Package 'mapSpain'

November 26, 2020

Type Package

Title Administrative Boundaries of Spain Including CCAA, Provinces and Municipalities

Version 0.1.0

Description Administrative Boundaries of Spain based on
the GISCO (Geographic Information System of the Commission) Eurostat database
<https://ec.europa.eu/eurostat/web/gisco>. It also provides a leaflet plugin
and the ability of downloading and processing static tiles.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

BugReports `https://github.com/dieghernan/mapSpain/issues`

URL `https://dieghernan.github.io/mapSpain/`, `https://github.com/dieghernan/mapSpain`

Depends R (>= 3.6.0)

Imports sf (>= 0.9),
countrycode (>= 1.2.0),
giscoR (>= 0.2.0),
raster (>= 3.0),
png (>= 0.1-5),
slippymath (>= 0.3.1),
leaflet (>= 2.0.0)

Suggests cartography (>= 2.4),
rgdal,
tinytest

R topics documented:

`mapSpain-package` *mapSpain package*

Description

This package provides Administrative Boundaries of Spain based on the GISCO (Geographic Information System of the Commission) Eurostat database.

Details

Package:	mapSpain
Type:	Package
Version:	See sessionInfo() or DESCRIPTION file
Date:	2020
License:	GPL-3
LazyLoad:	yes

Note

COPYRIGHT NOTICE

When data downloaded from this page is used in any printed or electronic publication, in addition to any other provisions applicable to the whole Eurostat website, data source will have to be acknowledged in the legend of the map and in the introductory page of the publication with the following copyright notice:

- EN: (C) EuroGeographics for the administrative boundaries
- FR: (C) EuroGeographics pour les limites administratives
- DE: (C) EuroGeographics bezüglich der Verwaltungsgrenzen

For publications in languages other than English, French or German, the translation of the copyright notice in the language of the publication shall be used.

If you intend to use the data commercially, please contact EuroGeographics for information regarding their licence agreements.

Author(s)

dieghernan, `https://github.com/dieghernan/`

Source

GISCO webpage

References

See `citation("mapSpain")`

See Also

Useful links:

- `https://dieghernan.github.io/mapSpain/`
- `https://github.com/dieghernan/mapSpain`
- Report bugs at `https://github.com/dieghernan/mapSpain/issues`

`addProviderEspTiles` *Leaflet plugin - Spanish providers*

Description

Add tiles of leaflet-providersESP to a **R** leaflet map.

Usage

```
addProviderEspTiles(
  map,
  provider,
  layerId = NULL,
  group = NULL,
  options = providerEspTileOptions()
)

providerEspTileOptions(...)
```

Arguments

`map, layerId, group, options`	See addTiles
`provider`	Name of the provider, see leaflet.providersESP.df.
`...`	Additional options. See providerTileOptions.

Details

`providerEspTileOptions` is a wrapper of `leaflet::providerTileOptions`

Value

Modified `map` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

Source

leaflet-providersESP leaflet plugin, **v1.2.0**.

See Also

leaflet.providersESP.df, esp_getTiles

tileOptions, providerTileOptions

Examples

```
library(leaflet)
  PuertadelSol <-
    leaflet() %>% setView(lat = 40.4166,
                          lng = -3.7038400,
                          zoom = 18) %>%
    addProviderEspTiles(provider = "IGNBase.Gris") %>%
    addProviderEspTiles(provider = "RedTransporte.Carreteras")

    PuertadelSol
```

---

esp_codelist *Spanish Code Translation Data Frame*

Description

A data frame used internally for translating codes and names of the different subdivisions of Spain. The data frame provides the hierarchy of the subdivisions including NUTS1 level, Autonomous Communities (equivalent to NUTS2), Provinces and NUTS3 level. See Note.

Format

data frame with codes as columns

- **nuts*.code**: NUTS code of each subdivision.
- **nuts*.code**: NUTS name of each subdivision.
- **codauto**: INE code of each autonomous community.
- **iso2.*.code**: ISO2 code of each autonomous community and province.
- **ine.*.name**: INE name of each autonomous community and province.
- **iso2.*.name.***: ISO2 name of each autonomous community and province. Several languages available.
- **cldr.*.name.***: CLDR name of each autonomous community and province. Several languages available.
- **ccaa.short.***: Short (common) name of each autonomous community. Several languages available.
- **cpro**: INE code of each province.
- **prov.shortname.***: Short (common) name of each province. Several languages available.

Note

Languages available are:

- en: English
- es: Spanish
- ca: Catalan
- ga: Galician
- eu: Basque

Although NUTS2 matches the first subdivision level of Spain (CCAA - Autonomous Communities), it should be noted that NUTS3 does not match the second subdivision level of Spain (Provinces). NUTS3 provides a dedicated code for major islands whereas the Provinces doesn't.

Ceuta and Melilla has an specific status (Autonomous Cities) but are considered as communities with a single provice (as Madrid, Asturias or Murcia) on this dataset.

Source

- INE: Instituto Nacional de Estadistica: `https://www.ine.es/`
- Eurostat (NUTS): `https://ec.europa.eu/eurostat/web/nuts/background`
- ISO: `https://www.iso.org/obp/ui/#iso:code:3166:ES`
- CLDR: `https://unicode-org.github.io/cldr-staging/charts/38/index.html`

Examples

```
data(esp_codelist)
```

`esp_dict_region_code` *Convert and translate Subdivision Names*

Description

Converts long subdivision names into different coding schemes and languages.

Usage

```
esp_dict_region_code(sourcevar, origin = "text", destination = "text")

esp_dict_translate(sourcevar, lang = "en", all = FALSE)
```

Arguments

`sourcevar`	Vector which contains the subdivision names to be converted.
`origin, destination`	One of `'text'`,`'nuts'`,`'iso2'`,`'codauto'` and `'cpro'`.
`lang`	Language of translatation. Available languages are: • es: Spanish • en: English • ca: Catalan

- ga: Galician
- eu: Basque

all	Logical. Should the function return all names or not? On FALSE it returns a character vector. See Value

Details

If no match is found for any value, the function displays a warning and returns NA for those values.

Note that mixing names of different administrative levels (e.g. Catalonia and Barcelona) may return empty values, depending on the destination values.

Value

esp_dict_region_code returns a vector of characters.

esp_dict_translate returns a character vector or a named list with each of the possible names of each sourcevar on the required language lang.

Author(s)

dieghernan, https://github.com/dieghernan/

Examples

```
vals <- c("Errioxa", "Coruna", "Gerona", "Madrid")

esp_dict_region_code(vals)
esp_dict_region_code(vals, destination = "nuts")
esp_dict_region_code(vals, destination = "cpro")
esp_dict_region_code(vals, destination = "iso2")

# From ISO2 to another codes

iso2vals <- c("ES-M", "ES-S", "ES-SG")
esp_dict_region_code(iso2vals, origin = "iso2")
esp_dict_region_code(iso2vals, origin = "iso2",
                     destination = "nuts")
esp_dict_region_code(iso2vals, origin = "iso2",
                     destination = "cpro")

# Mixing levels
valsmix <- c("Centro", "Andalucia", "Seville", "Menorca")
esp_dict_region_code(valsmix, destination = "nuts")

## Not run:

# Warning

esp_dict_region_code(valsmix, destination = "codauto")
esp_dict_region_code(valsmix, destination = "iso2")

## End(Not run)

vals <-  c("La Rioja", "Sevilla", "Madrid",
           "Jaen", "Orense", "Baleares")
esp_dict_translate(vals)
esp_dict_translate(vals, lang = "es")
esp_dict_translate(vals, lang = "ca")
esp_dict_translate(vals, lang = "eu")
esp_dict_translate(vals, lang = "ga")

esp_dict_translate(vals, lang = "ga", all = TRUE)
```

`esp_getTiles` *Get Tiles from Public Administations of Spanish.*

Description

Get static map tiles based on a spatial object. Maps can be fetched from various open map servers.

This function is a implementation of the javascript plugin leaflet-providersESP **v1.2.0**

Usage

```
esp_getTiles(
  x,
  type = "IDErioja",
  zoom = NULL,
  crop = TRUE,
  res = 512,
  bbox_expand = 0.05,
  transparent = TRUE,
  mask = FALSE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`x`	An `sf` object.
`type`	Name of the provider. See leaflet.providersESP.df.
`zoom`	the zoom level. If `NULL`, it is determined automatically (see `getTiles`). Only valid for WMTS.
`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.
`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`.
`update_cache, cache_dir, verbose`	See esp_get_nuts

Details

Results of `esp_getTiles` could be plotted using `tilesLayer`

For a complete list of providers see leaflet.providersESP.df.

Most WMS/WMTS providers provide tiles on EPSG:3857. In case that the tile looks deformed, try projecting first `x`:

`x <-sf::st_transform(x,3857)`

Tiles are cached under the path `cache_dir/[type]`

Value

A RasterBrick is returned.

Author(s)

dieghernan, `https://github.com/dieghernan/`

Source

leaflet-providersESP leaflet plugin, **v1.2.0**.

See Also

leaflet.providersESP.df, addProviderEspTiles, getTiles, tilesLayer.

`esp_get_can_box` *Get complementary lines when plotting Canary Islands.*

Description

When plotting Spain, it is usual to represent the Canary Islands as an inset (see `moveCAN` on esp_get_nuts). These functions provides complementary borders when Canary Islands are displaces.

`esp_get_can_box` is used to draw lines around the displaced Canary Islands.

`esp_get_can_provinces` is used to draw a separator line between the two provinces of the Canary Islands.

Usage

```
esp_get_can_box(style = "right", moveCAN = TRUE, epsg = "4258")

esp_get_can_provinces(moveCAN = TRUE, epsg = "4258")
```

Arguments

`style`	Style of line around Canary Islands. Three options available: `'left'`,`'right'` or `'box'`.
`moveCAN, epsg`	See esp_get_nuts

Value

A `LINESTRING` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

See Also

esp_get_nuts, esp_get_ccaa.

Examples

```
library(sf)

Provs <-  esp_get_prov()
Box <- esp_get_can_box()
Line <- esp_get_can_provinces()


plot(st_geometry(Provs), col = hcl.colors(4, palette = "Grays"))
plot(Box, add = TRUE)
plot(Line, add = TRUE)


# Displacing Canary

Provs_D <-  esp_get_prov(moveCAN = c(15, 0))
Box_D <- esp_get_can_box(style = "left", moveCAN = c(15, 0))
Line_D <- esp_get_can_provinces(moveCAN = c(15, 0))



plot(st_geometry(Provs_D), col = hcl.colors(4, palette = "Grays"))
plot(Box_D, add = TRUE)
plot(Line_D, add = TRUE)
```

`esp_get_ccaa` *Get Autonomous Communities boundaries of Spain*

Description

Loads a simple feature (`sf`) object containing the autonomous commmunities boundaries of Spain.

Usage

```
esp_get_ccaa(ccaa = NULL, ...)
```

Arguments

`ccaa`	A vector of names and/or codes for autonomous communities or `NULL` to get all the autonomous communities. See Details.
`...`	Additional parameters from esp_get_nuts.

Details

When using `ccaa` you can use and mix names and NUTS codes (levels 1 or 2), ISO codes (corresponding to level 2) or `codauto`. Ceuta and Melilla are considered as Autonomous Communities on this dataset.

When calling a NUTS1 level, all the Autonomous Communities of that level would be added.

Value

A `POLYGON/POINT` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

See Also

esp_get_hex_ccaa, esp_get_nuts, esp_get_prov, esp_get_munic, esp_codelist

Examples

```
library(sf)

# Random CCAA

Random <-
  esp_get_ccaa(ccaa = c("Euskadi",
                        "Catalunya",
                        "ES-EX",
                        "Canarias",
                        "ES52",
                        "01"))
plot(st_geometry(Random), col = hcl.colors(6))

# All CCAA of a Zone plus an addition

Mix <-
  esp_get_ccaa(ccaa = c("La Rioja", "Noroeste"),
               resolution = "20")
plot(
  Mix[, "nuts1.code"],
  pal = hcl.colors(2),
  key.pos = NULL,
  main = NULL,
  border = "white"
)
```

`esp_get_country` *Get boundaries of Spain*

Description

Loads a single `sf` object containing the boundaries of Spain.

Usage

```
esp_get_country(...)
```

Arguments

`...`	Additional parameters from esp_get_nuts.

Value

A `MULTIPOLYGON/MULTIPOINT` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

See Also

esp_get_nuts, esp_get_ccaa, esp_get_prov, esp_get_munic, esp_codelist

Examples

```
library(sf)

OriginalCan <- esp_get_country(moveCAN = FALSE)

plot(OriginalCan$geometry, col = hcl.colors(5))

MovedCan <- esp_get_country(moveCAN = TRUE)

plot(MovedCan$geometry, col = hcl.colors(5))
```

`esp_get_hex` *Get an hexbin map with the boundaries of Spain*

Description

Loads a hexbin map (`sf` object) with the boundaries of the provinces or autonomous communities of Spain.

Usage

```
esp_get_hex_prov(prov = NULL)

esp_get_hex_ccaa(ccaa = NULL)
```

Arguments

`prov`	See esp_get_prov
`ccaa`	See esp_get_ccaa

Details

Hexbin or grid map has an advantage over usual choropleth maps. In choropleths, a large polygon data looks more emphasized just because of its size, what introduces a bias. Here with hexbin, each region is represented equally dismissing the bias.

Value

A `POLYGON` object.

Author(s)

dieghernan, https://github.com/dieghernan/

See Also

esp_get_nuts, esp_get_ccaa, esp_get_prov, esp_get_munic, esp_codelist

Examples

```
library(sf)


ccaa <- esp_get_hex_ccaa()
plot(st_geometry(ccaa), col = hcl.colors(4))

prov <- esp_get_hex_prov()
plot(st_geometry(prov), col = hcl.colors(15))
```

`esp_get_munic` *Get municipalities boundaries of Spain*

Description

Loads a simple feature (`sf`) object containing the municipalities boundaries of Spain.

Usage

```
esp_get_munic(
  year = "2019",
  epsg = "4258",
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE,
  region = NULL,
  munic = NULL,
  moveCAN = TRUE
)
```

Arguments

`year, epsg, cache, update_cache, cache_dir, verbose, moveCAN`	See esp_get_nuts. Years from 2016 to 2019 available.
`region`	A vector of names and/or codes for provinces or `NULL` to get all the municipalities. See Details.
`munic`	A name or regex expression with the names of the required municipalities. `NULL` would not produce any filtering.

Details

When using `region` you can use and mix names and NUTS codes (levels 1, 2 or 3), ISO codes (corresponding to level 2 or 3) or `cpro`.

When calling a superior level (Province, Autonomous Community or NUTS1) , all the municipalities of that level would be added.

Value

A `POLYGON` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

Source

GISCO API

See Also

esp_get_nuts,esp_munic.sf, esp_codelist

Examples

```
library(sf)

Base <- esp_get_munic(region = c("Castilla y Leon"))
SAN <-
  esp_get_munic(
    region = c("Castilla y Leon"),
    munic = c("^San ", "^Santa ")
  )

plot(st_geometry(Base), col = "cornsilk", border = "grey80")
plot(st_geometry(SAN),
     col = "firebrick3",
     border = NA,
     add = TRUE)
```

esp_get_nuts *Get NUTS boundaries of Spain*

Description

Loads a simple feature (`sf`) object containing the NUTS boundaries of Spain.

Usage

```
esp_get_nuts(
  year = "2016",
  epsg = "4258",
  cache = TRUE,
  update_cache = FALSE,
  cache_dir = NULL,
  verbose = FALSE,
  resolution = "01",
  spatialtype = "RG",
  region = NULL,
  nuts_level = "all",
  moveCAN = TRUE
)
```

Arguments

`year` Release year. One of `"2003"`,`"2006"`, `"2010"`,`"2013"`,`"2016"` or `"2021"`

`epsg` projection of the map: 4-digit EPSG code. One of:

- `"4258"` - ETRS89
- `"4326"` - WGS84
- `"3035"` - ETRS89 / ETRS-LAEA
- `"3857"` - Pseudo-Mercator

`cache` A logical whether to do caching. Default is `TRUE`.

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

`cache_dir` A path to a cache directory. See Details.

`verbose` Display information. Useful for debugging, default is `FALSE`.

`resolution` Resolution of the geospatial data. One of

- `"60"` (1:60million),
- `"20"` (1:20million)
- `"10"` (1:10million)
- `"03"` (1:3million) or
- `"01"` (1:1million).

`spatialtype` Type of geometry to be returned:

- `"RG"`: Regions - `MULTIPOLYGON/POLYGON` object.
- `"LB"`: Labels - `POINT` object.

`region` Optional. A vector of region names, NUTS or ISO codes (see esp_dict_region_code).

`nuts_level`	NUTS level. One of `"0"` (Country-level), `"1"`,`"2"` or `"3"`. See `https://ec.europa.eu/eurostat/web/nuts/background`.
`moveCAN`	A logical `TRUE`,`FALSE` or a vector of coordinates `c(lat,lon)`. It places the Canary Island close to Spain's mainland. Initial position can be adjusted using the vector of coordinates. See Note.

Details

`cache_dir` = `NULL` (default) uses and creates `/mapSpain` directory in the temporary directory from `tempdir`. The directory can also be set with `options(mapSpain = "path/to/dir")` or `options(gisco_cache_dir = "path/to/dir")` (see `gisco_get`)

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

Value

A `POLYGON/POINT` object.

Note

Please check the download and usage provisions on `gisco_attributions`.

While `moveCAN` is useful for visualization, it would alter the actual geographical position of the Canary Islands. When using the output for spatial analysis or using tiles (esp_getTiles, addProviderEspTiles) this option should be set to `FALSE` in order to get the actual coordinates.

Author(s)

dieghernan, `https://github.com/dieghernan/`

Source

GISCO API

See Also

esp_nuts.sf, esp_dict_region_code, esp_codelist, gisco_get.

Examples

```
library(sf)

pal <- hcl.colors(5, palette = "Lisbon")

NUTS1 <- esp_get_nuts(nuts_level = 1, moveCAN = TRUE)
plot(st_geometry(NUTS1), col = pal)

NUTS1_alt <- esp_get_nuts(nuts_level = 1, moveCAN = c(15, 0))
plot(st_geometry(NUTS1_alt), col = pal)

NUTS1_orig <- esp_get_nuts(nuts_level = 1, moveCAN = FALSE)
plot(st_geometry(NUTS1_orig), col = pal)

AndOriental <-
  esp_get_nuts(region = c("Almeria", "Granada", "Jaen", "Malaga"))
```

```
plot(st_geometry(AndOriental), col = pal)

RandomRegions <- esp_get_nuts(region = c("ES1", "ES300", "ES51"))
plot(st_geometry(RandomRegions), col = pal)

MixingCodes <- esp_get_nuts(region = c("ES4", "ES-PV", "Valencia"))
plot(st_geometry(MixingCodes), col = pal)
```

esp_get_prov *Get Provinces boundaries of Spain*

Description

Loads a simple feature (`sf`) object containing the province boundaries of Spain.

Usage

```
esp_get_prov(prov = NULL, ...)
```

Arguments

`prov`	A vector of names and/or codes for provinces or `NULL` to get all the provinces. See Details.
`...`	Additional parameters from esp_get_nuts.

Details

When using `prov` you can use and mix names and NUTS codes (levels 1, 2 or 3), ISO codes (corresponding to level 2 or 3) or `cpro`.

Ceuta and Melilla are considered as provinces on this dataset.

When calling a superior level (Autonomous Community or NUTS1) , all the provinces of that level would be added.

Value

A `POLYGON/POINT` object.

Author(s)

dieghernan, `https://github.com/dieghernan/`

See Also

esp_get_hex_prov, esp_get_nuts, esp_get_ccaa, esp_get_munic, esp_codelist

Examples

```
library(sf)

# Random Provinces

Random <-
  esp_get_prov(prov = c("Zamora",
                        "Palencia",
                        "ES-GR",
                        "ES521",
                        "01"))
plot(st_geometry(Random), col = hcl.colors(6))

# All Provinces of a Zone plus an addition

Mix <-
  esp_get_prov(prov = c("Noroeste",
                        "Castilla y Leon", "La Rioja"),
               resolution = "20")
plot(
  Mix[, "nuts1.code"],
  pal = hcl.colors(3),
  key.pos = NULL,
  main = NULL,
  border = "white"
)
```

---

esp_munic.sf — *All Municipalities* POLYGON *object of Spain*

Description

A sf object including all municipalities of Spain as provided by GISCO (2019 version).

Format

A POLYGON data frame (resolution: 1:1million, EPSG:4258) object:

codauto INE code of each autonomous community.

ine.ccaa.name INE name of each autonomous community.

cpro INE code of each province.

ine.prov.name INE name of each province.

cmun INE code of each municipalty.

name Name of the municipality

LAU_CODE LAU Code (GISCO) of the municipality

geometry geometry field

Source

GISCO .geojson source

See Also

esp_get_munic

Examples

```
library(sf)

data("esp_munic.sf")
data("esp_nuts.sf")

Teruel.cpro <- esp_dict_region_code("Teruel", destination = "cpro")
Teruel.NUTS <- esp_dict_region_code(Teruel.cpro,
   origin = "cpro",
   destination = "nuts")

Teruel.sf <- esp_munic.sf[esp_munic.sf$cpro == Teruel.cpro, ]
Teruel.city <- Teruel.sf[Teruel.sf$name == "Teruel", ]

NUTS <-
  esp_nuts.sf[esp_nuts.sf$LEVL_CODE == 3 &
             esp_nuts.sf$NUTS_ID != Teruel.NUTS,]


plot(st_geometry(Teruel.sf), col = "cornsilk")
plot(st_geometry(Teruel.city), col = "firebrick3", add = TRUE)
plot(st_geometry(NUTS), col = "wheat", add = TRUE)
title(main = "Municipalities of Teruel",  line = 1)
```

`esp_nuts.sf` *All NUTS* `POLYGON` *object of Spain*

Description

A `sf` object including all NUTS levels of Spain as provided by GISCO (2016 version).

Format

A `POLYGON` data frame (resolution: 1:1million, EPSG:4258) object with 86 rows and fields:

COAST_TYPE COAST_TYPE

FID FID

NUTS_NAME NUTS name on local alphabet

MOUNT_TYPE MOUNT_TYPE

NAME_LATN Name on Latin characters

CNTR_CODE Eurostat Country code

URBN_TYPE URBN_TYPE

NUTS_ID NUTS identifier

LEVL_CODE NUTS level code (0,1,2,3)

geometry geometry field

Source

GISCO .geojson source

See Also

esp_get_nuts

Examples

```
library(sf)

nuts <- esp_nuts.sf
nuts3 <- subset(nuts, LEVL_CODE == 3)

unique(nuts3$MOUNT_TYPE)

plot(
  nuts3[, "URBN_TYPE"],
  pal = hcl.colors(3, palette = "Viridis"),
  main = "Urban type -  NUTS3 levels of Spain",
  key.pos = NULL
)
```

`leaflet.providersESP.df` *Public WMS and WMTS of Spain*

Description

A data frame containing information of different public WMS and WMTS providers of Spain

This function is a implementation of the javascript plugin leaflet-providersESP **v1.2.0**

Format

A data frame object with a list of the required parameters for calling the service:

provider Provider name

field Description of value

value INE code of each province.

Details

Providers available to be passed to `type` are:

- **IDErioja**: `IDErioja`
- **IGNBase**: `IGNBase.Todo`, `IGNBase.Gris`, `IGNBase.TodoNoFondo`, `IGNBase.Orto`
- **MDT**: `MDT.Elevaciones`, `MDT.Relieve`, `MDT.CurvasNivel`
- **PNOA**: `PNOA.MaximaActualidad`, `PNOA.Mosaico`
- **OcupacionSuelo**: `OcupacionSuelo.Ocupacion`, `OcupacionSuelo.Usos`
- **LiDAR**: `LiDAR`

- **MTN**: `MTN`
- **Geofisica**: `Geofisica.Terremotos10dias`, `Geofisica.Terremotos30dias`, `Geofisica.Terremotos365dias`, `Geofisica.VigilanciaVolcanica`
- **CaminoDeSantiago**: `CaminoDeSantiago.CaminoFrances`, `CaminoDeSantiago.CaminosTuronensis`, `CaminoDeSantiago.CaminosGalicia`, `CaminoDeSantiago.CaminosDelNorte`, `CaminoDeSantiago.CaminosAnc` `CaminoDeSantiago.CaminosCentro`, `CaminoDeSantiago.CaminosEste`, `CaminoDeSantiago.CaminosCatalane` `CaminoDeSantiago.CaminosSureste`, `CaminoDeSantiago.CaminosInsulares`, `CaminoDeSantiago.CaminosPi` `CaminoDeSantiago.CaminosTolosana`, `CaminoDeSantiago.CaminosPortugueses`
- **Catastro**: `Catastro.Catastro`, `Catastro.Parcela`, `Catastro.CadastralParcel`, `Catastro.CadastralZonin` `Catastro.Address`, `Catastro.Building`
- **RedTransporte**: `RedTransporte.Carreteras`, `RedTransporte.Ferroviario`, `RedTransporte.Aerodromo`, `RedTransporte.AreaServicio`, `RedTransporte.EstacionesFerroviario`, `RedTransporte.Puertos`
- **Cartociudad**: `Cartociudad.CodigosPostales`, `Cartociudad.Direcciones`
- **NombresGeograficos**: `NombresGeograficos`
- **UnidadesAdm**: `UnidadesAdm.Limites`, `UnidadesAdm.Unidades`
- **Hidrografia**: `Hidrografia.MasaAgua`, `Hidrografia.Cuencas`, `Hidrografia.Subcuencas`, `Hidrografia.POI`, `Hidrografia.ManMade`, `Hidrografia.LineaCosta`, `Hidrografia.Rios`, `Hidrografia.Humedales`
- **Militar**: `Militar.CEGET1M`, `Militar.CEGETM7814`, `Militar.CEGETM7815`, `Militar.CEGETM682`, `Militar.CECAF1M`
- **ADIF**: `ADIF.Vias`, `ADIF.Nodos`, `ADIF.Estaciones`
- **LimitesMaritimos**: `LimitesMaritimos.LimitesMaritimos`, `LimitesMaritimos.LineasBase`
- **Copernicus**: `Copernicus.LandCover`, `Copernicus.Forest`, `Copernicus.ForestLeaf`, `Copernicus.WaterWet`, `Copernicus.SoilSeal`, `Copernicus.GrassLand`, `Copernicus.Local`, `Copernicus.RiparianGreen`, `Copernicus.RiparianLandCover`, `Copernicus.Natura2k`, `Copernicus.UrbanAtlas`
- **ParquesNaturales**: `ParquesNaturales.Limites`, `ParquesNaturales.ZonasPerifericas`

Source

leaflet-providersESP leaflet plugin, **v1.2.0**.

See Also

esp_getTiles, addProviderEspTiles.

`pobmun19` *Population by municipality (2019)*

Description

A data frame with 8131 rows containing the population data by municipality in Spain (2019).

Source

INE: Instituto Nacional de Estadistica

Index