Better WebHelp

Contents

Better WebHelp

Better Webhelp (BWH) is a frameless html help plugin for the DITA Open Toolkit. It is an open source project available at https://github.com/Jorsek/BetterWebHelp. It is also available as an output scenario for easyDITA.

How To Structure Your Document

So, you want to use BWH, but you're not sure where to start. This page describes how best to structure your DITA document to best work with Better WebHelp.

For your document to best fit with the basic Better WebHelp, you should follow the general structure described on this page. You can, of course, use other set-ups, however, we may not have tested those, so let us know how they work!

The basic outline of your document should look like the following:

- Bookmap
  - Frontmatter
    - Preface
      - Topicref [with toc="no" attribute] (homepage)
  - Chapter [with href and format]
  - Chapter [with href and format]
  - etc...

This will produce a BWH website that loads the topic located in the preface as a homepage. The homepage will automatically contain links to each of the chapters with their short descriptions.

The navigation panel will consist of a list of the chapters. Usually, the href links on the chapters will be maps.

Important: If the href link is a map, the format attribute on the chapter must be set to "ditamap"

These content maps should look like the following:

- Map [with title attribute]
  - Topicref (intro page)
    - Topicref [with optional navtitle attribute]
    - Topicref [with optional navtitle attribute]
    - etc...

When a chapter is clicked in the navigation panel, it will load the "intro page" topic (which should just be a topic, not a map). Again, the intro page will auto populate with links to its child elements and their short descriptions. It will also expand the navigation panel to show the child topicrefs. If one of the child topicrefs links to a map it should look the same as above and it will behave in the same way.

Note: The value of the title attribute on the map element is what will display in the navigation panel for a chapter.

Installing BWH

The following guides will get you fully up and running with Better WebHelp on your local DITA Open Toolkit and on your instance of easyDITA.

Installing BWH into the DITA-OT

How to install the Better Web Help plugin into the DITA Open Toolkit on your computer.

You must first have the DITA Open Toolkit installed. I will use {dita.dir} as the installation directory.

1. Download the com.jorsek.bwh.zip file.
2. Unzip into {dita.dir}/plugins
3. Run by going into {dita.dir} in the command prompt and using the following command:
   ```
   ant -f build.xml -Dargs.input={input.file.ditamap} -
   Doutput.dir={output.directory} -Dtranstype=bwh
   ```
4.   **Tip:** If you don't want to create the Lunr Search Index on compile, you can set -Dnoindex=true

   ```
   ant -f build.xml -Dargs.input={input.file.ditamap} -
   Doutput.dir={output.directory} -Dtranstype=bwh -Dnoindex=true
   ```

You will get all required files in the specified {output.directory} and to view the results, post the folder on a webserver and navigate to that folder or site.

Set Up a Localhost Server

How to get a localhost server up and running on a mac.

This outlines how to set up a webserver on Mac OSX and has been tested on versions 10.9.3 and 10.10.1.

1. Navigate to *https://discussions.apple.com/docs/DOC-3083* and follow the steps for your version of OSX.
2. To add a BWH website to your local server, you need to create a folder to contain the files: (You can name the folder whatever you want. I am using "bwh" for this example.)
   ```
   sudo mkdir /Library/WebServer/Documents/bwh/
   ```
3. And you must change the permissions so that the DITA OT can write to it:
   ```
   sudo chmod -R 777 /Library/WebServer/Documents/bwh/
   ```
4. Then, when using the DITA OT to build your BWH website, specify the `outdir` to be `/Library/WebServer/Documents/bwh`.
5. Alternatively, you can copy a pre-built project into `/Library/WebServer/Documents`.

To view your website, navigate to http://localhost/bwh.

Installing BWH into easyDITA

How to install the Better Web Help plugin into your company's easyDITA.

More information about installing plugins can be found at: *Configuring the DITA OT in easyDITA*.

1. Go to http://{your-org}.easydita.com and log in as a dba user.
2. In the upper left hand corner, click "Manage Users and Accounts."
3. Select the "Plugin Management" tab and click "DITA Open Toolkit."
4. On this page, you should see "Scenarios" on the left, with "Available DITA Open Toolkits" and "DITA OT Plugins" on the right.
5. Next you must add a DITA Open Toolkit. If you already have an Available DITA OT, skip to Step 6.

a) Go to the *DITA OT SourceForce Page*.

b) This will list all the 1.8.* releases. Download the latest 1.8.*_full_easy_install_bin.zip.

> **Note:** BWH publishing for easyDITA requires at least version 1.8 of the DITA Open Toolkit, and has been tested with 1.8.3.

c) Return to the Plugin Management page and click the "New" button next to the "Available DITA Open Toolkits" header.

d) Select the zip file you just downloaded and it will automatically upload and install.

6. Now, you have to upload the plugin itself.

a) In order to add the plugin, first download the zip file from the *github page*.

b) Click the "New" button next to the "DITA OT Plugins" header and select the downloaded "com.jorsek.bwh_{release-date}.zip" file.

c) It will automatically upload and install.

7. Finally, you need to create a "Scenario" so that you can access the plugin from the Publish tab when viewing a document.

a) Click the "New" button next to the "Scenarios" header.

b) Enter a Scenario Name ("Better Web Help").

c) Select the DITA OT you just installed (in Step 5) from the dropdown list.

d) Click "Change" under "Select Ant File to Run" and select "build.xml" in the root directory of the DITA OT folder you uploaded in Step 5.

e) Add a parameter with the Name: "transtype" with Type: "Simple String Value" and Value "bwh".

f) Leave "User Settable" unchecked.

g) Finally, you should see a list of the installed Plugins on the right sidebar. Click the checkbox next to "com.jorsek.bwh_{release-date}.

h) Save your Scenario.

Now when looking at a resource (by clicking on its name), you should see the "Better Web Help" under the list of "Available Publishing Scenarios" if you add a New Publish within the DITA Open Toolkit.

> **Note:** When publishing documents with the Better Web Help plugin, the build file indexes all the documents on compile time (for offline search). Don't worry if the publish takes a long time (up to 30 minutes or more for many files).

Setting up Google Analytics with BWH

How to get Google Analytics up and running with your instance of Better Web Help.

The code is already built for using Google Analytics, however to track your website you need to enter in your own Tracking ID that you get when you create a GA account.

1. Navigate to www.google.com/analytics and click the "Access Google Analytics" button in the top right hand corner.

2. On the next page, click the "Sign Up" button.

3. Fill out the required form and click "Get Tracking ID"

4. You will then be brought to a page that lists your Tracking ID as well as a block of code. All you need to do is copy the Tracking ID and save it for later.

5. Next, you need to add the Google Analytics code to the webpage. Go to the root directory of your website and open "index.html" in a text editor.

6. Copy the code below and paste it just before the closing </head> tag (right after the <title>{your-title}</title> tag).

```
<script>
 (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||
function(){(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new
 Date();a=s.createElement(o),m=s.getElementsByTagName(o)
```

```
[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)})
(window,document,'script','//www.google-analytics.com/analytics.js','ga');

 ga('create', '{your-tracking-id}', {'cookieDomain': 'none'});
 ga('send', 'pageview');

</script>
```

7. Now simply replace {your-tracking-id} with the Tracking ID you saved from step 5.

8. Save and close index.html.

9. After a few hours, you should be able to navigate back to www.google.com/analytics/web and see your webpage listed there. Click on "All Web Site Data"

10. **Note:** If you see the "Sign Up" page again, make sure you're logged in with the correct account. You can check this with the drop down menu next to your email address in the top right hand corner.

11. This will bring you to the homepage of the Analytics Info.

12. To view pageview data and behavior flow click on "Overview" under "Behavior" in the left side panel. This page pageviews and top pages.

13. You can also click on "Behavior Flow" to see how users navigate your site.

14. Under "Site Content" select "All Pages" to see an overview of the pages viewed by your users. If you scroll down to the bottom of the page, you will see a list of the top pages viewed. You can filter these to only show searches by entering "#/q=" into the search bar at the top.

15. We also highly encourage you to use our Custom Dashboard. When logged in, click on the following link and you will be prompted to import the Dashboard:
    https://www.google.com/analytics/web/template?uid=4Ncv497jRk2VswVMX-P1Qg

16. The dashboard shows you a number of useful statistics, and you are welcome to customize the dashboard to your liking once you've downloaded it.

Hopefully this gets you starting with Google Analytics for you Better Web Help web site.

BWH Customization

Better WebHelp was designed to be completely customizable. You can customize any aspect of BWH, including custom headings and footers, custom colors and spacing and general styling.

BWH Styling

How the styling has been built for the Better Web Help DITA OT plugin.

The Better Web Help styling was built to be adaptive for mobile, tablet, and desktop. This was accomplished by adapting the frameless framework. To make the development simpler, we used LESS for CSS and compiled it before running the DITA OT.

The base styling is provided by the frameless.css stylesheet, compiled by frameless.less (both located in com.jorsek.bwh/resource/scripts/). This provides the basic styling for the webpage and for the drag-up nav panel on mobile.

This can be overwritten by side-drawer.css (as explained in BWH_Customization.pdf) which will change the nav panel to drag out from the left side.

Additionally, the user can add their own overwrite files. We have provided two example template files to get started: user-fonts.css and user-styling.css. These are automatically included in the <head> on build from the DITA OT, so you can just go ahead and edit them to change the styling.

Note: If you do change the user-fonts.css or user-styling.css files, you can safely rerun the DITA OT build script to the same folder and it will not overwrite your customizations.

Follow the examples to get more comfortable with how these customization files can be put to use.

Custom Fonts with Better Web Help

How to customize the fonts with output from the Better Web Help (bwh) DITA OT plugin.

This tutorial assumes that you have already built the website using the DITA OT.

1. Open the output directory that you supplied to the DITA OT.
2. Open "scripts/user-fonts.css."
3. You will see a comment at the top describing the classes and ids to specify to change a specific region:

```
/*
 *
 * Enter custom fonts below. Use the following key to change the specific
 * regions you require:
 *
 * #web-help-c1 {this will affect the navigation panel / table of contents}
 * #web-help-c2 {this will affect the entire content pane, excluding the
 * heading}
 * #web-help-c2 h1 {this will affect the title of the content pane}
 * #web-help-c2 .body {this will affect the body text of the content pane}
 * #web-help-c2 .related-links {this will affect the related links footer in
 * (the content pane}
 *
 * not recommended --> #heading {this will affect the heading (content from
 * heading.html)}
 *    edit styles within heading.html instead
 *
 */
```

4. Using the syntax below, enter the desired fonts.

```
{region} {font-family: {font1}, {font2}, {font3}, ...;}
```

The fonts on your web-help should now be changed. Note that if you rebuild the project to the same output directory, your custom fonts will remain. The build will not overwrite your user-fonts.css file.

BWH user-styling.css Width Example

An example on how to change the width of the columns in the Better Web Help DITA OT plugin output.

In the default output of the DITA OT, the widths of the columns are percent based. They change size depending on the size of the window. In this example, we will fix the width of the Navigation panel and adjust the size of the content pane accordingly.

1. Say you want to change the widths of the Navigation column and the Content column for your version of BWH.
2. Open the user-styling.css file located in bwh/scripts/.
3. The first comment gives you the IDs and classes of important regions of the document. The only ones we're going to worry about in this example are

```
#web-help-c1 {this will affect the navigation panel / table of contents}
#web-help-c2 {this will affect the entire content pane, excluding the
 heading}
```

4. If you scroll down in user-styling.css, you will see the templates for the different media (mobile, wide-mobile, tablet, desktop, and desktop zoom).
5. We don't want to affect the widths on mobile because the Navigation panel becomes a drawer, so under "desktop layout" ("min-width: 57em") let's add a line to fix the width of the Nav panel:

```
/*
 * desktop layout
 * 912-infinity px
 */
@media screen and (min-width: 57em) {
 #web-help-c1 {width: 15.1875em;}
}
```

6. If you are still using the default heading (with the easyDITA logo) you will notice that the panel is now the same width as the logo. We computed this by opening the developer tools on the website to measure the width of the logo (248px) minus the padding and margins on #web-help-c1 (5px) which gives us 243px. Then, since the body font size is 16px, 243px/16px/1em = 15.1875em.

7. Now, we also want fixed width for tablet, but we want it a little smaller, so we can add the following line of code:

```
/*
 * Tablet layout
 * 600-911 px
 * Zoomed in above 600 px
 */
@media screen and (min-width: 37.5em) {
 #web-help-c1 {width: 10.625em;}
}
```

8. However, now we need to adjust the width of the content pane (#web-help-c2) to match the navigation panel.

```
/*
 * Tablet layout
 * 600-911 px
 * Zoomed in above 600 px
 */
@media screen and (min-width: 37.5em) {
 #web-help-c1 {width: 10.625em;}
 #web-help-c2 {left: 13em; right: 0; width: auto;}
}
 /*
  * desktop layout
  * 912-infinity px
  */
@media screen and (min-width: 57em) {
 #web-help-c1 {width: 15.1875em;}
 #web-help-c2 {left: 18em; right: 0; width: auto;}
}
```

9. We rounded up on the left values so that there is a little extra padding between the navigation panel and the content pane.

At this point, you should have a constant width navigation panel when resizing your window and the content pane will resize appropriately.

BWH Custom Content

You can also customize the Content of your Better WebHelp site. BWH uses AJAX to automatically insert your custom headings and footers into the page. Follow the guides to get it set up.

Custom Heading & Logo

How to add a custom heading and logo to the Better Web Help (bwh) DITA OT plugin output.

The default page for the bwh output is automatically built to insert any heading it finds in the output directory. So, all you have to do is design it.

1. First, design a heading. It might look something like this:

```
<div>
 <img src="heading/logo.jpg"/>
 <h1>Web Help</h1>
</div>
```

2. There is styling already associated with <h1> and <img> tags, however you are welcome to override that styling with inline CSS, such as:

```
<h1 style="background-color: red; text-weight: bold; font-size: 28;">
 Web Help</h1>
```

3. Once the header is created, save it as "heading.html" and put it in the root directory of the DITA OT output.

4.   **Note:** The heading.html file will only contain what's necessary for the heading (i.e. it can begin with the \<div> as shown above). There is no need for \<html>, \<head>, or \<body> tags.

5. If you have other content such as images or other files, it is recommended to have a "heading" folder into which you can put all of those. As in Step 1, in your source for your heading, make sure each reference looks into the heading folder. (\)

6. The next time you load the page, your header should pop up on top of the web help. If it doesn't look the way you want, go back in and adjust the inline styling in heading.html.

So, to add a heading to your web help system, all you have to do is include a heading.html file in the root directory of your web app and you're all set.

Custom Footer

How to include a custom footer for a link to your support channel in you implementation of Better Web Help.

This assumes you have already built a Better Webhelp website, and the root directory is bwh/.

1. When you first build a BWH site, a default footer is loaded (footer.html). However, it contains a link to the easyDITA support website, so you should customize this file before launching your website.

2. Open footer.html in a text editing program, and you will see the following:

```
<span id="text">Need help? <a href="http://support.easydita.com/
home">Contact our support.</a></span>
```

3.   **Note:** The footer.html file will only contain what's necessary for the footer (i.e. it should begin with the \<span> as shown above). There is no need for \<html>, \<head>, or \<body> tags.

4. Feel free to change the text ("Need help? Contact our support.") and the href link ("http://support.easydita.com/home") to tailor it to your site.

5. Leave the \<span> surrounding it though. This will assure proper formatting on the site.

On each page load, footer.html will be automatically insterted into the page.

Including a Homepage

How to include a "homepage" (or preface) that loads when you're in the root directory in your implementation of Better Webhelp.

This assumes you are using a bookmap for your final build of your webpage.

1. Type up a dita file that will contain what you need in your homepage.

2.   **Note:** As of right now, the homepage can support a Title, Short Description, and Paragraphs.

3. Save the file and include it as a topicref in the preface of your final bookmap. (The preface is located under frontmatter: /bookmap/frontmatter/preface/topicref.)

4. Within the easyDITA map editor, right click on the preface.dita file you just added and select "Edit properties..."

5. For "toc" select "No" (located most of the way down on the right side).

6. Save the bookmap.

This will create a preface.html file with the contents of your homepage that's not included in the left navigation panel, and it will automatically load on the homepage.

Homepage Section Links

This document explains how the section links and descriptions are created for the homepage (preface page) in Better Webhelp.

When you create a homepage, you add it as the preface in your bookmap to have it automatically load in the root directory of your website. But you'll also notice that the resultant homepage contains section listings and descriptions as well.

When the homepage is created, it takes the navtitles of each chapter in the bookmap (chapter/topicmeta/navtitle) to create the section headings. Then if there is an href on the chapter, it will link the heading to that page and include the short description from that page underneath the section heading.

So, you can automatically create a nice homepage based on the structure of your bookmap by including a preface in the frontmatter and by including hrefs with short descriptions for each chapter heading.

Set Up a Side Nav Drawer
How to set up a sliding side nav panel on mobile for Better Web Help (bwh) plugin for the DITA OT.

This tutorial assumes you've already built the BWH site using the DITA Open Toolkit. I will use {root} as the root directory of the webpage (where you specified "outdir" in the DITA OT).

1. Open {root}/index.html in a text editing program (Notepad, TextEdit, etc).
2. You should see a document that begins with something like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta content="width=device-width, initial-scale=1.0, maximum-scale=1.0,
 user-scalable=0"
   name="viewport" />
  <meta name="viewport" content="width=device-width, minimal-ui" />
  <meta name="mobile-web-app-capable" content="yes" />
  <meta name="apple-mobile-web-app-capable" content="yes" />
  <link rel="icon" sizes="196x196" href="help-icon.png" />
  <link rel="apple-touch-icon" sizes="196x196" href="help-icon.png" />
  <link rel="stylesheet" type="text/css" href="scripts/frameless.css" />
  <link rel="stylesheet" type="text/css" href="scripts/user-fonts.css" />
  <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/
jquery/1.11.1/jquery.min.js">//</script>
  <script type="text/javascript" src="//ajax.googleapis.com/ajax/libs/
jqueryui/1.11.0/jquery-ui.min.js">//</script>
  <script type="text/javascript" src="scripts/typeahead.bundle.js">//</
script>
  <script type="text/javascript" src="scripts/queryString.js">//</script>
  <script type="text/javascript" src="scripts/dragdealer.js">//</script>
  <script type="text/javascript" src="scripts/lunr.js">//</script>
  <script type="text/javascript" src="scripts/lunrScripts.js">//</script>
  <script type="text/javascript" src="scripts/loadContent.js">//</script>
...
```

3. Right below the line that contains 'href="frameless.css"', add the following line:

```
<link rel="stylesheet" type="text/css" href="scripts/side-drawer.css" />
```

4. It should look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta content="width=device-width, initial-scale=1.0, maximum-scale=1.0,
 user-scalable=0"
   name="viewport" />
  <meta name="viewport" content="width=device-width, minimal-ui" />
  <meta name="mobile-web-app-capable" content="yes" />
  <meta name="apple-mobile-web-app-capable" content="yes" />
  <link rel="icon" sizes="196x196" href="help-icon.png" />
```

```
  <link rel="apple-touch-icon" sizes="196x196" href="help-icon.png" />
  <link rel="stylesheet" type="text/css" href="scripts/frameless.css" />
  <link rel="stylesheet" type="text/css" href="scripts/side-drawer.css" />
  <link rel="stylesheet" type="text/css" href="scripts/user-fonts.css" />
  <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/
jquery/1.11.1/jquery.min.js">//</script>
  <script type="text/javascript" src="//ajax.googleapis.com/ajax/libs/
jqueryui/1.11.0/jquery-ui.min.js">//</script>
  <script type="text/javascript" src="scripts/typeahead.bundle.js">//</
script>
  <script type="text/javascript" src="scripts/queryString.js">//</script>
  <script type="text/javascript" src="scripts/dragdealer.js">//</script>
  <script type="text/javascript" src="scripts/lunr.js">//</script>
  <script type="text/javascript" src="scripts/lunrScripts.js">//</script>
  <script type="text/javascript" src="scripts/loadContent.js">//</script>
...
```

When you next open your webpage, you should see a side drawer when the window size is less than 600 pixels wide.

Adding a Manifest File
How to add a webapp manifest file to your Better Web Help.

A webapp manifest tells your webpage how to cache pages. Based on how it is used, it can allow your website to be accessed offline. There is already a manifest file built for BWH. All you need to do is modify it to include any additional resources.

1. The manifest file is located in the root directory of your webpage (bwh/ on the github repository) and is called webapp.manifest.
2. Under the "CACHE:" heading, you will see a long list of files. Here, add any additional files your page requires (such as files in the heading/ folder or additional scripts or images)
3. **Note:** You shouldn't include all the content files here since you have to write each file out individually. That will be covered in the next step.

4. Under "NETWORK:" you see an asterisk. This means that the manifest will cache all other pages as they are encountered. This will cover all the content pages.
5. Then "FALLBACK:" says that if there is any page that isn't cached yet, it should load fallback.html. This file simply states "Oops, this files isn't cached yet, sorry!"
6. Once your manifest is built. You need to add it to the webpage. Change the opening <html> tag to the following:

```
<html manifest='webapp.manifest'>
```

When you next load the webpage if you have the developer tools open, you will see the manifest checking for each of the files listed (written out in the console).

Important: If you alter any of the files listed in the manifest, they will not update on the website until the manifest itself is updated. Simply update the version line to reload all cached items.