Package: raadfiles (via r-universe)

September 12, 2024

Title File Database Management for 'raadtools'

Version 0.1.4.9009

Description Tools for managing collections of files for the 'raad' family.

Depends R (>= 3.3.0)

Imports digest, fs (>= 1.6.4.9000), rlang, stringr, dplyr, memoise, tibble, vroom

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Suggests fastrls, testthat

Remotes r-lib/fs, AustralianAntarcticDivision/fastrls

Repository https://scar.r-universe.dev

RemoteUrl https://github.com/AustralianAntarcticDivision/raadfiles

RemoteRef HEAD

RemoteSha 458c3893bd623451c264d9086208a53ea6a93a05

Contents

`altimetry` *Altimetry products*

Description

Sea Surface Height measured by Altimetry and derived variables. SSALTO/DUACS Near-Real-Time Level-4 sea surface height and derived variables measured by multi-satellite altimetry observations over Global Ocean.

Usage

```
altimetry_daily_files(all = FALSE)
```

Arguments

`all`	return all files or only the final processing (NRT is included either way)

Details

In 2018/2019 the file servers migrated to 'my.cmems-du.au' and 'nrt.cmems-du.eu' (NRT) from 'ftp.sltac.cls.fr', but the files and file name scheme remained unchanged so no net effect (so far that we are aware of).

There are NRT (near-real-time) and final processed files, identifiable from the root domain and in the filename '^nrt_'. Both are returned by this function. The 'all' argument can be set to 'TRUE' to include all NRT files, which has overlapping processing dates that are ultimately consolidated into the daily sequence.

Value

tibble data frame of file names, data 'date', and 'processing_date'

Examples

```
## Not run:
  altimetry_daily_files()

## End(Not run)
```

`altimetry_antarctica_files` *Antarctic Polar Altimetry files*

Description

Sea Level Anomaly measured by Altimetry and derived variables

Usage

```
altimetry_antarctica_files()
```

Details

sla, formal_error, U, V

Value

data frame of file paths

Examples

```
## Not run:
  aaf <- altimetry_antarctica_files()

## End(Not run)
```

`altimetry_currents_polar_files` *Derived altimetry products*

Description

A polar-transformed copy of the 'u' and 'v' components of surface currents from [altimetry_daily_files]. Only available for the southern hemisphere.

Usage

```
altimetry_currents_polar_files(hemisphere = "south")
```

Arguments

hemisphere	south only for now

Details

The code that creates these derived files is at [raad-deriv](https://github.com/AustralianAntarcticDivision/raad-deriv).

Examples

```
## Not run:
  altimetry_currents_polar_files()

## End(Not run)
```

---

amps_files *AMPS files*

Description

Antarctic Mesoscale Prediction System GRIB files.

Usage

```
amps_files()

amps_model_files(time.resolution = "4hourly", grid = "d1", ...)

amps_d1files(time.resolution = "4hourly", ...)

amps_d2files(time.resolution = "4hourly", ...)
```

Arguments

time.resolution	a placeholder, defaults to "4hourly" and remains unused
grid	one of ‘d1‘ (30km resolution) or ‘d2‘ (10km resolution)
...	reserved, unused

Details

‘amps_files‘ returns all the files, ‘amps_model_files‘ returns the files with date set from the file name, ‘amps_d1files‘ and ‘amps_d2files‘ return only the 30km and 10 km resolution grids respectively.

Examples

```
## Not run:
  amps_files()
  amps_model_files()
  amps_d1files()
  amps_d2files()

## End(Not run)
```

`amsr_daily_files` *AMSR daily sea-ice concentration*

Description

Sea ice concentration files at 6.25 km resolution, southern hemisphere.

Usage

```
amsr2_3k_daily_files(type = c("tif", "hdf"))

amsre_daily_files()

amsr2_daily_files()

amsr_daily_files()
```

Arguments

`type`	tif or hdf

Details

'amsre_daily_files()' returns HDF files

'amsr2_daily_files()' returns TIF files

'amsr2_3k_daily_files()' returns TIF files

'amsr_daily_files()' returns HDF files

The HDF files require flipping about the y-axis, and setting the polar extent and the crs (projection) metadata. The TIF files don't require this, they are completely oriented and metadata-complete.

Value

tibble data frame of file names

Examples

```
## Not run:
  ## this combines amsr2 (2012-) and amsre (2002-2011)
  amsr_daily_files()

## End(Not run)
```

`argo_files` *Argo files.*

Description

ARGO, by default we have 'profiles', alternatively 'meta' for type

Usage

```
argo_files(type = c("prof", "meta", "traj", "tech", "Mprof"), dac = NULL)
```

Arguments

type	file type, i.e. prof, meta, traj, tech
dac	data acqusition centre e.g.* "aoml", "bodc", "coriolis", "csio", "csiro", "incois", all returned if not specified

Details

(No traj, tech, or Mprof at this time of writing 2022-11-21)

`cafe_monthly_files` *'Cafe' MODIS files*

Description

'Cafe' MODIS files

Usage

```
cafe_monthly_files()
```

Value

data frame of file names and date, 'date', 'fullname'

Examples

```
## Not run:
  cafe_monthly_files()

## End(Not run)
```

ccmp *Derived surface files from Remote Sensing Systems (RSS)*

Description

RSS CCMP_RT V2.1 derived surface winds (Level 3.0), variables 'uwnd', 'vwnd', 'nobs' 'u-wind vector component at 10 meters', 'v-wind vector component at 10 meters', and 'number of observations used to derive wind vector components' from [Remote Sensing Systems](http://www.remss.com/).

Usage

```
ccmp_6hourly_files()
```

Details

Each file contains four time steps at six hourly intervals aligned to the file base date.

Value

tibble data frame of file names, with columns 'fullname' and 'date'

References

"Mears et al., Journal of Geophysical Research: Oceans,124, 6997-7010, 2019, Hoffman et al., Journal of Atmospheric and Oceanic Technology, 2013; Atlas et al., BAMS, 2011; Atlas et al., BAMS, 1996". "Available for public use with proper citation".

Examples

```
## Not run:
  ccmp_6hourly_files()

## End(Not run)
```

cersat_daily_files *CERSAT daily sea-ice concentration*

Description

Sea ice concentration files at 12.5 km resolution, southern hemisphere.

Usage

```
cersat_daily_files()
```

Value

tibble data frame of file names

Examples

```
## Not run:
  cersat_daily_files()

## End(Not run)
```

`fasticefiles` *fast ice files*

Description

Data frame of all available fast ice files.

Usage

```
fasticefiles(
  product = c("circum_fast_ice", "binary_fast_ice"),
  mask = FALSE,
  ...
)
```

Arguments

`product`	which product
`mask`	if TRUE return mask file name
`...`	reserved for future use, currently ignored

Details

A data frame with file, date, fullname

Note that this product changed from the legacy 2000-2008 initial East Antarctic product "binary_fast_ice" to the circumpolar update "circum_fast_ice" 2000-2018 in Feburary 2021.

If you want the old files, use ‘product = "binary_fast_ice"‘, but it’s safe to assume the default product supersedes the old one.

The initial product was in Cylindrical Equal Area projection, while the circumpolar product uses the NSIDC-compatible polar stereographic (but with an unspecified extent, though implicit in the longitude and latitude arrays of the NetCDF files).

Exists in ’public.services.aad.gov.au/datasets/science’ (Feb 2021).

Value

data frame

References

Fraser, A. D., Massom, R. A., Ohshima, K. I., Willmes, S., Kappes, P. J., Cartwright, J., and Porter-Smith, R.: High-resolution mapping of circum-Antarctic landfast sea ice distribution, 2000–2018, Earth Syst. Sci. Data, 12, 2987–2999, https://doi.org/10.5194/essd-12-2987-2020, 2020.

[Fraser et al. 2018](https://doi.org/10.5194/essd-12-2987-2020)

`fsle_files` *Backward-in-time Finite-Size Lyapunov Exponents.*

Description

FSLE - MAPS OF FINITE SIZE LYAPUNOV EXPONENTS AND ORIENTATIONS OF THE ASSOCIATED EIGENVECTORS

Usage

```
fsle_files()
```

Details

These are daily files.

References

[https://www.aviso.altimetry.fr/en/data/products/value-added-products/fsle-finite-size-lyapunov-exponents.html](Finite-Size Lyapunov Exponents)

`geoid_files` *Earth Gravitation Model files*

Description

Global 2.5 Minute Geoid Undulations, a

Usage

```
geoid_files(all = FALSE, ...)
```

Arguments

`all`	return all files, or just the core grid files for GDAL? './w001001.adf'
`...`	: additional parameters, currently ignored

Details

Each file is an ESRI GRID raster data set of 2.5-minute geoid undulation values covering a 45 x 45 degree area. Each raster file has a 2.5-minute cell size and is a subset of the global 2.5 x 2.5-minute grid of pre-computed geoid undulation point values found on the EGM2008-WGS 84 Version web page. This ESRI GRID format represents a continuous surface of geoid undulation values where each 2.5-minute raster cell derives its value from the original pre-computed geoid undulation point value located at the SW corner of each cell.

Value

data frame of file paths

Examples

```
## Not run:
  geoid_files()

## End(Not run)
```

---

ghrsst *GHRSST files*

Description

The Group for High Resolution Sea Surface Temperature (GHRSST) files.

Usage

```
ghrsst_daily_files()
```

Value

tibble data frame of file names

Examples

```
## Not run:
  ghrsst_daily_files()

## End(Not run)
```

leads *Files containing relative lead frequencies for the Arctic and Antarctic*

Description

Average Lead-Frequency for the polar oceans for winter months November-April 2002/03-2018/19 based on daily lead composites as derived from MOD/MYD-29 IST 5 min granules.

Usage

```
iceclim_south_leadsfiles(all = FALSE)

iceclim_north_leadsfiles(all = FALSE)
```

Arguments

all	return all files, or just the core grid files (*.nc)?

References

F. Reiser, S. Willmes, G. Heinemann (2020): A new algorithm for daily sea ice lead identification in the Arctic and Antarctic winter from thermal-infrared satellite imagery.

Examples

```
## Not run:
  iceclim_south_leadsfiles()
  iceclim_north_leadsfiles()

## End(Not run)
```

ncep2_files *NCEP2 wind files*

Description

NCEP2 six-hourly reanalysis2 gaussian grid

Usage

```
ncep2_uwnd_6hr_files()

ncep2_vwnd_6hr_files()
```

Value

tibble data frame of file names

Examples

```
## Not run:
  ncep2_uwnd_6hr_files()
  ncep2_vwnd_6hr_files()

## End(Not run)
```

---

nsidc *NSIDC daily and monthly sea-ice concentration*

Description

Sea ice concentration files.

Usage

```
nsidc_south_monthly_files()

nsidc_north_monthly_files()

nsidc_monthly_files()

nsidc_south_daily_files()

nsidc_north_daily_files()

nsidc_daily_files()

nsidc_daily_files_v2(extra_pattern = NULL)

nsidc_monthly_files_v2(extra_pattern = NULL)
```

Arguments

extra_pattern	argument for restricted string matching

Value

tibble data frame of file names

Examples

```
## Not run:
  nsidc_south_monthly_files()
  nsidc_north_monthly_files()
  nsidc_monthly_files()
  nsidc_south_daily_files()
  nsidc_north_daily_files()
```

```
  nsidc_daily_files()

## End(Not run)
```

`oisst` *OISST v2 files*

Description

Optimally Interpolated Sea Surface Temperature, from https://www.ncei.noaa.gov/. These files contain four variables ‘sst‘, ‘anom‘, ‘err‘ and ‘ice‘ for sea surface temperature, sst anomaly, sst error and sea ice concentration on a regular global longitude latitude grid, with dimensions 1440x720 grid (0.25 degree spatial resolution).

Usage

```
oisst_daily_files()

oisst_monthly_files()
```

Details

At the time of writing (2021-01-18) the files are accessible at https://www.ncei.noaa.gov/data/sea-surface-temperature-optimum-interpolation/v2.1/access/avhrr/. See the [blueant](https://github.com/ package for a convenient way to obtain this data set named "NOAA OI 1/4 Degree Daily SST AVHRR".

These files can be accessed individually ‘raster‘ package function ‘raster‘ or as multiple layers with ‘brick‘ or ‘raster::stack‘. Use the ‘varname‘ argument to choose one of the four variables.

To obtain full NetCDF header metadata use ’ncdf4::open.nc(file)’ or ’RNetCDF::print.nc(RNetCDF::open.nc(file))’ to see the equivalent of ’ncdump -h’ output.

Optimally Interpolated version 2 SST moved from ’eclipse.ncdc.noaa.gov’, to ’www.ncei.noaa.gov’ at the end of 2017. Version 2 was superseded by version 2.1 during 2020.

Value

tibble data frame of file names, with columns ‘fullname‘ and ‘date‘

Examples

```
## Not run:
  oisst_daily_files()

## End(Not run)
```

par — *Each file contains four time steps at six hourly intervals aligned to the file base date.*

Description

Each file contains four time steps at six hourly intervals aligned to the file base date.

Usage

```
par_files(time.resolution = "8D")
```

Arguments

time.resolution	time resolution (only "8D" is available)

Value

tibble data frame of file names, with columns ‘fullname‘ and ‘date‘

References

oceandata.sci.gsfc.nasa.gov/MODISA/Mapped/8Day/4km/par

Examples

```
par_files()
```

polarview — *Polarview files*

Description

Imagery from www.polarview.aq

Usage

```
polarview_files(type = c("jpeg", "tarball"))
```

Arguments

type	jpeg or tarball

Details

The JPEGs are simple images, the GeoTIFFs are 16-bit integers (haven’t explored further)

Value

tibble data frame of file names, with columns ‘fullname‘ and ‘date‘

Examples

```
## Not run:
  files <- polarview_files()
  tiffiles <- polarview_files(type = "tarball")

## End(Not run)
```

---

raadfiles-admin *Raadfiles administration tools*

Description

Administration tools for managing a data library.

Usage

```
get_raad_data_roots()

get_raad_filenames(all = FALSE)

set_raad_data_roots(
  ...,
  replace_existing = TRUE,
  use_known_candidates = FALSE,
  verbose = TRUE
)

raad_filedb_path(...)

set_raad_filenames(clobber = FALSE)

run_build_raad_cache()
```

Arguments

all	if ‘TRUE‘ include ’data_deprecated’, expert-use only
...	input file paths to set
replace_existing	replace existing paths, defaults to TRUE
use_known_candidates	apply internal logic for known candidates (for internal use at raad-hq), defaults to FALSE
verbose	issue warnings?
clobber	by default do not ignore existing file cache, set to TRUE to ignore and set

Details

These management functions are aimed at raadtools users, but can be used for any file collection. The administration tools consist of **data roots** and control over the building, reading, and caching of the available file list. No interpretation of the underlying files is provided in the administration tools.

A typical user won't use these functions but may want to investigate the contents of the raw file list, with 'get_raad_filenames()'.

A user setting up a raadfiles collection will typically set the root directory/directories with 'set_raad_data_roots()', then run the file cache list builder with 'run_build_raad_cache()', and then 'set_raad_filenames()' to actually load the file cache into memory.

In a new R session there is no need to run 'set_raad_filenames()' directly as this will be done as the package loads. To disable this automatic behaviour use 'options(raadfiles.file.cache.disable = TRUE)' *before* the package is used or loaded. This is typically done when calling 'run_build_raad_cache()' in a cron task.

Every raadfiles file collection function (e.g. 'oisst_daily_files') will run 'get_raad_filenames' to obtain the full raw list of available files from the global in-memory option 'getOption("raadfiles.env")$raadfiles.filename.databas
and there is a low threshold probability that this will also trigger a re-read of the file listing from the root directories. To avoid this trigger either use that directly directly to get the in-memory file list, or set 'options(raadfiles.file.refresh.threshold = 0)' to prevent the trigger. (Set it to 1 to force it always to be read, also controlled by 'set_raad_filenames(clobber = TRUE)').

There is a family of functions and global options used for administration.

Administration functions

`set_raad_data_roots`	set data root paths, for normal use only one data root is needed
`set_raad_filenames`	runs the system to update the file listing and refresh it
`get_raad_data_roots`	returns the current list of visible root directories
`get_raad_filenames`	returns the entire list of all files found in visible root directories
`run_build_raad_cache`	scan all root directories and update the file listing in each

Options for use by administrators

`raadfiles.data.roots`	the list of paths to root directories
`raadfiles.file.cache.disable`	disable on-load setting of the in-memory file cache (never set automatically by the p
`raadfiles.file.refresh.threshold`	threshold probability of how often to refresh in-memory file cache (0 = never, 1 = e

Internal options, used by the package

Options used internally, and subject to control by adminstrator options and the running of admin functions (they may not be set).

`raadfiles.env`	an environment with the data frame of all file names from the data roots in a object named 'r
`raadfiles.database.status`	a status record of the in-memory filename database (timestamp)

`rema_8m_files` *Files for The Reference Elevation Model of Antarctica (REMA)*

Description

Return files for various products from REMA Release 1

Usage

```
rema_tile_files(all = FALSE, ...)

rema_100m_files(...)

rema_200m_files(filled = TRUE, ...)

rema_1km_files(filled = TRUE, ...)

rema_8m_files(...)

rema_8m_tiles()

rema_200m_dem_files()

.rema_file_filter(x)

rema_200m_dem_geoid_files()

rema_200m_slope_files()

rema_200m_aspect_files()

rema_200m_rugosity_files()

rema_200m_rock_files()

rema_100m_dem_files()

rema_100m_dem_geoid_files()
```

```
rema_100m_slope_files()

rema_100m_aspect_files()

rema_100m_rugosity_files()

rema_100m_rock_files()

rema_8m_dem_files()

rema_8m_dem_geoid_files()

rema_8m_slope_files()

rema_8m_aspect_files()

rema_8m_rugosity_files()

rema_8m_rock_files()
```

Arguments

all	for rema_tile_files, return all or just *.shp files; for other functions, if 'TRUE' include 'data_deprecated', expert-use only
...	additional parameters, currently ignored
filled	return 'filled' variant if available
x	pattern to detect

Details

'rema_8m_files' returns the base level 8 GeoTIFF files, there are 1516 files at 8m resolution.

Value

data frame of file names

References

https://www.pgc.umn.edu/tag/rema/

Examples

```
## Not run:
  rema_8m_files()
  rema_100m_files(filled = TRUE)

## End(Not run)
```

seapodym_weekly_files *SEAPODYM model files*

Description

Global ocean low and mid trophic levels biomass hindcast

Usage

```
seapodym_weekly_files()
```

References

http://www.cls.fr, http://www.seapodym.eu

smap *SMAP ocean surface salinity files*

Description

Remote Sensing Systems SMAP Level 3 Sea Surface Salinity Standard Mapped Image 8day running

Usage

```
smap_8day_files()
```

Value

tibble data frame of file names, with columns ‘fullname‘ and ‘date‘

Examples

```
## Not run:
  smap_daily_files()

## End(Not run)
```

sose_monthly_files *Southern Ocean State Estimate files*

Description

Files from Ocean State Estimation at Scripps for the Southern Ocean 'SOSE'.

Usage

```
sose_monthly_files(varname = "", iteration = "")
```

Arguments

varname	default is '' which is the first available, set to 'all' to return all file names without date expansion
iteration	default is '' which finds latest available, see details

Details

Iteration provided is latest available, otherwise this argument will be used to match with file names.

Dates in the files are extracted and expanded out for every time step, it's assumed this will be used in raadtools along with a 'level' argument to return a time step or time series.

Value

data frame of file names and date, 'date', 'fullname'

References

http://sose.ucsd.edu/

Examples

```
sose_monthly_files()
```

srtm *SRTM files*

Description

SRTM 90m Digital Elevation Database v4.1

Usage

```
srtm_files()
```

Details

DOI: 0.1080/13658810601169899

Value

tibble data frame of file names, with columns ‘fullname‘, ‘x‘, ‘y‘ (tiles)

data frame with ‘fullname‘ file path, ‘x‘, ‘y‘ tile column and row indices, ‘root‘ the data file root path

References

[https://cgiarcsi.community/data/srtm-90m-digital-elevation-database-v4-1/]

Examples

```
## Not run:
  srtm_files()

## End(Not run)
```

---

thelist *TheList*

Description

Local authority spatial data in Tasmania.

Usage

```
thelist_files(
  format = c("gdb", "tab", "shp", "asc", "xml", "lyr", "dbf", "zip", "all"),
  pattern = NULL
)
```

Arguments

format	is used to targe tspecific formats see Details
pattern	is used to string match generally, if this is not NULL then format is ignored

Details

TheList.tas.gov.au is the standard local mapping authority in Tasmania, it was recently upgraded and works very well, and most of the data including newish LiDAR is available within the opendata tree. Also check out tasmap.org for an alternative interface with access to the same services.

These files are broken into sub-regions, administrative areas within the state of Tasmania. At time of checking there were 19 sub-regions, and 544 or so layers (type within format) and 37,616 total files. GDB detection is different to the other more definite formats so the file sets won't be analogous atm.

There are Climate Futures Australia (CFA) layer indexes in here as well, it's on the todo list to build a comprehensive index (or access one).

The scheme uses the Map Grid of Australia 1994 (MGA94) on the Geocentric Datum of Australia 1994 (GDA94), an implementation of UTM Zone 55. GDA94 was rolled out in Australia in the early 2000s, Tasmania kept the old UTM scheme (it was AMG66, AGD66) but around the same time Victoria used the opportunity to move to a single-projection for the entire state, to avoid having to switch between zones. NSW took much longer to modernize and standardize around GDA94 and they stumbled forward with their three UTM zones (54, 55, 56), and while Tasmania did it quickly we only have the one zone (no one thought much about Macquarie Island) and Victoria did it more cleverly. I'm not sure how Queensland went, they were adding properties and roads at a very scary rate so probably took much longer than us. The software back then could only just handle an entire city worth of vector roads and cadastre, so experience with higher abstractions was pretty rare. As a nation, we could probably never have agreed on a national LCC projection given that the states had so much mess to sort out themselves, but that's what you see in the USA with its Albers family, and the elided Hawaii and and Alaskan montsrosities . During the time GDA94 was being rolled out the addressing system was being standardized for GPS and modern communication systems, the P-NAF was the original project that took the data from Australia Post. State of the art for address parsing in 2002 was Perl, Google Earth was but a keyhole glimmering in the background in early West Wing episodes, and the idea of "micro-services" was catching on among the venture capital elite. Today the echoes of Oracle and ESRI and RP-Data and ENVI and are still with us.

It was around this time that the large players made their mark in Australia (mid-1990s-early 2000s), MapInfo had a tight hold on many local government authorities (LGAs) because they had the best software, the best formats (TAB, MIF and georeferenced tile TAB for imagery), and somehow got integrated into many state departments. That's why these TAB and MIF formats are here still, shp was the poor interchange cousin, limited to DBF, short field names, no data above the 32-bit index limit, no mixed topologies in a single layer. Aerial imagery was just starting to make an impact and the future business and research interests being recognized. MrSID and ECW were used to integrate large imagery collections into single files and services, while their parent companies waged a furious legal battle around wavelet compression. LizardTech has mostly faded from the scene, but NearMap continues today with "reality as a service", they certainly had the long-game in mind this whole time.

Manifold was in version 5.0 in 2002, and it could read all of these formats as well as provide very accessible rendering, ability to create tiles with links betweeen drawings and images for creating tile sets. ECW was absolutely hot, and ERMapper (Nearmap) had a free viewer that is probably still the best one around until leaflet happened. The point of this long story was to explain that in the early 2000s these files were LARGE and no one had a hope of reading a road line, cadastral parcel, or even address point shapefile for an entire state. We read them in parts, and in pairs or more of parts while we slowly rendered our way around the country building map tile sets deprecated immediately when Google Earth arrived. These days it's a pain to get the file list into one object so you lapply it with the latest R GIS i/o package, but there's really no problem with memory.

This function is here to make it easy to get the file list for Tasmania.

tab, gdb, shp is sf/rgdal ready - gdb works with the directory name, it might work with some of the individual files - I don't know how GDAL otherwise resolves layer names in the same folder but you can give it the path name, this is probably why gdb/, though note that for raster /anything-next-to.adf does work

all will give every thelist file

tab is that glorious ancient format

gdb is a newcomer format, recently reverse engineered by Even

shp is the usual suspect

dbf is the usual suspect's data (ggplot2 calls this metadata)

asc is DEM e.g. list_dem_25m_break_o_day.asc (part of a statewide effort in the early 2000s to build a DEM for Tasmania, it was used to build a networked drainage and topography graph of the state's physical landscape, and this helped spur the development of a powerful imagery orthorectification system and led to some interesting commerical initiatives in general geospatial data)

csv is something else e.g. list_fmp_data.csv

xml,txt is probably just xml, probably only relevant to GDAL and ESRI list_fmp_data_statewide.txt.xml

lyr - style files?

zip - unpackage zips

Arguments are used to pattern match on different aspects of the file name so that anything can be pulled out.

Value

tibble data frame of file names

Examples

```
## Not run:
  thelist_files()

  ## to get statewide sets, find the individual groups first and pick one
  grps <- raadfiles:::thelist_groups()
  print(grps)
  #read_all <- function(pattern) {
  #files <- thelist_files(format = "shp", pattern = pattern)
  #do.call(rbind, lapply(files$fullname,  sf::read_sf))
  #}
  #x <- read_all(sample(grps, 1))

## End(Not run)
```

`topography` *Topographic data files*

Description

Obtain file names for various topographic data.

Usage

```
gebco23_files(all = FALSE, ...)

gebco21_files(all = FALSE, ...)

gebco19_files(all = FALSE, ...)

gebco14_files(all = FALSE, ...)

gebco08_files(all = FALSE, ...)

ramp_files(all = FALSE, ...)

ibcso_files(all = FALSE, ...)

ibcso_background_files(all = FALSE, ...)

ibcso_bed_files(all = FALSE, ...)

ibcso_digital_chart_files(all = FALSE, ...)

ibcso_rid_files(all = FALSE, ...)

ibcso_tid_files(all = FALSE, ...)

ibcso_sid_files(all = FALSE, ...)

cryosat2_files(all = FALSE, ...)

etopo1_files(all = FALSE, ...)

etopo2_files(all = FALSE, ...)

lakesuperior_files(all = FALSE, ...)

kerguelen_files(all = FALSE, ...)

george_v_terre_adelie_1000m_files(all = FALSE, ...)

george_v_terre_adelie_500m_files(all = FALSE, ...)

george_v_terre_adelie_250m_files(all = FALSE, ...)

george_v_terre_adelie_100m_files(all = FALSE, ...)

smith_sandwell_files(all = FALSE, ...)

smith_sandwell_unpolished_files(all = FALSE, ...)
```

```
smith_sandwell_lon180_files(all = FALSE, ...)

smith_sandwell_unpolished_lon180_files(all = FALSE, ...)

macquarie100m_57S_files(all = FALSE, ...)

macquarie100m_58S_files(all = FALSE, ...)
```

Arguments

all	return a larger set of files (for exploratory use only)
...	reserved

Details

Each function exists to match a specific data set, but the optional ‘all‘ argument may be used to easily discover a broader set of files that ship with the data, or that represent older versions, documentation and other metadata files.

There's no single format, there are GeoTIFFs, ArcInfo binary, ERStorage, NetCDF, NetCDF GMT, (Geo) PDF, and some VRT wrappers for handling raw binary files.

Value

data frame of ‘file‘ and ‘fullname‘ columns

GEBCO General Bathymetric Chart of the Oceans

Versions 2008, 2014, 2019, 2021, 2023.

IBCSO International Bathymetric Chart of the Southern Ocean

'is' ('is_PS71' tif, or grd), 'background_hq', 'bed' ('bed_PS71'), 'digital_chart', 'sid' ('sid_PS71')

ETOPO

Etopo1 and Etopo2, Lake Superior

Smith and Sandwell

Polished and unpolished, version 18.1 replaces 18.

Cryosat2

Cryosat2

Australian Antarctic

George V Terre Adelie, Kerguelen, Macquarie 100m

Examples

```
## Not run:
  gebco23_files()

## End(Not run)
```

WOA *World Ocean Atlas products*

Description

WOA find files

Usage

```
woa13_files()

woa09_files()

woa09_daily_files()
```

Details

Current returns all NetCDF files, without any date information, there's a mix of variables month/year climatologies.

Value

tibble data frame of file names

Examples

```
## Not run:
  woa13_files()

## End(Not run)
```

Index