Class Encoding < Object

1.9

An encoding describes how to map the binary data in the internal representation of strings into characters. Ruby has support for a large number of encodings built in—others can be loaded dynamically at runtime.

Encodings are identified by name (*UTF-** or *ISO-8859-1*, for example). They are represented by encoding objects. The Encoding class contains predefined constants for these encoding objects. Often there are multiple objects for the same encoding. For example, the constants Encoding::IBM860 and Encoding::CP860 are both representations of the encoding named *IBM860*. In Table 27.2 on the following page, we can see the names of the encodings are in the first column and the names on the constants in the Encoding class for the corresponding encoding object(s). An entry such as ISO-8859-1 – 11 indicates that there are 11 separate encodings (with the obvious names).

Encodings are used when opening files, creating strings, and so on. The methods that accept an encoding as a parameter will take either an encoding name or an encoding object. Use of the object is marginally faster.

Chapter 17 on page 264 is devoted to a discussion of encodings.

Class methods

aliases

Encoding.aliases → *hash*

Returns a hash whose keys are aliases for encodings and whose values are the corresponding base encoding names.

```
Encoding.aliases["BINARY"]   # =>   "ASCII-8BIT"
```

compatible?

Encoding.compatible?(*str1*, *str2*) → *enc* or nil

Determines whether two strings have compatible encodings (meaning, for example, that you could concatenate them). Returns the encoding of the string that would result from the concatenation or nil if the strings are not compatible.

```
# encoding: utf-8
ascii1 = "ant"
ascii2 = "bee"
iso    = "\xee"
iso.force_encoding(Encoding::ISO_8859_1)
utf    = "δog"

Encoding.compatible?(ascii1, ascii2)   # =>   #<Encoding:UTF-8>
Encoding.compatible?(ascii1, iso)      # =>   #<Encoding:ISO-8859-1>
Encoding.compatible?(ascii1, utf)      # =>   #<Encoding:UTF-8>
Encoding.compatible?(iso,    utf)      # =>   nil
```

default_external

Encoding.default_external → *enc*

Returns the default external encoding, used when reading and writing data from I/O streams.

```
Encoding.default_external   # =>   #<Encoding:UTF-8>
```

Table 27.2. Encoding Names and Class Names

Encoding name	Encoding::*xxx* class name(s)
ASCII-8BIT	ASCII_8BIT, BINARY
Big5	Big5, BIG5, CP950
CP51932	CP51932
CP850	CP850, IBM850
CP852	CP852
CP855	CP855
CP949	CP949
Emacs-Mule	Emacs_Mule, EMACS_MULE
EUC-JP	EUC_JP, EucJP, EUCJP
EUC-KR	EUC_KR, EucKR, EUCKR
EUC-TW	EUC_TW, EucTW, EUCTW
eucJP-ms	EucJP_ms, EUCJP_MS, EUC_JP_MS
GB12345	GB12345
GB18030	GB18030
GB1988	GB1988
GB2312	EUC_CN, EucCN, EUCCN
GBK	GBK, CP936
IBM437	IBM437, CP437
IBM737	IBM737, CP737
IBM775	IBM775, CP775
IBM852	IBM852
IBM855	IBM855
IBM857	IBM857, CP857
IBM860 – 6	IBM860 – 6, CP8600 – 6
IBM869	IBM869, CP869
ISO-2022-JP	ISO_2022_JP, ISO2022_JP
ISO-2022-JP-2	ISO_2022_JP_2, ISO2022_JP2
ISO-8859-1 – 11	ISO8859_1 – 11
ISO-8859-13 – 16	ISO8859_13 – 16
KOI8-R	KOI8_R, CP878
KOI8-U	KOI8_U
macCentEuro	MacCentEuro, MACCENTEURO
macCroatian	MacCroatian, MACCROATIAN
macCyrillic	MacCyrillic, MACCYRILLIC
macGreek	MacGreek, MACGREEK
macIceland	MacIceland, MACICELAND
MacJapanese	MacJapanese, MACJAPANESE, MacJapan, MACJAPAN
macRoman	MacRoman, MACROMAN
macRomania	MacRomania, MACROMANIA
macThai	MacThai, MACTHAI
macTurkish	MacTurkish, MACTURKISH
macUkraine	MacUkraine, MACUKRAINE
Shift_JIS	Shift_JIS, SHIFT_JIS, SJIS
stateless-ISO-2022-JP	Stateless_ISO_2022_JP, STATELESS_ISO_2022_JP
TIS-620	TIS_620
US-ASCII	US_ASCII, ASCII, ANSI_X3_4_1968
UTF-16BE	UTF_16BE, UCS_2BE
UTF-16LE	UTF_16LE
UTF-32BE	UTF_32BE, UCS_4BE
UTF-32LE	UTF_32LE, UCS_4LE
UTF-7	UTF_7, CP65000
UTF-8	UTF_8, CP65001
UTF8-MAC	UTF8_MAC, UTF_8_MAC
Windows-1250 – 1258	Windows_1250 – 1258, WINDOWS_1250 – 1258, CP1250 – 1258
Windows-31J	Windows_31J, WINDOWS_31J, CP932, CsWindows31J, CSWINDOWS31J
Windows-874	Windows_874, WINDOWS_874, CP874

default_external= Encoding.default_external = *enc*

Sets the default external encoding.

default_internal Encoding.default_internal → *enc* or nil

Returns the default internal encoding, used when transcoding data read and written. Returns nil if no default encoding is set.

default_internal= Encoding.default_internal = *enc*

Sets the default internal encoding.

```
Encoding.default_internal = 'utf-8'
Encoding.default_internal   # =>   #<Encoding:UTF-8>
```

find Encoding.find(*name*) → *enc*

Returns the encoding object for the given encoding name or throws an ArgumentError.

```
Encoding.find("Shift_JIS")   # =>   #<Encoding:Shift_JIS>
```

list Encoding.list → *array*

Returns a list of the encoding objects loaded into the current interpreter.

locale_charmap Encoding.locale_charmap → *name*

Returns the name of the charmap of the current locale. This is normally set externally, often in an environment variable or other operating-system context.

```
ENV["LANG"]               # =>   "en_US.UTF-8"
Encoding.locale_charmap   # =>   "UTF-8"
```

name_list Encoding.name_list → *array*

Returns a list of the names of loaded encodings.

```
Encoding.name_list.sort.first(5)   # =>   ["646", "ANSI_X3.4-1968",
                                          "ASCII", "ASCII-8BIT", "BINARY"]
```

Instance methods

dummy? *enc*.dummy? → true or false

Dummy encodings are placeholders for encodings that cannot be handled properly by the current mechanism of Ruby M17N, often because they are stateful.

```
Encoding::UTF_7.dummy?   # =>   true
Encoding::UTF_8.dummy?   # =>   false
```

name *enc*.name → *string*

Returns the name of *enc*.

```
Encoding::UTF_8.name     # =>   "UTF-8"
Encoding::CP65001.name   # =>   "UTF-8"
```

names *enc*.names → [⟨ *string* ⟩+]

Returns the name of *enc*, along with the names of *enc*'s aliases.

```
Encoding::UTF_8.names     # => ["UTF-8", "CP65001", "locale",
                                "external"]
Encoding::CP65001.names   # => ["UTF-8", "CP65001", "locale",
                                "external"]
Encoding::ASCII.names     # => ["US-ASCII", "ASCII", "ANSI_X3.4-1968",
                                "646"]
```