

# Package ‘weatherOz’

April 8, 2026

**Title** An API Client for Australian Weather and Climate Data Resources

**Version** 3.0.0

**Description** Provides automated downloading, parsing and formatting of weather data for Australia through API endpoints provided by the Department of Primary Industries and Regional Development (DPIRD) of Western Australia and by the Science and Technology Division of the Queensland Government's Department of Environment and Science (DES). As well as the Bureau of Meteorology (BOM) of the Australian government precis and coastal forecasts, and downloading and importing radar and satellite imagery files. DPIRD weather data are accessed through public APIs provided by DPIRD, <<https://www.dpird.wa.gov.au/online-tools/apis/>>, providing access to weather station data from the DPIRD weather station network. Australia-wide weather data are based on data from the Australian Bureau of Meteorology (BOM) data and accessed through SILO (Scientific Information for Land Owners) Jeffrey et al. (2001) <[doi:10.1016/S1364-8152\(01\)00008-1](https://doi.org/10.1016/S1364-8152(01)00008-1)>. DPIRD data are made available under a Creative Commons Attribution 3.0 Licence (CC BY 3.0 AU) license <<https://creativecommons.org/licenses/by/3.0/au/deed.en>>. SILO data are released under a Creative Commons Attribution 4.0 International licence (CC BY 4.0) <<https://creativecommons.org/licenses/by/4.0/>>. BOM data are (c) Australian Government Bureau of Meteorology and released under a Creative Commons (CC) Attribution 3.0 licence or Public Access Licence (PAL) as appropriate, see <<https://www.bom.gov.au/copyright>> for further details.

**License** GPL (>= 3)

**URL** <https://github.com/ropensci/weatherOz/>,  
<https://docs.ropensci.org/weatherOz/>

**BugReports** <https://github.com/ropensci/weatherOz/issues>

**Depends** R (>= 4.1.0)

**Imports** apsimx, clock, crayon, curl, crul (>= 1.6.0), data.table (>= 1.1.5), foreign, grDevices, jsonlite, knitr, lubridate, magick, methods, sf, stars, stats, terra, utils, xml2

**Suggests** covr, dplyr, ggplot2, ggthemes, grid, gridExtra, mailR, mapproj, maps, rmarkdown, roxyglobals, spelling, testthat (>= 3.0.0), usethis, vcr (>= 2.0.0), vdiff, withr

**VignetteBuilder** knitr

**Config/roxyglobals/filename** globals.R

**Config/roxyglobals/unique** FALSE

**Config/testthat/edition** 3

**Config/testthat/parallel** true

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**RoxygenNote** 7.3.3

**X-schema.org-applicationCategory** Tools

**X-schema.org-isPartOf** <https://ropensci.org>

**X-schema.org-keywords** dpird, bom, meteorological-data, weather-forecast, australia, weather, weather-data, meteorology, western-australia, australia-bureau-of-meteorology, western-australia-agriculture, australia-agriculture, australia-climate, australia-weather

**NeedsCompilation** no

**Author** Rodrigo Pires [aut, cre] (ORCID:

<https://orcid.org/0000-0001-7384-6849>),

Anna Hepworth [aut] (ORCID: <https://orcid.org/0000-0003-0204-6347>),

Rebecca O'Leary [aut],

Jonathan Carroll [aut] (ORCID: <https://orcid.org/0000-0002-1404-5264>),

James Goldie [aut] (ORCID: <https://orcid.org/0000-0002-5024-6207>),

Dean Marchiori [aut] (ORCID: <https://orcid.org/0000-0002-3430-7225>),

Paul Melloy [aut] (ORCID: <https://orcid.org/0000-0003-4253-7167>),

Mark Padgham [aut] (ORCID: <https://orcid.org/0000-0003-2172-5265>),

Hugh Parsonage [aut] (ORCID: <https://orcid.org/0000-0003-4055-0835>),

Keith Pembleton [ctb] (ORCID: <https://orcid.org/0000-0002-1896-4516>),

Contributed code and ideas for original bomrang package that was used in the creation of weatherOz.),

Maëlle Salmon [ctb] (ORCID: <https://orcid.org/0000-0002-2815-0399>),

Contributed to debugging a nasty little bug with CI where timezones caused tests to fail due to vcr not recognising the URL when run outside of Australia-Perth TZ! Suggested the use of local\_timzone().),

Max Moldovan [ctb] (ORCID: <https://orcid.org/0000-0001-9680-8474>),

Contributed valuable feedback on package usage leading to improvements in the package structure and functionality.),

Stephen Bradshaw [ctb] (ORCID: <https://orcid.org/0000-0003-3096-8787>),

Identified data structure issues and provided insights that improved package data quality and usability.),

Jimmy Ng [ctb],  
 Steve Collins [ctb] (Designed the hex logo for 'weatherOz' hex logo.),  
 Adam H. Sparks [aut] (ORCID: <<https://orcid.org/0000-0002-0061-8359>>),  
 Laurens Geffert [rev],  
 Sam Rogers [rev],  
 Western Australia Agriculture Authority (WAAA) [cph],  
 Curtin University [cph]

**Maintainer** Rodrigo Pires <rodrigo.pires@dpird.wa.gov.au>

**Repository** CRAN

**Date/Publication** 2026-04-08 05:30:02 UTC

## Contents

dpird_extreme_weather_values . . . . .	4
dpird_minute_values . . . . .	4
dpird_summary_values . . . . .	5
find_forecast_towns . . . . .	6
find_nearby_stations . . . . .	7
find_stations_in . . . . .	9
get_available_imagery . . . . .	11
get_available_radar . . . . .	13
get_coastal_forecast . . . . .	14
get_data_drill . . . . .	15
get_data_drill_apsim . . . . .	19
get_dpird_apsim . . . . .	21
get_dpird_availability . . . . .	23
get_dpird_extremes . . . . .	25
get_dpird_minute . . . . .	28
get_dpird_summaries . . . . .	30
get_key . . . . .	35
get_metno_daily_forecast . . . . .	36
get_metno_forecast . . . . .	37
get_patched_point . . . . .	39
get_patched_point_apsim . . . . .	43
get_precis_forecast . . . . .	46
get_radar_imagery . . . . .	47
get_satellite_imagery . . . . .	48
get_stations_metadata . . . . .	50
parse_coastal_forecast . . . . .	53
parse_precis_forecast . . . . .	55
silodaily_values . . . . .	56
south_west_agriculture_region . . . . .	57

**Index**

**58**

---

dpird\_extreme\_weather\_values

*A List of DPIRD Extreme Weather Data Values*

---

### Description

A [vector](#) object containing 57 items representing valid values to supply to `get_dpird_extremes()`'s *values* argument taken from the documentation for the DPIRD Weather 2.0 API.

### Usage

```
dpird_extreme_weather_values
```

### Format

A [vector](#) object of 57 items.

### Source

<https://www.dpird.wa.gov.au/online-tools/apis/>

### See Also

Other DPIRD: [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data: [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#), [silo\\_daily\\_values](#)

---

dpird\_minute\_values

*A List of DPIRD Minute Weather Data Values*

---

### Description

A [vector](#) object containing 12 items representing valid values to supply to `get_dpird_minute()`'s *values* argument taken from the documentation for the DPIRD Weather 2.0 API.

### Usage

```
dpird_minute_values
```

### Format

A [vector](#) object of 12 items.

**Source**

<https://www.dpird.wa.gov.au/online-tools/apis/>

**See Also**

Other DPIRD: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_summary\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_summary\\_values](#), [silodaily\\_values](#)

---

dpird\_summary\_values *A List of DPIRD Summary Weather Data Values*

---

**Description**

A [vector](#) object containing 75 items representing valid values to supply to `get_dpird_summary()`'s *values* argument taken from the documentation for the DPIRD Weather 2.0 API.

**Usage**

```
dpird_summary_values
```

**Format**

A [vector](#) object of 75 items.

**Source**

<https://www.dpird.wa.gov.au/online-tools/apis/>

**See Also**

Other DPIRD: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [silodaily\\_values](#)

---

find\_forecast\_towns *Find the Nearest Town With a BOM Forecast*

---

### Description

For a given latitude and longitude, find the nearest town that the BOM provides a forecast for.

### Usage

```
find_forecast_towns(longitude = 149.2, latitude = -35.3, distance_km = 100)
```

### Arguments

longitude	A numeric value of longitude in decimal degree (DD) format. By default, Canberra (approximately).
latitude	A numeric value of latitude in decimal degree (DD) format. By default, Canberra (approximately).
distance_km	A numeric value of the distance in kilometres from the latitude and longitude point beyond which values will not be returned.

### Value

A `data.table::data.table()` of all forecast towns (in this package) sorted by distance from *latitude* and *longitude*, ascending.

### Author(s)

Hugh Parsonage, <hugh.parsonage@gmail.com>, and James Goldie, <me@jamesgoldie.dev>, and Adam H. Sparks, <adamhsparks@gmail.com>

### See Also

Other BOM: [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#), [parse\\_precis\\_forecast\(\)](#)

Other metadata: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_stations\\_metadata\(\)](#)

### Examples

```
# find forecast towns near Esperance, WA
find_forecast_towns(longitude = 121.8913, latitude = -33.8614)
```

---

find\_nearby\_stations *Find the Nearest Weather Stations to a Given Geographic Point or Known Station*

---

### Description

Find nearby weather stations given geographic coordinates or a station code for both of the DPIRD and SILO weather station networks. Either a combination of *latitude* and *longitude* or *station\_code* must be provided. A DPIRD API key is only necessary to search for stations in the DPIRD network. If you are not interested in DPIRD stations in Western Australia, you may use this function to query only SILO stations for all of Australia without using a key.

### Usage

```
find_nearby_stations(
    longitude = NULL,
    latitude = NULL,
    station_code = NULL,
    distance_km = 100,
    api_key = NULL,
    which_api = "silo",
    include_closed = FALSE
)
```

### Arguments

longitude	A numeric value of longitude in decimal degree (DD) format. Optional and defaults to NULL. Required if <i>station_code</i> is not provided.
latitude	A numeric value of latitude in decimal degree (DD) format. Optional and defaults to NULL. Required if <i>station_code</i> is not provided.
station_code	A string with the station code for the station of interest. Optional and defaults to NULL. Required if <i>longitude</i> and <i>latitude</i> are not provided.
distance_km	A numeric value for distance to limit the search from the station or location of interest. Defaults to 100 km.
api_key	A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. If left as NULL, defaults to automatically detecting your key from your local .Renvirom, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected. Only used when <i>which_api</i> is DPIRD or all.
which_api	A string value that indicates which API to use. Defaults to <i>silo</i> only. Valid values are <i>all</i> , for both SILO (BOM) and DPIRD weather station networks; <i>silo</i> for only stations in the SILO network; or <i>dpird</i> for stations in the DPIRD network.
include_closed	A Boolean value that indicates whether closed stations in the DPIRD network should be included in the results. Defaults to FALSE with closed stations not included.

**Value**

A `data.table::data.table()` with `station_code`, `station_name`, `latitude`, `longitude`, `elev_m`, `state`, `owner`, and `distance`. Data are sorted by increasing distance from station or location of interest.

**Note**

You can request your own API key from DPIRD for free by filling out the form found at <https://www.dpird.wa.gov.au/online-tools/apis/>.

**Author(s)**

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam H. Sparks, <adamhsparks@gmail.com>

**See Also**

Other DPIRD: `dpird_extreme_weather_values`, `dpird_minute_values`, `dpird_summary_values`, `find_stations_in()`, `get_dpird_apsim()`, `get_dpird_availability()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_stations_metadata()`

Other SILO: `find_stations_in()`, `get_data_drill()`, `get_data_drill_apsim()`, `get_patched_point()`, `get_patched_point_apsim()`, `get_stations_metadata()`, `silodaily_values`

Other metadata: `find_forecast_towns()`, `find_stations_in()`, `get_available_imagery()`, `get_available_radar()`, `get_dpird_availability()`, `get_stations_metadata()`

**Examples**

```
## Not run:

# Note that queries to the DPIRD API require you to have your own API key.

# Query WA only stations and return DPIRD's stations nearest to the
# Northam, WA station, "NO", returning stations with 50 km of this station

wa_stn <- find_nearby_stations(
  station_code = "NO",
  distance_km = 50,
  api_key = "your_api_key",
  which_api = "dpird"
)

# Query stations nearest DPIRD's Northam, WA station, "NO" and return both
# DPIRD and SILO/BOM stations within 50 km of this station.

wa_stn <- find_nearby_stations(
  station_code = "NO",
  distance_km = 50,
  api_key = "your_api_key",
  which_api = "all"
)
```

```

# Query Wagga Wagga BOM station finding stations within 200 km of it, note
# that it is not necessary to provide an `api_key` for SILO queries of
# nearby stations.

wagga_stn <- find_nearby_stations(
  latitude = -35.1583,
  longitude = 147.4575,
  distance_km = 200,
  which_api = "silo"
)

## End(Not run)

```

---

find_stations_in	<i>Find Stations Within a Geospatially Defined Geographic Area of Interest</i>
------------------	--

---

## Description

Given an **sf** polygon or a bounding box as a vector with the minimum and maximum longitude and latitude values, find DPIRD or BOM stations in the SILO network that fall within that defined area or the station nearest the centroid of the area of interest.

## Usage

```

find_stations_in(
  x,
  centroid = FALSE,
  api_key = NULL,
  which_api = "all",
  include_closed = FALSE,
  crs = "EPSG:7844"
)

```

## Arguments

x	One of two types of object: <ul style="list-style-type: none"> <li>• A Vector A four-digit vector defining a bounding box of the area of interest in this order, 'xmin', 'ymin', 'xmax', 'ymax', or</li> <li>• An object of class <b>sf</b> defining the area of interest.</li> </ul>
centroid	Boolean A value of TRUE or FALSE indicating whether you want the centroid only to be used to find the nearest station to the centre of the area of interest. If "n" polygons are supplied, "n" stations are returned. Defaults to FALSE with all stations within the area of interest returned.

api_key	A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. If left as NULL, defaults to automatically detecting your key from your local .Renviro, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected. Only used when <i>which_api</i> is DPIRD or all.
which_api	A string value that indicates which API to use. Defaults to <i>siilo</i> only. Valid values are <i>all</i> , for both SILO (BOM) and DPIRD weather station networks; <i>siilo</i> for only stations in the SILO network; or <i>dpird</i> for stations in the DPIRD network.
include_closed	A Boolean value that indicates whether closed stations in the DPIRD network should be included in the results. Defaults to FALSE with closed stations not included.
crs	A string value that provides the coordinate reference system, AKA, "projection" to be used for the point extraction. Defaults to GDA 2020, EPSG:7844. <b>NOTE</b> This will override any <i>crs</i> value that your <code>.polygon</code> provides unless you specify it again here, <i>e.g.</i> , <code>crs = sf::st_crs(polygon_object_name)</code> .

### Value

a **data.table** object of weather station(s) within the defined area of interest in an unprojected format, EPSG:4326, WGS 84 – WGS84 - World Geodetic System 1984, used in GPS format.

### See Also

Other DPIRD: `dpird_extreme_weather_values`, `dpird_minute_values`, `dpird_summary_values`, `find_nearby_stations()`, `get_dpird_apsim()`, `get_dpird_availability()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_stations_metadata()`

Other SILO: `find_nearby_stations()`, `get_data_drill()`, `get_data_drill_apsim()`, `get_patched_point()`, `get_patched_point_apsim()`, `get_stations_metadata()`, `siilo_daily_values`

Other metadata: `find_forecast_towns()`, `find_nearby_stations()`, `get_available_imagery()`, `get_available_radar()`, `get_dpird_availability()`, `get_stations_metadata()`

### Examples

```
# using a (generous) bounding box for Melbourne, Vic using only the SILO API
# for BOM stations, so no API key is needed.

bbox <- find_stations_in(
  x = c(144.470215, -38.160476, 145.612793, -37.622934),
  which_api = "SILO",
  include_closed = TRUE
)
bbox

# Use the same bounding box but only find a single station nearest
# the centroid using only the SILO API for BOM stations
```

```
centroid <- find_stations_in(  
  x = c(144.470215, -38.160476, 145.612793, -37.622934),  
  which_api = "SILO",  
  include_closed = TRUE,  
  centroid = TRUE  
)  
centroid  
  
# Use the `south_west_agricultural_region` data to fetch stations only in the  
# south-western portion of WA and plot it with {ggplot2} showing open/closed  
# stations just to be sure they're inside the area of interest.  
  
# As this is in WA, we can use the DPIRD network, so we need our API key.  
# Using the `south_west_agricultural_region` {sf} object provided.  
  
sw_wa <- find_stations_in(  
  x = south_west_agricultural_region,  
  api_key = "your_api_key",  
  include_closed = TRUE  
)  
  
sw_wa
```

---

get\_available\_imagery *Get a List of Available BOM Satellite Imagery*

---

## Description

Fetch a listing of BOM GeoTIFF satellite imagery from <ftp://ftp.bom.gov.au/anon/gen/gms/> to determine which files are currently available for download. Files are available at ten minute update frequency with a 24-hour delete time. It is useful to know the most recent files available and then specify in the `get_satellite_imagery()` function. Ported from **bomrang**.

## Usage

```
get_available_imagery(product_id = "all")
```

## Arguments

product_id	Character. BOM product ID of interest for which a list of available images will be returned. Defaults to all images currently available.
------------	--

## Details

Valid BOM satellite Product IDs for GeoTIFF files include:

**IDE00420** AHI cloud cover only 2km FD GEOS GIS

**IDE00421** AHI IR (Ch13) greyscale 2km FD GEOS GIS

**IDE00422** AHI VIS (Ch3) greyscale 2km FD GEOS GIS  
**IDE00423** AHI IR (Ch13) Zehr 2km FD GEOS GIS  
**IDE00425** AHI VIS (true colour) / IR (Ch13 greyscale) composite 1km FD GEOS GIS  
**IDE00426** AHI VIS (true colour) / IR (Ch13 greyscale) composite 2km FD GEOS GIS  
**IDE00427** AHI WV (Ch8) 2km FD GEOS GIS  
**IDE00430** AHI cloud cover only 2km AUS equirect. GIS  
**IDE00431** AHI IR (Ch13) greyscale 2km AUS equirect. GIS  
**IDE00432** AHI VIS (Ch3) greyscale 2km AUS equirect. GIS  
**IDE00433** AHI IR (Ch13) Zehr 2km AUS equirect. GIS  
**IDE00435** AHI VIS (true colour) / IR (Ch13 greyscale) composite 1km AUS equirect. GIS  
**IDE00436** AHI VIS (true colour) / IR (Ch13 greyscale) composite 2km AUS equirect. GIS  
**IDE00437** AHI WV (Ch8) 2km AUS equirect. GIS  
**IDE00439** AHI VIS (Ch3) greyscale 0.5km AUS equirect. GIS

### Value

A vector of all available files for the requested Product ID(s).

### Author(s)

Adam H. Sparks, <adamhsparks@gmail.com>

### References

Australian Bureau of Meteorology (BOM) high-definition satellite images <https://www.bom.gov.au/australia/satellite/index.shtml>

### See Also

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#), [parse\\_precis\\_forecast\(\)](#)

Other metadata: [find\\_forecast\\_towns\(\)](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_stations\\_metadata\(\)](#)

### Examples

```

# Check availability of AHI VIS (true colour) / IR (Ch13 greyscale) composite
# 1km FD GEOS GIS images
imagery <- get_available_imagery(product_id = "IDE00425")

imagery

```

---

get\_available\_radar     *Get a List of Available BOM Radar Imagery*

---

### Description

Fetch a listing of available BOM RADAR imagery from <ftp://ftp.bom.gov.au/anon/gen/radar/> to determine which files are currently available for download. The files available are the most recent RADAR imagery for each location, which are updated approximately every 6 to 10 minutes by the BOM. Ported from **bomrang**.

### Usage

```
get_available_radar(radar_id = "all")
```

### Arguments

radar\_id            Numeric. BOM radar of interest for which a list of available images will be returned. Defaults to all images currently available.

### Details

Valid BOM RADAR ID for each location required.

### Value

A `data.table::data.table()` of all selected RADAR locations with location information and `product_ids`.

### Author(s)

Dean Marchiori, <[deanmarchiori@gmail.com](mailto:deanmarchiori@gmail.com)>, and Adam H. Sparks, <[adamhsparks@gmail.com](mailto:adamhsparks@gmail.com)>

### References

Australian Bureau of Meteorology (BOM) radar image <https://www.bom.gov.au/weather-and-climate/rain-radar-and-weather-maps>.

### See Also

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#), [parse\\_precis\\_forecast\(\)](#)

Other metadata: [find\\_forecast\\_towns\(\)](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_stations\\_metadata\(\)](#)

## Examples

```
# Check availability radar imagery for Wollongong (radar_id = 3)
imagery <- get_available_radar(radar_id = 3)

imagery
```

---

get\_coastal\_forecast *Get a BOM Coastal Waters Forecast*

---

## Description

Fetch the BOM daily Coastal Waters Forecast for a specified state or region.

## Usage

```
get_coastal_forecast(state = "AUS")
```

## Arguments

**state** Australian state or territory as full name or postal code. Fuzzy string matching via `base::agrep()` is done. Defaults to AUS returning all state forecasts, see details for further information.

## Details

Allowed state and territory postal codes, only one state per request or all using 'AUS':

**AUS** Australia, returns forecast for all states, NT and ACT

**ACT** Australian Capital Territory (will return NSW)

**NSW** New South Wales

**NT** Northern Territory

**QLD** Queensland

**SA** South Australia

**TAS** Tasmania

**VIC** Victoria

**WA** Western Australia

## Value

A `data.table::data.table()` of an Australia BOM Coastal Waters Forecast.

## Author(s)

Dean Marchiori, <deanmarchiori@gmail.com>, and Paul Melloy, <paul@melloy.com.au>

## References

Forecast data come from Australian Bureau of Meteorology (BOM) Weather Data Services <https://www.bom.gov.au/catalogue/data-feeds.shtml>.

And also,

Location data and other metadata come from the BOM anonymous FTP server with spatial data <ftp://ftp.bom.gov.au/anon/home/adfd/spatial/>, specifically the DBF file portion of a shapefile, <ftp://ftp.bom.gov.au/anon/home/adfd/spatial/IDM00003.dbf>.

## See Also

[parse\\_coastal\\_forecast](#)

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#), [parse\\_precis\\_forecast\(\)](#)

Other data fetching: [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

## Examples

```
get_coastal_forecast(state = "NSW")
```

---

```
get_data_drill
```

```
Get DataDrill Weather Data From SILO
```

---

## Description

Fetch nicely formatted weather data from the SILO API of spatially interpolated weather data (DataDrill). The daily climate surfaces have been derived either by splining or kriging the observational data. The returned values contain “source” columns, which denote how the observations were derived. The grid spans 112° to 154°, -10° to -44° with resolution 0.05° latitude by 0.05° longitude (approximately 5 km × 5 km).

## Usage

```
get_data_drill(
  longitude,
  latitude,
  start_date,
  end_date = Sys.Date(),
  values = "all",
  api_key = get_key(service = "SILO")
)
```

**Arguments**

longitude	A single numeric value representing the longitude of the point-of-interest to the hundredths ( <i>e.g.</i> , 0.05) of a degree.
latitude	A single numeric value representing the latitude of the point-of-interest to the hundredths ( <i>e.g.</i> , 0.05) of a degree.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.
values	A character string with the type of weather data to return. See <b>Available Values</b> for a full list of valid values. Defaults to all with all available values being returned.
api_key	A character string containing your API key, an e-mail address, for the request. Defaults to automatically detecting your key from your local .Renvirom, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

**Value**

a `data.table::data.table()` with the weather data queried with the weather variables in alphabetical order. The first eight columns will always be:

- longitude,
- latitude,
- elev\_m (elevation in metres),
- date (ISO8601 format, YYYYMMDD),
- year,
- month,
- day,
- extracted (the date on which the query was made)

**Column Name Details**

Column names are converted from the default returns of the API to be snake\_case formatted and where appropriate, the names of the values that are analogous between SILO and DPIRD data are named using the same name for ease of interoperability, *e.g.*, using `rbind()` to create a `data.table` that contains data from both APIs.

### Available Values

**all** Which will return all of the following values  
**rain (mm)** Rainfall  
**max\_temp (degrees C)** Maximum temperature  
**min\_temp (degrees C)** Minimum temperature  
**vp (hPa)** Vapour pressure  
**vp\_deficit (hPa)** Vapour pressure deficit  
**evap\_pan (mm)** Class A pan evaporation  
**evap\_syn (mm)** Synthetic estimate<sup>1</sup>  
**evap\_comb (mm)** Combination (synthetic estimate pre-1970, class A pan 1970 onwards)  
**evap\_morton\_lake (mm)** Morton's shallow lake evaporation  
**radiation (Mj/m<sup>2</sup>)** Solar exposure, consisting of both direct and diffuse components  
**rh\_tmax (%)** Relative humidity at the time of maximum temperature  
**rh\_tmin (%)** Relative humidity at the time of minimum temperature  
**et\_short\_crop (mm)** FAO56<sup>4</sup> short crop  
**et\_tall\_crop (mm)** ASCE<sup>5</sup> tall crop<sup>6</sup>  
**et\_morton\_actual (mm)** Morton's areal actual evapotranspiration  
**et\_morton\_potential (mm)** Morton's point potential evapotranspiration  
**et\_morton\_wet (mm)** Morton's wet-environment areal potential evapotranspiration over land  
**mslp (hPa)** Mean sea level pressure

### Value information

Solar radiation: total incoming downward shortwave radiation on a horizontal surface, derived from estimates of cloud oktas and sunshine duration<sup>3</sup>.

Relative humidity: calculated using the vapour pressure measured at 9am, and the saturation vapour pressure computed using either the maximum or minimum temperature<sup>6</sup>.

Evaporation and evapotranspiration: an overview of the variables provided by SILO is available here, [https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration\\_overview.pdf](https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration_overview.pdf).

### Data codes

Data codes Where possible (depending on the file format), the data are supplied with codes indicating how each datum was obtained.

- 0** Official observation as supplied by the Bureau of Meteorology
- 15** Deaccumulated rainfall (original observation was recorded over a period exceeding the standard 24 hour observation period)
- 25** Interpolated from daily observations for that date
- 26** Synthetic Class A pan evaporation, calculated from temperatures, radiation and vapour pressure
- 35** Interpolated from daily observations using an anomaly interpolation method
- 75** Interpolated from the long term averages of daily observations for that day of year

**Author(s)**

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam H. Sparks, <adamhsparks@gmail.com>

**References**

1. Rayner, D. (2005). Australian synthetic daily Class A pan evaporation. Technical Report December 2005, Queensland Department of Natural Resources and Mines, Indooroopilly, Qld., Australia, 40 pp.
2. Morton, F. I. (1983). Operational estimates of areal evapotranspiration and their significance to the science and practice of hydrology, *Journal of Hydrology*, Volume 66, 1-76.
3. Zajaczkowski, J., Wong, K., & Carter, J. (2013). Improved historical solar radiation gridded data for Australia, *Environmental Modelling & Software*, Volume 49, 64–77. DOI: [doi:10.1016/j.envsoft.2013.06.013](https://doi.org/10.1016/j.envsoft.2013.06.013).
4. Food and Agriculture Organization of the United Nations, Irrigation and drainage paper 56: Crop evapotranspiration - Guidelines for computing crop water requirements, 1998.
5. ASCE's Standardized Reference Evapotranspiration Equation, proceedings of the National Irrigation Symposium, Phoenix, Arizona, 2000.
6. For further details refer to Jeffrey, S.J., Carter, J.O., Moodie, K.B. and Beswick, A.R. (2001). Using spatial interpolation to construct a comprehensive archive of Australian climate data, *Environmental Modelling and Software*, Volume 16/4, 309-330. DOI: [doi:10.1016/S1364-8152\(01\)000081](https://doi.org/10.1016/S1364-8152(01)000081).

**See Also**

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_stations\\_metadata\(\)](#), [silos\\_daily\\_values](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

**Examples**

```
## Not run:
# requires an API key as your email address
# Source data from latitude and longitude coordinates (gridded data) for
# max and minimum temperature and rainfall for Southwood, QLD.
wd <- get_data_drill(
  latitude = -27.85,
  longitude = 150.05,
  start_date = "20221001",
  end_date = "20221201",
  values = c("max_temp", "min_temp", "rain"),
  api_key = "your_api_key"
)

## End(Not run)
```

---

get\_data\_drill\_apsim *Get DataDrill Weather Data in the APSIM Format From SILO*

---

## Description

Fetch APSIM .met file formatted weather data from the weather data from the SILO API of spatially interpolated weather data (DataDrill). The daily climate surfaces have been derived either by splining or kriging the observational data. The returned values contain “source” columns, which denote how the observations were derived. The grid spans 112° to 154°, -10° to -44° with resolution 0.05° latitude by 0.05° longitude (approximately 5 km × 5 km).

## Usage

```
get_data_drill_apsim(
  longitude,
  latitude,
  start_date,
  end_date = Sys.Date(),
  api_key = get_key(service = "SILO")
)
```

## Arguments

longitude	A single numeric value representing the longitude of the point-of-interest.
latitude	A single numeric value representing the latitude of the point-of-interest.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.
api_key	A character string containing your API key, an e-mail address, for the request. Defaults to automatically detecting your key from your local .Renvron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

## Details

Note that when saving, comments from SILO will be included, but these will not be printed as a part of the resulting met object in your R session.

## Value

An **apsimx** object of class ‘met’ with attributes.

### Included Values

- rain (mm)** Rainfall
- maxt (degrees C)** Maximum temperature
- mint (degrees C)** Minimum temperature
- vp (hPa)** Vapour pressure
- evap\_pan (mm)** Class A pan evaporation
- radiation (Mj/m<sup>1</sup>)** Solar exposure, consisting of both direct and diffuse components

### Value information

Solar radiation: total incoming downward shortwave radiation on a horizontal surface, derived from estimates of cloud oktas and sunshine duration<sup>2</sup>.

Evaporation and evapotranspiration: an overview of the variables provided by SILO is available here, [https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration\\_overview.pdf](https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration_overview.pdf).

### Data codes

Where the source code is a 6 digit string comprising the source code for the 6 variables. The single digit code for each variable is:

- 0** an actual observation;
- 1** an actual observation from a composite station;
- 2** a value interpolated from daily observations;
- 3** a value interpolated from daily observations using the anomaly interpolation method for CLIMARC data;
- 6** a synthetic pan value; or
- 7** an interpolated long term average.

### Saving objects

To save “met” objects the `apsimx::write_apsim_met()` is reexported. Note that when saving, comments from SILO will be included, but these will not be printed as a part of the resulting met object in your R session.

### Author(s)

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam Sparks, <adamhsparks@gmail.com>

### References

1. Rayner, D. (2005). Australian synthetic daily Class A pan evaporation. Technical Report December 2005, Queensland Department of Natural Resources and Mines, Indooroopilly, Qld., Australia, 40 pp.
2. Morton, F. I. (1983). Operational estimates of areal evapotranspiration and their significance to the science and practice of hydrology, *Journal of Hydrology*, Volume 66, 1-76.

**See Also**

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_stations\\_metadata\(\)](#), [silos\\_daily\\_values](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

Other APSIM: [get\\_dpird\\_apsim\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [reexports](#)

**Examples**

```
## Not run:
# requires an API key as your email address
# Source data from latitude and longitude coordinates (gridded data) for
# max and minimum temperature and rainfall for Southwood, QLD.
wd <- get_data_drill_apsim(
  latitude = -27.85,
  longitude = 150.05,
  start_date = "20220101",
  end_date = "20221231",
  api_key = "your_api_key"
)

## End(Not run)
```

---

get_dpird_apsim	<i>Get DPIRD Summary Weather Data in the APSIM Format From the Weather 2.0 API</i>
-----------------	--

---

**Description**

Automates the retrieval and conversion of summary data from the DPIRD Weather 2.0 API to an APSIM .met file formatted weather data object.

**Usage**

```
get_dpird_apsim(
  station_code,
  start_date,
  end_date = Sys.Date(),
  api_key = get_key(service = "DPIRD")
)
```

**Arguments**

station_code	A character string or factor from <code>get_stations_metadata()</code> of the BOM station code for the station of interest.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.
api_key	A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. Defaults to automatically detecting your key from your local .Renviron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

**Value**

An **apsimx** object of class ‘met’ with attributes.

**Saving objects**

To save “met” objects the `apsimx::write_apsim_met()` is reexported. Note that when saving, comments from SILO will be included, but these will not be printed as a part of the resulting met object in your R session.

**Author(s)**

Adam H. Sparks, <adamhsparks@gmail.com>

**See Also**

Other DPIRD: `dpird_extreme_weather_values`, `dpird_minute_values`, `dpird_summary_values`, `find_nearby_stations()`, `find_stations_in()`, `get_dpird_availability()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_stations_metadata()`

Other data fetching: `get_coastal_forecast()`, `get_data_drill()`, `get_data_drill_apsim()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_metno_daily_forecast()`, `get_metno_forecast()`, `get_patched_point()`, `get_patched_point_apsim()`, `get_precis_forecast()`, `get_radar_imagery()`, `get_satellite_imagery()`

Other APSIM: `get_data_drill_apsim()`, `get_patched_point_apsim()`, `reexports`

**Examples**

```
## Not run:
# Get an APSIM format object for Binnu
# Note that you need to supply your own API key

wd <- get_dpird_apsim(
```

```

    station_code = "BI",
    start_date = "20220101",
    end_date = "20221231",
    api_key = "your_api_key"
)

## End(Not run)

```

---

```
get_dpird_availability
```

*Get the Availability for DPIRD Weather Stations*

---

## Description

Fetch the availability metadata of weather stations in the DPIRD weather station network from the Weather 2.0 API.

## Usage

```

get_dpird_availability(
  station_code = NULL,
  start_date = NULL,
  end_date = NULL,
  values = "availability",
  api_key = get_key(service = "DPIRD")
)

```

## Arguments

station_code	A character string of the DPIRD station code for the station of interest. Defaults to NULL, returning metadata for all stations during the requested <i>start_date</i> and <i>end_date</i> interval.
start_date	A character string representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). This function will return data inclusive of this date. Defaults to NULL, returning data for the current year-to-date. Must be sent along with an <i>end_date</i> .
end_date	A character string representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). This function will return data inclusive of this date. Defaults to NULL, returning data for the current year-to-date. Must be sent with a <i>start_date</i> .
values	A character string with the type of availability metadata to return. See <b>Available Values</b> for a full list of valid values. Defaults to <i>availability</i> , returning metadata for all stations.

`api_key` A character string containing your API key from DPIRD, <https://www.dpird.wa.gov.au/online-tools/apis/>, for the DPIRD Weather 2.0 API. Defaults to automatically detecting your key from your local `.Renvirom`, `.Rprofile` or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

### Value

a `data.table::data.table()` with `station_code` and the requested metadata.

### Available Values

- `availability` (which will return all of the following values),
- `availabilityCurrentHour`,
- `availabilityLast7DaysSince9AM`,
- `availabilityLast7DaysSince12AM`,
- `availabilityLast14DaysSince9AM`,
- `availabilityLast14DaysSince12AM`,
- `availabilityLast24Hours`,
- `availabilityMonthToDateSince12AM`,
- `availabilityMonthToDateTo9AM`,
- `availabilitySince9AM`,
- `availabilitySince12AM`,
- `availabilityTo9AM`,
- `availabilityYearToDateSince12AM`, and
- `availabilityYearToDateTo9AM`

### Author(s)

Adam H. Sparks, <[adamhsparks@gmail.com](mailto:adamhsparks@gmail.com)>

### See Also

Other DPIRD: `dpird_extreme_weather_values`, `dpird_minute_values`, `dpird_summary_values`, `find_nearby_stations()`, `find_stations_in()`, `get_dpird_apsim()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_stations_metadata()`

Other metadata: `find_forecast_towns()`, `find_nearby_stations()`, `find_stations_in()`, `get_available_imagery()`, `get_available_radar()`, `get_stations_metadata()`

## Examples

```
## Not run:
# Note that you need to supply your own API key

# Here we check the up time for the current year for Westonia
WS001 <- get_dpird_availability(station_code = "WS001",
                              api_key = "your_api_key")

# Here we check the up time for 2017 for Binnu
BN <- get_dpird_availability(
  station_code = "BI",
  start_date = "20170101",
  end_date = "20171231",
  api_key = "your_api_key"
)

## End(Not run)
```

---

get\_dpird\_extremes      *Get DPIRD Extreme Weather Event Summaries*

---

## Description

Fetch nicely formatted individual extreme weather summaries from the DPIRD Weather 2.0 API.

## Usage

```
get_dpird_extremes(
  station_code,
  values = "all",
  api_key = get_key(service = "DPIRD")
)
```

## Arguments

station_code	A character string or factor from <a href="#">get_stations_metadata()</a> of the BOM station code for the station of interest.
values	A character string with the type of extreme weather to return. See <b>Available Values</b> for a full list of valid values. Defaults to all, returning the full list of values unless otherwise specified.
api_key	A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. Defaults to automatically detecting your key from your local .Renviron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

**Value**

a `data.table::data.table()` of one row with `station_code`, `station_name`, `latitude`, `longitude`, `date_time` of the query and the extreme weather information according to the value(s) selected.

**Available Values**

- all (which will return all of the following values),
- `erosionCondition`,
- `erosionConditionLast7Days`,
- `erosionConditionLast7DaysDays`,
- `erosionConditionLast7DaysMinutes`,
- `erosionConditionLast14Days`,
- `erosionConditionLast14DaysDays`,
- `erosionConditionLast14DaysMinutes`,
- `erosionConditionMonthToDate`,
- `erosionConditionMonthToDateDays`,
- `erosionConditionMonthToDateMinutes`,
- `erosionConditionMonthToDateStartTime`,
- `erosionConditionSince12AM`,
- `erosionConditionSince12AMMinutes`,
- `erosionConditionSince12AMStartTime`,
- `erosionConditionYearToDate`,
- `erosionConditionYearToDateDays`,
- `erosionConditionYearToDateMinutes`,
- `erosionConditionYearToDateStartTime`,
- `frostCondition`,
- `frostConditionLast7Days`,
- `frostConditionLast7DaysDays`,
- `frostConditionLast7DaysMinutes`,
- `frostConditionLast14Days`,
- `frostConditionLast14DaysDays`,
- `frostConditionLast14DaysMinutes`,
- `frostConditionMonthToDate`,
- `frostConditionMonthToDateDays`,
- `frostConditionMonthToDateMinutes`,
- `frostConditionMonthToDateStartTime`,
- `frostConditionSince9AM`,
- `frostConditionSince9AMMinutes`,

- `frostConditionSince9AMStartTime`,
- `frostConditionTo9AM`,
- `frostConditionTo9AMMinutes`,
- `frostConditionTo9AMStartTime`,
- `frostConditionYearToDate`,
- `frostConditionYearToDate`,
- `frostConditionYearToDateMinutes`,
- `frostConditionYearToDateStartTime`,
- `heatCondition`,
- `heatConditionLast7Days`,
- `heatConditionLast7DaysDays`,
- `heatConditionLast7DaysMinutes`,
- `heatConditionLast14Days`,
- `heatConditionLast14DaysDays`,
- `heatConditionLast14DaysMinutes`,
- `heatConditionMonthToDate`,
- `heatConditionMonthToDateDays`,
- `heatConditionMonthToDateMinutes`,
- `heatConditionMonthToDateStartTime`,
- `heatConditionSince12AM`,
- `heatConditionSince12AMMinutes`,
- `heatConditionSince12AMStartTime`,
- `heatConditionYearToDate`,
- `heatConditionYearToDateDays`,
- `heatConditionYearToDateMinutes`, and
- `heatConditionYearToDateStartTime`

**Author(s)**

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam Sparks, <adamhsparks@gmail.com>

**See Also**

Other DPIRD: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

## Examples

```
## Not run:
# Query Bonnie Rock station for wind erosion and heat extreme events
# Note that you need to supply your own API key

xtreme <- get_dpird_extremes(
  station_code = "BR",
  values = c("erosionCondition",
             "heatCondition"),
  api_key = "your_api_key"
)

## End(Not run)
```

---

get\_dpird\_minute

*Get DPIRD Weather Data by the Minute*


---

## Description

Fetch nicely formatted minute weather station data from the DPIRD Weather 2.0 API for a maximum 24-hour period.

## Usage

```
get_dpird_minute(
  station_code,
  start_date_time = lubridate::now() - lubridate::hours(24L),
  minutes = 1440L,
  values = "all",
  api_key = get_key(service = "DPIRD")
)
```

## Arguments

station_code	A character string or factor from <a href="#">get_stations_metadata()</a> of the BOM station code for the station of interest.
start_date_time	A character string representing the start date and time of the query in the format “yyyy-mm-dd-hh-mm” (ISO8601). Defaults to 24 hours before the current local system time, returning the most recent 24 hour observations rounded to the nearest minute. This function does its best to decipher many date and time formats but prefers ISO8601.
minutes	An integer value that provides the number of observations to be returned. Defaults to 1440 minutes for 24 hours of observations.
values	A vector of weather values to query from the API. See <b>Available Values</b> section for valid available codes. Defaults to all available values, all.

`api_key` A character string containing your API key from DPIRD, <https://www.dpird.wa.gov.au/online-tools/apis/>, for the DPIRD Weather 2.0 API. Defaults to automatically detecting your key from your local `.Renviron`, `.Rprofile` or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

### Value

a `data.table::data.table()` with `station_code` and the date interval queried together with the requested weather variables.

### Available Values

- all (which will return all of the following values),
- `airTemperature`,
- `dateTime`,
- `dewPoint`,
- `rainfall`,
- `relativeHumidity`,
- `soilTemperature`,
- `solarIrradiance`,
- `wetBulb`,
- `wind`,
- `windAvgSpeed`,
- `windMaxSpeed`, and
- `windMinSpeed`

### Note

Please note this function converts date-time columns from Coordinated Universal Time ‘UTC’ returned by the API to Australian Western Standard Time ‘AWST’.

### Author(s)

Adam H. Sparks, <adamhsparks@gmail.com>

### See Also

Other DPIRD: `dpird_extreme_weather_values`, `dpird_minute_values`, `dpird_summary_values`, `find_nearby_stations()`, `find_stations_in()`, `get_dpird_apsim()`, `get_dpird_availability()`, `get_dpird_extremes()`, `get_dpird_summaries()`, `get_stations_metadata()`

Other data fetching: `get_coastal_forecast()`, `get_data_drill()`, `get_data_drill_apsim()`, `get_dpird_apsim()`, `get_dpird_extremes()`, `get_dpird_summaries()`, `get_metno_daily_forecast()`, `get_metno_forecast()`, `get_patched_point()`, `get_patched_point_apsim()`, `get_precis_forecast()`, `get_radar_imagery()`, `get_satellite_imagery()`

**Examples**

```
## Not run:

# Note that you need to supply your own API key

get_dpird_minute(
  station_code = "SP",
  start_date_time = "2023-02-01 13:00:00",
  minutes = 1440,
  values = c("airTemperature",
             "solarIrradiance",
             "wind"),
  api_key = "your_api_key"
)

## End(Not run)
```

---

```
get_dpird_summaries   Get DPIRD Weather Data in Summarised Formats
```

---

**Description**

Fetch nicely formatted individual station weather summaries from the DPIRD Weather 2.0 API.

**Usage**

```
get_dpird_summaries(
  station_code,
  start_date,
  end_date = Sys.Date(),
  interval = c("daily", "15min", "30min", "hourly", "monthly", "yearly"),
  values = "all",
  api_key = get_key(service = "DPIRD")
)
```

**Arguments**

station_code	A character string or factor from <a href="#">get_stations_metadata()</a> of the BOM station code for the station of interest.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.

interval	A character string that indicates the time interval to monthly or yearly. For intervals shorter than 1 day, the time period covered will be midnight to midnight, with the end_date time interval being before midnight - hour/minute values are for the end of the time period. Data for shorter intervals (15min, 30min) are available from January of the previous year.
values	A character string with the type of summarised weather to return. See <b>Available Values</b> for a full list of valid values. Defaults to all with all available values being returned.
api_key	A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. Defaults to automatically detecting your key from your local .Renviron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

### Value

a `data.table::data.table()` with station\_code and the date interval queried together with the requested weather variables in alphabetical order. The first ten columns will always be:

- station\_code,
- station\_name,
- longitude,
- latitude,
- year,
- month,
- day,
- hour,
- minute, and if month or finer is present,
- date (a combination of year, month, day, hour, minute as appropriate).

### Start Dates

The earliest available data start from August of 2000 for Vasse, “VA”.

### Column Name Details

Column names are converted from the default returns of the API to be snake\_case formatted and where appropriate, the names of the values that are analogous between SILO and DPIRD data are named using the same name for ease of interoperability, *e.g.*, using `rbind()` to create a `data.table` that contains data from both APIs. However, use with caution and don’t mix datasets of different time-steps, *i.e.*, this function gets many summary values not just “daily” time-step data. The functions that access the SILO API only provide access to daily data, so don’t mix (sub)hourly, monthly or yearly data from DPIRD with SILO.

**Available Values**

- all (which will return all of the following values),
- airTemperature,
- airTemperatureAvg,
- airTemperatureMax,
- airTemperatureMaxTime,
- airTemperatureMin,
- airTemperatureMinTime,
- apparentAirTemperature,
- apparentAirTemperatureAvg,
- apparentAirTemperatureMax,
- apparentAirTemperatureMaxTime,
- apparentAirTemperatureMin,
- apparentAirTemperatureMinTime,
- barometricPressure,
- barometricPressureAvg,
- barometricPressureMax,
- barometricPressureMaxTime,
- barometricPressureMin,
- barometricPressureMinTime,
- battery,
- batteryMinVoltage,
- batteryMinVoltageDateTime,
- chillHours,
- deltaT,
- deltaTAvg,
- deltaTMax,
- deltaTMaxTime,
- deltaTMin,
- deltaTMinTime,
- dewPoint,
- dewPointAvg,
- dewPointMax,
- dewPointMaxTime,
- dewPointMin,
- dewPointMinTime,
- erosionCondition,

- erosionConditionMinutes,
- erosionConditionStartTime,
- errors,
- etoShortCrop,
- etoTallCrop,
- evapotranspiration,
- evapotranspirationShortCrop,
- evapotranspirationTallCrop,
- frostCondition,
- frostConditionMinutes,
- frostConditionStartTime,
- heatCondition,
- heatConditionMinutes,
- heatConditionStartTime,
- observations,
- observationsCount,
- observationsPercentage,
- panEvaporation,
- panEvaporation12AM,
- rainfall,
- relativeHumidity,
- relativeHumidityAvg,
- relativeHumidityMax,
- relativeHumidityMaxTime,
- relativeHumidityMin,
- relativeHumidityMinTime,
- richardsonUnits,
- soilTemperature,
- soilTemperatureAvg,
- soilTemperatureMax,
- soilTemperatureMaxTime,
- soilTemperatureMin,
- soilTemperatureMinTime,
- solarExposure,
- wetBulb,
- wetBulbAvg,
- wetBulbMax,

- wetBulbMaxTime,
- wetBulbMin,
- wetBulbMinTime,
- wind,
- windAvgSpeed, and
- windMaxSpeed

**Note**

Please note this function converts date-time columns from Coordinated Universal Time ‘UTC’ to Australian Western Standard Time ‘AWST’.

**Author(s)**

Adam H. Sparks, <adamhsparks@gmail.com>, and Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>

**See Also**

Other DPIRD: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

**Examples**

```
## Not run:
# Note that you need to supply your own API key
# Use default for end date (current system date) to get rainfall

wd <- get_dpird_summaries(
  station_code = "CL001",
  start_date = "20171028",
  api_key = "your_api_key",
  interval = "yearly",
  values = "rainfall"
)

# Only for wind and erosion conditions for daily time interval

wd <- get_dpird_summaries(
  station_code = "BI",
  start_date = "20220501",
  end_date = "20220502",
  api_key = "your_api_key",
  interval = "daily",
  values = c(
```

```
    "wind",
    "erosionCondition",
    "erosionConditionMinutes",
    "erosionConditionStartTime"
  )
)

## End(Not run)
```

---

get\_key

*Get or Set Up API Keys*

---

## Description

Checks first to get key from your .Rprofile or .Renviron (or similar) file. If it's not found, then it suggests setting it up. Can be used to check that your key that R is using is the key that you wish to be using or for guidance in setting up the keys.

## Usage

```
get_key(service = c("DPIRD", "SILO", "METNO"))
```

## Arguments

service (character) The API host i.e., "DPIRD", "SILO", or "METNO".

## Details

The suggestion is to use your .Renviron to set up the API keys. However, if you regularly interact with the APIs outside of R using some other language you may wish to set these up in your .bashrc, .zshrc, or config.fish for cross-language use.

## Value

A string value with either a DPIRD Weather 2.0 API, SILO and METNO API key value.

## Examples

```
## Not run:
get_key(service = "DPIRD")
get_key(service = "SILO")
get_key(service = "METNO")

## End(Not run)
```

---

```
get_metno_daily_forecast
```

*Get Daily Weather Forecast Data from MET Weather API*

---

### Description

Retrieves daily aggregated weather forecast data from the Norwegian Meteorological Institute (MET.NO) locationforecast API. This is a convenience function that wraps `get_metno_forecast` and aggregates the hourly forecast data into daily summaries.

### Usage

```
get_metno_daily_forecast(
    latitude,
    longitude,
    days = 9,
    api_key,
    timeout = 30,
    max_retries = 3,
    retry_delay = 1,
    use_cache = TRUE,
    cache_dir = NULL,
    allow_stale_on_error = TRUE
)
```

### Arguments

<code>latitude</code>	Numeric. The latitude in decimal degrees for the forecast location. Must be within Australian limits (-44 to -10).
<code>longitude</code>	Numeric. The longitude in decimal degrees for the forecast location. Must be within Australian limits (112 to 154).
<code>days</code>	Numeric. The number of forecast days to return (1-9, default: 9).
<code>api_key</code>	Character. Your email address, required for the User-Agent header as per MET Weather API terms of service.
<code>timeout</code>	Numeric. Request timeout in seconds (default: 30).
<code>max_retries</code>	Numeric. Maximum number of retry attempts for failed requests (default: 3).
<code>retry_delay</code>	Numeric. Base delay between retries in seconds (default: 1.0).
<code>use_cache</code>	Logical. Use session-scoped cache and conditional revalidation for MET.NO responses. Defaults to TRUE.
<code>cache_dir</code>	Character. Optional directory path for cache files. If NULL (default), uses <code>file.path(tempdir(), "metno_cache")</code> .
<code>allow_stale_on_error</code>	Logical. If TRUE (default), return stale cached data when MET.NO returns HTTP 429 (rate limit exceeded) and stale cache is available.

## Details

The latitude and longitude are truncated to 4 decimal places as per MET Weather API Terms of Service. The daily aggregation includes minimum and maximum temperatures, total precipitation, average and maximum wind speeds, average relative humidity, average air pressure, average cloud fraction, and a dominant weather symbol for the day.

## Value

A data.table with daily aggregated weather forecasts, including: date, min\_temperature, max\_temperature, total\_precipitation, avg\_wind\_speed, max\_wind\_speed, avg\_relative\_humidity, avg\_pressure, avg\_cloud\_fraction, and dominant\_weather\_symbol.

## Author(s)

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>

## See Also

Other METNO: [get\\_metno\\_forecast\(\)](#), [metno\\_get\\_dominant\\_symbol\(\)](#), [metno\\_resample\\_data\\_table\(\)](#), [metno\\_timeseries\\_to\\_data\\_table\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

## Examples

```
## Not run:
# Example of how to use the function (replace with your actual email)
# daily_forecast <- get_metno_daily_forecast(
#   latitude = -27.5,
#   longitude = 153.0,
#   days = 7,
#   api_key = "your.email@example.com"
# )
# print(daily_forecast)

## End(Not run)
```

---

get\_metno\_forecast

*Get Weather Forecast Data from MET Weather API*

---

## Description

Retrieves weather forecast data from the Norwegian Meteorological Institute locationforecast API and returns the parsed metadata together with a tidy hourly data.table. The response headers Expires and Last-Modified are provided both in their raw RFC 1123 form and parsed to POSIXct for downstream logic.

**Usage**

```
get_metno_forecast(
  latitude,
  longitude,
  format = c("compact", "complete"),
  api_key = get_key(service = "METNO"),
  timeout = 30,
  max_retries = 3,
  retry_delay = 1,
  use_cache = TRUE,
  cache_dir = NULL,
  allow_stale_on_error = TRUE
)
```

**Arguments**

latitude	Numeric. Latitude in decimal degrees for the forecast location. Must be within Australian limits (-44 to -10).
longitude	Numeric. Longitude in decimal degrees for the forecast location. Must be within Australian limits (112 to 154).
format	Character. Either "compact" (default) or "complete" for the MET Weather API locationforecast endpoint variant.
api_key	Character. Email address required for the User-Agent header in accordance with MET Weather API terms of service.
timeout	Numeric. Request timeout in seconds (default: 30).
max_retries	Integer. Maximum number of retry attempts on transient failures (default: 3).
retry_delay	Numeric. Base delay between retries in seconds for exponential backoff (default: 1).
use_cache	Logical. Use session-scoped cache and conditional revalidation for MET.NO responses. Defaults to TRUE.
cache_dir	Character. Optional directory path for cache files. If NULL (default), uses <code>file.path(tempdir(), "metno_cache")</code> .
allow_stale_on_error	Logical. If TRUE (default), return stale cached data when MET.NO returns HTTP 429 (rate limit exceeded) and stale cache is available.

**Value**

A named list with elements:

`data` Hourly forecast as a `data.table`.

`raw` The full parsed GeoJSON response (list).

`metadata` A list containing request parameters, status code, retrieval timestamp, header information (`expires_raw`, `expires`, `last_modified_raw`, `last_modified`), and cache provenance in `metadata$cache`.

**Author(s)**

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>

**See Also**

Other METNO: [get\\_metno\\_daily\\_forecast\(\)](#), [metno\\_get\\_dominant\\_symbol\(\)](#), [metno\\_resample\\_data\\_table\(\)](#), [metno\\_timeseries\\_to\\_data\\_table\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

**Examples**

```
## Not run:
forecast <- get_metno_forecast(
  latitude = -31.95,
  longitude = 115.86,
  api_key = "your.email@example.com"
)
forecast$metadata$expires
utils::head(forecast$data)

## End(Not run)
```

---

get\_patched\_point      *Get PatchedPoint Weather Data From SILO*

---

**Description**

Fetch nicely formatted weather data from the SILO API derived from the BOM station observations (PatchedPoint) data.

**Usage**

```
get_patched_point(
  station_code,
  start_date,
  end_date = Sys.Date(),
  values = "all",
  api_key = get_key(service = "SILO")
)
```

**Arguments**

station_code	A character string of the BOM station code for the station of interest.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.
values	A character string with the type of weather data to return. See <b>Available Values</b> for a full list of valid values. Defaults to all with all available values being returned.
api_key	A character string containing your API key, an e-mail address, for the request. Defaults to automatically detecting your key from your local .Renviron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

**Value**

a `data.table::data.table()` with the weather data queried with the weather variables in alphabetical order. The first eight columns will always be:

- station\_code,
- station\_name,
- longitude,
- latitude,
- elev\_m (elevation in metres),
- date (ISO8601 format, "YYYYMMDD"),
- year,
- month,
- day,
- extracted (the date on which the query was made)

**Column Name Details**

Column names are converted from the default returns of the API to be snake\_case formatted and where appropriate, the names of the values that are analogous between SILO and DPIRD data are named using the same name for ease of interoperability, e.g., using `rbind()` to create a `data.table` that contains data from both APIs.

The SILO documentation provides the following information for the PatchedPoint data.

*These data are a continuous daily time series of data at either recording stations or grid points across Australia:*

- *Data at station locations consists of observational records which have been supplemented by interpolated estimates when observed data are missing. Datasets are available at approximately 8,000 Bureau of Meteorology recording stations around Australia.*
- *Data at grid points consists entirely of interpolated estimates. The data are taken from the SILO gridded datasets and are available at any pixel on a  $0.05^\circ \times 0.05^\circ$  grid over the land area of Australia (including some islands).*

### Available Values

**all** Which will return all of the following values

**rain** Rainfall

**max\_temp (degrees C)** Maximum temperature

**min\_temp (degrees C)** Minimum temperature

**vp (hPa)** Vapour pressure

**vp\_deficit (hPa)** Vapour pressure deficit

**evap\_pan (mm)** Class A pan evaporation

**evap\_syn (mm)** Synthetic estimate<sup>1</sup>

**evap\_comb (mm)** Combination (synthetic estimate pre-1970, class A pan 1970 onwards)

**evap\_morton\_lake (mm)** Morton's shallow lake evaporation

**radiation (Mj/m<sup>2</sup>)** Solar exposure, consisting of both direct and diffuse components

**rh\_tmax (%)** Relative humidity at the time of maximum temperature

**rh\_tmin (%)** Relative humidity at the time of minimum temperature

**et\_short\_crop (mm)** FAO56<sup>4</sup> short crop

**et\_tall\_crop (mm)** ASCE<sup>5</sup> tall crop<sup>6</sup>

**et\_morton\_actual (mm)** Morton's areal actual evapotranspiration

**et\_morton\_potential (mm)** Morton's point potential evapotranspiration

**et\_morton\_wet (mm)** Morton's wet-environment areal potential evapotranspiration over land

**mslp (hPa)** Mean sea level pressure

### Value information

Solar radiation: total incoming downward shortwave radiation on a horizontal surface, derived from estimates of cloud oktas and sunshine duration<sup>3</sup>.

Relative humidity: calculated using the vapour pressure measured at 9am, and the saturation vapour pressure computed using either the maximum or minimum temperature<sup>6</sup>.

Evaporation and evapotranspiration: an overview of the variables provided by SILO is available here, [https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration\\_overview.pdf](https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration_overview.pdf).

**Data codes**

The data are supplied with codes indicating how each datum was obtained.

- 0** Official observation as supplied by the Bureau of Meteorology
- 15** Deaccumulated rainfall (original observation was recorded over a period exceeding the standard 24 hour observation period).
- 25** Interpolated from daily observations for that date.
- 26** Synthetic Class A pan evaporation, calculated from temperatures, radiation and vapour pressure.
- 35** Interpolated from daily observations using an anomaly interpolation method.
- 75** Interpolated from the long term averages of daily observations for that day of year.

**Author(s)**

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam Sparks, <adamhsparks@gmail.com>

**References**

1. Rayner, D. (2005). Australian synthetic daily Class A pan evaporation. Technical Report December 2005, Queensland Department of Natural Resources and Mines, Indooroopilly, Qld., Australia, 40 pp.
2. Morton, F. I. (1983). Operational estimates of areal evapotranspiration and their significance to the science and practice of hydrology, *Journal of Hydrology*, Volume 66, 1-76.
3. Zajackowski, J., Wong, K., & Carter, J. (2013). Improved historical solar radiation gridded data for Australia, *Environmental Modelling & Software*, Volume 49, 64–77. DOI: [doi:10.1016/j.envsoft.2013.06.013](https://doi.org/10.1016/j.envsoft.2013.06.013).
4. Food and Agriculture Organization of the United Nations, Irrigation and drainage paper 56: Crop evapotranspiration - Guidelines for computing crop water requirements, 1998.
5. ASCE's Standardized Reference Evapotranspiration Equation, proceedings of the National Irrigation Symposium, Phoenix, Arizona, 2000.
6. For further details refer to Jeffrey, S.J., Carter, J.O., Moodie, K.B. and Beswick, A.R. (2001). Using spatial interpolation to construct a comprehensive archive of Australian climate data, *Environmental Modelling and Software*, Volume 16/4, 309-330. DOI: [doi:10.1016/S1364-8152\(01\)00081](https://doi.org/10.1016/S1364-8152(01)00081).

**See Also**

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_stations\\_metadata\(\)](#), [silos\\_daily\\_values](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

**Examples**

```
## Not run:
# requires an API key as your email address
# Source observation data for station Wongan Hills station, WA (008137)
wd <- get_patched_point(station_code = "008137",
  start_date = "2021-06-01",
  end_date = "2021-07-01",
  values = "all",
  api_key = "your_api_key")

## End(Not run)
```

---

```
get_patched_point_apsim
```

*Get PatchedPoint Weather Data in the APSIM Format From SILO*

---

**Description**

Fetch APSIM .met file formatted weather data from the SILO API derived from the BOM station observations (PatchedPoint) data.

**Usage**

```
get_patched_point_apsim(
  station_code,
  start_date,
  end_date = Sys.Date(),
  api_key = get_key(service = "SILO")
)
```

**Arguments**

station_code	A character string or factor from <a href="#">get_stations_metadata()</a> of the BOM station code for the station of interest.
start_date	A character string or Date object representing the beginning of the range to query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date.
end_date	A character string or Date object representing the end of the range query in the format “yyyy-mm-dd” (ISO8601). Data returned is inclusive of this date. Defaults to the current system date.
api_key	A character string containing your API key, an e-mail address, for the request. Defaults to automatically detecting your key from your local .Renviron, .Rprofile or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected.

## Details

The SILO documentation provides the following information for the PatchedPoint data.

*These data are a continuous daily time series of data at either recording stations or grid points across Australia:*

- *Data at station locations consists of observational records which have been supplemented by interpolated estimates when observed data are missing. Datasets are available at approximately 8,000 Bureau of Meteorology recording stations around Australia.*
- *Data at grid points consists entirely of interpolated estimates. The data are taken from the SILO gridded datasets and are available at any pixel on a  $0.05^\circ \times 0.05^\circ$  grid over the land area of Australia (including some islands).*

## Value

An **apsimx** object of class 'met' with attributes.

## Included Values

**rain (mm)** Rainfall

**maxt (degrees C)** Maximum temperature

**mint (degrees C)** Minimum temperature

**vp (hPa)** Vapour pressure

**evap\_pan (mm)** Class A pan evaporation

**radiation (Mj/m<sup>1</sup>)** Solar exposure, consisting of both direct and diffuse components

## Value information

Solar radiation: total incoming downward shortwave radiation on a horizontal surface, derived from estimates of cloud oktas and sunshine duration<sup>2</sup>.

Evaporation and evapotranspiration: an overview of the variables provided by SILO is available here, [https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration\\_overview.pdf](https://data.longpaddock.qld.gov.au/static/publications/Evapotranspiration_overview.pdf).

## Data codes

Where the source code is a 6 digit string comprising the source code for the 6 variables. The single digit code for each variable is:

**0** an actual observation;

**1** an actual observation from a composite station;

**2** a value interpolated from daily observations;

**3** a value interpolated from daily observations using the anomaly interpolation method for CLIMARC data;

**6** a synthetic pan value; or

**7** an interpolated long term average.

## Saving objects

To save “met” objects the `apsimx::write_apsim_met()` is reexported. Note that when saving, comments from SILO will be included, but these will not be printed as a part of the resulting met object in your R session.

## Author(s)

Rodrigo Pires, <rodrigo.pires@dpird.wa.gov.au>, and Adam Sparks, <adamhsparks@gmail.com>

## References

1. Rayner, D. (2005). Australian synthetic daily Class A pan evaporation. Technical Report December 2005, Queensland Department of Natural Resources and Mines, Indooroopilly, Qld., Australia, 40 pp.
2. Morton, F. I. (1983). Operational estimates of areal evapotranspiration and their significance to the science and practice of hydrology, *Journal of Hydrology*, Volume 66, 1-76.

## See Also

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_stations\\_metadata\(\)](#), [silos\\_daily\\_values](#)

Other APSIM: [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [reexports](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

## Examples

```
## Not run:
# requires an API key as your email address
# Source observation data for Wongan Hills station, WA (008137)
wd <- get_patched_point_apsim(
  station_code = "008137",
  start_date = "20220101",
  end_date = "20221231",
  api_key = "your_api_key"
)

## End(Not run)
```

---

get\_precis\_forecast    *Get a BOM Daily Précis Forecast*

---

## Description

Fetch nicely formatted daily précis forecast from the BOM, which contains seven-day town forecasts for a specified state or territory. Ported from **bomrang**.

## Usage

```
get_precis_forecast(state = "AUS")
```

## Arguments

state	Australian state or territory as full name or postal code. Fuzzy string matching via <code>base::agrep()</code> is done. Defaults to AUS returning all state bulletins, see Details for more.
-------	---

## Details

Allowed state and territory postal codes, only one state per request or all using 'AUS'.

**AUS** Australia, returns forecast for all states, NT and ACT

**ACT** Australian Capital Territory (will return NSW)

**NSW** New South Wales

**NT** Northern Territory

**QLD** Queensland

**SA** South Australia

**TAS** Tasmania

**VIC** Victoria

**WA** Western Australia

## Value

A `data.table::data.table()` of an Australia BOM précis seven day forecasts for BOM selected towns.

## Author(s)

Adam H. Sparks, <adamhsparks@gmail.com>, Keith Pembleton, <keith.pembleton@usq.edu.au>, and Paul Melloy, <paul@melloy.com.au>

## References

Forecast data come from Australian Bureau of Meteorology (BOM) Weather Data Services

<https://www.bom.gov.au/catalogue/data-feeds.shtml>

Location data and other metadata for towns come from the BOM anonymous FTP server with spatial data

<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/>, specifically the DBF file portion of a shape-file,

<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/IDM00013.dbf>.

## See Also

[parse\\_precis\\_forecast](#)

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_coastal\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#), [parse\\_precis\\_forecast\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#), [get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#)

## Examples

```
# get the short forecast for Western Australia
get_precis_forecast(state = "WA")
```

---

```
get_radar_imagery      Get BOM Radar Imagery
```

---

## Description

Fetch BOM radar imagery from <ftp://ftp.bom.gov.au/anon/gen/radar/> and return a **magick** image object. Files available are the most recent radar snapshot which are updated approximately every 6 to 10 minutes. It is suggested to check file availability first by using [get\\_available\\_radar\(\)](#).

## Usage

```
get_radar_imagery(product_id, path = NULL, download_only = FALSE)
```

## Arguments

product_id	Character. BOM product ID to download and import. Value is required.
path	Character. A character string with the name where the downloaded file is saved. If not provided, the default value NULL is used which saves the file in an R session temp directory.
download_only	Boolean. Whether the radar image is loaded into the environment as a <b>magick</b> object or just downloaded. Defaults to FALSE

**Details**

Valid BOM RADAR Product IDs for radar imagery can be obtained from `get_available_radar()`.

**Value**

A **magick** object of the most recent RADAR image snapshot published by the BOM. If `download_only = TRUE` there will be a NULL return value with the download path printed in the console as a message.

**Author(s)**

Dean Marchiori, <deanmarchiori@gmail.com>

**References**

Australian Bureau of Meteorology (BOM) radar images  
<https://www.bom.gov.au/weather-and-climate/rain-radar-and-weather-maps>

**See Also**

`get_available_radar()`

Other BOM: `find_forecast_towns()`, `get_available_imagery()`, `get_available_radar()`, `get_coastal_forecast()`, `get_precis_forecast()`, `get_satellite_imagery()`, `parse_coastal_forecast()`, `parse_precis_forecast()`

Other data fetching: `get_coastal_forecast()`, `get_data_drill()`, `get_data_drill_apsim()`, `get_dpird_apsim()`, `get_dpird_extremes()`, `get_dpird_minute()`, `get_dpird_summaries()`, `get_metno_daily_forecast()`, `get_metno_forecast()`, `get_patched_point()`, `get_patched_point_apsim()`, `get_precis_forecast()`, `get_satellite_imagery()`

**Examples**

```
# Fetch most recent radar image for Wollongong 256km radar
imagery <- get_radar_imagery(product_id = "IDR032")
imagery
```

---

`get_satellite_imagery` *Get BOM Satellite Imagery*

---

**Description**

Fetch BOM satellite GeoTIFF imagery from `ftp://ftp.bom.gov.au/anon/gen/gms/` and return a **terra** SpatRaster S4 class (see `[terra::rast()]`) or **stars** S3 stars object of GeoTIFF files. Files are available at ten minutes update frequency with a 24-hour delete time. It is suggested to check file availability first by using `get_available_imagery()`. Ported from **bomrang** with modifications.

**Usage**

```
get_satellite_imagery(product_id, scans = 1, compat = "terra")
```

**Arguments**

product_id	Character. BOM product ID to download and import as a <b>terra</b> SpatRaster S4 class (see [terra::rast()]) or <b>stars</b> S3 stars class object. A vector of values from <a href="#">get_available_imagery()</a> may be used here. Value is required.
scans	Integer. Number of scans to download, starting with most recent and progressing backwards, <i>e.g.</i> , 1 - the most recent single scan available, 6 - the most recent hour available, 12 - the most recent 2 hours available, etc. Negating will return the oldest files first. Defaults to 1. Value is optional.
compat	Character. A string indicating the R package with which the returned imagery should be formatted for use, one of terra or stars. Defaults to terra.

**Details**

Valid BOM satellite Product IDs for use with *product\_id* include:

- IDE00420** AHI cloud cover only 2km FD GEOS GIS
- IDE00421** AHI IR (Ch13) greyscale 2km FD GEOS GIS
- IDE00422** AHI VIS (Ch3) greyscale 2km FD GEOS GIS
- IDE00423** AHI IR (Ch13) Zehr 2km FD GEOS GIS
- IDE00425** AHI VIS (true colour) / IR (Ch13 greyscale) composite 1km FD GEOS GIS
- IDE00426** AHI VIS (true colour) / IR (Ch13 greyscale) composite 2km FD GEOS GIS
- IDE00427** AHI WV (Ch8) 2km FD GEOS GIS
- IDE00430** AHI cloud cover only 2km AUS equirect. GIS
- IDE00431** AHI IR (Ch13) greyscale 2km AUS equirect. GIS
- IDE00432** AHI VIS (Ch3) greyscale 2km AUS equirect. GIS
- IDE00433** AHI IR (Ch13) Zehr 2km AUS equirect. GIS
- IDE00435** AHI VIS (true colour) / IR (Ch13 greyscale) composite 1km AUS equirect. GIS
- IDE00436** AHI VIS (true colour) / IR (Ch13 greyscale) composite 2km AUS equirect. GIS
- IDE00437** AHI WV (Ch8) 2km AUS equirect. GIS
- IDE00439** AHI VIS (Ch3) greyscale 0.5km AUS equirect. GIS

**Value**

A **terra** SpatRaster S4 class (see [terra::rast()]) or **stars** S3 stars class object as selected by the user by specifying compat of GeoTIFF images with layers named by BOM product ID, timestamp and band.

**Note**

The original **bomrang** version of this function supported local file caching using **hoardr**. This version does not support this functionality any longer due to issues with CRAN and **hoardr**.

**Author(s)**

Adam H. Sparks, <adamhsparks@gmail.com>

**References**

Australian Bureau of Meteorology (BOM) high-definition satellite images  
<https://www.bom.gov.au/australia/satellite/index.shtml>.

**See Also**

[get\\_available\\_imagery\(\)](#)

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#),  
[get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#),  
[parse\\_precis\\_forecast\(\)](#)

Other data fetching: [get\\_coastal\\_forecast\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#),  
[get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#),  
[get\\_metno\\_daily\\_forecast\(\)](#), [get\\_metno\\_forecast\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#),  
[get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#)

**Examples**

```
# Fetch AHI VIS (true colour) / IR (Ch13 greyscale) composite 1km FD
# GEOS GIS {terra} `SpatRaster` object for most recent single scan
  available

imagery <- get_satellite_imagery(product_id = "IDE00425", scans = 1)
plot(imagery)

# Get a list of available image files and use that to specify files for
# download, downloading the two most recent files available

avail <- get_available_imagery(product_id = "IDE00425")
imagery <- get_satellite_imagery(product_id = avail, scans = 2)
plot(imagery)
```

---

get\_stations\_metadata *Get Weather Station Metadata for Both DPIRD and SILO Weather Stations*

---

**Description**

Download the latest station locations and metadata for stations in the SILO and DPIRD networks. For BOM stations that exist in SILO, but lack metadata from BOM, the rows will exist to indicate that the station is in the SILO data set, but there is no corresponding BOM metadata available.

**Usage**

```
get_stations_metadata(
  station_code = NULL,
  station_name = NULL,
  which_api = "all",
  api_key = NULL,
  include_closed = FALSE,
  rich = FALSE
)
```

**Arguments**

- |                |  |
|----------------|--|
| station_code   | An optional value that should be provided as a single string value or character vector of station codes for which to return metadata. If this or station_name are not provided, all station metadata is returned by default. If this and station_name are both provided, this takes precedence and values corresponding to this input will be returned.  |
| station_name   | An optional value that should be provided as either a single string or character vector of station names for which to return metadata. Fuzzy matching is used, <i>e.g.</i> , using <code>c("brisbane", "melbourne")</code> will return rows for "Brisbane", "Brisbane Aero", "Mt Brisbane", "City of Melbourne Bay", "Selbourne Kirnbrae", "Maroondah Weir Melbourne Water", "Melbourne Airport" and "Melbourne Botanical Gardens" station_name values. If this or station_code are not provided, all station metadata is returned by default. Using station_code will always override this argument if both are provided. |
| which_api      | A string value that indicates which API to use. Valid values are <code>all</code> , for both SILO (BOM data) and DPIRD APIs; <code>silo</code> for only stations from the SILO API (BOM data); or <code>dpird</code> for stations from the DPIRD Weather 2.0 API. Defaults to <code>all</code> .   |
| api_key        | A character string containing your API key from DPIRD, <a href="https://www.dpird.wa.gov.au/online-tools/apis/">https://www.dpird.wa.gov.au/online-tools/apis/</a> , for the DPIRD Weather 2.0 API. If left as <code>NULL</code> , defaults to automatically detecting your key from your local <code>.Renvirom</code> , <code>.Rprofile</code> or similar. Alternatively, you may directly provide your key as a string here. If nothing is provided, you will be prompted on how to set up your R session so that it is auto-detected. Only used when <i>which_api</i> is <code>DPIRD</code> or <code>all</code> .       |
| include_closed | A Boolean string indicating whether to include closed stations' metadata. Use <code>TRUE</code> to include these. Defaults to <code>FALSE</code> .   |
| rich           | A Boolean string indicating whether to return rich information about DPIRD's weather station(s), this does not affect the SILO stations' metadata, the variables for these observations will be <code>NA</code> . Defaults to <code>FALSE</code> .   |

**Value**

A `data.table::data.table()` of BOM weather stations' metadata for stations available from SILO and weather stations' metadata for stations available from DPIRD's Weather 2.0 API with the following columns sorted by state and station\_name.

<b>station_code:</b>	Unique station code. factor
<b>station_name:</b>	Unique station name. character
<b>start:</b>	Date observations start. date
<b>end:</b>	Date observations end. date
<b>latitude:</b>	Latitude in decimal degrees. numeric
<b>longitude:</b>	Longitude in decimal degrees. numeric
<b>state:</b>	State in which the station is located. character
<b>elev_m:</b>	Station elevation in metres. numeric
<b>source:</b>	Organisation responsible for the data or station maintenance. character
<b>include_closed:</b>	Station include_closed, one of 'open' or 'closed'. character
<b>wmo:</b>	World Meteorological Organisation, (WMO), number if applicable. numeric
<b>rich values</b>	
<b>capabilities:</b>	a list of the station's capabilities (data that it records). character
<b>probe_height:</b>	temperature probe height in metres. double
<b>rain_gauge_height</b>	rain gauge height in metres. double
<b>wind_probe_heights:</b>	wind probe heights always 3 metres, although some have 10 metre probes. integer

### Note

For stations in the SILO API, BOM does not report the exact date on which stations opened or closed, only the year. Therefore the `start` and `end` columns will indicate January 1 of the year that a station opened or closed, whereas stations in the DPIRD network have the date to the day. For BOM stations that are closed for the current year, this indicates that the station closed sometime during the current year prior to the request being made. NA in the current year indicates a station is still open.

There are discrepancies between the BOM's official station metadata, *e.g.* longitude and latitude values and SILO metadata. In these cases, the BOM metadata is used as it is considered to be the authority on the stations' locations.

The station names are returned by both APIs in full caps. For purposes of cleaner graphs and maps where these data may be used, this function converts them to proper name formats/title case with the first letter of every word capitalised excepting words like "at" or "on" and keeps acronyms like "AWS" or "PIRSA" or state abbreviations in the station names as all caps.

### Author(s)

Adam H. Sparks, <adamhsparks@gmail.com>

### References

Station location and other metadata are sourced from the Australian Bureau of Meteorology (BOM) webpage, Bureau of Meteorology Site Numbers: <https://www.bom.gov.au/climate/cdo/about/site-num.shtml> and [https://www.bom.gov.au/climate/data/lists\\_by\\_element/stations.txt](https://www.bom.gov.au/climate/data/lists_by_element/stations.txt) and the DPIRD Weather 2.0 API.

### See Also

Other DPIRD: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_dpird\\_apsim\(\)](#), [get\\_dpird\\_availability\(\)](#), [get\\_dpird\\_extremes\(\)](#), [get\\_dpird\\_minute\(\)](#), [get\\_dpird\\_summaries\(\)](#)

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [silos\\_daily\\_values](#)

Other metadata: [find\\_forecast\\_towns\(\)](#), [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_dpird\\_availability\(\)](#)

## Examples

```
## Not run:
# fetch metadata for all stations available in {weatherOz}
get_stations_metadata(api_key = "your_api_key")

## End(Not run)
```

---

parse\_coastal\_forecast

*Parse BOM Coastal Waters Forecast XML Files*

---

## Description

Parse local BOM daily coastal waters forecast XML file(s) for a specified state or territory or all of Australia.

## Usage

```
parse_coastal_forecast(state, filepath)
```

## Arguments

state	Required value of an Australian state or territory as full name or postal code. Fuzzy string matching via <code>base::agrep()</code> is done.
filepath	A string providing the directory location of the coastal forecast file(s) to parse. See Details for more.

## Details

Allowed state and territory postal codes, only one state per request or all using AUS.

**AUS** Australia, returns forecast for all states, NT and ACT

**ACT** Australian Capital Territory (will return NSW)

**NSW** New South Wales

**NT** Northern Territory

**QLD** Queensland

**SA** South Australia

**TAS** Tasmania

**VIC** Victoria

**WA** Western Australia

The *filepath* argument will only accept a directory where files are located for parsing. DO NOT supply the full path including the file name. This function will only parse the requested state or all of Australia in the same fashion as `get_coastal_forecast()`, provided that the files are all present in the directory.

**Value**

A `data.table::data.table()` of an Australia BOM Coastal Waters Forecast.

**Author(s)**

Dean Marchiori, <deanmarchiori@gmail.com>, and Paul Melloy, <paul@melloy.com.au>

**References**

Forecast data come from Australian Bureau of Meteorology (BOM) Weather Data Services  
<https://www.bom.gov.au/catalogue/data-feeds.shtml>.

Location data and other metadata come from the BOM anonymous FTP server with spatial data  
<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/>, specifically the DBF file portion of a shape-file,  
<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/IDM00003.dbf>.

**See Also**

[get\\_coastal\\_forecast](#)

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#),  
[get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#),  
[parse\\_precis\\_forecast\(\)](#)

Other parse: [metno\\_get\\_dominant\\_symbol\(\)](#), [metno\\_resample\\_data\\_table\(\)](#), [metno\\_timeseries\\_to\\_data\\_table\(\)](#),  
[parse\\_precis\\_forecast\(\)](#)

**Examples**

```
# parse the coastal forecast for Queensland

#download to tempfile() using basename() to keep original name
utils::download.file(url = "ftp://ftp.bom.gov.au/anon/gen/fwo/IDQ11290.xml",
  destfile = file.path(tempdir(),
    basename("ftp://ftp.bom.gov.au/anon/gen/fwo/IDQ11290.xml")),
  mode = "wb")

parse_coastal_forecast(state = "QLD", filepath = tempdir())
```

---

parse\_precis\_forecast *Parse BOM Précis Forecast XML Files*

---

### Description

Parse local BOM daily précis forecast XML file(s) of the seven-day town forecasts for a specified state or territory or all Australia. Ported from **bomrang**.

### Usage

```
parse_precis_forecast(state, filepath)
```

### Arguments

state	Required value of an Australian state or territory as full name or postal code. Fuzzy string matching via <code>base::agrep()</code> is done.
filepath	A string providing the directory location of the précis file(s) to parse. See Details for more.

### Details

Allowed state and territory postal codes, only one state per request or all using 'AUS'.

**ACT** Australian Capital Territory (will return NSW)

**NSW** New South Wales

**NT** Northern Territory

**QLD** Queensland

**SA** South Australia

**TAS** Tasmania

**VIC** Victoria

**WA** Western Australia

**AUS** Australia, returns forecast for all states, NT and ACT

The *filepath* argument will only accept a directory where files are located for parsing. DO NOT supply the full path including the file name. This function will only parse the requested state or all of Australia in the same fashion as `get_precis_forecast()`, provided that the files are all present in the directory.

### Value

A `data.table::data.table()` of Australia BOM précis seven-day forecasts for BOM selected towns.

### Author(s)

Adam H. Sparks, <adamhsparks@gmail.com>, and Keith Pembleton, <keith.pembleton@usq.edu.au>, and Paul Melloy, <paul@melloy.com.au>

**References**

Forecast data come from Australian Bureau of Meteorology (BOM) Weather Data Services  
<https://www.bom.gov.au/catalogue/data-feeds.shtml>

Location data and other metadata for towns come from the BOM anonymous FTP server with spatial data

<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/>, specifically the DBF file portion of a shapefile,

<ftp://ftp.bom.gov.au/anon/home/adfd/spatial/IDM00013.dbf>

**See Also**

[get\\_precis\\_forecast](#)

Other BOM: [find\\_forecast\\_towns\(\)](#), [get\\_available\\_imagery\(\)](#), [get\\_available\\_radar\(\)](#), [get\\_coastal\\_forecast\(\)](#), [get\\_precis\\_forecast\(\)](#), [get\\_radar\\_imagery\(\)](#), [get\\_satellite\\_imagery\(\)](#), [parse\\_coastal\\_forecast\(\)](#)

Other parse: [metno\\_get\\_dominant\\_symbol\(\)](#), [metno\\_resample\\_data\\_table\(\)](#), [metno\\_timeseries\\_to\\_data\\_table\(\)](#), [parse\\_coastal\\_forecast\(\)](#)

**Examples**

```
# parse the short forecast for Western Australia

# download to tempfile() using basename() to keep original name
utils::download.file(url = "ftp://ftp.bom.gov.au/anon/gen/fwo/IDQ11295.xml",
  destfile = file.path(tempdir(),
    basename("ftp://ftp.bom.gov.au/anon/gen/fwo/IDQ11295.xml")),
  mode = "wb")

parse_precis_forecast(state = "QLD", filepath = tempdir())
```

---

silo\_daily\_values      *A List of SILO Daily Weather Values*

---

**Description**

A [vector](#) object containing 18 items representing valid values to supply to [get\\_patched\\_point\(\)](#) and [get\\_data\\_drill\(\)](#)'s *values* argument taken from the documentation for the SILO API.

**Usage**

```
silo_daily_values
```

**Format**

A [vector](#) object of 57 items.

**Source**

<https://www.longpaddock.qld.gov.au/silo/about/climate-variables/>

**See Also**

Other SILO: [find\\_nearby\\_stations\(\)](#), [find\\_stations\\_in\(\)](#), [get\\_data\\_drill\(\)](#), [get\\_data\\_drill\\_apsim\(\)](#), [get\\_patched\\_point\(\)](#), [get\\_patched\\_point\\_apsim\(\)](#), [get\\_stations\\_metadata\(\)](#)

Other data: [dpird\\_extreme\\_weather\\_values](#), [dpird\\_minute\\_values](#), [dpird\\_summary\\_values](#)

---

south\_west\_agriculture\_region

*Western Australia Southwest Agriculture Region Geospatial Polygon*

---

**Description**

An **sf** object of the the WA South West Agricultural Region.

**Usage**

```
south_west_agricultural_region
```

**Format**

An `sf::sf()` polygon object

**Details**

The zone managed for intensive agricultural activities in South-Western Australia. Also known as the South West Agricultural Area or Clearing Line. This zone defines the easternmost extent of land cleared for agricultural purposes.

**Base data sets**

Western Australian Land Information Authority - Captured from photographic interpretation of best available orthophotography at date of capture, dates range between 2007 and 2010.

**Scale of capture**

1:20,000

**Coordinate Reference System**

EPSG:4326 - WGS 84 – WGS84 - World Geodetic System 1984, used in GPS <https://epsg.io/4326>

**Source**

Western Australia Department of Primary Industries and Regional Development under a [Creative Commons Attribution 4.0 Licence](#)

# Index

- \* **API**
  - get\_metno\_daily\_forecast, 36
  - get\_metno\_forecast, 37
- \* **APSIM**
  - get\_data\_drill\_apsim, 19
  - get\_dpird\_apsim, 21
  - get\_patched\_point\_apsim, 43
- \* **BOM**
  - find\_forecast\_towns, 6
  - get\_available\_imagery, 11
  - get\_available\_radar, 13
  - get\_coastal\_forecast, 14
  - get\_precis\_forecast, 46
  - get\_radar\_imagery, 47
  - get\_satellite\_imagery, 48
  - parse\_coastal\_forecast, 53
  - parse\_precis\_forecast, 55
- \* **DPIRD**
  - dpird\_extreme\_weather\_values, 4
  - dpird\_minute\_values, 4
  - dpird\_summary\_values, 5
  - find\_nearby\_stations, 7
  - find\_stations\_in, 9
  - get\_dpird\_apsim, 21
  - get\_dpird\_availability, 23
  - get\_dpird\_extremes, 25
  - get\_dpird\_minute, 28
  - get\_dpird\_summaries, 30
  - get\_stations\_metadata, 50
- \* **METNO**
  - get\_metno\_daily\_forecast, 36
  - get\_metno\_forecast, 37
- \* **SILO**
  - find\_nearby\_stations, 7
  - find\_stations\_in, 9
  - get\_data\_drill, 15
  - get\_data\_drill\_apsim, 19
  - get\_patched\_point, 39
  - get\_patched\_point\_apsim, 43
  - get\_stations\_metadata, 50
  - silodaily\_values, 56
- \* **data fetching**
  - get\_coastal\_forecast, 14
  - get\_data\_drill, 15
  - get\_data\_drill\_apsim, 19
  - get\_dpird\_apsim, 21
  - get\_dpird\_extremes, 25
  - get\_dpird\_minute, 28
  - get\_dpird\_summaries, 30
  - get\_metno\_daily\_forecast, 36
  - get\_metno\_forecast, 37
  - get\_patched\_point, 39
  - get\_patched\_point\_apsim, 43
  - get\_precis\_forecast, 46
  - get\_radar\_imagery, 47
  - get\_satellite\_imagery, 48
- \* **datasets**
  - dpird\_extreme\_weather\_values, 4
  - dpird\_minute\_values, 4
  - dpird\_summary\_values, 5
  - silodaily\_values, 56
  - southwest\_agriculture\_region, 57
- \* **data**
  - dpird\_extreme\_weather\_values, 4
  - dpird\_minute\_values, 4
  - dpird\_summary\_values, 5
  - silodaily\_values, 56
- \* **metadata**
  - find\_forecast\_towns, 6
  - find\_nearby\_stations, 7
  - find\_stations\_in, 9
  - get\_available\_imagery, 11
  - get\_available\_radar, 13
  - get\_dpird\_availability, 23
  - get\_stations\_metadata, 50
- \* **parse**
  - parse\_coastal\_forecast, 53
  - parse\_precis\_forecast, 55

- apsimx::write\_apsim\_met(), 20, 22, 45
- base::agrep(), 14, 46, 53, 55
- data.table::data.table(), 6, 8, 13, 14, 16, 24, 26, 29, 31, 40, 46, 51, 54, 55
- dpird\_extreme\_weather\_values, 4, 5, 8, 10, 22, 24, 27, 29, 34, 52, 57
- dpird\_minute\_values, 4, 4, 5, 8, 10, 22, 24, 27, 29, 34, 52, 57
- dpird\_summary\_values, 4, 5, 5, 8, 10, 22, 24, 27, 29, 34, 52, 57
- find\_forecast\_towns, 6, 8, 10, 12, 13, 15, 24, 47, 48, 50, 53, 54, 56
- find\_nearby\_stations, 4–6, 7, 10, 12, 13, 18, 21, 22, 24, 27, 29, 34, 42, 45, 52, 53, 57
- find\_stations\_in, 4–6, 8, 9, 12, 13, 18, 21, 22, 24, 27, 29, 34, 42, 45, 52, 53, 57
- get\_available\_imagery, 6, 8, 10, 11, 13, 15, 24, 47, 48, 50, 53, 54, 56
- get\_available\_imagery(), 48–50
- get\_available\_radar, 6, 8, 10, 12, 13, 15, 24, 47, 48, 50, 53, 54, 56
- get\_available\_radar(), 47, 48
- get\_coastal\_forecast, 6, 12, 13, 14, 18, 21, 22, 27, 29, 34, 37, 39, 42, 45, 47, 48, 50, 54, 56
- get\_coastal\_forecast(), 54
- get\_data\_drill, 8, 10, 15, 15, 21, 22, 27, 29, 34, 37, 39, 42, 45, 47, 48, 50, 53, 57
- get\_data\_drill\_apsim, 8, 10, 15, 18, 19, 22, 27, 29, 34, 37, 39, 42, 45, 47, 48, 50, 53, 57
- get\_dpird\_apsim, 4, 5, 8, 10, 15, 18, 21, 21, 24, 27, 29, 34, 37, 39, 42, 45, 47, 48, 50, 52
- get\_dpird\_availability, 4–6, 8, 10, 12, 13, 22, 23, 27, 29, 34, 52, 53
- get\_dpird\_extremes, 4, 5, 8, 10, 15, 18, 21, 22, 24, 25, 29, 34, 37, 39, 42, 45, 47, 48, 50, 52
- get\_dpird\_minute, 4, 5, 8, 10, 15, 18, 21, 22, 24, 27, 28, 34, 37, 39, 42, 45, 47, 48, 50, 52
- get\_dpird\_summaries, 4, 5, 8, 10, 15, 18, 21, 22, 24, 27, 29, 30, 37, 39, 42, 45, 47, 48, 50, 52
- get\_key, 35
- get\_metno\_daily\_forecast, 15, 18, 21, 22, 27, 29, 34, 36, 39, 42, 45, 47, 48, 50
- get\_metno\_forecast, 15, 18, 21, 22, 27, 29, 34, 37, 37, 42, 45, 47, 48, 50
- get\_patched\_point, 8, 10, 15, 18, 21, 22, 27, 29, 34, 37, 39, 39, 45, 47, 48, 50, 53, 57
- get\_patched\_point\_apsim, 8, 10, 15, 18, 21, 22, 27, 29, 34, 37, 39, 42, 43, 47, 48, 50, 53, 57
- get\_precis\_forecast, 6, 12, 13, 15, 18, 21, 22, 27, 29, 34, 37, 39, 42, 45, 46, 48, 50, 54, 56
- get\_precis\_forecast(), 55
- get\_radar\_imagery, 6, 12, 13, 15, 18, 21, 22, 27, 29, 34, 37, 39, 42, 45, 47, 47, 50, 54, 56
- get\_satellite\_imagery, 6, 12, 13, 15, 18, 21, 22, 27, 29, 34, 37, 39, 42, 45, 47, 48, 48, 54, 56
- get\_satellite\_imagery(), 11
- get\_stations\_metadata, 4–6, 8, 10, 12, 13, 18, 21, 22, 24, 27, 29, 34, 42, 45, 50, 57
- get\_stations\_metadata(), 22, 25, 28, 30, 43
- metno\_get\_dominant\_symbol, 37, 39, 54, 56
- metno\_resample\_data\_table, 37, 39, 54, 56
- metno\_timeseries\_to\_data\_table, 37, 39, 54, 56
- parse\_coastal\_forecast, 6, 12, 13, 15, 47, 48, 50, 53, 56
- parse\_precis\_forecast, 6, 12, 13, 15, 47, 48, 50, 54, 55
- reexports, 21, 22, 45
- sf::sf(), 57
- silo\_daily\_values, 4, 5, 8, 10, 18, 21, 42, 45, 53, 56
- south\_west\_agricultural\_region (south\_west\_agriculture\_region), 57
- south\_west\_agriculture\_region, 57
- vector, 4, 5, 56