3rd Edition
covers
Web 2.0
and DITA

Technical Writing 101

A Real-World Guide to Planning and Writing Technical Content

by

Alan S. Pringle
Sarah S. O’Keefe

3rd Edition
covers
Web 2.0
and DITA

Technical Writing 101

A Real-World Guide
to Planning and Writing
Technical Content

by

Alan S. Pringle
Sarah S. O'Keefe

Technical Writing 101: A Real-World Guide to Planning and Writing Technical Content

Third edition

Part of Chapter 13 is adapted from Scriptorium Publishing's white paper, "Structured Authoring and XML," which was originally published at www.scriptorium.com/structure.pdf.

Chapter 14 is adapted from Scriptorium Publishing's white paper, "Friend or Foe? Web 2.0 in Technical Communication," which was originally published at www.scriptorium.com/whitepapers/web2/web2intc.pdf.

Published by Scriptorium Press, the imprint of Scriptorium Publishing Services, Inc.

For information, contact:

Scriptorium Publishing Services, Inc.
PO Box 12761
Research Triangle Park, NC 27709-2761 USA
Attn: Scriptorium Press

www.scriptorium.com/books
books@scriptorium.com

ISBN: 978-0-9704733-7-0

About the authors

Alan S. Pringle is director of publishing operations at Scriptorium Publishing Services, Inc. He implements new processes for developing and distributing technical content. His responsibilities include managing schedules and budgets for complex consulting projects, automating production and localization tasks with XML-based workflows, and working on large-scale DITA conversions. Alan guides books through the entire publication process as the manager of Scriptorium's publishing imprint, Scriptorium Press. Alan edited *Publishing Fundamentals: Unstructured FrameMaker 8, Publishing Fundamentals: FrameMaker 7*, and *The WebWorks Publisher Cookbook*.

Sarah S. O'Keefe is president of Scriptorium Publishing. Since founding the company in 1996, Sarah has focused on efficiency—selecting the right publishing tools, creating templates, and training writers on how to use their tools. In 2002, she received her Certified Technical Trainer (CTT+) accreditation from CompTIA. Her presentations at international, national, and regional conferences have consistently earned high ratings, and she is an Associate Fellow of the Society for Technical Communication (STC). Her publishing credits include *Publishing Fundamentals: Unstructured FrameMaker 8*, *Publishing Fundamentals: FrameMaker 7* (originally published as *FrameMaker 7: The Complete Reference*), *The WebWorks Publisher Cookbook*, and numerous white papers.

Acknowledgments

Special thanks to the following people, who helped make this a better book:

- Bill Burns for contributing to the chapter on localization and internationalization
- Sean Byrne for drawing the illustrations
- David Kelly for designing the cover
- Larry Kunz for recommending updates in this edition
- Sheila Loring for completing the final review
- Terry Smith for reviewing the book and updating the index
- The staff at Scriptorium Publishing for contributing to the section on DITA

Contents

Preface

For technical writers, answering the obligatory "What do you do for a living?" question at a party can have many effects. It can:

- Create more questions: *"What's that?"*
- Draw blank stares
- Provoke some minor hostilities: *"Did you write that worthless online help that came with my word processing software?"*[1]

So, if you've decided that you want a career in technical communication, be prepared. Although you'll have a challenging, fast-paced job that changes as swiftly as the technology you write about, discussing your work at a party will be quite a conversation stopper.

What's in this book

Technical Writing 101 will show you that there's more to technical writing than just writing. The first major section of the book explains the skills you need as a

[1] Notice the skillful use of bullets, the hallmark of a good technical writer.

writer. It also describes some of the essential tools and techniques for delivering projects on schedule and on budget. The chapters in this section are:

- Chapter 1, “So, what’s a technical writer?”

  Explains what technical writers do and what skills they need.

- Chapter 2, “The technical writing process”

  Provides a high-level view of the technical writing process and explains the two predominant workflows authors follow to complete the steps in that process.

- Chapter 3, “Very necessary evils—doc plans and outlines”

  Explains two project management tools that every technical writer needs. Documentation plans provide writers with a roadmap to follow as they create materials. Typically, a plan includes a description of the target audience, the schedule, and a list of documents or online help to be developed. Outlines are just that—those hideous indented things you probably remember from high school. Unfortunately, you’ll discover that outlines are in fact a necessity for technical writers.

- Chapter 4, “The Tech Writer’s Toolbox”

  Focuses on the tools and technologies that you need to work successfully. Technical writers use a variety of writing and graphics packages to develop material.

In the second major section, you learn about how to get information, organize information, and (finally) write content. The chapters also describe other tasks in the content development process, such as creating visual content, technical editing, production editing, and indexing. The chapters in this section are:

- Chapter 5, “Getting information”

  Gives you tips on how to extract information from source documents and product developers. Many people respond well to bribery, especially when the bribe is edible and includes chocolate in some form.

- Chapter 6, “Finally—it’s time to start writing”

  Describes how to write content. When writing, it’s important to address the document’s audience and to divide your content into different types of information—interface, reference, conceptual, and procedural.

- Chapter 7, “Writing task-oriented information”

  Explains the basics of writing procedures. Because most technical documents tell users how to perform tasks, the ability to write good task-oriented information is a fundamental skill for all technical writers.

- Chapter 8, “Visual communication”

  Describes how to create and work with visuals. A graphic (or video clip) may not be worth exactly a thousand words, but visual content can often explain something with more clarity than any amount of text.

- Chapter 9, "The importance of being edited"

  Editing is an essential component of the content development process. A solid review by an editor or a fellow writer can make you look good by catching your mistakes before the client or users see them.

- Chapter 10, "Indexing"

  Explains the basics of writing a good index. A thorough, useful index is essential because readers often check the index first when looking for a particular piece of information. A good index can also save a company money—readers who quickly find the information they need are less likely to call customer support.

- Chapter 11, "Final preparation—production editing"

  Tells you how to ensure content is ready to be printed—or released as online help.

The third section explains some advanced topics. Chapters in this section are:

- Chapter 12, "Avoiding international irritation"

  Offers some tips on minimizing the hassles that occur when content is translated into other languages. Learning about the translation process before you start writing the English content can save your company a lot of time and money—and prevent many, many headaches.

- Chapter 13, "Structured authoring with XML"

  Explains structured authoring—a publishing workflow that defines and enforces consistent organization of information in documents—and how it affects technical writers. The chapter also offers an overview of the Darwin Information Typing Architecture (DITA), a structured authoring standard that many documentation groups have adopted.

- Chapter 14, "Web 2.0 and technical communication"

  Covers Web 2.0 technologies (such as blogs and wikis), which encourage the sharing of information on the Internet. These technologies have changed the way users get information about products, so technical writers need to consider Web 2.0 technologies when creating content.

The appendices provide information about getting a job, a list of resources, and a sample documentation plan. The appendices are:

- Appendix A, "Getting your first job as a technical writer"

  Gives you some pointers on tailoring your resume for technical writing jobs, interviewing, and putting together a portfolio.

- Appendix B, "Resources"

  Lists web sites, books, and organizations that are useful for technical writers.

- Appendix C, "Sample doc plan"

  Contains a sample documentation plan.

This book focuses on developing content for computer hardware and software. However, many of the concepts described apply to other forms of technical writing, such as writing about manufacturing environments, medical and pharmaceutical topics, and science.

If you're a talented writer with an interest in technical topics, writing technical content can be quite lucrative. This book gives you the advice and tools you'll need to get started in this challenging field.

1 So, what’s a technical writer?

What’s in this chapter

- Knowledge of technology
- Writing ability
- Organizational skills
- Strong detective (and people) skills

The short definition of “technical writer” is a person who writes about technical topics. But perhaps a better definition is someone who can explain complicated concepts in clear, easy-to-understand prose.

A technical writer is really a translator. You start with a complicated piece of technology, and your mission is to explain to a nonexpert how to use that technology.

This deceptively simple mission requires more than just writing ability and an understanding of technology—although both of those skills are critical, they aren’t enough.

As a technical writer, you need strong organizational skills because you have to organize the information you gather. In many cases, getting information isn’t difficult, but identifying what’s relevant for your readers can be challenging. Many experienced technical writers create several new books each year. They must absorb new information and then understand, organize, prioritize, and deliver it.

You also need people skills and investigative talent. The stereotypical crabby, difficult starving artist/writer won’t make it as a technical writer. The information you need resides in someone’s head, so you must be able to work with that person to (gently?) extract the information.

This chapter describes the four basic skill sets every technical writer needs:

- Knowledge of technology
- Writing ability
- Organizational skills
- Detective and people skills

You’ll find technical writers who are stronger in one skill set and weaker in another, but every successful technical writer needs to be competent in these four basic areas.

Knowledge of technology

The “technical” part of technical writing doesn’t necessarily mean that you must be a programmer, electrical engineer, or other hardcore techie. However, you should be comfortable with and have some basic knowledge about the technology you'll be documenting. For example, if you plan to become a software technical writer, you should understand how to use an operating system (such as Windows), and you should be comfortable using various applications.

You should also understand basic concepts such as databases, networking, the Internet, and file manipulation (opening, closing, and saving). You should be willing (or even eager) to learn about new technology as it develops. Remember, you're the one who's expected to write the content, so you need to be able to figure out how the hardware or software works.

That doesn't mean, though, that you're completely on your own. In most jobs, you have access to information about the product you're documenting and can ask the programmers or engineers for details.

If your job requires that you write about something particularly technical, you need a higher level of knowledge. For example, if you're documenting a C++ programming tool, you should have a basic grasp of C++ programming. You don't have to be a programmer, but you do need to understand the fundamentals of programming.

You can learn about new developments in technology by subscribing to computer magazines or online mailing lists, or by joining user groups in your community. See Appendix B for more information about resources for technical writers.

Typing 101

If you're a good typist, you may be tempted to put your typing speed on your resume. Don't. Decent typing speed is a basic necessity for any technical writer, much like breathing. By putting typing speed on your resume, you are implying that you will be evaluated only on how fast you can pound out words. And that's not the case—you need to pound out the *right* words.

Your typing technique is not important. Touch typing is nice, but hunting and pecking is also fine, provided that you can type at about 40–50 words per minute. If using a keyboard is a laborious, error-prone, and tedious task for you, consider getting some typing software and improving your typing speed.

If spending most of the day at a keyboard typing seems unappealing, technical writing is probably not the job for you.

Ignorance is bliss

Writing about new technology (often nonexistent at the beginning of the project—and sometimes surprisingly close to the end) means that you can't possibly be expected to know everything about the new stuff. Your initial lack of knowledge, however, works for you. As you learn how to use the new hardware or software, you encounter many of the same issues that the users (people who buy the product) will face as they get started.

This experience should play a critical part in how you write your content: "I had a hard time figuring out how to use the interface to edit the file, so I need to be sure I explain that thoroughly to new users."

Initial ignorance—or, even better, the ability to pretend ignorance—can be a valuable asset when you're writing content. But don't use this research technique to avoid learning about existing technology. By understanding what's already out there, you build a strong foundation for writing about what's to come.

Who treats the doctor and who documents for the writer?

As a technical writer, you need to be able to figure out new concepts with little or no assistance. Often (always?), product developers work long hours to meet aggressive deadlines. They do not want to spend a lot of time explaining the product to you, so it's up to you to learn about the product with little or no help. Sometimes, you can get information from other folks who know the product or are learning about it—for example, tech support personnel, trainers, and customer service reps. Save the limited time you'll get from the developers for the hard questions.

The Family Member Test

A side effect of your career as a technical writer is that your family and friends will assume that you're a computer expert. After all, if you spend your time writing about computers, surely you can explain computers to them.

You may not think of yourself as a computer expert, but you'll likely find that you do have enough expertise to help. If you're writing content for the general public, consider using the Family Member Test—whether the document you're writing will make sense to a family member who isn't a computer expert—to determine whether you're on the right track.

Writing ability

Writing skills are an essential component of technical communication. It seems terribly obvious to say that a technical *writer* should be able to write, but it's necessary to point this out. Enormous technical knowledge is not a substitute for writing ability; if you're not convinced, try reading material written by programmers sometime.[1]

The ability to break down complicated information into content that is appropriate for the document's audience is very important. Chapter 6, "Finally—it's time to start writing," covers this in detail.

The technical writer's mission is to create content (text and graphics) that communicates information to the reader. It's hard to identify exactly what makes good writing, but here are some general guidelines. Writing should be:

- Clear
- Easy to understand
- Not subject to misinterpretation
- Concise
- Easy to follow

It's easier to say what writing should *not* be:

- Confusing
- Redundant
- Wordy
- Poorly organized
- Inaccurate

Miss Thistlebottom was right…

An important component of clear, easy-to-understand writing is using correct grammar and spelling. As a technical writer, you must be able to produce content

[1] *Some* programmers are excellent writers.

that's grammatically correct and doesn't have any spelling errors. Notice that we say "produce." Nobody writes error-free prose during a first draft. But you should spell-check, review, and proofread your drafts before anyone else—including your editor[1]—sees them.

However, following the standard rules of language comes with a significant caveat. In some cases, adhering to the most formal rules of writing is not the best method for communicating with readers. For example, assume that you are writing content that tells workers on an assembly line how to use time management software. Which of the following steps would be better suited for that information?

> Select the name of the building in which you work.
> Select the name of the building you work in.

Even though the first example would probably make your high-school English teacher happier, it sounds pretentious. Your readers probably have well-tuned B.S. detectors, and it's unlikely that their priorities include avoiding a preposition at the end of a sentence. If creating a document with a casual tone will improve your reader's comprehension (and your credibility as the author), that goal is probably more important than following old-school rules of writing.

While writing, you will probably be required to follow style guidelines, either from a standard book (such as *The Chicago Manual of Style*) or from a company style guide. Style guidelines specify how you should capitalize section headings, for example, or punctuate bulleted lists.

[1] Assuming that you're lucky enough to have an editor.

NOTE: You probably won't agree with all of the guidelines your company follows. Don't waste your time arguing about them, though, unless you can show that your preferred style improves reader comprehension enormously. Otherwise, you're just imposing your personal style preferences on the company. Arguing about nitty-gritty style issues is a great way to waste time and make yourself look unproductive.

If your writing skills need work, practice writing about any topic. Writing recipes, for example, tests whether you can remember to document every step in the process. Ask a friend to try your recipe and see whether you forgot any steps.

By far the best way to improve your writing skills, though, is to write a lot and have someone else critique your writing. Many fiction writers belong to writing groups; you might consider finding a few other new technical writers and establishing a similar group.

Organizational skills

Almost every job requires some organizational skills, but technical writing demands quite a bit. You obviously need the ability to organize information in your document. But to succeed as a technical writer, you also need project and time management skills.

Planning a schedule for a book means scheduling your writing time, plus time for many other activities—creating art, getting drafts reviewed, editing, indexing, and preparing for final output. Depending on where you work, you may be responsible for all of these activities, or you may need to coordinate with others, such as editors, to add your book to their schedules. For example, if your company has a technical editor, notify the editor about the project so that your book is scheduled for an edit. If your company's editing is limited to peer reviews (another writer editing your work), find out who might be available to look at your document.

To ensure that you can meet project deadlines, it's critical that the time and resources required for each activity are clearly spelled out at a project's start.

All of this probably sounds a bit intimidating. After all, you want to be a writer, not a project manager! In many companies, entry-level or junior writers aren't required to create schedules; a senior writer or documentation manager does the planning work.

If you are the only writer in a company, however, you may have to handle scheduling right from the beginning. It's unlikely that you'll get any help from your coworkers (they probably don't know a thing about developing content), so scheduling and time management will be particularly important skills in this environment.

Many writers are confident about their writing and technical ability but are intimidated by the project management and scheduling requirements of the job. Consider taking a seminar on project management and be sure to maintain a calendar or spreadsheet with scheduling information. Project management software can also be helpful, especially for large projects (for example, projects with lengthy schedules, multiple deliverables, or many people involved).

Strong detective (and people) skills

To create content, you need information. And unlike writing a novel, you can't just make up that information (although creative extrapolation is often required when you can't get any information—more on that in Chapter 6). Where do you get the information you need? Sources include technical specifications, prototypes, and product developers, as explained in Chapter 5, "Getting information."

Often, the problem is not getting information but identifying what information is relevant. You might have to ferret through piles of technical specifications to find exactly what you need, so filtering information is an important part of technical writing.

Sometimes, you need information from a product developer or a subject matter expert (SME, which is pronounced "smee") who is extremely busy and may not want to make time for you. Learning how to communicate and get what you need from busy technical experts is a skill you cultivate over time. (Chapter 5, "Getting information," contains some helpful tips on how to extract information from documents and from people.)

Now that you have an idea of the skills you need as a technical writer, let's take a look at what you do with those skills to develop content.

Part-time contributors need technical writing skills, too

In many companies, employees other than technical writers contribute content for a product's documentation—often, product developers or those in the marketing department pitch in and provide some text.

These part-time contributors may not have the title of "technical writer," but they perform some of the same tasks. This book can be a helpful resource for them, too.

2 The technical writing process

What’s in this chapter

- What you can expect (maybe)
- Authoring with templates and with structure

Despite having the word “writing” in its name, technical writing includes many tasks, including visual design, editing, and project management. It’s likely you will complete some or all of those tasks, particularly in a small documentation group—and one-person documentation departments are not uncommon.

What you can expect (maybe)

Creating technical content, at a minimum, requires that you and your coworkers:

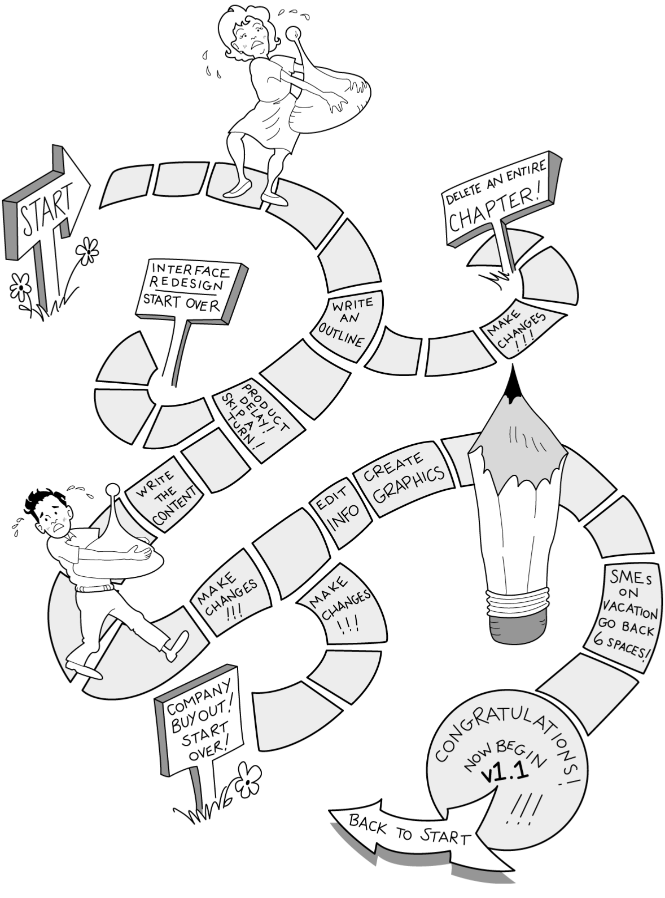
1 Identify the needed *deliverables*—the final products you turn over to the client or release to users at the

end of the project. Deliverables can include books, online help, web pages, tutorials, videos, and so on.

2. Develop an outline or a plan for each deliverable.
3. Create an overall project plan with a list of tasks for each deliverable and a draft schedule. Tasks include outlining, creating storyboards, writing, editing, designing visual content, reviewing, and so on.
4. Design the content's template or get your department's existing template. A *template* is a file that contains all of the paragraph styles, layouts, and other formatting information for your content. A template may also contain definitions for a document's structure; for example, the structure may require that every list have at least two list items. ("Authoring with templates and with structure" on page 40 has more information about authoring based on templates and structure.)
5. Create the content.
6. Create the visuals, including both still images and content with motion and audio.
7. Get the information reviewed by SMEs. Make the changes identified by the SMEs.
8. Edit the information and then have the material reviewed by a peer or technical editor. Make the changes noted. (Some departments have content edited before it's reviewed by SMEs.)
9. Index the content.

10 Produce the material (that is, clean up the formatting and get everything ready for the printer, the web, or other final delivery format).

During a real-life project, however, you'll get approximately halfway through step 6, at which point you'll discover that the developers have added a slew of new features to the product.


START
DELETE AN ENTIRE CHAPTER!
INTERFACE REDESIGN START OVER
WRITE AN OUTLINE
MAKE CHANGES !!!
PRODUCT DELAY! SKIP A TURN!
WRITE THE CONTENT
EDIT INFO
CREATE GRAPHICS
MAKE CHANGES !!!
MAKE CHANGES !!!
SMEs ON VACATION GO BACK 6 SPACES!
COMPANY BUYOUT! START OVER!
CONGRATULATIONS! NOW BEGIN v1.1 !!!
BACK TO START

You'll go back and document those features, test them against the product, and then discover that the developers also took out a couple of features without telling you. You check with a friendly developer around the corner and discover that those features are just "temporarily disabled"; development found some bugs but expects to correct the problems before the final release.

At this point, you have to make a decision. Do you assume that they will restore the features before the release, or do you delete the content? Or do you hedge your bets by making a copy of the information but removing it from the content for now? Every project is full of happy surprises like these. The rest of this book provides more information about how the technical writing process works and offers helpful information about how to handle the inevitable bumps in the road.

To complete the steps in the content development process, a technical writing department should use a template-based approach. Some groups go beyond templates by defining a structure that content must follow.

Authoring with templates and with structure

In general, you can classify the methodology for developing technical content into four levels:

- Chaos
- Page consistency
- Template-based authoring
- Structured authoring

The first two levels, chaos and page consistency, exemplify what you *shouldn't* do when creating content.

Chaos is exactly what you think it is: there is no consistency in the creation or the presentation of content. Each author develops content in a different manner, and users end up with information that is wildly inconsistent.

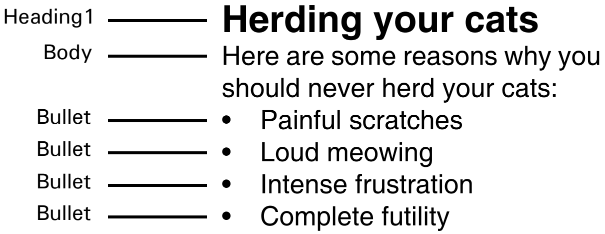
Page consistency means that content looks the same on paper (or other delivery format). However, when you look at the source files for the content, there is no consistency. For example, one author defines tab spaces to indent paragraphs, while another author presses the space bar five times to achieve the same look. Even though users see consistent formatting, a lot of time was wasted developing the information because authors didn't follow any standards.

In the chaos and page-consistency scenarios, it is impossible to create quality content with a high degree of efficiency. There are no repeatable processes in place to guide authors while they develop content. As a result, they struggle with the process of authoring instead of focusing their efforts on distilling information for product users.

Template-based authoring

At a minimum, professional writing departments implement templates. Because templates provide predetermined styles, writers don't spend time figuring out how to create particular formatting—they apply styles to add formatting. Figure 1 on page 42 shows the application of paragraph styles to text.

Figure 1: *Applying paragraph styles*



Herding your cats

Here are some reasons why you should never herd your cats:

- Painful scratches
- Loud meowing
- Intense frustration
- Complete futility

To help writers apply the correct styles to content, departments usually write documentation about the template or include that information in the department's style guide. The guidelines tell authors what styles to apply ("Use the Heading1 paragraph style for a section title."), and they often explain how authors must arrange content:

- A heading must be followed by an introductory paragraph.
- A bulleted list must contain at least two items.
- A graphic must have a caption.

An author's content is reviewed during the editing process to be sure it follows the rules. (Chapter 9, "The importance of being edited," describes editing tasks in more detail.)

Structured authoring

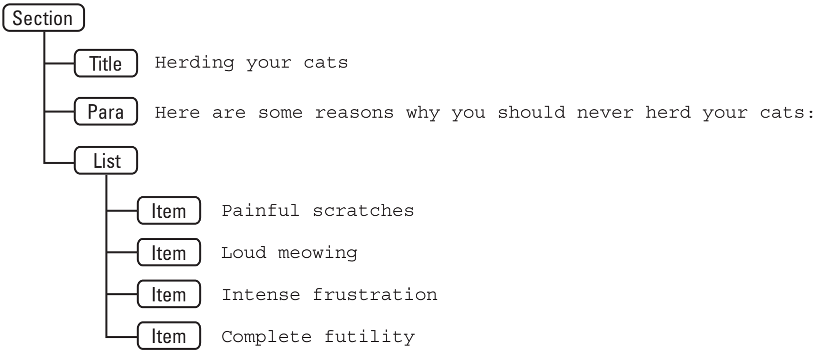
More and more technical writing departments are implementing *structured authoring*, a publishing workflow that defines and enforces consistent organization of content. In template-based publishing, the style guide lists content rules and explains what template styles should be applied, and an editor or another writer reviews content to ensure the information conforms to the approved styles.

In structured authoring, those rules are captured in a structure definition document, which is a template that defines a hierarchy. Writers work in software that *validates* their content; the software verifies that the content they create conforms to the rules in the structure definition document.

In addition to enforcing structure, structured authoring workflows separate formatting tasks from the creation of content. Instead of typing content and applying styles, authors assign elements to bits of content. Based on the location of an element within the content's hierarchy, formatting is automatically applied. Authors no longer apply paragraph styles or need to know which styles are allowed in particular contexts.

Figure 2 shows a hierarchical view of the cat-herding example in Figure 1.

Figure 2: *Hierarchy of a section*



Based on the location of each element within the hierarchy shown in Figure 2 on page 43, the structured authoring environment applies formatting automatically. For example, when a Title element is in a Section element, the Heading1 paragraph style is applied to that element. The formatting can look just like what is shown in the template-based example in Figure 1 on page 42, but there is an underlying structure that determines the formatting—and the author didn't apply any of the styles.

In some structured workflows, the author doesn't see the final formatted version (or versions if content is released in multiple outputs) while using an authoring tool to create content. Instead, the tool may show basic formatting that resembles a word processing document to make the content easier to view. In a later step, the workflow automatically applies the full formatting before the content is released.

Later chapters in this book explain how working in a template- or structure-based workflow affects the way you complete steps in the technical writing process. Chapter 13, "Structured authoring with XML," provides detailed information about structured authoring.

Are templates and structure *really* that important?

Yes, they are, which is why we're telling you about them so early in this book.

The demand for having content in multiple formats—print, web, and various types of online help—has risen. Despite this increased demand, some documentation groups now have fewer staff members to create all the content. Therefore, it is essential to have streamlined processes in place, and templates and structure boost authors' efficiency.

Templates and structure also provide the foundation for *single sourcing*, which is the process of using one set of files to create:

- Different versions of content—for example, user guides for two printers that have some common features. Both printers have the same paper feeder, so an author uses the content about the feeder in both manuals (and doesn't write separate versions for each guide). Only one of the printers has a memory card reader, so the author includes that information in just one of the guides.
- Multiple types of output—for example, printed books, online help, and web pages. Authors don't maintain separate source files for each type of output.

You can create different versions of content *and* multiple output types; the two aren't mutually exclusive. In short, single sourcing means writing content once and using it in multiple places.

Single sourcing requires that authors create content in a consistent manner—and following a template or structure ensures that consistency. The specifics of single-sourcing processes vary depending on the tools authors use to develop content and create final output, but consistent source files are the foundation of any single sourcing process.[1] You'll read more about tools

[1] It's also worth noting that consistent content looks more professional, is easier for readers to comprehend, and simplifies translation into other languages.

and approaches for single sourcing in "File conversion and single-sourcing utilities" on page 67 and "Structured authoring and single sourcing" on page 233.

Single sourcing is now considered standard practice for the technical writing industry, so you will likely do some form of it as you start your writing career. When we released the first two editions of *Technical Writing 101*, companies were still considering and just starting to adopt single sourcing, so we included a chapter on evaluating and implementing single-sourcing workflows. Now that single sourcing is standard practice, that chapter is outdated, so we have removed it from this edition. For more information about single sourcing and how documentation groups implemented it, download the chapter from the second edition at: www.scriptorium.com/singlesourcing.pdf

Templates and structure are good for your professional well-being

There's an indirect benefit to following your department's template or structure—it gives you more job security. If you don't follow the standards, you're creating an obstacle to your department's efficiency and therefore make yourself more expendable. Also, prospective employers like to know that a technical writer understands the importance of applying templates and following structure.

3 Very necessary evils—doc plans and outlines

What’s in this chapter

- ❖ What’s a doc plan?
- ❖ Outlining—it’s not just for high school papers anymore

You’ve got the job, and you’re ready to start a new project. All you need is some information about the product, and you’re ready to start writing—writing a documentation plan and an outline, that is.

You probably want to start writing the content instead of spending time on plans and outlines, but up-front planning and analysis are critical to good content. Formulating doc plans and outlines is the best way to complete this analysis. The time you spend planning your content is an investment; you’ll find that good planning saves you a lot of time later in the project.

In other words, you can plan at the beginning, or pay for it when you flounder in the middle of the project.

NOTE: If you're using complex content development strategies, such as single sourcing or database publishing, planning is *not* optional.

To create a documentation plan and outline for each deliverable, you need some information about the product. Most often, you get access to a prototype of the product or a technical specification ("spec") that lists the product's features. Although early prototypes and specs *never* give you all the information you need (and frequently are inaccurate because the product has changed), they usually provide enough information to start the project. Chapter 5, "Getting information," explains how to round up information.

Clients aren't always where you expect them

A client is your customer. If you're a freelance writer or if your company is creating content for another company, the identity of the client is obvious. But as a staff writer in a software company, your client is probably inside the company. It may be your documentation manager, or it may be the vice president of development in a small company.

However, your ultimate customers are always the same—the readers of your content. Don't forget that your goal is to deliver information to them.

What's a doc plan?

A documentation plan describes all the components of a content development project. For a sample documentation plan, refer to Appendix C. A doc plan should contain information about the following:

- Product description—a brief summary of what the product does. This might only be a sentence or two: "The FoogleGarber software lets users ensure that their privacy is maintained as they surf the web. FoogleGarber rejects all cookies and other identifying characteristics requested by a web server."
- Audience—characteristics of the product's users; in some cases, there may be more than one type of user. You should provide as much information as you can about each type of user. This should include their education level, demographic information, level of technological expertise, and the features that each type of user will use. The more you know about your audience, the easier it will be to write content that meets their requirements. See "Audience, audience, audience" on page 90 for more information.
- Deliverables—the names of the documents you will create, a brief description of each, and how they will be delivered (printed manuals, online help, HTML, and so on). This list should also include any rich media content (such as videos) you plan to deliver. (For more information on rich media, see "Using rich media content" on page 141.)
- Receivables—what the writers and other documentation team members need from others (such as the latest version of the product, access to the developers to get questions answered, existing content, and a template from the template designer).

- Style—what style guidelines will be followed (such as in-house style guidelines and *The Chicago Manual of Style*).
- Tasks—list of the actions required to complete the project and who is responsible for them. For example, the documentation team will handle information gathering, writing, editing, production editing, and indexing. The product developers will provide technical assistance and ensure the accuracy of content.
- Tools—what tools (such as desktop publishing and graphics software) the documentation team will use to create the content. Chapter 4, "The Tech Writer's Toolbox," provides more information about tools.
- Schedules—a schedule for each deliverable (see "Any formulas for writing doc plans?" on page 52 for more information about estimating dates).
  A schedule should also include dates for editing, illustration, indexing, and preparation for final output (print, online, or both).

Sometimes, the doc plan itself is a deliverable, particularly if you're a freelance or independent writer. Even if you're not required to deliver the doc plan to your client or your internal customers (such as the development team), it's a good idea to let them see the doc plan to ensure that everyone understands what they need to do to keep the deliverables on schedule.

Doc plans for external clients

If you are producing a documentation plan for an external client, the plan should also include the following items:

- Copyright—who will own the copyright of the completed content. It’s critical to spell out whether and how the copyright is transferred from the writer to the client company. If you’re in business on your own, get advice from an attorney on how to word this section of your proposal or documentation plan.
- Cost—a breakdown of the cost for each deliverable (see “Any formulas for writing doc plans?” on page 52 for more information). Cost may be a fixed price or an hourly rate.
- Disclaimer—All doc plans, contracts, and legal paperwork exchanged with a client should state that technical accuracy is the responsibility of the client or developer. For example, you could add the following text to your doc plan:

  “*Technical Communication Company* will make a reasonable effort to ensure the technical accuracy and completeness of the materials delivered to *Client*. However, *Client* is ultimately responsible for the content of the content. *Client* will review content for accuracy and completeness, and identify required changes to *Technical Communication Company* before the final delivery.”
- Terms—payment schedule, cancellation policy, and other legal information.

Who writes the doc plan?

In a group of writers or a documentation department, a senior writer or manager usually writes the documentation plan.

However, if you’re the only writer at a company, you’re it. Take the time to write a doc plan—even if it’s an informal document you don’t show to anyone else. As you create the plan, you’re forced to think about schedules, audience, and so on. Performing this analysis early in the project helps prevent nasty surprises later.

If your plan requires commitments from other people, be sure to consult them before you publish your schedule.

Have your doc plan edited if you plan to turn it over to an external client—particularly when it's part of a proposal. A grammatical error or misspelling in a doc plan isn't going to improve your chances of getting the project, and it makes you look bad.

Any formulas for writing doc plans?

A doc plan contains lots of numbers (page counts, schedules, and so on), so you might wonder whether there are any formulas to help you calculate those numbers.

Some reasonable estimates are as follows:

- 8 hours per completed page
- 8 hours per online help topic
- 40 hours per one hour of instruction time

A "completed" page includes all components of the writing process:

- Writing
- Editing
- Indexing
- Template design and document formatting
- Production

In your particular environment, you may find these numbers too high or too low.

To calculate page count, you might estimate that each task a user performs requires approximately two pages of content. Then, estimate how many tasks you need to document and throw in a few extra pages for the front matter and back matter. Creating these estimates is a bit of a black art, but this shouldn’t keep you from creating estimates. At the end of every project, be sure to look back at the estimate. If the estimated numbers don’t match the actual numbers, you need to identify why the estimate was inaccurate. As you complete more projects, you’ll gradually develop a feel for how much time a project really takes and be able to improve the accuracy of your estimates.

Even experienced writers cannot pinpoint exact page counts or days of work with certainty. Instead, use your estimates as an indicator of what to expect. If you think that some factor, such as an extremely complicated product or a large number of changes, will increase the time required, be sure to account for that in your estimate. It’s usually better to overestimate a little than to underestimate. The old cliché “underpromise and overdeliver” applies here.

If you have a deliverable on a very aggressive schedule, putting together your plan will give you an idea of just exactly how much trouble you’re in. Your plan can give you the ammunition you need to convince your manager that you need some temporary help.

Outlining—it's not just for high school papers anymore

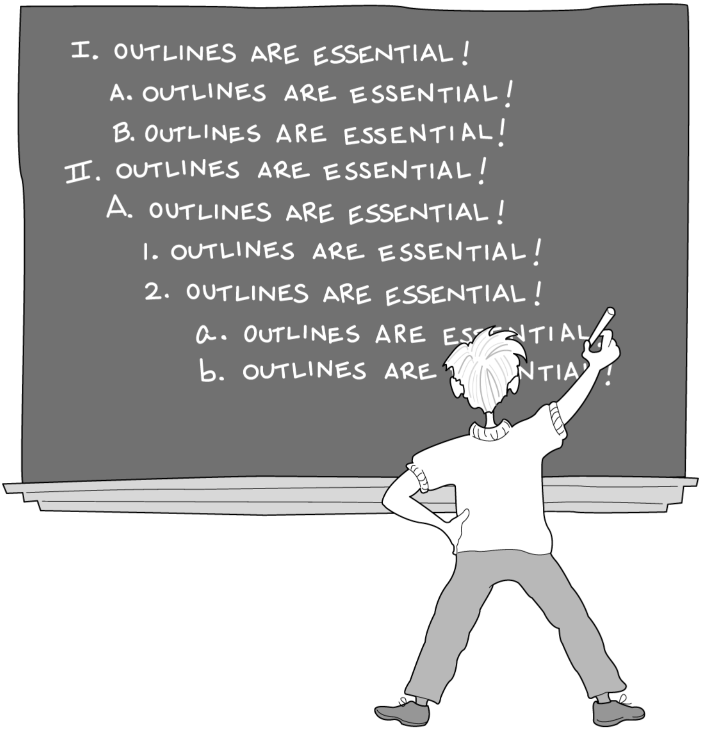
You probably have bad memories of being forced to create outlines for term papers in high school. Well, those outline-writing exercises are about to pay off—writing an outline for content is a very important step in the technical documentation process. Outlining requires you to think about what needs to be covered, and that sort of early analysis is essential for providing good information.

In high school or college, you were probably quite capable of writing a 5- or 10-page paper without an outline and thought that your instructor was being difficult by requiring an outline. In reality, you did create an outline even if you weren't made to turn it in; the outline was in your head. But now we're talking about writing manuals with hundreds of pages and online help with hundreds of topics. You will *not* be able to keep the outline for all of that in your head.

Creating an outline will help you break down the manual into manageable chunks. Let's face it: beginning a big documentation project is intimidating. If you create an outline, you can focus on just one section of content at a time and pretend that the rest of it doesn't exist.

What goes into the outline?

To write the outline, you need to know what the users need to accomplish with the product. Your best bet is to play with the prototype, if one is available. You should also review the spec (if it exists) and read every bit of information you can get your hands on. This might include product design documents, marketing plans, business plans, and interface design documents. User content for related products is also useful for gathering ideas about what your document should contain. See Chapter 5, "Getting information," for details about information sources.


I. OUTLINES ARE ESSENTIAL!
A. OUTLINES ARE ESSENTIAL!
B. OUTLINES ARE ESSENTIAL!
II. OUTLINES ARE ESSENTIAL!
A. OUTLINES ARE ESSENTIAL!
1. OUTLINES ARE ESSENTIAL!
2. OUTLINES ARE ESSENTIAL!
a. OUTLINES ARE ESSENTIAL
b. OUTLINES ARE ESSENTIAL!

Based on your research and the information you find in the existing documents, figure out what the users do with the product. For example, if the product you're documenting is graphics software, users often convert graphics from one file type to another; if the product is a printer, users will want to know how to print on envelopes. This is where your ignorance (or ability to fake ignorance) of a product is really useful. As a new user, you need to figure out how to complete certain tasks. Eventually, you'll write down what you discover for other users. Use the mindset of a new user while you figure out what needs to be documented.

Once you have a list of tasks, you'll need to organize them into an outline—or outlines, if you think that the tasks would be better suited to more than one deliverable.

How many deliverables should there be?

Here are some things to consider while determining how many different deliverables there might be:

- *Do you have different audiences with different information requirements?* For example, it's common to deliver a book or online help for regular users and another version for system administrators. Providing two sets of content separates out the day-to-day tasks from the high-level tasks that only users with special rights on the system can perform.
- *How much information do you have?* If you're looking at delivering about 100 pages of information, you can probably put everything in a single deliverable even if you do have multiple audiences. However, if you have a total of 1,000 pages of content for a single product, you may want to separate the material into several deliverables. Dividing up the information makes it easier for users to find what they're looking for.

- *Do you need to deliver information in different media?* One common approach is to deliver interface help ("The Add button does this.") or task-based information in the online help but to provide conceptual details in the printed documentation.
- *Are there delivery issues that you should consider?* For example, if most of the content is placed on the user's system as part of the software installation process, you should always provide printed installation information. Otherwise, end users are caught in a Catch-22—if the installer doesn't work, they can't check the content because it hasn't been installed yet!

Writing the outline

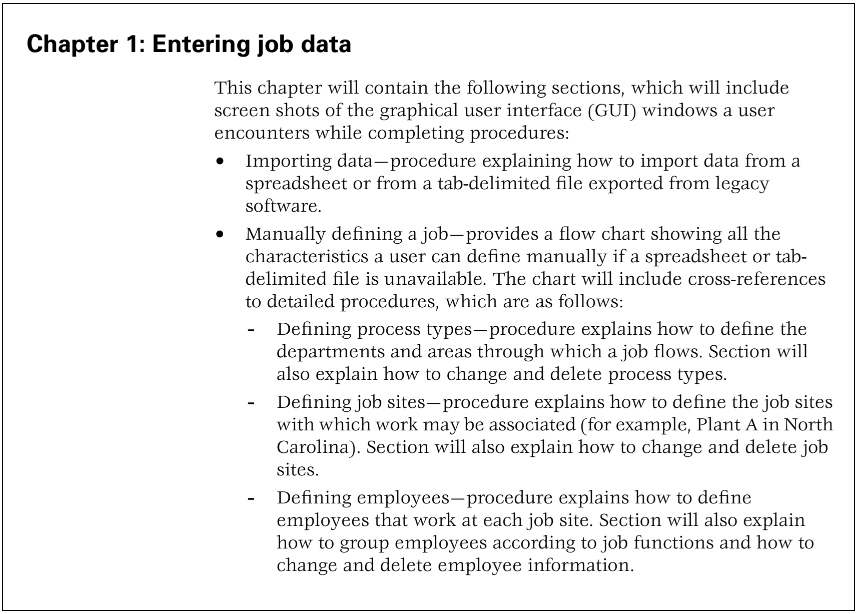
Some word processing programs (Microsoft Word, for example) have outline functions that can handle most of the formatting for you.

Instead of using a rigid outlining structure with Roman numerals, consider using headings, bullets, and sub-bullets, as Figure 3 on page 58 shows.

There's no single way to develop the outline. Do whatever makes sense to you.

Your outlines can be part of your documentation plan or stand-alone documents, but no matter what, be sure to get the outlines approved. Your manager or a more experienced writer can help identify any items you may have forgotten. If you're freelancing, be sure to get your client's approval of your outlines; this will ensure that your client understands where you're going with the content early in the process.

Figure 3: *Outlines don't have to be formal*



Chapter 1: Entering job data

This chapter will contain the following sections, which will include screen shots of the graphical user interface (GUI) windows a user encounters while completing procedures:

- Importing data—procedure explaining how to import data from a spreadsheet or from a tab-delimited file exported from legacy software.
- Manually defining a job—provides a flow chart showing all the characteristics a user can define manually if a spreadsheet or tab-delimited file is unavailable. The chart will include cross-references to detailed procedures, which are as follows:
    - Defining process types—procedure explains how to define the departments and areas through which a job flows. Section will also explain how to change and delete process types.
    - Defining job sites—procedure explains how to define the job sites with which work may be associated (for example, Plant A in North Carolina). Section will also explain how to change and delete job sites.
    - Defining employees—procedure explains how to define employees that work at each job site. Section will also explain how to group employees according to job functions and how to change and delete employee information.

NOTE: For deliverables that include audio and video, write a script or draw out a storyboard instead of creating an outline. For more information on rich media content, see "Using rich media content" on page 141.

Collaborating on outlines

If multiple people need to offer input on an outline, consider setting up an internal wiki; a *wiki* is a web site that encourages collaborative authoring by allowing visitors to modify content.

With an internal wiki, you limit access to those within your company or department. You could post a draft of an outline, and then your coworkers can contribute their ideas to it (and to any other information you post on the wiki). The wiki shows you when changes were made and who made them. Wikis also offer calendars and other functions that can be useful for planning and tracking content development efforts.

There are many free wiki technologies available (including TikiWiki at info.tikiwiki.org), but you will likely need assistance from a network administrator to set it up for internal use.

For more information on wikis and other collaborative technologies, see Chapter 14, “Web 2.0 and technical communication.”

4 The Tech Writer’s Toolbox

What’s in this chapter

- Content/text development tools for printed content
- Graphics software and clip art packages
- Rich media tools
- Help or web authoring tools
- File conversion and single-sourcing utilities
- Other helpful software
- Computers and ergonomics

Now that you have a doc plan and an outline, there is one last thing you need to consider before you dive into writing content: do you have all the tools you need to complete the job?

This chapter explains the software you need in your Tech Writer’s Toolbox.

NOTE: Some of the programs listed in this chapter are open source programs you can download for free. See page 242 for information on open source technology.

Content/text development tools for printed content

The most important tool in your toolbox is the one you use to write the text. There are many word processing, document processing, and desktop publishing packages that support template-based authoring, and each one has its advantages and disadvantages.

NOTE: You may not use a document processing tool if your department creates only online content. Instead, you may use just a help or web authoring tool; see page 67 for more information.

If you're working in a corporate environment, you probably won't get to choose the tool for developing content. You'll use whatever your manager hands you—and like it (maybe).

Many technical writers use a basic word processing program such as Microsoft Word. Although Word is adequate for short business documents, it is not designed for long, complex documents, and it often becomes unstable when you try to maintain very lengthy documents.

For writing documents that will be printed, you need a tool that can:

- Handle many embedded graphics
- Number steps and figure captions automatically
- Create complex tables
- Maintain cross-references
- Generate indexes and tables of contents

This list also applies to content in Portable Document Format (PDF).[1] A PDF file maintains the formatting of a printed document, but it gives you the benefits of online content, such as hypertext cross-references.

A good choice for printed and PDF documents is FrameMaker software. FrameMaker is specifically for creating and maintaining long technical documents.

[1] More information on the PDF format: www.adobe.com/products/acrobat/

Taking advantage of FrameMaker's powerful document processing features can greatly improve your productivity by reducing the amount of time you spend on tedious document maintenance.

If you plan to create different kinds of output from your document (for example, online help and HTML), keep in mind that you'll need to find an output path from your source document format to the other formats. See "File conversion and single-sourcing utilities" on page 67 for details.

If you're producing graphics-intensive shorter pieces (such as a newsletter), you might consider one of the desktop publishing packages, such as QuarkXPress or InDesign. These packages are less oriented toward book production but are better than FrameMaker at producing full-color, highly designed documents. If your content needs to look like a very expensive annual report, these tools may be right for you.

Contemplate that template

Templates aren't just for document processing. They are also common with other applications technical writers use: graphics programs, help authoring tools, and file conversion utilities. Regardless of what kind of program it's for, a template generally has the same purpose—to ensure consistency in presentation, which is an essential component of good technical communication.

If you are working in a structured authoring environment, you may use a tool for authoring in Extensible Markup Language (XML) instead of using a document processing or desktop publishing application. XML is a specification for storing structured content as text. See Chapter 13, "Structured authoring with XML," for more information. XML development tools include XMetaL, Arbortext Editor, In.Vision, and Oxygen.

Graphics software and clip art packages

To create the screen shots, illustrations, and other graphical elements in your content, you need several kinds of graphics software:

- Screen capturing—to take snapshots of items on your computer screen. If you're writing about software, you'll need pictures of the software, and screen capture software does that quite well. See "Displaying information from your computer screen" on page 136 for information about taking screen shots.
- Drawing—to draw shapes (particularly for flowcharts and technical illustrations).
- Graphics processing/editing—to change a file's format, to touch up and crop graphics, and so on.

Some software packages (such as Paint Shop Pro) can handle both screen shots and graphics processing. However, most drawing tools don't provide graphics processing and vice versa.

HiJaak, Paint Shop Pro, and SnagIt are popular for taking screen shots on Windows machines. For the Mac, Snapz Pro is available. For UNIX, try xv.

NOTE: Most operating systems have built-in (but basic) tools for screen shots.

If you need to create flowcharts, consider Visio or OpenOffice.org Draw. For drawing line art (for example, a detailed view of hardware), you need a package such as Illustrator. For graphics processing or editing, use a tool such as Photoshop or Paint Shop Pro. ("Using graphics in content" on page 129 has more information about using still images in content.)

A clip art package can be helpful when you need icons in your text (such as a stop sign to flag warnings). If you don't have much drawing ability or access to a graphic designer's services, well-chosen clip art can prevent quite a few headaches. Keep in mind that you want a high-quality selection of clip art and that some clip art packages have restrictions on the number of times you can use an image and where (for example, some permit you to use the images on the web only). In addition to buying packages of clip art, you can purchase individual files from web sites such as www.istockphoto.com.

Rich media tools

Now that more and more content is being delivered online, you can go beyond standard images by including videos and animations (often referred to as *rich media*). Tools such as Captivate and Camtasia record what happens on screen, and you can use the software to add narration to what is shown in a clip.

Flash enables the creation of animations and interactive content, including tutorials, presentations, and games. (The player for showing Flash files is free.) If you see an online ad that includes action and sound, there is a very good chance the content was created with Flash.

"Using rich media content" on page 141 has more information about rich media.

Help or web authoring tools

If you're developing online help or web-based materials, you'll need a tool to write the help or HTML. For online help, you can use tools such as RoboHelp and Flare, and to write HTML, you can use tools such as Dreamweaver.

If you're going to create documents for more than one type of output (for example, print and HTML), your best bet is to use a text development tool that has conversion capabilities (or that's compatible with a third-party conversion tool, as explained in the next section).

File conversion and single-sourcing utilities

There are many third-party tools that convert word-processing files to other formats, including online help, HTML, HTML Help, JavaHelp, and XML. Converting the material intended for print means that you don't have to spend time re-creating a different type of output in another authoring tool. Conversion tools include ePublisher and MIF2GO. RoboHelp and Flare also have conversion capabilities.

To convert documents to PDF format, you'll need Acrobat or another tool that converts files to PDF format. The Acrobat Reader, which lets users view PDF files, is available for free.

The ability to create multiple types of output from one set of files is called single sourcing, as mentioned in "Are templates and structure really that important?" on page 44. Most documentation departments have instituted some form of single sourcing. Some companies convert their word processing files with third-party tools to PDF format and online help. Other companies have more complex single-sourcing environments in which authors write small chunks of information that are later combined and transformed into print/PDF and online help deliverables. (You'll read more about such an environment in "Structured authoring with DITA" on page 234.)

This software is hard to use! Help!

Some of the tools listed in this chapter—particularly for text development, graphics, and file conversion—are not applications you can learn quickly (or even easily). Getting formal training is the best option, but it can be expensive. Many software manufacturers' web sites list companies that train people how to use their software. Your local chapter of the Society for Technical Communication (STC) may list (and even sponsor) training in your area. Also, check with your local community college about software-related continuing education classes. If you're lucky, your employer may reimburse you for the cost of training.

If training is not a possibility for you, consider getting a third-party reference or a workbook, and visit web-based discussion forums, particularly those at a tool's corporate site. Low-cost or free online tutorials, such as the free FrameMaker workbooks at wiki.scriptorium.com, are also available.

Other helpful software

There are some other software packages you'll need as a technical writer. Some of these applications are open source tools you can download for free. (For information about open source tools, see "Free but not cheap" on page 242.)

These tools, which aren't limited to the technical writing profession, include the following:

- Compression utility—Files can become quite large, particularly if they include graphics, so it's a good idea to have a tool such as 7-Zip (PC) or StuffIt (Mac) that compresses files before you send them to a coworker or the client. Compression is essential when sending large files via email.
- Communication software—You'll need a tool for email (such as Outlook or Thunderbird) and file transfer protocol (FTP). Sometimes, files are too big to send through email as attachments. FTP software lets you transfer the files over the Internet without using email. (Web browsers have some FTP capabilities, but the dedicated FTP clients are more fully featured.) A good rule of thumb is not to send an email attachment larger than two megabytes (MB). Don't forget to use your compression utility for larger files sent by FTP, too.

  On the PC, you can use commands at an MS-DOS prompt to send files via FTP, but most people prefer using software with a user interface, such as FileZilla.

  You can also use online file transfer services such as www.yousendit.com.

- Project management/time-tracking tool—Having software that tracks your schedule can be helpful, particularly when several people are working on a project. Microsoft Project is a widely used project-tracking tool, and there are also web-based programs such as Basecamp (www.basecamphq.com) and LiquidPlanner (www.liquidplanner.com).

  You can also use a web-based calendar that all team members access. That way, everyone can see when deliverables are due. You can create such a calendar for free at web sites such as Yahoo (www.yahoo.com) and Google (www.google.com). Even if you are the only person on a project, laying out your schedule can be very useful.

- Encryption software—If you're sending confidential information over the Internet, consider using encryption software such as Pretty Good Privacy (www.pgp.com) or GNU Privacy Guard (www.gnupg.org) for your email. Keep in mind that both the sender and the recipient will need the software.

NOTE: In your company, you may be required to use specific tools for email, project management, and so on. Your company may also have specific configurations for the tools.

Computers and ergonomics

Most newer computers have more than enough processing capability and memory to run the programs you'll use. If you're using an older computer, though, it's a good idea to check the system requirements of applications to ensure your computer meets the specifications.

You're going to spend a lot of time looking at text on your monitor, so get the biggest one you can afford (or talk your boss into buying a big one). At an absolute minimum, your monitor should be 17 inches. This will help minimize scrolling through your pages. Also consider using two smaller monitors instead of one large monitor; many systems today can support dual monitors.

Because technical writers do spend so much time in front of the computer, it's important to have an ergonomic work space to prevent repetitive motion injuries (including carpal tunnel syndrome) and other problems (such as eyestrain).

Some things to consider about your work environment and how you work in it include the following:

- Height and position of your chair and keyboard
- Posture
- Lighting

For resources about ergonomics, see "Ergonomics" on page 301.

5 Getting information

What's in this chapter

- Technical specifications and other development content
- Prototypes and software under development
- Legacy content
- Developers and subject matter experts
- Interviews with users

Your primary task is to give people the information they need to use technology. Ferreting out that information can be the most difficult aspect of technical writing. For this reason, technical communicators are sometimes called "information developers"—they develop useful information from various obscure sources.

This chapter explains several sources from which you can extract information.

NOTE: In general, you need some information about the product to write a doc plan and outlines for content, so you may have access to at least one source very early in the project. As work progresses on the project, you will probably use most of these sources.

Technical specifications and other development content

A technical or functional specification—known as a *spec*—is a document written by a product's developers. It explains the product's purpose and how it works. Typical information in a spec includes:

- Lists of menus and menu choices (software)
- Mock-ups of the interface (software)
- Illustrations of components (hardware)
- Lists of features or proposed features
- Information about how the product processes data (for example, how a connection to a database works)
- Information about the product's components
- Changes from the previous release
- Schedules for the product's release

The benefits of a spec

A good spec can provide you with an excellent overview of the product and answer many basic questions about the product. When the spec answers your questions, you don't have to track down a developer or SME to get your information, and that makes everybody happier.

The drawbacks of a spec

Most specs don't contain all the information you need, are inaccurate, and aren't updated to reflect a product's ever-changing functionality. Often, the spec doesn't exist at all.

In short, a *really* good spec is a mythical creature. If you ever see one, please contact your nearest tabloid—you could probably make a lot of money selling the story about your encounter.

Organizing menu information: the spec does not always know best

Because a software spec often lists every menu and menu choice in the order they occur on the interface, you may be tempted to write your document in a similar fashion—but don't! Instead, focus on the tasks a user performs and the order in which the user performs them. This may or may not match how the interface is organized.

For example, many software programs have a **Print** menu choice on the **File** menu. If you wrote the content according to the **Print** choice's order on the interface, information about printing would come early because it's on the first menu, **File.** In reality, however, printing may be one of the less important tasks a user performs, so it may not need such prominence in your content.

Prototypes and software under development

The term *prototype* is used a little differently in hardware and software. A hardware prototype often does not have all the working parts; it may consist of the product's "shell" without all the internal components. A software prototype is usually a "proof of concept"; the software performs more or less the functions that the final software will perform, but the interface is probably not what will appear in the final product.

As a writer, you will find hardware prototypes very useful because you can often see how the product will look and can infer what will happen "under the covers" from the design. But software prototypes can be a problem because the content must be very specific about how to manipulate the software's interface. In fact, most users (if they've ever thought about it) equate the interface with the product. The software developer will tell you, "Oh, the functionality is the same; we just tweaked the GUI."[1] But your end users don't care about the functionality; they care about how things look. So, a "tweak" in the interface can result in catastrophic rewriting requirements for the content.

Slightly better than prototype software is prerelease software. Usually, the software goes through a few stages, as shown in Table 1:

Table 1: *Software release cycle*

Development stage	Status
Alpha	Software works, sort of. Some major features are missing or not working. Expect alpha software to crash your system at regular intervals and possibly corrupt it. If at all possible, keep alpha software and the source files for your content on separate computers.
Beta	Software works, mostly. All major features are sort of working. This version is often made available to customers for testing. Expect beta software to work with minor glitches. It may crash occasionally, but you should be able to identify what causes the crashes.

[1] GUI (pronounced "gooey")—graphical user interface. Otherwise known as the windows that the user sees on screen.

***Table 1:** Software release cycle (continued)*

Development stage	Status
Release Candidate (RC)	Software is done—for the most part. All features are working and major bugs have been cleaned up. It's made available to more customers for testing. RC software should be stable with no crashes.
General Availability (GA)	Software is done and shipped to customers.

Many companies have additional components in their release cycle. Products may go through several alpha versions (Alpha 1, Alpha 2, Alpha 3) before moving on to beta status. Often, Release Candidate status requires a freeze on interface changes and no code changes except to correct bugs. In other words, RC status means that no more features will be added.

The distinction between beta and GA software is becoming fuzzier. Many companies now distribute software online and include automatic updating capability in their applications.

Of course, these release cycles are only guidelines, and they are for the most part breached more than they are followed. For example, it's common for companies to release software that's not quite ready to hit a particular shipping deadline. (Most often, this release coincides with the end of a quarter and has various implications for what's known as "booking revenue.")

The benefits of prototypes and prerelease software

Early "drafts" of the product in the form of prototypes and alpha or beta software are an essential source of information. If you're documenting software, you must have a copy of the program to write and test your procedures (and to take screen shots for your document). The same applies to hardware prototypes.

The drawbacks of prototypes and prerelease software

Like a spec or any other source of information, if the prototype or software you're using is not the latest version, the information you're writing could very well be inaccurate. Be sure that when the development team makes changes, you get the new version or are at a minimum notified of the changes.

NOTE: A good way of keeping track of changes is to get access to the developer's bug tracking system, which can provide invaluable information about what's being changed. At some companies, writers can add bugs they find to the tracking system.

If you are taking screen shots of a software interface or are creating hardware diagrams, it is especially important that you wait a bit for the product to stabilize and then take the screen shots or draw the illustrations.[1] Keeping in close contact with the product developers can help you figure out when it's safe (well, safer) to create graphics. Inevitably, you will have to retake screen shots or update drawings due to last-minute changes. But you can minimize the amount of rework you'll have to do by waiting as long as possible before creating illustrations or taking screen shots. Instead, insert placeholders that explain what a drawing or screen shot will show.

[1] Hardware diagrams often come from computer-aided design (CAD) programs, which create detailed, exact renditions of components.

TOP SECRET

Many products are considered confidential while they're under development. Companies don't want their competitors to hear about the new product or its features.

If you are working on a product whose features (or existence) have not yet been announced, you can expect some security requirements. Generally, you should avoid discussing the specifics of what you're working on with anyone who is not involved in the project.

Many companies ask employees to sign a nondisclosure agreement (NDA). The NDA spells out your obligations to keep information confidential and to safeguard proprietary information. Most NDAs are straightforward, but read each one. In some cases, a company will put intellectual property information into their NDA. The intellectual property and copyright agreements should be separate from the NDA, which should focus only on how you must handle confidential information that a company divulges to you.

If you're working on classified government information, expect much more stringent requirements. But nothing could top the company that required its technical writers to hide a prototype printer when they weren't working on the documentation. The writers were required to cover the prototype with a large box whenever they left the room so that no one could look in and see the printer. In addition, the development had to be done in a room with no windows, and the door had to be locked when the writers were not in the room. This is a true story. Really!

Small software companies are especially notorious for changing things up to the very last minute. If you're working at one of these companies, try to educate your managers about why these changes will cause problems for your content development efforts. You can also develop a process that accommodates those changes with iterative releases and content that is auto-updated along with the software.

Of course, a changing product can also cause difficulties for your text. One good approach is to ask the developers which parts of the product are more stable so that you can write about those first. Then, as other parts are completed, you can document them. This can save you from having to rewrite a topic several times as the product changes.

Legacy content

If you're documenting a new version of an existing product, there's a good chance that content already exists for the previous version. In some cases, you can use the existing content by updating the text to reflect changes and added functions.

The benefits of legacy content

Legacy content can help when you're drawing up doc plans for new content—you can ask your clients and the content's readers what they like (and don't like!) in the existing docs and then use that feedback to improve the content for the next version. If the content is well written, you can use it as a foundation for the new material and add, update, or delete information as necessary. Also, working with what's already been written can save a lot of time.

Reviewing content for similar products can also be helpful. For example, if you're documenting a printer that's in the same product line as one that's already on the market, you may be able to reuse some of the text from that product for the content you're developing.

The drawbacks of legacy content

Legacy content can be a problem when it's badly written. Reworking existing bad content into something useful takes as long as writing quality content from scratch. If the product has changed significantly since the last release, the legacy content may not have enough relevant information to make reuse worthwhile.

You'll run into another problem with legacy docs if you're new in a job or when you pick up a new client. You may review the legacy content and discover that it's poorly written, disorganized, and not useful for readers—but your client or manager thinks that the existing information is great and just wants you to "make a few updates" for the next release.

You'll need your diplomatic skills to solve this type of problem. You could try rewriting a brief section and explain the improvements you've made.

The best way to avoid this situation, though, is to ensure that you find out *before* you take the job what legacy content exists and what the client or manager's opinion of those documents is. Review the material before you accept the job and make sure that you and your potential employer agree on what needs to be done.

Developers and subject matter experts

The people who are developing a product—or the SMEs very familiar with a product—are the most important source of information a writer has. Not only do developers create the prototypes, but they also know which features work, how features should work when they don't, and what changes are on the horizon. Good communication between product developers and technical writers is essential to the success of a content project.

The benefits of developers and SMEs

Developers who promptly answer your questions and review content on schedule are your best allies on a content development project. Their review comments ensure that your content is technically accurate, and their knowledge of how the product is shaping up can cut the amount of time you spend rewriting material. If the developers warn you that particular features are going to change, you can hold off writing the material about those features until the developers tell you the features are stable.

The drawbacks of developers and SMEs

Sometimes developers are so busy working on the product, they feel they don't have the time to review document drafts. (In some extreme cases, developers may not place much value on content, so they don't bother to review it at all.) Reviews then run late, or they are very cursory and offer no real feedback. Sometimes, review comments focus on issues such as comma usage or capitalization instead of on the technical content.

To avoid problems with grammatical nitpicking, you should do three things:

- Ensure that the content drafts you deliver are grammatically correct and spell-checked and that they do not contain writing errors that will distract the reviewers.
- Remind the reviewers that you need their input on technical issues, not on grammar. If an editor or another writer will review the work for grammar and adherence to the style guide, tell technical

reviewers that they do not need to focus on those issues.

- Create a sign-off sheet and attach it to the draft, or use your department's electronic workflow tools for tracking draft content through the review process. Requiring reviewers to sign off on their reviews makes them more accountable (and can also provide you with some protection if technical errors aren't caught).

Getting a solid review from the developers is essential because only the developers know whether content is technically accurate. Without a thorough review by the development team or SMEs, your content will not be as useful to the product's users.

What are some of the ways you can ensure that developers give you the information you need? See the following sidebar, "(Almost) 30 ways to get information from developers."

(Almost) 30 ways to get information from developers

While many of the following suggestions are hardly serious, the basic message should be clear: establish a clear line of communication, be persistent but respectful, and remember that the developer is a person with interests and responsibilities outside of work.

1. Bring bagels and cream cheese to a morning review session.
2. Deliver content that requires only minimal changes from the developers. If the developers respect your work, they'll be more likely to deliver the information.
3. Figure out the developer's preferred method of communication and use it. If the developer prefers to talk at the water cooler, do it. If the developer prefers to be contacted by email, use email, even if his cube is across from yours.

(Almost) 30 ways to get information from developers

4. Be respectful of the developer’s time and other commitments. Try to group your questions instead of interrupting her constantly.
5. Consider using medieval torture devices, including the rack and the iron maiden.
6. Develop enormous expertise in your product. The product engineers are more likely to take your questions seriously if they think you know your stuff.
7. Restrict the questions you ask the developer to the really obscure stuff.
8. Brownies. And lots of them.
9. Attend development project meetings to stay on top of what’s going on with the project.
10. When working late, share your pizza.
11. Learn to read code and dig up information by scanning the code.
12. Make the developer’s life easy. When inserting queries for the developer in your document, use large, bold, and hard-to-miss type.
13. If you don’t want your writing style (and comma placement) criticized, don’t criticize the developer’s work.
14. Find out what the developer’s work schedule is and call (or visit) during those hours.
15. When reporting bugs to the developers, be thorough. Explain exactly what happens and how to reproduce it. Don’t make the developer figure it out.
16. Cold hard cash.
17. Save up questions and make a list to give to the developer instead of asking individual questions.
18. Deliver one or two chapters for review at a time instead of the entire document.
19. Split up reviews so that developers review only the material they are responsible for.
20. If all else fails, marry a developer. (This is not likely to help, though.)
21. Don’t ask stupid questions—do your research first.

(Almost) 30 ways to get information from developers

22. Don’t tell a developer that you could have designed a better interface in your sleep (even if it’s true).
23. Find out what the developer’s favorite snack is and just happen to have some during the next review meeting.
24. Schedule a standing weekly meeting to review questions.
25. Report bugs to the developer personally instead of the bug-tracking system—many developers are evaluated based on the numbers of bugs reported against their code.
26. Send your manager a note explaining the drop-dead date when you need review comments.
27. Copy your manager on correspondence to developers.
28. Copy the developer’s manager on requests for information.
29. Have your manager talk to the developer’s manager (a last-resort measure).

Interviews with users

You write content for a product's users, so it makes sense that they could provide you valuable information during the writing process. Unfortunately, this is by far the most difficult information to get.

The benefits of interviews with users

By talking to users of a product, you can get firsthand information about how the product is really used. Users can tell you what they stumbled on (so you can be sure your document clearly explains the issue), and they can show you the most frequently performed tasks (which can help you figure out in what order tasks should be described in the content). Sometimes, an interview with a user will prove that the document's outline does not reflect the tool's use in the real world, so you may need to revamp the structure of the outline (and maybe even your document, if you've already done a lot of writing) to reflect reality.

NOTE: It's also important to ask users how they find information about your company's products. Most users turn to search engines such as Google instead of going directly to company web sites, so your content strategies should take that into consideration. Knowing the terms and phrases users enter into search engines can help you (and your coworkers who maintain web content) refine the search-engine keywords associated with your company's online content.

The drawbacks of interviews with users

Tight budgets and aggressive schedules on content development projects often make user interviews impossible—interviews are expensive and time consuming, whether the interviews are done in person, with paper or online questionnaires, or over the phone. Even if you manage to get access to customers for the interviews, you sometimes end up with individuals who aren't the product's true users. You may end up talking to the person who supervises the employees using a product instead of the employees themselves. Feedback from such an individual can do more harm than good because the supervisor's opinion on how the product is used (or should be used) can be vastly different from the employees' day-to-day reality.

For new products, you may not know who the users will be, which makes it difficult to interview them.

Although technical communicators often act as user advocates by providing the information that the users want and need, there are cases where this might not give you the true picture. If you're documenting company policies and procedures in addition to how to use the product, you may need to talk with managers and supervisors to ensure that you get the appropriate information about corporate policy for your documents. This corporate information may conflict with how users would like to use the product.

6 Finally—it’s time to start writing

What’s in this chapter

- Audience, audience, audience
- Style and terminology
- Different types of content
- Topic-based writing and its benefits
- Dealing with the inevitable schedule changes
- Experience is the best teacher

Considering that “writer” is part of a technical writer’s title, it’s surprising how long it takes to get around to actually *writing* the content. Once you’ve created a documentation plan, drawn up outlines, and identified your information sources, you’re ready to get started.

So, how do you get started? Basically, you need to perform the same tasks the product’s user does, and while doing so, write down exactly what you do and what happens when you do it.

However, the content development process is not as simple as just writing down actions and results. You need to provide background information, such as a high-level explanation of what the product does and why, and information about why the user might want to perform a particular action. It's usually much more difficult to create that information than to write down tasks because the background or conceptual information requires that you truly understand the product and its uses.

While writing your material, you need to tailor the content to the audience, break down a task into discrete steps, and decide when to use graphics and tables. This analysis is an important part of writing good content.

This chapter explains the many points you should keep in mind while writing technical content.

Audience, audience, audience

The real estate agent's mantra is "Location, location, location"; your mantra should be "Audience, audience, audience." To create useful content, you must address your audience at the right level. For example, if you're documenting a product used by workers who have little prior computer experience, you may need to explain basic operating system procedures—for example, the difference between a single and double mouse click. However, if you're documenting a tool used by C++ programmers, an explanation about single- vs. double-click would not only be inappropriate, but it would also annoy experienced computer users. When content underestimates (or overestimates) its audience, users become frustrated with the information and begin to doubt its accuracy and even its usefulness. Such content is a failure.

TAKE MY PREFACE ... PLEASE! BUT SERIOUSLY, FOLKS,
YOU'VE BEEN A GREAT AUDIENCE...
USER GUIDE

Although it seems obvious that instructions on how to use the mouse aren't needed in programmer-level content, lack of attention to the target audience is a major problem in technical documents.

You’ve probably experienced this firsthand. Have you read a poorly written manual for a household appliances or an electronic device? There’s a very good chance the guide was a failure because it was unclear and because it overshot its audience’s knowledge or experience level. Manuals and online help should help people use products—not make them feel stupid. Writing to the correct audience will prevent a great deal of user frustration.

K.I.S.S.

The K.I.S.S. principle (Keep It Simple, Stupid) definitely applies to technical writing, but keep in mind that writing for your audience does not mean that you should be patronizing or condescending. Your readers will see right through this. You should also be careful about making assumptions about your audience based on demographics or educational level.

The rule of thumb for technical writing is that you should write at an eighth-grade level. You can use software to check your documents for complexity (some software will even evaluate the grade level for you), but here are some general guidelines on what “writing at an eighth-grade level” really means:

- Use clear declarative sentences.
- Avoid jargon, slang, and idioms.
- If you need to use complex terms or acronyms, explain them.
- Use headings, paragraphs, bullets, and steps to structure your writing into manageable chunks.

NOTE: In Chapter 12, "Avoiding international irritation," you'll find out that these and other guidelines in this chapter also make it easier to translate your documents.

Inclusive language

Using inclusive language is sometimes derided as "political correctness." It's true that some phrases can sound awkward ("his or her"), but it's possible to write a document that uses inclusive language without being obvious about it. For example, consider these choices:

1. The WinkleWart product helps the user manage his finances.
2. The WinkleWart product helps the user manage her finances.
3. The WinkleWart product helps the user manage his or her finances.
4. The WinkleWart product helps users manage their finances.
5. The WinkleWart product helps you manage your finances.

The first two options are not inclusive. The use of the so-called "generic he" in the first sentence could be construed as including women. However, it annoys many readers, so it's best to avoid it—even if it's technically grammatically defensible.

Option 2 might be acceptable if you're writing for an all-female audience. Option 3 is inclusive, but it's glaringly obvious you're trying to be inclusive. Consider the fourth and fifth options. Both of these options let you avoid the "he or she" construction gracefully.

In a few cases, it will be difficult to write around using "he" and "she." In such cases, you can alternate between the masculine and feminine pronouns.

NOTE: Common sense should prevail. If you're writing a document about pregnant women, "he or she" is probably not appropriate.

Occasionally, you'll be asked to document a product that is targeted toward a particular group—perhaps women or a particular ethnic or racial group. Take your audience into account, but don't insult your readers by assuming that sprinkling a few ethnic names throughout the document will, by itself, make your document appropriate for the audience.

Analyzing your audience without spending a fortune

You'll find entire books available on audience analysis; see "Audience and task analysis" on page 294 for resources. But in the real world, you rarely have the time (or money) to send out detailed user questionnaires, interview prospective users, or use other sophisticated techniques. There are, however, a number of questions you can ask to get a sense of your audience:

- *What is your audience's educational background?* If you're writing for college professors, you can make some assumptions about their level of literacy and education.
- *What is the demographic profile of your audience?* For example, where do they live, how old are they, and is English their primary language?

If you're writing for teenagers, don't assume that including references to the latest trends in popular culture will make your content more appealing to them. Pop culture changes quickly, so your references become dated, and the audience realizes you're trying too hard to appeal to them.

If your audience has limited English proficiency, it's a good idea to use graphics to convey your main points. (You might also consider translating the content.)

- *What is your audience's level of computer knowledge (or knowledge about the hardware)?* Professional computer users, such as programmers or system administrators, have different content requirements than computer novices.
- *How much do readers know about the subject matter?* If you're documenting how to use a legal database, the lawyers who use it are probably experts on the content in the database. They are, however, unlikely to be computer experts. In this case, you don't need to worry about explaining legal terms, but you do need to explain computer terms. If your product is a database of legal information for the general public, you cannot assume that the reader will have legal knowledge or computer knowledge.
- *How motivated is your reader?* Does the user want to use the product? If you are writing about a digital camera, you can probably assume that most of your readers are interested in using the camera and want to learn about it so that they can take good pictures. But what if you're writing about a database

application that's replacing a paper filing system in an office? The clerks who have been using the paper filing system for years know exactly how to use the old system. But now, they must learn a new, computer-based system, which may not be popular. In this case, your audience's motivation could be very low. The lower your audience's motivation, the easier (and shorter) your content should be. A more highly motivated reader is more willing to invest some time reading content to learn about the product.

- *What are the user's requirements?* What do the users need to do with the product? Do they need to know how to do everything or just a how to use a few important features? This question will help you assess whether your information should be strictly task-oriented (how to accomplish task X) or should include reference information (an exhaustive list of features, usually organized alphabetically or by some other scheme that doesn't address the order in which tasks are performed).
- *Do your readers have any physical characteristics or limitations that affect their reading ability?* These limitations will affect how you design and deliver the content. For example, older readers or readers with limited vision will not appreciate small, cramped fonts. Readers who are color blind will not notice that you've cleverly color-coded headings for them. If your audience includes people who are blind, you need to make sure that your content is comprehensible to someone who *hears* it or that it works in Braille.

Answering these questions will help you understand your audience and write content that better meets their requirements.

Accessibility standards

When creating content, always keep in mind that the information should be accessible to people with disabilities.

For example, if you are creating content for a web site, be sure to include alternate text that briefly explains each image; most web development tools make adding alternate text very easy. Screen reading software will read that alternate text to describe a graphic to a visually impaired person.

In 1998, Congress passed legislation that requires federal agencies to make their electronic content and information technology accessible. These standards, known as Section 508, apply to software, online information, telecommunication devices, and other products. Visit www.section508.gov for detailed information about Section 508 standards.

The law does not require that private companies comply with the Section 508 guidelines (unless those companies are providing services and tools for the government). Many companies institute Section 508 standards as best practices, or they implement other standards such as those of the Web Accessibility Initiative (www.w3.org/WAI).

While it seems like common sense to implement accessibility so all users (and potential customers) have access to a company's products and information, some large companies have been sued for having inaccessible web sites.

Style and terminology

Before you start writing, be sure to consult the style guidelines for your project. Most projects have editors that establish a project's guidelines—when to capitalize words, what terms to use when referencing parts of the product, and so on. If you learn the guidelines before you start writing, you can prevent (or at least minimize) the misery of fixing repeated stylistic errors in your text.

Chapter 9, "The importance of being edited," contains more information about style guidelines and the review process for documents.

Different types of content

You can divide up the content you need to create into several different categories. Again, you can find volumes of academic research on this topic, but here are some basic categories:

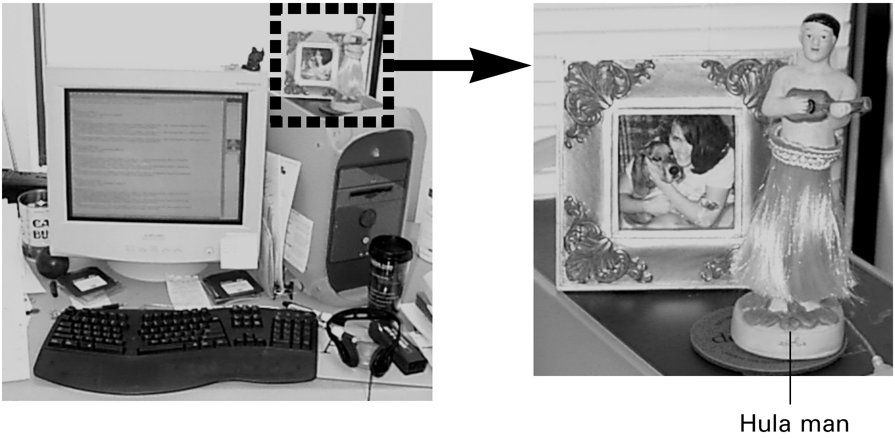
- Interface information
- Reference information
- Conceptual information
- Procedural information

The next few subsections describe each of these types of information and how you might handle them as you assemble content.

Interface information

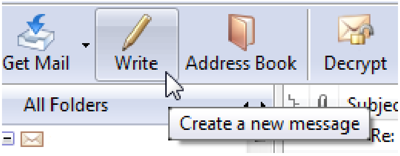
Interface information explains the function of a particular part on a product. For hardware, you'll often provide interface information as an illustration or a photograph. For example, you show a larger view of the product, and then zoom in on a particular part (as shown in Figure 4). You can further point out specific parts with *callouts,* which consist of lines and labels ("Hula man" in Figure 4).

Figure 4: *You can provide an overall view and then zoom in on the most important part of the picture in a separate image*



For software, you can provide similar labels electronically. Many applications include ToolTips, which are pop-up labels displayed when the user rests the cursor over a particular item for a few seconds (Figure 5).

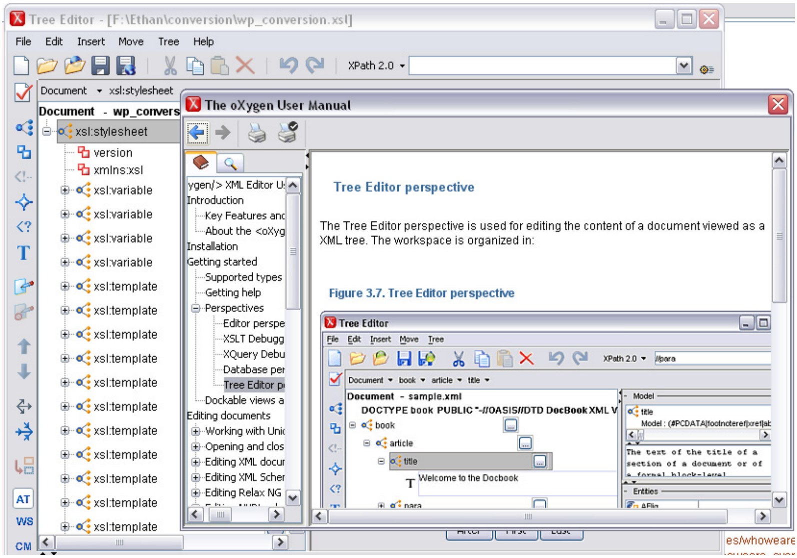
Figure 5: *ToolTip in Thunderbird email software*



If you're developing content for a software application, you can also provide context-sensitive help, which is triggered when the user selects an interface item and then a button or a key (such as F1). A small window with a sentence or two explaining a particular window or button is displayed, or the online help opens and displays the topic associated with the item.

As shown in Figure 6, context-sensitive help lets you provide more detail than ToolTips, which are normally limited to just a word or two.

Figure 6: *Context-sensitive help in oXygen XML editor displays the topic associated with the Tree Editor*



The Tree Editor perspective is used for editing the content of a document viewed as a XML tree. The workspace is organized in:

The text of the title of a section of a document or of a formal block-level

Some writers provide a list of all the interface items in a manual’s appendix. Consider whether the effort in assembling this information is worthwhile. It’s not likely that your readers will read an appendix just to see all the interface items explained. However, linking online interface descriptions to the items on the interface gives immediate assistance to users.

Building context-sensitive help can be technically challenging. It requires that you work closely with the software developers and use some complex tools.

NOTE: Many help authoring tools have context-sensitive help capabilities built in, but if you're working in an authoring environment based on the Darwin Information Architecture (DITA), your process will probably not accommodate building context-sensitive help. For more information about DITA, see "Structured authoring with DITA" on page 234.

Reference information

Reference information is data that readers need to look up (or "reference"). A dictionary is a great example of reference information. A standard dictionary provides an enormous amount of information about words, but it doesn't tell you anything about how to write sentences. So, reference information assumes that the reader knows what to do with the information; it provides content but no instructions. In technical writing, reference information includes glossaries and lists of functions for a programming language.

Writing reference information is not particularly difficult (although actually acquiring the information can be). Because the information is usually highly structured and organized in a way that's not up for debate (for example, alphabetically), you can create a skeleton for each reference item and then just fill in the required information.

The advantages you get from online formats, such as clickable indexes, full-text search, and hypertext links, are particularly useful when applied to reference information. In a book, you can provide information alphabetically and give readers indexes and cross-references, but online they can click on an item to see the related information. Consider making reference information available online to make searching faster and more powerful.

When writing reference information, keep in mind that users reading it are typically looking for a single piece of information. For example, they want to know how to use a certain function and nothing else. Giving them long narrative text is probably not going to be popular.

Conceptual information

Conceptual information is by far the most difficult information that you'll be asked to write. When you write conceptual information, you provide the "why" behind a feature. Reference information tells you what a function is, and procedural information tells how to use the function. But conceptual information explains under what circumstances feature A is a better choice than feature B. Typically, the introductory chapter in a book is conceptual; it discusses the product and explains what you can do with the product. In some cases, you might also be asked to explain why your product is better than the competition's.

It's difficult to write conceptual information because it requires an understanding beyond what you can learn from looking at a product's interface. For example, based on things you see inside a car, you can't provide the warning that excessive speed is a bad idea—the relationship between speeding and being pulled over by the police is not apparent from looking at a car's dashboard.

Procedural information

Procedural information consists of steps that tell a user how to perform a task. Because most content has a great deal of information about how to use a product, you'll probably spend a lot of time writing procedural information, also called task-oriented information.

The next chapter, "Writing task-oriented information," offers many pointers on how to write effective procedures.

NOTE: Not every product will have task-oriented and reference information. Some products often have just a task-oriented user guide that tells the user how to install and use the product. There may be no need for a programmer-level reference.

Topic-based writing and its benefits

Some companies write content as *topics*: small chunks of content that are combined to create a deliverable. Often, topics (also known as modules) are divided according to the type of information they present: procedural information for a particular feature goes into one topic, conceptual information goes into another topic, and so on.

Topic-based content provides the following benefits:

- Easy reuse—Reusing and sharing information in topics eliminates multiple authors writing the same information in different deliverables, which wastes time and introduces inconsistencies in terminology and presentation.

- Automatic updating of topics—When you modify information in a topic, it is automatically updated in the source file (or files) in which it's placed. Automatic updating eliminates the need for figuring out where topics are used in your source files and for making the same changes in the different locations. Making the change in a single location also ensures consistency.

To gain an understanding of how topic-based authoring works, suppose you are writing content for a group of printers that are similar; the advanced models have more features. Table 2 shows a feature map for the printers.

Table 2: *Features of three printer models*

Printer model number	500-sheet paper drawer	Secondary paper drawer	Color printing	Duplex printing attachment
Model A-1000	X	X		
Model B-2000	X	X	X	
Model C-3000	X	X	X	X

You and other authors write topics about the features, including the following topics:

- Adding paper to the 500-sheet paper drawer
- Using the secondary paper drawer

- Printing color documents
- Installing the duplex printing attachment

To create a deliverable for each of the models, you would gather topics; the method for assembling the topics depends on the software you're using. (Figure 46 on page 235 shows the process for a DITA-based workflow.)

If you are working in a topic-based workflow, it's likely you will write a particular set of topics instead of all the pieces that make up one manual or help set. Sometimes, authors don't even see all the topics assembled and formatted as a deliverable because those tasks are handled by another person or by the publishing workflow itself.

You can find more information about creating topic-based documentation in *Single Sourcing: Building Modular Documentation* by Kurt Ament (ISBN 9780815514916).

NOTE: If you're writing topic-based documentation, don't think that writing consistently isn't as important because you're writing smaller bits of information. Follow your department's standards so that the topic you are writing is similar to your other topics *and* that of other writers—it's highly unlikely you will write all the topics used in a deliverable. Differences in style and presentation become glaringly obvious when dissimilar topics are placed one after the other in a deliverable.

Dealing with the inevitable schedule changes

Regardless of whether you're writing interface, reference, conceptual, or procedural information, you'll deal with schedule changes on a project sooner or later (most likely sooner). Slipping schedules on a project fall into the same category as death and taxes—you can count on them. Now it's a bit of an exaggeration to say that there's *never* been a project during which dates were met, but those projects are in the minority.

Slipping schedules are common because a product's documentation is contingent upon the product itself. A product is going to change as it is developed—particularly a new one—so it's common sense that those changes will affect the content. Last-minute fixes, deletions, and additions to a product's functions most likely mean that there will be last-minute changes to the documents as well. A smart doc plan writer knows this and creates a schedule to accommodate changes. A good schedule also contains a "freeze point"—if the documents are to be delivered on time, the product's development must freeze on a particular day. If there are changes beyond the freeze point, the schedule slips accordingly.

Last-minutes changes sometimes mean that there are some trade-offs. For example, to make the deadline for delivery to the printer, you may have to forego a solid edit on a chapter about a function just added to the product. If you're documenting software that has automatic updates, you could release completed online help topics and then include the ones you're still writing in a future update.

Make these sorts of decisions with the involvement of the client and the documentation team. Even though time (or lack thereof) will play an important part in deciding on any compromises, it's important to keep the book's users in mind. Audience should *always* be a top concern, even when you're in panic mode because of last-minute changes. After all, if late changes to text make content less audience friendly, you undo much of the work you've completed already.

See "The reality of time constraints" on page 162 for more information about making compromises when time is tight.

Experience is the best teacher

Writing content for the first time is intimidating. There are many issues you need to remember while writing (audience, task breakdown, and so on), and then there are the slipping schedules, constantly changing information sources, and other issues that need your attention during the content development process.

If it all sounds like too much to handle, realize that new writers are rarely asked to draw up doc plans, create outlines, *and* write the content; senior writers and project managers often handle the planning and project management responsibilities. However, as you become a more experienced writer, your involvement in the management side of projects will most likely increase.

Many companies base technical writers' job descriptions on level of experience and the amount of management responsibility the job entails—for example, a level-one writer has a year of experience and has no management responsibilities, whereas a level-three writer has seven or more years of experience and acts as a project lead and as a mentor for level-one writers.

As you work on more and more projects, figuring out how to handle all the elements of a project becomes second nature. Technical writing seminars, a degree from a technical writing program, or this book alone will not make you a good writer. Courses, degree programs, and reference books on writing can help, but working on real-world projects is by far the best way to develop solid technical writing skills.

A vivid imagination can be most useful

Every once in a while, you'll face the Project from Hell (PFH). You'll know it when you see it, but the PFH usually involves getting no information, no software, no access to the developers, and no specs—nothing.

What do you do when you're asked to do the impossible: write about a piece of technology that you haven't seen (and that probably doesn't exist yet)?

One possibility in such desperate circumstances is to extrapolate. Take what little information you do have and make your best guess. If you do guess, make sure that material you write is clearly labeled as a guess—it's particularly important to flag fabricated information before you send your document to technical experts for review.

7 Writing task-oriented information

What’s in this chapter

- Elements of a procedure
- Introducing the procedure
- Breaking down a task into steps
- Including the results
- Adding notes, warnings, and cautions
- Using bulleted and numbered lists
- Letting illustrations tell the story
- Organizing information in tables
- Inserting cross-references

Task-oriented writing makes up the bulk of technical documentation. It’s also what beginner technical writers usually write first.

When you create an outline for content, you usually start by putting together a list of tasks that the user will perform. To begin documenting the software, you then try performing those tasks yourself and write down the steps needed to accomplish each task.

However, creating solid task-oriented content requires a lot more than just writing down what the user needs to do. You need to keep your audience in mind at all times, and you also need to consider questions such as the following:

- Does each step represent an action the user takes?
- Are the results of an action clearly explained?
- Would a graphic help explain an action more clearly?

This chapter explains what you need to think about while writing procedures.

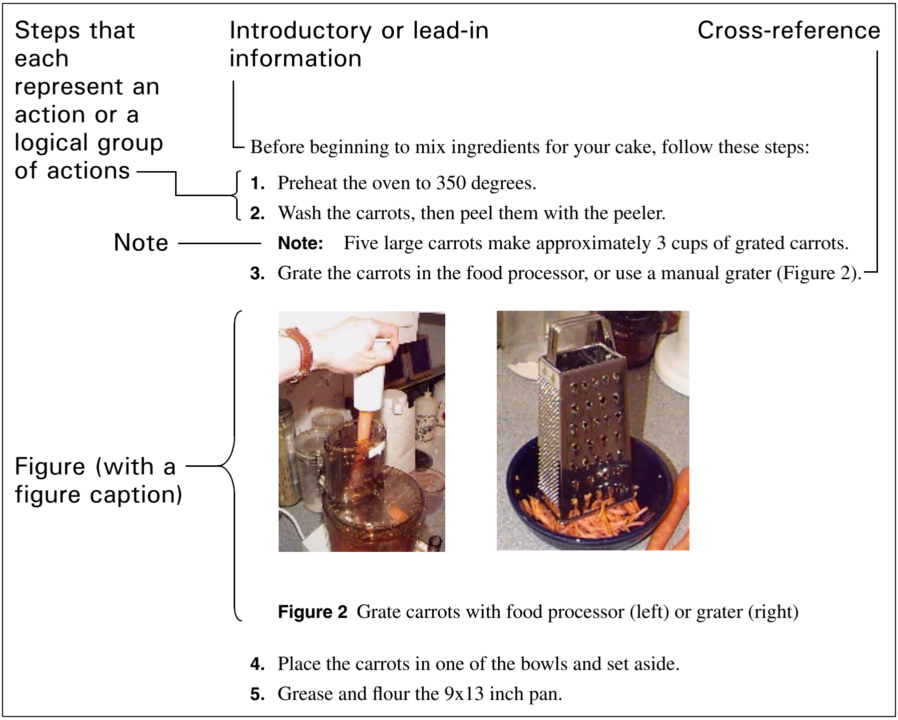
NOTE: This chapter is not an exhaustive resource on how to write task-oriented material. Instead, it explains many of the concepts that technical writers must know.

Elements of a procedure

Whether it explains how to use a piece of software or how to bake a cake, task-oriented information generally has the same elements—steps requiring action by the user, information about the results of those actions, and graphics, tables, or notes to clarify what the user does.

Figure 7 shows a sample procedure.

Figure 7: *Sample procedure*



Before beginning to mix ingredients for your cake, follow these steps:

1. Preheat the oven to 350 degrees.
2. Wash the carrots, then peel them with the peeler.

   Note: Five large carrots make approximately 3 cups of grated carrots.
3. Grate the carrots in the food processor, or use a manual grater (Figure 2).

Figure 2 Grate carrots with food processor (left) or grater (right)

4. Place the carrots in one of the bowls and set aside.
5. Grease and flour the 9x13 inch pan.

Steps that each represent an action or a logical group of actions
Introductory or lead-in information
Cross-reference
Note
Figure (with a figure caption)

The following sections provide details about how to handle the elements of a procedure.

NOTE: Some companies take a more minimal approach by eliminating lead-in sentences or information about the results of completing a step. If you're working in a structured authoring environment, the structure may not allow all of the components explained in this chapter.

Introducing the procedure

Before readers dive into a list of steps, they would probably like an idea of what they're about to accomplish. A lead-in sentence or paragraph can do just that.

Often, all that's needed is a simple sentence, like the one shown in Figure 7 on page 111. If you do use sentences for lead-ins, make sure all the lead-ins follow the same grammatical pattern. For example, if one lead-in reads

> To print a file, follow these steps:

the next one should read

> To delete a file, follow these steps:

and not

> You can delete a file by completing the following steps:

If necessary, your lead-in can be a short paragraph, particularly if you need explain any prerequisites for completing the procedure (for example, the user needs to complete another procedure before beginning the one being introduced). A lead-in can also briefly explain how completing a task fits into bigger scheme of using the product.

Breaking down a task into steps

You (or another member of your team) have already written an outline that breaks down the tasks you need to document. When you start writing the content, you continue the process of breaking down information—you break down the tasks in the outline into the steps within procedures.

By following a pattern in writing your procedures, you establish a rhythm that makes information easier to retrieve—the readers know what to expect. As Figure 7 on page 111 shows, a step can contain one action or a group of actions. Step 1 in the sample procedure tells the user to just turn on the oven, while Step 2 instructs the user to complete two actions that are closely associated with one another—washing and peeling. In content that explains software, it's common to group selecting a menu and a choice from that menu:

> Select the **Edit** menu, then **Copy.**

You can also group the actions of typing a command and pressing **Enter**:

> Type `ftp` at the command prompt, then press **Enter.**

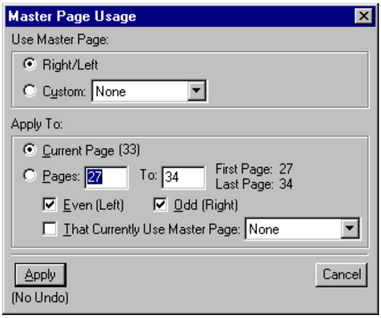
However, don't take grouping to extremes by combining multiple actions, particularly selecting menu choices from two different menus.

Including the results

When writing procedures, it's often helpful for the user if you include the result of each step. For example, if a step tells the user to select a menu choice, the step should include the result of selecting that choice (Figure 8 on page 114).

Figure 8: *Show the result of an action in a step*

1 Select the **Format** menu, then **Page Layout**, then **Master Page Usage**. The Master Page Usage dialog box is displayed (Figure 21).



Master Page Usage

Use Master Page:

● Right/Left

○ Custom: None

Apply To:

● Current Page (33)

○ Pages: 27 To: 34 First Page: 27 Last Page: 34

☑ Even (Left) ☑ Odd (Right)

□ That Currently Use Master Page: None

Figure 21 Master Page Usage dialog box

When including the result of an action, don't make it look like a step by assigning a paragraph tag that places a number before the text. If you're working with structure, don't assign a list item element to the result information: that inserts a step number in front of the text. Instead, use an element for a paragraph that's permitted within a procedure.

Following the example shown in Figure 8, the result for selecting the menu choices in Step 1 should *not* become the second numbered item in the list of steps:

1. Select the **Format** menu, then **Page Layout,** then **Master Page Usage.**
2. The Master Page Usage dialog box is displayed.

Basically, be sure each numbered list item in a procedure tells the user to perform an action. Depending on the template or structure you're using, you may be able to make the result more obvious by putting it on a separate line:

1. Select the **Format** menu, then **Page Layout,** then **Master Page Usage.**

   The Master Page Usage dialog box is displayed.

Adding notes, warnings, and cautions

You can use notes to include information in a procedure that doesn't quite fit into the action/result flow. Use notes for information that needs to be mentioned but that is a bit off-topic. A note contains information that is helpful to the user but that does not involve damage to the product or physical danger. See Figure 7 on page 111 for an example of a note. Your company's documentation may also include tips, hints, and other helpful tidbits.

If the information concerns possible damage to the product or danger to the user, use caution, warning, and danger paragraphs.

Usually, a caution tells the reader about a potential problem, but it does not involve damage to the product or injury to people.

A warning may explain certain actions that will damage components. In a software manual, a warning may point out that selecting a particular button on the interface will permanently delete a file.

A danger notation indicates the possibility of serious injury or death. Generally, danger notations are found just in hardware documentation, but sometimes content for software contains them, particularly if you're dealing with medical software. For example, for software that calculates medication dosages, a danger notation might point out that entering an incorrect weight for the patient could result in calculating the wrong dosage, which could kill a patient.

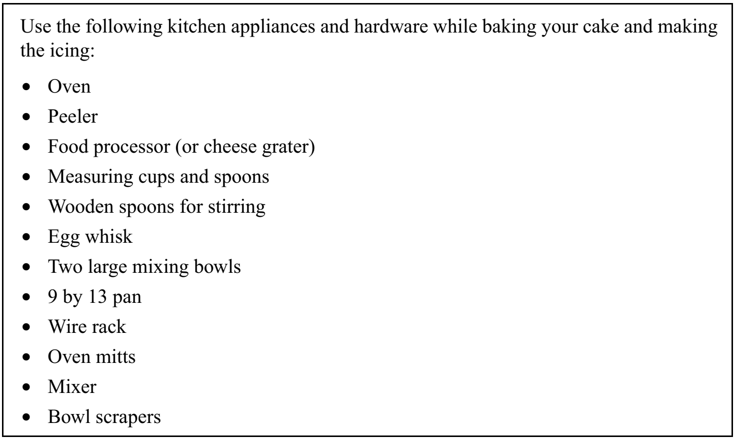
In a template-based workflow, your style guide can give you guidance on when to use and how to format notes, cautions, and other admonishments. If you are working in a structured authoring environment, the structure guides your placement of admonishments and applies the formatting to them. Companies often have very specific guidelines about including warnings and dangers to ensure user safety and to reduce product liability.

Be judicious about inserting notes into the text. Overusing notes diminishes their importance and can be an annoyance for readers.

Using bulleted and numbered lists

In general, use a numbered list to denote steps that the user must perform in a particular order, as shown in Figure 7 on page 111. For lists of items or choices (which do not have a particular order), use a bulleted list, as shown in Figure 9:

Figure 9: *Bullets don't denote a sequence of steps*



Use the following kitchen appliances and hardware while baking your cake and making the icing:

- Oven
- Peeler
- Food processor (or cheese grater)
- Measuring cups and spoons
- Wooden spoons for stirring
- Egg whisk
- Two large mixing bowls
- 9 by 13 pan
- Wire rack
- Oven mitts
- Mixer
- Bowl scrapers

If you have a list of three or more items, consider putting them into a bulleted list instead of listing them in a sentence. For example, if all the items listed in Figure 9 on page 117 were just part of a sentence, the user would have to wade through the following monster chunk of text:

> To make your cake and the icing, use the oven, peeler, food processor (or cheese grater), measuring cups and spoons, wooden spoons, egg whisk, two large mixing bowls, 9 by 13 pan, wire rack, oven mitts, mixer, and bowl scrapers.

The bulleted list communicates the same information in a much clearer way.

In structured authoring environments, it is possible to specify that a list contain a minimum number of items. Many documentation groups do not like lists that contain just one item, so they set the structure to require two or more items in a list.

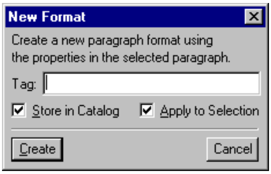
Letting illustrations tell the story

Sometimes, no amount of writing can explain something as clearly as a well-chosen screen shot or technical illustration.

In content for software, it is common practice to show the dialog boxes and windows a user encounters while completing a procedure, as shown in Figure 10:

***Figure 10:** Screen shots of a program's dialog boxes and windows make procedures easier to follow*

2 From the **Commands** drop-down list, select **New Format.** The New Format dialog box is displayed (Figure 14).



New Format

Create a new paragraph format using the properties in the selected paragraph.

Tag:

☑ Store in Catalog ☑ Apply to Selection

Figure 14 New Format dialog box

In content about hardware, a technical illustration or photograph may depict the action a user is performing in a step, as shown by the photograph with the food processor in Figure 7 on page 111.

NOTE: In online content, animations and video clips can be even more effective in showing how to complete an action.

If you do insert a graphic, don't assume it is self-explanatory. Include a figure caption to explain its purpose, and if necessary, point out particular items with callouts, which are the lines and bits of text that focus the reader's attention. See the "Hula man" callout in Figure 4 on page 99 and the "Oven" callout in Figure 12 on page 122 for examples.

When deciding whether to include a graphic, consider your audience. For example, if you're telling computer novices about how to open a file, you might want to include a screen shot of the standard Windows Open dialog box. However, for more computer-savvy audiences, such a screen shot would probably not be necessary because the users are accustomed to that standard dialog box. If your product uses a nonstandard Open dialog box, you might want to explain it and show a graphic, though.

In general, it's a writer's responsibility to take screen shots; some writers (particularly in smaller departments) create all visual content.If you're lucky enough to be working with graphic designers and technical illustrators, be sure to keep them informed about what you need. Also, check the schedule in the doc plan for graphic request cutoff dates.

See Chapter 8, "Visual communication," for more information about creating visuals for content.

Organizing information in tables

Tables are yet another way to communicate information (particularly information that is repetitive or has a pattern). To ensure a table's purpose is clear, use headers on your columns and (rows, if necessary). You can also add a table caption. Figure 11 shows a table with headers on the columns and a caption.

Figure 11: *Tables are useful for information that has a pattern*

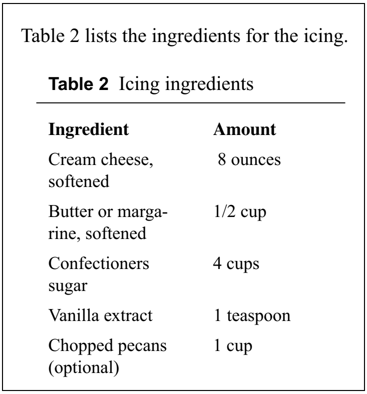


Table 2 lists the ingredients for the icing.

Table 2 Icing ingredients

Ingredient	**Amount**
Cream cheese, softened	8 ounces
Butter or margarine, softened	1/2 cup
Confectioners sugar	4 cups
Vanilla extract	1 teaspoon
Chopped pecans (optional)	1 cup

If your content is based on structure, the structure will specify whether a table has a caption, heading row, and so on. Some elements may be optional (for example, you can include a caption, but it is not required.)

Tables can be helpful in reference material (for example, tables listing commands, their parameters, and their uses).

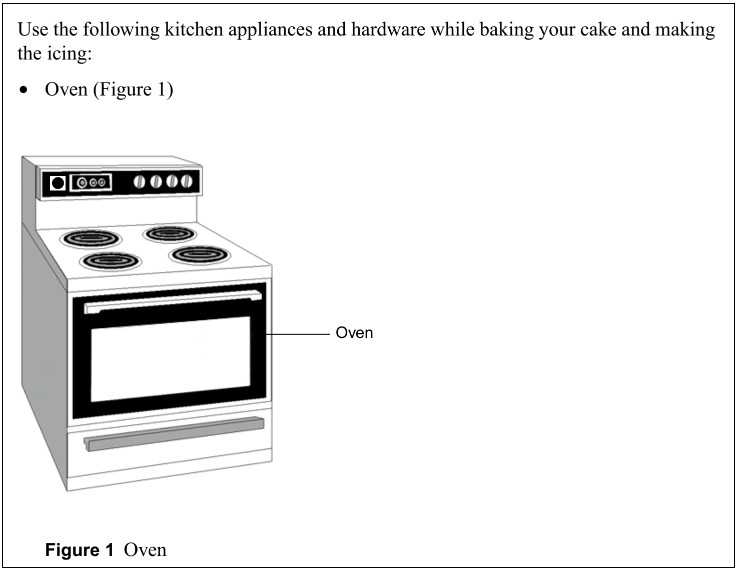
Inserting cross-references

It's not unusual for multiple figures or tables to appear in task-oriented content, so it's important that any text that refers to a figure or table be very specific about what's being referenced. The best way to ensure that the user references the correct item is to use a cross-reference.

Cross-references in printed documents

In printed documentation, placing a cross-reference can be as easy as inserting an identifier for the referenced item in parentheses (Figure 12).

Figure 12: *Simple cross-reference*



Use the following kitchen appliances and hardware while baking your cake and making the icing:

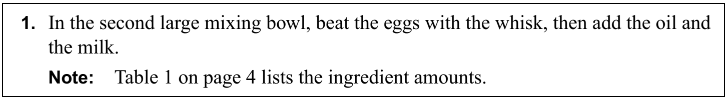
- Oven (Figure 1)

Figure 1 Oven

Figure 7 on page 111 and Figure 11 on page 121 also show basic uses of cross-references.

In printed documents, it's sometimes necessary to include the page number for the referenced item when the item is several pages forward or backward, as shown by the cross-references you just read in the preceding paragraph and by the note in Figure 13:

Figure 13: *Cross-reference with page number*



1. In the second large mixing bowl, beat the eggs with the whisk, then add the oil and the milk.

   Note: Table 1 on page 4 lists the ingredient amounts.

If your text development software has a cross-referencing feature, use it to insert references instead of just typing in the name and page number of the item you're referencing—this applies to workflows based on templates and on structure. When you're creating printed or PDF-based content, the book's pagination will change as you add and delete content, but your authoring tool will automatically update the page numbers in your cross-references. If you manually typed the page numbers for your cross-references, you'd have to verify that every reference's page number was correct before printing the document.

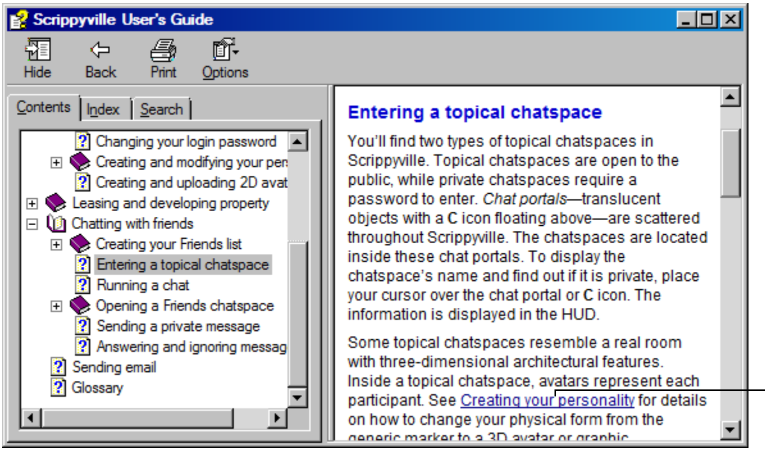
It's not necessary to include the page number if the referenced item is on the same page or a facing page in a printed book.

NOTE: Sometimes you won't know what pages in a manual will be facing one another until final production, so verifying whether cross-references need page numbers should be part of your production process. See "Page numbers in cross-references" on page 192 for more information.

Links in online content

In online content and PDF files, cross-references are generally *live links*; when users click on a link, the on-screen content jumps to where the cross-reference is pointing. To make links in online content distinct from standard text, they are usually highlighted with a different color, underlining, or both (Figure 14).

Figure 14: *Cross-reference link in help file*


Scrippyville User's Guide
Hide
Back
Print
Options
Contents
Index
Search
Changing your login password
Creating and modifying your per
Creating and uploading 2D avat
Leasing and developing property
Chatting with friends
Creating your Friends list
Entering a topical chatspace
Running a chat
Opening a Friends chatspace
Sending a private message
Answering and ignoring messag
Sending email
Glossary

Entering a topical chatspace

You'll find two types of topical chatspaces in Scrippyville. Topical chatspaces are open to the public, while private chatspaces require a password to enter. *Chat portals*—translucent objects with a **C** icon floating above—are scattered throughout Scrippyville. The chatspaces are located inside these chat portals. To display the chatspace's name and find out if it is private, place your cursor over the chat portal or **C** icon. The information is displayed in the HUD.

Some topical chatspaces resemble a real room with three-dimensional architectural features. Inside a topical chatspace, avatars represent each participant. See Creating your personality for details on how to change your physical form from the generic marker to a 3D avatar or graphic

Link is underlined.

If you are creating online content by converting source files for printed books, your conversion tool will likely create and format the links automatically—as long as the cross-references are set up in a way the tool recognizes. Generally, if you set up the references with the text development's tool cross-reference feature, the conversion tool will process them correctly. Page numbers are not needed in online links, so most conversion tools automatically strip that information.

If you directly author online content, the tool you're using probably includes a way to quickly set up cross-reference links in the correct format.

Links in online content aren't limited to just cross-references. You can also point to other content: web sites, videos, and so on. Before linking to *any* content, first verify the policies about referring to other resources. Many companies have strict rules about linking to online material (and that sometimes includes information created by the company itself). Also, keep in mind that web sites frequently change, so links will break when web pages are deleted, renamed, and so on.

8 Visual communication

What's in this chapter

- ❖ What sort of visuals should I use?
- ❖ Using graphics in content
- ❖ Using rich media content
- ❖ Hey, I'm a writer, not an artist!

Perhaps the name technical *writer* is a bit misleading. As you create your documents, you'll discover many places where providing information in a graphic or a video clip will make the content much easier to understand.

Traditional graphics include several different types of illustrations, such as:

- Line art (drawings)
- Screen shots
- Computer-Aided Design (CAD) graphics

In online content, you're not limited to providing just static images. You can include rich media that incorporates both visuals and sound: video clips, audio instructions, and animated screen shots.[1]

Rich media can also be interactive. For example, you can create multiple-choice quizzes that give readers instant feedback on whether they answered the questions correctly.

What sort of visuals should I use?

There's no single correct answer to this question. You generally use screen shots when you want to show a picture of what's being displayed on the computer screen. In online content, you can take that idea even further by including an animated screen shot with a voiceover explaining what is being shown in the clip.

Photographs are great for showing real-life images or highly detailed information. The same is true of video clips in online content—they show the exact process a user should follow to complete a task.

Line art lets you simplify a piece of machinery and show just the parts that you want the reader to focus on. CAD graphics show exact, detailed views of components. CAD drawings can also be used to create animations that zoom in on a particular part, for example. As always, you need to consider your audience and

[1] Animated screen shot showing how to complete a task in FrameMaker:
www.scriptorium.com/change.html

what they need to know before you decide which type of image or rich media to use.

This chapter focuses on the technical details of adding visuals to your content, such as the file formats you'll need.

NOTE: For information about designing visual information, consult *The Visual Display of Quantitative Information* by Edward Tufte (ISBN 9780961392147).

Using graphics in content

Line art, screen shots, and photographs are among the types of still images you can include in your content.

Understanding graphic file types

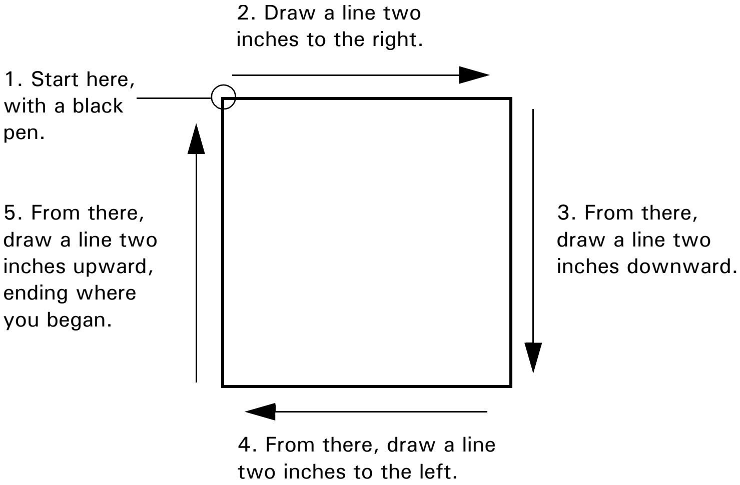
In the physical world, visual images have lots of different formats—oil paintings, pencil drawings, blueprints, photographs, and so on. When you store images on a computer, however, you only have two types of graphics—vector images and bitmap graphics.

In general, you set up drawings (for example, line art and flow charts) as vector images, and you use bitmap formats for photographs and screen shots.

Vector images

In mathematics, a *vector* is a line that has a starting point, a length, and a direction (in other words, an arrow). *Vector images* are graphics that are made up of vectors. The vector image file contains a series of mathematical equations that describe the image in the file. A simple image, such as a box, would be defined by instructions something like this: “Start in the upper left corner, with a black pen that’s two points wide. Draw a line two inches to the right. From there, draw a line two inches down. From there, draw a line two inches to the left. From there, draw a line two inches up, finishing where you started.” That’s a vector description, as shown in Figure 15.

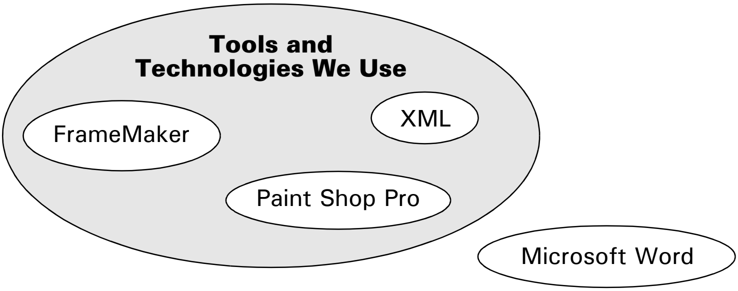
Figure 15: *How vector graphic information is stored*


2. Draw a line two inches to the right.
1. Start here, with a black pen.
5. From there, draw a line two inches upward, ending where you began.
3. From there, draw a line two inches downward.
4. From there, draw a line two inches to the left.

The content development tool you’re using probably has a drawing tool that will create vector images. Some tools, such as FrameMaker, have feature sets sufficient to create simple flow charts, and Microsoft Word has all that plus a fairly extensive set of useful symbols (such as flow chart boxes). Other tools for this kind of work include Adobe Illustrator, OpenOffice.org Draw, and Microsoft Visio. (Visio is especially good for flow charts.)

Figure 16 shows a typical example of vector art—a Venn diagram, which includes both graphics and text.

Figure 16: *Venn diagram*



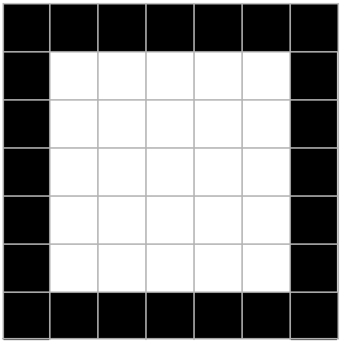
If you need to create a diagram or line art, vector images are a good choice; the images stay sharp even when you resize an image to appear larger in your content files. Also, the vector files are quite small because the vector descriptions are compact.

Vector images, however, are a poor choice for photographs and screen shots. Photographs consist of thousand or millions of dots, each of which uses a specific color. Vectors are not an efficient way to describe a photograph.

Bitmap images

Bitmap graphics handle photographs with ease. A bitmap graphic stores information in a grid. Each square in the grid has a unique address and a color. Unlike the vector image of the box in Figure 15, a bitmap graphic does not have any information about the box; instead, the bitmap stores the image by describing which squares are black (that is, the squares that contain part of the box's outline) and which ones are white (meaning they don't have the box's outline in them). Figure 17 shows how bitmap editors describe a picture.

Figure 17: *How bitmap images work*



Black, black, black, black, black, black, black.

Black, white, white, white, white, white, black.

Black, white, white, white, white, white, black.

Black, white, white, white, white, white, black.

Black, white, white, white, white, white, black.

Black, white, white, white, white, white, black.

Black, black, black, black, black, black, black.

This is the image, superimposed over the bitmap grid.

This is what the file might look like if it were written in English.

This is an inefficient way to draw a box—the vector version is much simpler. But for photographs and screen shots, bitmaps work well because you can't easily describe the image as a series of vectors.

File formats

Graphic files come in many different file formats, such as EPS, BMP, TIFF, GIF, JPEG, and PNG, just to name a few. Certain file formats are always bitmaps; others are always vector images. Table 3 provides some recommendations for graphic formats.

Table 3: *File formats for graphics*

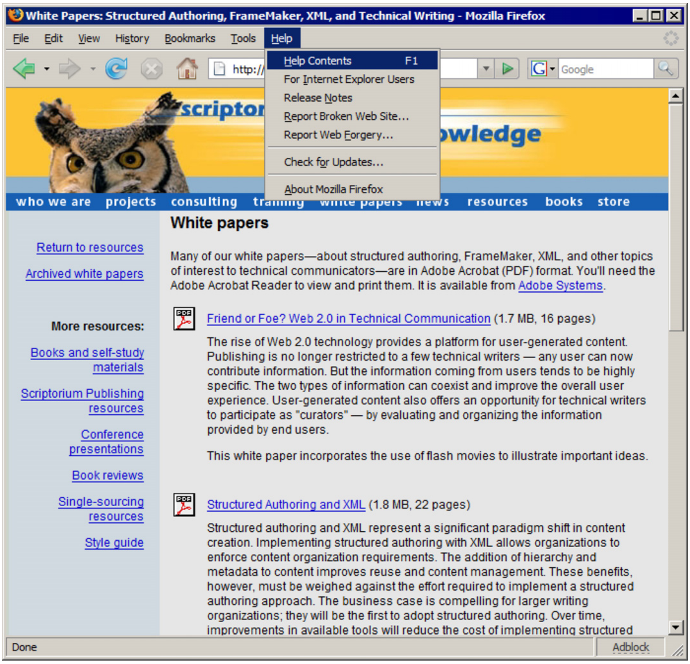
For this type of graphic	Use this file format
Vector image such as diagram or flow chart	• Encapsulated PostScript (EPS) • Windows Metafile (WMF) EPS is the standard format for vector images and the most widely accepted.
Photograph	Use Joint Photographic Experts Group (JPEG) format for graphics that need to be as small as possible. JPEG provides good compression for photographic images. Keep in mind, however, that JPEG compression is "lossy"—compressing the file reduces its quality. To preserve every data point, use Tagged Image File Format (TIFF).
Screen shot	• Windows Bitmap (BMP) • Sun Raster Image (RS) • PC Paintbrush (PCX) • Macintosh picture file (PICT) • CompuServe Graphics Interchange Format (GIF) • Portable Network Graphic (PNG) GIF and PNG are good choices because they automatically compress the graphic. The compression is "lossless"—it does not change the quality of the original image (but you may lose color information when saving an image as a GIF because that format supports only 256 colors).

Scope of an illustration

One important consideration in creating graphics is to give your readers *context.* You want to be sure that readers can identify the main point of the illustration immediately, but you also want them to understand what part of the whole you're highlighting in your graphic.

Consider, for example, Figure 18, which shows the application window for Mozilla Firefox with the **Help** menu displayed. If you're trying to show readers the location of the **Help Contents** choice in the **Help** menu, this is probably not the best way to do it.

Figure 18: *Providing too much information makes it difficult for readers to locate the item you want them to see*



White papers

Return to resources

Archived white papers

More resources:

Books and self-study materials

Scriptorium Publishing resources

Conference presentations

Book reviews

Single-sourcing resources

Style guide

Many of our white papers—about structured authoring, FrameMaker, XML, and other topics of interest to technical communicators—are in Adobe Acrobat (PDF) format. You'll need the Adobe Acrobat Reader to view and print them. It is available from Adobe Systems.

Friend or Foe? Web 2.0 in Technical Communication (1.7 MB, 16 pages)

The rise of Web 2.0 technology provides a platform for user-generated content. Publishing is no longer restricted to a few technical writers — any user can now contribute information. But the information coming from users tends to be highly specific. The two types of information can coexist and improve the overall user experience. User-generated content also offers an opportunity for technical writers to participate as "curators" — by evaluating and organizing the information provided by end users.

This white paper incorporates the use of flash movies to illustrate important ideas.

Structured Authoring and XML (1.8 MB, 22 pages)

Structured authoring and XML represent a significant paradigm shift in content creation. Implementing structured authoring with XML allows organizations to enforce content organization requirements. The addition of hierarchy and metadata to content improves reuse and content management. These benefits, however, must be weighed against the effort required to implement a structured authoring approach. The business case is compelling for larger writing organizations; they will be the first to adopt structured authoring. Over time, improvements in available tools will reduce the cost of implementing structured

The image is so large, you had to shrink it to fit on the page, making the lettering in the menu small and hard to read. In Figure 19, the menu choice is easy to see, but the image lacks context; the reader cannot easily tell where in the application the choice is located.

Figure 19: *Too little information leaves the reader lost*

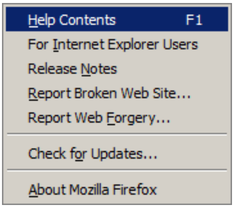
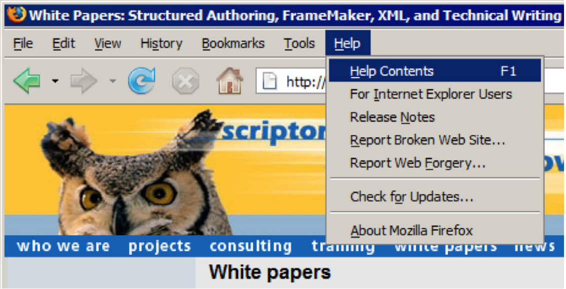


Figure 20 shows a better approach. In this image, you show the **Help** menu along with just enough of the main window to give the reader an idea of where this menu is displayed in the application.

Figure 20: *Now readers can locate the item you want them to see*



For every graphic you include in your document, consider how much context is needed so that the graphic makes sense to the reader.

Displaying information from your computer screen

Most software documentation uses pictures of the interface; these pictures are called screen shots. There are many programs for taking screen shots; for example, Paint Shop Pro, SnagIt, and Hijaak Pro are common for Windows machines, and Snapz Pro is a common Macintosh utility. All of these tools let you take a picture of an entire screen, part of the screen, or just an active window.

You don't necessarily need a separate utility to take screen shots. Both the Macintosh and Windows operating systems have built-in screen capture functions. They don't have all the bells and whistles of the commercial versions, but if you just need a couple of screen shots, they're usually good enough. Consult the help for your operating system for more information.

When colors scheme

When multiple authors are working on a project, all their computers should have the same display scheme to ensure consistency in screen shots. For example, on the Windows operating system, the colors and fonts in dialog boxes vary according to which desktop theme is selected. Refer to the online help for your operating system for information about changing display settings.

If you're working on an operating system that features translucent title bars on windows (Windows Vista, for example), consider changing the desktop theme to one with solid title bars. You probably don't want what's behind title bars visible in screen shots.

Authors should also check their screen resolution settings. The resolution affects the size of dialog boxes. Even if a project requires screen shots taken on multiple operating systems, it is a good idea to use the same (or similar) screen resolution to keep the graphics consistent in size.

Your style guide should specify the display and resolution information for taking screen shots.

Placing graphics in your content

After you create your illustrations or screen shots, you'll want to put them in your content. Before doing so, consider the following points:

- Whether to link or embed graphics
- Uniformity in the size of graphics

Linking vs. embedding graphics

When you're working with a text processing tool, the procedure for placing graphics varies depending on the application you're using, but many applications give you a choice between *linking* and *embedding* your graphics. When you link a graphic, you store a pointer to the image in the document file. When you embed a graphic, you put a complete copy of the graphic in your document file. Each of these techniques has advantages and disadvantages, which are summarized in Table 4:

Table 4: *Linking vs. embedding graphics*

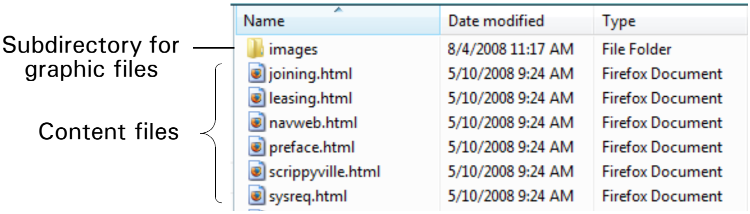
	Linked graphics	Embedded graphics
File size	Because linked graphics contain only a pointer to the original graphic file, the document file remains small.	When you embed the graphic, you insert a copy of the graphic into the document, which increases the file size by at least the size of the graphic.
Portability	To move a document file from one location to another, you must remember to also move all of the linked graphics.	If all graphics are embedded, you can move the document file and be sure that recipients will be able to see all the graphics.
Updating	When you update a graphic, the linked graphics are automatically updated.	When you update the graphic, you must reimport the graphic into the document.

For technical documentation projects, which tend to grow and evolve, it's usually best to use linked graphics.

NOTE: If you are working in a structured authoring environment based on Extensible Markup Language (XML), your content files refer to graphics with links. There isn't an option to embed images in XML content. The same is true if you are authoring HTML content for the web.

Figure 21 shows an easy way to organize your content and graphic files; the folder structure works well for source files created with a text processor (template- or structure-based) or a help authoring tool. In the example, all graphic files are in a subdirectory called "images."

Figure 21: *Directory structure for organizing content and graphic files*



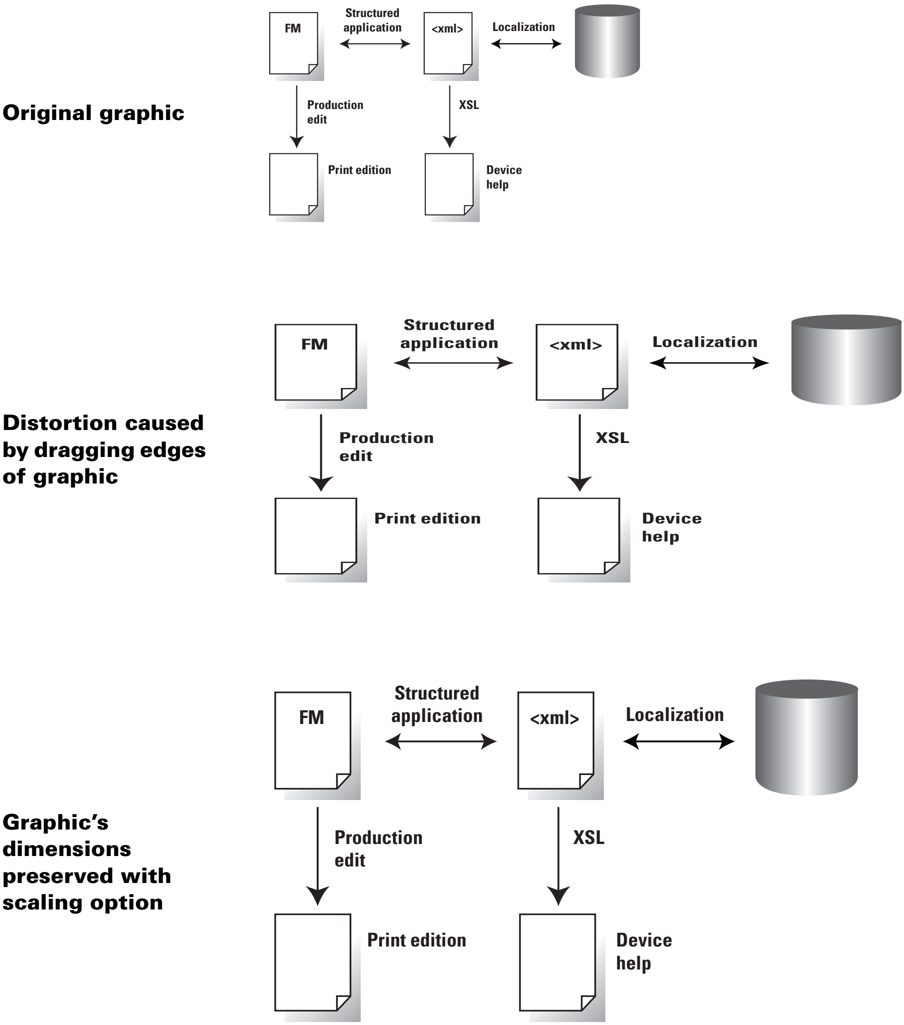
Name	Date modified	Type
images	8/4/2008 11:17 AM	File Folder
joining.html	5/10/2008 9:24 AM	Firefox Document
leasing.html	5/10/2008 9:24 AM	Firefox Document
navweb.html	5/10/2008 9:24 AM	Firefox Document
preface.html	5/10/2008 9:24 AM	Firefox Document
scrippyville.html	5/10/2008 9:24 AM	Firefox Document
sysreq.html	5/10/2008 9:24 AM	Firefox Document

Keeping graphics uniform in size

When you place a graphic in a document file, some programs will prompt you to specify the dots per inch (DPI) setting for the graphic. Use one DPI setting consistently—this is especially important if multiple authors are creating content. Even if your text processing application doesn't give you the ability to specify a DPI setting for a graphic, you should pick one method for inserting graphics to ensure consistent sizing. Check the user manual or online help for more information.

If you need to modify the size of a graphic after placing it in a file, it's usually best not to adjust the size by selecting the graphic and dragging a corner. Some applications have commands for scaling graphics; scaling preserves the ratio of the graphic's width and height. Figure 22 on page 140 shows what happens when a graphic's ratios aren't preserved during resizing.

Figure 22: *Resizing a graphic*



Original graphic

Distortion caused by dragging edges of graphic

Graphic’s dimensions preserved with scaling option

Using rich media content

More and more technical information is delivered online, so technical communicators are no longer limited to including still images in their content. Animations, videos, and other rich media can contribute greatly to a user's understanding of a product.

Creating animated screen shots and other rich media

Creating an animated screen shot may seem daunting in comparison to capturing a still screen image, but there are many software programs that simplify the process. Programs such as Captivate and Camtasia streamline the process of recording the action on your screen, adding callouts and titles, and including narration.

To prepare, consider writing a script; you can probably use some of the text you've already written. You can also write down a list of the actions you want to show in the video tutorial. While planning, evaluate the scope of the clip (as explained in "Scope of an illustration" on page 133): how much of the screen area should you show to demonstrate the feature?

While recording, keep in mind the speed at which you move your mouse, make menu selections, and so on. Moving too quickly may make it hard for users to follow the action, and it can also cause problems if you add narration later—some programs don't allow you to adjust timing.

If you make a mistake while recording, it doesn't you mean you've lost everything before the error. You can pause recording to gather your thoughts, restart the recording, and repeat the information you flubbed earlier. Continue with the rest of the demo, and you can remove the mistake during editing.[1]

If multiple writers are creating animated screen shots, all the writers need to have their screens set to the same display settings to ensure consistency in appearance. See "When colors scheme" on page 136 for more information. (Much of the advice for creating still images applies to motion files: a consistent look-and-feel for all visual communication makes your content easier to understand and gives it a professional appearance.)

You can use the Flash application to create multimedia content that combines still images, videos, audio, and interactive features (such as quizzes that verify the user's answers). Flash is an expensive and complex tool; it's not as easily learned as screen capture software.

Understanding rich media file formats

File formats for rich media are more complicated than those for static images—most formats are proprietary to the companies that make them, require specific players or browser plugins to play them, and may not work on all operating systems. Although basic players for the many proprietary formats are generally free, some become intrusive by displaying advertisements or bothersome messages suggesting users upgrade to a paid version of the program.

[1] Tom Johnson provides several excellent tips for creating video tutorials on his blog, I'd Rather Be Writing: www.idratherbewriting.com/2008/09/11/how-i-create-video-tutorials

At the time of this writing, the Flash file format is widely used, particularly for content shown in web browsers. The free web browser plugin for Flash is supported across many operating systems, and free standalone Flash players are available.

Even if you use another program to create rich media content, the software will likely give you the option of saving files in Flash format. You can also use a video converter program to convert files to Flash format.

Rich media content is rapidly evolving, so file formats will likely change, too. Another format will probably emerge as the standard—and may be the standard when you read this book.

Adding rich media files to your content

Newer content creation tools include features for adding rich media. Web development and help authoring tools often have an "Add Media" option that makes it easy to include multimedia files.

Some document processing tools let you add rich media files. Rich media content doesn't work in the printed manuals based on document processing files, but it does work in PDF files and other online content generated from those files.

NOTE: Even if your document or word processing tool doesn't allow you include rich media in the source files, you can save the content as PDF format and then use Acrobat to add the rich media files.

Depending on the tool you use to create PDF files, you may have the option of embedding or linking to rich media files. While embedding the rich media will increase the file size of the PDF file, keeping the rich media files separate from the PDF file means there is a greater chance the user may not get all of the necessary files.

Your department probably has a policy on whether you should embed or link to rich media files. If not, work with other members of your team to come up with a policy that works best for those who use the content. Also, if you're delivering content that's distributed online, be sure to include web development and networking teams in your discussions. They can tell you about any file size restrictions and other aspects to consider when delivering information over the Internet.

Hey, I'm a writer, not an artist!

Even if you have no artistic leanings, there is a good chance you will be responsible for creating the visuals in content you are writing. Most technical writers are required to take their own screen shots, and many must draw their own illustrations because no professional illustrators are available to help. It's a good idea to become familiar with the tools for creating simple illustrations.

At a minimum, you should be comfortable with the following:

- Taking screen shots
- Creating basic line drawings (such as Figure 16 on page 131)
- Importing illustrations into a document

Because animated screen shots are becoming prevalent in online help, learn how to create them, too. That skill will make you more marketable to prospective employers.

9 The importance of being edited

What's in this chapter

- Preventive measures
- Copy editing vs. technical editing vs. production editing
- Editing the content—what you and your editor can expect
- Editorial checklists
- Working with review comments
- The reality of time constraints

When you think of having your content edited, you may envision a severe-looking character tearing your text to shreds with very sharp red pencils. Instead, be thankful to get your text reviewed *at all.*

Fewer and fewer technical documentation departments employ full-time technical editors to review content, and many groups rely upon writers to review each other's work.[1]

Whether your reviewer is a fellow writer or a dedicated editor, the editing process can help you do the following:

- Improve the organization, tone, and consistency of your content
- Correct spelling and grammatical errors

[1] There are some departments that do not review content at all. We do not recommend this option, however. At a minimum, a department should institute a peer review policy so that writers review each other's content before it is released.

To ensure that content is consistent, a technical editor (or the writers reviewing each other's work) should establish style guidelines early in a project. Coming up with the rules at the outset of a project can save everyone lots of work—when all the writers know the rules up front, everyone's text will be much more consistent and will therefore require less rework later on.

An editor ensures that your material clearly communicates the necessary information at the appropriate audience level. That sort of feedback is invaluable. And, of course, every error caught during the editing process is one less mistake that users see—and that makes you look better.

This chapter explains the importance of editing content during the technical communication process. If you work in a department where there is no full-time editor, you and your fellow writers will handle the tasks described in this chapter.

Handling reviews electronically

Today, many departments rely on electronic review processes, particularly when employees are not in the same location. Instead of writing comments on pages of printed output, reviewers insert comments into an electronic version of the content. For example, Acrobat has tools that allow you to include notes and highlight text in a PDF file. Some companies have sophisticated electronic workflows that enable multiple people to include comments in a review copy. The writer can see all reviewers' comments combined into one file and then accept or reject the changes.

NOTE: Some companies also use software (such as the acrolinx IQ Suite) to automatically check grammar, adherence to style guidelines, and use of terminology.

Preventive measures

The sooner a team considers the editing process during a project, the better. In particular, there are several editing tasks to complete in the early phases of a project:

- Reviewing doc plans and schedules
- Establishing style guidelines
- Deciding on terminology
- Examining legacy content
- Reviewing early chapters

Reviewing doc plans and schedules

If you have a dedicated technical editor for your project, it's important to coordinate dates with the editor's schedule, particularly because an editor often supports multiple writers and will need to stagger reviews.

A schedule should also take an editor's workload into account—it's not reasonable to expect an editor to do a thorough edit of 1,000 pages in less than a week.

NOTE: A decent rule of thumb is that the editor will complete about 10 pages per hour, which means about 400 pages per week. However, that number is affected by the complexity of the content and the quality of the writing.

If you work on a team that doesn't have a dedicated editor, you still need to account for peer reviews when building doc plans and schedules. Allow enough time for writers to review each other's work, and schedule time for making revisions based on the reviews.

Establishing style guidelines

Most documentation departments already have style guidelines in place. These guidelines explain issues such as the following:

- Punctuation
- Capitalization
- Word choice and terminology
- Highlighting
- Acronym use

Many companies develop their own style guides and distribute them to all employees, either in printed or online form. However, some companies follow the guidelines in an established reference such as *The Chicago Manual of Style* and may distribute a smaller, separate document that details any exceptions and additions to the reference's rules. See "Editorial" on page 293 for a list of style references.

Scriptorium Publishing's style guide is available on the web at:

www.scriptorium.com/Standards/

If you're doing work for a client, be sure to ask about the client's style guidelines. You need to ensure that your work matches the client's guidelines.

Alienating an editor or a reviewer in one easy step…

The fastest road to a bad relationship with someone editing your content is trying to impose your personal style preferences on the style guide. Remember, your preferences are just that—preferences—and that's why it's a *style* guide, not a grammar guide. Telling the person reviewing your work that "only idiots use serial commas" is not going to build a good working relationship.

If you feel strongly that something in the style guide should be changed, approach the editor who manages the style guide; in the case of a department that has no editor, have a team meeting and decide whether the style guide should be changed.

Keep in mind, though, that every writer has a set of ingrained style preferences. If a department modified the style guide to accommodate every new writer, the style guide would change constantly—and frankly, the only thing worse than style guidelines you disagree with is style guidelines that change constantly!

Deciding on terminology

Style guides often contain word usage information that explains the correct way to use particular terms (for example, use *list box* instead of *listbox*) and what terms should be avoided altogether. This usage information, however, may not anticipate terminology specific to the project you're documenting. In that case, writers and editors should work together to draw up a list of terms for the new product. The editor or a writer on the team should compile the list of project-specific terms and ensure that all writers have a copy of the list.

Examining legacy content

If one of your information sources is legacy documents from the product's previous release, examine those documents for terminology and specific presentation methods at the beginning of the project. Check with the client about any of the terminology and presentation methods from the previous release that should be preserved.

A review of legacy information can also identify any weaknesses in presentation or organization that you should avoid in the new content. If you didn't write the earlier version, this is a pain-free way of getting some excellent feedback.

Reviewing early chapters

It can be very helpful for an editor or fellow writer to look at the first chapter or two that you write, particularly if you're a new writer. Early edits can give you an idea of what to avoid as you continue to write, and they can also ensure that all writers on a project are consistent in tone, audience, and presentation.

Copy editing vs. technical editing vs. production editing

Different types of editing include:

- Copy editing—editing text for spelling, adherence to grammar rules and to style guidelines, and general clarity.
- Technical editing—examining the content for organization, presentation, and consistency. Does the text flow logically? Do the text and graphics work together? Are similar types of information presented in the same manner? Also known as developmental editing.
- Production editing—ensuring that a book is ready for printing or that online help is ready for release.

On content development projects, a technical editor or fellow writer often does the copy editing and technical editing at the same time. A few companies have production specialists whose primary responsibility is preparing documents for printing and online distribution. Most companies don't have such luxuries, though, so you will likely end up doing the production work yourself. See Chapter 11, "Final preparation—production editing," for detailed information about production editing.

Editing the content—what you and your editor can expect

Before an editor or another writer starts reviewing your book, you should already have some idea of what will occur during the edit. There are three elements that establish these expectations:

- Doc plan and outlines—offer specifics about audience and content. The editor will ensure that the content follows what the doc plan and outline specify. (However, that's not to say that an outline is set in stone. Changes to the product in turn cause changes to the outline.)
- Schedule—determines the time an editor has for editing. A short turnaround time may mean that a review focuses on some issues and disregards others. Decisions of this nature should be made by project managers with input from all team members. See "The reality of time constraints" on page 162 for more information.

- Style guide—specifies the guidelines you should follow while writing. If content does not show basic adherence to style guidelines, the editor may ask you to correct style issues and resubmit the book for editing, so it's best to follow the rules from the start.

There are also specific things an editor or reviewer can expect from you and vice versa.

What an editor can expect from you

When it comes time for your content to be edited, you can't give the source files to a reviewer and assume that any problems will be fixed. At most companies, an editor or reviewer doesn't make changes to your source files for the following reasons:

- Getting comments instead of changed source files means that you have the final word on changes to your content. If you find a comment that you disagree with, you should check with the editor and figure out a compromise. If an editor or reviewer makes the changes to the source files, you do not have a chance to perform this quality control.
- Making changes based on review comments will improve your writing skills and adherence to style guidelines. For example, changing every reference to "checkbox" to "check box" may be unpleasant, but if the style guide mandates the use of "check box," making all of those changes will go a long way toward ensuring you remember to use the correct terminology in the future.

NOTE: Although writers should incorporate editing changes into their content as a general rule, editors are usually willing to help out to meet a looming deadline.

The content you turn over for editing should have the following characteristics:

- Basic adherence to correct spelling, rules of grammar, the style guide, and the template. It's quite understandable and common for a writer not to catch every error or inconsistency, but it is not acceptable for a writer to turn over content that hasn't been proofed or spell-checked. If text does not follow the basic rules, many reviewers will return it to you for review, correction, and resubmission. In many companies, the inability (or refusal) to follow style guidelines can lead to reprimands and poor performance appraisals—and possibly the loss of your job.
- Some attention paid to organization. Because a writer gets so close to content, it's often hard to see how changes in the order of paragraphs or subsections would improve the flow of information—the editor's second set of eyes can really help spot areas that could be improved with some reorganization. That being said, however, a writer should not just write material and expect a reviewer to place it in order.

Basically, paying attention to the style guide, the doc plan, and the outline—and not expecting an editor or reviewer to do your dirty work—will make life better for everyone.

The challenges of reviewing topic-based content

If your company is taking a topic-based approach to documentation, it's likely the order in which a reviewer or editor reads topics may not follow the order of topics for an actual deliverable.

Different deliverables (and even different versions of one deliverable) can be created by including or excluding particular topics. Because of all the possible topic combinations, it's tough—if not impossible—for a reviewer to get a handle on how content flows when topics are combined together for various deliverables.

Therefore, in topic-based writing environments, it may make more sense for the editing process to focus more on the consistency of how information is presented.

What you can expect from an editor

You can expect the following from an editor or a fellow writer reviewing your work:

- An electronic file with comments inserted into it, or a legible (or semilegible!) markup of a printed copy. Markup on paper should also use standard diacritical marks (Table 5 on page 158).
- A review that points out errors and makes commentary but that does not berate you. There shouldn't be color commentary such as "Why did you do that?!?!?!?" or "You fool!"
- A review that points out positive features. If you've done a good job of presenting information, the reviewer or editor should note that. Feedback doesn't have to be all negative.
- A list of issues that occur throughout the content (often called a "global list"). Instead of marking recurring errors (such as incorrect capitalization in headings) again and again, many editors will flag an error a few times and then write a comment

such as “Global. I won’t flag this again in this chapter.” Global lists reduce the amount of time a review takes. However, an editor should not abuse global lists—a global item such as “book doesn’t follow style guidelines; won’t mark all deviations” is too broad.

When you receive the global list, it’s your responsibility to go through the source files and make the changes globally. Sometimes, you can use a search-and-replace operation to make global changes, but do so with care. Using search-and-replace with little thought can lead to ugly results, which can take more time to clean up than if you had made the changes individually.

Table 5: *Some standard diacritical marks*

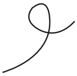
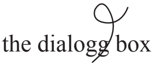
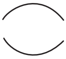
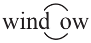
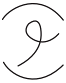
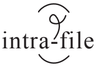
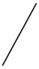
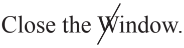
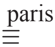
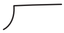
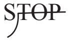
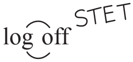
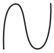
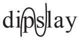
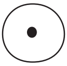
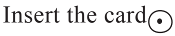
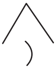
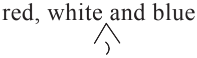
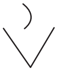
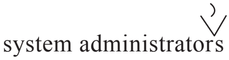
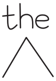
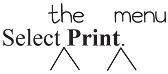
Symbol	Meaning	Example
	Delete	
	Close up	
	Delete and close up	
	Change to lowercase	
	Change to uppercase	

***Table 5:** Some standard diacritical marks (continued)*

Symbol	Meaning	Example
	Change to initial capitals	 S~~TOP~~
STET	Let it stand (that is, don't make the marked change)	 log off STET
	Transpose	 dipslay
	Insert period	 Insert the card
	Insert comma	 red, white and blue
	Insert apostrophe	 system administrators
	Insert text	 the menu Select **Print**.

Editorial checklists

Some companies give writers editorial checklists to follow before turning over a book for review. A sample checklist starts on the next page.

Editorial checklist

NOTE: This checklist is a general overview of what editors consider while editing documents. It is not meant as a substitute for the style guide. Please read the style guide and apply its guidelines while you write your document.

Readability and accessibility

___ Steps are in the imperative mood. For example: Select the **Update All** button.

___ Descriptive material is primarily in active voice. (However, don’t mangle a sentence to avoid passive voice.)

___ Latin abbreviations (for example, i.e., e.g., and etc.) are not in the text; use the English equivalents instead.

___ Text avoids gender-specific pronouns or constructions such as “he or she.” When possible, use plural forms and second person instead.

Capitalization

___ Headings and captions have sentence-style capitalization—not initial caps.

___ Menu names, button names, and the names of other GUI items reflect the capitalization on the interface.

Highlighting

___ Menu names, menu choices, and GUI items have the Menu character style applied.

___ First uses of terms have the Term character style applied.

___ Commands have the Command style applied, and variables in commands have the CommandVariable style applied.

Acronym usage

___ On first use in a chapter, an acronym is spelled out and followed by the acronym form in parentheses: local area network (LAN).

Figures and tables

___ Figures and tables have captions.

___ The text contains cross-references to each figure and table.

___ Tables are based on a table format in the template.

___ Tables spanning multiple pages have the table continuation variable in their captions.

___ Figures are inserted according to figure placement rules in the style guide.

Spell-checking

___ Document is spell-checked. Be sure that the Repeated Words, Two in a Row, Straight Quotes, Extra Spaces, Space Before, and Space After options in the Spelling Checker are on.

NOTE: The preceding checklist is for a template-based workflow. The "Highlighting" section mentions character styles the author applies to text. In a structured workflow, authors wrap the content in elements instead. For example, menu names and choices would be wrapped in the Menu element. Also, the structure would determine the correct placement of figures, as mentioned under "Figures and tables."

Working with review comments

When you receive comments from an editor or reviewer, spend a bit of time looking at the comments. Make a note of unclear comments or questions that you need clarified.

If the reviewer has suggested a change you don't agree with, discuss it with the reviewer. Often, a discussion about a change can lead to a compromise that ultimately serves the product's users better than what either you or the reviewer wanted. However, save your energy for the worthwhile battles. Arguing against rules in the style guide is a waste of everyone's time.

The reality of time constraints

Tight schedules (and a lack of resources) sometimes cut into the time available for the editing process. Although it's always best to anticipate time problems with a schedule that can tolerate some slippage (as mentioned in "Dealing with the inevitable schedule changes" on page 106), you can't always forecast every scheduling problem. If it becomes necessary to make some compromises, be sure to choose ones that have the least impact on the content. Also, if you're working with a documentation team, it's best to get everyone's input about what compromises make the most sense.

The following list describes the features of good content, starting with the most important. If you need to make sacrifices because of time, start at the bottom of the list.

- Technical accuracy—the book must describe the product correctly. (It's better to have an accurate book that contains spelling errors than an inaccurate document with pristine spelling.)
- Completeness—the book covers all features (or as many as possible).
- Organization—the content flows in a logical manner.

- Index—because the index is often the first thing a user reads in a book, having an index is important, but it's not as important as verifying technical accuracy.
- Grammar and spelling—correct grammar and spelling are essential to clear communication, but if you have to choose between editing a book for grammatical correctness and ensuring it's technically accurate, it's wiser to check for technical accuracy.
- Adherence to style guidelines—in a template-based environment, the application of style guidelines makes text more consistent and readable. That being said, having all menu names in bold type should be the least of your concerns if time isn't on your side.

  Structured workflows automatically apply style guidelines to content, so authors don't have to spend time assigning styles (or verifying they were assigned correctly in other writers' content).
- Production editing—ensuring that online help is formatted correctly and ready to be released or that you've checked line breaks, page breaks, running headers, and other items in printed content (as described in Chapter 11, "Final preparation—production editing"). While a cleanly formatted document gives an impression of professionalism, nicely formatted text that doesn't reflect the product isn't going to win you kudos from readers.

  As it does with the application of style rules, a structured workflow eliminates many production tasks, and that is yet another cost and (time) reduction that makes structured authoring a compelling choice for many documentation groups. See "The impact of structured authoring on production editing" on page 183 for details.

Just because some of the preceding items are more important than others, don't fall into the trap of thinking the less important items are always disposable. If you find you're consistently making sacrifices, you're not getting enough time or support to create good content. You should address the lack of time and resources with your manager. If you're working for a client, keep the client actively involved in making decisions about any compromises, and thoroughly document those discussions—recapping such a meeting via email is a very good idea. (Yes, this is paranoid, but a little paranoia can be a career survival strategy.)

10 Indexing

What’s in this chapter

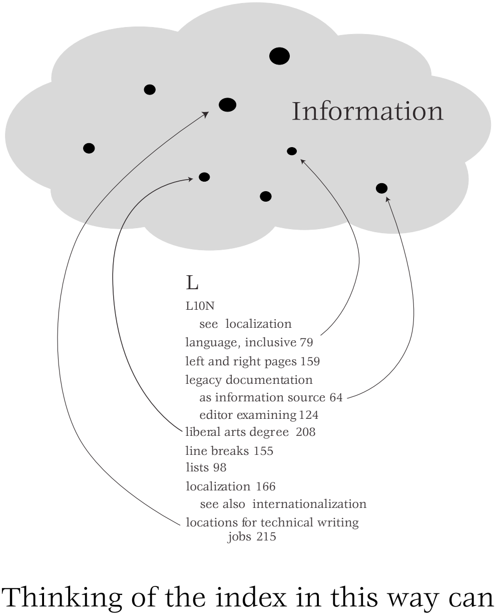
- What should I index?
- Cross-indexing
- Using primary and secondary entries
- “See” and “See also” entries
- How long should my index be?
- Editing your index
- Some helpful tips

A complete, user-friendly index is essential for useful documentation and online help—the index is one of the most heavily used components in technical content. Many users look at the index *before* they look at the table of contents.

Readers rarely read a user guide or help system in its entirety or in the order information is presented. Instead, they refer to the book or help when they are stuck and need assistance. If users can’t find an index reference to the information they need, they become frustrated. At that point, they may abandon your content altogether, call tech support, or look for other sources of information through Google or another search engine.[1]

You can minimize such failures by providing a complete index for your information. Professional indexers refer to each entry in the index as an "information access point" (Figure 23).

Figure 23: *Information access points*



Thinking of the index in this way can help you identify good index entries. Another way to view an index is to

[1] Users turning to other web resources for help instead of reading official documentation is a trend technical writers need to recognize and address. See Chapter 14, "Web 2.0 and technical communication," for more information.

compare it to the table of contents. The table of contents provides readers with a sequential list of topics in the book; it reflects the organizational scheme that the writer of the content chose (which might not be the scheme that the reader would have chosen). However, the index provides a list of topics in alphabetical order, so the readers can locate the information they need using a standard A–Z reference scheme.

Even though indexes are critical to help readers find information, very few companies employ professional indexers. The task of indexing usually falls on the writer, so having basic indexing skills is important. This chapter offers some tips on creating useful indexes.

Do I really need to index online content?
Can't the reader just do a search?

Full-text search in a help system or PDF file provides users with another method to locate information, but it is not a replacement for an index. Suppose you're using an application for the first time, and do you a search on the term you think describes an item you need clarified. If the author of the help didn't use the same term to describe that item, you'll get no results for that search. A good index handles situations like this by including variations of a term and "see also" entries; these techniques are described later in this chapter.

What should I index?

Let's start with how *not* to create an index. The following content is usually not indexed:

- The *front matter*, which includes the title page, table of contents, preface, and so on
- The *back matter*, which includes appendices, the glossary, and the index

In online help, some of this content isn't usually included, so indexing is not an issue.

NOTE: These are general rules. Sometimes, a preface or appendix contains important information that you should index.

Once you begin, don't index every occurrence of a particular word, or you'll end up with index entries that look like this:

chocolate 2, 4, 5, 7, 9, 11, 13, 16, 22, 25, 30

Instead, index only the places where important new information is provided:

chocolate 11, 16, 30

When there are three or more entries, you should break down the entries into subentries so that the reader can scan the list and find the exact topic quickly:

chocolate
  bittersweet 16
  milk 11
  mousse 30

See "Using primary and secondary entries" on page 172 for more information.

Also, don't index the name of every menu choice, dialog box, button, and so on. For example, the following entries are not useful because most readers don't look up topics by their menu location:

Special menu
  Cross-Reference 15, 18
  Equations 72–75
  Page Break 32, 60

Instead, you should generally focus on indexing definitions and tasks:

> **C**
> - cross-reference, inserting
>
> **E**
> - equation, inserting
>
> **I**
> - inserting
>   - cross-reference
>   - equation
>   - page break
>
> **P**
> - page break, inserting

Indexing definitions

Throughout technical content, you'll see information like this:

> Extensible Markup Language (XML) defines a standard for storing structured content in text files.

and then later:

> Documentation departments are increasingly adopting XML-based workflows.

The first sentence is useful. It provides information about what XML is, so a reader could look up that page and get the definition. You should provide an index entry for that sentence. But the second sentence is less useful, so it's probably not worth indexing.

If the defined term is an acronym, as in the XML example, you should also include an index entry for the acronym's spelled-out meaning:

Extensible Markup Language
  see XML

The "see" reference helps readers unfamiliar with the acronym find the information they need, and you're saved the extra work of indexing the information under both the acronym and its meaning (refer to ""See" and "See also" entries" on page 173 for more information).

If definitions are in a glossary, you don't need to index them. Because the glossary is in alphabetical order, the reader can easily find the terms there without referring to an index.

Indexing tasks

Readers use procedural information to figure out how to accomplish a task. Therefore, you should provide them with a way to access that information from the index. If you have content about printing, provide an index entry for "printing." If you have a section about replacing toner cartridges, try "replacing toner," "toner, replacing," and perhaps "ink, replacing."

However, before you index a task, consider your audience. For example, a section or topic discusses querying a database. If the audience consists of database experts, the entries "querying database" and "database, querying" should suffice. But if your audience contains users generally unfamiliar with databases, consider adding "accessing database."

Analyze who your readers are and try to determine how they might look up terms (see "Audience, audience, audience" on page 90 for information about audience analysis). For some procedures, you may need to include several synonyms for the same operation ("closing application," "exiting application," and "quitting application").

When indexing a task, be sure to index not only the action ("opening file," "closing file," and "printing file") but also the item involved in the action:

```
file
  opening 10, 34
  closing 15
  printing 55
```

Dual entries such as these reflect a good feature of indexing—cross-indexing.

Cross-indexing

Not everyone thinks exactly alike, and your index should anticipate this. By including variations of an entry, you can make sure that different readers find the information they need. (Professional indexers refer to indexes with cross-indexed entries as *permuted indexes*.) For example, suppose there is a paragraph about displaying the text that creates a document's running footer. To cover all your bases, your index could contain the following entries for that one paragraph:

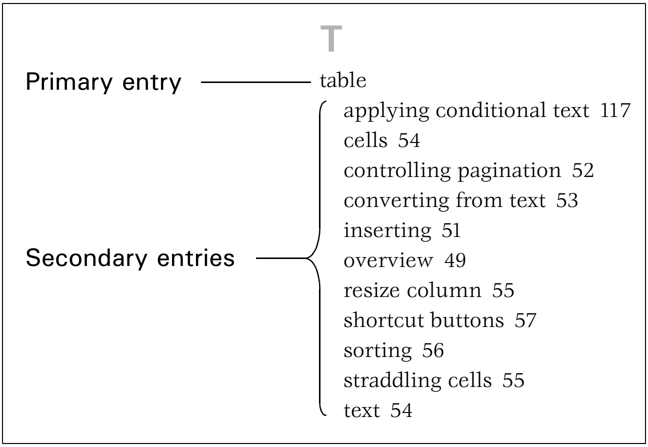
```
displaying running footer text
footer, displaying text
running footer, displaying text
text, displaying running footer
```

You will need to include these variations every time you index the topic. It's easy to forget all the variations you're using, so you can refresh your memory by generating a partial index showing the entries you've created.

Using primary and secondary entries

Primary and secondary entries organize index information into topics (Figure 24).

Figure 24: *Primary and secondary index entries*



T
table
applying conditional text 117
cells 54
controlling pagination 52
converting from text 53
inserting 51
overview 49
resize column 55
shortcut buttons 57
sorting 56
straddling cells 55
text 54

If the entries about tables weren't combined under the primary "table" entry, the reader would have to contend with something like this:

table, applying conditional text
table, cells
table, controlling pagination
table, converting from text
...

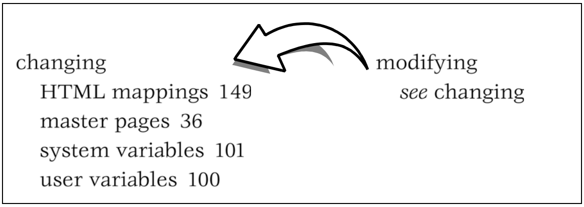
"Collapsing" all the table entries into secondary entries makes finding them much easier.

You can also create third-level entries, but it's probably best to avoid going further than that unless your book is extraordinarily long (more than 500 pages or so).

"See" and "See also" entries

In addition to cross-indexing, you can use "see" entries to be sure your index is useful to as many as people as possible. For example, some readers may look up "changing," but others may look up "modifying." Instead of including the same list of secondary entries for both terms, you could create a "see" entry that points the reader from "modifying" to "changing" (Figure 25). That way, your index works for people who look for either term, and it isn't padded with duplicate entries.

Figure 25: *"See" entry refers readers to the information they need without duplicating index entries*

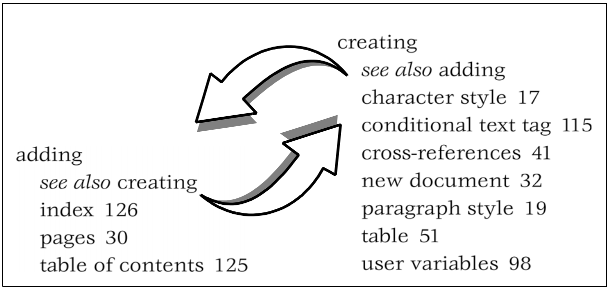


changing
  HTML mappings 149
  master pages 36
  system variables 101
  user variables 100

modifying
  see changing

A "see also" entry is similar to a "see" entry, but you use it when you have related entries in two locations (Figure 26 on page 174).

***Figure 26:** "See also" entries refer readers to related entries*



creating
- *see also* adding
- character style 17
- conditional text tag 115
- cross-references 41
- new document 32
- paragraph style 19
- table 51
- user variables 98

adding
- *see also* creating
- index 126
- pages 30
- table of contents 125

NOTE: "See" and "See also" entries may not work in online help. Before you create them, verify whether the help system and the authoring (or conversion) tool you use supports them.

Using the indexing features of your tools

To create index entries with most text processing and help authoring tools, you insert bits of code into the files—this is true for both template- and structure-based workflows. The tools require you to add the information in a particular way, so consult your program's documentation or help to learn about the correct syntax.

When you generate the index, the software scans the files for the index code. The index file includes entries based on what you typed in that coding, and the software alphabetizes the entries for you. If you're creating content for printed (and PDF) output, the page numbers for entries are automatically included.

If you create online content by converting text processing files, many conversion tools use the indexing information in those files to automatically assemble the index for the online help. Page numbers are not included because they don't apply to online help.

In general, you don't have to rebuild the index for content generated through conversion processes. If you do re-create indexes after conversion, it's time to reevaluate your processes and toolset.

How long should my index be?

The answer, as always, is "it depends." If the material is very dense and complex, the index might be longer than usual. If the information is very "fluffy," the index might be shorter. In technical content, however, the rule of thumb is that the index should contain approximately one page of two-column index entries for every 20 pages of documentation. Thus, a 100-page book would have about a 5-page index with two columns of entries (see the index of this book for an example of a two-column format).

Editing your index

Once you've tagged your index entries, you're only half done. Most indexes also require quite a bit of revision. To make the process easier, you may want to index a few chapters or topics, generate your index, and then review what you have done so far.

Here are some problems to look for:

- Cross-indexing inconsistencies:

  displaying running footer text 45, 62
  footer, displaying text 45, 62
  running footer, displaying text 45
  text, displaying running footer 45, 62

  In this example, the second page reference is missing from "running footer, displaying text."

You should also make sure that entries are cross-posted under both the action and the item receiving the action:

file
  closing 18
  opening 14
  printing 60

Confirm that you also have the entries “closing file,” “opening file,” and “printing file.”

- Page-range errors:

  querying databases 32–??

  In this case, the page range start and end tags don’t match.

NOTE: This example is taken from FrameMaker, where range errors are shown by double question marks. If you’re indexing in a different application, the problem may look a little different. Also, page ranges usually do not work in online help. If you are using a conversion tool to create online help from the files for printed books, verify how the tool handles them.

- Not enough detail in a broad page range:

  printing file 62–80

  A page range this long is useless. Go back to those pages to see where you can break down the topic into secondary entries:

  printing file
    crop marks, including 78
    ...
    print preview 62
    setup 64

Be sure to include your new subentries as their own main entries as well.

- Too much detail in subentries:

  ```
  variables
     creating 15
     deleting 16
     editing 15
     inserting in code 15
  ```

  In this case, you should probably collapse the entry into a range:

  ```
  variables 15–16
  ```

  As usual, confirm that entries exist for “creating variables,” “deleting variables,” and so on.

- Missing index entries referenced by other entries. For every “see” and “see also” reference, be sure that the entry you’re pointing to exists.

- Parallelism:

  ```
  file
     closing 32
     open 23
     saving 30
  ```

  In this example, “open” should be changed to “opening” to ensure consistency with the other subentries. Similarly, you should decide whether to refer to items as singular or plural (“file, opening” vs. “files, opening”) and be consistent among entries. Also look for consistency in the format and placement of “see” and “see also” entries.

You should review and revise your index several times before submitting it to the technical editor or peer reviewer (and don't be surprised if the editor flags a bunch of errors and inconsistencies you missed).

Some helpful tips

You may feel a little overwhelmed the first few times you index your content. Here are a few tips to make the process easier:

- Before you start indexing, be sure you understand how to insert index coding correctly. Reading the indexing section in your text processor's online help will probably save you headaches later.
- Don't put off editing your index until the end. Frequently regenerating your index and making the necessary revisions is the best way to make indexing more manageable.

- Place your index coding as close to the referenced topic as possible, usually at the beginning of the heading or sentence where the reference occurs. If the code is too far away from its intended reference, the page numbers may be incorrect in the index. You also want to ensure that if the topic is moved or deleted, the corresponding index information is automatically moved or deleted as well.
- Be careful when indexing symbols (such as \ and ;). Some symbols have special meanings in a program's indexing utility. Consult the program's manual or help for information on indexing symbols correctly.
- If you make changes to an index, be sure you change the index coding, not the entry in the generated index. Changes made directly to the index are usually lost when you regenerate the index.

11 Final preparation—production editing

What’s in this chapter

- The impact of structured authoring on production editing
- Preparing content for production editing
- Production editing for printed documents and PDF files
- Production checklists for printed and PDF content
- Preparing final output for printing and PDF distribution
- Production reviews for online content

Production editing is the process of finalizing the appearance of content. By the time content reaches the production cycle, it’s been written, edited, and proofed.

Now, it's time to verify the presentation of that content before distribution to users.

Just as the technical editing cycle ensures that content follows the style guidelines, the production editing process ensures that content follows the formatting guidelines in template-based workflows. For most text processing applications, this means verifying the correct styles from the template are applied correctly. A production edit on printed and PDF content also focuses on page elements, such as headers, footers, and page numbers, to ensure that they are placed consistently and contain the correct information throughout the book.

Like copy editing, production editing is an invisible art. When content is free of typographical errors, readers don't notice—you'll rarely hear someone say, "Wow, that book was great. It didn't have any typos in it." But readers notice typos, which make a book look unprofessional and amateurish. Similarly, production errors are noticed only when they occur, not when they are absent. Readers usually take note of inconsistent formatting and strange line breaks.

Today, few companies employ full-time, dedicated production editors. Writers often complete the production tasks—on their own work or on content created by other writers.

Writing departments that have implemented structured authoring generally do less production work, as explained in the next section.

The impact of structured authoring on production editing

In structured authoring environments, content and formatting are separated from each other. This separation means that you don't manually assign tags to content as you write it—the authoring environment automatically applies the formatting for you. (You'll read detailed information about structured authoring in Chapter 13, "Structured authoring with XML.") The tool you use to create structured content may show you what a formatted version looks like, but that's not always the case.

Because the authoring environment handles the formatting, authors and production editors don't need to do long reviews to verify that formatting has been applied correctly—the production process may be a matter of quickly reviewing output and ensuring there are no serious problems (such as missing graphics or paragraphs). If you are creating print and PDF output from structured source files, you may be able to modify unattractive page and line breaks, but that depends on the tools you are using.

Some documentation groups rely just on the automatic formatting and make few (if any) manual adjustments to output. The cost savings associated with minimizing or eliminating production reviews override any concerns about page breaks and similar formatting issues. Therefore, if you are working in a structured authoring environment, much of the remaining information in this chapter may not apply to you.

NOTE: Even for content produced in structured authoring environments, we recommend doing quick checks of output before distributing it. Human error can introduce problems into automated workflows—an author may leave a file out, forget to specify a particular option, and so on.

Preparing content for production editing

Before you complete a production review on your own content or submit the content for review by a production editor or fellow writer, keep the following tips in mind:

- Content should follow the approved template with few or no formatting overrides.
- The content should be complete: writing, technical reviews, and editing should be finished before doing production work.
- Graphics should be set up using the guidelines established by your company.

Focus on the formatting and not the content. A production editor generally has leeway to correct hyphenation and pagination problems but should not change the text. If you spot a possible problem in content while doing a production review, consult with the author before making any content changes.

Production editing for printed documents and PDF files

In an unstructured authoring workflow, the first step in production editing for printed books and PDF files is opening all the source files and applying the styles in the company template, thus removing all formatting overrides in the files. (See "The impact of structured authoring on production editing" on page 183 for information about production editing in structured authoring environments.)

The next step is to check the formatting catalogs for additions. If you're working with a production editor, expect to have any new styles challenged at this point. In a large company with a stable template, it's unlikely that you'll be allowed to add styles. In a smaller company where the template is still under development, you might be allowed to keep your custom styles, and the production editor may ask you for more information about the styles to figure out whether they should be made available to all writers as part of the standard template.

NOTE: Adhere to the template just as you adhere to the style guide.

In addition to checking compliance with the template, the production editor checks each page of a document for several items. The next few sections discuss some of those issues.

Hyphenation and bad line breaks

The production editor looks for hyphenation that might confuse the reader. These include some classics such as:

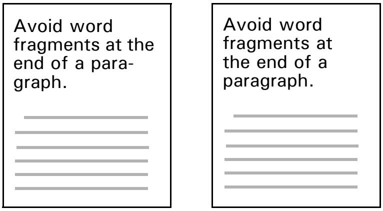
anal-ysis

When hyphenation like this occurs, the production editor will adjust the hyphenation points in a word to correct the problem. The result would be:

analy-sis

In addition to adjusting hyphenation to prevent misunderstanding, the production editor also looks at the end of each paragraph. Leaving a word fragment on the last line of a paragraph is disconcerting to the reader.
In Figure 27, the problem was corrected by forcing the word “the” in the second line to the third line. The result is a much more balanced paragraph.

Figure 27: *Avoid word fragments on their own line*



Avoid word
fragments at the
end of a para-
graph.

Avoid word
fragments at
the end of a
paragraph.

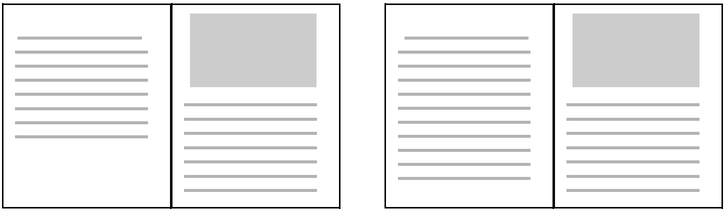
Many text preparation and word processing programs have automatic hyphenation that works really well—most of the time. In some cases, words will hyphenate across pages, or just a few syllables of a word will dangle at the end of a paragraph. Eliminate hyphenation of this nature.

Also, if there is a paragraph with a lot of hyphenation at the ends of lines, eliminating the hyphenation in one line may eliminate hyphenation of the next line. In general, it's best to avoid having two concurrent lines of text both ending with a hyphen. Some publishing packages let you prevent this by setting a preference or an attribute for the paragraph styles.

Page breaks and copyfitting

You may not have ever thought about this, but when you encounter a page in a book that's only half-filled with text, you generally assume that you're at the end of a chapter or section. If you turn the page and discover that the chapter continues (perhaps with a big figure that didn't fit in the space on the previous page), this causes some confusion. Copyfitting—adjusting how much text is on each page—can help you avoid these problems. As shown in Figure 28, you can move text from the right page over to the bottom of the left page to get rid of the empty space at the bottom of the left page.

Figure 28: *Large illustrations often cause copyfitting problems*



You also want to avoid having a heading at the bottom of the page and the introductory paragraph at the top of the next page. In most applications, you can eliminate this problem by requiring that a heading style always stay on the same page as the following paragraph.

NOTE: Do not press the **Enter** or **Return** key to create page breaks. Inserting blank lines in this manner is unnecessary in most publishing applications. Check the user guide or online help for your program to determine the correct way to force page breaks.

Other bad page breaks include a figure or table being on one page while its caption is on the next. This looks silly, but worse, it can also cause navigation problems. Because many documents' cross-references point to specific figure and table numbers, the caption needs to be on the same page as the referenced item.

Writing + fixing page breaks = waste of time

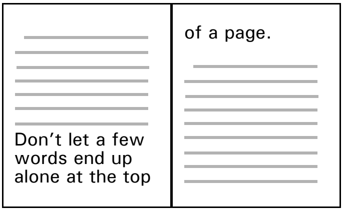
While writing content, don't worry about how pages are breaking. If you adjust pagination as you write, your efforts will be in vain—product changes and edits will mean additions and deletions to text, which in turn affect page breaks. Wait until you have finished writing and cutting in changes (in other words, wait for the production edit) to adjust pagination.

In some writing groups, setting page breaks counts as a template override, which will get you in trouble with your production editor. Keep that in mind when you're deciding whether to "tweak" your pagination.

Widows and orphans

In addition to paying attention to how paragraphs break across pages, you also need to watch how lines of text break across pages. Sometimes, the last few words of a sentence end up at the top of the page. Such a fragment is called a *widow* (Figure 29).

Figure 29: *Widows are a few words at the top of a page*



The opposite problem—the first line of a paragraph alone at the bottom of a page—is called an *orphan.*

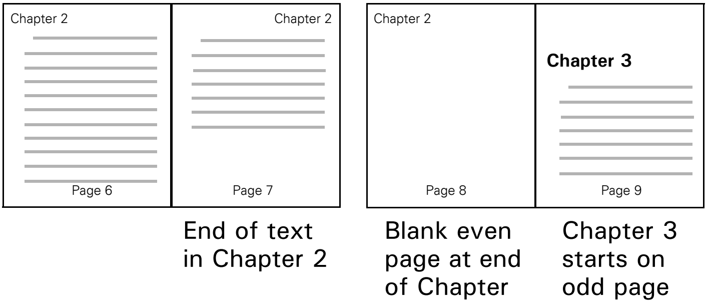
Most word processors and desktop publishing packages let you set up your paragraph styles to prevent widows and orphans. Nonetheless, you should check your pages to ensure that none sneaked into the book.

Right/left pagination

In a printed book, a chapter title page usually falls on a right page. That means that odd-numbered pages need to be on the right, and even-numbered pages need to be on the left. If a chapter's content ends on an odd page, add a blank even page (Figure 30 on page 190). Some applications automatically add blank pages to make a chapter end on an even page.

You may be required to mark blank pages with "This page intentionally left blank." Check with your editor and template designer to find out whether blank pages need special treatment in your manual.

***Figure 30:** Use a blank page to make a chapter end on an even page*



NOTE: In a PDF file that's being distributed online, odd/even pagination may not be a concern. The reader may print just a few pages or none at all, so inserting blank pages to maintain left and right pages is not necessary.

Running headers, running footers, and pagination

Running headers and *running footers* are the bits of text at the tops and bottoms of pages that contain information such as manual title, chapter name, subsection name, and page number—just look at the top and bottom of this page for an example. Before the advent of word processing software, checking footers and headers was crucial because they were inserted manually by the typesetter, who might put the wrong chapter title in a running header accidentally. Many software packages today handle this chore automatically, particularly if you have a good template, but that doesn't mean you should blindly trust the software (mostly because people using software do make mistakes). While performing a production edit, make sure that the headers and footers reflect the correct information. For instance, a common mistake is to use the chapter's running header in an appendix, which results in something like "Chapter A" instead of "Appendix A."

Just as you shouldn't completely rely on your software to generate headers and footers correctly every single time, you shouldn't assume that the software will always correctly number your pages. In some text processing programs, you can reset pagination at the start of a chapter, and if that happens accidentally, you'll end up with a second Page 1 in the middle of your book, and you probably don't want that.

So, although software really can help you focus on creating content by handling many cumbersome formatting tasks, you should always check during a production edit that the software did what you thought it would do.

Consistency in presentation of tables and figures

Even though much of a production edit focuses on text issues, you should also look at figures and tables to ensure they are consistently presented.

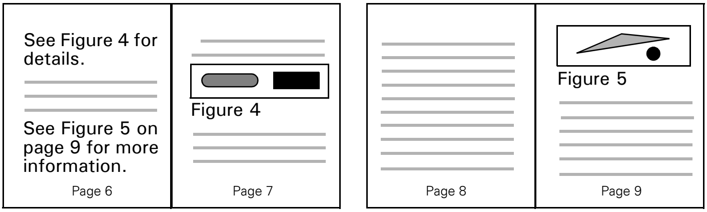
Figures should have the same amount of white space around them, and borders around them should be consistent. Tables should have similar shading and ruling. You may have two or three standard formats for your tables, and they should be used consistently. Because you can define table types in some software, it's easier today to ensure that tables look the same—just ensure that tables use the same table format.

When reviewing tables and figures, also ensure that they have numbers and captions (if your company's style mandates their usage). Also verify that cross-references to figures and tables are accurate—users get very frustrated when a step refers to the wrong figure.

Page numbers in cross-references

Once pagination is stable, check to see if cross-references to tables, figures, and the like need page numbers. Generally, if a cross-reference is on the same or facing page as the referenced item, including the page number in the reference is unnecessary (Figure 31).

Figure 31: *Cross-references to items on the same or facing page should not include page numbers*


See Figure 4 for details.
See Figure 5 on page 9 for more information.
Page 6
Figure 4
Page 7
Page 8
Figure 5
Page 9

Production checklists for printed and PDF content

Regardless of what software your company uses to develop content, there are several general things you need to look for during a production edit.

To help writers and production editors keep track of what they should check during a production edit, many companies create production checklists similar to the following:

Production checklist

NOTE: This checklist gives you a general idea of what production editors consider while editing documents.

Page-level editing

___ Are the correct styles (paragraph, character, table) used?

___ Are headers and footers correct?

___ Are heading styles sequential (heading 1 is followed by heading 2, not heading 3)?

___ Are registration and copyright symbols superscripted?

___ Are quotation marks curly instead of straight (except in computer code)?

___ Have graphics been scaled and placed uniformly, and do they have captions?

___ Do table titles and headings continue when tables breaks across pages?

___ Did you prevent hyphenation of words across two pages?

___ Is computer code in Courier font?

___ Are at least two bullets or steps together before a page break? (Avoid having one bullet or step on a line by itself before a page break.)

___ Did you prevent a page or line break between words containing hyphens, forward or backslashes, and em or en dashes?

___ Did you prevent a short word (four or fewer letters) from being on a line by itself?

___ Do cross-references to items not on the same or facing page include the page number?

___ Did you prevent hyphenation that would create 2-letter prefixes or suffixes? (Some software programs contain prefix and suffix hyphenation settings.)

___ In the table of contents and index, are page numbers on the same line as the entry? (Prevent the number from being on a line by itself when possible.)

Book-level editing

___ Do chapters start on the right-hand page (if a double-sided book)?

___ Has the book cover information been updated (title, author, version, and logo)?

___ Has the front matter been updated (copyright date and text, ISBN, credits, and part number)?

___ Are autonumbered styles numbering correctly (throughout the book or restarting in each chapter)?

___ Are index entry ranges missing (the start or end range is denoted by question marks)?

___ Do the table of contents and index have sufficient entry levels (up to three)?

Preparing final output for printing and PDF distribution

After you complete a production edit, it's time to get the files ready for the printer. Usually, this means creating a PDF file. Be sure to talk to your printer about the specifications for PDF files. Your printer may provide you with a job options file for Adobe Acrobat that ensures the PDF file is created correctly. For more information about Acrobat settings and job options, refer to the online help for Acrobat.

If you also create a PDF file to distribute to users, it's likely the PDF you give them will not be the same one you send the printer. Often, PDF files for distribution are smaller in size (which means they can be downloaded from the Internet more quickly), so graphics are compressed and somewhat reduced in quality. The PDF file you send to the printer shouldn't compress the graphics as much (or at all) to ensure the best quality printing.

Your department may have an Acrobat job options file for creating PDF files distributed to users. If a job options file doesn't exist, consider creating one so that everyone uses the same settings to create PDF files.

Production reviews for online content

Verifying the final output for online help is a bit different than reviewing the pages in a printed document. Online help doesn’t have page numbers and usually doesn’t include running headers and footers, so you don’t have to worry about those print-centric formatting items. Of course, there are some features of online help that aren’t in printed content—such as links and navigation—that you should check before releasing the content.

The following information applies to most types of online help. However, there are many different ways of providing online content, so all the advice may not be true for your company’s help.

NOTE: Many features of online help are created automatically, whether you develop your online content with a help authoring tool or with a conversion tool that converts the source files for your printed documentation. The formatting, links, navigation, and other features created by tools are generally reliable, but it’s still a good idea to check output to be sure.

Line breaks in online content

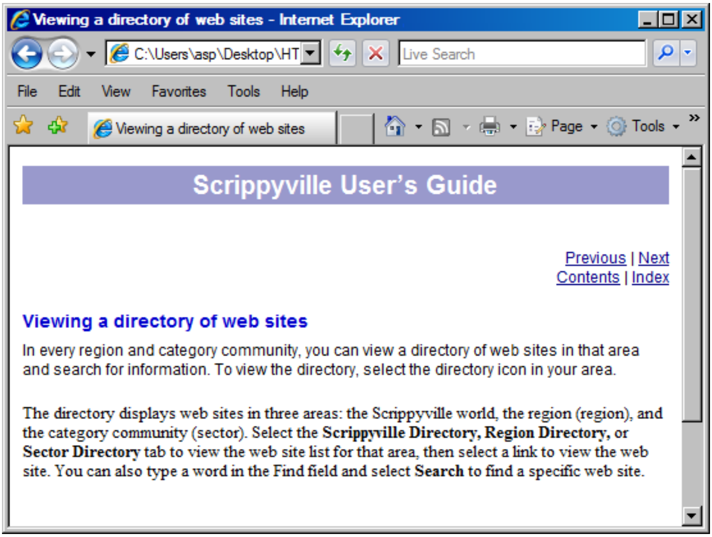
In general, you do not need to adjust for line breaks in online content. Users can adjust the window size of most online help systems and web browsers, so there is no way you can be sure at what width the content will be viewed.

Some online help systems—particularly those for hand-held devices and mobile phones—are displayed at a narrow width. Larger items such as big graphics and multicolumn tables may not display properly. In those cases, there is a good chance your department has workarounds; consult a senior writer or your manager. If there are no set methods, consider establishing guidelines to ensure consistency across the online help.

Consistent formatting

As with printed documentation, it is a good idea to verify that content is formatted consistently. For example, in Figure 32, the first and second paragraphs under the heading are in different fonts.

Figure 32: *Inconsistent paragraph formatting*



Viewing a directory of web sites - Internet Explorer
C:\Users\asp\Desktop\HT
Live Search
File
Edit
View
Favorites
Tools
Help
Viewing a directory of web sites
Page
Tools

Scrippyville User's Guide

Previous | Next
Contents | Index

Viewing a directory of web sites

In every region and category community, you can view a directory of web sites in that area and search for information. To view the directory, select the directory icon in your area.

The directory displays web sites in three areas: the Scrippyville world, the region (region), and the category community (sector). Select the **Scrippyville Directory, Region Directory,** or **Sector Directory** tab to view the web site list for that area, then select a link to view the web site. You can also type a word in the Find field and select **Search** to find a specific web site.

When you see such inconsistency, a good way to start troubleshooting is by viewing the source code for the output (if possible). If you're working with HTML-based content, you can open the file in an HTML or text editor to see the source code. Web browsers also give you the ability to view the source code. (In Internet Explorer, select the **View** menu, then **Source**.)

Seeing how the source code is different can often help you (or the person who manages the process for creating online help) figure out what's causing the problem.

Web browsers: anticipate the tangled webs they can weave

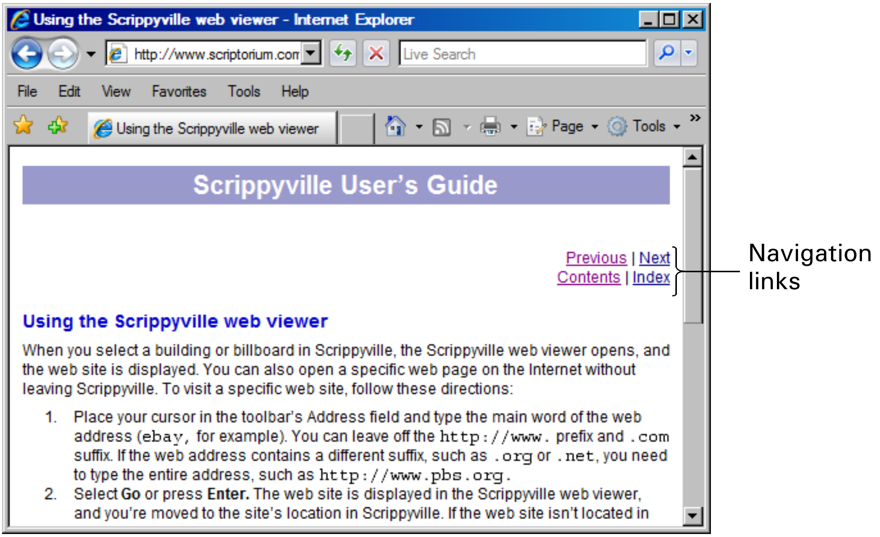
If you are creating content that users view in a web browser, verify how that content looks in different browsers and in different versions of the same browser; there is no guarantee your users will be using the latest release of a browser. Also, if the product you're documenting works on multiple operating systems, you need to verify how browsers on the different platforms show the content.

Because the display of formatting can be surprisingly different from one browser to another, you should do this sort of testing early in the process of creating the content. The differences may require you to make adjustments to your development process. You don't want to find out a particular browser isn't displaying your content correctly just before you need to release it.

Navigation and links

Before you release online help to users, test the navigation, index, and other similar features to ensure they work. Your help system may include a table of contents displayed on the left (Figure 14 on page 124), links at the top of each topic (Figure 33 on page 198), or other navigational aids.

Figure 33: *Navigation links*



Using the Scrippyville web viewer - Internet Explorer
http://www.scriptorium.com
Live Search
File Edit View Favorites Tools Help
Using the Scrippyville web viewer
Page
Tools
Scrippyville User's Guide
Previous | Next
Contents | Index
Navigation links

Using the Scrippyville web viewer

When you select a building or billboard in Scrippyville, the Scrippyville web viewer opens, and the web site is displayed. You can also open a specific web page on the Internet without leaving Scrippyville. To visit a specific web site, follow these directions:

1. Place your cursor in the toolbar's Address field and type the main word of the web address (`ebay`, for example). You can leave off the `http://www.` prefix and `.com` suffix. If the web address contains a different suffix, such as `.org` or `.net`, you need to type the entire address, such as `http://www.pbs.org`.
2. Select **Go** or press **Enter.** The web site is displayed in the Scrippyville web viewer, and you're moved to the site's location in Scrippyville. If the web site isn't located in

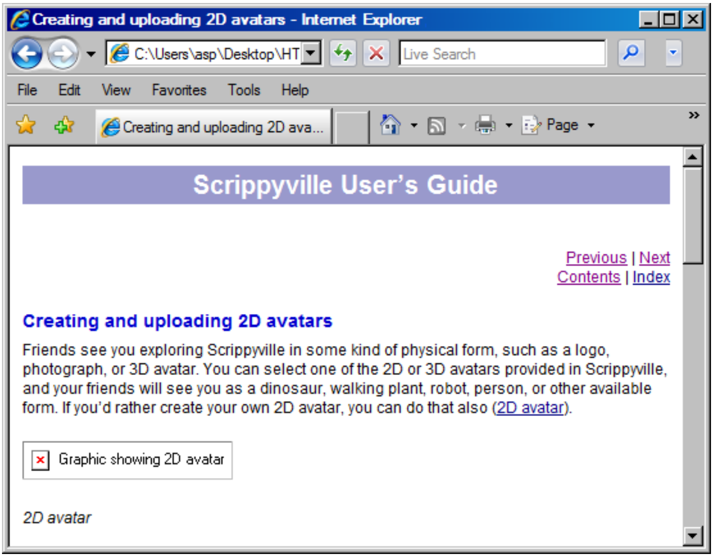
It's also wise to spot-check links in the content; those links can point to topics within the same help system, other help systems, and external web sites.

NOTE: Some help authoring tools have link-checking capabilities. Also, there are free web-based tools that will verify all the links in a web site.

Graphics and rich media

Verify that graphics are displayed correctly in your online content. Look for graphics that are distorted (like the graphic shown in Figure 22 on page 140) and for placeholders indicating missing graphics. Figure 34 shows a missing graphic in an HTML file. Internet Explorer displays an X symbol and the image's alternate text to indicate a missing image.

Figure 34: *Missing graphic in an HTML file*


Creating and uploading 2D avatars - Internet Explorer
C:\Users\asp\Desktop\HT
Live Search
File
Edit
View
Favorites
Tools
Help
Creating and uploading 2D ava...
Page

Scrippyville User's Guide

Previous | Next
Contents | Index

Creating and uploading 2D avatars

Friends see you exploring Scrippyville in some kind of physical form, such as a logo, photograph, or 3D avatar. You can select one of the 2D or 3D avatars provided in Scrippyville, and your friends will see you as a dinosaur, walking plant, robot, person, or other available form. If you'd rather create your own 2D avatar, you can do that also (2D avatar).

☒ Graphic showing 2D avatar

2D avatar

If you include rich media files in your content, check that the files play correctly. Also, verify that the *poster frame*—the still image shown when the file isn't playing—is displaying what you want it to. If not, you will probably need to modify the rich media file to display the correct frame.

12 Avoiding international irritation

What’s in this chapter

- ❖ Some basic definitions
- ❖ Did we mention audience?
- ❖ The myth about images
- ❖ Formatting that won’t hurt you
- ❖ Follow that template!
- ❖ Structured authoring and localization
- ❖ Think globally, act locally

As a technical writer, there’s a good chance you’ll write content that will be translated for other markets. The basic practices for good technical communication still apply when you write for an international audience. However, some practices that seem benign to source developers (“source” meaning the original language in which a document is written) can cause many problems with the redesign that occurs as part of the translation process. In addition, some writing techniques that may work well for informal writing or for nontechnical writing are wholly unsuitable when you write for an international audience.

This chapter provides some caveats about writing and designing content for international audiences. Although these guidelines are not exhaustive and cannot eliminate all redesign issues during translation, they can help minimize rewriting and redesign.

Some basic definitions

The term *translation* doesn’t adequately address all of the work that occurs when a product is redesigned for another market. The process involves far more than just rewriting the content in a different language. To encompass a broader set of requirements, the term *localization* was coined. Localization is the process of redesigning a product for a specific international market or *locale*. It incorporates translation, file preparation, reformatting, visual proofing, and additional tasks that ensure that the proper characters appear in the text, that the page layout works for all languages and locales, and that specific cultural references or designs are not included in the localized product.

Localization is often abbreviated L10N (which means, literally, L, then ten letters, then N). Ideally, localization involves what happens after a product is completed in its source language to make the product ready for other markets. More and more, this sequence is becoming skewed, where the localization process begins before the final version of the source product has been completed. Such scenarios often result in higher localization costs, lower product quality for localized versions, and a whole lot of frustration for everyone involved. Many localization vendors spend a significant amount of time educating clients about the localization process to avoid these problems.

One way to avoid problems during localization is to *internationalize* the product. *Internationalization* (or I18N—care to guess why?) is the practice of designing a product to be as culturally neutral as possible, accounting for issues such as language, design conventions, and tool limitations. By internationalizing your content, you can lessen the frustration that you may otherwise encounter after you finish a source product and send it through the localization process.

Did we mention audience?

As mentioned earlier in this book, technical writers must have a clear vision of the audience for whom a document is intended (see "Audience, audience, audience" on page 90). When you write for an international audience, this advice is even more crucial. Cultural factors, such as the educational system, affect the expectations of your audience. Content that works well for a U.S. audience is often inappropriate for other cultures.

In addition to differences in exposure to technology, people from different locales have different social mores and a different body of cultural knowledge. What might make sense to someone raised in the U.S. may have little relevance to someone from another locale. In the same respect, what passes for humor in one culture may be unfunny or even offensive in another culture. So technical writers need to be aware of the cultural context in which their work may be viewed and write appropriately for the situation.

Also, many of the common phrases we use to describe different geographic regions reflect our bias. Americans refer to the southwestern part of Asia as "the Middle East" and the southeastern part as "the Far East." (Ironically, from the United States, you usually fly west to get to the Far East.) Geography is relative to location, so every locale has its own "Middle East." Using more precise geographic descriptions can help you avoid such references. Avoid ethnocentric or U.S.-centric comments that sometimes creep into marketing statements or product descriptions. Although a technology may help make a U.S. business "more competitive against the Japanese," it can also work in the reverse. In a global market, writers need to adopt a global perspective.

Language is an eight-letter word

Okay, so maybe an eight-letter word doesn't sound all that offensive. However, language creates a dilemma for content developers. Language, according to George Lakoff and Mark Johnson in *Metaphors We Live By* (ISBN 9780226468013), is inherently metaphorical. The authors note just how often our everyday speech relies on figurative speech for comprehension.

The problem is that different cultures have different conventions, mores, and expectations about what is acceptable and what is not. For example, speakers of U.S. English would have no difficulty understanding the allusion in the heading of this section. An eight-letter word somehow matches a four-letter word in its offensiveness. However, to someone who speaks another language, the phrase “four-letter word” may have no significance.

In the U.S., we use an idiom to describe an event that occurs infrequently: *once in a blue moon*. Although most U.S. English speakers understand the idiom, very few actually know what a blue moon is. Nonetheless, idioms still manage to convey meaning within a locale. Outside a locale, the idiom often doesn’t work. For example, in Italy (which has a large population of Catholics) the same meaning is conveyed by the phrase, *every time the Pope dies*. Compare these to the equivalent in Ecuadoran Spanish, *every time there is an eclipse*. Each phrase is comprehensible if in context, but the immediacy of the meaning is not the same between locales. Imagine how the following U.S. English idioms might sound to non-U.S. ears:

- I got it from the horse’s mouth.
- He kicked the bucket.
- It’s like finding a needle in a haystack.
- That fellow is two bricks short of a load (or “dumber than a sack of hammers,” “a taco short of a combination plate,” and so on).

Idioms are based in regional and national dialects, and as such, they create problems for translators and for audiences.

More on mores

Beyond the metaphoric or figurative aspects of language, different cultures *expect* varying degrees of formality. In the U.S., audiences are accustomed to informal social interactions, using first names with people we barely know (if we know them at all), and interjecting humor into subjects that are normally dry and uninspiring. Other cultures are far more formal. Using first names in the workplace may be considered acceptable to U.S. employees, but such behavior in many European and Asian countries would be overly familiar and disrespectful. In many locales, coworkers are expected to refer to people by surnames and formal titles.

Hand-in-hand with the level of formality is the appropriateness of humor. Audiences in many non-U.S. locales expect technical subjects to be, well, technical and consider a serious tone to be a display of respect on the writer's part. Humor in such publications might strike these audiences as condescending or patronizing.

Some common-sense rules for international writing

Being aware of the potential for problems is only half the battle, so here are some methods you can use to avoid problems. Some of these ideas have already been covered in this book.

Use clear, unambiguous, and consistent terminology

Use one term for a concept and focus on clarity and simplicity. As long as the meaning of a sentence is clear, a translator can work with it. Using a term as more than one part of speech (such as “press Enter” and “Enter your password”) is likely to create confusion, so use a particular term for one meaning and as one part of speech (that is, always a noun or always a verb). Also, use common, basic terms that get your point across. Using an extensive vocabulary and academic-sounding prose is not likely to help you communicate more effectively, and the more subtle you try to get with language, the more likely those subtleties will be lost on your audience.

Really not good: If the printing device fails to initiate the startup sequence within an ordinate amount of time, verify whether it is necessary to reestablish any power or bidirectional connectivity.

Better: If the printer does not start immediately, confirm that it is plugged in and connected to the computer.

Use simple sentence constructions

Simple sentence structures provide fewer chances for translators to misunderstand your point. Also, make sure you include all optional parts of speech, including relative pronouns. These syntactic clues give translators more on which to base their decisions.

Not so good: If, when the application finishes saving the file, you want to close the application, you can do so by selecting **Exit** from the **File** menu.

Better: To close the application after it finishes saving the file, select the **File** menu, then **Exit.**

Don't use culture-specific metaphors, idioms, or allusions

As noted before, different cultures share different worldly experiences. Metaphors, idioms, and allusions are all part of that experience, so these linguistic devices often do not translate well into other languages or local conventions. Focus as much as possible on the literal meaning of your words to make your point. For example, write "discover the cause of a problem" instead of "ferret out the problem."

Don't refer to celebrities, sports, or politics

Celebrities in one country aren't necessarily well known elsewhere, and references to sports are lost in cultures where the athletic activities differ. For example, to a U.K. resident, American baseball looks a lot like a game called rounders. And the term "football" is used for at least three very different games internationally.

What people watch at the movie theater also depends on who distributes films in their region. Depending on cultural constraints imposed by the government, local tastes, or other factors, what appears on film or television and how nonnative films or programs may be promoted can vary considerably. (For example, in Beijing, the film *Nixon* was screened as *Nixon: The Big Liar.*)

Politics is simply a good topic to avoid because your audience will likely include people of widely divergent political beliefs.

Don’t interject humor

Like metaphors, idioms, and allusions, humor is also dependent on culture, and what people in one culture find funny may simply be ludicrous or downright offensive in another culture.

Don’t use first names or gender references in examples

In many cultures, people use only titles or surnames in the workplace. In countries where surnames can reveal caste or religious affiliation, some combinations of names could offend readers.

Instead, use titles or roles in examples. Using plural terms instead of singular can also help eliminate gender references.

Not good

Kathy, Bob, and Sharmilla

Still risky

Ms. Jones, Mr. Renshaw, and Dr. Srinivas

Better

The system administrator, the database administrator, and the senior network engineer

Hey, this book is full of humor and culture-specific references. The authors don't follow their own rules!

We freely admit this book doesn't follow the guidelines outlined in this chapter—and that's because this book is not a user manual translated into multiple languages.

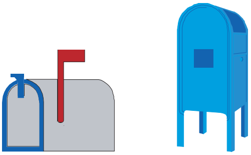
Culturally neutral writing is more easily localized, but it can also be dull and dry. We want this book to be informative *and* entertaining, and we wrote it for technical writers who create content for a Western, U.S.-centric audience. It's difficult (if not impossible) to write material that is both audience-specific and culturally neutral.

The myth about images

"A picture is worth a thousand words" may hold true in some situations, but words are far easier to translate. Many writers are unaware of the part culture plays in developing visual literacy. How audiences interpret graphic images depends on their native environments.

Take, for example, the types of icons that have appeared on email applications (Figure 35).

Figure 35: *U.S.-centric postal images*



Outside the U.S., these icons are not so easy to identify.[1] Most Europeans have never encountered mailboxes like these. For some audiences, the closest match they can find is a railroad tunnel.

Much ado about taboo

Another important fact to consider is that some cultures have taboos about how people are represented in images, especially when those images represent people of different genders. Some countries have guidelines about appropriate dress and behavior for people of different genders. These restrictions can include the following:

- Western-style dress for women
- Length of facial hair for men
- Physical position of women and men when interacting
- Position of hands or feet in relation to the perspective of the audience

[1] Even if users outside the U.S. understand what the mailbox icons mean, the use of decidedly American imagery on an internationally distributed application may cause resentment.

For example, placing the soles of feet or the palms of hands toward the audience is construed in some cultures as an insult. Because the left hand is considered unclean in many countries, displaying it in images can be inappropriate, depending on the context. If the audience may be offended by the representation, you need to respect that cultural perspective. Imagery that is inappropriate for a particular culture will distract the readers, which will make the content less effective.

Keep in mind that color can also communicate messages, some of which may be inadvertently interjected into your content if you are not aware of them. Whereas a hand with the palm facing toward the viewer can be offensive in some cultures, a red hand (often used to indicate critical warnings or cautions) can signify death. In the U.S., white is associated with weddings and purity, but in Japan, the color white signifies mourning. Although it's not desirable to eliminate all color from content, you must be aware of the potential for communicating unintended messages.

Screening your graphics

To make the most of your internationalization efforts, work with your localization provider to ensure that the content you deliver (graphic and otherwise) is appropriate for all audiences. Translators are typically natives of the locales they translate for, and they take pains to stay aware of the mores for their respective cultures. Requesting an analysis of the graphic content (or all content) from your localization provider before you begin writing can help ensure that the content you deliver meets the needs of all audiences without offending any of them.

Formatting that won’t hurt you

Layout and formatting decisions made during the development of the original source can have a significant impact on the localization process. Different languages have different requirements for space, and different locales have different conventions for layout and formatting. Keeping these differences in mind can help you avoid adding time to an already-tight schedule.

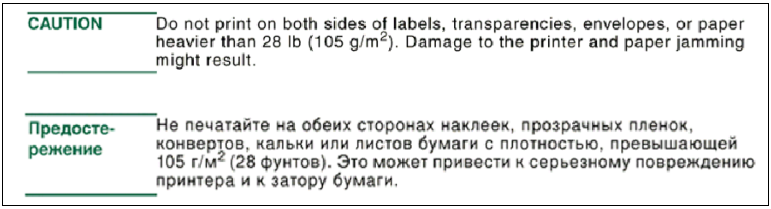
Text expansion

As a rule, translated content may expand up to 20 or 30 percent in length: English is a relatively compact language compared to many others. Shorter language components (words and phrases) in a target language can be double the length of the English source.

Text expansion complicates matters when template developers do not account for it in template design, and adjusting layouts for the increase can add many hours to a localization desktop publishing schedule. Desktop publishing is usually one of the most expensive aspects of the localization process.

To avoid additional localization costs, design paragraph styles to allow expansion. In Figure 36 on page 214, the margin for the text doesn’t account for expansion in the Russian translation, forcing the Russian side-heading text to wrap.

Figure 36: *Text expansion in side headings*



CAUTION

Do not print on both sides of labels, transparencies, envelopes, or paper heavier than 28 lb (105 g/m^2). Damage to the printer and paper jamming might result.

Предостережение

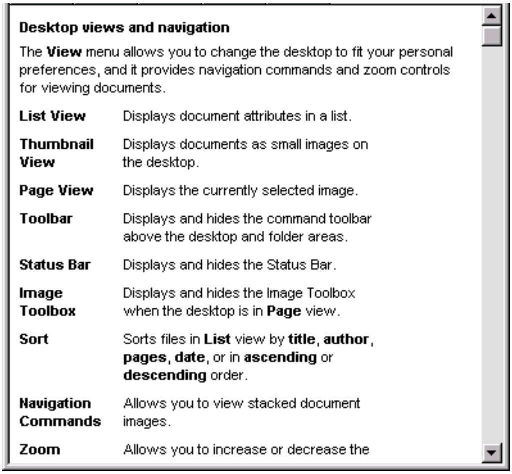
Не печатайте на обеих сторонах наклеек, прозрачных пленок, конвертов, кальки или листов бумаги с плотностью, превышающей 105 г/м^2 (28 фунтов). Это может привести к серьезному повреждению принтера и к затору бумаги.

A more localization-friendly approach for this style would be to have the side heading text as run-in text (that is, no hanging indentation) or as a heading above the body of the cautionary text.

Text expansion also causes problems with text in tables, especially if you're designing online help. The problem occurs when short text strings are used in narrow columns. As noted before, a single term or phrase can double in length. In this respect, shorter text strings cause more problems because hyphenation can occur more than once in long words. For some layout applications, columns expand automatically, but in online help and HTML, fixed width table cells have to be resized manually. Depending on how many tables you use, you can add considerable work to the localization schedule.

In Figure 37, some commands take up the entire column width.

Figure 37: *Table in a help file: No room for expansion*



Desktop views and navigation

The **View** menu allows you to change the desktop to fit your personal preferences, and it provides navigation commands and zoom controls for viewing documents.

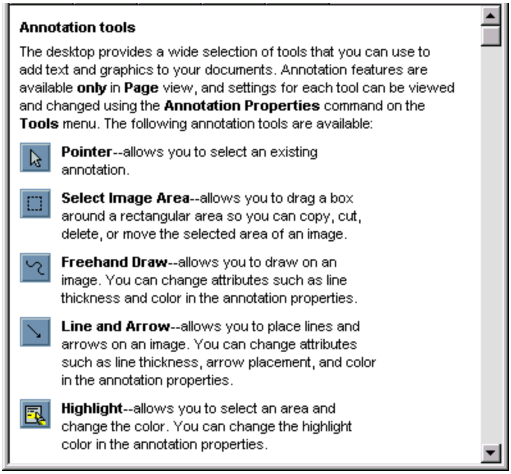
List View	Displays document attributes in a list.
Thumbnail View	Displays documents as small images on the desktop.
Page View	Displays the currently selected image.
Toolbar	Displays and hides the command toolbar above the desktop and folder areas.
Status Bar	Displays and hides the Status Bar.
Image Toolbox	Displays and hides the Image Toolbox when the desktop is in **Page** view.
Sort	Sorts files in **List** view by **title**, **author**, **pages**, **date**, or in **ascending** or **descending** order.
Navigation Commands	Allows you to view stacked document images.
Zoom	Allows you to increase or decrease the

In addition to forcing manual adjustment of the column, such designs also force the window parameters to change to allow the table to expand to the right.

Some table layouts don't cause expansion problems, primarily when they're designed without short text strings in narrow columns. As long as the text has room to expand, the column width is acceptable.

In Figure 38 on page 216, the left column contains icons. Because these do not require room for expansion, the table requires no adjustment.

Figure 38: *Table in a help file: Acceptable for expansion*



Annotation tools

The desktop provides a wide selection of tools that you can use to add text and graphics to your documents. Annotation features are available **only** in **Page** view, and settings for each tool can be viewed and changed using the **Annotation Properties** command on the **Tools** menu. The following annotation tools are available:

Pointer--allows you to select an existing annotation.

Select Image Area--allows you to drag a box around a rectangular area so you can copy, cut, delete, or move the selected area of an image.

Freehand Draw--allows you to draw on an image. You can change attributes such as line thickness and color in the annotation properties.

Line and Arrow--allows you to place lines and arrows on an image. You can change attributes such as line thickness, arrow placement, and color in the annotation properties.

Highlight--allows you to select an area and change the color. You can change the highlight color in the annotation properties.

Other problems with tables

A common technique in troubleshooting guides is using if/then scenarios (that is, “If you see this problem, then do this action”). This approach works fine as long as the text occurs in a standard paragraph flow. However, writers often use the same structure to sort lists of procedures. Basing any kind of table on a syntactical construction is likely to cause headaches during localization.

The example in Figure 39 might work well for English, but not all languages use the same word order as English. Some put verbs first or place both subject and object before the verb. How languages combine clauses also varies. During translation, the text in the headings could shift between columns and even between column headings and body cells.

***Figure 39:** Avoid using syntax to organize tables*

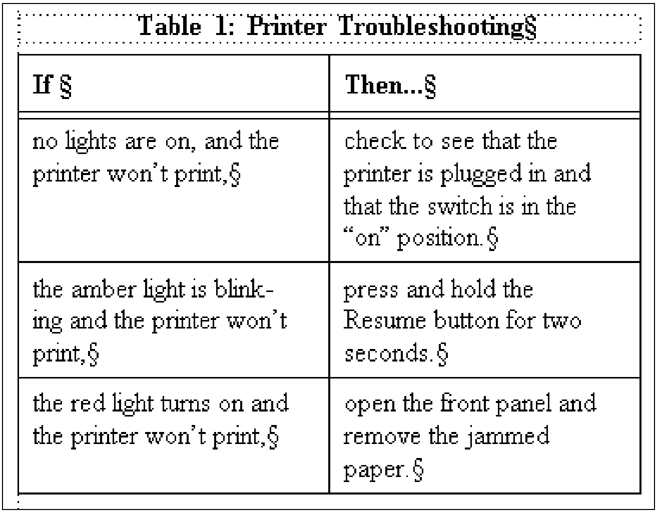


Table 1: Printer Troubleshooting§

If §	**Then...§**
no lights are on, and the printer won’t print,§	check to see that the printer is plugged in and that the switch is in the “on” position.§
the amber light is blinking and the printer won’t print,§	press and hold the Resume button for two seconds.§
the red light turns on and the printer won’t print,§	open the front panel and remove the jammed paper.§

In Figure 40 on page 218, the headings have been replaced with nouns that categorize the items in the column. This design works in any language and causes fewer linguistic contortions. Remember that languages vary in syntactical structure and that there is no correlation between word length in one language and word length in another.

Figure 40: *Using nouns for headings—a much better choice*

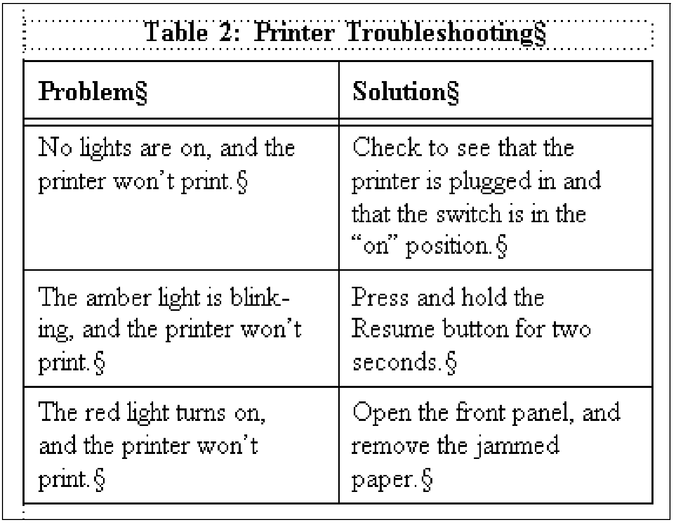


Table 2: Printer Troubleshooting§

Problem§	Solution§
No lights are on, and the printer won't print.§	Check to see that the printer is plugged in and that the switch is in the "on" position.§
The amber light is blinking, and the printer won't print.§	Press and hold the Resume button for two seconds.§
The red light turns on, and the printer won't print.§	Open the front panel, and remove the jammed paper.§

Follow that template!

If you're working in a template-based authoring environment, following the template can save a lot of time and expense during localization desktop publishing work. Styles give you global control over the look of paragraphs and characters, and a change that could affect dozens of paragraphs in dozens of files can be changed with a single button click instead of an hour or more of manual reformatting. Avoid using layout overrides (such as changing a single heading paragraph to fall after a page break). Instead, design specific styles for format variations. Use character styles for changes in the dominant font for a paragraph. Style overrides (for example, making a word bold or italic by selecting a button on the toolbar or a choice on the **Font** menu) can create a lot of manual reformatting. The more global you make the style control, the easier your documents will be to localize.

Structured authoring and localization

A big reason companies implement structured authoring is the automatic assignment of formatting. Desktop publishing expenses are drastically reduced because authors don't spend time formatting content.

Those savings become even more compelling when content is localized; you reduce the desktop publishing expense for each translated version. If your content is in English and six other languages, you cut desktop publishing expenses seven times—and release the content more quickly in all languages because you no longer need large blocks of time for desktop publishing tasks.

For details about structured authoring, see Chapter 13, "Structured authoring with XML."

Think globally, act locally

Even if your company's content isn't localized now, write with an international audience in mind. Avoiding slang, jargon, and culture-specific references makes your text more accessible to nonnative English speakers—and easier to translate when localization is a requirement.

13 Structured authoring with XML

What's in this chapter

- What is structured authoring?
- What is XML?
- The impact of structured authoring and XML on writers
- Structured authoring and single sourcing
- Structured authoring with DITA

Documentation departments increase productivity by implementing structured authoring, which enforces a structure for content and automatically assigns formatting.

Many structured authoring workflows are based on Extensible Markup Language (XML), which is a standard for defining the structure of content. This chapter explains structured authoring and XML, and it describes how they affect the documentation process—and you as a technical writer.

What is structured authoring?

Structured authoring is a publishing workflow that defines and enforces consistent organization of information.

Consider, for example, a simple structured document—a recipe. A typical recipe requires several components: a name, a list of ingredients, and instructions. The style guide for a particular cookbook states that the list of ingredients should always precede the instructions. In an unstructured authoring environment, the cookbook editor must review the recipes to ensure that the author has complied with the style guideline. In a structured environment, the recipe structure *requires* the specified organization.

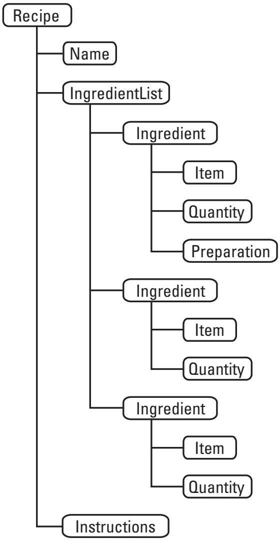
NOTE: As a new writer, you generally don't need to know how to create or modify a structured definition document, which can be quite complex. Instead, you just write content based on the rules established by the document.

Elements and hierarchy

Structured authoring is based on elements. An *element* is a unit of content; it can contain text or other elements. You can view the hierarchy of elements inside other elements as a set of nodes and branches.

Elements can be organized in hierarchical trees. In a recipe, the ingredient list can be broken down into ingredients, which in turn contain items, quantities, and preparation methods, as shown in Figure 41.

Figure 41: *Recipe hierarchy*



The element hierarchy allows you to associate related information explicitly. The structure specifies that the IngredientList element is a child of the Recipe element. The IngredientList element contains Ingredient elements, and each Ingredient element contains two or three child elements (Item, Quantity, and optionally Preparation).

In an unstructured, formatted recipe, these relationships are implied by how the type looks—for example, the recipe name is in large bold type, and the ingredient list items are in smaller type. The publishing software, however, does not capture the hierarchical relationship between the recipe name and the ingredient list—or the relationships of the components that make up each ingredient list item (item, quantity, and preparation).

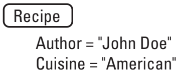
In structured documents, the following terms denote hierarchy:

- Tree—The hierarchical order of elements. (By the way, it's common to think of a family tree when considering the hierarchy of elements.)
- Branch—A section of the hierarchical tree.
- Leaf—An element with no descendant elements. Name, for example, is a leaf element in Figure 41 on page 223.
- Parent/child—A child element is one level lower in the hierarchy than its parent. In Figure 41 on page 223, Name, IngredientList, and Instructions are all children of Recipe. Conversely, Recipe is the parent of Name, IngredientList, and Instructions.
- Sibling—Elements are siblings when they are at the same level in the hierarchy and have the same parent element. Item, Quantity, and Preparation are siblings.

Element attributes

You can store additional information about the elements in attributes. An *attribute* is a name-value pair that is associated with a particular element. In the recipe example, attributes might be used in the top-level Recipe element to provide additional information about the recipe, such as the author and cuisine type (Figure 42).

Figure 42: *Attributes capture additional information about an element*



Attributes provide a way of further classifying information. If each recipe has a cuisine assigned, you could easily locate all Greek recipes by searching for the attribute. Without attributes, this information would not be available in the document. To sort recipes by cuisine in an unstructured document, a culinary expert would need to read each recipe.

When structured content is stored in a database, the attributes provide a way to search for specific information—think of attributes as information about information (also known as *metadata*).

Formatting structured documents

While writing a structured document, you don't apply paragraph formats to text as you do in unstructured word processing programs. When you put text in an element, formatting is assigned based on the element. Some applications show the formatting as you create content (like a word processor); other applications apply the formatting later when the content is processed for output.

What is XML?

Extensible Markup Language (XML) defines a standard for storing structured content in text files. The standard is maintained by the World Wide Web Consortium (W3C).[1]

XML is related to other markup languages, including:

- Hypertext Markup Language (HTML), which you may have used to build web pages. HTML is one of the simpler markup languages; it has a very small set of tags.
- Standard Generalized Markup Language (SGML), which is a heavy-duty markup language. Implementing SGML is an enormous undertaking. Because of this, SGML's acceptance has been limited to industries producing large volumes of highly structured information (for example, aerospace, telecommunications, and government).

XML is a simplified form of SGML that's designed to be easier to implement.[2] Publishing applications require complex frameworks to represent the structures of technical documents—including parts catalogs, training manuals, reports, and user guides—and XML can provide those structures.

[1] Detailed information: www.w3.org/XML

[2] SGML vs. XML details:www.w3.org/TR/NOTE-sgml-xml-971215

How are XML and structured authoring related?

Structured authoring is a concept. XML is a technology (or, more precisely, a specification) that lets you implement structured authoring using plain text files. The terms XML and structured authoring are often used almost interchangeably.

XML syntax

XML is a markup language, which means that content is enclosed by tags. In XML, element tags are enclosed in angle brackets:

```
<element>This is element text.</element>
```

A closing tag is indicated by a forward slash in front of the element name.

Attributes are stored inside the element tags:

```
<element my_attribute="my_value">This is element
text.</element>
```

Unlike HTML, XML does not provide a set of predefined tags. Instead, you define your own tags and the relationships among the tags. This makes it possible to define and implement a content structure that matches the requirements of your information. Figure 43 on page 228 shows an XML file that contains a recipe. Compare it to the recipe hierarchy shown in Figure 41 on page 223.

Figure 43: *A recipe in XML*

```
<Recipe Cuisine = "Italian" Author = "Unknown">
   <Name>Marinara Sauce</Name>
   <IngredientList>
      <Ingredient>
         <Quantity>2 tbsp.</Quantity>
         <Item>olive oil</Item>
      </Ingredient>
      <Ingredient>
         <Quantity>2 cloves</Quantity>
         <Item>garlic</Item>
         <Preparation>minced</Preparation>
      </Ingredient>
      <Ingredient>
         <Quantity>1/2 tsp.</Quantity>
         <Item>hot red pepper</Item>
      </Ingredient>
      <Ingredient>
         <Quantity>28 oz.</Quantity>
         <Item>canned tomatoes, preferably San Marzano</Item>
      </Ingredient>
      <Ingredient>
         <Quantity>2 tbsp.</Quantity>
         <Item>parsley</Item>
         <Preparation>chopped</Preparation>
      </Ingredient>
   </IngredientList>
   <Instructions>
      <Para>Heat olive oil in a large saucepan on medium. Add
      garlic and hot red pepper and sweat until fragrant. Add
      tomatoes, breaking up into smaller pieces. Simmer on
      medium-low heat for at least 20 minutes. Add parsley,
      simmer for another five minutes. Serve over long pasta.
      </Para>
   </Instructions>
</Recipe>
```

XML is said to be *well-formed* when basic tagging rules are followed. For example:

- All opening elements have a corresponding closing element, and empty elements use a terminating slash:

```
<element>This element has content</element>
<empty_element />
```

- Attribute information is enclosed in double quotes:

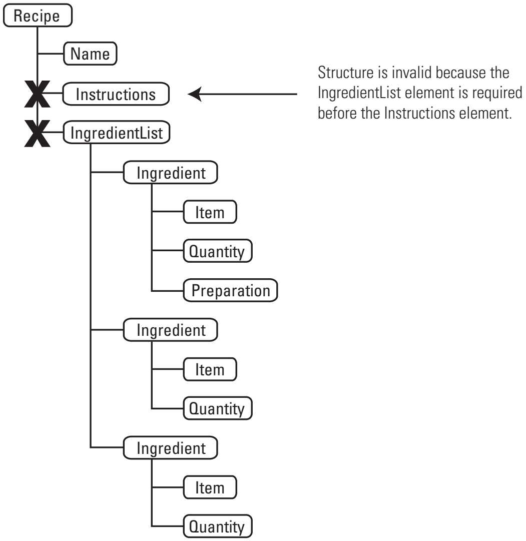
```
<element attribute="name">This is a legal
attribute</element>
<element attribute=name>This is not well-
formed.</element>
```

- Tags are nested and do not “cross over” each other:

```
<element>This is <strong>correct.</strong>
</element>
<element>This is <strong>not correct.</element>
</strong>
```

XML is said to be *valid* when the structure of the XML matches the structure specified in the structure definition. When the structure does not match, the XML file is *invalid* (Figure 44 on page 230).

Figure 44: *Invalid structure*



Defining structure in XML

Just as structured authoring relies upon a document that defines the hierarchy of elements, XML documents are based upon a definitions file. For example, a Recipe element definition might read as follows:

```
<!ELEMENT Recipe (Name, History?, IngredientList,
Instructions)>
```

The preceding definition states that Recipe element contains a required Name element, an optional History element, and required IngredientList and Instructions elements.

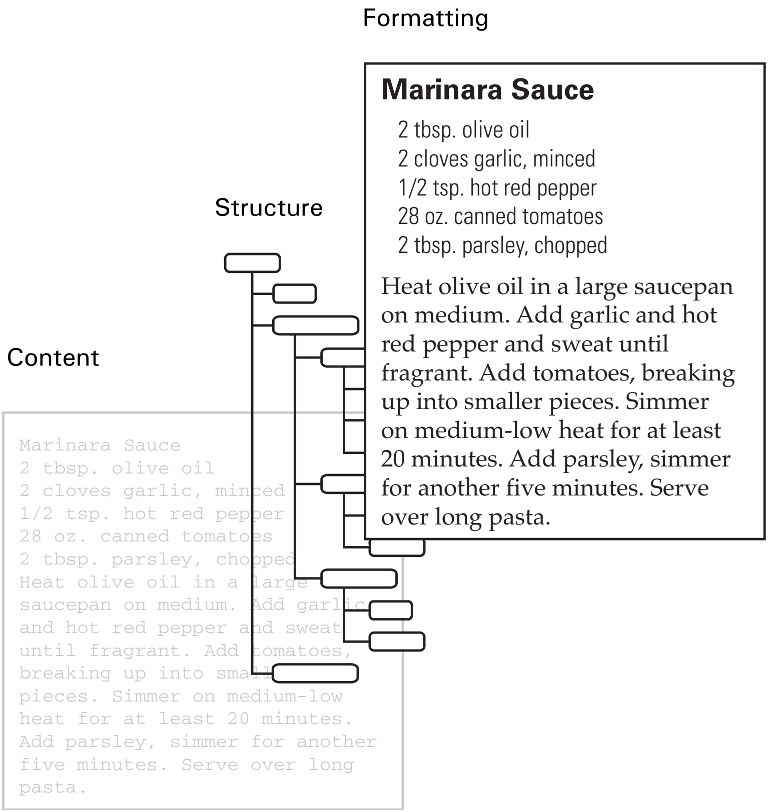
Unlike the definitions file for a structured authoring tool, the file for XML documents does not contain any formatting information because XML is text based. However, when you open XML content in some authoring tools, the tool recognizes all the elements *and* knows what formatting to apply because the authoring tool assigns formatting based on element names.

The impact of structured authoring and XML on writers

XML and structured authoring result in a completely different way of looking at information. Instead of following the familiar page- and paragraph-based method, structured authoring requires that writers consider information as a hierarchy with a separate formatting layer (Figure 45 on page 232).

As mentioned in previous chapters of this book, writers have fewer formatting and publishing responsibilities in structured workflows—these tasks are generally automated by the structure definition. You assign an element to text, and formatting is automatically applied based on the element's definition. You no longer need to worry about applying paragraph tags or knowing which tags are allowed in particular contexts.

Figure 45: *Representing a document as a series of layers*


Formatting
Marinara Sauce
2 tbsp. olive oil
2 cloves garlic, minced
1/2 tsp. hot red pepper
28 oz. canned tomatoes
2 tbsp. parsley, chopped
Heat olive oil in a large saucepan on medium. Add garlic and hot red pepper and sweat until fragrant. Add tomatoes, breaking up into smaller pieces. Simmer on medium-low heat for at least 20 minutes. Add parsley, simmer for another five minutes. Serve over long pasta.
Structure
Content
Marinara Sauce
2 tbsp. olive oil
2 cloves garlic, minced
1/2 tsp. hot red pepper
28 oz. canned tomatoes
2 tbsp. parsley, chopped
Heat olive oil in a large saucepan on medium. Add garlic and hot red pepper and sweat until fragrant. Add tomatoes, breaking up into smaller pieces. Simmer on medium-low heat for at least 20 minutes. Add parsley, simmer for another five minutes. Serve over long pasta.

Instead of wrestling with content rules and formatting problems, you focus on content and organization. Working within a structure increases your productivity and improves the quality and consistency of the final output.

NOTE: You probably won't work directly in XML files as a new writer. Most likely, you'll use a structured authoring tool that shows you the structure of content.

As you get more comfortable with structure and working with tagging languages, you may edit XML content directly in a text editor.

Structured authoring and single sourcing

Single sourcing—the process of creating multiple deliverables from one set of files—goes hand-in-hand with structured authoring. Many of the higher-end single-sourcing tools that completely automate the conversion of document processing files into online help formats work just as well with element-based content as with unstructured content.

The consistency created by structure rules is a big plus when it comes to single sourcing. To get the cleanest conversion possible during single sourcing, there can be no overrides to formatting in document files. Validated structured content can minimize and even eliminate such overrides because authoring tools enforce the defined structure and automatically apply formatting.

Also, you can transform XML content into online formats without the use of a third-party conversion tool. Extensible Stylesheet Language (XSL) enables the conversion of XML content into HTML, HTML Help, and other formats. Setting up the XSL templates for the conversion process is difficult. However, once the process is set up, conversion is highly automated, and there is no additional software cost. An XSL template consists of free, open source (but not necessarily simple) code, and there are many freeware XSL processors. The next section describes a structured authoring workflow that relies on XSL templates.

Structured authoring with DITA

Creating a structured authoring environment is a lot of work. Before a documentation group even begins with implementation, it must analyze content to figure out what the structure should be. This effort to analyze content is significant—and it doesn't include the time it takes to implement the structured workflow.

The Darwin Information Typing Architecture (DITA) is an open source, XML-based specification that supports software documentation efforts.[1] Some documentation groups find that the DITA structure is a reasonable match for their content, so they can bypass most (if not all) of the content modeling effort by adopting that standard.

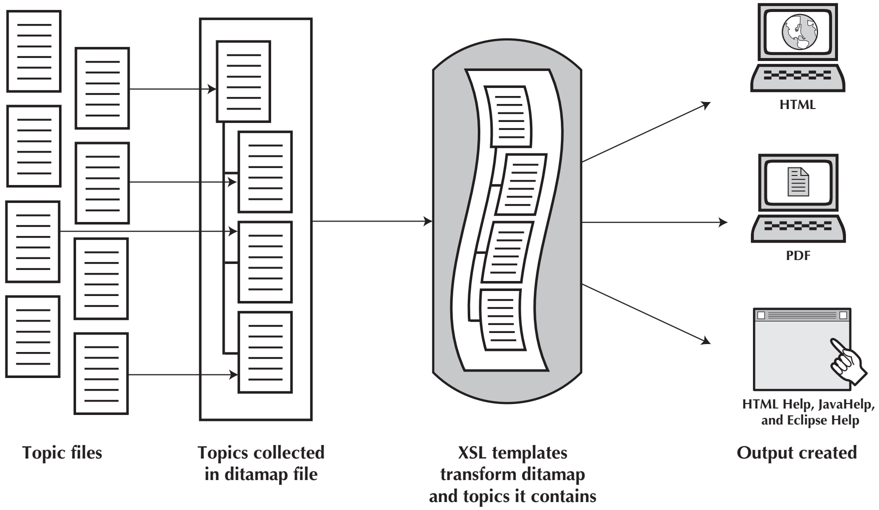
The widely used (and free) DITA Open Toolkit contains the files you need to implement a structured authoring environment based on the DITA standard. The toolkit includes the files that define the structure and the XSL templates that transform the XML content into output including HTML, PDF, and three types of online help.[2]

Figure 46 shows an overview of creating content with the DITA Open Toolkit, and the sections that follow describe the process in more detail.

[1] Detailed information: dita.xml.org

[2] Download the Open Toolkit: dita-ot.sourceforge.net

Figure 46: *Overview of DITA process*


HTML
PDF
HTML Help, JavaHelp, and Eclipse Help
Topic files
Topics collected in ditamap file
XSL templates transform ditamap and topics it contains
Output created

Writing topics

In a DITA environment, you write content as topics, as explained in “Topic-based writing and its benefits” on page 103.

The DITA structure defines four kinds of topics:

- Topic—provides a generic structure for information
- Concept—contains background information and examples
- Task—includes procedures (“how to” information)
- Reference—describes commands, parameters, and other features

Figure 47 shows the structure of a task topic:

Figure 47: *Sample topic provided in the Open Toolkit*

```
<task id="shovellingsnow" xml:lang="en-us">
<title>Shovelling snow</title>
<taskbody>
<context>
<p>Keep your driveway and sidewalks clear of snow and ice by
shovelling after any snowfall.</p>
</context>
<steps>
<step><cmd>Get the shovel out of the garage</cmd></step>
<step><cmd>Shovel the driveway, starting at the garage door
and working out to the street.</cmd></step>
<step><cmd>Shovel the sidewalk in front of your house.</cmd>
</step>
<step><cmd>Shovel the walk to your front door.</cmd></step>
</steps>
</taskbody>
<related-links>
<link href="../concepts/snowshovel.xml" format="dita"
type="concept"><linktext>Snow shovel</linktext></link>
</related-links>
</task>
```

To see samples of the other types of topics, download the Open Toolkit, which contains a directory of sample files.

NOTE: Some documentation groups modify the default structure of DITA topics and add new types of topics.

Collecting topics to create content

You create larger pieces of content by linking to multiple topics in a ditamap file, which may remind you of an outline or table of contents. Figure 48 shows part of a ditamap that includes the topic shown in the previous figure.

Figure 48: *Excerpt from a ditamap file provided in the Open Toolkit*

```
<map title="Eclipse content aggregated by a map">
<topicref href="tasks/garagetaskoverview.xml" type="concept">
  <topicref href="tasks/changingtheoil.xml" type="task"/>
  <topicref href="tasks/organizing.xml" type="task"/>
  <topicref href="tasks/shovellingsnow.xml" type="task"/>
  <topicref href="tasks/spraypainting.xml" type="task"/>
  <topicref href="tasks/takinggarbage.xml" type="task"/>
  <topicref href="tasks/washingthecar.xml" type="task"/>
</topicref>
...
</map>
```

Topic shown in Figure 47.

The ditamap file in Figure 48 creates hierarchy: there are six topics nested underneath the first topic, garagetaskoverview.xml.

Some authoring tools, including FrameMaker and XMetaL, have integrated support for the DITA structure. These tools let you create and modify topics and ditamaps in interfaces similar to word processors, but unlike word processing applications, the tools validate your content against the DITA structure. You can also edit ditamaps and topics in a text or XML editor, particularly if you are accustomed to working with the coding of other markup languages (such as HTML).

Creating output from DITA content

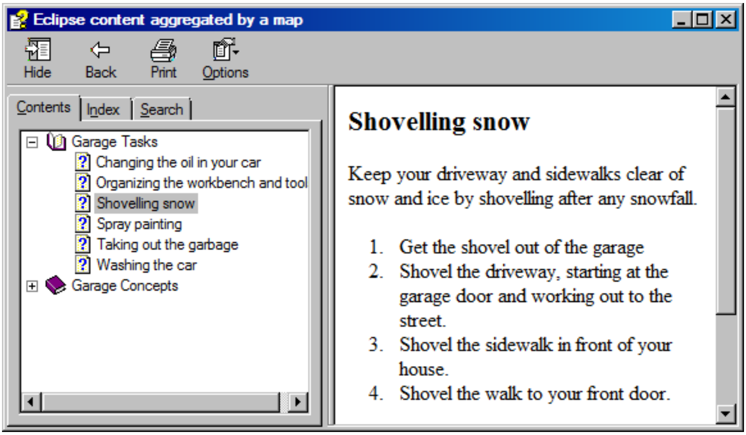
The DITA Open Toolkit also provides XSL stylesheets that transform XML content into multiple kinds of output, including PDF, HTML, and HTML Help. Generally, you transform ditamap files into output.

If you're using an authoring tool with built-in support for DITA, you likely create output by selecting menu choices or buttons, and the transformation process starts—the application calls the stylesheets that convert the DITA content. If you're not using such a tool, you transform content by typing a series of commands at the command prompt that run the stylesheets on your DITA content.

Figure 49 shows the HTML Help output for the ditamap file in Figure 48 on page 237.

The default output created with the XSL stylesheets is unlikely to have the look and feel that most documentation groups require; the default HTML Help output shown in Figure 49 is functional but not very attractive. It is possible to modify the stylesheets to change the appearance of the output (add company logos, use different fonts and colors for the text, and so on).

Figure 49: *HTML Help generated from sample files in the Open Toolkit*[1]



Shovelling snow

Keep your driveway and sidewalks clear of snow and ice by shovelling after any snowfall.

1. Get the shovel out of the garage
2. Shovel the driveway, starting at the garage door and working out to the street.
3. Shovel the sidewalk in front of your house.
4. Shovel the walk to your front door.

The good, the bad, and the ugly of DITA

If you're part of a documentation team that's considering a DITA implementation, you need to evaluate both the positive and negative aspects of DITA—and there are plenty of both. The following lists are by no means comprehensive, but they can serve as a good starting point for determining if DITA is a good fit for your department.

[1] Figure 49 does show HTML Help, even though the title bar mentions "Eclipse content," which is another type of output you can generate with the Open Toolkit. The title bar shows "Eclipse content aggregated by a map" because the sample ditamap file provided in the toolkit uses that text as the map title, as shown in Figure 48 on page 237. If you modified that map title value and generated the output again, the new HTML Help would show the updated text in the title bar.

Good

- DITA reduces content duplication and increases reuse of information through modular writing.
- DITA relies on open standards; using the Open Toolkit eliminates licensing costs for proprietary help conversion tools.
- The Open Toolkit provides paths to multiple output types by default: PDF/print, HTML, HTML Help, JavaHelp, and Eclipse Help.
- The DITA structure has elements with similar names to corresponding HTML tags, which can reduce the learning curve for those familiar with HTML.
- The Open Toolkit provides attribute-based conditional processing support: you can include or exclude content in output based on the values of attributes. For example, DITA provides a platform attribute. You can set that attribute to “win” for topics specifically about the Windows version of a product; for Macintosh, you can use “mac” as the value. When you create a deliverable for the Windows version, you include all the “win” topics and exclude the “mac” ones (and vice versa for the Macintosh deliverable).
- Growing adoption of the DITA standard means more writers are familiar with it.

Bad

- The Open Toolkit is far from perfect: you should expect to hit a few snags even in implementations with few changes to the default environment.
- Documentation for using DITA and the Open Toolkit isn't complete or reliable.
- Modifications you make to the Open Toolkit may not mesh well with later releases of the toolkit; this can be particularly problematic when a new release fixes a problem or adds a new feature you want to use.
- The Open Toolkit provides no support for context-sensitive help by default.

Ugly

- Default output for HTML and online help is not attractive, and modifying the stylesheets in the Open Toolkit to improve the formatting can be difficult.
- Default PDF output is rudimentary and not entirely reliable, and modifying the stylesheets in the Open Toolkit that control PDF processing is *very* difficult.

NOTE: The preceding lists were accurate at the time we wrote this content. Later versions of DITA and the Open Toolkit may address some of the issues.

Free but not cheap

The DITA Open Toolkit is open source, so you can download and use it at no charge. (For detailed information about the open source movement, visit www.opensource.org.) In the case of the Open Toolkit, that means getting a lot of stuff without paying any licensing fees: a predefined structure for software documentation and several XSL templates to create online outputs.

Despite paying no money for DITA (or any open source technology), that doesn't mean it's free. For example, the default output from the Open Toolkit won't match your company's formatting specifications. Someone in the documentation group needs to modify the templates to create the right output, or your company can a hire a consultant to make the changes. Regardless of who does the work, your company incurs costs.

Also, documentation for open source tools can be nonexistent, spotty, contradictory, and sometimes just *wrong*. This can lead to a lot of trial and error when installing and setting up open source technology. You may even decide to write documentation for the rest of your team so coworkers will know how to use the tool. These efforts result in costs.

In a few cases, open source tools may not work as expected (or not at all). Developers creating open source technology are usually volunteering their time, so fixing a problem that is affecting what you are trying to do may not be a priority. And even if the developers do agree to address the bug, that doesn't mean it will be fixed within a time frame that helps you.

In short, using open source tools is not free of costs, and there is some risk involved. However, many open source applications work as well or better than tools you buy.

14 Web 2.0 and technical communication

What's in this chapter

- Cardinality in communication
- User assistance in a Web 2.0 world
- Technical Writing 2.0
- Integrating Web 2.0 and user assistance

Technical writers are accustomed to being the gatekeepers for product information—they carefully organize product documentation, online help, and other user assistance for their readers.

Today, more and more users are also turning to unofficial sources on the Internet to get product information. *Web 2.0* technologies, which enable people to easily share information on the web, include the following methods of communication:

- Blogs (short for “web logs”)—web-based journals that contain commentary, often on a particular subject
- Forums—online bulletin boards where people can discuss particular products or topics
- Wikis—web sites that encourage collaborative authoring by allowing visitors to modify content

This chapter explores the implications of Web 2.0 content and describes how technical writers can integrate that content into their content development strategies.[1]

Cardinality in communication

The term *cardinality* is typically used in data modeling. It refers to the relationship between two objects and how they are connected. For example, a person should have a one-to-one relationship with a tax ID number (such as the U.S. Social Security number). But a person could have a one-to-many relationship with a phone number object (one person can have many phone numbers). The basic cardinality types are one-to-one, one-to-many, many-to-one, and many-to-many.

[1] The idea for this chapter was sparked by Andy Oram’s blog post, “What comes after the information age,” on O’Reilly Radar (radar.oreilly.com/archives/2007/09/what-comes-after-the-informati.html).

You can apply cardinality to verbal communication:

- One-to-one—a conversation between two people
- One-to-many—a professor giving a lecture to a hall full of students
- Many-to-one—a jury giving their verdict to a defendant
- Many-to-many—a group of people at a party, all talking simultaneously at the top of their lungs

For written and technical communication, cardinality is more complex.

One-to-one

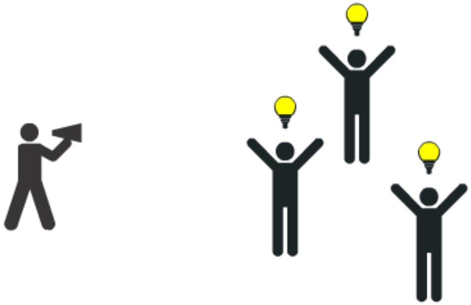
In technical communication, the most common example of one-to-one cardinality is phone-based technical support. The end user talks directly to a company support representative. One-to-one communication is usually quite effective—if something doesn't make sense at first, you can ask questions to clarify—but it is also expensive (Figure 50).

Figure 50: *Phone-based technical support is one-to-one communication*

One-to-many

Traditional publishing, including technical communication and user assistance, offers one-to-many communication. An author writes content that is consumed by many readers. One-to-many communication is much more cost-effective than one-to-one communication, but when writing for a large audience, authors necessarily create information that's more generic than a person-to-person conversation (Figure 51).

Figure 51: *Technical communication is one-to-many communication*

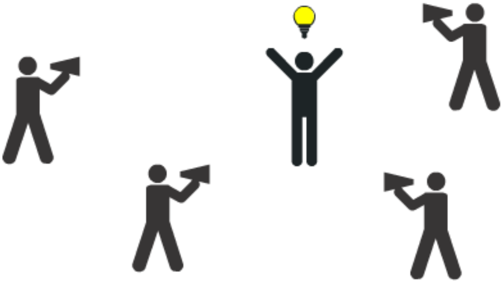


There are several ways of defining Web 2.0, but for technical communicators, the critical issue is that *Web 2.0 shifts communication from one-to-many cardinality into many-to-one and many-to-many.* The publishing channel is no longer restricted to professional authors; instead, anyone can provide information to anyone else.

Many-to-one

Many-to-one communication means that multiple people are providing answers to a single person. For example, Person A posts a question on a mailing list and receives offlist responses from a dozen people (Figure 52).

Figure 52: *Many-to-one communication involves multiple people providing information to a single person*

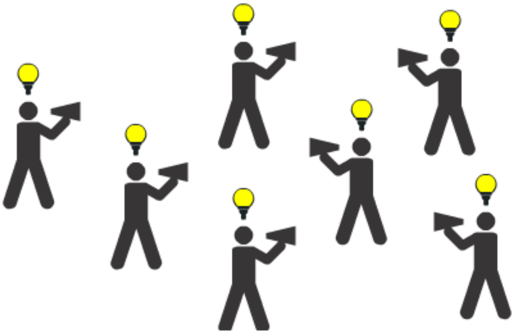


Many-to-one communication is arguably the least efficient method of communication because of the overlap in effort to provide answers to a single person. However, for the person receiving the information, it's often the fastest and the highest-quality communication channel. When several people provide answers, a consensus often emerges for the recipient.

Many-to-many

Many-to-many communication is at the center of the Web 2.0 concept. Everyone is requesting and receiving information, and roles shift fluidly from information provider to information recipient (Figure 53).

Figure 53: *In many-to-many communication, the distinction between information producers and information consumers disappears*



User assistance in a Web 2.0 world

The content produced by technical communicators is usually included with the product—as a printed manual in the product box or as online help accessible from the software's menu. In the past, third-party information was available, but often difficult and expensive to access. For instance, many trade computer books are available, but they are not included with the product—a reader must purchase a third-party book separately.

Today, the privileged position of official technical documentation is eroding—users who have a problem with their product often start by typing a query into Google instead of reaching for the manual or clicking on the help. In effect, professional technical communications are competing for attention with the collective intelligence of the web.

For many writers, this feels uncomfortable and vaguely insulting. As professionals, they feel that their efforts should be respected more than those of someone hiding behind a user name, such as JoeProductHater. In reality, however, technical writers are often constrained by corporate realities—they must be tactful about a product's limitations. By contrast, forum participants have few or no inhibitions about criticizing products or suggesting alternatives.

Several components of the web and of Web 2.0 contribute to the user-generated content revolution.

Search

Most books and online help provide tables of contents, indexes, and (online) full-text search. But to use these search features, the information seeker must already be using that book or help system.

Today, it's quite common to begin searching for information by simply typing a few words into a browser search box (Figure 54).

Figure 54: *Is your user assistance content exposed to search engines for indexing?*

user assistance web 2.0

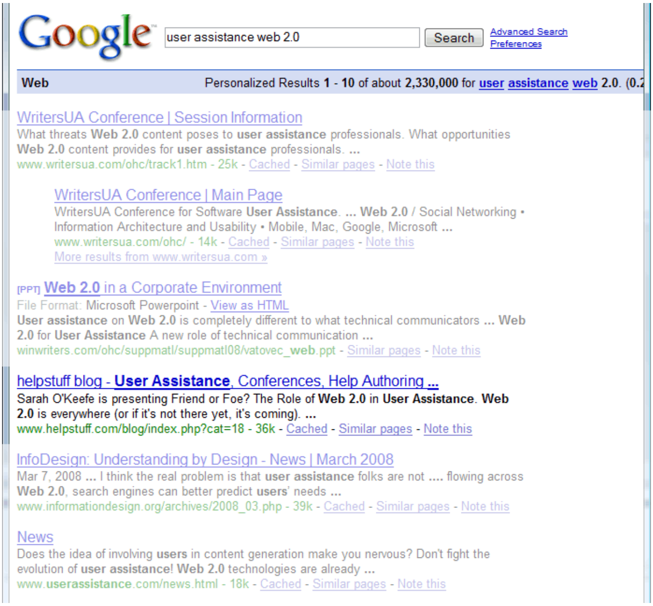
If you want people to find and use your content, you must:

- Publish in a location that is accessible for search engine indexing
- Create content and keywords that appear in the results of a search
- Provide useful, accurate information

Many technical writers are concentrating on the last item and ignoring the first two. But if nobody is finding their content, then it might as well not exist. Unfortunately, many writers see this list as a list of exclusive choices—they ask whether they should optimize keywords *instead of* ensuring that documents are technically accurate. The answer is that they must do both (Figure 55).

One significant challenge in search technology is to create a universal search that works across the web, online help, forums, and so on. Most message board software provides search functionality, but that search probably works differently from the typical web-based search. The search features built into web-based help systems likely use a third search algorithm. For a company that provides multiple types of information on their web site, this presents a serious dilemma. How should they aggregate and present search results from three (or more) different engines in a unified whole?

Figure 55: *How do I get to be first? (Google search results from April 1, 2008[1])*



Web Personalized Results **1** - **10** of about **2,330,000** for **user assistance web 2.0**. (0.2

WritersUA Conference | Session Information
What threats **Web 2.0** content poses to **user assistance** professionals. What opportunities **Web 2.0** content provides for **user assistance** professionals. **...**
www.writersua.com/ohc/track1.htm - 25k - Cached - Similar pages - Note this

WritersUA Conference | Main Page
WritersUA Conference for Software **User Assistance**. **... Web 2.0** / Social Networking • Information Architecture and Usability • Mobile, Mac, Google, Microsoft **...**
www.writersua.com/ohc/ - 14k - Cached - Similar pages - Note this
More results from www.writersua.com »

[PPT] **Web 2.0** in a Corporate Environment
File Format: Microsoft Powerpoint - View as HTML
User assistance on **Web 2.0** is completely different to what technical communicators **... Web 2.0** for **User Assistance** A new role of technical communication **...**
winwriters.com/ohc/suppmatl/suppmatl08/vatovec_**web**.ppt - Similar pages - Note this

helpstuff blog - **User Assistance**, Conferences, Help Authoring **...**
Sarah O'Keefe is presenting Friend or Foe? The Role of **Web 2.0** in **User Assistance**. **Web 2.0** is everywhere (or if it's not there yet, it's coming). **...**
www.helpstuff.com/blog/index.php?cat=18 - 36k - Cached - Similar pages - Note this

InfoDesign: Understanding by Design - News | March 2008
Mar 7, 2008 **...** I think the real problem is that **user assistance** folks are not **....** flowing across **Web 2.0**, search engines can better predict **users**' needs **...**
www.informationdesign.org/archives/2008_03.php - 39k - Cached - Similar pages - Note this

News
Does the idea of involving **users** in content generation make you nervous? Don't fight the evolution of **user assistance**! **Web 2.0** technologies are already **...**
www.**userassistance**.com/news.html - 18k - Cached - Similar pages - Note this

Each of the available options has limitations:

- Group search results by location. This is technically the easiest solution. Forum results go in one list, user assistance results in another list, knowledge base search results are in a third list. But from a searcher's point of view, it's pretty unfriendly.
- Ladder search results into a unified presentation. All of the #1 search results are presented, followed by all of the #2 search results, and so on. The problem with this approach is that if the relevant search results are concentrated in one type of result (for

[1] www.google.com/search?q=user+assistance+web+2.0

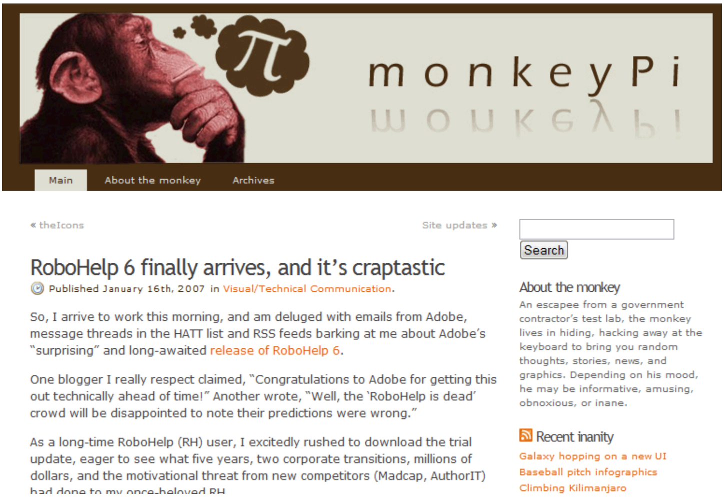
example, the user assistance results are much better than the forum results), the best information will not be at the top of the list.

The "correct" solution is to use a single search technology across all of the different types of content, but this may not be feasible due to technical limitations.

Blogs

Blogs are usually colorful and often insulting. Bloggers operate with little or no sympathy for the realities that drive corporate decision-making. Instead, they skewer products (and sometimes documentation) mercilessly (Figure 56).

Figure 56: *Bloggers are often not nice (web site captured on April 8, 2008[1])*



RoboHelp 6 finally arrives, and it's craptastic

Published January 16th, 2007 in Visual/Technical Communication.

So, I arrive to work this morning, and am deluged with emails from Adobe, message threads in the HATT list and RSS feeds barking at me about Adobe's "surprising" and long-awaited release of RoboHelp 6.

One blogger I really respect claimed, "Congratulations to Adobe for getting this out technically ahead of time!" Another wrote, "Well, the 'RoboHelp is dead' crowd will be disappointed to note their predictions were wrong."

As a long-time RoboHelp (RH) user, I excitedly rushed to download the trial update, eager to see what five years, two corporate transitions, millions of dollars, and the motivational threat from new competitors (Madcap, AuthorIT) had done to my once-beloved RH

About the monkey

An escapee from a government contractor's test lab, the monkey lives in hiding, hacking away at the keyboard to bring you random thoughts, stories, news, and graphics. Depending on his mood, he may be informative, amusing, obnoxious, or inane.

Recent inanity

Galaxy hopping on a new UI
Baseball pitch infographics
Climbing Kilimanjaro

[1] monkeypi.net/?p=133

For technical communicators, blogs open up several opportunities. First, the criticism that bloggers provide can be valuable ammunition in improving products. Technical writers are often stymied when they request product improvements based on issues that occurred while writing content for products. A blogger's impolite—and public—dissection of the deficiency may be just what's needed to move the issue up the priority list.

When a blogger complains about being unable to use a particular feature, is that a product deficiency or a problem with the documentation? Technical writers can use blog postings to identify places where user assistance should be improved.

Finally, and most controversially, technical writers and others can participate in blogging. One option is to comment on a blog entry. For example, if a blogger complains about a particular feature, the technical writer could provide a link to the online user assistance and request feedback on whether the instructions provided there are useful.

The most engaged (and terrifying) option would be for the technical writing group to start their own public blog to spark discussion with their readers.

NOTE: All of these approaches may be constrained by corporate policies, which should be thoroughly investigated first.

Blog comments for user assistance

A few companies, notably Adobe, allow readers to comment directly on their user assistance. Adobe's LiveDocs interface provides an Add Comment button at the bottom of each help topic. They also provide an easy way to generate a list of all comments for a particular product (Figure 57).

This presents some really interesting conundrums, like “Who owns the information?” and “What should you do about inaccurate comments?”

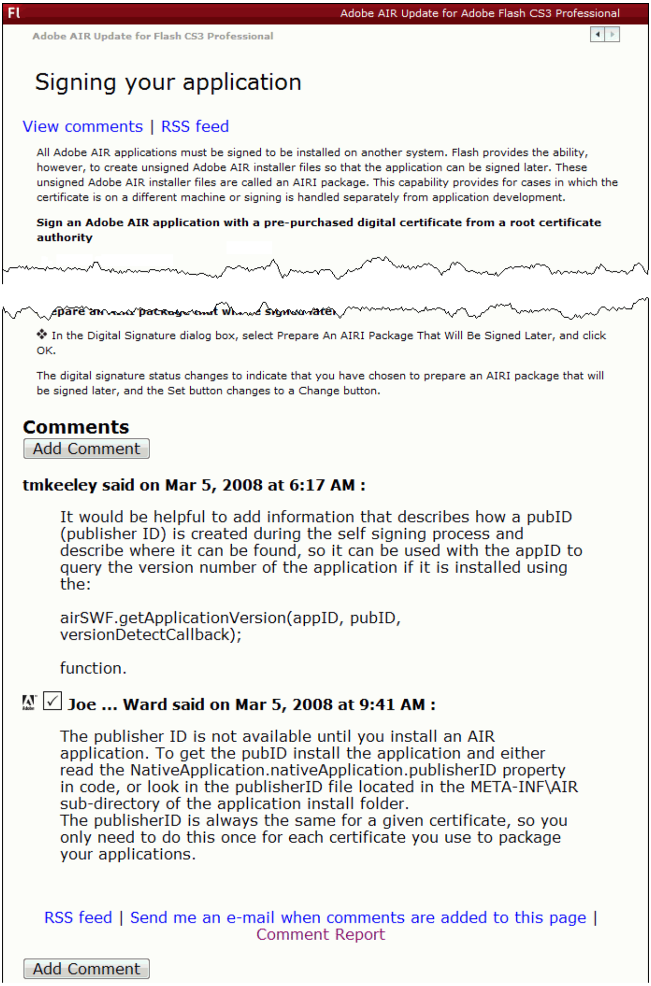
Adobe’s user assistance is available on the web to the general public. To add a comment, however, a reader must log in with an Adobe ID. Therefore, Adobe can associate comments with a unique email address. Although it’s obviously quite easy to set up multiple email accounts, this approach does presumably make readers think twice before posting something completely inappropriate.

Permitting readers to comment on your content is an interesting idea. It should help to:

- Identify areas where documents are inaccurate, confusing, or incomplete
- Identify areas that are controversial (where a lot of comments tend to accrue)
- Provide a venue for readers to participate in a (somewhat) controlled environment

It does present another interesting problem. If a commenter identifies a problem and the technical writer makes the needed updates and republishes the information, what should happen to the comment? Should it be deleted because it is no longer relevant? What if the comment is inaccurate? Should the comment be deleted in that case? Should the person deleting the comment explain why it is being deleted? Or should all comments be preserved permanently?

Figure 57: *Adobe LiveDocs allows reader comments (web site captured April 8, 2008*[1]*)*



Fl
Adobe AIR Update for Adobe Flash CS3 Professional
Adobe AIR Update for Flash CS3 Professional

Signing your application

View comments | RSS feed

All Adobe AIR applications must be signed to be installed on another system. Flash provides the ability, however, to create unsigned Adobe AIR installer files so that the application can be signed later. These unsigned Adobe AIR installer files are called an AIRI package. This capability provides for cases in which the certificate is on a different machine or signing is handled separately from application development.

Sign an Adobe AIR application with a pre-purchased digital certificate from a root certificate authority

❖ In the Digital Signature dialog box, select Prepare An AIRI Package That Will Be Signed Later, and click OK.

The digital signature status changes to indicate that you have chosen to prepare an AIRI package that will be signed later, and the Set button changes to a Change button.

Comments

Add Comment

tmkeeley said on Mar 5, 2008 at 6:17 AM :

It would be helpful to add information that describes how a pubID (publisher ID) is created during the self signing process and describe where it can be found, so it can be used with the appID to query the version number of the application if it is installed using the:

airSWF.getApplicationVersion(appID, pubID, versionDetectCallback);

function.

✓ **Joe ... Ward said on Mar 5, 2008 at 9:41 AM :**

The publisher ID is not available until you install an AIR application. To get the pubID install the application and either read the NativeApplication.nativeApplication.publisherID property in code, or look in the publisherID file located in the META-INF\AIR sub-directory of the application install folder.
The publisherID is always the same for a given certificate, so you only need to do this once for each certificate you use to package your applications.

RSS feed | Send me an e-mail when comments are added to this page | Comment Report

Add Comment

[1] livedocs.adobe.com/air/1/airextflash/

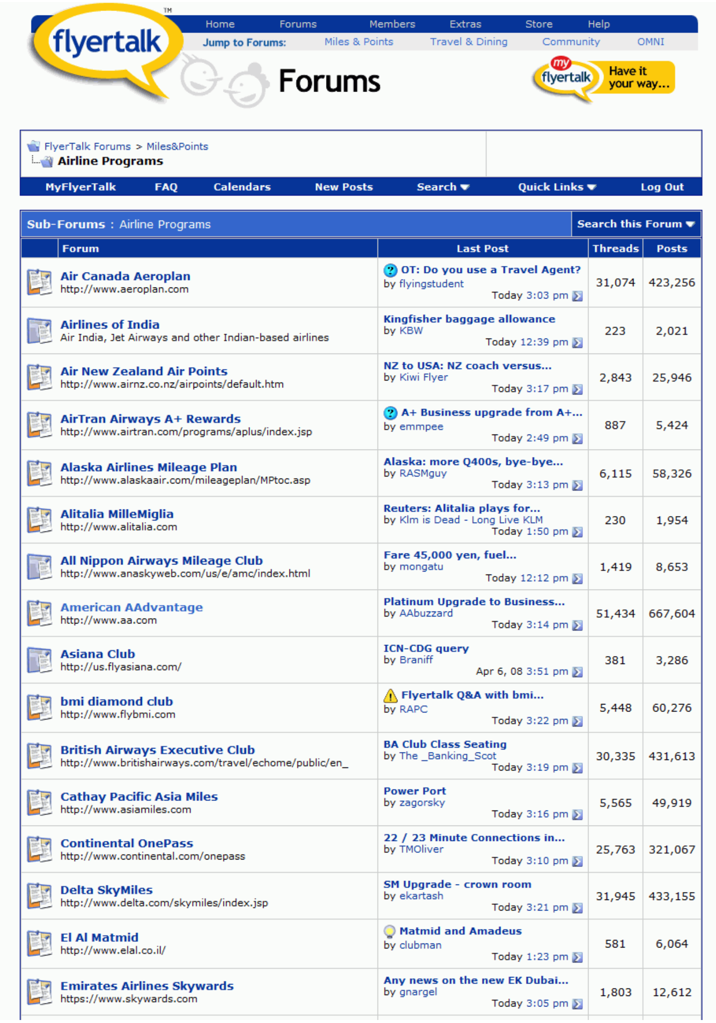
Forums

Online forums predate web technology; some may remember the old CompuServe bulletin boards. Today's web-based message boards provide a gathering spot for people to discuss particular products, games, or common interests.

Broadly, forums can be divided into "corporate" and "third-party" boards. That is, some companies provide forums for their own products and host the forums on their own web sites. But forums are easy to set up, and many forums exist independent of corporate control. For example, flyertalk.com provides a huge set of message board for frequent flyers with discussions of specific cities, frequent flyer programs, travel security, and much more (Figure 58). The discussions are often raucous, and the U.S. Transportation Security Administration is a particularly popular target. If you fly frequently, this web site provides invaluable resources, tip, tricks, and workarounds for making travel bearable.

Companies may want to control the discussion on corporate forums, but if forums are too heavily moderated, participants will migrate to another, less restrictive discussion option.

Figure 58: *Frequent-flyer forums definitely not endorsed by the airlines (web site captured April 8, 2008*[1]*)*



Forum	Last Post	Threads	Posts
Air Canada Aeroplan http://www.aeroplan.com	**OT: Do you use a Travel Agent?** by flyingstudent Today 3:03 pm	31,074	423,256
Airlines of India Air India, Jet Airways and other Indian-based airlines	**Kingfisher baggage allowance** by KBW Today 12:39 pm	223	2,021
Air New Zealand Air Points http://www.airnz.co.nz/airpoints/default.htm	**NZ to USA: NZ coach versus...** by Kiwi Flyer Today 3:17 pm	2,843	25,946
AirTran Airways A+ Rewards http://www.airtran.com/programs/aplus/index.jsp	**A+ Business upgrade from A+...** by emmpee Today 2:49 pm	887	5,424
Alaska Airlines Mileage Plan http://www.alaskaair.com/mileageplan/MPtoc.asp	**Alaska: more Q400s, bye-bye...** by RASMguy Today 3:13 pm	6,115	58,326
Alitalia MilleMiglia http://www.alitalia.com	**Reuters: Alitalia plays for...** by Klm is Dead - Long Live KLM Today 1:50 pm	230	1,954
All Nippon Airways Mileage Club http://www.anaskyweb.com/us/e/amc/index.html	**Fare 45,000 yen, fuel...** by mongatu Today 12:12 pm	1,419	8,653
American AAdvantage http://www.aa.com	**Platinum Upgrade to Business...** by AAbuzzard Today 3:14 pm	51,434	667,604
Asiana Club http://us.flyasiana.com/	**ICN-CDG query** by Braniff Apr 6, 08 3:51 pm	381	3,286
bmi diamond club http://www.flybmi.com	⚠ **Flyertalk Q&A with bmi...** by RAPC Today 3:22 pm	5,448	60,276
British Airways Executive Club http://www.britishairways.com/travel/echome/public/en_	**BA Club Class Seating** by The _Banking_Scot Today 3:19 pm	30,335	431,613
Cathay Pacific Asia Miles http://www.asiamiles.com	**Power Port** by zagorsky Today 3:16 pm	5,565	49,919
Continental OnePass http://www.continental.com/onepass	**22 / 23 Minute Connections in...** by TMOliver Today 3:10 pm	25,763	321,067
Delta SkyMiles http://www.delta.com/skymiles/index.jsp	**SM Upgrade - crown room** by ekartash Today 3:21 pm	31,945	433,155
El Al Matmid http://www.elal.co.il/	**Matmid and Amadeus** by clubman Today 1:23 pm	581	6,064
Emirates Airlines Skywards https://www.skywards.com	**Any news on the new EK Dubai...** by gnargel Today 3:05 pm	1,803	12,612

[1] flyertalk.com/forum/forumdisplay.php?f=374

Wikis

The largest and most well-known example of a wiki is Wikipedia (www.wikipedia.com). A wiki enables participants to author content collaboratively.

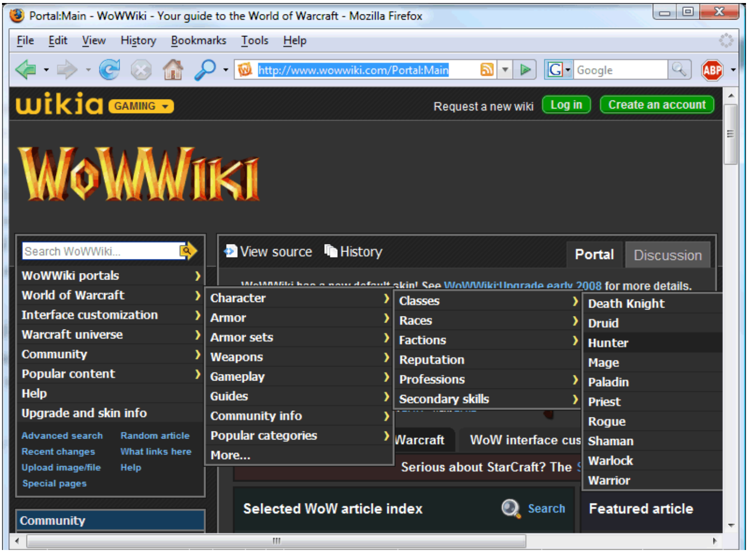
At its best, a wiki results in articles that capture the collective intelligence of the participants. In some cases, though, disagreements can escalate into article vandalism and other problems. In a wiki, participant A writes an article, participant B rewrites the article, and participant A gets very upset. People have a visceral reaction when their writing is changed.

Wikis are extremely useful for developing content that benefits from consensus. They are used heavily by gamers (perhaps because game designers tend to provide minimal documentation?). The World of Warcraft wiki, for example, is enormous (Figure 59).

Evaluating credibility

Technical writers have a built-in advantage in credibility. Their work ships with the product—on a CD, as a paper manual, or as user assistance that's connected directly to a software product. Readers assume that officially provided technical content has undergone some sort of quality control. (Although a company's reputation for technical documentation quality may increase or decrease readers' trust in the content.)

Figure 59: *Wiki navigation helps make information easier to find (web site captured April 8, 2008[1])*



For user-generated content, there are some qualitative criteria that readers use to evaluate whether a source is credible. These include the following:

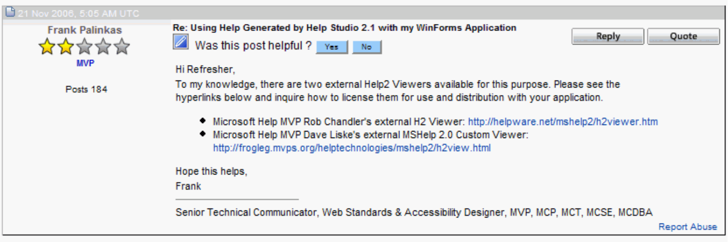
- Anonymity. A blogger whose identity is known is generally more credible than an anonymous blogger. However, like anonymous sources in journalism, blogging anonymously could actually increase credibility if the blogger has an excellent reason for remaining unnamed. For example, the blogger known as Mini-Microsoft (minimsft.blogspot.com) is thought to work inside Microsoft, but his (or her) blog clearly does not toe the corporate line. In this case, anonymity makes the

[1] www.wowwiki.com/Portal:Main

postings more compelling because we believe that Mini-Microsoft probably does have inside information which necessitates his or her anonymity.

- Post count. Most forums provide this information. Every time a poster publishes a message, the post count increases. Although by itself post count does not equal credibility, a higher post count indicates longer participation on the forum and greater investment in the topic at hand.
- Membership date. Mainly used in forums, a membership date lets you evaluate how long that person has been participating (under that ID) in the forums. Long-time participation, especially combined with regular posting, increases credibility.
- Member types. A general member might be someone who just joined and has fewer than 100 posts. A senior member might have at least six months on a forum and more than 100 posts. Moderators and administrators are connected to the forum management. Many corporate forums also identify participants who belong to the host company with their own membership categories. Finally, you see special member types, such as the Microsoft MVP or the Adobe Community Expert, that indicate people who have been recognized by the host company (Figure 60).

Figure 60: *A Microsoft MVP identifier (web site captured April 8, 2008*[1]*)*



21 Nov 2006, 5:05 AM UTC

Frank Palinkas
★★☆☆☆
MVP

Posts 184

Re: Using Help Generated by Help Studio 2.1 with my WinForms Application

Was this post helpful ? Yes No

Hi Refresher,
To my knowledge, there are two external Help2 Viewers available for this purpose. Please see the hyperlinks below and inquire how to license them for use and distribution with your application.

- Microsoft Help MVP Rob Chandler's external H2 Viewer: http://helpware.net/mshelp2/h2viewer.htm
- Microsoft Help MVP Dave Liske's external MSHelp 2.0 Custom Viewer: http://frogleg.mvps.org/helptechnologies/mshelp2/h2view.html

Hope this helps,
Frank

Senior Technical Communicator, Web Standards & Accessibility Designer, MVP, MCP, MCT, MCSE, MCDBA

- Badges. A badge provides graphic shorthand for that participant's level of seniority on the message board. For example, a technical support person from the host company might have one type of badge and a senior (external) member a different type.
- Ratings. Ratings are relatively new. In some forums, readers can vote on the usefulness of a particular contribution. Articles that get a lot of votes are highlighted in search results and elsewhere. Based on votes on the forum posts, a particular author might then also have a rating for providing especially useful or useless information.

Taken together, these items give readers a way to evaluate the reliability of a message.

[1] forums.microsoft.com/MSDN/ShowPost.aspx?PostID=947210&SiteID=1

Advantages and disadvantages of Web 2.0 content

User-generated content tends toward the following characteristics:

- Authentic. The people contributing content are rarely tactful; their opinions probably don't reflect official corporate positioning. The fact that information is unfiltered is one of its greatest assets to other readers.
- Passionate. It takes motivation to write a blog post, participate on a forum, or edit a wiki page. The people creating Web 2.0 content are passionate about the products they are writing about. Apathetic people don't bother to contribute.
- Specific. User-generated content tends to be about one person's experience rather than the general workings on the product, so it tends to be quite specialized. For example, a discussion of a database installation probably would describe how to configure a specific version (the one the author used) and not how to install, in general, any database. If the configuration being described matches your particular requirements, this could be a good thing.
- Not comprehensive. User-generated content will cover the information that users find interesting or compelling, unlike technical documentation, which is expected to provide universal coverage of the product. User-generated content will go much deeper than technical documentation in places, but it will also have enormous gaps in coverage.
- Not edited. With the possible exception of content in very large wikis, user-generated content is not

edited. The quality of the writing will be as good as the effort that the original author put into the message.

- Not curated. Users write about what they care about. Important but boring topics may not get much attention. Ratings and other voting mechanisms can help surface content from the ocean of information, but useful information may sink into oblivion because it is not linked or sufficiently searchable.

Where professional technical communicators are most valuable

Professional content creators add the most value where content curating is needed:

- Conceptual information and product overviews. A high-level description of a product requires time and attention from someone who understands the entire product. For complex products, discussions of the best implementation approach could be invaluable. Most user-generated content will focus on specific tasks that need to be completed rather than on providing perspective.
- Thorough coverage of all topics. Instead of focusing on interesting and exciting subject matter exclusively, a professional content creator will ensure that all of the needed topics are discussed in the documentation and user assistance.
- High production values. The professionals win on quality—user-generated content is rarely edited, copyfitted, or even designed.

- High-stakes documentation. If documenting by trial and error is out of the question, the professionals are necessary. Some obvious industries where “let the users figure it out” is probably not a very good idea include aerospace, defense, and medical devices.

Where users are most valuable

Users can (and will) play an important part in many industries where their participation is less risky. They are especially good at the following:

- Providing technical support and workarounds
- Correcting errors in the official content
- Highlighting things that are not working
- Demanding changes where they are needed

Technical Writing 2.0

Given the impending collision of technical writing and Web 2.0, what will Technical Writing 2.0 look like for professional content creators? Any strategy must take into account corporate policies (if they exist), but consider the following actions:

- Read blogs, forums, and wikis to keep up with user comments.
- Correct mistakes when you find them by commenting on blogs and forums or editing wiki pages. (Note that some wiki policies discourage editors

who have an affiliation with the topic in question from making changes. Be sure that you understand the wiki's culture before you make changes.)

- When you participate in online discussions, be clear about your corporate affiliation. You might choose not to give your entire name, but you could create an ID such as ProductXTechWriter that explains your position.
- Your power as a gatekeeper is greatly reduced. Although you can choose to omit information from the official product documentation, it's likely that the information will surface somewhere on the web. Instead, embrace your new role as a curator whose job it is to ensure that the best, most useful information is highlighted for readers. For this, product expertise is critical.
- Be cautious about product positioning or marketing content. If the official content glosses over problems, readers will probably choose unofficial, unapproved alternatives. It's not necessary for the official content to bash the product, but a dose of candor will help you keep your readers.

Integrating Web 2.0 and user assistance

Just as the availability of online help and printed technical manuals required a new approach to technical communication, the rise of Web 2.0 technology means that technical communicators need to assess content development strategies. Consider the following for products whose use does not have life-or-death consequences:

- Corporate technical content should provide comprehensive coverage of basic concepts with a particular focus on supplying conceptual information.
- The product web site should provide a platform for customer-generated content with user forums, wikis, and the like. The platform should include rating mechanisms (article ratings, voting, and badges) to help readers evaluate the credibility of contributors.
- Top contributors should be recognized, both online (badges) and with token gifts (shirts, key chains, and other swag).
- The product web site should include a blogging platform for corporate bloggers.
- The product web site should provide links across the various content types to help readers locate related information.
- The product web site should provide integrated search across the various content types.

For professional authors, it's tempting to ignore user-generated content and hope that it's a fleeting fad. This is, however, highly unlikely. Motivated users will always find a way to get their message out, and Web 2.0 technology makes it easy.

A Getting your first job as a technical writer

What’s in this appendix

- Demonstrating the skills of a technical writer
- Interviewing
- The portfolio
- Where should I look for a tech writing job?
- Professional organizations and networking sites
- Working as a freelance technical writer

After reading all about how the technical documentation process works, you’re probably wondering, “But how do I get my first technical writing job?”

Before you start your job search, consider your mission. To get a job as a technical writer, you need to convince an employer that you have mastered the basic skills that every technical writer needs. These are:

- An understanding of technology
- Writing ability
- Organizational skills
- Detective and people skills

Refer to Chapter 1 for a detailed discussion of each of these skill sets.

If you can prove that you have these skills, you can get a job as a technical writer. This chapter will show you how to reassess your current skills and make them relevant to technical writing.

Demonstrating the skills of a technical writer

Although new writers do grapple with "I need experience to get a job, but I need a job to get experience," you can minimize this problem. Experiences as a student, as a worker in another line of work, and as a volunteer can help show a prospective employer you've got the technological, writing, organizational, detective, and people skills required to succeed as a technical writer. You need to tailor your resume to highlight these skills.

Whenever you make changes to your resume, carefully check it for spelling, grammar, and formatting errors—and have a friend with writing skills look it over, too. Errors in resumes submitted for technical writing jobs will often automatically disqualify you from consideration.[1]

Alterations required

There's a big difference between tailoring your resume for a particular position and lying about your experience. Do not—EVER—lie on your resume. If you get caught, your career will take a hit. However, it's perfectly legitimate to change the description of a previous job to highlight past experience that's relevant to the job you want.

Another tricky area is software experience. If a particular job requires that you know a certain software package, do not just add that package to your resume. Again, getting caught could have serious consequences. Instead, find a copy of the software and spend some time learning it. Then, put it on your resume. If the employer asks, explain that you are self-taught. In many cases, the fact that you made the effort to learn the software will cancel out your lack of practical experience with it.

Assume that you do add a bunch of software to your resume and get hired on the strength of these fabrications. On day one, you show up on the job and are handed a new project and told to work in one of the software applications that you don't know. Now, you're stuck. You can't request training because you're supposed to know the application. You can struggle without assistance, but then the assignment will take much longer than it would take an expert software user. Either way, your inexperience will become apparent to your coworkers (and your boss). This is not a good way to start off in a new job.

Understanding of technology

You can demonstrate your understanding of technology in many different ways. Perhaps you studied computer science, math, or engineering in college. If so, your education will take care of the "Is this candidate technical enough?" question. If you are one of the many technical writers with a liberal arts degree, you'll need to show

[1] We once received a cover letter and resume that used different spellings of the candidate's first name.

that you have technical aptitude. Find a way to prove that you have some grasp of technology and that you are open to learning about new technology when you're looking for a tech writing job.

These days, almost everyone has had some exposure to basic computer software, such as word processors, email applications, web browsers, and spreadsheet programs. Basic computer literacy is a requirement for many jobs, not just technical writing. You'll want to be sure that this computer literacy is apparent from your resume. If you apply for a job by sending a resume via email, your actions prove that you can handle basic tasks on a computer.

Make a list of all the software that you've used. You may be surprised at how many different programs you already know. Add this list to your resume. If you have expert-level knowledge of any packages, be sure to point that out. And don't forget to list the operating systems you've used—Windows, Macintosh, UNIX, and Linux, for example. These kinds of details give the employer reading your resume a good feel for your level of technical knowledge.

If your list of software is painfully short, and if you're none too comfortable using a computer, you may have some work to do before you apply for your first technical writing job. Consider some of the following options.

Public libraries

Most public libraries have computers and Internet access. You can get free email accounts from many different providers, including mail.yahoo.com and mail.google.com. The librarian can probably help you out with this.

If you can get computer access at the library, spend some time getting familiar surfing the Internet and using email. Use web-based programs, such as Google Docs (docs.google.com) and ThinkFree (www.thinkfree.com), if library computers don't have word processing programs or other applications.

College computer lab

Most universities and colleges have computer labs. If you have access to one of these labs, you should find high-speed Internet access and a suite of word processing and other basic software applications.

Job training

If you're eligible for job training, you may be able to get into computer classes. This could include government programs and outplacement services. Outplacement is most often provided by a company that is laying off a large number of people. If you get caught in a layoff, find out what services your former employer provides and take advantage of them.

Buying a computer

This is the most expensive option, but it might also be the best one. Having a computer in your home means that you can practice your computer skills any time you want.

Building web pages

Many people build personal web pages. This is a great way to put technology to work for you and to show that you understand Internet technology. If you have some experience with web pages, list that on your resume.

If you've built web pages or a web site, consider providing the web addresses (URLs) of some web pages that you've built. But be careful—if your pages are full of strong personal opinions or political information, showing them to a potential employer could backfire in several ways. Political pages could cost you a job offer because the person reviewing your resume doesn't agree with your politics. Similarly, pointing to the web site for a particular church or other religious organization is asking for trouble. But if you helped put together a web site for your local animal shelter, it can't hurt to list that web site on your resume.[1]

If you were employed by an organization with a strong political viewpoint but didn't really agree with their politics, you might consider putting a disclaimer on your resume: "Built web pages for *xxx* organization. Do not necessarily endorse their point of view."

If you built the pages from scratch by writing HTML code in a text editor, say so. If you used a web design tool, specify which tool or tools you used. Don't forget about volunteer work, particularly if you're helping maintain the web pages (or even the web server) for a nonprofit group.

Another way to get some HTML skills is to download and modify open source HTML templates from sites such as www.oswd.org. To clearly demonstrate your

[1] We've received resumes pointing to pages with topics including feminist lesbians in China (no, really), fathers' rights, really terrible poetry, and much more.

contributions, you can show the same content with the original formatting and then with your modified formatting. Be careful not to pass off the template designer's work as your own.

When a prospective employer does a web search on your name, what turns up?

When a company is considering whether to interview (or even hire) you, it's likely an employee of that company will do a web search on your name. If you have a personal web site or a profile on a social networking site (such as Facebook and MySpace), those web pages may come up during a search. Think about whether you want prospective employers to see what's on those pages. For example, if you have posted photos showing your antics at a recent party, you may want to take them down.

Many search engines maintain older versions of web pages; search results include links labeled "cached" that point to previous versions. These cached versions mean that people can still see things recently deleted from a web page, so it's smart to think twice before posting anything to a personal web site or online profile. You never know who will see it—or if that content will come back to haunt you later.

Cheap (and free!) software

Many software manufacturers offer trial versions of their software that you can use to teach yourself. For example, Adobe (www.adobe.com) offers a 30-day trial of FrameMaker that includes all of the features of the licensed version. The same is true for XMetaL Author from JustSystems (na.justsystems.com).[1] Trial software is sometimes bundled with third-party books about applications.

[1] Companies sometimes change the length of trial periods and disable features in trial versions. Your experience with the trial versions of FrameMaker and XMetaL may differ from what is described.

If you're a student, check with software companies about buying academic versions of software. Academic versions are usually the same as full-license versions, but they cost significantly less—basically, you get a student discount, and it can be quite hefty.

You can also look for software on some of the online auction sites, such as www.ebay.com. You may be able to buy software below market price. Be careful, though, to check the seller's credentials; some of the software sold on auction sites is not legitimate. If you buy pirated software, you won't be able to upgrade it.

There are also some open source applications that can be very useful for putting together your resume and for learning basic computer skills. At www.openoffice.org, you can download a suite of no-cost tools for word processing, creating spreadsheets, drawing figures, and designing presentation slides. Be sure to list the open source tools you use on your resume—more and more companies are using them.

Software at work

If you're already in another line of work, there's a chance you may be using software specifically for your field. Even though the tool may not be used by technical writers, you should still list it on your resume to show that you use the particular industry's tools. You need to demonstrate you can master the tools of any trade.

Writing ability

Students, particularly liberal arts majors, can demonstrate writing ability by using a good term paper as a writing sample. Even if the paper is not about a technical topic, it can demonstrate that you understand the principles of grammar and how to organize content. Try to find an assignment that demonstrates your ability to perform research and organize the information you uncover into a coherent paper.

Many aspiring technical writers work for campus newspapers or magazines. Highlight that type of experience (and be sure to point out if you had a management position or helped with the design and layout in addition to writing).

Working for a college paper or magazine can also provide excellent writing samples. If you can show both article clippings and academic writing in a portfolio, you prove experience with different types of writing, which is a plus.

If your college days are in the distant past and you're already in the work force, you probably have some writing samples as well. Look for proposals, reports, press releases, marketing brochures, blog posts, or white papers that demonstrate writing ability. Remember to check into the volunteer work you've done for more possibilities.

See "The portfolio" on page 281 for more information about creating a portfolio.

Liberal arts degrees *can* be useful

Years ago, many employers considered liberal arts degrees to be as useful as a parka in the Sahara. Today, however, many companies—particularly those who employ technical writers—recognize the value of liberal arts degrees. Because liberal arts curricula generally include a great deal of writing, liberal arts graduates often a gain a better understanding of the craft of writing. Term papers also require research and analysis of a subject (whether Greek history or Shakespeare), and these abilities are very important in technical writing.

There's another less obvious benefit of a liberal arts background. Many liberal arts programs require students to take a wide variety of classes, which forces students to focus on different material every semester. The ability to juggle those different subjects and to switch gears mentally is a very valuable skill for workers in any field. It's particularly important in technical writing, where it's not uncommon to write documentation for a half-dozen different products in a year (and sometimes, you may find yourself writing about two completely unrelated products simultaneously).

Organizational skills

Your resume should clearly demonstrate that you can plan a schedule and deal with multiple tasks. Students should list extracurricular activities that involved any sort of managing (whether it be a project, people, or both), and those already in the workforce need to be sure their resume reflects any project management experience. Even volunteer work as a fund-raising coordinator is worth noting, particularly if you handled many different tasks.

Be wary, though, of listing activities that may tell your potential employer more than you want. For example, experience as the social chair of your fraternity or sorority is probably not going to convince your potential employer of your leadership abilities (your partying abilities, maybe, but that's usually not a plus).

Detective and people skills

College students can easily demonstrate detective or analytical skills with academic papers, such as theses, research projects, and term papers. Those already working can point to job responsibilities that require analytical ability. These might include projects where you were asked to evaluate several different options and write a memorandum recommending a particular decision. White papers are excellent examples of writing that requires analysis; if you have written any white papers, be sure to include the best ones in your portfolio and to list the experience on your resume.

If you have had a leadership role in extracurricular activities, you can use this to demonstrate your people skills. But by far the most important test of your people skills will be in an interview.

Interviewing

The personal interview is a critical part of the job hunting process. There are many excellent books on how to interview, but here are some basic pointers.

Dress professionally

Dress professionally for the interview, even if a company has a casual environment. If the company is very formal (typically banks, financial services companies, and government contractors), plan on formal business attire—suit and tie for men, business suit for women. If the company has a casual environment, you can probably dress less formally—dress shirt, pants, and a tie for men; separates for women.

Consider regional differences. For example, Austin is notorious (or maybe noted) for its casual dress codes. "Austin casual" means shorts, a T-shirt, and sandals. Other cities, such as Atlanta and Dallas, tend to be more corporate and have more formal dress codes.

Your goal should be to dress a notch or two better than the employees in the office. Your attire should show the employees that you are capable of looking like a professional. You may not be required to wear pantyhose or a tie for the next three years if you get the job, but it's a good idea to prove that you know how.

Don't oversell

Many job-hunting books tell you to sell yourself in the interview. You do need to do that, but don't overdo it. In addition to convincing the employer that you're right for the job, the interview is also an opportunity for you to assess whether the job is right for you. Ask questions about how the company works, what your responsibilities would be, and evaluate whether those responsibilities are what you want.

The interview is about figuring out whether there's a fit between you and the job. If you respond enthusiastically to every question ("I just *love* taking notes in meetings!"), you risk leaving the impression that you would be happy to perform duties that perhaps you don't really want to do.

What federal laws prohibit employment discrimination?

Title VII of the Civil Rights Act of 1964 (Title VII) prohibits employment discrimination based on race, color, religion, sex, or national origin.

The Equal Pay Act of 1963 (EPA) protects men and women who perform substantially equal work in the same establishment from sex-based wage discrimination.

The Age Discrimination in Employment Act of 1967 (ADEA) protects individuals who are 40 years of age or older.

Title I of the Americans with Disabilities Act of 1990 (ADA) prohibits employment discrimination against qualified individuals with disabilities in the private sector and in state and local governments.

Section 501 of the Rehabilitation Act of 1973 prohibits discrimination against qualified individuals with disabilities who work in the federal government.

The Civil Rights Act of 1991 provides monetary damages in cases of intentional employment discrimination.

Source: Equal Employment Opportunity Commission web site, www.eeoc.gov

Don't ask about salary

Asking about salary puts you in a very poor negotiating position. Wait until the employer brings up the topic. Many companies will ask you what salary you're looking for. The best response to this is to inquire about the salary range for the position. If they give you a range, you can say, "That sounds reasonable. I'm sure we can come to an agreement."

If you have previous relevant experience, you may be able to negotiate an offer higher than the base salary for an entry-level technical writer with no experience.

Starting salaries for technical writers vary tremendously by region. Members of the Society for Technical Communication (STC) can view a salary survey at the society's web site (www.stc.org). The U.S. Department of Labor also provides statistics on average salaries.

Be on time

Be on time for your interview. If you're not familiar with the company or the area, scout out the company the day before to ensure you know how to find it. Plan for traffic. If you arrive 15 minutes early, you can always sit in the parking lot for a few minutes before you go in. You should arrive at the reception desk (or at the interviewer's office) a little early, by five minutes or so. Do not show up earlier and expect the interviewer to see you immediately.

If you're going to be late, call and let the interviewer know immediately. Apologize, tell him or her how late you're going to be, and offer to reschedule for another time. If the problem is due to circumstances beyond your control (your plane has been delayed), point that out. If you're going to be late because the baby threw a fit when you dropped her off at daycare, you might want to be nonspecific about the problem and leave home earlier next time.

Send a thank-you note

Sending thank-you notes for interviews has gone out of fashion, and many applicants don't bother. You should. Sending a note immediately after the interview reiterating your interest in the position will make you stand out from all the other candidates who didn't bother. It also gives you an opportunity to mention something you might have forgotten to say in the interview. You can send your note via email, especially if the company intends to make the hiring decision soon.

The portfolio

A good collection of writing samples can demonstrate that you have writing and analytical abilities. If you've done different types of writing—term papers, news stories, and business writing (proposals, reports, and so on)—be sure your portfolio reflects the breadth of your writing experience.

However, most people applying for their first technical writing jobs probably don't have any true technical writing samples to show. Remedy this situation by writing some brief passages that show you can write task-oriented material.

You can do something as simple as the cake-baking procedure shown in Figure 7 on page 111, or you could document a few tasks for a piece of software (for example, how to create, save, and edit a file with the Notepad text editor, which comes with the Windows operating system). If you don't have the software (or ability) to draw figures, include placeholders that explain what's in the figures you would add to the procedure.

Consider contributing to the documentation for an open source tool. If you find instructions that aren't clear, make a PDF file of the content and then rewrite it. You can show a prospective employer how you improved the content by including both versions in your portfolio.

Be sure to keep a particular audience in mind while writing your tasks, and you might even want to specify the audience on the printout of the sample task you include in your portfolio. You'll really impress an interviewer if you demonstrate you know the importance of writing for your audience—the cardinal rule of good technical writing.

The sample procedures don't have to demonstrate something highly technical; it's more important that they show you can write for a specific audience and know how to break a task down into steps.

Consider creating a CD that includes PDF versions of your resume, portfolio pieces, and other any files that show your abilities; you can also place the files on a flash drive and let the interviewer copy the files from the drive.[1] Leaving files with prospective employers gives them the chance to review your samples in depth later.

What should my portfolio look like?

Professionals such as graphic designers and architects often carry enormous leather folders in which they showcase their best work.

For your samples, you can probably use something a little smaller—writing samples usually aren't oversized. Consider a basic leather folder. You can often find nice ones at your college bookstore or office supply stores. Any professional-looking folder that lets you organize your pieces will work.

[1] A prospective employer may not have the software you used to create your samples, so it is best to include files in PDF format or another format that a web browser can open. Applications for opening PDF files and web content are free, and they are often installed by default on systems.

If you a create a CD with sample pieces to leave with the interviewer, include a card or label that has your name and contact information.

Should I bring my portfolio to the interview?

Yes. Always bring your portfolio to the interview, even if the employer does not request it.

If you're interviewing by phone, and the employer wants you to send writing samples, be sure to send copies or electronic versions via email or on CD—it's best to keep your printed originals. You can also provide the addresses of web sites that show samples of your writing. Do not send any materials that are proprietary or confidential without getting written permission from the company for whom you created them. Sending confidential materials shows that you do not value a company or client's confidentiality requirements, and that will not score points with the interviewer.

Where should I look for a tech writing job?

To begin your search, investigate which cities have strong high-tech industries. Some of the best-known areas include:

- Silicon Valley (San Francisco Bay area), California
- Austin, Texas
- Boston, Massachusetts
- Research Triangle Park, North Carolina

- Portland, Oregon
- Seattle, Washington
- Washington, DC

If your home town isn't on the list, don't despair. Many other cities have significant high-tech concentrations in specific industries, such as:

- Dallas, Texas (telecommunications)
- Wichita, Kansas (aircraft)
- Melbourne, Florida (aerospace)

Contracting—getting your foot in the door

Many beginning technical writers start out working for a contract shop. It's often easier to get a job as a contractor than as a staff employee, especially if you have limited experience. The contract shop has an agreement with the client company. You do work for the client company, but your paycheck comes from the contract shop.

Some things to consider about contracting include:

- Contracting gives you the opportunity to get started and prove yourself. If you do a good job, there is sometimes the possibility of getting hired permanently by the client company.
- As a contractor, you are generally paid by the hour, which means that you'd get paid for any overtime. Not all staff technical writers are paid overtime.
- As a contractor, you are usually an hourly employee, so you get paid only for the hours that

you work. If you get sick, you may not have paid sick leave.

- If the workload decreases, contractors are usually let go before employees.
- A typical entry-level contract usually lasts six months or so. You can then move on to other contracts and gain a lot of experience in just a few years.

Contracting is not ideal, but it is easier to find contract employment than permanent, so it can be a good starting point.

Transferring within a company

If you are already employed but want to change jobs and become a technical writer, consider applying for a transfer. In many large companies, employees get preference over applicants from outside the company. If you can get your manager's approval, a transfer might be just the thing you need to get started as a technical writer.

Start-up companies

Small companies that are just getting started are sometimes willing to hire new technical writers. The trade-off is that they're generally not willing to pay very much. But if you can afford it, working for a start-up company could get you that all-important first job—and who knows, you might like the insanity that comes with an entrepreneurial company.

If you've got it, flaunt it

If you've been in the workforce for a while, you probably have some specialized knowledge. For example, if you've worked as a cable technician, you know quite a bit about installing and splicing cable. Try to find a company in which your experience will be relevant and useful. That company could be more willing to hire you as an entry-level technical writer because you already understand their technology.

Internships

Documentation departments sometimes offer summer internships to college students. Getting such an internship can be very valuable because it lets you see how a documentation department really works, even if interns are sometimes delegated less than glamorous work (such as transferring five sets of review comments into a single document or making copies of release notes).

In addition to giving you the opportunity to learn real-world technical writing skills and to observe a doc group in action, an internship can also give you a foot in the door of a prospective employer. It's not unusual for a company to groom interns for future full-time employment (but don't assume that an internship is a guarantee of future employment).

Professional organizations and networking sites

Professional organizations can help you with your job search by giving you a chance to meet people in the industry and to attend educational seminars. Networking sites, such as www.linkedin.com, can also be helpful. For a detailed list of resources, see Appendix B.

Working as a freelance technical writer

Some technical writers choose to work independently instead of working for a company or as a contractor through an agency. But freelancing is a difficult path, and you should be aware of some of the pitfalls before you try it.

What you need to make it as a freelancer

As an employee, you are provided with many things that you might take for granted: the equipment to do your job, a space in which to work, and sometimes even training. To be successful as a freelancer, you need outstanding skills—both job skills (writing, research, and the like) and business skills (bookkeeping, management, and the basics of banking and cash flow management).

The second major requirement for a successful freelancer is an excellent professional network. Networking and referrals are how you find jobs, so it's critical that you have a wonderful network that works for you. Ideally, you want to get a phone call every week that starts, “So-and-so recommended that I call you about doing some writing for our project.” Building this network takes years, so if you plan to become a freelancer some day, now is a good time to start working on your network.

The third talent that you need is the ability to sell. Because technical writers tend to be somewhat introverted, the idea of selling is often horrifying. “Can’t I just do a good job?” they wail. Doing a good job is, of course, critical to your freelancing success, but doing a good job doesn’t happen until you get the job—and for that, you need the ability to sell. Selling doesn’t have to mean acting like a smarmy salesman, but it does mean that you need the confidence to go to a potential client, present a proposal, ask for a particular amount of money, and ask for the project. *If you cannot or will not do any selling, you cannot make it as a freelancer.*

How much experience do I need to begin freelancing?

There are no official rules for this, but it usually takes a new writer about three years to learn the craft. In addition to mastering technical writing skills, you also need business skills. Many writers spend the first year or two of their career working as a contractor, then work as a staff employee for a few years, and then look at freelancing after that. A minimum of three to five years experience is probably a good idea before you dive into freelance work.

We've already given you two mantras:

- The real estate agent's mantra: location, location, location
- The technical writer's mantra: audience, audience, audience

Time for one more. The freelancer's mantra should be:

> Network, network, network

B Resources

What's in this appendix

- General technical writing
- Editorial
- Audience and task analysis
- Information design
- Indexing
- Management
- Single sourcing
- FrameMaker
- XML and DITA
- Other tools and technologies
- Professional organizations and conferences
- Social networking sites
- Ergonomics
- Job listings

This appendix lists some print and online resources for technical writers.

NOTE: The web site addresses in this appendix were accurate at the time of printing. If an address does not work, you may be able to find the information by going to the site's home page and doing a search on the topic.

General technical writing

Mailing lists

- techwr-l

  Mailing list for technical writers, technical editors, and others working in technical communications. Go to the techwr-l web site to subscribe:

  www.techwr-l.com

  Traffic and off-topic chatter on the list is very high, but you can find some valuable information every now and then. You can also search the archives for particular information.

- stccicsig-l

  Mailing list for technical writers interested in consulting and independent contracting (CIC).

  To subscribe, go to:

  mailman.stc.org/mailman/listinfo/stccicsig-l

Editorial

Print

- *The Chicago Manual of Style,* The University of Chicago Press, 15th edition, 2003, ISBN 9780226104034. The technical editor's bible. The companion web site to this must-have reference is at:

  www.chicagomanualofstyle.org/home.html
- *The Elements of Style,* William Strunk and E.B. White, 4th edition, 1999, ISBN 9780205309023. A classic style reference.
- *Microsoft Manual of Style for Technical Publications*, 3rd edition, 2004, ISBN 9780735617469.
- *Read Me First! A Style Guide for the Computer Industry,* 2nd edition, 2003, ISBN 978-0131428997.

NOTE: Your company may distribute its own style guide and dictionary. Be sure you have the latest releases of those documents.

Web sites

- www.scriptorium.com/Standards

  Scriptorium Publishing's style guide.
- whatis.techtarget.com

  A site that defines acronyms and technical terms.

Audience and task analysis

- *User and Task Analysis for Interface Design,* JoAnn T. Hackos and Janice C. Redish, 1998, ISBN 9780471178316.
- *The Persona Lifecycle: Keeping People in Mind Throughout Product Design,* John Pruitt and Tamara Adlin, 2006, ISBN 9780125662512.

Information design

- *The Visual Display of Quantitative Information,* Edward R. Tufte, 2nd edition, 2001, ISBN 9780961392147.
- *Show Me the Numbers: Designing Tables and Graphs to Enlighten,* Stephen Few, 2004, ISBN 9780970601995.
- *Dynamics in Document Design: Creating Text for Readers,* Karen A. Schriver, 1996, ISBN 9780471306368.

Indexing

- *Indexing Books,* Nancy C. Mulvany, 2nd edition, 2005, ISBN 9780226552767.

Management

Print

- *Information Development: Managing Your Documentation Projects, Portfolio, and People,* JoAnn T. Hackos, 2006, ISBN 9780471777113.

Mailing list

- stcmgmtpic-l

  Mailing list for technical writers interested in management issues.

  To subscribe, go to:

  mailman.stc.org/mailman/listinfo/mgmtsig

Single sourcing

- *Single Sourcing: Building Modular Documentation,* Kurt Ament, 2002, ISBN 9780815514916.

FrameMaker

Print

- *Publishing Fundamentals: Unstructured FrameMaker 8,* Sarah S. O'Keefe, Sheila A. Loring, Terry Smith, and Lydia K. Wong, 2008, ISBN 9780970473349.

Web sites

- www.adobe.com

  Adobe Systems makes FrameMaker software. The web site contains product information, support information, printer drivers, and more.

- www.frameusers.com

  An unofficial FrameMaker resource site. It contains information about FrameMaker service providers, add-on software, and more.

- www.frame-user.de

  Another unofficial FrameMaker resource site. If you speak German, this is the site for you. Even if you don't speak any German, check it out. The web site creator designed the site to look like a FrameMaker document; it's quite a unique look for a web site!

- wiki.scriptorium.com

  Scriptorium Publishing's wiki features free versions of workbooks that explain how to use the unstructured and structured interfaces of FrameMaker.

Mailing lists

- framers@frameusers.com

  A mailing list for FrameMaker users. The framers list has been around for at least 10 years; its membership includes both new FrameMaker users and experts, some of whom have subscribed to the list for many years. List traffic is heavy but generally fairly well-focused on issues related to FrameMaker.

To subscribe, send email to:

majordomo@frameusers.com

In the body of the message, type:

subscribe framers

- framers@omsys.com

  Similar to the framers list based at frameusers.com. Many topics are posted to both lists. This framers@omsys.com tends to be more focused on power user issues than the other list. The framers@omsys.com list also has less traffic than framers@frameusers.com. To subscribe, send email to:

  majordomo@omsys.com

  In the body of the message, type:

  subscribe framers

XML and DITA

- xml.silmaril.ie

  The XML FAQ. A resource for those who are new to XML.

- www.ibm.com/developerworks/xml/library/x-dita3

  The DITA FAQ. A good starting point for learning about DITA.

- www.scriptorium.com/papers.html

  Scriptorium Publishing offers free white papers on XML, DITA, and other topics.

Other tools and technologies

Web sites

- www.pdfzone.com

  A web site with resources for Adobe Acrobat users. It includes tips and tricks, resources, and more.

- www.guru.com

  An excellent site for freelancers of any type.

Mailing lists

- xml-doc

  A mailing list for technical authors interested in XML. To subscribe, visit:

  groups.yahoo.com/group/xml-doc/

  You will need to register with the web site to sign up for the list, but registration is free.

- Help Authoring Tools & Techniques (HATT)

  A mailing list for help authors. To subscribe, visit:

  groups.yahoo.com/group/HATT/

  You will need to register with the web site to sign up for the list, but registration is free.

- wwp-users

  A mailing list for WebWorks Publisher and ePublisher Pro users. To subscribe, visit:

  groups.yahoo.com/group/wwp-users/

  You will need to register with the web site to sign up for the list, but registration is free.

Professional organizations and conferences

- Society for Technical Communication (STC), www.stc.org

  The STC is a professional association for technical writers and other technical communicators. The STC web site includes information about the organization, its conference, and links to various local chapters. Chapters are mostly in the United States, but there are also several in other parts of the world.

- American Society for Training and Development (ASTD), www.astd.org

  The ASTD is a professional association for trainers and instructional designers.

- Association for Computing Machinery Special Interest Group for Documentation (ACM SIGDOC), www.acm.org/sigdoc

  The ACM SIGDOC is for senior technical communicators.

- Institute of Electrical and Electronics Engineers Professional Communications Society (IEEE PCS), www.ieeepcs.org

  The IEEE PCS helps technical communicators and engineers develop their writing skills. The IEEE PCS sponsors a conference each fall.

- tekom, www.tekom.de

  An organization for technical writers in Germany.

- WritersUA, www.writersua.com

  WritersUA sponsors conferences focused on providing user assistance and online help.

Social networking sites

- community.toc.oreilly.com

  The O'Reilly Tools of Change for Publishing community features forums, blogs, and events of interest to those working in publishing.

- thecontentwrangler.ning.com

  A social networking site for technical communicators.

- toc.oreilly.com/startwithxml

  The Start with XML project offers blog posts and forums about the motivations and methods for implementing XML workflows.

- www.linkedin.com

  A networking site for all professionals.

Ergonomics

- Occupational Safety & Health Administration, www.osha.gov

  The web site for the Occupational Safety & Health Administration contains a great deal of information about ergonomics and work-related injuries.

- Typing Injury FAQ, www.tifaq.com

  The Typing Injury FAQ explains repetitive motion injuries related to typing and how to prevent them.

Job listings

Most of the major job sites, such as monster.com and dice.com, list technical writing positions.

Many STC chapters post job listings on their web sites. To find your local chapter's web site, go to the main STC web site, www.stc.org.

C Sample doc plan

The following pages contain a sample doc plan. The sample does contain information about financial matters, but it does not contain specific pricing information.

Documentation plan

Mr. Rip Van Winkle
WinkleWart Technologies
Research Triangle Park, NC 27709

March 2, 2009

WinkleWart Technologies' WinklePlan software allows end users to import data about their mattress manufacturing environment, run simulations of different sleep environments, and output reports. The simulation process helps end users predict how new equipment and other changes will affect the quality of sleep.

The WinklePlan software needs documentation, including a *User's Guide, Quick Start,* and HTML Help. This proposal outlines how Scriptorium Publishing will develop and deliver the documentation.

Audience

The audience for the WinklePlan documentation consists of managers who work in a mattress manufacturing environment. Typically, the software users would be analysts who understand the environment very well and can build an accurate simulation model.

These individuals are typically somewhat computer-literate. They are familiar with Microsoft Windows operations. They are also likely to be familiar with Microsoft Excel.

Deliverables

This section outlines the scope of the project and what services Scriptorium Publishing will provide.

User's Guide

Scriptorium Publishing will develop a task-oriented *User's Guide* for the WinklePlan software. The *WinklePlan User's Guide* will explain the concepts underlying the software and how to use it. The document will contain the following sections:

- Introduction
- Importing data
- Sleeping
- Generating reports

The book will include a table of contents and index.

Scriptorium Publishing will write the manual, edit it, lay it out, index it, and production edit it.

Estimated page count: 100 pages

Quick Start

The *WinklePlan Quick Start* will explain how to complete a single project using WinklePlan software. It will provide a specific example so that new users can understand how the software works.

Because this book will be very short, we do not plan to include an index. The book will have a table of contents.

Scriptorium Publishing will write the manual, edit it, lay it out, and production edit it.

Estimated page count: 30 pages

HTML Help

To produce the online help, Scriptorium Publishing will to convert the content in the *User's Guide* to HTML Help.

Estimated topic count: 80 topics

Deliverable format

The books will be delivered in PDF format suitable for professional printing.

The online help will be delivered in HTML Help format.

Upon payment of all invoices, WinkleWart Technologies can also request and receive the FrameMaker source files.

Receivables

WinkleWart Technologies will provide the following to Scriptorium Publishing:

- Source files for the current version of the documentation.
- Any feature lists, functional specifications, product design documents, or asset lists that might be helpful in planning the project.
- A working version of the software with a stable, frozen interface. If the interface is not frozen, WinkleWart Technologies will note which portions are least likely to change.
- Ready access to technical experts.
- Timely reviews. Technical experts will return reviewed chapters within three business days.

  Note that a changing interface, difficulty in contacting technical experts, and delays in reviews will lead to change orders, an increase in the total price of this project, and delays in delivery.

Tools

Scriptorium Publishing will use some or all of the following tools and technologies:

- Adobe FrameMaker (the structured interface)—for authoring content based on the Darwin Information Typing Architecture (DITA) open standard
- Paint Shop Pro—for creating screen captures
- Adobe Acrobat—for producing PDF files
- DITA Open Toolkit—for creating HTML Help

Location

Work will be performed at the offices of Scriptorium Publishing.

Deadlines

Exact deadlines will be negotiated between Scriptorium Publishing and WinkleWart Technologies.

Intellectual property

Upon receiving payment in full for all invoices, Scriptorium Publishing will transfer all copyrights for the documentation to WinkleWart Technologies. Scriptorium Publishing retains ownership of any processes developed to complete the project.

Financial

Costs

Based on the information provided (PowerPoint presentation with outline of product), the cost for each deliverable is in the table on the next page.

Note that changes in the product will lead to change orders and a corresponding increase in cost. Furthermore, these estimates are based on the page counts provided in the deliverables section. If the final page count is more than 10 percent higher than the estimates in this proposal, change orders will result.

Fixed-price bid

Deliverable	Cost
User’s Guide	$
Quick Start	$
HTML Help	$
Book and HTML Help design	$
Total	**$$$$**
This is a fixed-price bid.	

Any estimates made by Scriptorium Publishing for the cost of any services shall be estimates only. Whenever estimated prices are quoted, Scriptorium Publishing shall use all reasonable efforts to perform the relevant services at the estimated price but in no event shall such estimates constitute a fixed price or not-to-exceed price agreement between the parties in respect of such services.

Whenever the parties agree to a fixed price or a not-to-exceed price, the applicable estimate must expressly so state.

Payment

For the fixed-price portions of the project, Scriptorium Publishing will invoice WinkleWart Technologies for one third of the cost upon approval of this proposal, one third after 30 days, and the final third at the project’s conclusion.

For the hourly portions of the project, Scriptorium Publishing will invoice WinkleWart Technologies for hours incurred at regular intervals (weekly or every other week). The invoices will include a breakdown of hours by task (for example, writing, editing, and illustration). Upon request, a breakdown of hours by individual and by day are available.

Invoice terms are net 15, and a charge of 1.5 percent per month will be applied to any overdue account. Overdue accounts that result in a collection by an attorney will be charged for reasonable attorney fees.

Cancellation

Should WinkleWart Technologies cancel this project, Scriptorium Publishing will charge WinkleWart Technologies at $/hour for the work already completed up to the date of notice of cancellation and for work required to terminate the project after notice of cancellation.

Proposal duration

This proposal is valid until March 16, 2009.

Agreement

By signing below, Scriptorium Publishing and WinkleWart Technologies agree that the specifications, costs, and terms described in this proposal are acceptable to both parties, and that work may begin.

Scriptorium Publishing Services, Inc.

By:____________________
Sarah S. O'Keefe

Date: March 2, 2009

WinkleWart Technologies

By:____________________
Rip Van Winkle

Date:____________

Index