DITA Open Toolkit
Release 4.3

Contents

Part 1 DITA Open Toolkit 4.3

DITA Open Toolkit, or *DITA-OT* for short, is a set of Java-based, open-source tools that provide processing for content authored in the *Darwin Information Typing Architecture*.

Note: While the DITA standard is owned and developed by OASIS, the DITA Open Toolkit project is governed separately. DITA-OT is an independent, open-source implementation of the DITA standard.

DITA-OT documentation

The DITA Open Toolkit documentation provides information about installing, running, configuring, and extending the toolkit.

- This first part includes Release Notes with information on the changes in the current release, and the Authoring formats and Output formats that are provided in the default installation of DITA-OT 4.3.
- Part 2 Installing DITA Open Toolkit on page 27 shows how to install the toolkit and run a build to verify the installation.
- explains the methods that can be used to publish DITA content to other formats, including the **`dita`** command, Ant, and the Java API, along with information on building output from a containerized environment such as Docker or GitHub Actions.
- explains how to adjust DITA Open Toolkit behavior via **`dita`** command arguments and options, parameter settings, and configuration properties.
- explains how to install, remove, and discover plug-ins, and create custom plug-ins to change the default transformations or add new output formats.
- contains information about resolving problems that you might encounter.
- Reference topics provide additional information about the DITA Open Toolkit Architecture, DITA specification support, and other DITA and DITA-OT resources.

Chapter 1 DITA Open Toolkit 4.3 Release Notes

DITA Open Toolkit 4.3.3 is a maintenance release that fixes issues reported in DITA-OT 4.3, which includes new **`init`** and **`validate`** subcommands that can be used to set up projects from a template and check files for errors before publishing. You can now publish multiple formats on the command line at once, add raw DITA to Markdown files, and publish bookmaps with PDF themes.

DITA-OT releases follow semantic versioning guidelines. Version numbers use the ***`major.minor.patch`*** syntax, where ***`major`*** versions may include incompatible API changes, ***`minor`*** versions add functionality in a backwards-compatible manner and ***`patch`*** versions are maintenance releases that include backwards-compatible bug fixes.

Tip: Download the `dita-ot-4.3.3.zip` package from the project website at dita-ot.org/download.

Requirements: Java 17

DITA-OT 4.3 is designed to run on Java version 17 or later and built and tested with the Open Java Development Kit (OpenJDK). Compatible Java distributions are available from multiple sources:

- You can download Oracle distributions from oracle.com/java under commercial license.
- Eclipse Temurin is the free OpenJDK distribution available from adoptium.net.
- Free OpenJDK distributions are also provided by Amazon Corretto, Azul Zulu, and Red Hat.
- Java versions are also available via package managers such as Chocolatey, Homebrew, or SDKMAN!

Note: The Java virtual machine is generally backwards compatible, so class files built with earlier versions should still run correctly with Java 17 and DITA-OT 4.3. If your DITA-OT installation contains plug-ins with custom Java code, you may need to recompile these with Java 17—but in most cases, this step should not be necessary.

DITA-OT 4.3.3

DITA Open Toolkit 4.3.3 is a maintenance release that includes the following bug fixes.

- Earlier versions of DITA-OT did not always cascade map-level attributes such as `@format` or `@scope` correctly. Processing has been updated to ensure that map metadata cascades as mandated by §2.2.4.4 of the DITA 1.3 specification: Cascading of metadata attributes in a DITA map. #4645

For additional information on the issues resolved since the previous release, see the 4.3.3 milestone and changelog on GitHub.

DITA-OT 4.3.2 released June 5, 2025

DITA Open Toolkit 4.3.2 is a maintenance release that includes the following bug fixes.

- The GitHub release workflow was updated in DITA-OT 4.3.1 to build Docker images for both `linux/amd64` and `linux/arm64` architectures, but this caused errors in GitHub Actions as reported in dita-ot-action#11. The release workflow has been updated to use the official Docker actions, which support multi-platform builds and push the resulting platform-specific images to Docker Hub. #4609
- Table processing has been updated to improve handling for broken tables. If the `@morerows` attribute value is not an integer, strict processing mode will now report an exception. If a row has fewer columns than the preceding row, processing will continue instead of failing with a Java exception. #4620, #4628
- Earlier versions of the map-first pre-processing routines in `preprocess2` did not apply stylesheets and parameters that were passed to the `mapref` processing phase by custom plug-ins. The `mapref` module configuration has been updated to ensure that `preprocess2` respects the same parameters as the original pre-processing routine. #4624, #4625
- Several dependencies have been upgraded to include the latest utility versions and fix security issues in bundled libraries. #4632
  - Apache Commons IO 2.19.0
  - ICU4J 77.1
  - Guava 33.4.8-jre
  - Jackson data binding library 2.19.0
  - Jing 20241231
  - Logback Classic Module 1.5.18
  - Saxon 12.7
  - SLF4J 2.0.17
  - XML Resolver 5.3.3
- The bundled Apache™ FOP version has been updated to 2.11, which includes PDFBox 3 and the latest versions of the Apache™ Batik SVG Toolkit and Apache™ XML Graphics Commons libraries. #4623, #4634

  (For details on recent changes, see the Apache FOP 2.11 Release Notes.)

For additional information on the issues resolved since the previous release, see the 4.3.2 milestone and changelog on GitHub.

DITA-OT 4.3.1 released March 23, 2025

DITA Open Toolkit 4.3.1 is a maintenance release that includes the following bug fixes.

- DITA-OT 4.2 and later versions produced broken links in the navigation ToC when the `@copy-to` attribute was defined on topic references, or the **`force-unique`** option was used. Generated temporary file names were used instead of the `@copy-to` attribute value. The `@copy-to` attribute value is now respected to ensure the correct links are written to the ToC. #4564, #4569

- DITA-OT 4.3 included a regression bug that generated output in the `out` subdirectory of the DITA-OT installation directory if the output location was not explicitly specified. In this case, output is now generated in the `out` subdirectory of the current directory as in DITA-OT 4.2.4 and earlier versions. #4589, #4594
- The new **`validate`** subcommand introduced in DITA-OT 4.3 can now also be run by setting the **`-f`** or **`--format`** options to `validate`. The **`dita`** command line interface has also been updated to prevent conflicts between the **`validate`** subcommand and the existing **`validate`** parameter. #4590, #4592, #4602, #4603
- DITA-OT Docker images can now be built for both 64-bit Linux AMD and ARM architectures. No changes are required to benefit from this enhancement. Docker should automatically select the image that corresponds to the current machine architecture. #4593
- Earlier versions of DITA-OT generated invalid nested paragraphs in HTML output when the `<lines>` element was used in a paragraph. Processing has been updated to ensure that lines content is treated as a block element and rendered in separate paragraphs. #4596, #4599
- Earlier versions of DITA-OT miscalculated the base directory when publications included resource-only topics that were outside of the root map directory. Resource-only topics are now ignored in this process to ensure that relative paths between resources are generated correctly. #4606
- Several dependencies have been upgraded to include the latest utility versions and fix security issues:
  - Ant 1.10.15 #4595
  - Logback 1.5.17 #4588
  - Saxon 12.5 #4595

For additional information on the issues resolved since the previous release, see the 4.3.1 milestone and changelog on GitHub.

DITA-OT 4.3 released February 15, 2025

DITA Open Toolkit Release 4.3 includes new **`init`** and **`validate`** subcommands that can be used to set up projects from a template and check files for errors before publishing. You can now publish multiple formats on the command line at once, add raw DITA to Markdown files, and publish bookmaps with PDF themes.

Preview Init subcommand

The new **`init`** subcommand initializes a project with files from a template. #4509, #4523

The initial implementation is a preview feature designed to illustrate how project templates work. You can use templates as a starting point for new publications with required metadata, media assets, or custom stylesheets, or provide examples of your organization’s preferred markup.

- For a list of available templates, run **`dita init --list`**
- To add files from a template to the current directory, run **`dita init`** ***`template`***

The folder hierarchy in the template will be copied to the current working directory by default. To write the files to a different location, add the **`--output`** option and specify the desired path.

The directory will be created if it doesn't exist. If any of the template files are already present, an error will appear.

> **Tip:** Sample project templates are provided in the new `org.dita.init` plug-in. If you have a common project structure that would be useful beyond your organization, you can contribute new templates to future DITA-OT versions, or create a custom plug-in with company-specific project templates.

Validate subcommand

A new **`validate`** subcommand can be used to check input files for errors before publishing. #4397, #4400

This command runs the pre-processing routines in strict mode and reports any errors or warnings. This is ideal for continuous integration scenarios, as it allows you to quickly check contributions for errors without building output.

A new **`depend.validate`** extension point defines an Ant target to run with the **`dita validate`** subcommand after pre-processing, so you can extend the default validation mechanisms with your own checks.

Multiple output formats from CLI

You can now publish multiple formats at once from a single **`dita`** command sequence. #4486

To produce several output formats from a single build, pass the **`--format`** option for each transformation, or use the **`-f`** shorthand. For example:

```
dita -i sample.ditamap -f html5 -f pdf
```

The result will be the same as if you had issued separate commands for each format:

```
dita --input=sample.ditamap --format=html5
dita --input=sample.ditamap --format=pdf
```

This can be used as a simple alternative to a DITA-OT project file that defines multiple deliverables.

Lightweight DITA and Markdown updates

The `org.lwdita` plug-in has been updated to version 5.9, which includes a series of bug fixes and support for additional DITA constructs in Markdown input, including:

- Language identifiers in fenced code blocks can now be processed with an optional prefix to enable syntax highlighting in environments that require different keywords. For example, if Markdown files contain code blocks with JavaScript code, they may start with ```` ```js ```` to display the code block with syntax highlighting on GitHub. The plug-in can now add a prefix like `language-` to the DITA `@outputclass` value, included in HTML5 as `class="language-js"` for HTML5-compliant syntax highlighting with libraries like Prism. #228

- Admonition types are no longer case-sensitive, so both `!!! note` and `!!! Note` will be rendered as DITA `<note>` elements, regardless of the capitalization of the `@type` keyword. #229
- If you need to include DITA content that has no equivalent markup in , you can now use raw DITA XML directly in Markdown files. You can use this approach to include things like `<xmlelement>` or `<hazardstatement>`, which would otherwise be impossible to express in Markdown. #217
- Earlier versions would crash when processing Markdown files that did not begin with a heading. These files are now converted to valid DITA with an empty `<title>` element, and an error message appears in the log to aid in debugging. #223

Bookmap support in PDF themes

The PDF theme plug-in `com.elovirta.pdf` has been updated to version 0.8 for better bookmap support. #111 You can now style the following bookmap elements in a YAML or JSON theme without building a custom PDF plug-in:

- `<part>`
- `<chapter>`
- `<appendix>`
- `<index>`

Table of contents (ToC) styles have moved to the root `style` key. ToC styling has also been extended for better bookmap support, so you can now specify styles for each level with dedicated keys such as `style-toc-part`, `style-toc-chapter`, etc.

Parts and chapters now also support their own local contents listings, which you can enable by setting the corresponding layout key, for example `chapter-layout: MINITOC`. You can then define styling for each level via keys like `style-part-toc-chapter`, or `style-chapter-toc-1`.

A new `default` theme provides basic styling such as font settings, indentation, and title numbering for a range of commonly used elements. This theme is not intended for publishing as is, but can serve as a foundation for custom themes, and reduce the number of elements you need to style yourself. To use the default theme as the baseline for your own custom theme, add `extends: default` to your theme file. #112, #114

Preview DITA 2.0 updates

In addition to the provided in DITA-OT 3.5 – 4.2, this release includes updated processing for the latest draft versions of the DITA 2.0 grammar files from OASIS (as of January 22, 2024).

- HTML5 processing now supports the `@height` and `@width` attributes on the DITA 2.0 `<video>` element to ensure that videos are scaled correctly. #4570
- HTML5 and PDF processing has been updated to support the new DITA 2.0 emphasis domain elements: `<em>` for emphasis and `<strong>` for strong emphasis. #4571

DITA documents that reference the draft grammar files can be parsed, and where features overlap with DITA 1.3, those features will work as expected.

Note: Other new or revised features proposed for DITA 2.0 are not yet supported. Additional features will be implemented in future versions of DITA-OT as the specification evolves.

Enhancements and changes

DITA Open Toolkit Release 4.3 includes the following enhancements and changes to existing features:

- To reduce page load times, HTML5 output now uses lazy loading for external images. #4001, #4005 Local and peer image resources are loaded eagerly as in previous versions, but images defined with `@scope="external"` are now output with the `@loading` attribute set to `"lazy"` by default. A new `set-image-loading` template mode allows custom plug-ins to override the default behavior if necessary.
- The Java code for the map-first pre-processing routines now includes Javadoc comments to document how the various stages are implemented. This documentation is not published separately, but is available to developers who need to extend map processing or topic processing in custom plug-ins. Many development environments extract and display the Javadoc information while viewing the source code. #4404
- The Java code has been modernized to use more standard library features and reduce dependencies on external libraries, and restructured with automatic refactoring tools to make it easier to read and maintain. #4407, #4441, #4442, #4444, #4498
- A new DOTJ088E error message makes it easier to identify XML parsing exceptions in the log. #4408 The error appears when the input is in some way invalid but can still be parsed. The message content begins with "`XML parsing error:`" and provides additional context from the parser in the `<reason>`. As with other error messages, custom plug-ins may override the message content or severity.
- DITA-OT uses the Xerces SecurityManager to protect against the so-called "billion laughs attack", an entity expansion technique that can cause XML parsers to run out of memory and overload the CPU when parsing maliciously crafted files. DITA-OT will now stop parsing and report an error when processing any files that exceed the entity limit imposed by the security manager library. #4542, #4556
- Several bundled dependencies have been upgraded to the latest versions:
  - Apache™ FOP 2.10 (including the Apache™ Batik SVG Toolkit and Apache™ XML Graphics Commons libraries) #4519, #4565

Bug fixes

DITA Open Toolkit Release 4.3 provides fixes for the following bugs:

- In earlier versions, installing a plug-in from a path that contained the at-sign character "@" failed with an InvalidArgumentException. The implementation has been updated to ensure these paths are handled correctly. #4354, #4558
- Earlier versions issued the DOTJ037W twice when running transformations with the **`validate`** parameter set to `false`. This warning has been moved from the Java code to Ant, which ensures that it only appears once for each transformation. #4377, #4396

Contributors

DITA Open Toolkit Release 4.3 includes code contributions by the following people:

1. Jarno Elovirta
2. Julien Lacour
3. Robert D Anderson
4. Roger Sheen
5. Andrei Pomacu
6. Chris Papademetrious
7. Jason Fox

For the complete list of changes since the previous release, see the changelog on GitHub.

Documentation updates

The documentation for DITA Open Toolkit Release 4.3 has been reorganized to simplify the navigation in HTML versions and reduce the number of parts in PDF output. All of the content from previous versions is still available, though arranged slightly differently. The diagram in Figure 1: DITA-OT 4.3 navigation changes on page 14 shows the previous structure on the left, and the new locations on the right.

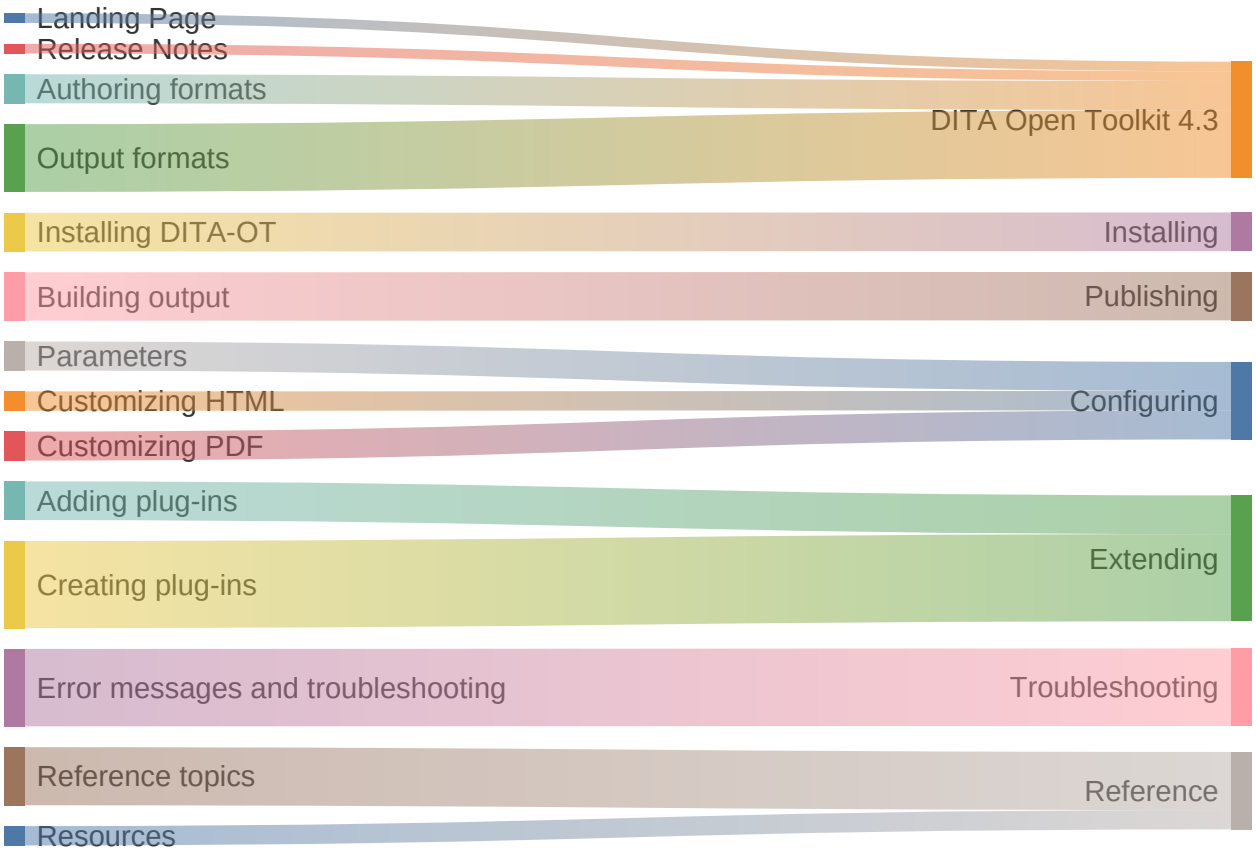

Figure 1: DITA-OT 4.3 navigation changes

The legacy Ant samples and garage sample files have been removed from the `docsrc/samples` subfolder of the installation directory. If your workflow relies on these files, you can restore them to the original location with the new **`init`** subcommand:

```
dita init samples path/to/dita-ot-dir/docsrc/samples
```

For additional information on documentation issues resolved in DITA Open Toolkit Release 4.3, see the 4.3 milestone in the documentation repository.

DITA Open Toolkit Release 4.3 includes documentation contributions by the following people:

1. Roger Sheen
2. Jarno Elovirta
3. Lief Erickson

4. Stefan Weil

For the complete list of documentation changes since the previous release, see the changelog.

Chapter 2 Authoring formats

In addition to standard DITA XML, DITA-OT supports several alternative input formats, including Markdown and the proposed *XDITA*, *MDITA* and *HDITA* authoring formats currently in development for Lightweight DITA.

Standard DITA XML

DITA Open Toolkit supports all released versions of the OASIS DITA specification, including 1.0, 1.1, 1.2, and 1.3. As of release 4.3, DITA-OT also provides an initial preview of features for the latest draft of the upcoming DITA 2.0 standard.

The DITA specification "defines a set of document types for authoring and organizing topic-oriented information, as well as a set of mechanisms for combining, extending, and constraining document types." The DITA 1.3 specification is the authoritative source of information on authoring DITA content in XML.

Tip: For details on how DITA Open Toolkit processes DITA XML content, see .

Markdown input

Markdown is a lightweight markup language that allows you to write using an easy-to-read plain text format and convert to structurally valid markup as necessary.

In the words of its creators:

> "The overriding design goal for Markdown's formatting syntax is to make it as readable as possible. The idea is that a Markdown-formatted document should be publishable as-is, as plain text, without looking like it's been marked up with tags or formatting instructions."

DITA Open Toolkit allows you to use Markdown files directly in topic references and export DITA content as Markdown.

These features enable lightweight authoring scenarios that allow subject matter experts to contribute to DITA publications without writing in XML, and support publishing workflows that include DITA content in Markdown-based publishing systems.

Adding Markdown topics

In 2015, the original *DITA-OT Markdown* plug-in introduced a series of conventions to convert Markdown content to DITA, and vice-versa. This Markdown flavor was called Markdown DITA.

The `markdown` format adds several complementary constructs to represent DITA content in Markdown, beyond those proposed for the MDITA format in the Lightweight DITA specification drafts.

Tip: For details on the differences in , see , , and .

To add a Markdown topic to a DITA publication, create a topic reference in your map and set the `@format` attribute to `markdown` so the toolkit will recognize the source file as Markdown and convert it to DITA:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE map PUBLIC "-//OASIS//DTD DITA Map//EN" "map.dtd">
<map>
  <topicref href="markdown-dita-topic.md" format="markdown"/>
</map>
```

When you add Markdown topics to a DITA publication as described above, the content is temporarily converted to DITA in the background when generating other output formats like HTML or PDF, but the Markdown source files remain unchanged.

Tip: This approach is recommended in cases where simple content is authored collaboratively over multiple versions, as Markdown topics can be edited by a wide range of authors and combined as necessary with more complex content maintained in DITA XML.

Converting Markdown to DITA

In cases where the Markdown input is a one-off contribution, members of the DITA authoring team can use the Markdown file as raw material that is easily converted to DITA and enriched with conditional processing attributes, conkeyrefs or other more complex semantics that have no equivalent in limited formats like Markdown.

If you prefer to maintain this content in DITA in the future, you can generate DITA output by passing the **`--format`**`=dita` option on the command line.

This converts all input files (both DITA XML and Markdown) to Normalized DITA. You can then copy the generated DITA files from the output folder to your project and replace references to the Markdown topics with their DITA equivalents.

Preview support for Lightweight DITA

DITA-OT provides preview support for the authoring formats proposed for Lightweight DITA, or "*LwDITA*". The *XDITA*, *MDITA* and *HDITA* formats are alternative representations of DITA content in XML, Markdown and HTML5.

Attention: Since Lightweight DITA has not yet been released as a formal specification, the implementation for *XDITA*, *MDITA* and *HDITA* authoring formats is subject to change. Future versions of DITA Open Toolkit will be updated as LwDITA evolves.

XDITA

XDITA is the LwDITA authoring format that uses XML to structure information. XDITA is a subset of DITA, with new multimedia element types added to support interoperability with HTML5. XDITA is designed for users who want to write DITA content but who do not want (or need) the full power of DITA.

The XDITA parser included in the `org.lwdita` plug-in provides preliminary support for XDITA maps and XDITA topics.

To apply XDITA-specific processing to topics in an XDITA map or a full DITA 1.3 map, set the `@format` attribute on a `<topicref>` to `xdita`:

```
<map>
  <topicref href="xdita-topic.xml" format="xdita"/>
</map>
```

Tip: For examples of cross-format content sharing between topics in *XDITA*, *HDITA*, extended-profile *MDITA*, and DITA 1.3, see the LwDITA sample files in the DITA-OT installation directory under `plugins/org.oasis-open.xdita.v0_2_2/samples`.

MDITA

MDITA is the LwDITA authoring format based on Markdown. It is designed for users who want to write structured content with the minimum of overhead, but who also want to take advantage of the reuse mechanisms associated with the DITA standard and the multi-channel publishing afforded by standard DITA tooling.

Recent proposals for LwDITA include two profiles for authoring MDITA topics:

- The "*Core profile*" is based on GitHub-Flavored Markdown and includes elements that are common to many other Markdown implementations.
- The "*Extended profile*" borrows additional features from other flavors of Markdown to represent a broader range of DITA content with existing plain-text syntax conventions.

The MDITA parser included in the `org.lwdita` plug-in provides preliminary support for these profiles and additional Markdown constructs as described in the .

To apply the stricter LwDITA-specific processing to Markdown topics, set the `@format` attribute to `mdita`:

```
<map>
  <topicref href="mdita-topic.md" format="mdita"/>
</map>
```

In this case, the first paragraph in the topic will be treated as a short description, for example, and additional metadata can be specified for the topic via a YAML front matter block.

Tip: For details on the differences in , see , , and .

HDITA

HDITA is the LwDITA authoring format based on HTML5, which is intended to support structured content authoring with tools designed for HTML authoring. HDITA also uses custom data attributes to provide interoperability with DITA.

The HDITA parser included in the `org.lwdita` plug-in provides preliminary support for these constructs.

To apply LwDITA-specific processing to HTML topics, set the `@format` attribute to `hdita`:

```
<map>
  <topicref href="hdita-topic.html" format="hdita"/>
</map>
```

Attention: The HDITA map format is not yet supported. To include HDITA content, use an XDITA map or a DITA 1.3 map.

Using conditional processing in MDITA and HDITA

When you set up conditional processing in MDITA and HDITA, use the `@data-props` attribute in the element that will have the conditional processing applied. In the `.ditaval` file, however, use the `@props` attribute.

Converting lightweight formats to DITA XML

When you add LwDITA topics to a DITA publication, the content is temporarily converted to DITA in the background when generating other output formats like HTML or PDF, but the source files remain unchanged.

If you prefer to maintain this content in DITA in the future, you can generate DITA output by passing the **`--format`**`=dita` option on the command line.

This converts all input files (both LwDITA formats and DITA XML) to Normalized DITA. You can then copy the generated DITA files from the output folder to your project and replace references to the lightweight topics with their DITA equivalents.

Chapter 3 Output formats

DITA Open Toolkit ships with several core transformations that convert DITA content to different output formats. Additional formats are available from the plug-in registry at dita-ot.org/plugins.

Tip: For information on how to install other formats, see .

PDF

The **pdf** transformation generates output in Portable Document Format.

This transformation was originally created as a plug-in and maintained outside of the main toolkit code. It was created as a more robust alternative to the demo PDF transformation in the original toolkit, and thus was known as PDF2. The plug-in was bundled into the default toolkit distribution with release 1.4.3.

To run the PDF transformation, set the **transtype** parameter to **pdf**, or pass the **--format**=**pdf** option to the **dita** command line.

```
dita --input=input-file --format=pdf
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

HTML5

The **html5** transformation generates HTML5 output and a table of contents (TOC) file.

The HTML5 output is always associated with the default DITA-OT CSS file (`commonltr.css` or `commonrtl.css` for right-to-left languages). You can use toolkit parameters to add a custom style sheet that overrides the default styles, or generate a `<nav>` element with a navigation TOC in topic pages.

To run the HTML5 transformation, set the **transtype** parameter to **html5**, or pass the **--format**=**html5** option to the **dita** command line.

```
dita --input=input-file --format=html5
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

Eclipse help

The **eclipsehelp** transformation generates XHTML output, CSS files, and the control files that are needed for Eclipse help.

In addition to the XHTML output and CSS files, this transformation returns the following files, where ***mapname*** is the name of the root map.

File name	Description
plugin.xml	Control file for the Eclipse plug-in
***mapname*.xml**	Table of contents
index.xml	Index file
plugin.properties	
META-INF/MANIFEST.MF	

To run the Eclipse help transformation, set the **transtype** parameter to **eclipsehelp**, or pass the **--format**=**eclipsehelp** option to the **dita** command line.

```
dita --input=input-file --format=eclipsehelp
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

HTML Help

The **htmlhelp** transformation generates HTML output, CSS files, and the control files that are needed to produce a Microsoft Compiled HTML Help (.chm) file.

In addition to the HTML output and CSS files, this transformation returns the following files, where ***mapname*** is the name of the root map.

File name	Description
***mapname*.hhc**	Table of contents
***mapname*.hhk**	Sorted index
***mapname*.hhp**	HTML Help project file

File name	Description
mapname.chm	Compiled HTML Help file **Note:** The compiled file is only generated if the HTML Help Workshop is installed on the build system.

To run the HTML Help transformation, set the **transtype** parameter to **htmlhelp**, or pass the **--format**=**htmlhelp** option to the **dita** command line.

```
dita --input=input-file --format=htmlhelp
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

Generating Markdown output

Along with Markdown input on page 17, DITA-OT provides three transformation types to convert DITA content to Markdown, including the original syntax, GitHub-Flavored Markdown, and GitBook.

The new output formats can be used to feed DITA content into Markdown-based publishing systems or other workflows that lack the ability to process DITA XML.

Markdown output can be generated by passing one of the following transformation types to the **dita** command with the **--format** option:

- To publish Markdown DITA files, use the **markdown** transtype.
- To generate GitHub-Flavored Markdown files, use the **markdown_github** transtype.

  Note: Since the GitHub format does not support definition lists, they are converted to unordered lists with bold terms. Attribute blocks with IDs, class names, and other custom attributes are also omitted, as GitHub does not support Pandoc header attributes or PHP Markdown Extra special attributes.
- To publish GitHub-Flavored Markdown and generate a SUMMARY.md table of contents file for publication via GitBook or mdBook, use the **markdown_gitbook** transtype.

Run the **dita** command and set the value of the output **--format** option to the desired format, for example:

```
dita --input=input-file --format=markdown
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

Attention: The *MDITA* format is not yet supported when generating output. To publish DITA content to Markdown, use one of the formats listed above.

Normalized DITA

The `dita` transformation generates normalized topics and maps from DITA input. The normalized output includes the results of DITA Open Toolkit pre-processing operations, which resolve map references, keys, content references, code references and push metadata back and forth between maps and topics.

In comparison to the source DITA files, the normalized DITA files are modified in the following ways:

- References from one DITA map to another are resolved
- Map-based links, such as those generated by map hierarchy and relationship tables, are added to the topics.
- Link text is resolved.
- Map attributes that cascade are made explicit on child elements.
- Map metadata such as index entries and copyrights are pushed into topics.
- Topic metadata such as navigation titles, link text and short descriptions are pulled from topics into the map.
- XML comments are removed.

Applications

Normalized output may be useful in situations where post-processing of DITA content is required, but the downstream systems are limited in their ability to resolve DITA references.

Tip: You can also use the normalized DITA transformation to convert Markdown or Lightweight DITA formats to DITA XML. You can then copy the generated DITA files from the output folder to your project and replace references to the lightweight topics with their XML equivalents.

Generating normalized DITA output

Run the **`dita`** command and set the value of the output **`--format`** option to **`dita`**:

```
dita --input=input-file --format=dita
```

where:

- ***`input-file`*** is the DITA map or DITA file that you want to process.

XHTML

The **xhtml** transformation generates XHTML output and a table of contents (TOC) file. This was the first transformation created for DITA Open Toolkit, and originally served as the basis for all HTML-based transformations.

The XHTML output is always associated with the default DITA-OT CSS file (`commonltr.css` or `commonrtl.css` for right-to-left languages). You can use toolkit parameters to add a custom style sheet to override the default styles.

To run the XHTML transformation, set the **transtype** parameter to **xhtml**, or pass the **--format**=**xhtml** option to the **dita** command line.

```
dita --input=input-file --format=xhtml
```

where:

- ***input-file*** is the DITA map or DITA file that you want to process.

Part 2 Installing DITA Open Toolkit

The DITA-OT distribution package can be installed on Linux, macOS, and Windows. It contains everything that you need to run the toolkit except for Java.

Before you begin

- Ensure that you have a Java Runtime Environment (JRE) or Java Development Kit (JDK).

  DITA-OT 4.3 is designed to run on Java version 17 or later and built and tested with the Open Java Development Kit (OpenJDK). Compatible Java distributions are available from multiple sources:

  - You can download Oracle distributions from oracle.com/java under commercial license.
  - Eclipse Temurin is the free OpenJDK distribution available from adoptium.net.
  - Free OpenJDK distributions are also provided by Amazon Corretto, Azul Zulu, and Red Hat.
  - Java versions are also available via package managers such as Chocolatey, Homebrew, or SDKMAN!
- If you want to generate HTML Help, ensure that you have HTML Help Workshop installed.

  You can download the Help Workshop from web.archive.org.

Procedure

1. Download the `dita-ot-4.3.3.zip` package from the project website at dita-ot.org/download.
2. Extract the contents of the package to the directory where you want to install DITA-OT.

   Note: The documentation refers to this location as the *DITA-OT installation directory*, or ***`dita-ot-dir`***.
3. Add the absolute path for the `bin` folder of the DITA-OT installation directory to the PATH environment variable.

   Tip: This defines the necessary environment variable that allows the **`dita`** command to be run from any location on the file system without typing the path to the command.

Chapter 4 Prerequisite software

The software that DITA-OT requires depends on the output formats you want to use.

Software required for core DITA-OT processing

DITA-OT requires the following software applications:

Java Development Kit (JDK) or Java Runtime Environment (JRE)

DITA-OT 4.3 is designed to run on Java version 17 or later and built and tested with the Open Java Development Kit (OpenJDK). Compatible Java distributions are available from multiple sources:

- You can download Oracle distributions from oracle.com/java under commercial license.
- Eclipse Temurin is the free OpenJDK distribution available from adoptium.net.
- Free OpenJDK distributions are also provided by Amazon Corretto, Azul Zulu, and Red Hat.
- Java versions are also available via package managers such as Chocolatey, Homebrew, or SDKMAN!

Note: This is the *only* prerequisite that you need to install. All other required software is provided in the distribution package, including Apache Ant™ 1.10.15, Saxon 12.7, and ICU for Java 77.1.

Software required for specific transformations

Depending on the type of output that you want to generate, you might need the following applications:

HTML Help Workshop

Microsoft no longer provides the software required for generating Compiled HTML Help (.chm) files. You can download an archived copy of the HTML Help Workshop from the Internet Archive's Wayback Machine at web.archive.org.

XSL-FO processor

Required for generating PDF output. Apache™ FOP *(Formatting Objects Processor)* 2.11

is included in the distribution package. You can download other versions from xmlgraphics.apache.org/fop. You can also use commercial FO processors such as Antenna House Formatter or RenderX XEP.

Chapter 5 Checking the DITA-OT version number

You can determine the DITA Open Toolkit version number from a command prompt.

Procedure

1. Open a command prompt or terminal session.
2. Issue the following command:

```
dita --version
```

Results

The DITA-OT version number appears on the console:

```
DITA-OT version 4.3.3
```

Chapter 6 First build with the `dita` command

You can publish output using the **dita** command-line tool. Build parameters can be specified on the command line, with `.properties` files, or in project files that define multiple deliverables.

About this task

The DITA-OT client is a command-line tool with no graphical user interface. To verify that your installation works correctly, you can build the HTML version of the documentation you are reading now.

Procedure

1. Open a terminal window by typing the following in the search bar:

Option	Description
Linux or macOS	Type `Terminal`.
Windows	Type `Command Prompt`.

2. Change directories to the `docsrc/samples` subfolder of the DITA-OT installation directory:

```
cd dita-ot-dir/docsrc/samples
```

3. At the command-line prompt, enter the following command:

```
dita --project=project-files/html.xml
```

   The HTML version of the documentation is generated in the `docsrc/samples/out` folder.

What to do next

Most builds require you to specify more options than are described in this topic. For more information, see Publishing with the **dita** command.

Index

Special Characters

A

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

W

X