Package ‘rdwd’

October 14, 2022

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

Version 1.6.0

Date 2022-05-30

Depends R(>= 2.10)

Imports berryFunctions (>= 1.21.11), pbapply

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

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 7.1.2

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

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-05-30 09:40:02 UTC
R topics documented:

addBorders .................................................. 3
checkIndex .................................................. 4
checkSuggestedPackage ..................................... 5
createIndex .................................................. 5
dataDWD ....................................................... 7
DEU .............................................................. 10
dirDWD .......................................................... 11
dwdbase .......................................................... 12
dwdparams ...................................................... 12
EUR ............................................................... 13
fileType .......................................................... 14
findID ............................................................. 16
index .............................................................. 17
indexFTP ......................................................... 18
ldist .............................................................. 20
locdir ............................................................. 21
metaInfo .......................................................... 22
nearbyStations ................................................ 23
newColumnNames .............................................. 24
plotDWD .......................................................... 25
plotRadar ........................................................ 26
projectRasterDWD ............................................ 29
rdwd ............................................................... 30
rdwdquiet ......................................................... 31
readDWD .......................................................... 31
readDWD.asc .................................................. 33
readDWD.binary ............................................... 35
readDWD.data .................................................. 36
readDWD.deriv ................................................ 38
readDWD.grib2 ............................................... 38
readDWD.meta .................................................. 40
readDWD.multia .............................................. 41
readDWD.nc .................................................... 43
readDWD.pdf ................................................... 44
readDWD.radar ............................................... 45
readDWD.raster ............................................... 47
readDWD.rklm .................................................. 48
readDWD.stand ............................................... 50
readMeta .......................................................... 52
readVars .......................................................... 53
rowDisplay ...................................................... 54
runLocalTests .................................................. 54
selectDWD ....................................................... 55
updateRdwd ..................................................... 58
validFileTypes ................................................. 59

Index .......................... 60
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 Bundeslaender line (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,  # fileIndex. DEFAULT: NULL
  mindex = NULL,  # metaIndex. DEFAULT: NULL
  gindex = NULL,  # geoIndex. DEFAULT: NULL
  excludefp = TRUE,  # Exclude false positives from geoIndex coordinate check results? DEFAULT: TRUE
  fast = FALSE,  # Exclude the 3-minute location per ID check? DEFAULT: FALSE
  warn = !quiet,  # Warn about issues? DEFAULT: !quiet (TRUE)
  logfile = berryFunctions::packagePath(file = "misc/ExampleTests/warnings.txt"),  # File to copy log to, appended to existing content. NULL to suppress. DEFAULT: "misc/ExampleTests/warnings.txt"
  quiet = rdwdquiet()  # Logical: Suppress progress messages? DEFAULT: FALSE through 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

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

data file and meta index of the DWD CDC FTP Server

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()

data file and meta index of the DWD CDC FTP Server

createIndex

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,
  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)
link4 <- "1_minute/precipitation/meta_data/Meta_Daten_ein_min_rr_00755.zip"
ind <- createIndex(link4, dir=tempdir())
ind
```

## 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**

```r
dataDWD(
  url,
  base = dwdbase,
  joinbf = FALSE,
)```
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 is not FALSE, the overwrite default is TRUE. DEFAULT: FALSE

overwrite Logical (vector): if force=TRUE, overwrite the existing file rather than append "/_1"/"_2" etc to the filename? DEFAULT: !isFALSE(force), i.e. true when force is specified.

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

method download.file method. Introduced in version 1.5.25 (2022-05-12) as triggered by https://github.com/brry/rdwd/issues/34. DEFAULT: getOption("download.file.method")

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
dataDWD

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
file Deprecated since rdwd version 1.3.34, 2020-07-28.
quiet 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=locdir(), 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]")
```r
# current and historical files:
link <- selectDWD("Potsdam", res="daily", var="kl", per="hr"); link
potsdam <- dataDWD(link, dir=locdir())
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**

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

See Also
addBorders, EUR

dirDWD directory management for rdwd

Description
Manage directories with useful messages in the rdwd package.

Usage
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
  # see source code of dataDWD and metaDWD
### dwdbase

**Description**

Base URLs to the DWD FTP Server

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

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

**Usage**

dwdbase

**Format**

An object of class character of length 1.

### dwdparams

**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 178 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

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
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; or contains multi_annual. Overrides meta.</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>data</td>
<td>.txt.gz</td>
<td>For data at /CDC/derived_germany/.</td>
</tr>
<tr>
<td>pdf</td>
<td>.pdf</td>
<td>only opens file in default viewer.</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>rklim</td>
<td>YW*.tar</td>
<td>For a file containing bin 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)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Jul 2020

See Also

readDWD()

Examples

ft <- read.table(header=TRUE, stringsAsFactors=FALSE, text="
type    daily_kl_recent_tageswerte_KL_03987_akt.zip
data    daily_kl_recent_KL_Tageswerte_Beschreibung_Sationen.txt
meta    multi_annual_mean_81-10_Temperatur_1981-2010_aktStandort.txt
multia    multi_annual_mean_81-10_Temperatur_1981-2010.txt
stand    subdaily_standard_format_kl_10381_00_akt.txt
deriv    derived_germany_soil_daily_historical_3987.txt.gz
pdf    DESCRIPTION_obsgermany_climate_monthly_kl_historical_en.pdf

radar    radolan_recent_bin_raa01-rw_10000-1802020250-dwd---bin.gz
binary    daily_radolan_historical_bin_2017_SF201712.tar.gz
raster    16_DJF_grids_germany_seasonal_air_temp_mean_188216.asc.gz
nc    daily_Project_TRY_humidity_RH_199509_daymean.nc.gz
asc    radolan_historical_asc_2018_RW-201809.tar
rklim    5_minutes_radolan_reproc_2017_002_bin_2020_YW2017_002_202006.tar

grib2    ftp_weather_nwp_cosmo-d2_005_T_2M.grib2.bz2
"
)

fileType(ft$filename)

stopifnot(fileType(ft$filename)==ft$type)

berryFunctions:::is.error(fileType("random_stuff.odt"), force=TRUE)

stopifnot(validFileTypes %in% ft$type)

stopifnot(ft$type %in% validFileTypes)
findID  

**find DWD weather station ID from name**

**Description**

Identify DWD weather station ID from station name

**Usage**

```r
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

### 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 `https://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, 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, dividebyten

**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
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 https://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:
gridindex <- indexFTP("", gridbase)
gridindex <- indexFTP(gridindex, gridbase, sleep=15)

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

Usage

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 https://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

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, "f" replaced with ",", ".txt" appended. DEFAULT: folder[1]
lldist

**Description**


**Usage**

```r
lldist(lat, long, data, r = 6371, i = 1L)
maxlldist(lat, long, data, r = 6371, fun = max, each = TRUE, ...)
```

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**

```r
## 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)
```
**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

---

**locdir**

*local data directory*

**Description**

This can be used to set a directory for DWD data across projects, thus avoiding multiple downloads of the same file.

Set the default for all subsequent calls with `options(rdwdlocdir="YOUR/PATH")`.

Currently, the `dataDWD()` dir defaults to a project specific folder at `getwd`. In the future, this may change to `locdir()`.

`locdir()` is used especially for the website, local tests and examples.

**Usage**

`locdir(dir = getOption("rdwdlocdir"), file = NULL, quiet = rdwdquiet())`

**Arguments**

- **dir**  
  Path to data directory. If `dir` does not exist, `tempdir()` is used instead (with a warning, unless `quiet=TRUE`). If `dir` is NULL, `locdir` tries "C:/DWDdata", then "~/DWDdata".
  
  `dir` can also be set with `options(rdwdlocdir="YOUR/PATH")` thanks to the DEFAULT: `getOption("rdwdlocdir")`

- **file**  
  Optional: path(s) at `dir`. DEFAULT: NULL

- **quiet**  
  Logical: suppress tempdir warning? DEFAULT: FALSE through `rdwdquiet()`
metaInfo

Value

charstring (directory)

Author(s)

Berry Boessenkool, <berry-b@gmx.de>, Apr 2019, Jun 2021

See Also

runLocalTests()

Examples

locdir()
oldopt <- options(rwdlocldir="-")
locdir()
stopifnot(locdir() == path.expand("-"))
options(oldopt) ; rm(oldopt)

Description

Information for a station ID on the DWD CDC FTP server

Usage

metaInfo(id, mindex = metaIndex, hasfileonly = TRUE)

Arguments

id Station ID (integer number or convertible to one)
mindex Index dataframe with metadata. DEFAULT: metaIndex
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
nearbyStations

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 = 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. 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 through rdwdquiet()
...

Value

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

---

**Examples**

```r
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")
```

---

**See Also**  
`selectDWD()`, `metaIndex`, website use case with `nearbyStations`
plotDWD

See Also
dwdparams, readVars(), readDWD() argument varnames, newColumnNames()

Examples

# mainly for internal usage

desc = "Quickly plot time series"

Usage
plotDWD(
  x,
  cn,
  monthaxis = TRUE,
  line0 = FALSE,
  xlab = "",
  ylab = cn,
  main = "",
  type = "l",
  lwd = 3,
  col = "blue",
  las = 1,
  mar = c(2.6, 3.1, 2.5, 0.5),
  mgp = c(1.9, 0.7, 0),
  keeppar = TRUE,
  ...
)

Arguments

x Data.frame, e.g. from readDWD.data
cn Column name (charstring)
monthaxis Draw nice axis? DEFAULT: TRUE
line0 Draw horizontal line at 0? DEFAULT: FALSE
xlab X axis label. DEFAULT: ""
ylab Y axis label. DEFAULT: cn
main Plot title. DEFAULT: ""
type graphics::plot type. DEFAULT: "l"
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",
  lwd = Line width. DEFAULT: 3
  col = Line color. DEFAULT: "blue"
  las = Label axis style. DEFAULT: 1 (all upright)
  mar = Plot margins. DEFAULT: c(2.6, 3.1, 2.5, 0.5)
  mgp = Margin placement. DEFAULT: c(1.9, 0.7, 0)
  keeppar = Keep las, mar and mgp as set with par, so later points are added in the right location? DEFAULT: TRUE
  ... = Further arguments passed to graphics::plot
)

Value

Nothing

Author(s)

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

See Also

readDWD()

Examples

link <- selectDWD("Potsdam", res="daily", var="kl", per="r")
clim <- dataDWD(link, dir=locdir(), varnames=TRUE)
plotDWD(clim, "TMK.Lufttemperatur", line0=TRUE, main="Potsdam")
plotRadar

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"
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`. DE-
FAULT: "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 re-
turned. `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=locdir())
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:
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=locdir())
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

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
- **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: "ll" 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 Germany's 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

```
```

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.
**rdwdquiet**

**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

---

**rdwdquiet**

*global quiet option for rdwd*

---

**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 \texttt{readDWD} are used for all subfunctions, e.g. \texttt{fread} is used only by \texttt{readDWD.data}, while \texttt{dividebyten} is used in \texttt{readDWD.raster} and \texttt{readDWD.asc}.

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

\textbf{Usage}

\begin{verbatim}
readDWD(
  file,
  type = fileType(file),
  varnames = FALSE,
  fread = NA,
  format = NA,
  tz = "GMT",
  dividebyten = TRUE,
  var = "",
  progbar = !quiet,
  quiet = rdwdquiet(),
  ...)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{file} Char (vector): name(s) of the file(s) downloaded with \texttt{dataDWD()}, e.g. "~/DWD-data/tageswerte_KL_02575_akt.zip" or "~/DWDdata/RR_Stundenwerte_Beschreibung_Stationen.txt"
  \item \texttt{type} Character (vector) determining which subfunction to call. DEFAULT: \texttt{fileType(file)}.
  \item \texttt{varnames} Logical (vector): Expand column names? Only used in \texttt{readDWD.data()}. DEFAULT: FALSE (for backward compatibility)
  \item \texttt{fread} Logical (vector): read fast? Used in \texttt{readDWD.data()}. DEFAULT: NA
  \item \texttt{format, tz} Format and time zone of time stamps, see \texttt{readDWD.data()}
  \item \texttt{dividebyten} Logical (vector): Divide the values in raster files by ten? That way, [1/10 mm] gets transformed to [mm] units. Used in \texttt{readDWD.radar(),readDWD.raster()} and \texttt{readDWD.asc()}. DEFAULT: TRUE
  \item \texttt{var} var for \texttt{readDWD.nc()}. DEFAULT: ""
  \item \texttt{progbar} Logical: present a progress bar with estimated remaining time? If missing and \texttt{length(file)==1}, \texttt{progbar} is internally set to FALSE, unless binary files are to be read. DEFAULT: !quiet
  \item \texttt{quiet} Logical: suppress messages? DEFAULT: FALSE through \texttt{rdwdquiet()}
  \item ... Further arguments passed to the internal \texttt{readDWD.*} subfunctions (see \texttt{fileType}) and from those to the underlying actual reading functions
\end{itemize}

\textbf{Value}

For observational data, an invisible \texttt{data.frame} of the desired dataset, or a named list of \texttt{data.frame}s if \texttt{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

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? See readDWD. 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()`

### probar

Show progress bars? `readDWD()` will keep probar=TRUE for asc files, even if length(file)==1. DEFAULT: !quiet, i.e. TRUE

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

#### Value

data.frame

#### Author(s)

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

#### See Also

`readDWD()`

#### Examples

```r
## Not run: # Excluded from CRAN checks, but run in localtests
# File selection and download:
datadir <- locdir()
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_%03d.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)
```
readDWD.binary

read 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,
  quiet = rdwdquiet(),
  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: sub(".tar.gz", "", file)
- **toraster**: Logical: convert output (list of matrixes + meta informations) to a list with dat (raster::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 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 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=locdir(), 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 <- 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=locdir(), 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 <- 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().
readDWD.data

Usage

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 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 strftime()) 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()

Value
data.frame

Author(s)

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

See Also

readDWD(), Examples in dataDWD()
**readDWD.deriv**  
**read derived dwd data**

**Description**
Read dwd data from `/CDC/derived_germany/`. Intended to be called via `readDWD()`.

**Usage**
```
readDWD.deriv(file, gargs = NULL, todate = TRUE, quiet = rdwdquiet(), ...)  
```

**Arguments**
- `file`  
  Name of file on harddrive, like e.g. DWDdata/soil_daily_historical Derived_germany soil_daily_historical
- `gargs`  
  If `fread=FALSE`: Named list of arguments passed to `R.utils::gunzip()`, see `readDWD.raster()`. DEFAULT: NULL
- `todate`  
  Logical: Convert char column 'Datum' or 'Monat' with `as.Date()`? The format is currently hard-coded. Monthly data gets mapped to yyyy-mm-15  DEFAULT: TRUE
- `quiet`  
  Ignored. 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.grib2**  
**read nwp forecast data**

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

readDWD.grib2(
  file,
  pack = "terra",
  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
pack Char: package used for reading. One of "terra" (the default), "stars" or "rgdal"
  (for the deprecated cosmo-d2 data). See issue. DEFAULT: "terra"
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()? Only used if pack="rgdal". DEFAULT: TRUE
quiet Silence readGDAL completely, including warnings on discarded ellps / datum. DEFAULT: FALSE through rdwdquiet()
...
Further arguments passed to stars::read_stars(), rgdal::readGDAL() or
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

## Not run: # Excluded from CRAN checks, but run in localtests
nwp_t2m_base <- "ftp://opendata.dwd.de/weather/nwp/icon-d2/grib/15/soiltyp"
nwp_urls <- indexFTP("", base=nwp_t2m_base, dir=tempdir())
# for p instead of soiltyp, icosahedral_model-level files fail with GDAL errors,
# see https://github.com/brry/rdwd/issues/28
# regular-lat-lon_pressure-level files work with pack="terra" or "stars"
nwp_file <- dataDWD(tail(nwp_urls,1), base=nwp_t2m_base, dir=tempdir(), joinbf=TRUE, dbin=TRUE, read=FALSE)
nwp_data <- readDWD(nwp_file)
terra::plot(nwp_data) # same map with sp::plot
addBorders() # the projection seems to be perfectly good :)

# index of GRIB files
if(FALSE){ # indexing takes about 6 minutes!
grib_base <- "ftp://opendata.dwd.de/weather/nwp/icon-d2/grib"
grib_files <- indexFTP("", base=grib_base, dir=tempdir())
for(f in unique(substr(grib_files, 1,3))) print(grib_files[which(substr(grib_files, 1,3)==f)[1]])
View(data.frame(grep("regular",grib_files, value=TRUE)))
}

## End(Not run)

---

**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, 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 local tests

link <- selectDWD(res="daily", var="kl", per="r", meta=TRUE)
link <- link[!grepl("mn4", link)]  # for mn4 file May 2022
link <- grep(".txt$", link, value=TRUE)
if(length(link)!=1) stop("length of link should be 1, but is ", length(link),
  ":
", berryFunctions::truncMessage(link, prefix="", sep="\n"))

file <- dataDWD(link, dir=locdir(), 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),
  ":
", 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**

```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 local tests

# Temperature aggregates (2019-04 the 9th file, 2022-05 the 8th):
durl <- selectDWD(res="multi_annual", var="mean_81-10", per="")[8]
murl <- selectDWD(res="multi_annual", var="mean_81-10", per="", meta=TRUE)[8]

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

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="grey90")
for(m in colnames(ma)[8:19])
  {
    raster::plot(DEU, border="grey40")
    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**

```r
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

```r
## Not run:  # Excluded from CRAN checks, but run in local tests

library(berryFunctions)  # for seqPal and colPointsLegend

url <- "daily/ProjectTRY/pressure/PRED_199606_daymean.nc.gz"  # 5 MB
url <- "daily/ProjectTRY/humidity/RH_199509_daymean.nc.gz"  # 25 MB
file <- dataDWD(url, base=gridbase, joinbf=TRUE, dir=locdir(), read=FALSE)
nc <- readDWD(file)
ncp <- plotRadar(nc, main=paste(nc@title, nc@z[[1]]), layer=1:3,
                 col=seqPal(), proj="nc", extent="nc")
str(nc, max.level=2)

raster::values(nc[[1]])  # obtain actual values into memory

raster::plot(nc[[1]])  # axes 0:938 / 0:720, the number of grid cells
raster::plot(ncp[[1]])  # properly projected, per default onto latlon

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)
```

Description

open pdf file. This leads to less failures in the new meta=TRUE

Usage

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

Arguments

- `file` Name of file on harddrive, like e.g. monthly_kl_historical_DESCRIPTION_obsgermany_climate_monthly
- `quiet` Ignored. DEFAULT: FALSE through `rdwdquiet()
- `...` Further arguments passed to `berryFunctions::openFile()` and from there to `system2()`
**readDWD.radar**

**Value**

`berryFunctions::openFile()` output

**system in selectDWD().**

Intended to be called via `readDWD()`.

**Author(s)**

Berry Boessenkool, <berry-b@gmx.de>, May 2022.

**See Also**

`readDWD()`

**Examples**

```r
## Not run: # Excluded from CRAN checks, but run in localtests
link <- selectDWD(res="hourly", var="sun", per="r", meta=TRUE)[2]
file <- dataDWD(link, dir=locdir(), read=FALSE)
readDWD(file)
```

## End(Not run)

---

**readDWD.radar**

**read dwd gridded radolan radar data**

**Description**

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

**Usage**

```r
readDWD.radar(
  file,
  gargs = NULL,
  toraster = TRUE,
  dividebyten = TRUE,
  quiet = rdwdquiet(),
  ...
)
```
**Arguments**

- **file**  
  Name of file on hard drive, like e.g. DWDdata/hourly/radolan/recent/bin/radolanbin.gz

- **gargs**  
  Named list of arguments passed to `R.utils::gunzip()`, see `readDWD.raster()`. DEFAULT: NULL

- **toraster**  
  Logical: convert output (list of matrices + meta information) to a list with data (`raster::stack`) + meta (list from the first subfile, but with vector of dates)? DEFAULT: TRUE

- **dividebyten**  
  Logical: Divide the numerical values by 10? See `readDWD`.toraster??? 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 `dwdradar::readRadarFile()` came from Henning Rust & Christoph Ritschel at FU Berlin.

**See Also**

- `readDWD()`, especially `readDWD.binary()`
- [https://wradlib.org](https://wradlib.org) for much more extensive radar analysis in Python
- Kompositformatbeschreibung at [https://www.dwd.de/DE/leistungen/radolan/radolan.html](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)
plotRadar(r$dat, main=paste("mm in 24 hours preceding", r$meta$date))
```

## End(Not run)
readDWD.raster

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, 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
- **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, use 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? See readDWD. DEFAULT: TRUE
- **quiet**: 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

```
## 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=locdir(), read=FALSE)
rf <- readDWD(localfiles[1])
rf <- readDWD(localfiles[1]) # runs faster at second time due to skip=TRUE
```
```r
raster::plot(rf)

plotRadar(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)
```

---

**readDWD.rklim**

read dwd gridded radklim binary data

---

**Description**

read gridded radklim binary data. Intended to be called via `readDWD()`.
Note: needs dwdradar >= 0.2.6 (2021-08-08)

**Usage**

```r
readDWD.rklim(
  file,
  exdir = NULL,
  unpacked = NULL,
  selection = NULL,
  toraster = TRUE,
  quiet = rdwdquiet(),
  progbar = !quiet,
  ...
)
```

**Arguments**

- **file**
  - Name of file on hard drive, like e.g. DWDdata/5_minutes_radolan_reproc_2017_002_bin_2020_YW2017.tar

- **exdir**
  - Directory to unzip into. If existing, only the needed files will be unpacked with `untar()`. Note that exdir size will be around 17 GB for 5-minute files. If `unpacked=FALSE`, exdir can contain other files that will be ignored for the actual reading. DEFAULT: basename(file) at tempdir

- **unpacked**
  - Manually indicate whether .tar.gz files within .tar file have already been unpacked before. DEFAULT: NULL: checks if `yw.*-bin` file(s) are present

- **selection**
  - Optionally read only a subset of the ~ 12 x 24 x 30/31 = 8640 files. Called as `f[selection]`. DEFAULT: NULL (ignored)

- **toraster**
  - Logical: convert to raster stack? see `readDWD.binary` DEFAULT: TRUE

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

- **progbar**
  - Show progress bars? DEFAULT: !quiet, i.e. TRUE

... 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>, Aug 2021.

See Also

`readDWD.binary()`, radar locations from https://www.dwd.de/DE/leistungen/radarklimatologie/radklimKompositformat_1_0.pdf?__blob=publicationFile&v=1

Examples

```r
## Not run: # Excluded from CRAN checks, but run in local tests
yw_link <- "/5_minutes/radolan/reproc/2017_002/bin/2020/YW2017.002_202006.tar"
yw_file <- dataDWD(url=yw_link, base=gridbase, joinbf=TRUE, dir=locdir(), read=FALSE)
x <- readDWD(yw_file, selection=3641:3644)
# 00:30 for tar files, 01:40 for unpacking.
# If you need a preselection argument, let me know.
raster::plot(x$dat)

f <- system.file("tests//raa01-yw2017.002_10000-2006131525-dwd---bin", package="dwdradar")
if(f="") stop("dwdradar test file not found")
# https://stackoverflow.com/a/72207233/1587132 on how to install with tests folder
x <- dwdradar::readRadarFile(f)
x$dat <- raster::raster(x$dat)
raster::plot(x$dat)
plotRadar(x$dat, extent=c(-360, 380, -4730, -3690))

radloc <- read.table(header=T, sep="", text="
ND, NM, NS , ED, EM, ES
53, 33, 50.4, 06, 44, 53.9
51, 07, 26.5, 13, 45, 48.5
51, 24, 18.5, 06, 57, 49.8
47, 52, 21.3, 08, 00, 24.6
54, 10, 23.2, 12, 06, 25.3
52, 28, 40.3, 13, 23, 13.0
54, 00, 15.8, 10, 02, 48.7
51, 07, 28.7, 13, 46, 07.1
49, 32, 26.4, 12, 24, 10.0
53, 20, 19.4, 07, 01, 25.5
51, 24, 20.2, 06, 58, 01.6
47, 52, 25.0, 08, 00, 13.0
51, 20, 06.0, 08, 51, 09.0
51, 18, 40.3, 08, 48, 07.2
50, 03, 06.0, 08, 34, 05.0
50, 01, 20.8, 08, 33, 30.7
53, 37, 16.5, 09, 59, 47.6
52, 27, 47.0, 09, 41, 53.9
52, 27, 36.2, 09, 41, 40.2
```
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,
  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/brry/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
## Not run: # Excluded from CRAN checks, but run in localtests

```r
link <- selectDWD(id=10381, res="subdaily", var="standard_format", per="r")
file <- dataDWD(link, dir=locdir(), 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)
```
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()`

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

```r
# 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**

```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**

```r
# 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 = locdir(),
  dir_exmpl = berryFunctions::packagePath(file = "misc/ExampleTests"),
  fast = FALSE,
  devcheck = !fast,
  radar = !fast,
  all_Potsdam_files = !fast,
  index = !fast,
)
selectDWD

indexfast = fast,
examples = !fast,
quiet = rdwdquiet()
)

Arguments

dir_data  Reusable data location. Preferably not under version control. DEFAULT: locdir()
dir_exmpl Reusable example location. DEFAULT: local directory
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

See Also

locdir()

selectDWD  Select data from the DWD CDC FTP Server

Description

Select data files for downloading with dataDWD(). The available res/var/per folders with datasets are listed online. Set res="", var="", per="" to avoid the default interactive selection. The arguments name/id and res/var/per can be vectors.
Usage

selectDWD(
  name = "", 
  res = NA, 
  var = NA, 
  per = NA, 
  id = findID(name, exactmatch = exactmatch, mindex = mindex, quiet = quiet), 
  exactmatch = TRUE, 
  mindex = metaIndex, 
  findex = fileIndex, 
  current = FALSE, 
  base = dwdbase, 
  meta = FALSE, 
  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 at base, e.g. "hourly", "daily", "monthly". See section 'Description' above and fileIndex. Use res="" for matching options from all resolutions. DEFAULT: NA for interactive selection

var: Char: weather variable of interest, e.g. "air_temperature", "cloudiness", "precipitation", "solar", "kl". See section 'Description' above and fileIndex. DEFAULT: NA for interactive selection

per: Char: desired time period, e.g. "recent" (up to date records from the last 1.5 years) or "historical" (long time series). Can be abbreviated. To get both datasets, use per="hr". DEFAULT: NA for interactive selection

id: Char/Number: station ID with or without leading zeros, e.g. "00614" or 614. Is internally converted to an integer. DEFAULT: findID(name, exactmatch, mindex)

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

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

findex: Single object: Index used to select filename, as returned by createIndex(). To use a current/custom index, see current and https://bookdown.org/brry/rdwd/fileindex.html. DEFAULT: fileIndex

current: Single logical when res/var/per is given: 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. See https://bookdown.org/brry/rdwd/fileindex.html. DEFAULT: FALSE

base: Single char: main directory of DWD ftp server. Must be the same base used to create findex. DEFAULT: dwdbase
selectDWD

meta Logical: select Beschreibung file from ismeta entries in findex? See metaIndex for a compilation of all Beschreibung files. See the 'Examples' section for handling pdf and txt files. DEFAULT: FALSE

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

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, rewritten May 2022

See Also
dataDWD(), metaIndex, website station selection chapter

Examples

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

# all files for all stations matching "Koeln":
tail(selectDWD("Koeln", res="", var="", per="", exactmatch=FALSE)) # 686 files
findID("Koeln", FALSE)

## Not run: # Excluded from CRAN checks to save time
# selectDWD("Potsdam") # interactive selection of res/var/per

# directly give station ID:
selectDWD(id="00386", res="daily", var="kl", per="historical")
selectDWD(id=537, "", "", "", "") # 8 files

# period can be abbreviated:
selectDWD(id="5419", res="daily", var="kl", per="h")

# selectDWD is vectorizable!
# since version 1.5.28 (2022-05-12) outer product, not elementwise comparison:
selectDWD("Freiburg", res="daily", var="kl", per="rh")
selectDWD("Freiburg", res=c("daily","monthly"), var="kl", per="r")
selectDWD("Freiburg", res=c("daily","monthly"), var="kl", per="hr")

# all files in all paths matching id:
head( selectDWD(id=c(1050, 386), res="",var="",per="") ) # 277 files
# all files in a given path (if ID is empty):
head( selectDWD(id="", res="daily", var="kl", per="recent") ) # 585 files
selectDWD(id=386, res="monthly", var="kl", per="h")
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

```r
updateRdwd(
  pack = "rdwd",
  user = "brry",
  vignette = NA,
  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: NA (changed to TRUE if rmarkdown and knitr are available)
- **quiet** Suppress version messages and `remotes::install_github()` output? DEFAULT: FALSE through `rdwdquiet()`
- **...** Further arguments passed to `remotes::install_github()`

Value

data.frame with version information
validFileTypes

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

See Also
remotes::install_github()

Examples
# updateRdwd()

---

validFileTypes   valid fileType values

Description
fileType values that have a reading subfunction readDWD.ftype().

Usage
validFileTypes

Format
An object of class character of length 13.
Index

* aplot
  - addToBorders, 3
  - plotRadar, 26
  - projectRasterDWD, 29
* character
  - findID, 16
  - rowDisplay, 54
* chron
  - readDWD, 31
* datasets
  - DEU, 10
  - dwdbase, 12
  - dwdparams, 12
  - EUR, 13
  - index, 17
  - metaInfo, 22
  - validFileTypes, 59
* data
  - dataDWD, 7
* debugging
  - runLocalTests, 54
* documentation
  - rdwd, 30
* file
  - dataDWD, 7
  - dirDWD, 11
  - fileType, 14
  - indexFTP, 18
  - locdir, 21
  - readDWD, 31
  - readMeta, 52
  - readVars, 53
  - selectDWD, 55
  - updateRdwd, 58
* hplot
  - plotDWD, 25
* manip
  - createIndex, 5
* package
  - checkSuggestedPackage, 5
  - rdwd, 30
* spatial
  - lldist, 20
  - plotRadar, 26
  -=, 16, 56
  - addBorders, 3, 11, 13
  - addBorders(), 28
  - as.Date(), 38
  - as.POSIXct(), 37
  - asc, 14, 30
  - berryFunctions::climateGraph(), 9
  - berryFunctions::convertUmlaut(), 40
  - berryFunctions::monthAxis(), 9
  - berryFunctions::newFilename(), 6, 20
  - berryFunctions::openFile(), 44, 45
  - berryFunctions::popleaf(), 54
  - berryFunctions::seqPal(), 27
  - binary, 14, 30
  - browseURL(), 9
  - cat(), 4
  - checkIndex, 4
  - checkIndex(), 6, 55
  - checkSuggestedPackage, 5
  - createIndex, 4, 5
  - createIndex(), 4, 17, 20, 56
  - crs, 30
  - data, 14
  - data.table::fread, 37
  - data.table::fread(), 19, 37, 38
  - dataDWD, 7
  - dataDWD(), 6, 11, 21, 31–33, 37, 52, 53, 55, 57
  - DEU, 3, 10, 13, 27
  - dirDWD, 11
  - download.file, 8
  - download.file(), 8, 9
INDEX

dwdbase, 6, 8, 12, 14, 17, 19, 41, 50, 56

dwdparams, 12, 24, 25, 53
dwdparams(), 24
dwdradar::readRadarFile(), 35, 46, 48

EUR, 3, 11, 13, 27

extent, 30

fileIndex, 4–6, 18, 56
fileIndex(index), 17
fileType, 14, 32
fileType(), 31, 33
findID, 16, 56
findID(), 17, 56
formatIndex, 50, 51
formatIndex(index), 17

geoIndex, 4–6, 54
geoIndex(index), 17
getOption, 21
getwd, 21
getwd(), 6, 8, 11, 19
graphics::plot, 25, 26
grepl(), 16, 56
grib2, 14
gridbase, 8, 14, 17
gridbase(dwdbase), 12
gridIndex, 18
gridIndex(index), 17

index, 7, 17
indexFTP, 18
indexFTP(), 5–8, 17, 56, 57

lldist, 20
locdir, 21
locdir(), 55

max(), 21
maxlldist(lldist), 20
meta, 14
metaIndex, 4–6, 16, 22–24, 56, 57
metaIndex(index), 17
metaInfo, 22
metaInfo(), 16, 17
multia, 14

nc, 14, 30
ncdf4::nc_open(), 43
ncdf4::ncvar_get(), 43

nearbyStations, 23
newColumnNames, 24
newColumnNames(), 25, 37, 53
newFilename, 9

par, 26
pdf, 14
plotDWD, 25
plotRadar, 3, 26
plotRadar(), 30
print, 22
projectRaster, 30
projectRasterDWD, 29
projectRasterDWD(), 27, 28, 33

R.utils::bunzip2(), 39
R.utils::gunzip(), 46, 48
raster, 14, 30
raster::brick(), 43
raster::crs(), 29
raster::extent(), 29
raster::plot(), 3, 28
raster::raster(), 47
raster::raster(), 34, 39, 47
raster::stack, 35, 46
raster::stack(), 33
raster::writeRaster(), 33
RCurl::getURL(), 18, 20
rdwd, 30
rdwd-package(rdwd), 30
rdwdquiet, 31
rdwdquiet(), 4, 6, 9, 11, 16, 20, 21, 23, 28,
29, 32, 34, 35, 37–41, 43, 44, 46–48,
51, 53, 55, 57, 58
read.fwf(), 40, 50, 51
read.table(), 37, 38, 41, 50, 52
readDWD, 31, 33, 46, 47
readDWD(), 8, 9, 13–15, 25–38, 33–43, 45–48,
50–53
readDWD.asc, 32, 33
readDWD.asc(), 32
readDWD.binary, 35, 48
readDWD.binary(), 46, 49
readDWD.data, 25, 32, 36
readDWD.data(), 14, 24, 32, 41, 50
readDWD.deriv, 38
readDWD.grib2, 38
readDWD.meta, 40
readDWD.multia, 41
readDWD.nc, 43
readDWD.nc(), 32
readDWD.pdf, 44
readDWD.radar, 45
readDWD.radar(), 32, 36
readDWD.raster, 32, 47
readDWD.raster(), 32, 38, 39, 43, 46
readDWD.rklim, 48
readDWD.stand, 50
readLines(), 19
readMeta, 52
readMeta(), 14, 33, 53
readr::read_fwf(), 50, 51
readVars, 53
readVars(), 12, 13, 24, 25, 33, 52
remotes::install_github(), 58, 59
requireNamespace(), 5
rgdal::readGDAL, 39
rgdal::readGDAL(), 39
rklim, 14
rowDisplay, 54
runLocalTests, 54
runLocalTests(), 22

selectDWD, 55
selectDWD(), 7–9, 16, 17, 23, 24, 33, 45
setwd(), 11
stand, 14
stars::read_stars(), 39
strptime(), 37
Sys.sleep(), 8, 19
system2(), 44

tempdir(), 21, 33
tempfile(), 6

untar(), 35, 48
updateIndexes(), 7, 17, 20, 50
updateRdwd, 58

validFileTypes, 59