Ruby - Feature #12831

/\X/ (extended grapheme cluster) can't pass unicode.org's GraphemeBreakTest

10/11/2016 04:01 PM - mtsmfm (Fumiaki Matsushima)

Status:	Closed	
Priority:	Normal	
Assignee:	naruse (Yui NARUSE)	
Target version:		

Description

I'm trying to replace Rails's grapheme implementation (http://api.rubyonrails.org/classes/ActiveSupport/Multibyte/Unicode.html#method-i-unpack_graphemes) with Ruby's extended grapheme cluster (/X/).
https://github.com/rails/rails/issues/26743

I noticed that Ruby's grapheme cluster can't pass unicode.org's GraphemeBreakTest.

Following test script will fail on Ruby 2.2/2.3

```
require 'rubygems'
require 'open-uri'
require 'test/unit'

UNICODE_VERSION =
  if Gem::Version.new(RUBY_VERSION) >= Gem::Version.new("2.3.0")
    "8.0.0"
  else
    "7.0.0"
  end

class TestGrapheme < Test::Unit::TestCase

# https://github.com/rails/rails/blob/v5.0.0.1/activesupport/test/multibyte_grapheme_break_conform
ance_test.rb#L37
  def test_breaks
    each_line_of_break_tests do |*cols|
      *clusters, comment = *cols
      string = clusters.map {|c| c.pack("U*") }.join
      assert_equal clusters, string.scan(/\X/).map(&:codepoints), comment
    end
  end

  def each_line_of_break_tests(&block)
    lines = 0
    max_test_lines = 0 # Don't limit below 21, because that's the header of the testfile
    URI.parse("http://www.unicode.org/Public/#{UNICODE_VERSION}
/ucd/auxiliary/GraphemeBreakTest.txt").open do |f|
      until f.eof? || (max_test_lines > 21 && lines > max_test_lines)
        lines += 1
        line = f.gets.chomp!
        next if line.empty? || line.start_with?("#")

        cols, comment = line.split("#")
        # Cluster breaks are represented by ÷
        clusters = cols.split("÷").map { |e| e.strip }.reject { |e| e.empty? }
        clusters = clusters.map do |cluster|
          # Codepoints within each cluster are separated by ×
          codepoints = cluster.split("×").map { |e| e.strip }.reject { |e| e.empty? }
          # codepoints are in hex in the test suite, pack wants them as integers
          codepoints.map { |codepoint| codepoint.to_i(16) }
        end

        # The tests contain a solitary U+D800 <Non Private Use High
        # Surrogate, First> character, which Ruby does not allow to stand
```

```
        # alone in a UTF-8 string. So we'll just skip it.
        next if clusters.flatten.include?(0xd800)

        clusters << comment.strip

        yield(*clusters)
      end
    end
  end
end
```

https://gist.github.com/mtsmfm/38f46882c3d4ccde35c269594fc24ebc

I found an issue on Onigmo (https://github.com/k-takata/Onigmo/issues/46)
but I couldn't on bugs.ruby-lang.org so I created this ticket.

I'm unfamiliar with grapheme so please tell me if I get something wrong.

Associated revisions

Revision c11e648799cf32d267875381d967e8228a07cea6 - 11/30/2016 05:29 PM - naruse (Yui NARUSE)

Regexp supports Unicoe 9.0.0's \X

- meta character \X matches Unicode 9.0.0 characters with some workarounds
  for UTR #51 Unicode Emoji, Version 4.0 emoji zwj sequences.
  [Feature #12831] [ruby-core:77586]

The term "character" can have many meanings bytes, codepoints, combined
characters, and so on. "grapheme cluster" is highest one of such words,
which means user-perceived characters.
Unicode Standard Annex #29 UNICODE TEXT SEGMENTATION specifies how to
handle grapheme clusters (extended grapheme cluster).
But some specs aren't updated to current situation because Unicode Emoji
is rapidly extended without well definition.
It breaks the precondition of UTR#29 "Grapheme cluster boundaries can be
easily tested by looking at immediately adjacent characters". (the
sentence will be removed in the next version)
Though some of its detail are described in Unicode Technical Report #51
UNICODE EMOJI but it is not merged into UTR#29 yet.

http://unicode.org/reports/tr29/
http://unicode.org/reports/tr51/
http://unicode.org/Public/emoji/4.0/

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@56949 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision c11e6487 - 11/30/2016 05:29 PM - naruse (Yui NARUSE)

Regexp supports Unicoe 9.0.0's \X

- meta character \X matches Unicode 9.0.0 characters with some workarounds
  for UTR #51 Unicode Emoji, Version 4.0 emoji zwj sequences.
  [Feature #12831] [ruby-core:77586]

The term "character" can have many meanings bytes, codepoints, combined
characters, and so on. "grapheme cluster" is highest one of such words,
which means user-perceived characters.
Unicode Standard Annex #29 UNICODE TEXT SEGMENTATION specifies how to
handle grapheme clusters (extended grapheme cluster).
But some specs aren't updated to current situation because Unicode Emoji
is rapidly extended without well definition.
It breaks the precondition of UTR#29 "Grapheme cluster boundaries can be
easily tested by looking at immediately adjacent characters". (the
sentence will be removed in the next version)
Though some of its detail are described in Unicode Technical Report #51
UNICODE EMOJI but it is not merged into UTR#29 yet.

http://unicode.org/reports/tr29/
http://unicode.org/reports/tr51/
http://unicode.org/Public/emoji/4.0/

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@56949 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 11/25/2016 06:37 AM - shyouhei (Shyouhei Urabe)

- Assignee set to naruse (Yui NARUSE)

- Tracker changed from Bug to Feature

- Status changed from Open to Assigned

#2 - 11/25/2016 01:18 PM - shyouhei (Shyouhei Urabe)

We looked at this issue in todays developer meeting and assigned it to Yui. But no one there had implementation of this. It might need some time.
Stay tuned!

#3 - 11/30/2016 05:29 PM - naruse (Yui NARUSE)

- Status changed from Assigned to Closed

Applied in changeset r56949.

Regexp supports Unicoe 9.0.0's \X

- meta character \X matches Unicode 9.0.0 characters with some workarounds
  for UTR #51 Unicode Emoji, Version 4.0 emoji zwj sequences.
  [Feature #12831] [ruby-core:77586]

The term "character" can have many meanings bytes, codepoints, combined
characters, and so on. "grapheme cluster" is highest one of such words,
which means user-perceived characters.
Unicode Standard Annex #29 UNICODE TEXT SEGMENTATION specifies how to
handle grapheme clusters (extended grapheme cluster).
But some specs aren't updated to current situation because Unicode Emoji
is rapidly extended without well definition.
It breaks the precondition of UTR#29 "Grapheme cluster boundaries can be
easily tested by looking at immediately adjacent characters". (the
sentence will be removed in the next version)
Though some of its detail are described in Unicode Technical Report #51
UNICODE EMOJI but it is not merged into UTR#29 yet.

http://unicode.org/reports/tr29/
http://unicode.org/reports/tr51/
http://unicode.org/Public/emoji/4.0/