

DocBook: From Syntax to Publication

**Norman Walsh
Sun Microsystems, Inc.**

Table of Contents

About DocBook

About Norman Walsh

A Little Background

DocBook V5.0

DocBook Markup

Publishing

Stylesheet Customization

Schema Customization

...

About DocBook

- **DocBook is a schema maintained by the DocBook Technical Committee of OASIS**
- **Available as an SGML or XML DTD, RELAX NG Grammar, or W3C XML Schema**
- **Particularly well suited to books and papers about computer hardware and software (though by no means limited to these applications)**
- **About 10 years old (it was actually 13 on 10 November 2005)**

About Norman Walsh

- **Member of the Java Web Technologies and Standards group at Sun Microsystems, Inc.**
- **Chair of the DocBook TC**
- **Active participant in web standards at W3C (XML Core, XSLT, TAG) and OASIS (DocBook, Entity Resolution, RELAX NG)**
- **Specification lead for JSR 206: *Java API for XML Processing***
- **Long-time markup geek**

A Little Background

A quick look at some features of structured documentation.

Structured Documentation

Benefits

Technical Challenges

Non-Technical Challenges

Storing Structured Documentation

XML or SGML or ...

OASIS

Evolution

Structured Documentation

- **Semantic rather than presentational**
- **Components have *identifiable* structure**
- **ASCII and Word (without templates) are not structured**
- **HTML and Word are somewhat structured**
- **DocBook is strictly structured**

Benefits

- **Multiple presentations from the same source (print, online, help, etc.)**
- **Documentation reuse**
- **Authors no longer have to worry about presentation**
- **Opportunities for improved authoring interfaces**

Technical Challenges

- **Relatively sophisticated processing required for presentation**
- **Document reuse requires careful management**
- **Users benefit from special authoring tools**

Non-Technical Challenges

- **Writing reusable documentation is different**
- **Authoring within a strict structure is different**
- **Authoring tools are different (and sometimes expensive)**

Storing Structured Documentation

- **XML is a natural system for storing structured documentation**
- **XML can be used to develop different vocabularies**
- **DocBook is an XML vocabulary designed for computer hardware and software documentation**

XML or SGML or ...

- **DocBook was historically an SGML DTD.**
- **DocBook V4.x is**
 - > **An XML or SGML DTD**
 - > **Has an unofficial RELAX NG Grammar**
 - > **Has an unofficial W3C XML Schema**
- **DocBook V5.x is**
 - > **A RELAX NG Grammar**
 - > **Has a non-normative XML DTD**
 - > **Has a non-normative W3C XML Schema**

OASIS

- **OASIS: The Organization for the Advancement of Structured Information Standards**
 - > **A non-profit, international consortium that creates interoperable industry specifications based on public standards such as XML and SGML. OASIS members include organizations and individuals who provide, use and specialize in implementing the technologies that make these standards work in practice.**
- **DocBook is a work product of the OASIS DocBook Technical Committee**

Evolution

- **DocBook is *stable*.**
- **Backwards incompatible changes can only occur at full version revisions (5.0, 6.0, etc.).**
- **Minor revisions (3.1, 4.1, 4.1.2) are always backwards compatible.**
- **Backwards incompatible changes have to be announced a full version before they are implemented.**
- ***Except* for the V5.0 release.**

DocBook V5.0

Introducing DocBook V5.0.

Design Goals

Big Picture Changes

Design Goals

- **Must “feel like” DocBook**
- **Most existing documents should still be valid or it should be possible to transform them in simple, mechanical ways into valid documents**
- **Must enforce as many constraints as possible in the schema. Some additional constraints are expressed with Schematron rules**
- **Must simplify many content models**

Design Goals (Continued)

- **Must give users the flexibility to extend or subset the schema in an easy and straightforward way**
- **Must allow the generation of a (non-normative) XML DTD and W3C XML Schema.**

Big Picture Changes

- **DocBook V5.0 is in the namespace `http://docbook.org/ns/docbook`.**
- **A version attribute is now required on the root element.**
- **Element and attribute names are case-sensitive (as they have been in the XML DTD since 4.0).**
- **Content models are generally simpler.**
- **Uses XLink; allows most inlines to be link elements.**

Big Picture Changes (Continued)

- **Many constraints are now enforced by the grammar. They are not new, but they could not previously be enforced by DTDs.**
- **A few simple datatypes are used.**
- **Accessibility has been improved with the introduction of `alt` and `annotation` tags.**

DocBook Markup

The elements of DocBook...

Element Classes

"Information Pool" Elements

Inlines

Inline Examples

Inline Categories

Linking

Paragraphs

Examples, Figures, Tables, ...

...

Element Classes

- **There are two main classes of elements in DocBook**
 - > **“Hierarchy” elements provide gross structure**
 - > **“Information Pool” elements provide prose markup**
- **The information pool could be reused in a new hierarchy**
- **Conversely, the hierarchy could be preserved with a new technical vocabulary**

"Information Pool" Elements

- **Inlines (publishing, linking, markup, user interfaces, programming, operating systems, ...)**
- **Examples, figures, tables, and equations**
- **Graphics (media objects)**
- **“Verbatim” (program listings, screens, ...)**
- **Admonitions (caution, warning, note, ...)**
- **Lists (ordered, itemized, simple, ...)**

Inlines

- There are roughly *100* inline *elements*:
- they identify commands (`ls`),
- code fragments (`x := 4`),
- dates (2005-10-06), etc.
- The `phrase` element is a general purpose wrapper.

Inline Examples

```
<para>There are roughly <emphasis>100
</emphasis> inline
<glossterm baseform="element">ele-
ments </glossterm>: they identify
commands (<command>ls</command>),
code fragments (<code>x := 4</code>),
dates (<date>2005-10-06</date>), etc.
The <tag>phrase</tag> element is a
general purpose
<phrase>wrapper</phrase>.</para>
```

Inline Categories

- **Technical** (`package`, `termdef`, ...)
- **Error related** (`errorcode`, `errorname`, ...)
- **Programming** (`function`, `varname`, ...)
- **Products** (`productname`, `trademark`, ...)
- **Operating system** (`envar`, `filename`, ...)
- **Markup related** (`tag`, `token`, `literal`, ...)
- **Bibliographic** (`citation`, `author`, ...)
- **Publishing related** (`acronym`, `footnote`, ...)
- **Graphic** (`inlinemediainobject`)

Inline Categories (Continued)

- **Keyboard related** (`keycap`, `shortcut`, ...)
- **Indexing** (`indexterm`)
- **GUI related** (`guiicon`, `guibutton`, ...)
- **Links** (`link`, `xref`, `olink`, `anchor`)

Linking

- **DocBook uses ID/IDREF linking**
 - > `<link linkend="someid">hot text</link>`
 - > `<xref linkend="someid" />`
- **DocBook V5.0 adds XLink**
 - > `<link xlink:href="someURI">hot text</link>`
 - > `<command xlink:href="#someid">ls</command>`
 - > **Experimental support for link bases**

Paragraphs

- The DocBook paragraph element is `para`.
- For paragraphs with titles, there's `formal-para`.
- In DocBook, `para` can contain “block” elements (tables, figures, procedures, etc.). The `simpara` element can only contain in-lines.

Examples, Figures, Tables, ...

- DocBook has `example`, `figure`, `table`, and `equation`. These elements are “formal” and are expected to have a title.
- If you don't want a title, use `informal-example`, `informalfigure`, `informaltable`, and `informalequation`.
- Tables come in two flavors:
 - > CALS tables and
 - > HTML tables

Graphics

- **Media objects (mediaobject):**
 - > **Images (imageobject),**
 - > **Video (videoobject),**
 - > **Audio (audioobject), and**
 - > **Text (textobject)**

MediaObject Example

```
<mediaobject>
<imageobject>
  <imagedata fileref="emc2.svg" />
</imageobject>
<imageobject>
  <imagedata fileref="emc2.png" format="PNG" />
</imageobject>
<textobject>
  <para>Energy is equal to mass times the speed
of light squared.</para>
</textobject>
<textobject>
  <phrase>E=mc2</phrase>
</textobject>
</mediaobject>
```

Mediaobject Semantics

- The objects inside a `mediaobject` (or `inlinemediobject`) are *alternatives*.
- The processor must choose exactly one.
- Either a `textobject` containing a single phrase or an `alt` element (in DocBook V5.0) can be used for “alternate text” for accessibility.
- A `textobject` may be used as the “long description” for accessibility.

Verbatim environments

- **Program listings:** `programlisting`
- **Screen shots:** `screen` (for command-line interfaces) and `screenshot` (for graphical UIs)
- **Literal layouts:** `literallayout`
- **Addresses:** `address`.

The `programlisting` and `screen` elements are generally monospaced; `literallayout` and `address` are usually in the same font as the body text.

Admonitions

- **note, tip, important, caution, and warning**

Note

This is a note.

```
<note>
```

```
<para>This is a note.</para>
```

```
</note>
```

Lists

- `itemizedlist` and `orderedlist`,
- `variablelist`, and
- `simplelist`

```
<itemizedlist>  
<listitem><para><tag>itemizedlist</tag> and  
<tag>orderedlist</tag>,  
</para></listitem>  
<listitem><para><tag>variablelist</tag>, and  
</para></listitem>  
<listitem><para><tag>simplelist</tag>  
</para></listitem>  
</itemizedlist>
```

Definition or “Variable” Lists

varlistentry Wraps each term (or terms) and the definition.

term Wraps each term, there may be more than one.

listitem Wraps the definition.

```
<variablelist>
```

```
<varlistentry>
```

```
<term><tag>varlistentry</tag></term>
```

```
<listitem><para>Wraps each term (or terms) and...
```

```
</para></listitem>
```

```
</varlistentry>
```

```
...
```

Simple Lists

They're simple...in a somewhat complicated way.

```
<simplelist>  
<member>Apples</member>  
<member>Oranges</member>  
<member>Pears</member>  
<member>Bananas</member>  
<member>Kiwi</member>  
</simplelist>
```

- **Inline, type="inline":
Apples, Oranges, Pears, Bananas, Kiwi**

Simple Lists (Continued)

- `Inline, type="vert", columns="2":`

Apples	Bananas
Oranges	Kiwi
Pears	

- `Inline, type="horiz", columns="3":`

Apples	Oranges	Pears
Bananas	Kiwi	

Special Purpose Markup

- **FAQs**
- **Function and command synopses**
- **Object-oriented programming classes, interfaces, methods, etc.**
- **Sets of messages**
- **EBNF diagrams**
- **MathML and SVG**

FAQs

```
<qandaset>
<qandadiv><title>Sample Questions</title>
<qandaentry>
<question><label>Q1</label>
<para>Question para 1</para>
</question>
<answer><label>A1</label>
<para>Answer para 1</para>
</answer>
</qandaentry>
</qandadiv>
<qandadiv><title>Imponderables</title>
<qandaentry> <!-- Some have no answers -->
<question><para>Why?</para></question>
</qandaentry>
```

FAQs (Continued)

```
<qandaentry>
```

```
<question>
```

```
<para>Why did the chicken cross the road?</para>
```

```
</question>
```

```
<answer>
```

```
<para>To get to the other side</para>
```

```
</answer>
```

```
<answer>
```

```
<para>Some other silly reason I've forgotten.
```

```
</para>
```

```
</answer>
```

```
</qandaentry>
```

```
</qandadiv>
```

```
</qandaset>
```

Function Synopsis

```
<funcsynopsis>
<funcsynopsisinfo>
#include <pwd.h>
</funcsynopsisinfo>
<funcprototype>
  <funcdef>struct passwd *<function>getpwnam</function></funcdef>
  <paramdef>const char * <parameter>name</parameter></paramdef>
</funcprototype>
<funcprototype>
  <funcdef>struct passwd *<function>getpwuid</function></funcdef>
  <paramdef>uid_t <parameter>uid</parameter></paramdef>
</funcprototype>
</funcsynopsis>
```

"Hierarchy" Elements

- **Set and Book**
- **Part and Reference**
- **Preface, Chapter, Appendix, Bibliography, Glossary, Index**
- **Article**
- **Section, Sect1...Sect5, SimpleSect**
- **RefEntry**
- **RefSect1...RefSect3**

A DocBook 4 Book

```
<book>
<bookinfo>
  <title>An Example Book</title>
  <author>
    <firstname>Norman</firstname>
    <surname>Walsh</surname>
  </author>
  <copyright>
    <year>2004</year>
    <holder>Sun Microsystems, Inc.</holder>
  </copyright>
  <contractnum>1234</contractnum>
  <contractsponsor>Our Favorite Sponsor
</contractsponsor>
</bookinfo>
```

A DocBook 4 Book (Continued)

```
<preface><title>Introduction</title>
<para>...</para>
</preface>
<chapter><title>The First Chapter</title>
<para>...</para>
</chapter>
<!-- ... -->
<appendix><title>An Appendix</title>
<para>...</para>
</appendix>
</book>
```

A DocBook 5 Book

```
<book xmlns="http://docbook.org/ns/docbook">
<info>
  <title>An Example Book</title>
  <author>
    <personname>
      <firstname>Norman</firstname>
      <surname>Walsh</surname>
    </personname>
  </author>
  <copyright>
    <year>2004</year>
    <holder>Sun Microsystems, Inc.</holder>
  </copyright>
  ...
</info>...
```

An Article

```
<article>
<articleinfo>
  <title>An Example Article</title>
  <author>
    <firstname>Norman</firstname>
    <surname>Walsh</surname>
  </author>
  <copyright>
    <year>2004</year>
    <holder>Sun Microsystems, Inc.</holder>
  </copyright>
</articleinfo>
<section><title>A Section</title>
<para>...</para>
</section>
```

An Article (Continued)

```
<appendix><title>An Appendix</title>  
<para>...</para>  
</appendix>  
</article>
```

Reference Pages

```
<refentry>  
<refmeta>  
<refentrytitle>getpwnam</refentrytitle>  
<manvolnum>3</manvolnum>  
</refmeta>
```

```
<refnamediv>  
<refname>getpwnam</refname>  
<refname>getpwuid</refname>  
<refpurpose>get password file entry</refpurpose>  
</refnamediv>
```

```
<refsynopsisdiv><title>Synopsis</title>  
<synopsis>  
#include <pwd.h>
```

Reference Pages (Continued)

```
#include <sys/types.h>
```

```
struct passwd *getpwnam(const char * name);
```

```
struct passwd *getpwuid(uid_t uid);
```

```
</synopsis>
```

```
</refsynopsisdiv>
```

```
<refsect1><title>Description</title>
```

```
<para>The <function>getpwnam</function> function  
returns a pointer to a structure containing the  
broken out fields of a line from
```

```
<filename>/etc/passwd</filename> for
```

```
the entry that matches the user name
```

```
<parameter>name</parameter>.
```

Reference Pages (Continued)

```
</para>  
<!--...-->  
</refsect1>  
<!--...-->  
</refentry>
```

Publishing

**Getting from here to there. DocBook goes in,
what comes out?**

DocBook to ...

XSLT

DocBook XSL Stylesheets

DocBook to HTML

Transforming to HTML

DocBook to PDF

Transforming to PDF

DocBook to ...

- **HTML or XHTML**
- **XSL Formatting Objects (then to PDF)**
- **HTML Help**
- **Java Help**
- **Unix “man” pages**
- **TeX**
- **PDF and PostScript**
- **RTF**

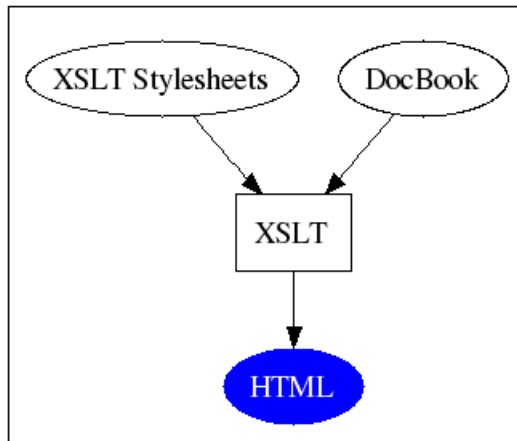
XSLT

- **XSL Transformations, part of the Extensible Style Language from the W3C**
- **Many processors available (Saxon, Xalan, xsltproc, ...)**
- **Only for XML; uses XML syntax and XPath as an expression language.**
- **Produces HTML, Formatting Objects, XML**
- **Formatting Objects can produce PDF (via PassiveTeX, FOP, RenderX, etc.)**

DocBook XSL Stylesheets

- **HTML or XHTML**
- **XSL Formatting Objects (then to PDF)**
- **HTML Help**
- **Java Help**
- **Unix “man” pages**

DocBook to HTML



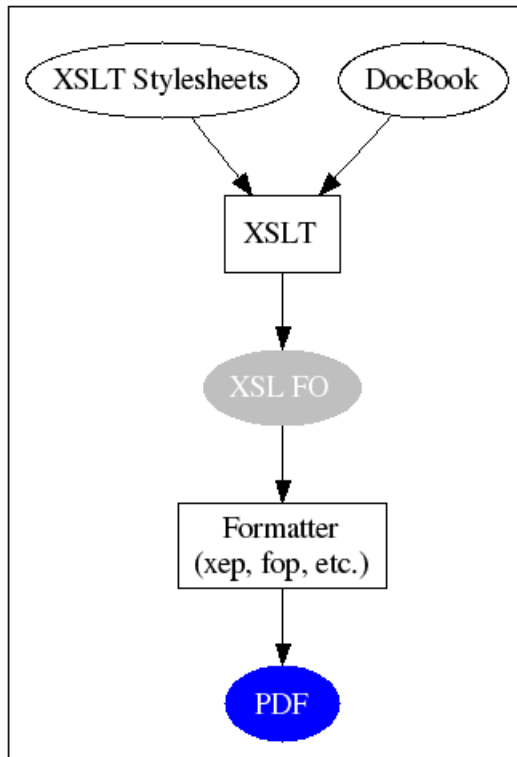
Publishing HTML

Transforming to HTML

```
java com.icl.saxon.StyleSheet -o example.html \  
    example.xml html/docbook.xsl
```

- **Processes example.xml**
- **With html/docbook.xsl**
- **Produces example.html**

DocBook to PDF



Publishing HTML

Transforming to PDF

```
java com.icl.saxon.StyleSheet -o example.fo \  
    example.xml fo/docbook.xsl  
java com.renderx.xep.XSLDriver example.fo
```

- **Saxon processes example.xml**
- **With fo/docbook.xsl**
- **And produces example.fo;**
- **XEP formats example.fo**
- **And produces example.pdf**

Stylesheet Customization

XSL Customizations

Easy: Setting Parameters

Easy Parameters

Pretty Easy: A Customization Layer

Customization Skeleton

Customization Example 1

Customization Example 2

A Little Harder

...

XSL Customizations

- **Easy**
- **Pretty easy**
- **A little harder**
- **SMOP (Simple Matter of Programming)**

Easy: Setting Parameters

```
java com.icl.saxon.StyleSheet -o example2.html \  
    example.xml html/docbook.xsl admon.graphics=1
```

- **Processes `example.xml`**
- **With `html/docbook.xsl` with `admon.graphics=1`**
- **Produces `example2.html`**

With the FO stylesheet, `example2.pdf` can be produced in the analogous way.

Easy Parameters

- **Admonition graphics**
- **Line numbering and other verbatim styles**
- **Automatic labelling of chapters, etc.**
- **CSS to use in the HTML result**
- **Page size (Letter/A4/etc.) and margins**
- **Number of columns on a page**
- **Page Header and footer styles**
- **Font families**
- **Profiling characteristics**

Pretty Easy: A Customization Layer

- **One XSLT stylesheet can “import” another.**
- **Useful if you want to set several parameters.**
- **Necessary if you want to set parameters that aren't simple string or numeric values.**
- **A step towards more aggressive customizations.**

Customization Skeleton

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

    <xsl:import href="/path/to/docbook/xsl/html/docbook.xsl"/>

    <!-- your stuff goes here -->
    <xsl:param name="admon.graphics" select="1"/>

</xsl:stylesheet>
```

Customization Example 1

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

  <xsl:import href="/path/to/docbook/xsl/html/docbook.xsl"/>

  <xsl:param name="admon.graphics" select="1"/>
  <xsl:param name="html.stylesheet">my.css</xsl:param>

  <xsl:attribute-set name="shade.verbatim.style">
    <xsl:attribute name="border">0</xsl:attribute>
    <xsl:attribute name="bgcolor">#E0E0E0</xsl:attribute>
  </xsl:attribute-set>
</xsl:stylesheet>
```

Customization Example 2

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
  <xsl:import href="/path/to/docbook/xsl/html/docbook.xsl"/>

  <xsl:param name="admon.graphics" select="1"/>
  <xsl:param name="html.stylesheet">my.css</xsl:param>

  <xsl:attribute-set name="shade.verbatim.style">
    <xsl:attribute name="border">0</xsl:attribute>
    <xsl:attribute name="bgcolor">#E0E0E0</xsl:attribute>
  </xsl:attribute-set>

  <xsl:template name="user.header.content">
    <div align="center">
      <h1>Your Corporate Header</h1>
    </div>
  </xsl:template>
</xsl:stylesheet>
```

A Little Harder

- **Changing title pages**
- **Replacing individual templates**

Changing Title Pages

- **Copy the title page templates file**
- **Modify the template(s) you want to change**
- **Rebuild the templates stylesheet**
- **Incorporate it into a customization layer**
- **Style your document with the new stylesheet**

Updating the Title Page Template

- Copy `.../html/titlepage.templates.xml` to `tpage.xsl`
- Edit the “book” template:

```
<t:titlepage t:element="book" t:wrapper="div"
            class="titlepage">
  <t:titlepage-content t:side="recto">
    <title/>
    <subtitle/>
    <corpauthor/>
    <authorgroup/>
    <author/>
    <othercredit/>
    <releaseinfo/>
```

Updating... (Continued)

```
<copyright />  
<legalnotice />  
<pubdate />  
<revision />  
<revhistory />  
<abstract />  
<contractnum />  
<contractsponsor />  
</t:titlepage-content >
```

- **Transform tpage.xml to tpage.xsl (with .../templates/titlepage.xsl)**

Incorporate it into a customization layer

- **Use the import/include two-step:**

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <xsl:import href="/sourceforge/docbook/xsl/html/docbook.xsl"/>
  <xsl:include href="tpage.xsl"/>

</xsl:stylesheet>
```

- **Transform with that stylesheet.**

Write a template for contractnum

- Improve the presentation with generated text:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
  <xsl:import href="/sourceforge/docbook/xsl/html/docbook.xsl"/>
  <xsl:include href="tpage.xsl"/>

  <xsl:template match="contractnum" mode="titlepage.mode">
    <div class="contract">
      <xsl:text>This work was funded by Contract </xsl:text>
      <xsl:apply-templates mode="titlepage.mode"/>
    </div>
  </xsl:template>
</xsl:stylesheet>
```

Uh, but what about internationalization?

- Putting generated text directly in templates isn't very friendly to internationalization.
- The DocBook stylesheets *support 59 languages* out of the box: Afrikaans, Albanian, Amharic, Arabic, Azerbaijani, Bangla, Basque, Bosnian, Bulgarian, Catalan, Chinese (Simplified), Chinese (Traditional), Croatian, Czech, Danish, Dutch, English, Estonian, Farsi, Finnish, French, German, Greek, Gujarati, Hebrew, Hindi, Hungarian, Indonesian, Irish, Italian, Japanese, Kan-

Uh, but what about I18N? (Continued)

nada, Korean, Latin, Lithuanian, Mongolian, Norwegian, Nynorsk, Oriya, Polish, Portuguese (Brazil), Portuguese, Punjabi, Romanian, Russian, Serbian in Cyrillic script, Serbian in Latin script, Slovak, Slovenian, Spanish, Swedish, Tagalog, Tamil, Thai, Turkish, Ukrainian, Vietnamese, Welsh, and Xhosa

- **It's not difficult to leverage this flexibility in your stylesheets.**

Write an I18N-friendly template

- **Internationalize the generated text:**

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
  <xsl:import href="/sourceforge/docbook/xsl/html/docbook.xsl"/>
  <xsl:include href="tpage.xsl"/>
  <xsl:template match="contractnum" mode="titlepage.mode">
    <div class="contract">
      <xsl:apply-templates select="." mode="object.title.markup"/>
    </div>
  </xsl:template>
</xsl:stylesheet>
```

- **Easy, right? But where does the text come from?**

I18N: Put it in the Stylesheet

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

  <xsl:import href="/sourceforge/docbook/xsl/html/docbook.xsl"/>
  <xsl:include href="tpage.xsl"/>

  <xsl:param name="local.l10n.xml" select="document('')"/>

  <l:i18n>
    <l:l10n language="en">
      <l:context name="title">
<l:template name="contractnum"
  text="This work was funded by Contract %n"/>
      </l:context>
    </l:l10n>
    <l:l10n language="de">
      <l:context name="title">
<l:template name="contractnum"
  text="Diese Arbeit wurde durch eine freundliche
```

I18N: Put it in the Stylesheet (Continued)

```
Spende unter Vertrag %n unterstützt"/>
  </l:context>
</l:l10n>
</l:i18n>

<xsl:template match="contractnum" mode="titlepage.mode">
  <div class="contract">
    <xsl:apply-templates select="." mode="object.title.markup"/>
  </div>
</xsl:template>

</xsl:stylesheet>
```

I18N: Handle the label

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

  <xsl:import href="/sourceforge/docbook/xsl/html/docbook.xsl"/>
  <xsl:include href="tpage.xsl"/>

  <xsl:param name="local.l10n.xml" select="document('')"/>

  <l:i18n>
    <l:l10n language="en">
      <l:context name="title">
<l:template name="contractnum"
  text="This work was funded by Contract %n"/>
      </l:context>
    </l:l10n>
    <l:l10n language="de">
      <l:context name="title">
<l:template name="contractnum"
  text="Diese Arbeit wurde durch eine freundliche
```

I18N: Handle the label (Continued)

```
Spende unter Vertrag %n unterstützt"/>
  </l:context>
</l:l10n>
</l:i18n>

<xsl:template match="contractnum" mode="titlepage.mode">
  <div class="contract">
    <xsl:apply-templates select="." mode="object.title.markup"/>
  </div>
</xsl:template>

<xsl:template match="contractnum" mode="label.markup">
  <xsl:apply-templates/>
</xsl:template>

</xsl:stylesheet>
```

S.M.O.P. (Simple Matter of Programming)

- **Rewrite entire sections of the stylesheets**
- **Support new markup constructs**

Schema Customization

Subsetting and extending DocBook.

The Role Attribute

Subsets

Extensions

DTD Customization Layer

RELAX NG Customization Layer

Restricting Role on Emphasis (DTD)

Restricting Role on Emphasis (RELAX NG)

Removing Procedures from the DTD

...

The Role Attribute

- All elements have a `role` attribute
- Stylesheets can key off of role values
- `<literal>` vs. `<literal role="widget-Spec">`
- DocBook *never* specifies role values

Subsets

- **Subsets constrain DocBook**
- **All documents that conform to the subset also conform to the full schema**
- **Enumeration of attribute values**
- **Removing elements**
- **Constraining content models**
- **Doesn't usually require stylesheet/tool customization**

Extensions

- **Extensions add new elements or attributes to DocBook**
- **Documents that conform to the extension may not conform to DocBook**
- **Adding new attributes or elements**
- **Extending content models**
- **Extensions can also remove elements**
- **Always requires stylesheet/tool customization**

DTD Customization Layer

In general, DTD customization layers look like this:

```
<!-- Turn off markup -->

<!-- Redefine parameter entities -->

<!-- include the base DTD -->
<!ENTITY % docbook PUBLIC "-//OASIS//DTD DocBook XML V4.3//EN"
          "http://www.oasis-open.org/docbook/xml/4.3/docbookx.dtd"
>
%docbook;

<!-- Redefine elements/attributes -->
```

RELAX NG Customization Layer

In general, RELAX NG customization layers look like this:

```
namespace db = "http://docbook.org/ns/docbook"  
  
include "/path/to/docbook.rnc" {  
    # redefine patterns  
}  
  
# define new patterns
```

Restricting Role on Emphasis (DTD)

```
<!ENTITY % emphasis.role.attrib
          role      (normal|emphasis)      #IMPLIED
>

<!ENTITY % docbook PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
          "http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd"
>

%docbook;
```

Restricting Role on Emphasis (RELAX NG)

```
namespace db = "http://docbook.org/ns/docbook"
default namespace = "http://docbook.org/ns/docbook"

include "docbook.rnc" {
  db.emphasis.role.attribute =
    attribute role { "normal"|"emphasis" }
}
```

Removing Procedures from the DTD

```
<!-- DocBook XML V4.3 No Procedures Subset -->
```

```
<!ENTITY % ebnf.block.hook "">
<!ENTITY % local.compound.class "">
<!ENTITY % compound.class
  "msgset|sidebar|qandaset
  %ebnf.block.hook;
  %local.compound.class;">

<!ENTITY % procedure.content.module "IGNORE">
<!ENTITY % task.content.module "IGNORE">
<!ENTITY % sidebar.element "IGNORE">
<!ENTITY % qandaset.element "IGNORE">
<!ENTITY % qandadiv.element "IGNORE">
<!ENTITY % question.element "IGNORE">
<!ENTITY % answer.element "IGNORE">
<!ENTITY % revdescription.element "IGNORE">
<!ENTITY % caution.element "IGNORE">
<!ENTITY % important.element "IGNORE">
<!ENTITY % note.element "IGNORE">
```

Removing Procedures... (Continued)

```
<!ENTITY % tip.element "IGNORE">
```

```
<!ENTITY % warning.element "IGNORE">
```

```
<!ENTITY % docbook.dtd PUBLIC "-//OASIS//DTD DocBook XML V4.3//EN"  
          "http://docbook.org/xml/4.3/docbookx.dtd">  
%docbook.dtd;
```

```
<!ENTITY % my.sidebar.mix  
  "%list.class; |%admon.class;  
  |%linespecific.class; |%synop.class;  
  |%para.class; |%informal.class;  
  |%formal.class;  
  |%genobj.class;  
  |%ndxterm.class;          |beginpage  
  %local.sidebar.mix;">
```

```
<!ELEMENT sidebar (sidebarinfo?,  
                  (%formalobject.title.content;)?,  
                  (%my.sidebar.mix;)+)>
```

Removing Procedures... (Continued)

```
<!ENTITY % my.qandaset.mix
  "%list.class;          |%admon.class;
  |%linespecific.class; |%synop.class;
  |%para.class;  |%informal.class;
  |%formal.class;
  |%genobj.class;
  |%ndxterm.class;
  %local.qandaset.mix;">

<!ELEMENT qandaset (blockinfo?, (%formalobject.title.content;)?,
  (%my.qandaset.mix;)*,
  (qandadiv+ |qandaentry+))>

<!ELEMENT qandadiv (blockinfo?, (%formalobject.title.content;)?,
  (%my.qandaset.mix;)*,
  (qandadiv+ |qandaentry+))>

<!ELEMENT question (label?, (%my.qandaset.mix;)+)>

<!ELEMENT answer (label?, (%my.qandaset.mix;)*, qandaentry*)>
```

Removing Procedures... (Continued)

```
<!ENTITY % my.revdescription.mix
  "%list.class; |%admon.class;
  |%linespecific.class; |%synop.class;
  |%para.class; |%informal.class;
  |%formal.class;
  |%genobj.class;
  |%ndxterm.class;
  %local.revdescription.mix;">
```

```
<!ELEMENT revdescription ((%my.revdescription.mix;)+)>
```

```
<!ENTITY % my.admon.mix
  "%list.class;
  |%linespecific.class; |%synop.class;
  |%para.class; |%informal.class;
  |%formal.class; |sidebar
  |anchor|bridgehead|remark
  |%ndxterm.class; |beginpage
  %local.admon.mix;">
```

Removing Procedures... (Continued)

```
<!ELEMENT caution (title?, (%my.admon.mix;)+)  
                %admon.exclusion;>
```

```
<!ELEMENT important (title?, (%my.admon.mix;)+)  
                %admon.exclusion;>
```

```
<!ELEMENT note (title?, (%my.admon.mix;)+)  
                %admon.exclusion;>
```

```
<!ELEMENT tip (title?, (%my.admon.mix;)+)  
                %admon.exclusion;>
```

```
<!ELEMENT warning (title?, (%my.admon.mix;)+)  
                %admon.exclusion;>
```

Removing Procedures (RELAX NG)

```
namespace db = "http://docbook.org/ns/docbook"  
default namespace = "http://docbook.org/ns/docbook"  
  
include "docbook.rnc" {  
    db.procedure = notAllowed  
}
```

Derived Schemas

A few examples of successful and interesting subsets and extensions.

SolBook: The Sun Documentation DTD

Simplified DocBook

Websites

Slides

DITA?

SolBook: The Sun Documentation DTD

- **The source for docs.sun.com**
- **Restrictions to aid authoring and enforce style**

Simplified DocBook

- **Only supports articles**
- **Far fewer block elements**
- **Far fewer inlines**
- **About 100 tags vs about 400**

Websites

- **Uses DocBook information pool**
- **Replaces most of the hierarchy**
- **A website is a tree of nested web pages**
- **Stylesheets support both flat and tabular, two-column navigation**
- **See nwalsh.com for an example.**

Slides

- **Based on simplified DocBook**
- **Replaces article with a set of slides**
- **Slides can be divided into sections**
- **Stylesheets support HTML and PDF**
- **This presentation is generated from Slides source**

DITA?

- **Darwin Information Typing Architecture**
 - > **A topic-oriented authoring paradigm.**
 - > **A cross-referencing scheme that's more practical than XML's flat ID space.**
 - > **Transclusion (aka SGML's conref, reinvented).**
 - > **An extensibility model based on “specialization”**

It can all be done in DocBook with a little customization.

Conclusions

Resources

Q&A

Resources

- **DocBook Home Page**
- ***DocBook: The Definitive Guide***
Norman Walsh and Leonard Mueller
O'Reilly & Associates, Inc.
1st Edition October 1999
ISBN: 1-56592-580-7
- ***DocBook XSL: The Complete Guide***
Bob Stayton

Resources (Continued)

**Sagehill Enterprises
3rd Edition February 2005**

- **DocBook resources at docbook.sf.net. (XSL Stylesheets, Customization Layers, etc.)**

DocBook V5.0: The Gory Details

A detailed look at the changes in DocBook V5.0.

What's Changed

What's Changed

- The `articleinfo`, `bookinfo`, etc. elements have all been renamed `info`.
- The `collabname`, `corpauthor`, `corpcredit`, and `corpname` elements have all been replaced by `orgname` and updated content models for `author`, `editor`, and `othercredit`.
- The `graphic`, `graphicco`, `inlinegraphic`, and `mediaobjectco` elements have been

What's Changed (Continued)

- removed in favor of `mediaobject` and `inlinemediaobject`.
- The `isbn`, `issn`, and `pubsnumber` elements have all been replaced by `biblioid`.
- The `uLink` tag has been replaced by ubiquitous linking (`linkend` or `xlink:href` on `link`).
- The `sgmltag` element has been replaced by `tag`.

What's Changed (Continued)

- **The following elements have been removed: `action`, `beginpage`, `highlights`, `interface`, `invpartnumber`, `medialabel`, `mod-espec`, `structfield`, `structname`.**
- **HTML and CALS tables are now interpreted correctly**