Package 'CatastRo'

April 12, 2024

Title Interface to the API 'Sede Electronica Del Catastro'

Version 0.3.1

Description Access public spatial data available under the 'INSPIRE' directive. Tools for downloading references and addresses of properties, as well as map images.

License GPL-2

URL https://ropenspain.github.io/CatastRo/, https://github.com/rOpenSpain/CatastRo

BugReports https://github.com/rOpenSpain/CatastRo/issues

Depends R (>= 3.6)

Imports dplyr, httr2 (>= 1.0.0), mapSpain (>= 0.7.0), rappdirs (>= 0.3.0), sf (>= 1.0.0), stringi, terra, tibble, xml2

Suggests ggplot2, knitr, png, rmarkdown, slippymath, testthat (>= 3.0.0), tidyterra

VignetteBuilder knitr

Config/Needs/website ropenspain/rostemplate, devtools, sessioninfo, remotes, sfheaders, rapidjsonr, jsonify, geometries

Config/testthat/edition 3

Config/testthat/parallel true

Copyright © Dirección General del Catastro <https://www.catastro.meh.es/>

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Author Ángel Delgado Panadero [aut, cph] (<https://orcid.org/0000-0002-8189-9251>), Iñaki Ucar [ctb] (<https://orcid.org/0000-0001-6403-5550>), Diego Hernangómez [aut, cre] (<https://orcid.org/0000-0001-8457-4658>)

Maintainer Diego Hernangómez <diego.hernangomezherrero@gmail.com>

Repository CRAN

Date/Publication 2024-04-12 12:50:11 UTC

R topics documented:

`catr_atom_get_address` *ATOM INSPIRE: Download all the addresses of a municipality*

Description

Get the spatial data of all the addresses belonging to a single municipality using the INSPIRE ATOM service. Additionally, the function also returns the corresponding street information on the fields with the prefix `tfname_*`.

Usage

```
catr_atom_get_address(
  munic,
  to = NULL,
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

munic	Municipality to extract, It can be a part of a string or the cadastral code. See catr_atom_search_munic() for getting the cadastral codes.
to	Optional parameter for defining the Territorial Office to which munic belongs. This parameter is a helper for narrowing the search.
cache	A logical whether to do caching. Default is TRUE. See **About caching** section on catr_set_cache_dir().
update_cache	A logical whether to update cache. Default is FALSE. When set to TRUE it would force a fresh download of the source file.
cache_dir	A path to a cache directory. On NULL value (the default) the function would store the cached files on the tempdir.
verbose	Logical, displays information. Useful for debugging, default is FALSE.

Value

A sf object.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

INSPIRE API functions: catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_building
catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_wfs_get_address_bbox(),
catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE ATOM services: catr_atom_get_address_db_all(), catr_atom_get_buildings(),
catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_atom_get_parcels_db_all(),
catr_atom_search_munic()

Other addresses: catr_atom_get_address_db_all(), catr_wfs_get_address_bbox()

Other spatial: catr_atom_get_buildings(), catr_atom_get_parcels(), catr_wfs_get_address_bbox(),
catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Examples

```
s <- catr_atom_get_address("Melque",
  to = "Segovia"
)

library(ggplot2)

ggplot(s) +
  geom_sf(aes(color = specification)) +
  coord_sf(
    xlim = c(376200, 376850),
```

```
    ylim = c(4545000, 4546000)
  ) +
  labs(
    title = "Addresses",
    subtitle = "Melque de Cercos, Segovia"
  )
```

`catr_atom_get_address_db_all` *ATOM INSPIRE: Reference database for ATOM addresses*

Description

Create a database containing the urls provided in the INSPIRE ATOM service of the Spanish Cadastre for extracting Addresses.

- `catr_atom_get_address_db_all()` provides a top-level table including information of all the territorial offices (except Basque Country and Navarre) listing the municipalities included on each office.
- `catr_atom_get_address_db_to()` provides a table for the specified territorial office including information for each of the municipalities of that office.

Usage

```
catr_atom_get_address_db_all(
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)

catr_atom_get_address_db_to(
  to,
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`cache`	A logical whether to do caching. Default is `TRUE`. See **About caching** section on `catr_set_cache_dir()`.
`update_cache`	A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.

cache_dir	A path to a cache directory. On NULL value (the default) the function would store the cached files on the tempdir.
verbose	Logical, displays information. Useful for debugging, default is FALSE.
to	Territorial office. It can be any type of string, the function would perform a search using base::grep().

Value

A tibble with the information requested.

- catr_atom_get_address_db_all() includes the following fields:
  - territorial_office: Territorial office, corresponding to each province of Spain expect Basque Country and Navarre.
  - url: ATOM url for the corresponding territorial office.
  - munic: Name of the municipality.
  - date: Reference date of the data. Note that **the information of this service is updated twice a year**.
- catr_atom_get_address_db_to() includes the following fields:
  - munic: Name of the municipality.
  - url: url for downloading information of the corresponding municipality.
  - date: Reference date of the data. Note that **the information of this service is updated twice a year**.

Source

https://www.catastro.minhap.es/INSPIRE/CadastralParcels/ES.SDGC.CP.atom.xml

See Also

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_buildings(), catr_atom_get_buildings_db_al
catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_wfs_get_address_bbox(),
catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE ATOM services: catr_atom_get_address(), catr_atom_get_buildings(), catr_atom_get_buildings
catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_atom_search_munic()

Other addresses: catr_atom_get_address(), catr_wfs_get_address_bbox()

Other databases: catr_atom_get_buildings_db_all(), catr_atom_get_parcels_db_all(), catr_atom_search_munic
catr_srs_values

Examples

```
catr_atom_get_address_db_all()
```

`catr_atom_get_buildings` *ATOM INSPIRE: Download all the buildings of a municipality*

Description

Get the spatial data of all the buildings belonging to a single municipality using the INSPIRE ATOM service.

Usage

```
catr_atom_get_buildings(
  munic,
  to = NULL,
  what = c("building", "buildingpart", "other"),
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`munic`	Municipality to extract, It can be a part of a string or the cadastral code. See `catr_atom_search_munic()` for getting the cadastral codes.
`to`	Optional parameter for defining the Territorial Office to which `munic` belongs. This parameter is a helper for narrowing the search.
`what`	Information to load. It could be: • `"building"` for buildings. • `"buildingpart"` for parts of a building. • `"other"` for others elements, as swimming pools, etc.
`cache`	A logical whether to do caching. Default is `TRUE`. See **About caching** section on `catr_set_cache_dir()`.
`update_cache`	A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.
`cache_dir`	A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

Value

A `sf` object.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings_
catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_wfs_get_address_bbox(),
catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE ATOM services: catr_atom_get_address(), catr_atom_get_address_db_all(),
catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_atom_get_parcels_db_all(),
catr_atom_search_munic()

Other buildings: catr_atom_get_buildings_db_all(), catr_wfs_get_buildings_bbox()

Other spatial: catr_atom_get_address(), catr_atom_get_parcels(), catr_wfs_get_address_bbox(),
catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Examples

```
s <- catr_atom_get_buildings("Nava de la Asuncion",
  to = "Segovia",
  what = "building"
)

library(ggplot2)
ggplot(s) +
  geom_sf() +
  coord_sf(
    xlim = c(374500, 375500),
    ylim = c(4556500, 4557500)
  ) +
  labs(
    title = "Buildings",
    subtitle = "Nava de la Asuncion, Segovia"
  )
```

---

catr_atom_get_buildings_db_all

ATOM INSPIRE: Reference database for ATOM buildings

Description

Create a database containing the urls provided in the INSPIRE ATOM service of the Spanish Cadastre for extracting buildings.

- `catr_atom_get_buildings_db_all()` provides a top-level table including information of all the territorial offices (except Basque Country and Navarre) listing the municipalities included on each office.
- `catr_atom_get_buildings_db_to()` provides a table for the specified territorial office including information for each of the municipalities of that office.

Usage

```
catr_atom_get_buildings_db_all(
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)

catr_atom_get_buildings_db_to(
  to,
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`cache`	A logical whether to do caching. Default is `TRUE`. See **About caching** section on `catr_set_cache_dir()`.
`update_cache`	A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.
`cache_dir`	A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.
`to`	Territorial office. It can be any type of string, the function would perform a search using `base::grep()`.

Value

A `tibble` with the information requested.

- `catr_atom_get_buildings_db_all()` includes the following fields:
  - `territorial_office`: Territorial office, corresponding to each province of Spain expect Basque Country and Navarre.
  - `url`: ATOM url for the corresponding territorial office.
  - `munic`: Name of the municipality.
  - `date`: Reference date of the data. Note that **the information of this service is updated twice a year**.
- `catr_atom_get_buildings_db_to()` includes the following fields:

- `munic`: Name of the municipality.
- `url`: url for downloading information of the corresponding municipality.
- `date`: Reference date of the data. Note that **the information of this service is updated twice a year**.

Source

https://www.catastro.minhap.es/INSPIRE/CadastralParcels/ES.SDGC.CP.atom.xml

See Also

INSPIRE API functions: `catr_atom_get_address()`, `catr_atom_get_address_db_all()`, `catr_atom_get_buildings()`, `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`, `catr_wfs_get_address_bbox()`, `catr_wfs_get_buildings_bbox()`, `catr_wfs_get_parcels_bbox()`, `catr_wms_get_layer()`

Other INSPIRE ATOM services: `catr_atom_get_address()`, `catr_atom_get_address_db_all()`, `catr_atom_get_buildings()`, `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`, `catr_atom_search_munic()`

Other buildings: `catr_atom_get_buildings()`, `catr_wfs_get_buildings_bbox()`

Other databases: `catr_atom_get_address_db_all()`, `catr_atom_get_parcels_db_all()`, `catr_atom_search_munic()`, `catr_srs_values`

Examples

```
catr_atom_get_buildings_db_all()
```

---

catr_atom_get_parcels *ATOM INSPIRE: Download all the cadastral parcels of a municipality*

Description

Get the spatial data of all the cadastral parcels belonging to a single municipality using the INSPIRE ATOM service.

Usage

```
catr_atom_get_parcels(
  munic,
  to = NULL,
  what = "parcel",
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

munic	Municipality to extract, It can be a part of a string or the cadastral code. See catr_atom_search_munic() for getting the cadastral codes.
to	Optional parameter for defining the Territorial Office to which munic belongs. This parameter is a helper for narrowing the search.
what	Information to load. It could be "parcel" for cadastral parcels or "zoning" for cadastral zoning.
cache	A logical whether to do caching. Default is TRUE. See **About caching** section on catr_set_cache_dir().
update_cache	A logical whether to update cache. Default is FALSE. When set to TRUE it would force a fresh download of the source file.
cache_dir	A path to a cache directory. On NULL value (the default) the function would store the cached files on the tempdir.
verbose	Logical, displays information. Useful for debugging, default is FALSE.

Value

A sf object.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels_db_all(), catr_wfs_get_address_bbox(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE ATOM services: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels_db_all(), catr_atom_search_munic()

Other parcels: catr_atom_get_parcels_db_all(), catr_wfs_get_parcels_bbox()

Other spatial: catr_atom_get_address(), catr_atom_get_buildings(), catr_wfs_get_address_bbox(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Examples

```
s <- catr_atom_get_parcels("Melque",
  to = "Segovia",
  what = "parcel"
)

library(ggplot2)
```

```
ggplot(s) +
  geom_sf() +
  labs(
    title = "Cadastral Zoning",
    subtitle = "Melque de Cercos, Segovia"
  )
```

`catr_atom_get_parcels_db_all` *ATOM INSPIRE: Reference database for ATOM cadastral parcels*

Description

Create a database containing the urls provided in the INSPIRE ATOM service of the Spanish Cadastre for extracting cadastral parcels.

- `catr_atom_get_parcels_db_all()` provides a top-level table including information of all the territorial offices (except Basque Country and Navarre) listing the municipalities included on each office.
- `catr_atom_get_parcels_db_to()` provides a table for the specified territorial office including information for each of the municipalities of that office.

Usage

```
catr_atom_get_parcels_db_all(
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)

catr_atom_get_parcels_db_to(
  to,
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`cache`	A logical whether to do caching. Default is `TRUE`. See **About caching** section on `catr_set_cache_dir()`.
`update_cache`	A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.

cache_dir	A path to a cache directory. On NULL value (the default) the function would store the cached files on the tempdir.
verbose	Logical, displays information. Useful for debugging, default is FALSE.
to	Territorial office. It can be any type of string, the function would perform a search using base::grep().

Value

A tibble with the information requested.

- catr_atom_get_parcels_db_all() includes the following fields:
  - territorial_office: Territorial office, corresponding to each province of Spain expect Basque Country and Navarre.
  - url: ATOM url for the corresponding territorial office.
  - munic: Name of the municipality.
  - date: Reference date of the data. Note that **the information of this service is updated twice a year**.
- catr_atom_get_parcels_db_to() includes the following fields:
  - munic: Name of the municipality.
  - url: url for downloading information of the corresponding municipality.
  - date: Reference date of the data. Note that **the information of this service is updated twice a year**.

Source

https://www.catastro.minhap.es/INSPIRE/CadastralParcels/ES.SDGC.CP.atom.xml

See Also

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_wfs_get_address_bbox(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE ATOM services: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_atom_search_munic()

Other parcels: catr_atom_get_parcels(), catr_wfs_get_parcels_bbox()

Other databases: catr_atom_get_address_db_all(), catr_atom_get_buildings_db_all(), catr_atom_search_munic(), catr_srs_values

Examples

```
catr_atom_get_parcels_db_all()
```

`catr_atom_search_munic` *ATOM INSPIRE: Search for municipality codes*

Description

Search for a municipality (as a string, part of string or code) and get the corresponding coding as per the Cadastre.

Usage

```
catr_atom_search_munic(
  munic,
  to = NULL,
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`munic`	Municipality to extract, It can be a part of a string or the cadastral code.
`to`	Optional parameter for defining the Territorial Office to which `munic` belongs. This parameter is a helper for narrowing the search.
`cache`	A logical whether to do caching. Default is `TRUE`. See **About caching** section on `catr_set_cache_dir()`.
`update_cache`	A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.
`cache_dir`	A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

Value

A `tibble`.

See Also

Other INSPIRE ATOM services: `catr_atom_get_address()`, `catr_atom_get_address_db_all()`, `catr_atom_get_buildings()`, `catr_atom_get_buildings_db_all()`, `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`

Other search: `catr_get_code_from_coords()`, `catr_ovc_get_cod_munic()`, `catr_ovc_get_cod_provinces()`

Other databases: `catr_atom_get_address_db_all()`, `catr_atom_get_buildings_db_all()`, `catr_atom_get_parcels_`
`catr_srs_values`

Examples

```
catr_atom_search_munic("Mad")
```

`catr_clear_cache` *Clear your* R*hrefhttps://CRAN.R-project.org/package=CatastRo***CatastRo** *cache dir*

Description

Use this function with caution. This function would clear your cached data and configuration, specifically:

- Deletes the **CatastRo** config directory (`rappdirs::user_config_dir("CatastRo", "R")`).
- Deletes the `cache_dir` directory.
- Deletes the values on stored on `Sys.getenv("CATASTROESP_CACHE_DIR")`.

Usage

```
catr_clear_cache(config = FALSE, cached_data = TRUE, verbose = FALSE)
```

Arguments

`config`	if `TRUE`, will delete the configuration folder of **CatastRo**.
`cached_data`	If this is set to `TRUE`, it will delete your `cache_dir` and all its content.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

Details

This is an overkill function that is intended to reset your status as it you would never have installed and/or used **CatastRo**.

Value

Invisible. This function is called for its side effects.

See Also

Other cache utilities: `catr_set_cache_dir()`

Examples

```
# Don't run this! It would modify your current state
## Not run:
catr_clear_cache(verbose = TRUE)

## End(Not run)

Sys.getenv("CATASTROESP_CACHE_DIR")
```

`catr_get_code_from_coords` *Get the cadastral municipality code from coordinates*

Description

This function takes as an input a pair of coordinates of a `sf` object and returns the corresponding municipality code for that coordinates.

See also `mapSpain::esp_get_munic_siane()` and `catr_ovc_get_cod_munic()`.

Usage

```
catr_get_code_from_coords(x, srs, verbose = FALSE, cache_dir = NULL, ...)
```

Arguments

`x`	It could be: • A pair of coordinates c(x,y). • A `sf` object. See **Details**.
`srs`	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `wfs_service` column. See **Details**.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.
`cache_dir`	A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.
`...`	Arguments passed on to `mapSpain::esp_get_munic_siane` `year` Release year. See **Details** for years available.

Details

When `x` is a numeric vector, make sure that the `srs` matches the coordinate values.

When `x` is a `sf` object, only the first value would be used. The function would extract the coordinates using `sf::st_centroid(x, of_largest_polygon = TRUE)`.

Value

A `tibble` with the format described in `catr_ovc_get_cod_munic()`.

See Also

mapSpain::esp_get_munic_siane(), sf::st_centroid().

Other search: catr_atom_search_munic(), catr_ovc_get_cod_munic(), catr_ovc_get_cod_provinces()

Examples

```
# Use with coords
catr_get_code_from_coords(c(-16.25462, 28.46824), srs = 4326)

# Use with sf
prov <- mapSpain::esp_get_prov("Caceres")
catr_get_code_from_coords(prov)
```

---

catr_ovc_get_cod_munic

OVCCallejero: Extract the code of a municipality

Description

Implementation of the OVCCallejero service ConsultaMunicipioCodigos.

Return the names and codes of a municipality. Returns both the codes as per the Cadastre and as per the INE (National Statistics Institute).

Usage

```
catr_ovc_get_cod_munic(cpro, cmun = NULL, cmun_ine = NULL, verbose = FALSE)
```

Arguments

cpro	The code of a province, as provided by catr_ovc_get_cod_provinces().
cmun	Code of a municipality, as recorded on the Spanish Cadastre.
cmun_ine	Code of a municipality, as recorded on National Statistics Institute. See INE: List of municipalities
verbose	Logical, displays information. Useful for debugging, default is FALSE.

Details

Parameter cpro is mandatory. Either cmun or cmun_ine should be provided.

On a successful query, the function returns a tibble with one row including the following columns:

- munic: Name of the municipality as per the Cadastre.
- catr_to: Cadastral territorial office code.

- `catr_munic`: Municipality code as recorded on the Cadastre.
- `catrcode`: Full Cadastral code for the municipality.
- `cpro`: Province code as per the INE.
- `catr_munic`: Municipality code as per the INE.
- `catrcode`: Full INE code for the municipality.
- Rest of fields: Check the API Docs.

Value

A `tibble`. See **Details**

References

ConsultaMunicipioCodigos.

See Also

`mapSpain::esp_get_munic()` to get shapes of municipalities, including the INE code.

OVCCoordenadas API: `catr_ovc_get_cod_provinces()`

Other search: `catr_atom_search_munic()`, `catr_get_code_from_coords()`, `catr_ovc_get_cod_provinces()`

Examples

```
# Get municipality by cadastal code
ab <- catr_ovc_get_cod_munic(2, 900)

ab

# Same query using the INE code

ab2 <- catr_ovc_get_cod_munic(2, cmun_ine = 3)

ab2
```

`catr_ovc_get_cod_provinces` *OVCCallejero: Extract a list of provinces with their codes*

Description

Implementation of the OVCCallejero service ConsultaProvincia.

Return a list of the provinces included on the Spanish Cadastre.

Usage

```
catr_ovc_get_cod_provinces(verbose = FALSE)
```

Arguments

verbose	Logical, displays information. Useful for debugging, default is FALSE.

Value

A tibble.

References

ConsultaProvincia.

See Also

OVCCoordenadas API: catr_ovc_get_cod_munic()

Other search: catr_atom_search_munic(), catr_get_code_from_coords(), catr_ovc_get_cod_munic()

Examples

```
catr_ovc_get_cod_provinces()
```

---

catr_ovc_get_cpmrc *OVCCoordenadas: Geocode a cadastral reference*

Description

Implementation of the OVCCoordenadas service Consulta CPMRC.

Return the coordinates for a specific cadastral reference.

Usage

```
catr_ovc_get_cpmrc(
  rc,
  srs = 4326,
  province = NULL,
  municipality = NULL,
  verbose = FALSE
)
```

Arguments

rc	The cadastral reference to be geocoded.
srs	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the ovc_service column.
province, municipality	Optional, used for narrowing the search.
verbose	Logical, displays information. Useful for debugging, default is FALSE.

Details

When the API does not provide any result, the function returns a tibble with the input parameters only.

On a successful query, the function returns a tibble with one row by cadastral reference, including the following columns:

- xcoord, ycoord: X and Y coordinates in the specified SRS.
- refcat: Cadastral Reference.
- address: Address as it is recorded on the Cadastre.
- Rest of fields: Check the API Docs.

Value

A tibble. See **Details**

References

Consulta CPMRC.

See Also

catr_srs_values, vignette("ovcservice", package = "CatastRo")

OVCCoordenadas API: catr_ovc_get_rccoor(), catr_ovc_get_rccoor_distancia(), catr_srs_values

Other cadastral references: catr_ovc_get_rccoor(), catr_ovc_get_rccoor_distancia()

Examples

```
# using all the arguments
catr_ovc_get_cpmrc("13077A01800039",
  4230,
  province = "CIUDAD REAL",
  municipality = "SANTA CRUZ DE MUDELA"
)

# only the cadastral reference
catr_ovc_get_cpmrc("9872023VH5797S")
```

`catr_ovc_get_rccoor` *OVCCoordenadas: Reverse geocode a cadastral reference*

Description

Implementation of the OVCCoordenadas service Consulta RCCOOR.

Return the cadastral reference found of a set of specific coordinates.

Usage

```
catr_ovc_get_rccoor(lat, lon, srs = 4326, verbose = FALSE)
```

Arguments

`lat`	Latitude to use on the query. It should be specified in the same in the CRS/SRS specified by `srs`.
`lon`	Longitude to use on the query. It should be specified in the same in the CRS/SRS specified by `srs`.
`srs`	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `ovc_service` column.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

Details

When the API does not provide any result, the function returns a `tibble` with the input parameters only.

On a successful query, the function returns a `tibble` with one row by cadastral reference, including the following columns:

- `geo.xcen`, `geo.ycen`, `geo.srs`: Input parameters of the query.
- `refcat`: Cadastral Reference.
- `address`: Address as it is recorded on the Cadastre.
- Rest of fields: Check the API Docs.

Value

A `tibble`. See **Details**

References

Consulta RCCOOR.

See Also

catr_srs_values, `vignette("ovcservice", package = "CatastRo")`

OVCCoordenadas API: `catr_ovc_get_cpmrc()`, `catr_ovc_get_rccoor_distancia()`, `catr_srs_values`

Other cadastral references: `catr_ovc_get_cpmrc()`, `catr_ovc_get_rccoor_distancia()`

Examples

```
catr_ovc_get_rccoor(
  lat = 38.6196566583596,
  lon = -3.45624183836806,
  srs = 4326
)
```

`catr_ovc_get_rccoor_distancia`

OVCCoordenadas: Reverse geocode cadastral references on a region

---

Description

Implementation of the OVCCoordenadas service Consulta RCCOOR Distancia.

Return the cadastral reference found on a set of coordinates. If no cadastral references are found, the API returns a list of the cadastral references found on an area of 50 square meters around the requested coordinates.

Usage

```
catr_ovc_get_rccoor_distancia(lat, lon, srs = 4326, verbose = FALSE)
```

Arguments

`lat`	Latitude to use on the query. It should be specified in the same in the CRS/SRS specified by `srs`.
`lon`	Longitude to use on the query. It should be specified in the same in the CRS/SRS specified by `srs`.
`srs`	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `ovc_service` column.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

Details

When the API does not provide any result, the function returns a `tibble` with the input parameters only.

On a successful query, the function returns a `tibble` with one row by cadastral reference, including the following columns:

- `geo.xcen`, `geo.ycen`, `geo.srs`: Input parameters of the query.
- `refcat`: Cadastral reference.
- `address`: Address as it is recorded on the Cadastre.
- `cmun_ine`: Municipality code as registered on the INE (National Statistics Institute).
- Rest of fields: Check the API Docs.

Value

A `tibble`. See **Details**

References

Consulta RCCOOR Distancia.

See Also

catr_srs_values, `vignette("ovcservice", package = "CatastRo")`

OVCCoordenadas API: `catr_ovc_get_cpmrc()`, `catr_ovc_get_rccoor()`, `catr_srs_values`

Other cadastral references: `catr_ovc_get_cpmrc()`, `catr_ovc_get_rccoor()`

Examples

```
catr_ovc_get_rccoor_distancia(
  lat = 40.963200,
  lon = -5.671420,
  srs = 4326
)
```

catr_set_cache_dir *Set your* R*hrefhttps://CRAN.R-project.org/package=CatastRo***CatastRo** *cache dir*

Description

`catr_set_cache_dir()` will store your `cache_dir` path on your local machine and would load it for future sessions.

Alternatively, you can store the `cache_dir` manually with the following options:

- Run `Sys.setenv(CATASTROESP_CACHE_DIR = "cache_dir")`. You would need to run this command on each session (Similar to `install = FALSE`).
- Write this line on your .Renviron file: `CATASTROESP_CACHE_DIR = "value_for_cache_dir"` (same behavior than `install = TRUE`). This would store your `cache_dir` permanently.

`catr_detect_cache_dir()` detects and returns the path to your current `cache_dir`.

Usage

```
catr_set_cache_dir(
  cache_dir = NULL,
  overwrite = FALSE,
  install = FALSE,
  verbose = TRUE
)

catr_detect_cache_dir(...)
```

Arguments

`cache_dir`	A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.
`overwrite`	If this is set to `TRUE`, it will overwrite an existing `CATASTROESP_CACHE_DIR` that you already have in local machine.
`install`	if `TRUE`, will install the key in your local machine for use in future sessions. Defaults to `FALSE`. If `cache_dir` is `FALSE` this parameter is set to `FALSE` automatically.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.
`...`	Ignored

Value

`catr_set_cache_dir()` is called for its side effects, and returns an (invisible) character with the path to your `cache_dir`.

`catr_detect_cache_dir()` returns the path to the `cache_dir` used in this session

About caching

Sometimes cached files may be corrupt. On that case, try re-downloading the data setting `update_cache = TRUE`.

If you experience any problem on download, try to download the corresponding file by any other method and save it on your `cache_dir`. Use the option `verbose = TRUE` for debugging the API query.

See Also

`rappdirs::user_config_dir()`

Other cache utilities: `catr_clear_cache()`

Examples

```
# Don't run this! It would modify your current state
## Not run:
catr_set_cache_dir(verbose = TRUE)

## End(Not run)


catr_detect_cache_dir()
```

`catr_srs_values` *Reference SRS codes for* R*hrefhttps://CRAN.R-project.org/package=CatastRo***CatastRo** *APIs*

Description

A `tibble` including the valid SRS (also known as CRS) values that may be used on each API service. The values are provided as EPSG codes.

Format

A `tibble` with 16 rows and columns:

SRS Spatial Reference System (CRS) value, identified by the corresponding EPSG code.

Description Description of the SRS/EPSG code.

ovc_service Logical. Is this code valid on OVC services?

wfs_service Logical. Is this code valid on INSPIRE WFS services?

Details

Table: Content of catr_srs_values

SRS	Description	ovc_service	wfs_service
3785	Web Mercator	FALSE	TRUE
3857	Web Mercator	FALSE	TRUE
4230	Geográficas en ED 50	TRUE	FALSE
4258	Geográficas en ETRS89	TRUE	TRUE
4326	Geográficas en WGS 80	TRUE	TRUE
23029	UTM huso 29N en ED50	TRUE	FALSE
23030	UTM huso 30N en ED50	TRUE	FALSE
23031	UTM huso 31N en ED50	TRUE	FALSE
25829	UTM huso 29N en ETRS89	TRUE	TRUE
25830	UTM huso 30N en ETRS89	TRUE	TRUE
25831	UTM huso 31N en ETRS89	TRUE	TRUE
32627	UTM huso 27N en WGS 84	TRUE	FALSE
32628	UTM huso 28N en WGS 84	TRUE	FALSE
32629	UTM huso 29N en WGS 84	TRUE	FALSE
32630	UTM huso 30N en WGS 84	TRUE	FALSE
32631	UTM huso 31N en WGS 84	TRUE	FALSE

References

- OVCCoordenadas.
- INSPIRE WFS Service.

See Also

sf::st_crs().

Other databases: catr_atom_get_address_db_all(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels_db_all(), catr_atom_search_munic()

Other INSPIRE WFS services: catr_wfs_get_address_bbox(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox()

OVCCoordenadas API: catr_ovc_get_cpmrc(), catr_ovc_get_rccoor(), catr_ovc_get_rccoor_distancia()

Examples

```
data("catr_srs_values")

# OVC valid codes
library(dplyr)

catr_srs_values %>% filter(ovc_service == TRUE)

# WFS valid codes

catr_srs_values %>% filter(wfs_service == TRUE)

# Use with sf::st_crs()

catr_srs_values %>%
```

```
  filter(wfs_service == TRUE & ovc_service == TRUE) %>%
  print() %>%
  # First value
  slice_head(n = 1) %>%
  pull(SRS) %>%
  # As crs
  sf::st_crs(.)
```

`catr_wfs_get_address_bbox`

WFS INSPIRE: Download addresses

Description

Get the spatial data of addresses The WFS Service allows to perform several types of queries:

- By bounding box: Implemented on `catr_wfs_get_address_bbox()`. Extract objects included on the bounding box provided. See **Details**.
- By street code: Implemented on `catr_wfs_get_address_codvia()`. Extract objects of specific addresses.
- By cadastral reference: Implemented on `catr_wfs_get_address_rc()`. Extract objects of specific cadastral references
- By postal codes: Implemented on `catr_wfs_get_address_postalcode()`. Extract objects of specific cadastral references

Usage

```
catr_wfs_get_address_bbox(x, srs, verbose = FALSE)

catr_wfs_get_address_codvia(codvia, del, mun, srs = NULL, verbose = FALSE)

catr_wfs_get_address_rc(rc, srs = NULL, verbose = FALSE)

catr_wfs_get_address_postalcode(postalcode, srs = NULL, verbose = FALSE)
```

Arguments

`x`	See **Details**. It could be: • A numeric vector of length 4 with the coordinates that defines the bounding box: `c(xmin, ymin, xmax, ymax)` • A `sf/sfc` object, as provided by the **sf** package.
`srs`	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `wfs_service` column. See **Details**.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.

codvia	Cadastral street code.
del	Cadastral office code.
mun	Cadastral municipality code.
rc	The cadastral reference to be extracted.
postalcode	Postal code.

Details

When x is a numeric vector, make sure that the srs matches the coordinate values. Additionally, when the srs correspond to a geographic reference system (4326, 4258), the function queries the bounding box on EPSG:3857 - Web Mercator, to overcome a potential bug on the API side.

When x is a sf object, the value srs is ignored. In this case, the bounding box of the sf object would be used for the query (see sf::st_bbox()). The query is performed using EPSG:3857 (Web Mercator). The result is provided always in the SRS of the sf object provided as input.

Value

A sf object.

API Limits

The API service is limited to a bounding box of 4km2 and a maximum of 5.000 elements.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

sf::st_bbox()

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE WFS services: catr_srs_values, catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox()

Other addresses: catr_atom_get_address(), catr_atom_get_address_db_all()

Other spatial: catr_atom_get_address(), catr_atom_get_buildings(), catr_atom_get_parcels(), catr_wfs_get_buildings_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Examples

```
ad <- catr_wfs_get_address_bbox(
  c(
    233673, 4015968, 233761, 4016008
  ),
```

```
  srs = 25830
)

library(ggplot2)

ggplot(ad) +
  geom_sf()
```

`catr_wfs_get_buildings_bbox`

WFS INSPIRE: Download buildings

Description

Get the spatial data of buildings. The WFS Service allows to perform two types of queries:

- By bounding box: Implemented on `catr_wfs_get_buildings_bbox()`. Extract objects included on the bounding box provided. See **Details**.
- By cadastral reference: Implemented on `catr_wfs_get_buildings_rc()`. Extract objects of specific cadastral references.

Usage

```
catr_wfs_get_buildings_bbox(x, what = "building", srs, verbose = FALSE)

catr_wfs_get_buildings_rc(rc, what = "building", srs = NULL, verbose = FALSE)
```

Arguments

`x`	See **Details**. It could be: • A numeric vector of length 4 with the coordinates that defines the bounding box: `c(xmin, ymin, xmax, ymax)` • A `sf/sfc` object, as provided by the **sf** package.
`what`	Information to load. It could be: • `"building"` for buildings. • `"buildingpart"` for parts of a building. • `"other"` for others elements, as swimming pools, etc.
`srs`	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `wfs_service` column. See **Details**.
`verbose`	Logical, displays information. Useful for debugging, default is `FALSE`.
`rc`	The cadastral reference to be extracted.

Details

When x is a numeric vector, make sure that the srs matches the coordinate values. Additionally, when the srs correspond to a geographic reference system (4326, 4258), the function queries the bounding box on EPSG:3857 - Web Mercator, to overcome a potential bug on the API side. The result is provided always in the SRS provided in srs.

When x is a **sf** object, the value srs is ignored. The query is performed using EPSG:3857 (Web Mercator) and the spatial object is projected back to the SRS of the initial object.

Value

A sf object.

API Limits

The API service is limited to a bounding box of 4km2 and a maximum of 5.000 elements.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

sf::st_bbox()

INSPIRE API functions: catr_atom_get_address(), catr_atom_get_address_db_all(), catr_atom_get_buildings(), catr_atom_get_buildings_db_all(), catr_atom_get_parcels(), catr_atom_get_parcels_db_all(), catr_wfs_get_address_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Other INSPIRE WFS services: catr_srs_values, catr_wfs_get_address_bbox(), catr_wfs_get_parcels_bbox()

Other buildings: catr_atom_get_buildings(), catr_atom_get_buildings_db_all()

Other spatial: catr_atom_get_address(), catr_atom_get_buildings(), catr_atom_get_parcels(), catr_wfs_get_address_bbox(), catr_wfs_get_parcels_bbox(), catr_wms_get_layer()

Examples

```
# Using bbox
building <- catr_wfs_get_buildings_bbox(
  c(
    376550,
    4545424,
    376600,
    4545474
  ),
  srs = 25830
)
library(ggplot2)
ggplot(building) +
```

```
  geom_sf() +
  labs(title = "Search using bbox")

# Using rc
rc <- catr_wfs_get_buildings_rc("6656601UL7465N")
library(ggplot2)
ggplot(rc) +
  geom_sf() +
  labs(title = "Search using rc")
```

`catr_wfs_get_parcels_bbox`

WFS INSPIRE: Download cadastral parcels

Description

Get the spatial data of cadastral parcels and zones. The WFS Service allows to perform several types of queries:

- By bounding box: Implemented on `catr_wfs_get_parcels_bbox()`. Extract objects included on the bounding box provided. See **Details**.
- By zoning: Implemented on `catr_wfs_get_parcels_zoning()`. Extract objects of a specific cadastral zone.
- By cadastral parcel: Implemented on `catr_wfs_get_parcels_parcel()`. Extract cadastral parcels of a specific cadastral reference.
- Neighbor cadastral parcels: Implemented on `catr_wfs_get_parcels_neigh_parcel()`. Extract neighbor cadastral parcels of a specific cadastral reference.
- Cadastral parcels by zoning: Implemented on `catr_wfs_get_parcels_parcel_zoning()`. Extract cadastral parcels of a specific cadastral zone.

Usage

```
catr_wfs_get_parcels_bbox(x, what = "parcel", srs, verbose = FALSE)

catr_wfs_get_parcels_zoning(cod_zona, srs = NULL, verbose = FALSE)

catr_wfs_get_parcels_parcel(rc, srs = NULL, verbose = FALSE)

catr_wfs_get_parcels_neigh_parcel(rc, srs = NULL, verbose = FALSE)

catr_wfs_get_parcels_parcel_zoning(cod_zona, srs = NULL, verbose = FALSE)
```

Arguments

x	See **Details**. It could be: • A numeric vector of length 4 with the coordinates that defines the bounding box: `c(xmin, ymin, xmax, ymax)` • A `sf/sfc` object, as provided by the **sf** package.
what	Information to load. It could be `"parcel"` for cadastral parcels or `"zoning"` for cadastral zoning.
srs	SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `wfs_service` column. See **Details**.
verbose	Logical, displays information. Useful for debugging, default is `FALSE`.
cod_zona	Cadastral zone code.
rc	The cadastral reference to be extracted.

Details

When `x` is a numeric vector, make sure that the `srs` matches the coordinate values. Additionally, when the `srs` correspond to a geographic reference system (4326, 4258), the function queries the bounding box on EPSG:3857 - Web Mercator, to overcome a potential bug on the API side. The result is provided always in the SRS provided in `srs`.

When `x` is a `sf` object, the value `srs` is ignored. The query is performed using EPSG:3857 (Web Mercator) and the spatial object is projected back to the SRS of the initial object.

Value

A `sf` object.

API Limits

The API service is limited to the following constrains:

- `"parcel`: Bounding box of 1km2 and a maximum of 500. elements.
- `"zoning"`: Bounding box of 25km2 and a maximum of 500 elements.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

`sf::st_bbox()`

INSPIRE API functions: `catr_atom_get_address()`, `catr_atom_get_address_db_all()`, `catr_atom_get_buildings()`, `catr_atom_get_buildings_db_all()`, `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`, `catr_wfs_get_address_bbox()`, `catr_wfs_get_buildings_bbox()`, `catr_wms_get_layer()`

Other INSPIRE WFS services: `catr_srs_values`, `catr_wfs_get_address_bbox()`, `catr_wfs_get_buildings_bbox()`

Other parcels: `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`

Other spatial: catr_atom_get_address(), catr_atom_get_buildings(), catr_atom_get_parcels(), catr_wfs_get_address_bbox(), catr_wfs_get_buildings_bbox(), catr_wms_get_layer()

Examples

```
cp <- catr_wfs_get_parcels_bbox(
  c(
    233673, 4015968, 233761, 4016008
  ),
  srs = 25830
)

library(ggplot2)

ggplot(cp) +
  geom_sf()
```

`catr_wms_get_layer` *WMS INSPIRE: Download map images*

Description

Get geotagged images from the Spanish Cadastre. This function is a wrapper of `mapSpain::esp_getTiles()`.

Usage

```
catr_wms_get_layer(
  x,
  srs,
 what = c("building", "buildingpart", "parcel", "zoning", "address", "admboundary",
    "admunit"),
  styles = "default",
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE,
  crop = FALSE,
  options = NULL,
  ...
)
```

Arguments

`x` See **Details**. It could be:

- A numeric vector of length 4 with the coordinates that defines the bounding box: `c(xmin, ymin, xmax, ymax)`

- A `sf/sfc` object, as provided by the **sf** package.

`srs` SRS/CRS to use on the query. To check the admitted values check catr_srs_values, specifically the `wfs_service` column. See **Details**.

`what` Layer to be extracted, see **Details**.

`styles` Style of the WMS layer. See **Details**.

`update_cache` A logical whether to update cache. Default is `FALSE`. When set to `TRUE` it would force a fresh download of the source file.

`cache_dir` A path to a cache directory. On `NULL` value (the default) the function would store the cached files on the `tempdir`.

`verbose` Logical, displays information. Useful for debugging, default is `FALSE`.

`crop` `TRUE` if results should be cropped to the specified `x` extent, `FALSE` otherwise. If `x` is an **sf** object with one `POINT`, crop is set to `FALSE`.

`options` A named list containing additional options to pass to the query.

`...` Arguments passed on to `mapSpain::esp_getTiles`

`res` Resolution (in pixels) of the final tile. Only valid for WMS.

`bbox_expand` A numeric value that indicates the expansion percentage of the bounding box of `x`.

`transparent` Logical. Provides transparent background, if supported. Depends on the selected provider on `type`.

`mask` `TRUE` if the result should be masked to `x`.

Details

When `x` is a numeric vector, make sure that the `srs` matches the coordinate values. When `x` is a `sf` object, the value `srs` is ignored.

The query is performed using EPSG:3857 (Web Mercator) and the tile is projected back to the SRS of `x`. In case that the tile looks deformed, try either providing `x` or specify the SRS of the requested tile via the `srs` parameter, that ideally would need to match the SRS of `x`. See **Examples**.

Value

A `SpatRaster` is returned, with 3 (RGB) or 4 (RGBA) layers, see `terra::RGB()`.

Layers

The parameter `what` defines the layer to be extracted. The equivalence with the API Docs equivalence is:

- `"parcel"`: CP.CadastralParcel
- `"zoning"`: CP.CadastralZoning
- `"building"`: BU.Building
- `"buildingpart"`: BU.BuildingPart
- `"address"`: AD.Address
- `"admboundary"`: AU.AdministrativeBoundary
- `"admunit"`: AU.AdministrativeUnit

Styles

The WMS service provide different styles on each layer (`what` parameter). Some of the styles available are:

- `"parcel"`: styles : `"BoundariesOnly"`, `"ReferencePointOnly"`, `"ELFCadastre"`.
- `"zoning"`: styles : `"BoundariesOnly"`, `"ELFCadastre"`.
- `"building"`, `"buildingpart"`: `"ELFCadastre"`
- `"address"`: `"Number.ELFCadastre"`
- `"admboundary"`, `"admunit"`: `"ELFCadastre"`

Check the API Docs for more information.

References

API Documentation.

INSPIRE Services for Cadastral Cartography.

See Also

`mapSpain::esp_getTiles()` and `terra::RGB()`. For plotting see `terra::plotRGB()` and `tidyterra::geom_spatraster`

INSPIRE API functions: `catr_atom_get_address()`, `catr_atom_get_address_db_all()`, `catr_atom_get_buildings()`, `catr_atom_get_buildings_db_all()`, `catr_atom_get_parcels()`, `catr_atom_get_parcels_db_all()`, `catr_wfs_get_address_bbox()`, `catr_wfs_get_buildings_bbox()`, `catr_wfs_get_parcels_bbox()`

Other spatial: `catr_atom_get_address()`, `catr_atom_get_buildings()`, `catr_atom_get_parcels()`, `catr_wfs_get_address_bbox()`, `catr_wfs_get_buildings_bbox()`, `catr_wfs_get_parcels_bbox()`

Examples

```
# With a bbox

pict <- catr_wms_get_layer(
  c(222500, 4019500, 223700, 4020700),
  srs = 25830,
  what = "parcel"
)

library(mapSpain)
library(ggplot2)
library(tidyterra)

ggplot() +
  geom_spatraster_rgb(data = pict)


# With a spatial object
```

```
parcels <- catr_wfs_get_parcels_neigh_parcel("3662303TF3136B", srs = 25830)


# Use styles

parcels_img <- catr_wms_get_layer(parcels,
  what = "buildingpart",
  srs = 25830, # As parcels object
  bbox_expand = 0.3,
  styles = "ELFCadastre"
)


ggplot() +
  geom_sf(data = parcels, fill = "blue", alpha = 0.5) +
  geom_spatraster_rgb(data = parcels_img)
```

Index