Package 'Rnightlights'

October 14, 2018

Version 0.2.3

Date 2018-10-11

Title Satellite Nightlight Data Extraction

Description Extracts raster and zonal statistics
from satellite nightlight rasters downloaded from the United States
National Oceanic and Atmospheric Administration (<http://www.noaa.gov>)
free data repositories. Both the DMSP-OLS annual and SNPP-VIIRS monthly
nightlight raster data are supported. Satellite nightlight raster tiles are
downloaded and cropped to the country boundaries using shapefiles from the GADM
database of Global Administrative Areas (<http://gadm.org>). Zonal statistics
are then calculated at the lowest administrative boundary for the selected
country and cached locally for future retrieval. Finally, a simple data
explorer/browser is included that allows one to visualize the cached data e.g.
graphing, mapping and clustering regional data.

License GPL-3

Imports cleangeo, compiler, data.table, doSNOW, dplyr, ff, ffbase,
foreach, lubridate, methods, raster, readr, reshape2, rgdal,
rgeos, R.utils, rvest, rworldmap, settings, snow, sp, stats,
stringr, utils, xml2

Suggests dendextend, DT, gdalUtils, ggdendro, ggplot2, leaflet,
plotly, RColorBrewer, shiny, shinydashboard, testthat

SystemRequirements aria2, curl, gdal, wget

RoxygenNote 6.0.1

BugReports https://github.com/chrisvwn/Rnightlights/issues

URL https://github.com/chrisvwn/Rnightlights

NeedsCompilation no

Author Christopher Njuguna [aut, cre, cph]

Maintainer Christopher Njuguna <chris.njuguna@gmail.com>

Repository CRAN

Date/Publication 2018-10-13 23:00:03 UTC

R topics documented:

`addCtryPolyIdx` *Add an index column to all layers of a polygon*

Description

Add an index column to all layers of a polygon

Usage

```
addCtryPolyIdx(ctryCode, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	The ISO3 ctryCode of the country polygon to download
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

None

Examples

```
## Not run:
Rnightlights:::addCtryPolyIdx(ctryCode="KEN")

## End(Not run)
```

addREADME *Add README file to the root data path*

Description

Add README file to the root data path

Usage

```
addREADME(to = getNlDataPath())
```

Arguments

to	The folder to add the README file to

Value

None

Examples

```
  ## Not run:
  Rnightlights:::addREADME()

## End(Not run)
```

allValid *Check if a vector/list of values given is valid as per the given validation function*

Description

Check if a vector/list of values given is valid as per the given validation function. The function will also print a warning showing the values that are invalid. One can stop the warning being printed by wrapping the function in the `suppressWarnings` function.

Usage

```
allValid(testData, testFun, ...)
```

Arguments

testData	The list/vector of values to validate
testFun	The validation function to test each value of testData against
...	Other parameters to pass on to the testFun

Value

TRUE/FALSE

Examples

```
Rnightlights:::allValid(c("KEZ", "UGA", "RWA", "TZA"), Rnightlights:::validCtryCodes)


Rnightlights:::allValid(c("2012", "2015"), validNlPeriods, "OLS.Y")
```

---

allValidCtryAdmLvls *Checks if all ctry admLevels are valid*

Description

Checks if all ctry admLevels are valid

Usage

```
allValidCtryAdmLvls(ctryCode = NULL, admLevels,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of the country of interest
admLevels	character The admLevel(s) to search for
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

logical whether inputted admLevels are valid

Examples

```
## Not run:
validCtryAdmLvls("KEN", "adm0")
#returns "KEN_adm1"

## End(Not run)
```

allValidCtryCodes *Check if all ctryCodes are valid*

Description

Check if all ctryCodes are valid

Usage

```
allValidCtryCodes(ctryCodes)
```

Arguments

ctryCodes	the ISO3 country codes to validate

Value

TRUE/FALSE

Examples

```
Rnightlights:::allValidCtryCodes(c("BDI", "ETH", "KEN", "RWA", "TZA", "UGA")) #returns TRUE

#returns FALSE. "United Arab Emirates" ISO3 code = "ARE"
Rnightlights:::allValidCtryCodes(c("UGA", "UAE"))
```

allValidNlPeriods *Check if all nlPeriods are valid for given nlTypes*

Description

Check if all nlPeriods are valid for given nlTypes

Usage

```
allValidNlPeriods(nlPeriods, nlTypes)
```

Arguments

nlPeriods	vector or list of character vectors The nlPeriods of interest
nlTypes	vector or list of character vectors type of nightlight

Value

logical TRUE/FALSE

Examples

```
Rnightlights:::allValidNlPeriods(c("201401", "201402"),"VIIRS.M")
 #returns TRUE

Rnightlights:::allValidNlPeriods("201203","VIIRS.M")
 #returns FALSE

Rnightlights:::allValidNlPeriods("2012","OLS.Y")
 #returns TRUE
```

`allValidNlTypes` *Checks if all given character strings are valid nlTypes*

Description

Checks if all given character strings are valid nlTypes

Usage

```
allValidNlTypes(nlTypes)
```

Arguments

`nlTypes` `character vector` string The nlTypes

Value

`logical vector` whether the strings are valid nlTypes

Examples

```
Rnightlights:::allValidNlTypes("VIIRS.D") #returns TRUE

Rnightlights:::allValidNlTypes("VIIRS") #returns FALSE
```

createCtryNlDataDF *Initiates the country nightlight dataframe with the country admin level data read from the polygon*

Description

Initiates the country admin level nightlight dataframe with the country admin level data read from the polygon. This includes admin levels, level names and area

Usage

```
createCtryNlDataDF(ctryCode = NULL, admLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	the ISO3 code of the country
admLevel	The country admin level of interest
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

dataframe with the country admin level data

Examples

```
## Not run:
initCtryNlData <- Rnightlights:::createCtryNlDataDF("KEN")
 #returns a data frame

## End(Not run)
```

createCtryStruct *Save ctry admin structure to text file for faster access*

Description

Save ctry admin structure to text file for faster access

Usage

```
createCtryStruct(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	The ISO3 ctryCode of the country
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

None

Examples

```
## Not run:
  Rnightlights:::createCtryStruct("KEN")

## End(Not run)
```

---

createNlDataDirs *Create required data subdirectories in the root data path*

Description

Create required data subdirectories in the root data path

Usage

```
createNlDataDirs()
```

Value

None

Examples

```
  ## Not run:
  Rnightlights:::createNlDataDirs()

## End(Not run)
```

createNlTilesSpPolysDF

Creates a tile Spatial Polygons DataFrame from the "nlTiles" dataframe

Description

Creates a Spatial Polygons DataFrame from the "nlTiles" dataframe of VIIRS tiles

Usage

```
createNlTilesSpPolysDF()
```

Value

TRUE/FALSE

Examples

```
tilesSpPolysDFs <- Rnightlights:::createNlTilesSpPolysDF()
```

ctryCodeToName

Convert a country ISO3 code to the full name

Description

Convert a country ISO3 code to the full name. Exposes the rworldmap function isoToName(ctryCode). #rworldmap::isoToName can resolve 2-letter ctryCodes but we only want 3-letter ISO3 codes. With no parameters returns a list of ctryCodes and their corresponding names as given by rworldMap::getMap@data

Usage

```
ctryCodeToName(ctryCodes)
```

Arguments

ctryCodes	The country Codes to search for

Value

Character The full country name if the ctryCode is found. If ctryCode is not supplied then return a list of all country codes and their corresponding names

Examples

```
ctryCodeToName("KEN") #returns Kenya

ctryCodeToName("ARE") #returns United Arab Emirates

ctryCodeToName("USA") #returns United States of America

ctryCodeToName("JAM") #returns Jamaica
```

`ctryNameToCode` *Convert country names to ISO3 codes*

Description

Convert country names to ISO3 codes. Searches the rworldmap map data. With no parameters returns a list of ctryNames and their corresponding codes as given by rworldMap

Usage

```
ctryNameToCode(ctryNames)
```

Arguments

`ctryNames`	Character vector common names of countries to convert

Value

Character ISO3 ctryCode if found else NA

Examples

```
ctryNameToCode("kenya") #returns "KEN"

ctryNameToCode("ken") #returns "KEN"

ctryNameToCode("jamaica") #returns JAM
```

`ctryShpLyrName2Num` *Get the integer number of the layer.*

Description

Get the integer number of the layer. Is the last character in the name and is a digit. E.g. for the 3rd layer in Kenya shapefile polygon named "KEN_adm3" the layer number is `3`. The lowest layer number is `0`

Usage

```
ctryShpLyrName2Num(ctryCode = NULL, layerName,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryCode`	The ctryCode to process
`layerName`	- the name of the polygon layer
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

Integer layer number

Examples

```
## Not run:
  Rnightlights:::ctryShpLyrName2Num("KEN", "KEN_adm1") #returns 1

## End(Not run)
```

`deleteNlDataCol` *Delete an aggregate nightlight data column in a country nightlights dataframe*

Description

Delete an aggregate nightlight data column in a country nightlights dataframe. The number of elements in the vector MUST match the number of rows in the country dataframe.

Usage

```
deleteNlDataCol(ctryCode = NULL, admLevel, nlType, nlPeriod, statType,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	country with the data column to remove
admLevel	admLevel to process
nlType	the type of nightlight data
nlPeriod	the nlPeriod that the dataCol belongs to
statType	the stat which produced the dataCol vector
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Examples

```
## Not run:
ctryNlDataDF <- Rnightlights:::deleteNlDataCol(ctryNlDataDF,
     "VIIRS.M", "201409", "sum")

## End(Not run)

## Not run:
Rnightlights:::deleteNlDataCol(ctryNlDataDF,
    "OLS.Y", "2012", "mean")

## End(Not run)
```

---

dnldCtryPoly *Download a country's polygon shapefile from* http://gadm.org

---

Description

Download a country's polygon shapefile from http://gadm.org

Usage

```
dnldCtryPoly(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL, downloadMethod = pkgOptions("downloadMethod"))
```

Arguments

ctryCode	The ISO3 ctryCode of the country polygon to download
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip
downloadMethod	The method used to download polygons

Value

TRUE/FALSE Success/Failure of the download

Examples

```
## Not run:
Rnightlights:::dnldCtryPoly("KEN")

## End(Not run)
```

`downloadNlTiles` *Download the listed tiles for a given nlType in a given nlPeriod*

Description

Download the listed tiles for a given nlType in a given nlPeriod

Usage

```
downloadNlTiles(nlType, nlPeriod, tileList)
```

Arguments

`nlType`	character The nightlight type
`nlPeriod`	character The nlPeriod to process in the appropriate format
`tileList`	integer vector or character vector of digits containing valid tile numbers as obtained by tileName2Idx for VIIRS. Ignore for nlType=="OLS"

Value

TRUE/FALSE if the download was successful

Examples

```
#download VIIRS tiles for "KEN" which are tiles 2 and 5 for the specified
    #time periods
## Not run:
Rnightlights:::downloadNlTiles("VIIRS.M", "201401", c(2, 5))

## End(Not run)

#same as above but getting the tileList automatically
## Not run:
Rnightlights:::downloadNlTiles(nlType="VIIRS.M",
    nlPeriod="201401",
    tileList=Rnightlights:::getCtryTileList(ctryCodes="KEN",
        nlType="VIIRS.M")
```

```
)

## End(Not run)

#returns TRUE if the download was successful or tile is cached locally
```

`downloadNlTilesOLS` *Download OLS nightlight tile*

Description

Download OLS nightlight tile

Usage

```
downloadNlTilesOLS(nlPeriod, downloadMethod = pkgOptions("downloadMethod"))
```

Arguments

`nlPeriod`	the nlPeriod of the tile
`downloadMethod`	The method to use for download.

Value

TRUE/FALSE Whether the download was successful

Examples

```
## Not run:
if(Rnightlights:::downloadNlTilesOLS("201405"))
  print("download successful")

## End(Not run)
```

`downloadNlTilesVIIRS` *Download VIIRS nightlight tile*

Description

Download VIIRS nightlight tile

Usage

```
downloadNlTilesVIIRS(nlPeriod, tileNum,
  downloadMethod = pkgOptions("downloadMethod"), nlType)
```

Arguments

nlPeriod	the nlPeriod of the tile to download
tileNum	the index of the tile as given by getNlTiles
downloadMethod	The method to use for download.
nlType	A character string of nlType

Value

TRUE/FALSE Whether the download was successful

Examples

```
## Not run:
if(Rnightlights:::downloadNlTilesVIIRS("201401", "1"))
  print("download successful")

## End(Not run)
```

---

existsCtryNlData *Check if VIIRS nightlight stats exist locally*

Description

Check if VIIRS nightlight data for the country exists in the country nightlight data file. First checks if the country nightlight data file exists.

Usage

```
existsCtryNlData(ctryCode = NULL, admLevel, nlTypes, nlPeriods, nlStats,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ISO3 code of the country
admLevel	character string The country admin level of interest
nlTypes	character The nlTypes
nlPeriods	character The nlPeriods
nlStats	character The nlStats to check for
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

TRUE/FALSE

Examples

```
Rnightlights:::existsCtryNlData("KEN", "KEN_adm0", "VIIRS.M","201401", "sum")
```

`existsCtryNlDataFile` *Check if a country admin level data file exists*

Description

Check if a country admin level data file exists. Stats are calculated and stored at the admin level of a country, hence, a country could have as many files as admin levels.

Usage

```
existsCtryNlDataFile(ctryCode = NULL, admLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryCode`	the ISO3 country code
`admLevel`	The country admin level of interest.
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

TRUE/FALSE

Examples

```
ctryCode <- "KEN"
admLevel <- "KEN_adm0"
message(Sys.time(), ": Data file for ", ctryCode,
    ifelse(Rnightlights:::existsCtryNlDataFile(ctryCode, admLevel),
        " FOUND", " NOT FOUND"))
```

existsCtryPoly *Checks if a country polygon exists*

Description

Checks if a country polygon exists

Usage

```
existsCtryPoly(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of the country of interest
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Examples

```
## Not run:
existCtryPoly("KEN")
#returns TRUE/FALSE

## End(Not run)
```

existsPolyFnamePath *Check if the decompressed country polygon has been downloaded and stored in the polygon folder*

Description

Check if the decompressed country polygon has been downloaded and stored in the polygon folder

Usage

```
existsPolyFnamePath(ctryCode = NULL,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	The ctryCode to process
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

TRUE/FALSE

Examples

```
Rnightlights:::existsPolyFnamePath("KEN")
 #returns TRUE/FALSE
```

`existsPolyFnameZip` *Check if the compressed country polygon has been downloaded and stored in the polygon folder*

Description

Check if the compressed country polygon has been downloaded and stored in the polygon folder

Usage

```
existsPolyFnameZip(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	The ctryCode of interest
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

TRUE/FALSE

Examples

```
Rnightlights:::existsPolyFnameZip("KEN")
 #returns TRUE/FALSE
```

`exploreData` *Run a web application to explore the processed nightlight data cached locally*

Description

Run a web application to perform some exploratory data analysis. The application written in shiny either loads demo data or the data processed and cached locally.

Usage

```
exploreData()
```

Value

None

Examples

```
## Not run:
exploreData()

## End(Not run)
```

`fnAggRadGdal` *Calculate zonal statistics using GDAL*

Description

Calculate zonal statistics using GDAL. Alternative to fnAggRadRast and faster. Modified from http://www.guru-gis.net/efficient-zonal-statistics-using-r-and-gdal/

Usage

```
fnAggRadGdal(ctryCode, admLevel, ctryPoly, nlType, nlPeriod,
  nlStats = pkgOptions("nlStats"), gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	character string the ISO3 country code to be processed
admLevel	character string The admin level to process. Should match the ctryPoly given but no checks are made currently.
ctryPoly	Polygon the loaded country polygon layer
nlType	the nlType of interest
nlPeriod	character string the nlPeriod to be processed
nlStats	character vector The stats to calculate
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

data.frame of polygon attributes and the calculated stats, one column per stat

Examples

```
#read the Kenya polygon downloaded from GADM and load the lowest admin level (ward)
## Not run:
ctryPoly <- readCtryPolyAdmLayer(ctryCode="KEN",
    Rnightlights:::getCtryShpLowestLyrNames(ctryCodes="KEN"))

#calculate the sum of radiances for the wards in Kenya
sumAvgRadRast <- Rnightlights:::fnAggRadGdal(ctryCode="KEN", ctryPoly=ctryPoly,
    nlType="VIIRS.M", nlPeriod="201401", nlStats=c("sum","mean"))

## End(Not run)
```

---

fnAggRadRast *Calculate statistics on a nightlight raster that fall within a polygon*

Description

Calculate stats on the radiance of the pixels in a nightlight raster that fall within a polygon and its subpolygons using the raster package. Given a country polygon with subpolygons representing lower admin levels, it will crop and mask the raster to each subpolygon and calculate the total radiance for the polygon and return a vector of total radiances that matches the subpolygons

Usage

```
fnAggRadRast(ctryPoly, ctryRastCropped, nlType, nlStats, custPolyPath = NULL)
```

Arguments

ctryPoly	The polygon of the admin level/region of interest. In general is a country polygon with sub-regions usually the lowest known admin level as given by the GADM polygons.
ctryRastCropped	The raster containing nightlight radiances to sum. Usually will have already be cropped to the country outline
nlType	Character vector The nlType to process
nlStats	The statistics to calculate
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

data.frame of polygon attributes and the calculated stats, one column per nlStat

Examples

```
#read the Kenya polygon downloaded from GADM and load the lowest admin level (ward)
## Not run:
ctryPoly <- readCtryPolyAdmLayer(ctryCode="KEN",
    Rnightlights:::getCtryShpLowestLyrNames(ctryCodes="KEN"))

# the VIIRS nightlight raster cropped earlier to the country outline
ctryRastCropped <- raster::raster(Rnightlights:::getCtryRasterOutputFnamePath(ctryCode="KEN",
    nlType="VIIRS.M", nlPeriod="201401"))

#calculate the sum of radiances for the wards in Kenya
sumAvgRadRast <- Rnightlights:::fnAggRadRast(ctryPoly=ctryPoly,
    ctryRastCropped=ctryRastCropped, nlType="VIIRS.M", nlStats=c("sum","mean"))

## End(Not run)
```

---

getAllGadmVersions *Return a vector of GADM versions*

Description

Return a vector of GADM versions

Usage

```
getAllGadmVersions()
```

Value

character vector valid GADM versions

Examples

```
## Not run:
Rnightlights:::getAllGadmVersions()

## End(Not run)
```

`getAllNlCtryCodes` *Get all known valid ISO3 country codes*

Description

Get a list of all known valid ISO3 country codes.

Usage

```
getAllNlCtryCodes(omit = "none")
```

Arguments

`omit`	The ctryCodes to exclude from processing based on observed behaviour. The option can take the following values: • `missing` ctryCodes that are in `rworldmap` but not on `GADM` • `long` ctryCodes that take very long to process using 'rast' options. May improve using more processors or using 'gdal' options. • `error` ctryCodes whose polygons have caused processing to crash in tests. Not extensively tested and may work fine on other systems. • `all` Omit a combination of all the above options • `none` Do not omit any ctryCodes

Value

character vector of country codes

`getAllNlPeriods` *Generate a list of all possible nlPeriods for a given nlType*

Description

Generate a list of all possible nlPeriods for a given nlType

Usage

```
getAllNlPeriods(nlTypes)
```

Arguments

nlTypes	types of nightlights to process

Value

a named list of character vector nlPeriods

Examples

```
getAllNlPeriods("OLS.Y")
 #returns a vector of all years from 1994 to 2013

getAllNlPeriods("VIIRS.M")
 #returns a vector of all yearMonths from 201401 to present

getAllNlPeriods(c("OLS.Y", "VIIRS.Y"))
 #returns a list with 2 named vectors, one for each annual nlType
```

---

getAllNlTypes *Lists supported nightlight types*

Description

Lists supported nightlight types. The names are in the form "nlType.avgPeriod" where avgPeriod is the period over which the raw data is averaged i.e. Daily (D), Monthly (M) and Yearly (Y). Currently the main nlTypes are "OLS" and "VIIRS". OLS only has yearly averaged data while VIIRS has daily, monthly and yearly e.g. VIIRS daily = "VIIRS.D", OLS yearly = "OLS.Y"

Usage

```
getAllNlTypes()
```

Value

character vector of supported nlTypes

Examples

```
getAllNlTypes()
```

`getCtryNlData` *Returns nightlight statistics for the given ctryCode and nlType in the given nPeriods*

Description

Returns nightlight data for the given ctryCode and nlStats in the given nlPeriods and of the specified nlType. Note that getCtryNldata only processes one ctryCode at a time. `ignoreMissing` plays a significant role here. It can take 3 values:

- NULL (default) only return data if found for all nlPeriods and all nlStats provided otherwise return NULL.
- TRUE return any partial data that is found for the provided nlPeriods and nlStats. Ignore any missing data.
- FALSE return all data that is found and call `processNlData` to download and process any missing nlPeriods and nlStats.

Farther, if `nlPeriods` is missing, it is assigned values based on the value of ignoreMissing. If ignoreMissing is FALSE, nlPeriods is assigned all existing nlPeriods to date. This is the equivalent of retrieving all nightlight data for the given country and stats. If ignoreMissing is TRUE or NULL then the existing data is returned.

Usage

```
getCtryNlData(ctryCode = NULL, admLevel, nlTypes, nlPeriods,
  nlStats = pkgOptions("nlStats"), ignoreMissing = NULL,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL,
  downloadMethod = pkgOptions("downloadMethod"),
  cropMaskMethod = pkgOptions("cropMaskMethod"),
  extractMethod = pkgOptions("extractMethod"), source = "local", ...)
```

Arguments

`ctryCode`	the ISO3 code of the country. Only 1 country can be processed at a time.
`admLevel`	The country admin level of interest. Only 1 admLevel can be processed at a time.
`nlTypes`	a vector of nlTypes. The nightlight types to process.
`nlPeriods`	a vector of nlPeriods. Must be appropriate nlPeriods for the nlType.
`nlStats`	a vector of nlStats. If not supplied defaults to all nlStats as listed in pkgOptions("nlStats").
`ignoreMissing`	controls how the function behaves if any data is not found in the data file. • NULL (default) only return data if found for ALL nlPeriods and ALL stats provided otherwise return NULL • TRUE return any partial data that is found for the provided nlPeriods and stats. Ignore any missing data

	• FALSE return all data that is found and call `processNlData` to download and process any missing nlPeriods and stats
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip
`downloadMethod`	The method used to download polygons
`cropMaskMethod`	The method used to crop and mask the satellite raster
`extractMethod`	The method used to extract data and perform calculations on the satellite raster
`source`	"local" or "remote" Whether to download and process the data locally or to download the pre-processed data from a remote source/repo
`...`	other arguments

Value

dataframe of data for one country in one nlType in one or multiple nlPeriods

Examples

```
#NOTE: missing stats implies all stats as given by pkgOptions("nlStats")

#long running examples which also require large downloads
## Not run:
getCtryNlData("KEN", "KEN_adm0", "VIIRS.M", ignoreMissing=NULL)
    #returns either all requested data if it exists i.e. all nlPeriods
    #and all nlStats for KEN otherwise NULL

## End(Not run)

## Not run:
getCtryNlData("KEN", "KEN_adm0", "OLS.Y", ignoreMissing=TRUE)
    #Returns all requested data if it exists i.e. all nlPeriods and all
    #nlStats for KEN but omits any missing data

## End(Not run)

## Not run:
getCtryNlData(ctryCode="KEN", "KEN_adm0", "VIIRS.Y", ignoreMissing=FALSE)
    #Returns all requested data i.e. all nlPeriods and all
    #nlStats for KEN. All missing data will be downloaded and processed

## End(Not run)

## Not run:
getCtryNlData("KEN", "KEN_adm0", "VIIRS.M", nlPeriods=c("existingNlPeriod", "missingNlPeriod"),
    nlStats=c("sum", "unknownStat"), ignoreMissing=NULL)
    #Returns NULL due to missingNlPeriod and unknownStat not already existing
    #(ignoreMissing=NULL returns all data if exists or if any is missing returns NULL)

## End(Not run)
```

```
## Not run:
getCtryNlData("KEN", "KEN_adm0", "VIIRS.D", nlPeriods=c("existingNlPeriod", "missingNlPeriod"),
    nlStats=c("existingStat", "missingStat"), ignoreMissing=TRUE)
   #Returns existingStat for existingNlPeriods omits missingNlPeriod and missingStat
   #(ignoreMissing=TRUE returns only existing data)

## End(Not run)

## Not run:
getCtryNlData("KEN", "KEN_adm0", "VIIRS.M", nlPeriods=c("existingNlPeriod", "missingNlPeriod"),
    nlStats=c("sum", "unknownStat"), ignoreMissing=FALSE)
    #Runs processNlData for missingStat in "missingNlPeriod" and returns
    #"existingStat" and "missingStat" for both "existingNlPeriod" and
    #"missingNlPeriod"
    #(ignoreMissing=FALSE must return all data: forces processing of any missing)

## End(Not run)
```

`getCtryNlDataColName` *Construct the name of a nightlight data column given the nightlight type and nlPeriod*

Description

Construct the name of a nightlight data column given the nightlight type and nlPeriod Used in creating and retrieving data columns from the nightlight data file

Usage

```
getCtryNlDataColName(nlPeriod, nlStat, nlType)
```

Arguments

`nlPeriod`	character vector The nlPeriod to process
`nlStat`	character vector The stat to be stored in the column
`nlType`	character vector The type of nightlight

Value

character string

Examples

```
Rnightlights:::getCtryNlDataColName("201612", "sum", nlType="VIIRS.M")
```

`getCtryNlDataFname` *Construct the name of the country data file.*

Description

Construct the name of the data file. This function can be altered to name the file as required and consistently retrieve the name. Used in the function getCtryNlDataFnamePath to concat the directory path and this filename. Currently all nlTypes are stored in one file. Can be altered to separate VIIRS and OLS data files for example.

Usage

```
getCtryNlDataFname(ctryCode = NULL, admLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryCode`	The ctryCode of interest
`admLevel`	The country admin level of interest
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

Character filename of the country data file

Examples

```
ctryCode <- "KEN"
admLevel <- "KEN_adm0"
Rnightlights:::getCtryNlDataFname(ctryCode, admLevel)
#returns string of name of the ctry data file
```

`getCtryNlDataFnamePath`

Construct the full path to save the file containing the country data

Description

Construct the full path to save the file containing the country data. Note it does not indicate if the file exists

Usage

```
getCtryNlDataFnamePath(ctryCode = NULL, admLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character string The ctryCode of interest
admLevel	character string The admin level of interest
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

Character string the full path to the data file of a country admin level

Examples

```
#get the full path to the file containing data for KEN counties
getCtryNlDataFnamePath("KEN", "KEN_adm0")

#@export only due to exploreData() shiny app
```

---

getCtryPolyAdmLevelNames

Get the list of admin level names in a polygon shapefile

Description

Get the list of admin level names in a polygon shapefile. It returns all official names starting from 1 to the specified `lowestAdmLevel`. If not `lowestAdmLevel` is not specified, all admin level names are returned

Usage

```
getCtryPolyAdmLevelNames(ctryCode = NULL, lowestAdmLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of the country of interest
lowestAdmLevel	integer The lowest admin level number to return
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character vector of admin level names

Examples

```
## Not run:
Rnightlights:::getCtryPolyAdmLevelNames("KEN")
#returns vector [1] "County"       "Constituency" "Ward"
#if KEN shapefile exists otherwise errors

## End(Not run)
```

`getCtryPolyUrl` *Get the GADM url from which to download country polygons*

Description

Get the url from which to download country polygons. Polygons are downloaded from http://gadm.org. This provides the url to the zipped ESRI Shapefile which when decompressed contains a directory with the different country admin level boundary files. A sample url returned for Afghanistan: http://biogeo.ucdavis.edu/data/gadm2.8/shp/AFG_adm_shp.zip

Usage

```
getCtryPolyUrl(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	character string The ctryCode of interest
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

Character string url of the zipped ESRI shapefile for the ctryCode

Examples

```
ctryCode <- "KEN"
Rnightlights:::getCtryPolyUrl(ctryCode)
 #returns url for the zipped country ESRI shapefile
```

getCtryRasterOutputFname

Constructs the name of the output raster

Description

Constructs the name of the output raster

Usage

```
getCtryRasterOutputFname(ctryCode, nlType, nlPeriod,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	the ctryCode of interest
nlType	the nlType of interest
nlPeriod	the nlPeriod of interest
gadmVersion	The GADM version to use
custPolyPath	The path to a custom polygon as an alternative to using GADM polygons

Value

Character the name of country raster for a country and a given nlType and nlPeriod

Examples

```
Rnightlights:::getCtryRasterOutputFname("KEN","VIIRS.M", "201412")
```

getCtryRasterOutputFnamePath

Get the full path to the file where the cropped VIIRS country raster is stored.

Description

Get the full path to the file where the cropped VIIRS country raster is stored. This file is created when processing the country before extracting the data. It can be used to re-process a country much faster

Usage

```
getCtryRasterOutputFnamePath(ctryCode, nlType, nlPeriod,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	the ctryCode of interest
nlType	the nlType of interest
nlPeriod	the nlPeriod of interest
gadmVersion	The GADM version to use
custPolyPath	The path to a custom polygon as an alternative to using GADM polygons

Value

Character full path to the cropped VIIRS country raster for a country and a given year and month

Examples

```
## Not run:
getCtryRasterOutputFnamePath("KEN","VIIRS.M", "201412")

## End(Not run)

#export for exploreData() shiny app
```

---

getCtryShpAllAdmLvls *Get all the admLevels in a country*

Description

Get all the admLevels in a country

Usage

```
getCtryShpAllAdmLvls(ctryCodes = NULL,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCodes	character The ctryCode of the country of interest
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

logical whether inputted admLevels are valid

Examples

```
## Not run:
getCtryShpAllAdmLvls("KEN")
#returns "KEN_adm1"

## End(Not run)
```

`getCtryShpLowestLyrNames` *Get the name of the lowest ctry admin level*

Description

Get the name of the lowest ctry admin level

Usage

```
getCtryShpLowestLyrNames(ctryCodes = NULL,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryCodes`	`character` ctryCodes the ctryCodes of interest
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

character string The name of the lowest admin level

`getCtryShpLyrNames` *Get the standard names of polygon layers in a country*

Description

Get the standard name of a polygon layer for a country. Used to refer to a polygon layer by name i.e. for CTRYCODE & lyrNum="0": lyrName="CTRYCODE_adm0", lyrNum="1": lyrName="KEN_adm1". Note this is different from the official country administration level name.

Usage

```
getCtryShpLyrNames(ctryCodes = NULL, lyrNums, dnldPoly,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCodes	the ISO3 codes for the countries
lyrNums	the layer numbers starting from 0 = country level, 1 = first admin level
dnldPoly	logical If the country polygon doesn't exist should we download it?
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

Character layer name

Examples

```
## Not run:
#requires KEN polygon shapefile to exist in the polygons folder
getCtryShpLyrNames("KEN","1")
  #returns "KEN_adm1"

## End(Not run)

#export only due to exploreData() shiny app
```

---

getCtryStructAdmLevelNames

Get the list of admin level names in a polygon shapefile

Description

Get the list of admin level names in a polygon shapefile. It returns all official names starting from 1 to the specified lowestAdmLevel. If not lowestAdmLevel is not specified, all admin level names are returned

Usage

```
getCtryStructAdmLevelNames(ctryCode = NULL, lowestAdmLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of the country of interest
lowestAdmLevel	integer The lowest admin level number to return
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character vector of admin level names

Examples

```
## Not run:
Rnightlights:::getCtryPolyAdmLevelNames("KEN")
#returns vector [1] "County"       "Constituency" "Ward"
#if KEN shapefile exists otherwise errors

## End(Not run)
```

---

getCtryStructFname *Construct the name for the country struct file*

Description

Construct the name for the country struct file

Usage

```
getCtryStructFname(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	The ISO3 ctryCode of the country
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character string The filename

Examples

```
Rnightlights:::getCtryStructFname("KEN")
```

`getCtryStructFnamePath` *Construct the full path to the country struct file*

Description

Construct the full path to the country struct file

Usage

```
getCtryStructFnamePath(ctryCode = NULL,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryCode`	The ISO3 ctryCode of the country
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

character string The file path

Examples

```
Rnightlights:::getCtryStructFnamePath("KEN")
```

`getCtryTileList` *Returns a list of VIIRS nightlight tiles that a country or countries intersects with*

Description

Given a list of countries, this function will provide alist of VIIRS nightlight tiles that intersect with them. This helps in processing multiple countries by determining which nightlight tiles are required for processing by allowing the download of all required tiles before processing. Note all VIIRS_* nlTypes have the same nlTiles.

Usage

```
getCtryTileList(ctryCodes, nlType, omitCountries = "none")
```

Arguments

ctryCodes	character vector of country codes to process
nlType	character string The nlType of interest
omitCountries	countries to exclude from processing. This is helpful when the number of countries to exclude is smaller than the number to process e.g. when one wants to process all countries and exclude countries that take long to process i.e. omitCountries = "long"

Value

TRUE/FALSE

Examples

```
Rnightlights:::getCtryTileList(ctryCodes=c("BDI", "KEN", "RWA", "TZA", "UGA"),
    nlType="VIIRS.M", omitCountries="none")

#only 1 tile for OLS
Rnightlights:::getCtryTileList(ctryCodes=c("BDI", "KEN", "RWA", "TZA", "UGA"),
    nlType="OLS.Y", omitCountries="none")
    #returns "DUMMY"
```

---

getNlDataPath *Gets the root path to the file directory"*

Description

Gets the root path to the file directory"

Usage

```
getNlDataPath()
```

Value

Returns the folder containing the root of the current data path as a @character string.

See Also

To set the directory where package data files are stored, see @see "setNlDataPath".

Examples

```
print(getNlDataPath())
```

getNlDir — *Get the paths to the various data locations*

Description

Get the paths to the various locations of nightlights data generated by the Rnightlights package. These correspond to the various "dir..." options in the pkgOptions settings

Usage

```
getNlDir(dirName)
```

Arguments

dirName	character vector The name of the directory to retrieve

Examples

```
getNlDir("dirRasterOutput")

getNlDir("dirNlTiles")

getNlDir("dirPolygon")

getNlDir("dirZonals")
```

getNlTifLclNameOLS — *Constructs the filename used to save/access the decompressed OLS .tif file*

Description

Constructs the filename used to save/access the decompressed OLS .tif file

Usage

```
getNlTifLclNameOLS(nlPeriod)
```

Arguments

nlPeriod	the nlPeriod in which the tile was created

Value

a character vector filename of the .tif VIIRS tile

Examples

```
#using default dirNlTiles
## Not run:
Rnightlights:::getNlTifLclNameOLS("2004")
 #returns "OLS_2004.tif"

## End(Not run)
```

`getNlTiles` *Create mapping of nightlight tiles*

Description

Creates a data.frame mapping nightlight tile names to their vertice coordinates. This is used to identify nightlight tiles as well as to build a spatial polygons dataframe used to plot the tiles. OLS only has one tile for the whole world and thus has a dummy entry. OLS is included to prevent code duplication by writing separate functions for OLS.

Usage

```
getNlTiles(nlType)
```

Arguments

`nlType`	the nlType of interest

Value

A data.frame of names of tiles and lon-lat coordinate of top-left corner of each

Examples

```
Rnightlights:::getNlTiles("VIIRS.M")

Rnightlights:::getNlTiles("OLS.Y")
```

getNlTileTifLclNameOLS

Constructs the filename of the decompressed OLS .tif file

Description

Constructs the filename of the decompressed OLS .tif file

Usage

```
getNlTileTifLclNameOLS(nlPeriod)
```

Arguments

nlPeriod	the nlPeriod in which the tile was created

Value

a character vector filename of the .tif OLS tile

Examples

```
#using default dirNlTiles
## Not run:
Rnightlights:::getNlTileTifLclNameOLS("2004")
 #returns "OLS_2004_00N180W.tif"

## End(Not run)
```

getNlTileTifLclNamePath

Constructs the full path used to save/access the downloaded tile .tgz file

Description

Constructs the full path used to save/access the downloaded tile .tgz file

Usage

```
getNlTileTifLclNamePath(nlType, nlPeriod, tileNum)
```

Arguments

nlType	the nlType of interest
nlPeriod	the nlPeriod in which the tile was created
tileNum	the index of the tile as given in nlTileIndex

Value

a character string filename of the compressed .tgz VIIRS tile

Examples

```
## Not run:
Rnightlights:::getNlTileZipLclNamePath("OLS.Y", "2012", 1)
 #returns "/dataPath/OLS.Y_2012_00N180W.tgz"

## End(Not run)

## Not run:
Rnightlights:::getNlTileZipLclNamePath("VIIRS.M", "201412", 1)
 #returns "/dataPath/VIIRS.M_201412_75N180W.tgz"

## End(Not run)
```

---

getNlTileTifLclNamePathOLS

Constructs the full path used to save/access the decompressed OLS .tif file

---

Description

Constructs the full path used to save/access the decompressed OLS .tif file

Usage

```
getNlTileTifLclNamePathOLS(nlPeriod, tileNum)
```

Arguments

nlPeriod	the year in which the tile was created
tileNum	ignored

Value

a character vector filename of the .tif OLS tile

Examples

```
#using default dirNlTiles
## Not run:
Rnightlights:::getNlTileTifLclNamePathOLS("2012", 1)
 #returns "/dataPath/tiles/OLS_2012.tif"

## End(Not run)
```

`getNlTileTifLclNamePathVIIRS` *Constructs the full path used to save/access the decompressed VIIRS .tif file*

Description

Constructs the full path used to save/access the decompressed VIIRS .tif file

Usage

```
getNlTileTifLclNamePathVIIRS(nlPeriod, tileNum, nlType)
```

Arguments

`nlPeriod`	the yearMonth in which the tile was created
`tileNum`	the index of the tile as given in nlTileIndex
`nlType`	The particular VIIRS type e.g. VIIRS_D for daily VIIRS

Value

a character vector filename of the .tif VIIRS tile

Examples

```
#using default dirNlTiles
## Not run:
Rnightlights:::getNlTileTifLclNamePathVIIRS("201401", 1)
 #returns "/dataPath/tiles/VIIRS_2014_01_75N180W.tif"

## End(Not run)
```

getNlTileTifLclNameVIIRS

Constructs the filename of the decompressed VIIRS .tif file

Description

Constructs the filename of the decompressed VIIRS .tif file

Usage

```
getNlTileTifLclNameVIIRS(nlPeriod, tileNum, nlType)
```

Arguments

nlPeriod	the nlPeriod in which the tile was created
tileNum	the index of the tile as given in nlTileIndex
nlType	character string the nlType

Value

a character vector filename of the .tif VIIRS tile

Examples

```
#using default dirNlTiles
## Not run:
Rnightlights:::getNlTileTifLclNameVIIRS("201401", 1, "VIIRS.M")
 #returns "VIIRS_201401_75N180W.tif"

## End(Not run)
```

getNlTileZipLclNameOLS

The name with which to save the OLS tile locally

Description

The name with which to save the OLS tile locally

Usage

```
getNlTileZipLclNameOLS(nlPeriod)
```

Arguments

nlPeriod	The year of the OLS tile

Value

character string filename

Examples

```
## Not run:
Rnightlights:::getNlTileZipLclNameOLS("2012")
#returns "OLS.Y_2012_00N180W.tar"

## End(Not run)
```

---

getNlTileZipLclNamePath

Constructs the full path used to save/access the compressed downloaded tile

---

Description

Constructs the full path used to save/access the compressed downloaded tile Calls the relevant function for the given nlType

Usage

```
getNlTileZipLclNamePath(nlType, nlPeriod, tileNum)
```

Arguments

nlType	the nlType of interest
nlPeriod	the nlPeriod in which the tile was created
tileNum	the index of the tile as given in nlTileIndex

Value

a character string filename of the compressed .tgz VIIRS tile

Examples

```
## Not run:
Rnightlights:::getNlTileZipLclNamePath("VIIRS.M", "201401", 1)
 #returns "/dataPath/VIIRS_2014_01_75N180W.tgz"

## End(Not run)

## Not run:
Rnightlights:::getNlTileZipLclNamePath("OLS.Y", "2004", 1)
 #returns "/dataPath/OLS.Y_2004_00N180W.tar"

## End(Not run)
```

`getNlTileZipLclNameVIIRS` *Constructs the filename used to save/access the downloaded VIIRS tile .tgz file*

Description

Constructs the filename used to save/access the downloaded VIIRS tile .tgz file

Usage

```
getNlTileZipLclNameVIIRS(nlPeriod, tileNum, nlType)
```

Arguments

`nlPeriod`	The nlPeriod in which the tile was created
`tileNum`	The index of the tile as given in nlTileIndex
`nlType`	character the nlType

Value

A character string filename of the compressed .tgz VIIRS tile

Examples

```
## Not run:
Rnightlights:::getNlTileZipLclNameVIIRS("201401", 1)
 #returns "./tiles/VIIRS_2014_01_75N180W.tgz"

## End(Not run)
```

getNlUrlOLS *Function to return the url of the OLS tile to download*

Description

Function to return the url of the OLS tile to download given the year

Usage

```
getNlUrlOLS(nlPeriod)
```

Arguments

nlPeriod	The nlPeriod of the tile for which to return the tile download URL

Value

character string Url of the OLS tile file

Examples

```
## Not run:
tileUrl <- Rnightlights:::getNlUrlOLS("1999")

## End(Not run)
```

getNlUrlVIIRS *Function to return the url of the VIIRS tile to download*

Description

Function to return the url of the VIIRS tile to download given the year, month, and nlTile index

Usage

```
getNlUrlVIIRS(nlPeriod, tileNum, nlType)
```

Arguments

nlPeriod	character string the nlPeriod
tileNum	The integer index of the tile to download as given by `getNlTiles`
nlType	character the nlType

Value

Character string Url of the VIIRS tile file

Examples

```
## Not run:
tileUrl <- Rnightlights:::getNlUrlVIIRS("20171231", "1", "VIIRS.D")

tileUrl <- Rnightlights:::getNlUrlVIIRS("201401", "1", "VIIRS.M")

tileUrl <- Rnightlights:::getNlUrlVIIRS("2015", "1", "VIIRS.Y")

## End(Not run)
```

`getPolyFname` *Returns the directory name of the unzipped shapefile downloaded from http://gadm.org without the path*

Description

Returns the directory name of the unzipped shapefile downloaded from http://gadm.org without the path

Usage

```
getPolyFname(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	character the ISO3 code of the country
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

character name of shapefile directory

Examples

```
Rnightlights:::getPolyFname("KEN")
 #returns "KEN_adm_shp"
```

getPolyFnamePath *Get the path of the unzipped polygon directory downloaded from GADM.ORG*

Description

Get the path of the unzipped polygon directory downloaded from GADM.ORG. Note the polygons are in ESRI Shapefile format thus when unzipped create a directory with the name <ctrycode>_adm_shp e.g. KEN_adm_shp. The directory will contain a number of files including the .shp file. `rgdal::readOGR` can read a shapefile polygon when given the directory path. It will determine which files to read.

Usage

```
getPolyFnamePath(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	character the ISO3 code of the country
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

character path to polygon shapefile directory

Examples

```
Rnightlights:::getPolyFnamePath("KEN")
 #returns "dataPath/polygons/KEN_adm_shp"

#@export only due to exploreData() shiny app
```

getPolyFnameRDS *Get the filename of the polygon zip file as downloaded from* http://GADM.ORG

Description

Get the filename of the polygon zip file as downloaded from http://GADM.ORG

Usage

```
getPolyFnameRDS(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	character the ISO3 code of the country
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character path to zip

Examples

```
Rnightlights:::getPolyFnameZip("KEN")
 #returns "path/to/"
```

---

getPolyFnameZip *Get the filename of the polygon zip file as downloaded from* http://GADM.ORG

Description

Get the filename of the polygon zip file as downloaded from http://GADM.ORG

Usage

```
getPolyFnameZip(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	character the ISO3 code of the country
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character path to zip

Examples

```
Rnightlights:::getPolyFnameZip("KEN")
 #returns "path/to/"
```

getTilesCtryIntersectVIIRS *Get a list of tiles that a country polygon intersects with*

Description

Create a dataframe mapping each country in the rworldmap to the VIIRS tiles which they intersect with and thus need to be retrieved to process their nightlight imagery. Since some functions use this dataframe for long-term processing, omitCountries can eliminate countries that should be excluded from the list hence from processing. Countries can be added in the omitCountries function. Default is "none".

Usage

```
getTilesCtryIntersectVIIRS(ctryCode)
```

Arguments

ctryCode	The country's ISO3 code

Value

None

Examples

```
Rnightlights:::getTilesCtryIntersectVIIRS("KEN")
```

insertNlDataCol *Insert an aggregate nightlight data column in a country nightlights dataframe*

Description

Insert an aggregate nightlight data column in a country nightlights dataframe. The number of elements in the vector MUST match the number of rows in the country dataframe.

Usage

```
insertNlDataCol(ctryNlDataDF, dataCol, statType, nlPeriod, nlType)
```

Arguments

ctryNlDataDF	dataframe with the country data to save
dataCol	the numeric vector to be inserted as a column
statType	the stat which produced the dataCol vector
nlPeriod	the nlPeriod that the dataCol belongs to
nlType	the type of nightlight data

Value

the updated dataframe

Examples

```
## Not run:
ctryNlDataDF <- Rnightlights:::insertNlDataCol(ctryNlDataDF,
    dataCol, "sum", "201409", "VIIRS.D")

## End(Not run)

## Not run:
ctryNlDataDF <- Rnightlights:::insertNlDataCol(ctryNlDataDF,
    dataCol, "mean", "2012", "OLS.Y")

## End(Not run)
```

---

listCtryNlData *List available data*

Description

List available data. If source is "local" it lists data cached locally. If source is remote lists available data on the remote repository.

Usage

```
listCtryNlData(ctryCodes = NULL, admLevels = NULL, nlTypes = NULL,
  nlPeriods = NULL, polySrcs = NULL, polyVers = NULL, nlStats = NULL,
  source = "local")
```

Arguments

ctryCodes	character vector of ctryCodes to filter by
admLevels	A character vector of admLevels to filter by
nlTypes	A character vector of nlTypes to filter by
nlPeriods	A character vector of nlPeriods to filter by
polySrcs	The source of polygons e.g. GADM or CUST to filter by
polyVers	The version of the polygon source to filter by
nlStats	The stats to filter by
source	Character string. Whether to check data availability "local" or "remote" Not in use.

Value

a list of countries and the periods and stats for each

Examples

```
#list all data
listCtryNlData()

#list all data available for KEN
listCtryNlData(ctryCodes = "KEN")

#list all VIIRS.* data available for ECU
listCtryNlData(ctryCodes = "ECU", nlTypes = "VIIRS")

#list available OLS.Y data for KEN and RWA in 2012 & 2013
listCtryNlData(ctryCodes = c("KEN","RWA"), nlPeriods = c("2012", "2013"), nlTypes = "OLS.Y")
```

---

listCtryNlRasters *List available cropped country rasters*

Description

List available data. If source is "local" it lists data cached locally. If source is remote lists available data on the remote repository.

Usage

```
listCtryNlRasters(ctryCodes = NULL, nlPeriods = NULL, nlTypes = NULL,
  polySrcs = NULL, polyVers = NULL, nlStats = NULL, source = "local")
```

Arguments

ctryCodes	A character vector of ctryCodes to filter by
nlPeriods	A character vector of nlPeriods to filter by
nlTypes	A character vector of nlTypes to filter by
polySrcs	The source of polygons e.g. GADM or CUST to filter by
polyVers	The version of the polygon source to filter by
nlStats	The stats to filter by
source	Character string. Whether to check data availability "local" or "remote". Default is "local".

Value

a list of existing cropped rasters

Examples

```
#list all rasters
listCtryNlRasters()

#list all rasters available for KEN
listCtryNlRasters(ctryCodes = "KEN")

#list all VIIRS rasters available for ECU
listCtryNlRasters(ctryCodes = "ECU", nlTypes = "VIIRS")

#list available OLS rasters for KEN and RWA in 2012 & 2013
listCtryNlRasters(ctryCodes = c("KEN","RWA"), nlPeriods = c("2012", "2013"), nlTypes = "OLS.Y")
```

---

listNlTiles *List locally cached tiles*

Description

List the tiles which have been downloaded previously and are currently cached in the local tiles folder

Usage

```
listNlTiles(nlTypes = NULL, nlPeriods = NULL, tileName = NULL,
  source = "local")
```

Arguments

nlTypes	A character vector of nlTypes to filter by
nlPeriods	A character vector of nlPeriods to filter by
tileName	Character vector tileNames to filter by
source	Character string. Whether to check data availability. "local" or "remote". Default is "local".

Value

a list of locally cached nlTiles or NULL

Examples

```
#list all tiles
listNlTiles()

#list all VIIRS tiles
listNlTiles(nlTypes = "VIIRS")

#list all VIIRS tiles available in the years 2014-2015. Note VIIRS data
#starts in 201401
listNlTiles(nlTypes = "VIIRS.M", nlPeriods = nlRange("201401", "201512"))

#filter data
listNlTiles(nlTypes = "OLS.Y", nlPeriods = c("2012", "2013"))
```

---

mapAllCtryPolyToTilesVIIRS

Create a mapping of all countries and the tiles they intersect

---

Description

This is simply another name for mapCtryPolyToTilesVIIRS with ctryCodes="all"

Usage

```
mapAllCtryPolyToTilesVIIRS(omitCountries = pkgOptions("omitCountries"))
```

Arguments

omitCountries	A character vector or list of countries to leave out when processing. Default is "none"

Value

None

Examples

```
#no countries omitted
## Not run:
tileMap <- Rnightlights:::mapAllCtryPolyToTilesVIIRS()

## End(Not run)

#no countries omitted
## Not run:
tileMap <- Rnightlights:::mapAllCtryPolyToTilesVIIRS(omitCountries="none")

## End(Not run)

#include countries that take long to process
## Not run:
tileMap <- Rnightlights:::mapAllCtryPolyToTilesVIIRS(omitCountries=c("error", "long"))

## End(Not run)
```

`mapCtryPolyToTilesVIIRS` *Create a mapping of all countries and the tiles they intersect*

Description

Create a dataframe mapping each country in the rworldmap to the VIIRS tiles which they intersect with and thus need to be retrieved to process their nightlight imagery. Since some functions use this dataframe for long-term processing, omitCountries can eliminate countries that should be excluded from the list hence from processing. Countries can be added in the omitCountries function. Default is "none".

Usage

```
mapCtryPolyToTilesVIIRS(ctryCodes = "all",
  omitCountries = pkgOptions("omitCountries"))
```

Arguments

`ctryCodes`	A character vector or list of countries to map. Default is `"all"`
`omitCountries`	A character vector or list of countries to leave out. Default is `"none"`

Value

ctryCodeTiles A data frame of countries and the tiles they intersect with as give by `getNlTiles`

Examples

```
#map all countries
## Not run:
tileMap <- Rnightlights:::mapCtryPolyToTilesVIIRS()

## End(Not run)

#map all countries, no countries omitted
## Not run:
tileMap <- Rnightlights:::mapCtryPolyToTilesVIIRS(ctryCodes="all", omitCountries="none")

## End(Not run)

#will not omit countries that do not have polygons on GADM
## Not run:
tileMap <- Rnightlights:::mapCtryPolyToTilesVIIRS(omitCountries=c("error", "missing"))

## End(Not run)
```

`masqOLS` *Extract raster pixel values within the boundaries of a polygon*

Description

Extract raster pixel values within the boundaries of a polygon

Usage

```
masqOLS(shp, rast, i, retVal)
```

Arguments

`shp`	the country Polygon layer as SpatialPolygon
`rast`	the clipped country raster
`i`	the index of the polygon in the country polygon layer (shp)
`retVal`	Whether to return the raster data as a vector, or data.frame with spatial context NULL returns a vector of all values, colrowval returns a data.frame with row, col and raster value while lonlatval returns a data.frame with lon,lat and val.

Value

numeric vector of radiances

Examples

```
## Not run:
ctryPoly <- rgdal::readOGR(getPolyFnamePath("KEN"), getCtryShpLyrNames("KEN", 1))
ctryRaster <- raster::raster(getCtryRasterOutputFnamePath("KEN", "OLS", "1999"))
temp <- NULL
KenAdm1Sum <- NULL
for (i in 1:length(ctryPoly@polygons))
{
 temp$name <- as.character(ctryPoly@data$NAME_1[i])
 temp$sum <- sum(masqOLS(ctryPoly, ctryRaster, i), na.rm=T)

 KenAdm1Sum <- rbind(KenAdm1Sum)
}

## End(Not run)
```

`masqVIIRS` *extract data from a raster using one polygon in a multipolygon*

Description

extract data from a raster using one polygon in a multipolygon. Modified from https://commercedataservice.github.io/tutorial_viirs_part1/

Usage

```
masqVIIRS(ctryPoly, ctryRast, idx, retVal)
```

Arguments

`ctryPoly`	the country Polygon layer as SpatialPolygon
`ctryRast`	the clipped country raster
`idx`	the index of the polygon in the country polygon layer (shp)
`retVal`	Whether to return the raster data as a vector, or data.frame with spatial context NULL returns a vector of all values, colrowval returns a data.frame with row, col and raster value while lonlatval returns a data.frame with lon,lat and val.

Value

numeric vector of radiances

Examples

```
## Not run:
ctryPoly <- rgdal::readOGR('path/to/polygon.shp')

ctryRaster <- raster::raster('path/to/raster.tif')

#get the sum of nightlight pixels in the first polygon in a multipolygon
sumPolygon1 <- sum(masqVIIRS(ctryPoly, ctryRaster, 1), na.rm=T)

## End(Not run)
```

`myZonal` *Calculate zonal statistics. Used internally*

Description

Calculate zonal statistics. Used internally by zonalpipe. Modified from http://www.guru-gis.net/efficient-zonal-statistics-using-r-and-gdal/

Usage

```
myZonal(rast, zone, nlStats, digits = 0, retVal = NULL, na.rm = TRUE, ...)
```

Arguments

`rast`	the country raster
`zone`	the zonal country polygon layer
`nlStats`	a character list of statistics to calculate
`digits`	round off to how many decimals
`retVal`	Whether to return the raster data as a vector, or data.frame with spatial context NULL returns a vector of all values, colrowval returns a data.frame with row, col and raster value while lonlatval returns a data.frame with lon,lat and val.
`na.rm`	how to handle NAs
`...`	Other params to pass to the nlStats functions e.g. na.rm

Value

numeric value result of the given nlStat function

newNlType *Convert pre-0.2.0 nlType names to their new names*

Description

Convert pre-0.2.0 nlType names to their new names. Pre 0.2.0 has only 2 nlTypes i.e. OLS and VIIRS. They are renamed as follows:

- OLS => "OLS.Y"
- (VIIRS) => "VIIRS.M"

Usage

```
newNlType(oldNlType)
```

Arguments

oldNlType	character The old nlType i.e. "OLS" or "VIIRS"

Value

character The new nlType i.e. "OLS.Y" or "VIIRS.M"

Examples

```
Rnightlights:::newNlType("VIIRS")
 #returns "VIIRS.M"
```

nlCleanup *Clean up the environment after processing (Not yet implemented)*

Description

Clean up the environment after processing (Not yet implemented)

Usage

```
nlCleanup()
```

Examples

```
 ## Not run:
 Rnightlights:::nlCleanup()

## End(Not run)
```

nlInit *Initialize some important variables and create directory structure*

Description

Initialize some important variables and create directory structure

Usage

```
nlInit(omitCountries = "none")
```

Arguments

omitCountries	character string/vector CtryCodes to exclude from processing

Examples

```
 ## Not run:
 Rnightlights:::nlInit()

## End(Not run)
```

nlRange *Create a range of nlPeriods*

Description

Create a range of nlPeriods. Returns a list of character vectors of nlPeriods filling in the intermediate nlPeriods. NOTE: Both start and end range must be valid and of the same type.

Usage

```
nlRange(startNlPeriod, endNlPeriod, nlType)
```

Arguments

startNlPeriod	the nlPeriod start
endNlPeriod	the nlPeriod end
nlType	the nlType

Value

character vector of nlPeriods

Examples

```
#get OLS years between 2004 and 2010
nlRange("2004", "2010", "OLS.Y")

#get VIIRS yearMonths between Jan 2014 and Dec 2014
nlRange("201401", "201412", "VIIRS.M")
```

`orderCustPolyLayers` *Order polygon shapefile layers in custom polygons*

Description

Order polygon shapefile layers in custom polygons

Usage

```
orderCustPolyLayers(ctryCode, custPolyPath = NULL)
```

Arguments

`ctryCode`	The ISO3 ctryCode of the country polygon to download
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Details

Add an index column to all layers of a polygon

Value

None

Examples

```
## Not run:
Rnightlights:::addCtryPolyIdx(ctryCode="KEN")

## End(Not run)
```

pkgOptions *Set or get options for the Rnightlights package*

Description

Set or get options for the Rnightlights package

Usage

```
pkgOptions(...)
```

Arguments

...	Option names to retrieve option values or [key]=[value] pairs to set options.

Value

if an option name is supplied as a parameter this returns the value, else a list of all options is returned.

Supported options

The following options are supported

configName.VIIRS.D (character) The regex to uniquely identify the tile file to use out of the downloaded tile .tgz. The version 1 monthly series is run globally using two different configurations.

The first excludes any data impacted by stray light. The second includes these data if the radiance vales have undergone the stray- light correction procedure (Reference). These two configurations are denoted in the filenames as "vcm" and "vcmsl" respectively. The "vcmsl" version, that includes the stray-light corrected data, will have more data coverage toward the poles, but will be of reduced quality.

It is up to the users to determine which set is best for their applications. The annual versions are only made with the "vcm" version, excluding any data impacted by stray light.

configName.VIIRS.M (character) The regex to uniquely identify the tile file to use out of the downloaded monthly .tgz tile. Has the same options as configName.VIIRS.D

configName.VIIRS.Y (character) The regex to uniquely identify the tile file to use out of the downloaded tile .tgz. The annual products can have other values for the config shortname (Field 5). They are:

- vcm-orm *(VIIRS Cloud Mask - Outlier Removed)*: This product contains cloud-free average radiance values that have undergone an outlier removal process to filter out fires and other ephemeral lights.
- vcm-orm-ntl *(VIIRS Cloud Mask - Outlier Removed - Nighttime Lights)*: This product contains the "vcm-orm" average, with background (non-lights) set to zero.
- vcm-ntl *(VIIRS Cloud Mask - Nighttime Lights)*: This product contains the "vcm" average, with background (non-lights) set to zero.

`cropMaskMethod` (`character`) The method to use to clip the nightlight raster tiles to the country boundaries

`deleteTiles` (`character`) whether to delete tiles after processing may be useful where diskspace is a concern

`dirNlData` (`character`) The directory to store the extracted data files in

`dirNlRoot` `character`) The root directory storing the package data

`dirNlTiles` (`character`) The directory in which to store the downloaded VIIRS raster tiles

`dirPolygon` (`character`) The directory to store the downloaded country administration level polygons

`dirRasterOutput` (`character`) The directory in which to store the clipped country rasters

`dirRasterWeb` (`character`) The directory in which to store the rasters resampled for web display

`dirZonals` (`character`) The directory in which to store the zonal statistics country polygon

`downloadMethod` (`character`) The download method to use

`extractMethod` (`character`) The method to use to extract data from the rasters

`gdalCacheMax` (`character`) The maximum memory gdal should use in gdal_rasterize

`ntLtsIndexUrlOLS` (`character`) The url with the OLS tile index

`ntLtsIndexUrlVIIRS` (`character`) The url with the VIIRS tile index

`numCores` (`character`) The number of processor cores to use when extractMethod = "raster"

`omitCountries` (`character`) The countries to exclude in processing

`stats` (`character`) The statistics to calculate for country regions. The default are sum and mean. Any other aggregate statistics can be included. Also any aggregate function accessible in the current environment can be added.

`tmpDir` (`character`) Change the temporary directory for processing rasters. Not in use

Examples

```
#retrieve the current cropMaskMethod
pkgOptions("cropMaskMethod")

#set the cropMaskMethod
pkgOptions(cropMaskMethod="gdal")

#retrieve all options
pkgOptions()
```

pkgReset *Reset global options for the Rnightlights package*

Description

Reset global options for the Rnightlights package

Usage

```
pkgReset()
```

Examples

```
#get cropMaskMethod
pkgOptions("cropMaskMethod") #returns default "rast"

#set cropMaskMethod to "gdal"
pkgOptions(cropMaskMethod="gdal") #sets to "gdal"

#check cropMaskMethod has changed
pkgOptions("cropMaskMethod") #returns "gdal"

#reset pkgOptions
pkgReset()

#check cropMaskMethod has been reset
pkgOptions("cropMaskMethod") #returns default "rast"
```

plotCtryWithTilesVIIRS *Plot a country boundary with the VIIRS tiles and world map*

Description

Plot a country boundary as defined in the **rworldmap** package along with the VIIRS nightlight tiles for a visual inspection of the tiles required for download in order to process a country's nightlight data. Output corresponds to that of `getCtryNlTiles()`

It utilizes `rworldmap::rwmgetISO3()` to resolve country codes as well as names.

Usage

```
plotCtryWithTilesVIIRS(ctry)
```

Arguments

ctry	character the 3-letter ISO3 country code e.g. "KEN" or a common name of the country e.g. "Kenya" as found valid by rworldmap::rwmgetISO3()

Value

None

Examples

```
#by ctryCode
## Not run: plotCtryWithTilesVIIRS("KEN")
```

---

processNLCountry *Processes nightlights for an individual country in a particular nlPeriod*

Description

Given a countryCode, yearMonth and preferred processing methods cropMaskMethod and extractMethod, this function will first check if the data already exists in the cache. First it will check if the data file exists and if it does not it will create a dataframe of the country data containing only the administrative properties and move to processing. If the data file exists it will check to see if the particular year month already exists. If it exists, it will exit with a message. If it does not exist, it will load the country data file and move on to processing.

Usage

```
processNLCountry(ctryCode, admLevel, nlType, nlPeriod,
  nlStats = pkgOptions("nlStats"),
  downloadMethod = pkgOptions("downloadMethod"),
  cropMaskMethod = pkgOptions("cropMaskMethod"),
  extractMethod = pkgOptions("extractMethod"),
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of interest
admLevel	character The country admin level in the given ctryCode at which to calculate stats
nlType	character The nlType of interest
nlPeriod	character The nlPeriod of interest
nlStats	the statistics to calculate. If not provided will calculate the stats specified in pkgOptions("nlStats")
downloadMethod	The method used to download polygons and rasters

cropMaskMethod	character Whether to use rasterize or gdal-based functions to crop and mask the country rasters
extractMethod	("rast" or "gdal") Whether to use rasterize or gdal-based functions to crop and mask the country rasters
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Details

Processing consists of:

- Reading in the country polygon in ESRI Shapefile format
- Reading in the tiles that the particular country intersects with and then clipping the tile(s) to the country boundary
- Extract the data from the clipped raster and compute various statistics at the lowest admin level in the country.
- Finally, these stats are appended to the data frame and written to the data file.

NOTE: processNLCountry() assumes that all inputs are available and will not attempt to download them. It should ideally be called from the function processNlData() which does all the preparation for processing. processNlData() which can process multiple countries and time periods will download all the required tiles and polygons prior to calling processnlcountry. getCtryNlData can also be used with the option ignoreMissing=FALSE which will call processNlData in the background.

Value

None

Examples

```
#calculate only the sum of monthly VIIRS radiances for Dec 2014 using gdal
#for both cropMask and extraction for KEN
## Not run:
Rnightlights:::processNLCountry("KEN", "KEN_adm2", "VIIRS.M", "201412", "gdal", "gdal", "sum")

## End(Not run)
```

`processNlData` *Downloads nightlight tiles and country polygons and calls the function to process them*

Description

Downloads nightlight tiles and country polygons in preparation for the appropriate functions to process them. Given a list of countries and nlPeriods and an nlType, processNlData will first determine which countries and periods do not actually have data i.e. have not been previously processed. From the list of countries and prediods that need processing, it determines which nightlight tiles require to be downloaded. At the same time, it downloads any country polygons which have not already been downloaded.

Usage

```
processNlData(ctryCodes, admLevels, nlTypes, nlPeriods,
  nlStats = pkgOptions("nlStats"), custPolyPath = NULL,
  gadmVersion = pkgOptions("gadmVersion"),
  downloadMethod = pkgOptions("downloadMethod"),
  cropMaskMethod = pkgOptions("cropMaskMethod"),
  extractMethod = pkgOptions("extractMethod"))
```

Arguments

`ctryCodes`	the list of countries to be processed
`admLevels`	the list of admin levels in the given countries at which to calculate stats
`nlTypes`	the types of nightlights to process
`nlPeriods`	the nlPeriods of interest
`nlStats`	the statistics to calculate. If not provided will calculate the stats specified in `pkgOptions("nlStats")`
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip
`gadmVersion`	The GADM version to use
`downloadMethod`	The method used to download rasters and polygons
`cropMaskMethod`	The method used to crop and mask satellite rasters
`extractMethod`	The method used to extract and perform functions on raster data

Details

processNlData then calls processNlCountry with the nlType supplied as a parameter. The processing is essentially the same for all nlTypes.

This is the main entry-point to the package and the main user-facing function. However, it works in batch mode and caches all data without returning any data to the user. However, it will return TRUE/FALSE depending on whether it completed successfully. Since it saves state regularly it can be run multiply in case of failure until it finally returns TRUE. This is where the only constraints

are downloading and processing errors due to bandwidth or other resource constraints. A good example is running a long-running batch on an AWS EC2 spot-priced machine where the server may be decommissioned at any time. See more in the examples.

Value

None

Examples

```
#long running examples which may require large downloads
## Not run:
#Example 1: process monthly VIIRS nightlights for all countries at the
    lowest admin level and for all nlPeriods available e.g. to create a
    local cache or repo

    processNlData() #process VIIRS nightlights for all countries and all periods

#Example 2: process monthly VIIRS nightlights for all countries in 2014 only

   nlPeriods <- getAllNlPeriods("VIIRS.M") #get a list of all nightlight periods to present-day

    nlPeriods <- nlPeriods[grep("^2014", nlPeriods)] #filter only periods in 2014

    processNlData(nlTypes="VIIRS.M", nlPeriods=nlPeriods)

#Example 3: process OLS nightlights for countries KEN & RWA from 1992
#     to 2000

    cCodes <- c("KEN", "RWA")

    nlPeriods <- getAllNlPeriods("VIIRS.M")

    nlPeriods <- nlRange("1992", "2000", "OLS.Y")

    processNlData(ctryCodes=cCodes, nlPeriods=nlPeriods)

#Example 4: process VIIRS nightlights for countries KEN & RWA in 2014 Oct to 2014 Dec only

    processNlData(ctryCodes=c("KEN", "RWA"), nlTypes="VIIRS.M",
        nlPeriods=c("201410", "201411", "201412"))

#Example 5: process all nightlights, all countries, all stats in one thread

   processNlData()

#Example 6: process all VIIRS monthly nightlights, all countries, all stats with each
#   year in a separate thread. Create a separate R script for each year as follows:

    library(Rnightlights)
```

```
    nlPeriods <- getAllNlPeriods("VIIRS.M")

    nlPeriods_2014 <- nlPeriods[grep("^2014", nlPeriods)]

    processNlData(nlPeriods=nlPeriods_2014)

    #Run the script from the command line as:

    #R CMD BATCH script_name_2014.R

## End(Not run)
```

`readCtryPolyAdmLayer` *Read a country admLevel polygon*

Description

Read a country admLevel polygon. Reads the saved RDS format of the shapefile which is saved as part of `dnldCtryPoly` by default. Otherwise, it will try to read the shapefile. If it fails it returns null.

Usage

```
readCtryPolyAdmLayer(ctryCode = NULL, admLevel, polyType = "rds",
  dnldPoly = TRUE, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

`ctryCode`	`character` The ctryCode of the country of interest
`admLevel`	`character` The name to search for
`polyType`	`character` Whether to read the shapefile or the RDS format.
`dnldPoly`	`logical` If the country polygon doesn't exist should we download it?
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

`SpatialPolygonsDataFrame` The admLevel polygon layer or NULL if not found

Examples

```
## Not run:
readCtryPolyAdmLayer("KEN", "KEN_adm1")
#returns "KEN_adm1"

## End(Not run)
```

readCtryStruct *Reads the ctry admin structure from struct text file*

Description

Reads the ctry admin structure from struct text file

Usage

```
readCtryStruct(ctryCode = NULL, gadmVersion = pkgOptions("gadmVersion"),
  custPolyPath = NULL)
```

Arguments

ctryCode	The ISO3 ctryCode of the country structure to read
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

None

Examples

```
## Not run:
Rnightlights:::createCtryStruct("KEN")

## End(Not run)
```

removeDataPath *Deletes a root data path all sub-directories*

Description

Deletes a root data path and all sub-directories. It can be the current directory or a previously used data path. It will only delete if it has the default folder structure of a root data path.

Usage

```
removeDataPath(dataPath = file.path(getNlDataPath(), ".Rnightlights"))
```

Arguments

dataPath	The path to the root folder to be deleted

Value

None

Examples

```
  ## Not run:
  Rnightlights:::removeDataPath(getNlDataPath())

## End(Not run)
```

`saveCtryNlData` *Save a data frame of a country's data to the appropriate location*

Description

Saves the data frame created from processNlCountry* to the appropriate location. Note: This function does not perform any validation error checking and will overwrite existing data. Use with caution.

Usage

```
saveCtryNlData(ctryNlDataDF, ctryCode = NULL, admLevel,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

`ctryNlDataDF`	dataframe with the country data to save
`ctryCode`	the ctryCode to which the data belongs
`admLevel`	the country admin level to process
`gadmVersion`	The GADM version to use
`custPolyPath`	Alternative to GADM. A path to a custom shapefile zip

Value

None

Examples

```
## Not run:
Rnightlights:::saveCtryNlData(ctryNlDataDF, ctryCode)

## End(Not run)
```

searchAdmLevel *Search for the admLevel by official name*

Description

Search for the admLevel by official name. Expects the shapefile to already exist.

Usage

```
searchAdmLevel(ctryCodes = NULL, admLevelNames, dnldPoly = TRUE,
  downloadMethod = pkgOptions("downloadMethod"),
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCodes	character The ctryCodes of the country of interest
admLevelNames	character The names to search for
dnldPoly	logical If the country polygon doesn't exist should we download it?
downloadMethod	The method used to download polygons
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

character vector of admin level names

Examples

```
## Not run:
searchAdmLevel("KEN", "county")
#returns "KEN_adm1"

## End(Not run)
```

searchCountry *Search for a country by name or code*

Description

Serves as a frontend to the ctryNameToCode and ctryCodeToName functions in one.

Usage

```
searchCountry(searchTerms, extended = FALSE)
```

Arguments

searchTerms	The country code/name to search for
extended	Whether to do partial searches

Value

data.frame A mapping of the ctryCode to ctryName if the supplied searchTerm is a ctryCode or vice-versa if the searchTerm was a ctryName

Examples

```
searchCountry("KEN") #returns Kenya

searchCountry("Tanzania") #returns United Republic of Tanzania

searchCountry("uk", TRUE) #returns United Kingdom and Ukraine

searchCountry("rwa", TRUE) #returns Rwanda and Norway
```

---

setNlDataPath *Sets the root path to the package data directory*

Description

By default, this function will set the root path to ~/.Rnightlights/.

Usage

```
setNlDataPath(dataPath)
```

Arguments

dataPath	The path

Value

Returns (invisibly) the old root path.

Examples

```
## Not run:
Rnightlights:::setNlDataPath("/new/path")

## End(Not run)
```

setupDataPath *Interactively allows the user to set up the default root path*

Description

Interactively allows the user to set up the default root path where data is cached

Usage

```
setupDataPath(newDataPath = tempdir(), ...)
```

Arguments

newDataPath	Default root path to set
...	Not used.

Value

character string Returns (invisibly) the root path, or @NULL if running a non-interactive session.

See Also

Internally, @see "setNlDataPath" is used to set the root path. The "base::interactive" function is used to test whether R is running interactively or not.

tileIdx2Name *Get the name of a tile given its index*

Description

Get the name of a VIIRS tile as given by getNlTiles() given its index

Usage

```
tileIdx2Name(tileNum, nlType)
```

Arguments

tileNum	index as given by getNlTiles()
nlType	the nlType of interest

Value

Character name of the tile

Examples

```
Rnightlights:::tileName2Idx("00N060W", "VIIRS.M") #returns 6
```

`tileName2Idx` *Get the index of a tile given its name*

Description

Get the index of a VIIRS tile as given by getNlTiles() given its name

Usage

```
tileName2Idx(tileName, nlType)
```

Arguments

`tileName`	name as given by getNlTiles()
`nlType`	the nlType of interest

Value

Integer index of the tile

Examples

```
Rnightlights:::tileName2Idx("00N060W", "VIIRS.M")
```

`tilesPolygonIntersectVIIRS`
Get the list of VIIRS tiles that a polygon intersects with

Description

Get the list a VIIRS tiles that a polygon intersects with

Usage

```
tilesPolygonIntersectVIIRS(shpPolygon)
```

Arguments

`shpPolygon`	a SpatialPolygon or SpatialPolygons

Value

Character vector of the intersecting tiles as given by `getNlTiles`

Examples

```
## Not run:
#download shapefile if it doesn't exist
ctryShapefile <- Rnightlights:::dnldCtryPoly("KEN")

#read in shapefile top layer
ctryPoly <- readCtryPolyAdmLayer("KEN",
    Rnightlights:::getCtryShpLyrNames("KEN",0))

#get list of intersecting tiles
tileList <- Rnightlights:::tilesPolygonIntersectVIIRS(ctryPoly)

## End(Not run)
```

---

upgradeRnightlights *Perform upgrade functions to new package versions as required*

Description

Perform upgrade functions to new package versions as required General Upgrade functions pre-0.2.0 to 0.2.0:

- Rename tiles
- Rename output rasters
- Rename data files
- Rename data column names
- Remove zonal rasters which will be regenerated when required

Usage

```
upgradeRnightlights()
```

Value

TRUE/FALSE

Examples

```
## Not run:
  Rnightlights:::upgradeRnightlights()
  #returns TRUE/FALSE

## End(Not run)
```

validCtryAdmLvls *Checks if ctry admLevels are valid*

Description

Checks if ctry admLevels are valid

Usage

```
validCtryAdmLvls(ctryCode = NULL, admLevels,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	character The ctryCode of the country of interest
admLevels	character The admLevel(s) to search for
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

logical whether inputted admLevels are valid

Examples

```
## Not run:
validCtryAdmLvls("KEN", "adm0")
#returns "KEN_adm1"

## End(Not run)
```

validCtryCodes *Check if country codes are valid*

Description

Check if country codes are valid

Usage

```
validCtryCodes(ctryCodes)
```

Arguments

ctryCodes	the ISO3 country codes to validate

Value

named logical vector TRUE/FALSE

Examples

```
validCtryCodes(c("KEN", "UGA")) #returns TRUE TRUE

validCtryCodes("UAE") #returns FALSE. "United Arab Emirates" ISO3 code = "ARE"
```

`validCtryNlDataDF` *Check if a country dataframe is valid*

Description

Check if a country dataframe is valid

Usage

```
validCtryNlDataDF(ctryNlDataDF)
```

Arguments

`ctryNlDataDF`	the country dataframe

Value

TRUE/FALSE

Examples

```
## Not run:
  Rnightlights:::validCtryNlDataDF(nlCtryDataDF) #returns TRUE

## End(Not run)
```

validGadmVersions *Check whether GADM versions are valid*

Description

Check whether GADM versions are valid

Usage

```
validGadmVersions(gadmVersions)
```

Arguments

gadmVersions	The GADM versions to validate

Value

logical vector

Examples

```
## Not run:
Rnightlights:::validGadmVersions("2.8")

## End(Not run)
```

validNlPeriods *Check if an nlPeriod is valid for a given nightlight type*

Description

Check if an nlPeriod is valid for a given nightlight type. Vectorized to allow checking multiple nlPeriods of corresponding nlTypes. If a single nlType is given all nlPeriods are checked in that nlType. If multiple nlTypes are given, then a corresponding number of nlPeriods is expected e.g. if nlPeriods is a vector each entry must correspond to the nlType. If multiple nlPeriods are to be tested per nlType then a list of vectors is expected, one for each nlType.

Usage

```
validNlPeriods(nlPeriods, nlTypes)
```

Arguments

nlPeriods	the nlPeriods of interest
nlTypes	type of nightlight

Value

named logical list of TRUE/FALSE

Examples

```
validNlPeriods(c("201401", "201402"),"VIIRS.M")
 #returns
 #$VIIRS.M
 #201401 201402
 #  TRUE   TRUE

validNlPeriods("201203","VIIRS.M")
 #returns FALSE

validNlPeriods("2012","OLS.Y")
 #returns TRUE
```

`validNlStats` *Check if given statistics are valid*

Description

Check if given statistics are valid. A valid statistic is one which is a function available in the current environment and returns a valid value to match.fun

Usage

```
validNlStats(nlStats)
```

Arguments

`nlStats`	the statistics to check

Value

named logical TRUE/FALSE

Examples

```
Rnightlights:::validNlStats(c("sum", "mean"))
 #returns TRUE TRUE

Rnightlights:::validNlStats("unknownFunction")
 #returns FALSE
```

validNlTileNameVIIRS *Check valid VIIRS nightlight tile name*

Description

Check if a tile name is valid for a given VIIRS nightlight type.

Usage

```
validNlTileNameVIIRS(tileName, nlType)
```

Arguments

tileName	the name of the tile
nlType	character the nlType

Value

TRUE/FALSE

Examples

```
Rnightlights:::validNlTileNameVIIRS("00N060W", "VIIRS.M")
 #returns TRUE
```

validNlTileNumVIIRS *Check valid tile number for a given VIIRS nightlight type*

Description

Check if a tile number is valid for a given VIIRS nightlight type.

Usage

```
validNlTileNumVIIRS(nlTileNum, nlType)
```

Arguments

nlTileNum	the index of the tile
nlType	A character string of nlType

Value

TRUE/FALSE

Examples

```
Rnightlights:::validNlTileNumVIIRS("1", "VIIRS.M")
 #returns TRUE

Rnightlights:::validNlTileNumVIIRS("9", "VIIRS.D")
 #returns FALSE
```

`validNlTypes` *Checks if a given character string is a valid nlType*

Description

Checks if a given character string is a valid nlType

Usage

```
validNlTypes(nlTypes)
```

Arguments

`nlTypes`	`character vector` The nlTypes

Value

`logical` whether the strings are valid nlTypes

Examples

```
validNlTypes("VIIRS.D") #returns TRUE

validNlTypes("VIIRS") #returns FALSE
```

`ZonalPipe` *Create a zonal file if it does not exist and calculate the zonal stats*

Description

Create a zonal file if it does not exist and calculate the zonal stats by calling the myZonal function. Modified from http://www.guru-gis.net/efficient-zonal-statistics-using-r-and-gdal/

Usage

```
ZonalPipe(ctryCode, admLevel, ctryPoly, path.in.shp, path.in.r, path.out.r,
  path.out.shp, zone.attribute, nlStats,
  gadmVersion = pkgOptions("gadmVersion"), custPolyPath = NULL)
```

Arguments

ctryCode	the ctryCode of interest
admLevel	The country admin level of interest
ctryPoly	the SpatialPolygonsDataFrame country polygon to process
path.in.shp	The path to the country shapefile
path.in.r	The path to the raster tile
path.out.r	The path where to save the output zonal raster
path.out.shp	The path to save the output zonal shapefile (Ignored)
zone.attribute	The zonal attribute to calculate
nlStats	The stats to calculate
gadmVersion	The GADM version to use
custPolyPath	Alternative to GADM. A path to a custom shapefile zip

Value

TRUE/FALSE

Index