Package ‘rdwd’

April 8, 2021

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

Version 1.5.0

Date 2021-04-08

Depends R(>= 2.10)

Imports berryFunctions (>= 1.18.19), pbapply

Suggests RCurl, leaflet, knitr, markdown, testthat, roxygen2, devtools, remotes, bit64, data.table, OSMscale, raster, R.utils, ncdf4, readr, dwdradar, XML, sp, rgdal

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_node_en.html> for more information).
 Choose observational time series from meteorological stations with 'selectDWD().'.
 Find raster data from radar and interpolation according to <https://bookdown.org/brry/rdwd/raster-data.html>.
 Download (multiple) data sets with progress bars and no re-downloads through 'dataDWD().'.
 Read both tabular observational data and binary gridded datasets with 'readDWD().'.

License GPL (>= 2)

Encoding UTF-8

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

RoxygenNote 7.1.1

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

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2021-04-08 21:40:06 UTC

1
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>addBorders</td>
<td>3</td>
</tr>
<tr>
<td>checkIndex</td>
<td>4</td>
</tr>
<tr>
<td>checkSuggestedPackage</td>
<td>5</td>
</tr>
<tr>
<td>createIndex</td>
<td>5</td>
</tr>
<tr>
<td>dataDWD</td>
<td>7</td>
</tr>
<tr>
<td>DEU</td>
<td>10</td>
</tr>
<tr>
<td>dirDWD</td>
<td>11</td>
</tr>
<tr>
<td>dwdbase</td>
<td>12</td>
</tr>
<tr>
<td>dwdparams</td>
<td>12</td>
</tr>
<tr>
<td>EUR</td>
<td>13</td>
</tr>
<tr>
<td>fileType</td>
<td>14</td>
</tr>
<tr>
<td>findID</td>
<td>15</td>
</tr>
<tr>
<td>index</td>
<td>16</td>
</tr>
<tr>
<td>indexFTP</td>
<td>17</td>
</tr>
<tr>
<td>lldist</td>
<td>20</td>
</tr>
<tr>
<td>localtestdir</td>
<td>21</td>
</tr>
<tr>
<td>metaInfo</td>
<td>21</td>
</tr>
<tr>
<td>nearbyStations</td>
<td>22</td>
</tr>
<tr>
<td>newColumnNames</td>
<td>24</td>
</tr>
<tr>
<td>plotRadar</td>
<td>25</td>
</tr>
<tr>
<td>projectRasterDWD</td>
<td>27</td>
</tr>
<tr>
<td>rdwd</td>
<td>29</td>
</tr>
<tr>
<td>rdwdquiet</td>
<td>30</td>
</tr>
<tr>
<td>readDWD</td>
<td>30</td>
</tr>
<tr>
<td>readDWD.asc</td>
<td>32</td>
</tr>
<tr>
<td>readDWD.binary</td>
<td>33</td>
</tr>
<tr>
<td>readDWD.data</td>
<td>35</td>
</tr>
<tr>
<td>readDWD.grib2</td>
<td>36</td>
</tr>
<tr>
<td>readDWD.meta</td>
<td>38</td>
</tr>
<tr>
<td>readDWD.multia</td>
<td>39</td>
</tr>
<tr>
<td>readDWD.nc</td>
<td>40</td>
</tr>
<tr>
<td>readDWD.radar</td>
<td>42</td>
</tr>
<tr>
<td>readDWD.raster</td>
<td>43</td>
</tr>
<tr>
<td>readDWD.stand</td>
<td>45</td>
</tr>
<tr>
<td>readMeta</td>
<td>46</td>
</tr>
<tr>
<td>readVars</td>
<td>47</td>
</tr>
<tr>
<td>rowDisplay</td>
<td>48</td>
</tr>
<tr>
<td>runLocalTests</td>
<td>49</td>
</tr>
<tr>
<td>selectDWD</td>
<td>50</td>
</tr>
<tr>
<td>updateRdwd</td>
<td>53</td>
</tr>
</tbody>
</table>

**Index**  

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
</tr>
</tbody>
</table>
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 Bundesland borders (DEU). NA to suppress. DEFAULT: "grey80"
eu  Color for countries line (EUR). 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

plotRadar, DEU, EUR, website raster chapter

Examples

if(requireNamespace("raster", quietly=TRUE)){
  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)
}


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

```r
checkIndex(
  findex = NULL,
  mindex = NULL,
  gindex = NULL,
  excludefp = TRUE,
  fast = FALSE,
  warn = !quiet,
  logfile = localtestdir(".", "misc/ExampleTests/warnings.txt"),
  quiet = rdwdquiet()
)
```

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: !quiet (TRUE)
- **logfile**: File to copy log to, appended to existing content. NULL to suppress. DEFAULT: "misc/ExampleTests/warnings.txt"
- **quiet**: Logical: Suppress progress messages? DEFAULT: FALSE through `rdwdquiet()`

Value

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

Author(s)

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

See Also

- `createIndex`
checkSuggestedPackage

Examples

```r
data(fileIndex); data(metaIndex); data(geoIndex)
# ci <- rwd:::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

```r
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.
createIndex

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

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
checklog     Logfile for `checkIndex()`. DEFAULT: `tempfile()`
quiet        Logical: Suppress messages about progress and filenames? DEFAULT: FALSE through `rdwdquiet()`
...          Further arguments passed to `dataDWD()` for the meta part.
dataDWD

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(), website index chapter

Examples

```r
## 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, checkwarn=FALSE)
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 dataset is downloaded into `dir`. If `read=TRUE`, it is also read and processed.

dataDWD handles vectors of URLs, displays progress bars (if the package `pbapply` is available) and by default does not re-download data already in `dir` (but see argument `force` to update files).

To solve "errors in download.file: cannot open URL", see [https://bookdown.org/brry/rdwd/fileindex.html](https://bookdown.org/brry/rdwd/fileindex.html).

Usage

dataDWD(
  url,
  base = dwdbase,
  joinbf = FALSE,
  dir = "DWDdata",
  force = FALSE,
  overwrite = FALSE,
)


```r
read = TRUE,
dbin = TRUE,
dfargs = NULL,
sleep = 0,
progbar = !quiet,
browse = FALSE,
ntrunc = 2,
file = NULL,
quiet = rdwdquiet(),
```

Arguments

- **url**: Char (vector): complete file URL(s) (including base and filename.zip) as returned by `selectDWD()`. Can be a vector with several FTP URLs.
- **base**: Single char: base URL that will be removed from output file names. DEFAULT: `dwdbase`
- **joinbf**: Logical: paste base and file url together? Needed mostly for data at `gridbase`. 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: if `force!=FALSE`, you might want to set `overwrite=TRUE` as well. If `force=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
- **read**: Logical: read the file(s) with `readDWD()`? If FALSE, only download is performed and the filename(s) returned. DEFAULT: TRUE
- **dbin**: Logical: Download binary file, i.e. add mode="wb" to the `download.file()` call? See Website for details. DEFAULT: TRUE
- **dfargs**: Named list of additional arguments passed to `download.file()` Note that mode="wb" is already passed if `dbin=TRUE`
- **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, see note in `indexFTP()`. DEFAULT: 0
- **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
- **ntrunc**: Single integer: number of filenames printed in messages before they get truncated with message "(and xx more)". DEFAULT: 2
Deprecated since rdwd version 1.3.34, 2020-07-28.

Logical: suppress message about directory / filenames? DEFAULT: FALSE through rdwdquiet()

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, 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 outputs / 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$MESS_DATUM, data=tail(potsdam,1500), type="l")
# The straight line marks the jump back in time
# Keep only historical data in the overlap time period:
DEU <- potsdam[potsdam$MESS_DATUM != 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

Use directly with:
load(system.file("extdata/DEU.rda",package="rdwd"))

Obtained with the code:

url <- "https://gisco-services.ec.europa.eu/distribution/v2/nuts/shp/NUTS_RG_03M_2021_4326_LEVL_1.shp.zip"
tf <- tempfile(fileext=".zip")
download.file(url, tf) # 0.9 MB
unzip(tf, exdir="misc/vign") ; rm(url, tf)

DEU <- raster::shapefile("misc/vign/NUTS_RG_03M_2021_4326_LEVL_1.shp")
DEU <- DEU[DEU$CNTR_CODE == "DE","NUTS_NAME"]
raster::plot(DEU) ; axis(1, line=-1) ; axis(2, line=-1)

save(DEU, file="inst/extdata/DEU.rda", version=2)
tools::resaveRdaFiles("inst/extdata/DEU.rda", version=2)

Author(s)

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

**See Also**

- addBorders
- EUR

---

**dirDWD**

*directory management for rdwd*

**Description**

Manage directories with useful messages in the rdwd package.

**Usage**

```r
dirDWD(dir = "DWDdata", quiet = rdwdquiet())
```

**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 through `rdwdquiet()`

**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**

```r
# see source code of dataDWD and metaDWD
```
**dwdparams**

---

**dwdbase**

*DWD FTP Server base URL*

**Description**

Base URLs to the DWD FTP Server

- **dwdbase**: observed climatic records at
  
  `ftp://opendata.dwd.de/climate_environment/CDC/observations_germany/climate`
  
  See overview of available datasets and usage suggestions.

- **gridbase**: spatially interpolated gridded data at
  
  `ftp://opendata.dwd.de/climate_environment/CDC/grids_germany`
  
  See usage suggestions

**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. The `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 174 rows and 2 columns.

**Author(s)**

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

Map of Western European countries through the rworldmap package

Format

SpatialPolygonsDataFrame (package "sp") with 32 rows

Details

Use directly with:
load(system.file("extdata/EUR.rda",package="rdwd"))

Obtained with the code:

```r
EUR <- rworldmap::getMap("low")
EUR <- raster::crop(EUR, c(-11,25, 40,60))
raster::crs(EUR) <- raster::crs(DEU)
raster::plot(EUR)
save(EUR, file="inst/extdata/EUR.rda", version=2)
tools::resaveRdaFiles("inst/extdata/EUR.rda", version=2)
```

Author(s)

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

See Also

addBorders, DEU
fileType

determine DWD file type

Description

determine which subfunction to call in `readDWD()` from the file extension (ext).

The first block is for **observational data** (overview), the second for **gridded data** (overview).
Click on the type for the subfunction documentation, e.g. `data` for `readDWD.data()`.

<table>
<thead>
<tr>
<th>type</th>
<th>ext</th>
<th>notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>.zip</td>
<td>For regular data at <code>dwdbase</code>.</td>
</tr>
<tr>
<td>meta</td>
<td>.txt</td>
<td>For Beschreibung.txt files. For zip files containing station meta information, see <code>readMeta()</code>.</td>
</tr>
<tr>
<td>multia</td>
<td>[SO]</td>
<td>[SO]: file ends with &quot;Standort.txt&quot;. Overrides <code>meta</code>.</td>
</tr>
<tr>
<td>stand</td>
<td>[SF]</td>
<td>[SF]: file contains &quot;standard_format&quot;. For subdaily/standard_format files.</td>
</tr>
<tr>
<td>radar</td>
<td>.gz</td>
<td>For when the file contains a single binary file.</td>
</tr>
<tr>
<td>binary</td>
<td>.tar.gz</td>
<td>The common radolan format, as far as I can tell.</td>
</tr>
<tr>
<td>raster</td>
<td>.asc.gz</td>
<td>E.g. for seasonal data at <code>gridbase</code>.</td>
</tr>
<tr>
<td>nc</td>
<td>.nc.gz</td>
<td>For netcdf files.</td>
</tr>
<tr>
<td>asc</td>
<td>.tar</td>
<td>For a file containing asc files.</td>
</tr>
<tr>
<td>grib2</td>
<td>.grib2.bz2</td>
<td>For an nwp forecast file.</td>
</tr>
</tbody>
</table>

Usage

```
fileType(file)
```

Arguments

- `file` Filename(s) with extension.

Value

Character (vector)
findID

Description

Identify DWD weather station ID from station name

Usage

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

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: metaIndex
quiet Logical: suppress length warnings? DEFAULT: FALSE through rdwdquiet()
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

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
indexFTP

Create a recursive index of an FTP Server

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, per, hasfile
- **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, dividbyte

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()`, website index chapter

Examples

```r
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")
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.

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 (fileIndex) and gridded datasets (gridIndex). If they are out of date, please let me know!

Getting banned from the FTP Server

Normally, this shouldn’t happen anymore: since Version 0.10.10 (2018-11-26), a single RCurl handle is used for all FTP requests and since version 1.0.17 (2019-05-14), the file tree provided by the DWD is used to obtain all folders first, eliminating the recursive calls.

There’s a provision if the FTP server detects bot requests and denies access. If RCurl::getURL() fails, there will still be an output which you can pass in a second run via folder to extract the remaining dirs. You might need to wait a bit and set sleep to a higher value in that case. Here’s an example:

```r
gridindex <- indexFTP("", gridbase)
gridindex <- indexFTP(gridindex, gridbase, sleep=15)
```

Of course, with a higher sleep value, the execution will take longer!

Usage

```r
indexFTP(
    folder = "currentfindex",
    base = dwdbase,
    is.file.if.has.dot = TRUE,
    exclude.latest.bin = TRUE,
    fast = TRUE,
    sleep = 0,
    dir = "DWDdata",
    filename = folder[1],
    overwrite = FALSE,
    quiet = rdwdquiet(),
    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/
indexFTP

weather/tree.html. With "currentgindex" and gridbase, the grid folders in the tree are used. DEFAULT: "currentgindex"

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

exclude.latest.bin
Exclude latest file at opendata.dwd.de/weather/radar/radolan? RCurl::getURL indicates this is a pointer to the last regularly named file. DEFAULT: TRUE

fast
Read tree file with data.table::fread() (1 sec) instead of readLines() (10 secs)? 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, see note above. 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 through rdwdquiet()

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)

Value
a vector with file paths

Author(s)
Berry Boessenkool, <berry-b@gmx.de>, Oct 2016

See Also
createIndex(), updateIndexes(), website index chapter

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

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

# End(Not run)

lldist

## Example

### 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))`. 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)

**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**

```r
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**

```r
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**

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

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)

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 = rdwdquiet(),
  ...
)
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. DEFAULT: 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 through rdwdquiet()
...

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, website use case with nearbyStations

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, newColumnNames()

Examples

# mainly for internal usage
plotRadar

plotRadar products on a pretty map

Description

Convenience function to plot radar products on a pretty map. Creates a separate plot for each layer, a selection is possible.

Usage

plotRadar(
  x,
  layer = NULL,
  main = x@title,
  land = "gray80",
  sea = "cadetblue1",
  de = "grey80",
  eu = "black",
  col = berryFunctions::seqPal(),
  xlim = NULL,
  ylim = NULL,
  zlim = NULL,
  axes = TRUE,
  las = 1,
  mar = c(2.5, 3.5, 2.5, 5),
  keeppar = TRUE,
  project = TRUE,
  proj = "radolan",
  extent = "radolan",
  adjust05 = FALSE,
  targetproj = "ll",
  quiet = rdwdquiet(),
  ...
)

Arguments

x          raster object, e.g. 'dat' element of object returned by readDWD().
layer      Optional: selected layer(s) to be plotted. DEFAULT: NULL
main       Graph title(s). Use "" to suppress. Note output@title is set to main! DEFAULT: x@title
land       Color of land areas in the map. DEFAULT: "gray80"
sea        Color of sea areas in the map. DEFAULT: "cadetblue1"
de         Color of Deutschland Bundesland borders (DEU). DEFAULT: "grey80"
eu         Color of Europe country borders (EUR). DEFAULT: "black"
plotRadar

col
Color palette for the data itself. DEFAULT: berryFunctions::seqPal()

xlim
xlim. DEFAULT: NULL, i.e. taken from x extent (after reprojection if project=TRUE)

ylim
ylim. DEFAULT: NULL, i.e. taken from y extent (after reprojection if project=TRUE)

zlim
zlim. 3 Options: two-number vector, zlim="ind" for individual zlim per layer, or NULL for range of selected layer(s). DEFAULT: NULL

axes
Draw axes? DEFAULT: TRUE

las
LabelAxisStyle for axes. DEFAULT: 1 (all upright)

mar
Vector with plot margins. DEFAULT: c(2.5, 3.5, 2.5, 5)

keeppar
Logical: keep the margins set with par, so later points etc are added in the right location? DEFAULT: TRUE, opposite to sf::plot with reset=TRUE, see https://github.com/cran/sf/blob/master/R/plot.R

project
Project the data before plotting? Not needed if projectRasterDWD() has already been called. DEFAULT: TRUE

proj
current projection, see projectRasterDWD(), used only if project=TRUE. DEFAULT: "radolan"

extent
current extent, see projectRasterDWD(), used only if project=TRUE. DEFAULT: "radolan"

adjust05
Logical: Adjust extent by 0.5m to match edges? DEFAULT: FALSE

targetproj
target projection, see projectRasterDWD(), used only if project=TRUE. DEFAULT: "ll"

quiet
suppress progress messages? DEFAULT: FALSE through rdwdquiet()

...
Further arguments passed to raster::plot()

Value
raster object, projected (if project=TRUE). If length(layer)==1, only that selected layer is returned. output@title is set to main.

Author(s)
Berry Boessenkool, <berry-b@gmx.de>, Feb 2020

See Also
projectRasterDWD(), addBorders(), readDWD(), website raster chapter

Examples
# See homepage in the section 'See Also'
## Not run: ## Excluded from CRAN checks: requires internet connection
link <- "seasonal/air_temperature_mean/16_DJF/grids_germany_seasonal_air_temp_mean_188216.asc.gz"
rad <- dataDWD(link, base=gridbase, joinbf=TRUE, dir=tempdir())
radp <- plotRadar(rad, proj="seasonal", extent=rad$extent, main="plotRadar ex")
plotRadar(radp, ylim=c(52,54), project=FALSE) # reuses main

# plotRadar equivalent, map only country borders:
```r
radpm <- projectRasterDWD(rad[[1]], proj="seasonal", extent=rad@extent)
raster::plot(radpm)
addBorders()

# several layers
url <- "daily/Project_TRY/pressure/PRED_199606_daymean.nc.gz"  # 5 MB
nc <- dataDWD(url, base=gridbase, joinbf=TRUE, dir=localtestdir())
ncp3 <- plotRadar(nc, main=paste(nc@title, nc@z[[1]]), layer=1:3,
col=terrain.colors(100), proj="nc", extent="nc")
plotRadar(ncp3, layer=3:4, project=FALSE) # still has all layers
plotRadar(ncp3, layer=4:5, project=FALSE, zlim="ind") # individual zlims per layer
plotRadar(ncp3, layer=1, project=FALSE, zlim=c(1016,1020))

ncp1 <- plotRadar(nc, layer=1, proj="nc", extent="nc") # much faster projection
# no longer has layers 2-4:
berryFunctions::is.error(plotRadar(ncp1, layer=1:4, project=FALSE), TRUE, TRUE)

## End(Not run)
```

---

**projectRasterDWD**  
*project DWD raster data*

## Description

Set projection and extent for DWD raster data. Optionally (and per default) also reprojects to latlon data.

**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](https://github.com/brry/rdwd/blob/master/misc/ExampleTests/Radartests.pdf)

In `raster::plot`, use `zlim` with the original range if needed.

## Usage

```r
projectRasterDWD(
  r,
  proj = "radolan",
  extent = "radolan",
  adjust05 = FALSE,
  targetproj = "ll",
  quiet = rdwdquiet()
)
```

## Arguments

- `r`  
  Raster object
projectRasterDWD

proj Current projection to be given to r. Can be
- a `raster::crs()` input (e.g. a projection character string),
- NULL to not set proj+extent (but still consider targetproj),
- or a special charstring for internal defaults, namely: "radolan" (readDWD.binary + .asc + .radar), "seasonal" (.raster) or "nc" (.nc).
DEFAULT: "radolan"

extent Current `raster::extent()` to be given to r. Ignored if proj=NULL. Can be an extent object, a vector with 4 numbers, or "radolan" / "rw" / "seasonal" / "nc" with internal defaults. DEFAULT: "radolan"

adjust05 Logical: Adjust extent by 0.5m to match edges? DEFAULT: FALSE

targetproj r is reprojected to this `raster::crs()`. Use NULL to not reproject (i.e. only set proj and extent). DEFAULT: "II" with internal default for lat-lon.

quiet Logical: suppress progress messages? DEFAULT: FALSE through `rdwdquiet()`

Details

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

Value

Raster object with projection and extent, invisible

Author(s)

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

See Also

plotRadar()
`raster::crs/extent/projectRaster`
readDWD.binary / raster / asc / radar / nc
website raster chapter

Examples

# To be used after readDWD.binary etc
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>

See Also

- USA data: countyweather, rnoaa
- World data: Global Surface Summary of the Day
- Dutch data (Netherlands): https://github.com/bvhest/KNMIr
- Canadian data: https://cran.r-project.org/package=weathercan
- UK data website https://www.metoffice.gov.uk/climate/uk/data
**Description**

Global quiet option. The default `rdwdquiet()` is `FALSE`. Just write the following in your code and all subsequent calls will be quiet:

```r
options(rdwdquiet=TRUE)
```

**Usage**

```r
rdwdquiet()
```

**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(s) are read, processed and returned as a data.frame / raster object.

For observational data, new users are advised to set `varnames=TRUE` to obtain more informative column names.

`readDWD` will call internal (but documented) subfunctions depending on the argument `type`, see the overview in `fileType()`.

Not all arguments to `readDWD` are used for all subfunctions, e.g. `fread` is used only by `readDWD.data`, while `dividebyten` is used in `readDWD.raster` and `readDWD.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**

```r
readDWD(
  file,
  type = fileType(file),
  varnames = FALSE,
  fread = NA,
  format = NA,
  tz = "GMT",
  dividebyten = TRUE,
  var = "",
  progbars = !quiet,
```
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"

- **type**
  
  Character (vector) determining which subfunction to call. DEFAULT: `fileType(file)`.

- **varnames**
  
  Logical (vector): Expand column names? Only used in `readDWD.data()`. DEFAULT: FALSE (for backward compatibility)

- **fread**
  
  Logical (vector): read fast? Used in `readDWD.data()`. DEFAULT: NA

- **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

- **var**
  
  var for `readDWD.nc()`. DEFAULT: ""

- **probar**
  
  Logical: present a progress bar with estimated remaining time? If missing and length(file)==1, probar is internally set to FALSE. DEFAULT: !quiet

- **quiet**
  
  Logical: suppress messages? DEFAULT: FALSE through `rdwdquiet()`

...Further arguments passed to the internal readDWD.* subfunctions (see `fileType`) and from those to the underlying actual reading functions

Value

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

For gridded data, raster objects.

Author(s)

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

See Also

- `dataDWD()`, `readVars()`, `readMeta()`, `selectDWD()`, `fileType()`
- [https://bookdown.org/brry/rdwd](https://bookdown.org/brry/rdwd)

Examples

# see dataDWD and readDWD.* subfunctions
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,
  quiet = rdwdquiet(),
  progbar = !quiet,
  ...
)

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)
quiet  Suppress progress messages? DEFAULT: FALSE through rdwdquiet()
progbar  Show progress bars? readDWD() will keep progbar=TRUE for asc files, even if length(file)==1. DEFAULT: !quiet, i.e. TRUE
...

Value

data.frame

Author(s)

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

read dwd gridded radolan binary data

Description

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

Usage

```r
readDWD.binary(
  file,
  exdir = sub(".tar.gz$", ",", file),
  toraster = TRUE,
  quiet = rdwdquiet(),
)```

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,
    dbin=TRUE, 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)
plotRadar(asc[[1]], main=names(asc)[1])

viddir <- paste0(tempdir(),"/RadolanVideo")
dir.create(viddir)
png(paste0(viddir,"/Radolan_001.png"), width=7, height=5, units="in", res=300)
plotRadar(asc, layer=1:3, main=names(asc)) # 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)
```
progbar = !quiet,
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)

to raster
Logical: convert output (list of matrixes + meta informations) to a list with dataraster::stack + meta (list from the first subfile, but with vector of dates)? DEFAULT: TRUE

quiet
Suppress progress messages? DEFAULT: FALSE through rdwdquiet()

progbar
Show progress bars? readDWD() will keep progbar=TRUE for binary files, even if length(file)==1. DEFAULT: !quiet, i.e. 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 to raster, 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 local tests

# SF file as example: ----

SF_link <- "~/daily/radolan/historical/bin/2017/SF201712.tar.gz"
SF_file <- dataDWD(url=SF_link, base=gridbase, joinbf=TRUE, # 204 MB
    dir=localtestdir(), read=FALSE)
# exdir radardir set to speed up my tests:
readDWD.data <- "C:/Users/berry/Desktop/DWBbinarySF"
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 <- plotRadar(SF_rad$dat, layer=1:3, main=SF_rad$meta$date)
plotRadar(SF_radp, layer=1, project=FALSE)

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

# 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**

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

**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. When called from `readDWD()`, `fread=NA` can also be used, which means TRUE if R package `data.table` and system command `unzip` are available. Hint for Windows users: `unzip` comes with Rtools. See [https://bookdown.org/brry/rdwd/fread.html](https://bookdown.org/brry/rdwd/fread.html)  
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 through `rdwdquiet()`

**...**  
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.grib2**  
**read nwp forecast data**

### Description

read gridded numerical weather prediction data. Intended to be called via `readDWD()`.

### Usage

`readDWD.grib2(file, bargs = NULL, toraster = TRUE, quiet = rdwdquiet(), ...)`
Arguments

file  Name of file on harddrive, like e.g. cosmo-d2_germany_regular-lat-lon_single-level_2021010100_005_T_2M.grib2.bz2

bargs  Named list of arguments passed to `R.utils::bunzip2()`, see `gargs` in `readDWD.raster()`. DEFAULT: NULL

toraster  Logical: convert `rgdal::readGDAL` output with `raster::raster()`? DEFAULT: TRUE

quiet  Silence `readGDAL` completely, including warnings on discarded ellips / datum. DEFAULT: FALSE through `rdwdquiet()`

...  Further arguments passed to `rgdal::readGDAL()`.

Value

rgdal or raster object, depending on `toraster`

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jan 2021.

See Also

`readDWD()`

https://www.dwd.de/EN/ourservices/nwp_forecast_data/nwp_forecast_data.html

https://www.dwd.de/EN/aboutus/it/functions/Teasergroup/grib.html

Examples

```r
## Not run: # Excluded from CRAN checks, but run in localtests
# Deactivated 2021-04-08 since readDWD.grib2 -> rgdal::readGDAL -> Error:
# **.grib2 is a grib file, but no raster dataset was successfully identified.
warning("readDWD.grib2 is not tested due to unresolved problems.")
if(FALSE){
  nwp_t2m_base <- "ftp://opendata.dwd.de/weather/nwp/icon-d2/grib/03/p"
  nwp_urls <- indexFTP("", base=nwp_t2m_base, dir=tempdir())
  nwp_file <- dataDWD(nwp_urls[6], base=nwp_t2m_base, dir=tempdir(),
                      joinbf=TRUE, dbin=TRUE, read=FALSE)
  nwp_data <- readDWD(nwp_file, quiet=TRUE)
  plotRadar(nwp_data, project=FALSE)
  nwp_data_rgdal <- readDWD(nwp_file, toraster=FALSE)
  sp::plot(nwp_data_rgdal)
}
## End(Not run)
```
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

```r
readDWD.meta(file, quiet = rdwdquiet(), ...)
```

Arguments

- `file` Name of file on harddrive, like e.g. `DWDdata/daily_kl_recent_KL_Tageswerte_Beschreibung_Stationen.txt`
- `quiet` Ignored. DEFAULT: FALSE through `rdwdquiet()`
- `...` Further arguments passed to `read.fwf()`

Value

data.frame

Author(s)

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

See Also

`readDWD()`

Examples

```r
## 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))
```
readDWD.multia

## 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**

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

**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)
- **quiet** Ignored. DEFAULT: FALSE through `rdwdquiet()`
- **...** Further arguments passed to `read.table()`

**Value**

data.frame

**Author(s)**

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

**See Also**

`readDWD()`
Examples

```r
## 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, main="annual average ~ elevation")
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)

load(system.file("extdata/DEU.rda", package="rdwd"))
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)
```

---

**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 = rdwdquiet(),
  ...
)

Arguments

file Name of file on harddrive, like e.g. DWDdata/grids_germany/daily/Project_TRY/humidity/RH_199509_daymean.nc.gz

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 ... as varname. DEFAULT: TRUE

quiet Logical: Suppress time conversion failure warning? DEFAULT: FALSE through rdwdquiet()

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 <- plotRadar(nc, main=paste(nc@title, nc@z[[1]]), layer=1:3,
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, quiet = rdwdquiet(), ...)

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(), see readDWD.raster(). DEFAULT: 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

quiet Ignored. DEFAULT: FALSE through rdwdquiet()

... 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 `dwdadar::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

```r
## 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)

plotRadar(r$dat, main=r$meta$date)
```

## 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

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

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`
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

Logical: Divide the numerical values by 10? DEFAULT: TRUE

Ignored. DEFAULT: FALSE through `rdwdquiet()`

Further arguments passed to `raster::raster()`

Value

`raster::raster` object

Author(s)

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

See Also

`readDWD()`

Examples

```r
## 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])
rf10 <- readDWD(localfiles[1], dividebyten=FALSE)
raster::plot(rf)
pplotRadar(rf, proj="seasonal", extent=rf@extent)
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)
```
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

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

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/berry-b/rdwd/blob/master/R/updateIndexes.R. DEFAULT: formatIndex
- **quiet**: Ignored. DEFAULT: FALSE through rdwdquiet()
- **...**: 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

```r
## 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**

```r
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()`
readVars

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

```r
readVars(file, params = dwdparams, quiet = rdwdquiet(), progbar = TRUE)
```

Arguments

- `file` : Char (vector): name(s) of the file(s) downloaded with `dataDWD()`, e.g. "~/DWD-data/tageswerte_KL_02575_akt.zip"
- `params` : data.frame: Parameter explanations. DEFAULT: `dwdparams`
- `quiet` : Suppress message about non-abbreviated parameters? DEFAULT: FALSE through `rdwdquiet()`
- `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, newColumnNames()
readMeta() for complete Metadaten_Parameter file.
website use case

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,
  devcheck = !fast,
  radar = !fast,
  all_Potsdam_files = !fast,
  index = !fast,
  indexfast = fast,
  examples = !fast,
  quiet = rdwdquiet()
)

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
- devcheck: Run devtools::check()? DEFAULT: !fast
- 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
- index: Run checkIndex()? DEFAULT: !fast
- indexfast: fast option passed to checkIndex(). DEFAULT: !fast
- examples: Run Examples (including donttest sections) DEFAULT: !fast
- quiet: Suppress progress messages? DEFAULT: FALSE through rdwdquiet()

Value
Time taken to run tests in minutes

Author(s)
Berry Boessenkool, <berry-b@gmx.de>, Apr-Oct 2019
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](https://bookdown.org/brry/rdwd/available-datasets.html). To use an updated index (if necessary), see [https://bookdown.org/brry/rdwd/fileindex.html](https://bookdown.org/brry/rdwd/fileindex.html).

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 `findID()` using `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:

<table>
<thead>
<tr>
<th>case</th>
<th>id</th>
<th>path</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>base (and some warnings)</td>
</tr>
<tr>
<td>2</td>
<td>&quot;xx&quot;</td>
<td>&quot;&quot;</td>
<td>All file names (across paths) for station id</td>
</tr>
<tr>
<td>3</td>
<td>&quot;&quot;</td>
<td>&quot;xx&quot;</td>
<td>The zip file names at path</td>
</tr>
<tr>
<td>4</td>
<td>&quot;xx&quot;</td>
<td>&quot;xx&quot;</td>
<td>Regular single data file name</td>
</tr>
</tbody>
</table>

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 txt file at path. This is why case 3 with the default `meta=FALSE` only returns the data file names (ending in .zip) and not the description and Beschreibung txt/pdf files. Open those in a browser with

```r
dfdfpath <- grep("daily/kl/h.*DESCRIPTION", fileIndex$path, value=TRUE)
browseURL(paste0(dwdbase, "/", dfpdfpath))
```

Let me know if besides `meta`, `pdf` is needed for automated opening.

Usage

```r
selectDWD(
            name = "",
            res = NA,
```
selectDWD

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

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 resolution 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 variable of interest, like e.g. "air_temperature", "cloudiness", "precipitation", "soil_temperature", "solar", "kl", "more_precip" See above and in fileIndex. DEFAULT: NA for interactive selection
per Char: desired time period. 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
base Single char: main directory of DWD ftp server. Must be the same base used to create findx. DEFAULT: dwdbase
outvec Single logical: if path or ID length > 1, instead of a list, return a vector? (via unlist()). DEFAULT: per %in% c("rh","hr")
findx Single object: Index used to select filename, as returned by createIndex(). To use a current/custom index, see https://bookdown.org/brry/rdwd/fileindex.html. DEFAULT: fileIndex
remove_dupli Logical: Remove duplicate entries in the fileIndex? If duplicates are found, a warning will be issued, unless quiet=TRUE. The DWD updates files on the server quite often and sometimes misses removing the old files, leading to duplicates, usually with differences only in the date range. A semi-current (manually updated) list of duplicates is on github. Before reporting, run updateRdwd() to see if fileIndex has been updated. I email the DWD about duplicates when I find them, they usually fix it soon. If remove_dupli=TRUE, only the file with the longer timespan will be kept.
**selectDWD**

This is selected according to filename, which is not very reliable, hence manual checking is recommended. DEFAULT: TRUE

**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

**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, exactmatch, mindex)

**mindex**

Single object: Index with metadata passed to findID(). DEFAULT: metaIndex

**exactmatch**

Logical passed to findID(): match name with ==)? Else with grepl(). DEFAULT: TRUE

**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

**quiet**

Suppress id length warnings? DEFAULT: FALSE through rdwdquiet()

... 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, website station selection chapter

**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")
updateRdwd

# 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)

---

updateRdwd

## Update rdwd development version

### Description

Update rdwd to the latest development version on github, if necessary. If the version number or
date is larger on github, `remotes::install_github()` will be called.

### Usage

`updateRdwd(
  pack = "rdwd",
  user = "brry",
  vignette = TRUE,
  quiet = rdwdquiet(),
  ...
)

### Arguments

- **pack**: Name of (already installed) package. DEFAULT: "rdwd"
- **user**: Github username. repo will then be user/pack. DEFAULT: "brry"
- **vignette**: build_vignettes in `remotes::install_github()`? DEFAULT: TRUE
- **quiet**: Suppress version messages and `remotes::install_github()` output? DEFAULT: FALSE through `rdwdquiet()
- **...**: Further arguments passed to `remotes::install_github()`

## End
Value
data.frame with version information

Author(s)
Berry Boessenkool, <berry-b@gmx.de>, Nov 2019

See Also
remotes::install_github()

Examples
# updateRdwd()
Index

* aplot
  * addBorders, 3
  * plotRadar, 25
  * projectRasterDWD, 27
* character
  * findID, 15
  * rowDisplay, 48
* chron
  * readDWD, 30
* datasets
  * DEU, 10
  * dwdbase, 12
  * dwdparams, 12
  * EUR, 13
  * index, 16
  * metaInfo, 21
* data
  * dataDWD, 7
* debugging
  * runLocalTests, 49
* documentation
  * rdwd, 29
* file
  * dataDWD, 7
  * dirDWD, 11
  * fileType, 14
  * indexFTP, 17
  * localtestdir, 21
  * readDWD, 30
  * readMeta, 46
  * readVars, 47
  * selectDWD, 50
  * updateRdwd, 53
* manip
  * createIndex, 5
* package
  * checkSuggestedPackage, 5
  * rdwd, 29
* spatial
  * lldist, 20
  * plotRadar, 25
  * ==, 15, 52
  * addBorders, 3, 11, 13
  * addBorders(), 26
  * as.POSIXct(), 36
  * asc, 14, 28
  * berryFunctions::climateGraph(), 9
  * berryFunctions::convertUmlaut(), 38
  * berryFunctions::monthAxis(), 9
  * berryFunctions::newFilename(), 6, 19
  * berryFunctions::popleaf(), 48
  * berryFunctions::seqPal(), 26
  * binary, 14, 28
  * browseURL(), 8
  * cat(), 4
  * checkIndex, 4
  * checkIndex(), 6, 49
  * checkSuggestedPackage, 5
  * createIndex, 4, 5
  * createIndex(), 4, 16, 17, 19, 51
  * crs, 28
  * data, 14
  * data.table::fread, 36
  * data.table::fread(), 19, 36
  * dataDWD, 7
  * dataDWD(), 6, 11, 30, 31, 36, 46–48, 50, 52
  * DEU, 3, 10, 13, 25
  * dirDWD, 11
  * download.file(), 8, 9
  * dwdbase, 6, 8, 12, 14, 16, 19, 39, 45, 51
  * dwdparams, 12, 24, 47, 48
  * dwdparams(), 24
  * dwdradar::readRadarFile(), 34, 42, 43
  * EUR, 3, 11, 13, 25
  * extent, 28
INDEX

fileIndex, 4–6, 18, 51
fileIndex(index), 16
fileType, 14, 31
fileType(), 30, 31
findID, 15
findID(), 17, 50–52
formatIndex, 45
formatIndex(index), 16
geoIndex, 4–6, 48
geoIndex(index), 16
getwd(), 6, 8, 11, 19
grepl, 14
grib2, 14
gribbase, 8, 14, 16, 17
gribbase(dwdbase), 12
gridIndex, 18
gridIndex(index), 16
index, 7, 16
indexFTP, 17
indexFTP(), 5–8, 16, 17, 52
lldist, 20
localtestdir, 21
localtestdir(), 49, 50
max(), 20
maxlldist(lldist), 20
meta, 14
metaIndex, 4–6, 15, 22, 23, 52
metaIndex(index), 16
metaInfo, 21
metaInfo(), 16, 17
multia, 14
nc, 14, 28
ncdf4::nc_open(), 41
ncdf4::ncvar_get(), 41
nearbyStations, 22
newColumnNames, 24
newColumnNames(), 24, 36, 48
newFilename(), 9
plotRadar, 3, 25
plotRadar(), 28
print, 22
projectRaster, 28
projectRasterDWD, 27
projectRasterDWD(), 26, 32
R.utils::bunzip2, 37
R.utils::gunzip(), 41, 42, 44
radar, 14, 28
raster, 14, 28
raster::brick(), 41
raster::crs(), 28
raster::extent(), 28
raster::plot(), 3, 26
raster::raster, 44
raster::raster(), 32, 37, 44
raster::stack, 34, 42
raster::stack(), 32
raster::writeRaster(), 32
RCurl::getURL(), 18, 19
rdwd, 29
rdwd-package(rdwd), 29
rdwdquiet, 30
rdwdquiet(), 4, 6, 9, 11, 15, 19, 23, 26, 28, 31, 32, 34, 36–39, 41, 42, 44, 45, 47, 49, 52, 53
read.fwf(), 38, 45
read.table(), 36, 39, 45, 46
readDWD, 30
readDWD(), 8, 9, 13–15, 24–26, 32–45, 47, 48
readDWD.asc, 30, 32
readDWD.asc(), 31
readDWD.binary, 33
readDWD.binary(), 43
readDWD.data, 30, 35
readDWD.data(), 14, 24, 31, 39, 45
readDWD.grib2, 36
readDWD.meta, 38
readDWD.multia, 39
readDWD.nc, 40
readDWD.nc(), 31
readDWD.radar, 42
readDWD.radar(), 34
readDWD.raster, 30, 43
readDWD.raster(), 31, 37, 41, 42
readDWD.stand, 45
readLines(), 19
readMeta, 46
readMeta(), 14, 31, 48
readr::read_fwf(), 45
readVars, 47
readVars(), 12, 13, 24, 31, 47
remotes::install_github(), 53, 54
requireNamespace(), 5
rgdal::readGDAL, 37
rgdal::readGDAL(), 37
rowDisplay, 48
runLocalTests, 49
runLocalTests(), 21

selectDWD, 50
selectDWD(), 7–9, 16, 17, 23, 31
setwd(), 11
stand, 14
strptime(), 36
Sys.sleep(), 8, 19

tempdir(), 32
tempfile(), 6

unlist(), 51
untar(), 34
updateIndexes(), 7, 16, 19, 45
updateRdwd, 53
updateRdwd(), 51