Package: afriadmin (via r-universe)

August 28, 2024

Title African Administrative Boundary Polygons

Version 0.0.0.9002

Date 2021-12-12

Description Will make administrative boundary polygons for Africa easily accessible from R.

License GPL (>= 3) | file LICENSE

URL https://github.com/afrimapr/afriadmin

BugReports https://github.com/afrimapr/afriadmin/issues

Depends R (>= 2.10)

Imports sf, mapview, countrycode, grDevices, rgeoboundaries

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests knitr, GADMTools, rhdx, tmap, geojsonsf, raster

Remotes wmgeolab/rgeoboundaries

VignetteBuilder knitr

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

RemoteUrl https://github.com/afrimapr/afriadmin

RemoteRef HEAD

RemoteSha b36a276fd65dca828d2a9e44c88a7a6813c414a5

Contents

`afcountrynames` *return all africa country names (may not need the function, given that I've saved as data afcountries)*

Description

return all africa country names (may not need the function, given that I've saved as data afcountries)

Usage

```
afcountrynames(nameoriso3c = "name", filtercountries = NULL)
```

Arguments

`nameoriso3c`	whether to return vector of 'name' or 'iso3c' or 'both'
`filtercountries`	optional filter countries that are in 'gadm2' potential to add others

Value

character vector of african country names

Examples

```
afcountrynames()
```

afplotadmin *plot admin levels*

Description

*in progress aim to plot admin levels for a single country either on same map or facetted to make it easier to see the structure

Usage

```
afplotadmin(country, datasource = "gadm", addcontext = FALSE)
```

Arguments

country	a character vector of country names or iso3c character codes.
datasource	data source, initial default 'gadm'
addcontext	whether to plot neighbouring countries as well

Value

sf

afriadmin *Get admin polygons*

Description

returns admin polygons for specified countries and optionally plots map

Usage

```
afriadmin(
  country,
  level = 1,
  datasource = "geoboundaries",
  type = "simple",
  version = NULL,
  path = tempdir(),
  quiet = FALSE,
  plot = "mapview"
)
```

Arguments

country	a character vector of country names or iso3c character codes.
level	which admin level to return
datasource	data source, 'geoboundaries' or 'gadm'
type	for geoboundaries, boundary type defaults to 'simple' One of 'HPSCU', 'HPSCGS', 'SSCGS', 'SSCU', or 'precise' 'simple' 'precise standard' 'simple standard' Determines the type of boundary link you receive. More on details
version	for geoboundaries defaults to the most recent version of geoBoundaries available. The geoboundaries version requested, with underscores. For example, 3_0_0 would return data from version 3.0.0 of geoBoundaries.
path	where to save downloaded data for gadm, defaults to tempdir()
quiet	for geoboundaries, if TRUE no message while downloading and reading the data. Default to FALSE
plot	option to display map 'mapview' for interactive, 'sf' for static

Value

sf

Examples

```
sfnga2 <- afriadmin("nigeria", level=2)
```

---

african_country_names *african country names and iso 3 letter country codes*

Description

african country names and iso 3 letter country codes

Usage

```
afcountries
```

Format

dataframe

Source

rnaturalearth

african_country_polygons *africa country polygons from GADM*

Description

africa country polygons from GADM

Usage

```
sf_af_gadm0
```

Format

sf

Source

https://gadm.org/

alllevels *get all admin levels into a single object*

Description

*in progress

Usage

```
alllevels(country, datasource = "gadm", plot = "tmap")
```

Arguments

country	a character vector of country names or iso3c character codes.
datasource	data source, initial default 'gadm'
plot	option to display map 'mapview' for interactive, 'sf' for static

Value

sf

compareadmin *compare 2 sets of admin polygons for a country*

Description

returns a mapview (or potentially later sf plot object), most arguments take a vector allowing two layers to be specified

Usage

```
compareadmin(
  country,
  level = c(1, 1),
  datasource = c("geoboundaries", "gadm"),
  type = c("simple", "simple"),
  version = c(NULL, NULL),
  quiet = FALSE,
  colors = c("red", "blue"),
  lwds = c(2, 2),
 col.regions = list(RColorBrewer::brewer.pal(9, "YlGn"), RColorBrewer::brewer.pal(9,
    "BuPu")),
  alpha = c(0.9, 0.9),
  layer.names = NULL,
  plotlegend = TRUE,
  alpha.regions = c(0.1, 0.1),
  plot = "mapview",
  plotshow = TRUE
)
```

Arguments

country	a character vector of country names or iso3c character codes.
level	1
datasource	c('geoboundaries','gadm')
type	for geoboundaries, boundary type defaults to 'simple' One of 'HPSCU', 'HPSCGS', 'SSCGS', 'SSCU', or 'precise' 'simple' 'precise standard' 'simple standard' Determines the type of boundary link you receive. More on details
version	for geoboundaries defaults to the most recent version of geoBoundaries available. The geoboundaries version requested, with underscores. For example, 3_0_0 would return data from version 3.0.0 of geoBoundaries.
quiet	for geoboundaries, if TRUE no message while downloading and reading the data. Default to FALSE
colors	for the 2 sources of polygon boundaries default red,blue
lwds	line widths for the 2 sources of boundaries default 2,2
col.regions	colour palettes for polygon fills

alpha	transparency for the two source boundaries default 0.9,0.9
layer.names	optional names for the two source layers
plotlegend	whether to add legend
alpha.regions	transparency for the two source fills default 0.9,0.9
plot	option to display map 'mapview' for interactive, 'namestable' for a table of names, 'sf' for static
plotshow	whether to display plot

Value

sf

Examples

```
compareadmin("togo", level=2)
#comparing different types from geoboundaries
compareadmin("togo", level=2,
             datasource=c('geoboundaries','geoboundaries'),
             type=c('sscu','hpscu') )
```

---

country2iso *conversion from country names to iso3c code*

Description

#todo vectorise

Usage

```
country2iso(country)
```

Arguments

country	a character vector of country names

Value

character vector of iso3c codes

Examples

```
iso3c <- country2iso("nigeria")
```

GADM36SF

admin levels available by country from GADM (and names of the levels)

Description

admin levels available by country from GADM (and names of the levels)

Usage

```
GADM36SF
```

Format

```
dataframe
```

Source

https://cran.r-project.org/web/packages/GADMTools/index.html

hdxadmin

hdx admin layers starting to looking at downloading NOT WORKING YET

Description

#todo vectorise

Usage

```
hdxadmin(country, level = 2, plot = "sf")
```

Arguments

country	a character vector of country names
level	which admin level to return
plot	option to display map 'mapview' for interactive, 'sf' for static

Value

not sure yet

Examples

```
#hdxadmin("nigeria")
```

hdxlive *first go at interacting with hdx itos live admin layers from https://github.com/SimonbJohnson/cod_topo*

Description

NOTE I changed ZBM to ZMB for Zambia

Usage

```
hdxlive(
  country,
  level = 2,
  colr_hdx = rgb(0, 0, 1, alpha = 0.4),
  colr_gadm = rgb(1, 0, 0, alpha = 0.4),
  lwd = 2,
  legend = TRUE,
  add = FALSE,
  alpha = 0.5,
  plot = "compare_sf"
)
```

Arguments

country	a character vector of country names
level	which admin level to return
colr_hdx	colour for hdx boundaries in plots defaults blue semi-transparent
colr_gadm	colour for gadm boundaries in plots defaults red semi-transparent
lwd	line width in plots
legend	legend in sf plots TRUE/FALSE
add	whether to add sf plot to existing plot
alpha	transparency in mapview plots
plot	option to display map 'mapview' for interactive, 'sf' for static, 'compare' to sf compare with gadm

Details

start by checking whether I can open a few geojson files copied across

Value

not sure yet

Examples

```
#hdxlive("angola")
#hdxlive('benin',level=2)
#hdxlive('zimbabwe',level=2)
#to compare gadm and hdx
#hdxlive('malawi',level=2, plot='compare_mapview')
```

`iso2country` *conversion from iso3c code to country names*

Description

#todo vectorise

Usage

```
iso2country(iso3c)
```

Arguments

`iso3c`	a character vector of country codes

Value

character vector of country names

Examples

```
name <- iso2country("nga")
```

`maxadmin` *return max admin level for a country*

Description

initialy gadm but could be other sources too

Usage

```
maxadmin(country, datasource = "geoboundaries")
```

Arguments

`country`	a character vector of country names
`datasource`	data source, initial default 'gadm'

Details

#todo change this to work on downloads (or save a table) #todo add geoboundaries option #todo vectorise

Value

integer vector of max admin levels

Examples

```
maxlevel <- maxadmin("nigeria", datasource='geoboundaries')
```

Index