

SPRINGER BRIEFS IN  
APPLIED SCIENCES AND TECHNOLOGY

Sissi Closs

# DITA – the Topic- Based XML Standard A Quick Start



Springer

**SpringerBriefs in Applied Sciences  
and Technology**

More information about this series at <http://www.springer.com/series/8884>

Sissi Closs

# DITA – the Topic-Based XML Standard

A Quick Start

Sissi Closs  
Hochschule Karlsruhe Technik und  
Wirtschaft  
Karlsruhe  
Germany

ISSN 2191-530X ISSN 2191-5318 (electronic)  
SpringerBriefs in Applied Sciences and Technology  
ISBN 978-3-319-28348-7 ISBN 978-3-319-28349-4 (eBook)  
DOI 10.1007/978-3-319-28349-4

Library of Congress Control Number: 2016934007

© The Author(s) 2016

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

This Springer imprint is published by Springer Nature  
The registered company is Springer International Publishing AG Switzerland

# Contents

<b>1</b>	<b>What Is DITA?</b>	1
1.1	DITA Meets Zeitgeist	1
1.2	What Makes DITA so Special?	3
1.3	What Comes Free of Charge with DITA?	4
1.4	DITA History	5
1.5	What Does <i>DITA</i> Mean?	5
1.6	Where Can DITA Be Used?	5
<b>2</b>	<b>Basic Principle for DITA: Topic-Oriented Structuring</b>	7
2.1	What Is a Topic?	8
2.2	Why Topic Types?	8
<b>3</b>	<b>Sample Class Concept</b>	11
3.1	Raw Material for <i>Making Coffee</i>	11
3.2	Developing a Class Concept	11
<b>4</b>	<b>Implementation in DITA</b>	15
4.1	Elements for Block Structures and Inline Elements	15
4.2	DITA Task	16
4.3	DITA Concept	17
4.4	DITA Reference	18
4.5	DITA Glossentry	19
<b>5</b>	<b>Assembling Topics</b>	21
5.1	Nesting Topics	21
5.2	DITA Map	22
<b>6</b>	<b>Defining Relations</b>	23
6.1	Links in the Map	23
6.2	Links in the Topic	25
<b>7</b>	<b>Reusing Contents by Embedding</b>	27

<b>8</b>	<b>Addressing</b> . . . . .	31
8.1	Direct Addressing. . . . .	31
8.2	More Freedom Through Indirect Addressing . . . . .	31
<b>9</b>	<b>Variants</b> . . . . .	35
9.1	Different Maps. . . . .	35
9.2	Variables. . . . .	35
9.3	Filtering . . . . .	37
<b>10</b>	<b>Collaboration Under Full Control</b> . . . . .	39
<b>11</b>	<b>How Is an Information Product Produced?</b> . . . . .	41
<b>12</b>	<b>Production with DITA Open Toolkit</b> . . . . .	43
12.1	What's in DITA Open Toolkit? . . . . .	43
12.2	Installing DITA Open Toolkit . . . . .	43
12.3	Generating Output with DITA Open Toolkit . . . . .	46
12.4	The First Publication . . . . .	46
12.5	Parameters for Publication. . . . .	46
<b>13</b>	<b>DITA Specialization</b> . . . . .	49
<b>14</b>	<b>Why Use DITA?</b> . . . . .	51
	<b>Appendix: DITA 1.3 Overview</b> . . . . .	53
	<b>References</b> . . . . .	59
	<b>Index</b> . . . . .	61

# Introduction

The XML standard DITA (Darwin Information Typing Architecture) has established itself in technical communications in recent years. The systematic use of topic-based structures is a necessary precondition for efficient single sourcing and is also best suited to the modern style of content presentation and usage, particularly on mobile devices.

Therefore, DITA has recently been finding its way also into areas such as marketing, training, and corporate communications [4].



# Chapter 1

## What Is DITA?

DITA is often compared to the Lego system. Just as you can use Lego bricks to build the most varied replicas of the real world such as houses, cars, and landscapes, you can use DITA *topics* to create any information products necessary.

With both Lego and DITA, this modular system gives you flexibility and enables you to create all sorts of different objects from the same repertoire of pieces.

### Example

Figure 1.1 shows an excerpt from a topic repertoire on the subject of *Coffee*.

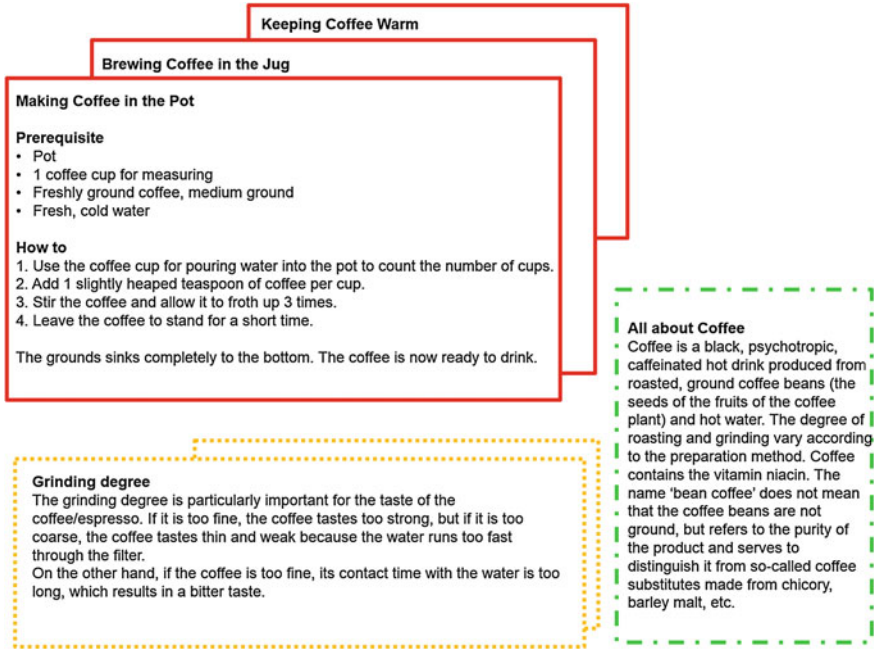
From this you can build a quick start guide for *Making Coffee* (Fig. 1.2), which describes only one of the many ways to make coffee and keep it warm. Prerequisites, notes, and all other topics are deliberately omitted from the quick start guide.

You can also compile a complete coffee book from the coffee-topic repertoire (Fig. 1.3), containing the same topics as in the quick start guide but with their complete content and also other topics.

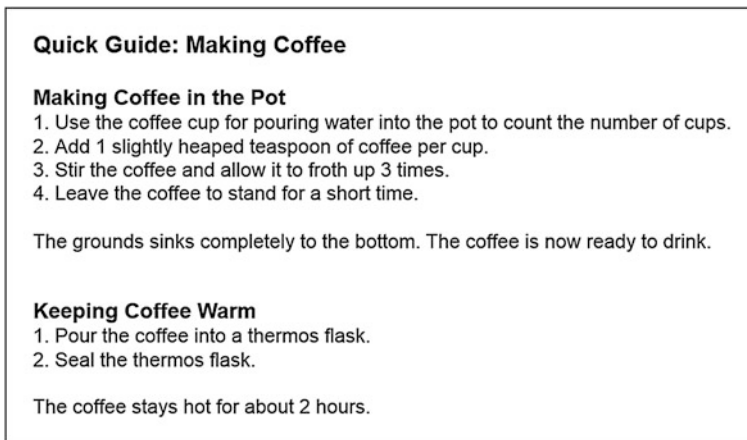
This example shows that in the quick start guide and the coffee book, not only the selection and combination of the topics can differ but also the content shown from a single topic, as well as the way the topics are presented.

## 1.1 DITA Meets Zeitgeist

Classical book-oriented production has reached its limits because requirements placed on information products have changed enormously. Users no longer want comprehensive books: instead, they want information packages that are individually tailored precisely to their profile, their situation, and their current needs. In addition, it must be possible to present the content optimally on the respective display medium.



**Fig. 1.1** Excerpt from a topic repertoire on the subject of *Coffee*



**Fig. 1.2** Quick start guide, *Making Coffee*

At the same time, the immense diversity of the products is growing continuously for the information provider, and their period of validity is getting shorter and shorter. This means that with traditional documents and redundant content

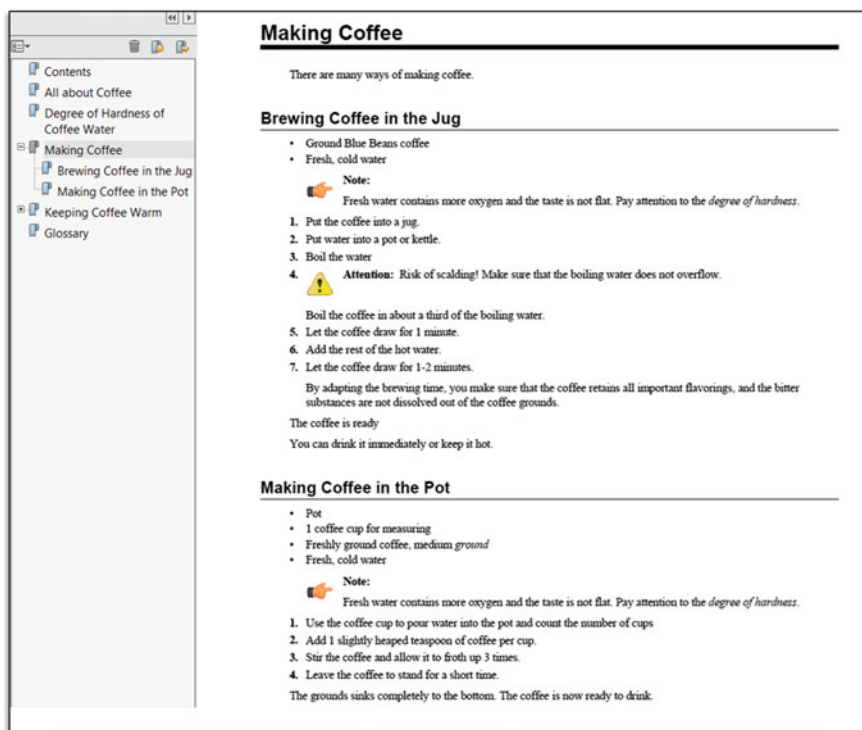


Fig. 1.3 Coffee book

maintenance, production is becoming ever more expensive and error-prone, particularly if the user expectations of today are to be met.

Topic-oriented structuring offers methods and techniques for meeting these challenges.

DITA builds on modularization, minimalism, reuse, and single sourcing, and provides an XML framework in which tried and tested methods and techniques can be implemented. These methods and techniques include information mapping<sup>®</sup> from Horn [10], class concept method<sup>®</sup> from Closs [2], function design<sup>®</sup> from Muthig and Schäfflein-Armbruster [12], and reuse methods from Rockley [15]. This creates confidence and offers a chance for good, sustainable solutions.

## 1.2 What Makes DITA so Special?

One reason why DITA is so widely accepted is its focus on topic-oriented structuring. Another is the fact that it is an open standard and is therefore not subject to license fees. This encourages its widespread use, which in turn means that

application know-how increases and more and more tools are developed to facilitate the use of DITA.

Moreover, DITA includes the concept of specialization, which means that the language can be adapted and extended according to prescribed rules without excessively endangering compatibility.

The DITA community is very large. More and more companies are using DITA for structuring and as a source format of their contents, and an increasing number of manufacturers of documentation tools are supporting DITA.

### **1.3 What Comes Free of Charge with DITA?**

The following are available free of charge for the DITA standard.

#### **Language definition**

The definition of the DITA language exists in the form of XML DTDs and XML schemas [14].

#### **Documentation of DITA standard**

For the DITA language, there is the official DITA specification in HTML, PDF, and CHM (Microsoft HTML Help) format. Essentially, the specification consists of two parts: a description of DITA concepts, and a reference of all elements and attributes of the language.

The document sources are structured with DITA and thus serve as examples that frequently occur in a similar form in practice.

#### **DITA Open Toolkit**

With the standard, you already have a suitable, cost-free DITA Open Toolkit (DITA OT) [6]. This contains a collection of scripts and programs that use DITA sources to generate different output formats. Using the DITA OT and further plugins where necessary, you can for example generate HTML, XHTML, PDF, EPUB, HTML5 with JQuery mobile, HTML Help, JavaHelp, EclipseHelp, DocBook, Troff, and RTF. DITA OT is being continuously developed and expanded. It is the reference implementation for the standard, which means that the functionality is implemented in the way intended by the standard.

DITA Open Toolkit also offers sample data for testing the scripts. To get an initial impression of the DITA world, the best and fastest way is to start with DITA Open Toolkit.

For DITA Open Toolkit, an installation manual and other documentation are provided.

#### **Sample applications**

In harmony with the open source spirit, many DITA users provide their DITA applications free of charge.

## Articles

The DITA concepts are described in numerous articles to be found on the OASIS site [7].

## 1.4 DITA History

In the late 1990s, IBM first defined DITA for its own documentation requirements.

To promote its spread and further development, IBM handed DITA over to OASIS (Organization for the Advancement of Structured Information Standards) in 2004 as an open-source architecture. In May 2005, DITA Version 1.0 was released as standard by OASIS; DITA Version 1.1 followed in August 2007, and Version 1.2 in November 2010. Version 1.3 is expected to appear at the end of 2015.

Where DITA details are mentioned in this book, they refer to DITA Version 1.3.

## 1.5 What Does *DITA* Mean?

Darwin Information Typing Architecture stands for core concepts used in DITA [5].

*Darwin* refers to Charles Darwin, the originator of the Theory of Evolution, which represents heredity and adaptability.

*Information Typing* refers to the typing of topics and maps.

*Architecture* means that there is a framework for topic and map types in the standard. This framework defines what they are intended for, and how they are to be used. DITA also defines rules and procedures for adapting and extending the architecture.

## 1.6 Where Can DITA Be Used?

DITA originally came from a technical documentation area, where it was used particularly in software documentation. This initially led to its reputation of being a software documentation standard only. But that is not true at all. The topic approach, combined with the flexibility of using suitable topic types for each kind of content, makes DITA a generic standard that can be used in any area for content structuring and as a source format. Successful DITA applications show that content reuse is possible on a broad scale and that for the first time synergies between classically separate fields such as marketing, training, and product information development can be achieved [4].

## Chapter 2

# Basic Principle for DITA: Topic-Oriented Structuring

DITA builds on topic-oriented structuring. The basic idea of this structuring principle is the division of content into pieces known as *topics* with the aim of assembling and reusing them flexibly. This structuring principle has a long history and was used in classical book production for lexicons and glossaries [2].

### Example

A term definition is a good topic example. A term is defined only once. Its definition can then be used anywhere where the term occurs and an explanation is needed. The topic advantages can be seen immediately:

- If the term definition has to be translated, it has to be translated only once. There is no redundant content that would cause multiple translations.
- Any changes are made in a single place only, and they are then consistently available everywhere.
- A term can be defined independently of the content in which it occurs.
- Creation and maintenance can be carried out by special experts.

But topic-oriented structuring really first came to life when content could be created and displayed digitally, and when important functions, in particular linking, could be technically implemented in an effective manner.

Topic-oriented structuring experienced a first heyday with the arrival of online help for software. Tools known as help authoring tools were developed both to support the creation of topics and to integrate technical functionality without any programming effort. However, the sources created with these tools are to a high degree tool-specific. In contrast, DITA offers a largely tool- and manufacturer-independent XML basis for the sources.

## 2.1 What Is a Topic?

Not every snippet of content is a good topic. As the example of the term definition shows, a topic should be a self-contained piece of content as context-independent as possible, containing a key statement and making sense on its own. The division of content into topics does not have to be just content- and usage-related: it can also have technical or organizational reasons. There are no rules governing the size of a topic. However, a topic should not be too large, for two reasons: first, there is a risk that the topic contains more than one key statement; second, it is difficult to display it on small devices. On the other hand, a topic should not be too small: it should be large enough to contain meaningful content that can be properly managed.

In the case of traditional book-oriented writing, content is created in context. Hierarchy and order of subjects are predefined. Explanation and instruction are often mixed up together, and the same content occurs redundantly but not identically at different places. In contrast, topics should be created separately from a concrete publication and context-independent so that they can be used in multiple ways. Each subject should be described only once (single point of truth). Just when a term is defined we do not necessarily know where that term is going to be used. The rule that applies when creating a topic is that its context is first revealed when topics are assembled for a special purpose and for a particular target group.

When presenting a topic on screen to the user, in the simplest case a topic is also displayed as a separate page. But this is not necessary. Content that is divided into several topics in the source can be presented contiguously if this is considered appropriate for the user.

For newcomers to topic-oriented structuring, dividing contents into topics is unfamiliar. The topic rules of the class concept method<sup>®</sup> help to find the right modularization [2].

### DITA topic

DITA topics should conform to the topic rules and they must have a title and a particular topic type. In file-based management, each DITA topic is usually stored as a separate file.

## 2.2 Why Topic Types?

Topics offer more flexibility, but they also require a lot of organizational effort because the number of topics can grow very fast and numerous topics have to be planned, managed, and organized properly in order to be found again. Classification or typing is a methodical approach to keep the amount of topics under control. With suitable classification criteria, you divide topics into different types characterized by features such as heading style, content type, etc. Instead of having to plan each topic individually, you just have to design a few topic types. Usually, you only need up to

ten topic types for the architecture of a specific documentation environment. For each topic type, as many topics as necessary can be created, in a controlled and consistent manner.

The design of topic types is a central task for information architects [3]. The class concept method<sup>®</sup> supports the iterative, agile development of topic types and their characteristic features [2].

### Example

A typical classification criterion is the content type. According to a proven modularization rule, explanations and background information should be separated from instructions (“separate what from how”). Therefore, DITA has contained the topic types `concept` and `task` right from the beginning [1]. The optimal structure of a task has been extensively researched: first the prerequisite, then the individual action steps, then the result. DITA has defined the suitable XML elements and structures for this.

### Advantages of classification

The topic types create a framework that ensures efficiency and quality and can guarantee the long-term stability of a topic pool.

### DITA topic types

Finding suitable topic types is not a simple task. This is where we see another advantage of DITA. As the name says, the standard supports typing. For the commonest content types such as step-by-step instructions, descriptions, and glossary entries, suitable topic types have become established throughout the years. DITA takes these up and, starting from the generic topic type, offers predefined specializations for established topic base types. Table 2.1 shows the base types in DITA 1.3.

Additionally, DITA offers topic types for specific applications. These include a series of topic types for the learning environment such as `LearningAssessment`, `LearningOverview`, `LearningPlan`, and `LearningSummary`.

The topic types have common elements such as the `title` element for the title, whereas specific elements characterize the type and structure of the content for which they are intended.

**Table 2.1** DITA topic types

DITA topic type	For
<code>concept</code>	Background information, concept, interdependencies, overview
<code>glossentry</code>	Glossary entry
<code>machinery task</code>	Instruction in engineering
<code>reference</code>	Facts, description of functions, commands, parameters
<code>task</code>	Instruction, procedure
<code>topic</code>	Content that does not suit any other topic type and basic type for specializations
<code>troubleshooting</code> (DITA 1.3)	Error message and removal



## Chapter 3

# Sample Class Concept

The development of an initial class concept is shown in the following using the coffee example.

### 3.1 Raw Material for *Making Coffee*

There often exists raw material (Fig. 3.1) that contains correct information but is neither meaningfully structured nor well formulated.

When you first look at the text in Fig. 3.1 for this well-known subject, the information first seems enough for making coffee. But if you look a bit closer, you see that important details are missing (such as how much coffee you need), and that the contents are not clearly structured (e.g., keeping the coffee hot is mentioned during the brewing phase). Moreover, there is no consistent style of writing. With unknown and difficult subjects, weak points like these result in the contents not being understood or not providing enough information to solve the task satisfactorily.

### 3.2 Developing a Class Concept

A topic-based solution can remove these weak points, and the class concept method helps to find systematically suitable topic types and define their features [2].

#### **Classifying contents**

The existing contents are analyzed and the different content types are marked. Figure 3.2 shows a meaningful modularization.

<p><b>Brewing in the jug</b></p> <p>Put ground coffee in a jug. Cover it with about a third of the boiling water. Caution! Acute risk of scalding! After about a minute, add the rest of the hot water. Leave the coffee to stand for one or two minutes. Pour the coffee into a preheated thermos flask. The adjusted brewing time ensures that the coffee retains all its flavors and the bitter substances are not released from the coffee-grounds.</p> <p><b>The water</b></p> <p>Use only fresh water and heat it from cold. The water then contains more oxygen and the taste is not flat, as it would be with low-oxygen water.</p> <p>You should use water with a hardness grade of five to six. If the water is softer, adding a pinch of salt can help. If the water is too hard, it spoils mainly the appearance of the drink. If necessary, you can use special filters to soften the water.</p> <p><b>Brewing coffee in the pot</b></p> <p>Simply measure how many coffee cups of water will go into your 5-liter pot. Use only fresh water! Add a slightly heaped teaspoon of coffee per cup. In the Arabian variant, add a few pinches of spice (cardamom or cinnamon) and 1 tsp of sugar.</p> <p>Bring it to the boil and stir it to make it foam 3x (the pot should therefore be only 2/3 full). Then leave it to stand for a short time (1 min). The grounds fall completely to the bottom, and you can then offer the finest and best-flavored coffee hot and to all guests simultaneously!</p>
---

Fig. 3.1 Raw material for *Making Coffee*

<p><b>Brewing in the jug</b></p> <p>Put ground coffee in a jug. Cover it with about a third of the boiling water. <b>Caution! Acute risk of scalding!</b> After about a minute, add the rest of the hot water. Leave the coffee to stand for one or two minutes. Pour the coffee into a preheated thermos flask. The adjusted brewing time ensures that the coffee retains all its flavors and the bitter substances <u>are not released</u> from the coffee grounds.</p>
<p><b>The water</b></p> <p>Use only fresh water and heat it from cold. The water then contains more oxygen and the taste is not flat, as it would be with low-oxygen water.</p> <p>You should use water with a hardness grade of five to six. If the water is softer, adding a pinch of salt can help. If the water is too hard, it spoils mainly the appearance of the drink. If necessary, you can use special filters to soften the water.</p>
<p><b>Brewing coffee in the pot</b></p> <p>Simply measure how many coffee cups of water will go into your 5-liter pot. <u>Use only fresh water!</u> Add a slightly heaped teaspoon of coffee per cup. <u>In the Arabian variant, add a few pinches of spice (cardamom or cinnamon) and 1 tsp of sugar.</u></p> <p>Bring it to the boil and stir it to make it foam 3x (the pot should therefore be only 2/3 full). Then leave it to stand for a short time (1 min). The grounds fall completely to the bottom, and you can then offer the finest and best-flavored coffee hot and to all guests simultaneously!</p>

Fig. 3.2 Modularizing

**Table 3.1** Class concept

Topic type	Label	Title	Writing style	Mapping → DITA
Instruction	t	Verbal: present participle plus object	Action step: imperative	task
Background information	c	<b>All about</b> <i>subject</i>		concept
Glossary entry	g	<i>Term</i> that is defined		glossentry
Facts	r	<i>Subject</i> substantive		reference
Note	n	None		note

The sample text contains instructions (frame), although these are not in the optimal shape as step-by-step instructions. There are notes. These are also marked (dotted frames) since the same note is intended to appear in several places. And there are some facts (double frame). Additionally, it makes sense to provide for background information and term definitions.

### Defining an initial class concept

With the class concept method<sup>®</sup>, we produce an initial class concept (Table 3.1).

You can successively refine the existing topic types by defining further features and add new topic types if necessary.

### Designing topic models

The important, DITA-independent work consists in designing a suitable topic model for every topic type defined in the class concept. This topic model should serve authors as a template. Existing good examples can be used as a starting point and adapted and extended with methods and techniques of minimalism, class concept method<sup>®</sup>, function design<sup>®</sup>, information mapping<sup>®</sup> as well as rules for comprehensible writing.

Figure 3.3 shows the task *Keeping Coffee warm in a Thermos Flask* according to a suitable task model.

Often it is not enough just to reformulate or make small structural adaptations to the raw material: you have to completely rewrite the contents.

In our example, this applies to the contents concerning the hardness of the water. It turns out that the raw material is incomplete. This is where the editorial work starts—i.e., you have to do some research to find missing information. Figure 3.4 shows the still incomplete topic.

Even if a class concept is still incomplete, you can implement it with DITA. If you later change the class concept, you can systematically revise the existing topic pool according to the changed class concept.

## Keeping Coffee Warm in a Thermos Flask

**PREREQUISITE**

- Thermos flask
- Warm hotplate

**NOTE:** Candles heaters can't be used.

---

**TASK**

1. Pour the coffee into a thermos flask.
2. Seal the thermos flask.

---

**RESULT:**

The coffee stays hot for about 2 hours.

**Fig. 3.3** Topic: *Brewing Coffee in the Jug*

## Hardness of Coffee Water

The *degree of hardness* of the coffee water is a decisive factor in the quality of the coffee. The following table shows the effects of the water hardness:

Degree of hardness	Effect	What to do
5-6	Coffee tastes good	Nothing
<5 (softer)	???	Add a pinch of salt to the water
>6 (harder)	Coffee becomes cloudy	Soften the water???with filters

**Fig. 3.4** Topic: *Hardness of Coffee Water*

## Chapter 4

# Implementation in DITA

For marking structures without making commitments to a particular layout or tool, XML languages have been used successfully for many years. DITA too is an XML language and it is specially equipped for marking topic structures in a suitable and display-neutral manner.

The implementation in DITA is shown in the following using the coffee example. Only the central DITA elements that occur in the topics are presented. The current, complete language description can be seen on the DITA page of OASIS [14].

The examples show that for a particular application, a small subset of the standard (which now consists of several hundred elements) is sufficient to mark the needed structures. That is reassuring but also means that defining the required DITA subset is a central initial task prior to implementation.

### 4.1 Elements for Block Structures and Inline Elements

For marking fundamental block and inline structures, DITA uses elements that are intentionally derived from the corresponding HTML elements.

#### DITA block elements

Table 4.1 shows the DITA elements for typical block elements in alphabetical order.

**Table 4.1** Selection of typical DITA block elements

Element	Marks
fig	Figure with optional caption
li	List item
note	Note
ol	Ordered list
p	Paragraph
section	Section
simpletable	Simple table
table	Table
title	Title
ul	Unordered list

**Table 4.2** Selection of typical DITA inline elements

Element	Marks
image	Image (graphic)
keyword	Keyword (significant word)
term	Term
xref	Cross-reference

**DITA inline elements**

Table 4.2 shows the DITA elements for typical inline elements in alphabetical order.

**4.2 DITA Task**

Table 4.3 shows specific task elements in alphabetical order.

**Table 4.3** Selection of specific task elements

Element	Marks
cmd	Command (step description)
context	Purpose of task
info	Additional information about the step description
prereq	Prerequisites to be met before carrying out the task described in the topic
result	Result of task
step	Single step
stepresult	Result of a single step
steps	Container for the single steps of the task
task	Root of a task topic
taskbody	Body of a task topic

```
- <task xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/"
id="id146QIOW005Z">
  <title>Brewing Coffee in the Jug</title>
  - <taskbody>
    + <prereq product="book">
    - <steps>
      - <step>
        <cmd>Put the coffee into a jug.</cmd>
      </step>
      + <step>
      + <step>
      + <step>
      + <step>
      + <step>
      + <step>
    </steps>
    - <result>
      <p>The coffee is ready</p>
      <p>You can drink it immediately or keep it hot.</p>
    </result>
  </taskbody>
</task>
```

Fig. 4.1 DITA task topic

Example of a task

You can tag the topic *Brewing Coffee in the Jug* with DITA as shown in Fig. 4.1.

4.3 DITA Concept

Table 4.4 shows specific concept elements in alphabetical order.

Example of a concept

You can tag the topic *All about Coffee* with DITA as shown in Fig. 4.2.

Table 4.4 Selection of specific concept elements

Element	Marks
conbody	Body of a concept topic
concept	Root of a concept topic

```
- <concept xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/" id="id158B0030401">
  <title>All about Coffee</title>
  - <conbody>
    <p>Coffee is a black, psychotropic, caffeinated hot drink produced from roasted,
    ground coffee beans (the seeds of the fruits of the coffee plant) and hot water. The
    degree of roasting and grinding vary according to the preparation method. Coffee
    contains the vitamin niacin. The name 'bean coffee' does not mean that the coffee
    beans are not ground, but refers to the purity of the product and serves to
    distinguish it from so-called coffee substitutes made from chicory, barley malt,
    etc.</p>
  </conbody>
</concept>
```

Fig. 4.2 DITA concept topic

## 4.4 DITA Reference

Table 4.5 shows specific reference elements in alphabetical order.

### Example of a reference

You can tag the topic *Hardness of Coffee Water* with DITA as shown in Fig. 4.3.

**Table 4.5** Selection of specific reference elements

Element	Marks
propdeschd	Header of the output column for the column with the brief description
properties	List of properties or parameters
property	A property or a parameter
refbody	Body of a reference topic
reference	Root of a reference topic

```

- <reference id="id158DG0TOHNA">
  <title>Hardness of Coffee Water</title>
  - <refbody>
    - <section>
      - <p>
        The
        <term keyref="HG">degree of hardness</term>
        of the coffee water is a decisive factor in the quality of the coffee. The
        following table shows the effects of the water hardness:
      </p>
      - <simplatable frame="all" relcolwidth="33* 33* 33*">
        - <sthead>
          - <stentry>
            <p>Degree of hardness</p>
          </stentry>
          - <stentry>
            <p>Effect</p>
          </stentry>
          - <stentry>
            <p>What to do</p>
          </stentry>
        </sthead>
        - <strow>
          <stentry>5-6</stentry>
          - <stentry>
            <p>Coffee tastes good</p>
          </stentry>
          <stentry>Nothing</stentry>
        </strow>
        + <strow>
        + <strow>
        </simplatable>
      </section>
    </refbody>
  </reference>

```

**Fig. 4.3** DITA reference topic



## 4.5 DITA Glossentry

Table 4.6 shows specific glossentry elements in alphabetical order.

### Example of a glossentry

You can tag the term definition for the degree of water hardness in DITA as shown in Fig. 4.4.

**Table 4.6** Selection of specific DITA glossentry elements

Element	Marks
glossbody	Details for the glossentry
glossdef	Definition of term (glossentry)
glossentry	Root of a glossentry topic
glossterm	Term that is defined

```
- <glossentry id="id146R9800RZW">
  <glossterm>Degree of hardness</glossterm>
  - <glossdef>
    Specifies the hardness of the water.
    - <note>
      <p>Ask your local council for the hardness of your water.</p>
    </note>
  </glossdef>
</glossentry>
```

**Fig. 4.4** DITA glossentry topic

## Chapter 5

# Assembling Topics

DITA provides several ways of assembling individual topics. You can nest topics or combine them in maps.

### 5.1 Nesting Topics

You can nest topics to form larger, continuous content blocks. The nesting defines the sequence and hierarchy of the participating topics. However, nesting can only take place outside the topic body.

The DITA configuration can define how and which topic types can be nested (if at all). The standard DITA configuration allows nesting with all topic types only for the generic root element `dita`.

Generally it is recommended that you manage each topic in a separate file. In some cases however, nesting can make sense in reducing management effort. With a file-based organization, the authors' work may be more efficient if for example all topics of one type are physically in one file, because in the event of a general revision, only one file has to be opened and functions like search and replace can be conducted more quickly. Also, the migration of contents can be easier if the topics are stored in a single file during the migration process.

## 5.2 DITA Map

The most flexible way of combining contents is provided by the DITA map. This resembles the well-known table of contents in online help programs and in the same way represents the framework for a complete information product. In a DITA map, you can combine topics according to different organization patterns and specify links for the topics listed in the map.

However, the DITA map has far more functions than the classical table of contents. The map defines the topics to be included in an information product, specifies their sequence, grouping, and hierarchy, and defines the relationships between the topics. Additionally, you use the map to define variables as well as metadata for characterizing and managing the information product. You can use a map to plan projects and information products and produce and adapt information products.

Figure 5.1 shows an extract from the DITA map for a coffee book.

### Submaps

Maps can contain submaps. You use submaps for giving a meaningful structure to maps for comprehensive and more complex information products, which makes them easier to handle.

### Example

It can be very useful to organize all glossary entries for a particular subject in a separate map, and to reference this as a submap in the maps for the information products.

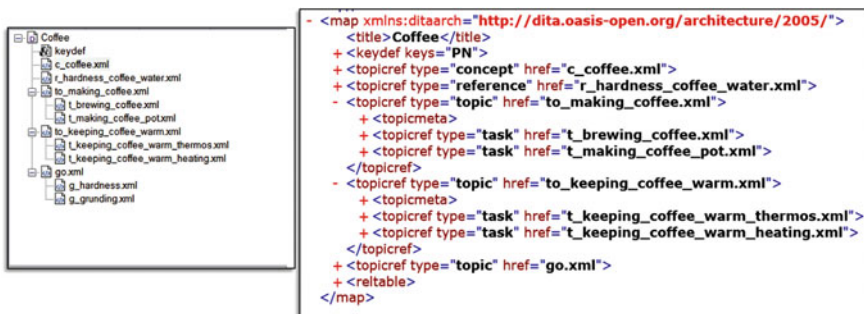


Fig. 5.1 DITA map

## Chapter 6

# Defining Relations

You place topics in relation to each other so as to show interrelated and therefore interesting content for the user. The best-known relations in documentation are cross references, which relate content items to each other either within a document or between different documents. A cross reference leads the reader from the current place in the text to other places that the author considers to be related to the current place, to explain the current place better, or to provide further information on the current place. In the electronic world, references are known as links. They are visible as such to the user and occur in different forms.

In general, DITA supports the principle that topic content should be kept as link-free as possible. This principle has advantages for both the reader and the author:

- Readers are not distracted and can absorb link-free topics better; they can navigate more effectively via systematically arranged links.
- For the author, the separation of contents and links simplifies all processes from creation to management and from maintenance to translation.

One of DITA's great strengths is the wide variety of ways of defining relations and links. Apart from the links that can automatically be generated from the map, you can set links explicitly in the topic or in the map.

### 6.1 Links in the Map

The most flexible method is to define the linking in the map and therefore to first define relations between topics when the topics are assembled in their respective combination.

These options are very interesting when it comes to the reuse factor. Since a topic can have different relations according to its context, a link in the topic would restrict its reusability. But if the link is defined in the map, you can combine the topic flexibly with other topics in different maps.

## Generated links

For relations that result from the arrangement of topics in the map, the standard provides that links can be automatically generated using suitable metadata (attributes)—for example, from a topic to its subordinate (child) topics.

### Example

If there are several topics on a single subject, it can help orientation according to the leveling principle to provide an overview.

For example, there are many ways of making coffee. For a fast overview, you provide an overview topic. If you get DITA to generate the links, you only need an introductory sentence in the overview topic. Figure 6.1 shows the DITA source of the overview topic.

The links to the various task topics result from the map and are generated if you set the `linking` attribute accordingly. Figure 6.2 shows a generated MS HTML help output in CHM format.

The advantages are obvious: the overview topic always automatically contains the correct and complete link list.

### Linking via a relationship table (`reltable`)

A map also offers the possibility to specify relations explicitly by means of a relationship table (`reltable`).

```
- <topic id="id158AN900UT6">
  <title>Making Coffee</title>
  - <body>
    <p>There are many ways of making coffee.</p>
  </body>
</topic>
```

**Fig. 6.1** DITA source for overview topic

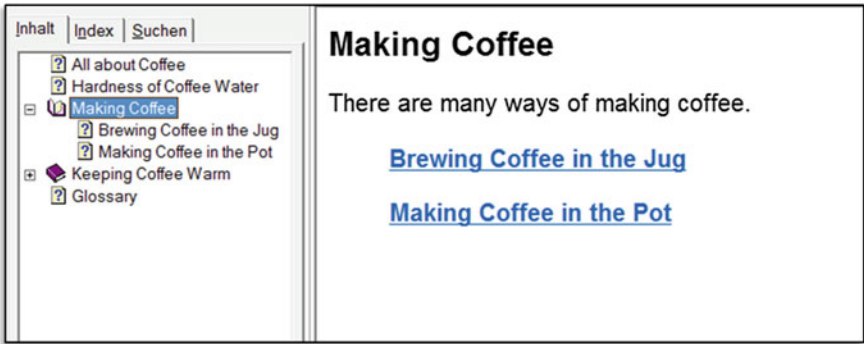


Fig. 6.2 Published overview topic with generated links

Task	Topic
<ul style="list-style-type: none"><li>Brewing Coffee in the Jug (t_brewing_coffee.xml)</li></ul> Brewing Coffee in the Jug	<ul style="list-style-type: none"><li>Keeping Coffee Warm (to_keeping_coffee_warm.xml)</li></ul> Keeping Coffee Warm

Fig. 6.3 Relationship table

Example

You can specify the link from the task *Brewing Coffee in the Jug* to the overview *Keeping Coffee Warm* in the coffee map by means of a relationship table.

In the WYSIWYG view (Fig. 6.3), the table looks like this.

Figure 6.4 shows the DITA source.

6.2 Links in the Topic

DITA also allows to set links in a topic:

- You can use `xref` to specify the classic cross-references that are well-known in paper documents and that can be used anywhere in topic content. They are often used to refer to a figure or table in the topic.
- You can also specify links in a separate `related-links` section following the topic body and separate from the actual topic contents. However, this may reduce the reusability of the topic since the referenced destinations may not suit every context.

```

- <map xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/">
  <title>Coffee</title>
  + <keydef keys="PN">
  + <topicref product="book" type="concept" href="c_coffee.xml">
  + <topicref type="reference" href="r_hardness_coffee_water.xml">
  + <topicref type="topic" href="to_making_coffee.xml">
  + <topicref type="topic" href="to_keeping_coffee_warm.xml">
  + <topicref type="topic" href="go.xml">
  - <reltable>
    <title/>
    - <relheader>
      <relcolspec type="task"/>
      <relcolspec type="topic"/>
    </relheader>
    - <relrow>
      - <relcell>
        + <topicref type="task" href="t_brewing_coffee.xml">
      </relcell>
      - <relcell>
        + <topicref type="topic" href="to_keeping_coffee_warm.xml">
      </relcell>
    </relrow>
    - <relrow>
      <relcell/>
      <relcell/>
    </relrow>
  </reltable>
</map>

```

Fig. 6.4 DITA relationship table (reltable)

## Chapter 7

# Reusing Contents by Embedding

A central aspect in a DITA-based documentation environment is the orientation to single sourcing and content reuse. You can reuse material from complete topics down to individual sentences.

### Example

You want to use a note at different places, not as a separate topic but embedded in the content of another topic.

DITA offers various ways of embedding.

The `conref` mechanism, well known from SGML, enables you to reuse the content of an element or an element group both within a topic and between different topics. Figure 7.1 illustrates the principle.

The `conref` mechanism in DITA exists as both a pull and a push variant. In the pull variant, you reference the target content to be embedded at the place where the content is to be inserted. In the push variant, you specify in the content to be embedded where and how the content is to be inserted.

With the `conref` mechanism, you can reuse any element or whole element groups with a unique identification (`id` attribute). However, you can only insert structurally equivalent content. If the element types are not compatible, no embedding takes place.

Reuse at the topic level is much easier to handle and should therefore always be the first choice. However, there are good reasons for embedding, such as with safety warnings. But clear rules are needed. You have to define the type of content to be embedded in this manner and where embedding can take place. The rules must always be obeyed: otherwise, there is a danger that the organization becomes too complex and unclear.





Fig. 7.1 Embedding contents

Example


In the coffee tasks, the note about fresh water is relevant in several coffee topics. Figure 7.2 shows two examples.

To avoid redundancy in the sources, we use the `conref` mechanism in the pull variant. To make it easier to manage, you create a collective topic for all notes, also called a warehouse topic. For the purpose of identification, every note receives an ID. The conventions for building the ID must be defined. Figure 7.3 shows the IDs assigned in the `id` attribute. The ID for notes starts with `N_`, and the ID for safety warnings with `SN_`.

The warehouse topic is the single point of truth for notes where a note is created and edited. This avoids redundancy, inconsistency, and errors, and keeps the size of

**Brewing Coffee in the Jug**

- Ground Blue Beans coffee
- Fresh, cold water

**Note:**  
Fresh water contains more oxygen and the taste is not flat. Pay attention to the *degree of hardness*.

**Making Coffee in the Pot**

- Pot
- 1 coffee cup for measuring
- Freshly ground coffee, medium *ground*
- Fresh, cold water


**Note:**  
Fresh water contains more oxygen and the taste is not flat. Pay attention to the *degree of hardness*.

Fig. 7.2 A repeatedly needed note

```

- <topic id="id146QK0608YK">
  <title>Note Warehouse Topic</title>
  - <body>
    - <note id="N_water" type="note">
      - <p>
        Fresh water contains more oxygen and the taste is not flat. Pay attention
        to the
        <term keyref="HG">degree of hardness</term>
      </p>
    </note>
    <note id="SN_scald" type="attention">Risk of scalding! Make sure that the boiling
    water does not overflow.</note>
  </body>
</topic>

```

Fig. 7.3 Warehouse topic for notes

```

- <task id="id146QI0W005Z">
  <title>Brewing Coffee in the Jug</title>
  - <taskbody>
    - <prereq product="book">
      - <ul>
        + <li>
          - <li>
            <p>Fresh, cold water</p>
            <note conref="n_notes.xml#id146QK0608YK/N_water"/>
          </li>
        </ul>
      </prereq>
      + <steps>
      + <result>
    </taskbody>
  </task>

```

Fig. 7.4 Embedding a note with conref

the sources down. It also enables the work to be distributed easily. For example, one person can write the notes while another writes the topics containing the notes.

Wherever a note is needed, it can be taken from the warehouse topic and embedded using `conref`. Figure 7.4 shows the embedding of the note about fresh water in the note element in the topic *Brewing Coffee in the Jug*.

## Chapter 8

# Addressing

For referencing, DITA since Version 1.2 has supported not only direct but also indirect addressing.

### 8.1 Direct Addressing

With direct addressing, you specify the destination with its actual address. But this results in considerable dependency. If the destination is renamed, moved, or deleted, the reference has to be edited to remain intact.

### 8.2 More Freedom Through Indirect Addressing

DITA 1.2 introduced indirect addressing to solve the problems of direct addressing. Instead of addressing the destination directly, you just specify a freely definable key for the destination. Then, when you create the map, you assign suitable destinations to the keys and thus define the target of the reference.

Ways of using indirect addressing have been improved and extended in DITA 1.3.

#### Example

The simplest way to demonstrate indirect addressing is with references to glossary entries.

Our coffee subject contains the phrase *degree of hardness*. For this phrase, you create in DITA a term definition as a `glossentry` topic (Fig. 8.1).

```

- <glossentry id="id146R9800RZW">
  <glossterm>Degree of hardness</glossterm>
  - <glossdef>
    Specifies the hardness of the water.
    - <note>
      <p>Ask your local council for the hardness of your water.</p>
    </note>
  </glossdef>
</glossentry>

```

Fig. 8.1 Glossentry topic for degree of hardness

```

- <topic id="id146QK0608YK">
  <title>Note Warehouse Topic</title>
  - <body>
    - <note id="N_water" type="note">
      - <p>
        Fresh water contains more oxygen and the taste is not flat. Pay attention
        to the
        <term keyref="HG">degree of hardness</term>.
      </p>
    </note>
    <note id="SN_scald" type="attention">Risk of scalding! Make sure that the
      boiling water does not overflow.</note>
  </body>
</topic>

```

Fig. 8.2 Indirect addressing using the key *HG* in the `term` element

The note about fresh water uses the term *degree of hardness*. As shown in Fig. 8.2, the key *HG* is assigned to the `keyref` attribute instead of the fixed address of the glossentry topic to make a reference to the definition of *degree of hardness*.

In a map for the coffee topics, the term definitions are referenced with `glossref`. The destination topic containing the term definition to be used in this map is assigned in the `href` attribute to the corresponding key in the `keys` attribute (Fig. 8.3).

Indirect addressing is a powerful feature. The topics in which references are used are independent of fixed addresses and do not need to be adapted if these addresses change. Moreover, from the same combination of topics, different variants can be produced if the keys in the map are assigned to different destinations.

```

- <map xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/">
  <title>Coffee</title>
  + <keydef keys="PN">
  + <topicref type="concept" href="c_coffee.xml">
  + <topicref type="reference" href="r_hardness_coffee_water.xml">
  + <topicref type="topic" href="to_making_coffee.xml">
  + <topicref type="topic" href="to_keeping_coffee_warm.xml">
  - <topicref type="topic" href="go.xml">
    - <topicmeta>
      <navtitle>Glossar</navtitle>
    </topicmeta>
    - <glossref keys="HG" type="glossentry" href="g_hardness.xml" print="yes">
      + <topicmeta>
    </glossref>
    + <glossref keys="MG" type="glossentry" href="g_grunding.xml" print="yes">
    </topicref>
    + <reltable>
  </map>

```

**Fig. 8.3** Destination assignment in the map in glossref

DITA 1.3 has extended the assignment options for keys in a map. You can now define validities for keys, and different branches of a map can have different assignments for the same key.

## Chapter 9

# Variants

DITA provides very good options for setting up an efficient variant management, including if the sources are to be managed only in the file system and without a content management system. These include the options of combining different maps from the same topic pool, and using variables and filtering. The basis is the map via which these methods can be used to produce the variants.

### 9.1 Different Maps

If required, you can produce different maps from a pool of topics for different purposes, target groups, and output media. If maps share only a few topics and only the topic content is changed but the combination in the maps is very stable, it can make sense to make several maps.

#### Example

From the coffee topics, you can produce a map for the quick start guide and a map for the complete coffee book.

### 9.2 Variables

You can use variables in DITA, typically for product names, version numbers, etc. In this case, you use keys as with indirect addressing. For a variable, you define a specific key. In the topics, you use only keys instead of fixed names. It is the map that combines the topics that defines the values of the variables. For this reason, you do not have to define the value until the information product is produced,

and you can generate different variants from the same topics by assigning different values to the variables in the maps.

### Example

If a coffee producer wants to use the coffee topics for coffee books that are branded with his or her coffee brands, a variable can be used in the topics for the coffee brand (Fig. 9.1).

In the map, a name such as *Blue Beans* is assigned to the variable *PN* (Fig. 9.2).

```
- <task id="id146QI0W005Z">
  <title>Brewing Coffee in the Jug</title>
  - <taskbody>
    - <prereq product="book">
      - <ul>
        - <li>
          - <p>
            Ground
            <keyword keyref="PN"/>
            coffee
          </p>
        </li>
      </ul>
    </prereq>
    + <steps>
    + <result>
  </taskbody>
</task>
```

Fig. 9.1 Variable *PN* for coffee brand

```
- <map xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/">
  <title>Coffee</title>
  - <keydef keys="PN">
    - <topicmeta>
      <navtitle/>
      - <keywords>
        <keyword>Blue Beans</keyword>
      </keywords>
    </topicmeta>
  </keydef>
  + <topicref type="concept" href="c_coffee.xml">
  + <topicref type="reference" href="r_hardness_coffee_water.xml">
  + <topicref type="topic" href="to_making_coffee.xml">
  + <topicref type="topic" href="to_keeping_coffee_warm.xml">
  + <topicref type="topic" href="go.xml">
  + <retable>
</map>
```

Fig. 9.2 Defining variable *PN* in the map

### 9.3 Filtering

Use filtering for producing different variants from the same map. Attributes and a filter file determine which content appears in the respectively produced information product.

If the maps are very large and the variants have a large number of common topics, and a lot of changes occur (i.e., many topics are added, deleted, or moved), it is better to generate the different variants using filtering instead of using different maps. This reduces maintenance efforts.

#### Attributes for filtering

DITA uses the attributes listed in Table 9.1 for filtering.

You define possible attribute values—e.g. for the audience attribute, the values for *beginner*, *normal*, *expert*. An attribute can have several values separated by spaces.

#### Filter file ditaval

For production, the filter file defines the contents to be included or excluded (with `include` or `exclude`), depending on the filter attributes and their values. The filter file is an XML file with the suffix `.ditaval`.

#### Example

The filter attribute `product` is used, and the possible values `book` and `quickGuide` are defined. The attributes are set accordingly in the topics (Fig. 9.3) and in the map (Fig. 9.4).

Table 9.1 Filter attributes

Attribute	For specifying
<code>audience</code>	Target group
<code>deliveryTarget</code> (DITA 1.3)	Output format in the map
<code>otherprops</code>	Individually defined filter criteria
<code>platform</code>	Area
<code>product</code>	Product

Fig. 9.3 `prereq` is to be used only in the coffee book (`product='book'`)

```
- <task id="id146QI0W005Z">
  <title>Brewing Coffee in the Jug</title>
  - <taskbody>
    + <prereq product="book">
    + <steps>
    + <result>
  </taskbody>
</task>
```



```
- <map xmlns:ditaarch="http://dita.oasis-open.org/architecture/2005/">
  <title>Coffee</title>
  + <keydef keys="PN">
  + <topicref product="book" type="concept" href="c_coffee.xml">
  + <topicref type="reference" href="r_hardness_coffee_water.xml">
  + <topicref type="topic" href="to_making_coffee.xml">
  + <topicref type="topic" href="to_keeping_coffee_warm.xml">
  + <topicref type="topic" href="go.xml">
  + <reltable>
</map>
```

**Fig. 9.4** The topic `c_coffee.xml` is to be used only in the coffee book (`product='`book`'`)

```
<val>
  <prop val="book" att="product" action="exclude"/>
</val>
```

**Fig. 9.5** Filter file `quickguide.ditaval` for producing the quick start guide

For the production of the quick start guide, you create a filter file for excluding all contents that are to appear in the coffee book only. Figure 9.5 shows the XML source.

DITA 1.3 has introduced branch filtering, which enables you to set different filter criteria for different branches of a map.

## Chapter 10

# Collaboration Under Full Control

Successful collaboration is an important success factor in any environment, but it needs suitable rules so that it does not get out of control. Since Version 1.2, DITA has provided good support for successful collaboration.

You can use constraints to restrict syntax without specialization. In this way, you can define conventions in a project environment more clearly for a team. Thus you can remove elements or set them as mandatory, and specify the element sequence.

Subject schemas provide a promising way of making clear rules. For a particular environment, they allow you to define specific attribute values and names for metadata without having to edit the DTDs.

Supplements for the terminology enable you to define the terminology right up to complete taxonomies. This enables you to lay the basis for semantic web functions: systematical, standardized, and integrated in the source content.

## Chapter 11

# How Is an Information Product Produced?

An information product is produced by combining topics in a map and defining an output format for them.

From a collection of topics, you can put together information products flexibly and according to your requirements on the basis of suitable maps: for different target groups, purposes, and output media. DITA is an XML language, so information products are generated from DITA sources according to the normal XML production process. Figure 11.1 shows the principal process and the tools required.

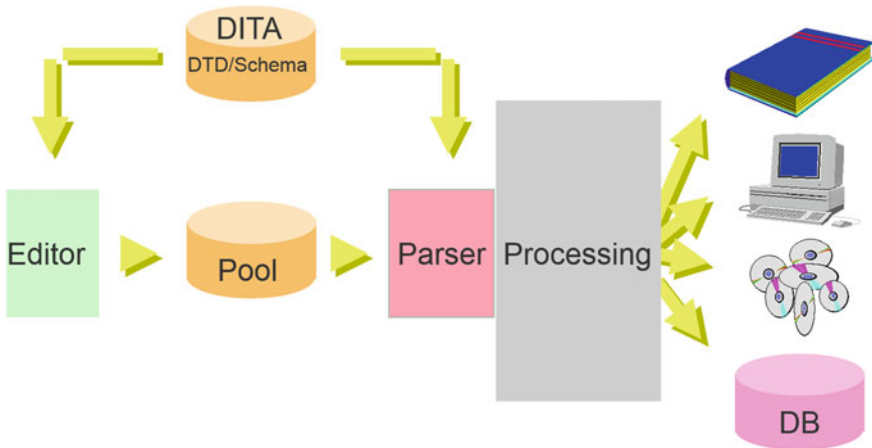
### Creation

Basically, you can use any editor for creating DITA sources. But if you use a simple text editor, it is more difficult. There are now many XML editors that are pre-configured for DITA and provide authors with good support: input assistance for selecting suitable elements, attributes and attribute values; and preconfigured layout views for writing, generating, and displaying in different formats via a connection to the DITA OT or your own output transformations.

### Management

In the simplest case, you can use the file system for managing DITA sources. DITA offers powerful functions for typical management tasks—in particular for reuse, variables, and variant control.

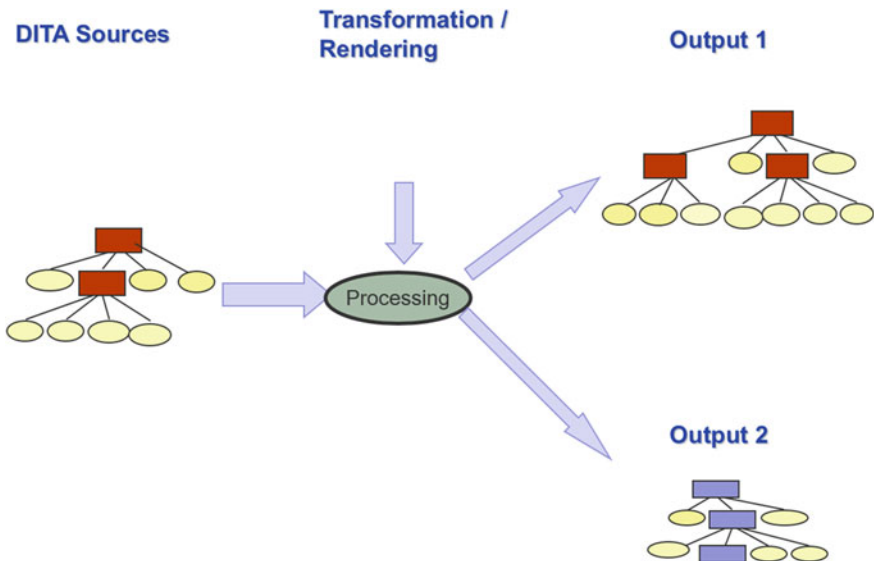
Even more comprehensive management support is provided by content management systems. There is a whole range of XML-based content management systems already preconfigured for DITA. As a rule, however, you have to adapt and complete the configuration according to your own needs.



**Fig. 11.1** XML production process

### Further processing

For generating information products, further processing means formatting for the output medium. In the simplest case, you just need a Cascading Stylesheet (CSS) for formatting DITA sources. For a more ambitious design, you use XSL stylesheets (Fig. 11.2).



**Fig. 11.2** Formatting with XSL stylesheets

# Chapter 12

## Production with DITA Open Toolkit

DITA Open Toolkit comes cost-free in different versions [6].

You can adapt the transformations that come with the toolkit to your own needs. But you need programming knowledge (CSS, XSL-FO, XSLT, ANT) according to the degree of your adaptation.

### 12.1 What's in DITA Open Toolkit?

Figure 12.1 shows the folders in the program directory of DITA Open Toolkit:

#### Folders in the DITA OT program directory

Table 12.1 lists the folder in the DITA OT program directory.

#### Folders in the plugins directory

The plugins directory contains the transformations as delivered for various output formats (Table 12.2). You adapt layouts in the respective subdirectories. If you have developed your own transformations or received any from third parties, install them in the plugins directory too.

### 12.2 Installing DITA Open Toolkit

The DITA Open Toolkit comes in several variants [6]:

- dita-ot-version.zip: contains the compiled DITA Open Toolkit for Windows
- dita-ot-version.tar.gz: contains the compiled DITA Open Toolkit for Linux and Mac
- Source Code: contains the Java source files of DITA Open Toolkit

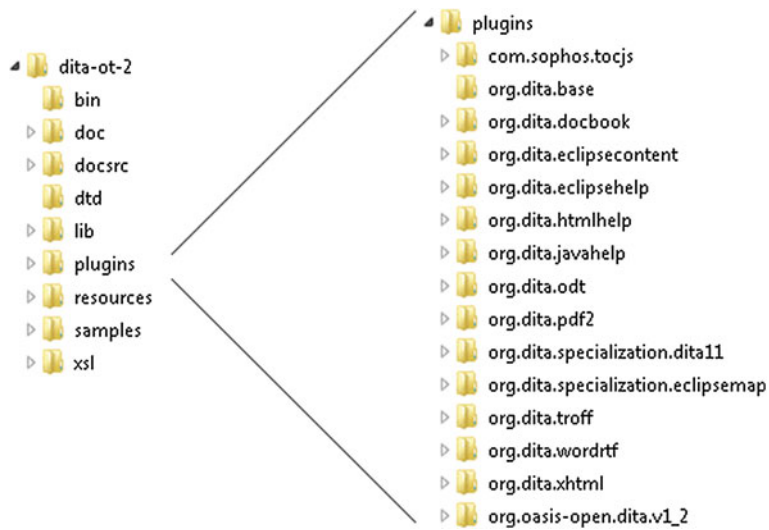


Fig. 12.1 Program directory of DITA Open Toolkit

Table 12.1 Program directory of DITA Open Toolkit

Folder	Contents
dita-ot-x	Base directory of DITA Open Toolkit, Version x
bin	Scripts for calling the DITA Open Toolkit
doc	Documentation of DITA Open Toolkit in HTML, PDF and CHM formats
docsrc	DITA sources of documentation of DITA Open Toolkit
dtd	DTDs
lib	Java program files
plugins	Folder for transformations, DTDs and own extensions
resources	Files for messages, etc.
samples	DITA sample documents and ANT sample scripts

Additional software

DITA Open Toolkit needs Java (JRE or JDK) in Version 7 or higher from Oracle (Java [11]). For a transformation to HTML help (CHM), you need Microsoft HTML Help Workshop [13].

**Table 12.2** Plugins directory of DITA Open Toolkit

Folder	Contents
<code>com.sophos.tocjs</code>	Transformation to XHTML, with Javascript frameset
<code>org.dita.base</code>	Base files for all transformations
<code>org.dita.docbook</code>	Transformation to Docbook
<code>org.dita.eclipsecontent</code>	Transformation to normalized DITA with Eclipse project files
<code>org.dita.eclipsehelp</code>	Transformation to Eclipse help
<code>org.dita.htmlhelp</code>	Transformation to HTML help
<code>org.dita.javahelp</code>	Transformation to Java help
<code>org.dita.odt</code>	Transformation to Open Document Format (Open Office)
<code>org.dita.pdf2</code>	Transformation to PDF
<code>org.dita.specialization.ditall</code>	DITA 1.1 DTDs and schemas
<code>org.dita.specialization.eclipsemap</code>	EclipseMap DTDs and schemas
<code>org.dita.troff</code>	Transformation to Troff
<code>org.dita.wordrtf</code>	Transformation to Rich Text Format
<code>org.dita.xhtml</code>	Transformation to XHTML and HTML5, basis for all HTML-based transformations
<code>org.oasis-open.dita.v1_2</code>	DITA 1.2 DTDs and Schemas

## Installing components

After downloading, unpack DITA Open Toolkit to a directory of your choice. For our examples, we use the following:

```
C:\dita-ot-2
```

Optional, but recommended for a simpler start of DITA Open Toolkit: add the pathname of the `bin` directory to the `PATH` system variable of your computer, here:

```
C:\dita-ot-2\bin
```

In addition to DITA Open Toolkit, Java must also be installed. DITA Open Toolkit is tested with Java Version 7.

If you want to create HTML help (CHM), install HTML Help Workshop.

## 12.3 Generating Output with DITA Open Toolkit

DITA Open Toolkit works with the command line. First, open the command line prompt (Windows) or a terminal window (Linux and Mac).

If you have added the pathname of the `bin` directory to the `PATH` system variable, you can start DITA Open Toolkit as follows:

```
dita
```

Otherwise, specify the path to DITA Open Toolkit, here:

```
C:\dita-ot-2\bin\dita
```

The command returns a brief overview of the call parameters.

## 12.4 The First Publication

To publish the DITA sample document (“garage sample”) to XHTML, enter the following (if you have extended the `PATH` system variable):

```
dita -f xhtml -i C:\dita-ot-2\samples\hierarchy.  
ditamap -o outdir
```

You will find the result in the `outdir` subdirectory of the current directory.

## 12.5 Parameters for Publication

To publish with the `dita` command, you have to specify at least the two parameters `-f` and `-i`. All other parameters are optional. Table [12.3](#) lists the parameters.



**Table 12.3** DITA Open Toolkit parameters

Parameter	Meaning
<code>-f &lt; output-format &gt;</code>	Desired publication format DITA Open Toolkit includes the following output formats: docbook, eclipsecontent, eclipsehelp, html5, htmlhelp, javahelp, odt, pdf, pdf2, tocjs, troff, wordrtf, xhtml
<code>-i &lt; input-file &gt;</code>	Absolute or relative path to the map of your DITA document
<code>-o &lt; output-directory &gt;</code>	Absolute or relative pathname of the directory to which the publication is to be written
<code>-filter &lt; input-file &gt;</code>	Absolute or relative path to the ditaval file
<code>-temp &lt; directory &gt;</code>	Absolute or relative pathname of the directory to which temporary files are to be written
<code>-v</code>	Generates detailed logging output
<code>-d</code>	Generates detailed debugging output
<code>-l</code>	Absolute or relative pathname to the file to which logging output is to be written
<code>-</code>	Parameters for transformation
<code>D &lt; parameter &gt;=&lt;value &gt;</code>	Specify <code>-D</code> for each parameter Possible parameters for transformations are described in the DITA Open Toolkit documentation
<code>-propertyfile &lt; file &gt;</code>	Parameters for transformation You can define a whole parameter set in a file, particularly if you always want to use the same parameters

## Chapter 13

# DITA Specialization

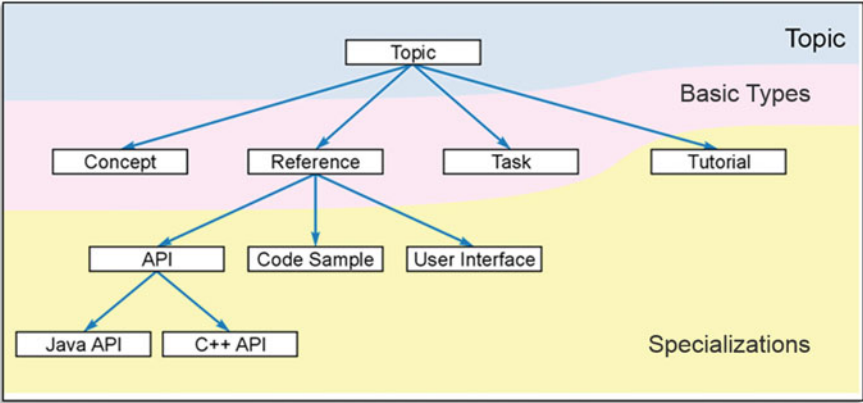
With DITA, you can define new domains and types on the basis of predefined basic types. Such adaptations and extensions are called specialization. Using the inheritance principle, definitions for the output types are passed on to derived new types and can be specifically adapted and extended according to requirements. Figure 13.1 shows the principle using the example of specializations of the `reference` topic type:

You can make adaptations and extensions in several ways:

- You can extend topic types.
- For contents not covered by existing elements, you can introduce new domains.
- For special information structures, you can create your own domains.

DITA prescribes the rules for specialization. A new topic type must be built on an existing one and must further restrict the content.

You should think carefully about a specialization. As a rule, the standard is sufficient for structuring contents meaningfully. And DITA is being continuously developed so that a complex specialization of your own could be a disadvantage if future standard versions also include these extensions. The `glossentry` and `troubleshooting` topic types are good examples of this.



**Fig. 13.1** Specializations of the reference topic type

## Chapter 14

# Why Use DITA?

DITA builds on topic-oriented structuring and therefore has all the benefits of this structuring technique if used correctly:

- Many different reuse options
- Efficient variant management
- Good ways of providing suitable access
- Support for collaboration

In combination with a suitable class concept, all documentation processes can be made more streamlined:

- For authors, it is easier to create and edit individual topics than a complete document.
- Authors with different expert knowledge can work simultaneously on topics. This makes revision faster without any loss of quality.
- You can test topics individually before the overall information product is finished.
- In the event of updates, only the new and revised topics have to be published.
- Translation of completed topics can start even if other topics still have to be edited.

DITA provides a formal framework for implementing tried and tested documentation techniques and practices that are suitable for meeting today's requirements. DITA is therefore excellently suited as a source format for equipping authors for fast and diverse developments.

Using DITA as a basis, you can efficiently set up a productive documentation environment for topic-based structuring, and develop it flexibly according to your needs.

The most notable advantages offered by DITA concepts and standardization are cost-savings and investment security. DITA provides a framework within which authors can start work immediately—without long-lasting structure-finding

processes that cost a lot of money. It is to be hoped that growing proliferation and broad tool support will ensure a long-term available technical basis.

Moreover, the wide use of the same basic architecture across corporate boundaries promotes the continuous growth of structuring know-how for information products from which new standards can be created. New standards can arise not only for contents at the lowest level but also for the resulting information products, such as different types of manuals, online helps, and mobile contents. In turn, the wide use of such standards will make exchangeability and automation possible in far more dimensions from those of today. Furthermore, authors can save a lot of time and work that they can profitably invest in the content.

The more DITA is used, the more the authors are supported by tools. The output of DITA sources to EPUB format already exists. There are Wikis on Drupal basis that use DITA as their source format, and more and more tools are being created to generate contents automatically for mobile display from DITA-tagged data.

### **Summary**

- DITA is an established standard suitable as a source format in any area and for all types of information products.
- Topic-based structuring is a good basis for meeting today's information management requirements flexibly and efficiently.
- Topic models (templates) of a class concept simplify the creation process and give authors more time for the actual contents.
- From the user's point of view, topics meet today's information expectations better than chapters or complete books. Good topics are short and easy to understand. They provide brief and targeted information. Equipped with metadata, they are suitable for just-in-time answers that can automatically be found on the basis of the current situation and the metadata.
- Standard environments can exchange contents with other standard environments and profit from the further developments of the standard and its tools.

## Appendix

### DITA 1.3 Overview

DITA 1.3 was released by OASIS on December 17, 2015 and is available on the OASIS web site [17].

This appendix summarizes the new DITA 1.3 features.

#### A.1 DITA 1.3 Editions

Version 1.3 is organized in three parts called editions:

- Base edition
- Technical content edition
- All-inclusive edition

Each edition is targeted at a different audience:

Figure A.1 from the DITA 1.3 overview, part 0, shows how the editions are related.

##### Base Edition

This edition contains the core DITA pieces for topic, map, and subject scheme map.

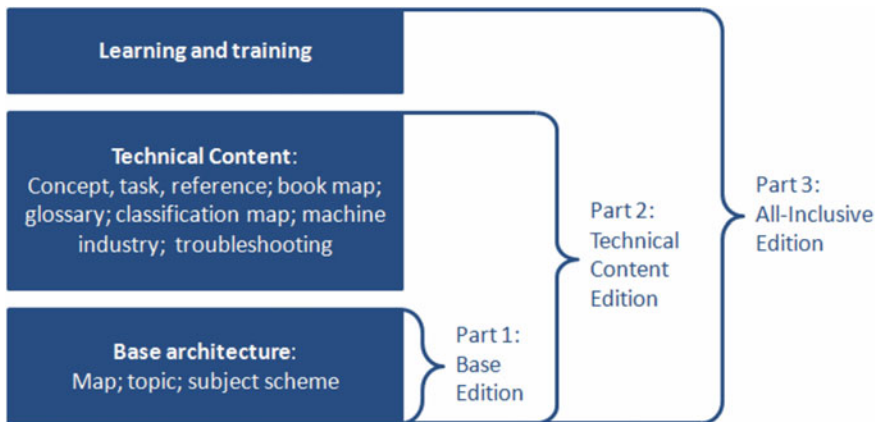
The base edition is designed for users who need only topics and maps and do not need the classic topic types of technical documentation such as concept, task, and reference.

This edition can be used just for authoring topics. It also can be used to develop specializations that do not need the classic topic types.

##### Technical Content Edition

This edition includes the base edition and the specializations for information typing: concept, task, and reference topics; machine industry task; troubleshooting topic; glossaries, bookmap, and classification map.

It is designed for authors who use topic types to modularize their content.



**Fig. A.1** DITA 1.3 editions

### All-Inclusive Edition

This edition includes the technical content edition and the specializations for learning and training.

It is designed for users who want all OASIS-approved specializations, as well as users who develop learning and training materials.

## A.2 What's New in DITA 1.3

DITA 1.3 provides a variety of enhancements.

Table [A.1](#) gives an overview of the DITA 1.3 features.

## A.3 Enhanced Support for Troubleshooting Information

DITA 1.3 offers various ways of providing troubleshooting information:

- `note` element
- `task` topic type
- `troubleshooting` topic type

### Troubleshooting enhancements for `note` element

With the new value `trouble` for the `type` attribute of the `note` element, a troubleshooting-related note can be explicitly identified.

**Table A.1** DITA 1.3 features

For	DITA 1.3 enhancement
Troubleshooting	Troubleshooting support is described in more detail in the following section
User assistance	<ul style="list-style-type: none"> <li>• New attributes for <code>resourceid</code>: <ul style="list-style-type: none"> <li><code>appid</code>: to specify an ID for an application</li> <li><code>ux-context-string</code>: to specify a context id for the topic</li> <li><code>ux-source-priority</code>: to specify how to resolve conflicts between <code>resourceid</code> definitions that exist in both a map and a topic</li> <li><code>ux-windowref</code>: to reference a window</li> </ul> </li> <li>• New element, <code>ux-window</code>, to specify a window or viewport in which an online help topic or Web page is displayed</li> </ul>
Version management	<ul style="list-style-type: none"> <li>• Further elements in topic prolog for version management</li> <li>• Release notes can be generated automatically</li> </ul>
Variant management	<ul style="list-style-type: none"> <li>• Branch filtering, which allows the application of different filters (<code>ditaval</code> conditions) to specific topic collections (branches) in a map</li> <li>• Scoped keys that support key definitions at different locations within a map structure. If a topic is re-used in several submaps, different key values can be specified, depending on the submap in which the topic appears</li> <li>• Expanded syntax for filtering attributes</li> </ul>
Addressing	<ul style="list-style-type: none"> <li>• New syntax for addressing an element within the same DITA topic (<code>#./label</code>)</li> <li>• Attribute <code>keyref</code> for object and param elements</li> <li>• Cross-deliverable linking</li> </ul>
Sorting	<ul style="list-style-type: none"> <li>• New <code>sort-as</code> element for use in sortable elements such as <code>title</code>, <code>searchtitle</code>, <code>navtitle</code>, <code>glossterm</code>, <code>dt</code>, <code>entry</code>, <code>stentry</code> where the base content is inadequate for sorting. The <code>sort-as</code> content is combined with the base content to construct the effective sort phrase</li> </ul>
Deliverables	<ul style="list-style-type: none"> <li>• New facility for key-based, cross-deliverable referencing</li> <li>• New conditional-processing attribute <code>deliveryTarget</code> replaces the <code>print</code> attribute and can have controlled values by using a subject scheme</li> <li>• Attribute <code>cascade</code> provides more control over cascading of metadata</li> </ul>
Vocabulary	<ul style="list-style-type: none"> <li>• Table extensions</li> <li>• <code>draft-comment</code>, <code>ph</code>, and <code>text</code> allowed in more places</li> <li>• More values for <code>style</code> attribute in a <code>ditaval</code> file</li> <li>• New <code>line-through</code> and <code>overline</code> elements</li> <li>• New <code>div</code> element for grouping elements in a topic</li> </ul>
Learning and training	<ul style="list-style-type: none"> <li>• New <code>learningObjectMap</code></li> <li>• New <code>learningGroupMap</code></li> <li>• New base domain and specialized domain for question-and-answer interactions</li> </ul>
Integration of other standards	MathML equations and SVG figures can be directly integrated into DITA topics
Specialization	Parts of structural specializations can be reused by other structural specializations without requiring one to be specialized from the other



**Troubleshooting enhancements for task topic type**

The new element `steptroubleshooting` in the content model for `step` is intended to contain information that might assist users when a step does not complete successfully or does not produce the expected result.

The new section `tasktroubleshooting` in the `task` topic type is intended to contain information that might assist users when a task does not produce the expected result or complete successfully.

**DITA troubleshooting topic**

A troubleshooting topic describes how a problem can be solved.

It begins with a condition that describes the problem, followed by one or more cause-remedy pairs. Each cause-remedy pair describes a potential solution to the problem described in the condition.

Table [A.2](#) shows specific troubleshooting elements in alphabetical order.

**Example of a troubleshooting topic**

For coffee which smells sour, the troubleshooting topic in Fig. [A.2](#) describes how to solve the problem.

The DITA source for this troubleshooting topic is shown in Fig. [A.3](#).

**Table A.2** Selection of specific troubleshooting elements

Element	For marking
cause	One potential source of the problem described in the <code>condition</code> element
condition	Problem to be solved
remedy	Step-by-step procedure as a potential solution for one cause of the problem described in the <code>condition</code> element
responsibleParty	Party responsible for performing a remedy procedure
troublebody	Body of a troubleshooting topic
troubleshooting	Root of a troubleshooting topic
troublesolution	Container element for cause and remedy information

## Coffee too sour

### Condition

Your coffee smells hollow and sourish.

### Cause: Coffee is too fresh

If the roasted coffee is too fresh, you will experience a lot of bubbles or quick dissipation.

### Remedy

Allow the beans a couple of days to settle and degas.

Keep in mind that coffee will go stale 21 days past the roast date.

### Cause: Not enough time to extract

Coffee that's too sour to the taste hasn't had enough time to extract.

### Remedy

Let the coffee draw for 1-2 minutes longer.

By adapting the brewing time, you make sure that the coffee retains all important flavorings, and the bitter substances are not dissolved out of the coffee grounds.

### Cause: Coffee grind too large

Particle size determines exactly how your coffee will extract, so a grind size that's large will require more extraction time than a grind size that's finer.

### Remedy

1. Bring your grind down a bit.
2. Try again.

Fig. A.2 Coffee troubleshooting

```
<troubleshooting id="trblshtng">
  <title>Coffee too sour</title>
  - <troublebody>
    - <condition>
      <title>Condition</title>
      <p>Your coffee smells hollow and sourish.</p>
    </condition>
    + <troubleSolution>
      - <troubleSolution>
        - <cause>
          <title>Cause: Not enough time to extract</title>
          <p>Coffee that's too sour to the taste hasn't had enough time to
            extract.</p>
        </cause>
        - <remedy>
          <title>Remedy</title>
          - <steps>
            - <step>
              <cmd>Let the coffee draw for 1-2 minutes longer.</cmd>
              <info>By adapting the brewing time, you make sure that the coffee
                retains all important flavorings, and the bitter substances are
                not dissolved out of the coffee grounds.</info>
            </step>
          </steps>
        </remedy>
      </troubleSolution>
    </troublebody>
  </troubleshooting>
```

Fig. A.3 DITA troubleshooting topic

# References

1. Bellamy L, et al (2012) DITA best practices: a roadmap for writing, editing, and architecting in DITA. IBM Press, Upper Saddle River
2. Closs S (2011) Single source publishing: modularer content für EPUB & Co., 2nd edn. entwickler.press, Frankfurt
3. Closs S (2014) Informationsarchitektur—Junge Disziplin mit großer Zukunft. Dok Magazin 3:69–72
4. Closs S (2014) Contextual content. tcworld e-magazine. <http://www.tcworld.info/e-magazine/technical-communication/article/contextual-content/>. Accessed 14 Aug 2015
5. Day D, Hargis G, Priestley M (2005) Frequently asked questions about the Darwin information typing architecture. <http://www.ibm.com/developerworks/library/x-dita3/#N104>. Accessed 14 Aug 2015
6. DITA OT: DITA open toolkit. <http://www.dita-ot.org/download>. Accessed 14 Aug 2015
7. DITA XML.org. <http://dita.xml.org>. Accessed 14 Aug 2015
8. Fritz M (2008) DITA in der Technischen Kommunikation—eine Entscheidungshilfe für den Einsatz, tekom, Stuttgart
9. Glushko RJ (2013) The discipline of organizing. MIT, Cambridge
10. Horn RE (1986) Engineering of documentation—the information mapping approach. Information Mapping Inc, Waltham
11. Java. <https://www.java.com/en/download/>. Accessed 24 Aug 2015
12. Muthig J, Schäfflein-Armbruster R (2014) “Funktionsdesign@—methodische Entwicklung von Standards”. In: Muthig J (Hrsg) Standardisierungsmethoden für die Technische Dokumentation 2nd Edition tcworld GmbH, Stuttgart, pp 41–73. See also <http://www.tcworld.info/e-magazine/technical-communication/article/technical-documentation-needs-standardization/>
13. MS Help WS: MS HTML help workshop. <http://www.microsoft.com/en-us/download/details.aspx?id=21138>. Accessed 14 Aug 2015
14. OASIS (2010) Darwin information typing architecture (DITA) Version 1.2. <http://docs.oasis-open.org/dita/v1.2/spec/DITA1.2-spec.pdf>. Accessed 14 Aug 2015
15. Rockley A (2003) Managing enterprise content, a unified content strategy. New Riders, Berkeley
16. DITA 1.3: Why three editions? <http://docs.oasis-open.org/dita/dita-1.3-why-three-editions/v1.0/dita-1.3-why-three-editions-v1.0.html>. Accessed 30 Jan 2016
17. OASIS package of the complete DITA 1.3 specifications and related files. <http://docs.oasis-open.org/dita/dita/v1.3/os/dita-v1.3-os.zip>. Accessed 30 Jan 2016

# Index

## A

Access, [51](#)  
Attributes for filtering, [37](#)  
audience, [37](#)

## B

Benefits, [51](#)  
Book-oriented production, [1](#)  
Branch filtering, [38](#)  
Branches, [33](#)

## C

Cascading stylesheet, [42](#)  
Class concept, [13](#), [51](#)  
Class concept method, [11](#)  
Classic cross-references, [25](#)  
Classification, [9](#)  
Collaboration, [39](#), [51](#)  
Concept, [9](#)  
Conref mechanism, [27](#)  
Constraints, [39](#)  
Content reuse, [5](#)  
Context-independent, [8](#)  
Conventions, [39](#)  
Cost-savings, [51](#)  
Cross references, [23](#)

## D

Darwin information typing architecture, [5](#)  
Defining variable, [36](#)  
`deliveryTarget`, [37](#)  
Designing topic models, [13](#)  
Developing “class concept”, [11](#)  
Direct addressing, [31](#)  
DITA block elements, [15](#)  
DITA community, [4](#)  
DITA concept, [17](#)

DITA glossentry, [19](#)

DITA history, [5](#)

DITA inline elements, [16](#)

DITA map, [22](#)

DITA open toolkit, [4](#), [43](#)

DITA reference, [18](#)

DITA specialization, [49](#)

DITA task, [16](#)

DITA topic, [8](#)

DITA topic types, [9](#)

`ditaval`, [37](#)

Documentation of DITA standard, [4](#)

## E

Embedding, [27](#)

Exchangeability, [52](#)

## F

Filter attributes, [37](#)

Filter file `ditaval`, [37](#)

Filtering, [37](#)

## G

Generating output with “DITA open toolkit”,  
[46](#)

Generic standard, [5](#)

Glossentry, [9](#)

Glossentry topic, [31](#)

## I

Indirect addressing, [31](#)

Information product, [41](#)

Information Typing, [5](#)

Installing DITA open toolkit, [43](#)

## J

Just-in-time answers, [52](#)

**K**

Key, [31](#)

Keys, [35](#)

**L**

Language definition, [4](#)

Lego system, [1](#)

Link-free, [23](#)

Links, [23](#)

machinery task, [9](#)

**M**

Modularization, [11](#)

**N**

Nesting topics, [21](#)

**O**

Open standard, [3](#)

Organization patterns, [22](#)

otherprops, [37](#)

**P**

Parameters for publication, [46](#)

platform, [37](#)

product, [37](#)

Program directory of DITA open toolkit, [44](#)

**R**

Reference, [9](#)

Related-links section, [25](#)

Relation, [23](#)

Relationship table, [24](#)

Reuse, [24](#), [27](#), [51](#)

Reusing, [7](#)

**S**

Security, [51](#)

Self-contained, [8](#)

Single sourcing, [27](#)

Source format, [5](#)

Subject schemas, [39](#)

Submaps, [22](#)

**T**

Table of contents, [22](#)

Task, [9](#)

Term definition, [7](#)

Topic-oriented structuring, [7](#)

Topic, [8](#), [9](#)

Topic types, [8](#)

Troubleshooting, [9](#)

**V**

Validities for keys, [33](#)

Variables, [35](#)

Variant management, [35](#), [51](#)

Variants, [35](#)

**W**

Warehouse topic, [28](#)

**X**

XML production process, [42](#)