Getting Started with Docbook

Tutorial

0.1, 14 Feb 2009

ZILOGIC SYSTEMS

Table of Contents

List of Listings

Chapter 1. XML Introduction

- XML - eXtensible Markup Language
- Markup Language - a language used to annotate text to make explicit the interpretation of the text, useful for computer/human processing.
- It is not very clear from the following text, what the text represents.

```
Abin    9444107528      23783120
Sarma   9444101103      23718291
Sameer  9444193010      20403030
```

- The same data marked up using XML.

```
<abook>
<entry>
        <name> Abin </name>
        <mobile> 9444107528 </mobile>
        <landl> 23783120 </landl>
</entry>
<entry>
        <name> Sarma </name>
        <mobile> 9444101103 </mobile>
        <landl> 23718291 </landl>
</entry>
<entry>
        <name> Sameer </name>
        <mobile> 9444193010 </mobile>
        <landl> 20403030 </landl>
</entry>
</abook>
```

- The annotations in angle brackets are called tags.
- Markup Languages were primarily developed for document and database publishing.
- XML is only a set of rules that says how a markup language should look like.
- The advantage is that the same XML parser can be used for all these XML based markup languages.
- As such XML does not define any tags by itself. The user can define a set of tags for an application and use it in his XML document.
- This is why XML is called "Extensible" Markup Language.
- Examples of XML applications are:
  - XML RPC
  - RSS
  - XHTML
  - Docbook
- For each application there is Document Type Defintion (DTD), that describes
  - What tags are available
  - How these tags can be nested

- What attributes are available for each tag

Chapter 2. Docbook Overview

- Docbook was originally intended for technical documentation related to computer hardware and software.
- Docbook is used in several key open source projects including the GNOME, KDE, FreeBSD and the Linux kernel.
- Docbook is NOT used to describe how the content should look like.
- Docbook is used to describe the meaning of the content. For example, rather than explaining how a source code listing is to be visually formatted, Docbook simply says that a particular block of text is a source code listing. It is up to an external processing tool to decide what font is to be used, whether the code is to be syntax highlighted, etc.
- A set of tools is used to convert Docbook, to a presentation format like HTML, RTF, PDF, Man pages, voice, etc.
- The transformation is done using XSLT. XSLT is a language used to describe how one XML document is to be converted into another XML or human-readable documents.
- An XSLT processor reads an XSLT stylesheet and transforms the input XML file. Examples of XSLT processors are
  - xsltproc — from the GNOME project
  - xalan — from the Apache XML project
  - saxon — by Michael Kay
- A bunch of stylesheets has been made available by the Docbook XSL project, maintained by Norman Walsh.

```
                Docbook XSL
                Stylesheets
                     |
                     v
Docbook XML --> xsltproc --> HTML Output
```

- Docbook advantages:
  - Content is separated out from the presentation. Technical writers can focus on the content, and the appearance will be taken care by the stylesheets.
  - Organization wide uniformity of document appearance — title pages, headers, footers, typography, …
  - Old documents can be easily re-generated to reflect changes in the stylesheet.
  - Multiple output format for print, web, voice, etc.

Chapter 3. Docbook XML

Listing 3.1. Hello Docbook

```
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
  "/usr/share/xml/docbook/schema/dtd/4.4/docbookx.dtd">

<article>

<articleinfo>
<title>Docbook Demo</title>
</articleinfo>

<section>
<title>Hello, world</title>

<para>This is my first DocBook file.</para>
</section>
</article>
```

- To transform using `xsltproc`

```
xsltproc -o <output-file> <stylesheet> <docbook-xml>
```

- Example invocation to transform to HTML.

```
$ STYLESHEET=/usr/share/xml/docbook/stylesheet/nwalsh/html/docbook.xsl
$ xsltproc -o hello.html $STYLESHEET hello.xml
```

- Document Type Declaration, is an instruction that associates the XML file with a DTD.

```
<!DOCTYPE ❶
 article  ❷
 PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN" ❸
 "/usr/share/xml/docbook/schema/dtd/4.4/docbookx.dtd" ❹
>
```

❶ Tells the processor that we are about to choose the DTD.
❷ Specifies the root element — book or article
❸ Specifies the DTD to use, with its Formal Public Indentifier
❹ Path of the DTD on the local system.

- The `<articleinfo>` tag is used to provide meta information about the document like the title, author, revision, date, etc …
- `<section>` starts a new section, and the section title is specified by the containing `<title>` tag.
- Paragraphs within a section are enclosed within `<para>`.
- Itemized list

```
<itemizedlist>
```

```
<listitem><para> Fruits </para></listitem>
<listitem><para> Vegetables </para></listitem>
<listitem><para> Spices </para></listitem>

</itemizedlist>
```

- 3 column table

```
<table>
<title>Mouse Mileage</title>
<tgroup cols="3">
<thead>
        <row>
                <entry>Month</entry>
                <entry>Week</entry>
                <entry>Feet Traveled</entry>
        </row>
</thead>
<tbody>
        <row>
                <entry>August</entry>
                <entry>1</entry>
                <entry>987</entry>
        </row>
        <row>
                <entry>August</entry>
                <entry>2</entry>
                <entry>657</entry>
        </row>
        <row>
                <entry>August</entry>
                <entry>3</entry>
                <entry>109</entry>
        </row>
</tbody>
</tgroup>
</table>
```

Chapter 4. Asciidoc

- Docbook is a way great to capture the content of a document.
- But Docbook has its drawbacks.
  - Steep learning curve
  - 400+ tags, compared to HTML which has 60+ tags
  - Hard to convince other colleagues to switch over
- Need a tool that has the power of Docbook and yet simple to learn and use.
- Enter Asciidoc, convert wiki-markup text to Docbook!

```
Docbook Demo
============
Vijay Kumar <vijaykumar@bravegnu.org>

Hello World
-----------

This is my first Docbook file.

List Example
------------

  * Fruits
    - Apple
    - Orange
    - Mango
  * Vegetables
  * Spices

Table Example
-------------

.Mouse Mileage
`-------`-------`------------
Month   Week    Feet Traveled
-----------------------------
August  1       987
August  2       657
August  3       109
-----------------------------
```

- To convert asciidoc files to Docbook.

```
asciidoc -b docbook <asciidoc-file>
```

Chapter 5. XSLT

- The stylesheets provided by the docbook-xsl project were used for the generation of HTML from Docbook source.
- Usually people would want to customize the stylesheets, so that the output fits their organization's document template — logo in header, section numbering, etc.
- docbook-xsl allows itself to be customized through a number of parameters.
- Before we look into docbook-xsl customization, we will take a peek into how XSLT works.

```
<weather type="Current conditions">
    <temp>76</temp>
    <wind>5</wind>
</weather>
```

- XSLT is used to specify rules to translate the source XML tree into result tree.
- Each rule is called a template in XSLT terminology.
- Each rule has a match condition associated with it, and is executed only when the match condition is triggered.
- The XSLT processor walks through the tree and when a match condition for any of the rule occurs, the rule is executed.
- A very simple stylesheet

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="weather">
<p>Got weather tag!</p>
</xsl:template>

<xsl:template match="temp">
<p>Got temp tag!</p>
</xsl:template>

<xsl:template match="wind">
<p>Got wind tag!</p>
</xsl:template>

</xsl:stylesheet>
```

- The need for XML namespaces.
- Processing recursively using <xsl:apply-templates/>

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="weather">
<h1>Current Weather</h1>
<ul>
<xsl:apply-templates/>
</ul>
</xsl:template>

<xsl:template match="temp">
<li>The temperature is <xsl:apply-templates/> degree Farenheit.</li>
</xsl:template>
```

```
<xsl:template match="wind">
<li>The wind speed is <xsl:apply-templates/> kmph.</li>
</xsl:template>

</xsl:stylesheet>
```

- The behaviour of templates can be customized using parameters.
- To define a parameter, the `<xsl:param>` can be used. The `name` attribute is used to specify the parameter name, and the content of the tag is the parameter's default value.
- To retreive the value of a parameter the `<xsl:value-of>` tag can be used.

```
<xsl:param name="title">Current Weather</xsl:param>

<xsl:template match="weather">
<h1><xsl:value-of select="$title"/></h1>
<ul>
<xsl:apply-templates/>
</ul>
</xsl:template>
```

- The default value for the parameter can be changed by passing the parameter value as an argument to xsltproc.

```
$ xsltproc --stringparam title "My Weather" stylesheet.xsl document.xml
```

- The stylesheets from the docbook-xsl project are highly customizable through parameters.
- Example stylesheet parameters.

section.autolabel	Specifies whether section titles are numbered.
formal.title.placement	Specifies the placement of titles for listings, tables and figures.

- Instead of specifying each parameter individually on the command line, a stylesheet can be created that uses the templates from the docbook-xsl project and overrides the parameter defintions.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:import href="/usr/share/xml/docbook/stylesheet/nwalsh/html/docbook.xsl"/>

<xsl:param name="section.autolabel">1</xsl:param>
<xsl:param name="formal.title.placement">
example after
table after
</xsl:param>

</xsl:stylesheet>
```

Chapter 6. XSL-FO

- XSLT is capable for tranforming one XML tree into another tree or a simple text format.
- XSLT cannot be used to directly generate PDFs, for one PDFs are not XML files.
- Enter XSL-FO, an XML application that describes how data will be presented to the reader — font size, colors, line spacing, page margins, headers and footers.
- XSL-FO can be later converted to PDF using FO processor. One such FO processor is Apache FOP.

```
$ fop -pdf out.pdf in.fo
```

Chapter 7. Toolchain Installation in Debian

The following packages should be installed.

docbook-xml	standard XML documentation system, for software and systems
docbook-xsl	stylesheets for processing DocBook XML files to various output
asciidoc	Highly configurable text format for writing documentation
xsltproc	XSLT command line processor
fop	XML to PDF Translator