Packt

Author Guidelines

Your guide to writing a book with Packt

Packt Books – An Introduction

Here at Packt, we're passionate about helping developers share their tech knowledge and insight. As you probably already know, all of our books are created ***by developers, for developers*** – so as author and expert developer you play an absolutely crucial role!

We want to ensure that all Packt content enables readers to:

- **Explore** – the latest cutting edge tech and keep tech skills razor sharp
- **Learn** – from expert developers with hands-on experience and unique insight
- **Evolve** – by learning smarter, faster – in a way which suits the reader

We've specially selected you to be a Packt author as you're a tech expert who can help the developer community to grow their skills by sharing your expertise, insight, and practical working knowledge.

Now it's time to start the writing process!

Whether you're a seasoned author or it's your very first time, we understand that writing a book can be a daunting process. This is why we've created this guide to the writing and book development process to give you a helping hand along the way.

As an author these guidelines will help you to:

- properly **plan, structure,** and **write** your book
- **understand what to expect** during the book writing process
- ensure you create a **well written (and commercially successful!)** book

Most importantly, remember that you're not alone! We'll be working closely with you along each step of the way to support you in creating a book that you're certain to be proud of.

We hope you find these guidelines useful – please take time to read through them before writing your book.

Packt›

Contents

01 Plan

02 Structure

03 Write – General Principles

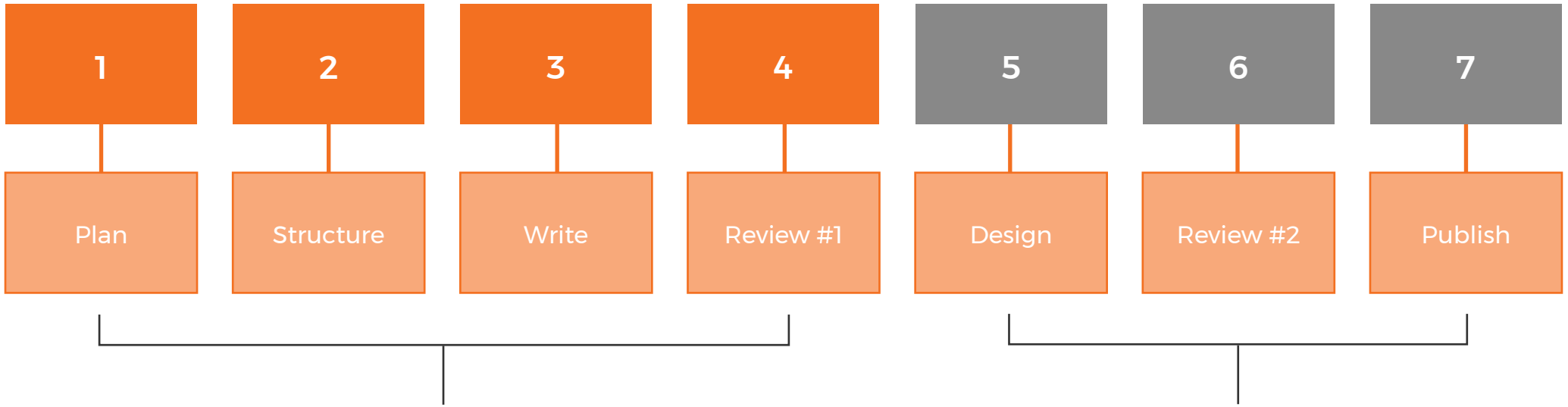
04 Write – The First Draft

05 Review

01 Plan

The Writing Process

Writing and publishing involves a number of different stages, as you can see below:



This guide covers **stages 1-4** – from planning through to the first review.

Stages 1-4 are where the magic happens!

Stages 5-7 happen in-house, so although you (the author) will be involved in few ways (such as proof reading), these stages are much less intensive.

Let's look at each of these stages in turn.

The Planning Process

PLAN

- Schedule
- Define Audience

Have you ever heard the saying
'Fail to prepare, prepare to fail'?

Well, when it comes to writing a book it's entirely true – **planning is everything!** There's also a 'right way' to write a book. Before we look at that, however, let's discuss the writing discipline.

Timeline and Schedule

Your editor will give you all the information you need about the schedule for writing your book. It's a really good idea to speak to them properly about this, and stick to the schedule as much as possible to make your life easier. Here are some handy tips for keeping to schedule:

1. **Be disciplined:** Not many of us like discipline! However, whether you're learning a programming language or writing a book, it's extremely important to set aside time – even when you don't feel like it. **Set aside an hour or two of *uninterrupted* time every day.**
2. **Set goals:** Write approximately 1,000 words a day. This will help you get into a rhythm and a flow – before you know it you'll be making progress in no time!
3. **Work to an outline:** This will keep you focused. We'll cover the outline in the next section.

Please be aware that it is normal to feel frustrated at times. You might even suffer with writer's block. Don't worry about it – it's completely normal, and every author goes through it. Writing a book is difficult but it's also a lot of fun – and a great thing to tick off your bucket list!

TIPS

- Go for a walk
- Eliminate distractions
- Change your environment
- Read a book
- Freewrite
- Listen to music
- Call a friend

Who is Your Audience?

This is probably **the most important question to ask** during the planning stage.

Who is the audience, **how** will they interact with the book and, **what** are their needs? You will need to be specific!

Always keep in mind your target audience: busy, working developers who don't always have the option of taking time out to learn new things. They're constantly learning in small sections, in a way that fits around their work. This isn't formal learning – it's driven by what they need to know for work, and is usually done on a 'need-to-know' basis.

You'll need to consider the knowledge level of the reader when they start the book.

Busy developers need information that is:

› Presented in a **logical and linear** fashion.

› **Easy to access** – so they can find what they're looking for quickly and efficiently.

› A **micro learning experience** – the sections and chapters in a Packt book should be **decontextualized**; in other words, the reader shouldn't need to read around for context in order to get the answers they need.

› **Practical and useful** in the real world.

Knowledge Level	Starting Point: Before Your Book	End Point: After Reading Your Book
New to tech	No knowledge or exposure	***Working knowledge:*** Basic proficiency with technology.
Working knowledge	Basic foundational knowledge/proficiency	***Advanced knowledge:*** A professional standard. Able to use the technology for their job.
Advanced	Professional proficiency/good working knowledge of tech	***Expert knowledge:*** A power user. Refinement and mastery. Focus on niche areas.

Who is Your Audience?

Set aside some time to **build a profile of the audience** and then write with that persona in mind.

Ask yourself the following questions, and write down your answers:

Question	Explanation
Who is your audience?	1. A general description of your audience
	2. What knowledge can be assumed?
	3. Key characteristics/unique challenges
What is important to them?	1. What is the key problem/issue your audience is dealing with?
	2. What are the need-to-know features?

1 2 3 This is just a start – you can probably think of some more questions of your own. The main thing is that you **understand your reader** (or **customer persona**) before you start writing for them.

Competition

Unless we are first to market, your book might fit into a crowded marketplace with a variety of alternative or substitute titles – just search Amazon or Safari for a few examples.

How does your book fit into the mix? What is unique about it? Once we've established this, then we can start to identify the **unique value proposition** that you're offering the working developer.

Take some time out to look at the best-selling books: check the contents pages and the topics they cover, get some insight into customers by reading the reviews, and check the content to see how the author has approached the topic. This will give you an idea of what customers like and what they are responding to.

02 Structure

How to Structure Your Book

02

STRUCTURE

- Table of contents (TOC)
- Review TOC

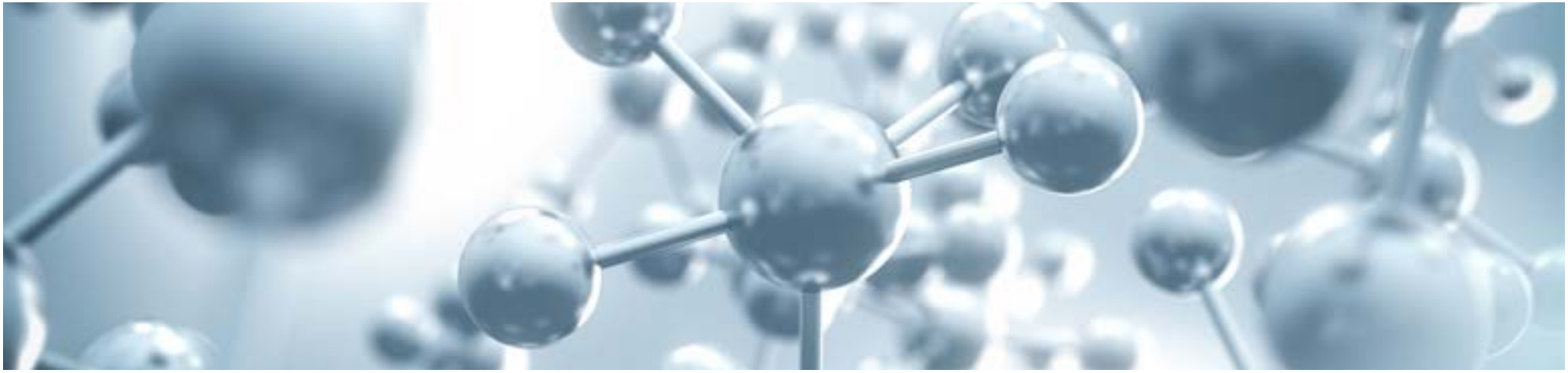
Once you've completed your plan, it's time to move on to the next stage in the process – developing the **structure** of the book.

How to develop the table of contents for your book

When buying a book, developers use the **table of contents** to help decide whether the book is right for them. **A good table of contents** can be the difference between whether or not your book sells!

Remember your **title, subtitle,** and **audience:** how are you going to deliver that promise?

Break it down into chunks, and that's the structure of your book. **Think of the table of contents like a map** – it helps the reader navigate your text. They should be able to dip in and out of it easily, or follow the path of the book's development.



Process

A book can be split up into activities which take place in sequence. Users perform these activities as they interact with the technology.

You've been specially chosen by Packt to write a book because your expertise in your field means that you will know what this process is. Here are some guidelines to develop this process:

1. **Process overview:** Do some research – how have other authors tackled the subject? What is the process or use cases that the reader will go through?
2. Create a **first draft table of contents.**
3. **Test this draft with other experts and with your editor** for areas that might have been missed – make sure you do this as it will really help you to get things right first time!
4. Finalize the table of contents.

In doing this, there are certain requirements that you will need to have considered; you looked at this during the planning stage.

Consider the user profile

You have considered the level of knowledge that your audience already has. Does the structure of your book match your readers' expectations and proficiency?

This is important when writing a table of contents – if you use terms that the reader won't understand then they won't connect with your book.

Now consider the reader's journey

This graph illustrates the learning experience that a reader goes through, and helps us to anticipate how they're going to feel as they use the book. Understanding this graph helps us to structure the book.

Structure

Whenever someone first begins learning a new subject or skill, there is a period of accelerated learning as **it is fun and fulfilling**.

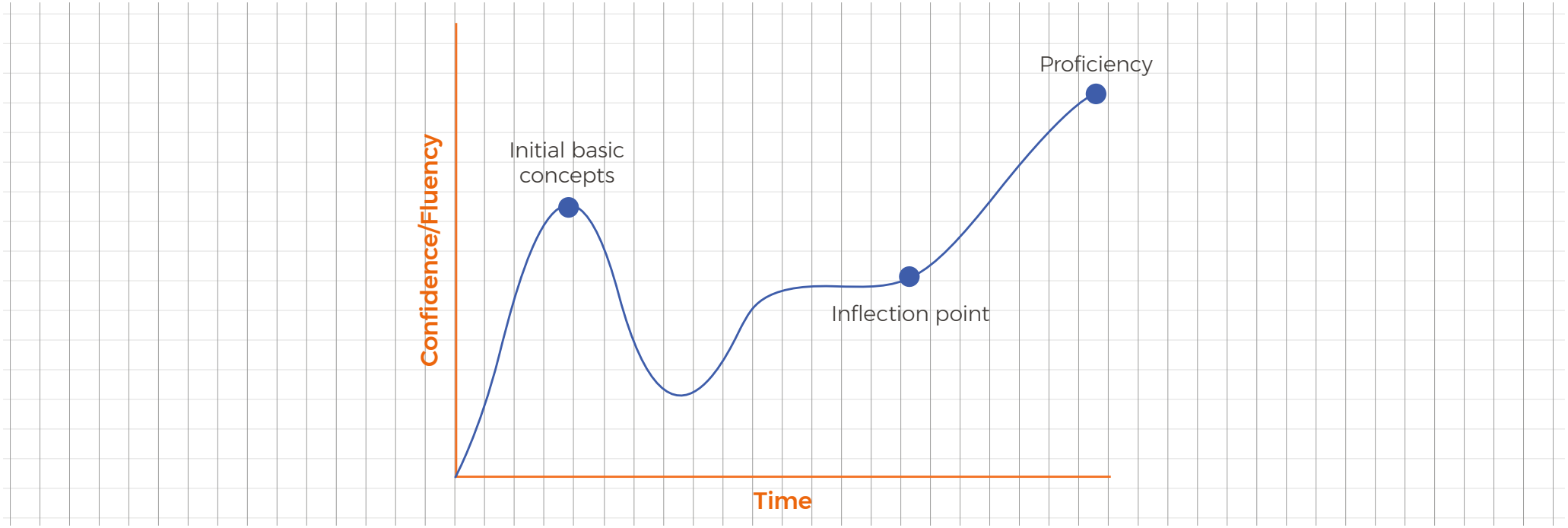
Shortly after they begin to realize how difficult a new skill actually is, they run out of the more immediate and basic concepts and start to feel ***discouraged***.

At a later point, the person might start to feel ***frustrated***. It's at this point that the brain begins neurally adapting to whatever it is they are learning. The brain starts pulling it deeper than surface level memorization, which means that it can do less processing to accomplish the same tasks!

Eventually, the person reaches the **inflection point**, which is when everything starts to 'click'. The learning becomes easier and accelerates the person to proficiency.

Using this illustration, you can predict the different levels of **confidence** a person might have as they learn a new subject, making it easier to prepare for what's ahead and not get stuck or give up at a low point.

Consider this as you decide how to structure your book.



How to Plan the Contents Page

General principles

Not all topics fit into a nice, neat, logical order, but these general principles are a good rough guide:

1. Go from **simple** to **advanced**

2. Go from **common** to **rare** – put the most useful information at the start of the book

3. Go from **early** to **late** – mirror the way that you'd conduct a project; for example, 'deployment' would be covered toward the end of the book.

Not sure what's right for your book? Then it might be that the order isn't especially important – choose whatever seems logical to you.

Consider the user experience

How will the reader interact with the book? For the most part, they'll be dipping in and out of it so it will be a form of micro-learning. The layout of your book should make it an easy-to-reference resource.

Does the table of contents make it easy for the reader to do this? Remember, the reader will care more about their **work topics** than the book itself – they've bought it because they want to solve a problem.

To make this easier, a good table of contents will be:

- **Well organized**
- **Easy to read**: Keep character titles to 50 characters or fewer.
- **Easy to understand:** Avoid using jargon or trying to be humorous.
- **To the point:** You don't need to be creative – get to the point and be specific.
- **Grammatically correct:** Spelling mistakes and bad grammar are a real turn off.
- **Practical:** Use action words such as build, create and explore.

...and will have a **logical progression.**

How to Plan the Contents Page

For example: If you were writing a book on Python programming, you might structure the sections as follows:

Introduction

Section 1: Essential concepts and skills

Section 2: Other concepts and skills

Section 3: Object-oriented programming

Section 4: Database and GUI programming

Appendix

If you were to break that down further into chapters, it might look like this:

Introduction: An introduction to Python programming

Section 1: Essential concepts and skills

- **Chapter 1:** Write your first program
- **Chapter 2:** Code control statements
- **Chapter 3:** Define and use functions and modules
- **Chapter 4:** Test and debut a program
- **Chapter 5:** Work with lists and tuples
- **Chapter 6:** Work with file I/O
- **Chapter 7:** Handling exceptions

Section 2: Other concepts and skills

Section 3: Object-oriented programming

Section 4: Database and GUI programming

Appendix

You'd then need to do the same for section 2 onwards, building them out into chapters.

Writing Chapter Sub-Headings

To expand your table of contents, start by writing the **introduction** and **bullet points** for each chapter:

Chapter introductions

The introduction to a chapter should be around 50–100 words explaining the content of the chapter. Describe in **three or four sentences** what the chapter is about.

Consider using the **5 Ws** (and one H): **Who, What, When, Where, Why,** and **How** to help craft it.

You should now have a chapter outline. Now break that down into logical sub-divisions. These should mirror the journey the reader will need to take in order to get there. Write this down in bullet point form.

Bullet points

Your bulleted list of learning outcomes will correspond with the main sections in the chapter. They will be:

- The **main topics** the reader will explore
- **Specific things the reader will do** with these topics

WHO

WHAT

WHEN

WHERE

WHY

HOW

Review

Finally, review your table of contents to check the following...

Question	Explanation	
Coverage	Is there anything missing?	☐
	Does the structure align with the reader's proficiency level?	☐
Reference	Does the table of contents help the reader navigate the content easily?	☐
	Is there a logical progression of content?	☐
	Easy to reference – can the reader dip in and out?	☐
Keywords	**IMPORTANT:** Chapter headings contain keywords for search optimization (table of contents appears on Amazon – so this part is very important!)	☐
Headings	To the point – specific and not too creative/funny	☐
	Grammatically correct – no spelling mistakes or bad punctuation	☐
	Easy to understand – headings relate to user proficiency	☐

03 Write

Before You Get Started

WRITE

- Style
- Write first draft
- Review process
- Revise draft

Now that you've planned and structured your book, the great news is that you're ready to start **writing** it!

Style

A Packt book is practical in nature, not theoretical as such – it isn't an academic textbook. Of course, all of our books contain theory, but it is **applied** theory that is useful to the working developer. It puts **knowledge into practice**.

A Packt book should:

- discuss the necessary details of a topic
- offer intuitive and informative explanations of concepts and theory
- show readers how to use these concepts and theories, and how to avoid common pitfalls

Get to the **practical core** of the book as quickly as you can, and then use the **practical examples** to teach the theory.

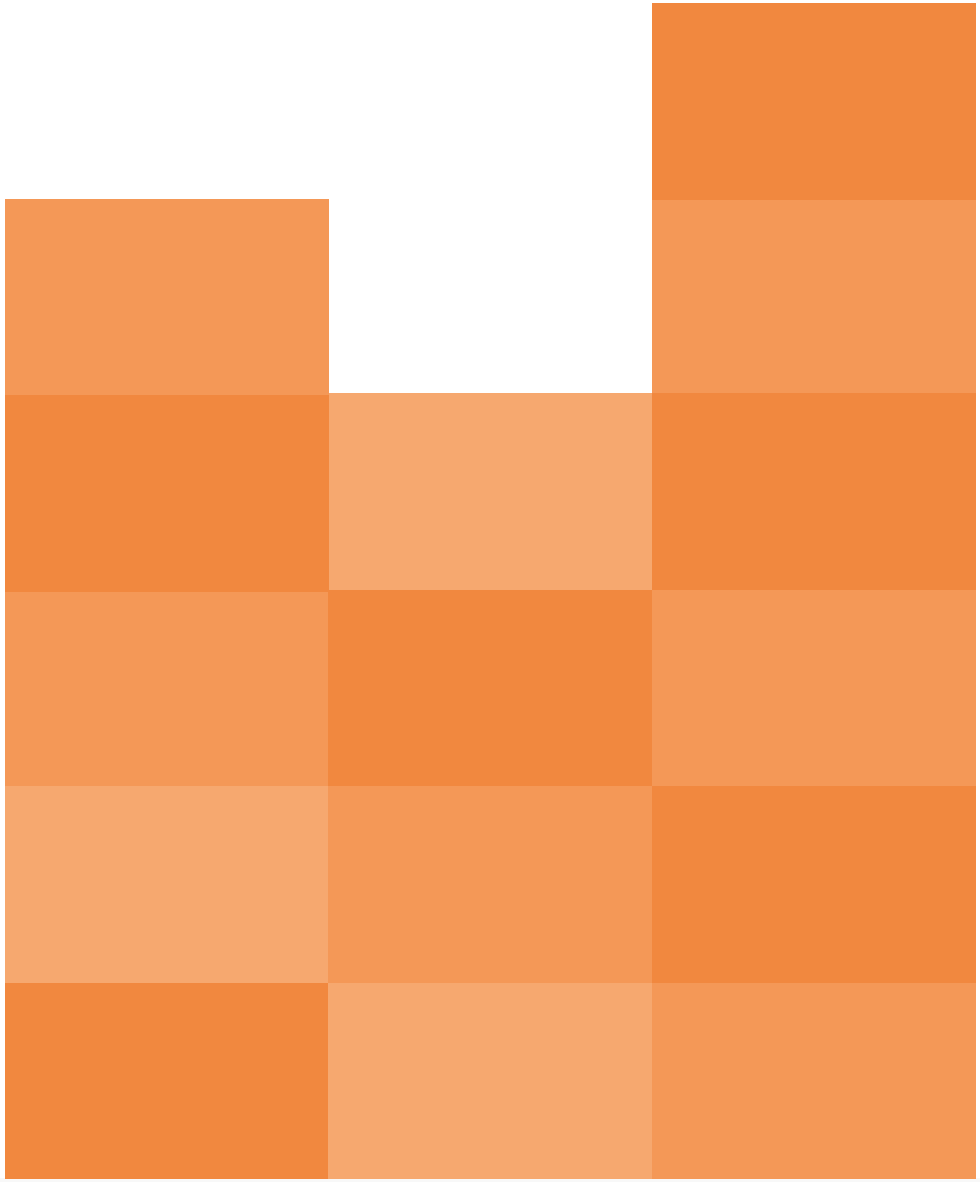
Packt books are written ***by*** developers, ***for*** developers.

Modular

Packt books are modular. This means that each chapter is a **standalone micro-learning unit.**

Think of each chapter like a lesson plan – it should relate to the chapter previously, but should also sit entirely independently of it. It needs to facilitate micro-learning. Not many readers will go through the book from cover to cover – they will dip in and out looking for the content that is relevant to them. For this reason, Packt books have a modular structure that lets the reader pick and choose the content they need, when they need it.

This means that just because you have written something in a chapter previously, you shouldn't assume the reader has read it. Write each chapter like an independent standalone module or lesson.



Language

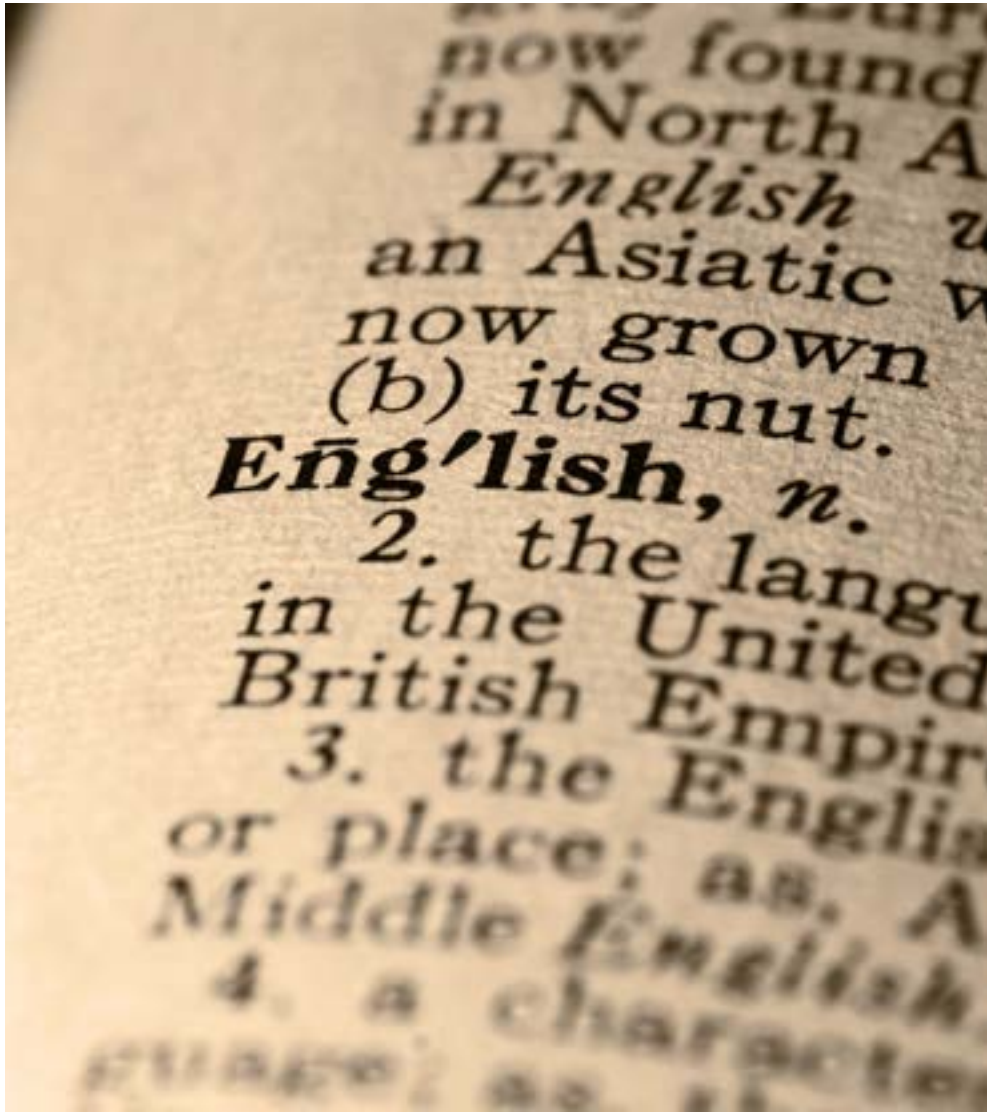
Packt books use American English. Your book should use **plain English**. Above all, readers are interested in what you have to say, but the way in which you say it may encourage them either to read on or to give up!

- *Use the language of everyday speech, not that of spokesmen, lawyers, or bureaucrats*

  (So, prefer **let** to permit, **people** to persons, **buy** to purchase, **rich** to wealthy, **show** to demonstrate, **break** to violate.)

  Complicated and long-winded sentences can make it really difficult to understand the **meaning** of a sentence – use **plain English** and **avoid waffle!**

- *Don't be too chatty.*
- *Don't be too didactic.* If too many sentences begin with Compare, Consider, Expect, Imagine, Look at, Note, Prepare for, Remember or Take, readers will think they are reading a textbook (or, indeed, a style book).
- Simple sentences help.

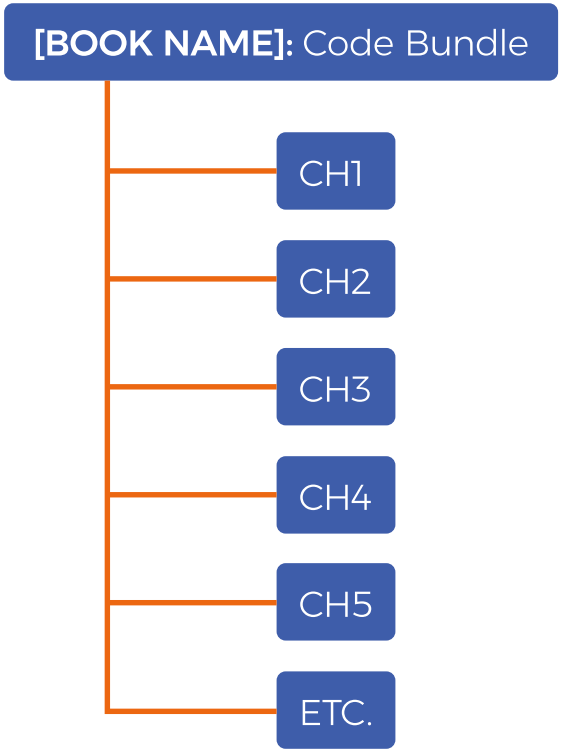


Code

Using practical code examples serves an important purpose: it illustrates the concepts by putting the learned material into action.

Try to ensure code never goes over a page in length. If you need to share a larger chunk of code, break it up with paragraphs of explanation at key points.

Include code that you want readers to see in the chapter text. It should be stored as follows:



Each chapter folder should contain the code bundle that is necessary to complete the exercises in the chapter – it is, in essence, a "code snapshot" required as a starting point for the exercises. If you have a project that you are building throughout the book then the starting point for one chapter would be the end point of the previous one. This is important as it allows the reader to pick up the book from a chapter without having to work through the previous chapters.

The download information should be captured in the **"Technical Requirements"** section at the start of each chapter. You will need to **list the GitHub URL** in order to direct users to the repository.

Submit your project files for the code download along with the chapter. You should put the code files for a chapter in a ZIP file with the format, ProductID_ChapterNumber_Code.zip. For example: **5482_02_Code.zip**.

Code is uploaded by the Technical Editor. One copy goes on the Packt website, and the other on GitHub. Your editor will add you as a contributor for that Git repo so that you can maintain the code after the book is published.

Images

Illustrating processes or architectural concepts with a diagram can make it much easier to understand your book.

Our production team can turn your rough diagrams into professional artwork, as long as your diagram is clear enough for us to understand.

Please note: We can't include images found on the web without permission; they need to be produced directly by you, or drawn by our graphics team. This includes images taken from official documentation.

Use screenshots to illustrate an instruction or show output. Try to take only the relevant window when possible. And make sure your windows aren't too big – resize them so that there isn't lots of empty space.

Image specifications: A separate document exists for image specifications that your editor will share with you. In short:

- Format: jpg/png/bmp
- Resolution: 300 dpi [minimum]
- Width: 5.5 inches (13.97cm, 139.7mm) [image width when placed in the book – maximum]
- Height: 7 inches (17.70 cm, 177.7mm) [image height when placed in the book – maximum]
- Image size: below 2 MB
- Image dimensions: below 4 million pixels (i.e. higher dimension value should be below 2000 pixels because image dimensions above 4 million pixels are rejected by Apple)

Plagiarism

A QUICK NOTE ABOUT PLAGIARISM...

Taking text, code, or images without the copyright holder's written permission leaves an author vulnerable to legal repercussions. It is never acceptable and the repercussions can be very expensive – don't be tempted. **Packt editors have specialized tools to check for plagiarized content, and we have a zero-tolerance attitude to plagiarism which will result in cancellation of your contract.**

It might be appropriate to quote from another source – that's OK, just make sure that it is referenced properly (see the example below:
Raschka, S. (2017). Python Machine Learning, Packt Publishing Limited).

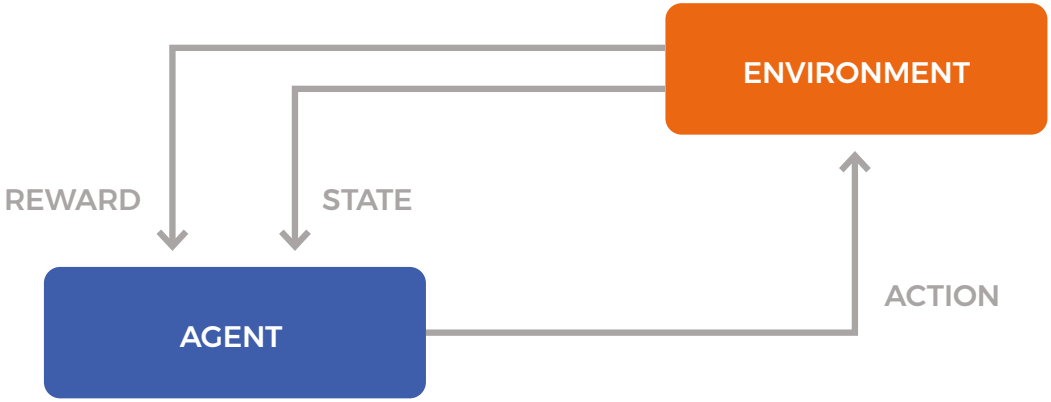
Lengthy quotes, however, can also cause legal issues, particularly if they're from published works. When in doubt just consult your editor, and they'll be able to help you out.

Images can be subject to copyright just the same as text. Even if you're able to find something through a simple Google image search, it is usually the property of someone else; Packt editors will also perform reverse image searches on anything in our books in order to prevent plagiarism. Even if you attribute the copyright owner alongside the image, we still face legal difficulties. We also can't claim fair use as we're a commercial entity. In short, make sure that all images are original!

Examples

You might be tempted to first write the theoretical background to a topic and then the practical; but **that would be a mistake!** Most people better pick up theory when it is presented alongside practical examples. This means the use of **examples** or **case studies** to illustrate concepts is important.

For example: If you were writing about reinforcement learning, then you might like to use the popular example of a chess engine. Here, the agent decides upon a series of moves depending on the state of the board (the environment), and the reward can be defined as win or lose at the end of the game:



Source: *Raschka, S. (2017). Python Machine Learning, Packt Publishing Limited*

04

Write –
The First Draft

Common Structural Components

04

All Packt books have the following parts:

01

PREFACE: This includes any necessary information about **setting up the development environment**.

02

INTRODUCTION: Familiarize the audience with the **big picture** and **major concepts** that you'll explore. Describe the content, the reason for its inclusion, and the sequence it is presented in. Describe the **learning outcome**. Finally, present the main topics you'll explore as bullet points.

03

BODY: The body of the text covers the main topics. It will be divided into chapters. The way the chapters are tackled will depend on the technology or topic.

04

LAST CHAPTER: A summary of everything that has been written. **Tell the reader what they have learned.**

Front Matter

Your author bundle contains a template for all front matter. This should be present in all Packt books and will contain the following:

1. **Title page**
2. **Copyright and credits**
3. **Dedication and acknowledgments**
4. **Mapt:** information about our subscription service
5. **Table of Contents**
6. **Contributor information:** that is, the author and technical reviewer's biographical details.
7. **Foreword:** this is optional. Add a foreword if you can obtain a well-known contributor. A **foreword** is written by someone other than the author - an expert in the field. Forewords help market the book: An opening statement by an eminent and well-published author adds credibility to your book. Forewords help by putting a stamp of approval on your work.

Preface

You will find the detail of the Preface in your author bundle. The preface is written by your Packt editor and (usually) contains some fairly standardized information. You will see the following sections:

1. **General introduction**
2. **Who this book is for:** A description of the reader profile. You'll have thought about this in quite a lot of detail during the planning stage, so please provide your editor with your insight and information.
3. **To get the most out of this book:** This section describes the software and downloads that will be necessary. For example: "You will need to install WebStorm version 10 to effectively run the code samples."
4. **You will see some icons:** "Warning" and "Tips" – which you can use throughout the book. These are "pull-out" sections that provide the reader with important information.
5. **Get in touch:** This section provides Packt contact details so that the reader can connect with us.

Introduction

The introduction covers a lot of the information that you'll have already worked out during the **Planning** phase.

1. **What is the problem that the book exists to solve?** Encourage the reader - with this book they will solve the problem!
2. **Introduction to the topic:** Familiarize the audience with the big picture and major concepts that you will explore. Describe the rationale behind your topic inclusion and the sequence that they are presented in. This should spell out:

   a) Context: general concepts of the topic.

   b) The area that you are going to be focusing on. This might include an explanation of terminology, for example.
3. **What the book covers:** a chapter by chapter description of the book.
4. **Prerequisites:**

   a) Knowledge: Inform the reader of the things that they need to know before they start. Spell out what knowledge you assume your reader has.

   b) Any additional installation and information they need for getting set up.
5. **Summary:** The learning outcome: Tell the reader what they will know at the end of the book.

Chapter Structure

We'll take a look at the chapter structure in more detail in the next section, as this can vary depending on the type of book, but there are some common elements. The opening page of a chapter should start on a right-hand folio and it will contain:

1. The **chapter number**
2. The **chapter header**
3. An **introduction** (50-100 words) explaining the content of the chapter
4. A **bulleted list of learning outcomes** (which should include page numbers for ease of reference)
5. **Technical Requirements:** A paragraph listing the exact technical prerequisites for completing the chapter. **This is where you add the Git Repo details for the chapter.**

Summary

At the end of each chapter there should be a summary – remind the reader what they have just learned. Aim for a maximum of around 200–300 words.

In the **The Author's Toolbox** section, we will look in more detail at layout devices that we can use in order to present information.

Length

As a rough guide:

- 30 pages per chapter
- 300 words per page unless there are code samples or illustrations (in which case it will be fewer)

The Last Chapter

Tell the reader what they now know and congratulate them on completing your book!

In the first chapter, you informed the reader of the problems they would solve, now show them how they solved those problems. Suggest the next steps that they might wish to take to move to the next level. Don't forget to discuss this with your editor – we have an extensive catalogue of helpful books that your readers might enjoy. These are added to the **Back Matter...**

Back Matter

A boilerplate exists for the Back Matter, which contains a section entitled **Further Thoughts**. This includes:

1. Examples of other Packt books that the reader might enjoy
2. An invitation to the reader to leave a review

05 Review

Review

Before completing your first draft, go through the checklists below to ensure you've captured everything that you need to.

Once you've submitted your first draft, you will get an opportunity to review it again. Your draft will be sent to a Technical Reviewer who will add comments, and then send them back to your editor. Your editor will then share those comments with you and give you an opportunity to respond to them and update your first draft.

Plan

Question	Explanation	
Who is your audience?	1. A general description of your audience	
	2. What knowledge can be assumed?	
	3. Key characteristics/unique challenges	
What is important to them?	1. What are the key problems/issues your audience is dealing with?	
	2. What are the need-to-know features?	

TOP TIPS

for writing a Packt book

1. Ensure that your heading titles are clearly indicative of what the reader will do/achieve in that section. The best way to do this is to use a gerund verb such as 'creating' or 'implementing'. This way, the reader knows exactly what to expect.

2. Always try to write in the active voice. You can read more about this here: https://learnenglish.britishcouncil.org/english-grammar-reference/active-and-passive-voice

3. Avoid referring to colours in images. Packt books are printed in grayscale, so hanging your point on a reference to the colour of a key, for example, will not translate to the reader.

4. Always remember to explain the purpose of tasks and reinforce the value of your point to your reader. For example, 'I'm writing an email to you. This email consists of my top tips for writing a Packt book. This will allow you to write your chapters much easier as you will have an insight into what makes a great book.' Here, I have related the information back to why this is useful to you. Keep this in mind for your readers, too. In the above example, notice I referred to you as 'you' rather than 'the author'? Using a direct address allows you to feel much more personally involved in what I'm saying to you. In your chapters, directly address your reader instead of calling them 'the reader'.

5. Good introductions state what we are going to explore, what lessons/skills the reader will gain, what topics we will cover, and why this is useful to the reader's real-world experiences and progress in the book itself.

6. Good summaries reflect on what we've covered, what we learned/what skills we gained, how those skills and lessons will become useful, what we will cover in the next chapter, and how this topic will help us moving forward.

7. Maintain logical flow in your chapters by creating links between sections. Ask yourself: why does this section come next?

8. When including images, code, tables, and links, ensure that you use a lead-in sentence which explains what we are looking at.

9. After including images, code, and tables, a follow-up sentence is necessary. This should explain what the figure showed us and elaborate on its purpose before moving on.

10. Instructional content should appear in numbered lists wherever possible.

Packt

Want to get in touch with Packt?

Want to get in touch and let us know your thoughts, feedback or anything else which springs to mind? We'd love to hear from you – drop us a line at authorsupport@packtpub.com

Useful Contact Details

- Author Support email: **authorsupport@packt.com**
- Packt Author Twitter handle: @packtauthors
- Packt Live email: **packtlive@packt.com**
- Packt Experts Facebook Author Community

Packt›