Package 'MazamaSpatialUtils'

September 28, 2019

Type Package

Version 0.6.4

Title Spatial Data Download and Utility Functions

Author Jonathan Callahan [aut, cre],
Tom Bergamaschi [aut],
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>

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

URL https://github.com/MazamaScience/MazamaSpatialUtils

BugReports https://github.com/MazamaScience/MazamaSpatialUtils/issues

Repository CRAN

Depends R (>= 3.1.0), sp

Imports countrycode, dplyr, geojsonio, lubridate, rgdal, rgeos, rlang, rmapshaper, rvest (>= 0.3.0), shiny, stringr, utils, xml2, magrittr

Suggests knitr, maps, markdown, readr, rmarkdown, testthat

Encoding UTF-8

VignetteBuilder knitr

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Date/Publication 2019-09-28 19:40:03 UTC

R topics documented:

`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 nnames*

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

CONUS *CONUS state codes*

Description

State codes for the 48 contiguous states +DC that make up the CONtinental US.

Usage

```
CONUS
```

Format

A vector with 49 elements

Details

CONUS state codes

`convertCARBAirBasins` *Convert California Air Resources Board basin shapefiles*

Description

Returns a SpatialPolygonsDataFrame for CARB air basins,

The California Air Basins layer is a polygon shapefile coverage representing the 15 California air basins, as defined in state statute and regulation. See the California Health and Safety Code, Section 39606 et seq. and California Code of Regulations, Title 17, Section 60100 et seq.

Air Basins are designated pursuant to California statute and regulation. Air Basins identify regions of similar meteorological and geographic conditions and consideration for political boundary lines, and are related to air pollution and its transport.

Usage

```
convertCARBAirBasins(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%.

Value

Name of the dataset being created.

Note

March, 2004 version.

References

https://www.arb.ca.gov/ei/gislib/gislib.htm

convertEEZCountries *Convert Exclusive Economic Zones countries shapefile*

Description

A previously downloaded file from 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

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".

Value

Name of the dataset being created.

References

http://www.marineregions.org/downloads.php#unioneezcountry

VLIZ (2014). Union of the ESRI Country shapefile and the Exclusive Economic Zones (version 2). Available online at http://www.marineregions.org/. Consulted on 2017-07-20.

Examples

```
## Not run:
convertEEZCountries("~/Data/Spatial/EEZ_land_union_v2_201410.zip")

## End(Not run)
```

convertGACC *Convert Geographic Area Coordination Center geojson, as defined by NIFC*

Description

Returns a SpatialPolygonsDataFrame for Geographic Area Coordination Centers (GACCs)

Usage

```
convertGACC(nameOnly = FALSE, simplify = TRUE)
```

Arguments

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

Details

A GACC 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://hub.arcgis.com/items/72213d9266eb4aefa4403a1bf21dfd61

See Also

setSpatialDataDir

convertGADM *Convert Global Administrative Areas (GADM) Shapefile*

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:

```
paste0('gadm_',countryCode,'_',admLevel)
```

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

Usage

```
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.

Examples

```
## 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

`nameOnly`	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

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/downloads/

http://www.statoids.com/ihasc.html

See Also

setSpatialDataDir

getState, getStateCode

---

convertNWSFireZones *Convert NWS Public Forecast Zones Shapefile*

Description

Returns a SpatialPolygonsDataFrame for NWS weather forecast zones.

Usage

```
convertNWSFireZones(nameOnly = FALSE, simplify = TRUE)
```

Arguments

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

Details

A weather forecast zones 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.

Note

zoneID is the unique identifier, and is the state code followed by zoneNumber.

References

https://www.weather.gov/gis/FireZones

See Also

setSpatialDataDir

`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)]
 [1] "Africa/Addis_Ababa"   "Africa/Asmara"      "Africa/Bamako"      "Africa/Bangui"      "Africa/B
 [6] "Africa/Blantyre"     "Africa/Brazzaville"   "Africa/Bujumbura"    "Africa/Conakry"     "Africa
[11] "Africa/Dar_es_Salaam"  "Africa/Djibouti"     "Africa/Douala"      "Africa/Freetown"     "Africa
[16] "Africa/Harare"      "Africa/Juba"        "Africa/Kampala"     "Africa/Kigali"      "Africa/Kin
[21] "Africa/Libreville"    "Africa/Lome"        "Africa/Luanda"      "Africa/Lubumbashi"    "Africa/L
[26] "Africa/Malabo"      "Africa/Maseru"      "Africa/Mbabane"     "Africa/Mogadishu"    "Africa/N
[31] "Africa/Nouakchott"    "Africa/Ouagadougou"   "Africa/Porto-Novo"    "Africa/Sao_Tome"     "Ameri
[36] "America/Antigua"     "America/Aruba"      "America/Cayman"     "America/Coral_Harbour" "Americ
[41] "America/Grenada"     "America/Guadeloupe"   "America/Kralendijk"   "America/Lower_Princes" "Ame
[46] "America/Montreal"    "America/Montserrat"   "America/St_Barthelemy" "America/St_Kitts"    "Ame
[51] "America/St_Thomas"    "America/St_Vincent"   "America/Tortola"     "Arctic/Longyearbyen"  "Asia
[56] "Asia/Bahrain"       "Asia/Chongqing"     "Asia/Harbin"        "Asia/Kashgar"       "Asia/Kuwait
[61] "Asia/Muscat"        "Asia/Phnom_Penh"     "Asia/Rangoon"       "Asia/Vientiane"     "Atlantic/S
[66] "Europe/Bratislava"    "Europe/Busingen"     "Europe/Guernsey"     "Europe/Isle_of_Man"   "Europ
[71] "Europe/Ljubljana"    "Europe/Mariehamn"    "Europe/Podgorica"    "Europe/San_Marino"    "Europ
[76] "Europe/Skopje"      "Europe/Vaduz"       "Europe/Vatican"     "Europe/Zagreb"      "Indian/Ant
[81] "Indian/Comoro"      "Indian/Mayotte"     "Pacific/Johnston"    "Pacific/Midway"     "Pacific/
[86] "Pacific/Yap"
```

References

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

See Also

setSpatialDataDir

convertWikipediaTimezoneTable

`convertPHDs` *Convert Public Health Districts Shapefile*

Description

Returns a SpatialPolygonsDataFrame for Public Health Districts for Washington, Oregon, Idaho, and California.

Usage

```
convertPHDs(nameOnly = FALSE)
```

Arguments

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

Details

A Public Health Districts 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://mazamascience.com/Shapefiles/PHDs.tgz

See Also

setSpatialDataDir

---

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

http://thematicmapping.org/downloads/

See Also

setSpatialDataDir

getCountry, getCountryCode

`convertSimpleCountriesEEZ` *Convert (Simple) World Borders Shapefile*

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

See Also

setSpatialDataDir

getCountry, getCountryCode

`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$timezone[is.na(WorldTimezones$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"
[13] "Europe/Isle_of_Man"    "Europe/Guernsey"       "Europe/San_Marino"
[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/

See Also

setSpatialDataDir

convertWikipediaTimezoneTable

Examples

```
## Not run:
setSpatialDataDir(getwd()) # directory
convertSimpleCountries()

## End(Not run)
```

---

convertStateLegislativeDistricts

Convert US dtate legislative districts shapefile

Description

a SpatialPolygonsDataFrame for US State Legislative Districts

Usage

```
convertStateLegislativeDistricts(stateCode, house = "Upper",
  nameOnly = FALSE)
```

Arguments

stateCode	ISO 3166-2 alpha-2 state code.
house	Character specifying either "Upper" or "Lower" house.
nameOnly	Logical specifying whether to only return the name without creating the file.

Details

A US State Legislative 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_sld.html

See Also

setSpatialDataDir

`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

```
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

https://www.worldwildlife.org/publications/terrestrial-ecoregions-of-the-world

---

`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

http://thematicmapping.org/downloads/

See Also

setSpatialDataDir

getCountry, getCountryCode

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

Description

Returns a SpatialPolygonsDataFrame for a simple world divisions

Usage

```
convertTMWorldBordersSimple(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.

References

http://thematicmapping.org/downloads/

See Also

setSpatialDataDir

getCountry, getCountryCode

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

Description

Returns a SpatialPolygonsDataFrame for US CBSAs

Usage

```
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()`.

The CBSA datasets was obtained from the following URL:

https://catalog.data.gov/dataset/tiger-line-shapefile-2017-nation-u-s-current-metropolitan-statistical-area-micropolitan-statist

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.

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.

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

convertUSCensusCounties *Convert US County Borders Shapefile*

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

See Also

setSpatialDataDir

getUSCounty

convertUSCensusStates *Convert US Census State Shapefile*

Description

Returns a SpatialPolygonsDataFrame for US States

Usage

```
convertUSCensusStates(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

See Also

setSpatialDataDir

---

convertWBDHUC *Convert USGS hydrologic unit shapefiles*

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

```
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:

```
curl https://prd-tnm.s3.amazonaws.com/StagedProducts/Hydrography/WBD/National/GDB/WBD_National_GDB.z
```

Typically, the raw data will be simplified using mapshaper.

With mapshaper, you can reduce the number of vertices in the polygons, greatly improving the efficiency of spatial searches. Experimentation shows 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:

```
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

`convertWeatherZones` *Convert NWS Public Forecast Zones Shapefile*

Description

Returns a SpatialPolygonsDataFrame for NWS weather forecast zones.

Usage

```
convertWeatherZones(nameOnly = FALSE, simplify = TRUE)
```

Arguments

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

Details

A weather forecast zones 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.

Note

zoneID is the unique identifier, and is the state code followed by zoneNumber.

References

https://www.weather.gov/gis/PublicZones

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

```
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

http://en.wikipedia.org/wiki/List_of_tz_database_time_zones

---

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"
[13] "Europe/Isle_of_Man"    "Europe/Guernsey"       "Europe/San_Marino"
[16] "Europe/Skopje"         "Europe/Sarajevo"       "America/Lower_Princes"
[19] "America/Marigot"       "Africa/Juba"
```

References

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

See Also

setSpatialDataDir

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.

dissolve *Aggregate shapes in a SpatialPolygonsDataFrame*

Description

Aggregate shapes in a spatial polygons dataframe. This is a convenience wrapper for `rmapshaper::ms_dissolve()`

Usage

```
dissolve(SPDF, field = NULL, sum_fields = NULL, copy_fields = NULL,
  ...)
```

Arguments

SPDF	object of class SpatialPolygonsDataFrame
field	proportion of points to retain (0-1; default 0.05)
sum_fields	fields to sum
copy_fields	fields to copy. The first instance of each field will be copied to the aggregated feature
...	arguments passed to `rmapshaper::ms_dissolve()`

Value

A spatial polygons dataframe with aggregated shapes.

Examples

```
regions <- dissolve(SimpleCountries, field = "UN_region", sum_fields = "area")
plot(regions)
regions@data
```

`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

```
getCountryName(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)
getCountryName(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, dataset = "WBDHU10_02", HUCs = 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
`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

```
getHUCName(lon, lat, dataset = "WBDHU10_02", HUCs = 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
HUCs	vector of Hydrologic Unit Codes
allData	logical specifying whether a full dataframe should be returned

Value

Vector of HUC names.

See Also

getSpatialData

---

getPolygonID *Get polygonID from SPDF of interest*

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

```
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 *Get package data directory*

Description

Returns the package data directory where spatial data is located.

Usage

```
getSpatialDataDir()
```

Value

Absolute path string.

See Also

dataDir

setSpatialDataDir

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

```
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

Value

Vector of state names in English.

See Also

getSpatialData

Examples

```
## 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

```
getStateCode(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 ISO-3166-2 alpha-2 state codes.

See Also

getSpatialData

Examples

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

## End(Not run)
```

`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

```
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

Examples

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

## End(Not run)
```

`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

```
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 specyfing 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/

See Also

SimpleTimezones

getSpatialData

Examples

```
lon <- seq(-120,-60,5)
lat <- seq(20,80,5)
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, useBuffering = 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
`useBuffering`	logical flag specifying the use of location buffering to find the nearest polygon if no target polygon is found

Value

Vector of county names in English.

References

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

See Also

getSpatialData

Examples

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

## End(Not run)
```

`getVariable` *Return SPDF 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

```
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

```
## Not run:
loadSpatialData("NaturalEarthAdm1")
lon <- seq(0, 50)
lat <- seq(0, 50)
getVariable(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 from 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 datastes 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

`MazamaSpatialUtils` *Mazama Science spatial data and utility functions.*

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
- stateCode – ISO 3166-2 alpha-2
- 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 +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.

runExample *Run Shiny app example*

Description

This function will run the specified shiny app. By default, the app will open in a new window. By default, it will run in the foreground in your R console, meaning that you have to stop the app to use R again. The default app is "map_app" which requires that the WBDHUC datasets and NaturalEarthAdm1 be downloaded to SpatialDataDir. They can be installed with convertWBDHUC.

Usage

```
runExample(appName = "map_app", ...)
```

Arguments

appName	app to run
...	parameters passed on to runApp().

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.

See Also

convertWorldTimezones

simplify *Simplify SpatialPolygonsDataFrame*

Description

Simplify a spatial polygons dataframe. This is a convenience wrapper for `rmapshaper::ms_simplify()`

Usage

```
simplify(SPDF, keep = 0.05, ...)
```

Arguments

SPDF	object of class SpatialPolygonsDataFrame
keep	proportion of points to retain (0-1; default 0.05)
...	arguments passed to `rmapshaper::ms_simplify()`

Value

A simplified spatial polygons dataframe.

Examples

```
## Not run:
FR <- subset(SimpleCountries, countryCode == 'FR')
par(mfrow = c(3, 3), mar = c(1, 1, 3, 1))
for (i in 9:1) {
  keep <- 0.1 * i
  plot(simplify(FR, keep), main=paste0("keep = ", keep))
}
layout(1)
par(mar = c(5,4,4,2)+.1)

## End(Not run)
```

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

```
stateToCode(stateNames, countryCodes = NULL,
  dataset = "NaturalEarthAdm1")
```

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.

See Also

convertNaturalEarthAdm1

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.

---

summarizeByPolygon *Summarize values by polygon*

Description

Given vectors of longitudes, latitudes and values, this function will summarize given values by spatial polygon using the FUN and return a dataframe with polygon IDs and summary values.

Usage

```
summarizeByPolygon(longitude, latitude, value, SPDF,
  useBuffering = FALSE, FUN, varName = "summaryValue")
```

Arguments

longitude	vector of longitudes
latitude	vector of latitudes
value	vector of values at the locations of interest
SPDF	SpatialPolygonsDataFrame with polygons used for aggregating
useBuffering	passed to MazamaSpatialUtils::getSpatialData()
FUN	function to be applied while summarizing (e.g. mean, max, etc.)
varName	variable name assigned to the summary variable

Value

A dataframe with the same rows as 'SPDF@data' but containing only two columns: 'polygonID' and the summary value.

Note

This function has not been thoroughly tested and should only be included in the package for experimental use only.

US_52 *US state codes*

Description

State codes for the 50 states +DC +PR (Puerto Rico).

Usage

```
US_52
```

Format

A vector with 52 elements

Details

US state codes

US_stateCodes *Dataframe of US state codes*

Description

US_stateCodes contains the following columns of data for the 50 United States plus the District of Columbia:

- `stateCode` – e.g. MT
- `stateName` – e.g. Montana
- `adm1_code` – e.g. USA-3515
- `code_hasc` – e.g. US.MT
- `fips` – e.g. US30

Format

A dataframe with 51 rows and 6 columns of data.

Index