OASIS Specification Publishing in DocBook XML

Working Draft 0.4, 08 February 2006

Document identifier:

oasis-specification-0.4

Locations:

Persistent version: `http://docs.oasis-open.org/templates/`
Current version: `http://docs.oasis-open.org/templates/DocBook/spec-0.4/oasis-specification-0.4.html`

Author:

G. Ken Holman, Crane Softwrights Ltd. <gkholman@CraneSoftwrights.com>

Abstract:

This Working Draft describes an environment for writing and publishing an OASIS specification using DocBook XML. It is an internal OASIS support document and not the basis of an OASIS specification in and of itself.

Related Work:

This methodology supersedes guidelines for the XML-based authoring and publishing of OASIS specifications described in http://www.oasis-open.org/spectools/docs/wd-spectools-docbook-template-03.html/ and supported by the stylesheets in http://www.oasis-open.org/spectools/stylesheets/.

Status:

This is a work in progress and does not at this time represent the consensus of any particular OASIS Technical Committee. There are no plans to make this a formal Committee Specification as it is merely an internal document made available to committee members to support the publishing process.

Please send comments on this document to G. Ken Holman at <gkholman@CraneSoftwrights.com>.

Notices:

Copyright © OASIS Open 2006. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS,

except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Table of Contents

Appendixes

1. Introduction

This document details an environment and methodology for writing an OASIS specification document using XML markup, and publishing the resulting document to HTML and printed results conforming to OASIS layout conventions.

While this has been prepared before, a new version of this environment and methodology is required to accommodate (a) the inclusion of revised OASIS specification metadata; and (b) the changes in the available stylesheets for the XML authoring environments developed years ago for OASIS specifications.

The stylesheets for this version of this environment will remain static while the prose methodology documents and samples may be revised from time to time. Should it be necessary to revise the stylesheets, this version of the environment will left unchanged on the OASIS web site and a new version of the stylesheets and prose documents will be started. Check the persistent location noted above for any revisions to this environment.

These stylesheets use XSLT [XSLT] as the transformation expression language to produce the final HTML result. XSLT is also used to produce the intermediate XSL-FO [XSL-FO] pagination semantics, which in turn are processed to produce printable results.

2. Authoring in XML

An important objective of using XML markup when writing content is to separate what you are writing from how it is formatted. Moreover, describing the individual components of your writing uniquely can allow machine processing of your content. Such machine processing can identify constructs and process them individually as required for the processed result.

The vocabulary of XML markup is the level of granularity used to identify constituent information items, and the collection of element and attribute labels applied to the granules.

Applying styling to documents is an example of machine processing. The processed result is a formatted representation of your document. When many authors use the same vocabulary, or a given author creates many documents with the same vocabulary, a single set of processes will produce consistently formatted results across the document set.

This process is analogous to using styles found in most desktop publishing applications, however by removing from the author the ability to inject arbitrary formatting of information items in their content, two benefits are realized: (a) the author no longer needs to think about formatting, only about appropriately labeling the information items in the content; and (b) authors cannot inadvertently format components of a document with incorrect or inconsistent results.

Many writers do, however, prefer the use of their favorite desktop publishing applications and OASIS has posted at `http://docs.oasis-open.org/templates/` a number of templates for widely-adopted word processing programs to be used in the creation of OASIS specifications. Users must be diligent in the use of styles in order to ensure their work conforms to the presentation conventions requested of OASIS. There is an obligation incumbent on the writer to verify their work has not violated the requested conventions, and there are no automated tools with which to validate the constraints have been respected.

When creating OASIS specifications using XML the burden of formatting is placed on the stylesheets, not on the writer. The obligation of the writer is only to be conformant to the document model for which the stylesheets have been designed, and there are automated validation tools with which the writer can validate the constraints have not been respected.

Any resulting violations to the formatting conventions can usually be ascribed to the stylesheets and, when repaired, the authored content usually does not need to change (barring any need to distinguish in the XML an ambiguity that is then being distinguished in the stylesheets).

2.1. DocBook vocabulary

This environment and methodology is based on using the `<article>` document type of the DocBook [DocBook] vocabulary for authoring the content in XML markup. These explanatory documents do not attempt to teach the DocBook vocabulary, as there are many online and printed references and manuals on the use of DocBook.

Note that there is a layer of additional constraints imposed on top of the DocBook metadata constructs in order to appropriately identify the constituent metadata components of the OASIS requirements for specifications. No validation constraints for this additional metadata layer are included in this environment, so it is the responsibility of the writer to visually confirm these needs have been satisfied in the rendered results. Missing components are typically the result of the writer not having correctly used the metadata indicators within the standard DocBook constructs.

The additional constraints are illustrated and documented in a pro forma template included with these materials.

2.2. Pro forma template

This environment includes a pro forma template of a DocBook XML instance with placeholders for all the OASIS specification metadata found in the word processing templates. The XML template has one of every item of metadata and can be used as a guideline when creating the metadata for a new document. The prose of the document includes a detailed narrative of the declaration and use of each of the metadata components.

The template also includes some limited guidelines to the use of the DocBook vocabulary that have been carried over from previous versions of the documentation. With time it is hoped to embellish more in this section for the benefit of the reader.

The pro forma template is included as part of the package of this environment and if the document you are reading is part of a completely installed collection then it can be found at `template/wd-spectools-docbook-template-0.4.html`.

2.3. Stylesheet association

Not included in the final published XML source of OASIS specifications are the stylesheet association [xml-assoc] processing instructions that are very handy conveniences to use during the authoring process. This methodology mandates their removal from the final published documents, but encourages their use when writing in order to streamline the writer's review of the formatted content.

Stylesheet association is not widely employed in the general XML community, typically due to (a) limited awareness by writers of their benefit; and (b) limited conforming support in web browsers.

An XSLT stylesheet association processing instruction is structured with two pseudo attributes as follows, indicating the type of the stylesheet being engaged and the location of the stylesheet file producing the HTML rendition:

```
<?xml-stylesheet type="text/xsl" href="place-URL-here"?>
```

The benefits of having embedded a stylesheet association processing instruction at the top of one's XML document are realized by opening the XML document being written in a web browser that (a) is aware of the processing instruction; (b) has a conforming XML processor with which to process the stylesheet files; and (c) has a conforming XSLT processor with which to engage the stylesheet files. The act of opening the XML document in the browser engages the HTML stylesheet against the XML content producing the HTML rendition on the browser canvas. At any time during the writing process, merely refreshing the web browser canvas re-engages the stylesheet producing the rendition of the revised content. This removes the necessity to engage the standalone invocation of the stylesheet environment to produce a reviewable rendition.

For example, this document was authored in an environment where the locally-installed offline version of the publishing environment is available in the local directory `p:/oasis/spec-0.4/`, thus allowing the following stylesheet association processing instruction placed at the top of the XML file before the document type declaration to render the document in an XSLT-aware web browser:

```
<?xml-stylesheet type="text/xsl" href="file:///p:/oasis/spec-0.4/stylesheets/
oasis-specification-html.xsl"?>
```

When a document is being authored in an environment where the remotely-installed online version of the publishing environment is available, the following stylesheet association processing instruction would perform equally as well (modulo the latency aspects of accessing resources over the Internet):

```
<?xml-stylesheet type="text/xsl" href="http://docs.oasis-open.org/
templates/DocBook/spec-0.4/stylesheets/oasis-specification-html.xsl"?>
```

Many people believe that stylesheet association is a transient property of an XML document and should not be included in any "official" normative XML content. This methodology, therefore, mandates that any such processing instruction that might be used by the author during the writing phase be removed from the final published XML instance.

3. Online publishing

The online publishing environment for this methodology is found at `http://docs.oasis-open.org/templates/DocBook/spec-0.4` complete with this documentation and directories for CSS stylesheets, XSLT stylesheets, and a pro forma template instance. There is no need to copy any files to your local machine environment in order to use the online publishing environment.

One needs to be connected to the Internet for online publishing to function.

See Appendix A, *Publishing choreography and orchestration (Non-Normative)* for example choreography and orchestration of these processes.

3.1. Validating the XML instance

To confirm that your DocBook instance conforms to the constraints of the DocBook vocabulary of elements and attributes, use any conforming DTD validating XML processor, process the XML instance checking it for being well-formed and valid. Ensure first that the document type declaration of your XML instance points the document type declaration's SYSTEM identifier to the online copy of the DocBook document type definition (DTD) as in the following.

```
<!DOCTYPE article
  PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
         "http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd"
```

Note that the PUBLIC identifier is handy in systems that use catalogues for dereferencing SYSTEM identifiers on the fly.

3.2. Creating the HTML rendition

To produce an HTML rendition run a conforming XSLT processor using your XML document as the source and `http://docs.oasis-open.org/templates/DocBook/spec-0.4/stylesheets/oasis-specification-html.xsl` as the stylesheet to create the final HTML result.

3.3. Creating the print renditions

To produce a print rendition run a conforming XSLT processor using your XML document as the source, and `http://docs.oasis-open.org/templates/DocBook/spec-0.4/stylesheets/oasis-specification-fo-a4.xsl` as the stylesheet to create the intermediate XSL-FO result for international A4 paper size (210mm by 297mm), and `http://docs.oasis-open.org/templates/DocBook/spec-0.4/stylesheets/oasis-specification-fo-us.xsl` as the stylesheet to create the intermediate XSL-FO result for US Letter paper size (8.5in by 11in).

The intermediate XSL-FO file needs to be interpreted into a formatted result by using an XSL-FO engine. Many such engines produce a Portable Document Format (PDF) final-form expression of the paginated result, while some will directly feed print systems and printers. For dissemination purposes it is recommended to produce PDF results and refer to them in the document collection.

4. Offline publishing

Three packages of files need to be downloaded to your local environment and then configured for offline publishing use.

While the offline publishing environment can be used very effectively in the development and writing of the specification documents, the final publicly-posted documents must be configured for online results. See Section 4.3, “OASIS specification stylesheets” for details on configuring for online results. See Section 5, “Packaging and check list” for reminders regarding publishing the final results.

See Appendix A, *Publishing choreography and orchestration (Non-Normative)* for example choreography and orchestration of these processes.

4.1. DocBook document model

An offline version of the DocBook document model is a verbatim copy of the online version found at `http://www.oasis-open.org/docbook/xml/4.4/`, and packaged in a single downloadable file in `http://www.oasis-open.org/docbook/xml/4.4/docbook-xml-4.4.zip`.

No further configuration of these files is necessary in order to be useful in the offline publishing environment.

For the purposes of this example, the extracted contents of that package are installed in the local "`p:\docbook`" directory. This generic directory name can be used elsewhere without having to be changed when the document models are updated.

For XML instances to engage the offline environment for validation, point the document type declaration's SYSTEM identifier to the offline copy of the DocBook document type definition (DTD) as in the following.

```
<!DOCTYPE article
  PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
         "file:///p:/docbook/docbookx.dtd"
```

4.2. DocBook stylesheets

An offline version of the DocBook stylesheets is a verbatim copy of the online version found in the directory `http://prdownloads.sourceforge.net/docbook`.

No further configuration of these files is necessary in order to be useful in the offline publishing environment.

For the purposes of this example, using `docbook-xsl-1.69.1.tar.gz` [http://prdownloads.sourceforge.net/docbook/docbook-xsl-1.69.1.tar.gz] or `docbook-xsl-1.69.1.zip` [http://prdownloads.sourceforge.net/docbook/docbook-xsl-1.69.1.zip] (being careful when looking in the long list to get the "`xsl`" package and not inadvertently download the "`xsl-doc`" package which does not have stylesheets), the extracted `docbook-xsl-1.69.1` directory is renamed "`xsl`" and moved to be the local "`p:\docbook\xsl`" directory. This generic directory name can be used elsewhere without having to be changed when the stylesheets are updated.

4.3. OASIS specification stylesheets

An offline version of the OASIS specification stylesheets is a slightly modified copy of the online version, where only a single file has to be changed to reflect the offline configuration and the nature of the target results.

The online version of this explanatory document you are reading points to the ZIP and TAR/GZ packages containing the OASIS specification stylesheets and related documentation and samples is found at `http://docs.oasis-`

open.org/templates/DocBook/spec-0.4/oasis-specification-0.4.html where the entire environment can be downloaded in the single package.

Note that if you are reading this document from a complete download of the environment, the stylesheets are already installed in the stylesheets/ directory.

The one file stylesheets/oasis-configuration.ent must be changed to reflect (1) the choices of installation subdirectories of the offline publishing environment; and (2) the nature of the result files being produced. The default configuration is for online publishing to produce results for use online.

To configure the directories file for offline publishing, change the following lines in that file that indicate the locations using URL syntax for the filenames (the p:/ names are merely examples being used by the author and are replaced with wherever you decide to install the files):

```
<!--locations of offline installation of support software-->
<!ENTITY offline-oasis-spec-directory  "file:///p:/oasis/spec-0.4/">
<!ENTITY offline-docbook-xsl-directory "file:///p:/docbook/xsl/">
```

Next, configure the stylesheets for offline use by indicating the offline stylesheets are included as follows (note the file that is mounted online is preconfigured for online use of the stylesheets):

```
<!--only one of the following two parameter entites can be "INCLUDE"-->
<!ENTITY % offline-stylesheets "INCLUDE">
<!ENTITY % online-stylesheets  "IGNORE">
```

Finally, leave the stylesheets configured to produce online results, or if you want to produce results for use locally on your machine without needing any online access, swap the "INCLUDE" and "IGNORE" strings in the following defaults:

```
<!--only one of the following two parameter entites can be "INCLUDE"-->
<!ENTITY % offline-results     "IGNORE">
<!ENTITY % online-results      "INCLUDE">
```

4.4. Creating the HTML rendition

To produce an HTML rendition run a conforming XSLT processor using your XML document as the source and your locally-installed equivalent to the example p:\oasis\spec-04\stylesheets\oasis-specification-html.xsl as the stylesheet to create the final HTML result.

4.5. Creating the print renditions

To produce a print rendition run a conforming XSLT processor using your XML document as the source, and your locally-installed equivalent to the example p:\oasis\spec-04\stylesheets\oasis-specification-fo-a4.xsl as the stylesheet to create the intermediate XSL-FO result for international A4 paper size (210mm by 297mm), and p:\oasis\spec-04\stylesheets\oasis-specification-fo-us.xsl as the stylesheet to create the intermediate XSL-FO result for US Letter paper size (8.5in by 11in).

The intermediate XSL-FO file needs to be interpreted into a formatted result by using an XSL-FO engine. Many such engines produce a Portable Document Format (PDF) final-form expression of the paginated result, while some will directly feed print systems and printers. For dissemination purposes it is recommended to produce PDF results and refer to them in the document collection.

5. Packaging and check list

The package of files for this environment and methodology includes the XML source, three renditions (HTML, PDF for the international A4 paper size and PDF for the US paper size), and a number of support files.

Before packaging the files to be posted it is recommended to consider the following check list:

- remove from the XML source any stylesheet association processing instruction
- ensure the XML source document type declaration SYSTEM identifier is pointing to the online reference and not an offline reference
- create online results when producing the final documents in order to ensure that all pointers (such as the OASIS logo and CSS stylesheet) are pointing to their correct online locations instead of offline locations

All of the required facilities for creating and testing ZIP and TAR/GZ files are available as core tasks in Ant [ant] as illustrated in the `template/package.xml` file in the pro forma template directory. In there you will see how to create a subdirectory of the files to be packaged, time stamp the files to a consistent date and time, package the files into both kinds of compressed archive files and uncompress the archive files into subdirectories suitable for testing.

A. Publishing choreography and orchestration (Non-Normative)

There is no obligation to automate the publishing process by choreographing the invocation of validation and production processes, however this methodology offers example orchestrations of these. Note that but for not having a core Ant [ant] task for the invocation of an XSL-FO processor, these guidelines would include such an example using that cross-platform Java-based build tool. It that absence, therefore, this section describes both a Windows command prompt and a Linux shell prompt that you may wish to configure for your own use.

The sample choreography invokes three separate processes in turn to produce the results. These processes are made available by Windows batch files and Linux shell scripts that are not included.

1. Windows batch file preparation

The example Windows batch file choreography in `template/wd-spectools-docbook-template.bat` assumes the presence of three support batch files on the command path summarized as follows:

- `call xmldtd.bat  input-XML-file`
- `call xslt.bat  input-XML-file  stylesheet-XSL-file  output-file`
- `call xslfo-pdf.bat  input-FO-file  output-PDF-file`

2. Linux shell script preparation

The example Linux shell script choreography in `template/wd-spectools-docbook-template.sh` [template/wd-spectools-docbook-template.bat] assumes the presence of three support shell scripts summarized as follows:

- `sh xmldtd.sh  input-XML-file`

- `sh xslt.sh input-XML-file stylesheet-XSL-file output-file`
- `sh xslfo-pdf.sh input-FO-file output-PDF-file`

3. Java-based sample applications

While any conforming XML, XSLT and XSL-FO engine can be used in a given environment, the following Java-based applications and their respective invocations are suitable. There is no obligation, however, to use these as any conforming implementations of validation, transformation and pagination can be used. Note these guidelines leave it up to the reader to download the required support and to appropriately set the required CLASSPATH environments for the following invocations to work.

To validate that an XML instance conforms to the constraints of a formal DTD expression, xjparse [xjparse] can be used to satisfy the `xmldtd` script as follows:

```
java com.nwalsh.parsers.xjparse %1
```

To transform and XML instance using and XSLT expression, Saxon [Saxon] can be used to satisfy the `xslt` script as follows:

```
java jar saxon.jar o %3 %1 %2
```

To transform an XSL-FO instance into a PDF file, FOP [FOP] can be used to satisfy the `xslfo-pdf` script as follows (though note that there may be some conformance challenges with FOP that might present some problems):

```
java org.apache.fop.apps.Fop fo %1 pdf %2
```

References

Normative

[DocBook] Norm Walsh *DocBook XML [http://www.docbook.org/xml/index.html], The DocBook 4.4 Document type [http://www.docbook.org/specs/cd-docbook-docbook-4.4.html]*. OASIS January 27, 2005

[RFC Keywords] S. Bradner *Key words for use in RFCs to Indicate Requirement Levels [http://rfc.net/rfc2119.txt]* Internet Engineering Task Force, March 1997

[XSL-FO] Sharon Adler, et al. *Extensible Stylesheet Language (XSL) Version 1.0 [http://www.w3.org/TR/2001/REC-xsl-20011015/]* W3C Recommendation 15 October 2001

[XSLT] James Clark *XSL Transformations (XSLT) Version 1.0 [http://www.w3.org/TR/1999/REC-xslt-19991116]* W3C Recommendation 16 November 1999

Non-normative

[ant] Apache Software Foundation *Ant Java-based Build Tool [http://ant.apache.org]* (Another Neat Tool).

[FOP] Apache Software Foundation *Formatting Objects Processor [http://xmlgraphics.apache.org/fop/]*.

[Saxon] Michael Kay *Saxon 6.5.5 XSLT 1.0 Processor [http://saxon.sourceforge.net/saxon6.5.5/index.html]*.

[xjparse] Norm Walsh *xjparse XML validation invocation [http://nwalsh.com/java/xjparse/]*.

[xml-assoc] James Clark *Associating Style Sheets with XML documents Version 1.0 [http://www.w3.org/1999/06/REC-xml-stylesheet-19990629]*. W3C Recommendation 29 June 1999.