Package ‘MazamaSpatialUtils’

September 18, 2018

Type Package

Version 0.5.4

Title Spatial Data Download and Utility Functions

Author Jonathan Callahan [aut, cre],
    Ruby Fore [aut],
    Will Leahy [aut],
    Helen Miller [aut],
    Henry Nguyen [aut],
    Robin Winstanley [aut],
    Alice Yang [aut]

Maintainer Jonathan Callahan <jonathan.s.callahan@gmail.com>

Depends R (>= 3.1.0), sp

Imports dplyr, lubridate, rgdal, rgeos, rmapshaper, rvest (>= 0.3.0),
    stringr, utils, xml2

Suggests knitr, maps, markdown, readr, testthat

Description A suite of conversion scripts to create internally standardized
    spatial polygons data frames. Utility scripts use these data sets to return
    values such as country, state, timezone, watershed, etc. associated with a
    set of longitude/latitude pairs. (They also make cool maps.)

License GPL-2

VignetteBuilder knitr

Repository CRAN

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Date/Publication 2018-09-18 04:50:03 UTC
R topics documented:

codeToCountry ........................................ 3
codeToState ........................................ 3
convertEEZCountries ................................ 4
convertGADM .......................................... 5
convertHMSSmoke ..................................... 6
convertIndianLands .................................. 7
convertISOCodeTable ................................ 7
convertNaturalEarthAdm1 .............................. 8
convertOSMTimezones ................................ 9
convertSimpleCountries ................................ 10
convertSimpleCountriesEEZ ............................ 11
convertSimpleTimezones ............................... 12
convertTerrestrialEcoregions ......................... 13
convertTMWorldBorders ................................ 14
convertTMWorldBordersSimple ........................ 14
convertUSCensusCBSA ................................ 15
convertUSCensusCongress ................................ 16
convertUSCensusCounties ................................ 17
convertUSCensusStates ................................ 18
convertWBDHUC ....................................... 19
convertWikipediaTimezoneTable ....................... 20
convertWorldEEZ ..................................... 21
convertWorldTimezones ................................ 21
countryToCode ........................................ 22
countryToCode ........................................ 23
countryToCode ........................................ 24
countryToCode ........................................ 25
countryToCode ........................................ 26
countryToCode ........................................ 26
countryToCode ........................................ 27
countryToCode ........................................ 28
countryToCode ........................................ 29
countryToCode ........................................ 29
countryToCode ........................................ 30
countryToCode ........................................ 31
countryToCode ........................................ 32
countryToCode ........................................ 33
countryToCode ........................................ 34
countryToCode ........................................ 35
countryToCode ........................................ 35
countryToCode ........................................ 36
countryToCode ........................................ 36
installSpatialData ................................... 37
MazamaSpatialUtils ................................... 37
setSpatialData ........................................ 38
setSpatialData ........................................ 38
setSpatialData ........................................ 39
**codeToCountry**

Convert Country Codes to Country Names

Description

Converts a vector of ISO 3166-1 alpha-2 codes to the corresponding English names.

Usage

codeToCountry(countryCodes)

Arguments

- `countryCodes` vector of country codes to be converted

Value

A vector of English country names or NA.

**codeToState**

Convert State Codes to State Names

Description

Converts a vector of ISO 3166-2 alpha-2 state codes to the corresponding English names.

Usage

codeToState(stateCodes, countryCodes = NULL, dataset = "NaturalEarthAdm1")

Arguments

- `stateCodes` vector of state codes to be converted
- `countryCodes` ISO-3166-1 alpha-2 country codes the state might be found in
- `dataset` name of dataset containing state-level identifiers
Details

For this function to work, you must first run `initializeSpatialData()` to download, convert and install the necessary spatial data.

Value

A vector of English state names or NA.

See Also

convertNaturalEarthAdm1

---

convertEEZCountries

Convert Exclusive Economic Zones Countries Shapefile

Description

A previously downloaded file from [http://www.marineregions.org/downloads.php?unioneezcountry](http://www.marineregions.org/downloads.php?unioneezcountry) is converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

Usage

`convertEEZCountries(dsnPath = NULL, nameOnly = FALSE)`

Arguments

dsnPath directory where EEZCountries .zip file is found
nameOnly logical specifying whether to only return the name without creating the file

Details


Value

Name of the dataset being created.

References


### Description

A SpatialPolygonsDataFrame file is downloaded from the Database of Global Administrative Areas (GADM) database with additional columns of data added. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`. Dataset and file names are generated like this:

```r
paste('gadm_', countryCode, '_', admLevel)
```

Level 0 will return the national outline. Level 1 will give state/province boundaries, etc.

### Usage

```r
convertGADM(countryCode = NULL, admLevel = 0, nameOnly = FALSE)
```

### Arguments

- `countryCode`: ISO-3166-1 alpha-2 country code
- `admLevel`: administrative level to be downloaded
- `nameOnly`: logical specifying whether to only return the name without creating the file

### Value

Name of the dataset being created.

### Note

Not all countries have the same number of levels. Many just have two levels while France has five.

### References

[https://gadm.org/data.html](https://gadm.org/data.html)

### Examples

```r
## Not run:
convertGADM('DE', 1)
## End(Not run)
convertHMSSmoke  

Convert NOAA Hazard Mapping System Smoke Shapefiles

Description

Previously downloaded smoke shapefiles from the NOAA Hazard Mapping System are converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir().

Usage

convertHMSSmoke(dsnPath = NULL, datestamp = NULL, nameOnly = FALSE)

Arguments

dsnPath directory where the HMS Smoke datasets are found
datestamp HMS datestamp in the format "YYYYmmdd"
nameOnly logical specifying whether to only return the name without creating the file

Details

The full set of archived HMS Smoke shapefiles can be downloaded from NOAA with the following command:

wget -R '.*.zip' ftp://satepsanone.nesdis.noaa.gov/FIRE/HMS/GIS/ARCHIVE/hms_smoke*

If no datestamp argument is used, all shapefiles in dsnPath will be converted. In this case, a vector of created dataset names is returned.

Value

Name of the dataset being created.

Note

Data files prior to August 13, 2007 do not contain the vital 'Density' column. For these files, NA will be used in the converted dataframes.

References

http://www.ospo.noaa.gov/Products/land/hms.html

See Also

setSpatialDataDir
convertIndianLands

Convert Indian Lands Shapefile

Description

A shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir()

Usage

convertIndianLands(nameOnly = FALSE)

Arguments

ameOnly logical specifying whether to only return the name without creating the file

Details

The USIndianLands shapefile represents lands administered by the Bureau of Indian Affairs, ie. Indian reservations and is compiled by the National Atlas of the United States of America.

Value

Name of the dataset being created.

References

https://nationalmap.gov/small_scale/atlasftp.html#indlanp
https://nationalmap.gov/small_scale/mld/indlanp.html

See Also

setSpatialDataDir

convertISOCodetable

Convert ISO country code Table to Dataframe

Description

Returns a dataframe version of the country code table with the following columns:

- countryName
- countryCode
- ISO3
- ISONumeric
convertNaturalEarthAdm1

Convert Level 1 (State) Borders Shapefile

Description
Returns a SpatialPolygonsDataFrame for a 1st level administrative divisions

Usage
convertNaturalEarthAdm1(nameOnly = FALSE)

Arguments
nameOnly logical specifying whether to only return the name without creating the file

Details
A state border shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir().

Within the MazamaSpatialUtils package the phrase 'state' refers to administrative divisions beneath the level of the country or nation. This makes sense in the United States. In other countries this level is known as 'province', 'territory' or some other term.

Value
Name of the dataset being created.

References
http://www.naturalearthdata.com/download
http://www.statoids.com/ihasc.html
convertOSMTimezones

See Also
setSpatialDataDir
getState, getStateCode

---

convertOSMTimezones  Convert OSM Timezone Shapefile

Description
A world timezone shapefile is downloaded from https://github.com/evansiroky/timezone-boundary-builder/releases and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir().

Usage
convertOSMTimezones(dsnPath = NULL, nameOnly = FALSE)

Arguments
dsnPath  optional directory where the timezones.shapefile.zip file is found (in case web access isn’t working)
nameOnly  logical specifying whether to only return the name without creating the file

Value
Name of the dataset being created.

Note
There are 86 timezones which have polygons but the associated rows in the dataframe have no data. These timezones also have no countryCode assigned. We hope to rectify this in a future release.

These are the missing timezones:

> OSMTimezones@data$timezone[is.na(OSMTimezones$countryCode)]

[51] "America/St_Thomas" "America/Ste_Vincent" "America/Tortola" "Arctic/Longyearbyen" "Asia/Bahrain" "Asia/Chongqing" "Asia/Harbin" "Asia/Kashgar" "Asia/Kowloon" "Asia/Kunming" "Asia/Mackay" "Asia/Makassar" "Asia/Malaysia" "Asia/Mongolia" "Asia/Novosibirsk" "Asia/Pearl_Harbour" "Asia/Pontianak" "Asia/PnomPenh" "Asia/Qingdao" "Asia/Quezon" "Asia/Riyadh" "Asia/Tokyo" "Asia/Urumqi" "Asia/Ulaanbaatar" "Asia/Vientiane" "Asia/West_Sahara" "Asia/Yakutsk" "Asia/Yerevan" "Australia/Adelaide" "Australia/Brisbane" "Australia/Darwin" "Australia/Douglas" "Australia/Lindeman" "Australia/Lord_Howe" "Australia/Melbourne" "Australia/Perth" "Australia/Sydney" "Europe/Athens" "Europe/Berlin" "Europe/Brussels" "Europe/Chisinau" "Europe/Dublin" "Europe/London" "Europe/Minsk" "Europe/Oslo" "Europe/Rome" "Europe/Valencia" "Europe/Volgograd" "Europe/Vienna" "Europe/Zaporozhye" "Europe/Zurich" "Pacific/Auckland" "Pacific/Chippewa" "Pacific/Guam" "Pacific/Honolulu" "Pacific/Kiritimati" "Pacific/Kwajalein" "Pacific/Marquette" "Pacific/Pago_Pago" "Pacific/Ponape" "Pacific/Puerto_Rico" "Pacific/Port_Moresby" "Pacific/Port_Vila" "Pacific/Sitka" "Pacific/Tasmania" "Pacific/Tongatapu" "Pacific/Tpunk" "Pacific/Truk" "Pacific/Ulasan" "Pacific/Wallis" "Pacific/Yakutat" "Pacific/Yoinsk" "Pacific/Yokohama"
convertSimpleCountries

Convert (Simple) World Borders Shapefile

Description
Returns a SpatialPolygonsDataFrame for a simple world divisions

Usage
convertSimpleCountries(nameOnly = FALSE)

Arguments
nameOnly logical specifying whether to only return the name without creating the file

Details
A world borders shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the package SpatialDataDir which is set with setSpatialDataDir().

This shapefile is a greatly simplified version of the TMWorldBorders shapefile and is especially suited for spatial searches. This is the default dataset used in getCountry() and getCountryCode(). Users may wish to use a higher resolution dataset when plotting.

Value
Name of the dataset being created.

Note
This is a non-exported function used only for updating the package dataset.

References
https://github.com/evansiroky/timezone-boundary-builder/releases

See Also
setSpatialDataDir
convertWikipediaTimezoneTable
convertSimpleCountriesEEZ

Description
Returns a SpatialPolygonsDataFrame for a simple world divisions

Usage
convertSimpleCountriesEEZ(dsnPath = NULL, nameOnly = FALSE)

Arguments
dsnPath directory where EEZCountries .zip file is found
nameOnly logical specifying whether to only return the name without creating the file

Details
A previously downloaded world borders shapefile is converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be saved in the data/ directory. The dataset can be downloaded from http://www.marineregions.org/download_file.php?name=EEZ_land_union_v2_201410.zip by answering the questions and clicking "download".

The SimpleCountriesEEZ shapefile is the same as the EEZCountries shapefile. Polygons for coastal countries include a 200 mile buffer, corresponding to their Exclusive Economic Zones, so this shapefile is especially suited for spatial searches. This is the default dataset used in getCountry() and getCountryCode(). Users may wish to use a higher resolution dataset when plotting.

Value
Name of the dataset being created.

Note
This is a non-exported function used only for updating the package dataset.

References
http://www.marineregions.org/downloads.php
convertSimpleTimezones

Convert SimpleTimezones Shapefile

Description

A world timezone shapefile is downloaded from http://efele.net/maps/tz/world/ and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir().

Usage

convertSimpleTimezones(nameOnly = FALSE)

Arguments

nameOnly logical specifying whether to only return the name without creating the file

Value

Name of the dataset being created.

Note

The following list of timezones have polygons but the associated rows in the dataframe have no data. These timezones also have no countryCode assigned. We hope to rectify this in a future release.

> WorldTimezones@data$timezone[is.na(WorldTimezones@data$countryCode)]
[1] "Europe/Zagreb" "Europe/Vatican" "America/Coral_Harbour"
[4] "Arctic/Longyearbyen" "uninhabited" "America/Kralendijk"
[7] "Europe/Jersey" "Europe/Bratislava" "America/St_Barthelemy"
[10] "Europe/Ljubljana" "Europe/Mariehamn" "Europe/Podgorica"
[16] "Europe/Skopje" "Europe/Sarajevo" "America/Lower_Princes"
[19] "America/Marigot" "Africa/Juba"

This is a non-exported function used only for updating the package dataset.

References

http://efele.net/maps/tz/world/
**convertTerrestrialEcoregions**

**See Also**
- setSpatialDataDir
- convertWikipediaTimezoneTable

**Examples**
```r
## Not run:
setSpatialDataDir(getwd())  # directory
convertSimpleCountries()

## End(Not run)
```

---

**convertTerrestrialEcoregions**

*Convert Terrestrial Ecoregion Shapefile*

**Description**

A shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`

**Usage**

```r
convertTerrestrialEcoregions(nameOnly = FALSE, simplify = TRUE)
```

**Arguments**

- `nameOnly`: logical specifying whether to only return the name without creating the file
- `simplify`: logical specifying whether to create a "_05" version of the file that is simplified to 5%

**Value**

Name of the dataset being created.

**References**

**convertTMWorldBorders**  
*Convert World Borders Shapefile*

**Description**

Returns a SpatialPolygonsDataFrame for world divisions

**Usage**

`convertTMWorldBorders(nameOnly = FALSE)`

**Arguments**

- `nameOnly`: logical specifying whether to only return the name without creating the file

**Details**

A world borders shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Value**

Name of the dataset being created.

**References**


**See Also**

- `setSpatialDataDir`
- `getCountry, getCountryCode`

---

**convertTMWorldBordersSimple**  
*Convert (Simple) World Borders Shapefile*

**Description**

Returns a SpatialPolygonsDataFrame for a simple world divisions

**Usage**

`convertTMWorldBordersSimple(nameOnly = FALSE)`

**Details**

A world borders shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Value**

Name of the dataset being created.

**References**


**See Also**

- `setSpatialDataDir`
- `getCountry, getCountryCode`
**Arguments**

- `nameOnly`  
  logical specifying whether to only return the name without creating the file

**Details**

A world borders shapefile is downloaded and converted to a `SpatialPolygonsDataFrame` with additional columns of data. The resulting file will be created in the package `SpatialDataDir` which is set with `setSpatialDataDir()`.

This shapefile is a greatly simplified version of the TMWorldBorders shapefile and is especially suited for spatial searches. This is the default dataset used in `getCountry()` and `getCountryCode()`.

Users may wish to use a higher resolution dataset when plotting.

**Value**

Name of the dataset being created.

**References**


**See Also**

- `setSpatialDataDir`  
- `getCountry`, `getCountryCode`  

---

**convertUSCensusCBSA**  
*Convert US Core Based Statistical Areas Shapefile*

**Description**

Returns a `SpatialPolygonsDataFrame` for US CBSAs

**Usage**

```r
convertUSCensusCBSA(nameOnly = FALSE, simplify = FALSE)
```

**Arguments**

- `nameOnly`  
  logical specifying whether to only return the name without creating the file

- `simplify`  
  logical specifying whether to create ".05", ".02" and ".01" versions of the file that are simplified to 5%, 2% and 1%
Details

A US Core Based Statistical Areas (CBSA) shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with setSpatialDataDir().

From the Census Bureau:

Metropolitan and Micropolitan Statistical Areas are together termed Core Based Statistical Areas (CBSAs) and are defined by the Office of Management and Budget (OMB) and consist of the county or counties or equivalent entities associated with at least one urban core (urbanized area or urban cluster) of at least 10,000 population, plus adjacent counties having a high degree of social and economic integration with the core as measured through commuting ties with the counties containing the core. Categories of CBSAs are: Metropolitan Statistical Areas, based on urbanized areas of 50,000 or more population; and Micropolitan Statistical Areas, based on urban clusters of at least 10,000 population but less than 50,000 population.

Boundaries are those defined by OMB based on the 2010 Census, published in 2013, and updated in 2015.

Value

Name of the dataset being created.

References


See Also

setSpatialDataDir
getUSCounty

convertUSCensusCongress

Convert US Congressional Districts Shapefile

Description

Returns a SpatialPolygonsDataFrame for US Congressional Districts for the 115th US House of Representatives

Usage

convertUSCensusCongress(nameOnly = FALSE)

Arguments

nameOnly logical specifying whether to only return the name without creating the file
**convertUSCensusCounties**

**Details**

A US congressional district shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Value**

Name of the dataset being created.

**References**

https://www.census.gov/geo/maps-data/data/cbf/cbf_cds.html

**See Also**

`setSpatialDataDir`

---

**Description**

Returns a SpatialPolygonsDataFrame for a US county divisions

**Usage**

`convertUSCensusCounties(nameOnly = FALSE)`

**Arguments**

- `nameOnly` logical specifying whether to only return the name without creating the file

**Details**

A US county borders shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Value**

Name of the dataset being created.

**References**

http://www2.census.gov/geo/tiger/GENZ2013
**convertUSCensusStates**

**Convert US Census State Shapefile**

**Description**

Returns a SpatialPolygonsDataFrame for US States

**Usage**

```r
classifyUSCensusStates(nameOnly = FALSE)
```

**Arguments**

- `nameOnly`: logical specifying whether to only return the name without creating the file

**Details**

A US state shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Value**

Name of the dataset being created.

**References**

[https://www.census.gov/geo/maps-data/data/cbf/cbf_state.html](https://www.census.gov/geo/maps-data/data/cbf/cbf_state.html)

**See Also**

- `setSpatialDataDir`
Description

Previously downloaded shapefiles from the USGS Watershed Boundary Dataset are converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

Usage

```r
convertWBDHUC(dsnPath = NULL, level = 8, extension = "", nameOnly = FALSE, simplify = FALSE)
```

Arguments

- `dsnPath`: directory where the WBD HUC datasets are found
- `level`: character or integer which must be 2, 4, 6, 8, 10, 12 or 14
- `extension`: character extension associated with mapshaper simplified files
- `nameOnly`: logical specifying whether to only return the name without creating the file
- `simplify`: logical specifying whether to create "_02" and "_01" versions of the file that are simplified to 2% and 1%

Details

The full WBD dataset can be downloaded from the USGS with the following command:

```bash
```

Typically, the raw data will be simplified using the command line version of `mapshaper`. (Installation instructions are found at this URL.)

With mapshaper, you can reduce the number of vertices in the polygons, greatly improving the efficiency of spatial searches. Experimentation at the mapshaper website show that a reduction to 1-2 of the original shapefile size still retains the recognizable shape of polygons, removing only the higher order "crenellations" in the polygons.

An example use of mapshaper would be:

```bash
mapshaper WBDHU2.shp --simplify 1
```

A full suite of `.shp`, `.shx`, `.dbf`, `.prj` files will be created for the new name `WBDHU2_02`.

Value

Name of the dataset being created.
References

http://nhd.usgs.gov/wbd.html

See Also

setSpatialDataDir

---

**convertWikipediaTimezoneTable**  
*Convert Wikipedia Timezone Table to Dataframe*

**Description**

Returns a dataframe version of the Wikipedia timezone table with the following columns:

- timezone – Olson timezone
- UTC_offset – hours between local timezone and UTC
- UTC_DST_offset – hours between local timezone daylight savings and UTC
- countryCode – ISO 3166-2 country code
- longitude – longitude of the Olson timezone city
- latitude – latitude of the Olson timezone city

**Usage**

```r
convertWikipediaTimezoneTable()
```

**Details**

Older named timezones from the table which are linked to more modern equivalents are not included in the returned dataframe.

**Value**

Dataframe with 399 rows and 6 columns.

**References**

**convertWorldEEZ**

**Convert World Exclusive Economic Zones Boundaries Shapefile**

**Description**
A world EEZ shapefile is downloaded and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Usage**
```
convertWorldEEZ(nameOnly = FALSE)
```

**Arguments**
- `nameOnly` logical specifying whether to only return the name without creating the file

**Value**
Name of the dataset being created.

**References**
http://www.marineregions.org/downloads.php

**See Also**
- `setSpatialDataDir`
- `getCountry`, `getCountryCode`

---

**convertWorldTimezones**

**Convert Timezone Shapefile**

**Description**
A world timezone shapefile is downloaded from http://efele.net/maps/tz/world/ and converted to a SpatialPolygonsDataFrame with additional columns of data. The resulting file will be created in the spatial data directory which is set with `setSpatialDataDir()`.

**Usage**
```
convertWorldTimezones(nameOnly = FALSE)
```

**Arguments**
- `nameOnly` logical specifying whether to only return the name without creating the file
Value

Name of the dataset being created.

Note

The following list of timezones have polygons but the associated rows in the dataframe have no data. These timezones also have no countryCode assigned. We hope to rectify this in a future release.

> worldtimezones@data$timezone[is.na(worldtimezones$countrycode)]
[1] "Europe/Zagreb"   "Europe/Vatican"   "America/Coral_Harbour"
[4] "Arctic/Longyearbyen" "uninhabited"   "America/Kralendijk"
[7] "Europe/Jersey"    "Europe/Bratislava" "America/St_Barthelemy"
[10] "Europe/Ljubljana" "Europe/Mariehamn" "Europe/Podgorica"
[16] "Europe/Skopje"    "Europe/Sarajevo" "America/Lower_Prices"
[19] "America/Marigot"  "Africa/Juba"

References

http://efele.net/maps/tz/world/

See Also

setSpatialDataDir
countryToCode
convertWikipediaTimezoneTable

countryToCode  

Convert Country Names to Country Codes

Description

Converts a vector of English country names to the corresponding ISO 3166-1 alpha-2 codes.

Usage

countryToCode(countryNames)

Arguments

countryNames  vector of country names to be converted

Value

A vector of ISO 3166-1 alpha-2 codes or NA.
getCountry

Return Country Names at Specified Locations

Description

Uses spatial comparison to determine which country polygons the locations fall into and returns the country name for those polygons.

If allData=TRUE, additional data is returned.

Usage

getCountry(lon, lat, dataset = "SimpleCountriesEEZ",
            countryCodes = NULL, allData = FALSE, useBuffering = FALSE)

Arguments

lon vector of longitudes in decimal degrees
lat vector of latitudes in decimal degrees
dataset name of spatial dataset to use
countryCodes vector of countryCodes
allData logical specifying whether a full dataframe should be returned
useBuffering logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

Value

Vector of country names in English.

References

http://www.naturalearthdata.com/downloads/10m-cultural-vectors/

See Also

SimpleCountries
getSpatialData

Examples

lon <- seq(0,50)
lat <- seq(0,50)
getCountry(lon, lat)
getCountryCode

Return Country ISO Codes at Specified Locations

Description

Uses spatial comparison to determine which country polygons the locations fall into and returns the country code strings for those polygons.

If allData=TRUE, additional data is returned.

Usage

getCountryCode(lon, lat, dataset = "SimpleCountriesEEZ",
               countryCodes = NULL, allData = FALSE, useBuffering = FALSE)

Arguments

lon       vector of longitudes in decimal degrees
lat       vector of latitudes in decimal degrees
dataset   name of spatial dataset to use
countryCodes   vector of countryCodes
allData          logical specifying whether a full dataframe should be returned
useBuffering    logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

Value

Vector of ISO-3166-1 alpha-2 country codes.

References

http://www.naturalearthdata.com/downloads/10m-cultural-vectors/

See Also

SimpleCountries
getSpatialData

Examples

lon <- seq(0,50)
lat <- seq(0,50)
getCountryCode(lon,lat)
getCountryName  

*Return Country Names at Specified Locations*

**Description**

Uses spatial comparison to determine which country polygons the locations fall into and returns the country name for those polygons.

If allData=TRUE, additional data is returned.

**Usage**

```r
getchountryname(lon, lat, dataset = "SimpleCountriesEEZ", countryCodes = NULL, allData = FALSE, useBuffering = FALSE)
```

**Arguments**

- `lon`: vector of longitudes in decimal degrees
- `lat`: vector of latitudes in decimal degrees
- `dataset`: name of spatial dataset to use
- `countryCodes`: vector of countryCodes
- `allData`: logical specifying whether a full dataframe should be returned
- `useBuffering`: logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

**Value**

Vector of country names in English.

**References**


**See Also**

- SimpleCountries
- getSpatialData

**Examples**

```r
lon <- seq(0,50)
lat <- seq(0,50)
getchountryname(lon,lat)
```
getHUC

Return HUCs at Specified Locations

Description

Uses spatial comparison to determine which HUC polygons the locations fall into and returns the
HUC identifier strings for those polygons.

If allData=TRUE, additional data is returned.

Usage

getHUC(lon, lat, SPDF, HUCs = NULL, allData = FALSE)

Arguments

lon vector of longitudes in decimal degrees
lat vector of latitudes in decimal degrees
SPDF spatial polygons dataset of HUCs
HUCs vector of Hydrologic Unit Codes
allData logical specifying whether a full dataframe should be returned

Value

Vector of HUC identifiers.

See Also

getSpatialData

gethucname

Return HUC Names at Specified Locations

Description

Uses spatial comparison to determine which HUC polygons the locations fall into and returns the
HUC names for those polygons.

If allData=TRUE, additional data is returned.

Usage

getHUCNname(lon, lat, dataset = "WBDHU10-ms", HUCs = NULL,
allData = FALSE)
**getPolygonID**

**Arguments**

- `lon` vector of longitudes in decimal degrees
- `lat` vector of latitudes in decimal degrees
- `dataset` name of spatial dataset to use
- `HUCs` vector of Hydrologic Unit Codes
- `allData` logical specifying whether a full dataframe should be returned

**Value**

Vector of HUC names.

**See Also**

`getSpatialData`

---

### Description

Extracts the the vector of unique polygon identifiers from SPDF.

This function is useful when writing code to aggregate data by polygon and calculate per-polygon statistics. Each unique SpatialPolygonsDataFrame will have a different set of data columns but each is guaranteed to have a column named `polygonID` that uniquely identifies each polygon.

This allows us to write code that aggregates by polygon without having to know whether the polygons represent, countries, timezones or HUCs, etc.

### Usage

`getPolygonID(SPDF)`

### Arguments

- `SPDF` spatial polygons dataset of interest

### Value

Vector of polygon identifiers.
getSpatialData

Return Spatial Data Associated with a Set of Locations

Description

All locations are first converted to SpatialPoints objects. The `sp::over()` function is then used to determine which polygon from SPDF each location falls in. The dataframe row associated with each polygon is then associated with each location.

Usage

```r
getSpatialData(lon, lat, SPDF, useBuffering = FALSE, verbose = FALSE)
```

Arguments

- `lon`: vector of longitudes in decimal degrees
- `lat`: vector of latitudes in decimal degrees
- `SPDF`: object of class SpatialPolygonsDataFrame
- `useBuffering`: logical flag specifying the use of location buffering to find the nearest polygon if not target polygon is found
- `verbose`: logical flag controlling detailed progress statements

Details

Occasionally for coastal locations the precise coordinates lie outside the boundaries of a low resolution SpatialPolygonsDataFrame. To account for this any location that remains unassociated after the first pass is then buffered to create a small circle around the original location. All polygons are then checked to see if there is any intersection with the now larger buffered locations. Each point is then checked for an intersecting polygon at the following radii: 1km, 2km, 5km, 10km, 20km, 50km, 100km, 200km. If a buffered location is more than 200km away from any polygon, a value of `NA` (or data frame row with all NAs) is returned for that location.

Missing or invalid values in the incoming `lon` or `lat` vectors result in NAs at those positions in the returned vector or data frame.

Value

Vector or dataframe of data.
getSpatialDataDir  

**getSpatialDataDir**  

*Get Package Data Directory*

**Description**

Returns the package data directory where spatial data is located.

**Usage**

```r
getspatialdatadir()
```

**Value**

Absolute path string.

**See Also**

dataDir
setSpatialDataDir

---

getState  

**getState**  

*Return State Names at Specified Locations*

**Description**

Uses spatial comparison to determine which 'state' polygons the locations fall into and returns the ISO 3166-2 2-character state code strings for those polygons.

Specification of countryCodes limits spatial searching to the specified countries and greatly improves performance.

If allData=TRUE, additional data is returned.

**Usage**

```r
getstate(lon, lat, dataset = "NaturalEarthAdm1", countryCodes = NULL, allData = FALSE, useBuffering = FALSE)
```

**Arguments**

- `lon`  
  vector of longitudes in decimal degrees
- `lat`  
  vector of latitudes in decimal degrees
- `dataset`  
  name of spatial dataset to use
- `countryCodes`  
  vector of country codes
- `allData`  
  logical specifying whether a full dataframe should be returned
- `useBuffering`  
  logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found
getStateCode

Value

Vector of state names in English.

See Also

getSpatialData

Examples

```r
## Not run:
lon <- seq(-140,-90)
lat <- seq(20,70)
getState(lon, lat)

## End(Not run)
```

---

**getStateCode**

*Return State ISO Codes at Specified Locations*

**Description**

Uses spatial comparison to determine which ‘state’ polygons the locations fall into and returns the ISO 3166 2-character state code strings for those polygons.

Specification of `countryCodes` limits spatial searching to the specified countries and greatly improves performance.

If `allData=TRUE`, additional data is returned.

**Usage**

```r
generateStateCode(lon, lat, dataset = "NaturalEarthAdmin1",
                  countryCodes = NULL, allData = FALSE, useBuffering = FALSE)
```

**Arguments**

- `lon`: vector of longitudes in decimal degrees
- `lat`: vector of latitudes in decimal degrees
- `dataset`: name of spatial dataset to use
- `countryCodes`: vector of country codes
- `allData`: logical specifying whether a full dataframe should be returned
- `useBuffering`: logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

**Value**

**getStateName**

*Return State Names at Specified Locations*

**Description**

Uses spatial comparison to determine which 'state' polygons the locations fall into and returns the ISO 3166-2 2-character state code strings for those polygons.

Specification of `countryCodes` limits spatial searching to the specified countries and greatly improves performance.

If `allData`=TRUE, additional data is returned.

**Usage**

```r
getStateName(lon, lat, dataset = "NaturalEarthAdm1",
             countryCodes = NULL, allData = FALSE, useBuffering = FALSE)
```

**Arguments**

- `lon`: vector of longitudes in decimal degrees
- `lat`: vector of latitudes in decimal degrees
- `dataset`: name of spatial dataset to use
- `countryCodes`: vector of country codes
- `allData`: logical specifying whether a full dataframe should be returned
- `useBuffering`: logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

**Value**

Vector of state names in English

**See Also**

`getSpatialData`
getTimezone

Examples

```r
## Not run:
lon <- seq(-140,-90)
lat <- seq(20,70)
getStateName(lon,lat)

## End(Not run)
```

g getTimezone

**Return Olson Timezones at Specified Locations**

**Description**

Uses spatial comparison to determine which timezone polygons the locations fall into and returns the Olson timezone strings for those polygons.

Specification of `countryCodes` limits spatial searching to the specified countries and greatly improves performance.

If `allData=TRUE`, additional data is returned.

**Usage**

```r
getTimezone(lon, lat, dataset = "SimpleTimezones", countryCodes = NULL,
             allData = FALSE, useBuffering = FALSE)
```

**Arguments**

- `lon`: vector of longitudes in decimal degrees
- `lat`: vector of latitudes in decimal degrees
- `dataset`: name of spatial dataset to use
- `countryCodes`: vector of `countryCodes`
- `allData`: logical specifying whether a full dataframe should be returned
- `useBuffering`: logical flag specifying the use of location buffering to find the nearest polygon if not target polygon is found

**Value**

Vector of Olson timezones.

**References**

[http://efele.net/maps/tz/](http://efele.net/maps/tz/)

**See Also**

SimpleTimezones

cSpatialData
getUSCounty

Examples

1 lon <- seq(-120,-60,5)
2 lat <- seq(20,80,5)
3 getTimezone(lon,lat)

getUSCounty

Return US County Name at Specified Locations

Description

Uses spatial comparison to determine which county polygons the locations fall into and returns the county name strings for those polygons.

Specification of stateCodes limits spatial searching to the specified states and greatly improves performance.

If allData=TRUE, additional data is returned.

Usage

getUSCounty(lon, lat, dataset = "USCensusCounties", stateCodes = NULL, allData = FALSE)

Arguments

lon vector of longitudes in decimal degrees
lat vector of latitudes in decimal degrees
dataset name of spatial dataset to use
stateCodes vector of stateCodes used to limit the search
allData logical specifying whether a full dataframe should be returned

Value

Vector of county names in English.

References

http://www.naturalearthdata.com/downloads/10m-cultural-vectors/

See Also

getSpatialData
**Examples**

```r
## Not run:
lon <- seq(-140,-90)
lat <- seq(20,70)
getUSCounty(lon,lat)

## End(Not run)
```

---

**getVariable**  
*Return SpatialDataframe Variable at Specified Locations*

**Description**

Uses spatial comparison to determine which polygons the locations fall into and returns the variable associated with those polygons.

If `allData=TRUE`, the entire dataframe is returned.

**Usage**

```r
getVariable(lon, lat, dataset = NULL, variable = NULL, 
countryCodes = NULL, allData = FALSE)
```

**Arguments**

- `lon` vector of longitudes in decimal degrees
- `lat` vector of latitudes in decimal degrees
- `dataset` name of spatial dataset to use
- `variable` name of dataframe column to be returned
- `countryCodes` vector of countryCodes
- `allData` logical specifying whether a full dataframe should be returned

**Value**

Vector or dataframe.

**See Also**

`getSpatialData`

**Examples**

```r
## Not run:
loadSpatialData('NaturalEarthAdm1')
lon <- seq(0,50)
lat <- seq(0,50)
gVariable(lon,lat,'NaturalEarthAdm1','gns_lang')

## End(Not run)
```
installSpatialData  

Install Spatial Datasets

Description

Install spatial datasets found at url into the directory previously set with setSpatialDataDir().

Usage

installSpatialData(urlBase = "http://mazamascience.com/RData/Spatial", file = "mazama_spatial_files-0.5.tar.gz")

Arguments

urlBase  location of spatial data files
file  name of the tar.gz file containing spatial datasets

Value

Nothing.

iso2ToIso3  

Convert From ISO2 to ISO3 Country Codes

Description

Converts a vector of ISO 3166-1 alpha-2 codes to the corresponding ISO 3166-1 alpha-3 codes.

Usage

iso2ToIso3(countryCodes)

Arguments

countryCodes  vector of country codes to be converted

Value

A vector of ISO3 country codes
iso3ToIso2  
Convert Between ISO3 to ISO2 Country Codes

Description
Converts a vector of ISO 3166-1 alpha-3 codes to the corresponding ISO 3166-1 alpha-2 codes.

Usage
iso3ToIso2(countryCodes)

Arguments
countryCodes  vector of country codes to be converted

Value
A vector of ISO2 country codes

loadSpatialData  
Load Spatial Datasets

Description
Load datasets found in the directory previously set with setSpatialDataDir(). Only files matching pattern will be loaded.

Core datasets available for the package include:
* TMWorldBorders – high resolution country polygons (higher resolution than SimpleCountries)
* NaturalEarthAdm1 – state/province polygons throughout the world
* USCensusCounties – county polygons in the United States
* WorldTimezones – high resolution timezone polygons (higher resolution than SimpleTimezones)

These can be installed with installSpatialData().

Usage
loadSpatialData(pattern = "*")

Arguments
pattern  regular expression used to match filenames

Value
Invisibly returns a vector of spatial dataset names loaded into the global environment.
See Also
setSpatialDataDir
installSpatialData

Description
This package contains code to convert various spatial datasets into .RData files with uniformly named identifiers including:

- countryCode – ISO 3166-1 alpha-2
- countryName – Country name
- timezone – Olson timezone
- longitude – degrees East
- latitude – degrees North
- area – m^2

The parameters listed above will be found in the @data slot of each spatial dataset whose source data has an equivalent field. The only field guaranteed to exist in every dataset is countryCode.

The following additional standards are applied during the data conversion process:

- all spatial data are converted to a purely geographic projection (CRS("+proj=longlat +ellps=GRS80 +datum=NAD83 +units=m +no_defs")
- no duplicated rows in the dataframe (conversion to multi-polygons)
- lowerCamelCase, human readable names replace original parameter names
- redundant, software-internal or otherwise unuseful data columns may be dropped
- parameters may be added to the @data dataframe
- latitude and longitude of polygon centroids may be added

Utility functions allow users to determine the country, state, county and timezones associated with a set of locations, e.g. environmental monitoring sites.

The uniformity of identifiers in the spatial datasets also makes it easy to generate maps with data from any dataset that uses standard ISO codes for countries or states.
setSpatialDataDir Set Package Data Directory

Description
Sets the package data directory where spatial data is located. If the directory does not exist, it will be created.

Usage
setSpatialDataDir(datadir)

Arguments
datadir directory where spatial datasets are created

Value
Silently returns previous value of data directory.

See Also
SpatialDataDir
getSpatialDataDir

SimpleCountries World Country Polygons

Description
SimpleCountries is a simplified world borders dataset suitable for global maps and quick spatial searches. This dataset is distributed with the package and is used by default whenever a dataset with country polygons is required.

Format
A SpatialPolygonsDataFrame with 246 elements and 7 columns of data.

Details
This dataset is equivalent to TMWorldBordersSimple but with fewer columns of data.

See Also
convertTMWorldBordersSimple
SimpleCountriesEEZ

**World Country EEZ Polygons**

**Description**

SimpleCountriesEEZ is a simplified world borders dataset with a 200 mile coastal buffer corresponding to Exclusive Economic Zones, suitable for quick spatial searches. This dataset is distributed with the package and is used by default whenever a dataset with country polygons is required.

**Format**

A SpatialPolygonsDataFrame with 261 elements and 6 columns of data.

**Details**

This dataset is equivalent to EEZCountries but with fewer columns of data.

**See Also**

- convertEEZCountries

---

SimpleTimezones

**World Timezone Polygons**

**Description**

SimpleTimezones is a simplified world timezones dataset suitable for global maps and quick spatial searches. This dataset is distributed with the package and is used by default whenever a dataset with timezone polygons is required.

**Format**

A SpatialPolygonsDataFrame with 1106 elements and 6 columns of data.

**Details**

This dataset is a simplified version of WorldTimezones. It was simplified with [http://mapshaper.org](http://mapshaper.org).

**See Also**

- convertWorldTimezones
**SpatialDataDir**

*Directory for Spatial Data*

**Description**

This package maintains an internal directory location which users can set using `setSpatialDataDir()`. All package functions use this directory whenever datasets are created or loaded.

The default setting when the package is loaded is `getwd()`.

**Format**

Absolute path string.

**See Also**

`getSpatialDataDir`

`setSpatialDataDir`

---

**stateToCode**

*Convert State Names to State Codes*

**Description**

Converts a vector of state names to an ISO 3166-2 two character state codes.

**Usage**

```r
stateToCode(stateNames, countryCodes = NULL, dataset = "NaturalEarthAdmin")
```

**Arguments**

- `stateNames`: state names to be converted
- `countryCodes`: ISO 3166-2 alpha-2 country codes the state might be found in
- `dataset`: name of dataset containing state-level identifiers

**Details**

For this function to work, you must first run `initializeSpatialData()` to download, convert and install the necessary spatial data.

**Value**

A vector of ISO 3166-2 codes or NA.
subsetHUC

See Also

cvtColorNaturalEarthAdm1

Examples

## Not run:
stateToCode("Washington")
stateToCode("Barcelona")
stateToCode("Shandong")

## End(Not run)

subsetHUC  Subset pre-formatted HUC files into smaller groupings.

Description

A SpatialPolygons Dataframe is broken into smaller pieces based on HUC code or state. The SpatialPolygons Dataframe must have the required fields 'stateCode', 'HUC', and 'allStateCodes' and is intended to come from the convertUSGSHUC function. The difference between stateCode and allStateCodes is that stateCode has just one two-digit ISO code while allStateCodes can have more than one. This allows us to include in the subset HUCs where part of the watershed is in the specified state even though the centroid is in a different state.

Usage

subsetHUC(SPDF, parentHUCs = NULL, stateCodes = NULL, allStateCodes = NULL)

Arguments

SPDF  a spatial polygons dataframe created using the convertUSGSHUC function
parentHUCs  a character vector specifying one or more containing HUCs
stateCodes  a character vector specifying one or more containing states
allStateCodes  similar to stateCode, but will also include HUCs who touch the state but whose centroid is in a different state.

Value

a SpatialPolygons Dataframe subsetted to the appropriate specifications.
Index

*Topic conversion
  codeToCountry, 3
  codeToState, 3
  countryToCode, 22
  iso2ToIso3, 35
  iso3ToIso2, 36
  stateToCode, 40

*Topic datagen
  convertEEZCountries, 4
  convertGADM, 5
  convertHMSSmoke, 6
  convertIndianLands, 7
  convertISOCodeTable, 7
  convertNaturalEarthAdm1, 8
  convertOSMTimezones, 9
  convertSimpleCountries, 10
  convertSimpleCountriesEEZ, 11
  convertSimpleTimezones, 12
  convertTMWorldEcoregions, 13
  convertTMWorldBorders, 14
  convertTMWorldBordersSimple, 14
  convertUSCensusCBSA, 15
  convertUSCensusCongress, 16
  convertUSCensusCounties, 17
  convertUSCensusStates, 18
  convertWBDHUC, 19
  convertWikipediaTimezoneTable, 20
  convertWorldEEZ, 21
  convertWorldTimezones, 21
  subsetHUC, 41

*Topic datasets
  SimpleCountries, 38
  SimpleCountriesEEZ, 39
  SimpleTimezones, 39

*Topic environment
  getSpatialDataDir, 29
  installSpatialData, 35
  loadSpatialData, 36
  setSpatialDataDir, 38
  SpatialDataDir, 40

*Topic locator
  getCountry, 23
  getCountryCode, 24
  getCountryName, 25
  getHUC, 26
  getHUCName, 26
  getPolygonID, 27
  getSpatialData, 28
  getState, 29
  getStateCode, 30
  getStateName, 31
  getTimezone, 32
  getUSCounty, 33
  getVariable, 34
  codeToCountry, 3
  codeToState, 3
  convertEEZCountries, 4
  convertGADM, 5
  convertHMSSmoke, 6
  convertIndianLands, 7
  convertISOCodeTable, 7
  convertNaturalEarthAdm1, 8
  convertOSMTimezones, 9
  convertSimpleCountries, 10
  convertSimpleCountriesEEZ, 11
  convertSimpleTimezones, 12
  convertTMWorldEcoregions, 13
  convertTMWorldBorders, 14
  convertTMWorldBordersSimple, 14
  convertUSCensusCBSA, 15
  convertUSCensusCongress, 16
  convertUSCensusCounties, 17
  convertUSCensusStates, 18
  convertWBDHUC, 19
  convertWikipediaTimezoneTable, 20
  convertWorldEEZ, 21
  convertWorldTimezones, 21
  countryToCode, 22
getCountry, 23
getCountryCode, 24
getCountryName, 25
getHUC, 26
getHUCName, 26
getPolygonID, 27
getSpatialData, 28
getSpatialDataDir, 29
getState, 29
getStateCode, 30
getStateName, 31
g getTimezone, 32
g getUSCounty, 33
g getVariable, 34

installSpatialData, 35
iso2ToIso3, 35
iso3ToIso2, 36

loadSpatialData, 36

MazamaSpatialUtils, 37
MazamaSpatialUtils-package
 (MazamaSpatialUtils), 37

setSpatialDataDir, 38
SimpleCountries, 38
SimpleCountriesEEZ, 39
SimpleTimezones, 39
SpatialDataDir, 40
stateToCode, 40
subsetHUC, 41