Ruby master - Feature #12034

RegExp does not respect file encoding directive

01/29/2016 03:52 PM - vo.x (Vit Ondruch)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

```
$ cat regexp-encoding.rb
# -*- encoding: binary -*-
puts ''.encoding
puts //.encoding

$ ruby regexp-encoding.rb
ASCII-8BIT
US-ASCII
```

The RegExp should have ASCII-8BIT encoding IMO.

Actually there is something different how Ruby 2.3 behaves with regards to encoding, since I cannot compile raindrops gem with Ruby 2.3 anymore due to this test error:

```
 1) Error:
TestLinux#test_unix_resolves_symlinks:
RegexpError: /.../n has a non escaped non ASCII character in non ASCII-8BIT script
    /builddir/build/BUILD/rubygem-raindrops-0.13.0/usr/share/gems/gems/raindrops-0.13.0/lib/raindr
ops/linux.rb:57:in `unix_listener_stats'
    /builddir/build/BUILD/rubygem-raindrops-0.13.0/usr/share/gems/gems/raindrops-0.13.0/test/test_
linux.rb:97:in `test_unix_resolves_symlinks'
```

This is the line where it fails:

http://bogomips.org/raindrops.git/tree/lib/raindrops/linux.rb#n57

History

#1 - 01/30/2016 02:49 AM - nobu (Nobuyoshi Nakada)

That encoding has never changed since 1.9.
It seems because File.readlink and File.realpath return locale strings.

#2 - 02/02/2016 10:14 AM - naruse (Yui NARUSE)

- Tracker changed from Bug to Feature

This is considered as a spec now.

Anyway the change is very tiny.

```
diff --git a/re.c b/re.c
index 3f7d227..3619711 100644
--- a/re.c
+++ b/re.c
@@ -2558,9 +2558,6 @@ rb_reg_initialize(VALUE obj, const char *s, long len, rb_encoding *enc,
             enc = fixed_enc;
         }
     }
-    else if (!(options & ARG_ENCODING_FIXED)) {
-        enc = rb_usascii_encoding();
-    }

     rb_enc_associate((VALUE)re, enc);
     if ((options & ARG_ENCODING_FIXED) || fixed_enc) {
```

#3 - 02/02/2016 06:51 PM - normalperson (Eric Wong)

nobu@ruby-lang.org wrote:

> That encoding has never changed since 1.9.
> It seems because File.readlink and File.realpath return locale strings.

Ugh, that isn't right to me since filesystem names (on *nix) can have
any byte besides "\0".

Anyways, workaround for raindrops:
http://bogomips.org/raindrops-public/20160202183136.21549-1-e%4080x24.org/raw

#4 - 02/07/2016 06:41 AM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> nobu@ruby-lang.org wrote:
>
> > That encoding has never changed since 1.9.
> > It seems because File.readlink and File.realpath return locale strings.
>
> Ugh, that isn't right to me since filesystem names (on *nix) can have
> any byte besides "\0".

How about fall back to ASCII-8BIT if we detect broken code range?

We try to be helpful by respecting FS encoding, but we need to
acknowledge symlinks can have any byte value from 1-0xFF

http://80x24.org/spew/20160207063040.31341-1-e%4080x24.org/raw

```
Subject: [PATCH v2] file.c (rb_file_s_readlink): do not set invalid encoding

With the exception of '\0', POSIX allows arbitrary bytes in a
symlink.  So we should not assume rb_filesystem_encoding()
is correct, and fall back to ASCII-8BIT if we detect strange
characters.

* file.c (rb_file_s_readlink): fall back to ASCII-8BIT
* test/ruby/test_file_exhaustive.rb (test_readlink_binary): add
  [ruby-core:73582] [Bug #12034]
---
 file.c                            | 10 +++++++++-
 test/ruby/test_file_exhaustive.rb | 16 ++++++++++++++++
 2 files changed, 25 insertions(+), 1 deletion(-)

diff --git a/file.c b/file.c
index 9f430a3..f880411 100644
--- a/file.c
+++ b/file.c
@@ -2768,7 +2768,15 @@ rb_file_s_symlink(VALUE klass, VALUE from, VALUE to)
 static VALUE
 rb_file_s_readlink(VALUE klass, VALUE path)
 {
-    return rb_readlink(path, rb_filesystem_encoding());
+    VALUE str = rb_readlink(path, rb_filesystem_encoding());
+    int cr = rb_enc_str_coderange(str);
+
+    /* POSIX allows arbitrary bytes with the exception of '\0' */
+    if (cr == ENC_CODERANGE_BROKEN) {
+   rb_enc_associate(str, rb_ascii8bit_encoding());
+    }
+
+    return str;
 }

 #ifndef _WIN32
diff --git a/test/ruby/test_file_exhaustive.rb b/test/ruby/test_file_exhaustive.rb
index 53b867e..730000b 100644
--- a/test/ruby/test_file_exhaustive.rb
+++ b/test/ruby/test_file_exhaustive.rb
```

```
@@ -549,6 +549,22 @@ def test_readlink
   rescue NotImplementedError
   end

+  def test_readlink_binary
+    return unless symlinkfile
+    bug12034 = '[ruby-core:73582] [Bug #12034]'
+    Dir.mktmpdir('rubytest-file-readlink') do |tmpdir|
+      Dir.chdir(tmpdir) do
+        link = "\xde\xad\xbe\xef".b
+        File.symlink(link, 'foo')
+        str = File.readlink('foo')
+        assert_predicate str, :valid_encoding?, bug12034
+        assert_equal link, str, bug12034
+      end
+    end
+  rescue NotImplementedError => e
+    skip "#{e.message} (#{e.class})"
+  end
+
   def test_readlink_long_path
     return unless symlinkfile
     bug9157 = '[ruby-core:58592] [Bug #9157]'
```

#5 - 02/07/2016 09:51 AM - nobu (Nobuyoshi Nakada)

Eric Wong wrote:

> How about fall back to ASCII-8BIT if we detect broken code range?

It may be desirable or undesirable, as it can cause unexpected failure later.

```
+        link = "\xde\xad\xbe\xef".b
+        File.symlink(link, 'foo')
+        str = File.readlink('foo')
+        assert_predicate str, :valid_encoding?, bug12034
+        assert_equal link, str, bug12034
```

Anyway, "\xde\xad\xbe\xef" is a valid string in some encodings, e.g., EUC-JP, ISO-8859-1, and so on.
Especially in ISO-8859 encodings, any bytes are valid.

#6 - 02/07/2016 10:41 PM - normalperson (Eric Wong)

nobu@ruby-lang.org wrote:

> Eric Wong wrote:
>
> > How about fall back to ASCII-8BIT if we detect broken code range?
>
> It may be desirable or undesirable, as it can cause unexpected failure later.

Current behavior causes failures now.

```
+        link = "\xde\xad\xbe\xef".b
+        File.symlink(link, 'foo')
+        str = File.readlink('foo')
+        assert_predicate str, :valid_encoding?, bug12034
+        assert_equal link, str, bug12034
```

> Anyway, "\xde\xad\xbe\xef" is a valid string in some encodings, e.g., EUC-JP, ISO-8859-1, and so on.
> Especially in ISO-8859 encodings, any bytes are valid.

I think that is fine as long as the strings are valid.
Returning invalid strings is the main problem, I think;
and we should stop doing that. Dir.entries and similar methods
have the same problem.

#7 - 02/07/2016 11:31 PM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> Returning invalid strings is the main problem, I think;
> and we should stop doing that. Dir.entries and similar methods
> have the same problem.

Maybe this, too:

```
Subject: [PATCH] string.c (rb_external_str_with_enc): fall back to ASCII-8BIT

Fall back to returning ASCII-8BIT instead of returning invalid
strings for things like Dir.entries.
---
 string.c                    | 4 ++++
 test/ruby/test_dir_m17n.rb | 3 ++-
 2 files changed, 6 insertions(+), 1 deletion(-)

diff --git a/string.c b/string.c
index e4a02eb..e390dfc 100644
--- a/string.c
+++ b/string.c
@@ -958,6 +958,10 @@ rb_external_str_with_enc(VALUE str, rb_encoding *eenc)
     return str;
      }
      rb_enc_associate(str, eenc);
+     if (rb_enc_str_coderange(str) == ENC_CODERANGE_BROKEN) {
+    rb_enc_associate(str, rb_ascii8bit_encoding());
+    return str;
+     }
      return rb_str_conv_enc(str, eenc, rb_default_internal_encoding());
 }

diff --git a/test/ruby/test_dir_m17n.rb b/test/ruby/test_dir_m17n.rb
index febfbc0..db5ac58 100644
--- a/test/ruby/test_dir_m17n.rb
+++ b/test/ruby/test_dir_m17n.rb
@@ -72,7 +72,8 @@ def test_filename_extutf8_invalid
         opts = {:encoding => Encoding.default_external} if /mswin|mingw/ =~ RUBY_PLATFORM
         ents = Dir.entries(".", opts)
         filename = "%FF" if /darwin/ =~ RUBY_PLATFORM && ents.include?("%FF")
-        assert_include(ents, filename)
+        assert_include(ents, filename.b)
+        ents.each { |f| assert_predicate f, :valid_encoding? }
       EOS
     }
   end unless /mswin|mingw/ =~ RUBY_PLATFORM
```

--
http://80x24.org/spew/20160207232116.15467-1-e%4080x24.org/raw

#8 - 02/09/2016 04:31 AM - nobu (Nobuyoshi Nakada)

- File 0001-string.c-rb_external_str_with_enc-fall-back-to-ASCII.patch added

- File 0002-follow-up-for-OS-X.patch added

It failed on OS X.

#9 - 02/09/2016 08:41 AM - normalperson (Eric Wong)

nobu@ruby-lang.org wrote:

> File 0001-string.c-rb_external_str_with_enc-fall-back-to-ASCII.patch added
> File 0002-follow-up-for-OS-X.patch added
>
> It failed on OS X.

So 0002 fixes things on OS X? Can you commit if you agree with this series?
I won't be around much the next few days.

#10 - 02/12/2016 09:03 PM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> nobu@ruby-lang.org wrote:
>
> > File 0001-string.c-rb_external_str_with_enc-fall-back-to-ASCII.patch added
> > File 0002-follow-up-for-OS-X.patch added
> >
> > It failed on OS X.
>
> So 0002 fixes things on OS X? Can you commit if you agree with this series?
> I won't be around much the next few days.

I will be around this weekend :) Shall I commit these patches?

#11 - 02/13/2016 06:41 PM - naruse (Yui NARUSE)

Eric Wong wrote:

> I think that is fine as long as the strings are valid.
> Returning invalid strings is the main problem, I think;
> and we should stop doing that. Dir.entries and similar methods
> have the same problem.
>
> How about fall back to ASCII-8BIT if we detect broken code range?

How should Ruby treat invalid paths is difficult problem.
Once I decided it is filesystem encoding but I agree to change if another encoding is practically better.

In this case both filesystem encoding and ASCII-8BIT won't work because it will raise Encoding::CompatibilityError
on paths.join even if it returns ASCII-8BIT strings instead of invalid filesystem encoding strings.

As an another use case to simply show filenames, retrieving filenames including invalid strings,
and call String#scrub works fine.

Therefore at this time I don't think changing into ASCII-8BIT isn't good thing.

Files

0001-string.c-rb_external_str_with_enc-fall-back-to-ASCII.patch	1.47 KB	02/09/2016	nobu (Nobuyoshi Nakada)
0002-follow-up-for-OS-X.patch	1.52 KB	02/09/2016	nobu (Nobuyoshi Nakada)