# Package 'r2country'

May 9, 2026

**Type** Package

**Title** Country Data with Names, Capitals, Currencies, Populations, Time, Languages and so on

**Version** 2.0.2.4.0

**Maintainer** Obinna Obianom <idonshayo@gmail.com>

**Description** Obtain information about countries around the globe. Information for names, states, languages, time, capitals, currency and many more. Data source are 'Wikipedia' <https://www.wikipedia.org>, 'TimeAndDate' <https://www.timeanddate.com> and 'CountryCode' <https://countrycode.org>.

**License** MIT + file LICENSE

**URL** https://r2country.obi.obianom.com

**BugReports** https://github.com/oobianom/r2country

**Depends** R (> 3.6)

**Imports** utils, stats, quickcode

**Suggests** rmarkdown, knitr, qpdf, testthat

**Encoding** UTF-8

**VignetteBuilder** knitr

**Language** en-US

**RoxygenNote** 7.2.3

**LazyData** true

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Obinna Obianom [aut, cre]

**Repository** CRAN

**Date/Publication** 2024-08-01 14:40:02 UTC

# Contents



---

`byContinent` *Fetch countries by continent*

---

## Description

Obtain country data matching language first one or more letters

## Usage

```
byContinent(
 name = c("asia", "europe", "africa", "north america", "south america", "oceania"),
  full.list = TRUE
)
```

## Arguments

| | |
|---|---|
| `name` | name of continent |
| `full.list` | whether to return only name of country or full list |

### Details

Note that choices for names of continent includes 'asia','europe','africa','north america','south america','oceania'

### Value

country data list matching continent

### Examples

```
# task 1: get only names of countries that contains with "africa" or "AFrica"
# note that the search in case-insensitive
byContinent("africa", full.list = FALSE)

# task 2: get only names of countries that contains with "ASIA" or "asia"
byContinent("asia", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country
byContinent("europe", full.list = TRUE)
```

---

## callingCodeOf *Fetch the calling code of a country*

---

### Description

With specified country name or names, get the associated calling code

### Usage

```
callingCodeOf
```

### Format

An object of class `list` of length 193.

### Value

a list containing all countries and their corresponding calling code

### Examples

```
# view the searchable countries, return first 6
head(names(callingCodeOf))

#task 0: check if the calling code of japan is included
#should be all in lower case
grep("japan",names(callingCodeOf), value = TRUE)
```

```
#task 1: check the calling code of nigeria
callingCodeOf$nigeria

#task 2: check the calling code of united states
callingCodeOf$`united states`


#task 3: check calling code of multiple countries
callingCodeOf[c("slovenia","romania","malaysia")]

#task 4: what if the calling code is not available
callingCodeOf[c("randomcountry","mexico","luxembourg")]
```

---

`capitalOf` *Fetch the latest capital of a country*

---

## Description

With specified country name or names, get the associated capital

## Usage

```
capitalOf
```

## Format

An object of class `list` of length 193.

## Value

a list containing all countries and their corresponding capital

## Examples

```
# view the searchable countries, return first 6
head(names(capitalOf))

#task 0: check if the capital of japan is included
#should be all in lower case
grep("japan",names(capitalOf), value = TRUE)

#task 1: check the capital of nigeria
capitalOf$nigeria

#task 2: check the capital of united states
capitalOf$`united states`


#task 3: check capital of multiple countries
```

```
capitalOf[c("slovenia","romania","malaysia")]

#task 4: what if the capital is not available
capitalOf[c("randomcountry","mexico","luxembourg")]
```

---

`city_time` *Dataset of current time in Cities*

---

## Description

Dataset containing time different in various cities with respect to USA/New York time

## Usage

```
data(city_time)
```

## Format

A data frame with the current time in various cities

**City** Names of cities

**Timediff** Time difference relative to New York time

## Value

cities and the current time relative to New York

## Source

Internal data consisting of cities and their corresponding time

## References

https://www.timeanddate.com

## Examples

```
# load the cities and languages dataset
data(city_time)

# view content of the city-language data
head(city_time)
```

## continentOf *Fetch the continent of a country*

### Description

Get the continent that a particular country belongs to

### Usage

```
continentOf
```

### Format

An object of class `list` of length 193.

### Value

a list containing all countries and corresponding continents

### Examples

```
# view the searchable countries, return first 15
names(continentOf)[1:15]

#task 1: view the continent of algeria
continentOf$algeria

#task 2: view the continent of nigeria
continentOf$nigeria

#task 3: view the continent of multiple countries
continentOf[c("niger","china","colombia")]

#task 4: if the continent is not available
continentOf[c("niger","china","randomtest")]
```

## COUNTRIES *Names of all countries in upper case*

### Description

Fetch the names of all countries in upper case

### Usage

```
COUNTRIES
```

### Format

An object of class `character` of length 193.

### Value

Names of all countries in the globe in upper case

### Examples

```
# list all the countries
COUNTRIES

# list only 5 countries
COUNTRIES[1:5]

# create a dataset with all countries of the world
data.frame(ID = 1, Names = COUNTRIES)
```

---

## `countries` *Names of all countries*

---

### Description

Fetch the names of all countries

### Usage

```
countries
```

### Format

An object of class `character` of length 193.

### Value

Names of all countries in the globe

### Examples

```
# list all the countries
countries

# list only 5 countries
countries[1:5]

# create a dataset with all countries of the world
data.frame(ID = quickcode::number(length(countries)), Names = countries)
```

countryEndsWith *Fetch countries that ends with specified characters*

## Description

Subset and return countries given a specified characters to search

## Usage

```
countryEndsWith(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| char | character to search for |
| full.list | whether to return only name of country or full list |

## Value

country data list matching a specified character

## See Also

[countryStartsWith()] for country search starting with specified characters, and [countryHas()] for countries that contain specified characters.

## Examples

```
# task 1: get only names of countries that end with "A" or "a"
# note that the search in case-insensitive
countryEndsWith("A", full.list = FALSE)

# task 2: get only names of countries that end with "No" or "no"
countryEndsWith("no", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country
countryEndsWith("no")
```

countryHas *Fetch countries that contains with specified characters*

## Description

Subset to obtain data for countries containing specified characters to search

## Usage

```
countryHas(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| char | character to search for |
| full.list | whether to return only name of country or full list |

## Value

country data list matching content from a specified character

## See Also

countryStartsWith for country search starting with specified characters, and [countryEndsWith()] for countries that end with a specified characters.

## Examples

```
# task 1: get only names of countries that contains with "ER" or "er"
# note that the search in case-insensitive
countryHas("er", full.list = FALSE)

# task 2: get only names of countries that contains with "LAND" or "lAND" or "land"
countryHas("lAND", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country
countryHas("many", full.list = TRUE)
```

`countryStartsWith` *Fetch countries that start with specified characters*

## Description

Subset and return countries given a specified characters to search

## Usage

```
countryStartsWith(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| `char` | character to search for |
| `full.list` | whether to return only name of country or full list |

## Value

country data list matching a specified character

## See Also

[countryEndsWith()] for country search ending in specified character, and [countryHas()] for countries that contain specified characters.

## Examples

```
# task 1: get only names of countries that start with "A" or "a"
# note that the search in case-insensitive
countryStartsWith("A", full.list = FALSE)

# task 2: get only names of countries that start with "No" or "no"
countryStartsWith("no", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country
countryStartsWith("no")
```

`country_calling_code` *Dataset of countries and their calling code*

## Description

Dataset containing country IDs and their calling code

## Usage

```
data(country_calling_code)
```

## Format

A data frame with the calling code of countries

**ID** country identifiers

**callingcode** Calling code of countries

## Value

calling code of countries

## Source

Internal data consisting of countries and their calling code

## References

https://countrycode.org

## See Also

Data `country_names` for linkage of IDs with country calling code

## Examples

```
# load the calling code dataset
data(country_calling_code)

# view content of the city-calling code data
head(country_calling_code)
```

`country_capital` *Dataset of countries and their capitals*

## Description

Dataset containing country IDs and their capitals

## Usage

```
data(country_capital)
```

## Format

A data frame with the capitals that countries belong to

**ID** country identifiers

**capital** capitals of various country

## Value

corresponding capitals of countries

## Source

Internal data consisting of countries and their capital

## References

https://www.wikipedia.org

## See Also

Data `country_names` for linkage of IDs with capital

## Examples

```
# load the capitals dataset
data(country_capital)
data(country_names)
finaldb <- cbind(country_names,country_capital)

# view content of the country and population
head(finaldb)
```

`country_continent` *Dataset of countries and their continents*

## Description

Dataset containing country IDs and their continents

## Usage

```
data(country_continent)
```

## Format

A data frame with the continents that countries belong to

**ID** country identifiers

**continent** continents of various country

## Value

corresponding continents of countries

## Source

Internal data consisting of countries and their continent

## References

https://www.wikipedia.org

## See Also

Data `country_names` for linkage of IDs with continent

## Examples

```
# load the continent dataset
data(country_continent)
data(country_names)
finaldb <- cbind(country_names,country_continent)

# view content of the country and population
head(finaldb)
```

`country_language` *Dataset of country official languages*

## Description

Data for all countries and their associated languages

## Usage

```
data(country_language)
```

## Format

A data frame with the languages of countries

**ID** Identifier for the countries

**officiallanguage** official languages of countries

## Value

country ID and official languages

## Source

Internal data consisting of all country IDs and their official languages

## References

https://www.wikipedia.org

## See Also

Data `country_names` for linkage of IDs with country names

## Examples

```
# load the country ID and languages
data(country_language)

# view content of the countries languages
head(country_language)
```

`country_money` *Dataset for Country Currencies*

## Description

Data for all country currency names and symbols

## Usage

```
data(country_money)
```

## Format

A data frame with all countries and columns for currency

**ID** Identifier for countries

**currency** name of the currency

**symbol** symbol of the currency

**isocode** The ISO code of the currency

**fractionalunity** The fractional unit of the currency

## Value

data for currencies by country

## Source

Internal data for currencies belonging to every country

## References

https://www.wikipedia.org

## See Also

`country_names` for linkage of IDs with country names

## Examples

```
data(country_money)

# view content of the countries currency
head(country_money)
```

country_names *Dataset of country names*

## Description

Data for all country currency names and associated ID

## Usage

```
data(country_names)
```

## Format

A data frame with the names of all countries

**ID** Identifier for the countries

**name** name of the countries

## Value

data for names of country

## Source

Internal data consisting of all country names

## References

https://www.wikipedia.org

## Examples

```
# load the country names
data(country_names)

# view content of the countries currency
head(country_names)
```

country_population *Dataset of countries and their latest population*

## Description

Dataset containing country IDs and their population by year

## Usage

```
data(country_population)
```

## Format

A data frame with the population of countries

**ID** country identifiers

**population2023** country population as of 2023

## Value

corresponding population size of countries

## Source

Internal data consisting of countries and their population

## References

https://www.wikipedia.org

## See Also

Data country_names for linkage of IDs with country population size

## Examples

```
# load the population dataset
data(country_population)
data(country_names)
finaldb <- cbind(country_names,country_population)

# view content of the country and population
head(finaldb)
```

currencyOf *Fetch the currency of a country*

## Description

With specified country name or names, get the associated currency

## Usage

```
currencyOf
```

## Format

An object of class `list` of length 193.

## Value

a list containing all countries and their corresponding currency

## Examples

```
# view the searchable countries, return first 6
head(names(currencyOf))

#task 0: check if the currency of spain is included
#should be all in lower case
grep("spain",names(currencyOf), value = TRUE)

#task 1: check the currency of spain
currencyOf$spain

#task 2: check the currency of singapore list
currencyOf$singapore # return a list of singapore
currencyOf$singapore['symbol'] #returns the symbol
currencyOf$singapore['isocode'] #returns the iso code
currencyOf$singapore['fractionalunity'] #returns the fractional unit


#task 3: check currencies of multiple countries
currencyOf[c("slovenia","romania","malaysia")]

#task 4: what if the currency is not available
currencyOf[c("randomcountry","mexico","luxembourg")]
```

languageEndsWith *Fetch countries data with official language ending in specified character*

## Description

Obtain country data matching language first one or more letters

## Usage

```
languageEndsWith(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| char | character to search in languages |
| full.list | whether to return only name of country or full list |

## Value

language list or country data list matching parts of a character search on languages

## See Also

countryStartsWith for country search starting with specified characters, and [countryEndsWith()] for countries that end with a specified characters.

## Examples

```
# task 1: get only language names that ends with "EN" or "en"
# note that the search in case-insensitive
languageEndsWith("eN", full.list = FALSE)

# task 2: get only language names that ends with "chi"
languageEndsWith("chi", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country with the language
languageEndsWith("sin", full.list = TRUE)

# searching text with no results
languageEndsWith("er", full.list = FALSE)
```

languageHas *Fetch countries data based on official language prefix*

## Description

Obtain country data matching language first one or more letters

## Usage

```
languageHas(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| char | character to search for |
| full.list | whether to return only name of country or full list |

## Value

country data list matching content from a specified character

## See Also

countryStartsWith for country search starting with specified characters, and [countryEndsWith()] for countries that end with a specified characters.

## Examples

```
# task 1: get only names of countries that contains with "ER" or "er"
# note that the search in case-insensitive
countryHas("er", full.list = FALSE)

# task 2: get only names of countries that contains with "LAND" or "lAND" or "land"
countryHas("lAND", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country
countryHas("many", full.list = TRUE)
```

`languageOf` *Fetch the official language of a country*

## Description

With specified country name or names, get the associated official language(s)

## Usage

```
languageOf
```

## Format

An object of class `list` of length 193.

## Value

a list containing all countries and their corresponding language

## Examples

```
# view the searchable countries, return first 6
head(names(languageOf))

#task 0: check if the language of japan is included
#should be all in lower case
grep("japan",names(languageOf), value = TRUE)

#task 1: check the language of nigeria
languageOf$nigeria

#task 2: check the language of united states
languageOf$`united states`


#task 3: check language of multiple countries
languageOf[c("slovenia","romania","malaysia")]

#task 4: what if the language is not available
languageOf[c("randomcountry","mexico","luxembourg")]
```

languageStartsWith *Fetch countries data based on official language prefix*

## Description

Obtain country data matching language first one or more letters

## Usage

```
languageStartsWith(char, full.list = TRUE)
```

## Arguments

| | |
|---|---|
| char | character to search in languages |
| full.list | whether to return only name of country or full list |

## Value

language list or country data list matching parts of a character search on languages

## See Also

countryStartsWith for country search starting with specified characters, and [countryEndsWith()] for countries that end with a specified characters.

## Examples

```
# task 1: get only language names that ends with "EN" or "en"
# note that the search in case-insensitive
languageStartsWith("eN", full.list = FALSE)

# task 2: get only language names that ends with "chi"
languageStartsWith("chi", full.list = FALSE)

# task 3: repeat task 2, but return full list for each country with the language
languageStartsWith("sin", full.list = TRUE)

# searching text with no results
languageStartsWith("er", full.list = FALSE)
```

populationOf *Fetch the latest population count of a country*

## Description

With specified country name or names, get the associated population

## Usage

```
populationOf
```

## Format

An object of class `list` of length 193.

## Value

a list containing all countries and their corresponding population

## Examples

```
# view the searchable countries, return first 6
head(names(populationOf))

#task 0: check if the population of japan is included
#should be all in lower case
grep("japan",names(populationOf), value = TRUE)

#task 1: check the population of nigeria
populationOf$nigeria

#task 2: check the population of united states
populationOf$`united states`


#task 3: check population of multiple countries
populationOf[c("slovenia","romania","malaysia")]

#task 4: what if the population is not available
populationOf[c("randomcountry","mexico","luxembourg")]
```

`timeIn` *Fetch the current time in a specific city*

## Description

Get the current time using specified city name

## Usage

```
timeIn
```

## Format

An object of class `list` of length 2226.

## Value

a list containing all cities and corresponding local time

## Examples

```
# view the searchable cities, return first 10
names(timeIn)[1:10]

#task 0: check if the time in Delhi is present
#should be all in lower case
grep("delhi",names(timeIn), value = TRUE)

#task 1: check the time in delhi
timeIn$`india, delhi, new delhi`

#task 2: check the time in Boston
timeIn$`usa, massachusetts, boston`

#task 3: view the time in multiple countries
timeIn[c(
"china, zhejiang, hangzhou",
"nigeria, kano, kano",
"usa, texas, garland"
)]

#task 4: what if the city is not available
timeIn[c("randomcity","york","jerusalem")]
```

# Index