Ruby trunk - Bug #14586

URI::RFC2396_Parser#unescape raises an exception if the input is mixed Unicode and percent-escapes

03/08/2018 01:55 AM - kivikakk (Ashe Connor)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.6.0dev (2018-03-07 trunk 62693) [x86_64-darwin17]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

Currently, the following test case passes:

```
def test_unescape
  p1 = URI::Parser.new
  assert_equal("\xe3\x83\x90", p1.unescape("\xe3\x83\x90"))
  assert_equal("\xe3\x83\x90", p1.unescape('%e3%83%90'))
end
```

But the following raises Encoding::CompatibilityError:

```
def test_unescape
  p1 = URI::Parser.new
  assert_equal("\xe3\x83\x90", p1.unescape("\xe3\x83\x90"))
  assert_equal("\xe3\x83\x90", p1.unescape('%e3%83%90'))
  assert_equal("\xe3\x83\x90\xe3\x83\x90", p1.unescape("\xe3\x83\x90%e3%83%90"))
end
```

The issue is in the definition of URI::RFC2396_Parser#unescape:

```
def unescape(str, escaped = @regexp[:ESCAPED])
  str.gsub(escaped) { [$&[1, 2].hex].pack('C') }.force_encoding(str.encoding)
end
```

The behaviour is as follows:

- If the String contains only ASCII characters (including percent-escapes), then substituting each result of [$&[1, 2].hex].pack('C') (which returns ASCII-8BIT) succeeds, because the String so far is safely coerced to ASCII-8BIT to let the concatenation work.
- If the String contains only ASCII + Unicode characters (and no percent-escapes), then gsub matches nothing.
- If the String contains both, however, then attempting to gsub individual ASCII-8BIT characters (which can't be coerced to UTF-8) fails with Encoding::CompatibilityError.

This patch:

1. Adds the test.
2. Records the original encoding of the input string, forces the encoding to ASCII-8BIT for the gsub, then forces the encoding back after gsub. If the percent-encoded characters aren't valid in the original encoding, that's up to the user, but this is better than just refusing to perform the unescape at all.
3. Corrects a minor doc mismatch.

Thanks to tenderlovemaking (Aaron Patterson) for helping me find this in upstream Ruby, who suggested naruse (Yui NARUSE) might be interested in reviewing this patch. We currently run with this monkey-patched at GitHub, and in Rails master.

Associated revisions

Revision 6db869c1 - 03/08/2018 09:38 AM - naruse (Yui NARUSE)

fix error if the input is mixed Unicode and percent-escapes

Reported by kivikakk (Ashe Connor) with tests and doc fix
Patch based on mame and fix by naruse
[Bug #14586]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@62695 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 62695 - 03/08/2018 09:38 AM - naruse (Yui NARUSE)

fix error if the input is mixed Unicode and percent-escapes

Reported by kivikakk (Ashe Connor) with tests and doc fix
Patch based on mame and fix by naruse
[Bug #14586]

Revision 62695 - 03/08/2018 09:38 AM - naruse (Yui NARUSE)

fix error if the input is mixed Unicode and percent-escapes

Reported by kivikakk (Ashe Connor) with tests and doc fix
Patch based on mame and fix by naruse
[Bug #14586]

History

#1 - 03/08/2018 05:58 AM - mame (Yusuke Endoh)

I'm unfamiliar with encoding, so this is just my impression: str.dup.force_encoding(Encoding::ASCII_8BIT) always creates a string object, so str.gsub(re) { [$&[1, 2].hex].pack('C').force_encoding(str.encoding) } might be better?

(And, this is not directly related to this ticket, I'm unsure how the "escaped" optional parameter is helpful.)

#2 - 03/08/2018 09:38 AM - naruse (Yui NARUSE)

- Status changed from Open to Closed

Applied in changeset trunk|r62695.

---

fix error if the input is mixed Unicode and percent-escapes

Reported by kivikakk (Ashe Connor) with tests and doc fix
Patch based on mame and fix by naruse
[Bug #14586]

Files

rfc2396-uri-encoding.patch	1.48 KB	03/08/2018	kivikakk (Ashe Connor)