Ruby - Feature #1106

Script encoding vs. default_internal: Implicitly transcode strings/regexps

02/04/2009 06:22 PM - tomel (Tom Link)

Status:	Rejected
Priority:	Normal
Assignee:	
Target version:	

Description

=begin
If I'm not mistaken, a related issue was discussed in the past (eg [1]). Anyway, please take a sec and consider the following scripts and input files:

FILE: test2.rb:

encoding: UTF-8

Encoding.default_internal = Encoding::UTF_8
Encoding.default_external = Encoding::UTF_8
require 'test2a'
File.readlines('test2.txt').each do |line|
p line, test2a(line)
end

FILE: test2a.rb

encoding: ISO-8859-1

p **ENCODING**
def test2a(x)
x =~ /[äöüÄÖÜß]/
end

FILE: test.txt (uft8 byte sequences; the second line should read "weiß", the third one "Bär" in UTF-8 encoding)
foo
weiÃŸ
BÃ¤r
bar

If I run

$ ruby -v
ruby 1.9.1p0 (2009-01-30 revision 21907) [i386-cygwin]
$ ruby test2.rb
#Encoding:ISO-8859-1
"foo\n"
nil
/home/t/src/tmp/test2a.rb:6:in test2a': invalid byte sequence in UTF-8 (ArgumentError) from test2.rb:9:in block in '
from test2.rb:8:in each' from test2.rb:8:in '

It seems the ISO-8859-1 encoded regexp in test2a.rb /[äöüÄÖÜß]/, is not transcoded to UTF-8. But since default_internal is set to UFT-8, ruby seems to expect a valid UTF-8 string. Please forgive me if my interpretation of that error message is wrong. It is quite possible that I missed something and that there already exists an easy solution to this problem, which I don't know of. If that is the case, I kindly ask you to tell me about it.

If this is the way, ruby 1.9.1 currently is supposed to work, I would humbly suggest to silently transcode all strings found in scripts to default_internal if non-nil. IMHO not transcoding strings doesn't make any sense and drives users who work with heterogeneous files to madness. If a string cannot be transcoded to default_internal, an error should be raised. Thanks.

[1] http://groups.google.com/group/ruby-core-google/browse_frm/thread/d6474429dd112926?hl=en
=end

History

#1 - 02/04/2009 06:53 PM - naruse (Yui NARUSE)

- Category set to M17N

=begin
Unicode codepoint is compatible with ISO-8859-1, converting regexp is not harmful.
But for example in Windows-1252, /[~-€]/ matches only Tilde and DEL and Euro Sign.
But after converted to UTF-8, /[~-€]/ means /[\u007E-\u20AC]/, so this matches many characters.

So converting literal regexp is harmful and this is why Ruby 1.9 doesn't convert literal encoding.
=end

#2 - 02/05/2009 08:07 AM - mike (Michael Selig)

=begin
There were a number of quite long discussions about String (and Regexp) literal encodings (as well as other encoding compatibility issues) last year. The decisions (as I recall) were:

- There should be NO silent transcoding in Ruby. This includes literals, and also when operating on strings with different encodings. The rationale was that transcoding was not always available, and was not always safe.
- If a program uses "default_internal" (which is likely to be UTF-8 in most circumstances) it is up to the programmer to make sure that string literals are in the default_internal encoding. This usually means that all the source file encodings need to be in the default_internal encoding (or are ASCII).
- Libraries being called by programs which may or may not use default_internal need to be careful about their use of String literals. ASCII literals are safe as long as both the source & default_internal encodings are both ASCII compatible (ruling out say UTF-16 and UTF-32). There was a discussion about whether default_internal and soure file encoding should be forced to be ASCII compatible. I am not sure what the outcome was.
  =end

#3 - 02/05/2009 05:18 PM - tomel (Tom Link)

=begin
Thanks for the summary. I'm not really sure what problem the current solution is supposed to solve but it's not mine to question it.

IMHO ruby should inform users about such conflicts or simply throw an error. Currently, ruby --verbose shows a warning message telling users that default_(in|ex)ternal was set. This message could be more informative. Also, it doesn't cover cases where default_internal wasn't set but two scripts have different encodings != ASCII.

I think additional warning messages should be displayed when a source file's encoding doesn't match default_internal or if the script encodings are heterogenous (number of encodings != ASCII >= 2).

=end

#4 - 10/11/2009 01:34 AM - naruse (Yui NARUSE)

- Status changed from Open to Rejected

=begin
script encoding affects only literals in the script.
I don't think that mixing different source is a problem.
=end