Package 'rdwd'

October 28, 2019

Title Select and Download Climate Data from 'DWD' (German Weather Service)

Version 1.2.0

Date 2019-10-27

Depends R(>= 2.10)

SystemRequirements Pandoc (>= 1.12.3), pandoc-citeproc

Imports berryFunctions (>= 1.17.0), pbapply

Suggests RCurl, leaflet, knitr, rmarkdown, testthat, roxygen2, data.table, OSMscale, raster, R.utils, ncdf4, readr, dwdradar, XML

Author Berry Boessenkool

Maintainer Berry Boessenkool <berry-b@gmx.de>

Description Handle climate data from the 'DWD' ('Deutscher Wetterdienst', see <https://www.dwd.de/EN/climate_environment/cdc/cdc.html> for more information). Choose files with 'selectDWD()', download and process data sets with 'dataDWD()' and 'readDWD()'.

License GPL (>= 2)

Encoding UTF-8

URL https://github.com/brry/rdwd

RoxygenNote 6.1.1

BugReports https://github.com/brry/rdwd/issues

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-10-28 05:40:11 UTC

R topics documented:

`addBorders` *add country and Bundesland borders to a map*

Description

add country and Bundesland borders to a map

Usage

```
addBorders(de = "grey80", eu = "black", add = TRUE, ...)
```

Arguments

de	Color for Bandeslaender line. NA to suppress. DEFAULT: "grey80"
eu	Color for countries line. NA to suppress. DEFAULT: "black"
add	Logical: add to existing plot? DEFAULT: TRUE
...	Further arguments passed to raster::plot

Value

invisible list with DEU and EUR

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2019

See Also

DEU, EUR

Examples

```
plot(1, xlim=c(2,16), ylim=c(47,55))
addBorders()
plot(1, xlim=c(2,16), ylim=c(47,55))
addBorders(de="orange", eu=NA)
```

---

checkIndex *check indexes*

Description

check indexes. Mainly for internal usage in createIndex. Not exported, so call it as rdwd:::checkIndex() if you want to run tests yourself. Further test suggestions are welcome!

Usage

```
checkIndex(findex = NULL, mindex = NULL, gindex = NULL,
  excludefp = TRUE, fast = FALSE, warn = TRUE)
```

Arguments

findex	fileIndex. DEFAULT: NULL
mindex	metaIndex. DEFAULT: NULL
gindex	geoIndex. DEFAULT: NULL
excludefp	Exclude false positives from geoIndex coordinate check results? DEFAULT: TRUE
fast	Exclude the 3-minute location per ID check? DEFAULT: FALSE
warn	Warn about issues? DEFAULT: TRUE

Value

Charstring with issues (if any) to be printed with `cat()`.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, May 2019

See Also

createIndex

Examples

```
data(fileIndex) ; data(metaIndex) ; data(geoIndex)
# ci <- checkIndex(findex=fileIndex, mindex=metaIndex, gindex=geoIndex)
# cat(ci)
```

---

checkSuggestedPackage *check suggested package for availability*

Description

check suggested package for availability, yielding an instructive error message if not

Usage

```
checkSuggestedPackage(package, functionname)
```

Arguments

package	Charstring: package to be checked for loadability
functionname	Charstring: function name to be used in the message

Value

invisible success logical value from requireNamespace

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2019

See Also

requireNamespace

---

createIndex *Create file and meta index of the DWD CDC FTP Server*

Description

This is mainly an internal function. Create data.frames out of the vector index returned by indexFTP. For fileIndex (the first output element) createIndex tries to obtain res, var, per, file, id, start and end from the paths. If meta=TRUE, metaIndex and geoIndex are also created. They combine all Beschreibung files into a single data.frame.
If you create your own index as suggested in selectDWD (argument findex), you can read the produced file as shown in the example section.

Usage

```
createIndex(paths, base = dwdbase, dir = "DWDdata",
  fname = "fileIndex.txt", meta = FALSE, metadir = "meta",
  mname = "metaIndex.txt", gname = "geoIndex.txt", overwrite = FALSE,
  checkwarn = TRUE, quiet = FALSE, ...)
```

Arguments

paths	Char: vector of DWD paths returned by indexFTP called with the same base value as this function
base	Main directory of DWD ftp server, defaulting to observed climatic records. DEFAULT: dwdbase
dir	Char: writeable directory name where to save the main output(s). Created if not existent. DEFAULT: "DWDdata" at current getwd()
fname	Char: Name of file in dir in which to write fileIndex. Use fname="" to suppress writing. DEFAULT: "fileIndex.txt"
meta	Logical: should metaIndex also be created from fileIndex? Uses dataDWD to download files if not present. DEFAULT: FALSE
metadir	Char: Directory (subfolder of dir) where original description files are downloaded to if meta=TRUE. Passed to dataDWD. "" to write in dir. DEFAULT: "meta"
mname	Char: Name of file in dir (not metadir) in which to write metaIndex. Use mname="" to suppress writing. DEFAULT: "metaIndex.txt"
gname	Filename for geoIndex. DEFAULT: "geoIndex.txt"

overwrite	Logical: Overwrite existing fname / mname / gname files? If not, "_n" is added to the filenames, see berryFunctions::newFilename. DEFAULT: FALSE
checkwarn	Logical: warn about checkIndex issues? DEFAULT: TRUE
quiet	Logical: Suppress messages about progress and filenames? DEFAULT: FALSE
...	Further arguments passed to dataDWD for the meta part.

Value

invisible data.frame (or if meta=TRUE, list with two data.frames) with a number of columns inferred from the paths. Each is also written to disc.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct-Nov 2016, June 2017

See Also

indexFTP, updateIndexes index, selectDWD

Examples

```
## Not run:  # Not tested with R CMD check because of file writing
link <- "daily/kl/historical/tageswerte_00699_19490101_19580630_hist.zip"
ind <- createIndex(link, dir=tempdir())
ind
link2 <- "daily/kl/historical/KL_Tageswerte_Beschreibung_Stationen.txt"
link3 <- "daily/kl/recent/KL_Tageswerte_Beschreibung_Stationen.txt"
ind2 <- createIndex(c(link,link2,link3), dir=tempdir(), meta=TRUE)
lapply(ind2, head)

## End(Not run)
```

---

dataDWD *Download data from the DWD CDC FTP Server*

Description

Get climate data from the German Weather Service (DWD) FTP-server. The desired .zip (or .txt) dataset is downloaded into dir. If read=TRUE, it is also read, processed and returned as a data.frame. To solve "errors in download.file: cannot open URL", see https://bookdown.org/brry/rdwd/station-selection.html#fileindex.

Usage

```
dataDWD(file, base = dwdbase, joinbf = FALSE, dir = "DWDdata",
  force = FALSE, overwrite = FALSE, sleep = 0, quiet = FALSE,
  progbar = !quiet, browse = FALSE, read = TRUE, ntrunc = 2,
  dfargs = NULL, ...)
```

Arguments

file	Char (vector): complete file URL(s) (including base and filename.zip) as returned by selectDWD. Can be a vector with several filenames.
base	Single char: base URL that will be removed from output file names. DEFAULT: dwdbase
joinbf	Logical: paste base and file together? DEFAULT: FALSE (selectDWD returns complete URLs already)
dir	Char: Writeable directory name where to save the downloaded file. Created if not existent. DEFAULT: "DWDdata" at current getwd()
force	Logical (vector): always download, even if the file already exists in dir? Use NA to force re-downloading files older than 24 hours. Use a numerical value to force after that amount of hours. Note you might want to set overwrite=TRUE as well. If FALSE, the file is still read (or name returned). DEFAULT: FALSE
overwrite	Logical (vector): if force=TRUE, overwrite the existing file rather than append "_1"/"_2" etc to the filename? DEFAULT: FALSE
sleep	Number. If not 0, a random number of seconds between 0 and sleep is passed to Sys.sleep after each download to avoid getting kicked off the FTP-Server. DEFAULT: 0
quiet	Logical: suppress message about directory / filenames? DEFAULT: FALSE
progbar	Logical: present a progress bar with estimated remaining time? If missing and length(file)==1, progbar is internally set to FALSE. Only works if the R package pbapply is available. DEFAULT: TRUE (!quiet)
browse	Logical: open repository via browseURL and return URL folder path? If TRUE, no data is downloaded. If file has several values, only unique folders will be opened. DEFAULT: FALSE
read	Logical: read the file(s) with readDWD? If FALSE, only download is performed and the filename(s) returned. DEFAULT: TRUE
ntrunc	Single integer: number of filenames printed in messages before they get truncated with message "(and xx more)". DEFAULT: 2
dfargs	Named list of additional arguments passed to download.file
...	Further arguments passed to readDWD, like fread, varnames etc. Dots were passed to download.file prior to rdwd 0.11.7 (2019-02-25)

Value

Presuming downloading and processing were successful: if read=TRUE, a data.frame of the desired dataset (as returned by readDWD), otherwise the filename as saved on disc (may have "_n" appended in name, see newFilename).
If length(file)>1, the output is a list of data.frames / vector of filenames.
The output is always invisible.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jun-Oct 2016

See Also

selectDWD. readDWD, download.file.
https://bookdown.org/brry/rdwd
Helpful for plotting: berryFunctions::monthAxis, see also berryFunctions::climateGraph

Examples

```
## Not run:  ## requires internet connection
# find FTP files for a given station name and file path:
link <- selectDWD("Fuerstenzell", res="hourly", var="wind", per="recent")
# download file:
fname <- dataDWD(link, dir=tempdir(), read=FALSE) ; fname
# dir="DWDdata" is the default directory to store files
# unless force=TRUE, already obtained files will not be downloaded again

# read and plot file:
wind <- readDWD(fname, varnames=TRUE) ; head(wind)
metafiles <- readMeta(fname)          ; str(metafiles, max.level=1)
column_names <- readVars(fname)       ; head(column_names)

plot(wind$MESS_DATUM, wind$F, main="DWD hourly wind Fuerstenzell", col="blue",
     xaxt="n", las=1, type="l", xlab="Date", ylab="Hourly Wind speed  [m/s]")
berryFunctions::monthAxis(1)


# current and historical files:
link <- selectDWD("Potsdam", res="daily", var="kl", per="hr"); link
potsdam <- dataDWD(link, dir=tempdir())
potsdam <- do.call(rbind, potsdam) # this will partly overlap in time
plot(TMK~MESS_DATUM, data=tail(potsdam,1500), type="l")
# The straight line marks the jump back in time
# Keep only historical data in the overlap time period:
potsdam <- potsdam[!duplicated(potsdam$MESS_DATUM),]


# With many files (>>50), use sleep to avoid getting kicked off the FTP server
#links <- selectDWD(res="daily", var="solar")
#sol <- dataDWD(links, sleep=20) # random waiting time after download (0 to 20 secs)

# Real life examples can be found in the use cases section of the vignette:
# browseURL("https://bookdown.org/brry/rdwd")

## End(Not run)
```

DEU *Map of German states (Bundeslaender) from GADM through the* raster *package*

Description

Map of German states (Bundeslaender) from GADM through the `raster` package

Format

Formal class 'SpatialPolygons' [package "sp"] with 4 slots

Details

Obtained with the code:

```
DEU1 <-raster::getData("GADM",country="DEU",level=1)
DEU <-rgeos::gSimplify(DEU1,tol=0.02,topologyPreserve=FALSE)
raster::plot(DEU1)
raster::plot(DEU)
save(DEU,file="data/DEU.rda")
tools::resaveRdaFiles("data/DEU.rda")
```

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, May 2018

See Also

addBorders, EUR

---

dirDWD *directory management for rdwd*

Description

Manage directories with useful messages in the rdwd package.

Usage

```
dirDWD(dir = "DWDdata", quiet = FALSE)
```

Arguments

dir	Char for dirDWD: writeable directory name. Created if not existent. DEFAULT: "DWDdata" at current getwd()
quiet	Logical: Suppress messages about creating dir? DEFAULT: FALSE

Value

dirDWD invisibly returns the prior working directory as per setwd.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct 2016

See Also

dataDWD

Examples

```
# see source code of dataDWD and metaDWD
```

---

dwdbase *DWD FTP Server base URL*

Description

base URL to DWD FTP Server

dwdbase: observed climatic records at
ftp://opendata.dwd.de/climate_environment/CDC/observations_germany/climate

gridbase: spatially interpolated gridded data at
ftp://opendata.dwd.de/climate_environment/CDC/grids_germany

Usage

```
dwdbase
```

Format

An object of class character of length 1.

---

dwdparams *DWD parameter explanations*

Description

Short German parameter explanations for the DWD abbreviations on the CDC FTP server.
These are manually created by me and might need to be expanded if the DWD adds more abbreviations.
readVars maps them to the variable abbreviations in the "Metadaten_Parameter.*txt" file in any given zip folder and will warn about missing entries.

Usage

```
dwdparams
```

Format

An object of class `data.frame` with 160 rows and 2 columns.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jun 2018

See Also

`readVars`, `readDWD`

Examples

```
head(dwdparams)
```

EUR *Map of Western European countries through the* `rworldmap` *package*

Description

Map of Western European countries through the `rworldmap` package

Format

SpatialPolygonsDataFrame [package "sp"] with 32 rows

Details

Obtained with the code:

```
EUR <-rworldmap::getMap("low")
EUR <-raster::crop(EUR,c(-5,20,40,60))
raster::plot(EUR)
save(EUR,file="data/EUR.rda",version=2)
tools::resaveRdaFiles("data/EUR.rda",version=2)
```

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2019

See Also

`addBorders`, `DEU`

findID *find DWD weather station ID from name*

Description

Identify DWD weather station ID from station name

Usage

```
findID(name = "", exactmatch = TRUE, mindex = metaIndex,
  quiet = FALSE)
```

Arguments

name	Char: station name(s) that will be matched in mindex to obtain **id**. DEFAULT: ""
exactmatch	Logical: Should name match an entry in mindex exactly (be ==)? If FALSE, name may be a part of mindex$Stationsname, as checked with grepl. This is useful e.g. to get all stations starting with a name (e.g. 42 IDs for Berlin). DEFAULT: TRUE
mindex	Single object: Index used to select id if name is given. DEFAULT: rdwd:::metaIndex
quiet	Logical: suppress length warnings? DEFAULT: FALSE

Value

Character string (vector) with ID(s)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct-Nov 2016

See Also

used in selectDWD, metaInfo

Examples

```
# Give weather station name (must be existing in metaIndex):
findID("Potsdam")
findID("potsDam") # capitalization is ignored
# all names containing "Hamburg":
findID("Hamburg", exactmatch=FALSE)
findID("Potsdam", exactmatch=FALSE)

# vectorized:
findID(c("Potsdam","Berlin-Buch"))

# German Umlauts are changed to ue, ae, oe, ss
```

```
findID("Muenchen", FALSE)
berryFunctions::convertUmlaut("M?nchen") # use this to convert umlauts in lists
```

---

index *Indexes of files and metadata on the DWD CDC FTP server*

Description

Created with indexFTP and createIndex used in updateIndexes.
In functions, you can access them with rdwd:::fileIndex etc.
fileIndex: A data.frame with the filenames (and derived information) at the default base value dwdbase.
metaIndex: A data.frame with the contents of all the station description files (..._Beschreibung_Stationen.txt) under dwdbase.
geoIndex: metaIndex distilled to geographic locations.
gridIndex: Vector of file paths at gridbase.
formatIndex: (modified) table from ftp://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/subdaily/standard_format/formate_kl.html

Format

fileIndex: data.frame with character strings. ca 260k rows x 8 columns:
res, var, per (see selectDWD), station id, time series start and end, and ismeta information, all according to path.
metaIndex: data.frame with ca 97k rows for 12 columns:
Stations_id,von_datum,bis_datum,Stationshoehe,geoBreite,geoLaenge,Stationsname,Bundesland,res,var,p
geoIndex: data.frame with ca 6k rows for 11 columns:
id,name,state,lat,lon,ele,nfiles,nonpublic,recentfile,display,col
gridIndex: Vector with ca 50k file paths at gridbase
formatIndex: data.frame with 140 rows for 12 columns:
Ke_Ind,Kennung,Label,Beschreibung,Einheit,Code-Tabellen,Zusatzinfo,Typ,Pos,Erlaubt,Fehlk,dividebyte

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, June-Nov 2016, June 2017, Oct 2019

Source

Deutscher WetterDienst / Climate Data Center FTP Server

See Also

createIndex, indexFTP, selectDWD, findID, metaInfo, https://bookdown.org/brry/rdwd

Examples

```
data(fileIndex)
data(metaIndex)
data(geoIndex)
head(fileIndex)
head(metaIndex)
head(geoIndex)

# in functions, you can use head(rdwd:::fileIndex) etc, but I don't export them
# because Hadley says 'Never @export a data set' in
# browseURL("http://r-pkgs.had.co.nz/data.html#data-data")
```

`indexFTP` *Create a recursive index of an FTP Server*

Description

Create a list of all the files (in all subfolders) of an FTP server. Defaults to the German Weather Service (DWD, Deutscher WetterDienst) OpenData server at ftp://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/.

The R package `RCurl` must be available to do this. If `RCurl::getURL` fails, usually because bot access is detected and denied, there will still be an output which you can pass in a second run via `folder` to extract the remaining dirs. You might want to wait a bit and set `sleep` to a higher value in that case. Here's an example:

```
gridindex <-indexFTP("",gridbase)
gridindex <-indexFTP(gridindex,gridbase,sleep=1)
```

Usage

```
indexFTP(folder = "currentfindex", base = dwdbase,
  is.file.if.has.dot = TRUE, sleep = 0, dir = "DWDdata",
  filename = folder[1], overwrite = FALSE, quiet = FALSE,
  progbar = !quiet, verbose = FALSE)
```

Arguments

`folder`	Folder(s) to be indexed recursively, e.g. "/hourly/wind/". Leading slashes will be removed. Use `folder=""` to search at the location of `base` itself. If `folder` is "currentfindex" (the default) and `base` is the default, `folder` is changed to all observational folders listed in the current tree file at ftp://opendata.dwd.de/weather/tree.html. With "currentgindex" and `gridbase`, the grid folders in the tree are used. DEFAULT: "currentfindex"
`base`	Main directory of FTP server. Trailing slashes will be removed. DEFAULT: `dwdbase`

`is.file.if.has.dot`
: Logical: if some of the input paths contain a dot, treat those as files, i.e. do not try to read those as if they were a folder. Only set this to FALSE if you know what you're doing. DEFAULT: TRUE

`sleep`
: If not 0, a random number of seconds between 0 and `sleep` is passed to `Sys.sleep` after each read folder to avoid getting kicked off the FTP-Server. DEFAULT: 0

`dir`
: Writeable directory name where to save the downloaded file. Created if not existent. DEFAULT: "DWDdata" at current `getwd()`

`filename`
: Character: Part of output filename. "INDEX_of_DWD_" is prepended, "/" replaced with "_", ".txt" appended. DEFAULT: folder[1]

`overwrite`
: Logical: Overwrite existing file? If not, "_n" is added to the filename, see `berryFunctions::newFilename`. DEFAULT: FALSE

`quiet`
: Suppress progbars and message about directory/files? DEFAULT: FALSE

`progbar`
: Logical: present a progress bar in each level? DEFAULT: TRUE

`verbose`
: Logical: write a lot of messages from `RCurl::getURL`? DEFAULT: FALSE (usually, you dont need all the curl information)

Details

It's not suggested to run this for all folders, as it can take quite some time and you may get kicked off the FTP-Server. This package contains an index of the climatic observations at weather stations: `View(rdwd:::fileIndex)`. If it is out of date, please let me know!

Value

a vector with file paths

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct 2016

See Also

`createIndex`, `updateIndexes`

Examples

```
## Not run:  ## Needs internet connection
sol <- indexFTP(folder="/daily/solar", dir=tempdir())
head(sol)

# mon <- indexFTP(folder="/monthly/kl", dir=tempdir(), verbose=TRUE)

## End(Not run)
```

`lldist` *distance between lat-long coordinates*

Description

Great-circle distance between points at lat-long coordinates. Mostly a copy of OSMscale::earthDist Version 0.5.3 (2017-04-19). https://github.com/brry/OSMscale/blob/master/R/earthDist.R#L57-L102. Copied manually to avoid dependency hell. Does not check coordinates. Not exported.

Usage

```
lldist(lat, long, data, r = 6371, i = 1L)

maxlldist(lat, long, data, r = 6371, fun = max, each = TRUE, ...)
```

Arguments

`lat, long`	Latitude (North/South) and longitude (East/West) coordinates in decimal degrees
`data`	Optional: data.frame with the columns `lat` and `long`
`r`	radius of the earth. Could be given in miles. DEFAULT: 6371 (km)
`i`	Integer: Index element against which all coordinate pairs are computed. DEFAULT: 1
`fun`	Function to be applied. DEFAULT: `max`
`each`	Logical: give max dist to all other points for each point separately? If FALSE, will return the maximum of the complete distance matrix, as if `max(maxlldist(y,x))`. For examples, see OSMscale::maxEarthDist DEFAULT: TRUE
`...`	Further arguments passed to fun, like na.rm=TRUE

Value

Vector with distance(s) in km (or units of `r`, if `r` is changed)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2016 + Jan 2017. Angle formula from Diercke Weltatlas 1996, Page 245

localtestdir *local test data directory*

Description

returns a directory used for local tests on Berry's computers. This is used in many examples to save the downloaded DWD data in this directory, thus avoiding multiple downloads of the same file.

Usage

```
localtestdir(packdir = ".", folder = "misc/localdata", file = NULL)
```

Arguments

packdir	Path to package directory. DEFAULT: "."
folder	Path inside package. DEFAULT: "misc/localdata"
file	Optional: path(s) at folder. DEFAULT: NULL

Value

charstring (directory)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Apr 2019

See Also

runLocalTests

Examples

```
localtestdir()
```

metaInfo *Information for a station ID on the DWD CDC FTP server*

Description

Information for a station ID on the DWD CDC FTP server

Usage

```
metaInfo(id, hasfileonly = TRUE)
```

Arguments

id	Station ID (integer number or convertible to one)
hasfileonly	Logical: Only show entries that have files? DEFAULT: TRUE

Value

invisible data.frame. Also prints the output nicely formatted.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Nov 2016

See Also

metaIndex

Examples

```
metaInfo(2849)
```

---

nearbyStations *Find DWD stations close to given coordinates*

Description

Select DWD stations within a given radius around a set of coordinates

Usage

```
nearbyStations(lat, lon, radius, res = NA, var = NA, per = NA,
  mindate = NA, hasfileonly = TRUE,
  statname = "nearbyStations target location", quiet = FALSE, ...)
```

Arguments

lat	Coordinates y component [degrees N/S, range 47:55]
lon	Coordinates x component [degrees E/W, range 6:15]
radius	Maximum distance [km] within which stations will be selected
res, var, per	Restrictions for dataset type as documented in selectDWD. Each can be a vector of entries. DEFAULTS: NA (ignored)
mindate	Minimum dataset ending date (as per metadata). DEFAULT: NA
hasfileonly	Logical: only return entries for which there is an open-access file available? DEFAULT: TRUE
statname	Character: name for target location. DEFAULT: "nearbyStations target location"
quiet	Logical: suppress progress messages? DEFAULT: FALSE
...	Further arguments passed to selectDWD

Value

metaIndex subset with additional columns "dist" and "url"

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Mar 2017

See Also

selectDWD, metaIndex

Examples

```
m <- nearbyStations(49.211784, 9.812475, radius=30,
    res=c("daily","hourly"), var= c("precipitation","more_precip","kl") ,
    mindate=as.Date("2016-05-30"), statname="Braunsbach catchment center")
# View(m)

# for a continued example of this, see the vignette in chapter
# use case: plot all rainfall values around a given point
# browseURL("https://bookdown.org/brry/rdwd")
```

---

newColumnNames *Enhance readDWD column names*

Description

Add short German parameter descriptions to the DWD abbreviations. This uses dwdparams to create column names like "TT_TU.Lufttemperatur" and "RSK.Niederschlagshoehe." Column names not in the abbreviation list will be left untouched.

Usage

```
newColumnNames(dataframe, variables = dwdparams, separator = ".")
```

Arguments

dataframe	Dataframe as returned by readDWD.data
variables	Dataframe as returned by readVars for a single file. Rownames must be variable abbreviations. There must be a "Kurz" column. DEFAULT: dwdparams
separator	Separator between abbreviation and long name. DEFAULT: "."

Value

The dataframe with new column names

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Apr 2019

See Also

dwdparams, readVars, readDWD argument varnames

Examples

```
# mainly for internal usage
```

---

projectRasterDWD *project DWD raster data*

Description

Set projection and extent for DWD raster data. Optionally (and per default) also reprojects to latlon data.
The internal defaults are extracted from the Kompositformatbeschreibung at https://www.dwd.de/DE/leistungen/radolan/radolan.html, as provided 2019-04 by Antonia Hengst.
The nc extent was obtained by projecting Germanys bbox to EPSG 3034 (specified in the DWD documentation). Using that as a starting point, I then refined the extent to a visual match, see developmentNotes.R

WARNING: reprojection to latlon changes values slightly. For the tested RX product, this change is significant, see: https://github.com/brry/rdwd/blob/master/misc/ExampleTests/Radartests.pdf
In raster::plot, use zlim with the original range if needed.

Usage

```
projectRasterDWD(r, proj = "radolan", extent = "radolan",
  latlon = TRUE, quiet = FALSE)
```

Arguments

r	Raster object
proj	Desired projection. Use NULL to not set proj+extent but still consider latlon. Can be a raster::crs output, a projection character string (will be passed to crs), or a special charstring for internal defaults, namely: "radolan" (readDWD.binary + .asc + .radar), "seasonal" (.raster) or "nc" (.nc). DEFAULT: "radolan"
extent	Desired extent. Can be an extent object, a vector with 4 numbers, or "radolan" / "rw" / "seasonal" / "nc" with internal defaults. DEFAULT: "radolan"
latlon	Logical: reproject r to lat-lon crs? DEFAULT: TRUE
quiet	Logical: suppress progress messages? DEFAULT: FALSE

Value

Raster object with projection and extent, invisible

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, May 2019

See Also

raster::crs, raster::projection, raster::extent, raster::projectRaster, readDWD.binary, readDWD.raster, re

Examples

```
# To be used after readDWD.binary, readDWD.raster, readDWD.asc
```

---

rdwd *Handle Climate Data from DWD (German Weather Service)*

Description

- find, select, download + read data from the German weather service DWD
- vectorized, progress bars, no re-downloads
- index of files + meta data
- observational time series from 6k meteorological recording stations (2.5k active)
-> rain, temperature, wind, sunshine, pressure, cloudiness, humidity, snow, ...
- gridded raster data from radar + interpolation
- european data stock slowly growing
For an introduction to the package, see https://bookdown.org/brry/rdwd.

Searchability Terms

Weather Data Germany download with R, Climate Data Germany
Deutscher Wetterdienst R Daten download Klimastationen
DWD Daten mit R runterladen, Wetter und Klimadaten in R

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, June-Nov 2016, June 2017

See Also

USA data: countyweather, rnoaa
World data: Global Surface Summary of the Day
Durch data: https://github.com/bvhest/KNMIr
Canadian data: https://cran.r-project.org/package=weathercan

readDWD *Process data from the DWD CDC FTP Server*

Description

Read climate data that was downloaded with dataDWD. The data is unzipped and subsequently, the file is read, processed and returned as a data.frame / raster object.
New users are advised to set varnames=TRUE to obtain more informative column names.

readDWD will call internal (but documented) functions depending on the arguments multia,meta,stand,binary,raster,nc
to read observational data (overview): readDWD.data,readDWD.multia,readDWD.stand,readDWD.meta
to read gridded data (overview): readDWD.binary,readDWD.raster,readDWD.radar,readDWD.nc,readDWD.asc
Not all arguments to readDWD are used for all functions, e.g. fread is used only by .data, while dividebyten is used in .raster and .asc.

file can be a vector with several filenames. Most other arguments can also be a vector and will be recycled to the length of file.

Usage

```
readDWD(file, quiet = FALSE, progbar = !quiet, fread = FALSE,
  varnames = FALSE, var = "", format = NA, tz = "GMT",
  dividebyten = TRUE, multia = grepl("Standort.txt$", file),
  meta = grepl(".txt$", file), stand = grepl("standard_format", file),
  binary = grepl(".tar.gz$", file), raster = grepl(".asc.gz$", file),
  nc = grepl(".nc.gz$", file), radar = grepl(".gz$", file),
  asc = grepl(".tar$", file), ...)
```

Arguments

file	Char (vector): name(s) of the file(s) downloaded with dataDWD, e.g. "~/DWD-data/tageswerte_KL_02575_akt.zip" or "~/DWDdata/RR_Stundenwerte_Beschreibung_Stationen.txt"
quiet	Logical: suppress messages? DEFAULT: FALSE
progbar	Logical: present a progress bar with estimated remaining time? If missing and length(file)==1, progbar is internally set to FALSE. DEFAULT: !quiet
fread	Logical (vector): read fast? See readDWD.data. DEFAULT: FALSE (some users complain it doesn't work on their PC)
varnames	Logical (vector): Expand column names? See readDWD.data. DEFAULT: FALSE
var	var for readDWD.nc. DEFAULT: ""
format, tz	Format and time zone of time stamps, see readDWD.data
dividebyten	Logical (vector): Divide the values in raster files by ten? Used in readDWD.raster and readDWD.asc. DEFAULT: TRUE

multia	Logical (vector): is the file a multi_annual file? Overrides meta, so set to FALSE manually if readDWD.meta needs to be called on a (manually renamed) Beschreibung file ending with "Standort.txt". See readDWD.multia. DEFAULT: TRUE for each file ending in "Standort.txt"
meta	Logical (vector): is the file a meta file (Beschreibung.txt)? See readDWD.meta. DEFAULT: TRUE for each file ending in ".txt"
stand	Logical (vector): is the file a subdaily/standard_format file? See readDWD.stand. DEFAULT: TRUE fo files containing "standard_format" in the name.
binary	Logical (vector): does the file contain binary files? See readDWD.binary. DEFAULT: TRUE for each file ending in ".tar.gz"
raster	Logical (vector): does the file contain a raster file? See readDWD.raster. DEFAULT: TRUE for each file ending in ".asc.gz"
nc	Logical (vector): does the file contain a netcdf file? See readDWD.nc. DEFAULT: TRUE for each file ending in ".nc.gz"
radar	Logical (vector): does the file contain a single binary file? See readDWD.radar. DEFAULT: TRUE for each file ending in ".gz"
asc	Logical (vector): does the file contain asc files? See readDWD.asc. DEFAULT: TRUE for each file ending in ".tar"
...	Further arguments passed to the internal readDWD.* functions and from those to the underlying reading functions documented in each internal function.

Value

Invisible data.frame of the desired dataset, or a named list of data.frames if length(file) > 1. readDWD.binary, readDWD.raster and readDWD.asc return raster objects instead of data.frames.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jul-Oct 2016, Winter 2018/19

See Also

dataDWD, readVars, readMeta, selectDWD
https://bookdown.org/brry/rdwd

Examples

```
# see dataDWD
```

`readDWD.asc` *read dwd gridded radolan asc data*

Description

read grid-interpolated radolan asc data. Intended to be called via `readDWD`.
All layers (following `selection` if given) in all .tar.gz files are combined into a raster stack with `raster::stack`.
To project the data, use `projectRasterDWD`

Usage

```
readDWD.asc(file, exdir = NULL, dividebyten = TRUE, selection = NULL,
  progbar = TRUE, ...)
```

Arguments

`file`	Name of file on harddrive, like e.g. DWDdata/grids_germany/hourly/radolan/historical/asc/2018_RW-201809.tar. Must have been downloaded with `mode="wb"`!
`exdir`	Directory to unzip into. Unpacked files existing therein will not be untarred again, saving up to 15 secs per file. DEFAULT: NULL (subfolder of `tempdir()`)
`dividebyten`	Divide numerical values by 10? If dividebyten=FALSE and exdir left at NULL (tempdir), save the result on disc with `raster::writeRaster`. Accessing out-of-memory raster objects won't work if exdir is removed! -> Error in .local(.Object, ...) DEFAULT: TRUE
`selection`	Optionally read only a subset of the ~24*31=744 files. Called as `f[selection]`. DEFAULT: NULL (ignored)
`progbar`	Show messages and progress bars? `readDWD` will keep progbar=TRUE for asc files, even if length(file)==1. DEFAULT: TRUE
`...`	Further arguments passed to `raster::raster`

Value

data.frame

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, April 2019

See Also

`readDWD`

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

# File selection and download:
datadir <- localtestdir()
radbase <- paste0(gridbase,"/hourly/radolan/historical/asc/")
radfile <- "2018/RW-201809.tar" # 25 MB to download
file <- dataDWD(radfile, base=radbase, joinbf=TRUE, dir=datadir,
                dfargs=list(mode="wb"), read=FALSE) # download with mode=wb!!!

#asc <- readDWD(file) # 4 GB in mem. ~ 20 secs unzip, 30 secs read, 10 min divide
asc <- readDWD(file, selection=1:5, dividebyten=TRUE)
asc <- projectRasterDWD(asc)

raster::plot(asc[[1]], main=names(asc)[1])
addBorders()

rng <- range(raster::cellStats(asc, "range"))
nframes <- 3 # raster::nlayers(asc) for all (time intensive!)
viddir <- paste0(tempdir(),"/RadolanVideo")
dir.create(viddir)
png(paste0(viddir,"/Radolan_%03d.png"), width=7, height=5, units="in", res=300)
dummy <- pbsapply(1:nframes, function(i)
         raster::plot(asc[[i]], main=names(asc)[i], zlim=rng)) # 3 secs per layer
dev.off()
berryFunctions::openFile(paste0(viddir,"/Radolan_001.png"))

# Time series of a given point in space:
plot(as.vector(asc[800,800,]), type="l", xlab="Time [hours]")

# if dividebyten=FALSE, raster stores things out of memory in the exdir.
# by default, this is in tempdir, hence you would need to save asc manually:
# raster::writeRaster(asc, paste0(datadir,"/RW2018-09"), overwrite=TRUE)

## End(Not run)
```

`readDWD.binary` *read dwd gridded radolan binary data*

Description

read gridded radolan binary data. Intended to be called via `readDWD`.

Usage

```
readDWD.binary(file, exdir = sub(".tar.gz$", "", file),
  toraster = TRUE, progbar = TRUE, selection = NULL, ...)
```

Arguments

file	Name of file on harddrive, like e.g. DWDdata/daily_radolan_historical_bin_2017_SF201712.tar.gz
exdir	Directory to unzip into. If existing, only the needed files will be unpacked with untar. Note that exdir size will be around 1.1 GB. exdir can contain other files, these will be ignored for the actual reading with dwdradar::readRadarFile. DEFAULT exdir: sub(".tar.gz$", "", file)
toraster	Logical: convert output (list of matrixes + meta informations) to a list with data (raster stack) + meta (list from the first subfile, but with vector of dates)? DEFAULT: TRUE
progbar	Show messages and progress bars? readDWD will keep progbar=TRUE for binary files, even if length(file)==1. DEFAULT: TRUE
selection	Optionally read only a subset of the ~24*31=744 files. Called as f[selection]. DEFAULT: NULL (ignored)
...	Further arguments passed to dwdradar::readRadarFile, i.e. na and clutter

Value

list depending on argument toraster, see there for details

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Dec 2018. Significant input for the underlying dwdradar::readRadarFile came from Henning Rust & Christoph Ritschel at FU Berlin.

See Also

readDWD, especially readDWD.radar
https://wradlib.org for much more extensive radar analysis in Python
Kompositformatbeschreibung at https://www.dwd.de/DE/leistungen/radolan/radolan.html for format description

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

# SF file as example: ----

SF_link <- "/daily/radolan/historical/bin/2017/SF201712.tar.gz"
SF_file <- dataDWD(file=SF_link, base=gridbase, joinbf=TRUE,   # 204 MB
                   dir=localtestdir(), read=FALSE)
# exdir radardir set to speed up my tests:
SF_exdir <- "C:/Users/berry/Desktop/DWDbinarySF"
if(!file.exists(SF_exdir)) SF_exdir <- tempdir()
# no need to read all 24*31=744 files, so setting selection:
SF_rad <- readDWD(SF_file, selection=1:10, exdir=SF_exdir) #with toraster=TRUE
if(length(SF_rad)!=2) stop("length(SF_rad) should be 2, but is ", length(SF_rad))

SF_radp <- projectRasterDWD(SF_rad$data)
raster::plot(SF_radp[[1]], main=SF_rad$meta$date[1])
```

```
addBorders()


# RW file as example: ----

RW_link <- "hourly/radolan/reproc/2017_002/bin/2017/RW2017.002_201712.tar.gz"
RW_file <- dataDWD(file=RW_link, base=gridbase, joinbf=TRUE,   # 25 MB
                   dir=localtestdir(), read=FALSE)
RW_exdir <- "C:/Users/berry/Desktop/DWDbinaryRW"
if(!file.exists(RW_exdir)) RW_exdir <- tempdir()
RW_rad <- readDWD(RW_file, selection=1:10, exdir=RW_exdir)
RW_radp <- projectRasterDWD(RW_rad$data, extent="rw")
raster::plot(RW_radp[[1]], main=RW_rad$meta$date[1])
addBorders()

# ToDo: why are values + patterns not the same?

# list of all Files: ----
data(gridIndex)
head(grep("historical", gridIndex, value=TRUE))

## End(Not run)
```

`readDWD.data` *read regular dwd data*

Description

Read regular dwd data. Intended to be called via `readDWD`.

Usage

```
readDWD.data(file, fread = FALSE, varnames = FALSE, format = NA,
  tz = "GMT", quiet = FALSE, ...)
```

Arguments

`file`	Name of file on harddrive, like e.g. DWDdata/daily_kl_recent_tageswerte_KL_03987_akt.zip
`fread`	Logical: read faster with `data.table::fread`? When reading many large historical files, speedup is significant. NA can also be used, which means TRUE if data.table is available. DEFAULT: FALSE
`varnames`	Logical (vector): add a short description to the DWD variable abbreviations in the column names? E.g. change `FX,TNK` to `FX.Windspitze,TNK.Lufttemperatur_Min`, see `newColumnNames`. DEFAULT: FALSE (for backwards compatibility)
`format`	Char (vector): Format passed to `as.POSIXct` (see `strptime`) to convert the date/time column to POSIX time format. If NULL, no conversion is performed (date stays a factor). If NA, `readDWD` tries to find a suitable format based on the number of characters. DEFAULT: NA

tz	Char (vector): time zone for as.POSIXct. "" is the current time zone, and "GMT" is UTC (Universal Time, Coordinated). DEFAULT: "GMT"
quiet	Suppress empty file warnings? DEFAULT: FALSE
...	Further arguments passed to read.table or data.table::fread

Value

data.frame

Author(s)

Berry Boessenkool, <berry-b@gmx.de>

See Also

readDWD, Examples in dataDWD

---

readDWD.meta *read dwd metadata (Beschreibung*.txt files)*

Description

read dwd metadata (Beschreibung*.txt files). Intended to be called via readDWD.
Column widths for read.fwf are computed internally.
if(any(meta)), readDWD tries to set the locale to German (to handle Umlaute correctly). It is hence not recommended to call rdwd:::readDWD.meta directly on a file!
Names can later be changed to ascii with berryFunctions::convertUmlaut.

Usage

```
readDWD.meta(file, ...)
```

Arguments

file	Name of file on harddrive, like e.g. DWDdata/daily_kl_recent_KL_Tageswerte_Beschreibung_Stationen.
...	Further arguments passed to read.fwf

Value

data.frame

Author(s)

Berry Boessenkool, <berry-b@gmx.de>

See Also

readDWD

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

link <- selectDWD(res="daily", var="kl", per="r", meta=TRUE)
if(length(link)!=1) stop("length of link should be 1, but is ", length(link),
                ":\n", berryFunctions::truncMessage(link,prefix="",sep="\n"))

file <- dataDWD(link, dir=localtestdir(), read=FALSE)
meta <- readDWD(file)
head(meta)

cnm <- colnames(meta)
if(length(cnm)!=8) stop("number of columns should be 8, but is ", length(cnm),
                        ":\n", toString(cnm))

## End(Not run)
```

`readDWD.multia` *read multi_annual dwd data*

Description

read multi_annual dwd data. Intended to be called via `readDWD`.
All other observational data at `dwdbase` can be read with `readDWD.data`, except for the multi_annual and subdaily/standard_format data.

Usage

```
readDWD.multia(file, fileEncoding = "latin1", comment.char = "\032",
  ...)
```

Arguments

`file`	Name of file on harddrive, like e.g. DWDdata/multi_annual_mean_81-10_Temperatur_1981-2010_aktStandort.txt or DWDdata/multi_annual_mean_81-10_Temperatur_1981-2010_Stationsliste_aktStandort.txt
`fileEncoding`	read.table file encoding. DEFAULT: "latin1" (needed on Linux, optional but not hurting on windows)
`comment.char`	read.table comment character. DEFAULT: "\032" (needed 2019-04 to ignore the binary control character at the end of multi_annual files)
`...`	Further arguments passed to `read.table`

Value

data.frame

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Feb 2019

See Also

readDWD

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

# Temperature aggregates (2019-04 the 9th file):
durl <- selectDWD(res="multi_annual", var="mean_81-10", per="")[9]
murl <- selectDWD(res="multi_annual", var="mean_81-10", per="", meta=TRUE)[9]

ma_temp <- dataDWD(durl, dir=localtestdir())
ma_meta <- dataDWD(murl, dir=localtestdir())

head(ma_temp)
head(ma_meta)

ma <- merge(ma_meta, ma_temp, all=TRUE)
berryFunctions::linReg(ma$Stationshoehe, ma$Jahr)
op <- par(mfrow=c(3,4), mar=c(0.1,2,2,0), mgp=c(3,0.6,0))
for(m in colnames(ma)[8:19])
  {
  berryFunctions::linReg(ma$Stationshoehe, ma[,m], xaxt="n", xlab="", ylab="", main=m)
  abline(h=0)
  }
par(op)

par(bg=8)
berryFunctions::colPoints(ma$geogr..Laenge, ma$geogr..Breite, ma$Jahr, add=F, asp=1.4)

data("DEU")
pdf("MultiAnn.pdf", width=8, height=10)
par(bg=8)
for(m in colnames(ma)[8:19])
  {
  raster::plot(DEU, border="darkgrey")
 berryFunctions::colPoints(ma[-262,]$geogr..Laenge, ma[-262,]$geogr..Breite, ma[-262,m],
                           asp=1.4, # Range=range(ma[-262,8:19]),
                           col=berryFunctions::divPal(200, rev=TRUE), zlab=m, add=T)
  }
dev.off()
berryFunctions::openFile("MultiAnn.pdf")

## End(Not run)
```

readDWD.nc *read dwd netcdf data*

Description

Read netcdf data. Intended to be called via readDWD.
Note that R.utils and ncdf4 must be installed to unzip and read the .nc.gz files.

Usage

```
readDWD.nc(file, gargs = NULL, var = "", toraster = TRUE,
  quiet = FALSE, ...)
```

Arguments

file	Name of file on harddrive, like e.g. DWDdata/grids_germany/daily/Project_TRY/humidity/RH_199509_d
gargs	Named list of arguments passed to R.utils::gunzip, see readDWD.raster. DEFAULT: NULL
var	if toraster=FALSE: Charstring with name of variable to be read with ncdf4::ncvar_get. If not available, an interactive selection is presented. DEFAULT: "" (last variable)
toraster	Read file with raster::brick? All further arguments will be ignored. Specify e.g. var through .... DEFAULT: TRUE
quiet	Logical: Suppress time conversion failure warning? DEFAULT: FALSE
...	Further arguments passed to raster::brick or ncdf4::nc_open

Value

raster::brick object. Alternatively, if toraster=FALSE, a list with time, lat, lon, var, varname, file and cdf. **cdf** is the output of ncdf4::nc_open.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2019

See Also

readDWD

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

library(berryFunctions) # for seqPal and colPointsLegend

url <- "daily/Project_TRY/pressure/PRED_199606_daymean.nc.gz"  #  5 MB
url <- "daily/Project_TRY/humidity/RH_199509_daymean.nc.gz"    # 25 MB
```

```
file <- dataDWD(url, base=gridbase, joinbf=TRUE, dir=localtestdir(), read=FALSE)
nc <- readDWD(file)
ncp <- projectRasterDWD(nc, proj="nc", extent="nc")
raster::plot(ncp[[1]], col=seqPal(), main=paste(nc@title, nc@z$"Date/time"[1]))
addBorders()
str(nc, max.level=2)

rng <- range(raster::cellStats(nc[[1:6]], "range"))
raster::plot(nc, col=seqPal(), zlim=rng, maxnl=6)

# Array instead of raster brick:
nc <- readDWD(file, toraster=FALSE)
image(nc$var[,,1], col=seqPal(), asp=1.1)
colPointsLegend(nc$var[,,1], title=paste(nc$varname, nc$time[1]))

# interactive selection of variable:
# nc <- readDWD(file, var="-") # uncommented to not block automated tests
str(nc$var)

## End(Not run)
```

`readDWD.radar` *read dwd gridded radolan radar data*

Description

read gridded radolan radar data. Intended to be called via `readDWD`.

Usage

```
readDWD.radar(file, gargs = NULL, toraster = TRUE, ...)
```

Arguments

`file`	Name of file on harddrive, like e.g. DWDdata/hourly/radolan/recent/bin/ raa01-rw_10000-1802020250-dwd—bin.gz
`gargs`	Named list of arguments passed to `R.utils::gunzip`. The internal defaults are: `remove=FALSE` (recommended to keep this so `file` does not get deleted) and `skip=TRUE` (which reads previously unzipped files as is). If `file` has changed, you might want to use `gargs=list(skip=FALSE,overwrite=TRUE)` or alternatively `gargs=list(temporary=TRUE)`. The `gunzip` default `destname` means that the unzipped file is stored at the same path as `file`. DEFAULT gargs: NULL
`toraster`	Logical: convert output (list of matrixes + meta informations) to a list with data (`raster stack`) + meta (list from the first subfile, but with vector of dates)? DEFAULT: TRUE
`...`	Further arguments passed to `dwdradar::readRadarFile`, i.e. `na` and `clutter`

Value

Invisible list with `dat` (matrix or raster, depending on `toraster`) and `meta` (list with elements from header)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Aug 2019. Significant input for the underlying `dwdradar::readRadarFile` came from Henning Rust & Christoph Ritschel at FU Berlin.

See Also

`readDWD`, especially `readDWD.binary`
`https://wradlib.org` for much more extensive radar analysis in Python
Kompositformatbeschreibung at `https://www.dwd.de/DE/leistungen/radolan/radolan.html` for format description

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests
# recent radar files
rrf <- indexFTP("hourly/radolan/recent/bin", base=gridbase, dir=tempdir())
lrf <- dataDWD(rrf[773], base=gridbase, joinbf=TRUE, dir=tempdir(), read=FALSE)
r <- readDWD(lrf)

rp <- projectRasterDWD(r$dat)
raster::plot(rp, main=r$meta$date)
addBorders()

## End(Not run)
```

---

readDWD.raster *read dwd gridded raster data*

Description

Read gridded raster data. Intended to be called via `readDWD`.
Note that `R.utils` must be installed to unzip the .asc.gz files.

Usage

```
readDWD.raster(file, gargs = NULL, dividebyten, ...)
```

Arguments

`file`	Name of file on harddrive, like e.g. DWDdata/grids_germany/seasonal/air_temperature_mean/16_DJF_grids_germany_seasonal_air_temp_mean_188216.asc.gz

gargs	Named list of arguments passed to R.utils::gunzip. The internal defaults are: remove=FALSE (recommended to keep this so file does not get deleted) and skip=TRUE (which reads previously unzipped files as is). If file has changed, you might want to use gargs=list(skip=FALSE,overwrite=TRUE) or alternatively gargs=list(temporary=TRUE). The gunzip default destname means that the unzipped file is stored at the same path as file. DEFAULT gargs: NULL
dividebyten	Logical: Divide the numerical values by 10? DEFAULT: TRUE
...	Further arguments passed to raster::raster

Value

raster::raster object

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Dec 2018

See Also

readDWD

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

rasterbase <- paste0(gridbase,"/seasonal/air_temperature_mean")
ftp.files <- indexFTP("/16_DJF", base=rasterbase, dir=tempdir())
localfiles <- dataDWD(ftp.files[1:2], base=rasterbase, joinbf=TRUE,
                      dir=localtestdir(), read=FALSE)
rf <- readDWD(localfiles[1])
rf <- readDWD(localfiles[1]) # runs faster at second time due to skip=TRUE
raster::plot(rf)

rfp <- projectRasterDWD(rf, proj="seasonal", extent=rf@extent)
raster::plot(rfp)
addBorders()

testthat::expect_equal(raster::cellStats(rf, range), c(-8.2,4.4))
rf10 <- readDWD(localfiles[1], dividebyten=FALSE)
raster::plot(rf10)
testthat::expect_equal(raster::cellStats(rf10, range), c(-82,44))

## End(Not run)
```

readDWD.stand *read subdaily/standard_format dwd data*

Description

read subdaily/standard_format dwd data. Intended to be called via readDWD.
All other observational data at dwdbase can be read with readDWD.data, except for the multi_annual and subdaily/standard_format data.

Usage

```
readDWD.stand(file, fast = TRUE, fileEncoding = "latin1",
  formIndex = formatIndex, ...)
```

Arguments

file	Name of file on harddrive, like e.g. DWDdata/subdaily_standard_format_kl_10381_00_akt.txt or DWDdata/subdaily_standard_format_kl_10381_bis_1999.txt.gz
fast	Logical: use readr::read_fwf instead of read.fwf? Takes 0.1 instead of 20 seconds but requires package to be installed. if fast=TRUE, fileEncoding is ignored. DEFAULT: TRUE
fileEncoding	read.table file encoding. DEFAULT: "latin1" (potentially needed on Linux, optional but not hurting on windows)
formIndex	Single object: Index used to select column widts and NA values. To use a current / custom index, see the source code of updateIndexes at https://github.com/brry/rdwd/blob/master/R/updateIndexes.R. DEFAULT: rdwd:::formatIndex
...	Further arguments passed to read.fwf or readr::read_fwf

Value

data.frame with column names as per formatIndex. "Q"-columns have "_parameter" appended to their name. A "Date" column has been added. NA-indicators have been processed into NAs.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct 2019

See Also

readDWD

Examples

```
## Not run:  # Excluded from CRAN checks, but run in localtests

link <- selectDWD(id=10381, res="subdaily", var="standard_format", per="r")
file <- dataDWD(link, dir=localtestdir(), read=FALSE)
sf <- readDWD(file)

sf2 <- readDWD(file, fast=FALSE) # 20 secs!
stopifnot(all.equal(sf, sf2))

plot(sf$Date, sf$SHK, type="l")

# Plot all columns:
if(FALSE){ # not run in any automated testing
tmp <- tempfile(fileext=".pdf")
char2fact <- function(x)
 {
 if(all(is.na(x))) return(rep(-9, len=length(x)))
 if(!is.numeric(x)) as.factor(x) else x
 }
pdf(tmp, width=9)
par(mfrow=c(2,1),mar=c(2,3,2,0.1), mgp=c(3,0.7,0), las=1)
for(i in 3:ncol(sf)-1) plot(sf$Date, char2fact(sf[,i]), type="l", main=colnames(sf)[i], ylab="")
dev.off()
berryFunctions::openFile(tmp)
}

## End(Not run)
```

`readMeta` *Process data from the DWD CDC FTP Server*

Description

Read climate meta info textfiles in zip folders downloaded with `dataDWD`.

Usage

```
readMeta(file, progbar = TRUE, ...)
```

Arguments

`file`	Char (vector): name(s) of the zip file(s) downloaded with `dataDWD`, e.g. "~/DWD-data/tageswerte_KL_02575_akt.zip"
`progbar`	Logical: present a progress bar with estimated remaining time? If missing and length(file)==1, progbar is internally set to FALSE. DEFAULT: TRUE
`...`	Further arguments passed to `read.table`

Value

Invisible named list of data.frames; or a list of lists, if length(file)>1.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, 2016 + March 2019

See Also

dataDWD, readVars, readDWD

Examples

```
# see dataDWD
```

---

readVars *Process data from the DWD CDC FTP Server*

Description

Read climate variables (column meta data) from zip folders downloaded with dataDWD. The metadata file "Metadaten_Parameter.*txt" in the zip folder file is read, processed and returned as a data.frame.
file can be a vector with several filenames.

Usage

```
readVars(file, progbar = TRUE)
```

Arguments

file	Char (vector): name(s) of the file(s) downloaded with dataDWD, e.g. "~/DWDdata/tageswerte_KL_02575_akt.zip"
progbar	Logical: present a progress bar with estimated remaining time? If missing and length(file)==1, progbar is internally set to FALSE. DEFAULT: TRUE

Value

data.frame of the desired dataset, or a named list of data.frames if length(file) > 1.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jun 2018

See Also

dataDWD, readDWD, dwdparams

Examples

```
# see dataDWD
```

`rowDisplay` *Create leaflet map popup from data.frame rows*

Description

Create display character string for leaflet map popup from data.frame rows. This function is not exported, as it is only internally useful. A generic version is available in `berryFunctions::popleaf`.

Usage

```
rowDisplay(x)
```

Arguments

x	data.frame with colnames

Value

Vector of character strings, one for each row in x.

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Feb 2017

See Also

`geoIndex`

`runLocalTests` *run local tests of rdwd*

Description

Run `rdwd` tests on local machine. Due to time-intensive data downloads, these tests are not run automatically on CRAN.

Usage

```
runLocalTests(dir_data = localtestdir(),
  dir_exmpl = localtestdir(folder = "misc/ExampleTests"), fast = FALSE,
  radar = !fast, all_Potsdam_files = !fast, examples = !fast,
  quiet = FALSE)
```

Arguments

dir_data	Reusable data location. Preferably not under version control. DEFAULT: localtestdir()
dir_exmpl	Reusable example location. DEFAULT: localtestdir(folder="misc/ExampleTests")
fast	Exclude many tests? DEFAULT: FALSE
radar	Test reading radar example files. DEFAULT: !fast
all_Potsdam_files	Read all (ca 60) files for Potsdam? Re-downloads if files are older than 24 hours. Reduce test time a lot by setting this to FALSE. DEFAULT: !fast
examples	Run Examples (including donttest sections) DEFAULT: !fast
quiet	Suppress progress messages? DEFAULT: FALSE

Value

Time taken to run tests in minutes

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Apr-Oct 2019

See Also

localtestdir

---

selectDWD *Select data from the DWD CDC FTP Server*

Description

Select files for downloading with dataDWD.
The available folders with datasets are listed at https://bookdown.org/brry/rdwd/available-datasets.html. To use an updated index (if necessary), see https://bookdown.org/brry/rdwd/station-selection.html#fileindex.

All arguments (except for mindex, findex and base) can be a vector and will be recycled to the maximum length of all arguments. If that length > 1, the output is a list of filenames (or vector if outvec=TRUE).
If station name is given, but id is empty (""), **id** is inferred via mindex. If res/var/per are given and valid (existing in findex), they are pasted together to form a **path**. Here is an overview of the behavior in each case of availability:

case	**id**	**path**	output
1	""	""	base (and some warnings)
2	"xx"	""	All file names (across paths) for station **id**
3	""	"xx"	The zip file names at **path**
4	"xx"	"xx"	Regular single data file name

For case 2, you can explicitly set `res="",var="",per=""` to avoid the default interactive selection. For case 3 and 4 (**path** given), you can set `meta=TRUE`. Then selectDWD will return the name of the station description file at **path**. This is why case 3 with `meta=FALSE` only returns the data file names (ending in .zip).

Usage

```
selectDWD(name = "", res = NA, var = NA, per = NA,
  exactmatch = TRUE, mindex = metaIndex, quiet = FALSE,
  id = findID(name, exactmatch = exactmatch, mindex = mindex, quiet =
  quiet), base = dwdbase, findex = fileIndex, current = FALSE,
  meta = FALSE, meta_txt_only = TRUE, outvec = any(per %in% c("rh",
  "hr")), ...)
```

Arguments

`name`	Char: station name(s) passed to `findID`, along with `exactmatch` and `mindex`. All 3 arguments are ignored if `id` is given. DEFAULT: ""
`res`	Char: temporal **res**olution available at `base`, usually one of `c("hourly","daily","monthly")`, see section 'Description' above. `res/var/per` together form the **path**. DEFAULT: NA for interactive selection
`var`	Char: weather **var**iable of interest, like e.g. `"air_temperature","cloudiness","precipitation","s` See above and in `View(rdwd:::fileIndex)`. DEFAULT: NA for interactive selection
`per`	Char: desired time **per**iod. One of "recent" (data from the last year, up to date usually within a few days) or "historical" (long time series). Can be abbreviated (if the first letter is "r" or "h", full names are used). To get both datasets, use `per="hr"` or `per="rh"` (and `outvec=TRUE`). `per` is set to "" if var=="solar". DEFAULT: NA for interactive selection
`exactmatch`	Logical passed to `findID`: match `name` with `==`)? Else with `grepl`. DEFAULT: TRUE
`mindex`	Single object: Index with metadata passed to `findID`. DEFAULT: `rdwd:::metaIndex`
`quiet`	Suppress id length warnings?
`id`	Char/Number: station ID with or without leading zeros, e.g. "00614" or 614. Is internally converted to an integer, because some DWD meta data files also contain no leading zeros. DEFAULT: findID(name, exaxtmatch, mindex)
`base`	Single char: main directory of DWD ftp server. Must be the same `base` used to create `findex`. DEFAULT: `dwdbase`
`findex`	Single object: Index used to select filename, as returned by `createIndex`.To use a current / custom index, use `myIndex <-createIndex(indexFTP("/daily/solar"))` (with desired path, of course). DEFAULT: `rdwd:::fileIndex`
`current`	Single logical for case 3/4 with given `path`: instead of `findex`, use a list of the currently available files at base/res/var/per? This will call `indexFTP`, thus requires availability of the `RCurl` package. DEFAULT: FALSE

meta	Logical: return metadata txt file name instead of climate data zip file? Relevant only in case 4 (path and id given) and case 3 for res="multi_annual". See metaIndex for a compilation of all metaData files. DEFAULT: FALSE
meta_txt_only	Logical: if meta, only return .txt files, not the pdf and html files? DEFAULT: TRUE
outvec	Single logical: if **path** or **ID** length > 1, instead of a list, return a vector? (via unlist). DEFAULT: per %in% c("rh","hr")
...	Further arguments passed to indexFTP if current=TRUE, except folder and base.

Value

Character string with file path and name(s) in the format "base/res/var/per/filename.zip"

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Oct 2016

See Also

dataDWD, metaIndex, https://bookdown.org/brry/rdwd

Examples

```
# Give weather station name (must be existing in metaIndex):
selectDWD("Potsdam", res="daily", var="kl", per="historical")

# all files for all stations matching "Koeln":
selectDWD("Koeln", res="", var="", per="", exactmatch=FALSE)
findID("Koeln", FALSE)

## Not run:  # Excluded from CRAN checks to save time

# selectDWD("Potsdam") # interactive selection of res/var/per

# directly give station ID, can also be id="00386" :
selectDWD(id=386, res="daily", var="kl", per="historical")

# period can be abbreviated:
selectDWD(id="00386", res="daily", var="kl", per="h")
selectDWD(id="00386", res="daily", var="kl", per="h", meta=TRUE)

# vectorizable:
selectDWD(id="01050", res="daily", var="kl", per="rh") # list if outvec=F
selectDWD(id="01050", res=c("daily","monthly"), var="kl", per="r")
# vectorization gives not the outer product, but elementwise comparison:
selectDWD(id="01050", res=c("daily","monthly"), var="kl", per="hr")

# all zip files in all paths matching id:
selectDWD(id=c(1050, 386), res="",var="",per="")
# all zip files in a given path (if ID is empty):
head(  selectDWD(id="", res="daily", var="kl", per="recent")   )
```

```
## End(Not run)
```

Index