

# Packt Style Guide

If we were to distill the contents of this style guide into one sentence, we would quote the 19th-century French writer Stendhal: "I see but one rule: to be clear". The more practical concerns, however, of how best to write code in a bullet point or the intricacies of heading punctuation require a more detailed and comprehensive approach.

A couple of useful reference books include Strunk and White's The Elements of Style and R. L. Trask's The Penguin Guide to Punctuation.

Furthermore, we base our style guide on The Chicago Manual of Style (CMOS), so for more obscure questions and issues, refer to the latest edition of CMOS.

## General Text Style

We'll begin with a handful of common concerns that both authors and editors raise. These rules apply to all text in our books, unless a specific rule overrides them.

- Use American English. For example, write *color* instead of *colour*, *among* instead of *amongst*, *while* instead of *whilst*.
- Use the serial comma in lists with more than two elements: say "dog, cat, and mat" rather than "dog, cat and mat."
- Avoid spaces before and after a slash (/), for example, *and/or*. However, there is an exception to this rule when the slash is meant to separate compounds instead of single words, for example, *World War I / World War II*. In these cases, consider replacing the slash with a suitable word such as "and" or "or."
- Replace ampersands (&) with the word "and."
- When referring to the book, call it "this book" or "the book", but never "our book" or "my book". Do not refer to the book as "the text" or "the work."
- Convert obscure symbols such as → and ∑ into icons.
- Other symbols – such as ®, ™, ©, %, and ° – are acceptable.
- Always write temperature as °C and °F, rather than "degrees Celsius" and "degrees Fahrenheit".
- Superscript and subscript are acceptable.

While the above rules apply to all Packt content, be sure to refer to specific [Series Guidelines](#) for more information and context.

## Chapter Structure

In general, a chapter should not exceed 40 pages (plus/minus 15% deviation is usually allowed); keep the Outline as a reference to determine the pre-decided page count per chapter. The following rules govern the intervening content.

### Title

The title should be brief but should contain enough information for readers to be able to glance at it and know what they'll be working on.

Capitalize the title of the chapter (use title case). Title case involves capitalization of all words other than articles, prepositions, and conjunctions (except when they appear at the beginning). This extends to hyphenated elements. For example, write "Bed-and-Breakfast", and not "Bed-and-breakfast." Similarly, where there are only two elements, both are capitalized. For example, use "User-Facing Framework" rather than "User-facing Framework."

Contrary to title case, sentence case involves capitalization of only the first word in the sentence – and, of course, any words that take a capital. Indeed, most text is written in sentence case.

Have a look at the following examples:

#### Title Case

- > Defining the Data Model
- > Server-Side Techniques with PHP and MySQL

#### Sentence Case

- > Defining the data model
- > Server-side techniques with PHP and MySQL

Chapter titles should also be SEO-compliant and should, ideally, include certain keywords that will increase the likelihood of independent chapters appearing high in search results.

### Introduction

Chapters begin with an Introduction. There should never be a heading before the Introduction, so the first line of the Introduction is an ordinary body text paragraph. On rare occasions, a chapter may begin with a quote. This is how a chapter typically begins:

- 1 Chapter title
2. Quote/body text (Introduction)
3. Body text
4. Bullet list (list of topics covered in the chapter), which should coincide with Level 1 Headings
5. **H1 - Section** (the first section of the chapter)

In general, the Introduction should not take up more than the first page of the chapter. The overall point of an Introduction is to make the reader understand why they should read on. A good Introduction:

- Is brief
- Relates the new topic to things the reader already knows
- Introduces the new topic as a solution to a problem that the reader can understand
- Makes the reader understand why the topic is important and help them decide whether they should read on
- Should avoid trying to define or explain the topic. It is very hard to explain anything that the reader hasn't seen
- Rarely needs more than three short paragraphs in addition to a list of the topics that the reader will encounter in the book

## Technical Requirements

The Introduction and bullet points should always be followed by a Technical Requirements section. This will detail the software required to understand the contents of the chapter, as well as a link to GitHub. This should always be written independent of other chapters, and it should never be assumed that the reader will be aware of the requirements just because they have been covered earlier in the book.

## Summary

Every chapter should have a Summary (except *Cookbooks*), which is the last section in the chapter (titled Summary and formatted as **H1 - Section**).

The Summary should recap the main points learned or covered in the chapter, and if suitable, advise the reader what they're going to do/read in the next chapter.

The Summary describes what the reader should take away from the chapter, so make this as positive as you can. Do not let the reader leave the chapter with a negative view of what they have just read. And avoid bullet points in the Summary – we want to avoid a boring recap of content.

## Chapter Elements

A number of different elements make up a Packt chapter:

- Headings
- Paragraphs
- Words that form the paragraphs
- Lists
- Code
- Tables
- Images

Anyone who works with content needs to map the author's work into our chapter elements. The author might at times supply material that does not fit our chapter elements, so we will need to adjust the material according to the standards we follow.

We'll look at each of these elements in turn now.

## Headings

Headings should convey as much information as possible about the text that follows to help readers locate information quickly. But we also, of course, use headings to segregate chapters into logical sections.

Follow these rules when editing headings:

- Headings take sentence case
- Prefer the gerund form over the infinitive: *Working* with PowerShell; *Using* Swift; *Running* the code; and so on
- Authors may phrase headings as questions, but ensure that the heading is clear and conveys enough information to the reader
- Do not change series-specific headings: a Cookbook's "How to do it" heading must remain unedited
- Avoid the following punctuation marks in headings: colons, semicolons, em dashes, and periods
- Include only the following punctuation marks in headings: commas, question marks, exclamation marks, and en dashes

## Text After a Heading

Do not follow a heading with anything but text in the paragraph style. This means that every heading should have text between it and the next heading and that code, vertical lists, images, and tables never directly follow a heading.

The How to do it... and How it works... sections in a Cookbook are exceptions to the above rules. Refer to the [Cookbook Series Guidelines](#) for more information.

Where possible, authors should avoid inserting "filler" text to adhere to these rules, as content should always be useful. However, any lead-in is better than no lead-in, so in cases where there is no lead-ins or interesting text between a heading and any of the elements mentioned above, the Content Development Editor (CDE) may need to insert a lead-in.

Ensure that sections flow into each other in a smooth way. When consecutive sections talk about seemingly unrelated topics, something could be missing.

## Heading Levels

There are four main levels of headings used in Packt chapters. The top-level heading, **H1 - Section**, is used for the main sections of the chapter (including the first and last headings in the chapter), and **H2 - Heading** through to **H4 - Subheading** are used for subsections.

The book's Table of Contents (TOC) is created from the headings present in the chapters of the book, and only includes headings up to **H2 - Heading**. The contents of each section of the book should be clear to the reader simply from reading the TOC.

**H1 - Section** is the "top level" heading and denotes the main sections of a chapter. A chapter must contain more than one "top level" heading (the Summary, and at least one main section). You will need to monitor heading levels carefully to ensure that **H1 - Section** is the highest level of heading used.

- The first heading of a chapter should be a "top level" heading, **H1 - Section**. This could mean, for example, that you end up with a chapter where all top-level headings are formatted in **H2 - Heading**, which is unacceptable.
- Do not allow the entire content of the chapter to sit inside one top-level heading. If a chapter has only one top-level heading (excluding the Summary), this suggests that only one topic is covered in the chapter. This again is unacceptable.
- Heading levels should increase in order. **H2 - Heading** should follow **H1 - Section**, and **H3 - Subheading** should follow **H2 - Heading**. The difference between the levels of two consecutive headings should not be more than 1. However, you can jump back any number of heading levels, so that **H1 - Section** can follow **H4 - Subheading**, for example.

There are 4 levels of heading that can be used in all Packt books – bear in mind that, by default, **H3 - Subheading** and **H4 - Subheading** do not feature in the TOC. If you find you need more than 4 levels of heading, you should carefully look at the heading structure again.

## Paragraph Text

Style paragraph text in one of the three ways:

- Body Text
- Important Notes
- Tips
- Bullets

### Body Text

An ordinary paragraph is called Body Text. Body text is any text which is not a heading, part of a list, table, code snippet, Important Note, or Tip. The default style for body text is always **P - Regular**.

### Important Notes

Some information may be so important to the reader that it should be highlighted to make it stand out from the ordinary text. See below:

## Important Note

An Important Note should provide readers with additional good-to-know information. For example, you can tell a reader to double check all dependencies before running a particular instruction via an Important Note.

The heading which is "Important Note" should have the **P - Callout Heading** style applied to it and the paragraph after it should have the **P - Callout** style applied.

If a single sentence in a paragraph seems particularly important, the paragraph should be edited so that the sentence forms a paragraph on its own. Applying the **Important Note** style to a single sentence will apply the style to the whole paragraph.

Apply the following rules when writing and editing Important Notes:

- Important Notes should ideally consist of a single paragraph, but if they consist of multiple paragraphs, ensure they are formatted as paragraphs
- Avoid using consecutive Important Notes
- If there is a lot of important information, it is worth considering whether there should be a heading here and a small section instead of the Important Note
- Do not allow Important Notes to take up more than half a page

## Tips

In the main text, this is how a Tip should appear:

## Tip

A tip is just that: useful advice, a hint, or a note for the reader. For example, you can advise the reader to install hardware of a higher capability than the basic requirements for the technology in order to make data processing faster.

## Quotes

A quote should be styled in **P - Italics** with quote marks around it, if it's within a paragraph:

*In this chapter, we will learn about the principles of...but before that, let us read this quote:"JavaScript (JS) is an interpreted computer programming language".*

If the quote is appearing on its own, use the **P - Quote** style without quote marks:

*JavaScript (JS) is an interpreted computer programming language.*

There are two instances where we can apply these styles. When we are in fact quoting someone, the name of the person who is quoted will be written as follows:

*En-dash / single hyphen<space><Name in P - Italics>*

The quote should be presented as follows:

*"Love all, trust a few, do wrong to none."*

*– William Shakespeare*

Where phrases are from the Internet, present them as below:

In a paper published by the IEEE in 2009, *The Unreasonable Effectiveness of Data* states that:

*"But invariably, simple models and a lot of data trump over more elaborate models based on less data."*

Whenever you come across an instance of a quote in your books, always verify the quote and then only apply the style to it. The quote should be exactly as said by the person or mentioned on the particular website.

Finally, do not edit a quote – even if it is grammatically incorrect or includes spelling mistakes.

## Words in Paragraph

How we style words in paragraphs is, obviously, important. However, there are some particular types of words that are found in paragraphs that we'll go into details below:

- Important/notable words (keywords)

- Code terms and URLs
- Terms that appear on the screen
- Keyboard keys
- E-mail addresses
- And more...

Highlighting Important Words

Depending upon the context, words can be highlighted using the **P - Italics** or **P - Keyword** style.

P - Italics

Apply **P - Italics** in the following cases:

- To reference a section, another book, a magazine, or a newspaper.
- For intense emphasis, usually to a single word or a few words.
- When referring to infinite values. For example, "1 to *n* values". It's also used when referring to a value from an equation. For example, "In the previous equation, the values *x* and *y*."
- Lesser-known foreign words, film names, and app names should be italicized.
- In the case of simple formulae, include this as part of the sentence and apply the **P - Italics** style to it. For example, "In Excel, it would look something like *(B2\*4+C2+D2)/6*." However, if the formula is complex, use MS Word's Equation option to insert the formula and center align it. For example:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

P - Keyword

The first appearance of an important term (with respect to the technology the book is about) in a piece of text should be highlighted using the **P - Keyword** style. This is a character style and is applied only to the currently selected word.

Once a term has been marked with the **P - Keyword** style, there is no need to mark it again. Words should not be marked more than once with the **P - Keyword** style unless the term has not been used for a long time since its introduction (such as being used in one of the later chapters in the book but introduced in one of the earlier chapters).

Marking words with the **P - Keyword** style is a good way to indicate the importance of that term not only to the reader but also to the indexer, who will often use these words during the creation of the book's index. Check out the below two examples for more context:

A **module** is an element of page content used throughout the system. You can choose your modules from the administration interface.

A set with a binary operation that satisfies these axioms is known as a **group**.

The following rules will give you a good idea of when and when not to apply the appropriate style. Apply:

- To the first occurrence of a word in the chapter or book. This includes terms along with their abbreviations. For example, **Microsoft Small Business Server 2011 Essentials (SBS2011E)**.
- To important/notable words that are relevant to the technology.
- When providing definitions or explaining terms in bullet lists (known as a term list – see later), the terms need to be in the **P - Keyword** style.
- To highlight terms that appear in a diagram or a figure (not in a screenshot) or to refer to items labeled by the author in a figure or screenshot.

Finally, the full term should always feature before the abbreviation. If the term is particularly prevalent and obvious to the reader, rather than reversing the keyword order, the abbreviation then the full term in parentheses, you do not need to include the full term. If the author insists on including the full term, reword the sentence to make it obvious that we have not got our own style incorrect – for example, **HTML** (short for **Hypertext Markup Language**).

Code Snippets and Command-line Code

Our code must look clear, clean, and simple. There are two styles for formatting code:

- **SC - Source** for general snippets of code

- **SC - Highlight** for highlighting individual lines of code to which the author would like the reader to pay specific attention. Note that this style is only supposed to be used within **SC - Source** blocks.

Have a look at the below example:

```
function detectAudioMP3Support() {  
    const audio = document.createElement('audio');  
    const canPlayMP3 = audio.canPlayType &&  
        audio.canPlayType('audio/mpeg; codecs="mp3"')  
    return canPlayMP3 === 'probably';  
}
```

You can find the exact instructions for the layout and spacing of code in the Code Formatting Guidelines. There are some general rules, but the specifics are different for each language or technology being used.

When editing code, keep in mind the following two important points:

- Format code with caution: adding extra spaces or introducing line breaks can break the code for certain technologies.
- We do not currently have a "code continues" character to go at the end of lines we have broken artificially. If you break a line in the chapter's code, the reader will also be expected to press the Enter key to start a new line, so this line break will have to be correct in the syntax of the code the user is writing.

Often, the author will want the reader to enter a particular command from a command-line prompt on their computer. This command is styled using the **P - Source** style, sometimes with a command prompt character (>) added to the start of the line. An example of Command line code is as follows:

```
>discover secret.py
```

Some important considerations for formatting code are as follows:

- A snippet of code should always be introduced with a lead-in sentence ending with a colon. This lead-in sentence should tell the reader the purpose of the code they are about to see.
- Do not allow certain characters (such as ...) to be used to indicate that code is missing. The author should explain, as carefully as they can, exactly where the code should go and should not resort to introducing symbols into the code that may not be completely understood by the reader.
- The code should not simply appear on the page for the reader to look at; there should always be some explanation of what the code does.
- Code comments are never a replacement for an explanation in the text.
- Try to keep each block of code to a maximum of 1 page. If you need to share a larger chunk of code, break it up with paragraphs of explanation at key points.
- Always double-check code comments and code for spelling errors.

### **SC - Heading and SC - Link**

Use **SC - Heading** in your code block to indicate the name of the corresponding code file that you want a reader to look at on GitHub. Use this style to add the name of the corresponding code file on GitHub whenever you truncate a piece of code that goes over a page in length. The name of the file needs to appear only above the code block.

Use **SC - Link** to add URLs at the end of or lead-in to the code block to lead users to the exact code file on GitHub. Use this style to add the link to the corresponding code file on GitHub whenever you truncate a piece of code that goes over a page in length. This link needs to be added only after the code block.

Example:



```

model.summary()

# train
#optim = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=OPTIM,
              metrics=['accuracy'])

history = model.fit(X_train, Y_train, batch_size=BATCH_SIZE,
                  epochs=NB_EPOCH, validation_split=VALIDATION_SPLIT,
                  verbose=VERBOSE)

print('Testing...')
score = model.evaluate(X_test, Y_test,
                      batch_size=BATCH_SIZE, verbose=VERBOSE)
print("\nTest score:", score[0])
print('Test accuracy:', score[1])

```

[https://github.com/PacktPublishing/Deep-Learning-with-Keras/blob/master/Chapter03/keras\\_CIFAR10\\_simple.py](https://github.com/PacktPublishing/Deep-Learning-with-Keras/blob/master/Chapter03/keras_CIFAR10_simple.py)

This is just an optional style and can be used if the author wants to.

## Code Terms and URLs

A code term is an expression that would appear in a computer program. This means anything such as the name of a variable, class, method, function, property, and so on.

The following is the complete list of **P - Code** instances:

- User input
- Twitter handles
- Code terms used in text
- Dummy URLs
- File names and names of folders, even if they appear in a screenshot
- Database table and field names, even if they are visible in the screenshot
- File extensions
- Path names

Code terms that appear in text (any form of text such as body text, Important Notes, lists, or tables) should be formatted in **P - Code**.

Check out the following example:

The **destroy** function removes all the data from your hard drive.

If you look at the definition of the **Dog** class, you will see the **Woof()** method. This method returns a **Noise** object

Keep in mind the following rules when writing and editing code:

- Where a code term has an article present – “**the** PageController”, for example – add the object type, (on this occasion, class), if there is one.
- If there is not an object type present or where a code term’s object type is part of the term – for example, **the** MainPage – delete the article. Similarly, where a code term has an object type but no article, add the article or delete the object type.
- Never enclose code terms in quotes, such as the “PageController” class. Remove the quotes and then apply the appropriate style – for example, “**the** PageController **class**”.
- Where a lowercase code term is at the beginning of sentence or paragraph, add an article and object type (or rephrase the paragraph) to avoid ambiguity as some code text terms include periods. This is not necessary for uppercase code terms or **P - Bold** terms (terms that appear on the UI/screen).

- Where a sentence is split over a block or line of code text (so there is half of the sentence before the code and the second half after the code), rephrase both fragments to create full sentences. A new line should start with a new sentence.

In addition to the above points, keep in mind the following when dealing with the styles themselves:

- Terms such as class names, object names, and command names that appear in a screenshot should be formatted in **P - Code**.
- Command lines that appear in a screenshot of a command-line window should be formatted in **P - Code**.
- Lines of code that appear in a screenshot should be formatted in **P - Code**.

If a URL is already present in the screenshot and the user is asked to click on it, it should be formatted in **P - URL**. However, if the user is asked to enter the URL, it should be formatted in **P - Code**.

URLs

URLs take the **P - URL** style. This style has the same definition as **P - Code** but is used to provide an extra layer of metadata. This means an online version of the text will have all the URLs as actual links that the reader can follow. Style all dummy URLs with **P - Code**.

Keep in mind the following rules when you encounter URLs in a text:

- Where a URL is the last item in a paragraph, it should be part of a complete sentence, with no paragraph break, and be finished with a full stop/period (otherwise the paragraph will look incomplete).
- If an author does not want a period to be included after the URL, finish the paragraph with a period, insert a paragraph break, and put the entire lead-in sentence and URL on a new line.
- Where the sentence is a standalone sentence (not at the end of a paragraph) or is part of a bulleted list, you may end the sentence with a colon, insert a paragraph break, then the URL, without a period.

Check out the below table for a couple of handy examples:

CORRECT	INCORRECT
<p>To improve the loading page's performance, we recommended that you use the minified version of the script.</p> <p>For more information, refer to: <a href="http://www.angularjs.com">http://www.angularjs.com</a></p> <p>To improve the loading page's performance, we recommended that you use the minified version of the script. For more information, refer to <a href="http://www.angularjs.com">http://www.angularjs.com</a>.</p>	<p>To improve the loading page's performance, we recommend that you use the minified version of the script. For more information, refer to: <a href="http://www.angularjs.com">http://www.angularjs.com</a></p> <p>To improve the loading page's performance, we recommended that you use the minified version of the script. For more information, refer to: <a href="http://www.angularjs.com">http://www.angularjs.com</a>.</p>

Filenames

Always style filenames and names of folders with the **P - Code** style – even if they appear in a screenshot.

The word “folder” should be used instead of directory when the author mentions files and their locations. Ensure to maintain consistency throughout the book. See the following example:

In the `C:\public\html\homepage.aspx` path, we have the following elements:

- `homepage.aspx` is the **filename**.
- `html` is the **folder** that contains the file.
- `c:\public\html` is the **full path** of the file.
- `.aspx` is the **extension**.
- **ASPX** is the type of file.

Note that when the extension of a file is written on its own, the extension should be in lowercase – even if the filename itself isn’t – and have the dot at the start of it. The extension should be in the **P - Code** style.

The type of file is always uppercase and does not include the dot at the start. Refer to the Packt Keyword list for the spelling of common file types. For example, a ZIP file has a .zip extension.

In Windows, folders are separated by the \ character, but on Linux and Mac, the separator character is /. Ensure that the author remains consistent with the use of the slash.

Using Plurals with P - Code

Often, you will encounter code terms that refer to more than one object. For example:



After dragging each *Button* control onto the form, you can look at the generated code to see the collection of *Buttons* that has been added to the form's control set.

The word *Buttons* is the problem. Because *Buttons* is set in the Code style, it should refer to an actual code snippet, but it doesn't: what we have is a collection of *Button* "things". In the above example, the word *Buttons* will refer to "*Button* objects" or "*Button* instances" – prefer such usage instead.

Put simply, never add an "s" to a code term in order to pluralize it. Find another way to do so...

This means that just as *Buttons* is unacceptable, neither is removing the formatting from the s to make *Buttons*. This looks even worse.

This problem becomes even more obvious when the plural requires "es". For example:

The constructor of the *Box* class takes 2 parameters for its height and width. Once we have supplied the height and width for each of the *Boxes* we want to create, we can draw them.

In this case, *Boxes* should be something like *Box* objects or *Box* instances. By putting *Boxes* in **P - Code**, we are suggesting that there is a code object called *Boxes*, but there probably isn't.

Do not apply this rule without thinking. Often there are code terms that are pluralized:

The table has a *Fields* property that contains a collection of *Field* objects.

The *Fields* property is correct; *Fields* is indeed the name of the property.

This next example illustrates how useful the above rules can be. This sentence could be incorrectly written as:

The table has a *Fields* property that contains a collection of *Fields*.

It is fair to say that this sentence has no useful information in its current form (and is wrong), and requires us to apply the rules mentioned earlier.

### Using Apostrophes with P - Code

Another thing you will see often is attempting to use an apostrophe with a **P - Code** term. Here are two possible ways of formatting the same sentence about the *destroy()* function:

1. *destroy()*'s parameters need to be ordered correctly or an error will result.
2. *destroy()*'s parameters need to be ordered correctly or an error will result.

The first one is obviously not right, because there is no code term called *destroy()*'s. The second one still looks rather clunky but correcting it is simple and removes any trace of an apostrophe. For example:

The parameters of the *destroy()* function need to be ordered correctly or an error will result.

Somewhat clunky, yes, but at least it's clear. In general, it is best to avoid using apostrophes with **P - Code** terms.

## Words that Appear on the Screen (P - Bold)

Technically, any code term will appear on the screen, unless the computer it runs on only generates paper output. However, here we are talking about words or expressions that form a part of the output (or input) of an application.

The following is a list of words that may appear on the screen:

- Words that appear in the user interface of an application
- Dialog boxes
- Labels
- Buttons
- Links for form elements

You can identify these terms based on the terms that are generally used with them. For example, "click on", "open", "go to", "hit", "on the x page", or similar terms that imply that they appear on the screen. To format these words, use the **P - Bold** style.

Never enclose words that appear on the screen in quotes – for example, click on the "Submit" button. Apply **P - Bold** instead.

Ideally, the words in **P - Bold** will be accompanied by a screenshot nearby to help the reader. If there is no such screenshot, the words that appear on the screen should still be marked with **P - Bold**.

Be sure to use **P - Bold** only for words that appear on the screen. Do not use it to apply special emphasis to words.

Where a **P - Bold** term has an article present – for example, "The **Name**" – the object type (on this occasion, field) should be added if there is one. If there is no object type, or where a **P - Bold** term's object is part of the term – for example, the **Main Page** – the article should be deleted. Note that this does not include proper nouns.

Where a **P - Bold** term is referring to a field on the screen that includes a colon – for example, the **Name:** field – the colon should be removed from the **P - Bold** term. For example, In the **Name** field, type your name. However, do not delete ellipses that appear on the screen.

User Input

The value that a user is asked to enter in the UI – a value to enter in a text field, for example – should be formatted in the **P - Code** style. This is because the value the user will enter is not already present. Readers should be able to distinguish between text already present on screen and text they are supposed to enter. So, any text that is present on the UI should be formatted as **P - Bold** and the text to be entered by the users should always be in **Code**. Check out the following table for examples of correct and incorrect usage:

CORRECT	INCORRECT
Click on the <b>Submit</b> button.	Click on the " <b>Submit</b> " button
Enter your name in the <b>Name</b> field.	Enter your <b>name</b> into the <b>Name</b> field.
Enter 203843 in the <b>Credit Card</b> number field.	Enter <b>203843</b> in the <b>Credit Card</b> number field.

Once P - Bold, Always P - Bold

Once you have formatted a term in **P - Bold**, if you use the same term again, be sure to use **P - Bold**. For example:

You enter the title of your article into the **Title** textbox. After you enter text into the **Title** textbox, you will see that some text has automatically appeared in the **Friendly URL** field.

But be aware of this exception: if the word should instead appear in a different style – such as *code* – use the new style as appropriate. Indeed, if we wanted to refer to these fields in the next chapter, we would still use the **P - Bold** style.

However, this does not mean that you keep applying **P - Bold** to any instance of the word. You would not write...

As we added the **title** to the **Title** textbox, this is now displayed as the page **title** in your web browser.

The term "Title" was marked as **P - Bold** as it refers to the textbox found on the screen, and whenever we refer to that term, we use the correct style. We do not apply the style every time we see the word "title" in our text.

Keyboard Keys

When you are instructing the reader to press certain keys on their keyboard, use the **P - Italics** style. If the reader must press more than one key simultaneously, the keys are separated with the + character, with a space before and after the + character. In this case, the + character must be set in the **Paragraph** style.

For keys that are words (such as Tab, Delete, Ctrl, or Enter), the word should be written the way it is present on the keyboard. Have a look at the following examples for more details:

- To stop your application, press *Ctrl + Alt + Delete* to bring up the Task Manager.
- If you press the *Q* key, your session will be terminated.

For keys without a written term on them, just write out the name of the key itself.

The Space bar (do not apply the **P - Italics** style as this is not the name of a key)

Finally, write all Apple-specific keys in the **P - Italics** style.

E-mails and Twitter

Both emails and Twitter handles take the **P - Code** style:

- For example, if the name is not filled out in the HTML form and the e-mail address is **contact@packtpub.com**, we are going to populate the **Name** property with **contact**.
- Our author tweets regularly: follow **@johnwritescode** for daily techie tidbits.

Keep in mind that Twitter is a proper noun and we should capitalize as such.

Lists

A list is a collection of items. There are two forms of lists used in Packt chapters:

- Bulleted lists
- Numbered lists

The standard formatting features of Word should not be used for formatting lists. Only Packt styles should be applied to format lists. Any of the standard formatting features for lists should be removed before re-applying the Packt styles.

## Bulleted Lists

Use a bulleted list for an unordered series of concepts, items, or options, rather than a sequence of events or steps. Items in a bulleted list are typically single sentences, put into a list form for visual impact.

A bulleted list always requires a lead-in from a paragraph, with the lead-in sentence finishing with a colon. The lead-in should always make sense and should not be a fragment.

The items in a bulleted list should be formatted in the **L - Bullets** style.

Follow these rules for bulleted lists:

- Capitalize the first word of each bulleted item unless the word is a code term or any other term that naturally appears in lowercase (such as jQuery).
- Commas, other punctuation marks, or conjunctions such as "and" or "or" should not be used at the end of a bullet point.
- Any term list should contain either only complete sentences or sentence fragments (such as single words/phrases), not a mix of both. For example, if a list contains two sentences and two fragments, the fragments should be converted into sentences.
- A full-stop should only be used at the end of a bullet item if the item contains >1 sentence, even if the bullet item forms a complete sentence.
- Golden rule – if there are too many variations, simply maintain **consistency**.

### Using bullets to indicate section ordering

Bullet lists are often used in chapters to indicate the sections/topics that are about to be covered. When this is done, the order of the items in the bullet list should match the order of the sections in the chapter, and it should be clear to the reader which section of the chapter each item corresponds to.

### Referring to items in a bullet list

Items in a bullet list will often include, for example, function names, activities, options, and properties. You should not refer to an item in a bullet list as a bullet, but refer to it as an item if you cannot find any other way of referring to it (e.g., the function, activity, or option).

If you want to refer to a particular bullet item, refer to it by its position in the list.

This means you talk about the first item (first function, first activity, and so on), second item, third item, and so on, with the last item in the list being referred to as the last item. Working backwards from the end of the list, there is the last item, second to last item, third to last item, and so on.

Do not refer to items in a bullet list by saying "item number 2" or "bullet point number 2", as the bullet list has no numbers associated with it. Example:

CORRECT	INCORRECT
<p>There are two methods in this class:</p> <ul style="list-style-type: none"><li>• toString()</li><li>• toXml()</li></ul> <p>The first method returns a string representation of the object.</p> <p>Once you have set up the system, you can either:</p> <ul style="list-style-type: none"><li>• Start using it</li><li>• Fall asleep</li></ul> <p>The second activity is often difficult to avoid following the lengthy installation process.</p>	<p>There are two methods in this class:</p> <ul style="list-style-type: none"><li>• toString()</li><li>• toXml()</li></ul> <p>Method 1 returns a string representation of the object.</p> <p>Once you have set up the system, you can either:</p> <ul style="list-style-type: none"><li>• Start using it</li><li>• Fall asleep</li></ul> <p>The second bullet is often difficult to avoid following the lengthy installation process.</p>

### Aligning text under bullet items

Occasionally, you may wish to break a bullet item, say to display an image, and then continue the bullet item. You can align the continued part of the item using the **L - Regular** style. Example:

- When you go to the specified URL, you will see a welcome window. It'll look something like this:



There will be a link that reads **set up the wiki**. If you were to follow this link, the universe would explode. We, therefore, advise against it.

Here the sentence that begins with "There is a link..." has the **L - Regular** style applied to it.

**Aligning code or command under bullet items**

If a block of code is present under the bullet, use the **L - Source** style. Example:

- If you're not using permalinks, your category feed code will be as follows:

```
<?php if (is_category()) { ?>
    <p><a href="<?php echo get_category_link($cat);?>
        &feed=rss">
        Subscribe to '<?php single_cat_title(); ?>' posts
    </a></p>
<?php } ?>
```

The process for command lines is to use a combination of **L - Source** + **SC - Highlight** and then indenting with spaces, as appropriate. Example:

- If you're not using permalinks, your command will be as follows:

```
Subscribe to '<?php single_cat_title(); ?>' posts
```

**Numbered Lists**

Use a numbered list for a set of steps, procedures, or other sequential lists. The items in a numbered list should be formatted in the **L - Numbers** style.

The following are the rules for numbered list items:

- Capitalize the first word of each numbered list item
- The numbers in the list should be consistent; they should increase by 1, and the first item in a new list should start at 1.

Note that a numbered list always requires a lead-in from a paragraph, with the lead-in sentence finishing with a colon. Do not just include a numbered list into the text without a lead-in. Also, a lead-in should not be a fragment.

Text can be aligned in numbered lists using the **L - Regular** style and the code can be aligned using **L - Source** style, as discussed above.

A numbered list of steps that extends beyond 20 steps (as a benchmark) should probably be broken down into smaller lists, with some text in between to join these lists together. A long list of steps may be difficult to follow for a reader. (If there are >20 steps but the list still flows and is easy to follow, it should be fine – you may take this decision subjectively, depending on the author's writing style.)

Tip: Try and keep each step to a single action. For example:

To exterminate the vampire, you need to:

1. Locate the vampire's coffin.
2. Visit the coffin by day.
3. Open the coffin.
4. Drive a wooden stake through the vampire's heart.

**Referring to items in a numbered list**

If the numbered list is a list of steps, then an item in the list should be referred to as a step. Otherwise, the default way of indicating an item in the numbered list is to use the word **item** if you cannot find any other way of describing what the item represents.

If you want to refer to a particular item in a numbered list, you can use either its position in the list or the number of the step.

This means you can talk about the first item, second item, third item, and so on or step 1, step 2, step 3, and so on.

When referring to one of the steps, you should always include the word "step" before the step number, and the word "step" should not have a capital S, unless it is at the start of a sentence. Example:



CORRECT	INCORRECT
<p>To exterminate the vampire, you need to:</p> <ol style="list-style-type: none"><li>1. Locate the vampire's coffin.</li><li>2. Visit the coffin by day.</li></ol> <p>Without step 1, you won't even have a vampire to exterminate.</p>	<p>To exterminate the vampire, you need to:</p> <ol style="list-style-type: none"><li>1. Locate the vampire's coffin.</li><li>2. Visit the coffin by day.</li></ol> <p>Without (1), you won't even have a vampire to exterminate.</p>

Vertical Lists

Regardless of kind – bullet or numbered – follow the below rules when composing vertical lists:

- Introduce a vertical list with a complete sentence followed by a colon.
- Start each list item with a capital letter unless the item begins with code or a proper noun styled in lowercase – jQuery or eBay, for example.
- Do not use a period unless an item forms one or more complete sentences (or as mentioned above, maintain **consistency**).

When referring to items in a bullet list, refer to an item unless you have a more appropriate word to hand: prefer function, activity, option, and so on. When referring to a specific item, refer to its position on the list. For example:

The third item in the above bullet list advises editors on the correct use of punctuation in a bullet list. The first, on the correct method of introduction.

In longer lists, consider referring to the 'second from last item' or the 'third from last item', and so on. Employ common sense when deciding on wording – but be sure, above all else, to be clear.

Term Lists

You will frequently come across lists of terms – their style doesn't deviate much from numbered and bulleted lists, but be aware of a handful of minor differences. A term list, then, is a list of terms, parameters, or similar items that are followed by a brief explanation or definition. Follow these rules:

- The term is set in **P - Keyword** and followed by a colon.
- While we should prefer full sentences to sentence fragments, term lists lend themselves nicely to fragments. Employ as appropriate.
- Capitalize the first letter of the definition unless – as mentioned above – it is a code fragment or a proper noun.

If the term is naturally set in either **P - Code** or **P - Code**, then this style is preserved, but the term is still followed by a colon. For example:

- **jQuery**: A cross-platform JavaScript library
- **Chicago**: An American city in the state of Illinois

As mentioned above, sentence fragments are appropriate in this context. Remember that you aren't writing full sentences, so you need not punctuate with a period. However, previously mentioned rules still apply: if one item takes a period, all must.

References

Authors will usually reference one of three sources: content in the same chapter, content in the same book, or content in a different book. Of course, an author could reference content from a variety of different sources, but these three are the main ones.

In general, avoid "blind" references, which provide no information about why you are referring to the material. The reader should know whether it is worth interrupting the current topic for the referenced material before following the reference.

With any kind of reference, the reference must always be correct:

- If it's a URL, the URL should exist and contain the specified content.
- If it's a reference to another part of the book, the reference should point to the correct chapter in the final book, not an earlier draft.
- If it's a reference to another book, the title, ISBN, and publisher should be correct.

Referencing Content from the Same Book

If the author is referencing some content from the same chapter, they will either reference something that they have covered or will cover. In either case, use words like "earlier" and "later" to indicate that we have covered the topic or will come across it later. Take care with the specific language: do not, for example, use the phrase "in a moment" to refer to content that readers will not encounter in a few paragraphs' time. Judge each case as it comes.

When it comes to referencing content from a different chapter in the same book, it's important to be clear where in the book the content can be found. As Packt chapters can be bought individually and read on Packt Library, readers need to be confident in knowing what you're referring to. Page numbers should always be avoided as they're not always consistent across different devices.

For a specific reference to another chapter in the same book, specify the exact chapter number and ensure that the word chapter is capitalized. The chapter number is followed by a comma and the chapter title in the reference. The style applied to the chapter number and chapter name should be **P - Italics**, and the comma should be in **Paragraph** style. If there is any possibility of the reference being difficult to find, include the title of the section in **P - Italics**. Every instance of a chapter reference should be followed by the respective chapter's name. When including the section title, be sure it is:

- Correct for the final version of the book, and not the title from an earlier draft of the chapter.
- Followed by the word "section". Do not use "section called XXXX", put the section title first then put the word "section".

See the following two examples:

- In *Chapter 3, Choosing and Installing Themes*, we listed the most useful template tags.
- In *Chapter 10, Using Swift*, we'll discuss the best uses of this feature.

There is little point in using expressions such as "on the previous page" as you cannot be certain that material will appear on the previous page after it has been laid out. Preferable expressions are 'as we just saw' or 'we're about to see'. The author should do this without prompt – only edit if his meaning is unclear.

Let's have a look at few more examples:

- You will see a list of the methods and properties of the **Dog()** class in *Chapter 5, The Alfresco APIs*.
- We saw how this worked in the *TheJavaScript API* section in *Chapter 2, Java Programming*.
- You will learn more about managing content with Muxo CMS in *Chapter 6, Playing around with CMS*.
- You will see more about managing content with Muxo CMS in a later chapter.

### Referencing Content from a Different Book

In order to reference another book entirely, simply state the title of the book, followed by the author and publisher (you only need to include the publisher once in the book). This should all be set in **P - Italics**, except the punctuation and spaces in between. This is even used for references to Packt books – write "Packt Publishing", not just "Packt".

Where a book's title is mentioned without the author's name, set it in **P - Italics**. However, where an author's name or person's name is present without being part of a full book reference, do not put the author's name in **P - Italics**.

See the following examples:

- For more information about ASP.NET Ajax, see *Sam's Teach Yourself ASP.NET Ajax in 24 Hours*, Joydip Kanjilal, *Sam's Publishing*.
- As stated by Joydip Kanjilal in *Sam's Teach Yourself ASP.NET Ajax in 24*, *Sams Publishing*, the reason we do this is...
- For more information, you might want to read *Sam's Teach Yourself ASP.NET Ajax in 24* or...
- As well as writing books, Joydip Kanjilal likes to...

### Tables

For clarity sake, always include a heading row and a heading for each column. Style code and command lines as normal; ditto vertical lists. Punctuation rules for vertical lists don't change in tables.

Indeed, a table should be simple – readers should be able to easily digest the information. If table content becomes excessive, it's worth reconsidering its use.

If you find that a single table extends more than one page, consider breaking up the table or presenting the content in some other way.

**Note that these tables will be converted into images later in the book. So it's best to avoid using tables, as their quality may not be excellent.**

### Images

We use three kinds of images: screenshots, diagrams, and photos. Be precise in your editing: use only the preceding three words. Indeed, 'image' will refer to a screenshot, a diagram, or a photo. Keep this in mind.

Use the **IMG - Caption** style on the Image and for its caption.

Note that every image in the chapter necessarily needs to have captions, except formulae and tables.

Furthermore, be aware of author's edits to images. Most of the time, Production will need to add the text to the image themselves in order to maintain the professional look and feel of an image. Some authors will have the required skills to edit an image themselves – apply common sense to decide on the quality of an author's edits. If in doubt, pass it on to Production.

The following points are useful to keep in mind when considering images:

- Unless the preceding text gives the reader sufficient context, include a lead-in sentence punctuated with a colon. If in doubt, include a lead-in.

- In case an image is followed by a bullet list, both the list and image must have separate lead-in sentences, rather than just one lead-in introducing both items. The lead-in should precede each individual item.

## Naming Conventions for Images and Captions

Use the following naming convention for image files: ProdID\_Figure ChapterNumber.ImageNumber. For example: B01234\_Figure 2.3. The product ID is B01234; the chapter number is 2; the image number is 3.

## Referring to Images

Be clear when referring to images. Use the precise word: screenshot, diagram, photograph, or caption. Do not use page numbers: they are unlikely to be correct in the final version and may change in subsequent editions of the book.

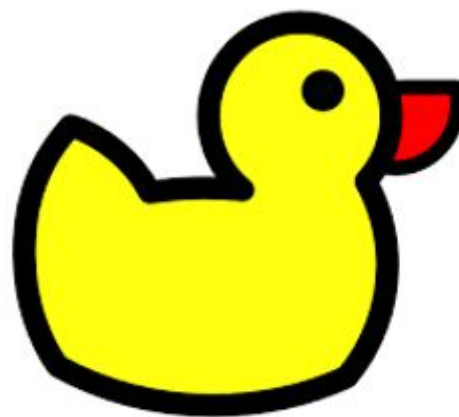
Above all, apply common sense. Be sure that you are being clear. Will readers know exactly what you are referring to?

Do not use 'above' or 'below'; rather, use 'following' or 'next' and 'previous' or 'preceding'. A good author will make this clear.

## Figure Captions

Format figure captions in the **IMG - Caption** style. Follow the below guidelines when writing captions:

- They should always start with the figure number followed by a hyphen and then the text. For example:



**Figure 1.1 – 2D Duck**

- Figure captions should be succinct and informative.
- They are there to explain what the image is, not give commentary or opinions.
- They should not provide information that was not mentioned in the body text.
- A figure caption should not be included where there is a lead-in present for the image or where it is clear what the image is representing as this may be repetitive for the reader.
- Begin a caption with a capital letter but do not conclude with a period.

## Avoiding Unnecessary Changes

Try to maintain the author's words and style as much or as closely as possible. Authors' styles can range from highly colloquial (almost like speech) to very formal (bordering on academic). We don't want to accidentally "edit out" the author's style.

The important thing is to ensure that the book is written in a clear, concise way that anybody can understand. Where you can do this and retain the author's style, you should. Where the meaning is unclear, you may need to make amendments that change the author's style. With this in mind, consider the following points:

- Leave contractions such as "isn't", "let's", and "aren't to" as they are. These need not be changed to "is not", "let us", and "are not".
- Do not change colloquialisms or idioms as long as readers will understand what the author means.
- Do not make unnecessary additions or deletions that would change the meaning and/or break the flow of the sentence.
- Change words where the original word was incorrectly used, but not where the word choice was correct: for example, change "like" where the meaning is "such as", but do not change "like" if the meaning is "similar to".
- Do not change words where the meaning of the replacement word is exactly the same as the original word: for example, do not change "similar to" to "like" as they mean the same thing.
- When you edit the text, try to make replacements that fit in with the author's style. For example, if the author's style is informal, do not make replacements that are formal, and vice versa.

SP - Editorial

This style should be used in two scenarios:

1. When you need to pass on any information inside the chapter to the Production Designer. Example:

Styles

- The magazine pages will be **portrait** and **facing**.
- Finally, it will be **10 pages** long.

When opening *Adobe InDesign 2020*, and as we covered in *Chapter 2, Getting Started with Adobe InDesign 2020*, we will find the Home screen, and from there we can create our first document. We actually have three ways to do that, as shown in the following screenshot:

- From the Home screen, click the **Create New** button, as shown in the left-hand image of the screenshot.
- From the **File** menu, click **New** and then select **Document...**, as shown in the right-hand image in the screenshot.
- You can also create a document by using the shortcut keys **Ctrl + N** (Windows) or **Cmd + N** (macOS):

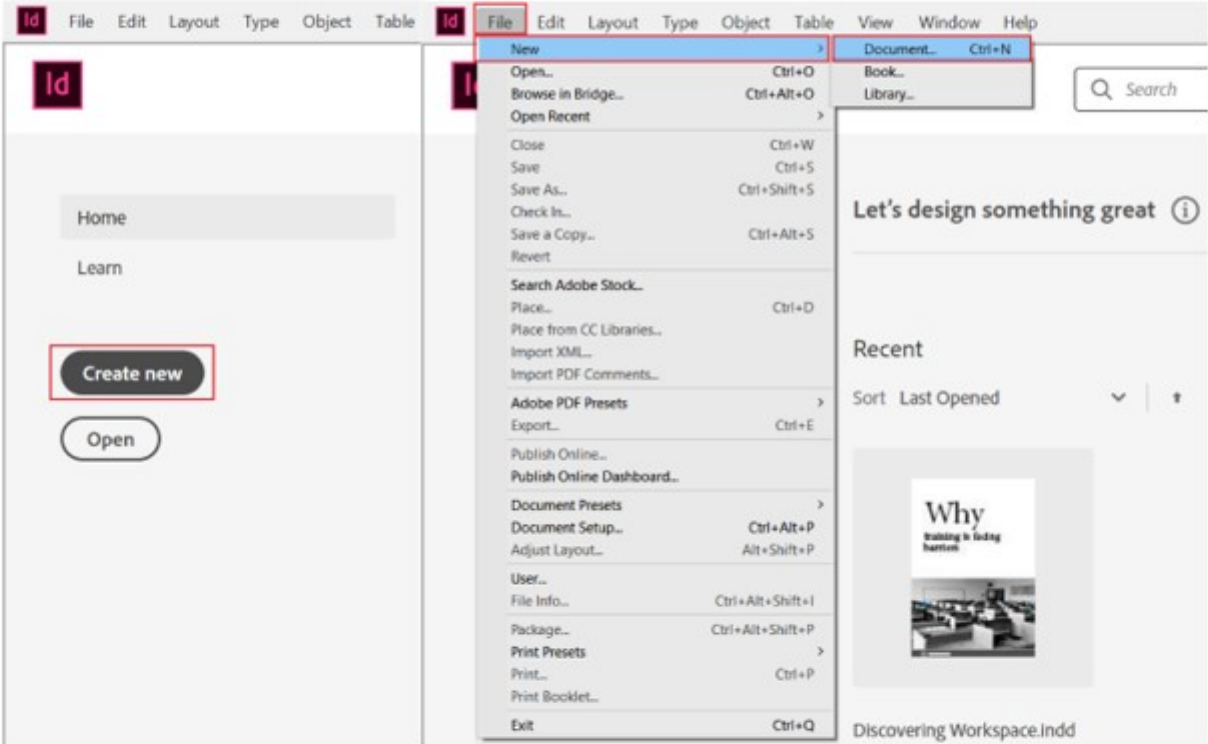


Figure 3.1: Creating a new document

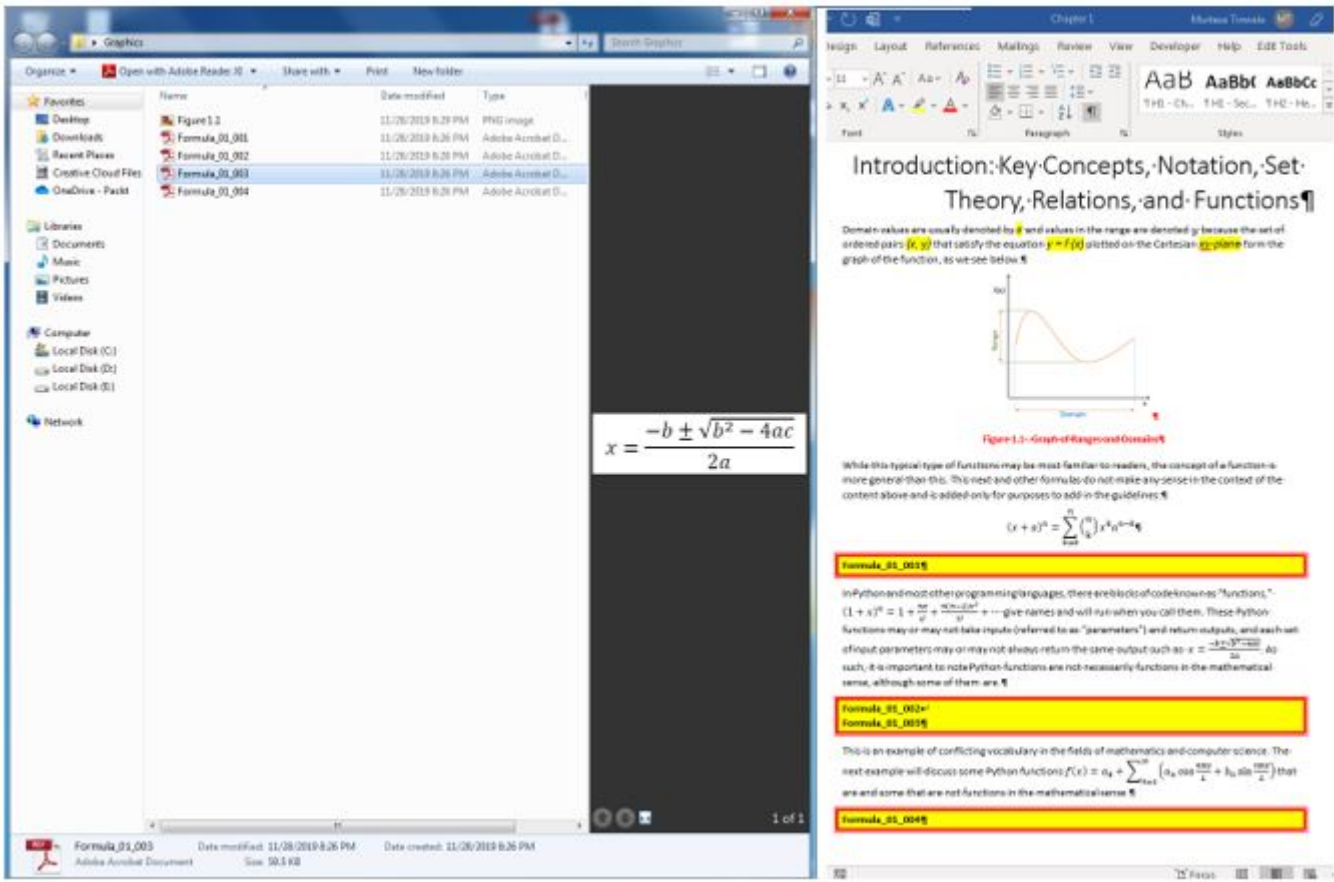
To Production Designer: Please do not break this image and the lead-in on two page, try to adjust it in a single page itself

Now a window will appear, giving you options to help you select the proper settings for your document. Let's discuss each tab of the window.

The text in the yellow box with red outline is the message, and this will not be a part of the actual book.

2. Formulae need to be referenced in the **SP - Editorial** style. Example:





## Acronyms and Abbreviations

Provided that the author specifies the full term at least once, feel free to employ acronyms and abbreviations, as appropriate. When the author first writes the term in full, follow with the abbreviated form between brackets or dashes. For the first appearance of the term and its abbreviation, consider highlighting the word and the abbreviation in **P - Keyword**.

The most common abbreviations do not need to be spelled out, such as PC, CPU, HTML, or even something like XML, depending on whether the target audience of the book will be familiar with this acronym.

If the abbreviated form of a term is introduced and then not used for several chapters, consider using the full form of the term again before subsequently employing the abbreviation.

When pluralizing acronyms, avoid the apostrophe. Write CDs rather than CD's. For acronyms that end with an 's', add 'es' – for example, OSes.

It is important to be consistent when using acronyms or other abbreviations. Do not switch between the full term and abbreviated form within the same chapter, except when introducing the abbreviation, obviously.

While acronyms are fine, you should avoid using abbreviated forms of words even though they would be fine in standard technical conversations. For example, do not use the expression "dev server" when talking about a "development server".

Completely avoid the following abbreviations: "e.g.", "i.e.", "etc.", and "viz.".

## Capitalization

We don't stray from standard practice when it comes to capitalization. Proper nouns take capitals; common nouns do not. Consult the following list for more specific cases:

- Capitalize the names and initials of people: R. Ritter, P. J. O'Rourke, T. Sowell.
- People's titles take capitals: Miss MacDonald, Mr. Smith, Dr. Carson. Be sure to add a period in case of contractions.
- Capitalize company and brand names, unless styled otherwise: eBay, jQuery, and so on.
- Capitalize the names of institutions: the University of Birmingham, for example, or the California Institute of Technology. Be aware of the definite article here: some will take a capital 't'; others will take a lowercase 't'.
- Geographical locations and nationalities take capitals: a Frenchman is usually from France.
- Days and months take capitals.
- For honors and awards, the universal rule is confusing and involves knowing whether the title is a proper name or not. To avoid confusion, we will never capitalize bachelor's and master's.
- Job titles are always in sentence case, unless part of a certification, such as Microsoft Most Valuable Professional.
- Department names should be in title case – the Birmingham Department of Technology, for example. Generic references, however, can be written as "technology department".
- Qualifications and degrees take sentence case.
- Field and industry names take sentence case.

Remember that we do not use capitalization to give emphasis; instead, employ the appropriate style.

Where you come across instances of capitalized terms that are not proper nouns, refer to the tech's official documentation to check the correct capitalization of the term. The order of preference for capitalization of terms is as follows:

1. Whichever way the official documentation presents the term, you should retain it to ensure that readers understand what is being referred to.
2. Where the official documentation does not present terms with initial capitals, change these terms to lowercase to avoid confusing the reader.
3. Where the official documentation is inconsistent and uses a mixture of capitalization and lowercase, pick one style and stick to it.
4. Where there is no clear common usage, retain the author's preference, as long as this will not cause confusion for readers. Again, remain consistent throughout.

## Numbers

Use the following rules when writing numbers:

- Spell out numbers zero through nine if the number does not precede a unit of measurement or is not used as input. Use digits for all other numbers.
- Where the usage includes numbers between zero through nine, and numbers above 10 (for example, "between one and 12"), either spell both out or use digits for both.
- For time periods such as days, months, or years, use digits (for example, "he worked there for 3-6 months").
- For time periods during the day, never start with a zero. For example, say 7:20 rather than 07:20. If it is important that it be in a four-digit format, include the zero.
- For round numbers of 1 million or more, use a numeral plus the word, even if the prefix number is less than 10.
- For numbers with 4 or more digits, insert a comma after every third digit as you move from right to left of the number. Do not add commas to page numbers, numbers in addresses, or numbers that have come from a program output.
- The decimal separator in a decimal number should always be a dot, not a comma.
- Hyphenate numbers and fractions when written in words.
- Do not use a symbol to indicate a fraction; write the fraction in words. Complicated fractions will be retained. For example, 1/5 will be changed to one-fifth but 1/52 will not be changed.
- Chapter numbers or page numbers are always digits.
- Avoid starting a sentence with a numeral. If necessary, add a modifier before a number. If starting a sentence with a number cannot be avoided, spell out the number.

For examples of some of the most common mistakes, see the following table:

Correct	Incorrect
One, two, three, four, five, six	1,2,3,
36 million 1.5 million	36 000 000 1,500,000
3,628 63,487,581 3.14159265	3628 6,348,758,1 3,146159265
Two-thirds	Two thirds
A half One half	½ 1/2
In Chapter 7	In Chapter seven
Microsoft Excel has 144 functions. Eleven example sites are included.	144 functions are available in Microsoft Excel. 11 example sites are included.

## Units

When writing units – centimeters, kilobytes, and so on – follow these rules:

- Leave a space between the value and unit.
- Use digits for all values, even if the number is under 10.
- Use the plural of the unit for unit values more than 1, less than -1, zero, and decimal values.
- Use a hyphen to connect a number to the unit only when a measurement is used as an adjective. In such a case, the unit is always singular.
- Repeat the unit of measurement for two or more quantities.

- The singular and plural forms of unit abbreviations are identical.
- Never use abbreviations for gigabits, kilobits, megabits, microseconds, nanoseconds, months, weeks, or years.

As above, consult the following table for common mistakes:

Correct	Incorrect
182 inches	182inches
93 cm	93cm
1 byte	One byte
100 bytes	Hundred bytes
0 megabits	0 megabit
0.5 megabits	0.5 megabit
1 byte	One byte
100 bytes	Hundred bytes
0 megabits	0 megabit
0.5 megabits	0.5 megabit
12-inch ruler	12 inch ruler
3.5-inch disk	3.5 inch disk
64 KB and 128 KB	64 and 128 KB
1 mile and 4 miles apart	1 and 4 miles apart
167 GB	167 GBs
18 gigabits	18 gb
400 Mbps (megabits per second)	400Mbps
50 MBps (Megabytes per second)	50MBps
10 GB hard disk	10-GB hard disk
10-gigabyte hard disk	10 gigabyte hard disk
5V for voltage 25mA for milliamps	5 V
	25 mA

## Dates

Follow these rules when writing dates:

- Dates are written in the following syntax: month, day, year. This is the American style.
- Always spell out the name of the month, and if the date is greater than (AD) 999, use four digits for the year.
- If the year is not part of the date, then use the format month, day.
- If the day is not part of the date, use the format month, year.
- Do not use ordinal numbers for dates.

Be aware of dates that are the output of a code example: the changing of the formatting of the output date could break the code.

See below for common mistakes:

Correct	Incorrect
December 13, 2006	13 December, 2006
	13/12/06
In the October 13 release	In the Oct 13 release
February 17	Feb 17th

## Important Notes:

1. In the Word templates, we cannot use sub-bullets or bullets within other bullets.
2. Always ensure **consistency** – if you're following an unconventional style for some reason, please ensure that the same style is applied at all instances in the book. Its best to make a note of such deviations as this will help you perform your final checks on all chapters of the book smoothly.