2BrightSparks
POWER WITH EASE

A Brief Introduction to Regular Expressions

Introduction

This article introduces regular expressions, also known as “regex”. People who are programmers or web designers may be more familiar with the functionalities of regular expressions, although some regular PC users also make use of this powerful search tool that extracts text by matching specific words, characters or patterns.

What are Regular Expressions?

A Regular Expression is a sequence of character strings that represents a search pattern. It is used to match complex patterns of text with minimal effort, which otherwise cannot be derived from conventional searching methods. Regular expressions are sometimes referred to as “advanced wildcards”. The majority of text editors can perform simple searches of specific words or a string of characters, but they are not powerful enough to identify varying strings of characters or patterns, like a variety of different phone numbers or email addresses. However, regular expressions can do this and much more.

How are Regular Expressions used?

Regular expressions can be used for all types of text-based manipulation tasks, but it is mainly used for matching, replacement and extraction. Regular expressions can be used to find text that matches a pattern, replace matched text with other text, or extract certain portions of the text for later use.

Regular expressions are widely supported in software applications and programming languages like Java, Oracle, Perl and many more. Regex is a powerful tool that programmers use to replace the time-consuming task of writing hundreds of lines of code with a single line of cryptic letters and symbols.

Several applications also integrate regular expressions into their search functions, albeit that some software designers integrate them in a slightly modified form. The backup and synchronization programs SyncBackSE and SyncBackPro can optionally use regular expressions in the Filters setting to match patterns to determine which files to copy and which to skip. Both of these programs also use regular expressions optionally in the Versioning setting to selectively keep or skip multiple revisions of files for backup.

Regular Expression Syntaxes and Examples

There are many symbols used in regular expressions to represent the different functionalities for each syntax. Some examples include the anchor, character classes, meta-characters, special characters, alternations and more.

Below are definitions of some commonly used syntaxes:

Character	Definition
^	The caret denotes the expression to match from the beginning of the string. It is an “anchor” character
.	The period is used to match any character
$	The dollar sign represents the expression to match from the end of the string. It is an “anchor” character
[]	Square brackets groups characters into character classes
()	Brackets group characters or patterns
{n}	Curly brackets matches exactly n times
|	The vertical bar symbol represents alternate matches
*	Asterisk matches 0 or more times
+	Plus sign matches 1 or more times
?	Question mark matches 1 or 0 times

Some examples of using regular expressions to match patterns in strings are presented below:

Expression	Matching Definition
[469]	Matches the single digit **4**, **6** or **9**
[0-9]	Matches any single digit from **0 - 9**
[A-Za-z0-9]	Matches any single character that is either an **uppercase** letter, a **lowercase** letter or a **digit**
^abc	Matches **abc** at the **beginning** of character strings
abc$	Matches **abc** at the **end** of character strings
a.c	Matches **a** followed by **any single character** and followed by **c**
Abc{3}d	Matches **ab** followed by **exactly three c’s** and followed by **d**
[lt](ie|ook)	Matches ‘**l**’ and ‘**t**’ to ‘**ie**’ or ‘**ook**’ for the words **lie**, **tie**, **look** and **took**
abc*d	Matches **ab** followed by **zero or more c’s** and followed by **d**
abc+d	Matches **ab** followed by **one or more c’s** and followed by **d**
abc?d	Matches **ab** followed by **an optional c** and followed by **d**

Summary

Although regular expressions can be complex, they can also be very useful in searching more conveniently for patterns in text. As the number of data files we keep grow every day, regular expressions can be very useful for those occasions when you need the Filters function in SyncBackSE and SyncBackPro to optionally search for only specific files to backup. Although 2BrightSparks does not offer bespoke help designing regex filters, users of their software will find valuable information about regex on Microsoft’s Quick Reference page and more detailed information at www.regular-expressions.info.