Package 'GhcnDaily'

February 19, 2015

Type Package

Title Downloads and processes GHCN daily

Version 1.5

Date 2012-07-21

Author Steven Mosher,

Maintainer Steven Mosher <mosher steven@gmail.com>

Depends R (>= 2.11.0), R.utils, R.oo, R.methodsS3, abind, ncdf

Description The GhcnDaily package provides the core functions required to download and format the GHCN Daily Data (TMax and Tmin) into Monthly datasets. It is currently restricted to operating with temperature data from the daily files.

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-07-22 06:20:17

NeedsCompilation no

R topics documented:

`GhcnDaily-package` *Ghcn Daily Climate Data*

Description

A package that downloads and processes GHCN daily data. NCDC maintains an ftp repository of climate stations that reported on a daily basis. The data consists of over 77000 stations with reporting periods giong back over 100 years. many "elements" of the climate are recorded. This package is primarly concerned with temperature measures, although some functions can be used to retrieve other "elements"

Details

Package:	GhcnDaily
Type:	Package
Version:	1.5
Date:	2012-07-21
License:	GPL (>=2)
LazyLoad:	yes
LazyData:	yes

1.0 Introduction

The purpose of this package is currently limited to downloading and formating NCDC's Global Historical Climate Network daily temperature data. The source files conatin more data than those two variables and future releases will handle them. The process flow proceeds roughly according to

these steps. A station inventory is downloaded to your local machine `downloadDailyInventory`. This copies a 21MB file to your workspace. The file contains information about all of the station files. For example the Id, the elements it contains (TMAX and TMIN) and the first and last year of data. You should first run the demo code `demo(DownloadDemo)`.. That will get you the base files to do the data file downloads. The Inventory can be read with the function `readDailyInventory` and processed to select subsets of data to download. One a list is estabilised you build a download list with `makeDownloadList`. This function merely creates a list of urls that used to fetch the station data. Data is downloaded via one of two function `downloadDailyData` or the more comprehensive `downloadDailyFiles`. The latter copies all data to disk. It can take a day on a fast connection at roughly 1000 files per hour. using `downloadDailyData` The data from the "dly" files is downloaded to the "DailyRaw" directory. Only TMAX and TMIN data is downloaded. The "Raw" data includes all data and the QA flags. This data in then processed by the function `convertRawToDat`. The QA flags are applied and you have clean data in the "DailyData" directory. The next step is to create a data array from all the individual files in the data directory. The function `createArray` is the workhorse function here. It takes an inventory of stations, and element (TMAX or TMIN) and reads the directory for all the files that are in the inventory. It systematically build a 3D array of data. The dimensions are 'station','month' and 'year'. That array can then be written to disk as one file containing all the station data. This is done with `writeNetCDF` or `writeGhcn`. NetCDF is the preferred format.

Author(s)

Steven Mosher

Maintainer: Steven Mosher <mosher steven@gmail.com>

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
## Not run:
  # get all the data
  demo(getAlldata)

## End(Not run)
```

`convertFiles` *Convert an entire directory of data*

Description

Ghcn daily files contain data beyong maximum and minimum temperatures (over 30 elements). The GhcnDaily package supports the download of these complete files. Converting these files to temperature only files can be accomplished by calling the function `convertFiles`

Usage

```
convertFiles(sourceDir = DAILY.FILES.DIRECTORY,
                  destDir = DAILY.DATA.DIRECTORY)
```

Arguments

sourceDir	The source directory where the .dly files are kept. By default this is "DailyFiles" which is the default value of DAILY.FILES.DIRECTORY
destDir	The destination directory where the .raw files are written. By default this is "DailyRaw" which is the default value of DAILY.DATA.DIRECTORY

Details

The .dly files are read and if the file contains a TMAX or TMIN element then that data is read out and written to a the .raw file in the destination directory. Filenames remain the same. That is the station Id is the filename

Value

Side effect is converting all the files in the source directory to data files in the destination directory

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt See the readme for documentation on the .dly files

Examples

```
## Not run:

  # write sample code

## End(Not run)
```

---

convertRawToDat *Converts raw files with QA flags to dat files*

Description

Dly files contain QA flags. When these files are converted to *.raw files the QA flags are retained. In order to further process the data this function will apply the QA flags and remove any temperatures that Fail QA. *.dat files are output by this function

Usage

```
convertRawToDat(QaFlags = c("D","G","I","K","L","M","N","O","R","S","T","W","X"),
    sourceDir = DAILY.DATA.DIRECTORY, destDir = DAILY.QA.DIRECTORY)
```

Arguments

QaFlags	This is a list of all the QA flags that will be applied. When a QA flag is applied to the daily data that data is set to NA. For a list of flags see the ftp readme.
sourceDir	The source directory of the *.raw files
destDir	The source directory of the *dat files

Details

Files are downloaded to the dailyData directory. Running this function applies the QA flags and removes bad values. the final format is *.dat

Value

side effect is to write a directory of data

Author(s)

Steven Mosher

References

see ftp readme

Examples

```
## Not run:
  # sample code

## End(Not run)
```

---

createArray *Create a 3D array of temperature data*

Description

Station data is comprised of a station Id and a series of temperatures in 1/10ths of a degree C. Other packages such as RghcnV3 make use of a 3D array to analyze this data. The data is stored in an array with dimensions -station, month, year. This function creates an entire collection of stations given the input parameters and returns an array. That array can then be written to disk `writeNetCdf` or processed further.

Usage

```
createArray(Inventory, element = "TMAX",
                indir = DAILY.QA.DIRECTORY, na.rm = meanFunction, ...)
```

Arguments

Inventory	An inventory of stations. This inventory is created by readDailyInventory. It should have one and only one data element, either TMAX or TMIN.
element	character string of "TMAX" or "TMIN".
indir	The input directory where the daily data files are located
na.rm	The variable controls how monthly means will be calculated from daily data. The default is to use the meanFunction. That function allows you to specify the allowabel number of days missing before an NA is returned for the monthly mean. Also na.rm can be set to TRUE or FALSE and in this case the stand mean function of R will be used. Also, the user can define his own function for calculating means and pass it to this function.
...	additional parameters passed through to the meanFunction

Details

The code reads a daily station .dat file and then processes that file to obtain monthly means. Then the monthly means are inserted into the data array at the proper location. dimension 1 is stations. Dimension 2 is months and dimension 3 is years. All years and months are filled with NA if they are missing in the source data so that the time series are continous from start to finish. Dimnames are also set for each of the dimensions and so the object is self documenting. (version 1.1 will add this as an attribute)

Value

Return a 3D array of temperature data. Monthly means

Note

The 3D array structure was inherited from Nick Stokes work

Author(s)

Steven Mosher

See Also

meanFunction

Examples

```
 ## Not run:
 # sample code

## End(Not run)
```

createStationMonthly *Converts a daily series of temperatures into a monthly series*

Description

GHCN Daily data consists of daily Tmax and daily Tmin data. This function takes daily data as an input and outputs a monthly mean. The monthly mean is provided for one measure. That is, if the function is fed Tmax, it returns Tmax monthly. Calculating the mean for a month is ultimately controlled by the function `meanFunction` which can also be user defined.

Usage

```
createStationMonthly(data, na.rm = meanFunction, ...)
```

Arguments

`data`	A matrix of temperatures created by `readDailyData` The matrix has station Id as rownames, Year in the first column month in the second, followed by data for 31 days (NAs for missing)
`na.rm`	This parameter controls the way a mean is calculated. set to FALSE and Rs standard `mean` function will be used. Any month with NA data will get an NA result. Set to TRUE and Rs standard `mean` function will be used. In most cases it should be set to the function `meanFunction`
`...`	The dots are provided to pass paramaters to `meanFunction` or any user defined function.

Details

Data is read in from `readDailyData` and passed to this function. Calculating a "mean" monthly temperature can be user defined by setting `na.rm` to the user function or to the function `meanFunction`. Essentially, the issue is how one computes a mean in the presence of NA. R's basic function `mean` provides two options. Ignore NAs or return NA if there is one single NA in the vector being averaged. The by using the `na.rm` variable the user can control how NAs are treated. You can define a function that will return a mean if 3 days are missing, 13 days, 2 consecutive days, etc.

Value

The function returns a long matrix with two columns. column number 1 is the date in fractional years. Missing months and missing years in the sequences are infilled. That is, the daily source data does not have records if an entire month is missing or an entire year is missing in a sequence. Ultimately, all the records of all the stations must be merged into a "complete" record with NAs for all missing data. This function does that for an individual measurand. TMAX or TMIN for the station provided is "infilled" with NA to give it a continuous gap free record form the first year of that station to the last year.

Author(s)

Steven Mosher

See Also

readDailyData, codemeanFunction

Examples

```
## Not run:
  # example code

## End(Not run)
```

DAILY.DATA.DIRECTORY *Directory for raw station data with QA flags*

Description

The local directory where all functions expect to find the raw source data. All files here have a .raw extension

Usage

```
DAILY.DATA.DIRECTORY
```

Format

The format is: chr "DailyRaw"

Details

The data directory name. Avoid changing it unless understand the code completely. Recall that on start this value will always be read into the variable.

Examples

```
print(DAILY.DATA.DIRECTORY)
```

DAILY.FILES.DIRECTORY *Directory for downloaded .dly files*

Description

The function `downloadDailyFiles` will download the entire dly file to this directory. These files contain all the data for a station , not just temperatures. Downloading all the files is a time consuming process and take over 24 hours even on a fast connection. There are 26000+ files that conatin temperatures plus other variables.

Usage

```
DAILY.FILES.DIRECTORY
```

Format

The format is: chr "DailyFiles"

Source

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
print(DAILY.FILES.DIRECTORY)
```

DAILY.QA.DIRECTORY *Data after QA flags applied*

Description

Data is downloaded with various QA flags. After they have have been applied the data is given a .dat extention and placed in the QA directory

Usage

```
DAILY.QA.DIRECTORY
```

Format

The format is: chr "DailyData"

Details

see ftp readme for flags

Examples

```
print(DAILY.QA.DIRECTORY)
```

`downloadCountry` *Downloads Country FIPS and name*

Description

Abbreviations for countries and their names

Usage

```
downloadCountry(url = GHCN.DAILY.COUNTRY.URL,
                dest = "countryCodes.txt", directory = getwd())
```

Arguments

`url`	url defaults to the URL value set globally.
`dest`	The destination name for this file on your local system
`directory`	The destination directory. It is advisable to keep this as your working directory for GhcnDaily

Details

Downloads a table of FIPS codes

Value

returns a handle to the local file

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt

Examples

```
## Not run:
  # maybe we should just build in the fips code



## End(Not run)
```

`downloadDailyData` *Downloads Daily temperature data*

Description

The .dly files on the ftp server contain more than temperature data. This function will process those files directly and only download the TMAX and TMIN data out of the files. It is considerable faster than downloading the entire file. These files are given a .raw extension and written to the destination directory

Usage

```
downloadDailyData(urlList, directory = DAILY.DATA.DIRECTORY)
```

Arguments

`urlList`	A list of urls to download. This list is created by the function `makeDownloadList`
`directory`	the destination directory. Other functions depend on finding the .raw files in this directory.

Details

The function makes a connection to the server and reads the files on the server and downloads only the TMIN an TMAX data. You should build a download list that conatins all the stations you want to download. If the process is interupted, merely create a new download list with the missing files.

Value

The side effect is files written to the local machine

Author(s)

Steven Mosher

See Also

codelinkdownloadDailyFiles

Examples

```
## Not run:
  # sample code

## End(Not run)
```

`downloadDailyFiles` *Downloads the GHCN daily files*

Description

Every station in GHCN Daily has a file of data. There are over 77000 files on the ftp. To initiate a download you need a list of the Urls and you specify a download directory. The files in the url list are downloaded to your local machine. The list of files to download is created by the function `makeDownloadList`. This function downloads the entire file (.dly). Files contain more than temperature data.

Usage

```
downloadDailyFiles(urlList, directory = DAILY.FILES.DIRECTORY)
```

Arguments

`urlList`	A vector of fully qualified urls that point to individual files. This list is created by `makeDownloadList`
`directory`	The default directory for storing local files. Note that this directory has .dly files while the data directory has .dat files

Details

The download list contains the urls and those urls are used to call the `download.file` function. If the download should fail or generate warnings for empty files Then rebuild the the urlList and resubmit the request. (a function for doing this will be added in a future release.)

Value

The side effect is local files created

Author(s)

Steven Mosher

See Also

`downloadDailyData`

Examples

```
## Not run:
 # sample code


## End(Not run)
```

`downloadDailyInventory`
Downloads an Inventory

Description

Downloads an inventory of stations for daily GHCN data. The inventory of stations is not a typical inventory. It contains metadata about the temperatures files themselves. Station "metadata" is provided in a different file. The inventory contains information about the first and last years of reporting and the "element reported". Some stations only report Tmin, others Tmax, and most report both.

Usage

```
downloadDailyInventory(url = GHCN.DAILY.INVENTORY.URL,
                dest = "DailyInv.txt", directory = getwd())
```

Arguments

`url`	The url where the daily data is maintained by NCDC
`dest`	The destination name for the inventory data
`directory`	Destination directory

Details

The relevant portion of the NCDC readme is repeated below:

ID: is the station identification code. Please see "ghcnd-stations.txt" for a complete list of stations and their metadata. LATITU

Value

returns a handle to the file

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
## Not run:
# sample code

## End(Not run)
```

`downloadDailyMetadata` *Download station data metadata*

Description

The station metadata consists of station location information, name, state, country, and various identifiers for other systems, such as WMO.

Usage

```
downloadDailyMetadata(url = GHCN.DAILY.METADATA.URL,
                      dest = "DailyMetadata.txt", directory = getwd())
```

Arguments

`url`	The url at NCDC where the data file resides
`dest`	The destination file name on you local machine
`directory`	The destination directory on your local machine

Details

The relevant portion of the readme are copied below ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

ID:is the station identification code. Note that the first two characters denote the FIPS country code, the third character is a net

Value

handle to the file

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt

Examples

```
## Not run:
```

```
  #write some stuff

## End(Not run)
```

GHCN.DAILY.COUNTRY.URL *Url of the country FIPS data*

Description

Url to the country FIPS codes

Usage

```
GHCN.DAILY.COUNTRY.URL
```

Format

The format is: chr "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt"

Details

current URL to the NCDC file of FIPS codes

Source

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt

Examples

```
print(GHCN.DAILY.COUNTRY.URL)
```

GHCN.DAILY.DATA.URL *Url for data*

Description

The base url that is used to generate the url list. All stations have their own seperate file. The files are named using the station ID. in order to download them all a 'url list' is created. that list uses this url as a "base" to build the complete url to the file, once the files have been chosen

Usage

```
GHCN.DAILY.DATA.URL
```

Format

The format is: chr "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/all/"

Details

The is the base url. All files under this directory have a name that is the station ID with a .dly suffix

Source

The sources for that data are covered in the NCDC readme and in the file itself

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
print(GHCN.DAILY.DATA.URL)
```

GHCN.DAILY.INVENTORY.URL *url for inventory*

Description

the url for the file of station inventories

Usage

```
GHCN.DAILY.INVENTORY.URL
```

Format

The format is: chr "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-inventory.txt"

Details

This url is used for downloading the file

Source

The source is NCDC see the readme below

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
print(GHCN.DAILY.INVENTORY.URL)
```

GHCN.DAILY.METADATA.URL

url to station metadata

Description

url to station metadata on the NCDCftp site. This file contains station location, name and identifier information

Usage

```
GHCN.DAILY.METADATA.URL
```

Format

The format is: chr "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt"

Source

NCDC is the source

References

urlftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
print(GHCN.DAILY.METADATA.URL)
```

makeDownloadList *Creates a list of urls to feed to the download function*

Description

This function takes an inventory as an input and uses the ID and the base URL to create a list of urls to download. It is supplied with an object that has station IDs. Those Ids are used in conjunction with the base url to create a list of urls to individual files on the ftp.

Usage

```
makeDownloadList(Inventory, baseUrl = GHCN.DAILY.DATA.URL)
```

Arguments

Inventory	The inventory you want to fetch
baseUrl	The base url that final urls are constructed from

Value

returns a vector of urls that you can iterate over. Simply, you get a vector of url strings that can then be passed to a download function to download the file.

Author(s)

Steven Mosher

Examples

```
## Not run:
   #add some sample code

## End(Not run)
```

meanFunction *A specialize function for calculating a mean*

Description

The mean function in R has limited options in calculating a mean in the presence of NA values. It is all or nothing. NA can either be completely ignored or NA is returned if any value is NA. This function provides the flexibility to specify the number of NAs on a monthly basis. This function is used internally by other functions. Source is provided to illustrate how a user can define their own function

Usage

```
meanFunction(data, dayMiss = 3)
```

Arguments

data	this data is a single line or month of daily data. It it contains 33 data elements. Year, Month and 31 values
dayMiss	The number of days that can be missing before an NA is returned. 3 days missing is set as the default. This can be changed by setting parameters in createStationMonthly

Details

The function calculates the number of reports missing. If the days missing exceeds the threshold (daysMiss) then NA is returned to the caller. Otherwise mean is called with na.rm = TRUE

Value

Return NA if the number of missing days exceeds daysMiss otherwise it returns the mean

Author(s)

Steven Mosher

See Also

createStationMonthly

Examples

```
## Not run:
  # sample

## End(Not run)
```

---

mergeInventory *A function to merge a Tmax inventory and Tmin Inventory*

Description

The Inventory file conatins records for stations acoording to the element they conatin. Most stations report both, some report only one. The time periods can also be different. This function takes two inventories and merges them to produce a consolidated record. The element variable is written to indicate if both are present and the smallest start year and largest end year are recorded

Usage

```
mergeInventory(Inventory1, Inventory2)
```

Arguments

Inventory1	An inventory with one element, for example TMAX
Inventory2	An Inventory with one element, for example TMIN

Details

The inventories are merged on the ID so that the result is the union of both. The elements are merged into a single field for example TMAX/TMIN or TMAX/NA if Tmin is missing for that station. The years are changed to reflect the earliest start year and the latest end year.

Value

Returns a merged Inventory

Author(s)

Steven Mosher

MONTHLY.DATA.DIRECTORY

A directory for monthly data

Description

Monthly data is complied into datasets that contain all stations. Both `writeNetCDF` and `writeGhcn` write data to this directory

Usage

```
MONTHLY.DATA.DIRECTORY
```

Format

The format is: chr "MonthlyData"

Details

A directory for Monthly datasets

Examples

```
print(MONTHLY.DATA.DIRECTORY)
```

readCountry *read country Fips codes*

Description

reads the FIPS codes

Usage

```
readCountry(filename = "countryCodes.txt")
```

Arguments

filename	Name of the file

Value

Returns a dataframe

Author(s)

Steven Mosher

References

insert

Examples

```
 ## Not run:
 #sample

## End(Not run)
```

readDailyData *A function to read Daily .dat files*

Description

Daily .dat files contain both TMIN and TMAX data elements This function reads an specified element and returns that data in a format expected by other functions such as `createMonthlyStation`

Usage

```
readDailyData(filename, element = "TMAX",
                      directory = DAILY.QA.DIRECTORY)
```

Arguments

filename	Filename to be read. do not include the directory as a part of the filename
element	character "TMAX" to return maximum temperatures "TMIN" for minimum temperatures
directory	The directory where the .dat files are located.

Details

The files are read and a matrix is returned with stationId in the rownames and Year followed by 12 columns of temperature data

Value

Returns a matrix of temperature data for futher processing.

Author(s)

Steven Mosher

Examples

```
 ## Not run:
 # sample code

## End(Not run)
```

---

readDailyFile *Reads a daily (.dly) file*

Description

This function read a .dly file and returns the TMIN and TMAX elements only.

Usage

```
readDailyFile(filename, directory = DAILY.FILES.DIRECTORY)
```

Arguments

filename	The filename (station ID) to be read
directory	The directory where the file is located

Details

This function reads a .dly file and extracts TMIN and TMAX and returns a dataframe for the station including the Id, Year, element and 31 columns of data for the days of the month and returns the 31 columns for QA flags

Value

A data.frame of data for the file.

Author(s)

Steven Mosher

See Also

readDailyData

Examples

```
## Not run:
#sample code

## End(Not run)
```

`readDailyInventory` *Read daily inventory*

Description

The inventory contains the station Id, lat and lon and metadata about the file, as opposed to the station. That means it conatins the "elements" of a station (Tmax,Tmin etc) as well as the years the station reported those elements.

Usage

```
readDailyInventory(filename = "DailyInv.txt", elements = NULL)
```

Arguments

`filename`	Filename for the inventory.
`elements`	If elements is set to NULL then all the stations will be returned. There are 31 different elements. If "TMAX" is supplied to this parameter only those stations with Tmax will return. You can supply a vector c("TMAX","TMIN"). Ordinarily you make one call for "TMIN" and another call for "TMAX" and then use `mergeInventory`

Details

The Inventories have elements as discussed in the NCDC readme. The function will read the entire list of stations and return those that satisfy the "elements" parameter: elements include

Value

returns an inventory of stations with the specified elements. elements are specified in section III of the readme. Copied below: PRCP = Precipitation (tenths of mm) SNOW = Snowfall (mm) SNWD = Snow depth (mm) TMAX = Maximum temperature (tenths of degrees C) TMIN = Minimum temperature (tenths of degrees C) ACMC = Average cloudiness midnight to midnight from 30-second ceilometer data (percent) ACMH = Average cloudiness midnight to midnight from manual observations (percent) ACSC = Average cloudiness sunrise to sunset from 30-second ceilometer data (percent) ACSH = Average cloudiness sunrise to sunset from manual observations (percent) AWND = Average daily wind speed (tenths of meters per second) DAEV = Number of days included in the multiday evaporation total (MDEV) DAPR = Number of days included in the multiday precipiation total (MDPR) DASF = Number of days included in the multiday snowfall total (MDSF) DATN = Number of days included in the multiday minimum temperature (MDTN) DATX = Number of days included in the multiday maximum temperature (MDTX) DAWM = Number of days included in the multiday wind movement (MDWM) DWPR = Number of days with non-zero precipitation included in multiday precipitation total (MDPR) EVAP = Evaporation of water from evaporation pan (tenths of mm) FMTM = Time of fastest mile or fastest 1-minute wind (hours and minutes, i.e., HHMM) FRGB = Base of frozen ground layer (cm) FRGT = Top of frozen ground layer (cm) FRTH = Thickness of frozen ground layer (cm) GAHT = Difference between river and gauge height (cm) MDEV = Multiday evaporation total (tenths of mm; use with DAEV) MDPR = Multiday precipitation total (tenths of mm; use with DAPR and DWPR, if available) MDSF = Multiday snowfall total MDTN = Multiday minimum temperature (tenths of degrees C; use with DATN) MDTX = Multiday maximum temperature (tenths of degress C; use with DATX) MDWM = Multiday wind movement (km) MNPN = Daily minimum temperature of water in an evaporation pan (tenths of degrees C) MXPN = Daily maximum temperature of water in an evaporation pan (tenths of degrees C) PGTM = Peak gust time (hours and minutes, i.e., HHMM) PSUN = Daily percent of possible sunshine (percent) SN*# = Minimum soil temperature (tenths of degrees C) where * corresponds to a code for ground cover and # corresponds to a code for soil depth.

SX*# = Maximum soil temperature (tenths of degrees C) where * corresponds to a code for ground cover and # corresponds to a code for soil depth. See SN*# for ground cover and depth codes.

THIC = Thickness of ice on water (tenths of mm) TOBS = Temperature at the time of observation (tenths of degrees C) TSUN = Daily total sunshine (minutes) WDF1 = Direction of fastest 1-minute wind (degrees) WDF2 = Direction of fastest 2-minute wind (degrees) WDF5 = Direction of fastest 5-second wind (degrees) WDFG = Direction of peak wind gust (degrees) WDFI = Direction of highest instantaneous wind (degrees) WDFM = Fastest mile wind direction (degrees) WDMV = 24-hour wind movement (km) WESD = Water equivalent of snow on the ground (tenths of mm) WESF = Water equivalent of snowfall (tenths of mm) WSF1 = Fastest 1-minute wind speed (tenths of meters per second) WSF2 = Fastest 2-minute wind speed (tenths of meters per second) WSF5 = Fastest 5-second wind speed (tenths of meters per second) WSFG = Peak guest wind speed (tenths of meters per second) WSFI = Highest instantaneous wind speed (tenths of meters per second) WSFM = Fastest mile wind speed (tenths of meters per second) WT** = Weather Type where ** has one of the following values: WV** = Weather in the Vicinity where ** has one of the following values:

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt data read includes:

ID: is the station identification code. Please see "ghcnd-stations.txt" for a complete list of stations and their metadata. LATITU

Examples

```
## Not run:
 x<-readDailyInventory(elements = "TMAX")


## End(Not run)
```

`readDailyMetadata` *reads the metadata for the daily stations*

Description

This function read the metadata for the station. This is different than the inventory metadata which reads the metadata about the files. This is station metadata: location, elevation, name, country and various identifier codes

Usage

```
readDailyMetadata(filename = "DailyMetadata.txt")
```

Arguments

`filename`	The filename you want to read

Details

The folloing data is read and returned ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

ID:is the station identification code. Note that the first two characters denote the FIPS country code, the third character is a net

Value

returns a data frame of station metadata as described above

Author(s)

Steven Mosher

References

ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt

Examples

```
## Not run:

   # I can probably do a test file in the external directory

## End(Not run)
```

`readNetCDF` *Reads a NetCDF file of temperatures*

Description

Files written by `writeNetCDF` can be read with this function. This merely wraps common `ncdf` functions for users who are not aquainted with the package. It also insures that the data structure (a 3D array) is properly named for use with other packages. Dimnames are written for the structure and station Ids are properly applied

Usage

```
readNetCDF(filename, directory = MONTHLY.DATA.DIRECTORY)
```

Arguments

`filename`	The filename. These have a .nc extension
`directory`	The directory where monthly data is written

Details

The NetCdf file conatins all the information required to build a 3D array of temperatures. The first dimension is stations, the second is months, and the third is years. the Dimnames are all written to the object before it is returned.

Value

A 3D array of temperatures. The attribute (TMAX or TMIN) can be read from the NetCDF file (global attributes. It is not associated with the data structure.

Author(s)

Steven Mosher

See Also

writeNetCDF

Examples

```
 ## Not run:
 #sample

## End(Not run)
```

---

writeGhcn *Writes a file in GHCNV3 format*

Description

GHCNV3 format has muliple lines of data for a given station. Each line conatins an Id, element, Year and 12 data elements for temperatures. The format here is not an exactly mimic of GhcnV3 format and can be read with a simnple `table.read` in R. However, the data elements are the same

Usage

```
writeGhcn(Temps, element, filename,
                 directory = MONTHLY.DATA.DIRECTORY)
```

Arguments

`Temps`	A 3D array of temperatures
`element`	The element "TMAX" or "TMIN". character string
`filename`	The base filename. The base file name is used to create a complete file name including the element, the time and a marker for the ghcn format. The extension is .dat
`directory`	The directory

Details

The 3D array is reformated into a ghcnV3 like format 15 columns of data: Id, Element, Year and 12 temperatures

Value

Side effect is a file being written

Author(s)

Steven Mosher

Examples

```
 ## Not run:
 #sample code

## End(Not run)
```

`writeNetCDF` *Writes a NetCDF file of a 3D data array*

Description

This function writes a NetCDF file of the 3D data array create by the function `createArray`. The file contains all the dimenional data, the temperatures, and the dimension names for the variables. In addition the station Ids are written to a variable.

Usage

```
writeNetCDF(Temps, element, filename,
                  directory = MONTHLY.DATA.DIRECTORY)
```

Arguments

`Temps`	A 3D array of temperatures
`element`	The element (TMAX or TMIN) being written. This should be supplied as a character string
`filename`	the base file name without extension. The file name is used to create a filename that includes the base name, the element, the data and the .nc extension
`directory`	The directory to write the file to

Details

The function defines and fills a NetCDF file of temperature and station names. A global attribute is set using the element

Value

Side effect is a file being written

Author(s)

Steven Mosher

See Also

codewriteNetCDF

Examples

```
## Not run:
 #sample code

## End(Not run)
```

Index