Ruby master - Bug #18471

Regex#match(re, position) with start of a string anchors ^ and \A

01/11/2022 08:14 AM - vlazar (Vlad Zarakovsky)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
<tr>
<td>Backport:</td>
<td>2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN</td>
</tr>
</tbody>
</table>

Description

Found this. Is it a bug in Regex#match(re, position)?

From Regex#match docs

> If the second parameter is present, it specifies the position in the string to begin the search.

```ruby
str = "hello world"
/\^world/.match(str, 6) # => nil
/\Aworld/.match(str, 6) # => nil
```

I would expect "specifies the position in the string to begin the search" would mean staring from position 6 in "hello world" should be equivalent to staring with position 0 in a string "world" and thus ^ or \A should match in this case too.

Consider another example. If I use the same with StringScanner#scan it works as I would expect:

```ruby
require "strscan"
str = "hello world"
scanner = StringScanner.new(str)
scanner.pos = 6
scanner.scan /\^world/ # => "world"
scanner.pos = 6
scanner.scan /\Aworld/ # => "world"
```

To me these 2 cases (at least looking into current API docs) should work the same and having a difference in behavior looks like a bug to me.

History

#1 - 01/11/2022 08:54 AM - sawa (Tsuyoshi Sawada)

I [...] expect [...] staring from position 6 in "hello world" [to] be equivalent to staring with position 0 in [the sub-]string "world"

I do not think so. Moving "the position in the string to begin the search" does not mean the string will be truncated. It only moves the cursor position in the same string. It does not create a new string or a new edge of a string.

On the other hand, I find the behavior of StringScanner#scan to be potentially confusing, but I think I can understand why it is designed so. To my understanding, as you read from a StringScanner instance, the matched part is consumed, the original content is truncated, and indeed new edges are created as you read from it. And the same thing seems to be happening when the position is moved. (Saying somewhat metaphorically. I do not know about the actual implementation.)

Thus, I think String and StringScanner differ in nature. And particularly, String#match is not destructive whereas StringScanner#scan, StringScanner#pos= are destructive.

#2 - 01/11/2022 08:55 AM - znz (Kazuhiro NISHIYAMA)

\G matches.

```ruby
str = "hello world"
/\Gworld/.match(str, 6)
```
#3 - 01/12/2022 12:45 AM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Rejected