Package 'Nippon'

May 16, 2018

Type Package

Title Japanese Utility Functions and Data

Version 0.7.1

Date 2018-05-16

Author Susumu Tanimura <aruminat@gmail.com> [aut, cre],
Hironobu Takahashi [cph],
Hajime Baba [cph],
Takatsugu Nokubi [cph]

Maintainer Susumu Tanimura <aruminat@gmail.com>

Depends stringr

Description Japan-specific data is sometimes too unhandy for R users to manage. The utility functions and data in this package disencumber us from such an unnecessary burden.

License GPL (>= 2)

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-05-16 12:45:30 UTC

R topics documented:

`Nippon-package` *Japanese Utility Functions and Data*

Description

Japan-specific data is sometimes too unhandy for R users to manage. The utility functions and data in this package disencumber us from such an unnecessary burden.

Details

The DESCRIPTION file:

Package:	Nippon
Type:	Package
Title:	Japanese Utility Functions and Data
Version:	0.7.1
Date:	2018-05-16
Author:	Susumu Tanimura <aruminat@gmail.com> [aut, cre], Hironobu Takahashi [cph], Hajime Baba [cph], Ta
Maintainer:	Susumu Tanimura <aruminat@gmail.com>
Depends:	stringr
Description:	Japan-specific data is sometimes too unhandy for R users to manage. The utility functions and data in th
License:	GPL (>= 2)
Encoding:	UTF-8
NeedsCompilation:	yes
Packaged:	2017-11-23 01:21:29
Repository:	CRAN
Date/Publication:	2017-11-23 01:24:13

Index of help topics:

```
JapaneseColors          Find RGB by Japanese color names
Nippon-deprecated       Deprecated function(s) in the Nippon package
Nippon-internal         Internal objects in the Nippon package
Nippon-package          Japanese Utility Functions and Data
jholiday                Calculate public holidays in Japan
jyear                   Conversion to Japanese year style
kakasi                  Interface to kakasi
kana2roma               Romanization of Japanese
kansu2arabic            Conversion form Kasuji (Chinese/Japanese
```

	numerals) to arabic forms
kata2hira	Conversion form Katakana to Hiragana
month.name.jp	The Japanese name of months
nippon.palette	Switch the color palette to JIS colors
prefectures	Prefectural names in Japan
sanitizeZenkaku	Sanitizing strings contaminated with fullwidth (zenkaku) charactors.
sjis2utf8	Wrapper of iconv for Japanese encoding
wareki2AD	Conversion from Japanese imperial year to Anno Domini
zen2han	Convert Japanese characters from fullwidth (zenkaku) to halfwidth (hankaku) forms, and vice versa.

To avoid troubles with Japanese strings, non-ASCII Japanese characters in R object is sanitized or converted into ASCII characters by utility functions in this package. Some common data for Japanese are planned to preset, for preventing from burdensome input. (Those will be reinforced in further version).

Author(s)

Susumu Tanimura <aruminat@gmail.com> [aut, cre], Hironobu Takahashi [cph], Hajime Baba [cph], Takatsugu Nokubi [cph]

Maintainer: Susumu Tanimura <aruminat@gmail.com>

`JapaneseColors` *Find RGB by Japanese color names*

Description

`JapaneseColors` returns RGB values from Japanese traditional color names, which are defined by Japanese Industrial Standard (JIS).

Usage

```
JapaneseColors(names)
```

Arguments

`names`	A chracter vector. The JIS name of Japanese traditional colors can be written in UTF-8 encoded Japanese (Kanji, or Hiragana), or Romaji (ASCII).

Details

The JIS Common Color Names (JIS Z 8102:2001) is definition of 269 colors by JIS. `JapaneseColors` provides the RGB value in conformity to the JIS Standards, referring the Japanese traditional color name. Note that this function only supports the JIS colors with Japanese traditional names (145 colors), and does not support the JIS colors with English names (124 colors).

Value

A character vector

Author(s)

Susumu Tanimura

References

JIS Z 8102:2001 (Names of non-luminous object colours)

K. Seino and I. Shimamori. Shikimeijiten. Tokyo:Sinkigensha, 2005.

See Also

`nippon.palette`

Examples

```
JapaneseColors(c("sangoiro","kuriiro"))
```

---

jholiday *Calculate public holidays in Japan*

Description

This function is to calculate public holidays in Japan for given year.

Usage

```
jholiday(year, holiday.names = TRUE)
is.jholiday(dates)
```

Arguments

`year`	A integer value, formatted as YYYY. A year should be in anno Domini, and in and after 1949. Only single value is accepted.
`holiday.names`	logical. If FALSE, names of holiday are suppressed. The default value is TRUE.
`dates`	A date value or vector of dates

Details

The function `jholiday` returns Japanese public holidays of given year according to the Public Holiday Law of 1948. All legal reforms have so far been followed, but users should be careful about holidays in the future because of possible change in law. The function `is.jholiday` answers to whether or not given date is holiday.

Value

The function `jholiday` returns an object of `Date` class with or without holiday names. The function `is.jholiday` returns a logical vector.

Note

There are several other R functions to calculate holidays, including public holiday in Japan; however, none of functions works correctly due to very complicated holiday system in Japan, especially the Happy Monday System and the citizens' holiday rule. Only `jholiday` may work correctly. In case you get wrong results by the function `jholiday`, please report to the author.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

References

Public Holiday Law http://www8.cao.go.jp/chosei/shukujitsu/gaiyou.html

Ephemeris Computation Workshop (1991) Sinkoyomi-benricho, Koseisha Koseikaku: Tokyo, 1991, ISBN:9784769907008.

See Also

`holiday` in **timeDate** package,

`holidays` in **TimeWarp** package, `Holidays` in **Holidays** package.

Examples

```
jholiday(2013)
d <- as.Date(c("2000-09-22","2013-11-04", "1968-01-27"))
is.jholiday(d)
```

---

jyear *Conversion to Japanese year style*

Description

`jyear` calculates a year of the reign of an Emperor, i.e., "gengo", which is widely used in official statistics and documents in Japan.

Usage

```
jyear(x, shift = FALSE, withAD = FALSE, ascii = FALSE)
```

Arguments

x	numeric. a Cristian year. It must be greater than 1867.
shift	logical. This is for manual adjustment in a particular year. The default value is FALSE. See **Details** for more information.
withAD	logical. If TRUE, the intact Christian year is also given in output. The default value is FALSE.
ascii	logical. If TRUE, an abbreviation of gengo is used: "M", "T", "S", and "H". If FALSE, the Kanji characters of gengo is provided. The default value is FALSE.

Details

Japan uses era systems, and on each emperor's reign it would constitute one era. The Japanese traditional era name is widely used in official statistics and documents instead of the Anno Domini system. Recently, the year is often written in traditional form with Christian year, e.g., H12 (2000), because Japanese year style without Christian one is confusing even for Japanese.

`jyear` calculates such Japanese year from the given Christian year. Is may be useful if the Japanese style is required in statistical graphics or documents.

Since `jyear` dose not take account of month and day, an unexpected output could be given for a particular year, during which an era changed to a new one. The `shift` option should be set manually as TRUE in the following period.

from January 1 to July 29 in 1912
from January 1 to December 24 in 1926
from January 1 to 7 in 1989

Value

character

Author(s)

Susumu Tanimura <aruminat@gmail.com>

Examples

```
jyear(2000,ascii=TRUE)
jyear(2000,withAD=TRUE,ascii=TRUE)
jyear(1989,ascii=TRUE)
jyear(1989,shift=TRUE,ascii=TRUE)
```

`kakasi` *Interface to kakasi*

Description

The kakasi is an interface to the external program kakasi, KAnji KAna Simple Inverter. It is useful especially when Japanese Kanji characters are subject to convert to Romaji (ASCII) characters.

Usage

```
kakasi(x, kakasi.option="-Ha -Ka -Ja -Ea -ka",
 ITAIJIDICTPATH = Sys.getenv("ITAIJIDICTPATH", unset = NA),
 KANWADICTPATH = Sys.getenv("KANWADICTPATH", unset = NA))
```

Arguments

x	A character vector
kakasi.option	A chracter string specifying the options passed to kakasi library/program
ITAIJIDICTPATH	A character string specifying the path to itaijidict. Environmental variable of itaijidict passed to kakasi library.
KANWADICTPATH	A character string specifying the path to kanwadict. Environmental variable of kanwadict passed to kakasi library.

Details

Japanese strings are often made up a mixture of Chinese characters (Kanji), Kana (Hiragana and Katakana) and Romaji (Latin phonetical pronunciation). The external program kakasi converts between these four different ways of writing Japanese. kakasi and Sys.kakasi are useful especially for sanitizing a character vector by converting Japanese (non-ASCII) to ASCII characters.

kakasi uses two basic dictionaries: itaijidict and kanwadict. These dictionaries are included in doc/share of Package directory after installation of Nippon package. Since the kakasi library looks up the environmental variables to find dictionary, ITAIJIDICTPATH and KANWADICTPATH are internally set using Sys.setenv at the time when kakasi is called first time. After the first call, kakasi continues to use the environmental variables. Until R session closes, these environmental variables never unset. To use alternative dictionary instead of the bundled, a user can set the environmental variables using Sys.setenv or as arguments of kakasi. For permanent setting of environmental variables, see help of Renviron.

Value

A character vector

Warning

Note that non-Japanese and non-ASCII characters are not filtered in kakasi.kakasi warns unless LC_CTYPE is "ja_JP.UTF-8" (Linux or MacOSX) or "Japanese_Japan.932" (Windows). It is not sure whether the function is workable in other locale.

Note

`Sys.kakasi` was removed in Nippon ver.0.6.

`kakasi` warns unless LC_CTYPE is "ja_JP.UTF-8" (Linux or MacOSX) or "Japanese_Japan.932" (Windows).

The accuracy of Kanji-Kana conversion with kakasi is a bit lower than with MeCab program (http://mecab.sourceforge.net/). Although MeCab does not have a function of Kana-Romaji conversion, MeCab could be an option if you wish more accurate results. RMeCab is available from http://rmecab.jp/wiki/.

For Windows users, please be known that R on Windows can use strings encoded by both "ja_JP.UTF-8" and "Japanese_Japan.932"; however, `kakasi` works only with "Japanese_Japan.932". If you have data encoded with UTF-8 on Windows, you should convert it to "Japanese_Japan.932 (CP932)" as shown in example.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

References

KAKASI - Kanji Kana Simple Inverter http://kakasi.namazu.org/

Examples

```
## Not run:
library(Nippon)
data(prefectures)
regions <- unique(prefectures$region)
regions
# Unix-like operating systems
kakasi(regions)
# Windows
regions.cp932 <- iconv(regions, from = "UTF-8", to = "CP932")
kakasi(regions.cp932)

## End(Not run)
```

---

kansu2arabic *Conversion form Kasuji (Chinese/Japanese numerals) to arabic forms*

Description

A function to convert from Kasuji (Chinese/Japanese numerals) to arabic forms.

Usage

```
kansu2arabic(s)
kansuExample()
```

Arguments

s	A character vector composed of Kansu characters

Details

Some numeric data (e.g., year) in Japan is written with "Kansu" characters (or "Kansuji") instead of using Arabic numeral systems. "Kansu" is the traditional Japanese numeral systems adopting Chinese characters. To make such traditional numerals manipulatable, `kansu2arabic()` convert them to safe numeric forms (i.e., Arabic numerals). The variant of "Kansu" characters for financial is also acceptable in `kansu2arabic()`, though the function dose not accept any other characters than "Kansu" characters.

`kansuExample()` merely generates an example of Kansu character vector in order to illustrate how `kansu2arabic()` works. This way is a detour to follow the rule of CRAN that prohibits to write non-ASCII characters in package source files.

Value

A numeric vector

Note

`kansu2arabic()` accepts only Kansu characters. Any other characters cause errors.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

Examples

```
x <-  kansuExample()
x
kansu2arabic(x)
```

---

kata2hira *Conversion form Katakana to Hiragana*

Description

Functions to convert from Katakana to Hiragana, vise versa.

Usage

```
kata2hira(x)
hira2kata(x)
ya.kata2hira(x)
ya.hira2kata(x)
hiragana()
katakana()
```

Arguments

x	A character vector including Japanese Hiragana or Katakana

Details

kata2hira and ya.kata2hira converts from Katakana to Hiragana. hira2kata and ya.hira2kata converts from Hiragana to Katakana. hiragana and katakana generate Hiragana and Katakana, respectively, from the UTF-8 code table. It may be useful when users need Kana characters where no Kana input method is available.

Value

A character vector

Note

The difference between kata2hira and ya.kata2hira or between hira2kata and ya.hira2kata is in the algorithm, causing difference cacluatiion cost and output results.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

Examples

```
library(Nippon)
(kata <- katakana()[26:30])
kata2hira(kata)
(hira <- hiragana()[21:25])
hira2kata(hira)
```

---

month.name.jp *The Japanese name of months*

Description

The traditional names of months in Japanese.

Usage

```
data(month.name.jp)
```

Format

character vector

Details

month.name.jp is a constant including the Japanese name of months

Examples

```
data(month.name.jp)
month.name.jp[which(month.name=="April")]
```

`Nippon-deprecated` *Deprecated function(s) in the Nippon package*

Description

Some functions are defunct in the Nippon package. See Details.

Details

`JapanPrefecturesMap` is deprecated. Please use `JapanPrefMap` in **NipponMap** package, which is rewrite with more modern **sf** package. `JapanPrefecturesMap` in this package is defunct.

`Nippon-internal` *Internal objects in the Nippon package*

Description

Internal objects in the Nippon package, which are only user-visible because of the special nature of the Nippon name space.

Usage

```
zenkaku
jiscolors
jdate
jpn.syllabary
jpn.syllabary.add
```

Format

zenkaku is a list, and jiscolors is a data frame. jpn.syllabary and jpn.syllabary.add are character data frames.

Details

Internal objects are loaded coincided with loading Nippon package. `zenkaku` and `jiscolors` are used internally in `zen2han` and `JapaneseColors`, respectively. `jpn.syllabary` and `jpn.syllabary.add` internally provide conversion tables for `hira2kata`, `kata2hira`, and `kana2roma`.

`zenkaku` has lower and upper case of alphabets and numbers in fullwidth form. `jiscolors` is data of JIS colors, including Kanji, Hiragana, and Romaji names.

`jpn.syllabary` is a conversion table, including Hiragana, Katakana, three main systems for the romanization of Japanese: Hepburn, Nihon-shiki and Kunrei-shiki.

`jpn.syllabary.add` is an unofficial conversion table, but it is widely used especially in ICT.

`nippon.palette` *Switch the color palette to JIS colors*

Description

`nippon.palette` switches the color palette to the Japanese Industrial Standard (JIS) color palette, replacing with the corresponding color in the default palette.

Usage

```
nippon.palette()
```

Details

JIS common color names (JIS Z 8102:2001) were defined by JIS as 269 colors. These colors are different from usual color in computers. For example, red is #BE0032 in JIS color but #FF0000 in usual.

Value

an optional character vector

Note

Use 'palette("default")' to restore the default color palette.

Author(s)

Susumu Tanimura

References

JIS Z 8102:2001 (Names of non-luminous object colours) K. Seino and I. Shimamori. Shikimei-jiten. Tokyo:Sinkigensha, 2005.

Examples

```
op <- par(mfrow=c(1,2))
palette("default")
n <- print(palette())
pie(rep(1,8),col=1:8,label=n)
nippon.palette()
pie(rep(1,8),col=1:8,label=n)
palette("default")
par(op)
```

`prefectures` *Prefectural names in Japan*

Description

`prefectures` provides the name of prefectures in Japan as well as the regional name to be grouped.

Usage

```
data(prefectures)
```

Format

A data frame with 47 observations on the following 3 variables.

`jiscode` a character vector

`name` a character vector

`region` a character vector

Details

The prefectures of Japan are the country's 47 subnational jurisdictions. `prefectures` provides the names in JIS code (JIS X 0401) order as UTF-8 encoded Japanese Kanji.The regions of Japan are assigned to corresponding prefecture, even though such regional division of Japan are not official administrative units.

Note

If you need the prefectural name in ASCII, `kakasi` can help to convert Kanji to Romaji (ASCII). See the example showed below.

Examples

```
data(prefectures)
head(prefectures)
## Not run:
## Obtain the name as Romaji (ASCII)
kakasi(head(prefecture$name))

## End(Not run)
```

romanization *Romanization of Japanese*

Description

Japanese characters in a string or character vector are romanized with the their sounds for the English-speaking world. While kakasi in **Nippon** package works for romanization of Japanese, alternative romanization of Japanese is limitedly available with kana2roma. Unlike the kakasi function, kana2roma works without any help of an external library.

Usage

```
kana2roma(x, type = c("Hepburn", "Nippon.shiki", "Kunrei.shiki"),
    cap = FALSE, ascii.only = TRUE)
```

Arguments

x	A character vector including Japanese Hiragana or Katakana
type	A character string specifying the type of romanization. Default is "Hepburn"
cap	logical. Capital letters to be uppercased, Default is FALSE
ascii.only	logical. Transcribed with ASCII characters only. Default is TRUE

Details

Japanese strings are often made up a mixture of Chinese characters (Kanji), Kana (Hiragana and Katakana) and Romaji (Latin phonetical pronunciation). kana2roma transcribes Kana to Romaji without any help of external programs, such as kakasi. It should be useful especially when users want to sanitize and make readable Japanese strings in data set for the English-speaking world. The function supports three main romanization systems. Although the Nihon-shiki (ISO3602 Strict) is the official system in Japan, Hepburn is most widely used especially for proper noun, and officially adopted in naming systems for railway station and roads. A variant of Hepburn is authorized by the Japanese Foreign Ministry for use in passports.

For place names or other proper nouns, set "cap = TRUE" in kana2roma (default is FALSE) to capitalize the first letters in Romaji strings.

Set "ascii.only = TRUE" in kana2roma (this is default) if a user needs to suppress non-ASCII Romaji. Otherwise, a pure romanization system may return values with non-ASCII codes, that is, macron.

Value

A character vector

Note

kana2roma supports only Kana (Hiragana and Katakana). All other characters are just ignored and output as it is. If users need convert from Kanji to Romaji, use kakasi instead of kana2roma.

Rigidly, there are many variants of the three main romanization systems with small differences. Yet another romanization is used in an input methods engine of computers. Since the function strictly and simply follows the three romanization systems, some Kana characters may be failed due to lack of authorized conversion rules. Yet, some unsupported conversion rules will be implemented as optional in the future.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

See Also

See Also as kakasi.

Examples

```
library(Nippon)
jpn <- c(hiragana()[21:25], katakana()[26:30])
kana2roma(jpn)
```

---

sanitizeZenkaku *Sanitizing strings contaminated with fullwidth (zenkaku) charactors.*

Description

Sanitizing strings unintensionally contaminated with fullwidth (zenkaku) charactors by converting characters from fullwidth (zenkaku) to halfwidth (hankaku) forms.

Usage

```
sanitizeZenkaku(s)
```

Arguments

s	A character vector. UTF-8 encoding is preferable.

Details

Occasionally a character vector is unintentionally contaminated with fullwidth (zenkaku) characters. sanitizeZenkaku remove Japanese fullwidth (zenkaku) alphabets, numbers, and symbols from the given character vector in order to make logical and factor vectors work properly. The alphabets, numbers, and symbols are substitute for halfwidth forms (aka. ASCII), while a fullwidth space is simply removed.

Value

A character vector. All alphabets, numbers, and symbols have their halfwidth from.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

See Also

zen2han

Examples

```
(n <- intToUtf8(c(65296 + 1:3, 12288)))
sanitizeZenkaku(n)
```

---

sjis2utf8 *Wrapper of iconv for Japanese encoding*

Description

These functions are to encode Japanese characters from SJIS/JIS/EUC-JP to UTF-8.

Usage

```
sjis2utf8(x, CP932=TRUE)
eucjp2utf8(x)
jis2utf8(x)
```

Arguments

x	A character vecter
CP932	logical. If you like to use Shift-JIS instread of CP932, set CP932 = FALSE. The default is TRUE

Details

The major Japanese encoding systems are Shift-JIS (CP932), JIS (ISO-2022-JP), EUC-JP, and recently UTF-8. Exchanging Japanese strings data between the different platforms is often the cause of unreadable illegal characters. Since iconv could be the solution of this issues, these functions are written for the handy use of iconv, in partifular, when importing an old dataset or from the different platform.

Value

A character vector

Author(s)

Susumu Tanimura <aruminat@gmail.com>

See Also

iconv and localeToCharset.

Examples

```
x <- iconv(zenkaku$upper, from="UTF-8", to="CP932")
sjis2utf8(x)
```

---

wareki2AD *Conversion from Japanese imperial year to Anno Domini*

Description

wareki2AD calculates a year for Anno Domini from Japanese imperial year, which is widely used in official statistics and documents in Japan.

Usage

```
wareki2AD(year)
```

Arguments

year	vector of Japanese calender year as strings.

Details

year should include numeric strings as a halfwidth (hankaku) forms (aka ACSII). If you have fullwidth (zenkaku) figures in the numeric part of year, you need to convert them to halfwidth characters before using wareki2AD.

Noted that the supported the names of an era, "gengo", inlcude "Meiji", "Taisho", "Showa", and "Heisei" only.

Value

character

Author(s)

Susumu Tanimura <aruminat@gmail.com>

Examples

```
## Not run:
yr <- paste0(intToUtf8(c(24179,25104)), 20, intToUtf8(24180))
wareki2AD(yr)

## End(Not run)
```

---

zen2han *Convert Japanese characters from fullwidth (zenkaku) to halfwidth (hankaku) forms, and vice versa.*

Description

This function is to convert Japanese characters between fullwidth (zenkaku) and halfwidth (hankaku) forms for avoiding trouble in Japanese string operation or for taking advantage of fullwidth (zenkaku) forms.

Usage

```
zen2han(s)
han2zen(s)
```

Arguments

s	A character vector. UTF-8 encoding is preferable.

Details

Japanese graphic characters are traditionally classed into fullwidth (zenkaku) and halfwidth (hankaku) form. Alphabets, numbers, and symbols can take either from, while Hiragana, Katakana, and Kanji are only available as fullwidth characters. It causes troubles in string manipulation such as matching or searching where the two forms of alphabets, numbers, and symbols are mixed in. Thus, the character data should be sanitized with this function.

The targeted zenkaku characters are numbers, alphabets, punctuation marks, and other special symbols. Katakana is not the target of zen2han because the halfwidth Katakana is rather a troublemaker.

han2zen functions reversely. This is useful for Japanese users to escape prohibitive characters in strings (e.g., '$' in a character vector).

Value

zen2han returns a character vector. All alphabets, numbers, and symbols have their halfwidth from.

han2zen returns a character vector. All alphabets, numbers, and symbols have their fullwidth from.

Author(s)

Susumu Tanimura <aruminat@gmail.com>

References

Halfwidth and Fullwidth Forms http://www.alanwood.net/unicode/halfwidth_and_fullwidth_forms.html

See Also

han2zen, showNonASCII

Examples

```
zenkaku
zen2han(as.character(zenkaku))
```

Index