

Oracle® Document Engineering

Content Reuse Solution Tutorial

Release 1

E56914-01

December 2014

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Deanna Bradshaw

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

1 Introduction to the Tutorial

What's in This Tutorial	1-1
Prerequisites	1-1
About the Practice Document That You Create	1-2
If You Have Questions About This Tutorial.....	1-3

2 Getting Started

Introduction to the Document Engineering Content Reuse Solution	2-1
Logging In to SDL LiveContent Architect and Arbortext Editor	2-1
The Arbortext Editor Interface.....	2-3

3 Working with Topics

Topics and Templates	3-1
Exercise: Creating Folders for the Practice Document	3-2
Exercise: Setting the Arbortext Editor Interface for This Tutorial.....	3-3
Exercise: Creating a Concept Topic.....	3-6
Exercise: Creating a Task Topic	3-8
Exercise: Creating a Reference Topic	3-10

4 Working with Text

Tags for Formatting Text	4-1
Exercise: Formatting a Graphical User Interface Control Element	4-2
Exercise: More Formatting	4-3
Exercise: Checking Spelling	4-4
Exercise: Inserting a Superscript Character	4-4

5 Working with Publications

Publications	5-1
Exercise: Creating a Publication	5-1
Exercise: Creating a Book Map	5-2
Exercise: Adding a Book Map to a Publication	5-3
Exercise: Adding Title and Copyright Page Information	5-5

6	Working with Examples	
	Informal and Formal Examples	6-1
	Exercise: Inserting a Block of Code	6-1
	Exercise: Inserting a Formal Example.....	6-4
	Exercise: Inserting a Wide Informal Example	6-6
7	Working with Tables	
	Informal and Formal Tables.....	7-1
	Exercise: Inserting an Informal Table	7-1
	Exercise: Inserting a Formal Table	7-4
8	Working with Images	
	Informal and Formal Figures	8-1
	Exercise: Adding a Screenshot to the Repository.....	8-1
	Exercise: Inserting an Informal Figure.....	8-2
	Exercise: Inserting a Formal Figure.....	8-5
	Exercise: Updating an Image	8-8
9	Working with Links	
	Links	9-1
	Exercise: Creating Cross-Topic Links and External Links.....	9-1
	Exercise: Creating Related Links	9-4
	Exercise: Creating a Link to an External Document Using a Topic ID	9-5
10	Completing the Publication	
	Publications: Chapters and Topics	10-1
	Exercise: Adding Chapters and Topics to a Publication	10-2
11	Creating Output	
	Output Formats and Publishing	11-1
	Exercise: Creating Output	11-1
A	Resetting Your SDL LiveContent Architect Password in DocBuilder	
	Logging In to DocBuilder	A-1
	Resetting Your SDL LiveContent Architect Password.....	A-1
B	Inserting Notes	
	Types of Notes Used in DITA	B-1
	Inserting Notes in DITA	B-1
C	Further Information	
	About SDL LiveContent Architect	C-1

About Short Descriptions	C-1
About Semantic Tags.....	C-3
Semantic Tags	C-3
About Figures and Images	C-5
Guidelines for Figures and Images	C-5
Attributes for Images	C-6
EPS Images	C-6
About Links	C-8
About Publications	C-10
Publication Options for Adding and Inserting	C-10
Overview of the Publication Lifecycle	C-10
Managing Topics in a Publication.....	C-11
Adding a Chapter to a Bookmap	C-11
Managing Publication Baselines	C-12
Releasing a Publication.....	C-12
About Accessibility.....	C-13
Formal Tables Accessibility Checklist	C-13
Informal Tables Accessibility Checklist	C-13
Figures Accessibility Checklist.....	C-14
Graphic Accessibility Checklist.....	C-15
About Oracle Publishing Options	C-16
Oracle Properties for Publishing Options.....	C-17
DITA Standards for Topic Types.....	C-17
DITA Standards for Task Topics.....	C-17
DITA Standards for Concept Topics	C-24
DITA Standards for Reference Topics	C-30
DITA Standards for Orientation Topics.....	C-35

Index

List of Figures

2-1	The Arbortext Editor Interface.....	2-3
3-1	Templates.....	3-2
4-1	Quick Tags Menu, Including Semantic and Typographical Tags for Formatting.....	4-2
C-1	Accessibility Elements in a Formal Table.....	0
C-2	Accessibility Elements in an Informal Table.....	0
C-3	Accessibility Elements in a Figure.....	C-15
C-4	Accessibility Elements in a Graphic.....	C-16

List of Tables

B-1	Types of Notes in DITA.....	0
C-1	Sample Reltable.....	C-8
C-2	Oracle Properties for Publishing Options.....	C-17

Introduction to the Tutorial

What's in This Tutorial

This tutorial leads you through how to create and publish a practice document using the Document Engineering **content reuse solution**, an integrated tools solution based on the Darwin Information Typing Architecture (DITA) model.

Each lesson contains an explanation followed by exercises. In the exercises, you create parts of a practice document. Work through the exercises in order. You can stop and resume the tutorial at any time. The tutorial takes about 5 hours to complete.

The exercises direct you to use values for metadata, such as **None** or **Admin**, that are appropriate only for this tutorial. Some values are determined by your assigned workflow. (Database documentation writers are assigned the database workflow. Everyone else is assigned the generic workflow.) For your own documentation, check with your documentation manager for the values that you should use for product names, functional areas, peer reviewers, functional area owners, architects, content group members, and so on.

This tutorial does not describe how to use variables and conditions, which are important for reusing content and enabling single-sourcing. These topics are described in *SDL LiveContent Architect Tutorial*, which includes exercises in which you reuse content by creating two different publications that use the same DITA map. The publications use variables for text and images, and the two publications differ by the values of their variables. You also practice how to conditionalize a step in a task topic and how to conditionalize an entire concept topic.

See Also:

[SDL LiveContent Architect Tutorial](#)

Prerequisites

Before you start this tutorial, ensure that you can access all the components of the content reuse solution.

You must:

- Install and be able to log in to Arbortext Editor
- Install and be able to log in to SDL LiveContent Architect Publication Manager
- Be able to log in to SDL LiveContent Architect web client at `https://dadvip0032.us.oracle.com/InfoShareAuthor/`
Ensure that pop-up window blocking is not selected in your browser.
- Be able to log in to DocBuilder at `https://docbuilder.us.oracle.com`

You should see **Arbortext Editor** and **SDL LiveContent Architect Publication Manager** displayed in the Windows **Start** menu or on the task bar. (In the **Start** menu, Arbortext may be listed under **PTC**.)

If you cannot access these components, then contact:

- Oracle Community Forum on CCMS and Arbortext Authoring Support at <https://community.oracle.com/community/employee/dita-authoring>.
- The help mailing list at doceng_helpdesk@oracle.com

This tutorial assumes that you are familiar with structured authoring and XML tags (elements).

See Also:

[CCMS and Arbortext Authoring Portal](#)

About the Practice Document That You Create

The practice document that you create is adapted from an old version of *Oracle B2B User's Guide*.

The practice document is abridged to fit the requirements of the tutorial. You can view the completed document as a PDF file at http://st-doc.us.oracle.com/id_common/dp/doceng/index.htm. You can also access the completed topics in the SDL repository and access the completed document (publication) in SDL LiveContent Architect Publication Manager.

The topics that you use for the lessons are in the SDL repository at the following locations:

- For the database workflow: General/Tutorials/YBBUG-Database/Practice Document-For Tutorial
- For the generic workflow: General/Tutorials/YBBUG-Generic/Practice Document-For Tutorial

The completed tutorial topics are at the following locations:

- For the database workflow: General/Tutorials/YBBUG-Database/Practice Document-Completed
- For the generic workflow: General/Tutorials/YBBUG-Generic/Practice Document-Completed

Note:

You cannot access topics that are created in a workflow different from your own.

When you are checking out a topic, if you see a message stating that someone else has already checked out the topic, then wait several minutes and try again.

If You Have Questions About This Tutorial

Ask questions at Oracle Community Forum on CCMS and Arbortext Authoring Support at <https://community.oracle.com/community/employee/dita-authoring>.

For questions or comments about this tutorial, put *AATUT*: (the tutorial doc ID) in the subject, as shown in the following figure.



CCMS and Arbortext Authoring

Overview Content People Subspaces

Watch for changes coming soon...

ASK CCMS AND ARBORTTEXT AUTHORIZING SUPPORT

AATUT: your question or comment about the tutorial

Similar questions already asked:

Search all results in CCMS and Arbortext Authoring Support →

Getting Started

Introduction to the Document Engineering Content Reuse Solution

The Document Engineering **content reuse solution** is an integrated tools solution for developing and publishing technical documentation based on the Darwin Information Typing Architecture (DITA) model.

The content reuse solution includes these components:

- Arbortext Editor: Use the editor for XML-based authoring.
- Topic templates: Use the templates for topic-based authoring.
- SDL repository: Use the SDL component content management system (CCMS) to store and share topics.
- SDL LiveContent Architect Publication Manager: Use the Publication Manager to **publish** content (create output formats).
- SDL LiveContent Architect web client: Use the web client to access the repository and update graphics. (Note that the web client is not the same as the Publication Manager.)
- DocBuilder: Use DocBuilder to create output formats and send the output formats to Oracle Review, DocArch, the id_common server, and Doc QA.

See Also:

[About SDL LiveContent Architect](#)

The architecture of SDL LiveContent Architect is shown in the figure.

[Logging In to DocBuilder](#)

Log in to DocBuilder to create output formats such as XHTML and PDF files from the XML files that you author in Arbortext.

Logging In to SDL LiveContent Architect and Arbortext Editor

Log in to SDL LiveContent Architect to access the repository and Publication Manager. Log in to Arbortext Editor to use the editor.

Your SDL LiveContent Architect user name and password are probably different from your Arbortext Editor user name and password. (Arbortext Editor uses Single Sign-On. SDL LiveContent Architect does not.) This tutorial recommends using the **Remember password** option for SDL LiveContent Architect. If you have already set the **Remember password** option for SDL LiveContent Architect, then, after you open SDL LiveContent Architect, start at Step 9. If you forget your SDL LiveContent Architect password, then reset it using DocBuilder.

If you cannot log in, send an email to doceng_helpdesk@oracle.com.

1. Open SDL LiveContent Architect Publication Manager.

An SDL LiveContent Architect Publication Manager window opens.

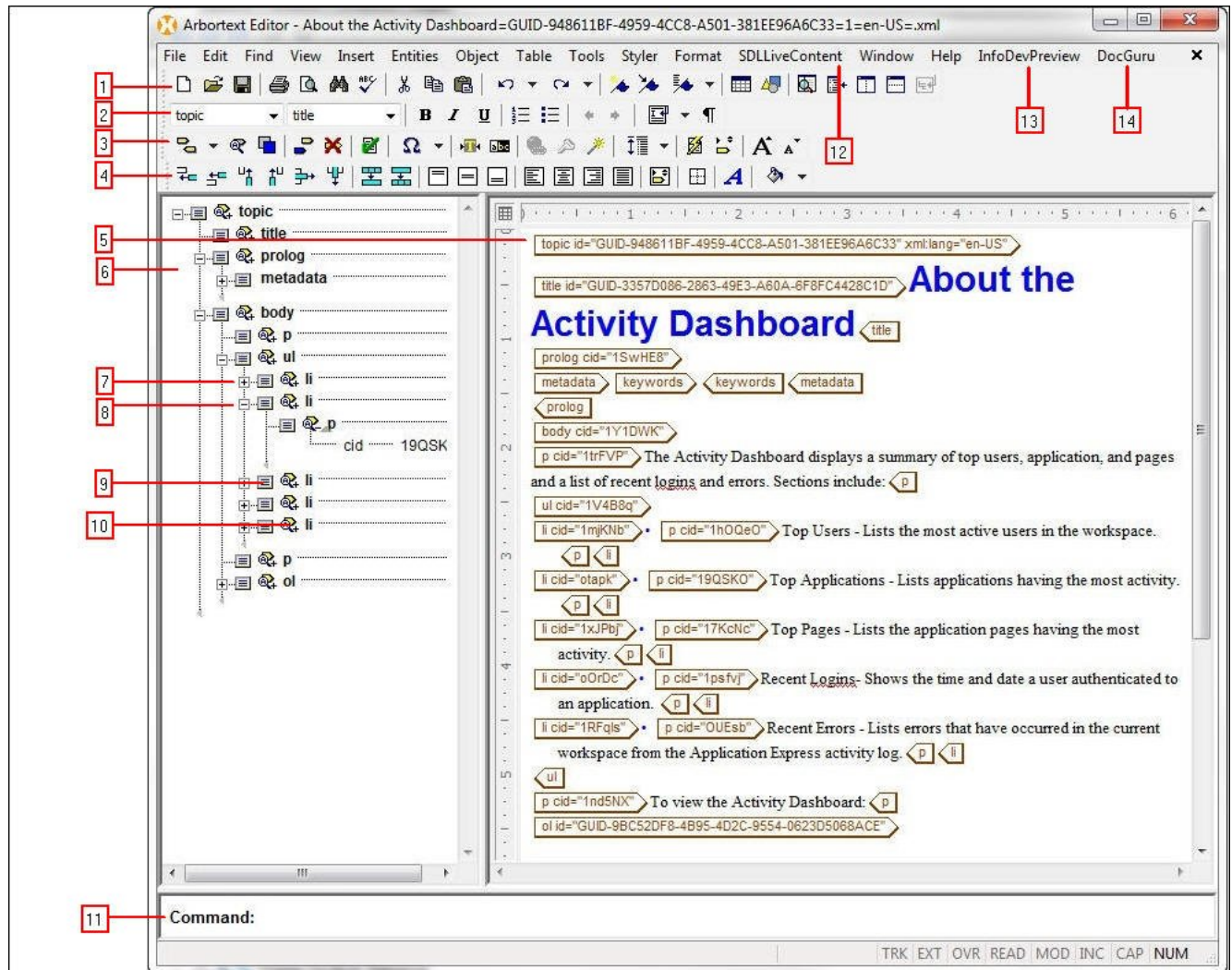
2. From the **Tools** menu, select **Accounts**.
3. If an account is not displayed in the list, then click **Add** and do the following. If an account is displayed, then go to Step 4.
 - a. In the Add Account dialog box, enter your Oracle global user ID followed by @prod (for example, jdoe@prod).
 - b. In the **SDL LiveContent Architect web service** field, enter the URL for the production server, <https://dadvip0032.us.oracle.com/InfoShareWS/>.
 - c. Click **Next**.
 - d. In the **User name** field, enter your SDL user name.
 - e. In the **Password** field, enter your SDL password.
 - f. Select **Remember password** and click **Next**.
 - g. Follow the prompts until you can click **Finish**.
4. In the Accounts dialog box, click **Properties**.
5. In the **General** tab, do the following:
 - a. In the **User name** field, enter your SDL LiveContent Architect user name if it is not already supplied.
 - b. In the **Password** field, enter your SDL LiveContent Architect password.
 - c. Select **Remember password**.
 - d. Click **Test Account**.
6. Click **OK** to close the dialog box.
7. Click **OK** to close the Account Properties dialog box.
8. Click **Close** to close the Accounts dialog box.
9. Open Arbortext Editor.
10. Log in to Arbortext using your Single Sign-On user name and password.
11. If you see the InfoDev Arbortext Extensions-Login Required dialog box, then log in again using your Single Sign-On user name and password.

A blank Arbortext Editor window opens.

The Arbortext Editor Interface

The Arbortext Editor interface is shown in the figure.

Figure 2-1 The Arbortext Editor Interface



1. Edit toolbar: A group of buttons that you use to perform editing operations such as Open, Print, and Copy.
2. Application toolbar: A group of drop-down lists and buttons that you use to format documents: bold, italic, and underlining; promote and demote elements; and convert tags to lists.
3. Markup toolbar: A group of options that you use to insert tags, modify attributes, and apply profiles.
4. Table toolbar: A group of buttons that you use to edit and format a table.
5. Edit view: A part of the Edit pane that displays the document.
6. Document map: A part of the Edit pane that displays a hierarchical outline of a document. Use it to navigate the document and select elements.

7. Expand (shown in the document map): To display the contents and child elements of an element, click the plus sign.
8. Collapse (shown in the document map): To collapse the contents of a tag, click the minus sign.
9. Tag icon (shown in the document map): An icon that represents a tag or element in the document.
10. Modify attributes icon (shown in the document map): An icon that appears when one or more attributes of an element have values that were edited.
11. Command-line bar: An area where you can enter Arbortext Command Line (ACL) commands. To display the command line, from the **Tools** menu, select the **Preferences** option and then select the **Command Line** box in the **Window** category.
12. SDLLiveContent menu: A menu that provides the Authoring Bridge features necessary to interact with the SDL server from Arbortext Editor.
13. InfoDevPreview menu: This menu is not available.
14. DocGuru menu: An InfoDev custom menu with options for converting DocBook content to DITA. (DocGuru is not used for this tutorial.)

After you create folders for the practice document (in the next exercise), you will set the Arbortext Editor interface so that your display matches what is described in this tutorial.

See Also:

[Arbortext Authoring Guide for DITA](#)

[Exercise: Setting the Arbortext Editor Interface for This Tutorial](#)

Set the Arbortext Editor interface so that your display matches what is described in this tutorial.

Working with Topics

Topics and Templates

A **topic** is the basic unit of DITA.

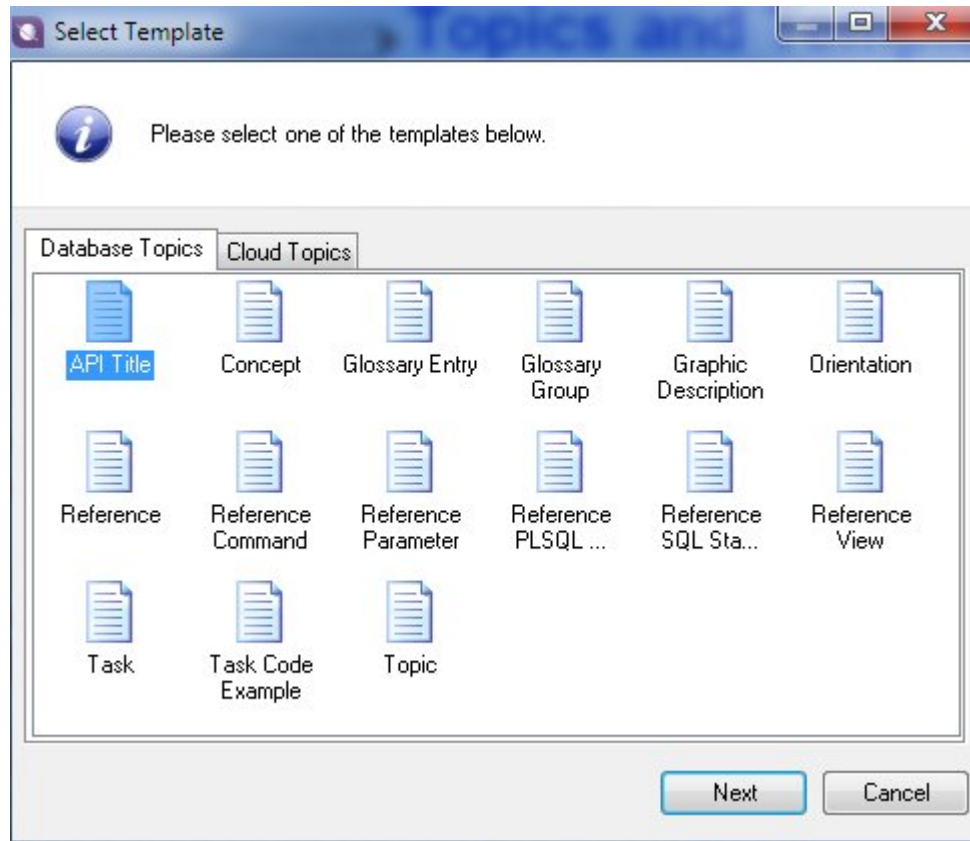
For the purpose of this tutorial, a topic is essentially the same as a *Section* in FrameMaker. A topic is called a **module** in SDL LiveContent Architect Publication Manager.

This tutorial introduces the following topics. You will arrange the topics within a hierarchy in chapters.

- Concept topic: Explains an idea that provides background for tasks or reference topics
- Task topic: Contains one sequence of steps
- Reference topic: Provides quick access to a set of facts
- Orientation topic: Serves as a topic organizer
- Graphic Description topic: Provides a description of an image for accessibility

Templates are provided for many topic types, as shown in [Figure 3-1](#).

Start with a template to create a topic. Templates for Oracle database topics and Oracle cloud topics are provided. In this tutorial, the database templates are used. Check with your documentation manager for which product area templates you should use.

Figure 3-1 *Templates*

See Also:

[DITA Standards for Topic Types](#)

All InfoDev documentation is built using only four topic types: task (<task>), concept (<concept>), reference (<reference>), and orientation (which uses <topic>). A glossary uses a different element (<glossgroup>), but is not considered a separate topic type.

Exercise: Creating Folders for the Practice Document

Create folders for the content that you will create.

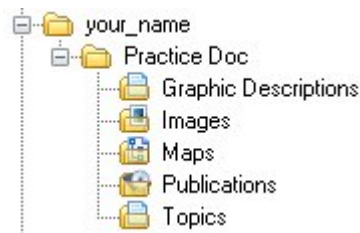
In this exercise, you will create a Graphic Descriptions folder. However, you can also place graphic descriptions in the **Topics** folder. The completed folders are shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Browse Repository**.
2. In the **Repository** tab, open the **Tutorials** folder and the *your_name* folder. If a folder with your name does not exist, create one as follows:
 - a. Right-click the **Tutorials** folder, select **New** and then **Folder**.
 - b. For the folder name, enter your first name, a space, and your last name. (Follow the pattern in the **Tutorials** folder.)
 - c. For the **Content type**, select **None**.
3. Right-click the *your_name* folder, select **New** and then **Folder**. Do the following:

- a. For the folder name, enter:
Practice Doc
 - b. For the **Content type**, select **None**.
 - c. Click **OK**.
4. Right-click the **Practice Doc** folder, select **New** and then **Folder**. Do the following:
- a. For the folder name, enter:
Topics
 - b. For the **Content type**, select **Module**.
 - c. Click **OK**.
5. Repeat Step 4 using these folder names and content types:

Folder Name	Content Type
Graphic Descriptions	Module
Images	Graphic
Maps	Master Document
Publications	Publication

The completed folders are shown in the following figure.



6. Minimize or close the Browse Repository window.

Exercise: Setting the Arbortext Editor Interface for This Tutorial

Set the Arbortext Editor interface so that your display matches what is described in this tutorial.

When you are comfortable with the Arbortext Editor interface, you can change the settings to your own preferences. If you want to see the effect of different interface settings, try other values before selecting the recommended values in the following steps. (Some of the recommended settings may also be the default settings.)

In this exercise, you will use **Check In As** to copy and move a practice topic, Set Me, from the practice repository to your own repository. Then, you will use the Set Me topic *in your repository* while you set the interface for the tutorial (not all options are available unless you have a topic open in Arbortext Editor). You will not use the Set Me topic again. The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.

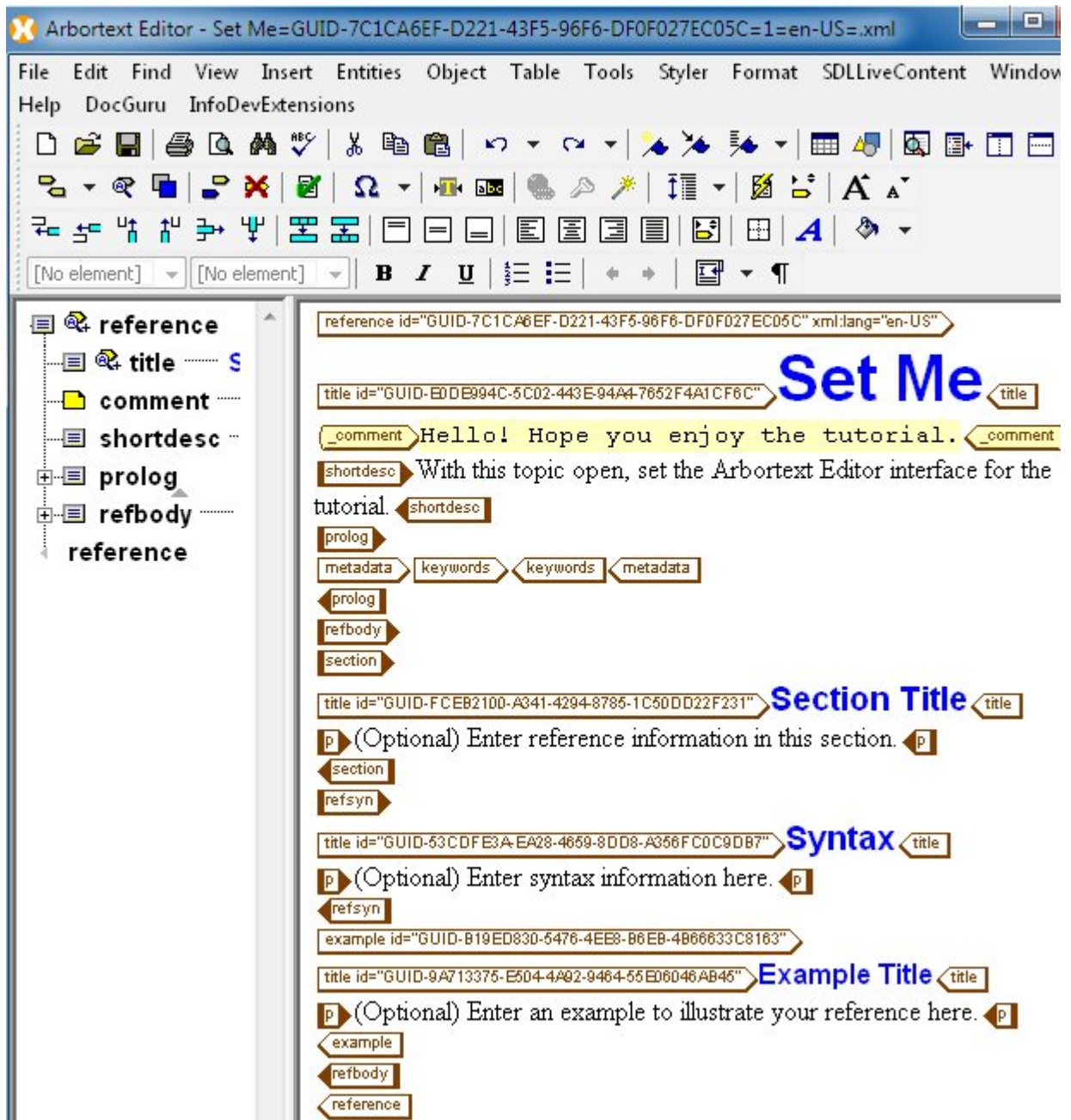
2. Open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - For Tutorial** folder, and the **Topics** folder.
3. Select the **Set Me** topic and click **Check Out**.
4. Copy and move the topic to your repository as follows:
 - a. From the **SDL LiveContent** menu, select **Check In As**.
 - b. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder, and click **Next**.
 - c. In the **General** tab, in the **Title** field, enter:

Set Me
 - d. Skip the **Version** tab.
 - e. Skip the **Workflow** tab (just for this exercise).
 - f. Ensure that **Immediately check out the object** is selected.
 - g. Click **OK**.
5. Display all menus and menu options as follows, and then click **OK**:
 - a. From the **Tools** menu, select **Preferences**.
 - b. From the **Category** options, select **Window**.
 - c. Under **Show**, select **Full Menus**. (You can select additional options to show according to your preferences.)
6. From the **View** menu, select **Full Tags**.
7. From the **View** menu, select **Toolbars**, and then select the **Edit** option. Repeat this step for each option.
8. From the **Window** menu, select **Left-Right** split.

This option is probably already set.
9. Place the cursor in the left pane and, from the **View** menu, select **Document Map**.
10. (Optional) Place the cursor over the divider between the two panes and adjust the size of the panes.
11. (Optional) Adjust the font size and tag size as follows:
 - a. Place the cursor in the right pane and, from the **View** menu, select **Font Size**, and then **Increase**.
 - b. From the **Tools** menu, select **Preferences**.
 - c. Click the **Advanced** button.
 - d. Scroll to **tagfontpercent**, select it, and click **Edit**.
 - e. In the **Value** field, enter 90 (or whatever value you prefer) and click **OK**.
 - f. Close the Advanced Properties dialog box.

g. Click **OK** to close the Preferences dialog box.

The topic is shown in the following figure.



12. From the **SDL LiveContent** menu, select **Check In** and then click **OK**.

The Set Me topic is now in the `Tutorials\your_name\Practice Doc\Topics` folder. Your Arbortext interface settings are saved for any topics that you open.

Exercise: Creating a Concept Topic

Create a concept topic about payload security.

In this exercise, you will also create an unordered (bulleted) list and check for completeness. Checking for completeness ensures that all the required components are contained in the topic. The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **New**.
2. In the **Database Topics** tab, select **Concept** and click **Next**.
3. In the **Repository** tab, open **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
4. Select the **Topics** folder and click **Next**.
5. In the **General** tab, in the **Title** field, enter:
`Payload Security`
6. Select the text that you just entered, and copy it into the paste buffer.
7. Skip the **Version** tab. In this tutorial, we will work with the default version number, 1.
8. In the **Workflow** tab, do the following:
 - a. From the **Status** field, select **30-Being Written** (for the database workflow) or **Generic-In progress** (for the generic workflow).
 - b. From the **Author** field, select your name.
 - c. From the **Functional Area Owner** field, select **Admin**.
 - d. From the **Architect** field, select **None**.
 - e. Leave the **Editor** and **Manager** fields set to **None**.
 - f. Ensure that **Immediately check out the object** is selected and click **OK**.
9. Select **Concept Title** and paste in `Payload Security`.
10. After the opening `shortdesc` tag, enter:
`Oracle B2B supports payload security by restricting access based on document type.`
11. Delete the opening and closing `draft-comment` tags and the boilerplate text.
12. After the required opening `p` tag, delete the boilerplate text, and enter:
`Document-type access is restricted based on the following user permissions:`
13. With the cursor after the closing `p` tag, add an unordered (bulleted) list as follows:
 - a. Press **Enter** and select **ul**.
 - b. With the cursor after the opening `p` tag, enter:

Admin permission for all document types

- c. With the cursor after the closing `li` tag, press **Enter**, select **li**, and enter:

Admin permission for specified document types

- d. (Optional) Add two more items to the list. Enter:

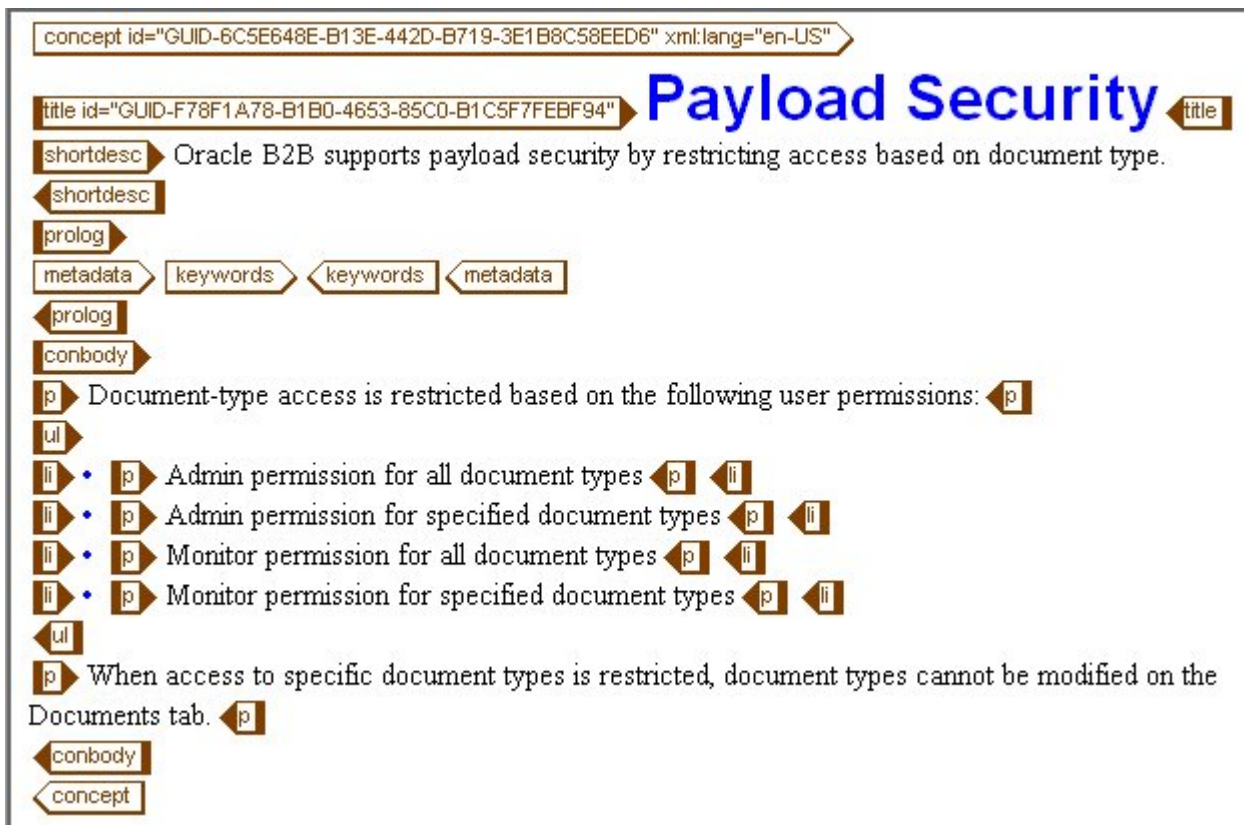
- Monitor permission for all document types
- Monitor permission for specified document types

14. With the cursor after the closing `ul` tag, press **Enter**, select **p**, and enter:

When access to specific document types is restricted, document types cannot be modified on the Documents tab.

15. Delete the opening and closing section and example tags and the boilerplate text.

The completed topic is shown in the following figure.



16. From the **Tools** menu, select **Check Completeness**.

The message *No completeness errors found* appears in the lower left corner.

17. From the **SDL Live Content** menu, select **Check In** and click **OK**.

See Also:

[About Short Descriptions](#)

A short description (<shortdesc>) is a mandatory element that summarizes a topic and explains its purpose. The descriptions appear in links and in search engine results. A short description gives just enough information so that users can determine whether they need to read the topic itself.

Exercise: Creating a Task Topic

Create a task topic about finding port information.

In this exercise, you will also create an ordered (numbered) list. The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **New**.
2. In the **Database Topics** tab, select **Task** and click **Next**.
3. In the **Repository** tab, open **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
4. Select the **Topics** folder and click **Next**.
5. In the **General** tab, in the **Title** field, enter:

`Finding Port Information`
6. Select the text that you just entered, and copy it into the paste buffer.
7. Skip the **Version** tab. In this tutorial, we will work with the default version number, 1.
8. In the **Workflow** tab, do the following:
 - a. From the **Status** field, select **30-Being Written** (for the database workflow) or **Generic-In progress** (for the generic workflow).
 - b. From the **Author** field, select your name.
 - c. From the **Functional Area Owner** field, select **Admin**.
 - d. From the **Architect** field, select **None**.
 - e. Leave the **Editor** and **Manager** fields set to **None**.
 - f. Ensure that **Immediately check out the object** is selected and click **OK**.
9. Select **Task Title** and paste in `Finding Port Information`.
10. With the cursor after the opening `shortdesc` tag, enter:

`Find port information on the Oracle Administration Console.`
11. After the opening `draft-comment` tag, delete the boilerplate text and enter a note to yourself, such as `Check on step 2`.
12. Delete the following opening and closing tags and the boilerplate text, which are not needed for this exercise: `context`, `prereq`, and `_comment`.

Notice that the template provides tags for entering steps.

- 13.** With the cursor after the first opening `cmd` tag, delete the boilerplate text and enter:

`Log in to the Oracle Administration Console.`

- 14.** Below Step 1, delete the opening and closing `stepresult` tags and the boilerplate text.

- 15.** With the cursor after the second opening `cmd` tag, delete the boilerplate text, and enter:

`In the Domain Structure pane, expand Environment and click Servers.`

- 16.** Below Step 2, after the opening `stepresult` tag, delete the boilerplate text and enter:

`The list of servers includes status messages.`

- 17.** With the cursor after the last closing `step` tag (not the `steps` tag), press **Enter**, select **step**, and enter:

`In the Servers table, find the port information in the Listen Port column.`

- 18.** (Optional) After the opening `result` tag, delete the boilerplate text and enter text that describes the result of the steps.

The completed topic is shown in the following figure.

task id="GUID-A803E2E8-6D98-40BF-B4CC-341CB6B62826" xml:lang="en-US"

title id="GUID-A71EF596-0A13-4F69-972E-FE83CCE161A0"

Finding Port Information

shortdesc Find port information on the Oracle Administration Console. draft-comment

Check on step 2. draft-comment shortdesc

prolog

metadata keywords keywords metadata

prolog

taskbody

steps

step 1. cmd Log in to the Oracle Administration Console. cmd step

step 2. cmd In the Domain Structure pane, expand Environment and click Servers.

cmd

stepresult The list of servers includes status messages. stepresult

step

step 3. cmd In the Servers table, find the port information for your server in the Listen Port column. cmd step

steps

result [Add an optional description of the results here.] result

taskbody

task

19. Save the topic and leave it open. You will return to this topic to apply bold to the graphical user interface elements.

Exercise: Creating a Reference Topic

Create a reference topic about supported protocols.

In this exercise, you will also create an index term, an informal table with reference information, and an unordered (bulleted) list in a table cell. The completed table is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **New**.
2. In the **Database Topics** tab, select **Reference** and click **Next**.
3. In the **Repository** tab, open **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
4. Select the **Topics** folder and click **Next**.
5. In the **General** tab, in the **Title** field, enter:

Supported Protocols

6. Select the text that you just entered, and copy it into the paste buffer.
7. Skip the **Version** tab. In this tutorial, we will work with the default version number, 1.
8. In the **Workflow** tab, do the following:
 - a. From the **Status** field, select **30-Being Written** (for the database workflow) or **Generic-In progress** (for the generic workflow).
 - b. From the **Author** field, select your name.
 - c. From the **Functional Area Owner** field, select **Admin**.
 - d. From the **Architect** field, select **None**.
 - e. Leave the **Editor** and **Manager** fields set to **None**.
 - f. Ensure that **Immediately check out the object** is selected and click **OK**.

9. Select **Reference Title** and paste in Supported Protocols.

10. With the cursor after the opening shortdesc tag, enter:

Oracle B2B supports industry-standard protocols in a range of industries, including health care, retail, IT, telecommunications, electronics, manufacturing, and the food industry.

11. After the opening keywords tag, press **Enter**, select **indexterm**, and enter a suitable index term, such as protocols. (The cursor is in the correct place for you to start entering text.)

12. Delete the following opening and closing tags and boilerplate text, which are not needed for this exercise: `_comment`, `draft-comment`, `section`, `refsyn`, and `example`.

13. With the cursor after the opening `refbody` tag, press **Enter**, select **table**, and do the following:

- a. For **Rows**, enter 5.
- b. For **Columns**, enter 2.
- c. Click **OK**.

14. With the cursor in the first row of the first column, from the **Table** menu, select **Convert to Header Row**.

15. Press **Enter**, select **p**, and enter:

Protocol Type

16. Press **Tab**, press **Enter**, select **p**, and enter:

Protocol Name

17. Press **Tab**, press **Enter**, select **p**, and enter:

Document

18. Press **Tab**, press **Enter**, and create an unordered list as follows:

- a. Select **ul** and enter the following between the **p** tags:

EDI EDIFACT

- b. With the cursor after the closing **li** tag, press **Enter**, select **li**, and enter the following between the **p** tags:

HL7

- c. (Optional) With the cursor after the closing **li** tag, press **Enter**, select **li**, and enter the following between the **p** tags:

RosettaNet PIP

19. With the cursor after the opening **table** tag, press **Enter**, select **desc**, and enter the follow description for accessibility:

The table contains two columns and five rows. The column with the Protocol Type heading lists the types of protocols. The column with the Protocol Name heading lists the names of the protocols that are available for each protocol type.

20. (Optional) Continue adding information to create the completed table.

The completed table is shown in the following figure.

p>Protocol Type<p>	p>Protocol Name<p>
p>Document<p>	<ul style="list-style-type: none"> • p>EDI EDIFACT<p> <ul style="list-style-type: none"> • p>HL7<p> <ul style="list-style-type: none"> • p>RosettaNet PIP<p>
p>Packaging<p>	<ul style="list-style-type: none"> • p>S/MIME 2.0<p> <ul style="list-style-type: none"> • p>SOAP<p> <ul style="list-style-type: none"> • p>XML Encrypt<p>
p>Transport<p>	<ul style="list-style-type: none"> • p>AQ<p> <ul style="list-style-type: none"> • p>SOAP<p> <ul style="list-style-type: none"> • p>SFTP<p>
p>Message Exchange<p>	<ul style="list-style-type: none"> • p>AS1-1.0, AS2-1.1<p> <ul style="list-style-type: none"> • p>MLLP-1.0<p>

21. Check in the topic.

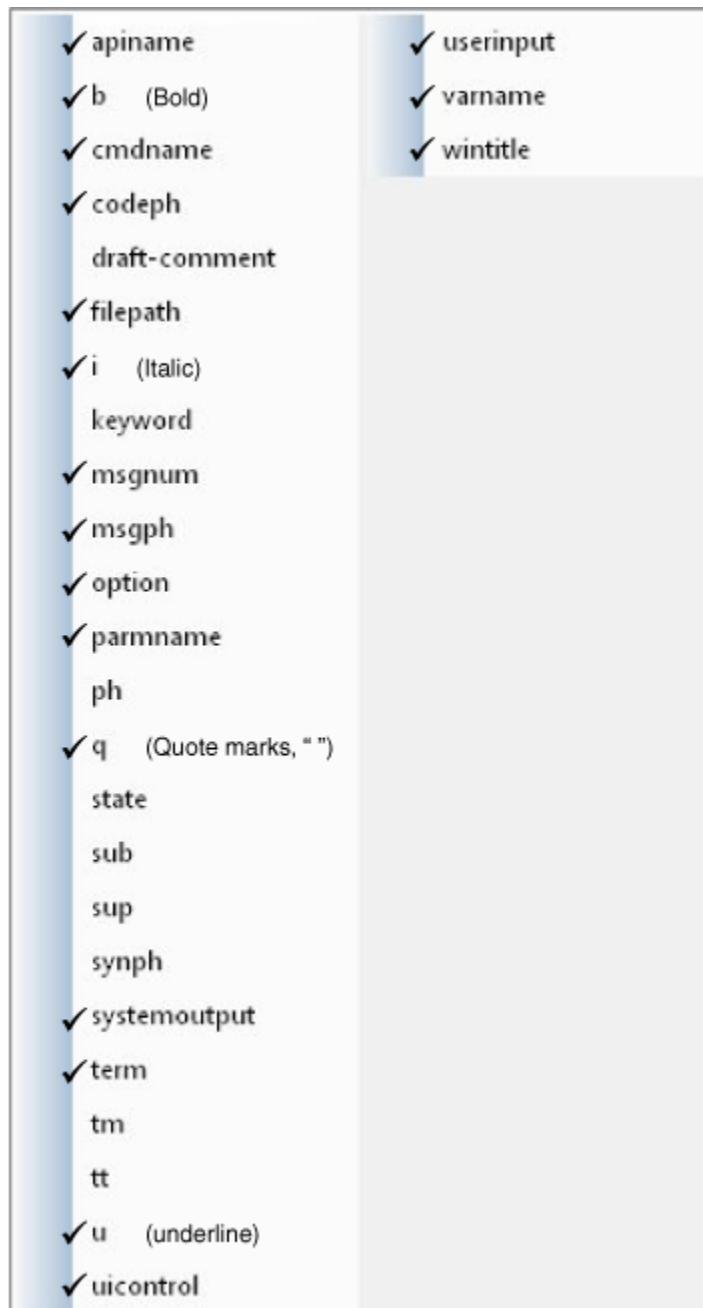
Working with Text

Tags for Formatting Text

Format text by wrapping it with a semantic tag or a typographical tag.

The `uicontrol` tag is an example of a semantic tag. It labels text as a graphical user interface control element. The `b` tag is an example of a typographical tag. It makes text bold. Use semantic tags whenever possible. [Figure 4-1](#) shows the format tag options that you see when you select text and press **Enter**, which displays the **Quick Tags** menu. The semantic and typographical tags are identified by a check mark. (The full names of the typographical tags are shown in parentheses. You can also enter **Ctrl+b** for bold, **Ctrl+i** for italic, and **Ctrl+u** for underline.)

Figure 4-1 Quick Tags Menu, Including Semantic and Typographical Tags for Formatting



See Also:

[Semantic Tags](#)

Semantic tags that you use to format inline text are described.

Exercise: Formatting a Graphical User Interface Control Element

Use the `uicontrol` tag to format a graphical user interface (GUI) element that has an action associated with it. The `uicontrol` tag formats text in bold.

1. In the Finding Port Information topic, select **Domain Structure**, press **Enter**, and select **uicontrol**.

2. (Optional) Repeat for other text that refers to a GUI element, such as **Environment**, **Servers** (twice), and **Listen Port**.

To use the Repeat Insert Markup feature (available from the **Edit** menu), press **Ctrl +Y** after selecting text.

3. Check in the topic.
4. Verify that the topic is in the *your_name* folder:
 - a. In Arbortext Editor, from the **SDL LiveContent** menu, select **Browse Repository**.
 - b. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.

You should see Finding Port Information in the list.

Exercise: More Formatting

Use semantic tags to identify the name of an API, the name of a command, and the name of a directory path.

The name of an API, the name of a command, and the name of a directory path should all appear in a monospace font. However, use a different semantic tag for each one to identify the different usages.

In this exercise, you will use **Check In As** to copy and move a practice topic from the practice repository to your own repository. Then you will use the topic in your repository to practice formatting.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. Open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - For Tutorial** folder, and the **Topics** folder.
3. Select the **Practice: Formatting** topic and click **Check Out**.
4. Copy and move the topic to your repository as follows:
 - a. From the **SDL LiveContent** menu, select **Check In As**.
 - b. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
 - c. Select the **Topics** folder and click **Next**.
 - d. In the **General** tab, in the **Title** field, enter:


```
Practice: Formatting
```
 - e. Skip the **Version** tab.
 - f. In the **Workflow** tab, complete the fields as you did in previous exercises.
 - g. Ensure that **Immediately check out the object** is selected.
 - h. Click **OK**.
5. In sentence A, select **SetCookie**, press **Enter**, and select **apiname**.

6. In sentence B, select **mkdir**, press **Enter**, and select **cmdname**.

The `cmdname` command correctly appears in monospace in PDF files, although you may see bold in the Arbortext interface.

7. In sentence C, select **/usr/ccs/bin**, press **Enter**, and select **filepath**.
8. In sentence D, change the incorrect **b** tag to the correct **uicontrol** tag as follows:
 - a. Double-click the word **Insert**
 - b. From the **Edit** menu, select **Change Markup**.
 - c. In the Change Markup dialog box, select **uicontrol** and click **OK**.
 - d. Double-click the word **Templates** and press **Ctrl+Y** (the repeat shortcut key).
 - e. (Optional) Repeat Step 7d for **Field** and **Placeholder**.
9. Save the topic and leave it open. You will return to this topic to practice inserting a superscript character.

Exercise: Checking Spelling

Check spelling as you type, or press **F7** to start the spelling checker.

The default setting in Arbortext Editor is to check spelling as you type. You can change this setting in the Preferences dialog box under the **Tools** menu. You may want to view the topic without tags displayed (under the **View** menu) if a missing space is flagged as a misspelling. Spaces can be hard to see with all the tags displayed.

1. From the **SDL Live Content** menu, select **Check Out**.
2. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder and the **Topics** folder.
3. Select the **Payload Security** topic and click **Check Out**.
4. Right-click **B2B** (which is flagged as a possible misspelled word with wavy red underlining), and select **Add to Dictionary**.
5. (Optional) Repeat the previous steps for other words that are flagged as a possible misspelled words. Use the context menu to select a suggestion, ignore the warning, or add the word to the dictionary.
6. Check in the topic.

Exercise: Inserting a Superscript Character

Use the Insert Symbol dialog box to insert symbols and special characters.

1. In the Practice: Formatting topic, place the cursor after the *c* in $E=mc$.
2. From the **Insert** menu, select **Symbol**.
3. Scroll to superscript ², select it, and click **Insert**.

The formula is now $E=mc^2$.
4. Close the Insert Symbol dialog box.

5. Check in the topic.

The Practice Formatting topic is now in the `Tutorials\your_name\Practice Doc\Topics` folder.

Working with Publications

Publications

For the purpose of this tutorial, a **publication** is essentially the same as a FrameMaker book.

The publication that you create in this tutorial is equivalent to a book because you add a *book* map as a component of the publication. (If you added a poster map or a white paper map as a component, then your publication would be a poster or a white paper instead of a book.)

Like a book, a publication with a book map contains front matter, chapters, appendixes, back matter, and so on. Just like when you created topics, you use templates to create the publication and the book map. By specifying properties for the publication and the book map, you provide information (metadata) that is needed for the title and copyright pages, such as the document title, part number, and copyright year or years.

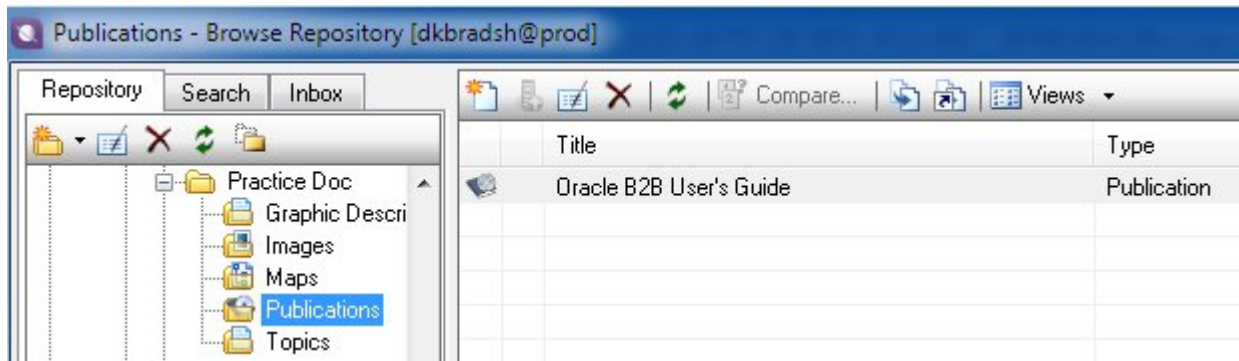
Exercise: Creating a Publication

When you create a publication, you are defining a document for publication.

The publication object is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Browse Repository**.
2. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
3. Select the **Publications** folder, and click the **New Object** (dog-eared page) icon.
4. Select **Empty Publication** and click **Next**.
5. In the Add Publication dialog box, do the following and click **OK**:
 - a. On the **General** tab, in the **Title** field, enter:
`Oracle B2B User's Guide (your name)`
 - b. From the **Publication type** list, select **User's Guide**.
 - c. On the **Version** tab, for **Product Name**, select **None**.
 - d. On the **Workflow** tab, for **Publication Owner**, select your name. For **Peer Reviewer**, select **None**.
 - e. On the **Front Matter** tab, for **Copyright Year, First Year**, enter the current year.

The Publications-Browse Repository window is shown in the following figure.



6. Right-click the **Oracle B2B User's Guide** publication, select **Properties**, and do the following:
 - a. On the **Version** tab, for the **Doc ID**, enter:
YBBUG
 - b. Click **OK**.
7. Minimize or close the Publications-Browse Repository window.

Exercise: Creating a Book Map

Create a book map, within which you will later add and arrange topics.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Browse Repository** (or maximize the repository window if it was minimized).
2. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
3. Select the **Maps** folder, and click the **New Object** icon.
4. In the Database Maps tab, select **Bookmap** and click **Next**.
5. In the Add Object dialog box, do the following:
 - a. On the **General** tab, in the **Title** field, enter:
Oracle B2B User's Guide
 - b. From the **Map type** list, select **Bookmap**.
 - c. On the **Version** tab, for **Product Name** and **Functional Area**, select **None**. Leave the rest of the fields as is or blank.
 - d. On the **Workflow** tab, from the **Status** list, select **Created** (for the database workflow) or **Generic-In progress** (for the generic workflow). For **Author** and **Publication Owner**, select your name. For **Functional Area Owner**, select **Admin**. Leave the rest of the fields set to **None**.
 - e. Click **OK**.
 - f. Click **Close**.

Exercise: Adding a Book Map to a Publication

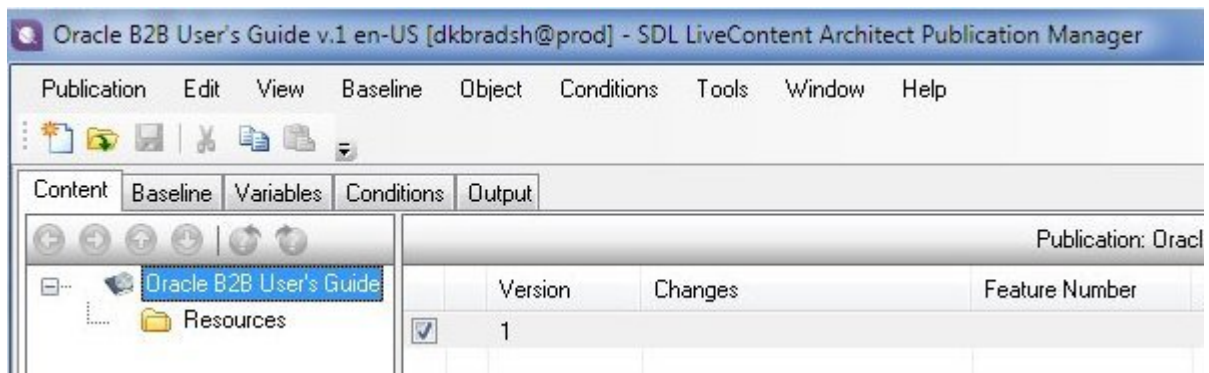
Add a book map to a publication so that the publication contains book components such as front matter (including the legal notices), chapters, and back matter.

The map object is shown at the end of the steps.

A book map, which you create in Arbortext Editor, is called a **master document** in SDL LiveContent Architect Publication Manager.

1. Open SDL LiveContent Architect Publication Manager.
2. From the **Publication** menu, select **Open**.
3. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Publications** folder.
4. Select the **Oracle B2B User's Guide** publication and click **Open**.

The SDL LiveContent Architect Publication Manager window is shown in the following figure.



5. Right-click **Oracle B2B User's Guide**, select **Add** and then **Master Document**.
6. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Maps** folder.
7. Select the **Oracle B2B User's Guide** book map and click **Add**.

The publication is shown in the following figure.

The screenshot shows the SDL LiveContent Architect Publication Manager interface. The title bar reads "Oracle B2B User's Guide v.1 en-US [dkbradsh@prod] - SDL LiveContent Architect Publication Manager". The menu bar includes "Publication", "Edit", "View", "Baseline", "Object", "Conditions", "Tools", "Window", and "Help". Below the menu is a toolbar with icons for file operations. The "Content" tab is active, showing a tree view on the left with the following structure:

- Oracle B2B User's Guide
 - Oracle B2B User's Guide
 - Book Title
 - Book Metadata
 - Front Matter
 - Chapter Chapter 1
 - Back Matter
 - Resources

On the right, a table lists the maps:

	Version	Changes	Map Type
<input checked="" type="checkbox"/>	1		Bookmap

Below the table, the content of the selected map is displayed:

Oracle® Database Book Title (platform) (optional)

Author: (primary author)
Author: (primary author)
Author: (contributing author)
Author: (contributing author)
Author: (contributor)
Author: (contributor)
[publisherinformation]:
[published]: January 2014
[bookid]:
[bookpartno]: XXXXX-00
[bookrights]:
[copyrfirst]: 2000
[copyrlast]: 2014
[bookowner]:
[organization]: Oracle and/or its affiliates

- [frontmatter Title not available]
 - licwarrantnotice_ak
 - restrictedrightslegend_ak
 - hazardnotice_ak
 - trademarknotice_ak
 - webcontentnotice_ak
 - {Condition: Beta in (external,internal)}
alphabetanotice_ak
 - {Condition: Beta=external}
revenuerecognitionnotice_ak
 - [booklists Title not available]
 - [toc Title not available]
 - [figurelist Title not available]
 - [tablelist Title not available]
 - [examplelist Title not available]

2. Chapter 1

- From the **Publication** menu, select **Save**.
- From the **Publication** menu, select **Close**.

10. Minimize SDL LiveContent Architect Publication Manager.

Exercise: Adding Title and Copyright Page Information

In Arbortext Editor, add the title and copyright page information to the book map.

In the following steps, delete any boilerplate text where you are instructed to enter text. The completed book map is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the repository, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Maps** folder.
3. Select the **Oracle B2B User's Guide** book map and click **Check Out**.
4. With the cursor after the opening `booklibrary` tag, enter the product name:

Oracle® B2B

Tip: From the **Insert** menu, select **Symbol** to insert ®.

5. With the cursor after the opening `mainbooktitle` tag, enter:

User's Guide

6. With the cursor after the opening `booktitlerelease` tag, enter:

Release 1

7. With the cursor after the opening `booktitleplatform` tag, delete the boilerplate text.
8. With the cursor after the first opening `author` tag, enter your name.
9. Delete the other opening and closing `author` tags.
10. With the cursor after the opening `month` tag, enter the current month.
11. With the cursor after the opening `year` tag, enter the current year.
12. With the cursor after the opening `bookpartno` tag, enter:

Q10229-04

13. With the cursor after the opening `year` tag that follows the opening `copyrfirst` tag, enter the current year.
14. With the cursor after the opening `year` tag that follows the opening `copyrlast` tag, delete the boilerplate text.

15. Within the `organization` tags, ensure that the text reads as follows:

Oracle and/or its affiliates. All rights reserved.

As you scroll down the file, notice that the legal notices are inserted for you.

16. Delete all the `_comment` and `glossarylist` tags, and the boilerplate text.

The completed book map is shown in the following figure.



17. Check in the book map.

Working with Examples

Informal and Formal Examples

Informal and formal examples typically use a monospaced font for text that is programming code.

Use the `example` and `codeblock` tags to create an informal example (an example that is not numbered and does not have a title). Use the `codeblock` tag, but do not wrap it in the `example` tag, for a block of code that is not an example.

Use the `example`, `title`, and `codeblock` tags to create a formal example (an example that is numbered, has a title, and can be cross-referenced). If you do not use the `codeblock` tag, then the example uses a normal font for text instead of a monospaced font for text.

Exercise: Inserting a Block of Code

Add a block of code (which is not an informal example) after a numbered step.

The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the **Repository** tab, open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - For Tutorial** folder, and the **Topics** folder.
3. Select the **Creating a weblogic User** topic, and click **Check Out**.
4. Copy and move the topic to your repository as follows:
 - a. From the **SDL LiveContent** menu, select **Check In As**.
 - b. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
 - c. Select the **Topics** folder and click **Next**.
 - d. In the **General** tab, in the **Title** field, enter:

```
Creating a weblogic User
```
 - e. Skip the **Version** tab.
 - f. In the **Workflow** tab, complete the fields as you did in previous exercises.
 - g. Select **Immediately check out the object** and click **OK**.
5. In Step 1 of the topic, note the use of the `info` tags and `codeblock` tags to wrap lines of programming code.

6. In Step 2 of the topic, with your cursor after the closing `cmd` tag, press **Enter**, select **info**, press **Enter** again, and select **codeblock**.
7. Enter the following on three lines. Press **Enter** after the first and the second line, but not after the third line.

```
dn: cn=Groups  
objectclass: top  
uniquemember: cn=weblogic
```

8. Delete the unneeded tags and the boilerplate text.

The completed topic is shown in the figure. (The figure shows an index term, which you were not instructed to create.)

task id="GUID-DD8E3799-3A23-4FB8-9625-F5FA723A2D49" xml:lang="en-US"

title id="GUID-E4745CEA-5AF7-461E-8133-08776211B385"

Creating a weblogic User

shortdesc For the weblogic user in Oracle Internet Directory (OID) to log in to Oracle B2B as an administrator and search for users, the OID Authenticator must have an Administrators group, and the weblogic user must be a member of that group. **shortdesc**

prolog

metadata **keywords** **indexterm** [\[index: weblogic user\]](#) **indexterm** **keywords** **metadata**

prolog

taskbody

context Create a weblogic user in OID using the LDAP browser. Create an Administrators group in OID to which you assign the weblogic user. **context**

steps

step 1. **cmd** Add the following to the **filepath** `users.ldif` **filepath** **file**: **cmd**

info

codeblock

```
dn: cn=weblogic,cn=Users,dc=us,dc=oracle,dc=com
objectclass: inetorgperson
objectclass: organizationalPerson
objectclass: person
objectclass: orcluser
objectclass: orcluserV2
objectclass: top
sn: weblogic
userpassword: welcome1
uid: weblogic
```

codeblock

info

step

step 2. **cmd** Add the following to the **filepath** `groups.ldif` **filepath** **file**: **cmd**

info

codeblock

```
dn: cn=Groups
objectclass: top
uniquemember: cn=weblogic
```

codeblock

info

step

steps

taskbody

task

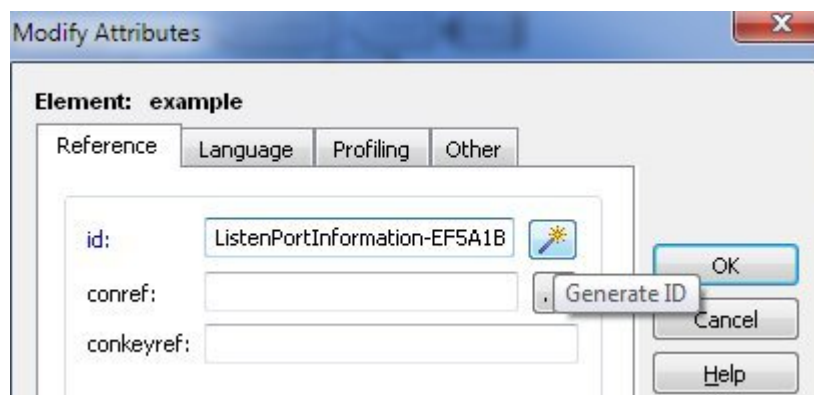
9. Check in the topic.

Exercise: Inserting a Formal Example

Insert a formal example and a title, generate an ID, and create a cross-reference to the formal example.

The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.
3. Select the **Finding Port Information** topic, and click **Check Out**.
4. With the cursor after the closing steps tag (not the step tag), press **Enter**, and select **example**.
5. Press **Enter** and select **title**.
6. With the cursor after the opening title tag, enter:
Listening Port Information
7. Create an ID for the example as follows:
 - a. With the cursor after the opening example tag, press **Ctrl+D**.
 - b. In the **Reference** tab, click the **Generate ID** icon next to the **id** field.



- c. Click **OK**.
 - d. Save the topic. (This step is required.)
8. With the cursor after the closing title tag, press **Enter**, select **p**, and enter:
You can also find listening port information for your server in the config.xml file, as shown in . (The space between *in* and the period is intentional.)
9. With the cursor after the closing p tag, press **Enter**, select **codeblock**, and enter the following. (Use the spacebar to indent.)

```
<server>
  <name>soa_server1</name>
  <listen-port>8008</listen-port>
```



```
...
</server>
```

10. With the cursor after the expression *as shown in* (followed by a space), press **Enter** and select **xref**.
11. In the Insert Hyperlink dialog box, select the **Finding Port Information** topic (the topic that is currently open) and click **Bookmark**.
12. In the Select dialog box, select the gray box labeled **example** and click **OK**.
13. In the Insert Hyperlink dialog box, delete the text in the **Text to display** field and click **Insert**.

The completed topic is shown in the following figure.

```

task id="GUID-33F55330-385C-4027-ABBB-F976A00410DF" xml:lang="en-US"
title id="GUID-A71EF596-0A13-4F69-972E-FE83CCE161A0" Finding Port Information title
shortdesc Find port information on the Oracle Administration Console. draft-comment Check on step 2. draft-comment
shortdesc
prolog
metadata keywords indexterm [index: port information] indexterm indexterm [index: Oracle Administration Console]
indexterm indexterm [index: Oracle WLS administration] index-see Oracle Administration Console index-see indexterm
keywords metadata
prolog
taskbody
steps
step 1. cmd Log in to the Oracle Administration Console. cmd step
step 2. cmd In the uicontrol Domain Structure uicontrol pane, expand uicontrol Environment uicontrol and click
uicontrol Servers uicontrol cmd step
step 3. cmd In the uicontrol Servers uicontrol table, find the port information for your server in the uicontrol Listen
Port uicontrol column. cmd step
steps
example id="example-142-D69C0A37"
title id="GUID-04B10C2B-068A-423E-9446-FBD8EEFF704E" Listening Port Information title
p You can also find listening port information for your server in the filepath config.xml filepath file, as shown in
xref href="GUID-33F55330-385C-4027-ABBB-F976A00410DF#GUID-33F55330-385C-4027-ABBB-F976A00410DF/example-142-D69C0A37"
xref p
codeblock
<server>
  <name>soa_server1</name>
  <listen-port>8008</listen-port>
  ...
</server>
codeblock
example
taskbody
task

```

14. Check in the topic.

Exercise: Inserting a Wide Informal Example

Insert an informal example that spans the margins.

You will see the wide format when you create the output. The completed topic is shown at the end of the steps.

1. Open a **Reference** template and create a topic named Tablespace Size that you check in to the `Tutorials\your_name\Practice Doc\Topics` repository.

2. After the opening title tag, enter:

Tablespace Size

3. With the cursor after the opening shortdesc tag, enter:

If you store a configuration larger than 150 megabytes, then extend or add a data file to increase the tablespace size.

4. With the cursor after the opening refbody tag, press **Enter** and select **example**.
5. With the cursor after the opening example tag, press **Enter** and select **codeblock**.
6. With the cursor after the opening codeblock tag, press **Ctrl+D** and do the following:
 - a. In the **Other** tab, from the **expanse** field, select **page**.
 - b. Click **OK**.

7. After the opening codeblock `expanse="page"` tag, enter:

```
ALTER TABLESPACE sh_mds add DATAFILE 'sh_mds01.DBF' SIZE 100M
autoextend on next 10M maxsize unlimited;
```

8. Press **Enter** and enter:

```
ALTER TABLESPACE sh_ias_temp add TEMPFILE 'sh_ias_temp01.DBF'
SIZE 100M autoextend on next 10M maxsize unlimited;
```

9. Delete the `_comment`, `draft-comment`, `section`, `refsyn`, and `example` tags and boilerplate text.

The completed topic is shown in the following figure.

reference id="GUID-88794816-E95B-4C7B-8201-4F13913D8D7F" xml:lang="en-US"

title id="GUID-C8BD9E75-007C-405C-89BB-3F9F56563269"

Tablespace Size

title

shortdesc If you store a configuration larger than 150 megabytes, then extend or add a data file to increase the tablespace size. **shortdesc**

prolog

metadata

keywords

keywords

metadata

prolog**refbody**

example

codeblock expand="page"

```
ALTER TABLESPACE sh_mds add DATAFILE 'sh_mds01.DBF' SIZE 10
0M autoextend on next 10M maxsize unlimited;
```

```
ALTER TABLESPACE sh_ias_temp add TEMPFILE 'sh_ias_temp01.DB
F' SIZE 100M autoextend on next 10M maxsize unlimited;
```

codeblock

example

refbody

reference

10. Check in the topic.

Working with Tables

Informal and Formal Tables

Use the `table` tag to insert informal and formal tables into a topic.

Informal tables are not numbered and do not have titles. Formal tables are numbered, have titles, and can be cross-referenced.

Tables must meet accessibility requirements. Use the `desc` tag to include a short description about the organization of the table (rows and columns) and its contents.

See Also:

[Inserting Notes](#)

Use the `note` tag to insert a note. (Notes are not related to tables.)

Exercise: Inserting an Informal Table

Insert an informal table within a step.

In this exercise, you will also add an unordered list within a step and create a cross-reference to a step number. The completed topic is shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the Repository tab, open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - For Tutorial** folder, and the **Topics** folder.
3. Select the **Adding a B2B Binding Component** topic, and click **Check Out**.
4. Copy and move the topic to your repository as follows:
 - a. From the **SDL LiveContent** menu, select **Check In As**.
 - b. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder, and click **Next**.
 - c. In the **General** tab, in the **Title** field, enter:

Adding a B2B Binding Component
 - d. Skip the **Version** tab.
 - e. In the **Workflow** tab, complete the fields as you did in previous exercises.
 - f. Select **Immediately check out the object** and click **OK**.
5. Add an unordered list within a step as follows:

- a. In Step 2 of the topic, with the cursor after the closing `cmd` tag, press **Enter**, select **info**, press **Enter** again, and select **ul**.
- b. With the cursor after the opening `p` tag, enter:

```
Select Exposed Services for receiving inbound messages.
```
- c. With the cursor after the closing `li` tag, press **Enter** and select **li**.
- d. With the cursor after the opening `p` tag, enter:

```
Select External References for sending outbound messages.
```
6. Add an informal table within a step as follows:
 - a. In Step 5 of the topic, with the cursor after the closing `cmd` tag, press **Enter**, select **info**, press **Enter** again, and select **table**.
 - b. In the Insert Table dialog box, enter information to create 4 rows and 2 columns, and click **OK**.
 - c. With the cursor in the first row of the first column, from the **Table** menu, select **Convert to Header Row**.
 - d. Add content to the table so that it looks like the table in the figure. (You can enter as little or as much content as you want.)
7. Add a cross-reference to a step number as follows:
 - a. In Step 5 of the topic, with the cursor after the opening `step` tag, click the **Generate ID** icon in the markup toolbar. (You can also press **Ctrl+D** and use the Modify Attributes dialog box.)
 - b. Save the topic. (This step is required.)
 - c. In Step 7 of the topic, with the cursor after the word *Step* followed by a space, press **Enter** and select **xref**.
 - d. In the Insert Hyperlink dialog box, from your **Topics** directory, select the **Adding a B2B Binding Component** topic (the topic that is currently open), and click **Bookmark**.
 - e. In the Select dialog box, select the gray box labeled **step** and click **OK**.
 - f. In the Insert Hyperlink dialog box, delete the text in the **Text to display** field, and click **Insert**.

The completed topic is shown in the following figure. (The figure shows an incomplete `desc` element, which you were not instructed to create.)

task id="GUID-B0779F09-0508-4671-8E52-B01E00D66215" xml:lang="en-US"

title id="GUID-B817837A-8A70-4806-9C05-3709FFC470CB"

Adding a B2B Binding Component

shortdesc Add a service or a reference binding component. shortdesc

prolog

metadata keywords indexterm [index: binding components indexterm], adding [indexterm] [indexterm] [indexterm] [index: adding [indexterm], binding components [indexterm] [indexterm] keywords metadata

prolog

taskbody

steps

step 1. cmd From the uicontrol Component Palette uicontrol, select uicontrol SOA uicontrol. cmd step

step 2. cmd Drag uicontrol B2B uicontrol to the uicontrol Exposed Services uicontrol or the uicontrol

External References uicontrol swim lane. cmd

info

ul

li • p Select uicontrol Exposed Services uicontrol for receiving inbound messages. p s

li • p Select uicontrol External References uicontrol for sending outbound messages. p li

ul

info

step

step 3. cmd On the B2B Configuration Wizard Welcome page, click uicontrol Next uicontrol. cmd step

step 4. cmd On the Service Name page, enter a name for the B2B service and click uicontrol Next uicontrol. cmd step

step id="OnTheB2BIntegrationTypePageSelectAn-F40050F3" 5. cmd On the B2B Integration Type page, select an integration

type: cmd

info

table

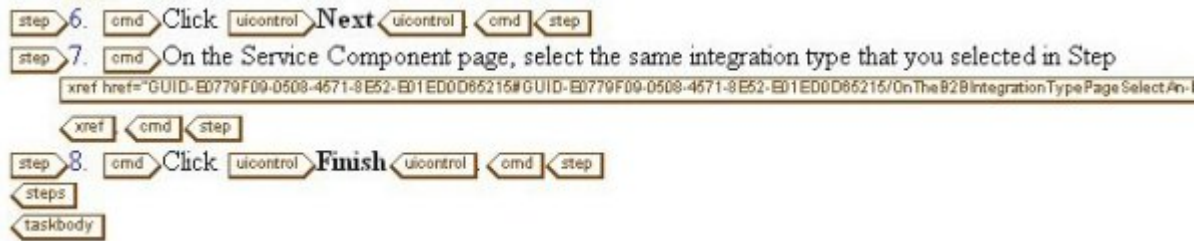
deso The table has 4 rows and 2 columns. K deso

Type	Description
Default	A B2B WSDL is generated for the SOA composite to communicate with Oracle B2B directly.
AQ	An AQ Adapter WSDL and JCA file are generated for the SOA composite to communicate with Oracle B2B through AQ queues.
JMS	A JMS Adapter WSDL and JCA file are generated for the SOA composite to communicate with Oracle B2B through JMS queues.

table

info

step



8. Check in the topic.

Exercise: Inserting a Formal Table

Insert a formal table in a concept topic.

In this exercise, you will also add a table title and a description of the table (for accessibility), generate an ID, and create a cross-reference to the formal table. When you enter the number of rows needed in the table, calculate the total number of rows, including the header row.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the **Repository** tab, open **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - For Tutorial** folder, and the **Topics** folder.
3. Select the **Document Definitions** topic and click **Check Out**.
4. Copy and move the topic to your repository as follows:
 - a. From the **SDL LiveContent** menu, select **Check In As**.
 - b. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, and the **Practice Doc** folder.
 - c. Select the **Topics** folder and click **Next**.
 - d. In the **General** tab, in the **Title** field, enter:
 Document Definitions
 - e. Skip the **Version** tab.
 - f. In the **Workflow** tab, complete the fields as you did in previous exercises.
 - g. Select **Immediately check out the object** and click **OK**.
5. In the Document Definitions Page section, with the cursor after the closing `p` tag, press **Enter** and select **table**.
6. In the Insert Table dialog box, enter information to create 5 rows and 2 columns, and click **OK**.
7. With the cursor in the first row of the first column, from the **Table** menu, select **Convert to Header Row**.
8. Add content to the table so that it looks like the following figure. (You can enter as little or as much content as you want.)

Option	Description
Search	Enter a definition name. Partial strings are matched if you type the beginning of the definition name. Partial strings with wildcards are not allowed.
Refresh	Refresh after a search to see all document definitions. The document definition list from the B2B server is retrieved.
B2B Configuration	Opens a browser to Oracle B2B, using the connect specified on the Server Connection page.
Use Routing ID	Selects all document definitions that use a routing ID.

9. With the cursor after the opening table tag, press **Enter**, select **title**, and enter:

Options on the Document Definitions Page

10. With the cursor after the closing title tag, press **Enter**, select **desc**, and enter:

The table contains two columns. The column with the Option heading lists the options displayed on the Document Definitions Page of Oracle B2B. The column with the Description heading describes each of the options. Excluding the heading row, there are four rows, each containing an option and a description of the option.

11. Create an ID for the table as follows:

- With the cursor after the opening table tag, click the **Generate ID** icon in the markup toolbar.
- Save the topic. (This step is required.)

12. Add a cross-reference to the formal table as follows:

- With the cursor after the opening p tag (so that the cursor precedes a space and the word *describes*), press **Enter** and select **xref**.
- In the Insert Hyperlink dialog box, from your **Topics** directory, select the **Document Definitions** topic (the topic that is currently open), and click **Bookmark**.
- In the Select dialog box, select the gray box labeled **table** and click **OK**.
- In the Insert Hyperlink dialog box, delete the text in the **Text to display** field and click **Insert**.

13. Check in the topic.

Working with Images

Informal and Formal Figures

Images can be informal figures, which are not numbered and do not have titles, or formal figures, which are numbered and have titles.

For an informal figure, use the `image` tag. For a formal figure, use the `fig` tag.

An image requires a graphic description to make the image accessible.

You can drag bitmap images that you create on your computer into the repository. These include screenshots and any images in GIF, JPG, or PNG formats. For a vector image, for which you need an EPS format for the PDF file and a GIF, JPG, or PNG format for the HTML files, use the SDL LiveContent Architect web client to add the image to the repository.

The placement of an image in the Arbortext Editor window will vary depending on your window size. Do not be concerned if your images do not align with the left margin.

See Also:

[Guidelines for Figures and Images](#)

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating figures and images.

Exercise: Adding a Screenshot to the Repository

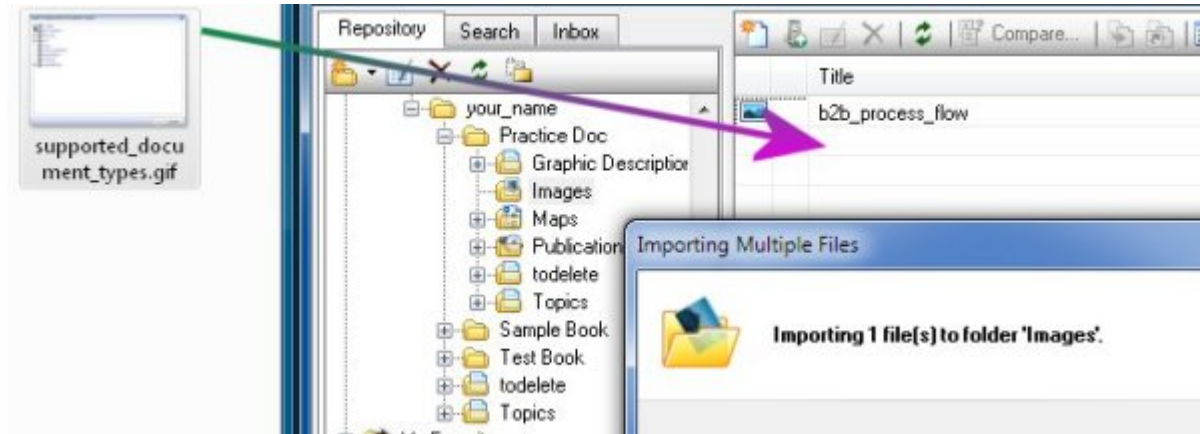
Create a screenshot and add it to the repository.

In this exercise, you will drag the screenshot file from your desktop into your **Images** folder. Dragging the screenshot file is shown at the end of the steps.

To add an EPS image to the repository, you must use the SDL LiveContent Architect web client.

1. Create a screenshot (of anything you like) and save it as a GIF, JPG, or PNG file on your desktop.
2. In Arbortext Editor, from the **SDL LiveContent** menu, select **Browse Repository**.
3. In the Repository tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Images** folder.
4. Ensure that the screenshot file on your desktop and the repository window are side-by-side (with no other files or windows in the drag path), and drag the screenshot file into the **Images** folder area. (If you have the screenshot file

displayed in an Explorer window, then ensure that the Explorer window and the repository window are side-by-side.)



5. In the **Images** folder area, right-click the image name, select **Properties**, and do the following:
 - a. From **Illustration type**, select **Screenshot**.
 - b. Click **OK**.
 - c. Minimize or close the repository window.

See Also:

[Importing Graphics Using the Browse Repository Dialog Box](#)

You import a graphic by dragging it from storage on your local computer to a folder in the **Browse Repository** dialog box. For a graphic with a separate EPS version for PDF output, use the Web Client to import the EPS file.

Exercise: Inserting an Informal Figure

Create a graphic description of your screenshot, insert the screenshot as an informal figure, and then link to the graphic description.

In this exercise, you can add as much or as little text as you want. The goal is to insert the informal figure within a step. The completed topic is shown at the end of the steps.

1. Create a graphic description topic as follows:
 - a. In ArborText Editor, from the **SDL LiveContent** menu, select **New**.
 - b. Select the **Graphic Description** template and click **Next**.
 - c. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Graphic Descriptions** folder, and then click **Next**.
 - d. On the **General** tab, in the **Title** field, enter the file name of your screenshot.
 - e. Skip the **Version** tab.
 - f. On the **Workflow** tab, complete the fields as you did in previous exercises.
 - g. Ensure that **Immediately check out the topic** is selected and click **OK**.

- i. Check in the topic.

Tip: Ensure that you open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.

- a. Delete the unneeded `draft-comment`, `context`, `prereq`, and `_comment` tags and the boilerplate text.

b. With the cursor after the opening `shortdesc` tag, enter any text that you like.

c. With the cursor after the first opening `<cmd` tag, delete the boilerplate text and enter any text that you like.

d. With the cursor after the first closing cmd tag, press **Enter** and select **info**.

e. With the cursor after the first opening `info` tag, press **Enter** and select **image**.

f. Select the screenshot and click **Insert**.

4. Add the link to the graphic description as follows:

- a. With the cursor after the opening image tag, press **Enter** and select **longdesc**.

b. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Graphic Descriptions** folder.

c. Select your graphic description and click **Insert.**

The completed topic is shown in the following figure. (Your topic will look different.)

task id="GUID-40371E20-E225-4FB5-8913-F3BAE72B9F02" xml:lang="en-US"

title id="GUID-824DB915-F88B-4943-A428-6BA9FDA91A8D" **My Task** title

shortdesc My task is all about my task. shortdesc

prolog

metadata keywords keywords metadata

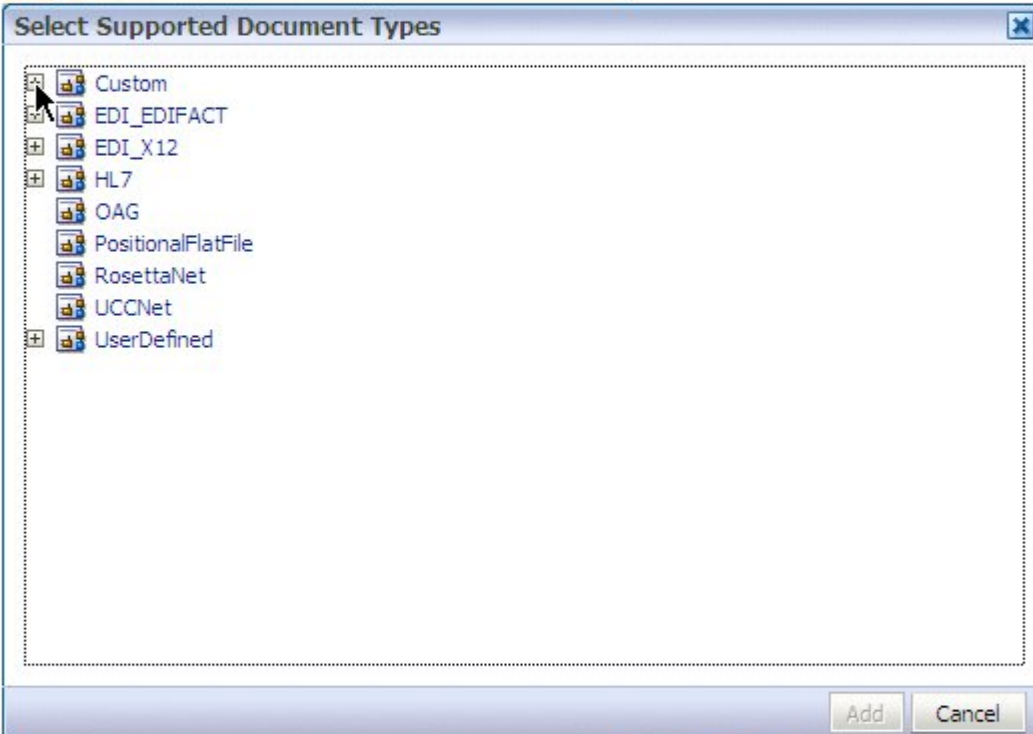
prolog

taskbody

steps

step 1. cmd Expand the uicontrol Custom uicontrol document type. cmd

info image href="GUID-4623AB35-61C1-4484-BFF5-A8660B50715E"



longdesc href="GUID-C723C3DA-1C5A-4AA3-A611-2DA778697899" image info

step

step 2. cmd (Required) Enter text of second step here. cmd

stepresult (Optional) Enter the result of the step here. stepresult

step

steps

result (Optional) Enter the result of the procedure here. result

taskbody

task

5. Check in the topic.

See Also:[Attributes for Images](#)

Use the Modify Attributes dialog box (**Ctrl+D**) to set the attributes for image and fig tags.

Exercise: Inserting a Formal Figure

Insert a flow diagram as a formal figure. The flow diagram is a vector image, for which there is an EPS version for the PDF output and a GIF version for the HTML output.

In this exercise, you will insert the GIF version, which is in the repository. You will also set the placement of the image so that it spans the margins, and add a title, an ID, and a cross-reference to the figure number. The completed topic is shown at the end of the steps.

1. Create a concept topic named Oracle B2B Process Flow. (Refer to previous exercises for detailed instructions.)

Tip: Ensure that you open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.

2. After the title tag, enter:

```
Oracle B2B Process Flow
```

3. With the cursor after the opening shortdesc tag, enter:

```
The Oracle B2B process flow starts with creating B2B
guideline files in the Oracle B2B Document Editor and
continues with using the Oracle B2B interface to create
document definitions, configure trading partners, and create
and deploy agreements.
```

4. With the cursor after the first opening p tag, delete the boilerplate text and enter:

```
shows the Oracle B2B process flow.
```

5. With the cursor after the closing p tag, press **Enter** and select **fig**.

6. Open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - Completed** folder, and the **Images** folder.

7. Select **b2b_process_flow** and click **Insert**. (You are reusing an image from another document.)

8. With the cursor after the opening fig tag, press **Enter**, select **title**, and enter:

```
Oracle B2B Process Flow
```

(The cursor is in the correct place for you to start entering text.)

9. Create an ID for the image and set the image to span the margins as follows:

- a. With the cursor after the opening fig tag, press **Ctrl+D**.
- b. In the **Reference** tab, click the **Generate ID** icon next to the **id** field.
- c. In the **Other** tab, from the **expansion** field, select **page**.

- d. Click **OK**.
 - e. Save the topic. (This step is required.)
10. Add a cross-reference to the formal figure as follows:
- a. With the cursor before the expression *shows the Oracle B2B Process flow* (with a space preceding *shows*), press **Enter** and select **xref**.
 - b. In the Insert Hyperlink dialog box, from your **Topics** directory, select the **Oracle B2B Process Flow** topic (the topic that is currently open), and click **Bookmark**.
 - c. In the Select dialog box, select the gray box labeled **fig** and click **OK**.
 - d. In the Insert Hyperlink dialog box, delete the text in the **Text to display** field and click **Insert**.
11. Delete the unneeded tags and boilerplate text.

The topic is shown in the following figure. However, the topic is not complete until you create a graphic description to which you link using the `longdesc` tag.

concept id="GUID-CDB1300B-7BD8-4B61-BEC4-41258BD7428F" xml:lang="en-US"

title id="GUID-783EB9F6-73BF-4F8D-938D-120F02FB1EE9"

Oracle B2B

Process Flow

shortdesc The Oracle B2B process flow starts with creating B2B guideline files in the Oracle B2B Document Editor and continues with using the Oracle B2B interface to create document definitions, configure trading partners, and create and deploy agreements. **shortdesc**

prolog

metadata

keywords

keywords

metadata

prolog

conbody

p

xref href="GUID-CDB1300B-7BD8-4B61-BEC4-41258BD7428F#GUID-CDB1300B-7BD8-4B61-BEC4-41258BD7428F/OracleB2B

xref shows the Oracle B2B process flow. **p**

fig expanse="page" id="OracleB2BProcessFlow-F0579189"

Figure 1. Oracle B2B Process Flow

image href="GUID-AC060240-247F-4F4E-B56A-DD6E76BF42F1"

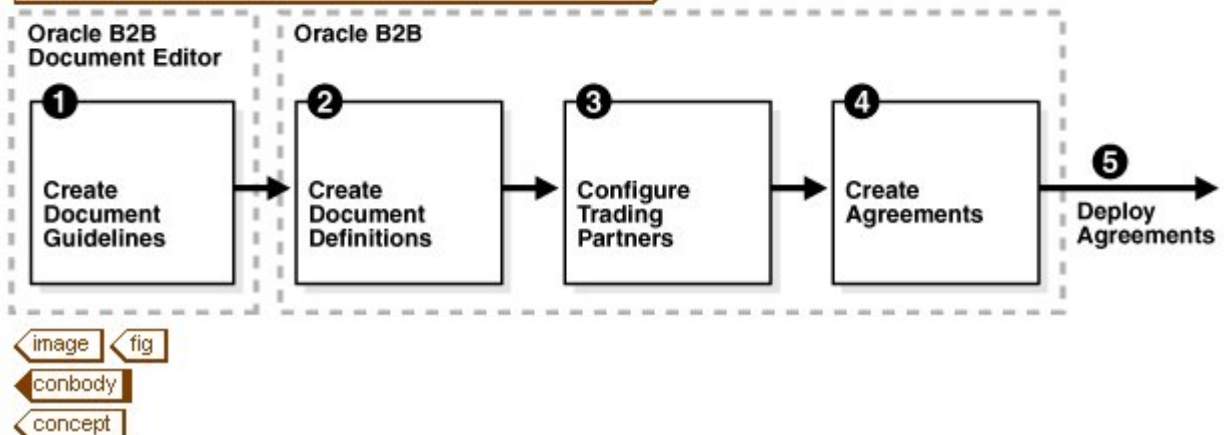


image **fig**

conbody

concept

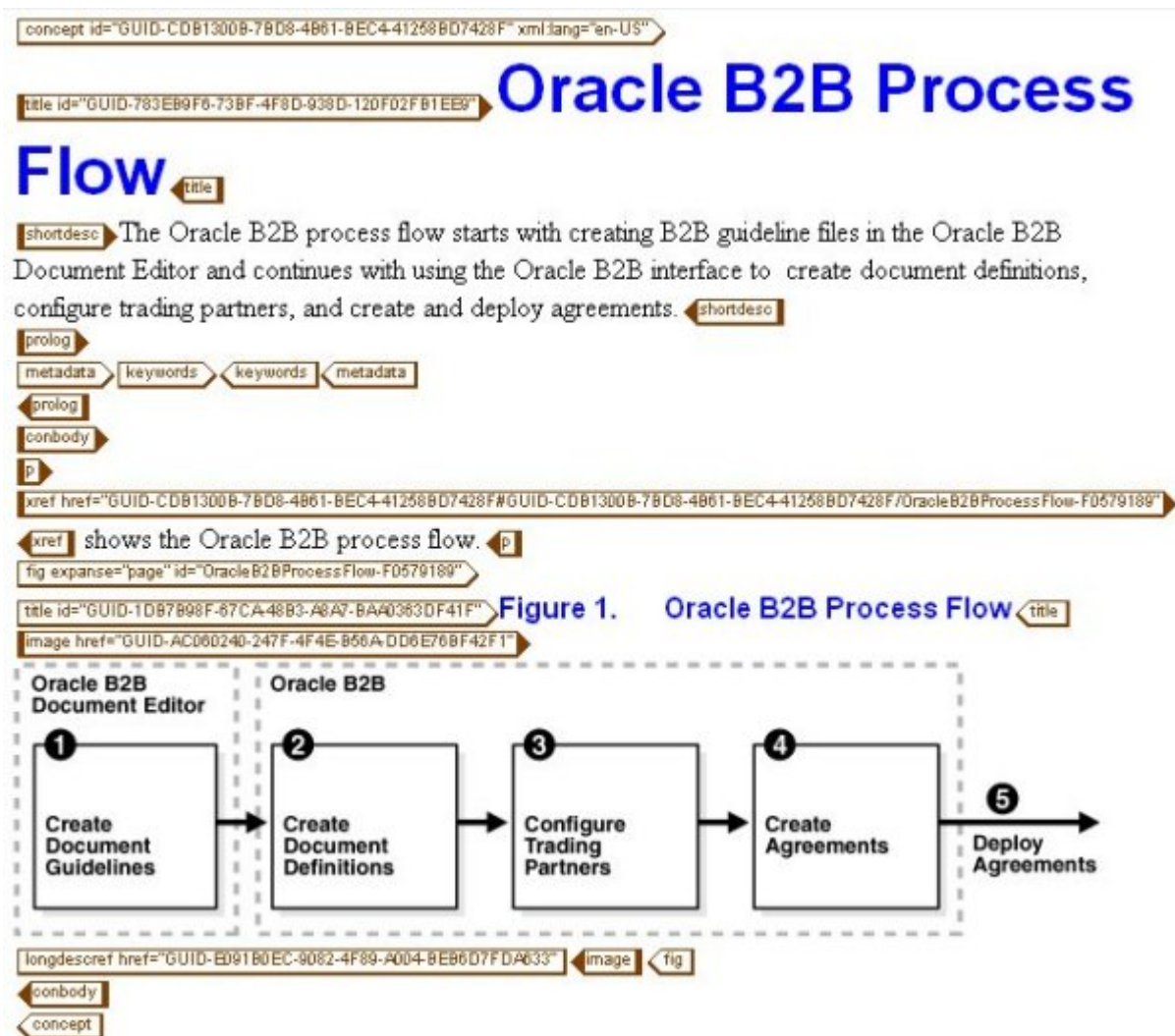
12. For the Oracle B2B Process Flow image, create a graphic description topic that you check in to `Tutorials\your_name\Practice Doc\Graphic Descriptions`. Enter (as much text as you like):

The image shows a flow diagram linking the following numbered blocks, displayed from left to right: 1. Oracle B2B Document Editor: Create Document Guidelines; 2. Oracle B2B: Create Document Definitions; 3. Oracle B2B: Configure Trading Partners; 4. Oracle B2B: Create Agreements; 5. Deploy Agreements.

13. In the Oracle B2B Process Flow topic, with the cursor after the opening image tag, press **Enter**, and select **longdesc**.

14. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Graphic Descriptions** folder, select your graphic description, and click **Insert**.

The completed topic is shown in the following figure.



15. Check in the topic.

See Also:

[EPS Images](#)

Add an EPS image to the repository using SDL LiveContent Architect web client.

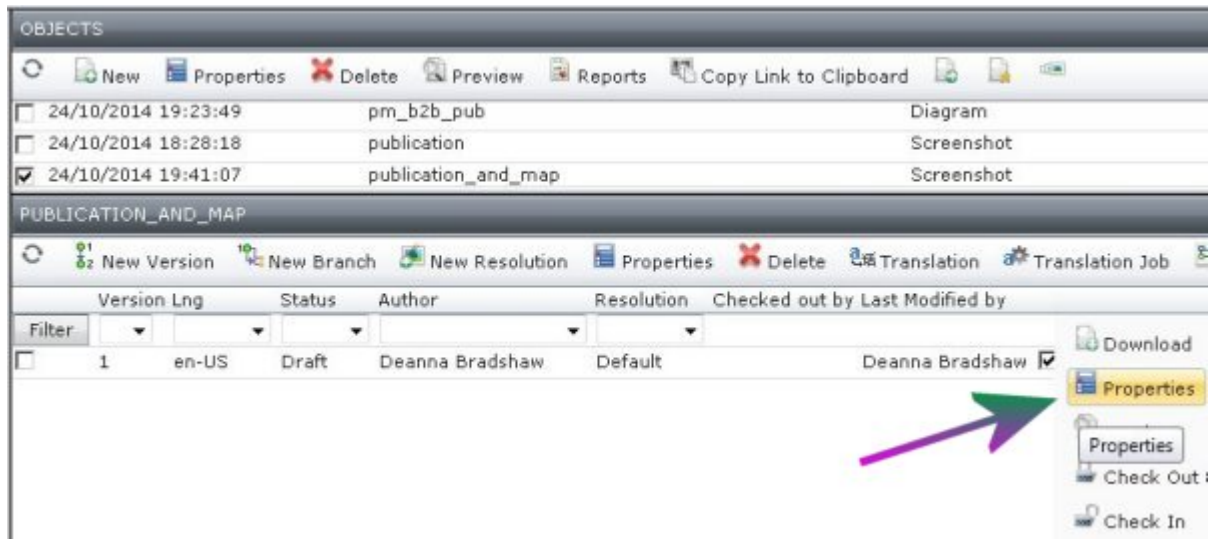
Exercise: Updating an Image

Crop the screenshot and then use SDL LiveContent Architect web client to overwrite the existing screenshot in the repository.

In this exercise, you must use the web client. (You cannot drag the updated screenshot into the repository.) The topic in which you inserted the screenshot is automatically updated with the new version of the screenshot.

1. Update the screenshot on your desktop in some noticeable way, such as cropping it, and save the file.

2. Log in to the SDL LiveContent Architect web client at <https://dadvip0032.us.oracle.com/InfoShareAuthor/>.
3. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Images** folder.
4. In the OBJECTS pane, click the *name* of the image (in the **Title** column). Do not click the check box, although the check box will be checked after you select the name of the image.



5. In the pane that displays details about your screenshot, select the check box next to your name, and click the **Properties** link that is in the lower-right area.
6. In the Illustration Document Language Properties dialog box, next to the **File** field, click the **Browse** button.
7. Select your updated screenshot and click **Open**.
The name of the file appears next to the **Browse** button.
8. Click the **Modify** button.

InfoShare - Document Window - Mozilla Firefox

https://dadvip0032.us.oracle.com/InfoShareAuthor/DocumentOperation.asp?Operation=MODIFY&TopType=CTIMG8

To modify the 'Document language of the Illustration' change the fields below.

Illustration Document Language Properties

Identifier (*):	<input type="text" value="GUID-1ABA7720-7B66-4EE5-B544-FB44B4B"/>	Unique identification name of the Illustration
Version (*):	<input type="text" value="1"/>	Version number
Language (*):	<input type="text" value="en-US"/> ...	Language code
Status (*):	<input type="text" value="Draft"/>	Indicator of the progress of the Illustration
Author (*):	<input type="text" value="Deanna Bradshaw"/>	Name of the person responsible for the object
Reviewer:	<input type="text"/>	Name of the reviewer
Localization Coordinator:	<input type="text"/>	Name of the localization coordinator
Graphic Artist:	<input type="text"/>	Graphic artist name
Last modified by:	<input type="text" value="Deanna Bradshaw"/>	Name of the user which has done the last modification to the document
Source language:	<input type="text"/>	The language that is used as source to translate the XML file to other languages
Comments:	<input type="text"/>	Various comments
Resolution:	<input type="text" value="Default"/>	Resolution of illustration
Format:	<input type="text" value="JPEG"/>	format
File:	<input type="button" value="Browse..."/> publication_and_map.jpg	Enter full pathname of the file you want to submit.

The screenshot is updated in the topic.

Working with Links

Links

You can create links within a topic, across topics (in the same or in a different publication), and to external URLs and documents.

In previous exercises, you used the `xref` tag to create cross-references within a topic (to a formal example, a formal table, a step number, and a formal figure). Limit the use of links created with the `xref` tag.

Instead, use these preferred methods for links:

1. Create cross-topic links and external links in a **relationship table**. A relationship table produces *See Also* links at the end of a topic in the HTML and PDF output. Use the `reltable` tag in the book map to create a relationship table.
2. Create cross-topic links as related links. Related links produce *See Also* links at the end of a topic in the HTML and PDF output. Use the `related-links` tag and the `link` tag at the end of a topic.

See Also:

About Links

There are three ways that links are created using LiveContent Architect and the DITA Open Toolkit: automatically based on the topic hierarchy, using relationship tables (reltables), and using the `xref` element.

Exercise: Creating Cross-Topic Links and External Links

Create *See Also* links at the end of a topic using a relationship table.

In this exercise, you will create a cross-topic link from the What Is Oracle B2B topic to the How Does Oracle B2B Fit into a SOA Implementation topic. You will also create external links: a link to a URL and a link to an external document. The completed relationship table is shown at the end of the steps.

1. Use **Check Out** and **Check In As** to copy and move the following practice topics to your own repository. (Refer to previous exercises for detailed instructions.) Do not select **Immediately check out the topic**. If a gray read-only window opens, then close it.

Topic Title	Check Out Location	Check In As Location
What Is Oracle B2B	Tutorials\YBBUG-Database\Practice Document-For Tutorial\Topics or Tutorials\YBBUG-Generic\Practice Document-For Tutorial\Topics	Tutorials\your_name\Practice Doc\Topics

Topic Title	Check Out Location	Check In As Location
How Does Oracle B2B Fit into a SOA Implementation	Tutorials\YBBUG-Database\Practice Document-For Tutorial\Topics or Tutorials\YBBUG-Generic\Practice Document-For Tutorial\Topics	Tutorials\your_name\Practice Doc\Topics
The weblogic User	Tutorials\YBBUG-Database\Practice Document-For Tutorial\Topics or Tutorials\YBBUG-Generic\Practice Document-For Tutorial\Topics	Tutorials\your_name\Practice Doc\Topics

2. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
3. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Maps** folder.
4. Select the **Oracle B2B User's Guide** book map and click **Check Out**.
5. With the cursor after the closing backmatter tag, from the **Insert** menu, select **Relationship Table**.
6. Identify the topic that is the source for the link (where you are linking *from*):
 - a. With the cursor in an empty row of the **Source** column, press **Enter** and select **topicref**.
 - b. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.
 - c. Select the **What Is Oracle B2B** topic and click **Insert**.
7. Identify the topic that is the target for the link (where you are linking *to*):
 - a. With the cursor in the **Target** column of the same row, press **Enter** and select **topicref**.
 - b. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.
 - c. Select the **How Does Oracle B2B Fit into a SOA Implementation** topic and click **Insert**.
8. Add a link from the **What Is Oracle B2B** topic to a URL:
 - a. In the row that contains the **What Is Oracle B2B** topic, in the **Target** column, place the cursor after the last closing marker (a filled-in wedge shape), press **Enter**, and select **anchorref**.
 - b. Press **Ctrl+D**, complete the following fields in the **Reference** tab, and click **OK**.

For This Field ...	Enter or Select This ...
href	http://www.oracle.com/us/products/middleware/soa/b2b-integration/overview/index.html
navtitle	Oracle B2B Integration

For This Field ...	Enter or Select This ...
scope	external
format	html

9. In the next row of the table, add a link to *Oracle SOA Suite Developer's Guide* (doc ID SOADG, a document in an Oracle library, but which is not available in the repository) as follows:
 - a. Repeat Step 6 with this change: Select **How Oracle B2B Fits into a SOA Implementation** as the source topic.
 - b. With the cursor in the **Target** column, press **Enter** and select **anchorref**.
 - c. Press **Ctrl+D**, complete the following fields in the **Reference** tab, and click **OK**.

For This Field ...	Enter or Select This ...
href	olink:SOADG
navtitle	Oracle SOA Suite Developer's Guide
scope	external
format	html

The relationship table is shown in the following figure.

backmatter

reliable

Source	Target
<p>Topicref: What Is Oracle B2B?</p> <p>Topicmeta:</p> <p>What Is Oracle B2B?</p>	<p>Topicref: How Does Oracle B2B Fit into a SOA Implementation</p> <p>Topicmeta:</p> <p>How Does Oracle B2B Fit into a SOA Implementation</p> <p>Anchorref: Oracle B2B Integration</p>
<p>Topicref: How Does Oracle B2B Fit into a SOA Implementation</p> <p>Topicmeta:</p> <p>How Does Oracle B2B Fit into a SOA Implementation</p>	<p>Anchorref: Oracle SOA Suite Developer's Guide</p>

reliable

bookmark

10. Check in the topic.

Exercise: Creating Related Links

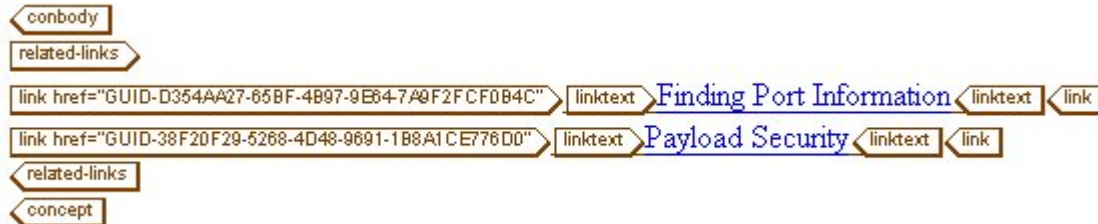
Create *See Also* links at the end of a topic using the `related-links` and `link` tags.

In this exercise, you will create a cross-topic link from The weblogic User topic to the Finding Port Information topic. Related links look like links produced from a relationship table, but they are inserted into the topic rather than into a relationship table. Inserting links into a topic decreases the reuse potential of the topic. The related link tags are shown at the end of the steps.

1. In Arbortext Editor, from the **SDL LiveContent** menu, select **Check Out**.
2. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder.
3. Select **The weblogic User** topic and click **Check Out**.
4. With the cursor before the closing `conbody` tag, press **Enter** and select **related-links**.

5. Press **Enter** and select **link**.
6. Open the **Tutorials** folder, the *your_name* folder, the **Practice Doc** folder, and the **Topics** folder, select **Finding Port Information**, and click **Insert**.
7. (Optional) After the closing link tag, press **Enter** and select **link**. Select another topic to insert.

The related-links and link tags are shown in the following figure.



8. Save the topic and leave it open. You will use it in the next exercise.

Exercise: Creating a Link to an External Document Using a Topic ID

Use `olink:` and a topic ID to create a link to an external document.

In this exercise, you will create a link to a topic in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* (doc ID OIDAG). Inserting links into a topic decreases the reuse potential of the topic. Use a relationship table instead. The completed link is shown at the end of the steps.

1. In The weblogic User topic, with the cursor after *user* followed by a period (before the first closing `p` tag), enter a space and:

See for more information about the Administrators group.

2. With the cursor after *See*, press **Enter** and select **xref**.

3. In the Insert Hyperlink dialog box, do the following:

- a. In the **Text to display** field, enter:

Oracle Fusion Middleware Administrator's Guide for Oracle
Internet Directory

- b. In the **Address** field, enter:

`olink:OIDAG023`

- c. Click **Insert**.

4. With the cursor before the closing `xref` tag, press **Ctrl+D**.
5. In the **scope** field, select **external** and click **OK**.
6. Add a space after *See* (before the opening `xref` tag).

The link is shown in the following figure.

concept id="GUID-B1B6345A-A346-44A4-ACB0-3C3EEDA7C2C2" xml:lang="en-US"

title id="GUID-D2CE579F-FC7C-430C-BF76-6D74FC825364"

The weblogic

User

For the weblogic user in Oracle Internet Directory (OID) to log in to Oracle B2B as an administrator and search for users, the OID Authenticator must have an Administrators group, and the weblogic user must be a member of that group.

prolog

metadata keywords indexterm [index: weblogic user] indexterm indexterm [index: Oracle Internet Directory] indexterm keywords metadata

prolog

conbody

Create a weblogic user in OID using the LDAP browser. Create an Administrators group in OID to which you assign the weblogic user. See [Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory](#) for more information about the Administrators group.

ol

1. Add the following to the `users.ldif` file:

codeblock

```
dn: cn=weblogic,cn=Users,dc=us,dc=oracle,dc=com
objectclass: inetorgperson
objectclass: organizationalPerson
objectclass: person
```

After postprocessing, cross-document links are enabled in documentation libraries.

Completing the Publication

Publications: Chapters and Topics

Add chapters and topics to complete the publication.

In SDL LiveContent Architect Publication Manager, use the **Insert Before** option to insert a chapter. Use the **Add Within** option to link to a topic from within the chapter. Images and graphic descriptions that are linked to the topics are added to the publication when you add the topic. Do not add images or graphic descriptions in a separate step.

You can also insert other book components, such as an appendix or a part page, and you can create submaps for chapters. The life cycle of a publication ends with the release of the publication.

See Also:

[Publication Options for Adding and Inserting](#)

In SDL LiveContent Architect Publication Manager, use the **Insert Before** and **Add Within** options to build a publication.

[Managing Topics in a Publication](#)

You manage the topics in a publication by adding topics to a bookmap or submap, selecting the version of a topic in a publication, and removing topics from a bookmap or submap. You accomplish these

tasks in either Arbortext Editor or the tree view of the Publication Manager.

Adding a Chapter to a Bookmap

You can add a chapter to a book map directly, or you can add a submap that contains (or will contain) the topics in the chapter to the bookmap.

Overview of the Publication Lifecycle

A publication goes through a lifecycle that includes creation, management, publication, and release.

Releasing a Publication

You release a publication when it is final for a product release. The product release can be a beta release or a production release.

Managing Publication Baselines

A baseline lists the versions of the objects that are used in a publication. Each publication has exactly one baseline. You manage baselines with the **Baseline** tab in Publication Manager.

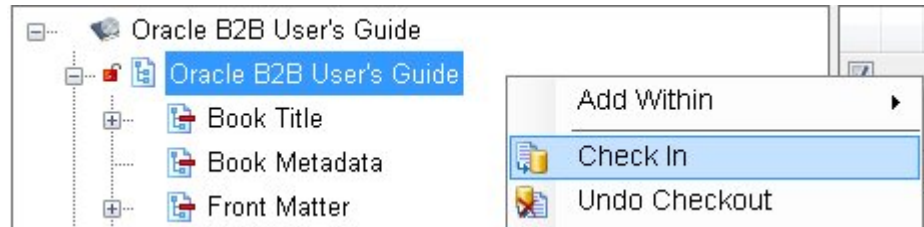
Exercise: Adding Chapters and Topics to a Publication

Add chapters and topics to complete the publication.

1. In SDL LiveContent Architect Publication Manager, from the **Publication** menu, select **Open**.
2. In the **Repository** tab, open the **Tutorials** folder, the *your_name* folder, the **Practice Doc Folder**, and the **Publications** folder.
3. Select the **Oracle B2B User's Guide** publication and click **Open**.
4. In SDL LiveContent Architect Publication Manager, in the **Content** tab, define the first chapter and add topics as follows:
 - a. In the tree structure, right-click **Chapter Chapter 1** and select **XML Attributes**.
 - b. In the **Navigation title** field, delete the text, and then enter the following and click **OK**:

Introduction to Oracle B2B
 - c. When you see the message *The object 'Oracle B2B User's Guide' has been changed by another application, and the new content has been loaded*, click **OK**.
 - d. Right-click **Chapter Introduction to Oracle B2B**, select **Add Within** and then **Topic Ref**.
 - e. In the Insert Values dialog box, click **Browse**.
 - f. Open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - Completed** folder, and the **Topics** folder, select the **What Is Oracle B2B** topic, and click **Insert**.
 - g. In the Insert Values for Topic Ref dialog box, click **OK**.
 - h. When you see the message *The object 'Oracle B2B User's Guide' has been changed by another application, and the new content has been loaded*, click **OK**.

- i. Right-click the **Oracle B2B User's Guide** book map (shown in the figure), select **Check In**, and click **OK**.



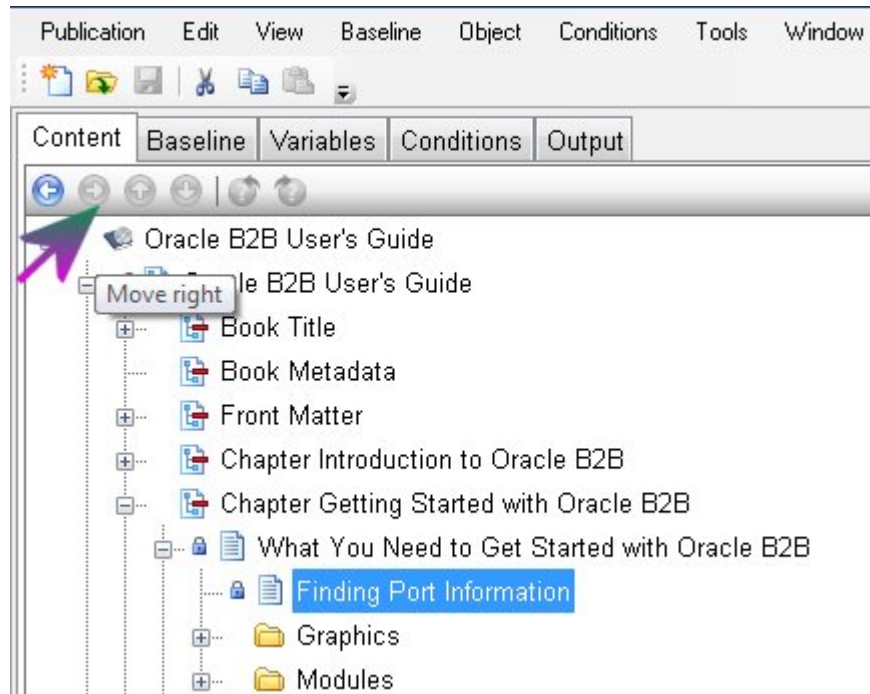
The lock icon changes from unlocked to locked.

5. Repeat Steps 4d through 4h to add the following topics to the first chapter. (You can add as many or as few as you like.)
 - Oracle B2B and Business-to-Business E-Commerce
 - Supported Protocols
 - Security Features of Oracle B2B
 - Payload Obfuscation
 - Payload Security
 - How Does Oracle B2B Fit into a SOA Implementation
6. Insert the second chapter and topics as follows:
 - a. In the tree structure, right-click **Back Matter**, and select **Insert Before** and then **Chapter**.
 - b. Click **OK**.
 - c. Right-click **Chapter** and select **XML Attributes**.
 - d. In the **Navigation title** field, delete any text, and then enter the following, and click **OK**:

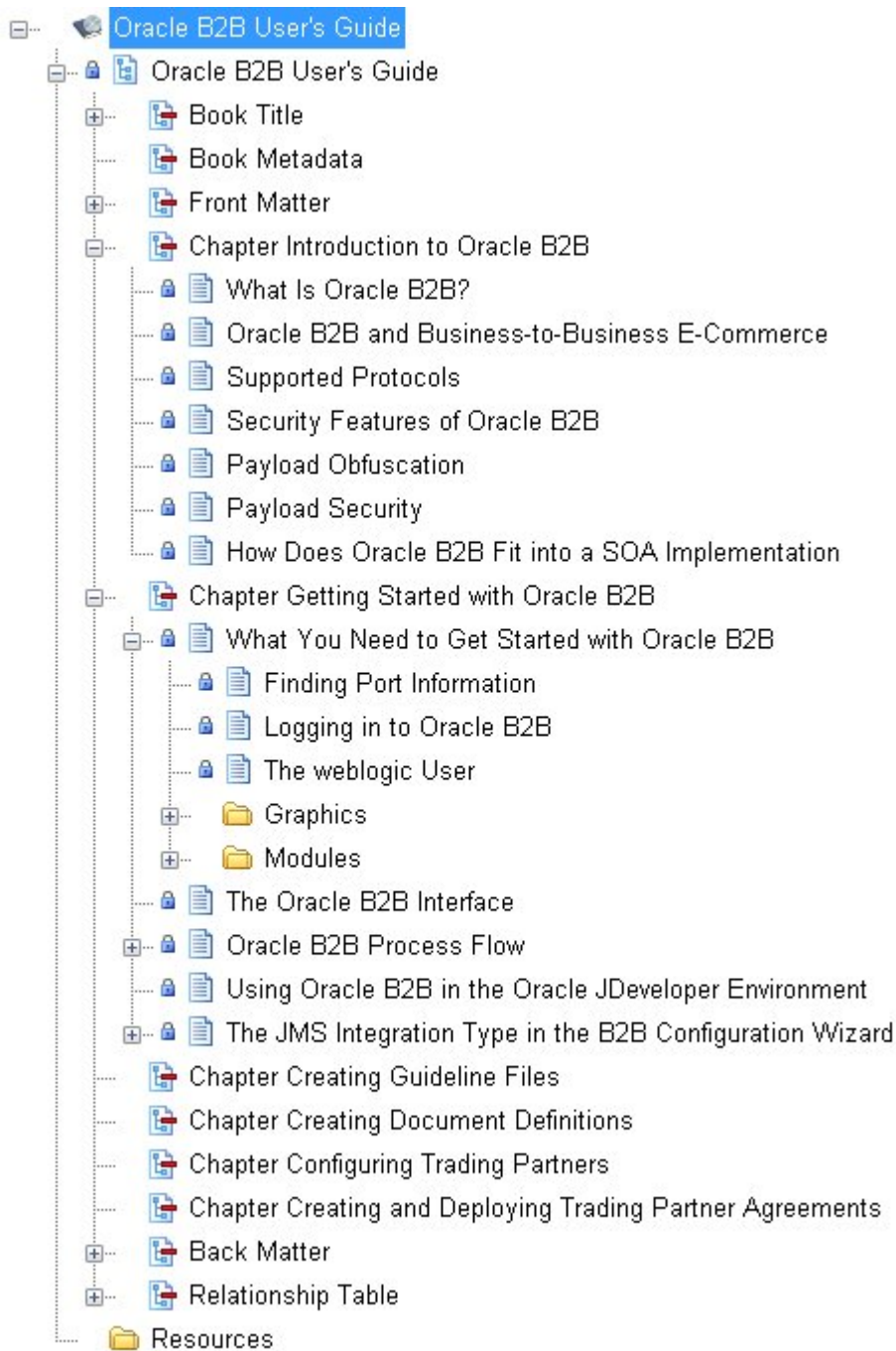
Getting Started with Oracle B2B
 - e. Click **OK**.
 - f. Right-click **Chapter Getting Started with Oracle B2B**, select **Add Within** and then **Topic Ref**.
 - g. In the Insert Values dialog box, click **Browse**.
 - h. Open the **Tutorials** folder, the **YBBUG - Database** folder or the **YBBUG - Generic** folder, the **Practice Document - Completed** folder, and the **Topics** folder, select the **What You Need to Get Started with Oracle B2B** topic, and click **Insert**.
 - i. In the Insert Values for Topic Ref dialog box, click **OK**.
 - j. Click **OK**.
 - k. Right-click the **Oracle B2B User's Guide** book map, select **Check In**, and then click **OK**.

The lock icon changes from unlocked to locked.

7. Repeat Steps 6f through 6j to add the **Finding Port Information** topic to the second chapter.
8. Click the **right arrow** (below the **Content** tab) to move the topic so that it is a child of the **What You Need to Get Started with Oracle B2B** topic, as shown in the following figure.



9. (Optional) Continue adding chapters and topics so that the publication looks like the following figure. (You can add as many or as few as you like.)



10. Right-click the **Oracle B2B User's Guide** book map, select **Check In**, and then click **OK**.

11. From the **Publication** menu, select **Save**. Leave the publication open for the next exercise.

Creating Output

Output Formats and Publishing

Specify output formats for a publication and then publish the publication.

DocBuilder creates output formats such as PDF files, XHTML files, and help files from the XML files that you author in Arbortext. You can access DocBuilder from within the SDL LiveContent Architect Publication Manager interface, or you can log in to DocBuilder directly. Use the SDL LiveContent Architect Publication Manager interface to specify the Oracle publishing options.

DocBuilder can also send the XHTML files to Oracle Review and Doc QA, send the output to the id_common server, and create an archivable ZIP file to send to DocArch.

See Also:

[Oracle Document Engineering DocBuilder Guide](#)

[About Oracle Publishing Options](#)

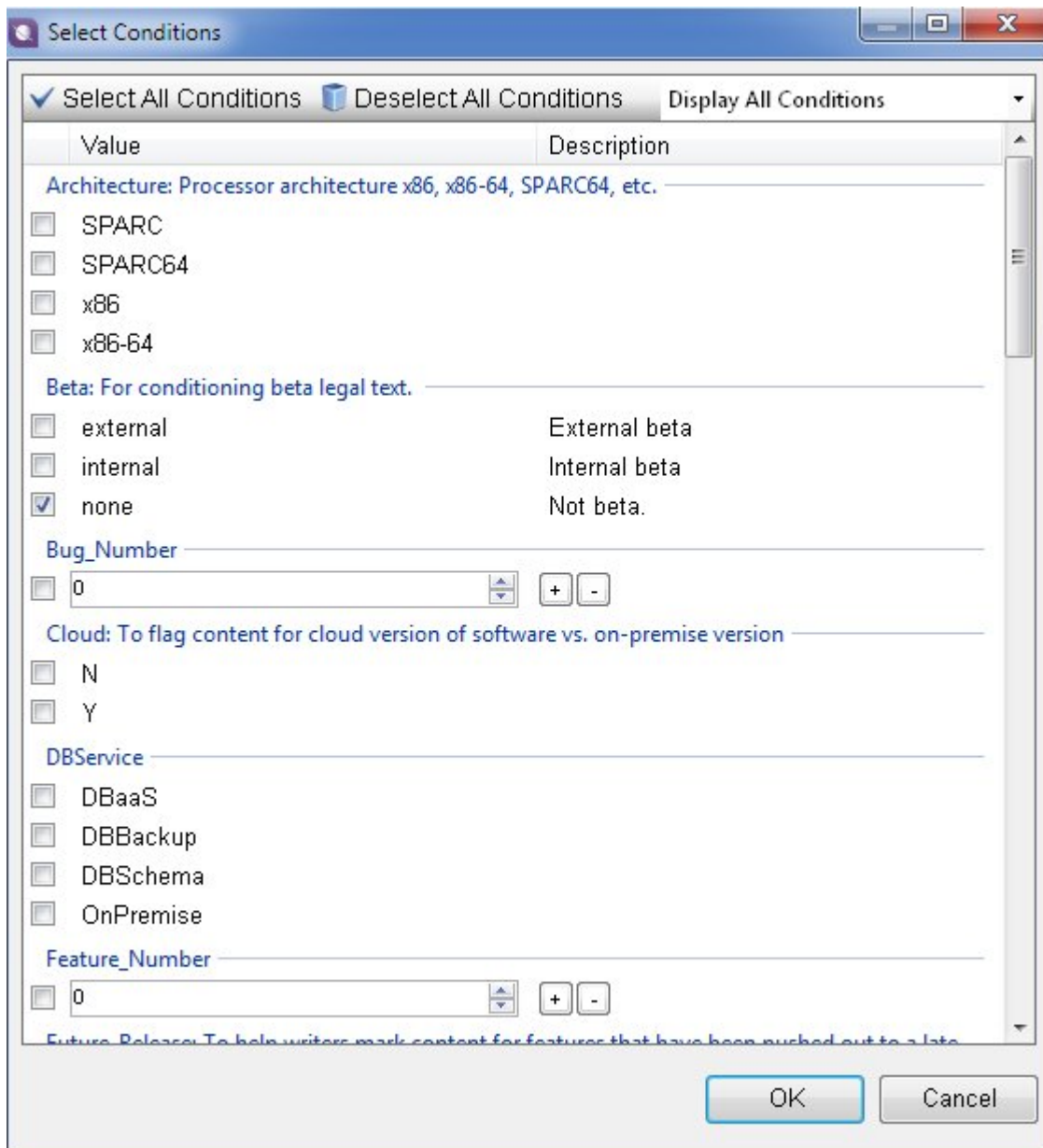
Set the Oracle publishing options in the Publishing Options tab.

Exercise: Creating Output

Create output formats such as PDF files, XHTML files, and help files.

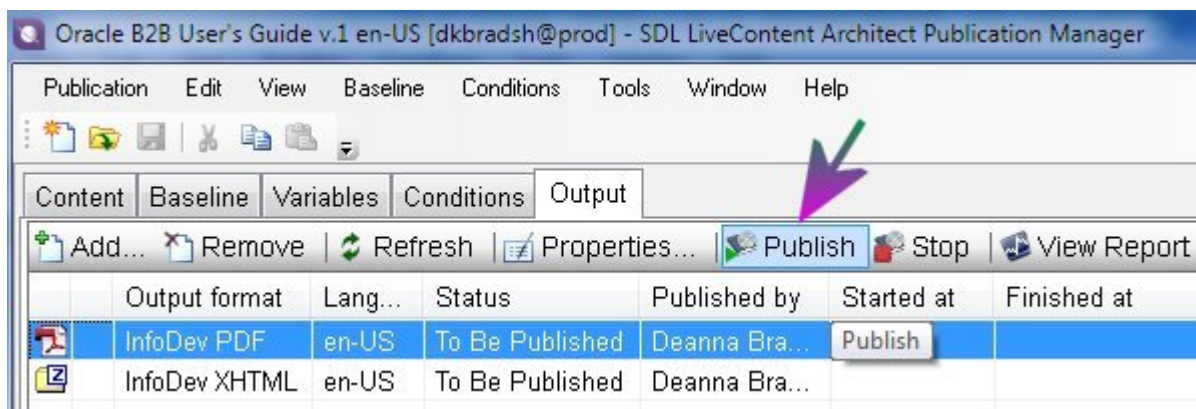
When you create XHTML files or help files and download the files to your computer, stylesheet formatting is not supplied, which can make the files difficult to review. To view the XHTML or help files with formatting, copy the dcommon directory to a location that is two levels higher than your XHTML or help files. A dcommon directory is included in the FrameMaker templates directory, and is generated in a FrameMaker book directory whenever you run the DARB XHTML Converter or run a book on Review Site Builder. The dcommon directory contains the blafdoc.css stylesheet and the images that you need for navigation icons.

1. In SDL LiveContent Architect Publication Manager, open the **Oracle B2B User's Guide** publication.
2. Set the conditions to hide the alphabetanotices and revenuerecognitionnotices legal notices:
 - a. Select the **Conditions** tab.
 - b. Click the **Modify** button.
 - c. In the **Select Conditions** dialog box, from the list in the upper-right corner, select **Display All Conditions**.



- d. In the **Beta: For conditioning beta legal text** section, select **None**.
 - e. Click **OK**.
 - f. From the **Publication** menu, select **Save**.
3. Add output formats as follows:
 - a. Click the **Output** tab.
 - b. Click the **Add** button.
 - c. In the Select Output Format dialog box, select **InfoDev PDF** and click **Next**.
 - d. In the **Publishing Options** tab, enter the following Oracle publishing options and click **OK**:

- In the **Part #** field, enter:
Q10229-04
 - In the **Print Date** field, enter:
December 2014
 - From the **Beta draft** list, select **none**.
 - From the **Auto-gen links** list, select **nofamily**.
You must select **nofamily** to create *See Also* links from a relationship table.
 - From the **Brand** list, select **DARB**.
 - Do not select **Include draft-comments**.
- e. Repeat Steps 3b through 3d. In Step 3c, select **InfoDev XHTML**.
4. Select the **InfoDev PDF** output format, and click the **Publish** button, as shown in the following figure.



5. When the date and time appear in the **Finished at** column, click the **Download** button to view the PDF file.

You will also receive an email notification with a link to the output file.

6. Repeat Steps 4 through 5 for the InfoDev XHTML output.

You can open or save the output file. After you download the XHTML output to your computer, open the `toc.htm` file.

Resetting Your SDL LiveContent Architect Password in DocBuilder

Logging In to DocBuilder

Log in to DocBuilder to create output formats such as XHTML and PDF files from the XML files that you author in Arbortext.

If you can log in to the SDL repository, then you have a user account on DocBuilder.

1. Go to DocBuilder at <https://docbuilder.us.oracle.com/>.
2. Log in using your Single Sign-On user name and password.

Resetting Your SDL LiveContent Architect Password

If you forget your SDL LiveContent Architect password, then you can reset it in DocBuilder.

Your SDL password provides access to the SDL repository and to SDL LiveContent Architect Publication Manager. Unlike DocBuilder, SDL does not use Single Sign-On. The reset process generates a temporary password, which you use to log in to SDL LiveContent Architect, and then change your temporary password to the password of your choice.

1. From the DocBuilder interface, under **Account Settings**, click **Reset SDL Password**.
2. Click the **Reset SDL Password** button.
3. Follow the instructions to complete the reset.
4. To return to the DocBuilder interface, in the **Builds** region, click **Builds**.

Inserting Notes

Use the `note` tag to insert a note. (Notes are not related to tables.)

Types of Notes Used in DITA

The InfoDev customizations use four types of notes, each of which has a particular semantic purpose. These note types are `note`, `warning`, `caution`, and `tip`. Use the existing semantic term for each note type. Do not create new types of notes in your document. These note types must mean the same thing across all documentation.

Table B-1 *Types of Notes in DITA*

Note Type	Usage	Output Label
Note	Includes important hints, guidance, or advice on using a program.	Note:
Caution	Indicates the possibility of damage to a program, system, or data. When used in a procedure, it should precede the step where the risk occurs.	Caution:
Warning	Indicates a potential hazard to people. When used in a procedure, it should precede the step where the risk occurs. When the output is built, the text is shown in bold typeface. Legally, warning notices can only be used to convey information about threat to people. Using otherwise can cause Oracle a legal problem.	Warning:
Tip		Tip:
See Also	Used to include a cross-reference to a related topic.	See Also:

Inserting Notes in DITA

To insert notes into your documentation, use the `note` element when editing a topic.

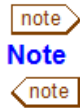
1. Working in Arbortext Editor, place the cursor where you want to insert the note in the text.

2. From the **Insert** menu, click **Markup**.

The Insert Markup dialog box displays.

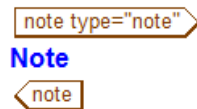
3. Select `note` from the list of markup elements and click **Insert**.

The `note` element displays in the text, as follows:



4. Place the cursor anywhere within the `note` element tags and click **Edit > Modify Attributes....**
5. Under the **Reference** tab, select **caution**, **note**, **tip**, or **warning** from the **type** dropdown menu, depending on the type of note you want to insert.

For example, if you want to insert a Note, then the markup will be as follows:



6. Place the cursor anywhere within the `note` element tags and type your note text.

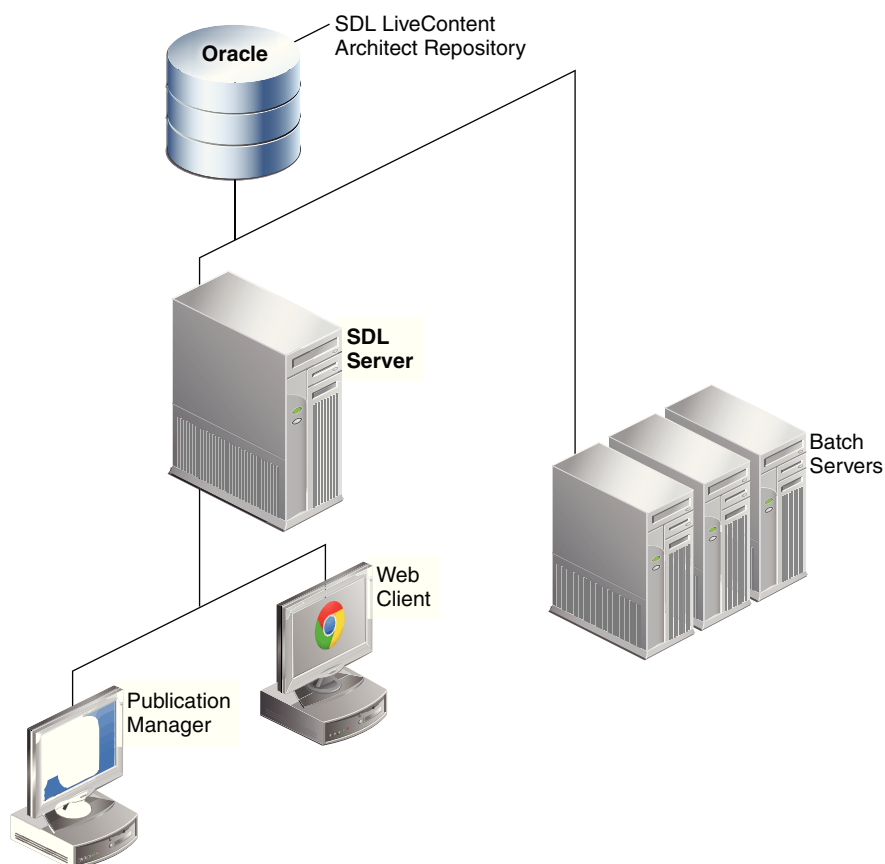
Further Information

About SDL LiveContent Architect

The architecture of SDL LiveContent Architect is shown in the figure.

SDL LiveContent Architect and Arbortext Editor are integrated with the SDL Authoring Bridge (not shown in the figure).

SDL LiveContent Architect



About Short Descriptions

A short description (`<shortdesc>`) is a mandatory element that summarizes a topic and explains its purpose. The descriptions appear in links and in search engine results.

A short description gives just enough information so that users can determine whether they need to read the topic itself.

The `<shortdesc>` element is valid between the `<title>` and main body of the topic. The short description is output in the following locations:

- The first paragraph of a topic
- In HTML, hover text for related links created either in a `related-links` element or in a relationship table (`reltable`) in the map
- The text under automatically generated child topic links at the end of a topic (The “link preview”)
- Search engine results

Within `<shortdesc>`, you may use `<keyword>` or any other valid elements. Typically, a short description contains one to three sentences, and contains no more than 50 words. As much as possible within these parameters, the short description should answer the five W's (who, what, when, where, why), and sometimes the one H (how).

Use the `<abstract>` element when the initial paragraph of a topic is unsuitable for use as a link preview or for summaries, because, for example, it contains lists or tables, or because only a portion of the paragraph is suitable. In this case, place the `<shortdesc>` element within the `<abstract>` element. The output contains all text within the `<abstract>` element, including whatever is wrapped in the `<shortdesc>` element, but links and search results show only the short description.

Example C-1 Short Description: Example

This example shows the short description for a conceptual topic about permanent tablespaces. The `<shortdesc>` appears after the `<title>` element, and before the `<prolog>` element.

```
<concept>
  <title>Permanent Tablespaces</title>
  <shortdesc>A permanent tablespace groups persistent schema
objects. The segments for objects in the tablespace are stored
physically in data files.</shortdesc>
  <prolog></prolog>
  <conbody>
    <p>Each database user is assigned a default permanent
tablespace.</p>
  </conbody>
</concept>
```

In the output, the text would appear in the following order (formatting is approximate):

Permanent Tablespaces

A permanent tablespace groups persistent schema objects. The segments for objects in the tablespace are stored physically in data files.

Each database user is assigned a default permanent tablespace.

Example C-2 Short Description Using `<abstract>`: Example

```
<abstract><shortdesc><shortdesc>

<concept>
  <title>Space Management in Data Blocks</title>
```

```
<abstract>As the database fills a data block from the bottom up,
the amount of free space between the row data and the block header
decreases. This free space can also shrink during updates, as when
changing a trailing null to a nonnull value. <shortdesc>The
database manages free space in the data block to optimize
performance and avoid wasted space.</shortdesc></abstract>
...
```

In the output, the first paragraph would look as follows (formatting is approximate):

Space Management in Data Blocks

As the database fills a data block from the bottom up, the amount of free space between the row data and the block header decreases. This free space can also shrink during updates, as when changing a trailing null to a non-null value. The database manages free space in the data block to optimize performance and avoid wasted space.

...

About Semantic Tags

Semantic tags reinforce the meaning of the formatted text.

Use semantic tags instead of typographical tags whenever possible.

Semantic Tags

Semantic tags that you use to format inline text are described.

DITA Tag	Used to Indicate	Output Format	Example
apiname	The name of an application programming interface (API), such as a function or procedure name, excluding PL/SQL and Java database program units, which are schema objects	Monospace	The DOC API will write documentation for you.
cmdname	The name of a command, including utility names, script names, and SQL statements	Monospace	Use the cd command to change the current working directory.
codeph	A phrase that should look similar to the monospace font used for programing code. This tag is not technically a semantic tag. It is for situations when there is no appropriate semantic tag.	Monospace	Use the EMP table to find employee information.
filepath	A directory path or file path	Monospace	Go to C:\Program Files (x86)\Cisco.

DITA Tag	Used to Indicate	Output Format	Example
menucascade	A series of hierarchical menu choices	Bold	Select File > Import > Files.
msgnum	A system message or error message number	No formatting	
msgph	A system message or error message	Monospace	Monospace
option	An option that can be used to modify a command, SQL statement, or subprogram, such as <code>-lrt</code> in the command <code>ls -lrt</code>	Monospace	Monospace
parmname	A parameter name, when describing parameters of an API, command, or script, including parameters of PL/SQL and Java database program units	Monospace	Monospace
systemoutput	Output generated by software or responses to a command	Monospace	Monospace
term	A new word or phrase that requires a definition	Bold	Bold
uicontrol	A graphical user interface control, such as a button, menu item, text field, and so on	Bold	Bold
userinput	Text that a user should input	Monospace	Monospace
varname	The name of a variable that must be supplied to a software application, or a placeholder for a user-supplied value	<i>Italic</i>	<i>Italic</i>
wintitle	The title of a window or dialog box in a graphical user interface	No formatting	

About Figures and Images

In DITA, insert graphics either as figures (`<fig>`), which have titles, or untitled images (`<image>`). Nearly all readers prefer to view an illustration when it helps them avoid reading text.

For graphics in concept topics and reference topics, use a `<fig>`. To insert images that refer to UI features in a task or in a reference table, use an `<image>`.

For both figures and images, you must include alternate text to make the illustrations accessible. Adding alternate text is important for making documentation accessible to disabled users. For navigation images and icons such as buttons and arrows, use the `<alt>` element to add a brief description. For all other graphic descriptions, create a separate topic using the Graphic Description template to contain the text description. Reference this topic using a `<longdesc>` element.

DITA supports the inclusion of screenshots. The InfoDev standard is to store the screenshot as a JPG rather than a GIF to enable better scaling, avoiding image distortion. The screenshot is scaled in the PDF, but renders full size in HTML. The InfoDev recommendation is to take the screenshot using HyperSnap, and then change the horizontal and vertical resolutions of the image by multiplying by 1.25, 1.33, 1.5, 1.66, or 2.0.

Guidelines for Figures and Images

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating figures and images.

When creating images and figures, consider the following guidelines:

- Use graphics when you need to insert images to refer to UI features in a task or in a reference table.
- For both figures and graphics, you must always include an alternate text description to make the illustrations accessible.
- Use `<fig>` for a formal figure, that is, a figure that is titled and can be cross-referenced. Use `<image>` for an informal (untitled) graphic.

Figure elements require a title and number, and should be used for most images. Graphic elements are informal images, which can be embedded within content units in document, such as in a procedure.

- For both figures and graphics, you must include an alternate text description to make the illustrations accessible.

Use the `<alt>` element to add a brief description of navigation images such as buttons and arrows. For longer, more complex descriptions (such as those required for diagrams), create a separate topic, of the Graphic Description type, containing the text description and reference this topic using a `<longdesc>` element. The `<longdesc>` element is an empty element. Use its `@href` attribute to reference the topic containing the graphic description.

- To get a wide figure, set the `@expanse` attribute to "page".
- To avoid placing the figure inline, set the `@placement` attribute to "break".
- When setting `@href`, specify a relative location.

- Do not use the @alt attribute. Use <alt> instead.

Guidelines for Writing Graphic Descriptions

When writing graphic descriptions, follow these guidelines:

- Create a new graphic description topic using the Graphic Description template.
- Use statements such as “at upper left is a...” or “the radio buttons on the left...” to help readers understand the layout of the picture.
- Do not use the word “graphic” because it is a keyword for most screen readers. Use “image,” “illustration,” or “figure” instead.
- Avoid including the image file name in the description. If you change the image file name, then you must update the description.

Guidelines for Taking Screenshots

When taking screenshots for use in by an <image>, follow these guidelines:

- Use HyperSnap 5.5 or later as your capture tool.
- Add a 1-pixel wide frame around the image.
- Change the horizontal and vertical resolutions of the image by multiplying by 1.25, 1.33, 1.5, 1.66, or 2.0.
- Save the image as a JPG.

Attributes for Images

Use the Modify Attributes dialog box (**Ctrl+D**) to set the attributes for image and fig tags.

To Do This ...	Use This Attribute ...	Applies To These Tags ...
Start the graphic on a new line	break (placement list)	image
Place the graphic in the same line as the surrounding text	inline (placement list)	image
Make the graphic span the margins	page (expanse list)	fig (To make an image span the margins, use the fig tag but delete the title tags.)

EPS Images

Add an EPS image to the repository using SDL LiveContent Architect web client.

See Step 6 for how to add an EPS image to the repository.

Importing Graphics Using the Browse Repository Dialog Box

You import a graphic by dragging it from storage on your local computer to a folder in the **Browse Repository** dialog box. For a graphic with a separate EPS version for PDF output, use the Web Client to import the EPS file.

When you create a screen shot, there is only one file to import into the repository. When a graphic artist creates a graphic, the graphic artist typically creates two versions of the image: one version for XHTML output and one version for PDF output.

Note: The folder that will contain the graphic must already exist in the repository. The content type of the folder must be "Graphic". To verify the content type of a folder, right-click the folder, and select **Properties**. The content type is on the **General** tab.

1. In Publication Manager, from the **Tools** menu, select **Browse Repository**.
2. In the **Repository** tab, locate and select the folder that will contain the graphic.
3. Locate the graphic source file in your computers file system.
4. Position the Browse Repository dialog box and your local file system window side by side on your computer screen so that both are visible.
5. Drag the graphic file from your local file system to the right pane of the **Browse Repository** dialog box.

Only drag and drop the screenshot version of the image, not the EPS file.

6. Complete the following steps to upload an EPS version of the image:

- a. Using your browser, go to the Web Client, and log in if necessary.

Access one of the following URLs in your web browser:

Application Instance	URL
Production	https://dadvip0032.us.oracle.com/InfoShareAuthor/
Development	https://dadvip0015.us.oracle.com/InfoShareAuthor/

- b. Click the **Repository** tab.
 - c. Locate and select the graphic file you imported in the previous steps.
 - d. In the bottom pane, select the check box on the right that corresponds to the version of the graphic you want to modify, and then click the **New Resolution** button.
 - e. In the **Add Illustration Document Language** dialog box, from the **Resolution** list, select **Print**.
 - f. Select **File**, click the **Choose File** button, and then select the EPS version of the graphic in your local file system.

- g. In the **Add Illustration Document Language** dialog box, click **OK**.

Connect to the following URL in a browser:

About Links

There are three ways that links are created using LiveContent Architect and the DITA Open Toolkit: automatically based on the topic hierarchy, using relationship tables (reltables), and using the `xref` element.

The system automatically creates links to child and parent topics in the topic hierarchy. Use reltables to create links to other topics in the repository and to external documents. Create links to elements within a topic or to elements in other topics using the `xref` element.

Links to Child and Parent Topics

In the hierarchy within a DITA map, a parent topic contains its children topics. Our system uses the automatic related links capability of the DITA Open Toolkit, which is based on the topic hierarchy.

The system generates the following links automatically:

- Each parent topic contains links to all of its children topics.
- Each child topic has a link to its parent topic.

Do not create these types of links using reltables or the `xref` element, because it will result in duplicate links in topics.

Links to Topics in the Repository and External Documents

Use reltables to create links to topics in the repository and external documents. A reltable is a table in a map that defines relationships between topics in the repository or relationships between topics and external documents. The system adds links to topics based on the relationships defined in each row of a reltable.

Reltables provide the following benefits:

- **Central management of links:** Reltables are only allowed in maps. Relationships between topics are defined in one location or a small number of locations instead of in hundreds or thousands of topics.
- **Easier maintenance of links:** Because links are defined in a small number of locations, updating or correcting links is easier.
- **Better reusability of topics:** Dependencies between topics are defined in maps instead in the topics. Therefore, it is easier to reuse topics without violating dependencies.

DITA supports several different types of reltables, but at Oracle, writers only create one type: a one-way reltable that has two columns. A one-way reltable creates links from one topic to another, and each link is from the source only topic to the target only topic. It does not create links from the target only topics to the source only topics.

For example, consider the following reltable:

Table C-1 *Sample Reltable*

Source	Target
Topicref to Topic 1	Topicref to Topic 3

Source	Target
Topicref to Topic 2	
Topicref to Topic A	Topicref to Topic B Topicref to Topic C

Given this reltable, the system automatically adds the following links:

- In Topic 1, the system adds a link to Topic 3.
- In Topic 2, the system adds a link to Topic 3.
- In Topic A, the system adds a link to Topic B and a link to Topic C.

Each source only topic must be a topic in the repository. A target only topic can be a topic in the repository or an external document.

An external document might be a topic in an Oracle library that is not available in your repository. An external document might also be a publication produced by a third party, such as a technical paper about relational databases. In general, avoid linking to non-Oracle sources unless it is absolutely necessary.

You enter external links in a reltable in one of the following forms:

- To link to a document or a topic in an Oracle library that is not available in your repository, enter `olink:` and the book ID or topic ID for the topic in the form `olink:BOOKID` or `olink:BOOKIDnnnnn`, respectively. For example, the book ID for the *Oracle Database Administrator's Guide* is `ADMIN`, and the topic ID for one topic in the book is `ADMIN11013`.
- To link to a publication produced by a third party, use the URL of the publication.

Links to Elements Within a Topic or in Other Topics

Because most topics are relatively short, links within a topic usually are not necessary. However, you can create links to the following types of elements in a topic:

- Figure
- Table
- Example
- Step

You create links to these elements by inserting an inline `xref` tag and selecting the element to which you want to link.

You also create links to elements in other topics with an inline `xref` tag. In this case, the target element in the other topic must be a figure, table, or example.

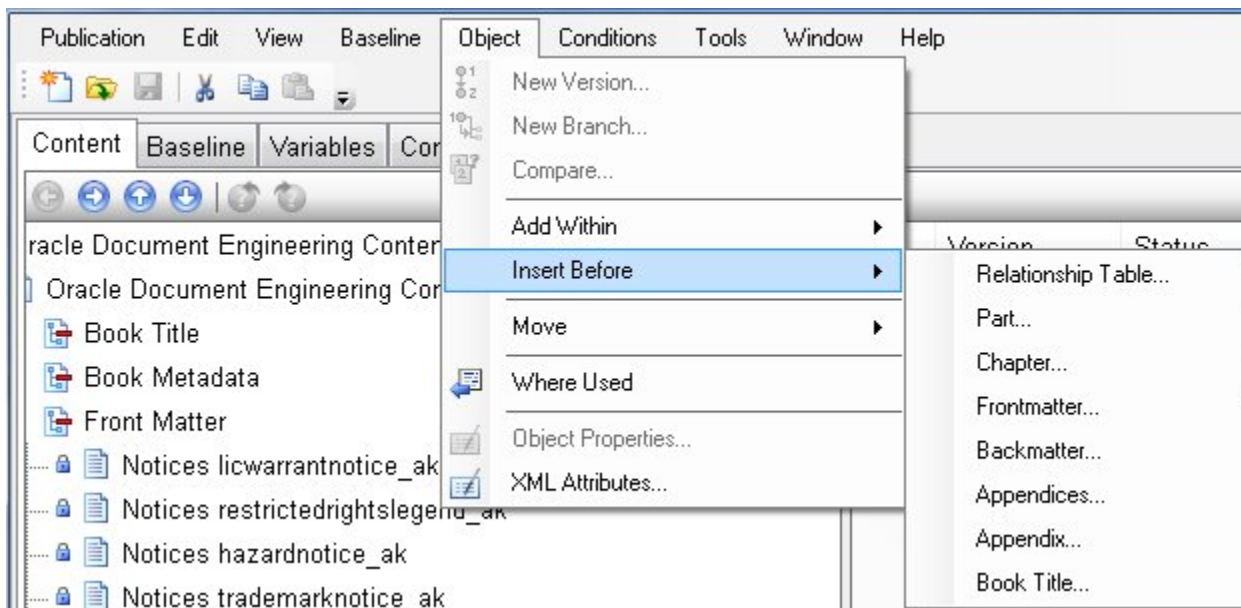
Note: Only use an inline `xref` tag to create links to elements within topics. Do not use an inline `xref` tag to create links to topics or publications.

About Publications

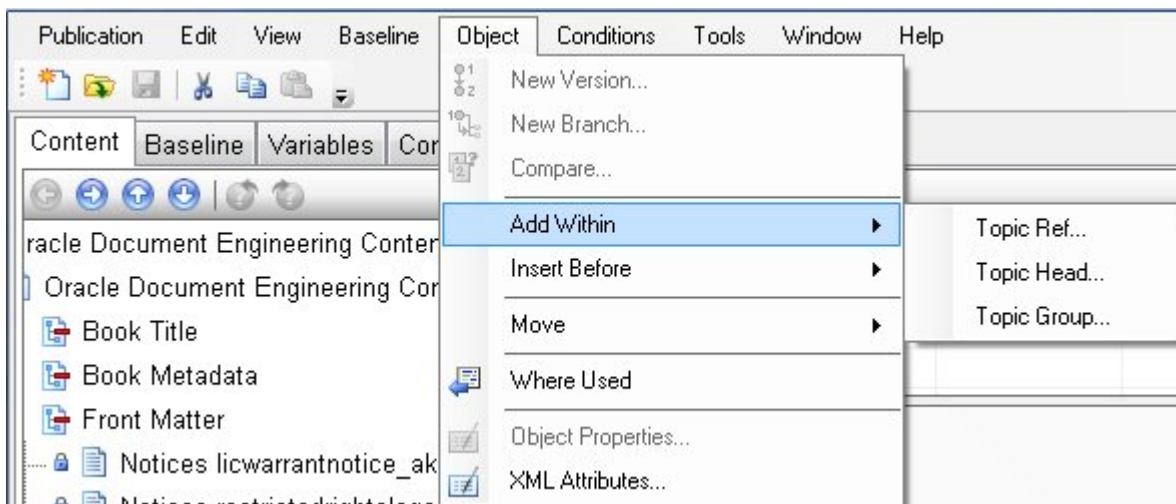
Publication Options for Adding and Inserting

In SDL LiveContent Architect Publication Manager, use the **Insert Before** and **Add Within** options to build a publication.

The figure shows the objects that you can insert into a publication.



The figure shows objects that you can link to from within a publication.



Overview of the Publication Lifecycle

A publication goes through a lifecycle that includes creation, management, publication, and release.

After a publication is created, writers can add, modify, and remove objects such as topics, and the publication owner can manage the publication's metadata and maps.

Writers document new features, edit existing content, and correct documentation bugs until the document is final for a specific product release. At that point, the publication owner publishes the final output for the release and releases the publication.

Releasing a publication freezes its baseline automatically. Therefore, before it can be modified for the next release, the publication owner must create a new version of the publication or create a new publication based on the publication.

A publication owner usually creates a new version of a publication when a new revision of the publication's part number has been ordered (for example, moving from -01 to -02 in the part number).

A publication owner usually creates a new publication based on the released publication when a new publication part number has been ordered for a new product release. A new part number is required when a new release of the product includes new features that must be documented in the publication.

Managing Topics in a Publication

You manage the topics in a publication by adding topics to a bookmap or submap, selecting the version of a topic in a publication, and removing topics from a bookmap or submap. You accomplish these tasks in either Arbortext Editor or the tree view of the Publication Manager.

A bookmap assembles DITA topics into a publication. A bookmap contains pointers (topicrefs) to topics, and these topicrefs are organized hierarchically like a table of contents. A bookmap can contain submaps, and each publication references a single bookmap.

Note: A bookmap differs from a regular DITA map because it includes elements for front matter and back matter, and it includes chapter, part, and appendix elements.

Adding a Chapter to a Bookmap

You can add a chapter to a book map directly, or you can add a submap that contains (or will contain) the topics in the chapter to the bookmap.

You have the following options when you add a chapter to a bookmap:

- Add the chapter to the bookmap directly.

After adding the chapter, you can add topics to the chapter with topic refs. The topic refs are included directly in the bookmap. It is best to use this method if you are sure the chapter will not need to be moved or copied to a different publication. The advantage adding the chapter directly to the publication is that there are fewer submaps to maintain.
- Add a submap to the bookmap.

After adding the submap, the chapter contains the topics in the submap. You can add topics to the submap with topic refs, and the topics are included in the chapter through the submap. The advantage of using a submap is that the submap can be moved or copied easily from one publication to another.

Before completing this task, the publication using the bookmap to which you want to add the chapter must exist.

1. With the publication open in Publication Manager, right-click the chapter that will immediately follow the chapter you are adding and select **Insert Before > Chapter**.

The added chapter appears in the bookmap in Publication Manager before the chapter that was selected.

2. Right-click the added chapter and select **XML Attributes**.
3. In the Insert Values for “Chapter” dialog box, complete one of the following actions:
 - If you are adding the chapter to the bookmap directly, then enter the title of the chapter in the **Navigation title** field.
 - If you are adding a submap to the bookmap, then click the **Browse** button and insert the submap in the **HRef** field. The title of the submap appears in the **Navigation title** field.

The text in the **Navigation title** field is for metadata purposes only. Each chapter starts with an orientation topic, and the title of this orientation topic is the chapter title in the output.

4. Click the **OK** button.

You can add topics to the chapter with topic refs if necessary.

Managing Publication Baselines

A baseline lists the versions of the objects that are used in a publication. Each publication has exactly one baseline. You manage baselines with the **Baseline** tab in Publication Manager.

Releasing a Publication

You release a publication when it is final for a product release. The product release can be a beta release or a production release.

You release a publication by releasing one of its output objects.

Releasing a publication freezes its baseline automatically. Therefore, before you can modify the publication for the next release, you must create a new version of the publication or a new publication based on the publication.

Before you can release a publication, all of the publication’s objects must be present and must be at “60 --Released” status.

The publication must have no invalid links, hyperlinks, or conref links.

At least one of the publication’s output formats must be published and have “Release Candidate” status.

1. In Publication Manager, open the publication by checking it out.
2. Click the **Output** tab.
3. Select the output format with “Release Candidate” status.
4. Click the **Release** button.

Note: The publication is released when the status of the selected output formats is changed to “Released.” Release both the PDF and XHTML output formats when the publication is final for a product release.

About Accessibility

Use these checklists to create accessible documentation.

Formal Tables Accessibility Checklist

Formal tables comply with accessibility requirements by using the desc element. It includes a short description of the contents and organization of the table.

Required Accessibility Elements for Formal Tables

The following table indicates the element or attributes that are required to make formal tables accessible:

Element	Purpose
desc	Includes a short description about the organization of the table (rows and columns) and its contents. Depending on the length of the description you may use a single ph to enclose the contents or different p elements.

Figure C-1 Accessibility Elements in a Formal Table

<p>title id="GUID-C26D27B0-7411-4875-869F-60AA0DAC7990" Table 1 Comparison of Heap-Organized Tables with Index-Organized Tables</p> <p>title</p> <p>desc cid="pLdYVW" Comparison of Index-Organized Tables with Ordinary Tables desc</p>	
p cid="zF46a" Heap-Organized Table p	p cid="1Lz9dY" Index-Organized Table p
p cid="booUq" The rowid uniquely identifies a row. Primary key constraint may optionally be defined. p	p cid="1G0wwI" Primary key uniquely identifies a row. Primary key constraint must be defined. p
p cid="1cy8nY" Physical rowid in codeph ROWID codeph indexterm cid="jPI0" [index: pseudocolumns] indexterm xref cid="24R0M3" format="dita" href="GUID-460C4D37-4C8D-48CE-8C22-CF" pseudocolumn xref allows building secondary indexes. p	p cid="1XK6a" Logical rowid in codeph ROWID codeph pseudocolumn allows building secondary indexes. p
p cid="2D2dx" Individual rows may be accessed directly by rowid. p	p cid="1to40I" Access to individual rows may be achieved indirectly by primary key. p
table	

Informal Tables Accessibility Checklist

Informal tables comply with accessibility requirements by using the desc element.

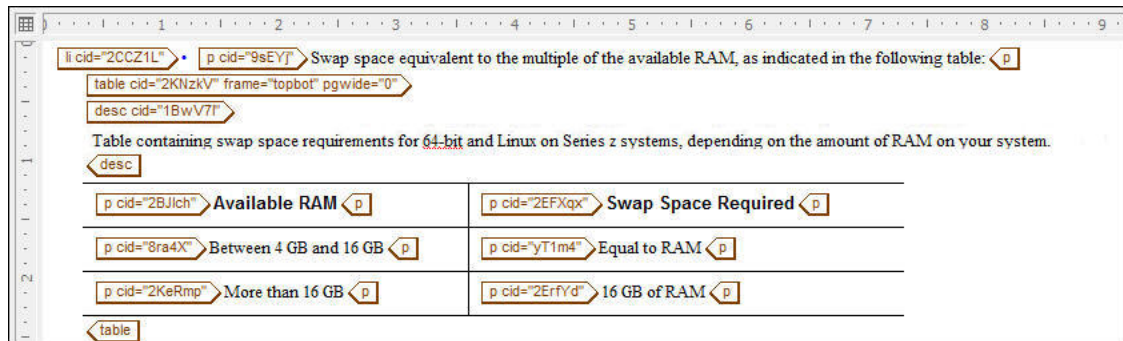
Required Accessibility Elements for Informal Tables

The following table indicates the elements or attributes that are required to make informal tables accessible

Element	Purpose
desc	Includes a short description about the organization of the table (rows and columns)

Element	Purpose
	and its contents. Depending on the length of the description you may use a single ph to enclose the contents or different p elements.

Figure C-2 Accessibility Elements in an Informal Table



Figures Accessibility Checklist

In DITA, figures comply with accessibility requirements by using the `alt` or `longdesctag` within their element structure. The element must provide a description in such a way that a user can understand the layout of an image, the flow of a diagram, or the action that is illustrated, without actually viewing the image.

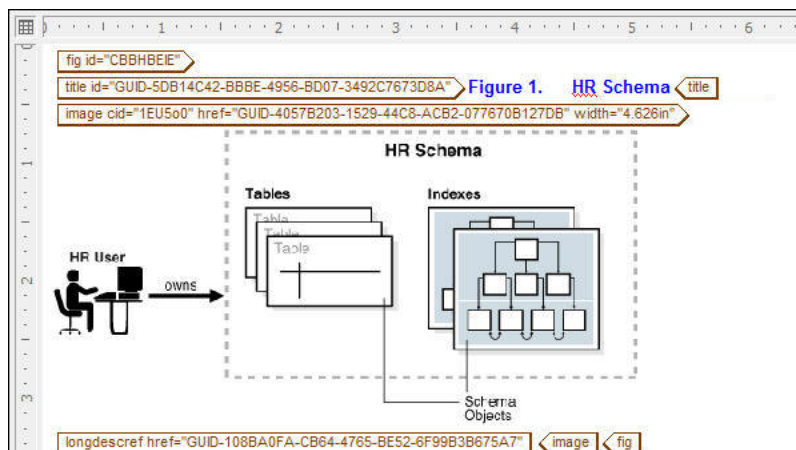
Required Accessibility Elements for Figures

The following table indicates the element or attributes that are required to make formal figures accessible:

Element	Purpose
<code>alt</code>	Contains an alternate text description equivalent to the information in the image. Use this element when describing simple graphics such as navigational arrows or buttons.
<code>longdescref</code>	References a Graphic Description topic that contains the description of the image through its <code>href</code> attribute. Use this element when describing complex graphics, such as screenshots or diagrams.

Note: Follow these guidelines as you write your alternate image description:

- Use statements such as “at upper left is a...” or “the navigation tree on the left...” to help readers understand the layout of the picture.
- Do not use the word *graphic* because it is a keyword for most screen readers. Use *image*, *illustration*, or *figure* instead.
- Avoid including the image file name in the description.. Otherwise, if you change the image file name, then you must update the description.

Figure C-3 Accessibility Elements in a Figure

Graphic Accessibility Checklist

In DITA, graphics comply with accessibility requirements by using the `alt` or `longdescref` tag within their element structure. The element must provide a description in a way that users can understand the layout of an image, the flow of a diagram, or the action that is illustrated, without actually viewing the image.

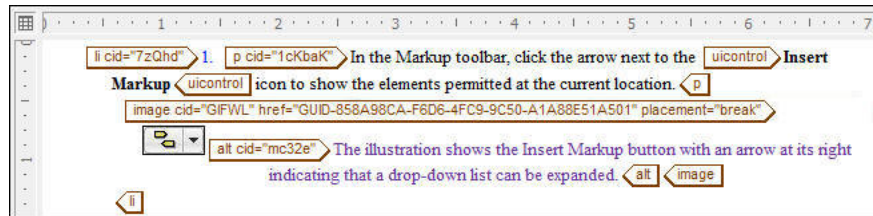
Required Accessibility Elements for Graphics

The following table indicates the element or attributes that are required to make formal tables accessible:

Element	Purpose
<code>alt</code>	Contains an alternate text description equivalent to the information in the image. Use this element when describing simple graphics such as navigational arrows or buttons.
<code>longdescref</code>	References a Graphic Description topic, that contains the description of the image, that contains the description of the image, through its <code>href</code> attribute. Use this element when describing complex graphics, such as screenshots or diagrams.

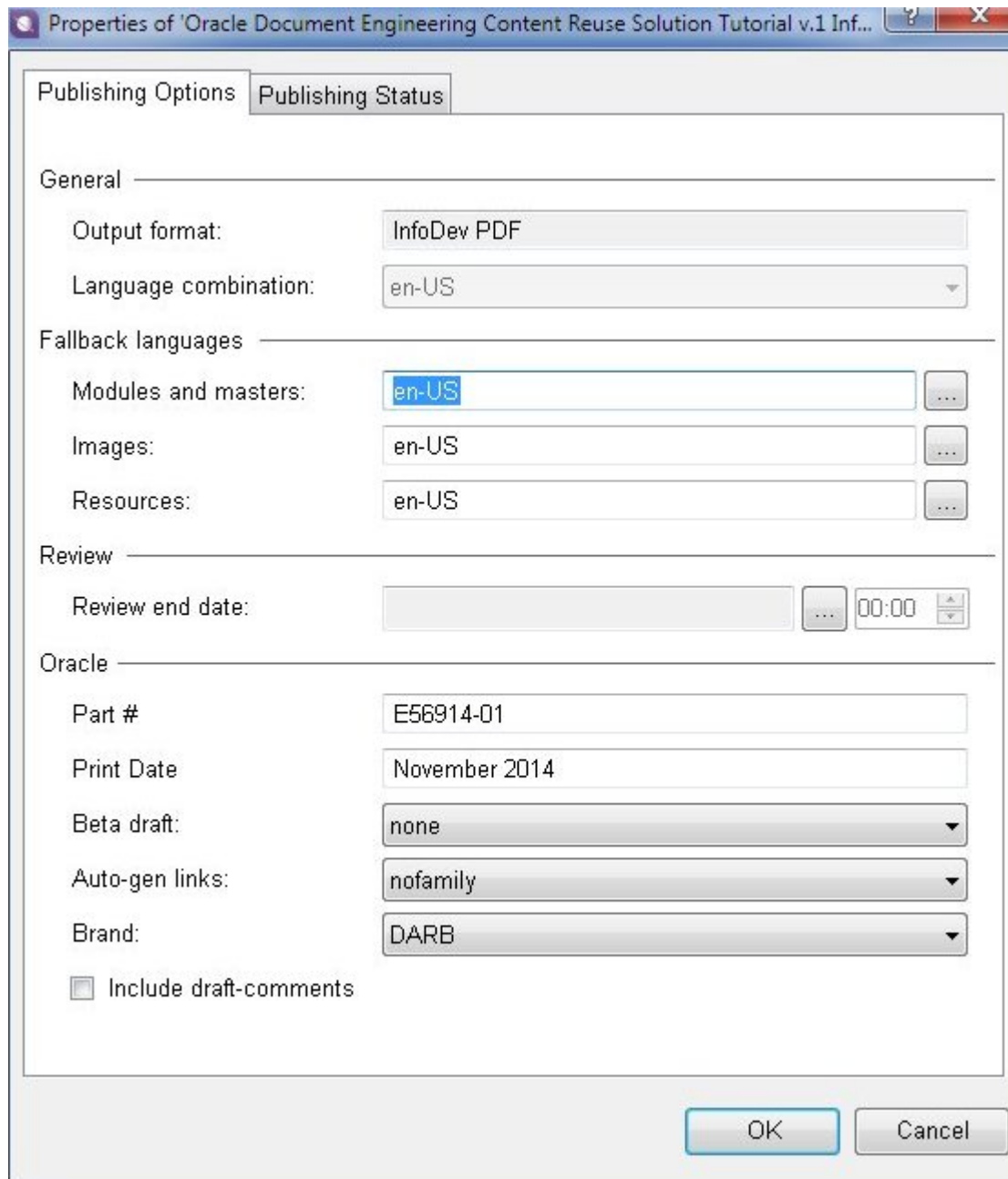
Note: Follow these guidelines as you write your alternate image description:

- Use statements such as “at upper left is a...” or “the navigation tree on the left...” to help readers understand the layout of the picture.
- Do not use the word *graphic* because it is a keyword for most screen readers. Use *image*, *illustration*, or *figure* instead.
- Avoid including the image file name in the description. Otherwise, if you change the image file name, then you must update the description.

Figure C-4 Accessibility Elements in a Graphic

About Oracle Publishing Options

Set the Oracle publishing options in the Publishing Options tab.



Oracle Properties for Publishing Options

When you add output formats for a publication, such as PDF and XHTML, you must specify publishing options for the output. The publishing options include properties that are specific to Oracle publications.

Table C-2 Oracle Properties for Publishing Options

Oracle Property	Description
Part #	The part number of the publication, including the revision number. For example, A12345-01.
Print Date	The print date of the publication, which is usually the current month and year. For example, July 2014.
Beta draft	Select <code>none</code> if the output is for a production release. Select <code>external</code> if the output is for a public beta release. Select <code>internal</code> if the output is for a private beta release.
Auto-gen links	Select <code>all</code> to automatically generate links from parent topics to their child topics. Each link gets a link preview, which is text just below the link that is the short description from the target topic. Links are generated for parent topics at all levels (chapter, H1, H2, and so on). Select <code>nofamily</code> to specify that no automatic links to child topics are generated. Use this value for most converted books, because they already have hand-coded internal TOCs in parent sections. Select <code>none</code> if you do not want to use automatically generated links.
Brand	
Include draft-comments	Select the check box to include the content in <code>draft-comment</code> tags in the output.

DITA Standards for Topic Types

All InfoDev documentation is built using only four topic types: task (`<task>`), concept (`<concept>`), reference (`<reference>`), and orientation (which uses `<topic>`). A glossary uses a different element (`<glossgroup>`), but is not considered a separate topic type.

DITA Standards for Task Topics

This section describes the standards for the `<task>` element. These standards are specific to the InfoDev implementation of DITA. They are intended for all authors in InfoDev.

For an introduction to task topics, see Chapter 2 in *DITA Best Practices* by Laura Bellamy. See also the DITA specification for task elements.

About Task Topics

A task topic, which uses the `<task>` element, explains how to achieve a goal using a procedure. A task answers the question, “How do I?” It is the most common topic type.

An effective task is short, retrievable, and reusable. If too much conceptual or reference information appears in a task, then expert users must wade through information that they might not need.

A task topic contains exactly one task. A task contains exactly one procedure. A template for a task topic is located in `System/Editor template/Topics`.

About Procedures

A procedure is a sequence of steps. Every task must have exactly one procedure, wrapped in either `<steps>` for a multi-step procedure, or `<steps-unordered>` for a single-step procedure.

As shown in the template, a procedure may be preceded by context and prerequisites, and followed by postrequisites, expected results, and examples. Each procedure step includes the following:

- A command in active voice (such as “Click **Go**”)
- Optional information, usually in the form of a paragraph element under the step
- Optional step results, usually the system response

About Complicated Tasks

Procedures that are more than 9 steps can be difficult to follow, especially when some steps are complex or have many sub-steps. If a complicated task consists of multiple separate tasks, then place each task in a separate topic, and use an orientation topic type to order the subordinate tasks. A benefit of this approach is that you can more easily reuse these shorter task topics elsewhere in the documentation set.

Main Elements in Task Topics

The main elements used for task elements include the task element, which encompasses the standard title, shortdesc, and prolog elements, followed by the task element. Within the task element is a collection of task-specific elements that are designed to accommodate prerequisites, the task itself, results, and examples.

Purpose

The `<task>` element is the top-level element for a task topic. A task provides either single-step or multi-step instructions. A task answers the question of “how to?” by explaining what to do and the order in which to do it.

Element Prototype

The following shows the typical sequence of elements for a two-step procedure:

```
<task>
  <title></title>
  <shortdesc></shortdesc>
  <prolog></prolog>
  <taskbody>
    <context></context>
    <prereq></prereq>
    <steps>
```

```

        <step><cmd></cmd><stepresult></stepresult></step>
        <step><cmd></cmd><stepresult></stepresult></step>
    </steps>
    <result></result>
    <example></example>
</taskbody>
</task>

```

Main Elements

This table shows the most relevant elements for tasks.

Task Topic Element	Mandatory ?	Description
<task>	Yes	Parent element, which encompasses all elements in the task topic
<shortdesc>	Yes	Standard short description element
<prolog>	No	Standard prolog element
<taskbody>	Yes	Includes a set of task-specific elements, which are described in the remainder of this table
<prereq>	No	Describe prerequisites the procedure may need. If you must refer the user to another section, then put the reference links needed for prerequisite in the related-links section, not in the prerequisite paragraph. You can use many common elements in the <prereq> element, such as ordered lists, unordered lists, and paragraphs.
<context>	No	Provides a purpose for the task. Use this element to describe what users will gain from completing the task. Though the context element may have some conceptual information, do not use it in place of a related concept topic.
<steps>	Yes unless <steps-unordered> used	Provides an ordered list of steps that the user must follow. The steps element contains individual step elements. Each step in a steps element appears as a numbered step. <steps> and <steps-unordered> are mutually exclusive.
<steps-unordered>	No unless <steps> not used	Used for a single-step procedure. Each step element under the steps-unordered element renders as a bulleted item. You can use one or more step element within the steps-unordered element. <steps> and <steps-unordered> are mutually exclusive.
<step>	Yes	Displays a step in either a multi-step (<steps>) or single-step (<steps-unordered>) procedure.
<substeps>	No	Describes each sub-step that a user must follow to complete the step. It has the same structure as the step element, but it cannot contain another level of <substeps> or the choice or choice-table element.

Task Topic Element	Mandatory ?	Description
<substep>	No	Breaks a step into a series of actions. This element contains one or more individual <substep> elements. Use the <substeps> element only when necessary.
<cmd>	No	Describes the action that the user must take in a step element. Write this action using imperative voice. This element must be the first element in the step element.
<choices>	No	Used within the <steps> element; the choices element provides a choice between one or more actions. It contains individual choice elements.
<choice>	No	Includes each choice that a user is presented with making. The choice items can be bullets when they are rendered.
<choicetable>	No	Similar to the choices element except that it provides two-column table so that you can include a description for each choice
<tutorialinfo>	NOT USED	Enables you to provide additional instructional information if the task is part of a tutorial. It appears after the <cmdname> element in the step element.
<stepresult>	No	Describes the result of completing a step, such as a dialog box opening. Use <stepresult> to assure users that they are on the right track, but do not use them for every step. For example, "The Create User page appears, with CREATE SESSION listed as the system privilege for user mweiss."
<info>	No	Provides additional information that users might need when they complete a step. The description must be brief and only contain minimal conceptual information. For example: "You do not need to specify additional users."
<stepxmp>	No	Provides an example of what happens when a step is completed. The example can include a few words, a paragraph, a figure, a table, or other information to illustrate the task. Step examples often provide specific data-entry characters.
<stepsection>	No	Provides expository text before a step element. Though the element is specialized from the li element and has the same content model as a list item, it is not designed to represent a step in a task. You can use either declarative or imperative voice. For example: "The remaining steps are very important." <ul style="list-style-type: none"> "The remaining steps are very important." "If you had installed the DBMS_CRMB package, then complete the remaining steps."
<result>	No	Provides the expected outcome of the task. It can include a final description that uses figures, tables, or audiovisual

Task Topic Element	Mandatory ?	Description
		cues that show the user that he or she has successfully completed the task. Add the result element after the steps element. For example, “After you complete this task, all user accounts are created.”
<code><example></code>	No	Displays the expected outcome of the entire task. It can include code samples, figures, screen shots, and other samples that show users that they have completed the task. The example element includes a title element, and a p (paragraph) element. From there, you can add the necessary elements you need, such as the <code><codeblock></code> element if you need to show code output.
<code><postreq></code>	No	Provides information that users should know after they have completed a task, or information about tasks that must be completed next. This kind of information can include actions that must be completed before the user can see expected results, such as restarting the computer. It can include information about what the user must read or cross-reference to verify that they properly completed the task. The <code><postreq></code> element often has links to the next task or tasks in the related-links section. For example: “After you enable Oracle Label Security, you must unlock the LBACSYS user account.”

Templates

Templates for a task and task example are located in `System/Editor template/Topics`. InfoDev recommends that you create new task topics using the templates.

Guidelines for Task Topics

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating task topics.

Consider the following guidelines when creating a task topic:

- Create a new task using the task template.
When creating a new topic, remember to delete unused elements inherited from the template. If you do not, empty lines can appear in the HTML and PDF output.
- Begin the `<title>` element with a gerund, as in “Creating a User” or “Backing Up a Tablespace.”
If relevant, put the goal of the task in the title, as in “Creating a User to Administer a PDB.” The goal is to make the title as informative as possible.
- Create a short description for the task topic (mandatory).
The short description appears in the HTML and PDF output as the first paragraph of the topic. Do not create a first paragraph for your topic that merely restates the short description.
- Create exactly one step-by-step procedure for every task topic (mandatory).

For multi-step procedures, use the `<steps>` element. In all other cases, including single-step procedures, use the `<steps-unordered>` element.

- Break complex procedures into smaller, individual task topics.

Procedures with more than 9 steps are difficult to follow. Create a parent topic, sometimes known as a high-level task, that describes the overall task flow. You can then nest the child topics under the high-level task in a logical order.

Guidelines for Content Surrounding a Procedure

Follow these guidelines when putting text before or after the procedure:

- Provide no more content before the procedure than is necessary to make sense of the procedure.

Sometimes the title and short description are sufficient in themselves to provide context. If the goal of the task is not apparent in the title or the short description, then you can introduce the procedure with the “To do X:” convention. In all other cases, use the `<context>` element for background information, and the `<prereq>` element for prerequisites. The `<prereq>` element is wrapped inside the `<taskbody>` element.

- Write prerequisites as imperative statements that call users to action.

Include links to tasks that users must complete before beginning the current task.

- Provide content after the procedure when appropriate using `<postreq>` (what to do next), and `<result>` (the change brought about by the procedure).

Often these elements are not needed. Ask, “Does the user really need to know this?”

- To create an example after a procedure, use the `<example>` element; to create an example after a step within a procedure, use the `<stepxmp>` element.

Guidelines for Steps in a Procedure

Follow these guidelines for elements within `<steps>` and `<steps-unordered>`:

- Begin each step with location context (when necessary), followed by an imperative.

For example, write “In the User window, click Create.” If the context is not needed, as in a sequence of steps performed on the command line, then begin with an imperative, as in “Run the BACKUP DATABASE command.” Place commands within the `<cmdname>` element. Ensure that a step never contains only an indicative statement such as “Now you back up the database.”

- When a step is optional, select `Modify Attributes`, select `Other`, and then set importance to `optional`.

In the HTML and PDF output, modifying this attribute generates a bold **Optional**: flag at the beginning of the step. Do not insert the literal word “Optionally” at the beginning of the step. Otherwise, the word “Optional” appears twice in the output.

- In a step that introduces sub-steps, use an imperative that states the goal of the sub-steps, such as “Load the statements into the SQL tuning set as follows.”

Avoid sentences such as “Follow these steps” as the introduction to sub-steps because such sentences give users no idea what action the sub-steps will describe.

- When a step presents the user with options, as in “Choose one of the following,” use the <choices> or <choicetable> element within the <step> element.
- When a step has sub-steps, use the <substep> element within the <step> element.
- To describe the outcome of a single step, use the <stepresult> element within <step>.

Use <stepresult> only for results are important or not obvious. For example, if the step closes a dialog, then do not add the step result “The dialog box closes.” Consider not inserting step results such as “The User dialog box appears.” Instead, in the next step, start with UI location, as in “In the User dialog box, create Create.”

- When instructing the user to select an item from a drop-down list, use the following form: “From the *drop_down_list_name* drop-down list, select *item_name*.” For example: From the **Content type** drop-down list, select **Module**.

Examples of Task Topics

These examples show some of the different elements that you can use within <task>.

Example C-3 Simple Multistep Task: Example

```
<task>
<title>Backing Up Server Parameter Files with RMAN</title>
<abstract>RMAN automatically backs up the current server parameter
file in certain cases. <shortdesc>The BACKUP SPFILE command backs
up the parameter file explicitly. The server parameter file that is
backed up is the one currently in use by the instance.</shortdesc></
abstract>
<taskbody>
<steps>
  <step>Start RMAN and connect to a target database and a recovery
catalog (if used).</step>
  <step>Ensure that the target database is mounted or open.
    <info>The database must have been started with a server
parameter file. If the instance is started with a client-side
initialization parameter file, then RMAN issues an error if you
execute <cmdname>BACKUP ... SPFILE</cmdname>.</info>
  </step>
  <step>Execute the <cmdname>BACKUP ... SPFILE</cmdname> command.
    <info>The following example backs up the server parameter file
to tape:</info>
  <stepxmp>BACKUP DEVICE TYPE sbt SPFILE;</stepxmp>
</step>
</steps>
</taskbody>
</task>
```

Example C-4 Simple Single-Step Task: Example

```
<task>
<title>Changing the Fast Recovery Area to a New Location</title>
<shortdesc>The fast recovery area is an Oracle Database managed
space that can be used to hold RMAN disk backups, control file
autobackups and archived redo log files.</shortdesc>
<taskbody>
<p>To move the fast recovery area of your database to a new
location:</p>
<steps-unordered>
  <step>In SQL*Plus, change the DB_RECOVERY_FILE_DEST
initialization parameter.</step>
  <info>For example, enter the following command to set the
```

```
destination to the ASM disk group disk1:</info>
  <stepxcmd>ALTER SYSTEM SET DB_RECOVERY_FILE_DEST='+disk1'
SCOPE=BOTH SID='*';</stepxcmd>
</step>
</steps-unordered>
<result>After you change the DB_RECOVERY_FILE_DEST initialization
parameter, all new fast recovery area files are created in the new
location.</result>
</taskbody>
</task>
```

DITA Standards for Concept Topics

This section describes the standards for the `<concept>` element. These standards are specific to the InfoDev implementation of DITA. They are intended for all authors in InfoDev.

For an introduction to concept topics, see Chapter 3 in *DITA Best Practices* by Laura Bellamy. See also the DITA specification for concepts elements.

About Concept Topics

A concept topic, which uses the `<concept>` element, explains or defines an idea as a means of providing background for tasks or reference topics.

A concept answers the following questions:

- “What is it?”
- “What is its purpose?”
- “How does it work?”

A good concept topic supports a task or reference, rather than the other way around. Because one of the functions of a concept is to define terms, a concept often cross-references glossary entries. However, a concept topic is not a list of glossary terms.

A concept differs from a glossary entry because of its length, the type of elements it contains, and its purpose. A typical glossary entry might define the concept and optionally state its purpose in more than three sentences. A concept topic usually has multiple paragraphs, and may contain bulleted lists, graphics, and tables.

A concept differs from a reference topic in its purpose. A reference topic either defines a UI component, or lists guidelines. A concept explains what something is and why it is important. A good analogy is the difference between a dictionary entry (reference topic) and an encyclopedia entry (concept topic).

A concept topic contains exactly one concept. A concept must never contain a step-by-step procedure or any other task information. A template for a concept topic is located in `System/Editor template/Topics`.

About Complicated Concepts

If you find that a topic describes more than one concept, then break this information into separate concept topics. Use an orientation topic to mention how these topics are related. The advantage of separating concepts is that users can read only what they need, and you can more easily reuse those concept topics elsewhere in your documentation.

About Irrelevant Concepts

Concept topics must only have information that is relevant to user tasks or reference. Because the primary source of doc is the functional spec, it is tempting to dump the concepts in a spec into a topic. However, concepts relevant for developers may not be relevant for users. Ask, “Does this concept have implications for a user task?” If a user cannot act on the information, then although the concept may be interesting, it is probably irrelevant and should be removed.

Main Elements in Concept Topics

The `<concept>` element is the top-level element for a concept topic. It encompasses the standard `<title>`, `<shortdesc>`, and `<prolog>` elements, followed by the `<conbody>` element. Within `<conbody>` you can use elements such as paragraphs, lists, tables, figures, examples, and sometimes sections.

Purpose

A concept provides background information for a task or reference topic, or introduces a feature. Unlike a reference topic, a concept explains and illustrates. A concept answers the following questions:

- “What is it?”
- “What is its purpose?”
- “How does it work?”

Element Prototype

The following shows a typical sequence of elements for a concept topic:

```
<concept>
  <title></title>
  <shortdesc></shortdesc>
  <prolog></prolog>
  <conbody>
    <p></p>
    <ul>
      <li><p></p></li>
      <li><p></p></li>
    </ul>
  </conbody>
</concept>
```

Main Elements

This table shows the most relevant elements for concepts.

Concept Topic Element	Mandatory ?	Description
<code><concept></code>	Yes	Parent element, which encompasses all elements in the concept topic
<code><shortdesc></code>	Yes	Standard short description element
<code><prolog></code>	No	Standard prolog element
<code><conbody></code>	Yes	Includes a set of concept-specific elements, which are described in the remainder of this table

Concept Topic Element	Mandatory ?	Description
<code><section></code>	No	Represents an organizational division in a topic. Use sections to organize subsets of information that are directly related to the concept. Sections are not hierarchical and cannot be nested. A section may have an optional title.
<code><sl></code>	No	Contains a simple list of items of short, phrase-like content, such as a list of characteristics. The output has no bullets.
<code></code>	No	Provides an unordered, bulleted list of items.
<code><dl></code>	No	Provides a list of terms and corresponding definitions. The term (<code><dt></code>) is usually flush left. The description or definition (<code><dd></code>) is usually either indented and on the next line, or on the same line to the right of the term. You can also provide an optional heading for the terms and definitions, using the <code><dlheadelement></code> , which contains header elements for those columns.
<code><codeblock></code>	No	Displays of program code. Like the <code><pre></code> element, content of this element has preserved line endings and is output in a monospace font.
<code><lines></code>	No	Represents dialogs, lists, text fragments, and so on. The <code><lines></code> element is similar to <code><pre></code> in that hard line breaks are preserved, but the font style is not set to monospace, and extra spaces inside the lines are not preserved.
<code><msgblock></code>	No	Contains a multi-line message or set of messages. The message block can contain multiple message numbers and message descriptions, each enclosed in a <code><msgnum></code> and <code><msgph></code> element. It can also contain the message content directly.
<code><screen></code>	No	Contains or refers to a textual representation of a computer screen or UI panel (window). Use <code><screen></code> to contain representations of text-based online panels, text consoles ("term" or "curses" windows, for example), or other text-based UI components. The default print representation is to enclose the screen within a box, suggesting a computer display screen. In contrast to graphical screen captures normally used to represent GUI parts (see the image element description), this element specifically supports constructions for which text is the primary content.
<code><fig></code> and <code><image></code>	No	A <code><fig></code> is a formal graphic element, requiring a title and an id, whereas <code><image></code> is informal. A <code><fig></code> , but not an <code><image></code> , can be cross-referenced. Use the <code><image></code> element to insert a graphic into a <code><fig></code> . The <code>@href</code> attribute points to the relative path location of the graphic file. Optionally, use the <code><alt></code> child element for alt text, similar to the FrameMaker AltText attribute. For a graphic description, use either <code><alt></code> or <code><longdesc></code> , both of which are associated with <code><image></code> .

Concept Topic Element	Mandatory ?	Description
		Set @exppanse to “page” to get a wide figure. Set @placement to “break”.
<table>	No	Organizes arbitrarily complex relationships of tabular information. This standard table markup allows column or row spanning and table captions or descriptions. An optional title allowed inside the table element provides a caption to describe the table. In addition, the table includes a <desc> element, which enables table description that is parallel with figure description.
<term>	No	Identifies glossary entries. To create a link for a glossary term, wrap the text in your topic in a <term> element. Modify the <term> attribute @keyref to associate the keyref with a <glosskey> element in the bookmap. A <glossarylist> is located in the bookmap, and has multiple <glossentry> elements as children. Each <glossentry> must be in its own topic, and typically, its own file. Create a <glossentry> topic by selecting the topic type “InfoDev DITA Glossary.” A <glossentry> contains a <glossterm> and a <glossdef> .

Templates

There is only one concept template. It is located in `System/Editor template/Topics`. InfoDev recommends that you create new concept topics using the template.

Guidelines for Concept Topics

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating concept topics.

Consider the following guidelines when creating a concept topic:

- Write one concept for each topic.
If a topic becomes too complex, break it into multiple concept topics, and group them using an orientation topic.
- Begin the title with a noun or noun phrase, as in “About User Privileges,” “Overview of User Privileges,” or “User Privileges.” Whether you use “About” or “Overview” is up to your discretion.
- Never include a step-by-step procedure for the user to follow. Tasks must go in a task topic.

However, a concept topic may use ordered lists to describe stages in a process (Oracle Database creates the SGA and background processes, and then opens the control file, and then opens the database).

- For <section> elements, give each a <title> element using the same naming conventions as the concept topic. Use sections judiciously.

The <conbody> element is the main body-level element for a concept. Like the <body> element of a general <topic>, <conbody> allows paragraphs, lists, and other elements as well as sections and examples. However, <conbody> has a

constraint such that a section or an example can be followed only by other sections or examples.

- Most concept topics benefit from a figure, although it is not required.

Guidelines for Ordering Conceptual Information

When deciding how to organize a concept topic, consider these guidelines:

- Use an inverted pyramid style, that is, put the most important idea at the beginning of the topic, and supplemental explanation afterward. Do not build up to the point. Get to the point straight away.
- Use the acronym “DRIES” as a rule of thumb. DRIES stands for: Definition, Reason (purpose), Implementation (how it works, what it contains), Example, and See Also (a <reltable>). For example, in “About Tablespaces,” begin by defining “tablespace,” then explain what the point of a tablespace is, explain how tablespaces work and what their components are, give an example of a CREATE TABLESPACE statement, and end with a <reltable> of related links.
- If a concept is too big to fit in one topic, create subtopics for Purpose, How It Works, and so on.

Reasons for Creating Concept Topics

When considering whether to create a concept topic, consider these guidelines:

- Create a concept only if a glossary entry is inadequate.
If a concept is adequately covered in one or two sentences, then make it part of a task, reference, or glossary instead. If you need more than a paragraph, or you need to explain how something works, then create a separate concept topic.
- Create concept topics to support tasks and goals, not the other way around.
Users read technical information to achieve a goal. The goal for users of most technical products is not to understand a concept but to complete a task, such as creating a tablespace, backing up a database, or dropping a user.

Examples of Concept Topics

These examples show some of the different elements that you can use within <concept>.

Example C-5 Simplest Concept: Example

In its simplest form, a concept topic has a short description, but no text within the <conbody>. The short description appears in the output as the first paragraph of the topic.

```
<concept>
<title>Savepoints</title>
<prolog>
  <metadata>
    <keywords>
      <indexterm>[index: savepoints]
      </indexterm>
      <indexterm>[index: transactions
                  <indexterm>, savepoints</indexterm>]
      </indexterm>
    </keywords>
  </metadata>
</prolog>
```

```

    </metadata>
  </prolog>
  <shortdesc>A savepoint is a user-declared intermediate marker
  within the context of a transaction. Internally, this marker
  resolves to an SCN. Savepoints divide a long transaction into
  smaller parts. If you use savepoints in a long transaction, then
  you have the option later of rolling back work performed before the
  current point in the transaction but after a declared savepoint
  within the transaction.</shortdesc>
  <conbody></conbody>
</concept>

```

Example C-6 Concept with Bulleted Lists: Example

This topic provides an overview of tables. There is no “how to” information, which must never appear in a concept. Notice also how the topic is relatively short even though tables are a huge topic. The related conceptual information appears in separate subtopics. For example, there is one subtopic on columns, another on rows, another on data types, another on integrity constraints, and so on.

```

<concept>
<title>Overview of Tables</title>
<shortdesc>A table is the basic unit of data organization in an
Oracle database. A table describes an entity, which is something of
significance about which information must be recorded. For example,
an employee could be an entity.</shortdesc>
<prolog>
  <metadata>
    <keywords>
      <keyword>tables</keyword>
    </keywords>
  </metadata>
</prolog>
<conbody>
<p>Oracle Database tables fall into the following basic categories:
<ul>
  <li><p>Relational tables</p>
    <p>Relational tables have simple columns and are the most
common table type. Example 2-1Example 2-1 shows a CREATE TABLE
statement for a relational table.</p></li>
  <li><p>Object tables</p>
    <p>The columns correspond to the top-level attributes of an
object type. See Overview of Object Tables.</p></li>
</ul>
<p>You can create a relational table with the following
organizational characteristics:
<ul>
  <li><p>A heap-organized table does not store rows in any
particular order. The CREATE TABLE statement creates a heap-
organized table by default.</p></li>
  <li><p>An index-organized table orders rows according to the
primary key values. For some applications, index-organized tables
enhance performance and use disk space more efficiently. See
Overview of Index-Organized Tables.</p></li>
  <li><p>An external table is a read-only table whose metadata is
stored in the database but whose data is stored outside the
database. See Overview of External Tables.</p></li>
</ul>
<p>A table is either permanent or temporary. A permanent table
definition and data persist across sessions. A temporary table
definition persists in the same way as a permanent table
definition, but the data exists only for the duration of a
transaction or session. Temporary tables are useful in applications
where a result set must be held temporarily, perhaps because the
result is constructed by running multiple operations.</p>

```

```
</conbody>  
</concept>
```

DITA Standards for Reference Topics

This section describes the standards for the `<reference>` element. These reference topic standards are specific to the InfoDev implementation of DITA. They are intended for all authors in InfoDev.

For an introduction to reference topics, see Chapter 4 in *DITA Best Practices* by Laura Bellamy. See also the DITA specification for concepts elements.

About Reference Topics

A reference topic, which uses the `<reference>` element, provides quick access to a set of facts (UI components or rules), without providing explanation.

A reference topic answers either of the following questions:

- What are the syntax and semantics of this UI component?
- What is the list of rules, restrictions, or guidelines for a task or reference topic?

A reference topic gives “facts without explanation,” which distinguishes it from a concept. A UI reference functions as a dictionary: users want to quickly find a definition, and then move on. A rules reference is analogous to a home appliance data sheet: a list of sentences of the form “Do this” and “Don’t do that.” A concept topic is similar to an wikipedia entry, which introduces a topic, states its purpose, and explains how it works using lists, figures, tables, and so on.

A reference also gives “facts without steps.” Thus, in contrast to task topics, reference topics never have procedures that explain how to achieve a goal.

In software documentation, UI reference topics describe components such as the following:

- SQL statements
- Commands
- Packages
- Utilities
- Error Messages
- CLI or GUI components

Topics enable users to learn the specifics of APIs, such as their syntax, input, expected output. A UI reference topic has a short description, followed by the reference material, which can include sections for items such as Overview, Security Model, Constants, Restrictions, Exceptions, Operational Notes, and Examples. In most cases, place each API, if it is small, into its own reference topic.

A reference topic contains exactly one type of reference information. A reference must never contain a step-by-step procedure or any other task information. The templates for reference topics are located in `System/Editor template/Topics`.

Main Elements in Reference Topics

The main elements for reference topics include the `<reference>` element, which encompasses the standard `<title>`, `<shortdesc>`, and `<prolog>` elements,

followed by the `<refbody>` element. Within `<refbody>` is a collection of reference-specific elements that present prerequisites, purpose sections, syntax, properties, examples, and so on.

Purpose

Reference topics document facts (either UI components, or lists of rules) without explanation. A reference topic answers either of the following questions:

- “What does this UI component do?”
- “What are the rules associated with the task or reference?”

Element Prototype

The following shows a typical sequence of elements for a reference topic:

```
<reference>
  <title></title>
  <shortdesc></shortdesc>
  <prolog></prolog>
  <refbody>
    <section><title>Purpose</title></section>
    <refsyn><title>Syntax</title></refsyn>
    <section><title>Parameters</title></section>
  </refbody>
</reference>
```

Main Elements

This table shows the most relevant elements for reference topics. Note that the `<p>` element is not permitted unless embedded in wrapping elements such as `<section>` and `<refsyn>`.

Concept Topic Element	Mandatory?	Description
<code><reference></code>	Yes	Parent element, which encompasses all elements in the reference topic
<code><shortdesc></code>	Yes	Standard short description element
<code><prolog></code>	No	Standard prolog element
<code><refbody></code>	Yes	Includes a set of reference-specific elements, which are described in the remainder of this table
<code><section></code>	No	Represents an organizational division in a topic. Use this element each subsection in the topic (such as Prerequisites, Properties, Syntax, Examples). Sections are not hierarchical and cannot be nested. A reference section must have a title. Do not embed <code><p></code> elements in sections without titles as a workaround for adding paragraphs.
<code><table></code>	No	Contains information such as parameter descriptions.
<code><properties></code>	No	Lists properties in a table by type, value, and description. For example, if you document how to design HTML pages, you might create a reference topic called “Cascading style sheet properties for body text” that includes a properties table describing the different cascading style sheet (CSS) properties for

Concept Topic Element	Mandatory?	Description
		<p>fonts. The table could describe font properties such as family, style, weight, or size.</p> <p>To represent multiple values for a type, create additional property elements, use only the <code><propvalue></code> element (and <code><propdesc></code> when needed) for each successive value, and specify the <code><proptype></code> element only the first time.</p>
<code><refsyn></code>	No	<p>Contains (usually) syntax or signature content. Examples are a command-line utility's calling syntax, and an API's signature. Typically, <code><refsyn></code> contains a brief, possibly diagrammatic description of the subject's interface or high-level structure. Use the <code><syntaxdiagram></code> element inside a <code><refsyn></code> element to structure syntax diagrams. The syntax diagram may be included as an image, or coded in DITA within <code><synph></code>.</p>
<code><example></code>	No	<p>Contains examples that illustrate or support the current topic. An <code><example></code> can contain both discussion and sample code or outputs.</p> <p>Use <code><codeblock></code> for sample code. Use <code><systemoutput></code> for output examples. Use <code><lines></code> for lines of text. For pre-formatted text such as email headers, use the <code><pre></code> element.</p>

Templates

Templates for reference topics are located in `System/Editor template/Topics`. InfoDev recommends that you create new concept topics using the template. The following reference templates are available:

- Reference
- Reference API
- Reference Command
- Reference Parameter
- Reference PL/SQL Package
- Reference SQL Statement
- Reference View

Guidelines for Reference Topics

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating reference topics.

Consider the following general guidelines when creating a reference topic:

- Use a noun phrase for the title. Examples include "Analytic Functions," "ALTER TABLE," "Database Resource Manager Data Dictionary Views," and "DBMS_REDACT". API reference topics must use the name of a command or API program unit.
- Do not include explanations or procedures. Reference topics are intended for quick lookups of facts. The mantra is "facts without explanation."

- For most reference information, use lists and tables rather than series of paragraphs.
- Paragraphs are not valid unless embedded in other elements. Do not create a section and then delete the `<title>` just to get around this restriction. (See “Guidelines for Sections” for a legitimate case in which you can delete `<title>`.)
- For very short reference topics, do not use the section element. An example might a short topic that contains one type of reference information, such as a table of predefined user roles.
- Do not use the `` element when a list has two parts. Use a table instead.
- Format the reference material consistently throughout.

For example, if you create several reference topics that have tables of procedure parameters, then use the same formatting for the tables, use the same introduction to the tables, and provide the same amount of information. Even small inconsistencies are distracting and potentially confusing.

Guidelines for Chunking

When deciding whether or how to break up reference information into separate topics, consider the following guidelines:

- Describe one type of reference material per topic, even for small APIs, as long as it is 1–2 pages.

For example, for the `DBMS_ALERT` topic, you would describe only API settings for the `DBMS_ALERT` package.

- If the material is long (the rule of thumb is over 2 pages), then use multiple tables or multiple topics for each subcategory.

For example, for a PL/SQL package, create one topic for each subprogram, not one topic for the entire package. Consider how the information will be searched for. For example, for large APIs, you could put each method into its own reference topic.

- When you are unsure whether to create a separate topic, consider how users will search for the information, and also whether the information is likely to grow over releases. If the information grows by 50% every release, then consider putting it into its own topic.

Guidelines for Tables

When creating tables in a reference topic, consider the following guidelines:

- Always use the `<table>` element, even if the table is very simple. Do not use the `<simpletable>` element. Provide the required table summary as a `<desc>` element.
- For table titles, use specific names, such as “System Privileges (Organized by the Database Object Operated Upon)”. For a very simple table in a small reference topic, omit the title. It is not necessary.
- Do not have too much information in the table. It becomes difficult for users to read.
- Do not have more than four or five columns in a table.

- Do not make the last column a catch-all column for comments, examples, and descriptions.

Guidelines for Syntax Diagrams

When creating a syntax section, consider the following guidelines:

- Place API calling syntax and method signatures within a `<refsyn>` section.
- You can include syntax as an image, or code it in DITA within a `<synph>` element.

Guidelines for Sections

When creating a `<section>`, consider the following guidelines:

- Use the `<section>` element in the `<refbody>` element for each section, such as “Prerequisites,” “Syntax,” and “Examples.”
- Always use exactly one `<title>` within each `<section>`. The only exception is when a topic is so short that it has no sections, and you cannot use necessary elements without deleting the `<title>`.
- You cannot nest `<section>` elements. They are not hierarchical.

Guidelines for Examples

When using the `<example>` element, consider the following guidelines:

- Use `<example>` to create a section that contains examples that illustrate or support the current topic. You can title an example, and use many elements such as lists, tables, and paragraphs within `<example>`.
- Use `<codeblock>` for sample code.
- Use `<systemoutput>` for output examples.
- Use `<lines>` for lines of text.
- Use the `<pre>` element for preformatted text such as email headers.

Guidelines for Property Tables

When creating a list of properties, consider the following guidelines:

- Use `<properties>` to provide a table of properties for the topic.
- Each property can include the type, value, and a description.
- To represent multiple values for a type, create additional property elements. Use only the `<propvalue>` element (and `<propdesc>` when needed) for each successive value. Specify the `<proptype>` element only the first time.

Examples of Reference Topics

These examples show some of the different elements that you can use within `<concept>`.

PL/SQL Procedure: Example

```
<reference>
  <title>INITIALIZE Procedure</title>
  <shortdesc>This procedure initializes the generator.</shortdesc>
<refbody>
<refsyn>
  <synph><kwd>DBMS_RANDOM.INITIALIZE (</kwd><var>val</var><kwd>IN
  BINARY_INTEGER);</kwd></synph>
</refsyn>
<section>
  <title>Usage Notes</title>
  <p>his procedure is obsolete because it calls the SEED
  Procedures.</p>
</section>
</refbody>
</reference>
```

DITA Standards for Orientation Topics

This section describes the standards for orientation topics, which use the `<topic>` element (there is no `<orientation>` element). These reference topic standards are specific to the InfoDev implementation of DITA. They are intended for all authors in InfoDev.

For an introduction to reference topics, see Chapter 1 in *DITA Best Practices* by Laura Bellamy. See also the DITA specification for `<topic>`.

About Orientation Topics

An orientation topic, which uses the `<topic>` element, provides a parent topic in the table of contents for a set of related topics, and introduces those topics. Thus, it serves as a high-level topic organizer.

An orientation topic can serve as a high-level task topic. It typically has a heading similar to that of a task topic (starting with a gerund), and is a parent topic for a series of topics that help users complete a high-level task such as installing, configuring, administering, and so on. Such a high-level task is explained further by a series of conceptual topics, task topics, and reference topics, all children of the high-level task topic.

Frequently, an orientation topic contains only the short description as its content. The short description provides introduction to the chapter or section. In some cases, the topic can include a few paragraphs of background or introductory information prior to the internal table of contents.

The topic also contains set of cross references to the contained subsections, which are automatically generated by the DITA Open Toolkit from the titles and short descriptions of the child topics of the orientation topic in the map. The title (as a link) and short description of each child topic are automatically added to the parent (orientation) topic during publishing. In addition, the DITA OT automatically adds a link to the parent topic in each child.

No `<orientation>` tag exists. Instead, use the generic `<topic>` tag for orientation topics.

Main Elements in Orientation Topics

The main elements for reference topics include the `<topic>` element, which encompasses the standard `<title>`, `<shortdesc>`, and `<prolog>` elements, followed by the `<body>` element. Within `<body>` you can place elements such as paragraphs, lists, and so on.

Purpose

The `<topic>` element is the top-level element for an orientation topic. An orientation topic is the high-level container for other topics. An orientation topic answers the question of “How is this group of related topics organized?”

Syntax

The following shows the typical sequence of elements in a simple orientation topic:

```
<topic>
  <title></title>
  <shortdesc></shortdesc>
  <prolog></prolog>
  <body></body>
</topic>
```

Main Elements

This table shows the most relevant elements for tasks.

Task Topic Element	Mandatory ?	Description
<code><topic></code>	Yes	Parent element, which encompasses all elements in the task topic.
<code><shortdesc></code>	Yes	Standard short description element. Often, the short description is the only text required in the orientation topic.
<code><prolog></code>	No	Standard prolog element.
<code><body></code>	Yes	Wraps any additional text necessary to orient the reader.
<code></code> , <code><p></code> , etc.	No	Use any of the standard elements to provide more orientation, if necessary.

Templates

The template for the orientation topic is located in `System/Editor template/Topics`. InfoDev recommends that you create new task topics using the templates.

Guidelines for Orientation Topics

These guidelines provide both requirements (what you must do) and best practices (what it is recommended to do) when creating orientation topics topics.

The orientation topic type provides a parent topic in the table of contents for a set of related topics, and introduces those topics. Consider the following guidelines when creating an orientation topic:

- Try to put all necessary introductory text in the `<shortdesc>`. If you need to add more paragraphs in the `<body>`, then it is completely acceptable; however, in most cases, the `<shortdesc>` is all that is necessary.
- Because the DITA Open Toolkit automatically creates an “internal table of contents,” do not create a list of subtopics manually.

The title (as a link) and short description (link preview) of each child topic are automatically added to the parent (orientation) topic during publishing. In addition, the DITA OT automatically adds a link to the parent topic in each child topic.

Note: If you want to manually put in an internal TOC, then you must disable the automatic generation of these links in the build settings in the output object of the publication or in DocBuilder.

- In the bookmap, set the `@collection-type` attribute of the `<topicref>` for the orientation topic to the value `family`.

This setting causes additional links to be generated within the child topics. Each child topic cross-references all its sibling topics in automatically generated lists of “Related Concepts,” “Related Tasks,” and “Related References.”

- When creating an orientation topic to serve as a high-level task topic, choose a title similar to that of a task topic (starting with a gerund).

An orientation topic is the parent topic for a series of topics that help users complete a high-level task such as installing, configuring, administering, and so on. Such a high-level task is explained by a series of conceptual topics, task topics, and reference topics, all of which are children of the high-level task topic. For example, an orientation topic named “Administering Tablespaces” might be the parent of topics named “About Tablespaces” (a concept), “Creating Tablespaces” (a task), “Taking a Tablespace Offline” (a task), and so on.

- If the orientation topic is not a high-level task topic, then use the same heading title rules as the orientation topic’s child topics. For example, if the orientation topic is the parent of conceptual topics, then use the title rules for conceptual topics.

Examples of Orientation Topics

These examples show some of the elements that you can use within `<topic>` to create an orientation topic.

High-Level Task Orientation: Example

Notice that there is no explicit internal table of contents, which might have been implemented as an unordered list. The heading begins with a gerund because the orientation topic contains task topics.

```
<topic>
  <title>Checking Memory Requirements</title>
  <shortdesc>Your target system must satisfy specific memory
requirements, which include RAM, swap space, and shared memory.</
shortdesc>
  <body>
  </body>
</topic>
```

Index

A

accessibility
 in figures, [C-14](#)
 in formal tables, [C-13](#)
 in graphics, [C-15](#)
 in informal tables, [C-13](#)
Arbortext Editor
 interface, [2-3](#), [3-3](#)

B

baselines
 freezing, [C-12](#)
bookmaps
 adding chapters to, [C-11](#)
 creating, [5-2](#)

C

chapters
 adding to a bookmap, [C-11](#)
checking spelling, [4-4](#)
concept topics
 guidelines, [C-27](#)
concept topics:main elements, [C-25](#)
concepts
 examples, [C-28](#)
creating
 book maps, [5-2](#)
 folders, [3-2](#)
 links, [9-1](#)
 publications, [5-1](#)
 reference topics, [3-10](#)
 task topics, [3-8](#)

D

Darwin Information Typing Architecture, [2-1](#)
DITA *See* Darwin Information Typing Architecture
DocBuilder
 logging in, [A-1](#)

F

figures
 accessibility of *See* accessibility
 formal, [8-5](#)
 informal, [8-2](#)
folders, [3-2](#)
formatting
 semantic tags, [4-1](#), [C-3](#)
 typographical tags, [4-1](#)

G

getting help, [1-3](#)
graphics
 accessibility of *See* accessibility
 importing, [C-7](#)

I

images *See* graphics
installation, [1-1](#)

L

lifecycle
 publications, [C-10](#)
linking
 external documents, [9-5](#)
 using olink: and topic IDs, [9-5](#)
links
 creating, [9-1](#)
logging in, [2-1](#), [A-1](#)

M

master documents, [5-1](#)
metadata, [1-1](#), [5-5](#)

N

notes
 types of, [B-1](#)

O

orientation topics
 examples, [C-37](#)
 guidelines, [C-36](#)
output
 creating, [11-1](#)
output formats
 Oracle properties, [C-17](#)

P

part numbers
 publications, [C-10](#)
passwords
 DocBuilder, [A-1](#)
 SDL, [A-1](#)
practice document, [1-2](#)
prerequisites, [1-1](#)
publications
 adding chapters, [10-2](#), [C-11](#)
 adding topics, [10-2](#)
 lifecycle, [C-10](#)
 output formats
 Oracle properties, [C-17](#)
 part numbers, [C-10](#)
 releasing, [C-12](#)

R

reference topics
 :main elements, [C-30](#)
 about, [C-30](#)
 examples, [C-35](#)
 guidelines, [C-32](#)
related links, [9-4](#)
relationship tables, [9-1](#)
reltable tag, [9-1](#)

S

SDL LiveContent Architect

SDL LiveContent Architect (*continued*)

 architecture, [C-1](#)
 Authoring Bridge, [C-1](#)
 Publication Manager, [C-1](#)
 web client, [C-1](#)
semantic tags, [4-1](#), [C-3](#)
short descriptions, about, [C-1](#)
special characters, [4-4](#)
spell checker, [4-4](#)
submaps
 adding to a bookmap, [C-11](#)
symbols, [4-4](#)

T

tables
 accessibility of *See* accessibility
 formal, [7-1](#), [7-4](#)
 informal, [7-1](#)
task topics, [3-8](#)
task topics:about, [C-18](#)
task topics:main elements, [C-18](#)
task topics:style conventions, [C-21](#)
tasks:examples, [C-23](#)
templates, [3-1](#)
title page information, [5-5](#)
topics
 concept, [3-6](#)
 orientation, [C-35](#)
 reference, [C-30](#)
topics:task topics, [C-18](#)
tutorial
 contents, [1-1](#)
typographical tags, [4-1](#)

U

uicontrol tag, [4-2](#)