Package: geotools (via r-universe)

November 26, 2024

Type Package

Title Support for Geography Packages

Version 1.1.0.9000

Maintainer Pascal Burkhard <pascal.burkhard@gmail.com>

Description Support functions for the GeoVerse packages.

License MIT + file LICENSE

URL https://geotools.nenuial.org

BugReports http://github.com/nenuial/geotools/issues/

Depends R (>= 3.5)

Imports cli, clock, countrycode, dplyr, glue, HMDHFDplus, httr2, ISOcodes, lifecycle, purrr, rappdirs, scales, settings, sf, stringr, tibble, wbstats, withr

Suggests forcats, santoku

Config/Needs/website nenuial/geopkg

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libicu-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

Repository https://nenuial.r-universe.dev

RemoteUrl https://github.com/Nenuial/geotools

RemoteRef HEAD

RemoteSha adad8f9ccaa1f819e9649aa797688e2bf00c856b

Contents

gtl_admin_code . . . 2
gtl_chk_api_call . . . 4
gtl_chk_idb_api_key . . . 4
gtl_chk_rnoaa_api_key . . . 5
gtl_country_list_noaa . . . 5
gtl_country_list_wb . . . 6
gtl_crs_proj . . . 6
gtl_crs_regional . . . 7
gtl_dwnl_api_json . . . 8
gtl_full_date . . . 8
gtl_gis_tissot_indicatrix . . . 9
gtl_hc_color_axis . . . 9
gtl_hc_discrete_color_axis . . . 10
gtl_hmd_codes . . . 10
gtl_koppen_code . . . 11
gtl_options . . . 12
gtl_opt_long_language . . . 12
gtl_opt_set_i18n . . . 13
gtl_opt_short_language . . . 14
gtl_relabel_dash . . . 14
gtl_sci_10_html . . . 15
gtl_swiss_canton_abbr . . . 16
gtl_swiss_canton_id . . . 16
gtl_translate_enfr . . . 17
gtl_translator . . . 17

Index **19**

`gtl_admin_code` *Provide custom countrycode function*

Description

Provide custom countrycode function

Usage

```
gtl_admin_code(..., country = c("China", "Switzerland", "Russia"))
```

Arguments

`...`	Arguments passed on to `countrycode::countrycode`
	`sourcevar` Vector which contains the codes or country names to be converted (character or factor)
	`origin` A string which identifies the coding scheme of origin (e.g., `"iso3c"`). See `codelist` for a list of available codes.

`destination` A string or vector of strings which identify the coding scheme of destination (e.g., `"iso3c"` or `c("cowc", "iso3c")`). See `codelist` for a list of available codes. When users supply a vector of destination codes, they are used sequentially to fill in missing values not covered by the previous destination code in the vector.

`warn` Prints unique elements from sourcevar for which no match was found

`nomatch` When countrycode fails to find a match for the code of origin, it fills-in the destination vector with `nomatch`. The default behavior is to fill non-matching codes with `NA`. If `nomatch = NULL`, countrycode tries to use the origin vector to fill-in missing values in the destination vector. `nomatch` must be either `NULL`, of length 1, or of the same length as `sourcevar`.

`custom_dict` A data frame which supplies a new dictionary to replace the built-in country code dictionary. Each column contains a different code and must include no duplicates. The data frame format should resemble `codelist`. Users can pre-assign attributes to this custom dictionary to affect behavior (see Examples section):

- "origin.regex" attribute: a character vector with the names of columns containing regular expressions.
- "origin.valid" attribute: a character vector with the names of columns that are accepted as valid origin codes.

`custom_match` A named vector which supplies custom origin and destination matches that will supercede any matching default result. The name of each element will be used as the origin code, and the value of each element will be used as the destination code.

`origin_regex` NULL or Logical: When using a custom dictionary, if TRUE then the origin codes will be matched as regex, if FALSE they will be matched exactly. When NULL, `countrycode` will behave as TRUE if the origin name is in the `custom_dictionary`'s `origin_regex` attribute, and FALSE otherwise. See examples section below.

`country` Select the custom dictionary, currently one of *China*, *Switzerland* or *Russia*

Value

A vector with the desired output

Examples

```
tibble::tribble(
  ~canton, ~population,
  "Neuchâtel", 176496,
  "Vaud", 805098,
  "Genève", 504128
) -> population

population |>
  dplyr::mutate(iso = gtl_admin_code(
    sourcevar = canton,
    origin = "canton.name.regex",
    destination = "iso",
```

```
  origin_regex = TRUE,
  country = "Switzerland"
))
```

`gtl_chk_api_call` *Check API connexion*

Description

Check API connexion

Usage

```
gtl_chk_api_call(call)
```

Arguments

`call`	An API url

Value

Returns an error if the API call fails

`gtl_chk_idb_api_key` *Check IDB API key*

Description

Function to check if the IDB API key is correctly set.

Usage

```
gtl_chk_idb_api_key()
```

Value

Returns an error if the IDB API key is not properly set

See Also

`idbr::idb_api_key()`

Examples

```
# Not run: fails if the IDB API key isn't set

gtl_chk_idb_api_key()
```

gtl_chk_rnoaa_api_key *Check NOAA API key*

Description

Function to check if the NOAA API key is correctly set.

Usage

```
gtl_chk_rnoaa_api_key()
```

Value

Returns an error if the NOAA API key is not properly set

Examples

```
# Not run: fails if the NOAA API key isn't set

gtl_chk_rnoaa_api_key()
```

gtl_country_list_noaa *NOAA country list*

Description

This function returns a list of World Bank countries for a choice selector in a Shiny app.

Usage

```
gtl_country_list_noaa()
```

Examples

```
# Not run: need the NCDC city database
gtl_country_list_noaa()
```

gtl_country_list_wb *World Bank country list*

Description

This function returns a list of World Bank countries for a choice selector in a Shiny app.

Usage

```
gtl_country_list_wb()
```

Examples

```
gtl_country_list_wb()
```

gtl_crs_proj *Get a defined CRS definition for a given code*

Description

A function that returns the corresponding CRS for different codes.

Usage

```
gtl_crs_proj(code)
```

Arguments

code	A string with the CRS identifier

Details

The current list of available codes is:

- eqearth
- equirec
- gallpeters
- goode
- hobodyer
- mercator
- robinson
- wintri

Value

A CRS object

Examples

```
gtl_crs_proj("eqearth")

gtl_crs_proj("robinson")
```

`gtl_crs_regional` *Get a defined proj4 string for a given country*

Description

A function that returns an ideal CRS for a given country.

Usage

```
gtl_crs_regional(country)
```

Arguments

`country`	A string with the country name

Details

The current list of possible countries:

- Russia

Value

A CRS object

Examples

```
gtl_crs_regional("Russia")
```

gtl_dwnl_api_json *Download JSON response from API*

Description

Download JSON response from API

Usage

```
gtl_dwnl_api_json(call)
```

Arguments

call	An API url

Value

JSON body

Examples

```
gtl_dwnl_api_json("https://dummyjson.com/products/1")
```

gtl_full_date *Format full date*

Description

This function formats a date in a long format (see return value). The format takes the current language setting into account (see gtl_opt_set_i18n()).

Usage

```
gtl_full_date(date)
```

Arguments

date	A date in YYYY-MM-DD format

Value

A formated day month year (%d %B %Y) date

See Also

gtl_opt_set_i18n()

Examples

```
# Set en english locale
gtl_opt_set_i18n("en_US")
gtl_full_date("2020-02-20")

# Set a french locale
gtl_opt_set_i18n("fr_FR")
gtl_full_date("2020-02-20")
```

`gtl_gis_tissot_indicatrix` *World Tissot Matrix*

Description

World Tissot Matrix

Usage

```
gtl_gis_tissot_indicatrix()
```

Value

A simple feature layer

Examples

```
gtl_gis_tissot_indicatrix()
```

`gtl_hc_color_axis` *Return a list of color breaks for Higchart maps*

Description

Return a list of color breaks for Higchart maps

Usage

```
gtl_hc_color_axis(breaks, palette)
```

Arguments

`breaks`	The data with the breaks (using chopped data with `santoku::santoku`)
`palette`	A color palette that can take a numeric argument

Value

A list of lists to use in colorAxis' dataClasses argument

Examples

```
data <- seq(1, 20)
data_cut <- santoku::chop(data, breaks = c(5, 10, 15))
gtl_hc_color_axis(data_cut, rainbow)
```

---

gtl_hc_discrete_color_axis *Return a list of color breaks for Higchart maps*

Description

Return a list of color breaks for Higchart maps

Usage

```
gtl_hc_discrete_color_axis(breaks, palette)
```

Arguments

breaks	The data with the breaks (factors !)
palette	A color palette that can take a numeric argument

Value

A list of lists to use in colorAxis' dataClasses argument

---

gtl_hmd_codes *Provide HMD country codes*

Description

This function returns a dataframe with the 47 countries for which the Human Mortality Database provides data. For each country there is a name and a code.

Usage

```
gtl_hmd_codes()
```

Value

A dataframe with 2 columns and 47 rows

Examples

```
gtl_hmd_codes()
```

`gtl_koppen_code` *Determine the Köppen climate based on temperature, precipitation and the latitude.*

Description

Determine the Köppen climate based on temperature, precipitation and the latitude.

Usage

```
gtl_koppen_code(temp, prec, lat)
```

Arguments

`temp`	A vector with temperatures in degree Celsius for each month
`prec`	A vector with precipitations in millimeters for each month
`lat`	The latitude in degrees

Value

A string with the Köppen climate determination

Examples

```
# Based on climate data for Geneva
gtl_koppen_code(
  temp = c(2.1, 3, 6.8, 10.4, 14.8, 18.4, 20.5, 20.2, 15.8, 11.5, 6, 2.7),
  prec = c(70.8, 60.6, 56.9, 69, 75.7, 78.8, 83.2, 81.4, 94.9, 97.6, 90.3, 85.2),
  lat = 6
)
```

gtl_options *Set or get options for my package*

Description

Set or get options for my package

Usage

```
gtl_options(...)
```

Arguments

...	Option names to retrieve option values or [key]=[value] pairs to set options.

Supported options

The following options are supported

`language` The default language

`country` The default country setting

`theme` The default ggplot theme

`mode` The default ggplot mode

`opacity` The default opacity

Examples

```
# Retrieve default options
gtl_options("language")
gtl_options("plot_standard_width")

# Change a setting and retrieve it
gtl_options(plot_standard_width = 25)
gtl_options("plot_standard_width")
```

gtl_opt_long_language *Return long language name*

Description

Function to get the current language setting in full.

Usage

```
gtl_opt_long_language()
```

Value

A string with the language name (lowercase)

See Also

gtl_opt_set_i18n()

Examples

```
# With a french locale
gtl_opt_set_i18n("fr_CH")
gtl_opt_long_language()

# With an english locale
gtl_opt_set_i18n("en_US")
gtl_opt_long_language()
```

---

gtl_opt_set_i18n *Set language and country settings*

Description

This function allows setting the default language and country settings using a valid i18n code.

Usage

```
gtl_opt_set_i18n(code)
```

Arguments

code	A valid i18n code (example: fr_CH)

Examples

```
# Swiss french locale
gtl_opt_set_i18n("fr_CH")
gtl_options("language")
gtl_options("country")

# American english locale
gtl_opt_set_i18n("en_US")
gtl_options("language")
gtl_options("country")
```

gtl_opt_short_language *Return short language code*

Description

Function that returns the short language code currently set if it is in the list of *valid* options. If the current language isn't in the list of *valid* options, the first element of the valid elements is returned.

Usage

```
gtl_opt_short_language(valid = c("en", "fr"))
```

Arguments

valid	A vector with valid language options

Value

A string

See Also

gtl_opt_set_i18n()

Examples

```
gtl_opt_set_i18n("fr_CH")
gtl_opt_short_language(valid = c("de", "fr"))

# With the current local not among the valid options
gtl_opt_set_i18n("en_US")
gtl_opt_short_language(valid = c("de", "fr"))
```

gtl_relabel_dash *Reformat santoku chop dash labels*

Description

This function can be used to reformat the levels produced by the santoku::lbl_dash() function.

Usage

```
gtl_relabel_dash(x)
```

Arguments

x	A string vector of labels

Value

A string vector with clean labels

See Also

santoku::lbl_dash()

Examples

```
data <- seq(1, 20)
data_cut <- santoku::chop(data,
  breaks = c(5, 7, 13),
  labels = santoku::lbl_dash(),
  extend = TRUE, drop = FALSE
)
forcats::fct_relabel(data_cut, gtl_relabel_dash)
```

---

gtl_sci_10_html *Format numbers using scientific notation for html*

Description

Format numbers using scientific notation for html

Usage

```
gtl_sci_10_html(x)
```

Arguments

x	Number to format

Value

A formatted expression

Examples

```
gtl_sci_10_html(10e6)
gtl_sci_10_html(10e9)
```

gtl_swiss_canton_abbr *Get Swiss canton abbreviations*

Description

This is a shortcut for Swiss canton abbreviation based on the canton name.

Usage

```
gtl_swiss_canton_abbr(canton_name)
```

Arguments

canton_name	A vector of Swiss canton names

Value

A vector of Swiss canton abbreviations

Examples

```
gtl_swiss_canton_abbr(c("Jura", "Genève", "Neuchâtel"))
```

gtl_swiss_canton_id *Get Swiss canton FSO ids*

Description

This is a shortcut for Swiss canton codes based on the canton name.

Usage

```
gtl_swiss_canton_id(canton_name)
```

Arguments

canton_name	A vector of Swiss canton names

Value

A vector of Swiss canton FSO ids

Examples

```
gtl_swiss_canton_id(c("Jura", "Genève", "Neuchâtel"))
```

gtl_translate_enfr *Translate function*

Description

Translate function

Usage

```
gtl_translate_enfr(english, french)
```

Arguments

english	English string
french	French string
	A function that returns the english or french argument depending on the current language setting.

Value

String depending on package language option

See Also

gtl_opt_set_i18n()

Examples

```
gtl_opt_set_i18n("fr_CH") # Set language to french
gtl_translate_enfr("Hello world!", "Bonjour le monde !")

gtl_opt_set_i18n("en_US") # Set language to english
gtl_translate_enfr("Hello world!", "Bonjour le monde !")
```

gtl_translator *Creates translation function for named string list*

Description

Creates translation function for named string list

Usage

```
gtl_translator(dictionary)
```

Arguments

dictionary	A list of translations

Value

A function that translates

Examples

```
translations <- list(
  "maison" = "house",
  "chapeau" = "hat",
  "chat" = "cat"
)

translate_to_english <- gtl_translator(translations)
translate_to_english("maison")
translate_to_english("chat")
```

Index

codelist, *2, 3*
countrycode::countrycode, *2*

gtl_admin_code, 2
gtl_chk_api_call, 4
gtl_chk_idb_api_key, 4
gtl_chk_rnoaa_api_key, 5
gtl_country_list_noaa, 5
gtl_country_list_wb, 6
gtl_crs_proj, 6
gtl_crs_regional, 7
gtl_dwnl_api_json, 8
gtl_full_date, 8
gtl_gis_tissot_indicatrix, 9
gtl_hc_color_axis, 9
gtl_hc_discrete_color_axis, 10
gtl_hmd_codes, 10
gtl_koppen_code, 11
gtl_opt_long_language, 12
gtl_opt_set_i18n, 13
gtl_opt_set_i18n(), *8*, *13, 14*, *17*
gtl_opt_short_language, 14
gtl_options, 12
gtl_relabel_dash, 14
gtl_sci_10_html, 15
gtl_swiss_canton_abbr, 16
gtl_swiss_canton_id, 16
gtl_translate_enfr, 17
gtl_translator, 17

idbr::idb_api_key(), *4*

santoku::lbl_dash(), *14, 15*
santoku::santoku, *9*