Package 'lingtypology'

December 28, 2016

Type Package

Title Lingustic Typology and Maping

Version 1.0.0

Depends R (>= 2.10)

Imports leaflet,
stats,
stringdist,
magrittr,
htmlwidgets,
grDevices,
rowr

Author George Moroz <agricolamz@gmail.com>

Maintainer George Moroz <agricolamz@gmail.com>

Description
This package provides R with the Glottolog database (http://glottolog.org/) and some more abilities for purposes of linguistic typology. The glottolog database containes the catalogue of languages of the world. This package helps researchers to make a linguistic maps, using philosophy of the Cross-Linguistic Linked Data project (http://clld.org/), which allows for while at the same time facilitating uniform access to the data across publications. A tutorial for this package is avaliable on GitHub wiki https://github.com/agricolamz/lingtypology/wiki.

License GPL (>= 2)

URL https://github.com/agricolamz/lingtypology

LazyData TRUE

RoxygenNote 5.0.1

R topics documented:

`aff.lang` *Get affiliation by languoid*

Description

Takes any vector of languoids and return affiliation.

Usage

```
aff.lang(x)
```

Arguments

`x`	A character vector of the languoids (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

`area.lang`, `country.lang`, `iso.lang`, `lat.lang`, `long.lang`

Examples

```
aff.lang("Korean")
aff.lang(c("Korean", "Polish"))
```

area.lang *Get macro area by languoid*

Description

Takes any vector of languoids and return macro area.

Usage

```
area.lang(x)
```

Arguments

x	character vector of the languoids (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, country.lang, iso.lang, lat.lang, long.lang

Examples

```
area.lang("Adyghe")
area.lang(c("Adyghe", "Aduge"))
```

circassian *Circassian villages in Russia*

Description

A dataset containes the list of the Circassian villages in Russia with genealogical affiliation, coordinates and district names. Most data collected during the fieldworks (2011-2016).

Usage

```
circassian
```

Format

A data frame with 8286 rows and 7 variables:

longitude longitude

latitude latitude

village name of the village

district names of the subjects of the Russian Federation: kbr — Kabardino-Balkar Republic, kch — Karachay-Cherkess Republic, kk — Krasnodar Krai, ra — Republic of Adygea, stv — Stavropol Krai

languoid names of the Circassian dialects

language according standard Circassian devision there are Adyghe and Kabardian languages

---

`countries` *Catalogue of countries names.*

Description

Catalogue of countries names.

Usage

```
countries
```

Format

A data frame with 86 rows and 3 variables:

common common name

official official name

abbreviation abreviated name

`country.lang` *Get country by languoid*

Description

Takes any vector of languoids and return affiliation.

Usage

```
country.lang(x, intersection = FALSE)
```

Arguments

x	character vector of the languoids (can be written in lower case)
intersection	logical. If TRUE, function reterns vector of countries, where all languoids from x argument are spoken.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, iso.lang, lat.lang, long.lang

Examples

```
country.lang("Udi")
country.lang(c("Udi", "Laz"))
country.lang(c("Udi", "Laz"), intersection = TRUE)
```

---

glottolog *Catalogue of languages of the world*

Description

A dataset containes the catalogue of languages of the world involving genealogical affiliation, macro-area, country, iso code, and coordinates.

Usage

```
glottolog
```

Format

A data frame with 8286 rows and 7 variables:

iso code based on ISO 639-3 http://www-01.sil.org/iso639-3/

languoid name of the languoid

affiliation genealogical affiliation

macro_area have six values Africa, Australia, Eurasia, North America, Papunesia, South America

country list of countries, where the language is spoken

latitude latitude

longitude longitude

Details

Glottolog 2.7. Hammarstrom, Harald & Forkel, Robert & Haspelmath, Martin & Bank, Sebastian. 2016. Max Planck Institute for the Science of Human History. Accessed on 2016-06-15.

Source

http://glottolog.org/

`is.glottolog` *Are these langoids in glottolog?*

Description

Takes any vector of linguoids or ISO codes and return a logical vector.

Usage

```
is.glottolog(x, response = FALSE)
```

Arguments

`x`	A character vector of linguoids (can be written in lower case)or ISO codes
`response`	logical. If TRUE, when languoid is absent, return warnings with a possible candidates.

Author(s)

George Moroz <agricolamz@gmail.com>

Examples

```
is.glottolog(c("Adyghe", "Russsian"))

# Add warning message with sugestions
is.glottolog(c("Adyge", "Russian"), response = TRUE)
# > FALSE TRUE
# Warning message:
# In is.glottolog(c("Adyge", "Russian"), response = TRUE) :
# Languoid Adyge is absent in our database. Did you mean Aduge, Adyghe?
```

iso.lang *Get ISO 639-3 code by languoid*

Description

Takes any vector of languoids and return ISO code.

Usage

```
iso.lang(x)
```

Arguments

x	A character vector of the languoids (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, country.lang, lat.lang, long.lang

Examples

```
iso.lang("Adyghe")
iso.lang(c("Adyghe", "Udi"))
```

lang.aff *Get languoids by affiliation*

Description

Takes any vector of affiliations and return languoids.

Usage

```
lang.aff(x, list = FALSE)
```

Arguments

x	A character vector of the affiliations (can be written in lower case)
list	logical. If TRUE, returns a list of languoids, if FALSE return a named vector.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

lang.country, lang.iso

Examples

```
lang.aff("Balto-Slavic")
lang.aff(c("East Slavic", "West Slavic"))
lang.aff(c("East Slavic", "West Slavic"), list = TRUE)
```

---

lang.country *Get languoids by country*

Description

Takes any vector of countries and return languoids.

Usage

```
lang.country(x, list = FALSE)
```

Arguments

x	character vector of the countries (can be written in lower case)
list	logical. If TRUE, returns a list of languoids, if FALSE return a vector.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

lang.aff, lang.iso

Examples

```
lang.country("Bali")
lang.country(c("Bali", "Luxembourg"))
lang.country(c("Bali", "Luxembourg"), list = TRUE)
## What languoids are both in North Korea and in South Korea?
lang.country("Korea")
```

lang.iso *Get languoid by ISO 639-3 code*

Description

Takes any vector of ISO codes and return languoids.

Usage

```
lang.iso(x)
```

Arguments

x	A character vector of the ISO codes.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

lang.aff, lang.country

Examples

```
lang.iso("ady")
lang.iso(c("ady", "rus"))
```

lat.lang *Get latitude by languoid*

Description

Takes any vector of languoids and return latitude.

Usage

```
lat.lang(x)
```

Arguments

x	A character vector of the languoids (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, country.lang, iso.lang, long.lang

Examples

```
lat.lang("Adyghe")
long.lang("Adyghe")
lat.lang(c("Adyghe", "Russian"))
long.lang(c("Adyghe", "Russian"))
```

---

long.lang *Get longitude by languoid*

Description

Takes any vector of languoids and return longitude.

Usage

```
long.lang(x)
```

Arguments

x	A character vector of the languoids (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, country.lang, iso.lang, lat.lang

Examples

```
lat.lang("Adyghe")
long.lang("Adyghe")
lat.lang(c("Adyghe", "Russian"))
long.lang(c("Adyghe", "Russian"))
```

makelink *Make a link for a languoid*

Description

Takes any vector of linguoids and return links.

Usage

```
makelink(x, popup = NULL)
```

Arguments

x	A character vector of linguoids (can be written in lower case)
popup	character vector of strings that will appear in pop-up window

Author(s)

George Moroz <agricolamz@gmail.com>

map.feature *Create a map*

Description

Map a set of linguoids and color them by feature

Usage

```
map.feature(languages, features = "none", popup = "",
  stroke.features = NULL, latitude = NULL, longitude = NULL,
  color = NULL, stroke.color = NULL, title = NULL, control = TRUE,
  legend = TRUE, radius = 5, stroke.radius = 9.5, opacity = 1,
  stroke.opacity = 1, ...)
```

Arguments

languages	character vector of linguoids (can be written in lower case)
features	character vector of features
popup	character vector of strings that will appear in pop-up window
stroke.features	additional independent stroke features
latitude	numeric vector of latitudes
longitude	numeric vector of longitudes

color	vector of colors
stroke.color	vector of stroke colors
title	of a legend
control	logical. If FALSE, function doesn't show layer control buttons.
legend	logical. If FALSE, function doesn't show legend.
radius	a numeric vector of radii for the circles.
stroke.radius	a numeric vector of stroke radii for the circles.
opacity	a numeric vector of marker opacity.
stroke.opacity	a numeric vector of stroke opacity.
...	further arguments of leaflet package.

Details

Takes any vector of linguoids and return a map.

Author(s)

George Moroz <agricolamz@gmail.com>

Examples

```
map.feature(c("Adyghe", "Russian"))

## Map all Slavic languages
map.feature(lang.aff(c("Slavic")))

## Color linguoids by feature
df <- data.frame(lang = c("Adyghe", "Kabardian", "Polish", "Russian", "Bulgarian"),
feature = c("polysynthetic", "polysynthetic", "fusion", "fusion", "fusion"))
map.feature(df$lang, df$feature)

## Adding pop-up
df <- data.frame(lang = c("Adyghe", "Kabardian", "Polish", "Russian", "Bulgarian"),
feature = c("polysynthetic", "polysynthetic", "fusion", "fusion", "fusion"),
popup = c("Adyghe", "Adyghe", "Slavic", "Slavic", "Slavic"))
map.feature(df$lang, df$feature, df$popup)

## Adding title
df <- data.frame(lang = c("Adyghe", "Kabardian", "Polish", "Russian", "Bulgarian"),
feature = c("polysynthetic", "polysynthetic", "fusion", "fusion", "fusion"),
popup = c("Adyghe", "Adyghe", "Slavic", "Slavic", "Slavic"))
map.feature(df$lang, df$feature, df$popup, title = "type of a language")

## Add your own coordinates
map.feature("Adyghe", latitude = 43, longitude = 57)

## Add you own colors
df <- data.frame(lang = c("Adyghe", "Kabardian", "Polish", "Russian", "Bulgarian"),
feature = c("polysynthetic", "polysynthetic", "fusion", "fusion", "fusion"),
```

```
popup = c("Adyghe", "Adyghe", "Slavic", "Slavic", "Slavic"))
map.feature(df$lang, df$feature, df$popup, color = c("green", "navy"))

## Remove control buttons
map.feature(lang.aff("Sign"), control = FALSE)
```

Index