TECHNICAL WHITE PAPER

HOW TO AVOID LEARNING XML

motive

CONTENTS

WHY WOULD I WANT TO AVOID LEARNING XML?

The title of this talk is actually an allusion to Mike Smith's article Don't learn XML (http://xml.oreilly.com/news/dontlearn_0701.html), where he suggests using the DocBook DTD and stylesheets to get started using XML for documentation rather than starting from scratch, writing your own DTD, stylesheets, and so on. My title is more than a little misleading. I'll explain how to get started with DocBook without learning much XML, but I also hope to show you how to get started learning the technologies that make a DocBook-based system tick.

Note

The html version of this page can be viewed with any browser. If you're viewing this page as html in the Opera web browser (http://www.opera.com), press F11 to view the slide show version in full screen.

GOALS OF THIS TALK

- To explain what DocBook *is*, what the pieces are, and how they fit together in a production environment.
- To show examples of what kind of output is possible from DocBook using the stock stylesheets and customizations of them and give you an idea of what it takes to create and maintain them.
- To give me a reason to play with Operashow (http://www.opera.com/support/tutorials/operashow/) and DocBook. Operashow is a feature of the Opera (http://www.opera.com) web browser that causes it to key off of simple css and present an html page as a slide show. The advantages of Operashow:
  - Very light weight
  - Easy to publish a fuller version of the talk on the Web
  - The published version can easily be made accessible to those with disabilities (http://tntluoma.com/opera/operashow/molly/) (contrast PowerPoint)
  - Opera is free and PowerPoint isn't
  - No temptation to use goofy DHTML effects

WHAT IS DOCBOOK?

What is this “DocBook” you speak of?

- “DocBook” is **not** an application in the sense that you're probably used to. It is not a program that runs on a computer like FrameMaker or Word.
- Strictly speaking, DocBook is a set of rules that defines how to *structure* a document. These rules can be and are expressed in English prose, as a DTD (Document Type Definition), and in a few schema languages. The English prose version is useful for authors

and stylesheet writers. The DTD and schemas are useful for applications such as XML editors, validators, and processors.

This structure is intended to provide the basis for the more specific needs of those who want to document computer software and hardware. You are expected to customize DocBook, at the very least removing some elements, there are two type of sections, recursive and non-recursive. You should pick one for your needs and remove the other.

- The set of rules is an open standard supported by Oasis (http://www.oasis-open.org/) ("OASIS is a not-for-profit, global consortium that drives the development, convergence and adoption of e-business standards.").
- The set of rules has been officially released as an SGML and XML DTD (Document Type Definition).
- The DocBook DTD has been around since 1991. The current version is 4.2, so it is a mature content model.

What does DocBook look like?

A DocBook XML document looks a lot like the xhtml source—both are instances XML DTDs. The key difference is that DocBook is primarily semantic markup for describing software and hardware documentation. xhtml is more general and less focused on semantics.

Example 1. Sample DocBook XML:

```
<para><command>ControlProcess</command> writes
its trace information to <filename>
<envar>$BJROOT</envar>/logs/ControlProcess_
<replaceable>hostname</replaceable>.out</filename>
(where <replaceable>hostname</replaceable> is the name
of the activation host on which
<command>ControlProcess</command> is running).</para>
```

Example 2. Sample XHTML:

```
<p><tt>ControlProcess</tt> writes
its trace information to <tt>
<i><tt>$BJROOT</tt></i>/logs/ControlProcess_
<i>hostname</i>.out</tt>
(where <i>hostname</i> is the name
of the activation host on which
<tt>ControlProcess</tt> is running).</p>
```

Example 3. The DocBook converted to html and rendered:

`ControlProcess` writes its trace information to `$BJROOT/logs/ControlProcess_`*`hostname`*`.out` (where *hostname* is the name of the activation host on which `ControlProcess` is running).

Ok, but what is DocBook *really*?

More broadly defined, DocBook is the DTD mention above and the mechanisms to transform DocBook documents to a useful format.

There are at least three separate existing mechanisms that I know of for transforming DocBook instances, each based on a different stylesheet language. If none of these mechanisms suit your needs, it is possible to create a new one from scratch, though that would be non-trivial. The important point is that DocBook is an open standard, so ultimately you own your data (http://www.troubleshooters.com/tpromag/200104/200104.htm).

Of the existing mechanisms, two are free (as in freedom and beer (http://www.gnu.org/fsf/fsf.html)) open source and a third is commercial:

- XSL stylesheets maintained by Norm Walsh and others at the DocBook Open Repository (https://sourceforge.net/projects/docbook/) for converting DocBook XML documents to html, chunked html, html help, xsl-fo (which can in turn be converted to postscript or pdf), UNIX man pages. In addition, html can be converted to text using a text browser like links or lynx.
- DSSSL (Document Style Semantics and Specification Language) stylesheets maintained by Norm Walsh and others at the DocBook Open Repository (https://sourceforge.net/projects/docbook/) for converting DocBook XML or SGML documents to html, chunked html, pdf, rtf, UNIX man pages. I have limited knowledge about the DSSSL stylesheets since I've primarily used the XSL stylesheets.
- FOSI stylesheets that are part of some Arbortext products for converting DocBook instances to html, html help, cross-browser html base help, and print. ...and possibly more, I have limited knowledge about Arbortext's products. I only looked at some demos briefly and long ago. Their content engine is pricey.

What can DocBook be made to look like?

XML facilitates creating multiple output from a single source. All of the documents linked below were created from the identical XML file (XML2PDF-presentation.xml). This talk also is what it is about. I wrote it in a form of DocBook (the slides DTD) and use XSLs to transform it into various output types.

- The slide show you're looking at now/a single, monolithic htmlIf you are viewing this page on the web, open this page in the Opera web browser (http://www.opera.com) and press F11 to view as a full screen slide show.
- “chunked” html : each chapter, section, and so on is broken into a separate html page to create an online book.
- HTML Help or RoboHELP's WebHelp : We also created chms from our server books because that format was more convenient in some circumstances. We used a less booklike variant for help sets. If you have a chm file, you can easily create WebHelp by opening the .hhp file in RoboHELP and generating the WebHelp. This requires owning RoboHELP.
- Print output using the stock DocBook XSLs: the output of the DocBook XSL stylesheets if you don't customize anything.
- The Motive pdf : this is the output from the stylesheets we use at Motive. After Motive acquired BroadJump, I customized the DocBook stylesheets to mimic the FrameMaker template they were using.

- Eclipse documentation plugins (http://help.eclipse.org/help21/index.jsp).
- The BroadJump pdf : this is the output from the stylesheets we used at BroadJump.
- The BroadJump “technical whitepaper” pdf : We created this stylesheet for a series of whitepapers that described our product's architecture.
- The DocBook 'slides' XSLs : A browser based slide show that doesn't depend on Opera like the one you're viewing now does. There are XSLs to create html with and without frames as well as fo/pdf.
- You can even create a Word doc out of it.

HOW DO I WRITE A DOCBOOK DOCUMENT?

This is easily the most difficult thing to get used to when composing in DocBook or DTD that tries to focus on semantics rather than presentation. There's not one short or even long answer to the question of what the best authoring environment is, but recently a couple of editors have made strides in solving the problem of how to represent semantics and reasonable presentation at the same time.

WYSIWYG is ultimately impossible any time you produce multiple outputs from a single source. This subject is discussed every few months on one of the XML or DocBook lists.

See a list (http://wiki.docbook.org/topic/DocBookAuthoringTools) of DocBook authoring tools at the DocBook Wiki.

In addition to providing help authoring instances of DocBook XML (or any DTD) by indicating what elements are valid at what points, these tools validate documents. When a tool validates a document, it compares the XML to its DTD to see if it conforms to the DTD. If the document does not conform to the DTD, the tools typically try to indicate what and where the problem is, though they can't always tell you exactly what is wrong.

- XMLMind XML Editor - XXE (Lite, but functional version free for “internal use”; ~$220 for a fully enabled version) (http://www.xmlmind.com/xmleditor/) : XMLMind uses CSS2 plus some extensions to present the document in a way that approximates what its final presentation might be. That is, lists look like lists, tables like tables, and so on. The editor also provides visual cues to indicate what element is currently selected and what your context is. A node-path bar shows the exact context and allows the writer to select a specific node. XMLMind supports xinclude, but has limited support for entities. If you're starting out, you can probably avoid the things that XMLMind can't do with entities, but if your existing docs or publishing system already requires that you use them, XMLMind may not be for you.
- Syntext Serna (~$254) (http://www.syntext.com/products/serna/) : Serna uses xslt and xsl-fo to style the document while you type and also uses tooltip-like visual cues to indicate where you are in the markup. Serna has full support for entities and allows you to edit them in context in the document. Another nice feature is that xrefs are resolved in the editing view, so it's even closer to wysiwyg.

- emacs + psgml mode and nsgmls(free): The one true editor. Great if you like to look at tags. psgml mode gives you quite a bit of help and it is very stable. You have to edit tables by hand, however.
- Arbortext Epic (c. $700?/seat): I have only demoed this briefly, but it has got a reputation of being the Rolls Royce of XML editors.
- XMetaL: BlastRadius, formerly Corel, and before that Softquad (c. $470?/seat): XMetaL has some nice features, but requires that you customize it to make it's WYSIWG interface usable. It's really unfortunate that they don't include a DocBook kit that works "out-of-the-box". I'm happy to share my macros etc. if anybody asks though.
- Adobe FrameMaker 7 ($870): Based on what I've read on mailing lists (http://www.getnet.net/~swhitlat/DocBook/Frame_Project_Readme.html), FrameMaker is not up to the task of providing you with and editing environment for DocBook. It contains a DocBook kit, but it would need lots of configuration before it would be useful.
- XML Spy/Authentic ($400 / $0): Really more suited to the needs of "dataheads" and forms. Authentic, which allows you to edit and validate documents, is now free. It contains a DocBook kit, but it would need lots of configuration before it would be useful.

There are a few other editors at various levels of maturity. I hear Oxygen and Morphon mentioned sometimes.

How do I transform a DocBook document?

I'm **not** discussing DSSSL or fosi here.

Most of the tools that transform XML documents are command line. Example using `xsltproc`:

```
xsltproc -o output.xml path/to/html/docbook.xsl input.xml
```

To use XSL to go to HTML Help, you must also run `hhc` or use the HTML Help workshop to compile the chm.

XSL does not take you directly to pdf. You use XSL to convert the DocBook document to xsl-fo, then use a fo renderer to convert that document to a pdf or postscript file.

There are several convenience tools (http://wiki.docbook.org/topic/DocBookTools) that hide the transformation process from you to one degree or another. You certainly need either one of these or something you write yourself (a batch file, shell script, perl script, make file, ant script)

What is fo and what is a fo renderer?

FO (Formatting Objects) is part of the w3c's XSL (http://www.w3.org/TR/xsl/) specification. It is like html in that it describes how content should be presented, but unlike html, fo focuses on the printed page, so it provides for headers, footers, page number, and so on.

> XSL is a language for expressing stylesheets. It consists of three parts: XSL Transformations (XSLT): a language for transforming XML documents,

> the XML Path Language (XPath), an expression language used by XSLT to access or refer to parts of an XML document. (XPath is also used by the XML Linking specification). The third part is XSL Formatting Objects: an XML vocabulary for specifying formatting semantics. An XSL stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary.
>
> — w3c (http://www.w3.org/Style/XSL/)

- RenderX's XEP (http://www.renderx.com)
- Antenna House's XSL Formatter (http://www.antennahouse.com)
- Apache's FOP (http://xml.apache.org/fop/)

See the DocBook Wiki (http://wiki.docbook.org/topic/DocBookPublishingTools) for a complete list.

How do I change the way the output looks?

For many aspects of a document's appearance, you can control the behavior of the stylesheets by changing parameters.

```
<xsl:param name="double.sided" select="1"/>
<xsl:param name="page.margin.inner">
  <xsl:choose>
    <xsl:when test="$double.sided != 0">1.25in</xsl:when>
    <xsl:otherwise>1in</xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="page.margin.outer">
  <xsl:choose>
    <xsl:when test="$double.sided != 0">0.75in</xsl:when>
    <xsl:otherwise>1in</xsl:otherwise>
  </xsl:choose>
</xsl:param>
```

The parameters are described in the documentation (http://docbook.sourceforge.net/release/xsl/current/doc/fo) that comes with the distribution.

Generated text

In addition to parameters, the stylesheets come with translations for strings that are generated. For example, one of the parameters lets you control how titles are formatted:

```
<l:context name="title-numbered">
   <l:template name="appendix" text="Appendix %n. %t"/>
   <l:template name="article/appendix" text="%n. %t"/>
   <l:template name="chapter" text="Chapter %n. %t"/>
   <l:template name="section" text="%n. %t"/>
</l:context>
```

Separate files exist for each language and the stylesheets come with translations for around 40 languages.

What if there is no parameter for the thing I want to change?

In that case, you have 2 choices:

1. If the thing you want to control would be generally useful to other users of the DocBook XSLs, submit an RFE to the maintainers.
2. Change the behavior of the XSLs by overriding the templates.

   Overriding the templates requires that you know 1) Enough XSL to change the right code, and 2) what you want the resulting XML to do. So if you're changing the fo stylesheets, you have to know enough FO to know what you want the stylesheets to do.

Bob Stayton's book, *DocBook XSL: The Complete Guide (http://www.sagehill.net/docbookxsl/index.html)* , is essential reading for learning how to customize the DocBook XSLs.

Glossary generation

One handy feature of the DocBook Open XSLs is that you can have a master glossary from which a custom glossary is built when you generate a document.

Note

The inclusion of terms in the glossary is *not* recursive. If the definitions of terms contain `glossterms`, these are not pulled into the glossary.

Example 4. Pointing the XSLs to your master glossary

```
<xsl:param name="glossary.collection"
select="'/local/path/to/glossary.xml'"/>
```

Example 5. A very small master glossary

```
<!DOCTYPE glossary PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
"http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd">
<glossary>
  <glossentry>
    <glossterm>0</glossterm>
     <glossdef>
      <para>Numeric zero, as opposed to the letter 'O'.</para>
    </glossdef>
  </glossentry>
</glossary>
```

Example 6. Using glossary term in your document

```
<para>There's no Roman numeral for <glossterm
   baseform="0">zero</glossterm>.</para>
```

Profiling

Another feature of the DocBook Open XSLs is the ability to filter out elements. This is like conditional text in FrameMaker.

Example 7. Some paras ready to be profiled

```
<para>A common introductory paragraph.</para>
<para os="Windows">A paragraph specific to Windows.</para>
<para os="UNIX;MacOSX;Linux">A paragraph pertains to several UNIXes.
<phrase userlevel="beginner">When in doubt, use <userinput>man -k</userinput>
</phrase>
</para>
```

Example 8. Running Saxon with profiling

```
java com.icl.saxon.StyleSheet -o sample.html sample.xml \
      ../html/profile-docbook.xsl \
      "profile.os=Windows" \
      "userlevel.os="beginner"
```

This behaves a little differently than FrameMaker. In FrameMaker, if you turn on a condition, then it appears even if it occurs within text that has been conditioned out.

Caution

Profiling, in part *because* it is more powerful than Frame's conditional text, provides an easy way for you to hang yourself.

The directory structure of the distribution

The XSL stylesheet distribution comes with several sets of stylesheets. The folder names indicate their purpose: `fo`, `html`, `htmlhelp`. See “Profiling” on page 9 for information about profiling.

`html/`	Use `docbook.xsl` to generate a flat html file.
	Use `profile-docbook.xsl` to generate a flat html file using profiling.
	Use `chunk.xsl`...
	Use `profile-chunk.xsl`...
`htmlhelp/`	Use `htmlhelp.xsl` to generate the collection of html pages, .hhc, .hhp, and .hhk files necessary to make a chm.
	Use `profile-htmlhelp.xsl` to generate the collection of html pages, .hhc, .hhp, and .hhk files necessary to make a chm using profiling.
`fo/`	Use `docbook.xsl` to generate an xsl-fo file.
	Use `profile-docbook.xsl` to generate an xsl-fo file using profiling.

REFERENCES AND RESOURCES

The problem with DocBook is not a lack of documentation. In fact, there are probably too many guides for getting started that well meaning people have posted on the web over the years, but

you may have difficulty figuring out which of these guides is current, which addresses your needs, and so on.

DocBook

- The official DocBook (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=docbook.org/) website.
- The DocBook Open Repository (http://sourceforge.net/projects/docbook/) at Sourceforge

  From here you can download the latest release of the stylesheets. The docs page includes a list of resources. Start with Five steps for finding answers to DocBook questions (http://sourceforge.net/docman/display_doc.php?docid=10430&group_id=21935).
- The docbook (http://lists.oasis-open.org/archives/docbook/)and docbook-apps (http://lists.oasis-open.org/archives/docbook-apps/) mailing lists
- The DocBook Wiki (http://wiki.docbook.org/topic/)
- Bob Stayton's *Using the DocBook XSL stylesheets (http://www.sagehill.net/xml/docbookxsl/index.html)*
- *DocBook, The Definitive Guide*: The online version is more up to date. (http://docbook.org/tdg/en/html/docbook.html)

Notice that you have to go to two separate places to get the DTD and stylesheets. That may seem strange (and even inconvenient), but there's a reason behind it. The DTD dictates the semantics of a DocBook document. There are some processing expectations associated with many elements, but it is not a requirement and further, there is no presupposition about what tool will be used to process the document. The stylesheets maintained at the DocBook Open Repository are not the last word on processing DocBook documents. You are free to build or buy a different implementation because you own your data.

XSLT

- Michael Kay's XSLT Programmer's Reference (http://www.wrox.com/Books/1861003129.htm)
- Ken Holman's XSL course and book (http://www.cranesoftwrights.com/)
- xsl-list (http://www.mulberrytech.com/xsl/xsl-list/) mailing list

XSLFO

- XSLFO lists: the yahoogrouops xsl-fo list (http://groups.yahoo.com/group/XSL-FO/) and the w3c xsl-fo list (http://lists.w3.org/Archives/Public/www-xsl-fo/).
- Zvon.org's XSL FO Reference (http://www.zvon.org/xxl/xslfoReference/Output/index.html)
- Dave Pawson's XSL-FO book (buy (http://www.oreilly.com/catalog/xslfo/) or read online (http://www.dpawson.co.uk/xsl/sect3/bk/index.html))

- Ken Holman's XSL-FO course and book (http://www.cranesoftwrights.com/)
- Eliot Kimber's Using XSL Formatting Objects for Production-Quality Document Printing (http://www.isogen.com/papers/production-quality-xsl-fo.pdf) presents the state of xsl-fo renderers
- An interesting looking article on xsl-fo that I haven't had a chance to read closely yet: What Is XSL-FO and When Should I Use It? (http://www.seyboldreports.com/TSR/free/0217/techwatch.html)

Questions and Answers

Is everything clear as mud?