

# **GREP in InDesign CS3**

**Michael Murphy**

[www.theindesigner.com](http://www.theindesigner.com) | [info@theindesigner.com](mailto:info@theindesigner.com)



**CERTIFIED EXPERT**  
InDesign®

## What is GREP?

GREP stands for General Regular Expression Parser. A regular expression is a means of describing patterns or conditions within text, and need not necessarily include one word or character of actual text. GREP allows search-and-replace operations that are based on those patterns and conditions, not on literal text. GREP's origins are in the realm of early computer languages that most people—myself included—find cryptic and off-putting. But GREP is not as formidable as it may initially seem, and once you experience its power first-hand, and understand its potential, any resistance you may have toward learning it will melt away.

The fundamental difference between a text search and a GREP search is that a text search looks for exact text and replaces it with other exact text. For example—find every instance of “dog” and replace it with “cat.” It's a one-to-one exchange.

A GREP search could look for every instance “dog” *or* “cat” *or* “hamster” *or* “parakeet” and replace all of them with “pet” in a single Find/Change operation. That's a conditional search. “Or” is a condition—this *or* that *or* another thing *or* something else.

More typically, GREP searches for patterns. Patterns describe text in general terms. Consider a paragraph. How would you describe a paragraph? A paragraph is a range of text characters followed by a hard return. That's a pattern. Two paragraphs in a row would be a range of text followed by a return followed by another range of text followed by another return. It doesn't matter what the words in those paragraphs are. GREP doesn't care. GREP only cares about the pattern.

Patterns are everywhere, whether you know it or not. Once you've used GREP a few times, you'll start to see patterns where you never noticed them before.

Even in a regular text search, if you leave “case sensitive” turned off and search for the word *sail*, the search isn't looking for *SAIL or sail*, it's looking for *S or s, then A or a, then I or i, then L or l*. If you turn on “whole word” it's looking for *the exact sequence of letters “sail” surrounded by word boundaries* (either the beginning of a line, end of a line, a white space or punctuation). That pattern defines what a “whole word” is—the letters just define what *specific* word it is. Therefore, it won't find *sailboat or assailant*.

With GREP, you can define your own patterns and conditions. Since you're not looking for specific words, you can have one search operation add to, delete from, or re-arrange that pattern while leaving the original text intact.

So how is this magic performed? How are those patterns and conditions described? It's accomplished through the use of special characters (or combinations of characters) called *metacharacters*.

## Metacharacters

Most of the time, a GREP search will work like a regular text search. Most characters match themselves. If you type a “t” and click Find, you’ll get a “t” as a match. However, if you type “\t”, you don’t get a backslash character followed by a “t”—you get a tab. The backslash makes the lower-case “t” a “metacharacter”—a pair of characters that represent another character.

Metacharacters are used in both the Text and GREP areas of the Find/Change dialog box. In Text searches, they’re typically used when the character itself cannot be typed using its standard keyboard equivalent. For example: a tab can’t be typed in the Find What field because, in that context, the tab key moves the cursor into the next field in the dialog box. Instead, the tab has to be represented by the metacharacter **^t**. You’ve probably seen these metacharacters many times when using special characters in a Find/Change operation or nested style instruction. In GREP, metacharacters are far more powerful.

Many metacharacters are comprised of two characters: a letter preceded by a specific modifier (typically a backslash). In the Text section of the Find/Change dialog box metacharacters begin with a caret (^). A tab character is represented as **^t**, a paragraph return as **^p**, and a soft return as **^n**, just to name a few. Metacharacters in the GREP section begin with a backslash (\) or a tilde (~). The backslash is the conventional GREP method of indicating a metacharacter, while the tilde is used only for characters that are unique to InDesign (i.e., flush space, anchored object marker, auto page number, etc.).

However, there are some metacharacters that are comprised of only one character. For example, a period means “any character (except a hard return)”, so if you need to include an *actual* period as part of your search, you have to “escape out” the period by adding a backslash before it. It can be a bit confusing at first, since the same character that makes a normal character into a metacharacter is also used to make a metacharacter into a normal character. The table below identifies these single-character metacharacters, their GREP function, and how to match them literally.

GREP METACHARACTER	WHAT IT MATCHES	TO FIND IT LITERALLY
<b>.</b> period	Wildcard: Any Character	\.
<b>^</b> caret	Location: Beginning of Paragraph	^\^
<b>\$</b> dollar sign	Location: End of Paragraph	\\$
<b>?</b> question mark	Repeat: Zero or One Time	\?
<b>*</b> asterisk	Repeat: Zero or More Times	\*
<b>+</b> plus sign	Repeat: One or More Times	\+
<b> </b> vertical divider	Match: Or	\
<b>(</b> opening parenthesis	Match: Marking Subexpression (start)	\(
<b>)</b> closing parenthesis	Match: Marking Subexpression (end)	\)
<b>[</b> opening square bracket	Match: Character Set (begin definition)	\[
<b>]</b> closing square bracket	Match: Character Set (end definition)	\]

## Metacharacter Types

Different metacharacters perform different functions, and they can be organized into different types. The simplest of these types is the **wildcard**. Examples of wildcards are any character (**.**), any digit (**\d**), any white space (**\s**), etc. These are fairly self-explanatory. They act as placeholder, standing in for different character types.

Another class of metacharacter controls **repetition**, meaning how many or how few times something is matched. Examples of repeat metacharacters are zero or one time (**?**), zero or more times (**\***), and one or more times (**+**). Combined with wildcards, repeat metacharacters form the foundation of useful and versatile queries like any character one or more times (**.+**) or any digit zero or more times (**\d\***). Repetition can be controlled more precisely by quantifying the number of matches with specific numeric ranges surrounded by curly brackets (**{}**). For instance, **\d{3}** finds any three digits in a row and **\d{3,6}** finds *at least* three digits in a row, but *no more than* six digits in a row.

The next class of metacharacters describes a **location** such as the beginning of a word (**\<**), beginning of a paragraph (**^**), end of a word (**\>**), or end of a paragraph (**\$**). Here again, combining these metacharacters with others makes search queries even more specific and powerful. For instance, the query **^\d+?\.**  will find any one or more digits at the beginning of a paragraph that are followed by a period.


Greater specificity can be built into GREP queries with **match** metacharacters. A *range* of characters to be found is defined by enclosing them in square brackets (**[ ]**) so that **[aeiou]** will match any vowel. Match metacharacters can also define a condition such as “or” (**|**). The GREP query for finding either of the whole words *dog or cat or goldfish or parakeet* would be: **\<cat\>|\<dog\>|\<parakeet\>|\<goldfish\>**

Preceding each word is the “beginning of word” metacharacter and following each is the “end of word” metacharacter. This prevents accidentally finding the “cat” in “location” or the “dog” in “dogma.” It could probably be left out for parakeet and goldfish.

Other match metacharacters define and “remember” sub-patterns (also known as sub-expressions) that can then be referred to and/or re-used later in the search query, or in the change portion of the Find/Change operation. For instance, in a 10-digit U.S. phone number, the overall pattern for the number 800-555-1212 would be **\d{3}-\d{3}-\d{4}**. However, for GREP to retain those sets of digits for future use, they must be enclosed in parentheses. That would look like this: **(\d{3})-(\d{3})-(\d{4})**.

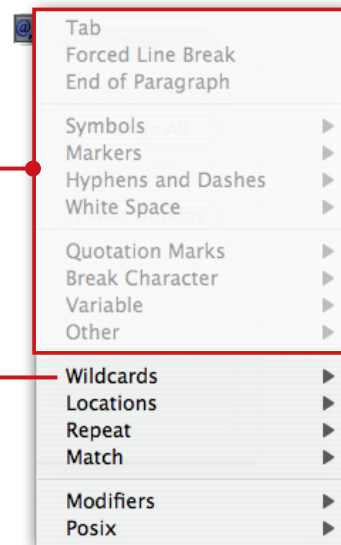
These sub-patterns play a key role in the Change To portion of a GREP search. Each sub-pattern can be referenced by the order in which it appears in the search. In the example above, recalling the area code is accomplished with the metacharacter **\$1**, meaning “Found Text 1” (the first of the three subpatterns in the phone number). The first three digits of the phone number would be **\$2** and the last four would be **\$3**. If, for some unfathomable reason, you wanted to swap the position of the area code and the first three digits, your Change To field would contain the following: **\$2-\$1-\$3**.

## Special Characters for Search

If the idea of learning and memorizing the many special characters, instructions, range limiters, and wildcards of GREP is an intimidating one, the Special Characters for Search flyout menu  at the end of the Find What field allows you to build your GREP queries element-by-element.

For a complete list, search the InDesign CS3 Help files for the word “metacharacter” and click the first match in the list: “Metacharacters for Searching” to view a table showing every possible metacharacter for text-based searches and GREP searches.

All special characters in this portion of the menu are the same as those in the Text area of the Find/Change dialog box. However, the “escape characters” that define them in GREP searches are either backslashes (\) or tildes (~), not carets (^).



### Wildcards

• Any Digit	\d
• Any Letter	[\u]
• Any Character	.
• Any White Space	\s
• Any Word Character	\w
• Any Uppercase Letter	\u
• Any Lowercase Letter	\l

### Locations

• Beginning of Word	\<
• End of Word	\>
• Word Boundary	\b
• Beginning of Paragraph	^
• End of Paragraph	\$

### Repeat

• Zero or One Time	?
• Zero or More Times	*
• One or More Times	+
• Zero or One Time ( <i>Shortest Match</i> )	??
• Zero or More Times ( <i>Shortest Match</i> )	*?
• One or More Times ( <i>Shortest Match</i> )	+?

### Match

• Marking Subexpression	()
• Non-marking Subexpression	(?:)
• Character Set	[]
• Or	
• Positive Lookbehind	(?<=)
• Negative Lookbehind	(?<!)
• Positive Lookahead	(?=)
• Negative Lookahead	(?!)

### Modifiers

• Case-insensitive On	(?i)
• Case-insensitive Off	(?-i)
• Multiline On	(?m)
• Multiline Off	(?-m)
• Single-line On	(?s)
• Single-line Off	(?-s)

### Posix

• [[[:alnum:]]]	<b>any letter or number</b>
• [[[:alpha:]]]	<b>any letter</b>
• [[[:digit:]]]	<b>any digit</b>
• [[[:lower:]]]	<b>any lowercase character</b>
• [[[:punct:]]]	<b>any punctuation character</b>
• [[[:space:]]]	<b>any white space character</b>
• [[[:upper:]]]	<b>any uppercase character</b>
• [[[:word:]]]	<b>any letter, number or underscore</b>
• [[[:x-digit:]]]	<b>any hexadecimal digit</b>
• [[[:a=]]]	<b>any character of a certain glyph set (such as a, à, á, â, ã, ä, å, Æ, Å, Á, Â, Ã, Ä and Å)</b>

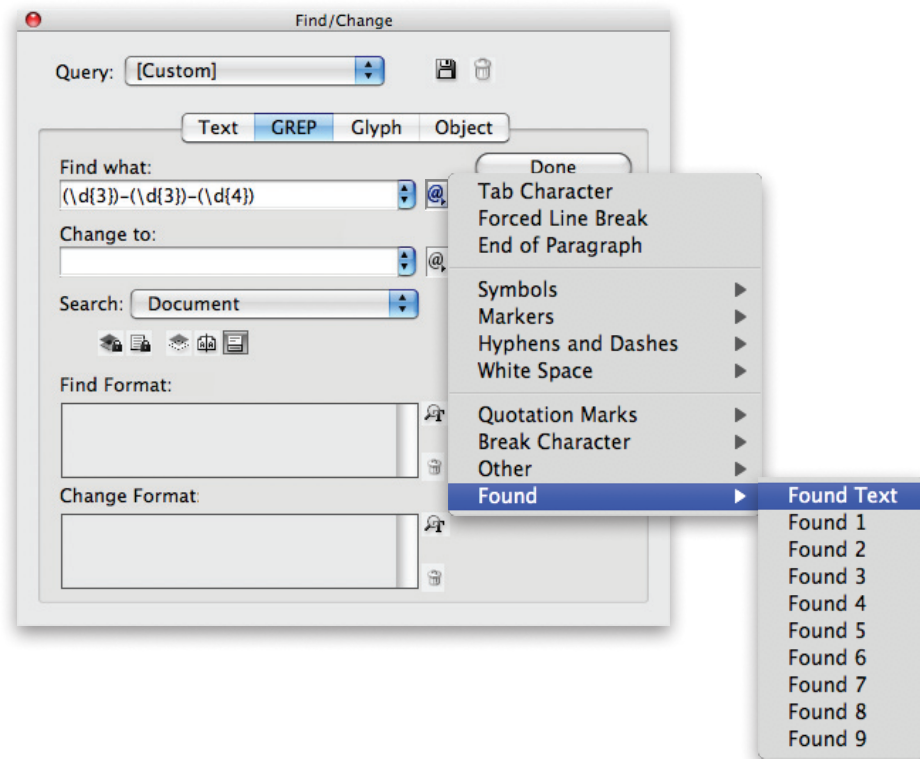
A complete list of metacharacters (text and GREP) is available in Adobe’s online help files at:  
[http://livedocs.adobe.com/en\\_US/InDesign/5.0/help.html?content=WSa285fff53dea4f8617383751001ea8cb3f-6f59.html](http://livedocs.adobe.com/en_US/InDesign/5.0/help.html?content=WSa285fff53dea4f8617383751001ea8cb3f-6f59.html)

## Special Characters for Replace

The Change To field also has a Special Characters flyout menu, but it has far fewer options than those available for the Find What field. When establishing “Change To” parameters, many of the metacharacter types (wildcards, location, match, repeat, etc.) have no use. For instance, there’s no way to use a wildcard like any digit in a replace instruction. What would that digit be? How would InDesign determine that?

Because of this, the Wildcards, Locations, Repeat, Match, Modifiers and Posix options do not appear in the Special Characters for Replace flyout menu. Instead, only one GREP-specific option is present: Found. The Found sub-menu includes Found Text and Found 1 through Found 9.

Found Text (**\$0**) refers to everything found by the search instruction, and the Found 1 (**\$1**) through Found 9 (**\$9**) options are a way to recall specific sub-patterns (or sub-expressions) from the search based on the order in which they appear. If your pattern contains more than 9 sub-patterns, you’ll have to split it into separate queries.



## Case-Sensitivity

All GREP searches are case-sensitive. However, case sensitivity can be turned off at any point within a query using the metacharacter (**?-i**), then turned back on again at any point using the metacharacter (**?i**).

## Limiting and Quantifying Searches

Grep searches are greedy. They will try to make the longest possible match. Suppose you're looking within a paragraph for any text that's enclosed in parenthesis. If you build a query to find *an opening parenthesis, followed by any one or more characters, followed by a closing parenthesis*, it would look like this: `\(.+\)`. Notice that the parentheses have been "escaped out" with backslashes because parentheses are the GREP metacharacter for defining a sub-expression. Unfortunately, if there is more than one piece of information in the paragraph that's enclosed in parentheses, this search query will match everything from the very first opening parenthesis to the very last closing parenthesis in the paragraph, and everything in-between. To prevent this, the "shortest match" limiter (`?`) must be added to the pattern just before the closing parenthesis, changing the meaning of the search to: *an opening parenthesis, followed by one or more characters up to the first closing parenthesis found*.

### `\(.+\)` matches all of this:

InDesign CS3 sports a new user interface that replaces palettes with panels. In the CS3 products, palette groups are now called panel stacks, and these stacks reside in a dock **or multiple docks, if that's how you prefer to work** that you can view either fully expanded, reduced to an icon and name mode, or collapsed down to a single column of icons (Figure 1). This minimized icon-only view replaces the hinged tab method for stowing away palettes of previous versions.

### `\(.+?\)` matches only this:

InDesign CS3 sports a new user interface that replaces palettes with panels. In the CS3 products, palette groups are now called panel stacks, and these stacks reside in a dock **or multiple docks, if that's how you prefer to work** that you can view either fully expanded, reduced to an icon and name mode, or collapsed down to a single column of icons (Figure 1). This minimized icon-only view replaces the hinged tab method for stowing away palettes of previous versions.

But what if you want to find text that's in parentheses, but only change the text *within* them, not the parentheses themselves? In that case, you want to find any string of text (computer geek speak for any one or more characters), but only if there's an opening parenthesis before it, and a closing parenthesis after it (and, as above, making the shortest match possible). To accomplish that, you must add conditional qualifiers to the search known as Positive Lookbehind (`?=`) meaning *only if this is present before the text*, and Positive Lookahead (`?<=`) meaning *only if this is present after the text*. The full search query would look like this: `?<=\(.+?\)=` and it would match the following:

InDesign CS3 sports a new user interface that replaces palettes with panels. In the CS3 products, palette groups are now called panel stacks, and these stacks reside in a dock **or multiple docks, if that's how you prefer to work** that you can view either fully expanded, reduced to an icon and name mode, or collapsed down to a single column of icons (Figure 1). This minimized icon-only view replaces the hinged tab method for stowing away palettes of previous versions.

GREP also includes Negative Lookbehind and Lookahead which you would use to impose the conditions *as long as this is not present before the text* or *as long as this is not present after the text*, respectively.

## Standardizing Phone Numbers Using GREP

In this example, a list of U.S. phone numbers using non-standard formatting (*left column*) needs to be formatted consistently (*right column*). I want to impose consistency where there apparently is none. Or isn't there? If you get past the inconsistent formatting, every U.S. phone numbers has one thing in common: it's made up of ten digits.

### WHAT I HAVE:

800-833-6687  
 888-649-2990  
 800-945-9120  
 1-800-642-3623  
 1-800-945-9120  
 800-642-3623  
 1 (800) 767-2775  
 (877) 412-7753  
 (800) 277-5356  
 (877) 412-7753  
 (877)427-5776  
 (408) 974-2042  
 (800) 793-2378  
 1-(888) 840-8433  
 1(800)-692-7753  
 (800) 780-5009  
 8009877543  
 1.800.800.2775  
 800.538.9696  
 800.300.3532  
 949.756.5108

### WHAT I WANT:

(800) 833-6687  
 (888) 649-2990  
 (800) 945-9120  
 (800) 642-3623  
 (800) 945-9120  
 (800) 642-3623  
 (800) 767-2775  
 (877) 412-7753  
 (800) 277-5356  
 (877) 412-7753  
 (877) 427-5776  
 (408) 974-2042  
 (800) 793-2378  
 (888) 840-8433  
 (800) 692-7753  
 (800) 780-5009  
 (800) 987-7543  
 (800) 800-2775  
 (800) 538-9696  
 (800) 300-3532  
 (949) 756-5108

The ten-digit phone number itself is a pattern. Within it, there are sub-patterns: a three-digit area code, followed by another three digits, then the last four digits.

The challenge is to build a Find/Change query that will find those patterns, account for and remove any inconsistencies in what exists around them, then apply preferred, consistent formatting, leaving the original ten digits of each number intact.

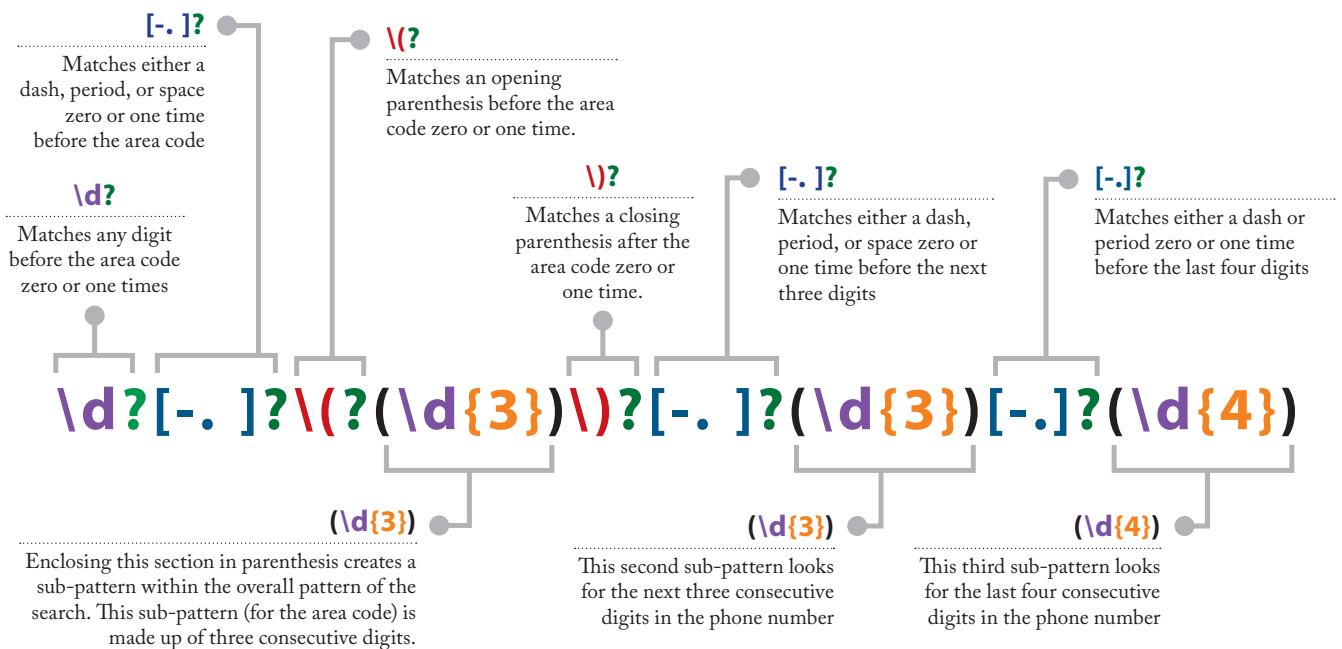
Using GREP, this can be done in one operation using metacharacters. Some metacharacters will stand in for the numbers themselves (as they are all different), other metacharacters will recognize that some parts of the search may or may not produce a match (i.e., zero or one instance; one or more instances, etc.), and others will consider any of a

range of characters (i.e, either this or that, whichever comes first) to be a match.

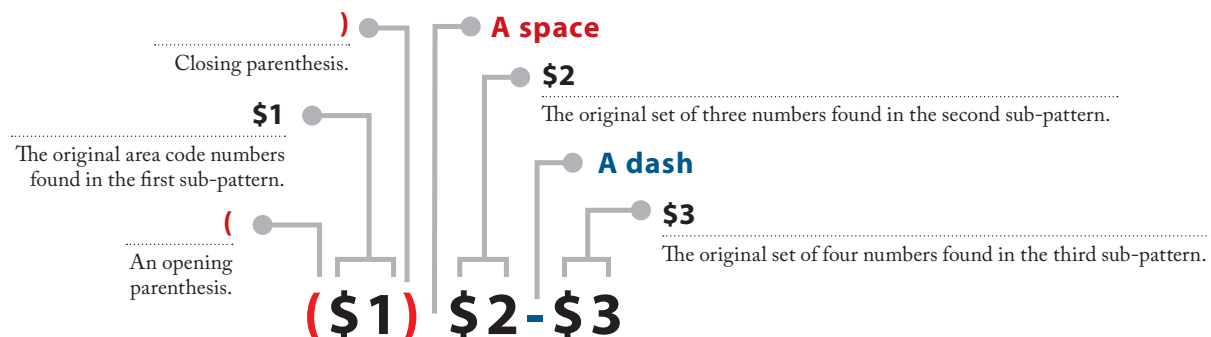
Key to understanding GREP is to think of your searches not in terms of the text itself, but rather in terms of the *idea* of the search. Since you're not searching for exact text, you need to put what you're searching for in the proper context.

On the next page, I've written a verbal description that takes a "big picture" look at what needs to be done. This is a good way to think about the *idea* of the search: what you are trying to accomplish. Below that, I've broken down the GREP query that will do it. Each piece of the written description is color-coded to associate the *idea* of the search with the corresponding metacharacters in the GREP query.

Find a digit (that may or may not be there), followed by either a dash, period or space (that may or may not be there), followed by an opening parenthesis (that may or may not be there), followed by a set of three consecutive digits, followed by a closing parenthesis (that may or may not be there), followed by either a dash, period or space (that may or may not be there), followed by a second set of three consecutive digits, followed by either a dash or a period (that may or may not be there), then a set of four consecutive digits.



Replace that with an opening parenthesis, the first set of three digits, a closing parenthesis, a space, the second set of three digits, a dash, then the last set of four digits.



## Italicizing Figure References Using GREP

An InDesign text search contains some wildcard options (any digit, any letter, any character, and any white space), but they have significant limitations compared to GREP. For example, a Text search for figure reference in parentheses within a document could use the search query **(Figure ^d)**, but that would only match Figures 1 through 9. No double-digit figures would be found, requiring a second search operation for **(Figure ^d^d)**.

However, neither of these would be of any help with references such as (Figure 1 and Figure 2) or (Figures 12 and 13). A GREP search, on the other hand, can be set up to match all possible figure references. At the same time, the query can be built to verify the *presence* of the surrounding parentheses, but exclude them from the results, allowing me to apply an italic character style in the Change operation to everything *except* the parentheses.

Below, the “idea” of the search is presented, followed by the GREP pattern (broken down step-by-step), followed by each potential match made by this search. The elements in each presentation of the query are color-coded to make the relationships clearer.

**Find, but do not include, an opening parenthesis**, followed by the *exact text* **Figure**, followed by a **letter s** (that *may or may not be there*), followed by a **space**, followed by **one or more digits**, followed by **any white space character** (that *may or may not be there*), followed by **any character or characters** (that *may or may not be there*), followed by **any white space character** (that *may or may not be there*), followed by a **digit or digits** (that *may or may not be there*), followed by **a closing parenthesis that does not get included in the results**.

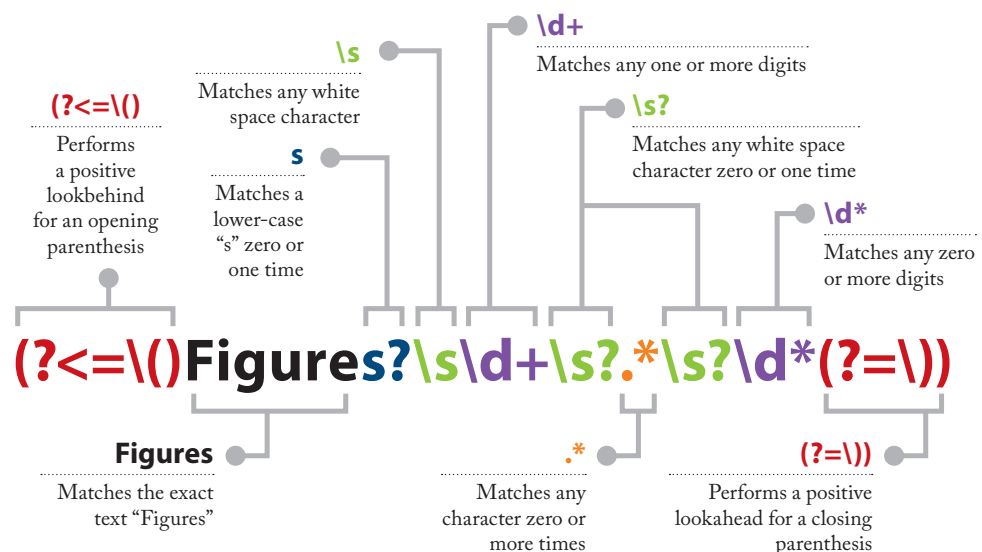
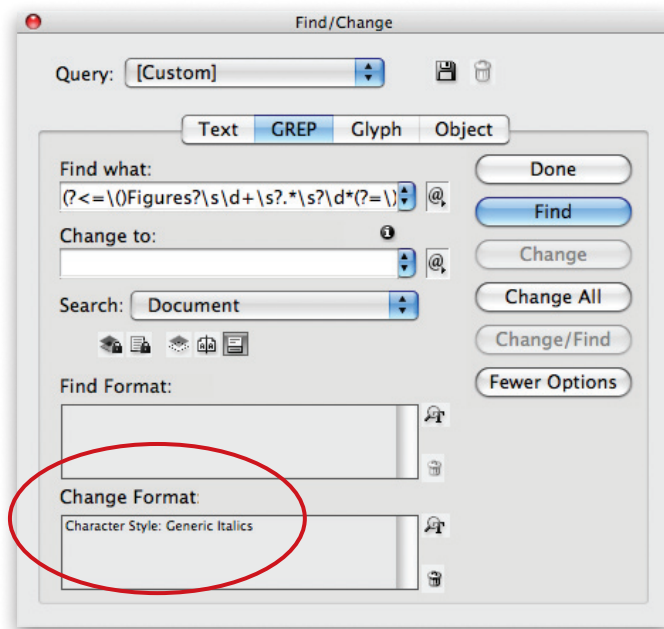


Figure 1   Figure 23   Figure 147   Figures 12 and 13   Figure 9 & Figure 10

Since the text itself will remain unchanged, there's no need to put anything in the Change To field. Applying the italics is accomplished through the Change Format options at the bottom of the Find/Change dialog. Typically, leaving the Change To field blank is the same as deleting the text (replacing it with nothing), but when formatting attributes are part of the Change operation, a blank Change To field means "leave the text itself unchanged."



Here, GREP is used in combination with InDesign's ability to apply formatting as part of a Find/Change operation. Any individual font attribute can be applied to the search results, but in this instance, I'm assigning a Character Style with only one defined attribute: Italics.

## Where to Learn More

Ben Forta's concise and clear book, *Teach Yourself Regular Expressions in 10 Minutes* is an excellent reference that will allow you to approach GREP from the ground up, or jump in anywhere that matches your current level of GREP knowledge, then build from there.

Mac users who have Bare Bones Software's outstanding BBEdit text editing application have an indispensable GREP resource available to them—the GREP Reference section of BBEdit's help documentation. It is both thoroughly explained and well organized.

Adobe's live online help documents at [http://www.adobe.com/go/learn\\_id\\_grep](http://www.adobe.com/go/learn_id_grep) provide much more information than the Help files that ship with the application itself. Better still, the documentation is appended by comments from users and Adobe staff and contains information about additional GREP-related resources.

“...thoughtful, articulate and well-prepared...brilliant...an invaluable resource...inspirational...shows you the how as well as the why...amazing... everything you need to know to be a proficient user...mind-blowing... a must for all graphic designers... phenomenal...an intelligent, elegant source of information.”\*



“The gold standard of podcast tutorials.”



Hosted by designer, writer, trainer, presenter, and Adobe Certified Expert Michael Murphy – “a veteran art director with an encyclopedic knowledge of Adobe’s InDesign”\*\* – The InDesigner is a free video podcast dedicated to empowering designers to embrace concepts and features in the application that will transform how they work.

information. instruction. insight.

**theindesigner**

[www.theindesigner.com](http://www.theindesigner.com)

\* excerpted from reviews posted on the iTunes Store

\*\* (The Top 40 Tech Podcasts, .net magazine, February 2007)