Why can't I get my regular expression pattern to match words that begin with %?

devblogs.microsoft.com/oldnewthing/20100423-00

April 23, 2010

Raymond Chen

A customer asked for help writing a regular expression that, in the customer's words, matched the string `%1` when it appeared as a standalone word.

Match	No match
%1	%1b
:%1:	x%1

One of the things that people often forget to do when asking a question is to describe the things that they tried and what the results were. This is important information to include, because it saves the people who try to answer the question from wasting their time repeating the things that you already tried.

Pattern	String	Result	Expected
\b%1\b	%1	No match	Match
\b%1\b	:%1:	No match	Match
\b%1\b	x%1	Match	No match
^..$	%1	Match	Match

That last entry was just to make sure that the test app was working, a valuable step when chasing a problem: First, make sure the problem is where you think it is. If the `^..$` hadn't worked, then the problem would not have been with the regular expression but with some other part of the program. "Is the `\b` operator broken?" No, the `\b` operator is working just fine. The problem is that the `\b` operator doesn't do what you think it does. For those not familiar with this notation, well, first you were probably confused by the `\b` in the original question and skipped the rest of this article. Anyway, `\w` matches A through Z (either

uppercase or lowercase), a digit 0 through 9, or an underscore. (It's actually more complicated than that, but the above description is good enough for the current discussion.) By contrast, `\W` matches every other character. And in regular expression speak, a "word" is a maximal contiguous string of `\w` characters. Finally, the `\b` operator matches the location between a `\w` and a `\W`, treating the beginning and end of the string as an invisible `\W`. I will stop mentioning the pretend `\W` at the ends of the string; just mentally insert them where applicable. Okay, let's go back to the original regular expression of `\b%1\b`. Notice that the percent sign is not one of the things which is matched by `\w`. Therefore, in order for the `\b` that comes before it to match, the character before the percent sign must be a `\W`. That way, the `\b` comes between a `\w` and a `\W`. The pattern `\b%1\b` means "A percent sign which comes after a `\w`, followed by a 1 which comes before a `\W`." Looking at it another way, the string `%1` breaks down like this:

`\W`	beginning of string (virtual)				
`\W`	`%`	`\w`	`1`	`\W`	end of string (virtual)

There is a `\b` between the `%` and the `1` and another one between the `1` and the end of the string, but there is no `\b` before the percent sign, because that location has `\W` on both sides. The question started off on the wrong foot: You are having trouble writing a regular expression that matches a word that begins with `%` because *there are no words which begin with* `%`. The percent sign is not a `\w` and therefore cannot be part of a word. What the customer is looking for is something more like `(?<!\w)%1\b`, a regular expression which means *a percent sign not preceded by a* `\w`*, followed by a 1 which comes before a* `\W`.

The customer realized the mistake once it was pointed out. "I keep forgetting that I can't get `%` included in `\w` just because I want it to."

Michael Kaplan covered this same topic some time ago

.

Raymond Chen

Follow