Adobe® FrameMaker® 7.0

Using the DocBook Starter Kit Online Manual

Using the DocBook Starter Kit

The DocBook DTD is a commonly used DTD that defines markup for technical books and other computer documentation. FrameMaker® software provides a starter kit for the DocBook DTD that allows you to translate between XML documents using that DTD and FrameMaker documents. FrameMaker ships with two DocBook starter kits—one kit to use with SGML versions of DocBook, and another kit to use with XML versions. These starter kits are structure applications. You can use each application as is or modify it to suit your organization.

What the DocBook starter kit accomplishes

If your organization is typical, it uses only some elements and attributes defined for DocBook. It may modify content models—for example, by removing optional elements, making optional elements required, or providing different semantics for possible attribute values and it probably has its own formatting requirements.

For these reasons, the DocBook starter kit does not attempt to interpret all elements in the DocBook DTD. You can use the starter kit to read any valid XML or SGML DocBook document into FrameMaker. However, you may have to augment or modify the starter kit before the software can appropriately interpret and format all the DocBook elements that your organization uses.

The XML starter kit supports DocBook version 4.1.2. The SGML starter kit supports DocBook 4.1.

What's included with the DocBook starter kit

Installation of the DocBook starter kit is optional when you install FrameMaker. If you're using FrameMaker on a UNIX® system and want information on how to install it, see *Installing Frame® Product*s. If you're using FrameMaker on a Windows ® or Macintosh system, see *Getting Starte*d.

If you use FrameMaker on a Macintosh or Windows platform, the DocBook starter kit is installed in the following directories:

XML DocBook	*install_dir*/structure/xml/xdocbook
SGML DocBook	*install_dir*/structure/sgml/docbook

where **install_dir** is the location of your FrameMaker installation.

If you use FrameMaker on a UNIX platform, the DocBook starter kit is installed in the following directories:

XML DocBook	$FMHOME/fminit/*uilanguage*/structure/xml/xdocbook
SGML DocBook	$FMHOME/fminit/*uilanguage*/structure/sgml/docbook

where **$FMHOME** is the directory in which FrameMaker is installed and **uilanguage** indicates a particular user-interface language, such as **usenglish** or **ukenglish**.

The starter kit directory contains two subdirectories:

app: the components of the structure application

src: the source code for the starter kit's structure API client.

The APP directory

The **app** directory contains the files needed to deliver the XDocBook starter kit as an XML structure application. It includes the following files:

Files for XML	Files for SGML	Purpose
docbookx.dtd	docbook.dtd	The parent file for the DocBook 4.1.2 DTD
dbcentx.mod dbgenent.mod dbhierx.mod dbnotnx.mod dbpoolx.mod	dbcent.mod dbgenent.mod dbhier.mod dbnotn.mod dbpool.mod	Various modules of the DocBook 4.1.2 DTD
calstblx.dtd	calstbl.dtd	The DocBook implementation of the CALS table model
edd	edd	Corresponding FrameMaker EDD, including formatting rules for the supported elements
rules	rules	Read/write rules
structapps.fm	structapps.fm	Application definition
template	template	Structured FrameMaker template to use on import
xdocbook.css		A css file that corresponds to the formatting in the template file—this is a convenience that saves users from having to generate a CSS every time the export a DocBook document to XML
	sgmldcl	The SGML Declaration file for use with DocBook

If you installed the DocBook starter kit when you installed FrameMaker itself, the default **structapps.fm** file contains the definition of the DocBook application. If you installed the DocBook starter kit after installing the rest of the product, you need to make this application definition in the in the starter kit directory available to your end users by incorporating it into the **structapps.fm** file you give to them.

For information on creating and editing **structapps.fm**, see the online manual, the *Structured FrameMaker Developer's Guide*.

The SRC directory

The **src** directory contains the source code for the starter kit's structure API client, and the files you need to build the client at your site. The directory includes the following files:

Files for XML	Files for SGML	Purpose
export.c	export.c	Event handler for the client's export functionality
import.c	import.c	Event handler for the client's import functionality
trnslate.c	trnslate.c	Interface from the structure API to standard FDK callbacks

Depending on your platform, the **src** directory may also contain files associated with building the client such as make or project files.

Which elements are supported

The DocBook starter kit supports a subset of the DocBook DTD (4.1 or 4.1.2), with formatting appropriate to the semantics of the elements. While you can import or export a document that uses any of the elements defined in this DTD without modification of the starter kit, you may need to modify it to provide formatting or other interpretation of elements outside this subset.

The supported subset is sufficient to import and format the user's manual for the DocBook DTD. In choosing the subset of elements to support, the following goals were achieved:

Most elements intended as inline elements (text ranges) are supported. Some of these elements (various links, for example) are supported only as markers or as placeholders for unprocessed attributes.

The supported elements are as follows:

accel	alt	answer	appendixinfo
articleinfo	artpagenums	attribution	authorblurb
authorgroup	authorinitials	beginpage	bibliodiv
biblioentry	bibliographyinfo	bibliomisc	bibliomixed
bibliomset	biblioset	blockquote	bookinfo
bridgehead	callout	caption	chapterinfo
citerefentry	citetitle	classsynopsisinfo	cmdsynopsis
cmdsynopsis	collabname	colophon	computeroutput
confdates	confgroup	confnum	confsponsor
conftitle	constant	constructorsynopsis	contractnum
contractsponsor	corpauthor	corpname	dedication
destructorsynopsis	envar	errorcode	errorname
errortype	exceptionname	fieldsynopsis	firstterm
footnoteref	foreignphrase	formalpara	funcdef
funcparams	funcprototype	funcsynopsis	funcsynopsisinfo
glossaryinfo	glossdef	glossdiv	glossentry
glosslist	glosssee	glosssseealso	glossterm
graphic	guibutton	guiicon	guilabel
guimenu	guimenuitem	guisubmenu	imagedata
imageobject	indexinfo	indexterm	informalequation
informalfigure	informaltable	initializer	inlineequation
inlinegraphic	inlinemediaobject	interface	interfacename
invpartnumber	isbn	issn	issuenum
itemizedlist	itermset	jobtitle	keycap
keycode	keycombo	keysym	keyword
keywordset	legalnotice	lineannotation	listitem
literallayout	lot	manvolnum	menuchoice

methodname	methodparam	methodsynopsis	modespec
modifier	mousebutton	msgaud	msgentry
msgexplan	msginfo	msglevel	msgmain
msgorig	msgrel	msgset	msgsub
msgtext	objectinfo	olink	ooclass
ooexception	oointerface	option	optional
orderedList	orgdiv	orgname	otheraddr
othercredit	othername	pagenums	paramdef
partinfo	partintro	phrase	pob
prefaceinfo	primaryie (client)	printhistory	productname
productnumber	programlisting	prompt	pubdate
publishername	pubsnumber	qandadiv	qandaentry
qandaset	question	refclass	refdescriptor
refentry	refentryinfo	refentrytitle	referenceinfo
refmeta	refmiscinfo	refname	refnamediv
refpurpose	refsect1	refsect1info	refsect2
refsect2info	refsect3	refsect3info	refsynopsisdiv
refsynopsisdivinfo	releaseinfo	remark	returnvalue
revdescription	revhistory	revnumber	revremark
sbr	screen	screeninfo	screenshot
secondaryie	sect1info	sect2info	sect3info
sect4info	sect5info	section	sectioninfo
seealsoie	seeie	seglistitem	segmentedlist
segtitle	setindex (drop content)	setindexinfo	setinfo
sgmltag	shortaffil	shortcut	sidebarinfo
simpara	simplelist	simplemsgentry	simplesect
structfield	structname	subject	subjectset
subjectterm	substeps	synopfragment	synopfragmentref
synopsis	systemitem	table	tbody
tertiaryie	textobject	tfoot	tgroup
thead	title	titleabbrev	toc (drop content - don't export.)
userinput	varargs	variablelist	varlistentry
varname	void	volumenum	wordasword
xref			

To support these elements appropriately, the starter kit includes read/write rules and a structure API client. This support includes the following features:

ULink elements

This starter kit supports the **ULink** element for both import and export. On import, **ULink** elements are translated as text insets—FrameMaker unwraps the **ULink** and imports the resulting ASCII file. On export to XML, FrameMaker translates text insets as **ULink** elements. (The value of the **URL** attribute is assumed to be a filename.)

White space

To avoid white space at the beginning of a FrameMaker paragraph, spaces at the beginning of **Para** elements are eliminated. When a **Para** element includes a **Screen** or **LiteralLayout** element, any spaces following the child element are also discarded. Finally, each sequence of spaces is condensed to a single space, except within the **Screen** and **LiteralLayout** elements.

List bullets

The value of the **Mark** attribute for **ItemizedList** (and the corresponding **Override** attribute for **ListItem**) can be an arbitrary string in DocBook. The starter kit supports the values **Bullet**, **Dash**, **Box**, and **Check**; the EDD considers other values invalid. The structure API client tests for the expected values on import.

Indexes

IndexTerm elements can be placed throughout a DocBook document to define terms that are to be indexed and linked to the indicated position. In DocBook, **IndexTerm** has child elements to identify different fields within an index entry (**Primary**, **Secondary**, **Tertiary**, **See**, and **SeeAlso**). FrameMaker uses index markers for this purpose, with punctuation within the marker text indicating the different fields. The starter kit's structure API client maps between the DocBook structure and index markers on both import and export. For example, the marker text **chocolate:cake** maps to

```
<indexterm><primary>chocolate</primary><secondary>cake</ secondary></indexterm>
```

This mapping does not account for font changes or special characters within an index marker, or for entity references and elements nested in the content of the children of an XML **IndexTerm** element.

Graphic elements

For **Graphic** and **InlineGraphic** elements in a FrameMaker document, the **Role** attribute controls whether the element is intended for an imported graphic or a new anchored frame with contents to be created using the drawing. The EDD defines the attribute as a a required choice with possible values **AnchoredFrame**, **ImportedGraphic**, and **AsIs**. When an author inserts a new element of this type, the following occurs:

- FrameMaker displays the Attributes for New Element dialog box. This will occur if New Element Options are set to prompt for attribute values. In that case, the user provides an attribute value.
- If the user chooses **AnchoredFrame** for the **Role** attribute, FrameMaker will then display the anchored frame dialog box.
- If the user does not set the value for **Role** or chooses a value other than **AnchoredFrame**, FrameMaker displays the Insert File dialog box.

When importing XML, the starter kit uses whatever value is assigned to **Role**. If **Role** has no value, or a value other than the choices defined in the EDD, the starter kit assigns a value of **AsIs**. This use of **AsIs** preserves document validity—it's necessary because **Role** is required in the EDD.

CALS tables

CALS tables consist of a Table element that contains one or more **TGroup** elements. FrameMaker translates CALS tables differently, depending on the number of **TGroup** elements within a **Table** element.

Multiple `TGroup` elements per `Table` If the Table element contains more than one **TGroup**, then FrameMaker translates the **Table** element as a container that holds a number of FrameMaker tables. The **Title** element is a child of **Table**, and is not a part of any FrameMaker table. As a result the title only appears on one page, even if the **Table** (made up of a number of **TGroup** elements) spans more than one page.

One `TGroup` per `Table` If the **Table** element only contains one **TGroup** element, FrameMaker translates the **Table** element as a FrameMaker table object. It unwraps the **TGroup** element and imports the **TGroup** content as the body, heading, and footing of the FrameMaker table. As a result, the **Title** element becomes a part of the FrameMaker table—if the table spans multiple pages the title will appear on each page.

Choosing how to translate CALS tables The starter kit supports both approaches to translating CALS. The EDD and read/write rules document both use conditional text to control how FrameMaker translates the tables in your XML document. However, you must make a choice for the entire document. If one table in your XML has multiple **TGroup** elements, then you must use that approach for the entire document.

To choose how FrameMaker will import CALS, change the Show/Hide Condition setting for the EDD and the read/write rules. After you make the Show/Hide setting, import the EDD into your FrameMaker template document.

If your XML document has multiple **TGroup** element in any **Table**, set the documents to show text with the **Multi-Tgroup** condition (this is the default setting). If your XML never has more than one **TGroup** per **Table** element, change the settings to show text with the **OneTgroup** condition.

A structured FrameMaker document cannot use the same element tag for both a container element and a table title element. However DocBook uses the **Title** element for a title in **Table**, and a title in other elements. To use the one-**TGroup** method of translating CALS, the EDD defines **TableTitle** for the titles of FrameMaker tables. On import, it converts any **Title** that is a child of **Table** into a **TableTitle**. On export, the starter kit converts all FrameMaker **TableTitle** elements to **Title** in XML.

TOC, Index, and other generated lists

In XML, a DocBook document may include page numbers in generated lists such as **TOC**, **LOT**, **Index**, or **SetIndex**. The starter kit ignores these page numbers, based on the assumption that such page numbers in the XML will not accurately match page numbers in the new FrameMaker document. Similarly, if a FrameMaker document is exported to XML it will probably be processed by some other tool and the FrameMaker page numbers will not be relevant. For both import and export the starter kit discards the contents of these elements but preserves their location.

Which elements are dropped

The DocBook starter kituses read/write rules to drop a number of elements in the DocBook DTD (4.1 or 4.1.2). By dropping these elements, the element content is lost from the document. If you import a DocBook document into FrameMaker, then export it to XML or SGML, the exported data will not include the content that was dropped. Following is a list of elements this starter kit explicitly drops:

area	areaset	areaspec	audiodata
audioobject	calloutlist	co	graphicco
imageobjectco	label	mediaobjectco	programlistingco
screenco	videodata	videoobject	