Package ‘geotopbricks’

March 4, 2024

Maintainer  Emanuele Cordano <emanuele.cordano@gmail.com>
License       GPL (>= 3)
Title         An R Plug-in for the Distributed Hydrological Model GEOtop
Type          Package
Author        Emanuele Cordano
Description    It analyzes raster maps and other information as input/output
               files from the Hydrological Distributed Model GEOtop. It contains functions
               and methods to import maps and other keywords from geotop.inpts file. Some
               examples with simulation cases of GEOtop 2.x/3.x are presented in the package.
               Any information about the GEOtop Distributed Hydrological Model source code
               is available on www.geotop.org. Technical details about the model are

Version 1.5.8.2
Date  2024-03-04
Repository CRAN
Depends  R (>= 2.10), methods, raster, stringr, zoo, sf
Imports   terra
Suggests soilwater
URL        https://zenodo.org/record/8228413, www.geotop.org,
           https://www.rendena100.eu/en/geotop-hydrological-model.html,
           https://github.com/ecor/geotopbricks
RoxygenNote 7.3.1
NeedsCompilation no
Date/Publication  2024-03-04 20:40:03 UTC

R topics documented:

  argsParser ................................................................. 2
  bondone ................................................................. 3
argsParser

Parser of an argument string

Description

This command parses ...

Usage

argsParser(option, args, sep = " ", novalue_response = NULL)
Arguments

- `option`: character strings containing options (or flag) whose values
- `args`: String containing all the arguments of an R script
- `sep`: separator character. Default is " ". If it is of length 2, the first is separator among different options, the second is between option name and its value.
- `novalue_response`: value used in case the option is missing. Default is `NULL`.

Examples

```r
args <- "--value 6 --fruit apple"
option <- "--fruit"
value <- argsParser(option=option, args=args)

option2 <- "--jobs"
value2 <- argsParser(option=option2, args=args)
value22 <- argsParser(option=option2, args=args, novalue_response="./")
args_b <- "value=6 , fruit=apple"
option3 <- "value"
value <- argsParser(option=option3, args=args_b, sep=c("","",""))
```

Description

It contains hourly meteorological data observed at MeteoTrentino T0327 station located at Monte Bondone-Viotte (Trentino, Easter Alps, Italy) from August 2004 to December 2012.

The `zoo` object `meteo` contains:

- `Iprec`: Hourly Precipitation Depth expressed in millimeters
- `AirT`: Air Temperature expressed in Celsius Degree
- `RH`: Relative Humidity in Percent
- `WinDir`: Wind Direction expressed in Degrees North Clockwise
- `WinSp`: Wind Direction expressed in meters per second
- `Swglob`: Short-Wave Radiation expressed in Watts per square meters

The corresponding time axis vector for each observation can be printed by typing `index(meteo)`.
Usage

data(bondone)

Format

Data frame, ‘zoo’ object

Details

This data set stores all meteorological information useful for a GEOtop simulation. The user can easily use the package with his/her own data after replacing the values of such variables.

Source

Original data are provided by Provincia Autonoma di Trento (https://www.meteotrentino.it/).
This dataset is intended for research purposes only, being distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY.

Description

Added implementation for 'brick' S4 method

@title brick

Usage

## S4 method for signature 'zoo'
brick(
  x,
  layer = 1,
  timerange = NULL,
  time = NULL,
  rows = 1:nrow(x),
  crs = NULL,
  use.read.raster.from.url = TRUE
)

## S4 method for signature 'GeotopRasterBrick'
brick(x)
Arguments

x  
a 'zoo' object returned by function pointer.to.maps.xyz.time or pointer.to.maps.xy.time or a GeotopRasterBrick-class object

layer  
layer at which raster maps are imported. If is NULL, maps ara no-zlayer distributed and zoo must be returend by pointer.to.maps.xy.time

timerange  
two-elements vector containing the time range at which geotop maps are imported

time  
vector of time instants at which geotop maps are imported

rows  
rows of zoo correspondig to the geotop maps that are imported. By default all rows of zoo are considered. It is calculated by time or timerange if they are not set as NULL.

crs  
coordinate system see RasterBrick-class

use.read.raster.from.url  
logical value. Default is TRUE. If TRUE the RasterLayer are read with read.raster.from.url, instead of raster (otherwise). It is recomended in case the files whose paths are contained in x are remote and are 'http' addresses. In this cases the stand-alone method raster(x) does not always work and use.read.raster.from.url is necessary.

Value

a RasterBrick-class containing the geopop maps indicated by x, which is already in a GeotopRasterBrick-class object or a 'zoo' object returned by function pointer.to.maps.xyz.time or pointer.to.maps.xy.time.

See Also

getvalues.brick.at.depth, vertical.aggregate.brick.within.depth

Examples

# TON TOSS
# See the examples in the functions listed in the 'SeeAlso' section
Usage

brick.decimal.formatter(
    file = NULL,
    file_prefix,
    formatter = "%04d",
    file_extension = ".asc",
    nlayers = 10,
    use.read.raster.from.url = FALSE,
    crs = NULL,
    start.from.zero = FALSE
)

Arguments

file filename of the 'brick' files containing the decimal formatter. It is NULL by
default, otherwise it replaces file.suffix, formatter and file_extension.

file_prefix character string suffix name of the 'brick' files.

formatter string value. Default is "%04d".

file_extension string value. Default is ".asc"

nlayers number of layers

use.read.raster.from.url logical value. Default is FALSE. (this is recommended in this function). If TRUE
the RasterLayer are read with read.raster.from.url, instead of raster (otherwise). It is recomended in case the files whose paths are contained in x are re-

remote and are 'http' addresses. In this cases the stand-alone method raster(x)
does not always work and use.read.raster.from.url is necessary.

crs coordinate system see RasterBrick-class, Default is NULL.

start.from.zero logical value. Default is FALSE. If TRUE the formatter starts from 0000, otherwise
it starts from 0001.

Value

the output is returned as a RasterBrick-class object

Examples

library(geotopbricks)
library(raster)
file <- system.file("example_files/examples/snowthickness",package="geotopbricks")
file <- paste(file,"SnowThickness0000L%04d.asc",sep="/")
# nlayers=15
nlayers <- 6 ## Only 6 layers are read to minimize the elapsed time of the example!!
b <- brick.decimal.formatter(file=file, nlayers=nlayers)
nlayers(b)
names(b)
Description

Extracts a brick or a raster layer from an output 3D Tensor or 2D map respectively.

Usage

```r
brickFromOutputSoil3DTensor(x, when, layers = "SoilLayerThicknesses", one.layer = FALSE, suffix = "L%04dN%04d.asc", time_formatter = "N%04d", suffix_one.layer = "N%04d.asc", wpath = NULL, tz = "A", start_date_key = "InitDateDDMMYYYYhhmm", end_date_key = "EndDateDDMMYYYYhhmm", timestep = "OutputSoilMaps", use.read.raster.from.url = FALSE, crs = NULL, projfile = "geotop.proj", start.from.zero = FALSE, secondary.suffix = NULL, only.map.filename = FALSE, add_suffix_dir = NULL, ...)
```

```r
rasterFromOutput2DMap(x, when, ...)
```

Arguments

- **x**: string. GEOtop keyword related to the 3D or 2D variable to be imported in R.
- **when**: `POSIXct-class` for date and time on which the variable `x` is requested.
- **layers**: number of soil layer or geotop keyword for soil layer (e.g., `SoilLayerThicknesses` or `SoilFile`). Default is `SoilLayerThicknesses`.
- **one.layer**: logical value. If `TRUE` a `RasterLayer-class` object is imported, otherwise a `RasterBrick-class` object is returned. Default for `brickFromOutputSoil3DTensor` is `FALSE`. 
suffix character string containing the decimal formatter used by GEOtop in the output file names. Default is "L%04dN%04.asc". A simple user is recommended not to modify the value of this argument and use the default value.

time_formatter, suffix_one.layer character string (suffix_one.layer is used for 2Dxy map) containing the decimal formatter used by GEOtop in the output file names to indicate time instant. Default is "N%04.asc". A simple user is recommended not to modify the value of this argument and use the default value.

wpath, tz, use.read.raster.from.url see get.geotop.inpts.keyword.value

start_date_key, end_date_key initial and final dates and times of the GEOtop simulation or alternatively the respective keywords of *.inpts file (Default)

timestep time step expressed in seconds every which the raster file has been created. It can be a string corresponding to the geotop keyword in the inpts file. Default value is "OutputSoilMaps".

crs, start.from.zero see brick.decimal.formatter. If crs is not NULL (Default), projfile is ignored.

projfile name of the *.proj file containing CRS information. See get.geotop.inpts.keyword.value. Default is "geotop.proj". If is NULL or NA or this file does not exist, it is not searched and read.. In case use.read.raster.from.url is TRUE and no NULL or NA values are assinged, the *.proj file is searched.

secondary.suffix String secondary suffix which can be added at the end of the Map file name (optional). Default is NULL and no secondary suffix is added.

only.map.filename logical value. If it is TRUE, only map file names are returned and maps are not imported. Default is FALSE.

add_suffix_dir, ... additional arguments for get.geotop.inpts.keyword.value or brickFromOutputSoil3DTensor

Details

These functions brickFromOutputSoil3DTensor and rasterFromOutput2DMap return 3D or 2D Raster-class objects respectively. rasterFromOutput2DMap is a wrapper function of brickFromOutputSoil3DTensor with the option one.layer=TRUE. The functions work with the following output keywords:

"SoilTempTensorFile",
"SoilAveragedTempTensorFile",
"SoilLiqContentTensorFile",
"SoilAveragedLiqContentTensorFile",
"SoilIceContentTensorFile",
"SoilAveragedIceContentTensorFile",
"SoilLiqWaterPressTensorFile",
"SoilTotWaterPressTensorFile" for \texttt{brickFromOutputSoil3DTensor};
"FirstSoilLayerTempMapFile",
"FirstSoilLayerAveragedTempMapFile",
"FirstSoilLayerLiqContentMapFile",
"FirstSoilLayerIceContentMapFile",
"LandSurfaceWaterDepthMapFile",
"ChannelSurfaceWaterDepthMapFile",
"NetRadiationMapFile",
"InLongwaveRadiationMapFile",
"NetLongwaveRadiationMapFile",
"NetShortwaveRadiationMapFile",
"InShortwaveRadiationMapFile",
"DirectInShortwaveRadiationMapFile",
"ShadowFractionTimeMapFile",
"SurfaceHeat FluxMapFile",
"SurfaceSensibleHeat FluxMapFile",
"SurfaceLatentHeat FluxMapFile",
"SurfaceTempMapFile",
"PrecipitationMapFile",
"CanopyInterceptedWaterMapFile",
"SnowDepthMapFile",
"GlacierDepthMapFile",
"SnowMeltedMapFile",
"SnowSublMapFile",
"GlacierMeltedMapFile",
"GlacierSublimatedMapFile",
"AirTempMapFile",
"WindSpeedMapFile",
"WindDirMapFile",
"RelHumMapFileadd_suffix_dir=NULLLe",
"SWEMapFile",
"GlacierWaterEqMapFile"
"SnowDurationMapFile",
"ThawedSoilDepthMapFile",
"ThawedSoilDepthFromAboveMapFile",
"WaterTableDepthMapFile",
"WaterTableDepthFromAboveMapFile",
"NetPrecipitationMapFile",
"EvapotranspirationFromSoilMapFile" for \texttt{rasterFromOutput2DMap}. 
create.geotop.inpts.keyword

Author(s)

Emanuele Cordano

See Also

get.geotop.inpts.keyword.value

Examples

library(geotopbricks)
## Not run:
# The data containing in the link are only for educational use
wpath <- 'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/idroclim_test1'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/idroclim_test1
x <- "SoilliqContentTensorFile"
tz <- "Etc/GMT-1"
when <- as.POSIXct("2002-03-22",tz=tz)

# Not Run because it elapses too long time!!!
# Please Uncomment the following lines to run by yourself!!!
# Please Uncomment the following lines to run by yourself!!!
b <- brickFromOutputSoil3DTensor(x,when=when,wpath=wpath,tz=tz,use.read.raster.from.url=TRUE)

# a 2D map:
x_e <- "SnowDepthMapFile"
# Not Run: uncomment the following line
m <- rasterFromOutput2DMap(x_e,when=when,wpath=wpath,timestep="OutputSnowMaps",
tz=tz,use.read.raster.from.url=TRUE)
## NOTE: set use.read.raster.from.url=FALSE (default)
## if the "wpath" directory is in the local file system.
# Not Run: uncomment the following line
plot(m)
## End(Not run)

create.geotop.inpts.keyword

Creates an 'geotop.inpts' files the keyword and their values of a date.frame like the one returned by declared.geotop.inpts.keywords

Description

Creates an 'geotop.inpts' files the keyword and their values of a date.frame like the one returned by declared.geotop.inpts.keywords
create.geotop.inpts.keyword

Usage

create.geotop.inpts.keyword(
  df,
  file = "geotop.inpts.copy",
  wpath = NULL,
  comment.lines = "default",
  header = "default",
  ...
)

Arguments

df    data frame returned by declared.geotop.inpts.keywords
file  connection or file name where to write 'df'
wpath complete path to file (optional). Default is NULL.
comment.lines string or vector of strings to add as comments for each keyword. If it is NULL the comment lines are omitted.
header string or vector of strings to add as a header. If it is NULL the header is omitted.
...
  further arguments for writeLines

Details

In case comment.lines and header are set equal to "default", they are suitably modified within the function code. See the example output.

See Also

writeLines, declared.geotop.inpts.keywords

Examples

library(geotopbricks)
## Not run:
# Simulation working path
wpath <-
'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/panola13_run2xC_test3'
# URL path (RAW VERSION) of
# https://github.com/ecor/geotopbricks_doc/tree/master/simulations/panola13_run2xC_test3
df <- declared.geotop.inpts.keywords(wpath=wpath)
create.geotop.inpts.keyword(df=df)
## End(Not run)
create.geotop.meteo.files

Creates geotop meteo files from (a list of) 'zoo' objects

Description

Creates geotop meteo files from (a list of) 'zoo' objects

Usage

create.geotop.meteo.files(
  x,
  format = "%d/%m/%Y %H:%M",
  file_prefix = "meteo",
  file_extension = ".txt",
  formatter = "%04d",
  na = "-9999",
  col.names = TRUE,
  row.names = FALSE,
  date_field = "Date",
  sep = ",",
  level = NULL,
  quote = FALSE,
  ...)

Arguments

x 'zoo' object or a list of 'zoo' object representing the meteorological station
format string format representing the date, see as.POSIXlt. Default is "%d/%m/%Y %H:%M" (which is the same format used in geotop.inpts keyword InitDateDDMMYYYYhhmm)
file_prefix string containing file prefix (full path). It correspos to the value of in geotop.inpts keyword MeteoFile
file_extension string containing the extensions of final files. Default is c(".txt")
formatter string value. It is the decimal formatter contained in the file name and used in case the tabular data are referred at several points. Default is "%04d". See sprintf.
na NA value indicator. Default is "-9999". See write.table.
col.names logical parameter. Default is TRUE. See write.table.
row.names logical parameter. Default is FALSE. See write.table.
date_field string value. Default is "Date", otherwise defined by the value of HeaderDateDDMMYYYYhhmmMeteo geotop keyword.
sep string value. Default is ",". See write.table.
declared.geotop.inpts.keywords

level integer argument. See get.geotop.inpts.keyword.value for major details. Default is NULL and is ignored.
quote logical parameter. Default is TRUE. See write.table.
...

See Also

write.table, get.geotop.inpts.keyword.value

Examples

library(geotopbricks)
data(bondone) ## It contains a "meteo" zoo object.

set.seed(12)

file_prefix <- paste(tempdir(),"meteo",sep="/")
level=2
out <- create.geotop.meteo.files(x=meteo,file_prefix=file_prefix,level=level)
## It exports the "meteo" zoo object into a ASCII file for GEOtop
head(readLines(out))
out

descended.geotop.inpts.keywords

Collects all keywords contained in the 'getop.inpts' configuration files and their values in a data frame object.

Description

Collects all keywords contained in the 'getop.inpts' configuration files and their values in a data frame object.

Usage

descended.geotop.inpts.keywords(
    wpath,
inpts.file = "getop.inpts",
    comment = "!",
extceptions = "Date",
    warn = FALSE,
    no.comment = c("!>!", "!>>"),
    ...
)


Arguments

- **wpath**: working directory containing GEOtop files.
- **inpts.file**: name of the GEOtop configuration file. Default is "geotop.inpts".
- **comment**: comment indicator character. Default is "!".
- **exceptions**: string vector. If keywords contain an element of this vector, the blank spaces in Value " " will not be removed.
- **warn**: logical argument of `readLines`. Default is FALSE.
- **no.comment**: string indicators read as comment ones by GEOtop but they do not indicate comments by "geotopbricks" package.
- **...**: further arguments of `readLines`.

Value

A data frame with two columns: `Keyword` and `Value`.

See Also

- `get.geotop.inpts.keyword.value`

---

**geotopbrick**

---

Description

`geotopbrick` method bla bla bla.

Usage

```r
geotopbrick(x = NULL, ...)
```

## Default S3 method:

```r
geotopbrick(x, ...)
```

## S3 method for class 'zoo'

```r
geotopbrick(x, layer = NULL, time = NULL, crs = NULL, timerange = NULL, ...)
```

## S3 method for class 'RasterLayer'

```r
geotopbrick(x, layer = NULL, time = NULL, aspath = zoo(NULL), ...)
```

## S3 method for class 'RasterBrick'

```r
geotopbrick(x, layer = NULL, time = NULL, aspath = zoo(NULL), ...)
```

## S3 method for class 'GeotopRasterBrick'

```r
geotopbrick(x = NULL, ...)
```

```r
g eotopbrick(  
   x,
layer = NULL,
time = NULL,
crs = NULL,
timerange = NULL,
ascpath = NULL,
...}

Arguments

- **x**: a 'zoo' object returned by function `pointer.to.maps.xyz.time` or `pointer.to.maps.xy.time` or a `GeotopRasterBrick-class` object
- **layer**: layer at which raster maps are imported. If is NULL, maps are no-z-layer distributed and `zoo` must be returned by `pointer.to.maps.xy.time`
- **time**: vector of time instants at which geotop maps are imported
- **crs**: coordinate system see `RasterBrick-class`
- **timerange**: two-elements vector containing the time range at which geotop maps are imported
- **ascpath**: NULL object or a "zoo" S3 object containing the names of ascii maps provided by GEOtop

Value

A `GeotopRasterBrick-class`

Description

A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!

Details

- **ascpath**: A "zoo" S3 object containing the names of ascii maps provided by GEOtop
- **index**: A "POSIXt" S3 object containing time or dates on which raster layers of brick are referred
- **layer**: character. Name of the vertical layer at which raster map are referred
- **brick**: A "RasterBrick-class" S4 object containing the Raster-Layer maps imported from GEOtop output files

# @note A GeotopRasterBrick object can be created by `new("GeotopRasterBrick", ...)`
get.geotop.inpts.keyword.value

**Author(s)**

Emanuele Cordano

**See Also**

Raster-class

**Examples**

```r
showClass("GeotopRasterBrick")
```

---

**get.geotop.inpts.keyword.value**

*Importing a GEOtop Keyword and its Value into R*

**Description**

It returns the values of a keyword of "geotop.inpts" file or data frame with the suitable format.

**Usage**

```r
get.geotop.inpts.keyword.value(
  keyword,
  inpts.frame = NULL,
  vector_sep = NULL,
  col_sep = ",",
  numeric = FALSE,
  format = "%d/%m/%Y %H:%M",
  date = FALSE,
  tz = "Etc/GMT-1",
  raster = FALSE,
  file_extension = ".asc",
  add_wpath = FALSE,
  wpath = NULL,
  use.read.raster.from.url = TRUE,
  data.frame = FALSE,
  formatter = "%04d",
  level = 1,
  date_field = "Date",
  isNA = -9999,
  matlab.syntax = TRUE,
  projfile = "geotop.proj",
  start_date = NULL,
  end_date = NULL,
)```
ContinuousRecovery = 0,
ContinuousRecoveryFormatter = "_crec%04d",
zlayer.formatter = NULL,
z_unit = c("centimeters", "millimeters"),
geotop_z_unit = "millimeters",
add_suffix_dir = NULL,
MAXNROW = 4,
header.only = FALSE,
...)

Arguments

keyword keyword name
inpts.frame data frame returned by declared.geotop.inpts.keywords or NULL. Default is
NULL.
vector_sep character value for the separator character if Keyword Value must be returned
as a vector, otherwise it is NULL. Default is NULL, but if numeric or date are
FALSE, vector_sep is set "," by default.
col_sep character value for the separator character of columns. It is used if Keyword
Value is returned as a data frame or zoo object or list of these objects. Default is
NULL, but is set ",".
numeric logical value. If TRUE the Value has numeric type, otherwise it is a string or
string vector. Default is FALSE.
format string format representing the date, see as.POSIXlt, used if date is TRUE. De-
dfault is "%d/%m/%Y %H:%M" (which is the format used in geotop.inpts keyword
InitDateDDMMYYYYhhmm)
date logical value. If TRUE the Value is retured as POSIXlt date, otherwise it is a
string or string vector. Default is FALSE.
tz format string representing the time zone, see as.POSIXlt, used if date is TRUE. De-
dfault is "Etc/GMT-1" (until the previous version it was "A") which means
UTC +1.
raster logical value. Default is FALSE. If TRUE function returns directly the raster map
as Raster-class object built with raster method.
file_extension Extension to be added to the keyword if keyword is a file name. Default is
".asc"
add_wpath logical value. Default is FALSE. If TRUE, the wpath string is attached to the
keyword string value. It is automatically set TRUE if raster is TRUE.
wpath working directory containing GEOtop files (included the inpts file). It is mandatory
if raster is TRUE. See declared.geotop.inpts.keywords.
use.read.raster.from.url logical value. Default is TRUE. If TRUE the RasterLayer are read with read.raster.from.url,
instead of raster (otherwise). It is recomended in case the files whose paths are
contained in x are remote and are 'http' addresses. In this cases the stand-alone
method raster(x) does not always work and use.read.raster.from.url is
necessary.
**data.frame**

logical value. It is an option for tabular data. If TRUE function returns directly a data frame or a list of data frames as `data.frame` or `zoo` objects imported from the keyword-related files using `read.table` function. In this case the argument `wpath` (see `declare.geotop.inpts.keywords`) is mandatory. Default is FALSE.

**formatter**

string value. It is the decimal formatter contained in the file name and used in case the tabular data are referred at several points. Default is "%04d". It is used in case `data.frame` is TRUE.

**level**

integer values. Numbers incating all the identification numbers of the files containing the requested data frames. Default is 1, corresponding to the decimal formatter "0001". See examples.

**date_field**

string value. Default is "Date", otherwise defined by the value of `HeaderDateDDMMYYYYhmmMeteo geotop keyword`. It is used only if the argument `data.frame` is TRUE. If it is NULL or NA the function return a list of generic `data.frame` object(s), otherwise `link{zoo}` object(s). See the arguments `tz` and `format` for Date formatting.

**isNA**

numeric value indicating NA in `geotop ascii files`. Default is -9999.00

**matlab.syntax**

logical value. Default is FALSE. If TRUE a vector is written in a string according to *.m file syntax. Warning: this syntax is not read by `GEOtop`.

**projfile**

filename of the `GEOtop` projection file. Default is `geotop.proj`.

**start_date, end_date**

null objects or dates in POSIXlt format between which the variables are returned. It is enabled in case that `date_field` is not NULL or NA and `data.frame` is TRUE. Default is NULL.

**ContinuousRecovery**

integer value. Default is 0. It is used for tabular output data and is the number of times `GEOtop` simulation broke during its running and was re-launched with 'Continuous Recovery' option.

**ContinuousRecoveryFormatter**

character string. Default is '_crec%04d'. It is used only for tabular output data and if ContinuousRecovery is equal or greater than 1.

**zlayer.formatter**

decimal formatter. It is used if `data.frame==TRUE` and the columns refers to different soil depths. Default is NULL.

**z_unit**

`z` coordinate measurement unit. `GEOtop` values expressed in millimeters which are converted to centimeters by default. Default is c("centimeters", "millimeters"). Otherwise can be the ratio between the unit and one meter. It is used if `zlayer.formatter=="z%04d"` or similar.

**geotop_z_unit**

`z` coordinate measurement unit used by `GEOtop`. Default is millimeters. It is used if `zlayer.formatter=="z%04d"` or similar.

**add_suffix_dir**

character string. Add a suffix at the directory reported in the keyword value

**MAXNROW**

maximum number accepted for `data.frame` output. Default is 4. It is used in case of `data.frame==TRUE`. In case the number of records in the function output is less than MAXNROW, function returns neither `data.frame` nor `zoo` objects but only the keyword value.
get.geotop.inpts.keyword.value

header.only logical value. Default is FALSE. If it is TRUE and data.frame==TRUE, only file header with variable names is returned by the function.

... further arguments of declared.geotop.inpts.keywords

Value

the keyword value

Note

If inpts.frame is NULL, inpts.frame will be obtained by calling the function declared.geotop.inpts.keywords with ... arguments.

Examples

library(geotopbricks)

#Simulation working path

wpath <- 'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/panola13_run2xC_test3'  
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/panola13_run2xC_test3
prefix <- get.geotop.inpts.keyword.value("SoilLiqWaterPressTensorFile",wpath=wpath)

slope <- get.geotop.inpts.keyword.value("SlopeMapFile",raster=TRUE,wpath=wpath)
bedrock_depth <- get.geotop.inpts.keyword.value("BedrockDepthMapFile",raster=TRUE,wpath=wpath)

layers <- get.geotop.inpts.keyword.value("SoilLayerThicknesses",numeric=TRUE,wpath=wpath)
names(layers) <- paste("L",1:length(layers),sep="")

##### set van genuchten parameters to estimate water volume
theta_sat <- get.geotop.inpts.keyword.value("ThetaSat",numeric=TRUE,wpath=wpath)
theta_res <- get.geotop.inpts.keyword.value("ThetaRes",numeric=TRUE,wpath=wpath)
alpahVG <- get.geotop.inpts.keyword.value("AlphaVanGenuchten",numeric=TRUE,wpath=wpath) # expressed in mm^-1

nVG <- get.geotop.inpts.keyword.value("NVanGenuchten",numeric=TRUE,wpath=wpath)

##### end set van genuchten parameters to estimate water volume

##### set meteo data

tz <- "Etc/GMT-1"  
## See help(timezones) In particular:
## Most platforms support time zones of the form Etc/GMT+n
## and Etc/GMT-n (possibly also without prefix Etc/),
## which assume a fixed offset from UTC (hence no DST).
## Contrary to some expectations
## (but consistent with names such as PST8PDT), negative offsets are times ahead of (east of) UTC,
## positive offsets are times behind (west of) UTC.

```r
start <- get.geotop.inpts.keyword.value("InitDateDDMMYYYYhhmm",
date=TRUE,wpath=wpath,tz=tz)
end <- get.geotop.inpts.keyword.value("EndDateDDMMYYYYhhmm",
date=TRUE,wpath=wpath,tz=tz)
```

```r
nmeteo <- get.geotop.inpts.keyword.value("NumberOfMeteoStations",
numeric=TRUE,wpath=wpath)
level <- 1:nmeteo
```

```r
## set meteo data
meteo <- get.geotop.inpts.keyword.value("MeteoFile",wpath=wpath,data.frame=TRUE,
level=level,start_date=start,end_date=end,tz=tz)
```

##### end set meteo data

## IMPORTING AN OUTPUT SOIL MOISTURE PROFILE:

```r
wpath <- paste0( 
'https://raw.githubusercontent.com/ecor/geotopbricks_doc/','
'master/simulations/Muntatschini_pnt_1_225_B2_004')
```

```r
## URL Path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/Muntatschini_pnt_1_225_B2_004
```

```r
SMC <- get.geotop.inpts.keyword.value("SoilLiqContentProfileFile",
wpath=wpath,data.frame=TRUE,date_field="Date12.DDMMYYYYhhmm.",
formatter="%04d")
SMCz <- get.geotop.inpts.keyword.value("SoilLiqContentProfileFile",
wpath=wpath,data.frame=TRUE,date_field="Date12.DDMMYYYYhhmm.",
formatter="%04d",zlayer.formatter="z%04d")
```

---

**get.geotop.points**

Get a *sf* object for Meteorological Stations or Control Points in a GEOtop simulation
Description

Get a sf object for Meteorological Stations or Control Points in a GEOtop simulation

Usage

```r
get.geotop.points(
  prefix = c("MeteoStation", "CoordinatePoint"),
  suffixes = c("Code", "Elevation", "Source"),
  coord_suffixes = list(MeteoStation = c("CoordinateX", "CoordinateY"),
                        CoordinatePoint = c("X", "Y")),
  wpath,
  ...,vector_sep = ",",
)
```

Arguments

- `prefix`: keyword prefix
- `suffixes`: keyword suffixes
- `coord_suffixes`: coordinate keyword suffixes. Default is `c("PointX","PointY")`
- `wpath`: GEOtop simulation path
- `vector_sep`: further arguments for `get.geotop.inpts.keyword.value`

Examples

```r
### See simulation template: "https://github.com/ecor/geotopbricks_doc/tree/master/template/sumava"

wpath <- "https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/template/sumava/"

## system.file("template/sumava",package="geotopbricks")

out <- get.geotop.points(wpath=wpath)

out <- get.geotop.points(prefix="CoordinatePoint",suffix=c("Code","Source"),wpath=wpath)

out <- get.geotop.points(prefix="MeteoStation",suffix=c("Code","Source"),wpath=wpath)
```

---

get.geotop.recovery.state

This function saves all spatially distributed information contained in the recovery folder into a comprehensive list object.

Description

This function saves all spatially distributed information contained in the recovery folder into a comprehensive list object.
Usage

get.geotop.recovery.state(
  recFolder,  
  xx = "0000",  
  formatter = "L%04d",  
  extension = ".asc",  
  nsoillayers = 10,  
  layersFromDir = FALSE,  
  ...
)

Arguments

recFolder directory when recovery maps are set. In GEOtop it is ...

xx character String. Default is "0000"

formatter string character for the the decimal formatter to be used. Default is "L%04d".

extension file extension used for ascii recovery map files. It must contains /quotesingle.Var/quotesingle.Var as the first character. Defaut is ".asc".

nsoillayers number of soil layers used in the GEOtop simulation.

layersFromDir logical value. If is TRUE the number of soil/snow (vertical) layers used in the GEOtop simulation is automatically calculated and cannot be assigned through nsoillayers.

... further arguments

Value

a list object containining all recovery raster maps.

Note

This function has been used with the built 1.225-9 of GEOtop.

Author(s)

Emanuele Cordano

See Also

brick.decimal.formatter,
raster.set.geotop.recovery.state,
write.vectorized.geotop.recovery.read.vectorized.geotop.recovery
getProjection

Examples

```r
library(geotopbricks)
example_Rscript <- system.file('template/example.geotop.recovery.state.R', package="geotopbricks")
exmple_Rscript

# Not Run because it elapses too long time!!!
# Please Uncomment the following line to run by yourself!!!
# source(example_Rscript)
```

getProjection

It reads the CRS metadata utilized in a GEOtop Simulation

Description

It reads the CRS metadata utilized in a GEOtop Simulation

Usage

```r
getProjection(x, cond = TRUE, ...)
```

Arguments

- `x` name and full path of the file containing CRS information
- `cond` logical value. If FALSE the function returns NA. Default is TRUE.
- `...` further arguments

Value

A string corresponding the projection and CRS if the argument cond is TRUE.

Examples

```r
library(geotopbricks)

wpath <- 'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/idroclim_test1'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/idroclim_test1
## Not run:

x <- paste(wpath,"geotop.proj",sep="/"")

crs <- getProjection(x)

## End(Not run)
```
**getvalues.brick.at.depth**

_Interpolates the values of a 'brick' at a certain depth and returns the map of brick values at the "depth" level_

---

**Description**

Interpolates the values of a 'brick' at a certain depth and returns the map of brick values at the "depth" level

**Usage**

`getvalues.brick.at.depth(x, depth, layers, i0 = NULL, verify = FALSE, ...)`

**Arguments**

- `x`: a `RasterBrick` or a three-dimensional array
- `depth`: depth map, generally a `RasterLayer` object
- `layers`: vector of layer thickness
- `i0`: a `Raster` containing the number of soil layer just over the bedrock. Default is `NULL` and is then calculated.
- `verify`: logical. Default is `FALSE`. If it is `TRUE`, it verifies that function is working correctly.
- `...`: further argument

**Value**

a list of `Raster` maps:

- `i0`: a `Raster` containing the number of soil layer just over the bedrock
- `val_z0`: a `Raster` containing the values of `x` at the `i0`-th layer
- `val_z1`: a `Raster` containing the values of `x` at the `(i0+1)`-th layer
- `z0`: a `Raster` containing the depth of the center of the `i0`-th layer
- `z1`: a `Raster` containing the depth of the center of the `(i0+1)`-th layer

**Note**

`x` and `depth` or `i0` must cover the same spatial region.

**See Also**

`vertical.aggregate.brick.within.depth`
Examples

```r
library(geotopbricks)
# The examples is the following R script contained in a 'inst' directory of the package source
f <- system.file("doc/examples/example.getvalues.brick.at.depth.R",package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=...,...) See file.copy documentation
```

---

**Description**

KML method for a GeotopRasterBrick object

**Usage**

```r
## S4 method for signature 'GeotopRasterBrick'
KML(
  x,
  filename,
  crs = as.character("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"),
  ...
)
```

**Arguments**

- `x` the GeotopRasterBrick object
- `filename` name of the KML file to produce
- `crs` character string containing the LatLon reference system. Default is "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs" (see [https://spatialreference.org/ref/epsg/4326/](https://spatialreference.org/ref/epsg/4326/)).
- `...` further argument for S4 method KLM for Raster object.

**Note**

A coordinate transformation is made with `projectRaster`.

**Examples**

```r
library(geotopbricks)
# The examples is the following R script contained in a 'inst' directory of the package source
f <- system.file("doc/examples/example.KML.GeotopRasterBrick.R",package="geotopbricks")
## Not run:
source(f) # Uncomment this line to run the example.
```
Description

Gets the maximum (scalar) values of a GeotopRasterBrick object

Usage

max_value(x, na.rm = TRUE, ...)

Arguments

x a GeotopRasterBrick object.
na.rm, ... further arguments for max.

Value

the maximum (scalar) values of a GeotopRasterBrick object

Description

Gets the minimum (scalar) values of a GeotopRasterBrick object

Usage

min_value(x, na.rm = TRUE, ...)

Arguments

x a GeotopRasterBrick object.
na.rm, ... further arguments for min.

Value

the minimum (scalar) values of a GeotopRasterBrick object
Description

Ops

Usage

## S4 method for signature 'GeotopRasterBrick, GeotopRasterBrick'
Ops(e1, e2)

## S4 method for signature 'GeotopRasterBrick, numeric'
Ops(e1, e2)

## S4 method for signature 'numeric, GeotopRasterBrick'
Ops(e1, e2)

Arguments

e1, e2 the GeotopRasterBrick or numeric objects

Details

Ops method for a GeotopRasterBrick object

Note

If e1 or e2 time index is not taken into account.

Description

plot method for a GeotopRasterBrick object

Usage

## S4 method for signature 'GeotopRasterBrick, ANY'
plot(x, y = NULL, ...)

Arguments

x the GeotopRasterBrick object

y further argument

... further argument for S4 method plot for Raster object.
See Also

KML

Examples

library(geotopbricks)
# The examples is the following R script conteined in a 'inst' directory of the package source
f <- system.file("doc/examples/example.plot.GeotopRasterBrick.R",package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=....,...) See file.copy documentation

Description

'pointer.to.maps.xyz.time' function (obsolete)

Usage

pointer.to.maps.xyz.time(
  wpath,
  map.prefix = "thetaliq",
  suffix = "L%04dN%04d.asc",
  zoo.index = NULL,
  ntime,
  nlayers
)

Arguments

wpath complete working path to *.asc maps are saved
map.prefix string prefix name map before
suffix z-time or time suffix plus file extention character string. Default for GEOtop application is "L%04dN%04d.asc" for xy+z+time maps or "N%04d.asc" for xy+time maps.
read.ascii.vectorized.brick

Read a text file containing values and metadata of a z-layer brick referred to a time instant (e.g. date). The file is formatted like an ascii format like 'geotop.inpts' file.

Description

Read a text file containing values and metadata of a z-layer brick referred to a time instant (e.g. date). The file is formatted like an ascii format like 'geotop.inpts' file.

Usage

read.ascii.vectorized.brick(
  file = NULL,
  comment = "!",
  crs = "",
  NAflag = -9999,
  matlab.syntax = FALSE,
  ...
)

Arguments

  file          file name to write
  comment       character. Comment indicator. Default is "!".
  crs           Character or object of class CRS. PROJ4 type description of a Coordinate Reference System (map projection) (optional). See brick or raster.
  NAflag        numeric. Dafault is -9999, see writeRasterxGEOTop.
  matlab.syntax logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
  ...           further arguments inserted as attribute
**Value**

the `RasterBrick-class` object

**See Also**

`write.ascii.vectorized.brick`

**Examples**

```r
# see the examples of read.ascii.vectorized.brick
```

---

**Description**

It imports a 'RasterLayer' object in Escri-Asci format from a URL 'http(s):/....<FILENAME>.asc'

**Usage**

```r
read.raster.from.url(x, header_nrow = 6, ...)
```

**Arguments**

- `x` the character string containing the URL address
- `...` additional arguments

**Value**

a 'RasterLayer' object

**Note**

This function reads a local or remote text files formatted as [https://en.wikipedia.org/wiki/Esri_grid](https://en.wikipedia.org/wiki/Esri_grid) and creates a 'RasterLayer' object.

**See Also**

`raster.readLines`
Description

Reads a text file like the one generated by write.vectorized.geotop.recovery

Usage

read.vectorized.geotop.recovery(
  file = file,
  comment = "!",
  matlab.syntax = TRUE,
  xx = "0000",
  formatter = "L%04d",
  extension = ".asc",
  NAflag = -9999,
  crs = "",
  ...
)

Arguments

file file name to write
comment character. Comment indicator. Default is "!".
matlab.syntax logical value. Default is TRUE. If TRUE the file syntax is like the one of a *.m Matlab script file.
formatter, extension, xx see get.geotop.recovery.state.
NAflag numeric. Default is -9999, see writeRasterxGEOtop.
crs Character or object of class CRS. PROJ4 type description of a Coordinate Reference System (map projection) (optional). See brick or raster.
... further arguments inserted as attribute

Value

a list object like get.geotop.recovery.state

See Also

write.vectorized.geotop.recovery
Examples

# see the examples of read.ascii.vectorized.brick

replace.keyword

It replaces some keyword values of geotop.inpts file with the ones of another *.inpts value

Description

It replaces some keyword values of geotop.inpts file with the ones of another *.inpts value

Usage

replace.keyword(
  x,
  y = "geotop.inpts",
  file.output = NULL,
  write.file.output = TRUE,
  wpath = NULL,
  ...
)

Arguments

x filename of the *.inpts with the "new" keyword value
y filename of the *.inpts with the "old" keyword value. Default is "geotop.inpts".
file.output filename where to write the comprehensive new geotop.inpts file. If it is NULL (default), the filename is assigned by y.
write.file.output logical value. If it is TRUE, the output of the function is written in the file file.output.
wpath working path to the GEOtop simulation folder containing the x and y files.
... further arguments

Details

This function replaces some keyword values of y with the ones indicated in y. It is useful to replace the meteo station metadata, for instance, when the meteorological station of a study case is modified. The function returns the new geotop.inpts file as a vector of character strings. If write.file.output==TRUE, the output is written in an external file, e.g. "geotop.inpts" newly (this option is suggested).

Author(s)

Emanuele Cordano
Examples

```r
library(geotopbricks)
wpath <- system.file('template/meteo_ex', package="geotopbricks")
x <- "meteo.inpts"
zl <- replace.keyword(x, wpath=wpath, write.file.output=FALSE)
```

Description

This function re-writes the recovery ascii raster maps in a given folder

Usage

```r
set.geotop.recovery.state(rec, newRecFolder, ...)
```

Arguments

- `rec` a list object returned by `get.geotop.recovery.state`
- `newRecFolder` directory where to write all recovery raster ascii maps
- `...` further arguments

Author(s)

Emanuele Cordano

See Also

`get.geotop.recovery.state, writeRasterxGE0top`

Examples

```
# See the examples of the 'get.geotop.recovery.state' function
```
vertical.aggregate.brick.within.depth

Aggregates with a mean or an addition on the vertical profile the values of a 'brick' within a certain depth and returns the vertical aggregated map

Description

Aggregates with a mean or an addition on the vertical profile the values of a 'brick' within a certain depth and returns the vertical aggregated map

Usage

vertical.aggregate.brick.within.depth(
  x,
  depth = NULL,
  layers = NULL,
  i0 = NULL,
  verify = FALSE,
  FUN = identity,
  divide.by.depth = FALSE,
  ...
)

Arguments

- **x**: a 'RasterBrick' or a three-dimensional array
- **depth**: depth map, generally a 'RasterLayer' object
- **layers**: vector of layer thickness
- **i0**: a 'Raster' containing the number of soil layer just over the bedrock. Default is NULL and is then calculated.
- **verify**: logical. Default is FALSE. If it is TRUE, it verifies that function is working correctly.
- **FUN**: function used for aggregation. If missing, identity is the default value.
- **divide.by.depth**: logical. If TRUE the function returns the 'mean' value, otherwise a a cumulate value. Default is FALSE.
- **...**: further argument for FUN

Value

- a list of 'Raster' maps:
  - **i0**: a 'Raster' containing the number of soil layer just over the bedrock
  - **z0**: a 'Raster' containing the depth of the center of the i0-th layer
  - **result**: a 'Raster' containing the aggregated map
Note

x and depth or i0 must cover the same spatial region.

See Also

getvalues.brick.at.depth.brick

Examples

library(geotopbricks)
# The examples is the following R script contented
# in a 'inst' directory of the package source
f <- system.file("doc/examples/example.vertical.aggregate.brick.within.depth.R", package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=.....) See file.copy documentation
## Arguments

- **b**
  - a RasterBrick-class or GeotopRasterBrick-class object
- **file**
  - file name to write
- **header**
  - character string vector for header text lines. If missing, a default header is written. Default is c("! header").
- **overwrite**
  - logical. Default is TRUE, see writeRaster.
- **NAflag**
  - numeric. Default is -9999, see writeRasterGEOTop.
- **matlab.syntax**
  - logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
- ... further arguments inserted as attribute

## Value

the string vector possibly written in file.

## Note

Add Quote if necessary. This function is NOT maintained and will be DEPRECATED.

## See Also

read.ascii.vectorized.brick

## Examples

```r
## Not Run
## library(geotopbricks)
## library(raster)
## file <- system.file("doc/examples/snowthickness",package="geotopbricks")
## file <- paste(file,"SnowThickness0000L%04d.asc",sep="/"")
## b <- brick.decimal.formatter(file=file,nlayers=15)
## nlayers(b)
## names(b)
## file <- "snow.txt"
## btext <- write.ascii.vectorized.brick(b,Date="1/1/2009",file="snow.txt")
## The printed object
## str(btext)
## bb <- read.ascii.vectorized.brick(file = file)
## bf <- abs(as.matrix(bb[[1]]-b[[1]]))<.Machine$double.eps^0.5
```
write.geotop.table  Writes an R object (data.frame or zoo) into a CSV file readable by GEOtop.

Description

Writes an R object (data.frame or zoo) into a CSV file readable by GEOtop.

Usage

```r
write.geotop.table(
  x,  
  file,  
  wpath = NULL,  
  tz = "Etc/GMT-1",  
  date_field = "Date12.DDMMYYYYhhmm.",  
  file_end = "",  
  sep = ",",  
  format = "%d/%m/%Y %H:%M",  
  na = "-9999",  
  ...  
)
```

Arguments

- **x**: R object (data.frame or zoo) to be exported and written.
- **file**: filename
- **wpath**: working path to the GEOtop simlation. If wpath is not NULL, filename will be put in wpath.
- **tz**: time zone. Default is "Etc/GMT-1". See `get.geotop.inpts.keyword.value` for further details.
- **date_field**: string used for date-time field. Default is "Date12.DDMMYYYYhhmm.". See `get.geotop.inpts.keyword.value` for further details.
- **file_end**: suffix of the file name (file) (optional). Default is "".
- **sep**: separator character. Default is ",". See `write.table` for further details.
- **format**: date time format. Default is "%d/%m/%Y %H:%M". See `get.geotop.inpts.keyword.value` for further details.
- **na**: string for unassigned values. Default is "-9999". See `write.table` for further details.
- **...**: further arguments for `write.table`. 
### Description

It writes a list object returned by `get.geotop.recovery.state` as a string vector or in a text file, following `*.inpts` or Matlab-like syntax.

### Usage

```r
write.vectorized.geotop.recovery(
  rec,
  file = NULL,
  header = NULL,
  overwrite = TRUE,
  NAflag = -9999,
  matlab.syntax = TRUE,
  ...
)
```

### Arguments

- **rec**: a list object returned by `get.geotop.recovery.state`
- **file**: ascii text file name where to write the string vector
- **header**: character string vector for header text lines. If missing, a default header is written. Default is `c("! header")` or the one assigned by `matlab.syntax`.
- **overwrite**: logical. Default is `TRUE`, see `writeRaster`.
- **NAflag**: numeric. Default is `-9999`, see `writeRasterGEOtop`.
- **matlab.syntax**: logical value. Default is `TRUE`. If `TRUE` the file syntax is like the one of a `*.m` Matlab script file.
- **...**: further arguments inserted as attribute

### Value

a string vector containing the `rec` variables.

### Note

Add Quote if necessary

### See Also

- `get.geotop.recovery.state`, `set.geotop.recovery.state`, `write.vectorized.variable.in.string`
write.vectorized.variable.in.string

   Writes one or more variables (scalars, vectors or Rasters) in a string each, following *.inpts or Matlab-like syntax.

Description

Writes one or more variables (scalars, vectors or Rasters) in a string each, following *.inpts or Matlab-like syntax.

Usage

write.vectorized.variable.in.string(  
   l,  
   NAflag = -9999,  
   matlab.syntax = FALSE,  
   ...  
)

Arguments

l          a list object contained the variables (scalars, vectors or Rasters) which will be written in a string each.
NAflag     numeric. Default is -9999, see writeRasterxGEOtop.
matlab.syntax logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
...        further arguments

Value

the string vector <NAME_VARIABLE>=<VALUES_VARIABLE>.

Note

Add Quote if necessary

See Also

read.ascii.vectorized.brick
writeRasterxGEOtop

Examples

```r
a <- 1:5
l <- list(v=a,a=a)
out <- write.vectorized.variable.in.string(l,matlab.syntax=TRUE)
out
```

writeRasterxGEOtop

This function uses `writeRaster` to create .asc maps which can be read by GEOtop

Description

This function uses `writeRaster` to create .asc maps which can be read by GEOtop

Usage

```r
writeRasterxGEOtop(
  x,
  filename = NULL,
  overwrite = TRUE,
  NAflag = -9999,
  use.decimal.formatter = FALSE,
  start.from.zero = FALSE,
  keyword,
  wpath,
  suffix.ext = ".asc",
  ...
)
```

Arguments

- **x**: a Raster object, see `writeRaster`. It can be also a `RasterBrick-class` object.
- **filename**: see `writeRaster`. It is a vector of string or one string containing a decimal formatter (see `brick.decimal.formatter`) in case `x` is a `RasterBrick-class` object.
- **overwrite**: logical. Default is `TRUE`, see `writeRaster`.
- **NAflag**: numeric. Dafault is `-9999`, see `writeRaster`.
- **use.decimal.formatter**: logical value. Default is `FALSE`. If it is `TRUE` or `x` is a `RasterBrick-class` object with `nlayers(x)!="length(filename)`, `filename` is considered as one string containing a decimal formatter (e.g. "%04d", see `brick.decimal.formatter`). Otherwise, if `filename` is considered as a vector string.
start.from.zero

  logical value. Default is FALSE. If TRUE the formatter starts from 0000, otherwise it starts from 0001.

keyword

  geotop keyword to be used to extract the raster file name from geotop.inpts file. This is enabled if filename is equal to NULL.

wpath

  simulation folder containing geotop.inpts file.

suffix.ext

  character string to be added to the keyword value, e.g. possible suffix and extension of the raster file name. Default is ".asc".

... further arguments of get.geotop.inpts.keyword.value or writeRaster

Note

  It makes use of system functions. It uses *.asc format for raster files. In case the file name filename is missing and then NULL, it must be imported by the simulation geotop.inpts file.

Examples

library(geotopbricks)

## Simulation working path

#file <- paste0("https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/template/",
  "rendena100/SnowDepthMapFile-2014-MA-mean-winter-2013-2014.asc")
file <- system.file("ex/elev.tif", package="terra")
elev <- raster(file)

elevfile <- rasterTmpFile()
extension(elevfile) <- ".asc"

writeRasterxGEOtop(x=elev,file=elevfile)

zoo-class

  A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!

Description

  A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!

Examples

  showClass("zoo")
# Index

* **classes**  
  GeotopRasterBrick-class, 15
* **dataset**  
  bondone, 3
* **methods**  
  Ops, 27
  
  argsParser, 2  
  as.POSIXlt, 12, 17
  
  bondone, 3  
  brick, 4, 6, 29, 31, 35  
  brick, GeotopRasterBrick-method (brick), 4  
  brick, zoo-method (brick), 4  
  brick.decimal.formatter, 5, 8, 10, 22, 40  
  brickFromOutputSoil3DTensor, 7, 8, 9  
  
  create.geotop.inpts.keyword, 10  
  create.geotop.meteo.files, 12
  
  data.frame, 18
  declared.geotop.inpts.keywords, 10, 11, 13, 17–19
  
  geotopbrick, 14
  GeotopRasterBrick, 25–27
  GeotopRasterBrick  
  (GeotopRasterBrick-class), 15
  GeotopRasterBrick-class, 15
  get.geotop.inpts.keyword.value, 8, 10, 13, 14, 16, 21, 37, 41
  get.geotop.points, 20
  get.geotop.recovery.state, 21, 31, 33, 38
  getProjection, 23
  getvalues.brick.at.depth, 5, 24, 35
  
  identity, 34
  
  KML, 25, 28
  KML, GeotopRasterBrick-method (KML), 25
  list, 31  
  max, 26  
  max_value, 26  
  meteo (bondone), 3  
  min, 26  
  min_value, 26
  
  Ops, 27
  Ops, GeotopRasterBrick, GeotopRasterBrick-method (Ops), 27
  Ops, GeotopRasterBrick, numeric-method (Ops), 27
  Ops, numeric, GeotopRasterBrick-method (Ops), 27
  
  plot, 27
  plot, GeotopRasterBrick, ANY-method (plot), 27
  pointer.to.maps.xy.time, 5, 15  
  pointer.to.maps.xy.time  
  (pointer.to.maps.xyz.time), 28  
  pointer.to.maps.xyz.time, 5, 15, 28  
  POSIXlt, 17
  projectRaster, 25
  
  raster, 5, 6, 17, 22, 29–31
  rasterFromOutput2DMap, 9
  rasterFromOutput2DMap  
  (brickFromOutputSoil3DTensor), 7
  read.ascii.vectorized.brick, 29, 36, 39
  read.raster.from.url, 5, 6, 17, 30
  read.table, 18
  read.vectorized.geotop.recovery, 22, 31
  readLines, 14, 30
  replace.keyword, 32
  
  set.geotop.recovery.state, 22, 33, 38
  sf, 20, 21
  sprintf, 12
system, 41
vertical.aggregate.brick.within.depth, 5, 24, 34
write.ascii.vectorized.brick, 30, 35
write.geotop.table, 37
write.table, 12, 13, 37
write.vectorized.geotop.recovery, 22, 31, 38
write.vectorized.variable.in.string, 38, 39
writeLines, 11
writeRaster, 36, 38, 40, 41
writeRasterxGEotop, 29, 31, 33, 36, 38, 39, 40
zoo, 3, 18
zoo-class, 41