Best practices for working with XML Documentation for Adobe Experience Manager

December 10, 2019

Contents

Introduction

This guide provides best practices for working with XML Documentation for Adobe Experience Manager (later referred to as *XML Documentation solution* in this guide). Following these best practices will help you set up, organize DITA content, translate and publish content, and develop processes around content creation, management, and publishing.

This guide is intended for the following type of audiences:

- **Authors**: Responsible for creating DITA content.
- **Publishers**: Responsible for running the publishing task to generate output in various formats.
- **Administrators**: Responsible for installing and managing XML Documentation solution.

Content structure

Information in this guide is organized as follows:

- *Recommendations for performance optimization*: This topic contains the recommendations for optimizing the performance of XML Documentation solution.
- *Content organization*: This topic describes the best practices for file naming, content referencing, and versioning.
- *Content migration and upload*: This topic describes the best practices for migrating content from any third-party DITA authoring tool, getting unstructured content from FrameMaker, or uploading DITA content directly into AEM.
- *Workflow offloading*: This topic describes the best practices around workflow offloading in AEM.
- *DITA-OT configurations*: This topic contains the best practices for DITA-OT configuration.
- *Publishing*: This topic contains information about optimizing your publishing process.
- *Other recommendations*: This topic contains information about optimizing other tasks and workflows in XML Documentation solution.

Recommendations for performance optimization

This topic talks about the performance optimization tasks that you may perform in your setup to achieve optimal performance. The following list covers the type of configurations required to be made, whether is it mandatory or optional, the exact change required in the system, when should you perform the required configuration,co and the final result that you achieve by making the change.

Configure data store

Is this configuration required?

Yes, this is a mandatory configuration.

***What is the change*?**

Set the `minRecordLength` property to a value of `100` under the configuration `org.apache.jackrabbit.oak.plugins.blob.datastore.FileDataStore`. For more information about file date store and S3 data store, see the Configuring node stores and data stores in AEM 6 article.

***NOTE:** For S3 data store, the cost for maintaining content in data store depends on the number of requests as well. Therefore, when doing this setting with S3—setup cost per request and cache size should also be considered.*

When to configure?

After the initial set up, but before any content is migrated. You must perform this change even on an existing system, which ensures that all new content is stored in data store instead of segment store.

Result of this change

The DITA files are saved in data store rather than the segment store. This keeps the segment store size under the recommended limits, which improves the responsiveness of the system.

Update Lucene index

Is this configuration required?

Yes, this is a mandatory configuration.

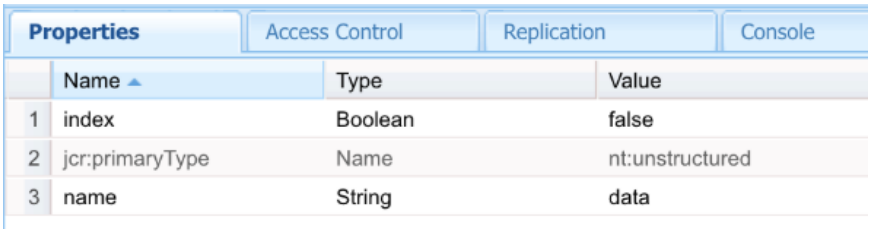
***What is the change*?**

Make the following changes in the Lucene index configuration:

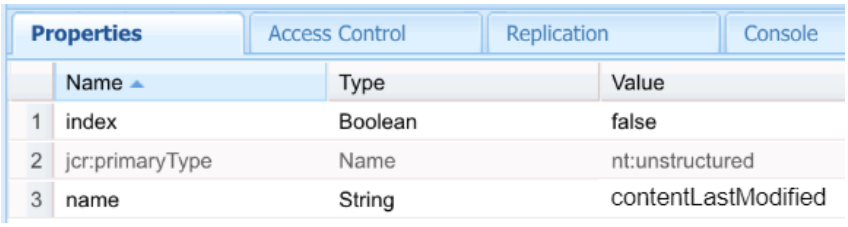
- Exclude `/content/fmdita` from `oak:index/lucene`.

NOTE: XML Documentation solution never uses Lucene indexes to search for content in `/content/fmdita` node.

- Add and then exclude `data` property from the index. The definition of `data` property along with other parameters is shown in the following screenshot:



- Add and then exclude `contentLastModified` property from the index. The definition of `contentLastModified` property along with other parameters is shown in the following screenshot:



When to configure?

If you are making this change on a new system before migrating content, then only updating `oak:index/lucene` is required. Otherwise, on an existing system where the content is already migrated, then after making the change in `oak:index/lucene`, rebuild indexes for Lucene (*which might take few hours to complete*).

Result of this change

This change prevents `/content/fmdita` node from getting indexed and stored in the segment store. The data and contentLastModified property update ensures that re-indexing is only performed at the updated topic and not across all DITA maps where it is being referred.

Java memory optimization

Is this configuration required?

Yes, this is a mandatory configuration.

***What is the change*?**

The JVM start parameters should be carefully tuned based on the infrastructure and the disk size. It is recommended that you consult Adobe Support to get help to access the ideal configuration. You may, however, try out the following configurations yourself:

– Set the JVM heap size to a minimum of 1/4th of the total available memory. Use the parameter `-Xmx<size>` to set the heap memory size.

– Ideally, the size of the heap `dump == xMx`.

– Configure the following parameters to ensure that GC does not run too frequently:

- `-XX:NewRatio`

- `-XX:ParallelGCThreads`
- `-XX:ConcGCThreads`

When to configure?

If you are making this change on an existing system, you need to restart the system. In case of a fresh installation, this change should be made in the start script (`.bat` or `.sh`) file before the system is started.

Result of this change

This results in optimal heap size and regulated execution of GC.

Client library minification on Author instance

Is this configuration required?

No, this is an optional configuration.

***What is the change*?**

The client libraries should be set to minified in the Authoring instances. This makes sure that there are less bytes to download when a user is browsing the system from different locations. To make this change, set the configuration in **HTML Library Manager** from the Felix console.

When to configure?

This can be done at run time through Felix console or via code deployment.

Result of this change

This change improves the load time of pages on Author instance as less bytes are transferred for loading the client libraries.

Configure concurrent publishing threads

Is this configuration required?

Yes, this is a mandatory configuration based on your use case.

***What is the change*?**

This change is required if you are using DITA-OT to publish output and a number of concurrent publishing threads is also defined.

By default, XML Documentation solution sets the publishing threads to the number of CPUs+1. However, it is recommended to set this value to half (1/2) or one-third (1/3) of the total number of CPUs. To do this, set the **Generation Pool Size** property under the configuration `com.adobe.fmdita.publish.manager.PublishThreadManagerImpl` as per the recommendations.

When to configure?

This can be done at run time through Felix console or via code deployment.

Result of this change

This change ensures that on a running Author instance all resources are not allocated for the publishing operations. This keeps the system resources available for authors as well, which results in a better user experience.

Configure batch size of nodes for AEM Site output generation

Is this configuration required?

Yes, this is a mandatory configuration based on your use case.

***What is the change*?**

This change is required if you are generating AEM Sites output.

Set the **Limit AEM Site Pages in Heap** property under `com.adobe.fmdita.config.ConfigManager` to a number based on your system's configuration. This property defines the batch size of nodes to be committed when the site pages are generated. For example, on a system with a larger number of CPUs and heap size, you can increase the default value from `500` to a larger number. You need to test run with the changed value to come to an optimal value for this property.

When to configure?

This can be done at run time through Felix console or via code deployment.

Result of this change

An increased number of the **Limit AEM Site Pages in Heap** property optimizes the AEM Site output generation process.

Patch Xerces Jar while using custom DITA-OT for publishing

Is this configuration required?

Yes, this is a mandatory configuration based on your use case.

***What is the change*?**

This change is required only if you use custom DITA-OT for publishing output.

Replace the Xerces Jar file in your custom DITA-OT package with the one shipped OOTB. The default OOTB `xercesImpl-2.11.0.jar` file is available within the `/etc/fmdita/dita_resources/DITA-OT.zip` file. Ensure that you rename the `xercesImpl-2.11.0.jar` file to match the old Xerces Jar file being replaced.

When to configure?

This can be done at run time.

Result of this change

This change reduces the publishing time and memory utilization while publishing DITA maps with a large number of topics.

Optimize number of post-processing threads

Is this configuration required?

Yes, this is a mandatory configuration based on your use case.

***What is the change*?**

This change is required if you are bulk uploading DITA content.

Set the **Post Process Threads** property under `com.adobe.fmdita.config.ConfigManager` to `1`.

When to configure?

This can be done at run time.

Result of this change

This change reduces the post-processing time on bulk upload of DITA files.

Content organization

This topic contains the best practices for naming DITA files, using content references, creating modular content, and versioning content.

File names

- Avoid spaces, apostrophe, or braces in filenames to simplify file references in your content. Remember that all references in DITA should be valid URLs. Having spaces in a filename will require references to be URL-encoded.
- Avoid non-ASCII characters in filenames. Use only alphanumeric filenames with underscore or hyphens.
- Avoid using `.xml` and instead use `.dita` file extension for your DITA files. Though doing this is not required, it has several benefits:
  - Using `.dita` file extension makes the files easier to recognize.
  - During the publish process, other `.xml` files might get generated to contain the metadata related to the publishing process. Using `.dita` extension avoids any conflict with the generated files and also avoids the possibility of publishing process overwriting one of the topic files.
  - Compulsorily use `.ditamap` or `.bookmap` extension for your map files. Not doing this will result in failures during publishing. For more information about DITA map, search for the *Definition of DITA maps* topic in OASIS documentation.

Folder hierarchy

- Do not store your content within the `/content/dam/projects` folder. The `projects` folder is used for specific tasks, such as review or translation workflow
- Adobe Experience Manager is designed to work best with hierarchical content repository. Having too many *(say, a thousand)* files at the same level or within a single folder might lead to poor user experience and performance.
- Have well-categorized folder structure for storing large number of files. For example, if your guide contains hundreds of files, then create folders for each chapter within your guide and place all chapter-related files in respective folders.

References

- Avoid having broken references.

- Broken references at top level in your DITA map hierarchy might cause complete failure of AEM Site output generation. This is because the references affect how temporary directories are created during publishing.
- It is ideal to have DITA map as the topmost entity in the entire content tree. If the content is organized such that the topics and other content is present at sibling or higher level, than DITA-OT might fail to handle such content.

  DITA-OT does provide an argument `generate.copy.outer`*, whose value can be set to 3 in DITA-OT command line arguments and it could successfully handle some cases. However, it does not handle all cases of up-level references. In such cases, it is best to define another DITA map that is a level above the highest folder level of all the referenced content and let that DITA map refers to the original DITA map. Then, use this highest level DITA map to publish your content.*

Modular authoring

- Use modular authoring approach wherein one topic represents one concept or subject.
- Modularized topics are easier to reuse across various documents.
- Use sub-maps to better organize your content. For example, if there is one user guide, you can have multiple sub-maps for each top-level topic, and each sub-section within a sub-map can be one DITA topic.
- Using a single large topic file becomes hard to maintain and pose challenges in reusing content.

Versioning of content

- For checking-in or out multiple files together, make sure all selected files are in the same state—checked-in or checked-out. Similarly, all selected files you wish to check-in should be checked-out by the same user.
- If auto check-in or check-out configuration is On, then the only way to check-in a checked-out file is by closing the file in the Web Editor.
- The file check-out and check-in buttons are always shown in the toolbar even if auto-configuration is enabled. The only difference with auto-checkout configuration is enabled is that when a user opens a file in the Web Editor, it is automatically checked-out.

RELATED LINKS:

Best Practices for Assets

Best Practices for Translating Assets Efficiently

Content migration and upload

Consider the following best practices while migrating content to AEM from any third-party DITA authoring tool, unstructured content from FrameMaker, or directly uploading DITA content to AEM.

- After uploading content, it is recommended to let the post-processing workflows complete before triggering the publish workflow or any other system intensive task.
- It is important not to disable the post-processing workflows or their launchers.
- When you upload an asset in AEM, workflow to extract metadata or create renditions is launched in AEM. Therefore, when you migrate a large amount of content, do it in batches of 500 with 2-5 minutes of interval in between. The interval between the workflows reduces the load on the system.
- Post-processing workflows also create a default set of output presets on DITA maps. Also, automatic updates for links also relies on the post-processing workflow.
- When a DITA map without any output preset is added to a map collection, it appears blank on the page.
- *Set Document State* workflow sets the document state of a new DITA map or topic content as Draft. This is done immediately after uploading a file or creating a map or a topic file. This also happens if you are copying or moving DITA content. Disabling this workflow will not set the document state of Draft for any new file.
- When moving, copying, or renaming DITA content, the internal DITA references are adjusted only if the movement is triggered through the Move/Copy option in the DAM UI. If you move, copy, or rename DITA content using the JCR API, or through CRXDE, or any other API, then you must explicitly fix any references to or from the moved or copied content.
- When same content is uploaded to same location on AEM assets via WebDAV or FrameMaker, then the content gets automatically versioned. This can lead to repository bloat in cases where users don't wish to keep older versions. AEM Assets version purging should be configured in such cases.
- DITA map creation in FrameMaker to DITA conversion works only for FrameMaker Publishing Server with FrameMaker (2017 release) Update 1 and later builds.
- Use in-band metadata wherever metadata is defined as inline *attribute-value* pairs.

NOTE: Elements metadata node gets created only till XML Documentation solution build 2.1. This feature will be removed from future builds. Users upgrading to newer builds from 2.1 or later will have to use a script to get rid of existing elements metadata.

Workflow offloading

XML Documentation solution's post-processing and publishing workflows do not support the traditional AEM model of workflow off-loading using sling job queues. However, load sharing can still be achieved by having separate instances for uploading assets, authoring, and publishing (or production).

- **Asset upload instance**: is where all assets are uploaded and processed. Replication agents can be used to replicate processed assets (even DITA content) to the authoring instance.
- **Authoring instance**: is where all authoring takes place. Replication agents can be configured to replicate reviewed and approved content to the production instance and published from there.
- **Production instance**: is where the publishing takes place. Depending on requirements this can be replaced with a publishing instance which takes care of publishing DITA content and replication agents replicate the published pages to a different server which then acts as production server.

RELATED LINKS:

Offloading Jobs (in AEM documentation)

DITA-OT configurations

The following guidelines are for configuring DITA-OT using Profiles (**Adobe Experience Manager (link) > XML Documentation > Profiles**) or the Configuration Manager.

Timeout

The DITA-OT timeout value defines how long XML Documentation solution waits for a DITA-OT child process to complete before considering it in a deadlock. If you are publishing large documents, you may need to increase this value. You can tell if XML Documentation solution is timing out while waiting for DITA-OT by looking at the output generation log file. For more information on accessing the log files, see the *View and check the log file* section in the XML Add-on User Guide. This property can be configured in the Profile.

Memory requirements

The amount of memory required by DITA-OT depends on a lot of factors like the size of content, nested references, number of files, and more. If you observe publishing failures with the output generation logs containing an `OutOfMemoryException` in DITA-OT stack trace, you will need to increase the amount of memory allocated to DITA-OT. For more details on how to change memory allocation from DITA-OT command-line prompt, see Increasing Java memory allocation. This property can be configured in the Profile.

Generation Pool Size

By default, the **Generation Pool Size** is set to the number of processors available to the Java Virtual Machine +1. This setting is available in the *com.adobe.fmdita.publish.manager.PublishThreadManagerImpl* component in the Configuration Manager.

NOTE: The **Generation Pool Size** configuration only controls how many concurrent external DITA-OT processes are launched and not the actual number of threads taken up by those DITA-OT processes. From XML Documentation solution's perspective, include both DITA-OT and FrameMaker Publishing Server tasks. For example, if thread pool configuration is set to 3, then at any given time a maximum of 3 concurrent DITA-OT processes will be active. Each of these DITA-OT processes can launch any number of further threads.

Publishing

This topic covers the best practices for managing publishing tasks and workflows.

Offload publishing to a different server

See the *Workflow offloading* section.

Permissions

To give a users permissions to use all XML Documentation solution's features and generate output in a specific format, you can add those users to the `Publishers` group created by XML Documentation solution. This ensures that the users have full access to all XML Documentation solution publishing functionalities.

In addition to the above permissions, read/write access to specific DAM folders can be configured for a user as per your requirements from the **useradmin** interface.

Output history

- Every output generation information is tracked as an entry under the following repository node: `/content/fmdita/metadata/outputHistory`
- If you observe that the output history tab in the DITA map console is empty even after you have generated multiple outputs, it could mean that your output history node is corrupted. You can clean the output history for a particular DITA map file by deleting its output history node.

  Note: Deleting the output history node for a DITA map deletes its complete output history.
- Output history nodes are named on the UUIDs of their corresponding DITA map files. You can find the output history node for a particular DITA map by performing the following steps:
  a) Using CRXDE, navigate to the DITA map file node.
  b) Find the `jcr:uuid` property on the DITA map node and copy its value.
  c) Navigate to `/content/fmdita/metadata/outputHistory/<value_you_copied_in_previous_step>`.
  d) The selected node is the output history node for the concerned DITA map.
- The output history nodes also store the output generation logs. Currently there is no automated way to clear old output history. However, user can write custom scripts to delete old output history nodes by checking `jcr:created` property of logs node.

PDF generation

When creating PDF output and storing them in DAM, you should take care that asset update workflow triggers a sub-asset creation workflow for the generated PDFs. These sub-assets take up a lot of disk space and the sub-asset creation workflow also takes up a lot of CPU. In most of the cases, you won't have a need of these sub-assets and should preferably disable these.

Using custom DITA-OT

XML Documentation solution provides an option to use custom DITA-OT package in place of the default package shipped with XML Documentation solution. Consider the following points when using a custom DITA-OT package:

- XML Documentation solution supports DITA 1.2 and 1.3 document types. Specialized DITA is also supported, however, it is recommended to use the base DITA specifications. Use specialized DITA only if your content needs are not met with the base DITA document types.
- The default DITA-OT package shipped with XML Documentation solution contains a plug-in to generate EPUB output. This plug-in is not available in the DITA-OT package available for download from the official DITA-OT website. If you are using a custom DITA-OT package, ensure that it contains the plug-in to generate EPUB output, else the EPUB output generation will fail.
- The PDF output preset uses the `pdfx` transformation that is supported by the default DITA-OT package. When using custom DITA-OT package, you must specify the transformation that should be used to generate the PDF output. Also, this transformation has to be supported by your custom DITA-OT plug-in.

Export custom DITA profile as package

You can export your custom DITA profiles as a package and then import it to the other instances of XML Documentation solution. This saves time when you have multiple instances of XML Documentation solution and want to have same DITA profile on all of them.

Perform the following steps to package and deploy your custom DITA profile on other XML Documentation solution instances:

1) Customize your DITA profile and save it.
2) Do the following to create a package:
    a) Open CRX package manager and click **Create package**.
    b) Enter package name and click **OK**.
    c) Click **Edit** to edit the package.
    d) On the **Edit Package** dialog, select **Filters** > **Add filter**.
    e) Select a profile from the root path `/content/fmdita/profiles/` and click **Done**.
    NOTE: You can check the name of the profile from the profile properties.
    f) Click **Save** to save your changes.

g) Click **Build** to create the package.

3) After your package is created, click **Download** to download the package to your local machine.
4) Import the downloaded package to the other instances of XLM Documentation solution using the package manager.
5) Once imported, open `com.adobe.fmdita.config.ConfigManager` and save it without any modifications at `http://<server-url>:<port>/system/console/configMgr` to enable the installed DITA profile.

For more information on how to work with a package, see Packaging Adobe Experience Manager 6 applications in AEM documentation.

Using post-generation workflows

For any custom post-generation workflow, you must put in the **Finalize Post Generation** step as the last step. Otherwise, the output generation will not be properly tracked in output history.

Publishing unstructured FrameMaker documents

- The custom `.sts` file for HTML publishing should be of the same FrameMaker version as that of the FrameMaker Publishing Server.
- The `joboptions.xml` file located at `/libs/fmdita/config/` should not be empty. This file specifies the distiller job options to be used for PDF creation of unstructured files.

Baseline

- Do not remove a baseline that is used in any output preset. Trying to generate output using a deleted baseline would result in a failure.
- If different versions of an asset are referenced in different topics and if that referenced file is picked automatically, the latest version of the referenced file is used while publishing. Make sure that same version for such asset is referenced everywhere to avoid undesirable results in published outputs.

AEM Site publishing

Page naming

Till XML Documentation solution version 2.0, all generated site page names were in lowercase. Starting from XML Documentation solution version 2.1, site page names have same case as the corresponding DITA filename. So anyone upgrading from 2.0 to a later version might need to modify external links to their site pages to take care of case changes.

Existing Output Pages configuration

For the **Existing Output Pages** configuration, you can choose either **Overwrite Content** or **Delete and Create** option. You must consider the following points when selecting the **Delete and Create** option:

- It is slower than the **Overwrite Content** option.
- It may deactivate pages from the published instances.
- It increases repository bloat, as revision snapshot are created.
- It should only be used sparingly when you specifically want a clean publish. For example, when the design template of your site has changed.

Activation to publish instance

When AEM Site output is generated with **Delete and Create** option selected for the **Existing Output Pages** setting, then the previously generated output is first deleted and then pages are re-created. If replication agent is configured on author instance to automatically replicate all changes to AEM site pages to publish instance, then it is possible that by the time pages get re-created, page deletion event is propagated on the publish instance by the replication agent. In that case, the AEM site pages will be marked **deactivated** on the publish instance.

AEM Site page creation logic

If you are re-publishing some content after changing the **Design Path** setting to a different template, it is recommended to either publish to a different destination or select the **Delete and Create** setting to avoid inconsistent output.

Topic chunking

When generating AEM Site output, you can choose how your site output splits topics. For example, you can choose to split a multi-topic document into individual files or a single file. The splitting of topics in the output is controlled by the chunking attribute. Correct usage of chunking attribute ensures that you get the desired output.

To ensure that your site output is chunked correctly, ensure that you set the ***@chunk*** attribute at **<topicref>** or **<mapref>** element only. In addition, it should be associated with the ***@copy-to*** attribute. The value of the ***@copy-to*** attribute is used to create output pages with that name and any regeneration uses the same name to overwrite the content instead of creating a copy of a page with new random IDs.

Based on the element where you specify the chunk attribute, ensure that:

- If ***@chunk***="`to-content`" is set at **<mapref>**, then it must also have the ***@copy-to*** attribute.
- If ***@chunk***="`to-content`" is set at **<topicref>** that does not have the ***@href*** to a topic, then it should also have the ***@copy-to*** attribute.

If you are generating output for a map that contains sub-map, and the ***@chunk*** attribute of the sub-map is set to "`to-content`" without the ***@copy-to*** attribute, then the site output is generated with random chunk IDs. In this case, if you regenerate the site output, you will get duplicate pages with new random chunk IDs. Hence it is recommended to always use the ***@copy-to*** attribute as per scenarios mentioned above.

Bulk activation of the site pages

For bulk activation of the site pages (or guides), the tree activation of AEM approach is not recommended. The tree activation results in considerably slowing down of the publish and author instances. It is recommended to use the packaging API provided by XML Documentation solution for bulk activation of the site pages. For more details about the package creation APIs, see the XML Documentation for Adobe Experience Manager API Reference guide.

AEM Site templates

You can customize the site template to control the structure of the created pages and also the existence of the default search and landing pages. See the *Customize AEM Site output design template* section in the XML Documentation for Adobe Experience Manager Installation and Configuration Guide.

Concurrent publishing of the same DITA map

This should be done only at different destination paths. If there are two or more publishing workflows writing to a common destination path, some or all of them might fail and the consistency of the published content is not guaranteed.

Guidelines on creating and overwriting output

When you publish your output in AEM Site format, then you have an option to manage existing pages in the destination. When you select **Overwrite Content** option in the **Existing Output Pages** setting, then for any existing page in the destination, XML Documentation solution recreates its `contentnode` and`headnode`. Then the new content is written in the newly created nodes. This option does not create any new revision of page.

The **Delete and Create** option in the **Existing Output Pages** setting deletes any existing pages in the destination path and then publishes the new content.

Sub-node publishing

XML Documentation solution supports non-destructive publishing, so that you can combine the published DITA content with content created through other means in a page. For this to work, it is important that your DITA map structure mirrors your site structure and the topic filenames match the page names (case-sensitive since version 2.1). Once you have your site structure and the corresponding DITA map hierarchy ready, you can configure the AEM Site template for this purpose:

- Use the `topicContentNode` and `topicHeadNode` properties of your AEM Site template node to specify the node paths where you want DITA content to be published.
- During publishing, XML Documentation solution cleans these nodes of any existing content and publishes new content in them without touching any other nodes in the page structure.
- You must select the **Overwrite Content** option in the **Existing Output Pages** setting in AEM Site preset so that XML Documentation solution does not delete and recreate the whole page.
- For more details about the Site template node and its properties, See the *Customize AEM Site output design template* section in the XML Documentation for Adobe Experience Manager Installation and Configuration Guide.

Element mapping

You may use the predefined DITA element mappings, or you can map DITA elements to your custom AEM components. If you plan to use custom AEM components, then consider the following recommendations:

- If you plan to override the default element mapping, it is recommended that you do not make the changes in the default `elementmapping.xml` file. You should create a new XML mapping file and place the file at another location, preferably inside the custom `apps` folder.
- If you are planning to override some (and not all) of the element mappings, you do not have to replicate the entire `elementmapping.xml` file. You need to create a new XML mapping file and define only the elements that you want to override.
- Use the **`<class>`** element with appropriate class value to ensure better compatibility.
- If a mapping requires an XPath, make sure it is enclosed in a `CDATA` tag to avoid parsing errors.
- Use the class-based XPath expressions instead of name based.

Move site pages

Site pages generated by XML Documentation solution follows the same rules for move/copy operations as the normal AEM Site pages. A move/copy operation could fail in case there is a mismatch between the source and target pages' templates. For detailed description of how a move/copy operation is evaluated for validity, see the *Template Availability* section in AEM documentation.

Other recommendations

This topic contains information about the AEM nodes used by XML Documentation solution, best practices for translating content, and other tasks and workflows that you can optimize.

AEM nodes overlaid by XML Documentation solution

For customizing the user interface in AEM, XML Documentation solution uses the following `/apps` nodes:

For AEM 6.5

- `/apps/cq/core/content/nav/tools/xmladdonconfigurations`
- `/apps/cq/core/content/nav/tools/xmladdonconfigurations`
- `/apps/cq/core/content`
- `/apps/cq/inbox/extensions/itemtyperegistry/task/review-task`
- `/apps/cq/inbox/extensions/itemtyperegistry/task/approval-task`
- `/apps/cq/inbox/extensions/itemtyperegistry/task/review-task-notification`
- `/apps/cq/inbox/extensions/itemtyperegistry/task/approval-task-notification`
- `/apps/dam/gui/content/assetdetails/jcr:content/content/items/assetdetail/items/viewer`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/filter`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/editTopic`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/openinfm`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/selectmap`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/keyresolution`
- `/apps/dam/gui/content/assetdetails/jcr:content/actions/lock`
- `/apps/dam/gui/content/assetdetails/jcr:content/rails/versionHistory`
- `/apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaTopic`
- `/apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaMap`
- `/apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaTemplate`
- `/apps/dam/gui/content/assets/jcr:content/actions/selection/edit`
- `/apps/dam/gui/content/assets/jcr:content/actions/selection/review`
- `/apps/dam/gui/content/assets/jcr:content/actions/selection/approval`

- /apps/dam/gui/content/assets/jcr:content/actions/selection/openinfm
- /apps/dam/gui/content/assets/jcr:content/actions/selection/editTopics
- /apps/dam/gui/content/assets/jcr:content/actions/selection/delete
- /apps/dam/gui/content/assets/jcr:content/actions/selection/lock
- /apps/dam/gui/content/assets/jcr:content/actions/selection/unlock
- /apps/dam/gui/content/assets/jcr:content/actions/selection/releaselock
- /apps/dam/gui/content/assets/jcr:content/actions/primary/pasteasset2
- /apps/dam/gui/content/assets/jcr:content/header/items/deletedita
- /apps/dam/gui/content/assets/jcr:content/rails/versionHistory
- /apps/dam/gui/content/assets/assetpage
- /apps/dam/gui/content/assets/moveassetwizard/jcr:content/head/clientlibs
- /apps/dam/gui/content/assets/moveassetwizard/jcr:content/body/items/form/items/wizard/items/ditaStep
- /apps/dam/gui/content/commons/availablecolumns
- /apps/dam/gui/coral/components/admin/contentrenderer/row/asset/asset.jsp
- /apps/dam/gui/coral/components/admin/customsearch
- /apps/settings/dam/search/facets
- /apps/dam/gui/content/nav/navigationlinks

For AEM 6.4 and 6.3

- /apps/cq/core/content
- /apps/cq/core/content/nav/tools/xmladdonconfigurations
- /apps/cq/inbox/extensions/itemtyperegistry/task/review-task
- /apps/cq/inbox/extensions/itemtyperegistry/task/approval-task
- /apps/cq/inbox/extensions/itemtyperegistry/task/review-task-notification
- /apps/cq/inbox/extensions/itemtyperegistry/task/approval-task-notification
- /apps/dam/gui/content/assetdetails/jcr:content/content/items/assetdetail/items/viewer
- /apps/dam/gui/content/assetdetails/jcr:content/actions/filter
- /apps/dam/gui/content/assetdetails/jcr:content/actions/editTopic
- /apps/dam/gui/content/assetdetails/jcr:content/actions/openinfm
- /apps/dam/gui/content/assetdetails/jcr:content/actions/selectmap
- /apps/dam/gui/content/assetdetails/jcr:content/actions/keyresolution
- /apps/dam/gui/content/assetdetails/jcr:content/actions/lock
- /apps/dam/gui/content/assetdetails/jcr:content/rails/versionHistory
- /apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaTopic

- /apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaMap
- /apps/dam/gui/content/assets/jcr:content/actions/secondary/create/items/ditaTemplate
- /apps/dam/gui/content/assets/jcr:content/actions/selection/edit
- /apps/dam/gui/content/assets/jcr:content/actions/selection/review
- /apps/dam/gui/content/assets/jcr:content/actions/selection/approval
- /apps/dam/gui/content/assets/jcr:content/actions/selection/openinfm
- /apps/dam/gui/content/assets/jcr:content/actions/selection/editTopics
- /apps/dam/gui/content/assets/jcr:content/actions/selection/delete
- /apps/dam/gui/content/assets/jcr:content/actions/selection/lock
- /apps/dam/gui/content/assets/jcr:content/actions/selection/unlock
- /apps/dam/gui/content/assets/jcr:content/actions/selection/releaselock
- /apps/dam/gui/content/assets/jcr:content/actions/primary/pasteasset2
- /apps/dam/gui/content/assets/jcr:content/header/items/deletedita
- /apps/dam/gui/content/assets/jcr:content/rails/versionHistory
- /apps/dam/gui/content/assets/assetpage
- /apps/dam/gui/content/assets/moveassetwizard/jcr:content/head/clientlibs
- /apps/dam/gui/content/assets/moveassetwizard/jcr:content/body/items/form/items/wizard/items/ditaStep
- /apps/dam/gui/coral/components/admin/contentrenderer/row/asset/asset.jsp
- /apps/dam/gui/content/nav/navigationlinks

Translation

- Component-based translation can be used even when the translation vendor does not support XML/DITA translation. However, asset metadata translation needs to be supported by the translation vendor. The Component-based translation is recommended only for machine-based translation process. For human translation, disable the Component-based translation option in the ConfigMgr.
- When component-based translation is enabled, references to assets in the source DITA file are automatically changed to point to the same asset under the destination language folder.
- Any common content that does not require translation should be stored outside of any language folder. This ensures that references from language copies to the shared content is relative. This helps in maintaining language copies and ensures that no patching of references is required when creating or updating language copies.
- The translation tab in the DITA map console of a DITA map shows all the assets reachable from this DITA map including any non-DITA assets like images or videos.

- When translating non-DITA assets, it is necessary to make sure that the asset properties that need to be translated are present in `/etc/workflow/models/translation/translation_rules.xml`.
- When translating assets using the translation tab in the DITA map console, it is imperative to accept or reject the translated content using the Accept or Reject buttons in the translation project.
- As accepting and rejecting translated files is a system-intensive process, you should accept or reject only a few 100 files at a time.

Search rules for indexing DITA file

- As the search serialization configuration uses XPath expressions for selecting elements/attributes for indexing, ensure that your XPath queries are not too wide.
- It is possible to create a single, very wide XPath query to serialize all elements in a DITA document. However, this approach is extremely inefficient and bloats the repository.
- Recommended approach is to first list down how many DITA element/attributes you want to index, then write specific XPath expressions targeting those elements/attributes only.
- While creating XPath expressions, use class-based matching to ensure that they work even if you have used DITA specialization.

Custom logger

- Preferably create a custom logger for `fmdita` specific logs (`com.adobe.fmdita`) so they can be analyzed easily.
- For information on how to create a custom logger, see Logging in AEM documentation.

Unicode support

- To enable support for Unicode characters, open the Configuration Manager and search for *Apache Sling Request Parameter Handling* configuration and set the value of *Default Parameter Encoding* to `UTF-8`.

FrameMaker to DITA

DITA map creation for FrameMaker to DITA works only for FrameMaker Publishing Server with FrameMaker (2017 release) Update 1 and later builds.

Map editor

- Keys redefined in a DITA map are shown in red in the Map Editor. Clicking on the key scrolls the page to the previously defined key.
- In a normal map, DITA maps dropped from the References rail are added as a *mapref* while topics are added as *topicref*.

Custom templates

- Custom topic templates can be added in `/content/dam/topic-templates`. Custom DITA map templates can be added in `/apps/fmdita/ditamapeditor/core/templates`. Ensure that you associate your custom template with the global or folder-level profile, otherwise it won't be available for authoring.
- It is recommended to set the Title and Description properties of a template that will help in easily identifying the template in the DITA map or topic creation page. These properties can be set using the Properties page in the Assets UI.
- The order of the templates in the Create Map or Create Topic page is determined by the `ranking` property set on the template. Templates are shown in ascending order of the `ranking` property.
- It is recommended that all custom templates have the property `jcr:mimeType = "application/xml"` set on `<customtemplate>/jcr:content/renditions/original/jcr:content`. This ensures that the new topics and maps are indexed properly.

Review

- If you have shared a document for review and you make some updates in the same document, the changes will not be seen by the reviewers until you Save a Revision of your document
- If you move a topic which is under review, then the links within the topic will break. You must Save a Revision of your topic after moving it, so that the links do not break.

DTD

- After specifying a Custom DTD in the DITA Profiles, it is imperative to clear the browser cache to make sure that the old DTDs are not used.
- It is recommended that the `catalog.xml` of the Custom DTD contain the Public IDs of all files in the DTD so as to avoid a DTD path resolution error. If this is not possible, it is recommended that the topic at the very least specifies the correct System ID. If the System ID is relative to the local file path from where the file is uploaded, it is recommended to select the **Add System ID Catalog** option at the time of creating or updating the Profile.

Overridden AEM features

- In AEM 6.2, the default inbox and notification features are customized to support the notifications sent during the review workflow. If you make any further changes in the inbox and notification features, you might lose the existing customizations.
- The move, copy & paste, and delete operations for AEM Assets have been overridden. The customized content for the move feature is placed at `/apps/dam/gui/content/assets/moveassetwizard`, for copy & paste at `/apps/dam/gui/content/assets/jcr:content/actions/primary/pasteasset2`, and for delete feature at `/apps/dam/gui/content/assets/jcr:content/header/items/deletedita` location. If you make any further changes in the move, copy & paste, and delete features, you might lose the existing customizations.

FrameMaker integration

- Users need to have read permission on root ("`/`") node of AEM repository to successfully connect to AEM repository from FrameMaker.
- The file check-in behavior in FrameMaker (2017 release) Update 1 is different from the file check-in experience in the XML web editor. In FrameMaker, when a checked out file is checked back in, the changes are not saved in the latest persisted version, but they are saved in the Latest version that does not have a version number associated with it. This behavior will be changes in the upcoming update for FrameMaker.

Content Reuse Report

To ensure proper working of the Content Reuse Report, you must enable the post-processing workflow.