Ruby master - Bug #15583

Regex: ? on quantified group {n} is interpreted as optional, should be lazy

02/01/2019 03:28 PM - davisjam (James Davis)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.6.1	**Backport:**	2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN

Description

The Ruby regex docs have this to say about repetition (specific link):

> The constructs described so far match a single character. They can be followed by a repetition metacharacter to specify how many times they need to occur. Such metacharacters are called quantifiers.
>
> - * - Zero or more times
> - ...
> - {n} - Exactly n times
> - ...

From this I conclude that the {n} construct is considered a quantifier metacharacter.

The docs go on to say

> Repetition is greedy by default: as many occurrences as possible are matched while still allowing the overall match to succeed. By contrast, lazy matching makes the minimal amount of matches necessary for overall success. A greedy metacharacter can be made lazy by following it with ?.

Since {n} is a greedy metacharacter, it seems like {n}? should make it lazy. In the particular case of {n}?, laziness is meaningless -- the regex engine must match n of whatever is being quantified, lazily or not. But I think other behavior is needlessly confusing. To make {n} optional, I think I should have to wrap it in parentheses: (a{n})?.

The docs make it sound like ? as "lazy" has stronger precedence than ? as "optional".
This make sense to me -- the "optional" meaning can be communicated using parentheses while the lazy meaning cannot.

Here is a test program to explore this behavior:

```
if /a{1,}?/.match("")
    puts "a{1,}? matched the empty string"
else
    puts "a{1,}? did not match"
end

if /a{1,3}?/.match("")
    puts "a{1,3}? matched the empty string"
else
    puts "a{1,3}? did not match"
end

if /a{,1}?/.match("")
    puts "a{,1}? matched the empty string"
else
    puts "a{,1}? did not match"
end

if /a{1}?/.match("")
    puts "a{1}? matched the empty string"
else
    puts "Did not match"
```

```
end
```

If ? attaches more strongly to quantifers (to mean non-greedy) than to arbitrary patterns (to mean optional), then I expect it to mean "non-greedy" in each of these cases. So the expected behavior is:

1. /a{1,}?/ *should not* match the empty string, since even non-greedily it must match at least 1 a.
2. /a{1,3}?/ *should not* match the empty string, since even non-greedily it must match at least 1 a.
3. /a{,1}?/ *should* match the empty string, since non-greedily it can match 0 a's.
4. /a{1}?/ *should not* match the empty string, since even non-greedily it must match at least 1 a.

Let's see how it behaves in Ruby 2.6.1:

```
(09:43:09) jamie@woody /tmp $ ruby -v
ruby 2.6.1p33 (2019-01-30 revision 66950) [x86_64-linux]
(09:43:12) jamie@woody /tmp $ ruby /tmp/t.rb
a{1,}? did not match
a{1,3}? did not match
a{,1}? matched the empty string
a{1}? matched the empty string
```

Cases 1-3 all behave as expected. However, case 4 matches the empty string, implying that in /a{1}?/ the ? interpreted to mean optional rather than non-greedy.
I find this inconsistency a bit confusing.

I tested this behavior in 7 other languages: Go, Java, JavaScript, Perl, PHP, Python, and Rust. In those languages, /a{1}?/ does not match the empty string (and is thus the {n}? notation interpreted as non-greedy rather than optional).

Perhaps this should be addressed via a docs change to avoid possible breakage. Here is some possible wording:

Repetition is greedy by default: as many occurrences as possible are matched while still allowing the overall match to succeed. By contrast, lazy matching makes the minimal amount of matches necessary for overall success. Most greedy metacharacters can be made lazy by following them with ?. For the {n} metacharacter, greedy and non-greedy behavior is identical and the ? instead makes the repeated pattern optional.

Associated revisions

Revision cd0e2089 - 08/28/2019 06:50 PM - jeremyevans (Jeremy Evans)

Document {n}? regexp pattern is optional and not non-greedy [ci skip]

While not consistent with {n,}?, {,m}?, and {n,m}?, it is arguably
more useful, as otherwise the ? would have no effect.

Fixes [Bug #15583]

History

#1 - 02/01/2019 03:28 PM - davisjam (James Davis)

- ruby -v set to 2.6.1

#2 - 02/02/2019 06:45 AM - naruse (Yui NARUSE)

- Backport changed from 2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN to 2.4: REQUIRED, 2.5: UNKNOWN, 2.6: UNKNOWN

Unfortunately it's expected behavior and the bug of documentation.

The upstream of Ruby's regexp, Onigmo, has such non greedy feature, but Ruby disables it.
https://github.com/k-takata/Onigmo/blob/master/doc/RE#L155-L156

It's because of compatibility of Ruby 1.8 and prior. Ruby allowed /a{n}?/ before it introduces Oniguruma/Onigmo, we cannot change the behavior.
If we introduce it, it requires some migration path to the new behavior.
If you want to have non greedy repetition with quantifier in a real world application, we consider the reasonable migration paths harder though it takes some years...

#3 - 02/02/2019 06:45 AM - naruse (Yui NARUSE)

- Backport changed from 2.4: REQUIRED, 2.5: UNKNOWN, 2.6: UNKNOWN to 2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN

#4 - 02/16/2019 02:21 AM - davisjam (James Davis)

Can we change the documentation? I am happy to propose additional text.

#5 - 02/16/2019 05:13 AM - duerst (Martin Dürst)

davisjam (James Davis) wrote:

> Perhaps this should be addressed via a docs change to avoid possible breakage.

Agreed.

> Here is some possible wording:
>
> Repetition is greedy by default: as many occurrences as possible are matched while still allowing the overall match to succeed. By contrast, lazy matching makes the minimal amount of matches necessary for overall success. Most greedy metacharacters can be made lazy by following them with ?. For the {n} metacharacter, greedy and non-greedy behavior is identical and the ? instead makes the repeated pattern optional.

The term "metacharacter" is usually used for single characters. So "the {n} metacharacter" sounds really strange. '{' and '}' are metacharacters, but "{n}" is not a metacharacter (because it's not a character in the first place).

This should be taken into account when rewriting the documentation.

#6 - 08/28/2019 06:55 PM - jeremyevans (Jeremy Evans)

- Status changed from Open to Closed

Applied in changeset git|cd0e208963bdf9ee2fec30e83bdc8f6bc77a7b8d.

Document {n}? regexp pattern is optional and not non-greedy [ci skip]

While not consistent with {n,}?, {,m}?, and {n,m}?, it is arguably
more useful, as otherwise the ? would have no effect.

Fixes [Bug #15583]

Files

t.rb	397 Bytes	02/01/2019	davisjam (James Davis)