DitaCraft User Guide

The Complete Guide to DITA Editing in VS Code

DitaCraft User Guide

The Complete Guide to DITA Editing in VS Code

DitaCraft
A Beginner-Friendly VS Code Extension
for DITA Authoring and Publishing

DitaCraft User Guide

The Complete Guide to DITA Editing in VS Code

Version 0.4.2

January 2025

Jeremy Jeanne

DitaCraft Project

Contents

Preface

Preface

Welcome to the DitaCraft User Guide.

About This Guide

This guide provides comprehensive documentation for DitaCraft, the Visual Studio Code extension for DITA authoring and publishing. Whether you are new to DITA or an experienced technical writer, this guide will help you get the most out of DitaCraft.

Intended Audience

This guide is intended for:

- **Technical Writers** who create and maintain DITA documentation
- **Documentation Managers** who oversee DITA projects
- **Developers** who write technical documentation alongside code
- **Content Strategists** evaluating DITA tooling options

Prerequisites

To use DitaCraft effectively, you should have:

- Visual Studio Code 1.80 or higher installed
- Basic familiarity with XML concepts
- Understanding of DITA fundamentals (helpful but not required)

How to Use This Guide

This guide is organized into the following sections:

Part I: Getting Started	Introduction to DitaCraft and installation steps
Part II: Using DitaCraft	Commands reference and detailed feature explanations
Part III: Configuration	Settings and customization options
Appendix: Keyboard Shortcuts	Quick reference for all keyboard shortcuts
Glossary	Definitions of key terms used throughout the guide

Conventions Used

This guide uses the following conventions:

Convention	Meaning
Bold text	UI elements, buttons, menu items
`Monospace`	Code, commands, file names, settings
`File paths`	Directory and file paths
`User input`	Text you should type

Version Information

This guide documents DitaCraft version 0.4.2. For the latest updates and release notes, visit the project repository.

Notice

Notices

Legal notices and trademark information.

Copyright

Trademarks

Disclaimer

Part I

Getting Started

Topics:

Chapter 1

Introduction to DitaCraft

DitaCraft is a comprehensive Visual Studio Code extension for editing and publishing DITA content.

What is DitaCraft?

DitaCraft is the easiest way to edit and publish your DITA (Darwin Information Typing Architecture) files directly from Visual Studio Code. It provides a complete authoring environment with:

- Full DTD validation using TypesXML
- Smart navigation with Ctrl+Click support
- Key space resolution for keyref and conkeyref
- Live HTML5 preview with scroll sync
- One-click publishing via DITA-OT
- Interactive map visualizer

Key Features

DitaCraft offers the following key features:

Smart Navigation	Ctrl+Click on `href`, `conref`, `keyref`, and `conkeyref` attributes to navigate directly to referenced content.
Full DTD Validation	Real-time validation against DITA 1.3 DTDs using TypesXML with 100% W3C conformance.
Live Preview	Side-by-side HTML5 preview with bidirectional scroll sync, theme support, and print mode.
DITA-OT Integration	Publish to HTML5, PDF, EPUB, and more with direct DITA Open Toolkit integration.
Map Visualizer	Interactive tree view showing your DITA map hierarchy with navigation support.

Supported File Types

DitaCraft supports the following DITA file types:

Extension	Description
`.dita`	DITA topics (concept, task, reference, topic)
`.ditamap`	DITA maps for organizing topics
`.bookmap`	Book-oriented maps with chapters and parts

Version Information

This guide covers DitaCraft version 0.4.2, which includes:

- Modular validation engine architecture
- Rate limiting for DoS protection
- 547+ comprehensive tests
- Enhanced security features

Chapter

2

Getting Started

Learn how to install and configure DitaCraft for your first DITA project.

Before installing DitaCraft, ensure you have:

- Visual Studio Code 1.80 or higher
- DITA-OT 4.2.1 or higher (for publishing features)

This guide walks you through installing DitaCraft and creating your first DITA file.

1. Install DitaCraft from VS Code Marketplace
    a) Open VS Code
    b) Press **Ctrl+P** (or **Cmd+P** on Mac)
    c) Type `ext install ditacraft` and press Enter
2. Configure DITA-OT path (optional, required for publishing)
    a) Download DITA-OT from dita-ot.org
    b) Extract to a location (e.g., `C:\DITA-OT-4.2.1`)
    c) Open Command Palette (**Ctrl+Shift+P**)
    d) Type `DITA: Configure DITA-OT Path`
    e) Select your DITA-OT installation directory
3. Create your first DITA topic
    a) Open Command Palette (**Ctrl+Shift+P**)
    b) Type `DITA: Create New Topic`
    c) Select topic type (concept, task, or reference)
    d) Enter a file name for your topic
    A new DITA topic file is created with proper DOCTYPE and structure.
4. Validate your DITA file
    a) Open your DITA file
    b) Press **Ctrl+Shift+V** to validate
    Validation results appear in the Problems panel. Errors and warnings are highlighted in the editor.
5. Preview your content (optional)
    a) Press **Ctrl+Shift+H** to open HTML5 preview
    A side-by-side preview panel opens showing your rendered DITA content.

You now have DitaCraft installed and configured. You can:

- Create DITA topics, maps, and bookmaps
- Validate your content in real-time
- Navigate between files using Ctrl+Click
- Preview and publish your documentation

Part

II

Using DitaCraft

Topics:

Chapter 3

Commands Overview

DitaCraft provides commands for validation, publishing, file creation, and navigation.

Accessing Commands

All DitaCraft commands are accessible via the Command Palette:

1. Press **Ctrl+Shift+P** (or **Cmd+Shift+P** on Mac)
2. Type `DITA:` to see all available commands
3. Select the desired command

Command Categories

DitaCraft commands are organized into the following categories:

Validation Commands	Validate DITA files for XML syntax and DTD conformance
Publishing Commands	Publish DITA content to HTML5, PDF, and other formats
File Creation Commands	Create new DITA topics, maps, and bookmaps
Navigation Commands	Navigate and visualize DITA content structure

Quick Command Reference

Command	Shortcut	Description
DITA: Validate Current File	Ctrl+Shift+V	Validate the active DITA file
DITA: Publish (Select Format)	Ctrl+Shift+B	Publish with format selection
DITA: Publish to HTML5	-	Quick publish to HTML5
DITA: Preview HTML5	Ctrl+Shift+H	Show live HTML5 preview
DITA: Show Map Visualizer	-	Display interactive map hierarchy
DITA: Create New Topic	-	Create a new DITA topic

Command	Shortcut	Description
DITA: Create New Map	-	Create a new DITA map
DITA: Create New Bookmap	-	Create a new bookmap
DITA: Configure DITA-OT Path	-	Set DITA-OT installation path

Chapter

4

Validation Commands

Commands for validating DITA files against XML syntax and DTD specifications.

DITA: Validate Current File

Shortcut: Ctrl+Shift+V (Windows/Linux) or **Cmd+Shift+V** (Mac)

Validates the currently active DITA file for:

- XML well-formedness (proper tags, attributes, encoding)
- DTD conformance (valid elements, required attributes)
- DITA structure (required elements like title, proper nesting)
- Content model validation (correct child elements)

Usage:

1. Open a `.dita`, `.ditamap`, or `.bookmap` file
2. Press **Ctrl+Shift+V** or run the command from Command Palette
3. View results in the Problems panel

Output:

- **Errors** - Critical issues that must be fixed (red squiggles)
- **Warnings** - Potential issues to review (yellow squiggles)
- **Information** - Suggestions and notes (blue squiggles)

Automatic Validation

DitaCraft automatically validates DITA files when:

- **On Open:** When you open a DITA file
- **On Save:** When you save a DITA file (if `autoValidate` is enabled)
- **On Change:** After you stop typing (with 500ms debounce)

To disable automatic validation, set `ditacraft.autoValidate` to `false` in settings.

Validation Engines

DitaCraft supports three validation engines:

Engine	Description	Requirements
`typesxml`	Full DTD validation with 100% W3C conformance (default, recommended)	None (bundled)

Engine	Description	Requirements
`built-in`	Lightweight XML and content model checking	None (bundled)
`xmllint`	External validation using libxml2	xmllint installed on system

To change the validation engine, set `ditacraft.validationEngine` in settings.

Rate Limiting

To prevent performance issues from rapid validation requests, DitaCraft includes rate limiting:

- Maximum 10 validation requests per second per file
- Excess requests are silently skipped (auto-validation) or show a warning (manual validation)
- Rate limits reset after the time window expires

Chapter 5

Publishing Commands

Commands for publishing DITA content to HTML5, PDF, and other output formats.

DITA: Publish (Select Format)

Shortcut: Ctrl+Shift+B (Windows/Linux) or **Cmd+Shift+B** (Mac)

Opens a format selection dialog and publishes the current DITA map or topic.

Available Formats:

- **html5** - Modern HTML5 output with responsive design
- **pdf** - PDF output (requires Apache FOP or similar)
- **xhtml** - XHTML 1.0 output
- **eclipsehelp** - Eclipse help system format
- **htmlhelp** - Microsoft HTML Help format
- **markdown** - Markdown output

Requirements:

- DITA-OT must be installed and configured
- Java Runtime Environment (JRE) 11 or higher

DITA: Publish to HTML5

Quickly publishes the current file to HTML5 format without format selection.

Usage:

1. Open a `.ditamap` or `.bookmap` file
2. Run the command from Command Palette
3. Output is generated in the `out` folder

Output Location:

By default, output is generated in `{workspace}/out/{format}`. You can customize this in settings with `ditacraft.outputDirectory`.

DITA: Preview HTML5

Shortcut: Ctrl+Shift+H (Windows/Linux) or **Cmd+Shift+H** (Mac)

Opens a live HTML5 preview panel alongside the editor.

Features:

- **Live Updates:** Preview refreshes automatically as you type
- **Scroll Sync:** Bidirectional scroll synchronization between editor and preview
- **Theme Support:** Light and dark theme support

- **Print Mode:** Toggle print-friendly styling
- **Conref Resolution:** Displays resolved conref content

Tip: The preview uses a lightweight renderer that does not require DITA-OT, making it instant for quick content review.

DITA: Configure DITA-OT Path

Opens a folder picker to select your DITA-OT installation directory.

Steps:

1. Download DITA-OT from dita-ot.org
2. Extract to a location on your system
3. Run this command and select the DITA-OT directory

Verification:

After configuration, DitaCraft verifies the installation by checking for:

- `bin/dita` or `bin/dita.bat` executable
- `lib/` directory with required JAR files
- Valid version information

Common Publishing Errors

If publishing fails, check the following:

DITA-OT not found	Run **DITA: Configure DITA-OT Path** to set the correct path.
Java not found	Install JRE 11+ and ensure `JAVA_HOME` is set correctly.
Build failed	Check the Output panel for detailed error messages from DITA-OT.

Chapter 6

File Creation Commands

Commands for creating new DITA topics, maps, and bookmaps.

DITA: Create New Topic

Creates a new DITA topic file with proper DOCTYPE and structure.

Topic Types:

Concept	For conceptual information that helps users understand a subject. Uses `<concept>` root element with `<conbody>`.
Task	For procedural information with step-by-step instructions. Uses `<task>` root element with `<taskbody>` and `<steps>`.
Reference	For reference information like API documentation or command syntax. Uses `<reference>` root element with `<refbody>`.
Topic	Generic topic type for content that doesn't fit other categories. Uses `<topic>` root element with `<body>`.

Usage:

1. Run the command from Command Palette
2. Select the topic type
3. Enter a file name (without extension)
4. The file is created in the current folder with `.dita` extension

DITA: Create New Map

Creates a new DITA map file for organizing topics.

Map Structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE map PUBLIC "-//OASIS//DTD DITA Map//EN"
 "map.dtd">
<map id="my-map">
    <title>My Map Title</title>
    <topicref href="topic1.dita"/>
    <topicref href="topic2.dita"/>
```

```
</map>
```

Usage:

1. Run the command from Command Palette
2. Enter a file name (without extension)
3. The file is created with `.ditamap` extension

After creation, add `<topicref>` elements to reference your topics.

DITA: Create New Bookmap

Creates a new bookmap for book-oriented publications.

Bookmap Structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE bookmap PUBLIC "-//OASIS//DTD DITA
 BookMap//EN" "bookmap.dtd">
<bookmap id="my-bookmap">
    <booktitle>
        <mainbooktitle>My Book Title</
mainbooktitle>
    </booktitle>
    <frontmatter>
        <booklists><toc/></booklists>
    </frontmatter>
    <chapter href="chapter1.dita"/>
    <backmatter>
        <booklists><indexlist/></booklists>
    </backmatter>
</bookmap>
```

Bookmap Elements:

- `<booktitle>` - Book title information
- `<bookmeta>` - Metadata (author, dates, rights)
- `<frontmatter>` - TOC, preface, dedication
- `<chapter>` - Main content chapters
- `<part>` - Groups of chapters
- `<appendix>` - Supplementary content
- `<backmatter>` - Index, glossary

File Naming Best Practices

- Use lowercase letters and hyphens (e.g., `getting-started.dita`)
- Avoid spaces and special characters
- Use descriptive names that reflect content
- Keep names concise but meaningful

Chapter 7

Navigation Commands

Commands for navigating and visualizing DITA content structure.

DITA: Show Map Visualizer

Opens an interactive tree view showing your DITA map hierarchy.

Features:

- **Hierarchical View:** Displays maps, chapters, and topics in a tree structure
- **Click Navigation:** Click any node to open the corresponding file
- **Real-time Updates:** Tree updates when files are added or removed
- **Status Indicators:** Shows validation status for each file

Usage:

1. Open a `.ditamap` or `.bookmap` file
2. Run the command from Command Palette
3. The visualizer opens in a side panel

Ctrl+Click Navigation

DitaCraft provides smart navigation using Ctrl+Click (Cmd+Click on Mac) on various attributes.

Supported Attributes:

Attribute	Navigation Behavior
`href`	Opens the referenced file
`conref`	Navigates to the content reference target (file#id)
`keyref`	Resolves the key and navigates to its target
`conkeyref`	Resolves the key and navigates to the content reference

Example:

```
<topicref href="topics/intro.dita"/>  <!-- Ctrl
+Click opens intro.dita -->
<p conref="shared.dita#shared/note1"/>  <!-- Ctrl
+Click opens shared.dita at #note1 -->
<ph keyref="product-name"/>  <!-- Ctrl+Click
 resolves key and navigates -->
```

Go to Definition

Press **F12** or right-click and select **Go to Definition** to navigate to referenced content.

This works on the same attributes as Ctrl+Click navigation.

Peek Definition

Press **Alt+F12** to peek at the definition without leaving your current file.

A small inline editor shows the referenced content, allowing you to:

- View the target content
- Make quick edits
- Navigate to the full file

Find All References

Press **Shift+F12** to find all places where the current element or ID is referenced.

Useful for:

- Finding all conrefs to a specific element
- Locating topic references in maps
- Understanding content reuse patterns

Chapter 8

Features Overview

DitaCraft provides a comprehensive set of features for DITA authoring in Visual Studio Code.

Core Features

DitaCraft transforms VS Code into a full-featured DITA authoring environment with the following capabilities:

Smart Navigation	Navigate between files using Ctrl +Click on href, conref, keyref, and conkeyref attributes. Jump directly to referenced content without manual file searching.
Full DTD Validation	Real-time validation against DITA 1.3 DTDs using TypesXML with 100% W3C conformance. Catch errors as you type with detailed diagnostic messages.
Live Preview	Side-by-side HTML5 preview with bidirectional scroll sync. See your formatted output without running a full build.
DITA-OT Integration	One-click publishing to HTML5, PDF, and other formats using DITA Open Toolkit integration.
Map Visualizer	Interactive tree view showing your DITA map hierarchy with click-to-navigate functionality.
Key Resolution	Full key space support for keyref and conkeyref resolution with intelligent key definition lookup.

Editor Enhancements

DitaCraft enhances the VS Code editing experience for DITA files:

- **Syntax Highlighting:** DITA-aware XML syntax highlighting
- **Auto-completion:** Element and attribute suggestions based on DTD
- **Bracket Matching:** Automatic tag matching and highlighting
- **Folding:** Collapse sections, steps, and other block elements
- **Outline View:** Navigate document structure in the Outline panel

Validation Features

DitaCraft offers multiple validation options:

- **Real-time Validation:** Errors appear as you type with debounced updates
- **Multiple Engines:** Choose from TypesXML (default), built-in, or xmllint
- **Detailed Diagnostics:** Line-precise error messages with fix suggestions
- **Rate Limiting:** Protection against excessive validation requests

Productivity Features

- **Quick File Creation:** Create topics, maps, and bookmaps from Command Palette
- **Content Reuse:** Full support for conref and keyref mechanisms
- **Reference Navigation:** Find all references to an element or topic
- **Workspace Support:** Works with single files or multi-root workspaces

Chapter

9

Smart Navigation

Navigate between DITA files using Ctrl+Click on reference attributes.

Overview

Smart Navigation allows you to quickly move between related DITA files by clicking on reference attributes. Hold **Ctrl** (or **Cmd** on Mac) and click on any supported attribute value to navigate to the target.

href Navigation

The `href` attribute is used in maps to reference topics and in topics to link to other content.

Examples:

```
<!-- In a ditamap -->
<topicref href="topics/getting-started.dita"/>

<!-- In a topic -->
<xref href="reference-guide.dita#config-settings"/
>
```

Ctrl+Click on the href value opens the referenced file. If the href includes a fragment identifier (#id), the cursor moves to that element.

conref Navigation

Content references (`conref`) pull content from one location to another. Navigation takes you to the source element.

Example:

```
<p conref="shared/warnings.dita#warnings/safety-
note"/>
```

Ctrl+Click navigates to `shared/warnings.dita` and positions the cursor at the element with `id="safety-note"`.

keyref Navigation

Key references use indirect addressing through keys defined in a map.

Map Definition:

```
<keydef keys="product-name">
    <topicmeta>
        <keywords><keyword>DitaCraft</keyword></
keywords>
```

```
    </topicmeta>
</keydef>

<keydef keys="install-guide" href="topics/
installation.dita"/>
```

Usage:

```
<ph keyref="product-name"/>  <!-- Resolves to
 "DitaCraft" -->
<xref keyref="install-guide"/>  <!-- Links to
 installation.dita -->
```

Ctrl+Click on a keyref:

1. Resolves the key using the current key space
2. Navigates to the key's target (href or definition)

conkeyref Navigation

Content key references combine key resolution with content referencing.

Example:

```
<!-- Key definition -->
<keydef keys="shared-content" href="shared/
common.dita"/>

<!-- Usage -->
<note conkeyref="shared-content/warning-note"/>
```

Ctrl+Click resolves `shared-content` to `shared/common.dita`, then navigates to `#warning-note`.

Navigation Tips

- **Hover Preview:** Hover over a reference to see a preview without navigating
- **Go Back:** Use **Alt+Left Arrow** to return to the previous location
- **Peek Definition:** Use **Alt+F12** to peek without leaving your file
- **Open to Side: Ctrl+Alt+Click** opens the target in a split editor

Chapter
10
Validation

DitaCraft validates DITA files against XML syntax and DTD specifications in real-time.

Validation Overview

DitaCraft provides comprehensive validation to ensure your DITA content is well-formed and conforms to DITA specifications. Validation runs automatically as you edit and can also be triggered manually.

What Gets Validated

XML Well-formedness	Checks for proper XML syntax: matching tags, valid attribute values, proper encoding, and correct entity references.
DTD Conformance	Validates elements and attributes against DITA 1.3 DTDs. Ensures elements appear in allowed contexts with required attributes.
Content Model	Verifies child elements appear in the correct order and quantity according to the DTD content model.
ID Uniqueness	Checks that ID attributes are unique within a topic.

Validation Engines

DitaCraft supports three validation engines, each with different capabilities:

Engine	Pros	Cons
`typesxml`	Full DTD support, 100% W3C conformance, no external dependencies	Slightly slower for very large files
`built-in`	Very fast, lightweight, basic content model checking	Limited DTD support, may miss some errors
`xmllint`	Industry-standard libxml2, excellent performance	Requires external installation

Recommendation: Use `typesxml` (the default) for the best balance of accuracy and convenience.

Diagnostic Levels

Validation issues are reported at three severity levels:

- **Error (Red):** Critical issues that must be fixed. The document is not valid DITA.
- **Warning (Yellow):** Potential issues that should be reviewed. The document may be valid but has problems.
- **Information (Blue):** Suggestions and notes. Not errors, but helpful for improvement.

Viewing Validation Results

Validation results appear in multiple places:

- **Editor Squiggles:** Colored underlines appear on problematic content
- **Problems Panel:** Full list of all diagnostics (**Ctrl+Shift+M**)
- **Status Bar:** Shows error/warning count for the current file
- **File Explorer:** File icons show validation status

Automatic Validation

By default, DitaCraft validates automatically:

- **On Open:** When you open a DITA file
- **On Save:** When you save a file (configurable)
- **On Change:** After you stop typing (500ms debounce)

To disable automatic validation, set `ditacraft.autoValidate` to `false` in settings.

Rate Limiting

To prevent performance issues, DitaCraft limits validation requests:

- Maximum 10 requests per second per file
- Excess requests are silently skipped during auto-validation
- Manual validation shows a warning if rate limited

This protects against excessive CPU usage when making rapid edits.

Chapter
11

Live Preview

View your DITA content as formatted HTML5 in a side-by-side preview panel.

Overview

The Live Preview feature renders your DITA content as HTML5 in real-time, allowing you to see how your documentation will look without running a full DITA-OT build.

Shortcut: **Ctrl+Shift+H** (Windows/Linux) or **Cmd+Shift+H** (Mac)

Preview Features

Live Updates	The preview refreshes automatically as you type. Changes appear within milliseconds, providing instant feedback on your edits.
Bidirectional Scroll Sync	Scrolling in the editor scrolls the preview to the corresponding position, and vice versa. Keep your place when reviewing long documents.
Theme Support	The preview adapts to VS Code's current theme. Light themes show light preview; dark themes show dark preview.
Print Mode	Toggle print mode to see how your content will appear when printed. Useful for reviewing PDF output styling.

Content Rendering

The preview renders the following DITA elements with appropriate styling:

- **Structure:** Sections, paragraphs, titles, and nested content
- **Lists:** Ordered, unordered, and definition lists
- **Tables:** Simple tables and complex tables with spanning
- **Code:** Code blocks and inline code with syntax highlighting
- **Notes:** Note, tip, warning, caution, and danger admonitions
- **Steps:** Task steps with numbering and substeps
- **Images:** Referenced images with alt text

Conref Resolution

The preview resolves content references (conref) and displays the referenced content inline. This lets you see your complete document with all reused content in place.

Example:

```
<p conref="shared/common.dita#common/legal-
notice"/>
```

In the preview, this shows the actual content from the referenced element, not just a placeholder.

Keyref Resolution

Key references are resolved using the current key space. If you have key definitions in your map, the preview shows the resolved values.

Preview Controls

The preview panel includes a toolbar with these controls:

- **Refresh:** Manually refresh the preview
- **Toggle Scroll Sync:** Enable/disable scroll synchronization
- **Toggle Print Mode:** Switch between screen and print styling
- **Open in Browser:** Open the preview in your default browser

Limitations

The live preview is designed for quick content review and has some limitations compared to full DITA-OT output:

- Styling may differ from your DITA-OT configuration
- Some advanced features like relationship tables are not rendered
- Custom plugins and extensions are not applied

For final output review, use **DITA: Publish to HTML5** to generate full DITA-OT output.

Chapter

12

Map Visualizer

View your DITA map structure as an interactive tree diagram.

Overview

The Map Visualizer provides a graphical representation of your DITA map hierarchy. It displays the relationship between maps, chapters, and topics in an easy-to-navigate tree view.

Command: DITA: Show Map Visualizer

Opening the Visualizer

1. Open a `.ditamap` or `.bookmap` file
2. Open Command Palette (**Ctrl+Shift+P**)
3. Type `DITA: Show Map Visualizer`
4. The visualizer opens in a side panel

Tree Structure

The visualizer displays your map hierarchy with different node types:

Map Node	The root of the tree, representing the ditamap or bookmap file.
Chapter/Part Nodes	Major divisions in bookmaps, shown as expandable folders.
Topic References	Individual topic files referenced by the map.
Nested Maps	Submaps referenced via `<mapref>` appear as expandable subtrees.

Navigating the Tree

- **Click:** Select a node and open the corresponding file in the editor
- **Double-click:** Open the file and close the visualizer panel
- **Expand/Collapse:** Click the arrow icons to show or hide child nodes
- **Right-click:** Access context menu with additional options

Visual Indicators

The visualizer uses icons and colors to convey information:

- **File Icons:** Different icons for maps, topics, and other file types
- **Validation Status:** Red indicators show files with errors
- **Missing Files:** Broken references are highlighted

- **Current File:** The currently edited file is highlighted

Real-time Updates

The visualizer updates automatically when:

- You add or remove topicref elements from the map
- You create or delete topic files
- Validation status changes for any file
- You switch between maps

Use Cases

- **Content Audit:** Review your document structure at a glance
- **Navigation:** Quickly jump to any topic in a large documentation set
- **Validation Review:** Identify which files have errors
- **Structure Planning:** Visualize how content is organized

Chapter 13

Key Resolution

Use keys for indirect addressing and content reuse in your DITA projects.

Overview

DITA keys provide a powerful mechanism for indirect addressing. Instead of using direct file paths, you define keys in your map and reference them throughout your topics. This enables:

- Easy updates when file locations change
- Conditional content through key scopes
- Centralized management of reusable content
- Simplified localization workflows

Defining Keys

Keys are defined in DITA maps using `<keydef>` elements or the `keys` attribute on `<topicref>`.

Basic Key Definition:

```
<keydef keys="product-name">
    <topicmeta>
        <keywords>
            <keyword>DitaCraft</keyword>
        </keywords>
    </topicmeta>
</keydef>
```

Key with href:

```
<keydef keys="install-guide" href="topics/
installation.dita"/>
```

Key on topicref:

```
<topicref keys="getting-started" href="topics/
getting-started.dita"/>
```

Using keyref

Reference keys using the `keyref` attribute to pull in text content or create links.

Text Replacement:

```
<!-- In your topic -->
<p>Welcome to <ph keyref="product-name"/>!</p>
```

```
<!-- Renders as -->
<p>Welcome to DitaCraft!</p>
```

Link Creation:

```
<xref keyref="install-guide">Installation Guide</
xref>
```

Using conkeyref

The `conkeyref` attribute combines key resolution with content referencing. It first resolves the key to a file, then pulls content from a specific element.

```
<!-- Key definition -->
<keydef keys="shared-warnings" href="shared/
warnings.dita"/>

<!-- Usage -->
<note conkeyref="shared-warnings/safety-warning"/>
```

This resolves `shared-warnings` to `shared/warnings.dita`, then pulls the element with `id="safety-warning"`.

Key Space

DitaCraft builds a key space from your root map. The key space includes:

- All key definitions in the root map
- Keys from referenced submaps
- Keys inherited through key scopes

Key Precedence: When the same key is defined multiple times, the first definition wins. Keys defined earlier in the map take precedence.

Key Scopes

Key scopes allow you to define variations of the same content for different contexts.

```
<topicgroup keyscope="admin">
    <keydef keys="audience">

 <topicmeta><keywords><keyword>administrators</
keyword></keywords></topicmeta>
    </keydef>
    <topicref href="admin-guide.dita"/>
</topicgroup>

<topicgroup keyscope="user">
    <keydef keys="audience">
        <topicmeta><keywords><keyword>end users</
keyword></keywords></topicmeta>
    </keydef>
    <topicref href="user-guide.dita"/>
</topicgroup>
```

DitaCraft Key Support

DitaCraft provides full support for DITA keys:

- **Key Resolution:** Automatically builds key space from root maps
- **Navigation:** Ctrl+Click on keyref navigates to the key's target
- **Preview:** Live preview shows resolved key values
- **Validation:** Warns about undefined or missing keys
- **Caching:** Key space is cached for performance with automatic invalidation

Part

III

Configuration

Topics:

Chapter 14

Settings Overview

Configure DitaCraft to match your workflow and preferences.

Accessing Settings

DitaCraft settings can be accessed in several ways:

1. Open VS Code Settings (**Ctrl+,** or **Cmd+,**)
2. Search for `ditacraft`
3. Or edit `settings.json` directly

Settings Levels

VS Code supports multiple settings levels:

User Settings	Apply to all VS Code instances. Stored in your user profile.
Workspace Settings	Apply only to the current workspace. Stored in `.vscode/settings.json`.
Folder Settings	Apply to a specific folder in a multi-root workspace.

Workspace settings override user settings, allowing project-specific configuration.

Settings Categories

DitaCraft settings are organized into these categories:

Category	**Description**
General	DITA-OT path and basic configuration
Validation	Validation engine and auto-validation settings
Publishing	Output directory and publishing options
Preview	Live preview behavior and styling

Quick Settings Reference

Setting	Default	Description
`ditacraft.ditaOtPath`	`""`	Path to DITA-OT installation
`ditacraft.validationEngine`	`"typesxml"`	Validation engine to use
`ditacraft.autoValidate`	`true`	Enable automatic validation
`ditacraft.outputDirectory`	`"out"`	Output directory for publishing
`ditacraft.previewScrollSync`	`true`	Enable preview scroll sync

Example settings.json

```
{
    "ditacraft.ditaOtPath": "C:/DITA-OT-4.2.1",
    "ditacraft.validationEngine": "typesxml",
    "ditacraft.autoValidate": true,
    "ditacraft.outputDirectory": "out",
    "ditacraft.previewScrollSync": true
}
```

Chapter 15

General Settings

Configure DITA-OT path and basic DitaCraft settings.

ditacraft.ditaOtPath

Type: String

Default: `""` (empty)

The path to your DITA Open Toolkit installation directory. This setting is required for publishing features.

Example Values:

- Windows: `"C:/DITA-OT-4.2.1"`
- macOS: `"/Users/username/dita-ot-4.2.1"`
- Linux: `"/opt/dita-ot-4.2.1"`

Tip: Use forward slashes (/) in paths, even on Windows. VS Code handles path conversion automatically.

How to Set:

1. Run **DITA: Configure DITA-OT Path** from Command Palette, or
2. Set the value manually in settings.json

Verification:

DitaCraft verifies the path by checking for:

- `bin/dita` (Unix) or `bin/dita.bat` (Windows)
- `lib/` directory with required JAR files

Java Configuration

DITA-OT requires Java 11 or higher. DitaCraft uses the following to locate Java:

1. `JAVA_HOME` environment variable
2. Java in system PATH

If Java is not found, publishing will fail with a "Java not found" error.

Setting JAVA_HOME:

```
# Windows (PowerShell)
$env:JAVA_HOME = "C:\Program Files\Java\jdk-17"

# macOS/Linux
export JAVA_HOME=/usr/lib/jvm/java-17-openjdk
```

Workspace Detection

DitaCraft automatically detects workspace context:

- **Single Folder:** Uses the folder as the workspace root
- **Multi-root Workspace:** Uses the folder containing the current file
- **No Workspace:** Uses the directory of the current file

This affects relative path resolution for href, conref, and key definitions.

Chapter 16

Validation Settings

Configure validation engine and automatic validation behavior.

ditacraft.validationEngine

Type: String (enum)

Default: `"typesxml"`

Options:

`"typesxml"`	Full DTD validation using TypesXML parser. Provides 100% W3C conformance with complete DTD support. Recommended for most users.
`"built-in"`	Lightweight validation with basic content model checking. Faster but may miss some errors. Useful for very large files.
`"xmllint"`	External validation using libxml2's xmllint tool. Requires xmllint to be installed on your system. Provides excellent performance and accuracy.

Choosing an Engine:

Use Case	Recommended Engine
General DITA authoring	`typesxml`
Very large files (1000+ elements)	`built-in` or `xmllint`
Strict DTD conformance required	`typesxml` or `xmllint`
Quick syntax checking only	`built-in`

ditacraft.autoValidate

Type: Boolean

Default: `true`

When enabled, DitaCraft validates DITA files automatically:

- When a file is opened
- When a file is saved
- After typing stops (with 500ms debounce)

Set to `false` to disable automatic validation. You can still validate manually using **Ctrl+Shift+V**.

When to Disable:

- Working with very large files that cause performance issues
- Editing files that are intentionally incomplete
- When you prefer manual validation control

Rate Limiting

DitaCraft includes built-in rate limiting to prevent performance issues:

- **Limit:** 10 validation requests per second per file
- **Behavior:** Excess requests are silently skipped during auto-validation
- **Manual Validation:** Shows a warning if rate limited

Rate limiting cannot be configured but ensures stable performance during rapid editing.

Setting Up xmllint

To use the `xmllint` engine, install libxml2:

Windows:

1. Download from zlatkovic.com
2. Add the bin directory to your PATH

macOS:

```
brew install libxml2
```

Linux (Debian/Ubuntu):

```
sudo apt-get install libxml2-utils
```

Verification:

```
xmllint --version
```

Chapter
17

Publishing Settings

Configure output directory and DITA-OT publishing options.

ditacraft.outputDirectory

Type: String

Default: `"out"`

The directory where published output is generated. Can be an absolute path or relative to the workspace root.

Examples:

- `"out"` - Creates `{workspace}/out/`
- `"build/output"` - Creates `{workspace}/build/output/`
- `"C:/Documentation/Output"` - Uses absolute path

Output Structure:

Within the output directory, DitaCraft creates subdirectories for each format:

```
out/
├── html5/
│   ├── index.html
│   └── topics/
├── pdf/
│   └── document.pdf
└── xhtml/
    └── ...
```

Default Publishing Format

DitaCraft supports publishing to multiple formats via DITA-OT:

Format	Description	Requirements
html5	Modern responsive HTML5	None (built-in)
pdf	PDF via Apache FOP	FOP (included in DITA-OT)
xhtml	XHTML 1.0 Transitional	None (built-in)
eclipsehelp	Eclipse Help System	None (built-in)
htmlhelp	Microsoft HTML Help	HTML Help Compiler
markdown	Markdown output	None (built-in)

DITA-OT Build Parameters

DitaCraft passes the following parameters to DITA-OT during publishing:

Parameter	**Value**
`-i`	Input file (current map or topic)
`-f`	Output format (html5, pdf, etc.)
`-o`	Output directory

For advanced DITA-OT configuration, use a `build.xml` or `plugin.xml` file in your project.

Clean Output

By default, publishing overwrites existing output in the target directory. To ensure a clean build:

1. Delete the output directory manually, or
2. Use a different output directory for each build

Viewing Published Output

After publishing completes:

- **HTML5:** Open `out/html5/index.html` in a browser
- **PDF:** Open the generated PDF file in a PDF reader
- **VS Code:** Right-click HTML files and select "Open with Live Server"

Chapter 18

Preview Settings

Configure live preview behavior and appearance.

ditacraft.previewScrollSync

Type: Boolean

Default: `true`

Enables bidirectional scroll synchronization between the editor and preview panel.

Behavior when enabled:

- Scrolling in the editor scrolls the preview to the corresponding position
- Scrolling in the preview scrolls the editor to match
- Cursor position in editor highlights the corresponding preview element

When to Disable:

- Working with very long documents where sync causes performance issues
- When you prefer independent scrolling of editor and preview
- When editing and viewing different sections simultaneously

Theme Integration

The preview automatically adapts to VS Code's current theme:

- **Light Themes:** White background, dark text
- **Dark Themes:** Dark background, light text
- **High Contrast:** Enhanced contrast for accessibility

Preview styling follows VS Code's color theme to provide a consistent experience.

Preview Refresh Behavior

The preview refreshes automatically with the following characteristics:

- **Debounce:** 300ms delay after typing stops
- **Preservation:** Scroll position is preserved during refresh
- **Efficiency:** Only changed content is re-rendered when possible

Preview Panel Controls

The preview panel toolbar provides these controls:

Refresh Button	Manually refresh the preview content.

Scroll Sync Toggle	Enable or disable scroll synchronization.
Print Mode Toggle	Switch between screen and print styling. Print mode shows how content will appear when printed.
Open in Browser	Open the preview in your default web browser for full-screen viewing.

Content Resolution

The preview resolves the following content types:

Content Type	**Resolution Behavior**
conref	Fetches and displays referenced content inline
keyref	Resolves key to display value or creates link
conkeyref	Resolves key then fetches referenced content
images	Displays images from relative or absolute paths

Note: If a reference cannot be resolved, the preview shows a placeholder indicating the missing content.

Known Limitations

The live preview has some limitations compared to full DITA-OT output:

- Custom DITA-OT plugins are not applied
- Relationship tables are not rendered
- Some advanced table features may display differently
- PDF-specific formatting is not shown

For final output review, use the publishing commands to generate full DITA-OT output.

Appendix A

Keyboard Shortcuts

Quick reference for all DitaCraft keyboard shortcuts.

Overview

DitaCraft provides keyboard shortcuts for common operations. This reference lists shortcuts for both Windows/Linux and macOS.

Validation

Windows/Linux	Mac	Command	Description
Ctrl+Shift+V	**Cmd+Shift+V**	DITA: Validate Current File	Validate the active DITA file against XML syntax and DTD

Publishing

Windows/Linux	Mac	Command	Description
Ctrl+Shift+B	**Cmd+Shift+B**	DITA: Publish (Select Format)	Publish with format selection dialog
Ctrl+Shift+H	**Cmd+Shift+H**	DITA: Preview HTML5	Open live HTML5 preview panel

Navigation

Windows/Linux	Mac	Action	Description
Ctrl+Click	**Cmd+Click**	Go to Definition	Navigate to href, conref, keyref, or conkeyref target
Ctrl+Alt+Click	**Cmd+Option +Click**	Open to Side	Open referenced file in split editor
F12	**F12**	Go to Definition	Navigate to the definition of the reference under cursor
Alt+F12	**Option+F12**	Peek Definition	Peek at definition inline without

Windows/Linux	Mac	Action	Description
			leaving current file
Shift+F12	**Shift+F12**	Find All References	Find all places where the current element is referenced
Alt+Left Arrow	**Ctrl+- (minus)**	Go Back	Return to previous location after navigation
Alt+Right Arrow	**Ctrl+Shift+- (minus)**	Go Forward	Move forward in navigation history

VS Code General Shortcuts

These VS Code shortcuts are useful when working with DITA files:

Windows/Linux	Mac	Action	Description
Ctrl+Shift+P	**Cmd+Shift+P**	Command Palette	Open Command Palette to access all DitaCraft commands
Ctrl+P	**Cmd+P**	Quick Open	Quickly open files by name
Ctrl+Shift+M	**Cmd+Shift+M**	Problems Panel	Show/hide validation errors and warnings
Ctrl+Shift+O	**Cmd+Shift+O**	Go to Symbol	Navigate to elements within the current file
Ctrl+G	**Ctrl+G**	Go to Line	Jump to a specific line number
Ctrl+/	**Cmd+/**	Toggle Comment	Comment/ uncomment XML content
Ctrl+Shift+[	**Cmd+Option+[**	Fold Region	Collapse the current XML element
Ctrl+Shift+]	**Cmd+Option+]**	Unfold Region	Expand the current XML element
Ctrl+K Ctrl+0	**Cmd+K Cmd+0**	Fold All	Collapse all foldable regions
Ctrl+K Ctrl+J	**Cmd+K Cmd+J**	Unfold All	Expand all folded regions

Windows/Linux	Mac	Action	Description
Ctrl+S	**Cmd+S**	Save	Save file (triggers auto-validation if enabled)
Ctrl+Z	**Cmd+Z**	Undo	Undo last edit
Ctrl+Shift+Z	**Cmd+Shift+Z**	Redo	Redo last undone edit

XML Editing Shortcuts

Windows/Linux	Mac	Action	Description
Ctrl+Space	**Ctrl+Space**	Trigger Suggestions	Show element and attribute auto-completion
Ctrl+Shift +Space	**Cmd+Shift +Space**	Parameter Hints	Show attribute value suggestions
Ctrl+.	**Cmd+.**	Quick Fix	Show available quick fixes for errors
F2	**F2**	Rename Symbol	Rename element ID across files
Ctrl+D	**Cmd+D**	Add Selection	Select next occurrence of current word
Ctrl+Shift+L	**Cmd+Shift+L**	Select All Occurrences	Select all occurrences of current word
Alt+Up/Down	**Option+Up/ Down**	Move Line	Move current line up or down
Shift+Alt+Up/ Down	**Shift+Option +Up/Down**	Copy Line	Duplicate current line above or below
Ctrl+Shift+K	**Cmd+Shift+K**	Delete Line	Delete the current line

Customizing Shortcuts

To customize keyboard shortcuts:

1. Press **Ctrl+K Ctrl+S** (Windows/Linux) or **Cmd+K Cmd+S** (Mac) to open Keyboard Shortcuts
2. Search for the command (e.g., "DITA: Validate")
3. Click the pencil icon to assign a new shortcut
4. Press your desired key combination

You can also edit `keybindings.json` directly for advanced customization.

Quick Cheat Sheet

Most frequently used DitaCraft shortcuts:

Action	Windows/Linux	Mac
Validate File	**Ctrl+Shift+V**	**Cmd+Shift+V**
Publish	**Ctrl+Shift+B**	**Cmd+Shift+B**
Preview HTML5	**Ctrl+Shift+H**	**Cmd+Shift+H**
Navigate to Reference	**Ctrl+Click**	**Cmd+Click**
Command Palette	**Ctrl+Shift+P**	**Cmd+Shift+P**
Problems Panel	**Ctrl+Shift+M**	**Cmd+Shift+M**