Package 'naijR'

December 6, 2021

Type Package

Title Operations to Ease Data Analyses Specific to Nigeria

Version 0.3.4

Date 2021-12-06

Depends R (>= 3.6), methods, utils

Imports RColorBrewer (>= 1.1.2), lifecycle (>= 0.2.0), magrittr (>= 1.5), mapdata (>= 2.3.0), maps (>= 3.3.0), rgdal (>= 1.4.4), rlang (>= 0.4.0)

Suggests covr, here, knitr, readxl, rmarkdown, sp (>= 1.4.2), testthat, usethis

Description A set of convenience functions as well as geographical/political data about Nigeria, aimed at simplifying work with data and information that are specific to the country.

License GPL-3

LazyData TRUE

Encoding UTF-8

RoxygenNote 7.1.1

VignetteBuilder knitr

RdMacros lifecycle

URL https://brovic.github.io/naijR/

BugReports https://github.com/BroVic/naijR/issues

NeedsCompilation no

Author Victor Ordu [aut, cre] (<https://orcid.org/0000-0003-3716-0668>)

Maintainer Victor Ordu <victorordu@outlook.com>

Repository CRAN

Date/Publication 2021-12-06 11:50:06 UTC

R topics documented:

as_state . . . 2
fix_mobile . . . 3
fix_region . . . 4
is_lga . . . 5
is_state . . . 5
lgas . . . 6
lgas_nigeria . . . 7
map_ng . . . 7
naijR . . . 9
states . . . 10

Index **12**

`as_state` *Explicit coercion between State and LGA names*

Description

Takes the names of either States or LGAs and converts them explicitly into objects of the other class.

Usage

```
as_state(x)

as_lga(x)
```

Arguments

`x`	A string representing either States or Local Government Areas (LGAs) that dually name one of these administrative regions.

Details

There are a few LGAs in the country that bear the same name as their State, and this could create some confusion when trying to use some of the the functionalities of this package. The States/LGAs in question are *Bauchi, Ebonyi, Ekity, Gombe, Katsina and Kogi*.

There as subtle differences in the way these functions handle data for States as against those for LGAs. In the case of States, an object of mode `character` is the preferred argument; alternatively, an object of class `states` will serve as long as it has only one element. For LGAs, the string is the preferred argument, since an object constructed with `lgas()` that is supplied a State's name as argument will list all the LGAs in that State. If a pre-formed `lgas` object is to be coerced to a `states` object, it should first be `unclassed` or explicitly coerced with `as.character`.

Value

In the case of `as_state`, an object of class `states`; with `as_lga`, an object of class `lgas`.

Examples

```
kt.st <- states("Katsina")  # Ensure this is a State, not an LGA.
kt.lg <- suppressWarnings(as_lga(kt.st))
is_state(kt.st)             # TRUE
is_lga(kt.lg)               # TRUE

## Where there's no ambiguity, it doesn't make sense to coerce
## This kind of operation ends with an error
## Not run:
as_state("Kano")
as_lga("Michika")

## End(Not run)
```

`fix_mobile` *Fix mobile numbers*

Description

Fixes up local mobile phone numbers to a uniform text format.

Usage

```
fix_mobile(x)
```

Arguments

`x`	A character vector of numerical strings.

Details

This format is specific to that used in a given location - for now the function is useful only for Nigeria mobile numbers, which come in the format expressed by the regex pattern `"^0[7-9][0-1][0-9]{8}$"`.

Value

The updated vector, usually the column of a data frame.

fix_region *Fix Region Names*

Description

Correct any misspelt names of administrative regions i.e. States and LGAs

Usage

```
fix_region(x, ...)

## S3 method for class 'states'
fix_region(x, ...)

## S3 method for class 'lgas'
fix_region(x, interactive = FALSE, quietly = FALSE, ...)

## Default S3 method:
fix_region(x, ...)
```

Arguments

x	An S3 object of class states or lgas. For fix_region.default, a character vector can be passed but only that for States will be interpretable.
...	Arguments passed to methods.
interactive	Logical. When TRUE, the function prompts the user to interactively select the correct LGA names from a list of available options.
quietly	Logical; default argument is FALSE.

Details

The function will look through a character vector and try to determine if State or LGA names have been wrongly entered. This presupposes that the atomic vector is of type character. It does not test any missing values in the vector, leaving them untouched.

Value

The transformed object. If all names are correct, the object is returned unchanged.

is_lga *Test for Local Government Areas*

Description

Checks a given object for Local Government Areas, represented as strings.

Usage

```
is_lga(x)
```

Arguments

x	An object of type `character`. This includes higher dimension object classes like `matrix` and `array`.

Value

A logical vector the same length as the input object. Each element that is not a valid Local Government Area will evaluate to `FALSE`.

is_state *Test an Object for States*

Description

Test an Object for States

Usage

```
is_state(x)
```

Arguments

x	A vector to be tested.

Details

An element-wise check of a supplied vector is carried out. To test an entire vector and return a single boolean value, functions such as `base::all` or `base::any` should be used.

Value

A logical vector.of same length as the input. If the input object is not even of type `character`, return the object unaltered, with a warning.

Note

The function throws a warning, when a missing value is among the elements. It works only for atomic vectors, throwing an error when this is not the case or when NULL is passed to it.

Examples

```
all(is_state(naijR::states()))
is_state(c("Maryland", "Baden-Baden", "Plateau", "Sussex"))
```

---

lgas *List Local Government Areas*

Description

List Local Government Areas

Usage

```
lgas(region = NA_character_, warn = TRUE, strict = FALSE)

lgas_ng(state = NA_character_)
```

Arguments

region	Context-dependent. Either State(s) of the Federation or Local Government Area(s) - internal checks are performed to determine what applies. In cases where States are synonymous to LGAs, the default behaviour is to use the State as a basis for selecting the LGAs. This can be modified with strict. The default value is NA_character_ and will return all 774 LGAs.
warn	logical; issue a warning when one or more elements are not actually Local Government Areas (or were misspelt).
strict	logical; in the event of a name clash between State/LGA, return only the specified LGA when this argument is set to TRUE.
state	Character; State(s) in the Federation of Nigeria. Default is NA_character_.

Value

If length of ng.state == 1L, a character vector containing the names of Local Government Areas; otherwise a named list, whose elements are character vectors of the LGAs in each state.

Note

There are six (6) LGAs that share names with their State - Bauchi, Ebonyi, Gombe, Katsina, Kogi and Ekiti.

lga_ng stands deprecated and will be removed in the next minor version. New code should use lgas instead.

Examples

```
how_many_lgas <- function(state) {
  require(naijR)
  stopifnot(all(is_state(state)))
  cat(sprintf("No. of LGAs in %s State:", state),
    length(lgas(state)),
    fill = TRUE)
}
how_many_lgas("Sokoto")
how_many_lgas("Ekiti")
```

---

lgas_nigeria *Local Government Areas of Nigeria*

Description

A dataset containing the 774 Local Government Areas of Nigeria

Usage

```
lgas_nigeria
```

Format

A dataframe with 774 rows and 2 columns

lga Local Government Area

state State of the Federation

---

map_ng *Map of Nigeria*

Description

Maps of the Federal Republic of Nigeria that are based on the basic plotting idiom utilised by maps:map and its variants.

Usage

```
map_ng(
  region = character(),
  data = NULL,
  x = NULL,
  y = NULL,
  breaks = NULL,
  categories = NULL,
  title = NULL,
  caption = NULL,
  show.neighbours = FALSE,
  show.text = FALSE,
  leg.x = 13L,
  leg.y = 7L,
  leg.title,
  leg.orient = c("vertical", "horizontal"),
  ...
)
```

Arguments

region	A character vector of regions to be displayed. This could be States or Local Government Areas.
data	An object containing data, principally the variables required to plot in a map.
x, y	Numeric object or factor (or coercible to one). See *Details*.
breaks	Numeric. A vector of length >= 1. If a single value i.e. scalar, it denotes the expected number of breaks. Internally, the function will attempt to compute appropriate category sizes or fail if out-of bounds. Where length is >= 3L, it is expected to be an arithmetic sequence that represents category bounds as for cut (applicable only to choropleth maps).
categories	The legend for the choropleth-plotted categories. If not defined, internally created labels are used.
title, caption	An optional string for annotating the map.
show.neighbours	Logical; TRUE to display the immediate vicinity neighbouring regions/countries.
show.text	Logical. Whether to display the labels of regions.
leg.x, leg.y	Numeric. Position of the legend.
leg.title	String. The legend Title
leg.orient	The orientation of the legend i.e. whether horizontal or vertical.
...	Further arguments passed to map

Details

The default value for region is to print all State boundaries. data enables the extraction of data for plotting from an object of class data.frame. Columns containing States are identified. The argument also provides context for quasiquotation when providing the x and y arguments.

For x and y, when both arguments are supplied, they are taken to be point coordinates, where x represent longitude and y latitude. If only x is supplied, it is assumed that the intention of the user is to make a choropleth map, and thus, numeric vector arguments are converted into factors i.e. number classes. Otherwise factors or any object that can be coerced to a factor should be used.

For plain plots, the col argument works the same as with map. For choropleth maps, the colour provided represents a (sequential) colour palette based on RColorBrewer::brewer.pal. The available colour options can be checked with getOption("choropleth.colours") and this can also be modified by the user.

Value

An object of class maps containing the data used to draw the map and which can be used for additional calls to map or other similar functions (e.g. graphics::plot.default).

Note

When adjusting the default colour choices for choropleth maps, it is advisable to use one of the sequential palettes. For a list of of available palettes, especially for more advanced use, review RColorBrewer::display.brewer.all

Examples

```
## Not run:
map_ng() # Draw a map with default settings
map_ng(states("sw"))
map_ng("Kano")

## End(Not run)
```

---

naijR *naijR: Operations to Ease Data Analyses Specific to Nigeria*

Description

The naijR package is essentially an R package about Nigeria and for Nigeria.

Details

naijR contains a number of functions that facilitate the management of data sets of interest including data cleaning and wrangling, as well as making available a number of facilities for geo-spatial data visuatlisation.

`states` *States of the Federal Republic of Nigeria*

Description

States of the Federal Republic of Nigeria

Print regions

Return the First or Last Parts of a Region Object

Usage

```
states(states, gpz = NULL, all = TRUE, warn = TRUE)

## S3 method for class 'regions'
print(x, ...)

## S3 method for class 'regions'
head(x, ...)

## S3 method for class 'regions'
tail(x, ...)
```

Arguments

`states`	One or more States of the Federation.
`gpz`	Geopolitical zone. Default is `NULL`; optionally `"nc"`,`"ne"`,`"nw"`,`"se"`,`"ss"` and `"sw"` (see "Details").
`all`	logical; whether to include the Federal Capital Territory in the result.
`warn`	logical; issue a warning when one or more elements are not actually States (or were misspelt).
`x`	The object of class `region`
`...`	Arguments to `tail.default`

Details

gpz A geo-political zone, in the Nigerian context, is a national subdivision that groups contiguous states. Historically, they arise from subnational administrative divisions known as 'regions' that existed at the time of the country's independence. There are 6 zones - North-Central, North-East, North-West, South-East, South-South and South-West.

Value

The States of Nigeria as a whole or by zones, as an S3 object of class `states`.

Examples

```
states()  # lists names of all States
states(gpz = "se")  # lists States in South-East zone
```

Index

∗ **datasets**
lgas_nigeria, 7

as_lga *(*as_state*)*, 2
as_state, 2

cut, *8*

fix_mobile, 3
fix_region, 4

head.regions *(*states*)*, 10

is_lga, 5
is_state, 5

lgas, 6
lgas_ng *(*lgas*)*, 6
lgas_nigeria, 7

map, *8, 9*
map_ng, 7
maps:map, *7*

naijR, 9
naijR-package *(*naijR*)*, 9

print.regions *(*states*)*, 10

states, 10

tail.regions *(*states*)*, 10