The Java™ Tutorials

Trail: Essential Classes
Lesson: Regular Expressions

The Java Tutorials have been written for JDK 8. Examples and practices described in this page don't take advantage of improvements introduced in later releases.

Boundary Matchers

Until now, we've only been interested in whether or not a match is found *at some location* within a particular input string. We never cared about *where* in the string the match was taking place.

You can make your pattern matches more precise by specifying such information with *boundary matchers*. For example, maybe you're interested in finding a particular word, but only if it appears at the beginning or end of a line. Or maybe you want to know if the match is taking place on a word boundary, or at the end of the previous match.

The following table lists and explains all the boundary matchers.

Boundary Construct	Description
`^`	The beginning of a line
`$`	The end of a line
`\b`	A word boundary
`\B`	A non-word boundary
`\A`	The beginning of the input
`\G`	The end of the previous match
`\Z`	The end of the input but for the final terminator, if any
`\z`	The end of the input

The following examples demonstrate the use of boundary matchers `^` and `$`. As noted above, `^` matches the beginning of a line, and `$` matches the end.

```
Enter your regex: ^dog$
Enter input string to search: dog
I found the text "dog" starting at index 0 and ending at index 3.

Enter your regex: ^dog$
Enter input string to search:       dog
No match found.

Enter your regex: \s*dog$
Enter input string to search:             dog
I found the text "            dog" starting at index 0 and ending at index 15.

Enter your regex: ^dog\w*
Enter input string to search: dogblahblah
I found the text "dogblahblah" starting at index 0 and ending at index 11.
```

The first example is successful because the pattern occupies the entire input string. The second example fails because the input string contains extra whitespace at the beginning. The third example specifies an expression that allows for unlimited white space, followed by "dog" on the end of the line. The fourth example requires "dog" to be present at the beginning of a line followed by an unlimited number of word characters.

To check if a pattern begins and ends on a word boundary (as opposed to a substring within a longer string), just use `\b` on either side; for example, `\bdog\b`

```
Enter your regex: \bdog\b
Enter input string to search: The dog plays in the yard.
I found the text "dog" starting at index 4 and ending at index 7.

Enter your regex: \bdog\b
```

```
Enter input string to search: The doggie plays in the yard.
No match found.
```

To match the expression on a non-word boundary, use `\B` instead:

```
Enter your regex: \bdog\B
Enter input string to search: The dog plays in the yard.
No match found.

Enter your regex: \bdog\B
Enter input string to search: The doggie plays in the yard.
I found the text "dog" starting at index 4 and ending at index 7.
```

To require the match to occur only at the end of the previous match, use `\G`:

```
Enter your regex: dog
Enter input string to search: dog dog
I found the text "dog" starting at index 0 and ending at index 3.
I found the text "dog" starting at index 4 and ending at index 7.

Enter your regex: \Gdog
Enter input string to search: dog dog
I found the text "dog" starting at index 0 and ending at index 3.
```

Here the second example finds only one match, because the second occurrence of "dog" does not start at the end of the previous match.

Previous page: Capturing Groups
Next page: Methods of the Pattern Class