Placeholders

There may be times when you want to add information to your text that either isn't yet known or may change. For example, you may want to:

- include the page number in the header or footer;
- include the word count and current date on your title page;
- add numbers to your chapter titles.

This is where placeholders come in. Placeholders are unique tag words enclosed in angled brackets that you can type anywhere in Scrivener, and which are replaced with specific information in the final compiled document. For instance, if you enter the placeholder <$surname>, when you create your manuscript using Compile, this will be replaced with the current user's surname. <$wc> will be replaced with the current word count.
There is also a set of placeholders to help you create project templates, for when you need to include unknown user information, such as adding a name and address to a title page.
In general, you won't want to litter your writing with placeholders: they are better used for setting up title pages, creating template projects, and setting up reusable Compile options. (For instance, the auto-number placeholders can be used in the title options of Compile to generate chapter numbers; they are used in many of Scrivener's built-in Compile formats.)
You may never need to use placeholders - they are most useful when you want to create your own project templates or Compile formats rather than using the ones that are already provided. For those who want to get their hands dirty, though, provided below is a complete list of the placeholders you can use in Scrivener.

Template Placeholders

Template placeholders are useful when putting together a project you intend to turn into a template using File > Save as Template. When a Scrivener project is created from a template, all of the following placeholders will be replaced with the relevant data before the project appears on screen. You can thus use these placeholders to create templates that include user-specific information such as name and address, which can be useful for setting up title pages.
Tip: To create a new project without having these placeholders automatically replaced, hold down the Alt key while clicking on "Create" in the Project Templates dialog. (This is useful for updating templates or creating your own templates based on existing ones.)

<$template_firstName>	User's forename of first name as set in Contacts.
<$template_lastName>	User's surname or last name as set in Contacts.
<$template_fullName>	User's full name, from Contacts or account settings.
<$template_initial>	First letter of user's forename, as set in Contacts.
<$template_street>	User's street name, taken from Contacts.
<$template_city>	User's city or town, taken from Contacts.
<$template_state>	User's state or county, taken from Contacts.
<$template_ZIP>	User's ZIP or postcode, taken from Contacts.
<$template_country>	User's country, taken from Contacts.
<$template_phoneNumber>	User's phone number, taken from Contacts.
<$template_email>	User's e-mail address, taken from Contacts.
<$template_projectName>	The project title, derived from the project file name.

Scriptwriting Placeholders

These placeholders can only be used in Script Settings (Format > Scriptwriting > Script Settings).

<$mediaPlaybackTime>	Intended for transcription purposes. When the other editor shows a video or sound file, on insertion this tag is replaced with the file's current playback time. Use it inside the "Insert" fields in the Tab/Return area of Script Settings to make it so that hitting tab automatically enters the current playback time of the sound or movie file in the other editor. To specify a custom time format, include the format after a colon and before the closing bracket, like this: <$mediaPlaybackTime:HH:mm:ss> (Only valid in the Tab/Return settings, in the "Insert" field for either "Tabbing on an empty line" or "Tabbing after typing".)

Compile Placeholders

The following placeholders can be entered anywhere in your project (or in the Compile settings) and will be replaced with specific information in your compiled document.
Escaping Compile placeholders: In very rare circumstances (for instance, if you were writing a book about Scrivener), you may find that you want to write *about* a placeholder and so have it appear in the final text without being replaced. For this, Compile placeholders can be "escaped" using the backslash character. Thus, "\<$date>" will appear in the compiled document

as “<$date>”, without being replaced with the current date.

Page Numbers

<$p>	When used in the header or footer, gets replaced with the current page number. When used in the main text and it has an internal document link associated with it, the <$p> placeholder will be replaced with the page number on which the linked document appears if possible.
<$p-r> <$P-R>	When used in the header or footer, gets replaced with the current page number as Roman numerals (use <$p-r> for lowercase Roman numerals and <$P-R> for uppercase Roman numerals). Unlike <$p>, which can show negative numbers for front pages, Roman numerals always start at “i”. Mainly intended for use on front matter pages. (Note that these placeholders may not be supported by all formats.)
<$pagecount>	When used in the header or footer, gets replaced with the page count (note that this placeholder only works in headers and footers). Note that for formats other than standard Print and PDF, this will count *all* pages, even front matter pages that do not have page numbers (for Print and PDF, front matter without page numbers is not included in the count).

Headers and Footers

Not all placeholders are supported in headers and footers. In addition to the special header and footer placeholders specified below (which can *only* be used in headers and footers), only the placeholders listed under the sections *Page Numbers*, *Current Date and Time*, and *User and Project Information* are supported in headers and footers.

<$pageGroupTitle>	Gets replaced with the title of the document that first comes after the page break most recently preceding the header or footer in which it occurs. In practice, this is useful for placing chapter titles in headers or footers. For instance, if you have a chapter in a single text document starting on a new page, then that document’s title will replace “<$pageGroupTitle>”; if your chapters are broken down into smaller sections placed inside folders, and the folders mark the start of each new chapter, then the folder’s title will replace “<$pageGroupTitle>” throughout the chapter headers and footers. Note that if the section title cannot be calculated for any reason (for instance, because there is no text in the section associated with any documents), <$pageGroupTitle> may fall back on using the abbrevated project title (working the same as “<$abbr_projecttitle>” - see *User and Project Information* for details on where the abbreviated project title is drawn from).
<$pageGroupParentTitle>	The same as <$pageGroupTitle>, except that instead of inserting the title of document that first comes after the most recent page break, it inserts that document’s *parent’s* title.

Comments, Footnotes and Layout

<$--ENDNOTES-->	Tells the compiler where to place the endnotes. If this placeholder is not present, the endnotes will get placed at the end - this placeholder just provides a way to customise the placement of the endnotes. (Note that this placeholder cannot be used with script or MultiMarkdown formats, and can only be used with RTF and .docx if “Flatten footnotes…” is selected in the “RTF Compatibility” compile options.)
<$--COMMENTS-->	Tells the compiler where to place linked comments. If this placeholder is not present, linked comments will be placed at the end of the text before any endnotes. Note that this placeholder can only be used with the HTML format.
<$BLANK_PAGE>	Tells the compiler to leave this page blank. At the end of compiling the text, Scrivener goes through looking for potentially blank pages and removes them, but if it finds this placeholder on an otherwise blank page, it just removes the placeholder and leaves the page blank.

Document Variables

Document variable placeholders can be placed anywhere inside the text of a document (note that they cannot be used in the Compile panel’s header and footer fields).

<$title>	Gets replaced with the document title during the Compile process (that is, the title of the document in the binder in which this placeholder occurs). If associated with

	an internal document link, gets replaced with the title of the linked document.
<$parenttitle>	Gets replaced with the title of the parent of the document (e.g. the folder in which the document is contained). If associated with an internal document link, gets replaced with the title of the parent of the linked document.
<$title_no_spaces>	The same as <$title>, but strips all spaces from the title. Potentially useful for compound placeholders. (If associated with an internal document link, gets replaced with the title of the linked document.)
<$parenttitle_no_spaces>	The same as <$parenttitle>, but strips all spaces from the parent title. Potentially useful for compound placeholders. (If associated with an internal document link, gets replaced with the title of the parent of the linked document.)
<$level*N*_title>	Gets replaced with the title of the document at level *N* in the current outline branch. For instance, "<$level1_title>" will be replaced with the title of the document at level 1 of the current Compile group on the same branch as the document in which the placeholder appears. If associated with an internal document link, gets replaced with the title of the document at level *N* in the outline branch of the linked document.
<$level*N*_title_no_spaces>	The same as <$level*N*_title>, but strips all spaces from the title. Potentially useful for compound placeholders. (If associated with an internal document link, gets replaced with the title of the document at the given level in the branch of the linked document.)
<$position>	Gets replaced with the position number of the document within its parent folder or container. Potentially useful for compound placeholders.
<$parentposition>	The same as <$position>, except uses the position of the parent within *its* parent folder or container. Potentially useful for compound placeholders.
<$linkID>	During Compile, each document is assigned a unique number that is used to reference the document in anchor links for HTML and ebook formats. The <$linkID> tag allows you to access and use this unique number in any format. If associated with an internal document link, gets replaced with the unique number of the linked document.
<$label>	Gets replaced with the name of the document label during the Compile process. If associated with an internal document link, gets replaced with the name of the document label of the linked document.
<$status>	Gets replaced with the name of the document status during the Compile process. If associated with an internal document link, gets replaced with the name of the document status of the linked document.
<$keywords>	Gets replaced with a list of keywords associated with the document during the Compile process. If associated with an internal document link, gets replaced with the keywords of the linked document.
<$synopsis>	Gets replaced with the document synopsis during the Compile process. If associated with an internal document link, gets replaced with the synopsis of the linked document.
<$linecount>	Gets replaced with the line count of the document during the Compile process. If associated with an internal document link, the line count will show the number of lines in the linked document.
<$subDocCount>	Gets replaced with the number of child documents the document has during the Compile process. If associated with an internal link, the count will show the number of child documents for the linked document.
<$descendantCount>	Gets replaced with the number of descendant documents the document has during the Compile process. (Descendants include children of children.) If associated with an internal link, the count will show the number of descendants for the linked document.
<$docTarget>	Gets replaced with the document target during the Compile process. If associated with an internal link, the linked document's target will be used.
<$revision>	Gets replaced with the revision number of the document. (The revision number is the number of snapshots associated with the document plus one.)
<$createdDate> <$shortCreatedDate> <$mediumCreatedDate> <$longCreatedDate> <$fullCreatedDate>	Gets replaced with the created date of the document, using one of the formatting styles specified in the System Preferences. (Note that <$createdDate> is the same as <$shortCreatedDate>). To specify a custom time format, include the format after a colon and before the closing bracket, like this: <$createdDate:HH:mm:ss>
<$createdTime> <$shortCreatedTime> <$mediumCreatedTime> <$longCreatedTime> <$fullCreatedTime>	Gets replaced with the created time of the document, using one of the formatting styles specified in the System Preferences. (Note that <$createdTime> is the same as <$shortCreatedTime>).
<$modifiedDate>	Gets replaced with the modified date of the document, using one of the

<$shortModifiedDate> <$mediumModifiedDate> <$longModifiedDate> <$fullModifiedDate>	formatting styles specified in the System Preferences. (Note that <$modifiedDate> is the same as <$shortModifiedDate>). To specify a custom time format, include the format after a colon and before the closing bracket, like this: <$modifiedDate:HH:mm:ss>
<$modifiedTime> <$shortModifiedTime> <$mediumModifiedTime> <$longModifiedTime> <$fullModifiedTime>	Gets replaced with the modified time of the document, using one of the formatting styles specified in the System Preferences. (Note that <$modifiedTime> is the same as <$shortModifiedTime>).
<$custom:…>	Can be used to insert custom metadata in the draft. For instance, if you have added a custom metadata field entitled “Locations” to your project, “<$custom:Locations>” will get replaced with the information in the Locations field for the document during the Compile process. (If associated with an internal document link, gets replaced with the custom metadata of the linked document.)
<$htmlref>	Used by ebook Compile only. Gets replaced with the HTML reference to the document linked to this tag (this tag must be associated with a document link). For instance, this might be replaced with “body2.html#doc10” during Compile to ePub or Kindle formats. This tag provides a way for users to write custom table of contents files in pure HTML if necessary.

Current Date and Time

<$date>	Gets replaced with the current date during the Compile process, using the short date format defined in the user’s System Preferences. To specify a custom time format, include the format after a colon and before the closing bracket, like this: <$date:HH:mm:ss>
<$shortdate>	Gets replaced with the current date during the Compile process, using the short date format defined in the user’s System Preferences.
<$mediumdate>	Gets replaced with the current date during the Compile process, using the medium date format defined in the user’s System Preferences.
<$longdate>	Gets replaced with the current date during the Compile process, using the long date format defined in the user’s System Preferences.
<$fulldate>	Gets replaced with the current date during the Compile process, using the full date format defined in the user’s System Preferences.
<$time>, <$shorttime>	Gets replaced with the current time during the Compile process, using the short time format defined in the user’s System Preferences.
<$mediumtime>	Gets replaced with the current time during the Compile process, using the medium time format defined in the user’s System Preferences.
<$longtime>	Gets replaced with the current time during the Compile process, using the long time format defined in the user’s System Preferences.
<$fulltime>	Gets replaced with the current time during the Compile process, using the full time format defined in the user’s System Preferences.
<$year>	Gets replaced with the current year during the Compile process.
<$shortnumericalmonth>	Gets replaced with the current month during the Compile process, using single or double digits (i.e. “3” for March and “11” for November).
<$numericalmonth>	Gets replaced with the current month during the Comple process, using double digits (i.e. “03” for March” and “11” for November).
<$shortmonth>	Gets replaced with the current month during the Compile process, using the abbreviated month name (e.g. “Mar” for March).
<$month>, <$longmonth>	Gets replaced with the current month during the Compile process, using the full month name.
<$day>, <$shortday>	Gets replaced with the current day of the month during the Compile process, using single or double digits.
<$longday>	Gets replaced with the current day of the month during the Compile process, using double digits only.
<$shortweekday>	Gets replaced with the current day of the week during the Compile process, using the abbreviated weekday name (e.g. “Tues”).
<$weekday>, <$longweekday>	Gets replaced with the current day of the week during the Comple process, using the full weekday name.

User and Project Information

<$surname>, <$lastname>	Gets replaced with the user's surname or last name during the Compile process. The information is taken from the metadata pane of the Compile panel. If the placeholder appears in uppercase, the surname will be uppercased too.
<$forename>, <$firstname>	Gets replaced with the user's forename or firstname name during the Compile process. The information is taken from the metadata pane of the Compile panel. If the placeholder appears in uppercase, the forename will be uppercased too.
<$initial>	Gets replaced with the first letter of the user's forename during the Compile process.
<$author>, <$name>, <$fullname>, <$username>	Gets replaced with the user's full name during the Compile process. The information is taken from the metadata pane of the Compile panel. If the placeholder appears in uppercase, the user's name will be uppercased too.
<$compilegroup>	Gets replaced with the name of the group currently being compiled (as selected in the "Contents" pane of the Compile panel). If the placeholder appears in uppercase, the name will be uppercased too.
<$draftname>	Gets replaced with the title of the Draft folder during the Compile process. If the placeholder appears in uppercase, the title will be uppercased too.
<$projecttitle>, <$projectname>	Gets replaced with the project name during the Compile process. The project name is taken from the metadata pane of the Compile panel. If the placeholder appears in uppercase, the project name will be uppercased too.
<$abbr_projecttitle>, <$abbr_projectname>, <$abbr_title>	Gets replaced with the abbreviated project name during the Compile process. The abbreviated project name is taken from metadata pane of the Compile panel. If the placeholder appears in uppercase, the abbreviated project name will be uppercased too.

Statistics

<$wc>	Gets replaced during the Compile process with the total word count of the text currently being compiled.
<$wc50>	Gets replaced during the Compile process with the total word count of the text currently being compiled, rounded to the nearest 50 words.
<$wc100>	Gets replaced during the Compile process with the total word count of the text currently being compiled, rounded to the nearest 100 words.
<$wc500>	Gets replaced during the Compile process with the total word count of the text currently being compiled, rounded to the nearest 500 words.
<$wc1000>	Gets replaced during the Compile process with the total word count of the text currently being compiled, rounded to the nearest 1000 words.
<$cc>	Gets replaced during the Compile process with the total character count of the text currently being compiled.
<$cc50>	Gets replaced during the Compile process with the total character count of the text currently being compiled, rounded to the nearest 50 characters.
<$cc100>	Gets replaced during the Compile process with the total character count of the text currently being compiled, rounded to the nearest 100 characters.
<$cc500>	Gets replaced during the Compile process with the total character count of the text currently being compiled, rounded to the nearest 500 characters.
<$cc1000>	Gets replaced during the Compile process with the total character count of the text currently being compiled, rounded to the nearest 1000 characters.
<$doccount>	Gets replaced during the Compile process with the number of documents currently being compiled.
<$draftTarget>	Gets replaced during the Compile process with the project Draft target.
<$sessionTarget>	Gets replaced during the Compile process with the project session target.

Auto-Numbering

<$n>	Gets replaced with Arabic numerals during the Compile process. The number is incremented each time a <$n> placeholder is encountered in the text, so "<$n>, <$n>, <$n>" would become "1, 2, 3" in the compiled text.
<$sn>	The same as <$n> but intended to be used for sub-numbering. The count restarts each time an <$n> placeholder is encountered. Thus, "<$n> (<$sn>, <$sn>), <$n> (<$sn>, <$sn>)" would become "1 (1, 2), 2 (1, 2)" in the compiled text.
<$np>	The same as <$n> but automatically reset at the start of each new page. This

	placeholder is only supported when compiling to PDF or printing.
<$r>	Gets replaced with lowercase Roman numerals during the Compile process. The number is incremented each time a <$r> placeholder is encountered in the text, so "<$r>, <$r>, <$r>" would become "i, ii, iii" in the compiled text.
<$R>	Gets replaced with uppercase Roman numerals during the Compile process. The number is incremented each time a <$R> placeholder is encountered in the text, so "<$R>, <$R>, <$R>" would become "I, II, III" in the compiled text.
<$l>	Gets replaced with lowercase outline (alphabetical) numbering during the Compile process. The number is incremented each time an <$l> placeholder is encountered in the text, so "<$l>, <$l>, <$l>" would become "a, b, c" in the compiled text.
<$L>	Gets replaced with uppercase outline (alphabetical) numbering during the Compile process. The number is incremented each time an <$L> placeholder is encountered in the text, so "<$L>, <$L>, <$L>" would become "A, B, C" in the compiled text.
<$w>	Gets replaced with numbers as lowercase words (using the current language settings) during the Compile process. The number is incremented each time a <$w> placeholder is encountered in the text, so "<$w>, <$w>, <$w>" would become "one, two, three" in the compiled text.
<$t>	Gets replaced with numbers as title-case words (using the current language settings) during the Compile process. The number is incremented each time a <$t> placeholder is encountered in the text, so "<$t>, <$t>, <$t>" would become "One, Two, Three" in the compiled text.
<$W>	Gets replaced with numbers as uppercase words (using the current language settings) during the Compile process. The number is incremented each time a <$W> placeholder is encountered in the text, so "<$W>, <$W>, <$W>" would become "ONE, TWO, THREE" in the compiled text.
<$hn>, <$ahn>, <$aon>, <$hn_0>, <$hn_level*N*>	Gets replaced during the Compile process with hierarchical numbering based on the level of the document in which the placeholder occurs relative to the Draft folder or compile group (depending on the current compile settings). So occurrences of the <$hn> placeholder in the second document in the Draft folder may get replaced with the number "2"; occurrences of the placeholder in the third subdocument of the eighth document in the Draft folder may be replaced with "8.3". You can assign internal document links to the <$hn> placeholder to refer to other sections - if the <$hn> placeholder has a document link associated with it, it will be replaced with the hierarchical numbering of the linked document. Note, however, that this only works if the linked document contains an <$hn> placeholder itself (for instance, in its title). You can use <$ahn> for alphabetical hierarchical number (a.1, a.1.2 and so on), and <$AHN> for a capitalised version of this. <$aon> provides alphanumerical outline numbering (I.A.1.a.i). Using <$hn_0> will cause the first number to start at 0 (other numbers still start at 1). You can set the starting level using the <$hn_level*N*> variant, where "N" represents the starting level. For instance, if you use <$hn_level1>, then "1", "2" and so on would be used for items nested in the first level deep rather than for top-level documents. Note that whichever tag is first found in a document - <$hn>, <$ahn>, <$hn_0> or <$hn_level*N*> - will determine the hierarchical number used for that document throughout. When creating a link to a document so that the document link will be replaced with the hierarchical number of that document, always use the <$hn> tag. Hierarchical numbering restarts whenever a different tag type is found.
Restarting auto-numbering streams: <$rst>, <$rst_…>	Place <$rst> immediately before any of the auto-numbering placeholders to restart the numbering. So, for instance, "<$w>, <$w>, <$w>, <$rst><$w>, <$w>" would become "one, two, three, one, two" in the compiled text. Alternatively, you can place <$rst_*X*> anywhere in the text, replacing the "X" with the letter used in the auto-numbering placeholder you wish to restart. E.g. <$rst_R> would restart the uppercase Roman numeral auto-numbering from that point onwards.
Using named auto-numbering streams: <$n:…>, <$w:…>	You can assign names to any of the auto-numbering variables to create unique streams by inserting a colon and any name of your choosing between the auto-numbering letter and the final bracket. For instance, you could use "<$t:part>" (where "part" is the name you have chosen) as the auto-numbering placeholders for the titles of parts in a book, and "<$t:chapter>" for the chapters. In this example, the text "Part <$t:part>, Chapter <$t:chapter>, Chapter <$t:chapter>, Chapter <$t:chapter>, Part <$t:part>, Chapter <$t:chapter>, Chapter <$t:chapter>" would result in "Part One, Chapter One, Chapter Two, Part Two,

Chapter Three, Chapter Four" in the compiled text. Please note that names cannot contain any whitespace (such as spaces).
You can refer to the numbers generated in this way using the format <$n#keyword>. This is replaced with the current number without incrementing it. For instance, the text, "Figure <$n:figure>, Figure <$n:figure>, Please see figure <$n#figure>" would result in "Figure 1, Figure 2, Please see figure 2" in the compiled text. If you add a document link to a number reference, the reference placeholder will be replaced with the numerical value of the placeholder as it was for the linked document. (E.g. If you are using <$t:chapter> in chapter titles and use <$t#chapter> linked to a document inside Chapter Four, then "<$n#chapter>" will be replaced with "Four".)

Figure and table numbering:
<$n:table:myTableName>
<$n:figure:myFigureName>

You can create special auto-numbering placeholders that include a name *and* a keyword to enable you to refer back to auto-numbers, for instance for referring to tables and figures. The format of such auto-numbering placeholders is this:

<$[auto-number-type]:[name]:[keyword]>

Subsequent instances of placeholders that use the same auto-number type, name and keyword will be replaced with the same number as was generated for the first instance of that combination; only placeholders that have a different keyword will cause the number to be incremented. This is best explained with an example:

[An image]
Figure <$n:figure:myImage>

[Another image]
Figure <$n:figure:nextImage>

[A table]
Table <$t:table:myTable>

See Figure <$n:figure:myImage>, Figure <$n:figure:nextImage> and Table <$t:table:myTable>.

When compiled, this will result in the following:

[An image]
Figure 1

[Another image]
Figure 2

[A table]
Table One

See Figure 1, Figure 2 and Table One.

Note how the placeholder "<$n:figure:myImage>" occurring later in the text was replaced with the same number ("1") generated for the first instance of that placeholder.

If you want to create a forward reference, for the reference use the format:

<$[auto-number-type]#[name]:[keyword]>
(i.e. Use a hash instead of the first colon.)

E.g:

<$n:eg:foo> Example sentence.

See example <$n#eg:foo> and compare it with <$n#eg:bar> below.

<$n:eg:inbetweener> Another sentence.
<$n:eg:bar> The final sentence.

When compiled, this will result in the following:

1 Example sentence.

See example 1 and compare it with 3 below.

2 Another sentence.
3 The final sentence.

	Using names and keywords in auto-numbering placeholders can thus allow you to refer to table and figure numbers. (Note that names and keywords cannot contain any whitespace characters such as spaces.)
Restarting named auto-numbering streams: <$rst_KEYWORD>	You can restart one of the named auto-numbering streams by using the <$rst_KEYWORD> placeholder anywhere before an occurrence of one of the placeholders. For instance, <$rst_imageNumber> would restart placeholders such as <$n:imageNumber> or <$n:imageNumber:myImage>.
Compound placeholders	You can create compound placeholders by using other placeholders inside the named auto-numbering or figure and table numbering placeholders. For instance, it is possible to use "<$n:figure:<$parentposition>>" as a valid figure auto-numbering placeholder, because the document variables such as <$parentposition> will get replaced before the auto-numbering variables.
Making placeholders more readable	You can use the Compile panel's "Replacements" pane to make compound and other placeholders more readable while you are writing your text. For instance, in the Stageplay UK project template, act and scene numbers appear in the text as "ACT [N] SCENE [R]". This phrase is then replaced in the Compile panel's replacement pane with "ACT <$R:ACT:<$PARENTPOSITION>> SCENE <$N:<$PARENTPOSITION>>". This allows each scene to reference its act number, because a folder is used to hold the scenes of each act, so "<$PARENTPOSITION>" can be used to give each act a unique placeholder. The result is this: "ACT I SCENE 1", "ACT I SCENE 2", ACT I SCENE 3", "ACT II SCENE 1", "ACT II SCENE 2", and so on.

Images and Text

<$img:imgName> <$img:imgPath> <$img:imgNameOrPath;w=*x*;h=*y*>	You can use the <$img…> placeholder to have images inserted into your text during Compile. This can be useful if you want to insert an image into a title prefix or suffix, if you want to keep images out of the text while writing, or if you want to include certain images only conditionally (you could use the "Replacements" pane of Compile to remove image placeholders for images you don't want to appear in a particular Compile format, for instance). Image placeholders support image documents that have been imported into the project - in which case you should use the name of the document on its own - or file paths to images on disk. For instance: <$img:My Image Document> <$img:~/Pictures/My Image File.png> (If the image file is stored in the same directory as, or a subdirectory of, the .scriv project, a relative path can be used.) You can also define the width and height of the image thus: <$img:My Image Document;w=400;h=200> As in HTML, specifying only one of either the width or height will cause the other dimension to be scaled proportionally: <$img:~/Pictures/My Image File.png;w=400> For ePub 3 and Mobi KF8 ebook export, you can also set a width percentage, like this: <$img:Img Doc;ebook=50%> The "ebook" setting will be ignored by other formats. To set a fixed width for other formats and a percentage width for ebooks, just use both (the "ebook" setting will override the width and height settings for ebooks if present): <$Img:Img Doc;w=400;ebook=75%> Remember that the image will use the same formatting as the placeholder, so if you want the image to be centred with no line spacing, you must format the placeholder accordingly. Tip: If you have more than one image in the project with the same name, you can specify the exact image by using an internal document link.
<$include> <$include:textNameOrPath>	You can use the <$include…> placeholder to have text from an external document inserted into the text during Compile. This can be useful if you have text snippets that need repeating throughout for some reason. To insert the main text of any document in the current project, either use the <$include> tag and apply a document link to the whole tag that points to the document whose text you wish to insert, or use <$include:docName>, where "docName" is the name of the document you wish to be inserted. (Using document links is the more reliable method.) To insert the text of an external file, either use the <$include> tag with a link to the external file, or use <$include:docPath>, where "docPath" is either the full or relative path of the document whose text you wish to insert (relative paths should

	be relative to the folder in which the project is stored). Note that only plain text UTF-8 files and RTF files are supported, and only the text of such files will be included (images, footnotes and comments are ignored in external RTF files that are inserted in this manner). The <$include…> tag is only supported in the following areas: the main text; notes; title prefixes and suffixes; section layout prefixes and suffixes. You can also use it in the ebook "Description" metadata field, but only if it appears on its own.

Miscellaneous

<$toc>	Used when compiling ebook formats only. If the <$toc> placeholder appears in an otherwise blank document when compiling to an ebook format, and if that document's title matches the title assigned to the table of contents (in the Compile options), then this document will be replaced with an automatically-generated table of contents. This allows the user to determine where the automatically-generated table of contents should be placed in cases where it is not wanted at the very start of the ebook. (Tip: Set up your Compile options so that there is a page break before this document and the document's formatting is not overridden.)
<$ebook_start>	Used when compiling to Kindle .mobi format only. If this is present in a document and "Book begins after front matter" is not ticked in the compile options, the exported Kindle file will be set to open at the section that contains this placeholder. Note that it may not work as expected if you place it in a table of contents page, because some Kindle devices do not seem to like this.
<$nav_start>, <$nav_end>	The <$nav_start> and <$nav_end> tags can be used to tell the navigation (table of contents) document in an ePub3 file where the <nav epub:type="toc"> block should start and end. If these tags are not included in the contents document, the <nav> block will start from either the top of the file or after the first empty line above a block including links, and will finish at the end of the file.
<$char_name>	For scriptwriting: <$char_name> can be inserted into the (MORE) field of the scriptwriting options in the Text Layout area of Compile (for PDF and Print only). It gets replaced by the character name so that you can have, for instance, "(JOHN/CONT'D OVER)" below dialogue that is split across pages.