→ Regular expressions

17th October 2019

Multiline mode of anchors ^ $, flag "m"

The multiline mode is enabled by the flag `m`.

It only affects the behavior of `^` and `$`.

In the multiline mode they match not only at the beginning and the end of the string, but also at start/end of line.

Searching at line start ^

In the example below the text has multiple lines. The pattern `/^\d/gm` takes a digit from the beginning of each line:

```
let str = `1st place: Winnie
2nd place: Piglet
3rd place: Eeyore`;

alert( str.match(/^\d/gm) ); // 1, 2, 3
```

Without the flag `m` only the first digit is matched:

```
let str = `1st place: Winnie
2nd place: Piglet
3rd place: Eeyore`;

alert( str.match(/^\d/g) ); // 1
```

That’s because by default a caret `^` only matches at the beginning of the text, and in the multiline mode – at the start of any line.

Please note:

“Start of a line” formally means “immediately after a line break”: the test `^` in multiline mode matches at all positions preceeded by a newline character `\n`.

And at the text start.

Searching at line end $

The dollar sign `$` behaves similarly.

The regular expression `\d$` finds the last digit in every line

```
let str = `Winnie: 1
Piglet: 2
Eeyore: 3`;

alert( str.match(/\d$/gm) ); // 1,2,3
```

Without the flag `m`, the dollar `$` would only match the end of the whole text, so only the very last digit would be found.

> **Please note:**
>
> “End of a line” formally means “immediately before a line break”: the test `$` in multiline mode matches at all positions succeeded by a newline character `\n`.
>
> And at the text end.

Searching for \n instead of ^ $

To find a newline, we can use not only anchors `^` and `$`, but also the newline character `\n`.

What’s the difference? Let’s see an example.

Here we search for `\d\n` instead of `\d$`:

```
let str = `Winnie: 1
Piglet: 2
Eeyore: 3`;

alert( str.match(/\d\n/gm) ); // 1\n,2\n
```

As we can see, there are 2 matches instead of 3.

That’s because there’s no newline after `3` (there’s text end though, so it matches `$`).

Another difference: now every match includes a newline character `\n`. Unlike the anchors `^` `$`, that only test the condition (start/end of a line), `\n` is a character, so it becomes a part of the result.

So, a `\n` in the pattern is used when we need newline characters in the result, while anchors are used to find something at the beginning/end of a line.

Previous lesson Next lesson

Share Tutorial map

Comments

- If you have suggestions what to improve - please submit a GitHub issue or a pull request instead of commenting.
- If you can't understand something in the article – please elaborate.
- To insert a few words of code, use the `<code>` tag, for several lines – use `<pre>`, for more than 10 lines – use a sandbox (plnkr, JSBin, codepen…)

about the projectcontact usterms of usage privacy policy