Ruby master - Feature #11577

Add encodeURIComponent compatible API for URI

10/09/2015 01:40 PM - naruse (Yui NARUSE)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

How about adding encodeURIComponent/decodeURIComponent compatible API?

There's already have some methods:

- URI.escape: context aware but deprecated.
- URIencode_www_form: application/x-www-form-urlencoded, which encodes spaces into '+'
- URIencode_www_form_component: above component

So it doesn't have non-form URI escape method.

```
diff --git a/lib/uri/common.rb b/lib/uri/common.rb
index 1444ae8..0017ae3 100644
--- a/lib/uri/common.rb
+++ b/lib/uri/common.rb
@@ -1,4 +1,5 @@
-#--
+#
+# -*- frozen-string-literal: true -*-
 # = uri/common.rb
 #
 # Author:: Akira Yamada <akira@ruby-lang.org>
@@ -329,27 +330,72 @@ module URI
     DEFAULT_PARSER.make_regexp(schemes)
   end

+  TBLENCURICOMP_ = {} # :nodoc:
   TBLENCWWWCOMP_ = {} # :nodoc:
-  256.times do |i|
-    TBLENCWWWCOMP_[i.chr] = '%%%02X' % i
-  end
-  TBLENCWWWCOMP_[' '] = '+'
-  TBLENCWWWCOMP_.freeze
+  TBLDECURICOMP_ = {} # :nodoc:
   TBLDECWWWCOMP_ = {} # :nodoc:
   256.times do |i|
     h, l = i>>4, i&15
-    TBLDECWWWCOMP_['%%%X%X' % [h, l]] = i.chr
-    TBLDECWWWCOMP_['%%%x%X' % [h, l]] = i.chr
-    TBLDECWWWCOMP_['%%%X%x' % [h, l]] = i.chr
-    TBLDECWWWCOMP_['%%%x%x' % [h, l]] = i.chr
+    c = i.chr.freeze
+    k = sprintf('%%%X%X', h, l).freeze
+    TBLENCURICOMP_[c] = TBLENCWWWCOMP_[c] = k
+    TBLDECURICOMP_[k] = TBLDECWWWCOMP_[k] = c
+    k = sprintf('%%%x%X', h, l).freeze
+    TBLDECURICOMP_[k] = TBLDECWWWCOMP_[k] = c
+    k = sprintf('%%%X%x', h, l).freeze
+    TBLDECURICOMP_[k] = TBLDECWWWCOMP_[k] = c
+    k = sprintf('%%%x%x', h, l).freeze
+    TBLDECURICOMP_[k] = TBLDECWWWCOMP_[k] = c
   end
+  TBLENCWWWCOMP_[' '] = '+'
+  TBLENCWWWCOMP_.freeze
+  TBLENCURICOMP_.freeze
```

```
   TBLDECWWWCOMP_['+'] = ' '
   TBLDECWWWCOMP_.freeze
+  TBLDECURICOMP_.freeze
+
+  # Encode given +str+ to URL-encoded form data.
+  #
+  # This method doesn't convert *, -, ., 0-9, A-Z, _, a-z, but does convert SP
+  # (ASCII space) to + and converts others to %XX.
+  #
+  # If +enc+ is given, convert +str+ to the encoding before percent encoding.
+  #
+  # This is an implementation of
+  # http://www.ecma-international.org/ecma-262/6.0/#sec-encodeuricomponent-uricomponent
+  #
+  # See URI.encode_www_form_component, URI.encode_www_form
+  def self.encode_component(str)
+    str = str.to_s.dup
+    if !str.ascii_only?
+      enc = str.encoding
+      if enc != Encoding::ASCII_8BIT && enc != Encoding::UTF_8
+        str.encode!(Encoding::UTF_8, invalid: :replace, undef: :replace)
+      end
+      str.force_encoding(Encoding::ASCII_8BIT)
+    end
+    str.gsub!(/[^\-_.!~*'()0-9A-Za-z]/, TBLENCURICOMP_)
+    str.force_encoding(Encoding::US_ASCII)
+  end
+
+  # Decode given +str+ of URL-encoded form data.
+  #
+  #
+  # This doesn't decodes + to SP.
+  #
+  # This is an implementation of
+  # http://www.ecma-international.org/ecma-262/6.0/#sec-decodeuricomponent-encodeduricomponent
+  #
+  # See URI.decode_www_form_component, URI.decode_www_form
+  def self.decode_component(str, enc=Encoding::UTF_8)
+    raise ArgumentError, "invalid %-encoding (#{str})" if /%(?!\h\h)/ =~ str
+    str.b.gsub(/%\h\h/, TBLDECURICOMP_).force_encoding(enc)
+  end

   HTML5ASCIIINCOMPAT = defined? Encoding::UTF_7 ? [Encoding::UTF_7, Encoding::UTF_16BE, Encoding:
:UTF_16LE,
     Encoding::UTF_32BE, Encoding::UTF_32LE] : [] # :nodoc:

-  # Encode given +str+ to URL-encoded form data.
+  # Encode given +str+ in application/x-www-form-urlencoded format.
   #
   # This method doesn't convert *, -, ., 0-9, A-Z, _, a-z, but does convert SP
   # (ASCII space) to + and converts others to %XX.
@@ -373,7 +419,7 @@ module URI
     str.force_encoding(Encoding::US_ASCII)
   end

-  # Decode given +str+ of URL-encoded form data.
+  # Decode given +str+ in application/x-www-form-urlencoded format.
   #
   # This decodes + to SP.
   #
@@ -457,7 +503,7 @@ module URI
       if isindex
         if sep.empty?
           val = key
-          key = ''
+          key = String.new
         end
```

```
          isindex = false
        end
@@ -471,7 +517,7 @@ module URI
        if val
          val.gsub!(/\+|%\h\h/, TBLDECWWWCOMP_)
        else
-         val = ''
+         val = String.new
        end

        ary << [key, val]
diff --git a/test/uri/test_common.rb b/test/uri/test_common.rb
index 5620415..c6b5633 100644
--- a/test/uri/test_common.rb
+++ b/test/uri/test_common.rb
@@ -54,6 +54,34 @@ class TestCommon < Test::Unit::TestCase
     assert_raise(NoMethodError) { Object.new.URI("http://www.ruby-lang.org/") }
   end

+  def test_encode_component
+    assert_equal("%00%20!%22%23%24%25%26'()*%2B%2C-.%2F09%3A%3B%3C%3D%3E%3F%40" \
+                 "AZ%5B%5C%5D%5E_%60az%7B%7C%7D~",
+                 URI.encode_component("\x00 !\"\#$%&'()*+,-./09:;<=>?@AZ[\\]^_`az{|}~"))
+    assert_equal("%E6%9F%8A", URI.encode_component(
+                   "\x95\x41".force_encoding(Encoding::Shift_JIS)))
+    assert_equal("%E3%81%82", URI.encode_component(
+                   "\x30\x42".force_encoding(Encoding::UTF_16BE)))
+    assert_equal("%E3%81%82", URI.encode_component(
+                   "\e$B$\"\e(B".force_encoding(Encoding::ISO_2022_JP)))
+  end
+
+  def test_decode_component
+    assert_equal(" +!\"\#$%&'()*+,-./09:;<=>?@AZ[\\]^_`az{|}~",
+                 URI.decode_component(
+                   "%20+%21%22%23%24%25%26%27%28%29*%2B%2C-.%2F09%3A%3B%3C%3D%3E%3F%40" \
+                   "AZ%5B%5C%5D%5E_%60az%7B%7C%7D%7E"))
+    assert_equal("\xA1\xA2".force_encoding(Encoding::EUC_JP),
+                 URI.decode_component("%A1%A2", "EUC-JP"))
+    assert_equal("\xE3\x81\x82\xE3\x81\x82".force_encoding("UTF-8"),
+                 URI.decode_component("\xE3\x81\x82%E3%81%82".force_encoding("UTF-8")))
+
+    assert_raise(ArgumentError){URI.decode_component("%")}
+    assert_raise(ArgumentError){URI.decode_component("%a")}
+    assert_raise(ArgumentError){URI.decode_component("x%a_")}
+    assert_nothing_raised(ArgumentError){URI.decode_component("x"*(1024*1024))}
+  end
+
   def test_encode_www_form_component
     assert_equal("%00+%21%22%23%24%25%26%27%28%29*%2B%2C-.%2F09%3A%3B%3C%3D%3E%3F%40" \
                  "AZ%5B%5C%5D%5E_%60az%7B%7C%7D%7E",
```