Ruby master - Bug #13671

Regexp with lookbehind and case-insensitivity raises RegexpError only on strings with certain characters

06/22/2017 11:28 PM - dschweisguth (Dave Schweisguth)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.4.1	**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

Here is a test program:

```
def test(description)
  begin
    yield
    puts "#{description} is OK"
  rescue RegexpError
    puts "#{description} raises RegexpError"
  end
end

test("ass, case-insensitive, special") { /(?<!ass)/i =~ '□' }
test("bss, case-insensitive, special") { /(?<!bss)/i =~ '□' }
test("as,  case-insensitive, special") { /(?<!as)/i  =~ '□' }
test("ss,  case-insensitive, special") { /(?<!ss)/i  =~ '□' }
test("ass, case-sensitive,   special") { /(?<!ass)/  =~ '□' }
test("ass, case-insensitive, regular") { /(?<!ass)/i =~ 'x' }
```

Running the test program with Ruby 2.4.1 (macOS) gives

```
ass, case-insensitive, special raises RegexpError
bss, case-insensitive, special raises RegexpError
as,  case-insensitive, special is OK
ss,  case-insensitive, special is OK
ass, case-sensitive,   special is OK
ass, case-insensitive, regular is OK
```

The RegexpError is "invalid pattern in look-behind: /(?<!ass)/i (RegexpError)"

Side note: in the real code in which I found this error I was able to work around the error by using (?i) after the lookbehind instead of //i.

Running the test program with Ruby 2.3.4 does not report any RegexpErrors.

I think this is a regression, although I might be wrong and it might be saving me from an incorrect result with certain strings.

Related issues:

Related to Ruby master - Bug #14838: RegexpError with double "s" in look-behi... **Open**

History

#1 - 06/23/2017 08:49 AM - Hanmac (Hans Mackowiak)

did some checks on my windows system to check how deep the problem is.
i used "ä" as variable.

the same problem happens when you try to use match function too:
/(?<!ass)/i.match('ä')
also happen for
Regexp.union(/(?<!ass)/i, /ä/)

but i still don't understand why it does crash with ass, while ss works.

might have something todo how regexp are stored internal

#2 - 07/14/2017 09:51 AM - naruse (Yui NARUSE)

I created a ticket in upstream: https://github.com/k-takata/Onigmo/issues/92

#3 - 08/27/2018 02:35 AM - gotoken (Kentaro Goto)

I encountered a non ss case. Is this a same problem?

```
% ruby -ve '"".match(/(?<=ast)/ui)'
ruby 2.6.0dev (2018-08-27 trunk 64549) [x86_64-linux]
-e:1: invalid pattern in look-behind: /(?<=ast)/i
```

It was reproduced in version 2.4 and 2.5.
#14838 seems to be duplicate.

#4 - 08/27/2018 03:46 AM - znz (Kazuhiro NISHIYAMA)

You can use (?:s) instead of s for workaround.

```
$ ruby -ve '/(?<=ast)/iu'
ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-darwin17]
-e:1: invalid pattern in look-behind: /(?<=ast)/i
-e:1: warning: possibly useless use of a literal in void context
$ ruby -ve '/(?<=a(?:s)t)/iu'
ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-darwin17]
-e:1: warning: possibly useless use of a literal in void context
```

#5 - 08/27/2018 03:47 AM - znz (Kazuhiro NISHIYAMA)

- Related to Bug #14838: RegexpError with double "s" in look-behind assertion in case-insensitive unicode regexp added

#6 - 08/27/2018 05:44 AM - gotoken (Kentaro Goto)

Thanks znz. The workaround is helpful. And I understood what was happened.
https://github.com/k-takata/Onigmo/issues/92#issuecomment-373981492 shows how some combinations of letters are variable length.

For example, "ss" and "st" are mapped "ß" ("\u00DF") and "ﬆ" ("\uFB06").
Those combinations are listed in ftp://ftp.unicode.org/Public/UNIDATA/SpecialCasing.txt

By the way, this expansion by //i option looks over kill for me.
I wish case sensitivity and SpecialCasing mapping were separated...

#7 - 08/27/2018 06:02 AM - shyouhei (Shyouhei Urabe)

gotoken (Kentaro Goto) wrote:

> By the way, this expansion by //i option looks over kill for me.
> I wish case sensitivity and SpecialCasing mapping were separated...

I know how you feel. Too bad we are just doing what Unicode specifies to do.

See also http://unicode.org/faq/casemap_charprop.html#11

#8 - 08/27/2018 06:31 AM - gotoken (Kentaro Goto)

Thanks shyouhei for your pointing out.

I imagine another Rexexp option, say //I, which is almost the same as //i except for never-applying SpecialCasing mapping.
This change extends Unicode matching indeed but does not introduce incompatibilities, IMHO.
A difficulty is the implementation is on the upstream library and cruby is just a user.

#9 - 08/29/2018 10:20 AM - duerst (Martin Dürst)

gotoken (Kentaro Goto) wrote:

> For example, "ss" and "st" are mapped "ß" ("\u00DF") and "ﬆ" ("\uFB06").
> Those combinations are listed in ftp://ftp.unicode.org/Public/UNIDATA/SpecialCasing.txt
>
> By the way, this expansion by //i option looks over kill for me.
> I wish case sensitivity and SpecialCasing mapping were separated...

I still have to verify this, but currently I strongly suspect that the problem is NOT in SpecialCasing, but in how Onigmo (/Oniguruma?) implement it.

#10 - 04/07/2020 06:40 PM - mauromorales (Mauro Morales)

FYI The issue has been addressed in Onigmo https://github.com/k-takata/Onigmo/pull/116 and has already been released in version 6.2.0. I tried it by applying the changes using Ruby 2.6.6 and it works as expected.

Files

test.rb	531 Bytes	06/22/2017	dschweisguth (Dave Schweisguth)