Mastering Technical Writing

From DITA to MadCap to universal design principles, everything that you need to begin writing successful technical content
Introduction

This booklet is for those who are aspiring technical writers, or those who are just beginning technical writing. In 6 brief sections, you will learn the foundations of technical communication, and will be ready to get out there and begin producing your own technical content.

Some important topics that will be covered include:

> DITA and how it works
> Single source documentation
> MadCap Flare
> Accessibility and universal design
Table of Contents

Part 1: The Roots 1
   What is DITA? 2
   How does it work? 2

Part 2: All About XML 3
   What is XML? 4
   What is the history of XML? 4
   How is XML relevant to technical documentation? 5
   What does XML look like? 5

Part 3: Single Source 6
   What is single source documentation? 7
   What are its benefits and challenges? 7
   Ideal Single Source Documentation 8
   Realistic Single Source Documentation 8

Part 4: MadCap Flare 9
   What is MadCap Flare? 10
   How does it work? 10
   What are some good Flare resources? 11

Part 5: Accessibility 12
   What is accessibility? 13
   How do I learn more about accessibility? 13

Part 6: Universal Design 14
   What is universal design? 15
   What are the 7 Principles of Universal Design? 15

Works Cited 16
Before there was MadCap Flare, there was DITA. In this chapter, you will be going back to the roots of technical documentation. This will make it much easier for you to learn Flare later on!
What is DITA?

The Darwin Information Typing Architecture, or DITA, is an XML-based architecture (rather than a tool) that serves to organize technical content. It supports the reuse of content, which makes technical writing easier and more efficient. DITA began as an IBM internal initiative. It was created to replace IBM’s IBM ID Doc, which was a more book-oriented architecture. DITA is now considered an OASIS Standard.

DITA can publish information in a variety of formats including PDF, XHTML, Microsoft Compressed HTML Help (.chm), and many others.

How does DITA work?

DITA is based on the creation and organization of topic types. Topics are a facet of XML, and are defined as abstract subjects characterized by names, relationships, and resources. Each topic pertains to a single subject. DITA contains rules for the kinds of topics that can be used, how they can be written, and how they can be grouped together. This strict, topic-based documentation helps to solve several issues that typically occur while producing technical content, including:

- Lack of modular format
- Inconsistent writing styles from the author(s)
- Redundancy
- Lack of separation between content and format
- Lack of content reuse
Part 2: All About XML

In order to use DITA, and ultimately MadCap Flare, you need to understand XML.
What is XML?

XML, or Extensible Markup Language, is a type of computer language that facilitates universal data access. XML is a “plain-text, unicode-based meta-language’ (Microsoft). More simply put, it’s a language for defining markup languages. What makes XML individual is that it is not tied to any programming language or operating system. XML provides a simple format and structure to tag all levels of information with the end goal to provide a desired structure. XML makes information more accessible and easy to organize and retrieve information in a computer format. One of the most defining traits of XML is that there are no designated tags as there is in HTML. As shown in the example below, any defining tag you want to give a set of data you can, and the computer will recognize and save that XML data, compiling it when you do a given search. An example from below would be the tag <city>.

What is the history of XML?

XML began with the creation of the Standardized Generalized Markup Language (SGML) by Charles Goldfarb while working as an employee of IBM in 1970. Although SGML has the term ‘markup language’ in its actual name, it is not a defined markup language, only a language which specifies and helps to define markup languages.

The whole purpose of creating SGML was to make it possible for computers to recognize a set of tags so that a given computer could understand a SGML document even 100 years from now. After the creation of SGML, it took around twenty years for XML to come from it, as one of the most standardized computer languages used in the world. XML took 20 weeks to be developed by over 100 computer programmers worldwide. It was finally designed and finished on February 10, 1998, released as XML 1.0.
How is XML relevant to technical documentation?

XML is not tied to any programming language, operating system or software vendor. Unlike HTML it does not have a fixed vocabulary. Instead, one can define vocabularies specific to particular applications or industries using XML. In fact, it is fairly straightforward to produce or consume XML using a variety of programming languages. The adaptability that XML has makes it ideal for all different kinds of programming platforms and operating systems. Besides being able to represent both structured and semi-structured data, XML is widely used because of its accessibility. XML is accepted as the go to mark up language for data representation.

Here are some quick and easy introductory links on XML and its use today. Along with tutorials, examples, and resources, these sources provide a clear overview of the markup language:

http://www.w3schools.com/xml/xml_usedfor.asp
http://www.xmlfiles.com/xml/xml_usedfor.asp

What does XML look like?

```xml
<?xml version="1.0" encoding="UTF-8"?>
<customers>
  <customer id="1">
    <name>Woodworks</name>
    <city>Baltimore</city>
    <projects>
      <project id="1">Medusa</project>
    </projects>
  </customer>
  <customer id="4">
    <name>Hardware Shop</name>
  </customer>
</customers>
```
Every good technical writer needs single source documentation. DITA and MadCap Flare help you to create and organize this type of documentation.
What is single source documentation?

Single-source documentation is a technique used by tech writers in which the same source content is used in multiple documents and media types. You manage it in one place, and only need to edit the original document to make changes across all forms.

What are its benefits and challenges?

Benefits:

> Research and editing is only done once, on one document.
> The document can be stored in one place and reused.
> There are fewer chances for errors because all the information is in the same place and one correction affects everything.
> You only have to modify the file once to make changes/fixes.
> It’s easier to maintain consistent corporate branding.
> It makes it easy to new programs and browser types.
> You can export relevant information to target specific groups.

Challenges:

> There’s a lot of the same information to sift through.
> If you miss a mistake it appears everywhere.
> All the information is linked, so you can’t change only one document.
> It requires a sophisticated approach to technical documentation that many companies can’t afford yet.
Ideal Single Source Documentation

Realistic Single Source Documentation

Part 3: Single Source | 8
Part 4: MadCap Flare

Now we’re getting to the good stuff. MadCap Flare is one of the most useful tools a technical writer has. This section will teach you all about it.
What is MadCap Flare?

MadCap Flare, like DITA, is an architecture for producing technical documentation. RoboHelp, its predecessor, was purchased by Macromedia in 2003. By 2005, Macromedia had abandoned the program, and let the RoboHelp team go. The team then contacted RoboHelp’s original creator, Bjorn Backland, and together they designed MadCap Flare. RoboHelp has since been restarted by Adobe and is one of MadCap Flare’s competitors. Adobe FrameMaker is also a competitor.

The idea behind MadCap Flare is “write once, publish anywhere.” This makes the lives of technical writers easier as they can avoid having to keep track of multiple versions of a document.

How does it work?

Unlike a word-processing program, MadCap Flare organizes documents according to topics, and formats them so that they can be converted to many different forms of technical documentation. Some of these forms include software and API documentation, user guides, policy and procedure manuals, and knowledge bases. The documentation can also be produced in print, online, and on mobile devices.
What are some good Flare resources?

The MadCap Flare website has plenty of resources for new Flare users. If you click on the Help & Support tab, then click on MadCap Flare, a window opens to Flare’s online help. Here you can find PDF guides that will teach you about Flare. There are also numerous video tutorials if you prefer to learn that way located here http://www.madcapsoftware.com/videos/.

Here are some of the videos located on the site:

“Where Do I Begin?”
“Overview Of MadCap Flare”
“What’s New In Flare 11”
“Meet MadCap Contributor”

MadSkills, Inc., which is a subsidiary of MadCap Software, offers training both in a classroom and online. MadCap Software also has its own forum, where users can assist each other with Flare issues.

In addition to MadCap’s resources, there are guides available to teach you Flare, like Five Steps to Flare Fourth Edition by Lorraine Kupka. There are also certified MadCap Flare trainers that you can consult for lessons. A couple of the more popular trainers are Paul Pehrson from DocGuy Training, and Eddie VanArsdall from Content Insomnia.
Part 5: Accessibility

It is important for your technical content to be available in a variety of formats. This way you are not limiting who can read your material.
What is accessibility?

Accessibility means anyone can access information, regardless of their learning preferences or disabilities. Documentation is accessible if there are in-file aspects to assist with reading/understanding the material, like a site that can easily be translated to a different language. If it’s not possible to make a certain piece of documentation accessible, there should be an alternative format that is accessible.

How do I learn more about accessibility?

Here are a few resources:

> W3C site
> Learning about users (how old are they? What are they using to access it? What barriers may they encounter?)
> Using programs like Flare that alert you when you’re not being accessible, or Dreamweaver which lets you know when you enter elements that will need to be adjusted to be accessible
> User testing
> Web Content Accessibility Guidelines (WCAG) has a shared standard for web content accessibility and offers tips and explanations for making basic content more accessible.

Web accessibility standards:

Rehabilitation Act Section 508 requires federal agencies to follow certain technology accessibility standards. Web Content Accessibility Guidelines (WCAG) were created by Web Accessibility Initiative (WAI) as part of the World Wide Web Consortium (W3C).
Part 6: Universal Design

This goes hand in hand with accessibility. You want to make sure that your design allows anyone to access your content.
What is universal design?

Universal design is a way of designing documentation so that it can be accessed and understood by as inclusive an audience as possible. Any type of documentation or environment should be made for its audience’s use, and should consider the potential roadblocks in the audience’s understanding.

Disability Act 2005 defines universal design as:
1. The design and composition of an environment so that it may be accessed, understood and used:
   i. To the greatest possible extent
   ii. In the most independent and natural manner possible
   iii. In the widest possible range of situations
   iv. Without the need for adaptation, modification, assistive devices or specialised solutions, by any persons of any age or size or having any particular physical, sensory, mental health or intellectual ability or disability, and
2. Means, in relation to electronic systems, any electronics-based process of creating products, services or systems so that they may be used by any person.

What are the 7 Principles of Universal Design?

1) Equitable use
   > Useful/marketable to people with a variety of abilities
2) Flexibility in Use
   > Accommodates preferences and abilities
3) Simple and intuitive use
   > Easy to understand regardless of user’s past experience/language
4) Perceptible information
   > Communicates information regardless of user’s senses
5) Tolerance for error
   > Small mistakes won’t make the entire thing crash
6) Low physical effort
   > Can be used comfortable without a lot of stress/physical effort
7) Size and space for approach and use
   > Should be easily accessible regardless of user’s body size/posture/mobility
Works Cited

Resources used:

http://www.topicmaps.org/xtm/
http://www.xmlmind.com/tutorials/DITA/#topic_structure
http://www.w3schools.com/xml/xml_usedfor.asp
http://www.xhtmlfiles.com/xml/xml_usedfor.asp
http://www.madcapsoftware.com/products/flare/#multiChannel
http://www.pcworld.com/article/225283/MadCap_Flare.html
http://www.ebstc.org/newsletter/0307/Article_1.htm
http://www.madcapsoftware.com/products/flare/
http://webhelp.madcapsoftware.com/flare11/Content/Home.htm
http://www.madcapsoftware.com/services/training/
http://www.northcoastwriters.com/fiddleheadpubs/

Image credits:

http://agilemodeling.com/images/singleSourceInformationRealistic.jpg
http://agilemodeling.com/images/singleSourceInformationIdeal.gif
http://i1-win.softpedia-static.com/screenshots/MadCap-Flare_1.png
http://www.stylusstudio.com/images/figures/sql_xml_xml_fragment.gif