March 2022

**Docs for developers**

An engineer’s field guide to technical writing

By Jared Bhatti, Zachary Sarah Corleissen, Jen Lambourne,David Nunes, Heidi Waterhouse



Docs for developers: book review

Reviewed by Natalia Dowding

Unlike many other publications in the Technical Writing field, this book is written for developers who face the necessity to create their own documentation, rather than for writers. The aim of the authors was to equip developers with a comprehensive guide to creating successful and effective documentation.

This relatively small volume covers several topics, leading the aspiring writer through all steps of documentation creation, from audience analysis to maintenance and ~~deprecating~~ removing the obsolete content. Each of these topics is a big subject on its own. The authors opted for extracting the essentials and leaving the details behind. There are extra references and a list of useful resources aat the end of the book.

To illustrate the process and the pitfalls that developers who are undertaking writing tasks encounter, the authors include a story of a fictional software company named Corg.ly, that develops a creative toolkit for translating dog barks into English.

At the end of the development process, the company faces the need to create the required user documentation for their product, and each chapter demonstrates through daily situations, how the writing process should be tackled.

## 

## What is between the book covers

The book follows the documenting process step by step in its working order, from planning and user research to maintenance. The first part of the book deals with the writing and publishing aspects of the job, taking us through each stage of the process. The second part introduces the practicalities of gathering feedback, establishing quality metrics and maintaining the content.

Throughout the book, the authors show lots of sympathy for developers. They seek to convince the reader that they are not alone to have problems with writing. And this is not because of their lack of ability, but because writing is difficult. Feeling stuck during their writing pretty is common among the writers and the authors offer several ways to overcome this problem.

Indeed, writing is not what developers do daily and there are significant differences between coding and writing.

Facing a blank page can be daunting to anyone, including experienced writers. The book claims that a good amount of planning is key, hence writing is part of a process.

## Getting into the details

 In trying to help the developers, the authors employ a systematic approach to writing process.

In the first chapter, the authors introduce the reader to the audience analysis.

The authors insist that collecting some data about the user is better than nothing. Understanding the needs of your users can make all the difference in deciding on the purpose of your documentation and on what should be included in it.

The authors describe the sources of gathering user information, such as ticket data, user interviews, surveys. After that they discuss methods of summarising user information by creating user personas, user stories and user journey maps.

## Planning and drafting documentation

Once the developers have learned more about his audience, the authors move on to discussing the outline of the future documentation. Here the reader will find about the most common content types and will learn how to plan his documentation.

Time pressure often overshadows the importance of preparation. To challenge this situation the authors, bring forward a case of writer's block. They argue that analysing the audience and planning the content may help to alleviate the problem of writing block when facing a blank page.

For developers, who might see writing as a vague, creative process, this will help them to start on the right foot.

The authors start by introducing different types of content. They explain the purpose of each type and provide a simple outline for each type.

Text, letter

Description automatically generated

Figure Conceptual guide

For example, tutorials are procedural while design notes are conceptual. For each category authors discuss the structure of the document: a how-to guide will likely contain a number of steps to achieve a necessary result.

 The authors will start by explaining how to design an outline, and then introduce the most common elements of content, such as lists and headers.

Authors will explain how different types of content serve a specific user goal, for example, the purpose of “Getting started” documentation is to help the new user to get up and running, while the reference documentation contains the information he will come back to when coding.

The authors will go through most common types of content and explain their purpose and structure, such as:

* Code comments
* README files
* Getting started guides
* Conceptual documentation
* Procedural documentation
  + How-to-guides
  + Tutorials
* Reference documentation
* Glossary
* Change documentation
* Troubleshooting documentation

In the end of the book the reader will find a list of resources containing templates for different types of content.

As a result, the developer will come up with the outline of the content and a clear understanding of what he needs to write (see Figure 1).

What I find particularly attractive in this book, is that with any new task the authors provide a checklist for the developer to go through before moving to the next task (see Figure 2).

Table

Description automatically generated

Figure 2 Content outline

Table

Description automatically generated

Figure 3 Checklist for planning documentation

## Trial and error

Drafting is, probably, one of the most difficult stages of a writing project. Both coding and writing are creative processes, but they are unquestionably done differently. What authors achieved here is that they present drafting as a logical, methodical job, rather than spontaneous and mysterious activity. This must seem very appealing to the developers, who often see writing as impulsive work, dependent on waves of inspiration.

Introducing structure is important for a beginner writer and having a **pattern or template** to follow eliminates the difficult task of designing some sort of structure from scratch,

With editing - another challenging task for the beginner writer - the authors propose to do it in several runs. Each time you look at one aspect of the document. As before, the authors provide a list of aspects to check. In this way, a confusing and chaotic process becomes something methodical, and, therefore, attainable. The authors draw parallels with coding: editing is like testing.

It was particularly interesting for me to read about providing and receiving feedback. The authors talk about *plussing technique* as a constructive way to provide a critique. The idea behind is that the reviewers may criticise only if they can give a practical suggestion in return.

## Visual elements and code samples

After having discussed the purely textual aspect of the document, the authors embark on code samples and visuals.

Unlike writing, dealing with code samples lies within the developers' comfort zone.

 Facing a relentless stream of new technologies, apps, and tools to take on board, developers know all too well how important code to und samples can make a huge difference to the comprehension of new material.

To help the developers to create effective code samples, the authors summarise the central principles to follow: the code samples should be well explained, concise and simple. If the sample is executable code, then it should work and be trustworthy. In summary, code samples should demonstrate exactly what they are expected to do, every additional code elaboration should be avoided.

The authors also bring up the subject of designing effective code samples. They discuss the questions language choice, code presentation, testing and maintaining them relevant.

Visual materials are another powerful tool to improve users' comprehension. As with code samples, developers deal with charts and diagrams in their daily life.

The briefly touches upon three types of visual materials:

* Screenshots
* Diagrams
* Video

Much can be said of numerous tools and apps available on the market, but the authors keep their focus on the principles. They stress the importance of considering accessibility issues, to maintain consistent shapes and colours in diagrams and keeping them clear.

The authors also leave some resources for developers to look at.

Diagram

Description automatically generated

Figure 4 Example of flowchart

Diagram

Description automatically generated

Figure 5 Overcomplicated flowchart

After all the writing and other materials have been finished, it is time to publish the documentation. The chapter that discussed the publishing stage is not large. Here the authors concentrate on finalising and approving documentation, choosing the publishing medium, and planning.

## Improving and maintaining documentation

Gathering the feedback and evaluating the quality are often overlooked and the authors explain how important it is to maintain the communication with the user to build up the trust

 The authors discuss first the practices of collecting the user’s feedback, then of integrating it into the documentation.

Several methods of feedback collection are examined, including collecting user feedback directly through documentation pages, using support issues, sentiment analysis, user surveys and user councils.

The feedback provided needs to be integrated into the documentation. The integration process that the authors describe here mirrors that of resolving incidents in the software industry. It starts with triaging and if the issue cannot be resolved, the request is sent back to the user, asking for more details.

The authors emphasise that documentation is a way of connecting to the users; they advise to stay in touch with them, follow up on submitted incidents and thank them for feedback.

In this chapter the authors will talk about how to create a strategy for collecting analytics and to make sure those analytics are useful in measuring documentation quality. They discuss different ways to assess the success of documentation: from an organisational point of view or users’ point of view. But they start from defining the quality of the documentation.

What the authors achieve here is that they show how to define the concept of quality and, once that is done, choose the useful analytics, rather than going the other way round.

The quality assessment in the book is divided into two categories: functional and structural. Assessing the functional quality allows evaluating if the documentation serves its goal by being accessible, purposeful, accurate, findable, and complete.

In its turn, the structural quality helps to decide if the document is written and structured well by being clear, concise, and consistent.

After that the authors go through some examples of analytic tools, such as page views and bounce rate. Then they explain how to choose appropriate analytics to evaluate the quality of content (see Figure 5).

Table

Description automatically generated

Figure Questions about quality and corresponding metrics

## Information Architecture (IA)

The authors reserve a separate chapter to Information Architecture, a discipline studying structuring and labelling of the content.

The authors bring attention that organising content is important to ensure that the user can find what he wants and that the documentation can grow without breaking the content structure.

The chapter touches upon the background: the sequential, hierarchical and web-like types  of content structure, followed by the main site navigation principles, including information categorisation, landing pages, navigation cues.

After that the authors cover the practical aspects of organising content:  auditing the existing content, outlining the new information architecture for your documentation and migrating the content to the new structure.

When speaking about outlining the new information architecture, the authors repeatedly advise to refer to users’ expectations. In deciding upon the content organisation, the reader will be briefly introduced to the card sorting method of information categorisation and then testing the new information organisation by collecting users’ feedback.

## Documentation Maintenance

At last, the authors touch on the topic of maintenance. Ensuring that the documentation is updated regularly and is aligned with releases is critical to maintaining the trust of the users. In a few short sections, the authors manage to distil the best practices for maintenance and deprecating documentation.

When talking about the maintenance, the authors go through the list of essential points, including planning, assigning persons, responsible for documents, coordinating documentation with releases, automating the maintenance, checking the links and the grammar.

What was especially interesting is that the authors recommend rewarding the documentation maintenance efforts, reminding that recognition is a great way to motivate a person.

In the end, the authors mention the process of announcing the documentation obsolete when corresponding software becomes obsolete. They point out the importance to retain the obsolete content and to flag it such to warn the user. Only when the documentation is not needed or not helpful to the user, it may be removed.

## Reviewer’s comments

The first impression from the book was that in simplifying the presentation of material the authors went a little too far.  For example, the story of Corg.ly would look appealing to any dog lover, it seemed to be oversimplified.

However, after giving it another thought, I came to appreciate the book much more.

The book was conceived as a practical guide to developers writing their documentation. As a "How to" guide to writing. And it achieves this purpose.

The authors manage to squeeze a great amount of material covered in 200 pages. The book is a good example of practices presented in it: clear, concise and consistent.

Information in the book is covered in a very logical and succinct way.  But one definition surprised me. It was the explanation of a paragraph. Presenting a paragraph as a section of text which provides a context or may contain a story, seems to be confusing and hardly logical.

 Overall, I feel that the book might benefit from more examples and resources.

Having said that, I like the overall style of the book: clear, cheerful, and sympathetic. Bearing in mind that the community of developers is far from being uniform and the authors did a very good job of accommodating this highly diverse crowd.

I would highly recommend this book to anybody who is looking to create documentation. Step by step, holding the book as a reference and a guide, the book is a great guide for the beginner writer to overcome all the hurdles and to succeed with well written and well organised documentation.