
DocBook V5.0

The Transition Guide

01 March 2006

This version:

<http://docbook.org/docs/howto/2006-03-01/>

Latest version:

<http://docbook.org/docs/howto/>

Previous versions:

<http://docbook.org/docs/howto/2005-12-28/>

<http://docbook.org/docs/howto/2005-10-27/>

Authors:

Jirka Kosek, <jirka@kosek.cz>

Norman Walsh, <ndw@nwalsh.com>

Dick Hamilton, <rlhamilton@frii.com>

Table of Contents

Introduction	2
Finally in a namespace	2
Relaxing with DocBook	2
Why switch to DocBook V5.0?	3
Schema jungle	4
Toolchain	4
Editing DocBook V5.0	4
Validating DocBook V5.0	9
Processing DocBook V5.0	10
Markup changes	11
Improved cross-referencing and linking	11
Renamed elements	12
Removed elements	13
Converting DocBook V4.x documents to DocBook V5.0	14
What About Entities?	14
Customizing DocBook V5.0	15
FAQ	15
Bibliography	20

This document is targeted at DocBook users who are considering switching from DocBook V4.x to DocBook V5.0. It describes differences between DocBook V4.x and V5.0 and provides some suggestions about how to edit and process DocBook V5.0 documents. There is also section devoted to conversion of legacy documents from DocBook 4.x to DocBook V5.0.

At the time of this writing the current version of DocBook V5.0 was 5.0b3. However almost all information in this document is general and it is applicable to any newer version in DocBook V5.0 series.

Introduction

The differences between DocBook V4.x and V5.0 are quite radical in some aspects, but the basic idea behind DocBook is still the same and almost all element names are unchanged. Because of this it is very easy to become familiar with DocBook V5.0 if you know any previous version of DocBook. You can find a complete list of changes in [DB5SPEC], here we will discuss only the most fundamental changes.

Finally in a namespace

All DocBook V5.0 elements are in the namespace `http://docbook.org/ns/docbook`. XML namespaces are used to distinguish between different element sets. In the last few years, almost all new XML grammars have used their own namespace. It is easy to create compound documents that contain elements from different XML vocabularies. DocBook V5.0 is following this design rule. Using namespaces in your documents is very easy. Consider this simple article marked up in DocBook V4.5:

```
<article>
  <title>Sample article</title>
  <para>This is a really short article.</para>
</article>
```

The corresponding DocBook V5.0 article will look very similar:

```
<article xmlns="http://docbook.org/ns/docbook" ...>
  <title>Sample article</title>
  <para>This is a really short article.</para>
</article>
```

The only change is the addition of a default namespace declaration (`xmlns="http://docbook.org/ns/docbook"`) on the root element. This declaration applies the namespace to the root element and all nested elements. Each element is now uniquely identified by its local name and namespace.

Note

The namespace name `http://docbook.org/ns/docbook` serves only as an identifier. This resource is not fetched during processing of DocBook documents and you are not required to have an Internet connection during processing. If you access the namespace URI with a browser, you will find a short explanatory document about the namespace. In the future this document will probably conform to (some version of) RDDL and provide pointers to related resources.

Relaxing with DocBook

For more than a decade, the DocBook schema was defined using a DTD. However DTDs have serious limitations and DocBook V5.0 is thus defined using a very powerful schema language called RELAX NG. Thanks to RELAX NG, it is now much easier to create customized versions of DocBook, and some content models are now cleaner and more precise.

Using RELAX NG has an impact on the document prolog. The following example shows the typical prolog of a DocBook V4.x document. The version of the DocBook DTD (in this case 4.5) is indicated in the document type declaration (!DOCTYPE) which points to a particular version of the DTD.

Example 1. DocBook V4.5 document

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V4.5//EN"
    'http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd'>
<article lang="en">
  <title>Sample article</title>
  <para>This is really very short article.</para>
</article>
```

In contrast, DocBook V5.0 does not depend on DTDs anymore. This means that there is no document type declaration and the version of DocBook used is indicated with the version attribute instead.

Example 2. DocBook V5.0 document

```
<?xml version="1.0" encoding="utf-8"?>
<article xmlns="http://docbook.org/ns/docbook" version="5.0" xml:lang="en">
  <title>Sample article</title>
  <para>This is really very short article.</para>
</article>
```

As you can see, DocBook V5.0 is built on top of existing XML standards as much as possible, for example the lang attribute is superseded by the standard xml:lang attribute.

Another fundamental change is that there is no direct indication of the schema used. Later in this document, you will learn how you can specify a schema to be used for document validation.

Note

Although we recommend the RELAX NG schema for DocBook V5.0, there are also DTD and W3C XML Schema versions available (see the section called “Where to get the schemas”) for tools that do not yet support RELAX NG.

Why switch to DocBook V5.0?

The simple answer is “because DocBook V5.0 is the future”. Apart from this marketing blurb, there are also more technical reasons:

- *DocBook V4.x is feature frozen.* At the time of this writing DocBook V4.5 is the last version of DocBook in the V4.x series. Any new DocBook development, like the addition of new elements, will be done in DocBook V5.0. It is only a matter of time before useful, new elements will be added into DocBook V5.0, but they are not likely to be back ported into DocBook V4.x. DocBook V4.x will be in maintenance mode and errata will be published if necessary.
- *DocBook V5.0 offers new functionality.* Even the current version of DocBook V5.0 provides significant improvements over DocBook V4.x. For example there is general markup for annotations, a new and flexible system for linking, and unified markup for information sections using the info element.
- *DocBook V5.0 is more extensible.* Having DocBook V5.0 in a separate namespace allows you to easily mix DocBook markup with other XML based languages like SVG, MathML, XHTML or even FooBarML.
- *DocBook V5.0 is easier to customize.* RELAX NG offers many powerful constructs that make customization much easier than it would be using a DTD.

Schema jungle

Schemas for DocBook V5.0 are available in several formats at <http://www.oasis-open.org/docbook/xml/5.0b3/> (or the mirror at <http://docbook.org/xml/5.0b3/>). Only the RELAX NG schema is normative and it is preferred over the other schema languages. However, for your convenience there are also DTD and W3C XML Schema versions provided for DocBook V5.0. But please note that neither DTDs nor XML schemas are able to capture all the constraints of DocBook V5.0. This means that a document that validates against the DTD or XML schema is not necessarily valid against the RELAX NG schema and thus may not be a valid DocBook V5.0 document.

DTD and W3C XML Schema versions of the DocBook V5.0 grammar are provided as a convenience for users who want to use DocBook V5.0 with legacy tools that don't support RELAX NG. Authors are encouraged to switch to RELAX NG based tools as soon as possible, or at least to validate documents against the RELAX NG schema before further processing.

Where to get the schemas

The latest versions of schemas can be obtained from the following locations:

RELAX NG schema	http://docbook.org/xml/5.0b3/rng/docbook.rng
RELAX NG schema in compact syntax	http://docbook.org/xml/5.0b3/rng/docbook.rnc
DTD	http://docbook.org/xml/5.0b3/dtd/docbook.dtd
W3C XML Schema	http://docbook.org/xml/5.0b3/xsd/docbook.xsd

These schemas are also available from the mirror at <http://www.oasis-open.org/docbook/xml/5.0b3/>.

DocBook documentation

Detailed documentation about each DocBook V5.0 element is presented in the reference part of *DocBook: The Definitive Guide*¹.

Note

Other parts of the book have not yet been updated to reflect the changes made in DocBook V5.0. Please do not be confused by this.

Toolchain

This section briefly describes tools and procedures to edit and process content stored in DocBook V5.0.

Editing DocBook V5.0

Because DocBook is an XML based format and XML is a text based format, you can use any text editor to create and edit DocBook V5.0 documents. However using “dumb” editors like Notepad is not very productive. You will do better if you use an editor that supports XML. Although there are DTD and W3C XML Schemas available for DocBook V5.0, which means you can use any editor that works with DTDs or W3C XML Schemas, we recommend that you use the RELAX NG grammar with DocBook V5.0. The rest of this section contains an overview of XML editors (listed

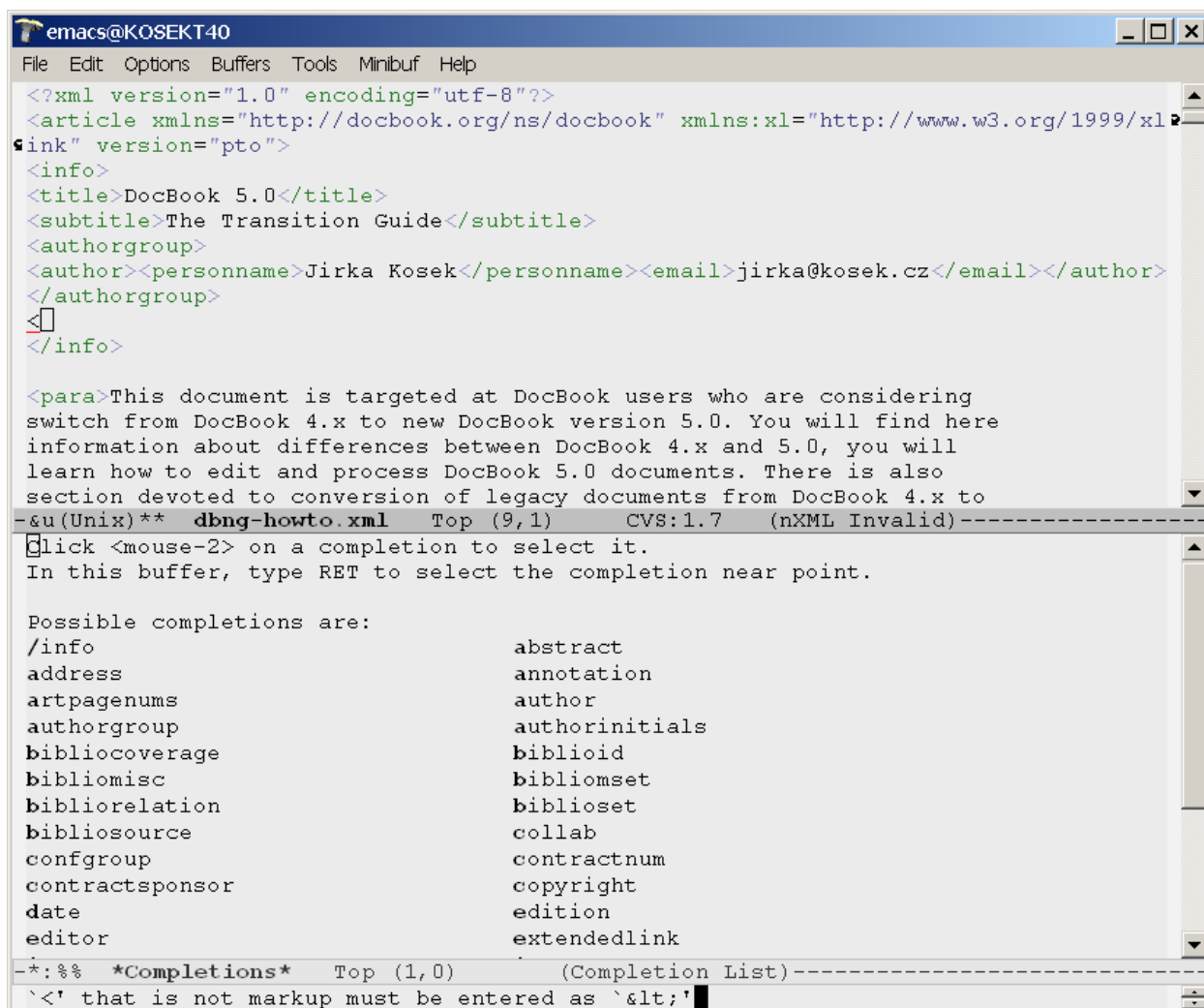
¹ <http://docbook.org/tdg5/en/html/pt02.html>

in alphabetical order) that are known to work with RELAX NG schemas and that offer guided editing based on the RELAX NG schema.

Emacs and nXML

nXML mode² is an add-on for the GNU Emacs text editor. By installing nXML you can turn Emacs into a very powerful XML editor that offers guided editing and validation of XML documents.

Figure 1. Emacs with nXML mode provides guided editing and validation



nXML uses a special configuration file named `schemas.xml` to associate schemas with XML documents. Often you will find this file in the directory `site-lisp/nxml/schema` inside the Emacs installation directory. Adding the following line into the configuration file, will associate DocBook V5.0 elements with the appropriate schema:

```
<namespace ns="http://docbook.org/ns/docbook" uri="/path/to/docbook.rnc"/>
```

² <http://www.thaiopensource.com/nxml-mode/>

Note

Please note that nXML ships with a file named `docbook.rnc`. This file contains the RELAX NG grammar for DocBook V4.x. Be sure that you associate the DocBook V5.0 namespace with the corresponding DocBook V5.0 grammar.

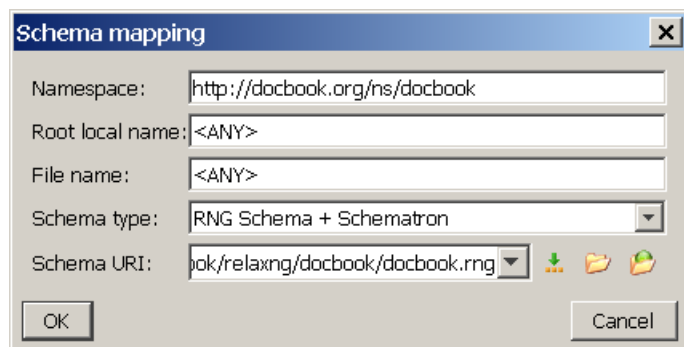
If you can't edit the global `schemas.xml` file, you can create this file in a directory with your document. nXML will find associations placed there also. In this case you must create a complete configuration file like:

```
<locatingRules xmlns="http://thaiopensource.com/ns/locating-rules/1.0">
  <namespace ns="http://docbook.org/ns/docbook" uri="/path/to/docbook.rnc" />
</locatingRules>
```

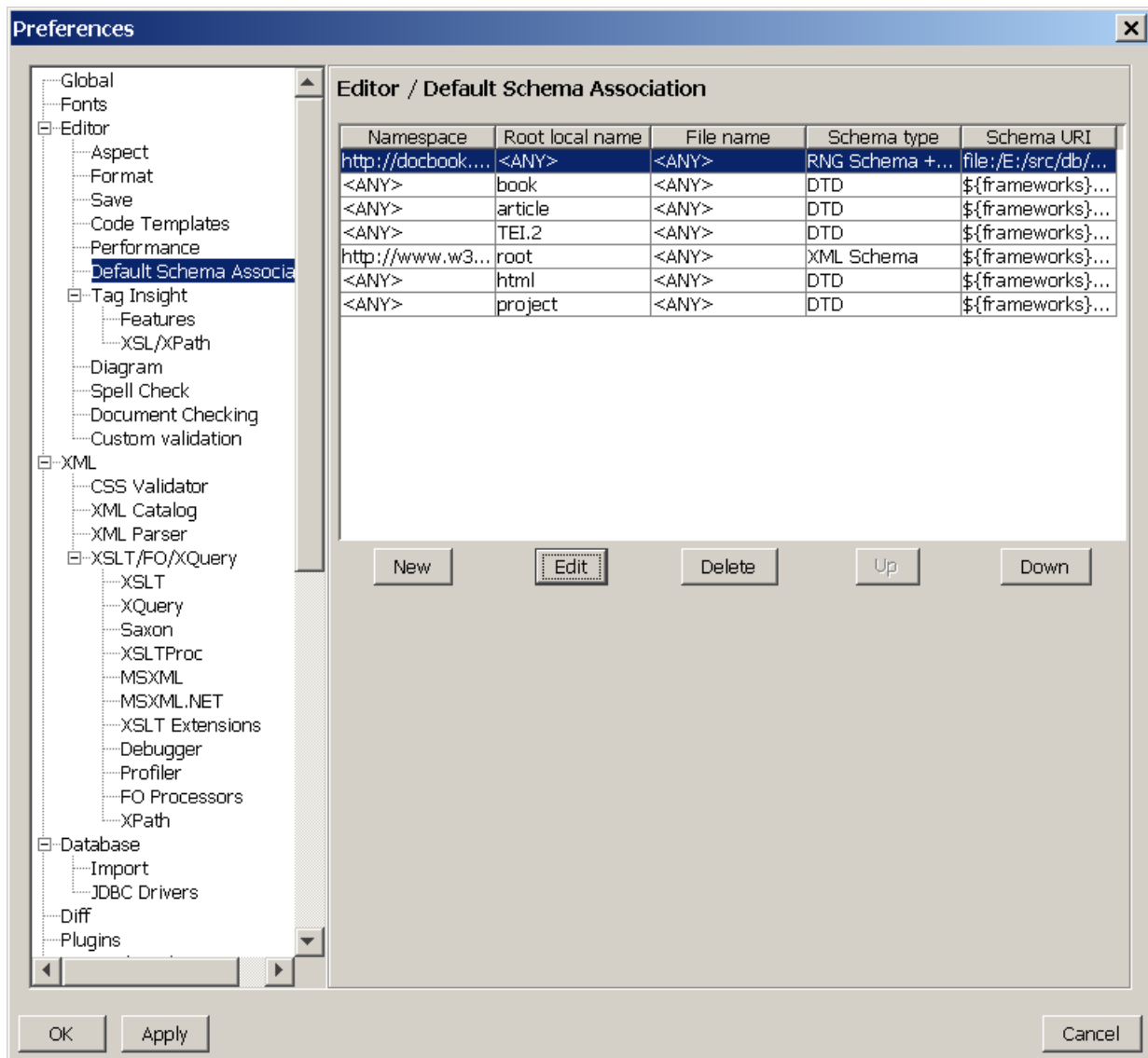
oXygen

oXygen is a feature rich XML editor. It has built-in support for many schema languages including RELAX NG. If you want to smoothly edit and validate DocBook 5.0 documents you should associate the DocBook namespace with the corresponding schema. Go to Options → Preferences... → Editor → Default Schema Associations. Then click the New button to add a new association. Type in the DocBook namespace and the RELAX NG schema location, choose the RNG Schema + Schematron type of schema as, and confirm your choice by clicking the OK button.

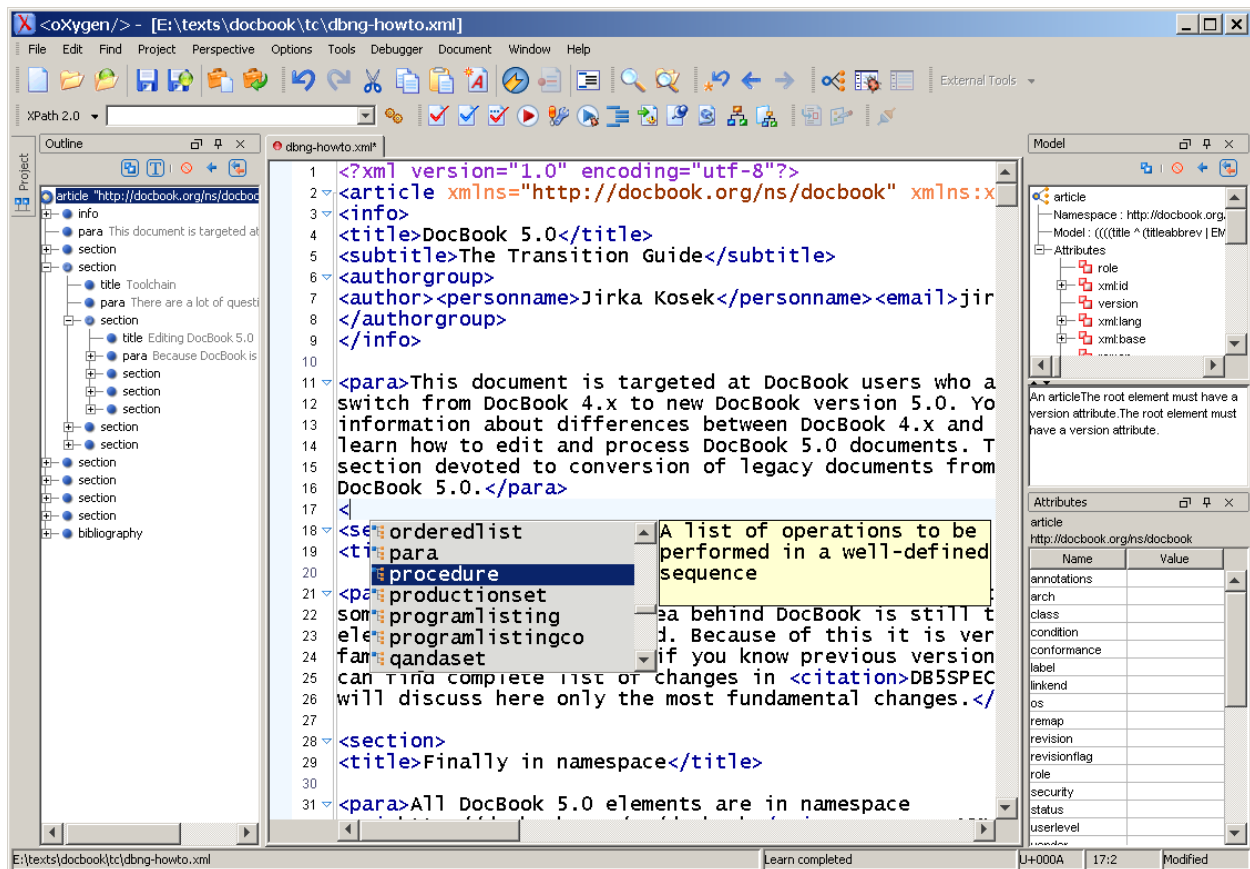
Figure 2. Adding a new schema association in oXygen



Because oXygen comes with preconfigured associations for DocBook V4.x, you must move your newly added configuration to the top of the list (using the Up button). That way you will be able to use oXygen with both DocBook V4.x and DocBook V5.0.

Figure 3. DocBook V5.0 association must precede associations for DocBook V4.x

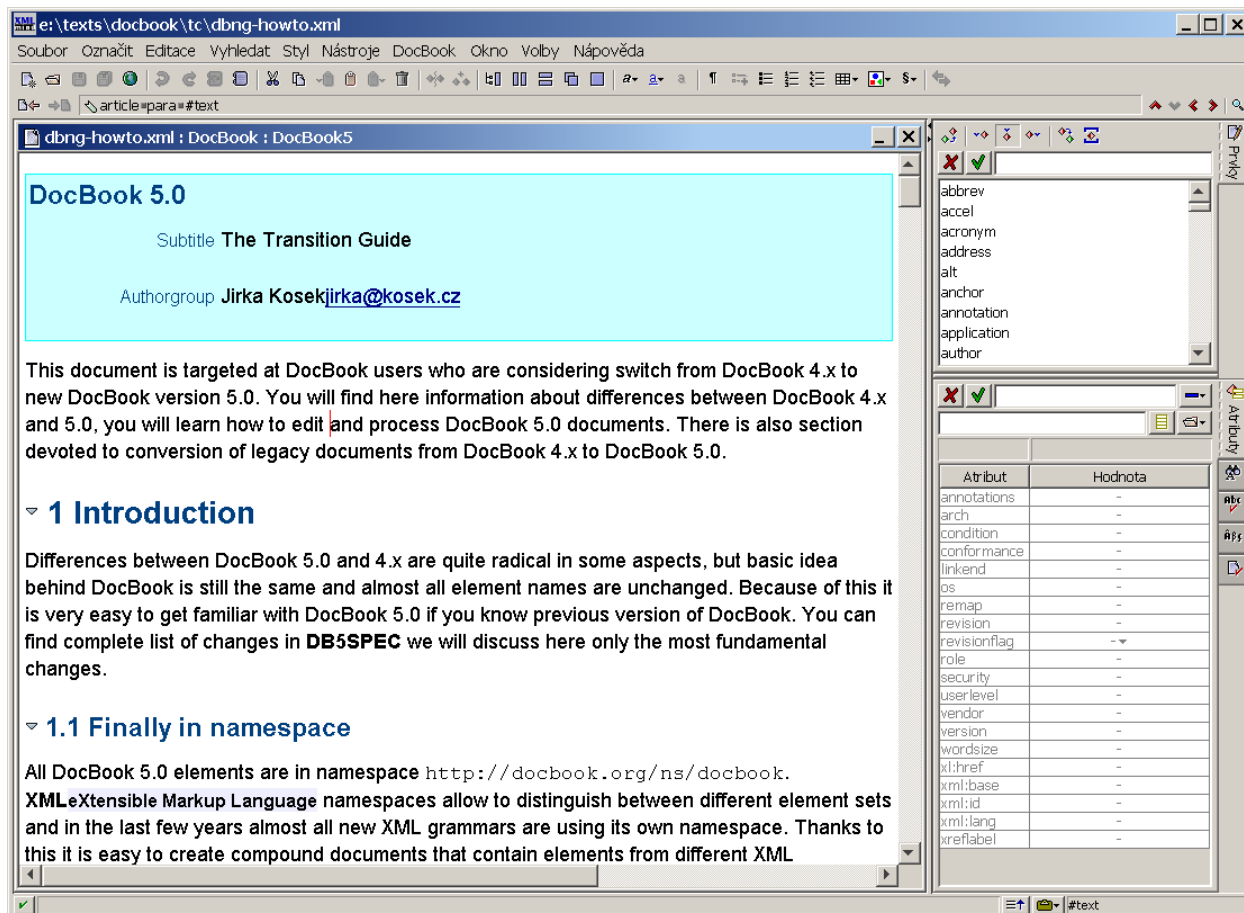
Now you can close the preference box by clicking on the OK button. From this time oXygen will assist you with writing DocBook V5.0 content and you will be able to validate your documents against both RELAX NG and Schematron schemas.

Figure 4. DocBook V5.0 document opened in oXygen

XML Mind XML editor

XML Mind XML editor (XXE) is a visual validating XML editor that provides a wordprocessor-like interface to users. It is available in two versions, Standard and Professional. The Standard version is free and provides everything you need to edit DocBook V5.0 documents.

Figure 5. XML Mind XML Editor – feels almost like MS Word but real DocBook V5.0 markup is created



Since version 2.11, XXE comes bundled with a DocBook V5.0 configuration. Unfortunately this configuration is not enabled by default. You must copy the contents of the directory `XXE_install_dir/doc/rnsupport/config/docbook5/` into `XXE_install_dir/addon/config/docbook5/` and restart XXE to activate it. After restarting XXE you will be able to create (a template for articles is provided) and edit DocBook V5.0 documents.

The RELAX NG schema provided with XXE may be outdated. If you want to use XXE with the latest schema just grab a fresh copy of `docbook.rng` and copy it into `XXE_install_dir/addon/config/docbook5/docbook.rng`.

Validating DocBook V5.0

If you are not using a RELAX NG based validating editor when you create documents, we strongly recommend that you validate your documents before processing them. Only after successful validation you can be sure that your document is really DocBook V5.0 and that processing tools will be able to process it correctly.

You can find a list of RELAX NG validators at <http://relaxng.org/#validators>. It is best to use validators with support for embedded Schematron rules inside RELAX NG schemas. Schematron is a rule-based validation language which is used to impose additional constraints on DocBook documents. Schematron rules assert conditions which cannot be expressed in a pure RELAX NG schema.

Sun Multi-Schema XML Validator (MSV) is able to validate an XML document against a RELAX NG schema and Schematron rules at the same time. To install and use MSV follow these steps:

1. Download `relames.zip` from <https://msv.dev.java.net/servlets/ProjectDocumentList?folderID=101>.
2. Unpack the downloaded file into an arbitrary directory.
3. Validate your document using the following command:

```
java -Xss512K -jar /path/to/relames.jar /path/to/docbook.rng document.xml
```

Note

The switch `-Xss512K` increases the stack size of the Java virtual machine. This is necessary because the DocBook schema is quite large. If you get stack overflow errors from MSV, increase this value. You may get spurious error messages if the value is too small, so if you get a stack overflow error, ignore other error messages and try a larger value for the stack size. If you are not using Sun's Java implementation, please consult the documentation for your virtual machine to learn how to increase the stack size.

There is also an on-line DocBook V5.0 validator³ that validates DocBook V5.0 documents against the normative RELAX NG schema with embedded Schematron rules.

Processing DocBook V5.0

Part of DocBook's great success can be attributed to the availability of free tools that can be used to transform DocBook content into various target formats including HTML and PDF. The DocBook XSL Stylesheets are very popular tools.

DocBook XSL Stylesheets

The DocBook stylesheets are designed to process content written in different versions of DocBook (for example 3.1 and 4.2). Recent versions of the stylesheets are also able to process DocBook V5.0 with some limitations.

You can process DocBook V5.0 documents with the DocBook XSL stylesheets exactly the same way as you process DocBook V4.x documents. You do not need special software, you can stick to your preferred XSLT processor, be it Saxon, xsltproc, Xalan or whatever else.

During document processing, the stylesheets strip namespaces from DocBook V5.0 to get a document which will be very similar to DocBook V4.x. This is necessary because from the XSLT point of view elements from different namespaces are distinct and can not be easily processed by the same set of templates. This process is completely transparent to the user. If you are processing DocBook V5.0 documents, the only difference is that you will see the following additional message:

```
Stripping NS from DocBook 5/NG document.
```

```
Processing stripped document.
```

Although you can successfully use the existing stylesheets to process DocBook V5.0, there are some limitations. To support some of the new features of DocBook V5.0, the existing stylesheets would require a significant rewrite. A rewrite is unlikely because a new version of stylesheets is currently under development.

The unsupported features include:

- general annotations;
- general XLink links on all elements;

³ <http://badame.vse.cz/docbookvalidator/>

During namespace stripping, the base URI of the document is lost. This means that in rare situations, relatively referenced resources like images or program listings can be processed incorrectly. The stylesheets attempt to compensate for this problem, but it is possible that there are corner cases where they will fail.

XSLT 2.0 based re-implementation

XSLT 1.0 is missing some important features. To work around these missing features, the current DocBook XSL stylesheets use some implementation-specific extensions. XSLT 2.0 adds many new and previously missing features into the language. A new set of DocBook stylesheets is being implemented based on XSLT 2.0 to take advantage of these features and to fully support DocBook V5.0.

The XSLT 2.0 based stylesheets have many new features, including:

- seamless integration of profiling (conditional documents) with external bibliographies and glossaries;
- no need for (most) external extensions;
- internationalized indexes;
- easy to customize titlepage templates;

The XSLT 2.0 based stylesheets are still under development. At this writing, they only support HTML and chunked HTML output. As time permits, the stylesheet developers will be adding other formats. Since the stylesheets are developed in the limited free time the developers have, there's no specific schedule.

There are not very many XSLT 2.0 implementations available. But, if you want to try the new stylesheets, grab a snapshot of the development version from <http://docbook.sourceforge.net/snapshots/docbook-xsl2-snapshot.zip> and unpack it somewhere. Then download and install Saxon 8 from <http://saxon.sf.net>.

To transform a DocBook V5.0 document to a single HTML page use the command:

```
java -jar /path/to/saxon8.jar -o output.html document.xml /path/to/docbook-xsl2-snapshot/html/docbook.xsl
```

To transform a DocBook V5.0 document to a set of chunked HTML pages use the command:

```
java -jar /path/to/saxon8.jar document.xml /path/to/docbook-xsl2-snapshot/html/chunk.xsl
```

Markup changes

This section describes the most common markup changes between DocBook V4.x and V5.0. You can find a complete list of changes in [DB5SPEC].

Improved cross-referencing and linking

In DocBook V4.x the attribute `id` is used to assign a unique identifier to an element. In DocBook V5.0 this attribute is renamed `xml:id` in order to comply with [XMLID].

Now you can use almost any inline element as the source of a link, not just `xref` or `link`. For example, the following DocBook 4.x content:

```
<section id="dir">
  <title>DIR command</title>
  <para>...</para>
</section>
```

```
<section id="ls">
  <title>LS command</title>
  <para>This command is a synonym for <link linkend="dir"><command>DIR</command></link> command.</para>
</section>
```

is written in DocBook V5.0 as:

```
<section xml:id="dir">
  <title>DIR command</title>
  <para>...</para>
</section>
```

```
<section xml:id="ls">
  <title>LS command</title>
  <para>This command is a synonym for <command linkend="dir">DIR</command> command.</para>
</section>
```

The linkend attribute was added to all inline elements together with the href attribute from the XLink namespace. This means that you can use any inline element as the source of a hypertext link. To use XLinks you have to declare the XLink namespace (most often on the root element of your document):

```
<article xmlns="http://docbook.org/ns/docbook"
         xmlns:xl="http://www.w3.org/1999/xlink" version="5.0">
  <title>Test article</title>

  <para><application xl:href="http://www.gnu.org/software/emacs/emacs.html">Emacs</application>
    is my favourite text editor.</para>
  ...
```

The ulink element was removed from DocBook V5.0 in favor of XLink linking. Instead of the DocBook V4.x ulink element:

```
<ulink url="http://docbook.org">DocBook site</ulink>
```

you can now use link

```
<link xl:href="http://docbook.org">DocBook site</link>
```

XLink links may contain a fragment identifier, which you can use instead of linkend to form cross-references inside a document; for example:

```
<command xl:href="#dir">DIR</command>
```

However XLink links are not checked during validation, while xml:id/linkend links are checked for ID/IDREF consistency. One place where the XLink-based, fragment identifier scheme is useful is when XInclude is being used, since XML ID/IDREF links cannot span XInclude boundaries. You can use whichever approach better suits your needs.

Renamed elements

Some elements were renamed to better express their meaning or to reduce the total number of elements available in DocBook.

Table 1. Renamed elements

Old name	New name
sgmltag	tag
bookinfo, articleinfo, chapterinfo, *info	info
authorblurb	personblurb
collabname, corppauthor, corpccredit, corpname	orgname
isbn, issn, pubnumber	biblioid
lot, lotentry, tocbac, tocchap, tocf, tocleve1, tocleve2, tocleve3, tocleve4, tocleve5, tocp	tocdiv
graphic, graphicco, inlinegraphic, mediaobjectco	mediaobject and inlinemediaobject
ulink	link

Removed elements

The following elements were removed from DocBook V5.0 without direct replacements: action, beginpage, highlights, interface, invpartnumber, medialabel, modespec, structfield, structname. If you use one or more of these elements, here are some suggestions as to how to re-code them in DocBook V5.0.

Table 2. Recommended mapping for removed elements

Old name	Recommended mapping
action	Use <phrase remap="action">.
beginpage	Remove: beginpage is advisory only and has tended to cause confusion. A processing instruction or comment should be a workable replacement if one is needed.
highlights	Use abstract. Note that because highlights has a broader content model, you may need to wrap contents in a para inside abstract.
interface	Use one of the “gui*” elements (guibutton, guicon, guilabel, guimenu, guimenuitem, or guisubmenu).
invpartnumber	Use <biblioid class="other" otherclass="medialabel">. The productnumber element is another alternative.
medialabel	Use <citetitle pubwork="mediatype">, where mediatype is the type of media being labeled (e.g., cdrom or dvd).
modespec	No longer needed. The current processing model for olink renders modespec unnecessary.
structfield, structname	Use varname. If you need to distinguish between the two, use <varname remap="structname or structfield">. In some contexts, it may also be appropriate to use property for structfield.

Converting DocBook V4.x documents to DocBook V5.0

The DocBook V5.0 schema ships with an XSLT 1.0 stylesheet that is designed to transform valid DocBook V4.x documents to valid DocBook V5.0 documents.

To convert your document, `doc.xml` in the examples below, follow these steps:

1. Check the validity of your DocBook XML V4.x document. The conversion tool assumes that the input document is valid. If the input document contains markup errors, the results will be unpredictable at best.
2. Transform `doc.xml` to `newdoc.xml` with the `db4-upgrade.xsl` stylesheet included in the DocBook V5.0 distribution that you are using.
3. Check the validity of your DocBook XML V5.0 document against the DocBook V5.0 RELAX NG grammar.

In the vast majority of cases, the resulting document should be valid and your conversion process is finished.

If the document is not valid, please report the problem. (Over time, we'll have more experience with the sorts of things that can go wrong and we'll update this document to reflect that experience.)

What About Entities?

Using XSLT to transform existing documents to DocBook V5.0 has one potential disadvantage: it removes all entity references from your document.

If preserving entities is an important aspect of your production work flow, you will have to engage in a semi-manual process to preserve them.

1. Open your existing document using your favorite editing tool. You must use a tool that *is not* XML-aware, or one that allows you to edit markup “in the raw”.
2. Replace all occurrences of the entity references that you want to preserve with some unique string. For example, if you want to preserve “`∏`” references, you could replace them all with “`[[[Product]]]`” (assuming that the string “`[[[Product]]]`” doesn't occur anywhere else in your document).
3. Copy the document type declaration off of your document and save it some place. The document type declaration is everything from “`<!DOCTYPE`” to the closing “`>`”.
4. Perform the conversion described in the section called “Converting DocBook V4.x documents to DocBook V5.0”.
5. Open the new document using your favorite editing tool. Replace all occurrences of the unique string you used to save the entity references with the corresponding entity references.
6. Paste the document type declaration that you saved onto the top of your new document.
7. Remove the external identifier (the `PUBLIC` and/or `SYSTEM` keywords) from the document type declaration. A document that begins:

```
<!DOCTYPE book [  
<!ENTITY someEntity "some replacement text">  
>
```

is perfectly well-formed. If you don't remove the references to the DTD, then your parser will likely try to validate against DocBook V4.0 and that's not going to work. Alternatively, you could refer to the DocBook V5.0 DTD.

Tip

Steps 2 and 5 from previous procedure can be automated using the cloak script⁴ written by Michael Smith.

External Parsed Entities

External parsed entities, entities which load part of a document from another file, are a special case. These can often be replaced with XInclude elements.

The Perl script `db4-entities.pl`, also included in the DocBook V5.0 distribution attempts to perform this replacement for you. To use the script, perform the following steps:

1. Process your document with `db4-entities.pl`. The script expects a single filename and prints the XInclude version on standard output.
2. Process the XInclude version as described in the section called “Converting DocBook V4.x documents to DocBook V5.0”.

Customizing DocBook V5.0

TBD

FAQ

1. Authoring

- 1.1. How do I attach a schema to a DocBook V5.0 document when I do not want to use DTDs and `!DOCTYPE`?

There is no standard way of associating a RELAX NG schema with a document. Most tools provide some mechanism for performing this association, consult the documentation for your application. In some tools you must specify schema manually each time you want to edit/process your document.

- 1.2. How do I use entities like `–` in DocBook V5.0?

Modern schema languages (including RELAX NG and W3X XML Schema) do not provide any means to define entities that can be used for easier typing of special characters. Some editors provide functions or special toolbars that allow you to easily pick necessary character and insert it into document as a raw Unicode character or a numeric character reference.

Another possibility is to include entity definitions in the prolog of your document. Entity definition files⁵ are now maintained by W3C. You can reference definition files with entity definitions you are interested in and then reference imported entities. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE article [
<!ENTITY % isopub SYSTEM "http://www.w3.org/2003/entities/iso8879/isopub.ent">
%isopub;
```

⁴ <http://docbook.sourceforge.net/outgoing/cloak>

⁵ <http://www.w3.org/2003/entities/>

```
]>
<article xmlns="http://docbook.org/ns/docbook" version="5.0">
<title>DocBook V5.0 &ndash; the superb documentation format</title>
...
```

1.3. How to modularize documents?

You can use XInclude⁶ for this task. There is an alternative schema for DocBook V5.0 that contains XInclude elements. This is necessary to make some XML editors happy. This schema can be found in files that end with letters “xi”, e.g. `docbookxi.rnc` instead of `docbook.rnc`.

2. Stylesheets

2.1. Will the current DocBook XSL stylesheets (XSLT 1.0 based implementation) be maintained and improved in the future since work on a new XSLT 2.0 based implementation has started?

Yes, the current stylesheets (like 1.69.1) will be supported and improved further because they are very widely deployed and work with many existing XSLT processors.

Surely there will be a point in a future when all new development will be switched to the XSLT 2.0 based implementation. But this will not happen until all features of the current stylesheets are implemented in the new stylesheets and until there is more than one usable XSLT 2.0 processor available.

3. Schema customizations

3.1. How can I extend the DocBook schema with MathML elements?

The basic DocBook schema allows elements from the MathML namespace to appear inside the `equation` element. This means that you can validate a DocBook+MathML document, but MathML content will be ignored during the validation. You will also not be able to use guided editing for the MathML content.

If you need strict validation of MathML content or guided editing for MathML, you can easily extend the base DocBook schema with the MathML schema.

Procedure 1. Extending the DocBook schema with the MathML schema

1. Download the MathML RELAX NG schema from <http://yupotan.sppd.ne.jp/relax-ng/mml2.html> and unpack it somewhere (e.g. into a `mathml` subdirectory).
2. Create a schema customization in compact syntax—`dbmathml.rnc`:

```
namespace html = "http://www.w3.org/1999/xhtml"
namespace mml = "http://www.w3.org/1998/Math/MathML"
namespace db = "http://docbook.org/ns/docbook"

include "/path/to/docbook.rnc" {
  db._any.mml = external "mathml/mathml2.rnc"
  db._any =
    element * - (db:* | html:* | mml:*) {
      (attribute * { text }
       | text
       | db._any)*
    }
```

⁶ <http://www.w3.org/TR/xinclude/>


```
}  
}
```

Or, alternatively, you can use the XML syntax of RELAX NG—`dbmathml.rng`:

```
<?xml version="1.0" encoding="UTF-8"?>  
<grammar xmlns="http://relaxng.org/ns/structure/1.0">  
  
  <include href="/path/to/docbook.rng">  
    <define name="db._any.mml">  
      <externalRef href="mathml/mathml2.rng"/>  
    </define>  
  
    <define name="db._any">  
      <element>  
        <anyName>  
          <except>  
            <nsName ns="http://docbook.org/ns/docbook"/>  
            <nsName ns="http://www.w3.org/1999/xhtml"/>  
            <nsName ns="http://www.w3.org/1998/Math/MathML"/>  
          </except>  
        </anyName>  
        <zeroOrMore>  
          <choice>  
            <attribute>  
              <anyName/>  
            </attribute>  
            <text/>  
            <ref name="db._any"/>  
          </choice>  
        </zeroOrMore>  
      </element>  
    </define>  
  </include>  
  
</grammar>
```

3. Now use the customized schema (`dbmathml.rnc` or `dbmathml.rng`) instead of the original DocBook schema.

3.2. How can I extend the DocBook schema with SVG elements?

The situation is the same as with MathML support. You can use elements from the SVG namespace inside the `imageobject` element.

Procedure 2. Extending the DocBook schema with the SVG schema

1. Download the SVG RELAX NG schema from <http://www.w3.org/Graphics/SVG/1.1/rng/rng.zip> and unpack it somewhere (e.g. into an `svg` subdirectory).
2. Create a schema customization in compact syntax—`dbsvg.rnc`:

```
namespace html = "http://www.w3.org/1999/xhtml"  
namespace db = "http://docbook.org/ns/docbook"
```

```
namespace svg = "http://www.w3.org/2000/svg"

include "/path/to/docbook.rnc" {
  db._any.svg = external "svg/svg11.rnc"
  db._any =
    element * - (db:* | html:* | svg:*) {
      (attribute * { text }
       | text
       | db._any)*
    }
}
```

Or, alternatively, you can use the XML syntax of RELAX NG—`dbsvg.rng`:

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0">

<include href="/path/to/docbook.rng">
  <define name="db._any.svg">
    <externalRef href="svg/svg11.rng"/>
  </define>

  <define name="db._any">
    <element>
      <anyName>
        <except>
          <nsName ns="http://docbook.org/ns/docbook"/>
          <nsName ns="http://www.w3.org/1999/xhtml"/>
          <nsName ns="http://www.w3.org/2000/svg"/>
        </except>
      </anyName>
      <zeroOrMore>
        <choice>
          <attribute>
            <anyName/>
          </attribute>
          <text/>
          <ref name="db._any"/>
        </choice>
      </zeroOrMore>
    </element>
  </define>
</include>

</grammar>
```

3. Now use the customized schema (`dbsvg.rnc` or `dbsvg.rng`) instead of the original DocBook schema.

3.3. Is it possible to use the previous two customizations for MathML and SVG together?

Yes, you can create a special schema customization that combines both MathML and SVG with the DocBook schema. In compact syntax, the merged schema is:

```
namespace html = "http://www.w3.org/1999/xhtml"
namespace mml = "http://www.w3.org/1998/Math/MathML"
namespace db = "http://docbook.org/ns/docbook"
namespace svg = "http://www.w3.org/2000/svg"
```

```
include "/path/to/docbook.rnc" {
  db._any.mml = external "mahtml/mathml2.rnc"
  db._any.svg = external "svg/svg11.rnc"
  db._any =
    element * - (db:* | html:* | mml:* | svg:*) {
      (attribute * { text }
       | text
       | db._any)*
    }
}
```

Or alternatively in the full RELAX NG syntax:

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0">

<include href="/path/to/docbook.rng">
  <define name="db._any.mml">
    <externalRef href="mathml/mathml2.rng"/>
  </define>

  <define name="db._any.svg">
    <externalRef href="svg/svg11.rng"/>
  </define>

  <define name="db._any">
    <element>
      <anyName>
        <except>
          <nsName ns="http://docbook.org/ns/docbook"/>
          <nsName ns="http://www.w3.org/1999/xhtml"/>
          <nsName ns="http://www.w3.org/1998/Math/MathML"/>
          <nsName ns="http://www.w3.org/2000/svg"/>
        </except>
      </anyName>
      <zeroOrMore>
        <choice>
          <attribute>
            <anyName/>
          </attribute>
          <text/>
          <ref name="db._any"/>
        </choice>
      </zeroOrMore>
    </element>
  </define>
</include>

</grammar>
```

4. Tool specific problems

- 4.1. I'm using Altova XMLSpy to validate DocBook V5.0 instances against the W3C XML Schema (`docbook.xsd`). XMLSpy complains about undefined `xml:id` attributes?

XMLSpy always uses its own bundled version of `xml.xsd` which unfortunately doesn't define the `xml:id` attribute. The bundled version of `xml.xsd` is hardwired into the program and cannot be replaced by a newer version. To solve this problem you must upgrade to version 2006 SP1.

Bibliography

[RNCTUT] Clark, James – Cowan, John – MURATA, Makoto: RELAX NG Compact Syntax Tutorial. Working Draft, 26 March 2003. OASIS. <http://relaxng.org/compact-tutorial-20030326.html>

[XMLID] Marsh, Jonathan – Veillard, Daniel – Walsh, Norman: `xml:id` Version 1.0. W3C Recommendation, 9 September 2005. <http://www.w3.org/TR/xml-id/>

[DB5SPEC] Norman, Walsh: The DocBook Schema. Working Draft 5.0a1, OASIS, 29 June 2005. <http://www.docbook.org/specs/wd-docbook-docbook-5.0a1.html>