Package: mregions2 (via r-universe)

September 4, 2024

Type Package

Title Access Data from Marineregions.org: Gazetteer & Data Products

Version 1.1.1

Maintainer Salvador Jesús Fernández Bejarano
<salvador.fernandez@vliz.be>

Description Explore and retrieve marine geospatial data from the Marine Regions Gazetteer <https://marineregions.org/gazetteer.php?p=webservices> and the Marine Regions Data Products <https://marineregions.org/webservices.php>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Config/testthat/edition 3

Imports checkmate, glue, httr2, magrittr, sf, utils, rdflib, ISOcodes, memoise, cli, dplyr, xml2, wrapr, methods, curl, digest

RoxygenNote 7.3.2

URL https://github.com/ropensci/mregions2, https://docs.ropensci.org/mregions2/

BugReports https://github.com/ropensci/mregions2/issues

Suggests ows4R (>= 0.3), httptest2, jsonlite, knitr, leaflet, leaflet.extras2, mapview, rmarkdown, testthat, wk, purrr, withr

Config/Needs/website jsonld

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Depends R (>= 2.10)

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

RemoteUrl https://github.com/ropensci/mregions2

RemoteRef main

RemoteSha e769c00687251e914a2192c19ee9c3845706e143

Contents

gaz_geometry . . . 2
gaz_relations . . . 4
gaz_rest . . . 5
gaz_rest_geometries . . . 5
gaz_rest_names_by_mrgid . . . 6
gaz_rest_records_by_lat_long . . . 7
gaz_rest_records_by_name . . . 8
gaz_rest_records_by_names . . . 9
gaz_rest_records_by_source . . . 10
gaz_rest_records_by_type . . . 10
gaz_rest_record_by_mrgid . . . 11
gaz_rest_relations_by_mrgid . . . 12
gaz_rest_sources . . . 13
gaz_rest_source_by_sourceid . . . 13
gaz_rest_types . . . 14
gaz_rest_wmses . . . 15
gaz_search . . . 16
gaz_search_by_source . . . 17
gaz_search_by_type . . . 18
gaz_sources . . . 19
gaz_types . . . 20
MRGID . . . 21
mrp_colnames . . . 22
mrp_col_unique . . . 23
mrp_get . . . 24
mrp_list . . . 26
mrp_ontology . . . 27
mrp_view . . . 27

Index **31**

`gaz_geometry` *Get the geometries of a Marine Regions Geo-Object*

Description

Get the geometries of a Marine Regions Geo-Object

Usage

```
gaz_geometry(x, ...)

## S3 method for class 'numeric'
gaz_geometry(x, ...)

## S3 method for class 'mr_df'
gaz_geometry(x, ...)
```

Arguments

x	object to retrieve the geometries from. Accepted: • (integer) A valid Marine Regions Gazetteer Identifier (MRGID) • A data frame retrieved with mregions2 via its functions `gaz_search()`, `gaz_search_by_source()`, `gaz_search_by_type()` or `gaz_relations()`. See details.
...	Arguments passed on to `gaz_rest_geometries` `format` (character) The preferred output format. One of: • "sfc": Simple Feature geometry object. See 'sf' • "wkt": Geometry representation as Well-Known Text • "rdf": Geometry as an object of class 'rdf". See 'rdflib' Default is "sfc" `multipart` (logical) Some Geo-Objects are compound of more than one part. • If FALSE, returns singlepart geometries (e.g. POLYGON, LINESTRING) • If TRUE (default), returns multipart geometries (e.g. MULTIPOLYGON, MULTILINESTRING)

Details

You can pass the output of most `gaz_*` functions to `gaz_geometry()` to retrieve the geometry the gazetteer entry. The data frame is then transformed into a sf::sf object.

Developer info:

This is done in the method `gaz_geometry.mr_df()`. `mr_df` is a class defined in this package to ensure the data frame passed to gaz_geometry has a variable with MRGID.

Value

A sfc object (default), a sf data frame, a WKT string or an RDF object

Examples

```
gaz_geometry(3293)
gaz_geometry(3293, format = "wkt")
gaz_geometry(3293, format = "rdf")

gaz_search(3293) |> gaz_geometry()
```

`gaz_relations` *Walk the hierarchy of the MarineRegions Gazetter given a Gazetteer MRGID or Gazetteer entries*

Description

Walk the hierarchy of the MarineRegions Gazetter given a Gazetteer MRGID or Gazetteer entries

Usage

```
gaz_relations(x, ...)

## S3 method for class 'numeric'
gaz_relations(x, ...)

## S3 method for class 'mr_df'
gaz_relations(x, ...)
```

Arguments

`x`	the object from which the relations are retrieved. Can be: • (integer) A valid Marine Regions Gazetteer Identifier (MRGID), passed to `gaz_rest_relations_by_mrgid()` • A data frame retrieved with mregions2 via its functions `gaz_search()`, `gaz_search_by_source()` or `gaz_search_by_type()`.
`...`	Arguments passed on to `gaz_rest_relations_by_mrgid` `with_geometry` (logical) Add geometries to the result data frame? Default = FALSE `direction` (character) Must be one of upper, lower, both: • `upper`: lists all parents of the record. • `lower`: lists all childs of the record. • `both`: lists parents and childs of the record (default) `type` (character) Must be one of partof, partlypartof, adjacentto, similarto, administrativepartof, influencedby, all.

Details

You can pass the output of most `gaz_*` functions to `gaz_relations()` to retrieve the related gazetteer entries

Developer info:

This is done in the method `gaz_relations.mr_df()`. `mr_df` is a class defined in this package to ensure the data frame passed to gaz_relations has a variable with MRGID.

Value

A data frame with Gazetteer entries

Examples

```
# Get the relations of the Belgian Exclusive Economic Zone
gaz_search("Belgian Exclusive Economic Zone") |> gaz_relations()

# Or using its mrgid
gaz_relations(3293)
```

`gaz_rest` *Marine Regions Gazetteer RESTful services (Documentation)*

Description

RESTful service REST (REpresentational State Transfer) is a simple stateless architecture that generally runs over HTTP.

mregions2 makes use of the RESTful API created and maintained by Marine Regions. The functions with names starting as `gaz_rest_*` perform HTTP requests to read the Marine Regions REST API. They are closer to the definition of each function in the Marine Regions REST API. All the gazetteer functions such as `gaz_search()`or `gaz_relations()` make use of these `gaz_rest_*` functions.

Value

Returns a help page: `gaz_rest` is not a function but documentation.

See Also

`gaz_rest_geometries()`, `gaz_rest_names_by_mrgid()`, `gaz_rest_record_by_mrgid()`, `gaz_rest_records_by_lat_l`
`gaz_rest_records_by_name()`, `gaz_rest_records_by_names()`, `gaz_rest_records_by_source()`,
`gaz_rest_records_by_type()`, `gaz_rest_relations_by_mrgid()`, `gaz_rest_sources()`, `gaz_rest_source_by_sourc`
`gaz_rest_types()`, `gaz_rest_wmses()`

Examples

```
?gaz_rest
```

`gaz_rest_geometries` *Get the geometries associated with a gazetteer record*

Description

Get the geometries associated with a gazetteer record

Usage

```
gaz_rest_geometries(mrgid, format = "sfc", multipart = TRUE, ...)
```

Arguments

`mrgid`	(integer) A valid Marine Regions Gazetteer Identifier (MRGID)
`format`	(character) The preferred output format. One of: • "sfc": Simple Feature geometry object. See 'sf' • "wkt": Geometry representation as Well-Known Text • "rdf": Geometry as an object of class 'rdf". See 'rdflib' Default is "sfc"
`multipart`	(logical) Some Geo-Objects are compound of more than one part. • If FALSE, returns singlepart geometries (e.g. POLYGON, LINESTRING) • If TRUE (default), returns multipart geometries (e.g. MULTIPOLYGON, MULTILINESTRING)
`...`	reserved for internal use

Value

A sfc object (default), a sf data frame, a WKT string or an RDF object

See Also

gaz_rest

Examples

```
gaz_rest_geometries(3293)
gaz_rest_geometries(3293, format = "wkt")
gaz_rest_geometries(3293, format = "rdf")
```

`gaz_rest_names_by_mrgid` *Get the names for a given MRGID*

Description

Get the names for a given MRGID

Usage

```
gaz_rest_names_by_mrgid(mrgid)
```

Arguments

`mrgid`	(integer) A valid Marine Regions Gazetteer Identifier (MRGID)

Value

a vector with all the names of a Marine Regions Gazetteer entry

See Also

gaz_rest, MRGID

Examples

```
gaz_rest_names_by_mrgid(3293)
gaz_rest_names_by_mrgid(14)
```

`gaz_rest_records_by_lat_long`

Get all gazetteer records where the geometry intersects with the given latitude and longitude

---

Description

Get all gazetteer records where the geometry intersects with the given latitude and longitude

Usage

```
gaz_rest_records_by_lat_long(
  latitude,
  longitude,
  with_geometry = FALSE,
  typeid = NULL
)
```

Arguments

`latitude`	(double) A decimal number which ranges from -90 to 90. Coordinates are assumed to be in WGS84
`longitude`	(double) A decimal number which ranges from -180 to 180. Coordinates are assumed to be in WGS84
`with_geometry`	(logical) Add geometries to the result data frame? Default = FALSE
`typeid`	(numeric) Restrict to one or more placetypeIDs. Retrieve a list of placetypeIDs with `gaz_rest_types()`

Value

A data frame with Gazetteer entries

See Also

gaz_rest

Examples

```
gaz_rest_records_by_lat_long(51.21551, 2.927)
gaz_rest_records_by_lat_long(51.21551, 2.927,
                             with_geometry = TRUE,
                             typeid = c(255, 259))
```

`gaz_rest_records_by_name`

Get Gazetteer Records for a given name

Description

Get Gazetteer Records for a given name

Usage

```
gaz_rest_records_by_name(
  name,
  with_geometry = FALSE,
  typeid = NULL,
  language = NULL,
  like = TRUE,
  fuzzy = TRUE
)
```

Arguments

`name`	(character) Term to search in the Marine Regions Gazetteer
`with_geometry`	(logical) Add geometry to the result data frame? Default = FALSE
`typeid`	(numeric) Restrict to one or more placetypeIDs. Retrieve a list of placetypeIDs with `gaz_rest_types()`
`language`	(character) Restrict to one language. Provide as a 2 digits ISO-639. See ISOcodes::ISO_639_2.
`like`	(logical) Add a '%'-sign before and after the name? (SQL LIKE function). Default = TRUE
`fuzzy`	(logical) Use Levenshtein query to find nearest matches? Default = TRUE

Value

A data frame with Gazetteer entries

See Also

gaz_rest, gaz_rest_records_by_name

Examples

```
gaz_rest_records_by_name("Belgian Exclusive Economic Zone", with_geometry = TRUE)
gaz_rest_records_by_name("Bélgica", language = "es")
gaz_rest_records_by_name("Belgium", typeid = c(350, 351))
```

`gaz_rest_records_by_names`

Get Gazetteer Records for all given names

Description

Get Gazetteer Records for all given names

Usage

```
gaz_rest_records_by_names(
  names,
  with_geometry = FALSE,
  like = TRUE,
  fuzzy = TRUE
)
```

Arguments

`names`	(character) Vector with the terms to search in the Marine Regions Gazetteer
`with_geometry`	(logical) Add geometry to the result data frame? Default = FALSE
`like`	(logical) Add a '%'-sign before and after the name? (SQL LIKE function). Default = TRUE
`fuzzy`	(logical) Use Levenshtein query to find nearest matches? Default = TRUE

Value

A data frame with Gazetteer entries

See Also

gaz_rest, gaz_rest_records_by_name

Examples

```
gaz_rest_records_by_names(
  c("Belgian Exclusive Economic Zone", "Dutch Exclusive Economic Zone")
)
```

gaz_rest_records_by_source

Retrieve Gazetteer Records by Source

Description

Retrieve Gazetteer Records by Source

Usage

```
gaz_rest_records_by_source(source, with_geometry = FALSE)
```

Arguments

source	(character) A source from gaz_rest_sources()
with_geometry	(logical) Add geometries to the result data frame? Default = FALSE

Value

A data frame with Gazetteer entries

See Also

gaz_rest

Examples

```
gaz_rest_records_by_source("ICES Ecoregions")
```

gaz_rest_records_by_type

Retrieve Gazetteer Records by Placetype

Description

Retrieve Gazetteer Records by Placetype

Usage

```
gaz_rest_records_by_type(type, with_geometry = FALSE)
```

Arguments

type	(character) The placetype from gaz_rest_types()
with_geometry	(logical) Add geometries to the result data frame? Default = FALSE

Value

A data frame with Gazetteer entries

See Also

gaz_rest, gaz_rest_types()

Examples

```
gaz_rest_records_by_type("FAO Subdivisions")
gaz_rest_records_by_type("EEZ")
```

---

gaz_rest_record_by_mrgid

Get one record for the given MRGID

Description

Get one record for the given MRGID

Usage

```
gaz_rest_record_by_mrgid(mrgid, with_geometry = FALSE, rdf = FALSE)
```

Arguments

mrgid	(integer) A valid Marine Regions Gazetteer Identifier (MRGID)
with_geometry	(logical) Add geometry to the result data frame? Default = FALSE
rdf	(logical) Return an object of class rdflib::rdf?

Value

A data frame with the Gazetteer entry

See Also

gaz_rest, MRGID

Examples

```
gaz_rest_record_by_mrgid(3293)
gaz_rest_record_by_mrgid(3293, with_geometry = TRUE)
gaz_rest_record_by_mrgid(3293, rdf = TRUE)
```

`gaz_rest_relations_by_mrgid` *Retrieve Gazetter Relations by MRGID*

Description

Retrieve Gazetter Relations by MRGID

Usage

```
gaz_rest_relations_by_mrgid(
  mrgid,
  with_geometry = FALSE,
  direction = "both",
  type = "all"
)
```

Arguments

`mrgid`	(integer) A valid Marine Regions Gazetteer Identifier (MRGID)
`with_geometry`	(logical) Add geometries to the result data frame? Default = FALSE
`direction`	(character) Must be one of upper, lower, both: • `upper`: lists all parents of the record. • `lower`: lists all childs of the record. • `both`: lists parents and childs of the record (default)
`type`	(character) Must be one of partof, partlypartof, adjacentto, similarto, administrativepartof, influencedby, all.

Value

A data frame with Gazetteer entries

See Also

List of types (Object Properties), gaz_rest, MRGID

Examples

```
gaz_rest_relations_by_mrgid(7378)
```

gaz_rest_sources *Get all the Marine Regions sources*

Description

Get all the Marine Regions sources

Usage

```
gaz_rest_sources()
```

Details

gaz_search() is a memoised function from gaz_rest_search(). See `memoise::memoise()`.

Value

a data frame with three columns:

- `sourceID`: the identifier of the source in the Marine Regions Gazetteer database.
- `source`: the name of the source.
- `sourceURL`: if available, the URL of the source.

See Also

gaz_rest, `gaz_search_by_source()`, `gaz_rest_records_by_source()`, `gaz_rest_source_by_sourceid()`

Examples

```
# This
gaz_rest_sources()

# is the same as
gaz_sources()
```

gaz_rest_source_by_sourceid *Get the name of a source by providing a sourceID*

Description

Get the name of a source by providing a sourceID

Usage

```
gaz_rest_source_by_sourceid(sourceid)
```

Arguments

sourceid	(integer) A valid sourceID

Value

a named vector with the source name and, if available, the url to the source.

See Also

gaz_rest, gaz_sources()

Examples

```
gaz_rest_source_by_sourceid(390)
gaz_rest_source_by_sourceid(657)
```

---

gaz_rest_types *Get all the place types of the Marine Regions Gazetteer*

Description

Get all the place types of the Marine Regions Gazetteer

Usage

```
gaz_rest_types()
```

Value

a data frame with three columns:

- typeID: the identifier of the place type in the Marine Regions Gazetteer database.
- type: the name of the place type.
- description: if available, the description of the place type.

See Also

gaz_rest

Examples

```
# This
gaz_rest_types()

# is the same as
gaz_types()
```

gaz_rest_wmses *Get WMS information for a given MRGID*

Description

Get WMS information for a given MRGID

Usage

```
gaz_rest_wmses(mrgid)
```

Arguments

mrgid	(integer) A valid Marine Regions Gazetteer Identifier (MRGID)

Value

a data frame with information from the WMS services including:

- value: the value to filter on
- MRGID : see MRGID
- url: the base URL of the WMS service
- namespace: see mrp_view() details
- featureType: see mrp_view() details
- featureName: see mrp_view() details

See Also

gaz_rest, MRGID, mrp_view()

Examples

```
gaz_rest_wmses(3293)
```

gaz_search *Search in the Marine Regions Gazetteer by names, MRGID or reverse geocode with a pair of WGS84 coordinates x and y*

Description

Search in the Marine Regions Gazetteer by names, MRGID or reverse geocode with a pair of WGS84 coordinates x and y

Usage

```
gaz_search(x, ...)

## S3 method for class 'character'
gaz_search(x, ...)

## S3 method for class 'numeric'
gaz_search(x, ..., y = NULL)

## S3 method for class 'sfg'
gaz_search(x, ...)

## S3 method for class 'sf'
gaz_search(x, ...)

## S3 method for class 'sfc'
gaz_search(x, ...)
```

Arguments

x	object to perform the search with. Can be: • (character) Free text search • (integer) A valid Marine Regions Gazetteer Identifier (MRGID) • (double) Longitude in WGS84 • Aditionally, you can pass objects of class sf::sf or sf::sfc with geometry of class POINT
...	Arguments passed on to gaz_rest_record_by_mrgid, gaz_rest_records_by_name, gaz_rest_records_by_names, gaz_rest_records_by_lat_long with_geometry (logical) Add geometry to the result data frame? Default = FALSE rdf (logical) Return an object of class rdflib::rdf? typeid (numeric) Restrict to one or more placetypeIDs. Retrieve a list of placetypeIDs with gaz_rest_types() language (character) Restrict to one language. Provide as a 2 digits ISO-639. See ISOcodes::ISO_639_2.

like (logical) Add a '%'-sign before and after the name? (SQL LIKE function). Default = TRUE

fuzzy (logical) Use Levenshtein query to find nearest matches? Default = TRUE

y (double) Latitude in WGS84 (Optional)

Value

A data frame with Gazetteer entries

Examples

```
# Look-up a name in the Gazetteer
gaz_search("North Sea")

# Get the entries of two known MRGID including their geometry
gaz_search(c(14, 17), with_geometry = TRUE)

# Maybe the name is in another language...
gaz_search("Noordzee", language = "nl")

# Get all the records intersecting with the longitude 51.21551 and latitude 2.927
# restricting to some placetypes
gaz_search(x = 2.927, y = 51.21551, typeid = c(255, 259))
```

---

gaz_search_by_source *Retrieve Gazetteer Records by Source*

Description

Retrieve Gazetteer Records by Source

Usage

```
gaz_search_by_source(x, ...)

## S3 method for class 'character'
gaz_search_by_source(x, ...)

## S3 method for class 'numeric'
gaz_search_by_source(x, ...)
```

Arguments

x source as free text or sourceID as integer

... Arguments passed on to gaz_rest_records_by_source

with_geometry (logical) Add geometries to the result data frame? Default = FALSE

Value

A data frame with Gazetteer entries

See Also

gaz_sources()

Examples

```
# Check out all sources
gaz_sources()

# Look up by source name
gaz_search_by_source("Gazetteer of Greenland")

# Or query by SourceID
gaz_search_by_source(386)
```

---

gaz_search_by_type *Retrieve Gazetteer Records by Placetype*

Description

Retrieve Gazetteer Records by Placetype

Usage

```
gaz_search_by_type(x, ...)

## S3 method for class 'character'
gaz_search_by_type(x, ...)

## S3 method for class 'numeric'
gaz_search_by_type(x, ...)
```

Arguments

x	A place type. Either: • (character) The name of a place type. • (integer) The typeid of a place type.
...	Arguments passed on to gaz_rest_records_by_type type (character) The placetype from gaz_rest_types() with_geometry (logical) Add geometries to the result data frame? Default = FALSE

Value

A data frame with Gazetteer entries

See Also

gaz_types()

Examples

```
# This
gaz_search_by_type("EEZ")

# is the same as
gaz_search_by_type(70)
```

---

gaz_sources *Get all the Marine Regions sources*

Description

Get all the Marine Regions sources

Usage

```
gaz_sources()
```

Details

gaz_search() is a memoised function from gaz_rest_search(). See memoise::memoise().

Value

a data frame with three columns:

- sourceID: the identifier of the source in the Marine Regions Gazetteer database.
- source: the name of the source.
- sourceURL: if available, the URL of the source.

See Also

gaz_rest, gaz_search_by_source(), gaz_rest_records_by_source(), gaz_rest_source_by_sourceid()

Examples

```
# This
gaz_rest_sources()

# is the same as
gaz_sources()
```

`gaz_types` *Get all the place types of the Marine Regions Gazetteer*

Description

Get all the place types of the Marine Regions Gazetteer

Usage

```
gaz_types()
```

Value

a data frame with three columns:

- `typeID`: the identifier of the place type in the Marine Regions Gazetteer database.
- `type`: the name of the place type.
- `description`: if available, the description of the place type.

See Also

gaz_rest

Examples

```
# This
gaz_rest_types()

# is the same as
gaz_types()
```

MRGID *Marine Regions Global Identifier or MRGID (Documentation)*

Description

Many functions of mregions2 make use of the argument `mrgid` or return data with the numeric variable `MRGID`.

But what is this identifier?

This is an unique and persistent identifier of each entry in the Marine Regions Gazetteer. This identifier consists in a URI containing a number, unique for each entry in the Marine Regions Gazetteer. The R package mregions2 uses this number in its functions, and it should be considered a synonym of the standard definition of MRGID.

See the section details for a more in depth definition of the `MRGID`

Details

From https://marineregions.org/mrgid.php :

Standards:

Place names change over time, and the same names may be used for different locations. Available gazetteers may find locations of some marine place names, but a truly global standard for marine place names is lacking. Marine Regions tries to establish for the first time a standardized list of georeferenced marine place names and marine areas. In order to preserve the identity of the marine geographic objects from the database, and to name and locate the geographic resources on the web, we promote the Marine Regions Geographic IDentifier, or the MRGID.

MRGID:

The Marine Regions Geographic IDentifier is:

- *unique* by using a URI (Uniform Resource Identifier), it's unique across the internet. Syntax `http://marineregions.org/mrgid/<number>`
- *persistent* we will never delete, nor change the concept behind an MRGID
- *resolvable* pointing your client to an MRGID will return - the reply of the webservice call `getGazetteerRecordByMRGID`, when using content negotiation (`text/turtle` or `application/ld+json`) - the webpage of the MRGID, when using a browser

For an identifier to be persistent, it requires the governing body to arrange for the identifier to be available for the long term. Use of the MRGID, as URI and persistent identifier has the commitment of the Flanders Marine Institute, issuing the identifier to maintain the http domain registration, and a strategy for managing the domain and the web servers.

Value

Returns a help page: `MRGID` is not a function but documentation.

Examples

```
?MRGID
```

`mrp_colnames` *Get the names of the columns and data type of the data product*

Description

Get the names of the columns and data type of the data product

Usage

```
mrp_colnames(layer)
```

Arguments

`layer`	(character) Identifier of the data product. See mrp_list

Details

This function becomes useful to write CQL or OGC filters that you can pass to `mrp_get()` or `mrp_view()` as it allows you to know the column names and the data types beforehand. Use it together with `mrp_col_unique()` to know all the possible values in the column name that you want to query on.

The actual description of each column is available only to the Maritime Boundaries products. See https://marineregions.org/eezattribute.php

Value

A data frame with the column names and data type in the Marine Regions data product

See Also

mrp_list to describe the list of products, `mrp_col_unique()` to get the unique values of a the columns of a data product, useful to write queries that can be passed to `mrp_get()` or `mrp_view()` via the arguments `cql_filter` or `filter`.

Examples

```
mrp_colnames("eez")
mrp_colnames("ecoregions")
```

`mrp_col_unique` *Get all the possible values of a column of a Marine Regions data product*

Description

Get all the possible values of a column of a Marine Regions data product

Usage

```
mrp_col_unique(layer, colname)

mrp_col_distinct(layer, colname)
```

Arguments

`layer`	(character) Identifier of the data product. See mrp_list
`colname`	(character) Column name in the data product. See `mrp_colnames()`

Details

This function becomes useful to write CQL or OGC filters that you can pass to `mrp_get()` or `mrp_view()` as it helps to know all the possible values in the column name that you want to query on beforehand. Use it together with `mrp_colnames()` to know the columns and data types in the data product.

Geometry columns:

Note that columns of type `geometry` are forbidden as their performance is sub-optimal and would likely crash your R session.

Value

A numeric or character vector with the unique values of a column of a Marine Regions data product.

See Also

mrp_list to describe the list of products, `mrp_colnames()` to get the names and data type of the columns of a data product, useful to write queries that can be passed to `mrp_get()` or `mrp_view()` via the arguments `cql_filter` or `filter`.

Examples

```
mrp_col_unique("ecs", "pol_type")
mrp_col_unique("ecs_boundaries", "line_type")
```

`mrp_get` *Get a data product*

Description

Get a data product

Usage

```
mrp_get(
  layer,
  path = getOption("mregions2.download_path", tempdir()),
  cql_filter = NULL,
  filter = NULL,
  count = NULL
)
```

Arguments

`layer`	(character) Identifier of the data product. See mrp_list
`path`	(character) Path to save the requests. Default is `base::tempdir()`. See details.
`cql_filter`	(character) Contextual Query Language (CQL) filter. See details.
`filter`	(character) Standard OGC filter specification. See details.
`count`	(numeric) Maximum number of features to be retrieved.

Details

This function uses WFS services to download the Marine Regions layers as ESRI Shapefiles.

Caching:

By default, the layers are downloaded to a temporal directory (`base::tempdir()`). You can provide a path in the `path` argument. But you can also set a path with `# options("mregions2.download_path" = "my/path/`

Because it is possible to add filters, each request is identified with a crc32 hash, provided with `digest::digest()` and attached to the file downloaded.

Once a layer is downloaded, it will be read from the cache during the next two weeks. To avoid this, simply delete the layers in the cache path.

Filters:

Both the Contextual Query Language (CQL) filter and the standard OGC filter specification allow to query the server before performing a request. This will boost performance as you will only retrieve the area of your interest. It is possible to query on attributes, but also perform geospatial queries. For instance, you can query a bounding box of interest.

CQL filters are possible only in geoserver. Marine Regions uses a geoserver instance to serve its data products. A tutorial on CQL filters is available in the geoserver web site.

Value

An sf object with the Marine Regions data product

See Also

mrp_list to describe the list of products, `mrp_view()` to visualize the data product in advance, `mrp_colnames()` and `mrp_col_unique()` to get the name, data type and unique values of a the columns of a data product, useful to query with the arguments `cql_filter` or `filter`

Examples

```
# Set cache path. Default is a temporal directory
options(mregions2.download_path = tempdir())

getOption("mregions2.download_path")
#> [1] "/tmp/RtmpARLgoE"

# See the list of all data products
mrp_list

# We want the Exclusive Economic Zones of Portugal. Let's first visualize the product:
mrp_view("eez")

# See all the columns on this data product
mrp_colnames("eez")

# We should query on sovereign
# See all the possible values of sovereign1, sovereign2 and sovereign3
sov1 = mrp_col_unique("eez", "sovereign1")
sov2 = mrp_col_unique("eez", "sovereign2")
sov3 = mrp_col_unique("eez", "sovereign3")

# Is Portugal a value in the sovereign1, 2 and 3?
"Portugal" %in% sov1
#> [1] TRUE

"Portugal" %in% sov2
#> [1] FALSE

"Portugal" %in% sov3
#> [1] FALSE

# Portugal is only in sovereign1. Let's write a CQL filter to get only
# the EEZs of Portugal, or those where Portugal is a party of a dispute or a joint regime
portugal_eez <- mrp_get("eez", cql_filter = "sovereign1 = 'Portugal'")

# If you perform this request again, it will be read from the cache instead
portugal_eez <- mrp_get("eez", cql_filter = "sovereign1 = 'Portugal'")
#> Cache is fresh. Reading: /tmp/RtmpARLgoE/eez-1951c8b7/eez.shp
#> (Last Modified: 2023-04-24 17:45:16)

# You can also limit the number of features to be requested
```

```
mrp_get("eez", count = 5)
```

`mrp_list` *Available data products at Marine Regions*

Description

A data frame including the name, abstract and some other relevant about each data product in Marine Regions.

Usage

```
mrp_list
```

Format

`mrp_list`**:**

A data frame with 21 rows and 7 columns:

title Data product name

namespace Workspace in geoserver

layer Identifier of the data product. Use in `mrp_get()`

license terms of use of the data products

citation preferred citation of the data products

doi ISO 26324 Digital Object Identifier)

imis Url of the data products in the Integrated Marine Information System (IMIS)

abstract Description of the data product

Details

Direct downloads are also available at: https://marineregions.org/downloads.php

Source

https://marineregions.org/sources.php https://marineregions.org/eezmethodology.php

Examples

```
mrp_list
```

mrp_ontology *Marine Regions Data Products Ontology*

Description

More information available at `vignette("mrp_ontology", package = "mregions2")`

Usage

```
mrp_ontology
```

Format

`mrp_ontology`**:**

A data frame with 374 rows and 4 columns:

layer Identifier of the data product. Use in `mrp_get()`

colname Name of the columns of each data product.

type Data type of the column.

definition Definition of the column.

Source

https://marineregions.org/sources.php https://marineregions.org/eezattribute.php

Examples

```
mrp_ontology
```

mrp_view *Visualize a Marine Regions data product without downloading.*

Description

Visualize a Marine Regions data product without downloading.

A series of helpers are available to ease the selection of the data products. Example: instead of running

```
mrp_view("eez")
```

You can use

```
mrp_view_eez()
```

Try `mrp_view_*()`with the identifier of the data product (see mrp_list)

Usage

```
mrp_view(layer, cql_filter = NULL, filter = NULL)

mrp_view_eez(...)

mrp_view_eez_boundaries(...)

mrp_view_eez_12nm(...)

mrp_view_eez_24nm(...)

mrp_view_eez_internal_waters(...)

mrp_view_eez_archipelagic_waters(...)

mrp_view_high_seas(...)

mrp_view_ecs(...)

mrp_view_ecs_boundaries(...)

mrp_view_iho(...)

mrp_view_goas(...)

mrp_view_eez_iho(...)

mrp_view_eez_land(...)

mrp_view_longhurst(...)

mrp_view_cds(...)

mrp_view_eca_reg13_nox(...)

mrp_view_eca_reg14_sox_pm(...)

mrp_view_worldheritagemarineprogramme(...)

mrp_view_lme(...)

mrp_view_ecoregions(...)

mrp_view_seavox_v18(...)
```

Arguments

layer	(character) Identifier of the data product. See mrp_list

cql_filter	(character) Contextual Query Language (CQL) filter. See details.
filter	(character) Standard OGC filter specification. See details.
...	pass the cql_filter and filter parameters to mrp_view() when using one of the helpers

Details

This function uses WMS services to load quickly a Leaflet viewer of a Marine Regions data product. It uses the EMODnet Bathymetry Digital Terrain Model as background layer.

Filters:

Both the Contextual Query Language (CQL) filter and the standard OGC filter specification allow to query the server before performing a request. This will boost performance as you will only retrieve the area of your interest. It is possible to query on attributes, but also perform geospatial queries. For instance, you can query a bounding box of interest.

CQL filters are possible only in geoserver. Marine Regions uses a geoserver instance to serve its data products. A tutorial on CQL filters is available in the geoserver web site.

Value

A leaflet map with a data product visualized via WMS

See Also

mrp_list to describe the list of products, mrp_colnames() and mrp_col_unique() to get the name, data type and unique values of a the columns of a data product, useful to query with the arguments cql_filter or filter, mrp_get() to get the data products as a simple feature object.

Examples

```
# You can pass a product name from mrp_list
mrp_view('eez')

# Or use the helper
mrp_view_eez()

# Example: filter a the Ecoregions 'Azores Canaries Madeira' with mrgid 21885
# You can check the names of the columns beforehand with mrp_colnames('ecoregions')
mrp_view_ecoregions(filter = "
  <Filter>
    <PropertyIsEqualTo>
      <PropertyName>ecoregion</PropertyName>
      <Literal>Azores Canaries Madeira</Literal>
    </PropertyIsEqualTo>
  </Filter>
")

# OGC filter are very verbose... but luckily you can use a CQL filter instead
mrp_view_ecoregions(cql_filter = "ecoregion = 'Azores Canaries Madeira'")

# View all the Extended Continental Shelf (ECS) boundary lines published during the first
```

```
# decade of the 21st century
mrp_view_ecs_boundaries(
  cql_filter = "doc_date > '2000-01-01' AND doc_date < '2009-12-31'"
)

# Or as timestamp
mrp_view_eez_boundaries(
 cql_filter = "doc_date AFTER 2000-01-01T00:00:00Z AND doc_date BEFORE 2009-12-31T00:00:00Z"
)
```

Index

∗ **datasets**
    mrp_list, 26
    mrp_ontology, 27

base::tempdir(), *24*

digest::digest(), *24*

gaz_geometry, 2
gaz_geometry.mr_df(), *3*
gaz_relations, 4
gaz_relations(), *3*
gaz_relations.mr_df(), *4*
gaz_rest, 5, *6–15*, *19, 20*
gaz_rest_geometries, *3*, 5
gaz_rest_geometries(), *5*
gaz_rest_names_by_mrgid, 6
gaz_rest_names_by_mrgid(), *5*
gaz_rest_record_by_mrgid, 11, *16*
gaz_rest_record_by_mrgid(), *5*
gaz_rest_records_by_lat_long, 7, *16*
gaz_rest_records_by_lat_long(), *5*
gaz_rest_records_by_name, *8*, 8, *9*, *16*
gaz_rest_records_by_name(), *5*
gaz_rest_records_by_names, 9, *16*
gaz_rest_records_by_names(), *5*
gaz_rest_records_by_source, 10, *17*
gaz_rest_records_by_source(), *5*, *13*, *19*
gaz_rest_records_by_type, 10, *18*
gaz_rest_records_by_type(), *5*
gaz_rest_relations_by_mrgid, *4*, 12
gaz_rest_relations_by_mrgid(), *4, 5*
gaz_rest_source_by_sourceid, 13
gaz_rest_source_by_sourceid(), *5*, *13*, *19*
gaz_rest_sources, 13
gaz_rest_sources(), *5*, *10*
gaz_rest_types, 14
gaz_rest_types(), *5*, *7, 8*, *10, 11*, *16*, *18*
gaz_rest_wmses, 15
gaz_rest_wmses(), *5*
gaz_search, 16
gaz_search(), *3, 4*
gaz_search_by_source, 17
gaz_search_by_source(), *3, 4*, *13*, *19*
gaz_search_by_type, 18
gaz_search_by_type(), *3, 4*
gaz_sources, 19
gaz_sources(), *14*, *18*
gaz_types, 20
gaz_types(), *19*

ISOcodes::ISO_639_2, *8*, *16*

memoise::memoise(), *13*, *19*
mregions2, *3–5*, *21*
MRGID, *3, 4*, *6, 7*, *11, 12*, *15, 16*, 21
mrp_col_distinct *(*mrp_col_unique*)*, 23
mrp_col_unique, 23
mrp_col_unique(), *22*, *25*, *29*
mrp_colnames, 22
mrp_colnames(), *23*, *25*, *29*
mrp_get, 24
mrp_get(), *22, 23*, *26, 27*, *29*
mrp_list, *22–25*, 26, *27–29*
mrp_ontology, 27
mrp_view, 27
mrp_view(), *15*, *22, 23*, *25*, *29*
mrp_view_cds *(*mrp_view*)*, 27
mrp_view_eca_reg13_nox *(*mrp_view*)*, 27
mrp_view_eca_reg14_sox_pm *(*mrp_view*)*, 27
mrp_view_ecoregions *(*mrp_view*)*, 27
mrp_view_ecs *(*mrp_view*)*, 27
mrp_view_ecs_boundaries *(*mrp_view*)*, 27
mrp_view_eez *(*mrp_view*)*, 27
mrp_view_eez_12nm *(*mrp_view*)*, 27
mrp_view_eez_24nm *(*mrp_view*)*, 27
mrp_view_eez_archipelagic_waters
        *(*mrp_view*)*, 27
mrp_view_eez_boundaries *(*mrp_view*)*, 27
mrp_view_eez_iho *(*mrp_view*)*, 27

mrp_view_eez_internal_waters
(mrp_view), 27
mrp_view_eez_land (mrp_view), 27
mrp_view_goas (mrp_view), 27
mrp_view_high_seas (mrp_view), 27
mrp_view_iho (mrp_view), 27
mrp_view_lme (mrp_view), 27
mrp_view_longhurst (mrp_view), 27
mrp_view_seavox_v18 (mrp_view), 27
mrp_view_worldheritagemarineprogramme
(mrp_view), 27

place type, *18*

rdflib::rdf, *11*, *16*

sf::sf, *3*, *16*
sf::sfc, *16*