

Inhaltsverzeichnis

DocBook Publishing.....	1
Kapitel 1. Einführung.....	2
1.1. Voraussetzungen.....	2
Kapitel 2. DocBook Publishing online lesen.....	3
Kapitel 3. Software.....	4
3.1. Überblick.....	4
3.2. Java.....	4
3.3. HTMLDOC.....	4
3.4. W3M.....	5
3.5. HTML Help Workshop.....	5
3.6. Python.....	5
3.7. Das Framework selbst.....	6
3.8. OpenOffice–Writer.....	6
Kapitel 4. Hello World–Projekt.....	8
Kapitel 5. Kleines Tutorial.....	10
5.1. Book oder Article.....	10
5.2. Ein einfaches DocBook Buch.....	10
5.3. Ein Artikel.....	11
5.4. Online Ressourcen.....	12
Kapitel 6. DocBook Publishing mit OpenOffice.....	13
6.1. Zusammenspiel zwischen OpenOffice und DocBook.....	13
6.2. Weiterführende Funktionen.....	13
Kapitel 7. Download DocBook Publishing.....	15

DocBook Publishing

Version 1.02

24. Februar2003

Copyright © 2003 Stefan Rinke

Diesen Artikel gibt's online auf [Stefan Rinke, Articles](#)

Kapitel 1. Einführung

DocBook Publishing ist eine Beschreibung eines Frameworks für das Publishing mit DocBook. Es beschreibt, wie man auf einfache Weise eine Publikation oder einen Artikel in den unterschiedlichsten Formaten erzeugen kann. Sie können diesen Artikel online lesen oder ihn in verschiedenen Formaten herunterladen. Dieses ganze Framework beruht auf der Arbeit von sehr vielen Leuten, die viele Teile dazu beigetragen und der Allgemeinheit als Open-Source zur Verfügung gestellt haben. Herausheben möchte ich Mark Pilgrim, der mich mit seinen Publikationen, insbesondere Dive Into Python, zu diesem Artikel inspiriert hat. Ich habe nichts weiter getan, als das ganze unter Windows und auf deutsch mit allen notwendigen Tools zum Laufen zu bringen und nun meinerseits damit zu arbeiten: Dieser Artikel hier ist selbstverständlich damit verfasst.

Um es noch einmal hervorzuheben: es geht mir hier nicht so sehr um DocBook an sich, hier will ich auf andere Sites verweisen, vielmehr soll ein Setup beschrieben werden, wie man das Ganze praktisch zum Laufen bekommt. Außerdem sind auf der WebSite auch alle erforderlichen Tools zum Download (auch als Binary) angeboten, so dass es nicht schwer fallen sollte, selbst ein System zusammen zu stellen, um so erfolgreich Dokumente mit DocBook schreiben zu können.

1.1. Voraussetzungen

Es ist wichtig zu verstehen, dass sich bei diesem Framework nicht um eine WYSIWYG Lösung handelt oder eine Klick-Klack-Anwendung (eine Windows-Applikation, die sich per Maus bedienen lässt). Vielmehr werden Artikel im *Quelltext* in DocBook (also XML) geschrieben und anschließend kompiliert. Das erinnert manchen vielleicht noch an die alten TEX-Zeiten, wo das Schreiben eines Dokuments so ähnlich abgelaufen ist.

Wer sich also gänzlich vor der Kommandozeile scheut, der sollte vielleicht lieber die Finger davon lassen und sich weiter mit gängigen Officetools bescheiden.

Der Umgang mit XML und etwas XSLT sollte einem ebenso geläufig sein, damit man nicht schon an den einfachsten Dingen scheitert. Alles Weitere will ich hier versuchen so weit zu erläutern, dass man schon bald ein erstes Ergebnis erzielen kann.

Kapitel 2. *DocBook Publishing* online lesen

Im Moment ist dieser Artikel noch nicht ganz vollendet, d.h. ich arbeite ständig eigene Erweiterungen oder Kommentare von Usern ein. Bald wird auf eine Revision–History zur Verfügung stehen, die die Veränderungen von Version zu Version zeigt.

Kapitel 3. Software

3.1. Überblick

Um alle Formate erzeugen zu können, werden einige Software-Pakete benötigt.

- Die Build-Umgebung stützt sich auf Ant, ein sehr populäres Buildtool von der Apache Group. Ebenso sind die XML-Tools insbesondere der Stylesheet-Prozessor, auf *Java* angewiesen. D.h. man braucht eine Java Laufzeit Umgebung mindestens Version 1.3.
- Um die Wordversion eines Artikels zu erzeugen, wird natürlich ein richtiges *Microsoft Word* benutzt. Dies funktioniert natürlich nur auf Microsoft Windows und dies ist auch das einzige, was nicht frei erhältlich ist.
- Die Help-File-Formate benötigen ein Tool von Microsoft, welches frei von Microsoft heruntergeladen werden kann: hhc.exe der Hilfe-Compiler
- Das PDF Format wird durch das Tool HTMLDOC erzeugt. Sourcecode und Binary gibt's ebenfalls frei zum Download.
- Das Text-Plain Format wird über HTML und das Tool w3m, ein Textbrowser ähnlich wie Lynx, erzeugt.
- Verschiedene Support-Funktionen wurden von Mark in Python realisiert, so dass auch ein Python auf dem System installiert sein sollte.
- Wer schließlich mit OpenOffice sein DocBook erstellen möchte, der muss natürlich ein OpenOffice installieren.
- Das Paket ooo2sDbk konvertiert OpenOffice Dokumente nach DocBook.

3.2. Java

Java wird am besten direkt bei Sun heruntergeladen. Unter <http://java.sun.com/j2se/1.3/download.html> kann man die Runtime 1.3.1_06^[1] für Windows herunterladen. Sun liefert ein ganz normales Setup, so dass die Installation keine großen Probleme bereiten sollte.

Wichtig: Damit das Haupt-Batch-File, welches Ant startet, das Java auch findet, muss man eine Environmentvariable JAVA_HOME setzen. Alternativ kann man den Java-Installations-Pfad auch direkt in das Batchfile reinschreiben.

Wenn man alles richtig gemacht hat, dann kann man an der Kommandozeile folgende Tests machen:

```
[C:\ ]$ echo %JAVA_HOME%
C:\Programme\JavaSoft\jdk1.3.1_06
[C:\ ]$ %JAVA_HOME%\bin\java
Usage: java [-options] class [args...]
(to execute a class)
or java -jar [-options] jarfile [args...]
(to execute a jar file)
...
```

Außerdem werden zum Bauen und Transformieren per XSLT noch eine paar Java-Bibliotheken (JARs) gebraucht. Da diese immer dieselben sind für alle Projekte installiert man sie am besten im Lib-Verzeichnis von JAVA_HOME (hier C:\Programme\JavaSoft\jdk1.3.1_06\lib. Zum Download gibt's diese Bibliotheken hier.

3.3. HTMLDOC

HTMLDOC gibt's unter <http://www.easysw.com/htmldoc/> als Quellcode und als Binary, wobei das Binary in irgendeiner Form eine eingeschränkte Funktionalität haben soll. Ich hab das nicht weiter ausprobiert und stattdessen ein neues Binary kompiliert.

Alles was benötigt wird ist das `htmldoc.exe` an sich und ein paar Zeichensatz-Dateien, die das Binary als Default unter `C:\Program Files\htmldoc` erwartet. In diesem ZIP-File sind alle diese Dateien zusammengefasst. Zu beachten ist noch, dass das Binary `htmldoc.exe` selbst unter dem PATH auch gefunden werden muss. Das erreicht man z.B. dadurch, dass man es in Windows-Verzeichnis kopiert. Wer das nicht mag, der legt am besten ein eigenes Verzeichnis für derartige Tools an (bei mir ist das `C:\Programme\bin`) und erweitert den Pfad entsprechend, so dass dieses Verzeichnis mit einbezogen wird. Testen kann man das am besten dadurch, dass man auf der Eingabeaufforderung einfach mal `htmldoc` eingibt. Dann sollte etwa folgendes erscheinen.

```
[C:\ ]$ htmldoc
ERROR: No HTML files!
HTMLDOC Version 1.8.21 Copyright 1997-2002 Easy Software Products, All Rights Reserved.
This software is governed by the GNU General Public License, Version 2, and is based in part on the w

Usage:
  htmldoc [options] filename1.html [ ... filenameN.html ]

Options:

  --batch filename.book
  ...
```

3.4. W3M

Das Tool `w3m` für die Ausgabe von Plaintext gibt unter <http://www.w3m.org/> zum Download. Das Einzige was im Zusammenhang mit diesem Framework hier benötigt wird ist allerdings das `w3m.exe` Binary und die `CGYWIN-DLL` ^[2]. Das kann man alternativ auch direkt hier herunterladen. Ebenso wie in vorherigen Abschnitt beschrieben, kopiert man das EXE und die DLL in ein Verzeichnis, was über den Pfad gefunden wird. Wenn alles korrekt ist dann sieht man an der Kommandozeile folgendes:

```
[C:\ ]$ w3m
version w3m/0.1.9
usage: w3m [options] [URL or filename]
options:

-t tab set tab width
...
```

3.5. HTML Help Workshop

Der Hilfecompiler von Microsoft ist direkt bei Microsoft erhältlich z.B. unter dieser URL. Das File `HTMLHELP.EXE` Version 1.32 ist das benötigte File und wird nach dem Download einfach gestartet. Das Setup installiert den *HTML Help Workshop* in der Regel unter `C:\Programme\HTML Help Workshop`. Wiederum gilt, soll alles aus dem Stand laufen, muss auch dieses Verzeichnis über den Pfad gefunden werden. Alternativ kann man auch das `hhc.exe` in das Toolsverzeichnis kopieren. Wenn alles korrekt ist dann sieht man an der Kommandozeile folgendes:

```
[C:\ ]$ hhc
Usage: hhc <filename>
where <filename> = an HTML Help project file
Example: hhc myfile.hhp
```

3.6. Python

Python kann man entweder von <http://www.python.org/> beziehen oder bei ActiveState das ActivePython herunterladen. Nach erfolgreichem Setup gilt das schon mehrfach gesagte: Python muss über den Pfad erreichbar sein. Beide Distributionen erledigen das normalerweise schon selber. Testen kann man wie immer direkt auf der

Kommandozeile:

```
[C:\ ]$ python
ActivePython 2.2.1 Build 222 (ActiveState Corp.) based on
Python 2.2.1 (#34, Apr 15 2002, 09:51:39) [MSC 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Um den interaktiven Modus wieder zu verlassen, drücken Sie Ctrl-Z.

Wichtig: Damit das Erzeugen des echten Worddokuments funktioniert, müssen die win32com Extension installiert sein. Testen kann man auch das am schnellsten per Kommandozeile:

```
>>> import win32com
>>> print win32com
<module 'win32com' from 'C:\Python22\Lib\site-packages\win32com\__init__.pyc'>
```

3.7. Das Framework selbst

So nun ist eigentlich alles komplett, jetzt fehlt nur noch ein erstes Projekt. Sie können z.B. im Abschnitt Download direkt dieses Projekt herunter laden und mit **make all** alle Seiten und Ausgabeformate erzeugen.

Etwas einfacher und übersichtlicher ist es aber trotzdem zuerst mit dem Hello World –Projekt zu beginnen. Wie das aussieht soll im folgenden Abschnitt beschrieben werden.

3.8. OpenOffice–Writer

Dieser Bestandteil ist optional, weil man im Prinzip mit jedem beliebigen Editor ein DocBook–Quellcode–File erstellen kann. Allerdings ist es vielleicht gerade für den Anfänger interessant OpenOffice zu benutzen. Der Writer des OpenOffice hat den Vorteil, dass man praktisch WYSIWYG editieren kann, ohne auch nur zu Grunde liegende XML zu Gesicht zu bekommen. Dieser Vorteil kann aber natürlich auch zum Nachteil werden, weil die umfassende Kontrolle über alle Aspekte des DocBook–Publishings verloren geht. OpenOffice kann in der Version 1.0.1 unter <http://www.openoffice.org/> heruntergeladen werden.

3.8.1. OOo2sDbk

Um DocBook mit OpenOffice erzeugen zu können, braucht man ein Paket, welches von OpenOffice–Format nach DocBook konvertiert. Ebenso ein Paket stellt OOo2sDbk dar. Es wird im Kapitel 6 DocBook mit OpenOffice ausführlich erläutert.

Die Installation dieses Pakets ist sehr einfach. Man muss lediglich das Zip–Archiv ooo2sdbk.zip in das Verzeichnis der Python Site–Packages entpacken. Dann ist das Modul allgemein verfügbar. Wichtig ist, dass Sie das Paket von dieser Website herunterladen, da es etwas erweiterte Funktionen bietet als das Original.

Wenn alles richtig installiert ist, dann testet man entsprechend:

```
>>> import ooo2sdbk
>>> print ooo2sdbk
<module 'ooo2sdbk' from 'C:\Python22\Lib\site-packages\ooo2sdbk\__init__.pyc'>
```

[1] alternativ kann auch die 1.4.1 verwendet werden.

^[2] eine Umgebung mit der viele Unix-Programme und Tools sehr leicht auf Windows portiert werden können (siehe auch <http://www.cygwin.com/>)

Kapitel 4. Hello World–Projekt

Die Standard–Verzeichnisstruktur sieht immer so aus:

- Das Common–Verzeichnis: enthält alle Bibliotheken und Tools, die zum Erstellen des Projekts gebraucht werden. Alles hier ist im Wesentlichen erst mal immer gleich und unabhängig vom Projekteinhalt.
Anmerkung: Anmerkung
Das ist natürlich nicht ganz richtig, denn hier wird insbesondere das Aussehen der erstellten Dokumente per XSL Stylesheet festgelegt.
- Das Build–Verzeichnis: enthält die Batch–Datei zum Aufrufen von Ant: make.bat (bzw. deren Unixvariante make.sh), die Datei build.xml, die das Bauen steuert und das Verzeichnis mit den Quelldokumenten: xml.
- Das XML–Verzeichnis: enthält alle XML–Dateien, die das eigentliche Dokument oder Buch ausmachen. Im Falle von Hello World kommt man hier natürlich auch mit nur einer Datei aus.

Die Steuerdatei für Ant sieht wie folgt aus:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE project [
    <!ENTITY common "../common">
    <!ENTITY buildcommon SYSTEM "../common/build_common.xml">
]>
<project name="HelloWorld" default="html" basedir=".">(1)
    <property name="lang" value="de"/>
    <target name="postprocess"/>
    <property name="project" value="HelloWorld"/>
    <property name="javahome" value="..." />
    <property name="fileversion" value="1.0"/>(2)
    &buildcommon;
</project>
```

Was hier also eigentlich nur festgelegt wird, ist wie das Projekt heißt und welche Dokumentenart standardmäßig erzeugt werden.

- (1) Definiert Projektname und Standard–Target (hier html).
- (2) Definiert eine Eigenschaft des Projekts. Hier der Projektname, was in der gemeinsamen Steuerdatei für Ant build_common.xml als Variable gebraucht wird.

Was jetzt noch fehlt ist der Inhalt in diesem Fall die Datei HelloWorld.xml und zwar im XML–Verzeichnis.

Für das allereinfachste HelloWorld steht hier nicht mehr als:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
"../../common/docbook/dtd/docbookx.dtd">
<article lang="de">
    <articleinfo>
        <title>Hello World</title>
        <author>
            <firstname>Stefan</firstname>
            <surname>Rinke</surname>
        </author>
    </articleinfo>
    <section>
        <title>Hello World</title>
        <para>Das ist alles. </para>
    </section>
```

</article>

Nun ist endgültig alles beisammen, was man zum ersten Testen braucht. Nachdem alle Tools installiert sind, sollte jetzt das Aufbauen des ersten Projekts kein Problem mehr sein. Wenn anschließend der Aufruf von

```
make all
BUILD SUCCESSFUL
Total time: 11 seconds
```

liefert. Dann sollten im dist-Verzeichnis unterhalb von build alle Formate des soeben erzeugten HelloWorld-Dokuments zu finden sein.

Kapitel 5. Kleines Tutorial

5.1. Book oder Article

Die nachfolgenden Ausführungen sind größtenteils an das Referenzwerk von Norman Walsh & und Leonard Mueller angelehnt (siehe hier).

Ein Buch ist typischerweise ein etwas größeres Werk als ein kleiner Artikel. Es umfasst von seiner Struktur einige Meta-Informationen (<bookinfo>), die den Autor, den Titel und einen Copyrightvermerk beschreiben. Es kann ein Vorwort haben und besteht aus Kapiteln, die wiederum in Abschnitte gegliedert sind. Zusätzlich können Anhänge folgen, sowie ein Literaturverzeichnis, ein Glossar und ein Index.

Ein Artikel ist die kleinere Ausgabe, der eben nicht soviel umfasst wie ein komplettes Buch, sondern im wesentlichen die Struktur eines einzelnen Kapitels hat. Er enthält aber genauso Meta-Informationen (<articleinfo>) und kann auch Anhänge, Literaturverzeichnis, Glossar und Index umfassen.

5.2. Ein einfaches DocBook Buch

Der XML Ausschnitt unten zeigt die typische Struktur eines DocBook Buches.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
"./../common/docbook/dtd/docbookx.dtd">
<book lang="de">
  <title>Mein erstes DocBook Buch</title>
  <bookinfo>(1)
    <author>
      <honorific>Dipl.-Ing</honorific>
      <firstname>Stefan</firstname>
      <surname>Rinke</surname>
    </author>
  </bookinfo>
  <preface>(2)
    <title>Vorwort</title>
    <para>In einem Vorwort stehen Dinge über das Buch an sich.</para>
  </preface>
  <chapter>(3)
    <title>Einführung</title>
    <para>Dies ist ein einfacher Artikel
      <indexterm>(4)
        <primary>Artikel</primary>
        <secondary>DocBook-Artikel</secondary>
      </indexterm>.
      Das ist nur ein Beispielabsatz.</para>
    </chapter>
  <appendix>(5)
    <title>Anhang 1</title>
    <para>Das ist ein Anhang.</para>
  </appendix>
  <glossary>(6)
    <glossdiv>
      <title>Ein Glossar</title>
      <glossentry>
        <glossterm id="DocBook">DocBook</glossterm>
        <glosssee>DocBook ist ein Standard zur Erstellung...</glosssee>
      </glossentry>
    </glossdiv>
```

```

</glossary>
<index/>(7)
</book>

```

- (1) Der Bookinfo-Teil ist eigentlich optional, allerdings wird man in den meisten Fällen als Autor nicht unerwähnt bleiben wollen.
 - (2) Auch das Vorwort muß nicht unbedingt dabei sein. Wenn das Werk etwas größer ausfällt, dann hat das Vorwort jedenfalls mindestens diese Struktur.
 - (3) Hier folgen nun ein oder mehrere Kapitel, die wiederum mit einem Titel `<title>` beginnen und in Absätze `<para>` gegliedert sind.
 - (4) Auf diese Weise erzeugt man einen Indexeintrag. Der Zweit-Eintrag ist optional.
 - (5) Nun folgen noch ein oder mehrere Anhänge.
 - (6) Wer möchte kann noch ein Glossar anfügen.
 - (7) Diese Tag markiert nur die Stelle, an der der fertige Index eingefügt werden soll.
- Wie ein solches Buch in HTML aussehen kann ^[3], sieht man hier.

5.3. Ein Artikel

Ganz entsprechend stellt sich ein einfacher Artikel dar.

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
"../../common/docbook/dtd/docbookx.dtd">
<article lang="de">
  <title>Mein erster Artikel</title>
  <articleinfo>
    <author>
      <honorific>Dipl.-Ing</honorific>
      <firstname>Stefan</firstname>
      <surname>Rinke</surname>
    </author>
  </articleinfo>
  <para>Dies ist ein einfacher Artikel. Dieser Ausdruck
    <indexterm>
      <primary>Ausdruck</primary>
    </indexterm>
    soll im Index erscheinen.</para>
  <appendix>
    <title>Anhang 1</title>
    <para>Das ist ein Anhang.</para>
  </appendix>
  <glossary>
    <glossdiv>
      <title>Ein Glossar</title>
      <glossentry>
        <glossterm id="DocBook">DocBook</glossterm>
        <glosssee>DocBook ist ein Standard zur Erstellung...</glosssee>
      </glossentry>
    </glossdiv>
  </glossary>
  <index/>
</article>

```

Auch wie das aussieht, kann man sich direkt online ansehen.

5.4. Online Ressourcen

Wer in das Thema DocBook tiefer einsteigen möchte, der sei an dieser Stelle auf einige andere Online Ressourcen verwiesen.

5.4.1. DocBook: The Definitive Guide

Da ist zunächst das Standardwerk: DocBook: The Definitive Guide man kann es direkt online lesen und es liefert wirklich alles vom Einstieg bis zur Referenz.

DocBook: The definitive Guide, Norman Walsh & Leonard Mueller

5.4.2. Tutorial von Stephan Wiesner

Ein sehr gutes Tutorial hat Stephan Wiesner geschrieben. Man kann es online nachlesen.

5.4.3. Online Tutorial von Lars Trieloff

Ein sehr gutes Tutorial von Lars Trieloff kann man unter <http://trieloff.net/doctutorial/build/index.html> nachlesen.

^[3] das ist natürlich von den definierten Stylesheets abhängig

Kapitel 6. DocBook Publishing mit OpenOffice

Eine sehr einfache Möglichkeit mit DocBook zu arbeiten ist die Verwendung von OpenOffice als Editor. OpenOffice verwendet von Hause aus XML als Dateiformat, daher ist es naheliegend an eine Konversion von OpenOffice Writer Dateien zum DocBook zu denken. In der Tat ist der Weg relativ leicht. Folgende Schritte sind notwendig:

- Entpacken der OpenOffice Datei
- Zusammenfügen der einzelnen XML-Dateien zu einer großen, flachen XML-Datei
- Eingebettete Bilder auslagern
- Die große XML-Datei mit Hilfe eines Stylesheets zu DocBook zu transformieren

Damit besser verständlich ist, wie das genau vor sich geht, soll der Vorgang kurz erläutert werden.

OpenOffice speichert seine Dokumente in Dateien mit der Endung swx. Diese Dateien sind eigentlich ZIP-Archive^[4], die mehrere XML-Dateien enthalten. Man kann die einzelnen Dateien also leicht extrahieren und neu zusammenstellen.

Um am Ende besser mit XSLT transformieren zu können, packt man nun die einzelnen XML-Dateien zu einer großen zusammen, wobei die verschiedenen Teilebereiche weiterhin durch Namespaces getrennt bleiben.

Bilder werden von OpenOffice direkt in die XML-Datei eingebettet und als Base64 in der XML-Datei gespeichert. Leider können die üblichen DocBook Stylesheets mit so eingebetteten Bildern nichts anfangen sondern erwarten, dass Bilder extern zur XML-Datei gespeichert werden.

All diese Schritte werden von einem kleinen Python-Modul von Eric Bellot erledigt. Es kann entweder hier bezogen werden, ist aber auch Teil des dieses Frameworks selbst.

Die anschließende Transformation wird mit Hilfe eines speziellen Stylesheets mit einem beliebigen XSLT-Prozessor (hier saxon) durchgeführt. Das Stylesheet von Eric Bellot wurde für dieses Framework noch etwas verbessert und es existieren mehrere Varianten davon, je nach dem, ob man ein `Book` oder einen `Article` schreiben möchte.

6.1. Zusammenspiel zwischen OpenOffice und DocBook

OpenOffice verwendet für Formatierungen Absatzformate, Zeichenformate und Rahmenformate. Diese sind benannt und an bestimmte Layoutregeln geknüpft. Die Verbindung zwischen DocBook und OpenOffice wird genau über diese vordefinierten Formate hergestellt. Wenn ein Satz beispielweise einen `Filename` enthält, dann wird er in OpenOffice mit dem Zeichenformat `Filename` formatiert und später in `<filename>...</filename>` konvertiert.

Mit Absatzformaten für Aufzählungen oder nummerierte Listen verhält es sich genauso, in OpenOffice werden diese wie gewohnt verwendet und dann automatisch in die DocBook Entsprechungen umgesetzt. Damit das immer richtig funktioniert, sollte man allerdings immer eine bestimmte Dokumentvorlage bzw. ein leeres Startdokument verwenden. Ein solches Dokument ist dem Framework beigelegt.

Die meisten korrespondierenden Formate / Tags sind intuitiv, die ausführlichere Darstellung liefert Eric Bellot auf seiner Seite.

6.2. Weiterführende Funktionen

6.2.1. Book oder Article

Wie schon erwähnt gibt es zwei verschiedene Stylesheets bzw. ein parametrierbares, um wahlweise ein `Book` oder einen `Article` erzeugen zu können. Das wird über die `Build-Property doc-type` in der Steuerdatei festgelegt.

6.2.2. CalloutList

Um sogenannte `CalloutLists` erzeugen zu können, wurde ein zusätzliches Aufzählungsformat mit Namen `CalloutList` aufgenommen.

Referenzen werden mit Querverweisen erzeugt. Wichtig hierbei ist, dass die Namen der Textmarken mit `co.` beginnen. Wenn diese Konventionen eingehalten werden, erzeugt das Stylesheet automatisch die richtigen `<co id= co.test >` und `<callout arearefs= co.test >`. Wie das genau funktioniert, kann man sich am besten in diesem Dokument ansehen.

6.2.3. Notiz einfügen

Wenn man mit OpenOffice eine Notiz einfügt, wird diese einfach wörtlich ins DocBook Dokument übernommen. Man kann damit beliebige DocBook-Tags erzeugen. Allerdings ist die Kontrolle über die genaue Position im Dokument nicht möglich, so dass Struktur-Tags, die an ein bestimmtes Inhaltsmodell geknüpft sind, manchmal auf diese Weise nicht sinnvoll eingefügt werden können. Was aber z.B. Problemlos möglich ist, sind Indexeinträge.

^[4] Man kann sie tatsächlich z.B. einfach mit WinZip öffnen

Kapitel 7. Download DocBook Publishing

- PDF
- HTML
- Microsoft Word
- Windows Help
- Plaintext
- HTML (eine Datei)
- XML (DocBook)
- OpenOffice
- Build Framework (für Anwender des Frameworks)