Name

herold — HTML to DocBook converter

Synopsis

`herold` [OPTIONS]

Description

The reuse of HTML content in presentation-neutral form is a frequent problem. One possible solution is to convert HTML to DocBook XML, because DocBook is a semantic markup language for documentation, which enables its users to create document content that captures the logical structure of the content.

The command line tool herold can be used to convert HTML to DocBook. Because HTML elements are often used not as intended, the possibilities for such a transformation are somewhat limited. herold is part of the dbdoclet suite of tools. For more information visit http://www.dbdoclet.org.

Options

--docbook-add-index, -x	Automatically add an index element at the end of the document.
--docbook-decompose-tables, -T	Decomposes the tables from the HTML code into single paragraphs. This can be useful, if a document contains a lot of tables for formatting reasons.
--docbook-encoding, -d	Specifies the encoding of the generated DocBook XML files.
--docbook-root-element, -r	The root element of the document. Possible values are: book, article, reference, part, chapter or section. The default value for this option is 'article'
--docbook-title, -t	The title for the resulting document.
--in, -i	Specifies the HTML input file.
--help, -h	Prints a help page on the console.
--html-encoding, -s	Specifies the encoding of the HTML source files, such as ISO-8859-1.

--out, -o	Specifies the DocBook XML destination file.
--profile, -p	A profile file with predefined settings.
--verbose, v	Enables the verbosity for the console output.
--version, -V	Displays the version of herold.

Configuration

The details of a transformation are controlled by a profile file. A profile file offers more possibilities to influence the transformation than the command line arguments. The following example shows a typical profile file.

```
 1: transformation html2docbook;
 2:
 3: section section-detection  {
 4:     attribute-class = ["^MsoHeading(\d+)$"];
 5:     section-numbering-pattern = "((\d+\.)+)?\d*\.?\p{Z}*";
 6: }
 7:
 8: section list-detection {
 9:     itemized-attribute-class = ["^MsoListBullet(\w*)$", "Aufzhlung(\w+)$];
10:     itemized-strip-prefix = [ "-", "o", "\u00b7" ];
11:     ordered-attribute-class = ["^MsoListNumbered(\w*)$"];
12:     ordered-strip-prefix = [ "\d+\.\s+" ];
13: }
14:
15: section HTML {
16:     encoding = "windows-1252";
17:     exclude = [ "//p[starts-with(@class, 'MsoToc')]", "" ];
18: }
19:
20: section DocBook {
21:     abstract = """<title>Lorem ipsum</title>
22: <para>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
23: do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
24: enim ad minim veniam, quis nostrud exercitation ullamco laboris
25: nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
26: reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
27: pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
28: culpa qui officia deserunt mollit anim id est laborum.sed, dolor
29: amet.</para>""";
30:     add-index = true;
31:     author-email = "me@somewhere.de";
32:     author-firstname = "Michael";
33:     author-surname = "Fuchs";
34:     chunk-elements = [ "chapter", "section", "appendix" ];
35: // Syntax: chunk-<CHUNK-ELEMENT>-depth = <INT>;
36:     chunk-section-depth = 3;
37:     collapse-protected-space = "true";
38:     copyright-holder = "Ingenieurbüro Michael Fuchs";
39:     copyright-year = "2012";
40:     corporation = "";
41:     create-condition-attribute = false;
42:     create-prolog = true;
```

```
43:     create-remap-attribute = false;
44:     create-xref-label = false;
45:     decompose-tables = false;
46:     detect-trapped-br = true;
47:     documentation-id = "doc01";
48:     document-element = "book";
49:     encoding = "UTF-8";
50:     hyphenation-char = "soft-hyphen";
51:     image-data-formats = [ "gif", "base64" ];
52:     image-path = "./figures";
53:     language = "de";
54:     release-info = "Version 3.1";
55:     table-style = "all";
56:     title = "Tutorial";
57:     title-normalize-space = true;
58:     use-absolute-image-path = false;
59: }
60:
```

Syntax

A profile file consists mainly of sections. Sections are used to group parameters which share the same context. Every section must start with the keyword `section` followed by the name of the section. After the name comes the block of parameters, which is surrounded by curly braces. Parameters can be of type String, Number, Boolean or Array. Strings must be framed with double quotes. If the String contains newlines, use three double quotes instead of one. Arrays are framed with square brackets. Inside an array, the elements must be comma separated. Every assignment must be finished by a semicolon. Multi line comments have the form `/* my comment */` , single line comments look like `// my comment\n`.

Mandatory Elements

A profile for herold must start with the line `transformation html2docbook;`.

Section section-detection

The section `section-detection` is used to detect section elements in HTML code and to strip off any numbering prefix from the titles.

Many authoring tools allow deeply nested sections. While exporting HTML, it happens, that the nesting becomes deeper than six levels. HTML provides header elements for up to six levels, h1-h6, but no h7 or even more. At this point, the formatting is normally done with the help of CSS and div or p elements. herold is able to detect the header element of HTML, but it can not know about the export format of a specific tool. To solve this problem even for some cases, you can specify the parameter `attribute-class`. It consists of a list of regular expressions, which are matched against the class attribute of each HTML element. If a match is found, the element is considered as a section element. The regular expression can have group, which is interpreted as level indicator. The

group must be the first group and it must match against a number, e.g. `^heading(\d+)$`. If the level can not be detected, a level of seven is assumed.

Because DocBook XSL stylesheets take care of the section numbering while transforming the DocBook XML to a specific output, it is often necessary to strip the numbering already defined in the HTML page. Otherwise you end up with two numbering texts in front of your titles. To help herold with the detection of numbering patterns, use the parameter `section-numbering-pattern`.

attribute-class
: A regular expression, which is applied to every p and div element. If the expression matches, the current element is handled as a section element. If the regular expression has groups, the first group will be used as nesting level, otherwise level seven is assumed.

section-numbering-pattern
: Normally you want to get rid of the section numbering that comes with the HTML data, because it becomes part of the title text in DocBook. The section numbers will the appear twice in your target media. One from HTML and one from the DocBook XSL processing. The parameter section-numbering-pattern defines a regular expression, which is matched against the beginning of every section title. If it matches, the matching part is removed.

Section list-detection

Sometimes lists are not represented with ul, ol or dl tags, but they are represented as p tags with additional css formatting. If you use a tool, which creates or exports HTML with such a construct, the conversion will end up with para elements, instead of the corresponding list elements in DocBook. To recreate the lists in some cases, you can use the section `list-detection`. The parameters `itemized-attribute-class` and `ordered-attribute-class` let you define lists of regular expression, which should match against the class attribute of listitem elements in the HTML. herold tries to rebuild the proper list structure from this information, even for nested lists.

Section HTML

The section HTML defines parameters, which control the loading and parsing of the HTML input data.

encoding
: The character set used to read the input stream.

exclude Defines an array of xpath expressions. All matches are removed from the HTML DOM tree before transformation.

Section DocBook

abstract	The text for the abstract element of the info section. If the text is structured with newlines, use three double quotes as delimiters. If the text starts with a "<" character, it is embedded into an abstract element, otherwise the text is embedded into an para element inside of an abstract element. The text will parsed and can contain DocBook elements.
add-index	If set to true, an index element is inserted at the end of the DocBook XML.
chunk-elements	Defines an array of element names. If an element of this list is detected while writing the output, the element and all child nodes will be written to a separate file. This new file will be included into the parent file with an xi:include tag. Recursive structures result in recursive includes. You might want to use this, if you are transforming big HTML files and the resulting DocBook XML file becomes uncomfortable large.
chunk-<CHUNK-ELEMENT>-depth	Defines the depth for a chunk element, until the chunking should be executed, eg chunk-section-depth = 3. If an element defined for chunking is nested recursivley, you might want to control the depth to which the chunking should be done. The default depth is 1, which means only the topmost element is separated.
create-xref-label	if set to false, anchor elements doesn't get a xreflabel attribute.
decompose-tables	If set to true, tables structures will be ignored. The content of the table cells will be inserted into the DocBook XML as a sequence of paragraphs. This parameter can be useful if your HTML contains tables for formatting purposes. Normally you want to get rid of them, because they tamper the logical structure.

document-element	The document element you want to use. Must be one of article, book, part or reference.
encoding	The character set which will be used for writing the output file.
image-data-formats	An array of image formats. These formats will be inserted as imageobject elements, additionally to the format found in the src attribute of the corresponding img element. The original format is inserted twice with the roles "html" and "fo". The other formats are inserted as "html-<FORMAT>" and "fo-<FORMAT>".
title	The title of the resulting document. If this parameter is undefined, herold tries to dected the title from the head section of the HTML data.
use-absolute-image-path	If you want absolute image paths in the fileref attribute of the imagedata element, set this parameter to true.

Copyright