

Package: mapping (via r-universe)

March 10, 2025

Type Package

Version 1.4.1

Date 2023-12-08

Title Automatic Download, Linking, Manipulating Coordinates for Maps

Description Maps are an important tool to visualise variables distribution across different spatial objects. The mapping process requires to link the data with coordinates and then generate the correspondent map. This package provide coordinates, linking and mapping functions for an automatic, flexible and easy approach of external functions. The package provides an easy, flexible and automatic unit. Geographical coordinates are provided in the package and automatically linked with the input data to generate maps with internal provided functions or external functions. Provide an easy, flexible and automatic approach to potentially download updated coordinates, to link statistical units with coordinates and to aggregate variables based on the spatial hierarchy of units. The object returned from the package can be used for thematic maps with the build-in functions provided in mapping or with other packages already available.

Depends R (>= 3.5)

Imports tmap (>= 3.3-3), cartography (>= 2.3.0), graphics(>= 3.6.1), ggplot2 (>= 3.2.1), sf(>= 1.0-0), utils, stats, dplyr(>= 0.8.3), leaflet(>= 2.0.3), tmaptools(>= 2.0-2), viridisLite(>= 0.3.0), grid(>= 3.6.1), httr(>= 1.4.1), curl(>= 4.3), htmltools(>= 0.5.0), leafpop(>= 0.0.5), leafsync(>= 0.1.0), mapview(>= 2.7.8), geojsonio(>= 0.9.2), jsonlite(>= 1.7.1), stringr(>= 1.4.0), s2(>= 1.0.6), stringi(>= 1.6.2)

Suggests knitr, DiagrammeR(>= 1.0.6.1), rmarkdown, validate, unrepX

URL <https://mappinguniverse.github.io/mapping/index.html>

BugReports <https://github.com/mappinguniverse/mapping/issues>

LazyData yes

LazyDataCompression xz**Encoding** UTF-8**RoxygenNote** 6.1.1**License** GPL (>= 2)**NeedsCompilation** no**Author** Alessio Serafini [aut, cre], Giancarlo Ferrara [aut]**Maintainer** Alessio Serafini <srf.alessio@gmail.com>**Date/Publication** 2023-12-12 12:30:02 UTC**Additional_repositories** <https://cranhaven.r-universe.dev>**Config/pak/sysreqs** libfontconfig1-dev libfreetype6-dev libgdal-dev
gdal-bin libgeos-dev make libicu-dev libjq-dev libpng-dev
libprotobuf-dev libxml2-dev libssl-dev libproj-dev
protobuf-compiler libprotoc-dev libsqlite3-dev libudunits2-dev
libnode-dev libx11-dev zlib1g-dev**Repository** <https://cranhaven.r-universe.dev>**RemoteUrl** <https://github.com/cranhaven/cranhaven.r-universe.dev>**RemoteRef** package/mapping**RemoteSha** 0fbf6aa09dc6eaab3a3b15bc90d21bb99aac37d7**RemoteSubdir** mapping

Contents

checkNamesDE	3
checkNamesEU	5
checkNamesIT	6
checkNamesUK	8
checkNamesUS	9
checkNamesWR	10
DE	12
EU	14
FR	16
getNamesDE	18
getNamesEU	19
getNamesFR	19
getNamesIT	20
getNamesUK	21
getNamesUS	22
getNamesWR	23
IT	23
loadCoordDE	26
loadCoordEU	27
loadCoordFR	28
loadCoordIT	29

loadCoordUK	31
loadCoordUS	32
loadCoordWR	34
mapPalette	35
mapping	36
mapping.options	38
mappingDE	41
mappingEU	43
mappingFR	46
mappingIT	48
mappingUK	51
mappingUS	52
mappingWR	55
names	58
popDE	59
popEU	59
popFR	60
popIT	60
popUK	61
popUS	61
popWR	62
saveObj	62
tax_wedge_ocde	63
UK	64
US	66
usa_election	68
WR	69
Index	72

checkNamesDE	<i>Check Germany names</i>
--------------	----------------------------

Description

Check the differences between the names (or codes) given in input and the names (or codes), of the corresponding selected Germany statistical unit.

Usage

```
checkNamesDE(id, unit = c("state", "district", "municipal", "municipality"),
             matchWith = c("name", "code", "code_full"), return_logical = FALSE,
             print = TRUE, use_internet = TRUE)
```

Arguments

<code>id</code>	character vector with names or codes						
<code>unit</code>	the type of European statistical unit to check						
<code>matchWith</code>	the type of id to check: <table> <tr> <td><code>"name"</code></td> <td>if unit names</td> </tr> <tr> <td><code>"code"</code></td> <td>if unit code</td> </tr> <tr> <td><code>"code_full"</code></td> <td>if unit complete code</td> </tr> </table>	<code>"name"</code>	if unit names	<code>"code"</code>	if unit code	<code>"code_full"</code>	if unit complete code
<code>"name"</code>	if unit names						
<code>"code"</code>	if unit code						
<code>"code_full"</code>	if unit complete code						
<code>return_logical</code>	a logical value indicating whether nomatched id are returned.						
<code>print</code>	a logical value indicating whether print the nomatched names						
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.						

Details

The function provides a check between id name or code in the dataset and the corresponding selected Germany statistical unit. `unit` starts from the largest aggregate, "state", to the smallest, "municipality".

Value

Returns a string vector with nomatched names or a boolean vector indicating whether or not the id matched.

Author(s)

Alessio Serafini

See Also

[checkNamesEU](#), [checkNamesUS](#), [checkNamesWR](#), [checkNamesUK](#)

Examples

```
data("popDE")
ck <- checkNamesDE(popDE$code_state, unit = "state", matchWith = "code_full")
str(ck)
```

checkNamesEU	<i>Check European names</i>
--------------	-----------------------------

Description

Check the differences between the names (or codes) given in input and the names (or codes), as provided by Eurostat, of the corresponding selected European statistical unit.

Usage

```
checkNamesEU(id,
              unit = c("nuts0", "nuts1", "nuts2", "nuts3", "urau"),
              year = c("2021", "2016", "2013", "2010", "2006", "2003"),
              matchWith = c("nuts", "id", "iso2", "iso3", "country_code"),
              scale = c("20", "60"), return_logical = FALSE,
              print = TRUE, use_internet = TRUE)
```

Arguments

id	character vector with names or codes	
unit	the type of European statistical unit to check	
year	year of the analysis	
matchWith	the type of id to check:	
	"nuts"	if nuts names
	"id"	if nuts id
	"iso2"	if iso2 code
	"iso3"	if iso3 code
	"country_code"	if Eurostat code
scale	the scale of the map.	
return_logical	a logical value indicating whether nomatched id are returned.	
print	a logical value indicating whether print the nomatched names	
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.	

Details

The function provides a check between id name in the dataset and the European statistical unit. unit starts from the largest aggregate, "nuts0" (European country), to the smallest, "nuts3". Since unit can change over the years, the year of the data has to be provided.

The single unit can be coded in different ways, with names, id or iso standard.

Value

Returns a string vector with nomatched names or a boolean vector indicating whether or not the id matched.

Author(s)

Alessio Serafini

See Also

[checkNamesIT](#), [checkNamesUS](#), [checkNamesWR](#)

Examples

```
data("popEU")
data("popEUnuts2")

# Check only the country
ck <- checkNamesEU(id = popEU$GEO,
                   unit = "nuts0", matchWith = "id")
ck1 <- checkNamesEU(id = popEU$GEO, unit = "nuts0",
                   matchWith = "id", return_logical = TRUE,
                   print = FALSE)

popEU[ck1,]

ck2 <- checkNamesEU(id = popEUnuts2$GEO,
                   unit = "nuts2",
                   matchWith = "id")
```

checkNamesIT

Check Italian names

Description

Check the differences between the names (or codes) given in input and the corresponding names (or codes), as provided by ISTAT, of the selected Italian statistical unit.

Usage

```
checkNamesIT(id,
             unit = c("ripartizione", "regione", "provincia", "comune"),
             year = c("2021", "2020", "2019", "2018", "2017"),
             matchWith = c("name", "code", "number"),
             return_logical = FALSE, print = TRUE, use_internet = TRUE)
```

Arguments

<code>id</code>	character vector with names or codes
<code>unit</code>	the type of Italian statistical unit to check
<code>year</code>	year of the analysis
<code>matchWith</code>	the type of id to check: "name" if unit names "code" if unit code "number" if unit number code
<code>return_logical</code>	a logical value indicating whether nomatched id are returned
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.

Details

The function provides a check between id name or code in the dataset and the corresponding selected Italian statistical unit. `unit` starts from the largest aggregate, "ripartizione", to the smallest, "comune". Since unit can change over the years, the year of the data has to be provided.

Value

Returns a string vector with nomatched names or a boolean vector indicating whether or not the id matched.

Author(s)

Alessio Serafini

See Also

[checkNamesEU](#), [checkNamesUS](#), [checkNamesWR](#)

Examples

```
data("popIT")
ck <- checkNamesIT(popIT$ID, unit = "provincia")
str(ck)
```

```
ck <- checkNamesIT(popIT$ID, unit = "provincia", return_logical = TRUE)
str(ck)
```

checkNamesUK	<i>Check United Kingdom names</i>
--------------	-----------------------------------

Description

Check the differences between the names (or codes) given in input and the names (or codes) of the corresponding selected United Kingdom statistical unit.

Usage

```
checkNamesUK(id, unit = c("country", "county"),
             year = c("2020", "2019"),
             matchWith = c("name", "code"),
             scale = c("500", "20"), return_logical = FALSE,
             print = TRUE, use_internet = TRUE)
```

Arguments

id	character vector with names or codes				
unit	the type of European statistical unit to check				
year	year of the analysis				
matchWith	the type of id to check: <table> <tr> <td>"name"</td> <td>if unit names</td> </tr> <tr> <td>"code"</td> <td>if unit code</td> </tr> </table>	"name"	if unit names	"code"	if unit code
"name"	if unit names				
"code"	if unit code				
scale	the scale of the map.				
return_logical	a logical value indicating whether nomatched id are returned.				
print	a logical value indicating whether print the nomatched names				
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.				

Details

The function provides a check between id name or code in the dataset and the corresponding selected United Kingdom statistical unit. `unit` starts from the largest aggregate, "country", to the smallest, "county". Since `unit` can change over the years, the year of the data has to be provided.

Value

Returns a string vector with nomatched names or a boolean vector indicating whether or not the id matched.

Author(s)

Alessio Serafini

See Also[checkNamesEU](#), [checkNamesUS](#), [checkNamesWR](#), [checkNamesDE](#)**Examples**

```
data("popUK")
ck <- checkNamesUK(popUK$name, unit = "country")
str(ck)
```

checkNamesUS	<i>Check USA names</i>
--------------	------------------------

Description

Check the differences between the names given in input and the names, as provided by United States Census of Bureau, of the corresponding USA statistical unit.

Usage

```
checkNamesUS(id,
              unit = c("country", "region", "division", "state",
                       "county", "district",
                       "district_county", "urban_area"),
              year = c("2018"), matchWith = c("name", "id", "number"),
              scale = c("20", "50", "500"), return_logical = FALSE,
              print = TRUE, use_internet = TRUE)
```

Arguments

id	character vector with names
unit	the type of USA statistical unit to check
year	year of the analysis
matchWith	the type of id to check if unit is set to "states"
scale	the scale of the map
return_logical	a logical value indicating whether nomatched id are returned.
print	a logical value indicating whether print the nomatched names
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.

Details

The function provides a check between id names in the dataset and the USA unit. unit starts from the largest aggregate, "country", to the smallest, "district". Since unit can change over the years, the year of the data has to be provided.

The single state can be coded in different ways, with names, id or number.

Value

Returns a string vector with nomatched names or a boolean vector indicating whether or not the id matched.

See Also

[checkNamesIT](#), [checkNamesEU](#), [checkNamesWR](#)

Examples

```
data("popUS")

ck <- checkNamesUS(popUS$id, unit = "state")
```

checkNamesWR	<i>Check World country names</i>
--------------	----------------------------------

Description

Check the differences between the names (or codes) given in input and the names (or codes) of the worldwide countries.

Usage

```
checkNamesWR(id,
  unit = c("country", "nato", "ocde",
    "continent", "region", "subregion",
    "region_wb", "type_income", "type_economy"),
  matchWith = c("country", "iso2", "iso3", "iso3_eh",
    "iso3_numeric", "iso3_un", "iso2_wb",
    "iso3_wb", "name_formal", "name_wb"),
  res = c("low", "hi"), return_logical = FALSE,
  print = TRUE, use_internet = TRUE)
```

Arguments

id	character vector with names or codes																				
unit	the type of world statistical unit																				
matchWith	the type of id to check: <table> <tr> <td>"country"</td> <td>if country names</td> </tr> <tr> <td>"iso2"</td> <td>if iso2 code</td> </tr> <tr> <td>"iso3"</td> <td>if iso3 code.</td> </tr> <tr> <td>"iso3_eh"</td> <td>if iso3_eh code</td> </tr> <tr> <td>"iso3_numeric"</td> <td>if iso3 numeric code</td> </tr> <tr> <td>"iso3_un"</td> <td>if iso3 United Nations</td> </tr> <tr> <td>"iso2_wb"</td> <td>if iso2 World Bank</td> </tr> <tr> <td>"iso3_wb"</td> <td>if iso3 World Bank</td> </tr> <tr> <td>"name_formal"</td> <td>if formal names</td> </tr> <tr> <td>"name_wb"</td> <td>if World Bank names</td> </tr> </table>	"country"	if country names	"iso2"	if iso2 code	"iso3"	if iso3 code.	"iso3_eh"	if iso3_eh code	"iso3_numeric"	if iso3 numeric code	"iso3_un"	if iso3 United Nations	"iso2_wb"	if iso2 World Bank	"iso3_wb"	if iso3 World Bank	"name_formal"	if formal names	"name_wb"	if World Bank names
"country"	if country names																				
"iso2"	if iso2 code																				
"iso3"	if iso3 code.																				
"iso3_eh"	if iso3_eh code																				
"iso3_numeric"	if iso3 numeric code																				
"iso3_un"	if iso3 United Nations																				
"iso2_wb"	if iso2 World Bank																				
"iso3_wb"	if iso3 World Bank																				
"name_formal"	if formal names																				
"name_wb"	if World Bank names																				
res	map resolution																				
return_logical	a logical value indicating whether nomatched id are returned																				
print	a logical value indicating whether print the nomatched names																				
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.																				

Details

The function provides a check between id name in the dataset and the worldwide country names. The single unit can be coded in different ways, with names, id or iso standards.

Value

Returns a string vector with no matched names or a boolean vector indicating whether or not the id matched.

Author(s)

Alessio Serafini

See Also

[checkNamesIT](#), [checkNamesEU](#), [checkNamesUS](#)

Examples

```
data("popWR")

ck <- checkNamesWR(id = popWR$country, matchWith = "country")
ck
ck1 <- checkNamesWR(id = popWR$country_code, matchWith = "iso3", return_logical = TRUE)
ck1
```

DE

*Object of class UK***Description**

Creates an object with data and coordinates of class DE for Germany statistical units to use with mapping functions or available in other R "maps" packages.

Usage

```
DE(data, colID = NULL,
    unit = c("state", "district", "municipal", "municipality"),
    matchWith = c("name", "code", "code_full"), subset = NULL,
    add = NULL, new_var_names = NULL, aggregation_fun = sum,
    aggregation_unit = NULL, aggregation_var = NULL, facets = NULL,
    check.unit.names = TRUE, dir = NULL, use_cache = TRUE,
    print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

<code>data</code>	a data.frame object with variables to display						
<code>colID</code>	character value or columns number indicating the column with unit names or codes						
<code>unit</code>	the type of Italian statistical unit						
<code>matchWith</code>	the type of id to check: <table> <tbody> <tr> <td><code>"name"</code></td> <td>if unit names</td> </tr> <tr> <td><code>"code"</code></td> <td>if unit code</td> </tr> <tr> <td><code>"code_full"</code></td> <td>if unit complete code</td> </tr> </tbody> </table>	<code>"name"</code>	if unit names	<code>"code"</code>	if unit code	<code>"code_full"</code>	if unit complete code
<code>"name"</code>	if unit names						
<code>"code"</code>	if unit code						
<code>"code_full"</code>	if unit complete code						
<code>subset</code>	a formula indicating the condition to subset the data, see the Details						
<code>add</code>	a formula to add new transformed variables starting from the variables in the data						
<code>new_var_names</code>	a character value or vector indicating the names of the new variables created in add						
<code>aggregation_fun</code>	function to use when data are aggregated						
<code>aggregation_unit</code>	variable name by which the unit are aggregated						
<code>aggregation_var</code>	variable name with value to aggregate						
<code>facets</code>	variable(s) name to split the data						
<code>check.unit.names</code>	a logical value indicating if the colID names are checked with unit names						

<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package, the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class DE, with data and coordinates to use in functions which perform map.

See Also

[EU](#), [WR](#), [US](#), [UK](#)

Examples

```
data("popDE")
de <- DE(data = popDE, colID = "code_state", unit = "state", matchWith = "code_full")

### Adding two variables

de2 <- DE(data = popDE, colID = "code_state", unit = "state", matchWith = "code_full",
          add = ~I(population_2020/1000) + I(population_2020/100) )

### Adding to variables and names
de3 <- DE(data = popDE, colID = "code_state", unit = "state", matchWith = "code_full",
          add = ~I(population_2020/1000) + I(population_2020/100),
          new_var_names = c("ratio1", "ratio2"))
```

EU *Object of class EU*

Description

Creates an object with data and coordinates of class EU for European countries to use with mapping functions or available in other R "maps" packages.

Usage

```
EU(data, colID = NULL,
    unit = c("nuts0", "nuts1", "nuts2", "nuts3", "urau"),
    year = c("2021", "2016", "2013", "2010", "2006", "2003"),
    matchWith = c("nuts", "id", "iso2", "iso3", "country_code"),
    scale = c("20", "60"), show_eu = TRUE,
    subset = NULL, add = NULL, new_var_names = NULL,
    aggregation_fun = sum, aggregation_unit = NULL, aggregation_var = NULL,
    facets = NULL, check.unit.names = TRUE, dir = NULL,
    use_cache = TRUE, print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

data	a data.frame object with variables to display										
colID	character value or columns number indicating the column with unit names										
unit	the type of European statistical unit										
year	year of the analysis										
matchWith	the type of id to check: <table style="margin-left: 2em;"> <tbody> <tr> <td>"nuts"</td> <td>if nuts names</td> </tr> <tr> <td>"id"</td> <td>if nuts id</td> </tr> <tr> <td>"iso2"</td> <td>if iso2 code</td> </tr> <tr> <td>"iso3"</td> <td>if iso3 code</td> </tr> <tr> <td>"country_code"</td> <td>if Eurostat code</td> </tr> </tbody> </table>	"nuts"	if nuts names	"id"	if nuts id	"iso2"	if iso2 code	"iso3"	if iso3 code	"country_code"	if Eurostat code
"nuts"	if nuts names										
"id"	if nuts id										
"iso2"	if iso2 code										
"iso3"	if iso3 code										
"country_code"	if Eurostat code										
scale	the scale of the map										
show_eu	logical value set to TRUE indicating if the entire map is drawn or only the coordinates linked to the input data										
subset	a formula indicating the condition to subset the data, see the Details										
add	a formula to add new transformed variables starting from the variables in the data										
new_var_names	a character value or vector indicating the names of the new variables created in add										
aggregation_fun	function to use when data are aggregated										

<code>aggregation_unit</code>	variable name by which the unit are aggregated
<code>aggregation_var</code>	variable name with value to aggregate
<code>facets</code>	variable(s) name to split the data
<code>check.unit.names</code>	a logical value indicating if the colID names are checked with unit names
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package (as provided by Eurostat), the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class EU, with data and coordinates to use in functions which perform map.

See Also

[WR](#), [IT](#), [US](#), [DE](#), [UK](#)

Examples

```
data("popEU")

popEU <- popEU

euNuts2 <- EU(data = popEU, colID = "GEO",
              unit = "nuts2", matchWith = "id")
str(euNuts2)

euNuts2_1 <- EU(data = popEU, colID = "GEO",
```

```

      unit = "nuts2",matchWith = "id",
      add = ~I(male/total) + I(female/total))
str(euNuts2_1)

euNuts2_2 <- EU(data = popEU, colID = "GEO",
  unit = "nuts2",matchWith = "id",
  add = ~I(male/total) + I(female/total),
  new_var_names = c("Per_Male", "Per_Female"))
str(euNuts2_2)

```

FR

*Object of class FR***Description**

Creates an object with data and coordinates of class FR for France statistical units to use with mapping functions or available in other R "maps" packages.

Usage

```

FR(data, colID = NULL, unit = c("region"),
  year = c("2021", "2020", "2019"), matchWith = c("name", "code"),
  subset = NULL, add = NULL, new_var_names = NULL,
  aggregation_fun = sum, aggregation_unit = NULL, aggregation_var = NULL,
  facets = NULL, check.unit.names = TRUE, dir = NULL, use_cache = TRUE,
  print = FALSE, use_internet = TRUE, crs = NULL)

```

Arguments

data	a data.frame object with variables to display				
colID	character value or columns number indicating the column with unit names or codes				
unit	the type of Italian statistical unit				
year	year of the analysis				
matchWith	the type of id to check: <table style="margin-left: 2em;"> <tbody> <tr> <td>"name"</td> <td>if unit names</td> </tr> <tr> <td>"code"</td> <td>if unit code</td> </tr> </tbody> </table>	"name"	if unit names	"code"	if unit code
"name"	if unit names				
"code"	if unit code				
subset	a formula indicating the condition to subset the data, see the Details				
add	a formula to add new transformed variables starting from the variables in the data				
new_var_names	a character value or vector indicating the names of the new variables created in add				

<code>aggregation_fun</code>	function to use when data are aggregated
<code>aggregation_unit</code>	variable name by which the unit are aggregated
<code>aggregation_var</code>	variable name with value to aggregate
<code>facets</code>	variable(s) name to split the data
<code>check.unit.names</code>	a logical value indicating if the <code>colID</code> names are checked with <code>unit</code> names
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package, the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class `FR`, with data and coordinates to use in functions which perform map.

See Also

[EU](#), [WR](#), [US](#), [DE](#)

Examples

```
data("popFR")

fr <- FR(data = popFR)

### Adding two variables

fr2 <- FR(data = popFR,
```

```
add = ~I(population/1000) + I(population/100) )

### Adding to variables and names
fr3 <- FR(data = popFR,
          add = ~I(population/1000) + I(population/100),
          new_var_names = c("ratio1", "ratio2"))
```

getNamesDE	<i>Germany names</i>
------------	----------------------

Description

Retrieves Germany statistical unit names.

Usage

```
getNamesDE(unit = c("state", "district", "municipal", "municipality"),
           all_levels = TRUE)
```

Arguments

unit	the type of statistical units
all_levels	a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels

See Also

[getNamesIT](#), [getNamesEU](#), [getNamesWR](#), [getNamesUK](#)

Examples

```
getNamesDE()

getNamesDE(unit = "district")
getNamesDE(unit = "district", all_levels = FALSE)
```

getNamesEU	<i>European names</i>
------------	-----------------------

Description

Retrieves European statistical unit names.

Usage

```
getNamesEU(year = c("2021", "2016", "2013", "2010", "2006", "2003"),
            unit = c("nuts0", "nuts1", "nuts2", "nuts3"), id = FALSE, all_levels = TRUE)
```

Arguments

year	year of the analysis
unit	the type of statistical unit
id	boolean value indicating whether the ids are returned instead of names
all_levels	a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels.

See Also

[getNamesIT](#), [getNamesUS](#), [getNamesWR](#), [getNamesUK](#), [getNamesDE](#)

Examples

```
getNamesEU()

getNamesEU(unit = "nuts1")
getNamesEU(unit = "nuts1", all_levels = FALSE, id = FALSE)
getNamesEU(unit = "nuts1", all_levels = FALSE, id = TRUE)
```

getNamesFR	<i>France names</i>
------------	---------------------

Description

Retrieves France statistical unit names.

Usage

```
getNamesFR(year = c("2021", "2020", "2019"),
            unit = c("region"), all_levels = TRUE)
```

Arguments

year year of the analysis
unit the type of statistical units
all_levels a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels

See Also

[getNamesIT](#), [getNamesEU](#), [getNamesWR](#), [getNamesDE](#)

Examples

```
getNamesFR()
```

```
getNamesFR(all_levels = FALSE)
```

getNamesIT

Italian names

Description

Retrieves Italian statistical unit names.

Usage

```
getNamesIT(year = c("2021", "2020", "2019", "2018", "2017"),  
          unit = c("ripartizione", "regione", "provincia", "comune"), all_levels = TRUE)
```

Arguments

year year of the analysis
unit the type of Italian statistical unit
all_levels a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels.

See Also

[getNamesEU](#), [getNamesUS](#), [getNamesWR](#), [getNamesUK](#), [getNamesDE](#)

Examples

```
getNamesIT()
getNamesIT(unit = "provincia")
getNamesIT(unit = "provincia", all_levels = FALSE)
```

getNamesUK	<i>United Kingdom names</i>
------------	-----------------------------

Description

Retrieves United Kingdom statistical unit names.

Usage

```
getNamesUK(year = c("2020", "2019"),
            unit = c("country", "county"),
            all_levels = TRUE)
```

Arguments

year	year of the analysis
unit	the type of statistical units
all_levels	a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels

See Also

[getNamesIT](#), [getNamesEU](#), [getNamesWR](#), [getNamesDE](#)

Examples

```
getNamesUK()

getNamesUS(unit = "county")
getNamesUK(unit = "county", all_levels = FALSE)
```

`getNamesUS`*USA names*

Description

Retrieves USA statistical unit names.

Usage

```
getNamesUS(year = "2018",
            unit = c("region", "division", "state", "county",
                    "district", "district_county", "urban_area"),
            id = FALSE, all_levels = TRUE)
```

Arguments

<code>year</code>	year of the analysis
<code>unit</code>	the type of statistical units
<code>id</code>	boolean value indicating whether the ids are returned instead of names
<code>all_levels</code>	a logical value indicating if all levels are returned or only the unit names

Value

A character vector or a data frame with unit names and corresponding associated levels

See Also

[getNamesIT](#), [getNamesEU](#), [getNamesWR](#), [getNamesUK](#), [getNamesDE](#)

Examples

```
getNamesUS()

getNamesUS(unit = "state")
getNamesUS(unit = "state", all_levels = FALSE)

getNamesUS(unit = "county")
getNamesUS(unit = "county", all_levels = FALSE)
```

getNamesWR	<i>World countries names</i>
------------	------------------------------

Description

Retrieves world country names, ids and iso.

Usage

```
getNamesWR(unit = c("all", "country", "name_formal",  
                    "name_wb", "iso2", "iso3",  
                    "iso3_eh", "iso3_numeric", "iso3_un",  
                    "iso2_wb", "iso3_wb"))
```

Arguments

unit the type of names

Value

A character vector or a data frame with unit names and corresponding associated levels.

See Also

[getNamesIT](#), [getNamesUS](#), [getNamesEU](#), [getNamesUK](#), [getNamesDE](#)

Examples

```
getNamesWR()  
getNamesWR("iso3")
```

IT	<i>Object of class IT</i>
----	---------------------------

Description

Creates an object with data and coordinates of class IT for Italy to use with mapping functions or available in other R "maps" packages.

Usage

```
IT(data, colID = NULL,
    unit = c("none", "ripartizione", "regione", "provincia", "comune"),
    year = c("2021", "2020", "2019", "2018", "2017"),
    matchWith = c("name", "code", "number"),
    show_it = TRUE, subset = NULL, add = NULL,
    new_var_names = NULL, aggregation_fun = sum,
    aggregation_unit = NULL, aggregation_var = NULL,
    facets = NULL, check.unit.names = TRUE, dir = NULL,
    use_cache = TRUE, print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

<code>data</code>	a data.frame object with variables to display						
<code>colID</code>	character value or columns number indicating the column with unit names or codes						
<code>unit</code>	the type of Italian statistical unit						
<code>year</code>	year of the analysis						
<code>matchWith</code>	the type of id to check: <table style="margin-left: 40px;"> <tr> <td><code>"name"</code></td> <td>if unit names</td> </tr> <tr> <td><code>"code"</code></td> <td>if unit code</td> </tr> <tr> <td><code>"number"</code></td> <td>if unit number code</td> </tr> </table>	<code>"name"</code>	if unit names	<code>"code"</code>	if unit code	<code>"number"</code>	if unit number code
<code>"name"</code>	if unit names						
<code>"code"</code>	if unit code						
<code>"number"</code>	if unit number code						
<code>show_it</code>	logical value set to TRUE indicating if the entire map is drawn or only the coordinates linked to the input data						
<code>subset</code>	a formula indicating the condition to subset the data, see the Details						
<code>add</code>	a formula to add new transformed variables starting from the variables in the data						
<code>new_var_names</code>	a character value or vector indicating the names of the new variables created in add						
<code>aggregation_fun</code>	function to use when data are aggregated						
<code>aggregation_unit</code>	variable name by which the unit are aggregated						
<code>aggregation_var</code>	variable name with value to aggregate						
<code>facets</code>	variable(s) name to split the data						
<code>check.unit.names</code>	a logical value indicating if the colID names are checked with unit names						
<code>dir</code>	local directory in which shape files are stored						
<code>use_cache</code>	a logical value indicating whether use the cache						
<code>print</code>	a logical value indicating whether print the nomatched names						

`use_internet` a logical value indicating wheter the coordinates are downloaded from <https://github.com/mappinguniverse/geospatial>. If FALSE the maps downloaded during package installation will be used

`crs` coordinate reference system. Look at [st_crs](#)

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package (as provided by ISTAT), the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class IT, with data and coordinates to use in functions which perform map.

See Also

[EU](#), [WR](#), [US](#), [DE](#), [UK](#)

Examples

```
data("popIT")

it <- IT(data = popIT, unit = "provincia", year = "2019")

### Adding two varaibles

it2 <- IT(data = popIT, unit = "provincia", year = "2019",
          add = ~I(maschi/totale) + I(femmine/totale) )

### Adding to variables and names
it3 <- IT(data = popIT, unit = "provincia", year = "2019",
          add = ~I(maschi/totale) + I(femmine/totale),
          new_var_names = c("Per_Maschi", "Per_Femmine") )
```

`loadCoordDE`*Get Germany coordinates*

Description

Loads and returns names, id, and coordinates for Germany statistical unit, to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordDE(unit = c("state", "district", "municipal", "municipality"),
            unit_subset = NULL, matchWith = NULL, dir = NULL,
            use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

<code>unit</code>	the type of Italian statistical unit to link
<code>unit_subset</code>	character vector of unit names to extract
<code>matchWith</code>	the type of id
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether to use the cache
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from DE folder <https://github.com/mappinguniverse/geospatial/tree/master/DE>.

If unit is not specified, state borders are loaded.

Value

A data.frame object with column indicating names, id, and the geometry to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordEU](#), [loadCoordWR](#), [loadCoordUS](#), [loadCoordUK](#)

Examples

```
DE_coords = loadCoordDE(unit = "state")
str(DE_coords)

## Load subset

coords_de <- loadCoordDE(unit = "district",
                          unit_subset = "bayern",
                          matchWith = "state")
```

loadCoordEU

Get European coordinates

Description

Loads and returns names, id, and coordinates for European countries, to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordEU(unit = c("nuts0", "nuts1", "nuts2", "nuts3", "urau"),
            year = c("2021", "2016", "2013", "2010", "2006", "2003"),
            scale = c("20", "60"), unit_subset = NULL,
            matchWith = NULL, dir = NULL,
            use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

unit	the type of European statistical unit to link
year	year of the analysis
scale	the scale of the map
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored

use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
crs	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from EU folder <https://github.com/mappinguniverse/geospatial/tree/master/EU>.

If unit is not specified, borders of the European countries are loaded.

Value

A data.frame object with columns indicating names, ids, iso and the geometries to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordIT](#), [loadCoordWR](#), [loadCoordDE](#), [loadCoordUK](#)

Examples

```
EU_coords = loadCoordEU(unit = "nuts0")
str(EU_coords)

coords_eu_it_de <- loadCoordEU(unit = "nuts0", unit_subset = c("italy", "germany"))
str(coords_eu_it_de)
```

loadCoordFR

Get France coordinates

Description

Loads and returns names, id, and coordinates for France statistical unit, to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordFR(unit = c("region"), year = c("2021", "2020", "2019"),
            unit_subset = NULL, matchWith = NULL, dir = NULL,
            use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

unit	the type of Italian statistical unit to link
year	year of the analysis
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored
use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
crs	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from FR folder <https://github.com/mappinguniverse/geospatial/tree/master/FR>.

If unit is not specified, country borders are loaded.

Value

A data.frame object with column indicating names, id, and the geometry to map.

Examples

```
FR_coords = loadCoordFR(unit = "region", year = "2020")
str(FR_coords)
```

loadCoordIT

Get Italian coordinates

Description

Loads and returns names, ids, and coordinates for Italian statistical unit, ready to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordIT(unit = c("none", "ripartizione", "regione", "provincia", "comune"),
            year = c("2021", "2020", "2019", "2018", "2017"),
            unit_subset = NULL, matchWith = NULL,
            dir = NULL, use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

unit	the type of Italian statistical unit to link
year	year of the analysis
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored
use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
crs	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from IT folder <https://github.com/mappinguniverse/geospatial/tree/master/IT>.

unit="none" (default) indicates that the border of Italy is returned.

Value

A data.frame object with column indicating names, id, and the geometry to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordEU](#), [loadCoordWR](#), [loadCoordUS](#), [loadCoordDE](#), [loadCoordUK](#)

Examples

```
IT_coords = loadCoordIT(unit = "regione", year = "2020")
str(IT_coords)

## Italy

IT_coords = loadCoordIT()
str(IT_coords)

coords_it<- loadCoordIT(unit = "regione", unit_subset = c(5, 10), matchWith = "number")
str(coords_it)
```

loadCoordUK	<i>Get United Kingdom coordinates</i>
-------------	---------------------------------------

Description

Loads and returns names, id, and coordinates for United Kingdom statistical unit, to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordUK(unit = c("country", "county"),
            year = c("2020", "2019"), scale = c("500", "20"),
            unit_subset = NULL, matchWith = NULL, dir = NULL,
            use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

unit	the type of Italian statistical unit to link
year	year of the analysis
scale	the scale of the map
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored
use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
crs	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from UK folder <https://github.com/mappinguniverse/geospatial/tree/master/UK>.

If unit is not specified, country borders are loaded.

Value

A data.frame object with column indicating names, id, and the geometry to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordEU](#), [loadCoordWR](#), [loadCoordUS](#), [loadCoordDE](#)

Examples

```
UK_coords = loadCoordUK(unit = "country", year = "2020")
str(UK_coords)
```

```
## Load subset
```

```
coords_uk <- loadCoordUK(unit = "county", unit_subset = "england", matchWith = "country")
coords_uk <- loadCoordUK(unit = "county", unit_subset = "hartlepool", matchWith = "county")
```

loadCoordUS

Get USA coordinates

Description

Loads and returns names, ids, and coordinates for USA, to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordUS(unit = c("country", "region", "division", "state",  
                    "county", "district", "district_county", "urban_area"),  
            year = c("2018"), scale = c("20", "50", "500"),  
            unit_subset = NULL, matchWith = NULL, dir = NULL,  
            use_cache = TRUE, use_internet = TRUE, crs = NULL)
```

Arguments

unit	type of USA unit to link
year	year of the analysis
scale	the scale of the map
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored
use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
crs	coordinate reference system. Look at st_crs

Details

Coordinates are downloaded from the Github repo <https://github.com/mappinguniverse/geospatial> from US folder <https://github.com/mappinguniverse/geospatial/tree/master/US>.

If unit is not specified, borders of the USA countries are loaded.

Value

A data.frame object with columns indicating names, ids, and the geometry to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordIT](#), [loadCoordWR](#), [loadCoordDE](#), [loadCoordUK](#), [loadCoordEU](#)

Examples

```
US_coords = loadCoordUS(unit = "state")
str(US_coords)
```

```
coords_us<- loadCoordUS(unit = "state", unit_subset = c("Florida", "California"))
```

loadCoordWR

Get worldwide countries coordinates

Description

Loads and returns names, ids, iso, and coordinates for world countries, ready to use with mapping functions and other "map" functions that accept an sf object.

Usage

```
loadCoordWR(unit = c("country", "nato", "ocde",
                    "continent", "region", "subregion",
                    "region_wb", "type_income", "type_economy"),
            res = c("low", "hi"), unit_subset = NULL,
            matchWith = NULL, dir = NULL, use_cache = TRUE,
            use_internet = TRUE, crs = NULL)
```

Arguments

unit	the type of world statistical unit
res	resolution
unit_subset	character vector of unit names to extract
matchWith	the type of id
dir	local directory in which shape files are stored
use_cache	a logical value indicating whether to use the cache
use_internet	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used.
crs	coordinate reference system. Look at st_crs

Details

Coordinates are download from the Github repo <https://github.com/mappinguniverse/geospatial> from world folder <https://github.com/mappinguniverse/geospatial/tree/master/world>.

Value

A data.frame object with column indicating names, id, iso and the geometry to map.

Author(s)

Alessio Serafini

References

<https://github.com/mappinguniverse/geospatial>

See Also

[loadCoordIT](#), [loadCoordEU](#), [loadCoordUS](#), [loadCoordDE](#), [loadCoordUK](#)

Examples

```
WR_coords = loadCoordWR(res = "low")
str(WR_coords, 1)
```

```
WR_ocde = loadCoordWR(unit = "ocde",res = "low")
str(WR_ocde, 1)
```

```
WR_continent = loadCoordWR(unit = "continent",res = "low")
str(WR_continent, 1)
```

mapPalette

Color palette

Description

Returns different color palette

Usage

```
mapPalette(type, nclass = NULL )
```

Arguments

type	character value indicating the color palette
nclass	number of classes

Value

A character vector with palettes.

mapping	<i>Static maps</i>
---------	--------------------

Description

Function to produce static maps from an object of class sf, IT, EU, US, or WR.

Usage

```
mapping(data = NULL, var = NULL, colID = NULL,
        type = c("static", "interactive"),
        typeStatic = c("tmap", "choro.cart", "typo", "bar"),
        add_text = NULL, subset = NULL, facets = NULL, aggregation_fun = sum,
        aggregation_unit = NULL, options = mapping.options(), ...)
```

Arguments

data	an object of class sf, IT, EU, US, or WR
var	character value(s) or columns number(s) indicating the variable to plot
colID	character value or columns number indicating the column with unit names
type	if generates static or interactive map
typeStatic	type of static map
add_text	character name indicating the column with text labels
subset	a formula indicating the condition to subset the data. See the details
facets	variable(s) name to split the data
aggregation_fun	function to use when data are aggregated
aggregation_unit	variable name by which the unit are aggregate
options	a list with options using mapping.options function
...	further arguments

Details

It is a general function to map data. We can externally provide the coordinates with the variable to map, or the coordinates and the data to link.

If coordinates are provided and data is NULL, the function map the var in coordinates. If data is not NULL, then the function link data and coordinates, and the var is get from the data provided in input. If only data are provided without coordinates, the function search the colID among the the coordinates dataset provided by <https://github.com/mappinguniverse/geospatial>, to link the ids with coordinates. For search look at SearchNames

Value

Returns a map. For tmap type, the function also returns a tmap object.

References

- Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.
- Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>
- Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, 84(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingIT](#), [mappingEU](#)

Examples

```
library(dplyr)
library(sf)

data("popIT")
popIT <- popIT
coords <- loadCoordIT(unit = "provincia", year = '2019')
cr <- left_join(coords, popIT, by = c("provincia" = "ID"))

#####
# Statics #
#####

mapping(cr)

mapping(cr, var = "maschi")

nc = st_read(system.file("shape/nc.shp", package="sf"))
class(nc)
mapping(nc)
mapping(nc, var = "AREA", options = mapping.options(legend.position = c("left", "bottom")))

#####
# Interactive #
#####

mapping(cr, type = "interactive")
mapping(cr, var = "maschi", type = "interactive")

nc = st_read(system.file("shape/nc.shp", package="sf"))
class(nc)
mapping(nc, type = "interactive")
mapping(nc, var = "AREA", type = "interactive")
```

mapping.options	<i>Default values for mapping functions</i>
-----------------	--

Description

Set or retrieve default values used in mapping functions availables in **mapping** package.

Usage

```
mapping.options(...)
```

Arguments

... A single character vector, or a named list. The form name = value can be used to change a single option or list(name1 = value1, name2 = value2) can be used to change several arguments. If no arguments are provided, then the function returns all the current options.

Details

The function change globally the option for the current R session, and locally if used in the mapping function, with the options argument, for example, options = mapping.options(legend.frame = FALSE, "title.position" = "left").

Many different options are used for the function in **tmap** package. For more details, look at [tm_layout](#), [tm_borders](#), and [tm_fill](#).

Available options are the following:

palette.cont = "YlGnBu" palette for countinuous data

palette.cat = "Accent" palette for categorical data

palette.cont.vector = NULL a string vector with color names for continuous data

palette.cat.vector = NULL a string vector with color names for categorical data

nclass = 5 number of classes for continuous data

check.unit.names = TRUE a logical value indicating whether the input id names are checked before the link with the coordinates

use_cache = TRUE a logical value indicating whether the cache is used to load the shape file

use_internet = TRUE a logical value indicating whether the data are downloaded from internet or whether a internet connection is available

alpha = 1 transparency

breaks = NULL a numerical value indicating the breaks

interval.closure = "left" a logical value indicating where the interval are closed

labels = NULL a character vector with labels of the classes
NA.color = "grey" color for NA values
NA.text = "Missing" label for NA values
col.style = "order" type of color scale for numeric data. For other method look ad [tm_fill](#)
map.frame = TRUE a logical value indicating whether the frame is drawn
border.lwd = 1 line width of the borders
border.col = "black" color of the borders
border.type = "solid" border type
border.alpha = NA transparency of the borders
title = NULL main title
title.position = "center" main title position
title.color = "black" color of main title
title.fontface = 1 main title font face
title.size = 1 main title size
legend.title = NA title of the legend
legend.show = TRUE a logical value indicating whether include the legend
legend.only = FALSE a logical value indicating whether include the legend without map
legend.position = c("right", "top") legend position
legend.digits = 5 legend digits
legend.outside = FALSE a logical value indicating whether the legend is included outside the map
legend.outside.facetes = TRUE a logical value indicating whether the legend is included outside the facetes
legend.width = 1 width of the legend
legend.title.position = c("right", "top") legend title position
legend.title.size = 1 legend title size
legend.title.fontface = 1 legend title font space
legend.title.color = "black" legend title color
legend.text.color = "black" legend title color
legend.text.size = 0.5 legend title color size
legend.text.align = "left"
legend.text.fontface = 1
legend.frame = TRUE a logical value indicating whether the frame is drawn for the legend
legend.decimal.mark = "."
legend.format = "fg"
legend.big.mark = ", "
legend.text.separator = "-"
facets.free.scale = FALSE

```
facetes.cols = NA
facetes.rows = NA
interactive.tiles = "CartoDB.Positron"
interactive.popup.vars = NULL
interactive.popup.id = TRUE
interactive.popup.closeButton = TRUE
interactive.popup.width.max = 150
interactive.popup.width.min = 35
interactive.highlight.weight = 3
interactive.highlight.color = "black"
interactive.highlight.alpha = 1
interactive.highlight.front = TRUE
interactive.control.collapse = TRUE
interactive.layer.control.position = c("left", "top")
interactive.hovered.id = TRUE
text.size = 0.5
text.col = "black"
text.fontface = 1
text.shadow = FALSE
text.alpha = NA
credits.source = NULL
credits.author = NULL
credits.size = 0.7
credits.fontface = NA
credits.color = "black"
credits.align = "left"
credits.position = c("left", "bottom")
popup.vars = NA a character vector indicating the variable to popup in interactive maps
compass = NULL a character vector indicatin the type of compass (look at tm\_layout)
style = "white" style (look at tm\_style)
crs = NULL
```

Options may be reset using `mapping.options()`.

Value

Return a list with options.

References

Tennekes M (2018). “tmap: Thematic Maps in R.” *Journal of Statistical Software*, *84*(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

Examples

```
mapping.options()

# A single options
mapping.options("title.position")

# Globally
mapping.options("title.position" = "left")
mapping.options("title.position")
```

mappingDE	<i>Static maps for Germany</i>
-----------	--------------------------------

Description

Function to produce static maps for Germany statistical unit.

Usage

```
mappingDE(data, var = NULL, colID = NULL,
          type = c("static", "interactive"),
          typeStatic = c("tmap", "choro.cart", "typo", "bar"),
          unit = c("state", "district", "municipal", "municipality"),
          matchWith = c("name", "code", "code_full"), dir = NULL,
          add_text = NULL, subset = NULL, facets = NULL,
          aggregation_fun = sum, aggregation_unit = NULL,
          options = mapping.options())
```

Arguments

data	a data.frame object with variables to display or a DE object produced by DE function. If object of class DE, arguments unit, year, and matchWith will be ignored
var	character value(s) or columns number(s) indicating the variable to plot
colID	character value or columns number indicating the column with unit names
type	if generates static or interactive map
typeStatic	type of static map

unit	the type of Italian statistical unit						
matchWith	the type of id to check: <table> <tr> <td>"name"</td> <td>if unit names</td> </tr> <tr> <td>"code"</td> <td>if unit code</td> </tr> <tr> <td>"code_full"</td> <td>if unit complete code</td> </tr> </table>	"name"	if unit names	"code"	if unit code	"code_full"	if unit complete code
"name"	if unit names						
"code"	if unit code						
"code_full"	if unit complete code						
dir	local directory in which shape files are stored						
add_text	character name indicating the column with text labels						
subset	a formula indicating the condition to subset the data. See the details section						
facets	variable(s) name to split the data						
aggregation_fun	function to use when data are aggregated						
aggregation_unit	variable name by which the unit are aggregated						
options	a list with options using mapping.options function						

Details

If data is a object of class "DE" generated using the [DE](#) function, the argument unit, because the object already contains the coordinates.

The aggregation_unit provides an aggregation for a user specified variable in data, or for larger statistical unit, automatically provided when the function link the data with the coordinates. For example, if data are of type municipal, we will have variables for larger aggregate unit, that is district and state variables. Look at [DE](#) for more details.

subset provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For tmap type, the function also returns a tmap object.

See Also

[mappingWR](#), [mappingEU](#), [mappingUS](#), [mappingUK](#)

Examples

```
data("popDE")

de <- DE(data = popDE, colID = "code_state",
         unit = "state", matchWith = "code_full",
         check.unit.names = FALSE)
```

```
#####
# Statics #
#####

mappingDE(data = de, var = "population_2020")

mappingDE(data = de, var = "population_2020",
          subset = ~I(state == "bayern"))

#####
# Interactive #
#####

mappingDE(data = de, var = "population_2020", type = "interactive")

mappingDE(data = de, var = "population_2020",
          subset = ~I(state == "bayern"),
          type = "interactive")
```

mappingEU

Static maps for Europe

Description

Function to produce static maps for European statistical unit.

Usage

```
mappingEU(data, var = NULL, colID = NULL,
          type = c("static", "interactive"),
          typeStatic = c("tmap", "choro.cart", "typo", "bar"),
          unit = c("nuts0", "nuts1", "nuts2", "nuts3", "urau"),
          year = c("2021", "2016", "2013", "2010", "2006", "2003"),
          matchWith = c("nuts", "id", "iso2", "iso3", "country_code"),
          scale = c("20", "60"), dir = NULL, show_eu = TRUE,
          add_text = NULL, subset = NULL, facets = NULL,
          aggregation_fun = sum, aggregation_unit = NULL,
          options = mapping.options())
```

Arguments

<code>data</code>	a <code>data.frame</code> object with variables to display or a EU object produced by EU function										
<code>var</code>	character value(s) or columns number(s) indicating the variable to plot										
<code>colID</code>	character value or columns number indicating the column with unit names										
<code>type</code>	if generates static or interactive map										
<code>typeStatic</code>	type of static map										
<code>unit</code>	the type of European statistical unit to check										
<code>year</code>	year of the unit										
<code>matchWith</code>	the type of id to check: <table style="margin-left: 40px;"> <tr> <td><code>"nuts"</code></td> <td>if nuts names).</td> </tr> <tr> <td><code>"id"</code></td> <td>if nuts id.</td> </tr> <tr> <td><code>"iso2"</code></td> <td>if iso2 code.</td> </tr> <tr> <td><code>"iso3"</code></td> <td>if iso3 code.</td> </tr> <tr> <td><code>"country_code"</code></td> <td>if Eurostat code</td> </tr> </table>	<code>"nuts"</code>	if nuts names).	<code>"id"</code>	if nuts id.	<code>"iso2"</code>	if iso2 code.	<code>"iso3"</code>	if iso3 code.	<code>"country_code"</code>	if Eurostat code
<code>"nuts"</code>	if nuts names).										
<code>"id"</code>	if nuts id.										
<code>"iso2"</code>	if iso2 code.										
<code>"iso3"</code>	if iso3 code.										
<code>"country_code"</code>	if Eurostat code										
<code>scale</code>	the scale of a map										
<code>dir</code>	local directory in which shape files are stored										
<code>show_eu</code>	logical value set to <code>TRUE</code> indicating if the map entire map is drawn or only the coordinates linked to the input data										
<code>add_text</code>	character name indicating the column with text labels										
<code>subset</code>	a formula indicating the condition to subset the data. See the details										
<code>facets</code>	variable(s) name to split the data										
<code>aggregation_fun</code>	function to use when data are aggregated										
<code>aggregation_unit</code>	variable name by which the unit are aggregated										
<code>options</code>	a list with options using <code>mapping.options</code> function										

Details

If `data` is a object of class "EU" generated using the [EU](#) function, the arguments `unit`, `year`, and `matchWith` are ignored, because the object already contains the coordinates.

The `aggregation_unit` provides an aggregation for a user specified variable in `data`, or for larger statistical unit, automatically provided when the function link the data with the coordinates. For example, if `data` are of type `nut2`, we will have variables for larger aggregate unit, that is `nuts1` and `nuts0` variables. Look at [EU](#) for more details.

`subset` provide an expression to subsetting the data using a formula, with the logical operators. For example `data` can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Returns a map. For tmap type, the function also returns a tmap object.

References

Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.

Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>

Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, 84(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingIT](#), [mappingUS](#), [mappingDE](#), [mappingUK](#)

Examples

```
data("popEU")
popEU <- popEU
euNuts2 <- EU(data = popEU, colID = "GEO", unit = "nuts2", matchWith = "id")

#####
# Statics #
#####

mappingEU(data = euNuts2, var = "total")
mappingEU(data = euNuts2, var = c("male", "female"))

mappingEU(data = euNuts2, var = "total", subset = ~I(nuts0_id == "IT"))
mappingEU(data = euNuts2, var = "total",
           subset = ~I(nuts0_id == "ES"), facets = "nuts2")

mappingEU(data = euNuts2, var = "total", typeStatic = "choro.cart")

mappingEU(data = euNuts2, var = "total", aggregation_unit = "nuts0", aggregation_fun = sum)
mappingEU(data = euNuts2, var = c("male", "female"),
           aggregation_unit = "nuts0", aggregation_fun = sum)

### Europe

eu1 <- loadCoordEU()
mappingEU(data = eu1)

#####
# Interactive #
```

```
#####

mappingEU(data = euNuts2, var = "total", type = "interactive")
mappingEU(data = euNuts2, var = c("male","female"), type = "interactive")

mappingEU(data = euNuts2, type = "interactive",
          var = "total", subset = ~I(nuts0_id == "IT"))
mappingEU(data = euNuts2, var = "total", type = "interactive",
          subset = ~I(nuts0_id == "ES"))

mappingEU(data = euNuts2, var = "total", type = "interactive")

mappingEU(data = euNuts2, var = "total", type = "interactive",
          aggregation_unit = "nuts0",
          aggregation_fun = sum)
mappingEU(data = euNuts2, var = c("male","female"), type = "interactive",
          aggregation_unit = "nuts0", aggregation_fun = sum)
```

mappingFR

Static maps for France

Description

Function to produce static maps for France statistical unit.

Usage

```
mappingFR(data, var = NULL, colID = NULL,
          type = c("static", "interactive"),
          typeStatic = c("tmap", "choro.cart", "typo", "bar"),
          unit = c("region"), year = c("2021", "2020", "2019"),
          matchWith = c("name", "code"),
          dir = NULL, add_text = NULL, subset = NULL, facets = NULL,
          aggregation_fun = sum, aggregation_unit = NULL,
          options = mapping.options())
```

Arguments

data	a data.frame object with variables to display or a UK object produced by FR function. If object of class FR, arguments unit, year, and matchWith will be ignored
var	character value(s) or columns number(s) indicating the variable to plot
colID	character value or columns number indicating the column with unit names
type	if generates static or interactive map

typeStatic	type of static map
unit	the type of Italian statistical unit
year	year of the unit
matchWith	the type of id to check: <div style="margin-left: 40px;"> "code" if unit code "name" if unit names). </div>
dir	local directory in which shape files are stored
add_text	character name indicating the column with text labels
subset	a formula indicating the condition to subset the data. See the details section
facets	variable(s) name to split the data
aggregation_fun	function to use when data are aggregated
aggregation_unit	variable name by which the unit are aggregated
options	a list with options using mapping.options function

Details

If data is a object of class "UK" generated using the [UK](#) function, the arguments unit, and year are ignored, because the object already contains the coordinates.

subset provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For tmap type, the function also returns a tmap object.

References

Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.

Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>

Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, 84(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingEU](#), [mappingUS](#), [mappingDE](#)

Examples

```

data("popFR")

fr <- FR(data = popFR)

#####
# Statics #
#####

mappingFR(data = fr, var = "population", subset = ~I(region == "corse"))

```

mappingIT

Static maps for Italy

Description

Function to produce static maps for Italian statistical unit.

Usage

```

mappingIT(data, var = NULL, colID = NULL,
           type = c("static", "interactive"),
           typeStatic = c("tmap", "choro.cart", "typo", "bar"),
           unit = c("none", "ripartizione", "regione", "provincia", "comune"),
           year = c("2021", "2020", "2019", "2018", "2017"),
           matchWith = c("name", "code", "number"), dir = NULL, show_it = TRUE,
           add_text = NULL, subset = NULL, facets = NULL,
           aggregation_fun = sum, aggregation_unit = NULL,
           options = mapping.options())

```

Arguments

data	a data.frame object with variables to display or a IT object produced by IT function. If object of class IT, arguments unit, year, and matchWith will be ignored
var	character value(s) or columns number(s) indicating the variable to plot
colID	character value or columns number indicating the column with unit names

type	if generates static or interactive map
typeStatic	type of static map
unit	the type of Italian statistical unit
year	year of the unit
matchWith	the type of id to check: <ul style="list-style-type: none"> "name" if unit names). "code" if unit code
dir	local directory in which shape files are stored
show_it	logical value set to TRUE indicating if the map entire map is drawn or only the coordinates linked to the input data
add_text	character name indicating the column with text labels
subset	a formula indicating the condition to subset the data. See the details section
facets	variable(s) name to split the data
aggregation_fun	function to use when data are aggregated
aggregation_unit	variable name by which the unit are aggregated
options	a list with options using mapping.options function

Details

If data is a object of class "IT" generated using the `IT` function, the arguments `unit`, and `year` are ignored, because the object already contains the coordinates.

The `aggregation_unit` provides an aggregation for a user specified variable in data, or for larger statistical unit, automatically provided when the function link the data with the coordinates. For example, if data are of type `provincia`, we will have variables for larger aggregate unit, that is `regione` and `ripartizione` variables. Look at `IT` for more details.

`subset` provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For `tmap` type, the function also returns a `tmap` object.

References

- Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.
- Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>
- Tennekes M (2018). "tmap: Thematic Maps in R." *Journalstatistical Software*, *84*(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingEU](#), [mappingUS](#), [mappingDE](#), [mappingUK](#)

Examples

```

data("popIT")

it <- IT(data = popIT, unit = "provincia", year = "2019", check.unit.names = FALSE)

#####
# Statics #
#####

mappingIT(data = it, var = "totale")

mappingIT(data = it, var = "totale", subset = ~I(regione == "Lazio"))

mappingIT(data = it, var = "totale", facets = "ripartizione")

mappingIT(data = it, var = c("maschi", "femmine"))
mappingIT(data = it, var = "totale", typeStatic = "choro.cart")

mappingIT(data = it, var = "totale",
          aggregation_unit = "ripartizione",
          aggregation_fun = function(x) sum(x, na.rm = TRUE))

### Italy

it1 <- loadCoordIT()
mappingIT(data = it1)

#####
# Interactive #
#####

mappingIT(data = it, var = "totale", type = "interactive")

mappingIT(data = it, var = c("maschi", "femmine"), type = "interactive")

mappingIT(data = it, var = "totale", subset = ~I(regione == "Lazio"), type = "interactive")

mappingIT(data = it, var = "totale", type = "interactive",
          aggregation_unit = "ripartizione",
          aggregation_fun = function(x) sum(x, na.rm = TRUE))

```

 mappingUK

Static maps for United Kingdom

Description

Function to produce static maps for United Kingdom statistical unit.

Usage

```
mappingUK(data, var = NULL, colID = NULL,
          type = c("static", "interactive"),
          typeStatic = c("tmap", "choro.cart", "typo", "bar"),
          unit = c("country", "county"), year = c("2020", "2019"),
          matchWith = c("name", "code"), scale = c("500", "20"),
          dir = NULL, add_text = NULL, subset = NULL,
          facets = NULL, aggregation_fun = sum, aggregation_unit = NULL,
          options = mapping.options())
```

Arguments

data	a data.frame object with variables to display or a UK object produced by UK function. If object of class UK, arguments unit, year, and matchWith will be ignored
var	character value(s) or columns number(s) indicating the variable to plot
colID	character value or columns number indicating the column with unit names
type	if generates static or interactive map
typeStatic	type of static map
unit	the type of Italian statistical unit
year	year of the unit
matchWith	the type of id to check: <ul style="list-style-type: none"> "name" if unit names). "code" if unit code
scale	the scale of a map
dir	local directory in which shape files are stored
add_text	character name indicating the column with text labels
subset	a formula indicating the condition to subset the data. See the details section
facets	variable(s) name to split the data
aggregation_fun	function to use when data are aggregated
aggregation_unit	variable name by which the unit are aggregated
options	a list with options using <code>mapping.options</code> function

Details

If data is a object of class "UK" generated using the [UK](#) function, the arguments `unit`, and `year` are ignored, because the object already contains the coordinates.

The `aggregation_unit` provides an aggregation for a user specified variable in data, or for larger statistical unit, automatically provided when the function link the data with the coordinates. For example, if data are of type `county`, we will have variables for larger aggregate unit, that is country variables. Look at [UK](#) for more details.

`subset` provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For `tmap` type, the function also returns a `tmap` object.

References

Giraud, T. and Lambert, N. (2016). *cartography: Create and Integrate Maps in your R Workflow*. JOSS, 1(4). doi: 10.21105/joss.00054.

Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal* 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>

Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, *84*(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingEU](#), [mappingUS](#), [mappingDE](#)

Examples

```
data("popUK")
```

```
uk <- UK(data = popUK, unit = "county", matchWith = "code", check.unit.names = FALSE)
```

mappingUS

Static maps for USA

Description

Function to produce static maps for USA unit.

Usage

```
mappingUS(data, var = NULL, colID = NULL,
          type = c("static", "interactive"),
          typeStatic = c("tmap", "choro.cart", "typo", "bar"),
          unit = c("country", "region", "division", "state",
                  "county", "district", "district_county", "urban_area"),
          year = c("2018"), matchWith = c("name", "id", "number"),
          scale = c("20", "50", "500"), dir = NULL, show_us = TRUE,
          add_text = NULL, subset = NULL, facets = NULL,
          aggregation_fun = sum, aggregation_unit = NULL,
          options = mapping.options())
```

Arguments

<code>data</code>	a data.frame object with variables to display or a US object produced by US function
<code>var</code>	character value(s) or columns number(s) indicating the variable to plot
<code>colID</code>	character value or columns number indicating the column with unit names
<code>type</code>	if generates static or interactive map
<code>typeStatic</code>	type of static map
<code>unit</code>	the type of European statistical unit to check.
<code>year</code>	year of the unit
<code>matchWith</code>	the type of id to check if unit is set to "states"
<code>scale</code>	the scale of a map
<code>dir</code>	local directory in which shape files are stored
<code>show_us</code>	logical value set to TRUE indicating if the map entire map is drawn or only the coordinates linked to the input data
<code>add_text</code>	character name indicating the column with text labels
<code>subset</code>	a formula indicating the condition to subset the data. See the details section
<code>facets</code>	variable(s) name to split the data
<code>aggregation_fun</code>	function to use when data are aggregated
<code>aggregation_unit</code>	variable name by which the unit are aggregated
<code>options</code>	a list with options using <code>mapping.options</code> function

Details

If `data` is a object of class "US" generated using the [US](#) function, the arguments `unit`, `year`, and `matchWith` are ignored, because the object already contains the coordinates.

The `aggregation_unit` provides an aggregation for a user specified variable in data, or for larger statistical unit, automatically provided when the function link the data with the coordinates. For example, if data are of type county, we will have variables for larger aggregate unit, that is state and region variables. Look at [US](#) for more details.

subset provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For tmap type, the function also returns a tmap object.

References

Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.

Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>

Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, *84*(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingWR](#), [mappingIT](#), [mappingEU](#), [mappingDE](#), [mappingUK](#)

Examples

```
data("popUS")

us <- US(data = popUS, unit = "state")

#####
# Statics #
#####

mappingUS(data = us, var = "population")
mappingUS(data = us, var = "population",
  subset = ~I(id == "california" | id == "texas"))
mappingUS(data = us, var = "population",
  subset = ~I(id == "california" | id == "texas"), facets = "id")

mappingUS(data = us, var = "population", typeStatic = "choro.cart")

#####
# Interactive #
#####

mappingUS(data = us, var = "population", type = "interactive")
mappingUS(data = us, var = "population", type = "interactive",
  subset = ~I(id == "california" | id == "texas" | id == "new york" ))
```

mappingWR

*Static maps for World countries***Description**

Function to produce static maps for world countries.

Usage

```
mappingWR(data, var = NULL, colID = NULL,
           type = c("static", "interactive"),
           typeStatic = c("tmap", "choro.cart",
                          "typo", "bar"),
           unit = c("country", "nato", "ocde", "continent",
                   "region", "subregion", "region_wb",
                   "type_income", "type_economy"),
           matchWith = c("country", "iso2", "iso3",
                        "iso3_eh", "iso3_numeric",
                        "iso3_un", "iso2_wb", "iso3_wb",
                        "name_formal", "name_wb"),
           res = c("low", "hi"), dir = NULL, show_wr = TRUE,
           add_text = NULL, subset = NULL,
           facets = NULL, aggregation_fun = sum, aggregation_unit = NULL,
           options = mapping.options(legend.position = c("left", "bottom")))
```

Arguments

data	a data.frame object with variables to display or a WR object produced by WR function												
var	character value(s) or columns number(s) indicating the variable to plot												
colID	character value or columns number indicating the column with unit names												
type	if generates static or interactive map												
typeStatic	type of static map												
unit	the type of world statistical unit												
matchWith	the type of id to check: <table> <tbody> <tr> <td>"country"</td> <td>if country names).</td> </tr> <tr> <td>"iso2"</td> <td>if iso2 code.</td> </tr> <tr> <td>"iso3"</td> <td>if iso3 code.</td> </tr> <tr> <td>"iso3_eh"</td> <td>if iso3_eh code.</td> </tr> <tr> <td>"iso3_numeric"</td> <td>if iso3 numeric code.</td> </tr> <tr> <td>"iso3_un"</td> <td>if iso3 United Nations.</td> </tr> </tbody> </table>	"country"	if country names).	"iso2"	if iso2 code.	"iso3"	if iso3 code.	"iso3_eh"	if iso3_eh code.	"iso3_numeric"	if iso3 numeric code.	"iso3_un"	if iso3 United Nations.
"country"	if country names).												
"iso2"	if iso2 code.												
"iso3"	if iso3 code.												
"iso3_eh"	if iso3_eh code.												
"iso3_numeric"	if iso3 numeric code.												
"iso3_un"	if iso3 United Nations.												

"iso2_wb"	if iso2 World Bank.
"iso3_wb"	if iso3 World Bank.
"name_formal"	if formal names.
"name_wb"	if World Bank names.
res	map resolution
dir	local directory in which shape files are stored
show_wr	logical value set to TRUE indicating if the map entire map is drawn or only the coordinates linked to the input data
add_text	character name indicating the column with text labels
subset	a formula indicating the condition to subset the data. See the details section
facets	variable(s) name to split the data
aggregation_fun	function to use when data are aggregated
aggregation_unit	variable name by which the unit are aggregated
options	a list with options using mapping.options function

Details

If data is a object of class "WR" generated using the `WR` function, the arguments `unit`, `year`, and `matchWith` are ignored, because the object already contains the coordinates.

The `aggregation_unit` provides an aggregation for a user specified variable in data, or for larger statistical unit, automatically provided when the function link the data with the coordinates.

`subset` provide an expression to subsetting the data using a formula, with the logical operators. For example data can be subsetting as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

Return a map. For `tmap` type, the function also returns a `tmap` object.

References

- Giraud, T. and Lambert, N. (2016). cartography: Create and Integrate Maps in your R Workflow. JOSS, 1(4). doi: 10.21105/joss.00054.
- Pebesma, E., 2018. Simple Features for R: Standardized Support for Spatial Vector Data. The R Journal 10 (1), 439-446, <https://doi.org/10.32614/RJ-2018-009>
- Tennekes M (2018). "tmap: Thematic Maps in R." *Journal of Statistical Software*, *84*(6), 1-39. doi: 10.18637/jss.v084.i06 (URL: <https://doi.org/10.18637/jss.v084.i06>).

See Also

[mappingEU](#), [mappingIT](#), [mappingUS](#), [mappingDE](#), [mappingUK](#)

Examples

```

data("popWR")
popWR <- popWR

wr <- WR(data = popWR, colID = "country_code",
         matchWith = "iso3_eh", check.unit.names = FALSE,
         res = "low")

#####
# Statics #
#####

mappingWR(data = wr, var = "total")

mappingWR(data = wr, var = c("male","female"))
mappingWR(data = wr, var = "total", subset = ~I(iso2 == "IT"))
mappingWR(data = wr, var = "total", subset = ~I(region == "Americas"))

mappingWR(data = wr, var = "total", facets = "continent")
mappingWR(data = wr, var = "total",
         subset = ~I(continent == "South America"),
         facets = "name_wb")

mappingWR(data = wr, var = "total", typeStatic = "choro.cart")

mappingWR(data = wr, var = "total", aggregation_unit = "continent",
         aggregation_fun = function(x) sum(x, na.rm = TRUE))
mappingWR(data = wr, var = "total", aggregation_unit = "subregion",
         aggregation_fun = function(x) sum(x, na.rm = TRUE))

## World countries

wr1 <- loadCoordWR()
mappingWR(data = wr1)

#####
# Interactive #
#####

mappingWR(data = wr, var = "total", type = "interactive")
mappingWR(data = wr, var = c("male","female"), type = "interactive")
mappingWR(data = wr, var = "total", subset = ~I(iso2 == "IT"), type = "interactive")

```

```
mappingWR(data = wr, var = "total",
           subset = ~I(region == "Americas"), type = "interactive")
mappingWR(data = wr, var = "total", type = "interactive",
           aggregation_unit = "continent",
           aggregation_fun = function(x) sum(x, na.rm = TRUE))
mappingWR(data = wr, var = "total", type = "interactive",
           aggregation_unit = "subregion",
           aggregation_fun = function(x) sum(x, na.rm = TRUE))
```

names

Statistical Unit Names

Description

Statistical unit names.

Usage

```
data("namesWR")
data("namesEU")
data("namesIT")
data("namesUS")
data("namesDE")
data("namesFR")
data("namesUK")
```

Format

A list with all names divided for year and type of units.

Details

Look at [getNamesWR](#), [getNamesEU](#), [getNamesIT](#), [getNamesUS](#), [getNamesUK](#), [getNamesDE](#), [getNamesFR](#)

Source

World Bank, Eurostat, United States Census and Istat

Examples

```
data(namesWR)
str(namesWR)

data(namesEU)
str(namesEU)
```

```
data(namesIT)
str(namesIT)
```

```
data(namesUS)
str(namesUS)
```

popDE *German Population*

Description

German bund population for year 2020

Usage

```
data("popDE")
```

Format

A data frame.

popEU *European population*

Description

European population for year 2018

Usage

```
data("popEU")
data("popEUnuts2")
```

Format

A data frame with 2252 observations on the following 5 variables.

TIME year

GEO names

total total

male number of male

female number of female

Source

<https://ec.europa.eu/eurostat/data/database>

popFR	<i>French Population</i>
-------	--------------------------

Description

French regions population for year 2021

Usage

```
data("popFR")
```

Format

A data frame.

popIT	<i>Italian Population</i>
-------	---------------------------

Description

Italian provincia population for year 2018

Usage

```
data("popIT")
```

Format

A data frame with 107 observations on the following 4 variables.

ID names

maschi number of male

femmine number of femal

totale total

Source

Istat

popUK	<i>United Kingdom Population</i>
-------	----------------------------------

Description

United Kingdom county population for year 2020

Usage

```
data("popUK")
```

Format

A data frame.

popUS	<i>USA population</i>
-------	-----------------------

Description

USA population for year 2019

Usage

```
data("popUS")
```

Format

A data frame with 52 observations on the following 2 variables.

id names

population total population

Source

<https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html>

popWR	<i>World population</i>
-------	-------------------------

Description

Country population for year 2018

Usage

```
data("popWR")
```

Format

A data frame with 269 observations on the following 5 variables.

country a factor with countries
country_code a factor with code
total total
male number of male
female number of female

Source

World Bank

saveObj	<i>Save mapping obj</i>
---------	-------------------------

Description

Save output from loadCoord function, sf objects, IT, EU, WR, and US in different format

Usage

```
saveObj(obj, name, as = c("RData", "csv", "json", "geojson", "shp"), ...)
```

Arguments

obj	Output from loadCoord function, sf objects, IT, EU, WR, and US
name	output name
as	format
...	further arguments

Value

No return value.

Examples

```
## Not run:

data("popIT")
it <- IT(data = popIT, unit = "provincia", year = "2019")
saveObj(it, name = "it.RData")

## End(Not run)
```

tax_wedge_ocde	<i>OCDE tax wedge</i>
----------------	-----------------------

Description

Tax wedge for OCDE countries

Usage

```
data("tax_wedge_ocde")
```

Format

A data frame with 74 observations on the following 7 variables.

country_code a factor with country code

year a character vector with year

family_type a factor with family levels

average_rate_employees_SSC a numeric vector with Social Securities Contribution by employees

average_rate_employer_SSC a numeric vector with Social Securities Contribution by employers

net_personal_average_tax_rate a numeric vector with personal average tax rate

average_tax_wedge a numeric vector with average tax wedge

Source

OECD (2020), Tax wedge (indicator). doi: 10.1787/cea9eba3-en (Accessed on 30 November 2020).
<https://data.oecd.org/tax/tax-wedge.htm>

Examples

```
data(tax_wedge_ocde)
str(tax_wedge_ocde)
```

UK *Object of class UK*

Description

Creates an object with data and coordinates of class UK for United Kindome statistical units to use with mapping functions or available in other R "maps" packages.

Usage

```
UK(data, colID = NULL, unit = c("country", "county"),
    year = c("2020", "2019"), matchWith = c("name", "code"),
    scale = c("500", "20"), subset = NULL, add = NULL,
    new_var_names = NULL, aggregation_fun = sum,
    aggregation_unit = NULL, aggregation_var = NULL,
    facets = NULL, check.unit.names = TRUE, dir = NULL,
    use_cache = TRUE, print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

data	a data.frame object with variables to display				
colID	character value or columns number indicating the column with unit names or codes				
unit	the type of Italian statistical unit				
year	year of the analysis				
matchWith	the type of id to check: <table style="margin-left: 2em;"> <tbody> <tr> <td>"name"</td> <td>if unit names</td> </tr> <tr> <td>"code"</td> <td>if unit code</td> </tr> </tbody> </table>	"name"	if unit names	"code"	if unit code
"name"	if unit names				
"code"	if unit code				
scale	the scale of the map				
subset	a formula indicating the condition to subset the data, see the Details				
add	a formula to add new transformed variables starting from the variables in the data				
new_var_names	a character value or vector indicating the names of the new variables created in add				
aggregation_fun	function to use when data are aggregated				
aggregation_unit	variable name by which the unit are aggregated				
aggregation_var	variable name with value to aggregate				
facets	variable(s) name to split the data				

<code>check.unit.names</code>	a logical value indicating if the colID names are checked with unit names
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package, the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class UK, with data and coordinates to use in functions which perform map.

See Also

[EU, WR, US, DE](#)

Examples

```
data("popUK")
uk <- UK(data = popUK, unit = "county", matchWith = "code")

### Adding two variables

uk2 <- UK(data = popUK, unit = "county", matchWith = "code",
          add = ~I(population/1000) + I(population/100) )

### Adding to variables and names
uk3 <- UK(data = popUK, unit = "county", matchWith = "code",
          add = ~I(population/1000) + I(population/100),
          new_var_names = c("ratio1", "ratio2"))
```

US

*Object of class US***Description**

Creates an object with data and coordinate of class US for United States of America to use with mapping functions or available in other R "maps" packages.

Usage

```
US(data, colID = NULL,
    unit = c("country", "region", "division", "state", "county",
            "district", "district_county", "urban_area"),
    year = c("2018"), matchWith = c("name", "id", "number"),
    scale = c("20", "50", "500"), show_us = TRUE,
    subset = NULL, add = NULL, new_var_names = NULL,
    aggregation_fun = sum, aggregation_unit = NULL, aggregation_var = NULL,
    facets = NULL, check.unit.names = TRUE, dir = NULL, use_cache = TRUE,
    print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

<code>data</code>	a data.frame object with variables to display
<code>colID</code>	character value or column numbers indicating the column with unit names.
<code>unit</code>	the type of US statistical unit
<code>year</code>	year of the analysis
<code>matchWith</code>	the type of id to check if unit is set to "states"
<code>scale</code>	the scale of the map
<code>show_us</code>	logical value set to TRUE indicating if the entire map is drawn or only the coordinates linked to the input data
<code>subset</code>	a formula indicating the condition to subset the data. See the details.
<code>add</code>	a formula to add new transformed variables starting from the variables in the data
<code>new_var_names</code>	a character value or vector indicating the names of the new variables created in add.
<code>aggregation_fun</code>	function to use when data are aggregated
<code>aggregation_unit</code>	variable name by which the unit are aggregated
<code>aggregation_var</code>	variable name with value to aggregate
<code>facets</code>	variable(s) name to split the data

<code>check.unit.names</code>	a logical value indicating if the <code>colID</code> names are checked with unit names.
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the `id` in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the `unit.names` (or codes) available in the package (as provided by USA Census of Bureau), the `check.unit.names` provides a preliminary check.

`subset` provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class `US`, with data and coordinates to use in functions which perform map.

See Also

[WR](#), [EU](#), [IT](#), [DE](#), [UK](#)

Examples

```
data("popUS")

us <- US(data = popUS, colID = "id", unit = "state")
str(us, 1)

us1 <- US(data = popUS, colID = "id", unit = "state", add = ~I(population/100))
str(us1, 1)

us2 <- US(data = popUS, colID = "id", unit = "state",
          add = ~I(population/100), new_var_names = c("population/100"))
str(us2, 1)
```

usa_election	<i>Usa Election</i>
--------------	---------------------

Description

2008 and 2016 Usa presidential election

Usage

```
data("usa_election")
```

Format

A data frame with 51 observations on the following 19 variables.

state_id a character vector
electoral_votes_obama a numeric vector
electoral_votes_mccain a numeric vector
votes_obama a numeric vector
votes_mccain a numeric vector
votes_others_08 a numeric vector
total_votes_08 a numeric vector
electoral_votes_trump a numeric vector
electoral_votes_clinton a numeric vector
votes_trump a numeric vector
votes_clinton a numeric vector
votes_others_16 a numeric vector
total_votes_16 a numeric vector
total_population_08 a numeric vector
total_citizen_08 a numeric vector
total_registered_08 a numeric vector
total_population_16 a numeric vector
total_citizen_16 a numeric vector
total_registered_16 a numeric vector

Source

<https://www.census.gov/topics/public-sector/voting/data.html> <https://www.fec.gov/introduction-campaign-finance/election-and-voting-information/>

Examples

```
data(usa_election)  
str(usa_election)
```

WR *Object of class WR*

Description

Creates an object with data and coordinate of class WR to use with mapping function or available in other R "maps" packages.

Usage

```
WR(data, colID = NULL,
    unit = c("country", "nato", "ocde", "continent",
            "region", "subregion", "region_wb",
            "type_income", "type_economy"),
    matchWith = c("country", "iso2", "iso3", "iso3_eh",
                 "iso3_numeric", "iso3_un", "iso2_wb",
                 "iso3_wb", "name_formal", "name_wb"),
    res = c("low", "hi"), show_wr = TRUE, subset = NULL,
    add = NULL, new_var_names = NULL,
    aggregation_fun = sum, aggregation_unit = NULL, aggregation_var = NULL,
    facets = NULL, check.unit.names = TRUE, dir = NULL, use_cache = TRUE,
    print = FALSE, use_internet = TRUE, crs = NULL)
```

Arguments

data	a data.frame object with variables to display																				
colID	character value or columns number indicating the column with unit names																				
unit	the type of world statistical unit																				
matchWith	the type of id to check: <table> <tbody> <tr> <td>"country"</td> <td>if country names</td> </tr> <tr> <td>"iso2"</td> <td>if iso2 code</td> </tr> <tr> <td>"iso3"</td> <td>if iso3 code</td> </tr> <tr> <td>"iso3_eh"</td> <td>if iso3_eh code</td> </tr> <tr> <td>"iso3_numeric"</td> <td>if iso3 numeric code</td> </tr> <tr> <td>"iso3_un"</td> <td>if iso3 United Nations</td> </tr> <tr> <td>"iso2_wb"</td> <td>if iso2 World Bank</td> </tr> <tr> <td>"iso3_wb"</td> <td>if iso3 World Bank</td> </tr> <tr> <td>"name_formal"</td> <td>if formal names</td> </tr> <tr> <td>"name_wb"</td> <td>if World Bank names</td> </tr> </tbody> </table>	"country"	if country names	"iso2"	if iso2 code	"iso3"	if iso3 code	"iso3_eh"	if iso3_eh code	"iso3_numeric"	if iso3 numeric code	"iso3_un"	if iso3 United Nations	"iso2_wb"	if iso2 World Bank	"iso3_wb"	if iso3 World Bank	"name_formal"	if formal names	"name_wb"	if World Bank names
"country"	if country names																				
"iso2"	if iso2 code																				
"iso3"	if iso3 code																				
"iso3_eh"	if iso3_eh code																				
"iso3_numeric"	if iso3 numeric code																				
"iso3_un"	if iso3 United Nations																				
"iso2_wb"	if iso2 World Bank																				
"iso3_wb"	if iso3 World Bank																				
"name_formal"	if formal names																				
"name_wb"	if World Bank names																				
res	map resolution																				
show_wr	logical value set to TRUE indicating if the entire map is drawn or only the coordinates linked to the input data																				
subset	a formula indicating the condition to subset the data, see the Details																				

<code>add</code>	a formula to add new transformed variables starting from the variables in the data
<code>new_var_names</code>	a character value or vector indicating the names of the new variables created in add
<code>aggregation_fun</code>	function to use when data are aggregated
<code>aggregation_unit</code>	variable name by which the unit are aggregated
<code>aggregation_var</code>	variable name with value to aggregate
<code>facets</code>	variable(s) name to split the data
<code>check.unit.names</code>	a logical value indicating if the colID names are checked with unit names
<code>dir</code>	local directory in which shape files are stored
<code>use_cache</code>	a logical value indicating whether use the cache
<code>print</code>	a logical value indicating whether print the nomatched names
<code>use_internet</code>	a logical value indicating wheter the coordinates are downloaded from https://github.com/mappinguniverse/geospatial . If FALSE the maps downloaded during package installation will be used
<code>crs</code>	coordinate reference system. Look at st_crs

Details

The function links (automatically) the id in the data and the coordinates for the given unit.

Since the names (or codes) provided in the data given in input must be checked with the unit names (or codes) available in the package, the `check.unit.names` provides a preliminary check.

subset provide an expression to subset the data, using a formula with the logical operators. For example, sub-samples of the data can be selected as follows: `~I("Variable 1" == "condition 1" & "Variable 2" != "condition 2")` or for example, `~I("Variable 1" > "condition 1" | "Variable 2" != "condition 2")`.

Value

An object of class WR, with data and coordinates to use in functions which perform map.

See Also

[EU](#), [IT](#), [US](#), [DE](#), [UK](#)

Examples

```
data("popWR")

wr <- WR(data = popWR, colID = "country_code",
         matchWith = "iso3_ah", res = "low")
str(wr, 1)
```

```
wr1 <- WR(data = popWR, colID = "country_code",
          matchWith = "iso3_eh", res = "low",
          add = ~I(male/total) + I(female/total))
str(wr1)

wr2 <- WR(data = popWR, colID = "country_code",
          matchWith = "iso3_eh", res = "low",
          add = ~I(male/total) + I(female/total),
          new_var_names = c("Per_Male", "Per_Female"))
str(wr2)
```

Index

* datasets

- names, 58
 - popDE, 59
 - popEU, 59
 - popFR, 60
 - popIT, 60
 - popUK, 61
 - popUS, 61
 - popWR, 62
 - tax_wedge_ocde, 63
 - usa_election, 68
- checkNamesDE, 3, 9
- checkNamesEU, 4, 5, 7, 9–11
- checkNamesIT, 6, 6, 10, 11
- checkNamesUK, 4, 8
- checkNamesUS, 4, 6, 7, 9, 9, 11
- checkNamesWR, 4, 6, 7, 9, 10, 10
- DE, 12, 15, 17, 25, 41, 42, 65, 67, 70
- EU, 13, 14, 17, 25, 44, 65, 67, 70
- FR, 16, 46
- getNamesDE, 18, 19–23, 58
- getNamesEU, 18, 19, 20–23, 58
- getNamesFR, 19, 58
- getNamesIT, 18–20, 20, 21–23, 58
- getNamesUK, 18–20, 21, 22, 23, 58
- getNamesUS, 19, 20, 22, 23, 58
- getNamesWR, 18–22, 23, 58
- IT, 15, 23, 48, 49, 67, 70
- loadCoordDE, 26, 28, 30, 32, 33, 35
- loadCoordEU, 27, 27, 30, 32, 33, 35
- loadCoordFR, 28
- loadCoordIT, 28, 29, 33, 35
- loadCoordUK, 27, 28, 30, 31, 33, 35
- loadCoordUS, 27, 30, 32, 32, 35
- loadCoordWR, 27, 28, 30, 32, 33, 34
- mapPalette, 35
- mapping, 36
- mapping.options, 38
- mappingDE, 41, 45, 47, 50, 52, 54, 56
- mappingEU, 37, 42, 43, 47, 50, 52, 54, 56
- mappingFR, 46
- mappingIT, 37, 45, 48, 54, 56
- mappingUK, 42, 45, 50, 51, 54, 56
- mappingUS, 42, 45, 47, 50, 52, 52, 56
- mappingWR, 37, 42, 45, 47, 50, 52, 54, 55
- names, 58
- namesDE (names), 58
- namesEU (names), 58
- namesFR (names), 58
- namesIT (names), 58
- namesUK (names), 58
- namesUS (names), 58
- namesWR (names), 58
- popDE, 59
- popEU, 59
- popEUnuts2 (popEU), 59
- popFR, 60
- popIT, 60
- popUK, 61
- popUS, 61
- popWR, 62
- saveObj, 62
- st_crs, 13, 15, 17, 25, 26, 28–31, 33, 34, 65, 67, 70
- tax_wedge_ocde, 63
- tm_borders, 38
- tm_fill, 38, 39
- tm_layout, 38, 40
- tm_style, 40

UK, [13](#), [15](#), [25](#), [47](#), [51](#), [52](#), [64](#), [67](#), [70](#)

US, [13](#), [15](#), [17](#), [25](#), [53](#), [65](#), [66](#), [70](#)

usa_election, [68](#)

WR, [13](#), [15](#), [17](#), [25](#), [55](#), [56](#), [65](#), [67](#), [69](#)