

DHSC Best Practices Guide

Contents

- DITA and User Assistance.....3**
- Developing Custom DITA-based Help Systems.....4**
 - TOCJS and TOCJSBIS Plug-ins.....4
- [INTERNAL] DHSC Best Practices Proposal.....11**

DITA and User Assistance

This topic describes the relationship between DITA and user assistance and Help systems in particular.

DITA can be used in the process of creating user assistance, and especially of Help systems, but is not currently (and may never be) used as a user assistance format itself.

DITA is a storage and authoring format, not a delivery format; it is a presentation-neutral format. The separation of content and form is fundamental to DITA's design; content is written in DITA, and must be transformed to a reading format before it can be delivered to the reader.

In principle, content written in DITA can be *transformed* to any reading format. In practice, it's not that simple. Before DITA can be transformed, a transformation process has to be devised. Many DITA authoring and publishing tools come with standard transformers for most common delivery formats, such as PDF, RTF and HTML. The *DITA Open Toolkit*, an open source collection of utilities and documentation to help writers work with DITA, includes basic transformers for PDF, RTF, HTML, DocBook, Eclipse Help, and Microsoft® HTML Help.

User assistance content is not defined by its format. A document in Microsoft HTML Help format isn't necessarily a Help system; user assistance is defined by the nature of the content. Conversely, user assistance content doesn't have to be delivered in a *traditional* Help format.

DITA promotes a single-source approach to documentation, so user assistance may commonly be one of a number of deliverables produced from a *repository* of information topics. The process of producing simple Help systems from DITA content using the standard DITA Open Toolkit transformers is straight-forward. It is a little more complicated to deliver such DITA-generated content for context-sensitive Help, but still readily achievable. Likewise, in principle, it is a trivial matter to incorporate DITA content into embedded user assistance and user interface elements, using standard XML tools and techniques. There is not yet a standard approach to user assistance, so there is also no standard way of using DITA in this way. Different organisations tend to develop their own individual, custom approaches, using inhouse technical expertise to do so.

Moving beyond simple Help systems, however, is currently difficult, but not impossible. The DITA Technical Committee is developing some changes to the DITA standard to allow these processes to be simplified. However, the apparent simplicity or complexity of using DITA for Help authoring will be in future determined by the capabilities of DITA editing and publishing tools. When it comes down to it, DITA is just a standard, and good tools are needed to work with good standards.

Developing Custom DITA-based Help Systems

TOCJS and TOCJSBIS Plug-ins

This topic provides an overview of the DITA-OT plug-ins named `tocjs` and `tocjsbis`.

The `tocjs` DITA-OT plug-in and its more recent enhancement named `tocjsbis` generate a JavaScript-based table of contents page for any DITA topics that you reference in your `.ditamap` file.

Overview

These plug-ins are very popular in the DITA community; we even use them on the DITA Technical Committee for our DITA 1.2 specifications.

The screenshot shows a help system interface. On the left is a navigation pane with a tree view. The tree is expanded to show 'DITAVAL elements', which includes 'alt-text' (highlighted), 'endflag', 'prop', 'revprop', 'startflag', 'style-conflict', and 'val'. Below this are sections for 'Commonly referenced attribute', 'Domains', and 'Appendix'. On the right is the main content area for the 'alt-text' element. It has a title 'alt-text' and a description: 'An element allowed inside either startflag or endflag to provide alternate text for an image, when the imageref attribute sets an image to be used for flagging. The default alternate text for revprop start of change is a localized translation of "Start of change". The default alternate text for revprop end of change is a localized translation of "End of change".' Below the description are sections: 'Contains' with 'text data', 'Contained by' with 'startflag' and 'endflag', and 'Example' with the text 'See the example in the <val> description.'

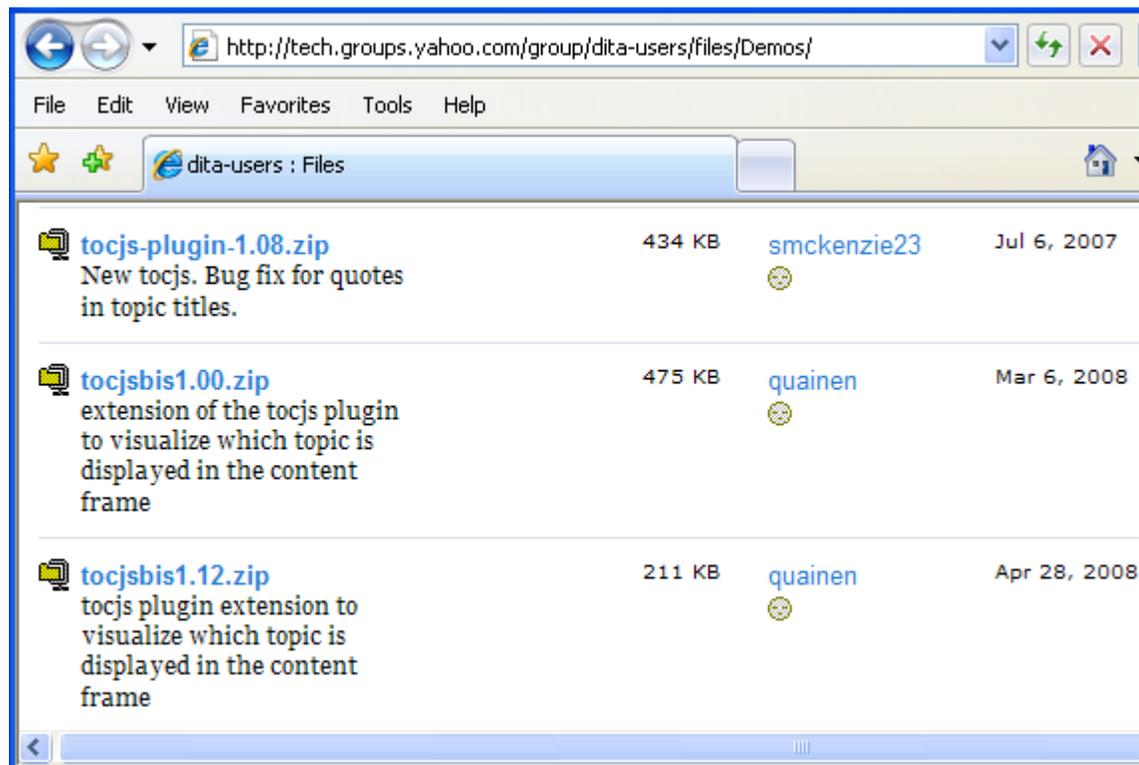
- `tocjs`: The `tocjs` plug-in was developed by Shawn McKenzie, currently working at Sophos in beautiful Vancouver, BC (Canada). The `tocjs` plug-in executes after the standard DITA-OT XHTML transform, so each `tocjs` TOC entry knows the name of its target XHTML topic.
- `tocjsbis`: The `tocjsbis` plug-in was written by Nadege Quaine and adds the important feature of topic synchronization, i.e. the highlighted topic entry in the TOC updates in sync with the topic being displayed in the contents frame. The plug-in achieves synchronization by adding a unique ID to each XHTML output topic (`<meta content="id-tocjsbis_about" name="DC.Identifier" />`) and by synchronizing the TOC entry against that topic ID. If you have generated HTML output from RoboHelp or WebWorks Publisher, this technique should be familiar.

If you are generating HTML output of any sort from your DITA sources, you should test one or both of these plug-ins. I use tocjsbis in my context-sensitive Help builds where I work. To the extent that the tree control in tocjs and tocjsbis are based on the Yahoo tree control library, you can customize the way that the final TOC tree displays and behaves in your Help system. They can be tricky, but customizations work.

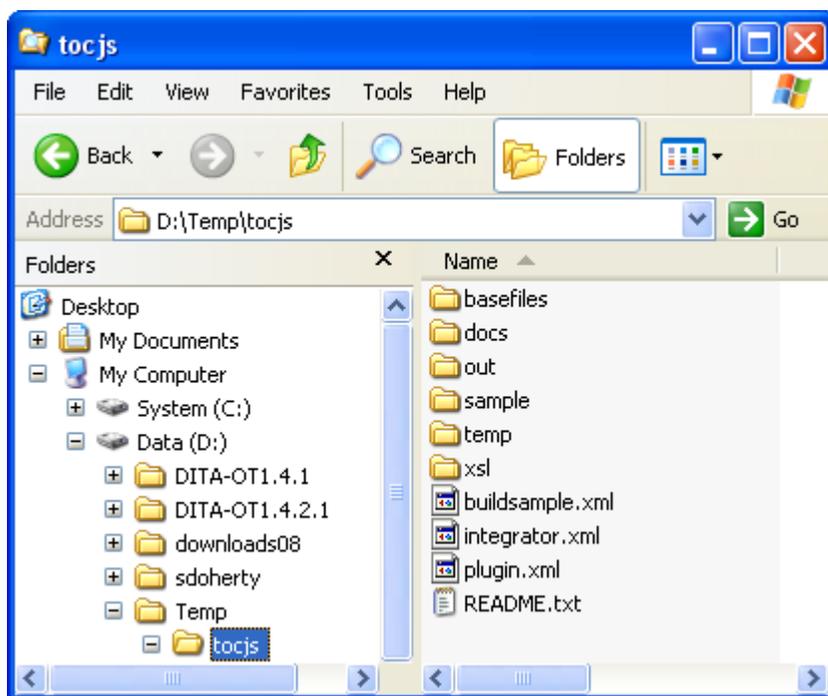
Setup and configuration

The tocjs and tocjsbis plug-ins are free downloads from the Yahoo DITA-OT site.

<http://tech.groups.yahoo.com/group/dita-users/files/Demos/>



After you have unzipped these archives to a local directory, you can browse the documentation to get a feel for what tocjs offers and how you can install it in your DITA-OT directory.

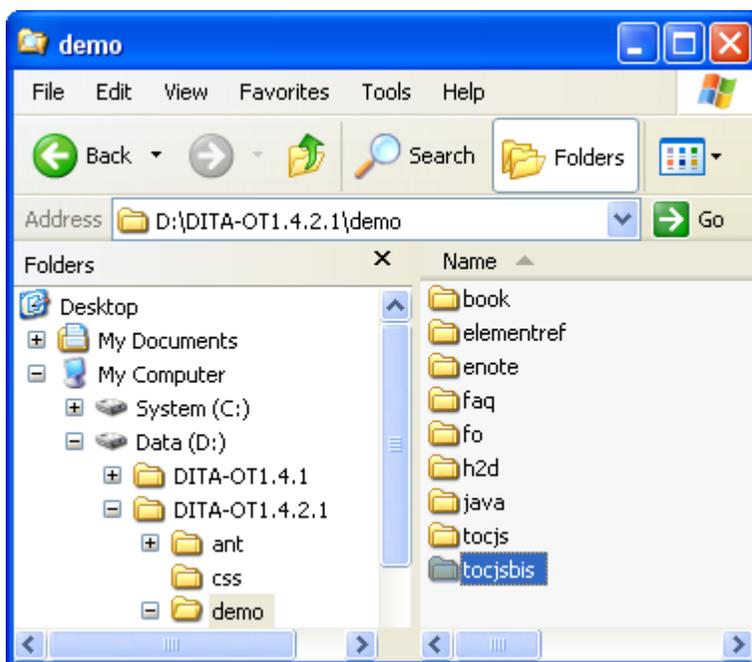


The pre-built documentation for tocjs lives in the `/docs` subdirectory.



Installing tocjs or tocjsbis is very straightforward.

1. Copy the un-achived tocjs or tocjsbis subdirectory into the demo subdirectory of your DITA Open Toolkit directory.



2. Open a shell command window from your DITA-OT directory.
3. Enter the following command to integrate the new plugin or plugins with your current DITA-OT environment.

```
ant -f integrator.xml
```

4. Change directory into the `demo/tocjs` or `demo/tocjsbis` subdirectory and enter the following command to build the sample DITA topics.

```
ant -f demo/tocjs/buildsample.xml sample2tocjs
```

5. Load the newly generated `demo\tocjs\out\sample\frameset.html` file in your browser.

If you are considering customizations to `tocjs` or `tocjsbis`, consult the documentation for the Yahoo UI tree control at the following URL.

<http://developer.yahoo.com/yui/treeview/>

Yahoo! UI Library: TreeView

The YUI TreeView Control provides a rich, compact visual presentation of hierarchical node data. Support is provided for several common node types, for the association of custom metadata with each node, and for the dynamic loading of node data (via in-page data or via XMLHttpRequest using [Connection Manager](#)) to navigate large datasets with a small initial payload.



On This Page:

- [Getting Started](#)
- [Using TreeView](#)
- [Known Issues](#)
- [YUI on Mobile Devices](#)
- [Support & Community](#)
- [Filing Bugs and Feature Requests](#)

Quick Links:

- [Examples](#): Explore examples of the TreeView Control in action.
- [API Documentation](#): View the full API documentation for the TreeView Control.
- [Release Notes](#): Detailed change log for the TreeView Control.
- [License](#): The YUI Library is issued under a BSD license.
- [Download](#): Download the TreeView Control as part of the full YUI Library on SourceForge.

That is about all that is involved with installing and configuring tocjs and tocjsbis.

Authoring

These plug-ins piggy-back whatever investment you have already made in authoring your DITA topics and map files. Beyond setting up a new ant script for tocjs or tocjsbis, there is nothing additional required.

Integration

The tocjs and tocjs plug-ins present few integration problems or opportunities, especially as regards managing context sensitivity between Help output and the calling software application. Many DITA Help writers customize the `frameset.html` file that ships with the tocjs plug-in to personalize or brand the final product. Here's what Shawn McKenzie does with tocjs on his Sophos corporate website.

The screenshot shows the Sophos Web Appliance Help interface. At the top, the header reads "SOPHOS WEB APPLIANCE HELP" with a globe icon on the right. Below the header is a search bar. On the left, a "Contents" sidebar is visible, listing various topics such as "About Your WS1000", "Getting Started", "Configuration", and "Reports". The main content area is titled "About Your WS1000" and features a photograph of the WS1000 hardware device. Below the image, there is introductory text explaining the importance of preventing network threats like viruses and spyware, and how the Sophos Web Appliance helps with web access policies.

Wrapping other navigational devices around the tocjs output is not difficult. I add the conventional tabs for tri-pane Help systems.

The screenshot shows a custom navigation bar with three tabs: "Contents", "Search", and "Support". Below the tabs, there are links for "Expand all" and "Collapse all". A tree view is displayed below, listing the following items: "About tocjs", "Installing tocjs", "Customizing tocjs for your project", and "Known Issues".

In terms of translation, note that all the text strings associated with entries in the TOC tree are stored in one file named `toctree.js`. These files can be localized, for sure, but they are not very friendly. To make life easier on our sisters and brothers in L10N, you can post-edit this `toctree.js` file to swap JavaScript resource strings for literal strings.

Output

Here again is a link to a live demo of tocjs running at Sophos.

<http://ca-repo1.sophos.com/docs/ws1000/>

Summary

If you are comfortable customizing XHTML output to build an HTML-based help system, you should consider tocjs and tocjsbis. They are sufficiently lightweight to work in Help systems or web-based portals.

Stan Doherty

OASIS DITA Help Subcommittee

[INTERNAL] DHSC Best Practices Proposal

This document describes the general contents for the proposed *Best Practices Guide for Authoring Help in DITA*.

Overview

This document follows up on the proposal for a best practices document approved by the DITA Help Subcommittee (hereafter DHSC) on June 5, 2008 and updated on June 26. The general notion here is that our subcommittee has devoted time to reviewing the current options for developing Help in DITA. Until we can develop Help specializations for DITA 1.3, these options are the only things that DITA-based organizations can implement, short of developing something new from scratch. We would be doing the DITA community in general a service by summarizing our findings and providing links to relevant Help-oriented resources.

Proposed Contents and Contributors

I propose that we keep things pretty simple. If we can develop one DITA topic per content area, I can pull these together into something buildable and reviewable pretty quickly. Following the general shape of the demos that we had over the past few months, here's what I would propose for a starting outline.

Section	Topic	Contributor
Introduction	Introduction	Tony Self (HyperWrite) Stan Doherty (Individual)
Developing DITA-based Help for Existing Help Environments	Introduction	Volunteer??? Stan Doherty (Individual)?
-	EclipseHelp	Jeff Antley (IBM)
-	CSHelp Plug-in	Jeff Antley (IBM)
-	AIRHelp	Scott Prentice (Individual)
-	PTC Help	Chris Goolsby (PTC) Dan Cunningham (PTC)?
-	HTMLHelp	Deborah Pickett (Moldflow)
-	WinANT outputs and optins	Tony Self (HyperWrite)
Developing Custom DITA-based Help Systems	Introduction Need matrix of options/choices.	Stan Doherty (Individual)
-	WinANT context ID management	Tony Self (HyperWrite)
-	TOCJS plugin	Stan Doherty (Individual)
-	HTMLSearch plugin	Stan Doherty (Individual)

Section	Topic	Contributor
-	XHTML (DITA-OT)	Stan Doherty (Individual)
-	CSHelp Plug-in	Jeff Antley (IBM)
-	Dynamically-rendered HTML	Tony Self (HyperWrite)
-	VistaHelp-style expandable content	Tony Self (HyperWrite)
Using DITA Source with Existing HATs	Introduction	TBD???
-	MadCap Flare	TBD???
-	RoboHelp	TBD???
Resources and Demos	Resources	Joe Welinske
-	Demos (links)	Joe Welinske
Futures?	Introduction	Tony Self (HyperWrite) ?

Topic Structure

At the risk of sounding "pushy," I'd recommend that we start with a generic DITA topic that has a common set of headings. If writers need to rearrange them or delete them, fine ... but at least we'd be starting from a common set of items to discuss. Obviously, where there is existing documentation for a plugin or tool, we point off to it ... where there is little or no documentation, we may need to do a little writing.

Here's my take on some common <section> headings to start with.

- *Overview*: discussion of what the plug-in or tool does (features) and why someone might want to use it
- *Setup and configuration*: where to download or purchase it, how to install (high-level) it, and how to configure it (high-level)
- *Authoring*: what to do (if anything) in DITA source files to take advantage of the plugin or tool
- *Integration*: what does someone do to integrate this plugin or tool with an engineering environment (context IDs at a minimum)
- *Output*: screen shots or links to live output
- *Summary*: final comments, evaluations, recommendations

Stan Doherty