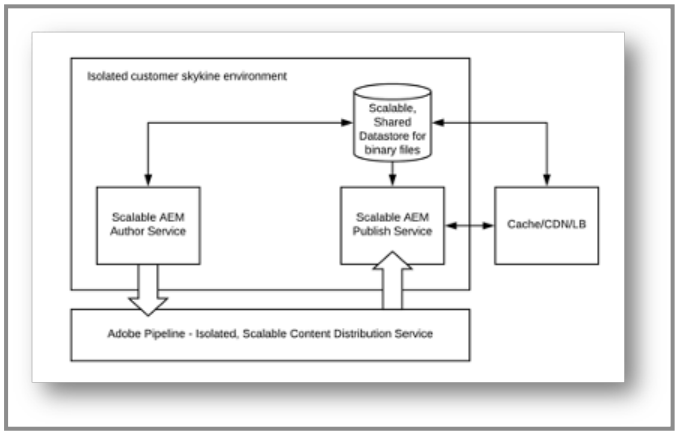
No content updates are allowed to the publish tier. The publish tier allows only read access to the content in the persistence layer.

For more information about the AEMaaCS runtime architecture and the Publish Service , see [1].

For more information about the Replication Service, see [2].

Apache Sling Content Distribution

Publishing operations, also known as content replication, from the author tier to the publish tier now follows a subscription pattern. AEM as a Cloud Service now uses the Apache Sling Content Distribution capability to move content from the author tier to the publish tier. Content is moved to the publish tier by using a pipeline service. The pipeline service runs on Adobe I/O, outside of the AEM runtime. As a result, publishing operations do not consume resources on the author tier.



Learn more about Sling Content Distribution in Helpx [3].

AEM as a Cloud Service Publishing Process

The publishing process is automated, including automatic self-configuration when publish nodes are added, removed, or recycled during runtime. A single publish or unpublish request can include multiple resources but will return a single status applied to all. As an atomic operation, it succeeds for all resources in the AEM Publish Service or fails for all. This ensures that the resources within the AEM Publish Service are never in an inconsistent state.

For more information about the AEM as a Cloud Service publishing process, see [4].

Golden Master

The golden master is a specialized publish node, never accessed by any end user, from which all the nodes of the publish service are created. It automates the lifecycle of publish nodes. Maintenance operations such as compaction are performed on the content repository attached to the golden master.

Publish nodes are recycled daily. As a result, publish nodes do not require any sort of routine maintenance or downtime.

Exercise 1: Publish a page

Scenario: An author has published a page. The DevOps personnel validate the page publication using the Distribution tools and logs.

Prerequisites:

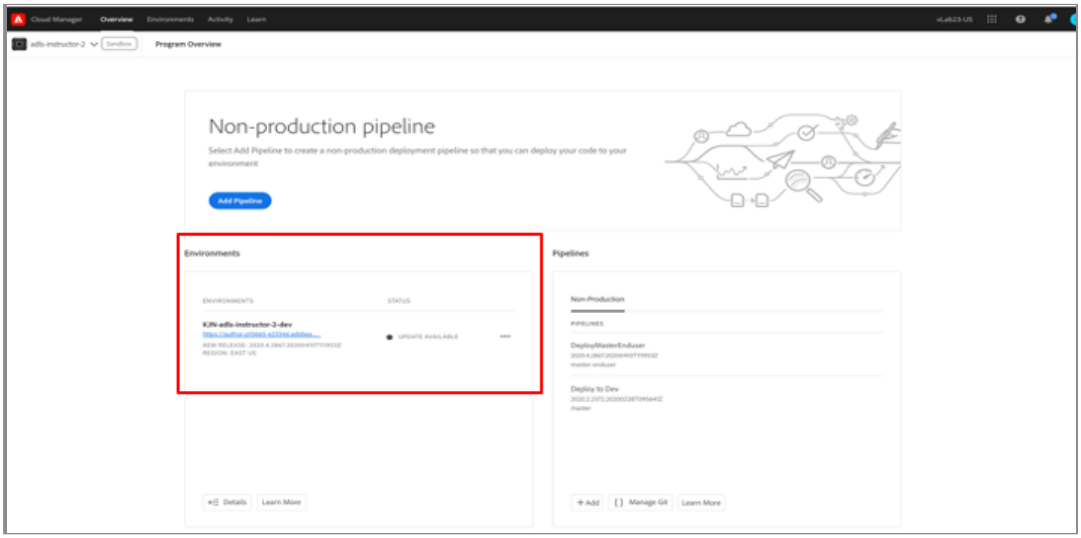
- Author and Publish services
- WKND site and associated templates have been published.

This exercise includes the following tasks:

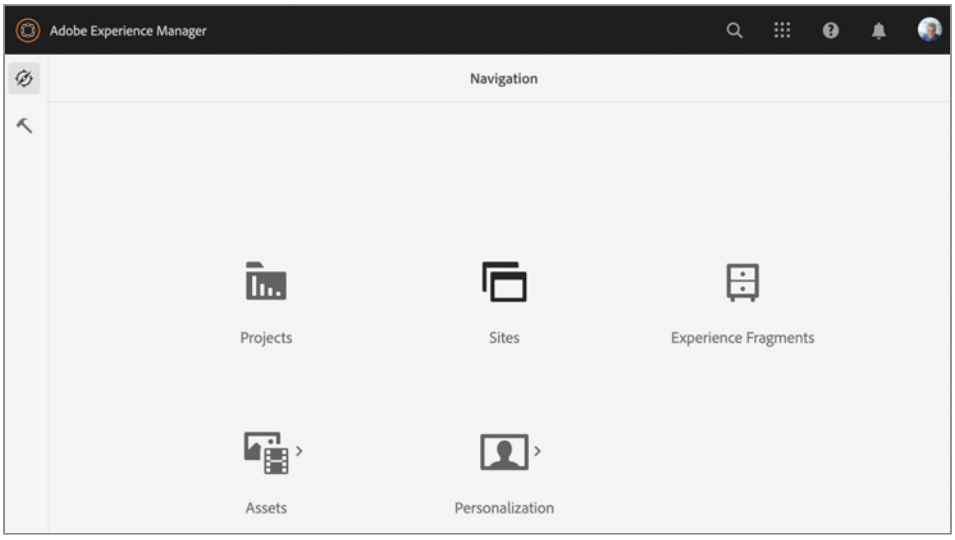
1. Create a new page in the WKND US site
2. Publish the page
3. Verify page publication

Task 1: Create a new page in the WKND US site

1. Use the URL given to you by your instructor to navigate to the **Program Overview** page for your Cloud Manager Program, as shown:



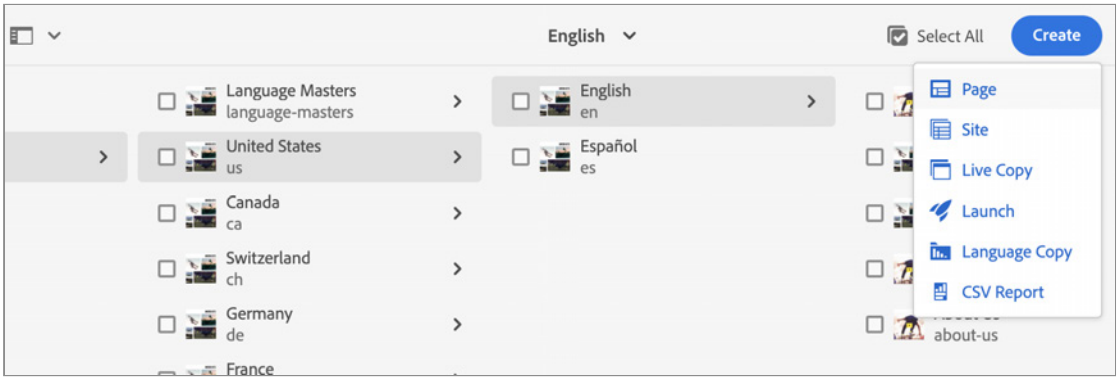
2. In the **Environments** section of the Overview page, click the author service link. You are logged into AEM, as shown:



3. If you have not already logged into the author service in your Program environment, use the credentials given to you by your instructor to log into AEM.

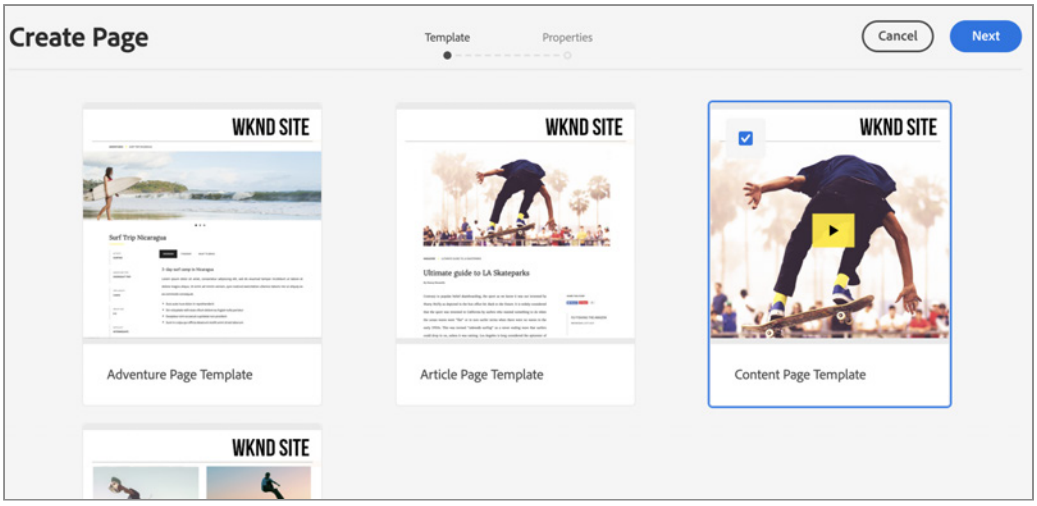
4. Using Global Navigation, navigate to **Sites** > **WKND Site** > **United States** > **English**.

5. Click the **Create** button and select **Page** from the dropdown menu, as shown:

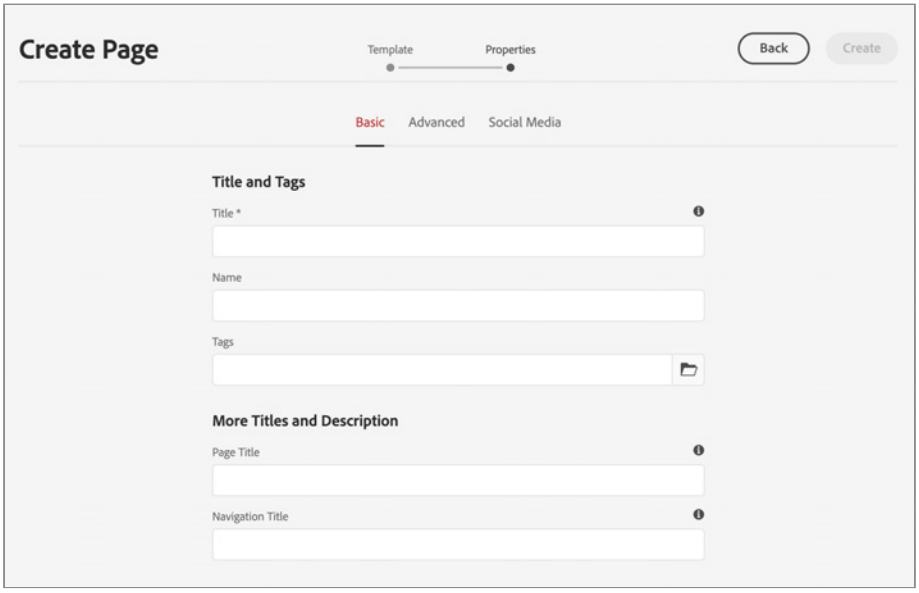


The **Create Page** window opens.

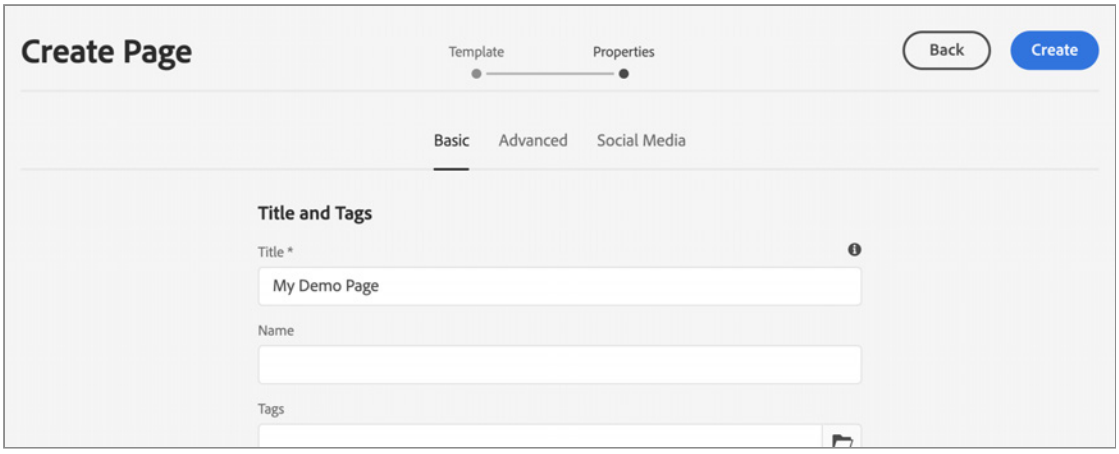
6. Select the **Content Page** template and click **Next**, as shown:



The **Create Page** Properties window opens, as shown:



7. In the Create Page dialog, enter the title as **My Demo Page** and click **Create**, as shown:

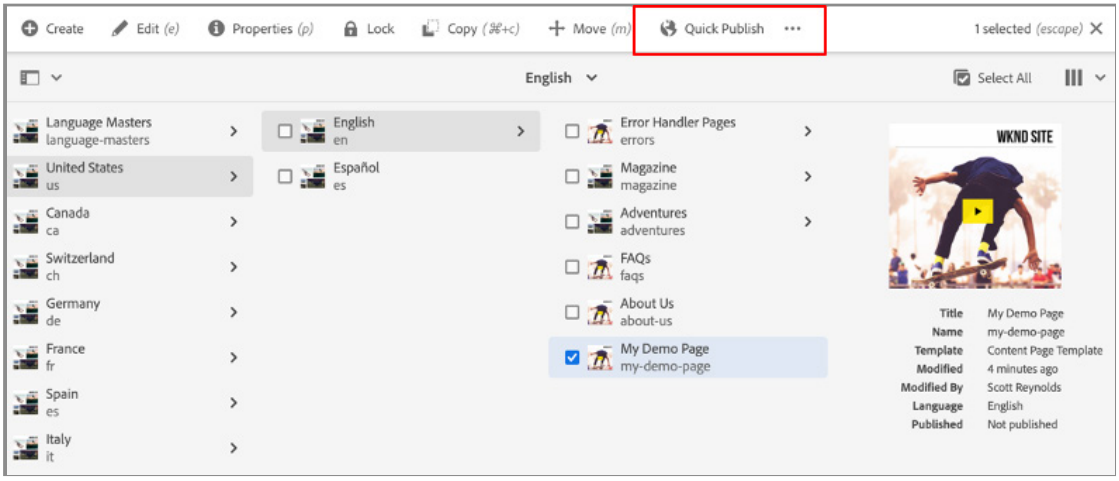


8. Click **Done** in the Success window. The Demo Page is added, as shown:

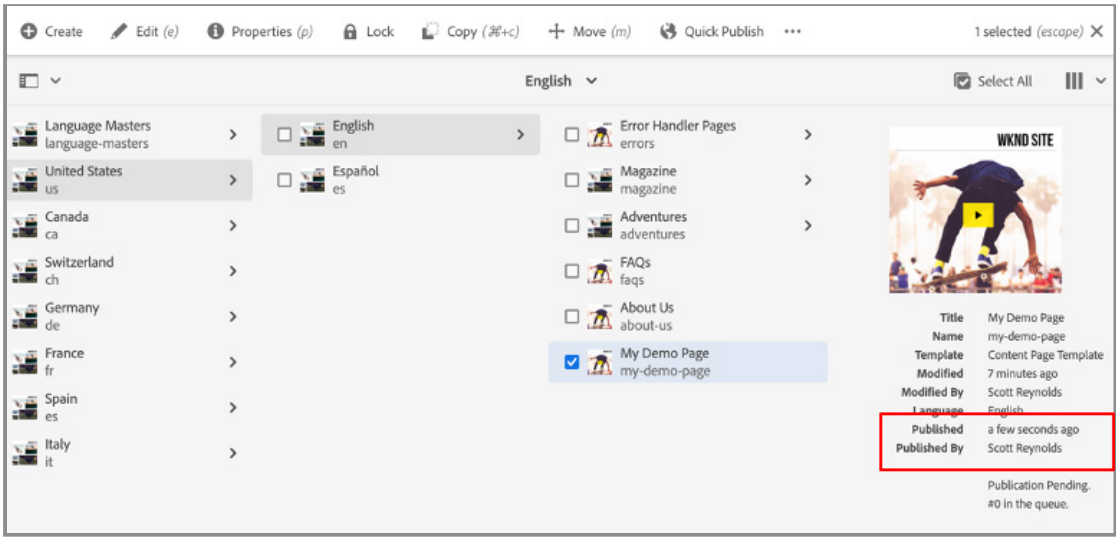
Task 2: Publish the page

1. Select the **My Demo Page** that you have just created.

2. From the Action bar, click **Quick Publish** to publish the page, as shown:



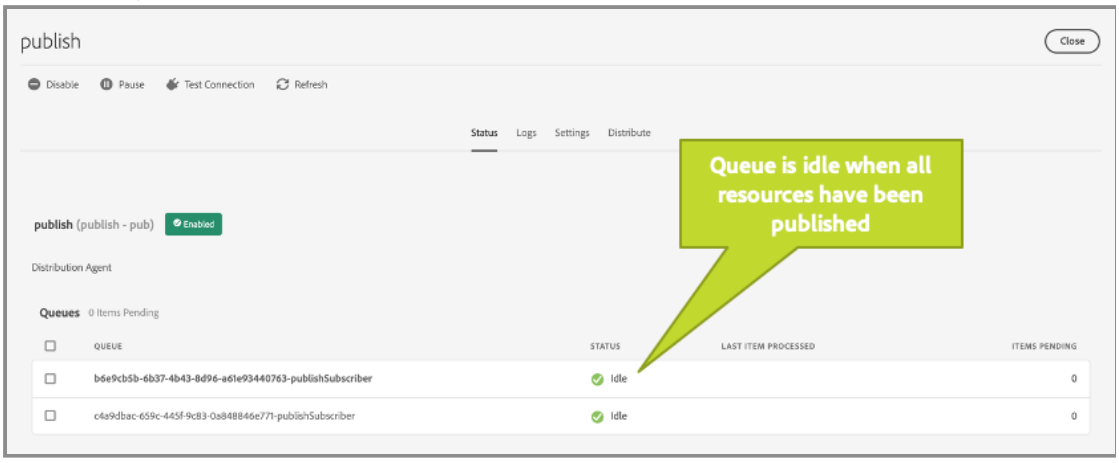
3. In the Quick Publish Window, click **Publish** to publish the page and all references. The Page is published, as shown:



Notice that the page metadata shows that the page has been published.

Task 3: Verify page publication

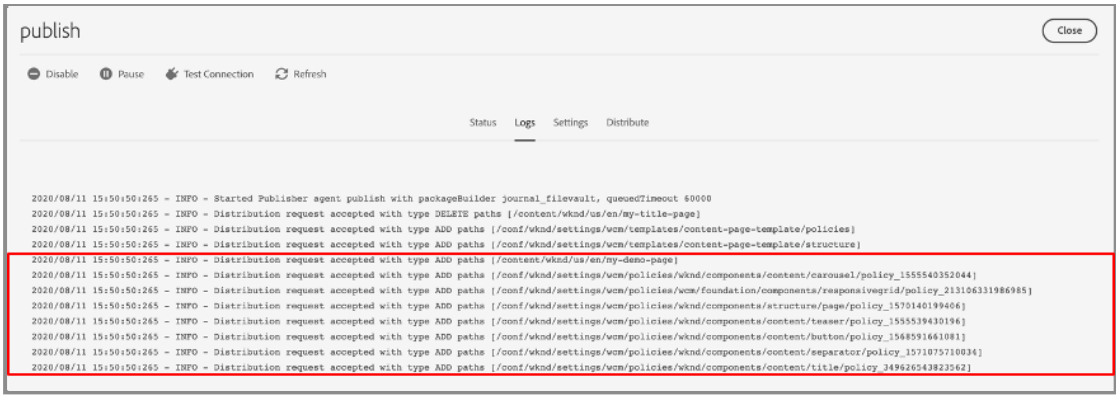
1. Using Global Navigation, navigate to **Tools** > **Deployment** > **Distribution** > **publish**. The **Publish** page opens , as shown:



2. Click one of the queues to open it and verify that the queue is empty, as shown:

3. Close the queue window.

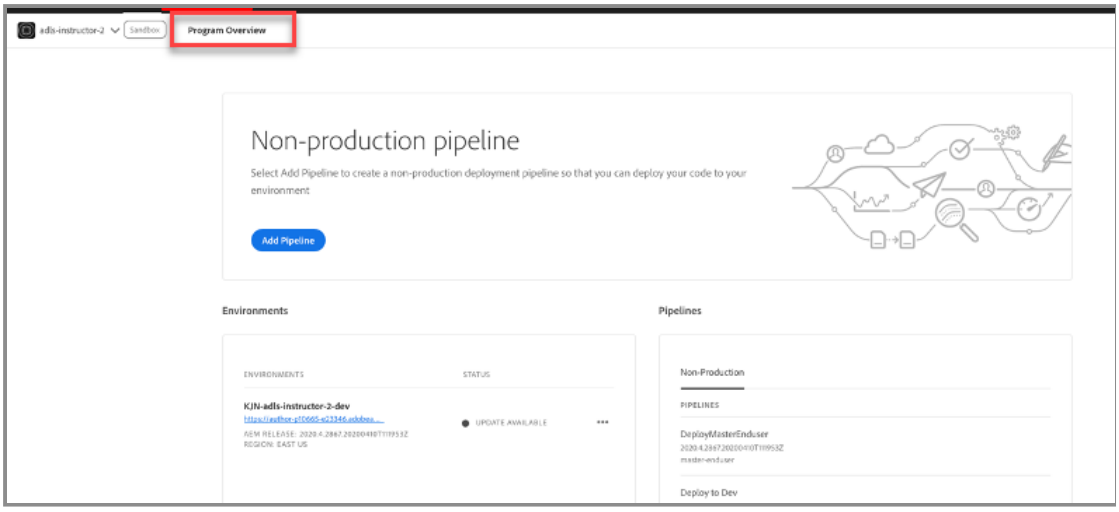
4. Click the **Logs** tab to open the distribution log, as shown:



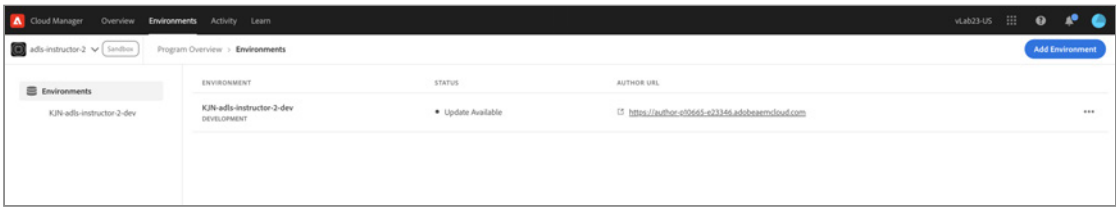
You will notice that the **My Demo Page** has been published along with any unpublished resources, for example component policies associated with the template.

5. Close the **Logs** tab.

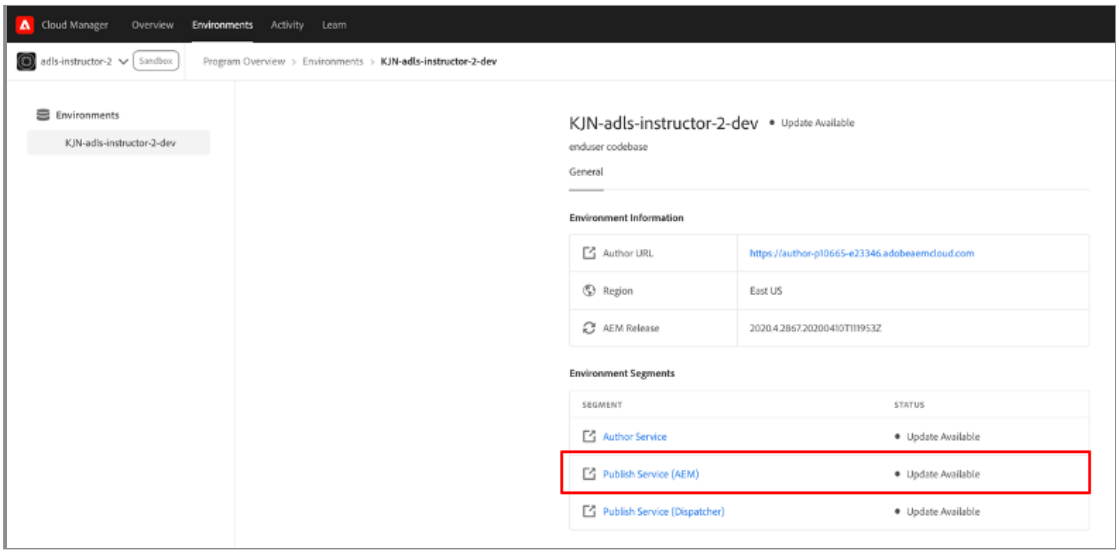
6. In your browser, return to the Cloud Manager **Program Overview** tab, as shown:



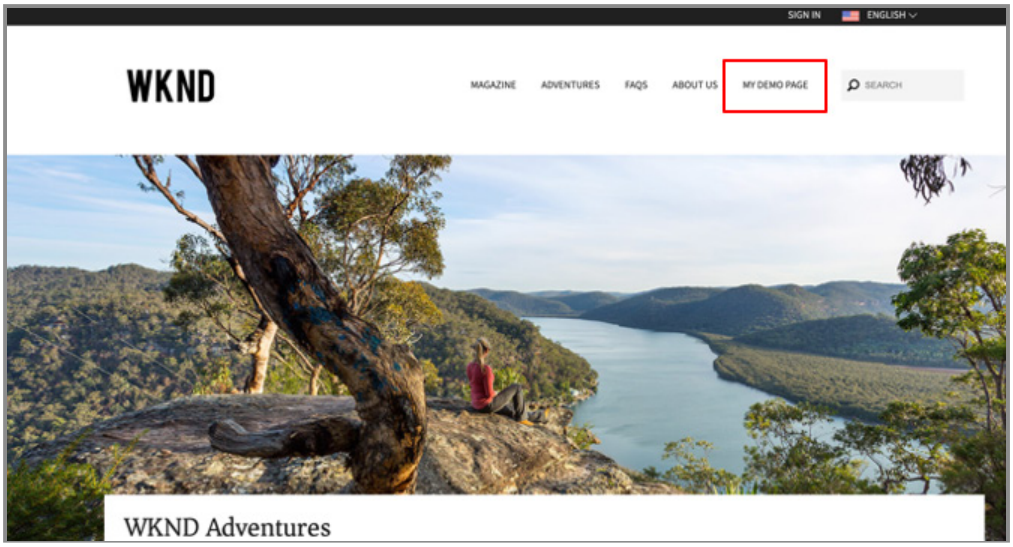
7. Click **Environments** in the black bar at the top. This will list the environments defined for your program, as shown:



8. Click the desired environment to open the information page for that environment, as shown:



9. Click the Publish Service (AEM). This will open the root page for the website in the publish tier, as shown:



10. Observe the link for the newly created **My Demo Page** at the top navigation.
    You have learned how to publish a page and then verify the publication using the Distribution Agent and its queues.

Exercise 2: Publish a tree of nodes from JCR

Scenario: A major release of the site content is complete and ready to be published. A devOps team member needs to publish the new site content and associated templates.

Prerequisites:

- Author and Publish services

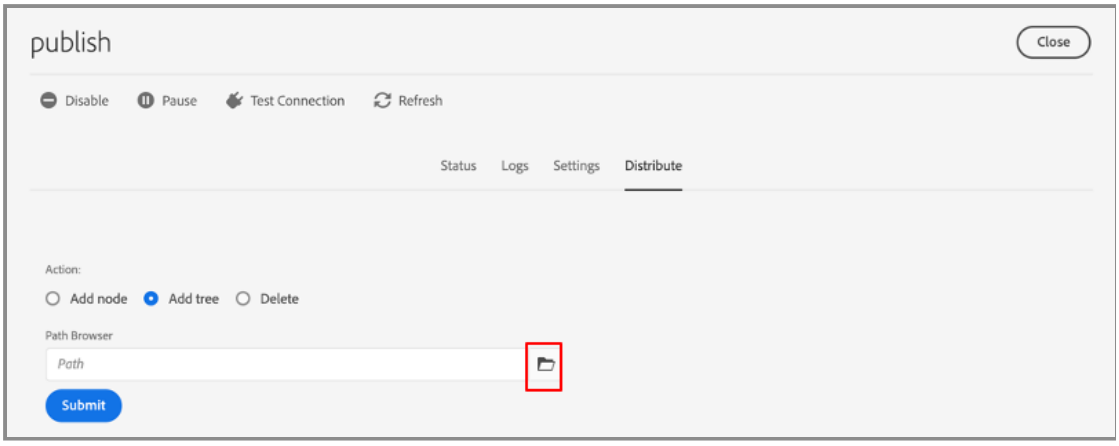
1. On a browser, using Global Navigation, navigate to **Sites** > **WKND Site** > **United States** > **English**.

2. Using the skills that you have learned, create a sub-tree of pages under the English page with the following titles:

   - **Festivals**
     - **Berlin Jazz Festival**
     - **Sonoma Wine Festival**

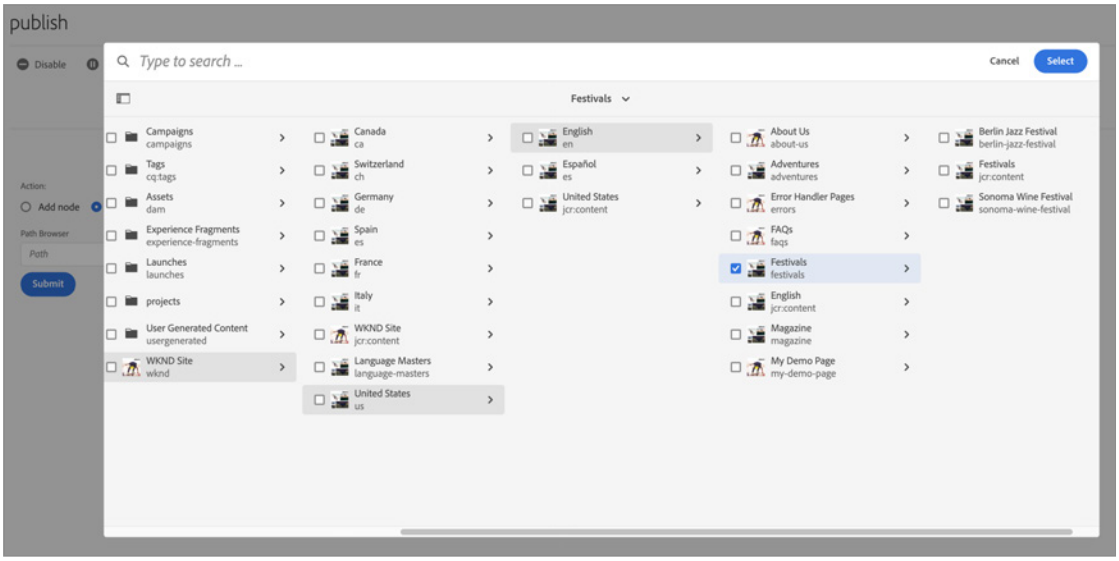
3. On a browser, using Global Navigation, navigate to **Tools** > **Deployment** > **Distribution** > **publish**. The **Publish** page opens.

4. Open the **Distribute** tab and select **Add Tree**, as shown:

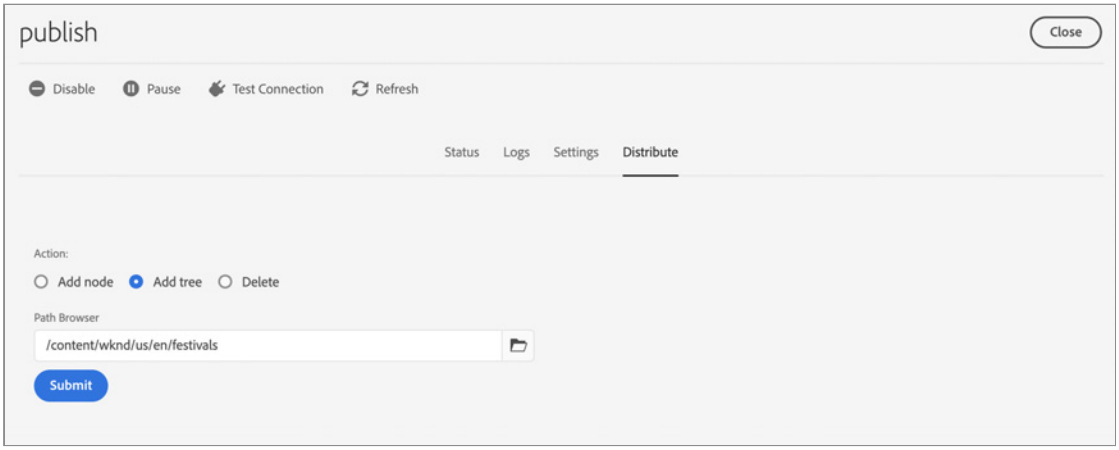


   The Path Selection dialog box opens.

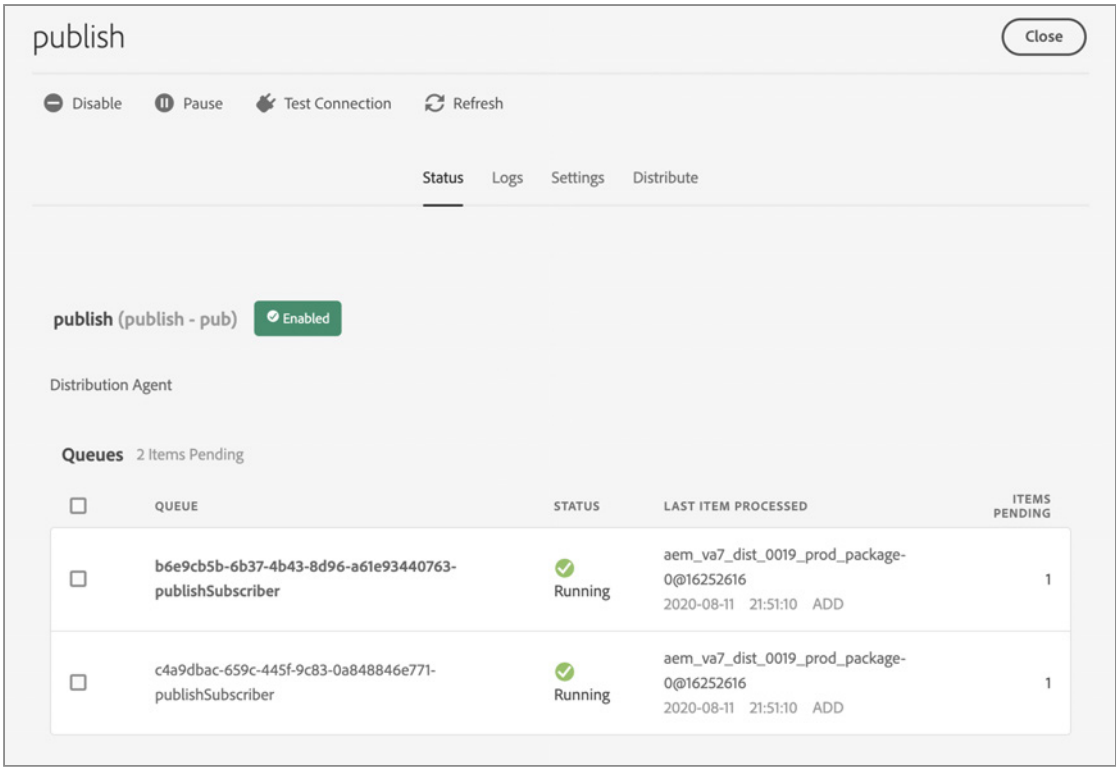
5. Using the Path Selection dialog box at the right of the Path Browser, navigate to **content** > **WKND Site** > **United States** > **English**,as shown:



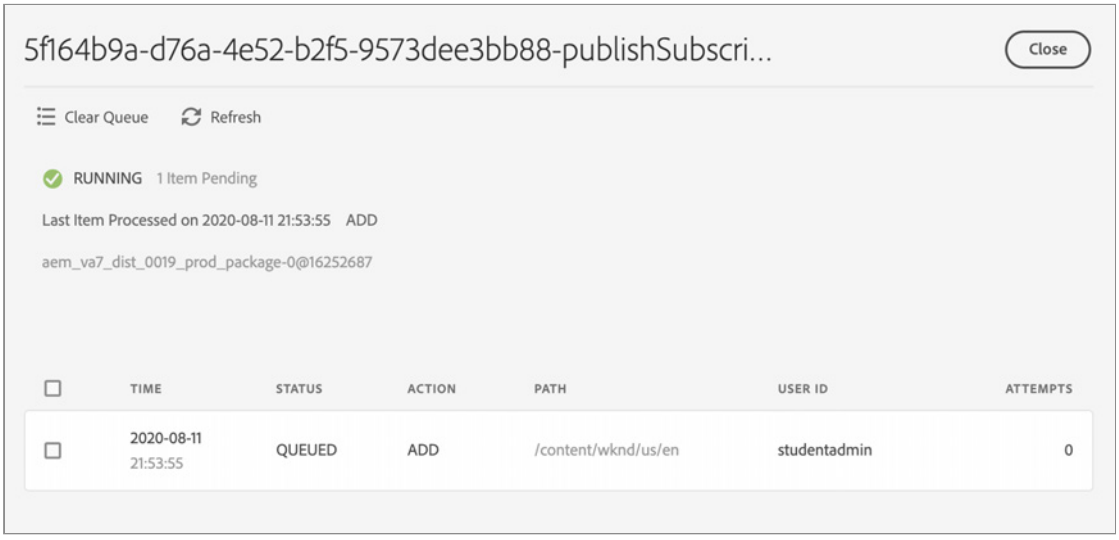
6. Select **Festivals** and click the **Select** button to select the path, as shown:



7. Click **Submit**. You will notice the queues are now running, as shown:



8. Click one of the queues, as shown:



Once the queues are idle, the content distribution is complete. You can verify that the pages were published by checking for them in the publish service.

9. On the browser tab with the publish service, find the **Festivals** link at the top navigation of the English page.

10. Click the **Festivals** link to open the Festivals page.

11. There is no secondary navigation in the WKND site, but you can check that the child pages were published by adding the child page name to the URL. For example, https://publish-p10665-e23346.adobeaemcloud.com/content/wknd/us/en/festivals/berlin-jazz-festival.html

Caution: Your URL will be unique to the publish service in your environment.

```
You have successfully published an entire sub-tree.
```

Content Delivery Network (CDN)

AEMaaCS is shipped with a built-in CDN. The main purpose of CDN is to reduce latency by delivering cacheable content from the CDN nodes at the edge, near the browser. It is fully managed and configured for the optimal performance of AEM applications.

The AEM-managed CDN will satisfy most customer's performance and security requirements. The customers can optionally point to it from their own CDN, which they will need to manage. Customer-managed CDNs are allowed on a case-by-case basis, based on meeting certain prerequisites including, but not limited to, the customer having a legacy integration with their CDN vendor that is difficult to abandon.

Learn more about the AEM as a Cloud Service Content Delivery Flow the Helpx [5].

AEM Dispatcher

The AEM dispatcher is an Apache HTTP Web server module that provides a security and performance layer between the CDN and the AEM publisher. The dispatcher should be part of the local development setup. The dispatcher is AEM's caching tool. This tool also provides a security layer and helps protect your AEM publisher from attack.

The dispatcher contains the mechanisms to generate and update static HTML, based on the content of the dynamic site. Among other items, you can specify in detail which:

- Documents are stored as the static files and which are always generated dynamically
- Requests are allowed through the web server
- Client headers are passed to the browser

For more information about configuring the AEM dispatcher, see [6].

For more information about the AEM dispatcher in the Cloud, see [7].

Exercise 3: Validate the dispatcher configuration locally (Optional)

Scenario: Before deploying the application release candidate to the Cloud Manager pipeline, the dispatcher module of the AEM project must be configured.

Prerequisites:

- AEM Project defined, based on Maven AEM Archetype 23
- AEM SDK with the correct version
- local Author and Publish services

Note: At this time, this exercise can only be done on a local Linux or Mac OS install. Your instructor will demonstrate this exercise.

This exercise includes the following tasks:

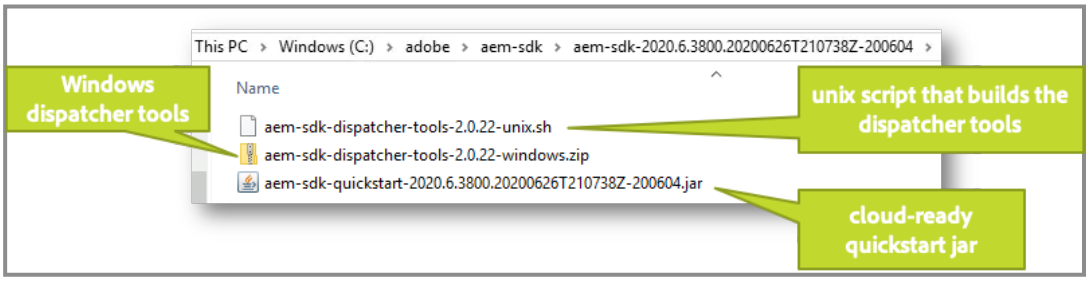
1. Unpack the AEM SDK dispatcher tools
2. Configure the dispatcher module for your AEM project
3. Locally validate the dispatcher configuration

Task 1: Unpack the AEM SDK dispatcher tools

Examine the configuration files that make up the AEM Project dispatcher module. Note the file names and how they map to the configuration files that will be deployed to the webserver: httpd.conf and dispatcher.any. Open the files and notice the files that can be modified and the files that cannot be modified.

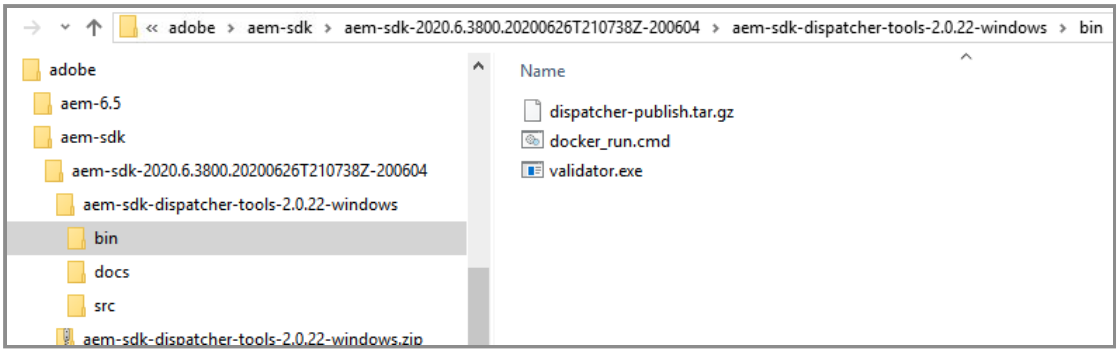
1. Using a **File Explorer**, navigate to your AEM SDK, for example **C:\adobe\ aem-sdk**.
   If you have already unpacked the AEM SDK, go to step 3.
2. The AEM SDK come packaged as an archive. The archive file name format is **aem-sdk-[version].zip**. Extract the SDK.

Caution: The file must be extracted. Attempting to use the File Explorer to navigate down into the archive will result in failure of the dispatcher tools.



---

3. Extract the windows dispatcher tools, as shown:

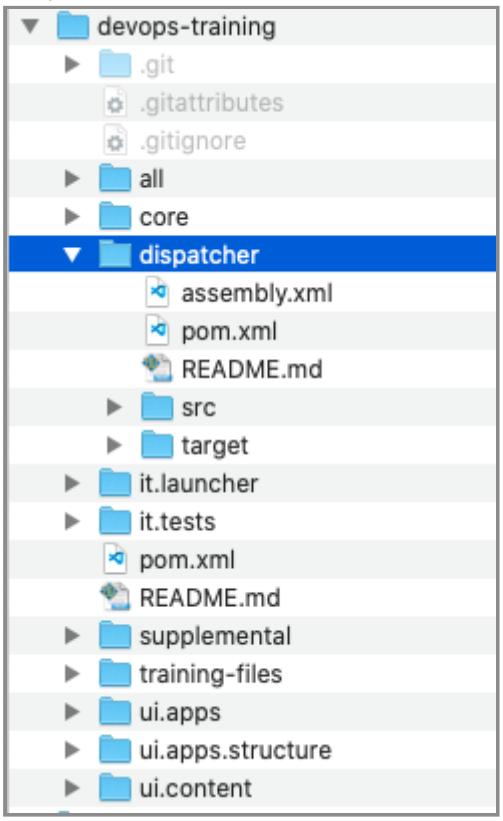


This will create a folder with the same name as the zip file.

You will notice a \bin folder that contains the docker_run_cmd script, which will start the Windows docker image containing the configured Apache web server, and the validator.exe application that you will use to locally validate the dispatcher configuration. The \src folder contains a sample dispatcher configuration. There is also a \docs folder containing documentation and helpful guides.

Task 2: Configure the dispatcher module for your AEM project

1. Using a **File Explorer**, navigate to the dispatcher module in your AEM project. For example, \dispatcher.



2. Open the **pom.xml** file and take note of the < build > element, as shown:

3. Open the **assembly.xml** file. Note that the element includes all the files under the dispatcher module's src folder, as shown:

4. Navigate to the dispatcher module configuration files, < **AEM Project** >**\dispatcher\src**

   a. Notice the differences between the earlier versions of the dispatcher configuration files (**conf.d** and **dispatcher.any**) and the AEM as a Cloud Service dispatcher configuration files.

   b. Identify the purpose of each file.

   c. Identify which files can be changed and what cannot.

Task 3: Locally validate the dispatcher configuration

Dispatcher configuration files can be validated locally through the use of the validator tool provided as part of the AEM Dispatcher SDK.

1. Using a **File Explorer**, navigate to the AEM SDK folder, for example, **C:\adobe\aem-sdk**.

2. Open a terminal window in the < AEM SDK Windows Tools > folder. For example: C:\adobe\aem-sdk\aem-sdk-[version]\aem-sdk-dispatcher-tools-[version]

3. Use the following command to run the validator tool and learn what the subcommands are

```
$ bin\validator.exe
```

```
C:\adobe\aem-sdk\aem-sdk-2020.6.3800.20200626T210738Z-200604\aem-sdk-dispatcher-tools-2.0.22-windows>bin\validator.exe
Cloud manager validator 2.0.19

    Subcommands are:

    - any:        dump dispatcher configuration files
    - collect:    collect all Apache directive used in multiple configuration files
    - dispatcher: validate one or more dispatcher configuration files
    - httpd:      validate one or more httpd configuration files
    - full:       validate both httpd and dispatcher configuration files
    - whitelist:  prints the whitelisted Apache directives
```

4. Run the validator to validate your AEM project dispatcher module structure and create the deployment files. The command has the form:

```
      $ validator full -d <dispatcher configuration deployment location>
<location of dispatcher package>
```

where < dispatcher configuration deployment location > is the location to write the generated dispatcher configuration files and < location of dispatcher packagae > is the location of the dispatcher configuration files in the dispatcher module.
For example:

```
   $ bin\validator.exe full -d out \Users\adlsadmin\devops-
training\dispatcher\src
```

```
kjnelson:dispatcher-sdk-2.0.23/ $ ./bin/validator full -d out /Users/kjnelson/AdobeAEM/devops-training2/dispatcher/src
Cloud manager validator 2.0.21
2020/08/17 11:17:15 No issues found
kjnelson:dispatcher-sdk-2.0.23/ $
```

Note: You have specified the **dispatcher configuration deployment location** to be in the SDK dispatcher tools folder for convenience. You can specify whichever location you wish. This is the set of files that will be deployed to the webserver in the docker image.

Note: The dispatcher configuration files will not be written to the deployment folder until all errors are resolved.

---

5. Navigate to the folder and verify that the configuration files have been generated, as shown:

You have validated and generated dispatcher configuration files so that they can be tested locally before the AEM Project is deployed to the Cloud Manager pipeline.

Demo: Test the Dispatcher configurations (Optional)

Scenario: Once the final dispatcher configuration files are generated, the devOps group wants to test the configuration to ensure that the values are set properly.

Prerequisites:

- Local machine not virtual machine
- Docker is installed and configured
- local AEM publish instance

Note: This exercise can only be done on a local install. The webserver is deployed in a docker image. The docker image (a virtual image) cannot be run inside another virtual image, such as the ReadyTech image is used for our virtual classroom. Your instructor will demonstrate the steps.

This exercise includes the following tasks:

1. Set up the local publish instance.
2. Test the dispatcher configuration.

Task 1: Set up the local publish instance

1. Start a local AEM cloud-ready quickstart jar as a **publish** instance, running on port 4503.
2. Using a File Explorer, navigate to < AEM Project >.
3. Using the Maven skills you have learned, deploy the AEM Project that you built from AEM Maven Archetype 23 to the publish instance.

```
$ mvn clean install -PautoInstallPackagePublish -Padobe-public
```

Task 2: Test the dispatcher configuration

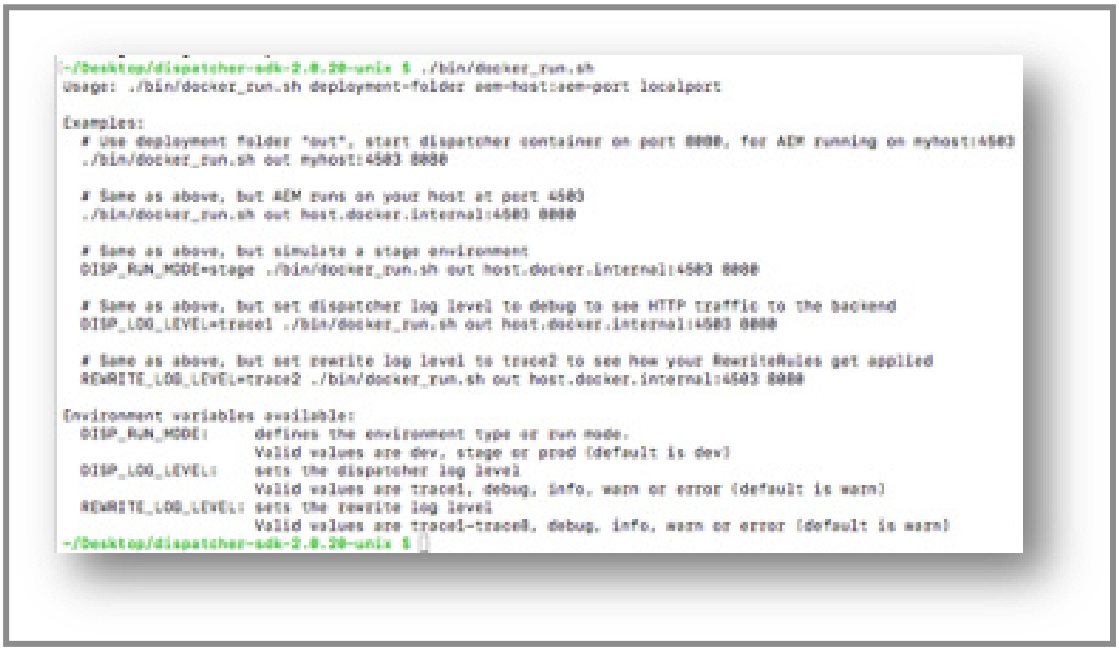
Deploy the dispatcher configuration to the Docker image:

Caution: This exercise can only be done if you are not running AEM in a virtual image, for example ReadyTech. Local testing of the dispatcher configuration for AEM as a Cloud Service is accomplished by deploying the generated dispatcher configuration files to a local docker image running an Apache HTTP Web server with the Dispatcher module. If you are running in a virtual image, your instructor will demonstrate this functionality.

Note: You must obtain your own Docker login to run Docker.

1. Run the docker script to explore the usage examples, as shown:

`$ ./bin/docker_run.sh` OR `$ bin\docker_run.cmd`



2. Run the docker script to deploy the dispatcher configuration files to the docker image containing the Apache HTTP web server. The command has the form, as shown:

`$ <dispatcher SDK location>/bin/docker_run.sh <dispatcher configuration deployment location>  <local machine IP address>:4503  <docker webserver port>`

For example:

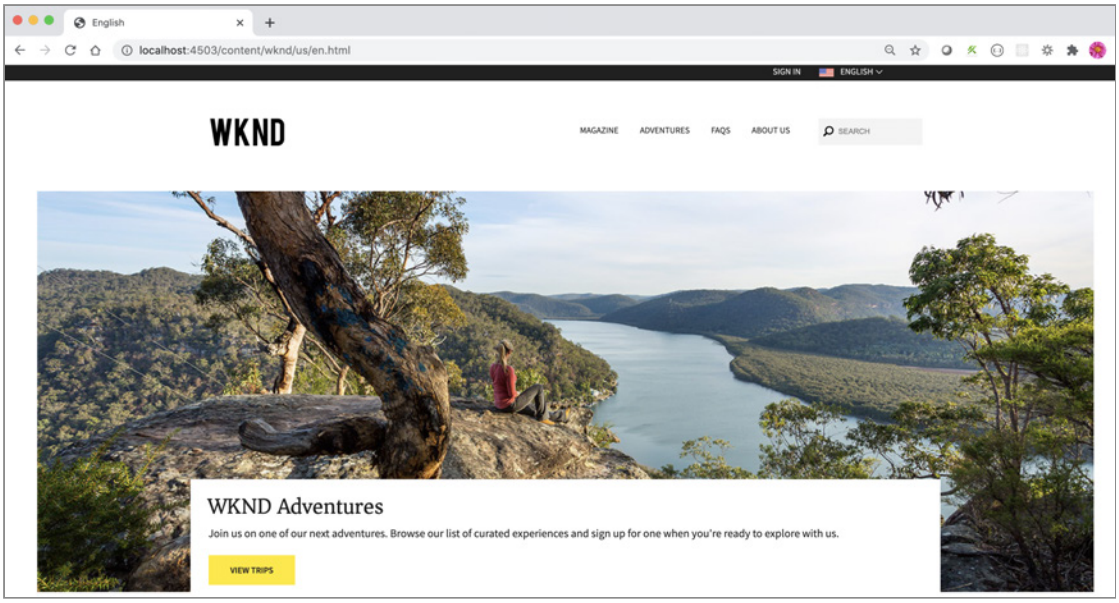
`$ ./bin/docker_run.sh out 192.168.0.04:4503 8080` OR `$ bin\docker_run.cmd out 192.168.0.04:4503 8080`

Caution: You must use the IP address of your local machine. It will not work with 'localhost'.

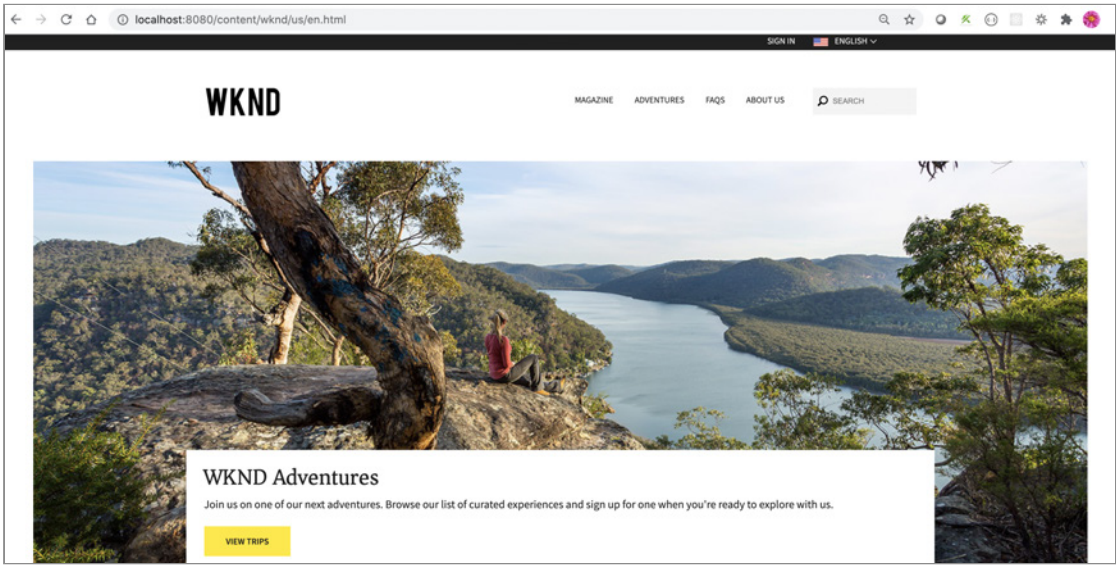
Tip: Once you start the docker image, do not resize the terminal window. It will cause the image to exit with a SIGWINCH error.

Note: The < dispatcher configuration deployment location > is the location where the generated configuration files were written by the validator script.

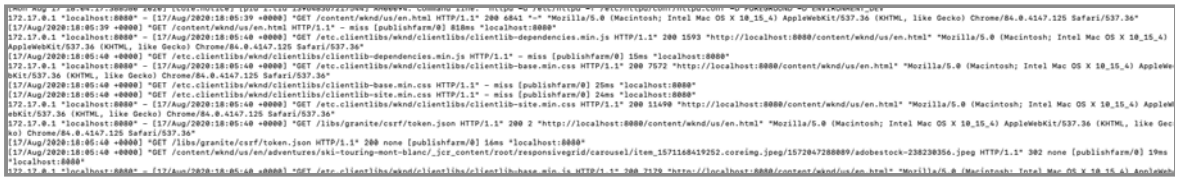
3. Access the publish instance web site http://localhost:4503/content/wknd/us/en.html directly through the local publish instance, as shown:



4. Access the web site through the web server/dispatcher http://localhost:8080/content/wknd/us/en.html, as shown:



5. Take note of the messages logged to the terminal window where you initiated the docker script, as shown:



You have successfully started the dispatcher-enabled webserver inside the provided docker image. In addition, you have accessed the website through the webserver and validated the webserver configuration.

References

1. AEMaaCS Architecture. https://docs.adobe.com/content/help/en/experience-manager-cloud-service/core-concepts/architecture.html ↩

2. Replication Service. https://docs.adobe.com/content/help/en/experience-manager-cloud-service/operations/replication.html ↩

3. Apache Sling Content Distribution. https://sling.apache.org/documentation/bundles/content-distribution.html ↩

4. Publishing Content. https://docs.adobe.com/content/help/en/experience-manager-cloud-service/core-concepts/architecture.html#content-distribution ↩

5. AEM Content Delivery Flow. https://docs.adobe.com/content/help/en/experience-manager-cloud-service/implementing/content-delivery/overview.html ↩

6. AEM Dispatcher Overview. https://docs.adobe.com/content/help/en/experience-manager-dispatcher/using/dispatcher.html ↩

7. AEM Dispatcher in the Cloud. https://docs.adobe.com/content/help/en/experience-manager-cloud-service/implementing/content-delivery/disp-overview.html ↩