Package ‘bdl’

February 24, 2023

Type Package

Title Interface and Tools for ‘BDL’ API

Version 1.0.5

Description Interface to Local Data Bank (‘Bank Danych Lokalnych’ - ‘bdl’) API


License GPL-3

Depends R (>= 3.5.0)

Imports methods, magrittr, tibble, jsonlite, httr, dplyr, tidyr,

  utils, stats, ggplot2, ggrepur, randomcolorR, purrr, sf,
  tmplotools, tmap, progress

Encoding UTF-8

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

RoxygenNote 7.2.2

URL https://statisticspoland.github.io/R_Package_to_API_BDL/

BugReports https://github.com/statisticspoland/R_Package_to_API_BDL/issues

Author Marzena Szpadel [aut],
Krzysztof Kania [aut, cre],
Statistics Poland [cph, fnd]

Maintainer Krzysztof Kania <contact@krzysztofkania.com>

Date/Publication 2023-02-24 15:00:02 UTC

R topics documented:

  attribute_info .............................................................. 2
  bdl ............................................................... 3
attribute_info

Information about attribute

Description

Retrieve information about attribute.

Usage

attribute_info(attrId, lang = c("pl", "en"), ...)
Details

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`.

Value

A named list.

Examples

```r
# attribute_info("1")
```

---

**bdl**

*bdl: Interface and Tools for 'BDL' API*

---

**Description**


**Package options**

`'bdl.api_private_key'` String with BDL API key which you can get at https://api.stat.gov.pl/Home/BdlApi?lang=en Example: `options(bdl.api_private_key = "11111111-2222-3333-4444-555555555555")`

**Author(s)**

**Maintainer:** Krzysztof Kania <contact@krzysztofkania.com>

Authors:

- Marzena Szpadel <M.Szpadel@stat.gov.pl>

Other contributors:

- Statistics Poland [copyright holder, funder]

**See Also**

Useful links:

- [https://statisticspoland.github.io/R_Package_to_API_BDL/](https://statisticspoland.github.io/R_Package_to_API_BDL/)
generate_map

Generate quick map

Description

Generate given NUTS level map with data from given variable

Usage

generate_map(
  varId,
  year,
  unitLevel = 2,
  unitParentId = NULL,
  aggregateId = NULL,
  palette = "Blues",
  style = NULL,
  n = 10,
  names = FALSE,
  borderLevel = NULL,
  lang = c("pl", "en"),
  ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>varId</td>
<td>A single variable Id. Use search_variables or get_variables to find variable id code.</td>
</tr>
<tr>
<td>year</td>
<td>A single year from 2010-2023 range.</td>
</tr>
<tr>
<td>unitLevel</td>
<td>A map and data NUTS level - number from 1 to 6. Use get_levels to find more info.</td>
</tr>
<tr>
<td>unitParentId</td>
<td>A 12 character NUTS id code of interested unit. Use search_units or get_units to find unit id code.</td>
</tr>
<tr>
<td>aggregateId</td>
<td>An aggregate id. Use get_aggregates for more info.</td>
</tr>
<tr>
<td>palette</td>
<td>A palette name or a vector of colors. See tmaptools::palette_explorer() for the named palettes. Use a &quot;-&quot; as prefix to reverse the palette.</td>
</tr>
<tr>
<td>style</td>
<td>Method to process the color scale. Options available are &quot;sd&quot;, &quot;equal&quot;, &quot;pretty&quot;, &quot;quantile&quot;, &quot;kmeans&quot;, &quot;hclust&quot;, &quot;bclust&quot;, &quot;fisher&quot;, &quot;jenks&quot;, and &quot;log10_pretty&quot;.</td>
</tr>
<tr>
<td>n</td>
<td>Preferred number of classes. Default is 10.</td>
</tr>
<tr>
<td>names</td>
<td>Logical that determines whether the unit names are shown.</td>
</tr>
<tr>
<td>borderLevel</td>
<td>Adds contours of units on specified level - number from 1 to 6. Use get_levels to find more info.</td>
</tr>
<tr>
<td>lang</td>
<td>A language of returned data, &quot;pl&quot; (default), &quot;en&quot;</td>
</tr>
<tr>
<td>...</td>
<td>Other arguments passed on to GET. For example a proxy parameters, see details.</td>
</tr>
</tbody>
</table>
**get_aggregates**

**Details**

Generate quickly map for given NUTS level, using BDL data. Default level is 2.
Maps available for year: 2010-2020
Provide unit parent id to narrow the map for specific regions.
Generating lower (levels 5 and 6) level maps can take some time.
This function requires external map data "bdl.maps" loaded to global environment. You can get data here: Map download. Download data and double-click to load it to environment.
To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

**Value**

A tmap map.

**Examples**

```r
# generate_map(varId = "60559", year = "2017")
```

---

**Description**

Retrieve all aggregates with information.

**Usage**

```r
get_aggregates(
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...)
```

**Arguments**

- `sort`: A type of sorting, "id" (default), "-id", "name", "-name"
- `lang`: A language of returned data, "pl" (default), "en"
- `...`: Other arguments passed on to `GET`. For example a proxy parameters, see details.

**Details**

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).
get_attributes

Value
A dataset as a tibble.

Examples

```r
# get_aggregates()
```

---

get_attributes  Get all attributes

Description
Retrieve all attributes with information.

Usage

```r
get_attributes(
  sort = c("id", "-id", "Display", "-Display"),
  lang = c("pl", "en"),
  ...
)
```

Arguments

- **sort**: A type of sorting, "id" (default), "-id", "Display", "-Display"
- **lang**: A language of returned data, "pl" (default), "en"
- **...**: Other arguments passed on to `GET`. For example a proxy parameters, see details.

Details
To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`.

Value
A dataset as a tibble.

Examples

```r
# get_attributes()
```
get_data_by_unit

Get data by unit Id’s from BDL API

Description
Retrieve data for given units from BDL with specified format.

Usage
get_data_by_unit(
  unitId,
  varId,
  year = NULL,
  type = c("code", "label"),
  aggregateId = NULL,
  lang = c("pl", "en"),
  ...
)

Arguments
unitId A single 12 character NUTS id code or vector of multiple unit id codes. If multiple unit codes are used, some columns are not available. Use search_units or get_units to find unit id code.
varId A vector of variable Id's. Use search_variables or get_variables to find variable id code.
year A vector of years. If NULL (default) returns data for all available years.
type A type of variables returned, "code" (default), "label"
aggregateId An aggregate id. Use get_aggregates for more info.
lang A language of returned data, "pl" (default), "en"
... Other arguments passed on to GET. For example a proxy parameters, see details.

Details
Data to retrieve from The BDL Web Services can be filtered with arguments. To get JSON data from specified directory with custom filters use get_request directly.

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value
A dataset as a tibble.
get_data_by_unit_locality

Examples

# get_data_by_unit(unitId = "023200000000", varId = "3643")
# get_data_by_unit(unitId = "023200000000", varId = c("3643", "2137", "148190"),
#                  type = "label")

# Multi variable download
# get_data_by_unit(unitId = c("023200000000", "020800000000"),
#                  varId = c("3643", "2137", "148190"))

description

Get data by unit locality Id from BDL API

Usage

get_data_by_unit_locality(
  unitId,
  varId,
  year = NULL,
  type = c("code", "label"),
  lang = c("pl", "en"),
  ...
)

Arguments

unitId A 12 character NUTS unit locality id with 7 characters locality individual id, separated by dash or vector of multiple unit id codes. If multiple unit codes are used, some columns are not available. Use search_unit_localities or get_unit_localities to find unit id code.

varId A vector of variable Id’s. Use search_variables or get_variables to find variable id code.

year A vector of years. If NULL (default) returns data for all available years.

type A type of variables returned, "code" (default), "label"

lang A language of returned data, "pl" (default), "en"

... Other arguments passed on to GET. For example a proxy parameters, see details.

Details

Data to retrieve from The BDL Web Services can be filtered with arguments. To get JSON data from specified directory with custom filters use get_request directly.

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).
**get_data_by_variable**

**Value**

A dataset as a tibble.

**Examples**

```r
# get_data_by_unit_locality(unitId = "030210106062-0189782", varId = "415", type = "label")

# Multi variable download
# get_data_by_unit_locality(unitId = c("030210106062-0189782", "030210106062-0189753"),
#   varId = "415")
```

---

**Description**

Retrieve data for a given variable for multiple units from BDL with specified format.

**Usage**

```r
get_data_by_variable(
  varId, 
  unitParentId = NULL, 
  unitLevel = NULL, 
  year = NULL, 
  aggregateId = NULL, 
  lang = c("pl", "en"),
  ...
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>varId</code></td>
<td>A single variable Id or vector of multiple variable id’s. If multiple id’s are used, some columns are not available. Use <code>search_variables</code> or <code>get_variables</code> to find variable id code.</td>
</tr>
<tr>
<td><code>unitParentId</code></td>
<td>A 12 character NUTS id code of parent unit. Use <code>search_units</code> or <code>get_units</code> to find unit id code. If NULL (default) and <code>unitLevel</code> not set up, returns all available units for variable.</td>
</tr>
<tr>
<td><code>unitLevel</code></td>
<td>A number from 0 to 6, filters the returned unit by its level. If NULL (default) no level filters apply. Use <code>get_levels</code> to find more info.</td>
</tr>
<tr>
<td><code>year</code></td>
<td>A vector of years. If NULL (default) returns data for all available years.</td>
</tr>
<tr>
<td><code>aggregateId</code></td>
<td>An aggregate id. Use <code>get_aggregates</code> for more info.</td>
</tr>
<tr>
<td><code>lang</code></td>
<td>A language of returned data, &quot;pl&quot; (default), &quot;en&quot;</td>
</tr>
<tr>
<td><code>...</code></td>
<td>Other arguments passed on to <code>GET</code>. For example a proxy parameters, see details.</td>
</tr>
</tbody>
</table>
get_data_by_variable_locality

Details

Data to retrieve from The BDL Web Services can be filtered with arguments. To get JSON data from specified directory with custom filters use `get_request` directly.

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`.

Value

A dataset as a tibble.

Examples

```r
# get_data_by_variable(varId = "3643", unitParentId = "030200000000")
# get_data_by_variable("420", year = "2000", unitLevel = 6)

# Multi variable download
# get_data_by_variable(varId =c("415","420"), unitParentId = "030210423000")
```

get_data_by_variable_locality

*Get data by variable Id for localities from BDL API*

Description

Retrieve data for a given variables for multiple unit localities from BDL with specified format.

Usage

```r
get_data_by_variable_locality(
  varId,  
  unitParentId,  
  year = NULL,  
  lang = c("pl", "en"),  
  ...  
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>varId</td>
<td>A single variable id or vector of multiple variable id's. If multiple id’s are used, some columns are not available. Use <code>search_variables</code> or <code>get_variables</code> to find variable id code.</td>
</tr>
<tr>
<td>unitParentId</td>
<td>A 12 character NUTS id code of interested unit. Use <code>search_units</code> or <code>get_units</code> to find unit id code. If NULL (default) and unitLevel not set up, returns all available units for variable.</td>
</tr>
<tr>
<td>year</td>
<td>A vector of years. If NULL (default) returns data for all available years.</td>
</tr>
<tr>
<td>lang</td>
<td>A language of returned data, &quot;pl&quot; (default), &quot;en&quot;</td>
</tr>
<tr>
<td>...</td>
<td>Other arguments passed on to <code>GET</code>. For example a proxy parameters, see details.</td>
</tr>
</tbody>
</table>
get_levels

Details

Data to retrieve from The BDL Web Services can be filtered with arguments. To get JSON data from specified directory with custom filters use get_request directly.

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value

A dataset as a tibble.

Examples

# get_data_by_variable_locality(varId = "415", unitParentId = "011212006063")
# get_data_by_variable_locality("420", year = "2008", unitParentId = "070000000000")

# Multi variable download
# get_data_by_variable_locality(varId =c("415","430"), unitParentId = "011212006063")

get_levels

Get all levels

Description

Retrieve all levels with information.

Usage

get_levels(sort = c("id", "-id", "name", "-name"), lang = c("pl", "en"), ...)

Arguments

sort A type of sorting, "id" (default), "-id", "name", "-name"
lang A language of returned data, "pl" (default), "en"
... Other arguments passed on to GET. For example a proxy parameters, see details.

Details

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value

A dataset as a tibble.

Examples

# get_levels()
get_panel_data

Get panel data by unit and variable Id’s from BDL API

Description
Retrieve data for given units from BDL with specified format.

Usage
get_panel_data(unitId, varId, year = NULL, ggplot = FALSE, ...)

Arguments
unitId A single 12 character NUTS id code or vector of multiple unit id codes. If multiple unit codes are used, some columns are not available. Use search_units or get_units to find unit id code.
varId A single Id or vector of variable Id’s. Use search_variables or get_variables to find variable id code.
year A vector of years. If NULL (default) returns data for all available years.
ggplot Output in a long format suitable for ggplot2. Allows to plot results directly with ggplot function.
... Other arguments passed on to GET. For example a proxy parameters, see details.

Details
Data to retrieve from The BDL Web Services can be filtered with arguments. To get JSON data from specified directory with custom filters use get_request directly.
To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).

Value
A dataset as a tibble.

Examples
# get_panel_data(unitId = "030210101000", varId = "60270")
# get_panel_data(unitId = "030210101000", varId = c("60270", "461668"))
# get_panel_data(unitId = c("030210101000", "030210105000", "030210106000"),
#                varId = c("60270", "461668"), year = c(2013:2016))
# get_panel_data(unitId = c("030210101000", "030210105000", "030210106000"),
#                varId = c("60270", "461668"), ggplot = TRUE)
get_request

Get JSON response from BDL API

Description

Retrieve data from BDL API in JSON format.

Usage

get_request(dir, id, filters = NULL, ...)

Arguments

dir          A directory of the dataset.
id          A name for the dataset of interested.
filters     A named list of filters. Names of list objects are bdl filter names and values are vectors with specified filter values. If NULL (default) the whole dataset is returned. See API documentation for more on filters and limitations per query.

...          Other arguments passed on to GET. For example a proxy parameters, see details.

Details

Data to retrieve from The BDL Web Services can be specified with filters. If no specific filters required, it's recommended to use data query like get_data_by_unit_locality, than to use get_request directly.

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value

A JSON raw data.

Examples

# get_request(dir = "data/By-Variable", id = "3643")
# get_request(dir = "data/By-Unit", id = "023200000000", filters = list(year = c("2000","2010"),
#               var-Id" = c("2137","148190")))
# get_request(dir = "data/By-Variable", id = "3643", filters = list(year = c("2000","2010"),
#               unit-Level" = 2, lang = "en"))
get_subjects

Get subject id codes.

Description

Retrieve all subjects id’s or sub-subjects.

Usage

```r
get_subjects(
  parentId = "",
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)
```

Arguments

- **parentId**: A parent subject id code. If not specified returns all top level subjects. Use `search_subjects` to find subject codes.
- **sort**: A type of sorting, "id" (default), "-id", "name", "-name"
- **lang**: A language of returned data, "pl" (default), "en"
- **...**: Other arguments passed on to `GET`. For example a proxy parameters, see details.

Details

To get all top level subjects skip the `parentId` parameter or list sub-subjects for given parent subject.

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`.

Value

A dataset as a tibble.

Examples

```r
# get_subjects()
# get_subjects("K3")
# get_subjects("G7")
```
**get_units**

Get unit NUTS codes.

---

**Description**

Retrieve all unit codes or sub to given unit.

**Usage**

```r
get_units(
  parentId = "",
  level = NULL,
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)
```

**Arguments**

- **parentId** A 12 character NUTS id code of parent unit. Use `search_units` to find unit id code.
- **level** A number from 0 to 6, filters the returned unit by its level. If `NULL` (default) no level filters apply. Use `get_levels` to find more info.
- **sort** A type of sorting, "id" (default), "-id", "name", "-name"
- **lang** A language of returned data, "pl" (default), "en"
- **...** Other arguments passed on to `GET`. For example a proxy parameters, see details.

**Details**

To get all units skip the `parentId` parameter. *Warning!* Downloading all unit can take around 1 minute.

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = http::use_proxy(url, port, username, password))`.

**Value**

A dataset as a tibble.

**Examples**

```r
# get_units(level = 2)
# get_units("010000000000")
```
get_unit_localities  Get unit locality codes.

Description
Retrieve unit locality codes.

Usage
get_unit_localities(
  parentId,
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)

Arguments
parentId  A 12 character NUTS id code of parent unit. Use search_units to find unit id code.
sort  A type of sorting, "id" (default), "-id", "name", "-name"
lang  A language of returned data, "pl" (default), "en"
...  Other arguments passed on to GET. For example a proxy parameters, see details.

Details
To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id,
  filters, config = http::use_proxy(url, port, username, password)).

Value
A dataset as a tibble.

Examples
# get_unit_localities("030210106062")
get_variables

**Get variable id codes.**

Description

Retrieve variables for given subjectId.

Usage

```r
get_variables(
  subjectId, 
  level = NULL, 
  year = NULL, 
  sort = c("id", "-id", "subjectId", "-subjectId"), 
  lang = c("pl", "en"), 
  ... 
)
```

Arguments

- `subjectId` A subject id code. If not specified returns all top level subjects. Use `search_subjects` or `get_subjects` to get subject id.
- `level` A number from 0 to 6, filters the returned unit by its level. If NULL (default) no level filters apply. Use `get_levels` to find more info.
- `year` A vector of years. If NULL (default) returns data for all available years.
- `sort` A type of sorting, "id" (default), "-id", "subjectId", "-subjectId"
- `lang` A language of returned data, "pl" (default), "en"
- `...` Other arguments passed on to `GET`. For example a proxy parameters, see details.

Details

Variables for specified subject optionally filtered by level and year.

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = http::use_proxy(url, port, username, password))`.

Value

A dataset as a tibble.

Examples

```r
# get_variables("P2425")
```
line_plot

Generate quick line plot

Description
Generate line plot for one unit/multiple variables or variable/multiple units

Usage
line_plot(
  data_type = c("unit", "unit.locality", "variable", "variable.locality"),
  unitId = NULL,
  varId = NULL,
  year = NULL,
  aggregateId = NULL,
  lang = NULL,
  unitParentId = NULL,
  unitLevel = NULL,
  ...
)

Arguments

- **data_type**
  A type of data used for generating plot, "unit" (default), "unit.locality", "variable", "variable.locality"

- **unitId**
  A 12 character NUTS unit id or locality 12 character id with 7 characters locality individual id, separated by dash.

- **varId**
  A vector of variable Id's (data_type equal "unit" or "unit.locality") or single variable (data_type equal "variable" or "variable.locality"). Use search_variables or get_variables to find variable id code.

- **year**
  A vector of years. If NULL (default) returns data for all available years.

- **aggregateId**
  An aggregate id. Use get_aggregates for more info.

- **lang**
  A language of returned data, "pl" (default), "en"

- **unitParentId**
  A 12 character NUTS id code of interested unit. (Used only with data_type equal "variable" or "variable.locality") Use search_units or get_units to find unit id code.

- **unitLevel**
  A number from 0 to 6, filters the returned unit by its level. (Used only with data_type equal "variable") If NULL (default) no level filters apply. Use get_levels to find more info.

  ...

  Other arguments passed on to GET. For example a proxy parameters, see details.

Details
Generate quickly ‘ggplot2’ plot, using BDL data.
Plot multiple variable values for one unit or one variable value for multiple units.

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).
Value
A ggplot2 plot.

Examples

```r
# line_plot(data_type = "unit", unitId = "000000000000", varId = c("415","420"))
```

pie_plot

Generate quick pie plot

Description
Generate pie plot for variable/multiple units

Usage

```r
pie_plot(
  data_type = c("variable", "variable.locality"),
  varId,
  year,
  unitParentId = NULL,
  unitLevel = NULL,
  aggregateId = NULL,
  label = T,
  lang = c("pl", "en"),
  ...
)
```

Arguments

data_type A type of data used for generating plot, "variable" (default), "variable", "variable.locality"

varId A variable Id. Use `search_variables` or `get_variables` to find variable id code.

year A single year. If NULL (default) returns data for all available years.

unitParentId A 12 character NUTS id code of interested unit. Use `search_units` or `get_units` to find unit id code.

unitLevel A number from 0 to 6, filters the returned unit by its level. (Used only with data_type equal "variable") If NULL (default) no level filters apply. Use `get_levels` to find more info.

aggregateId An aggregate id. Use `get_aggregates` for more info.

label Logical; if TRUE (default) adds labels.

lang A language of returned data, "pl" (default), "en"

... Other arguments passed on to `GET`. For example a proxy parameters, see details.
Details
Generate quickly ‘ggplot2’ plot, using BDL data.
Pie plot one variable value for multiple units on single year.
To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value
A ggplot2 plot.

Examples
# pie_plot(data_type ="variable","1","2018",unitParentId="04221430000", unitLevel = "6")

scatter_2var_plot
Generate quick scatter correlation plot

Description
Generate scatter correlation plot for 2 variables

Usage
scatter_2var_plot(
data_type = c("variable", "variable.locality"),
varId,
year = NULL,
unitParentId = NULL,
unitLevel = NULL,
aggregateId = NULL,
lang = c("pl", "en"),
...
)

Arguments
data_type A type of data used for generating plot, "variable"(default), "variable.locality"
varId A vector of 2 variable Id’s. Use search_variables or get_variables to find variable id code.
year A vector of years. If NULL (default) returns data for all available years.
unitParentId A 12 character NUTS id code of interested unit. (Used only with data_type equal "variable" or "variable.locality") Use search_units or get_units to find unit id code.
unitLevel A number from 0 to 6, filters the returned unit by its level. (Used only with data_type equal "variable") If NULL (default) no level filters apply. Use get_levels to find more info.
aggregateId  An aggregate id. Use `get_aggregates` for more info.
lang  A language of returned data, "pl" (default), "en"
...  Other arguments passed on to GET. For example a proxy parameters, see details.

Details

Generate quickly 'ggplot2' scatter correlation plot, using BDL data.
Scatter plot 2 variables for given units with regression line, confidence interval and correlation coefficient.
To use a proxy to connect, a `use_proxy` can be passed to GET. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`.

Value

A ggplot2 plot.

Examples

```r
# scatter_2var_plot(data_type = "variable", c("415", "60559"), unitLevel = "2")
```

---

**search_subjects**

**Search for subject codes**

**Description**

Search for given phrase in subject names

**Usage**

```r
search_subjects(
  name,
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)
```

**Arguments**

- **name**  A phrase to search.
- **sort**  A type of sorting, "id" (default), "-id", "name", "-name"
- **lang**  A language of returned data, "pl" (default), "en"
- **...**  Other arguments passed on to GET. For example a proxy parameters, see details.

**Details**

To use a proxy to connect, a `use_proxy` can be passed to GET. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`. 
# search_units

**Value**

A dataset as a tibble.

**Examples**

```r
# search_subjects("samochody")
# search_subjects("car", lang = "en")
```

**Description**

Search for a given phrase in unit names.

**Usage**

```r
search_units(
  name,
  level = NULL,
  year = NULL,
  kind = NULL,
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)
```

**Arguments**

- **name**: A phrase to search.
- **level**: A number from 0 to 6, filters the returned unit by its level. If NULL (default) no level filters apply. Use `get_levels` to find more info.
- **year**: A vector of years. If NULL (default) returns data for all available years.
- **sort**: A type of sorting, "id" (default), ",id", "name", ",-name"
- **lang**: A language of returned data, "pl" (default), "en"
- **...**: Other arguments passed on to `GET`. For example a proxy parameters, see details.

**Details**

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = httr::use_proxy(url, port, username, password))`. 
search_unit_localities

Value
A dataset as a tibble.

Examples
# search_units("wro")
# search_units("pol", type = "5")

search_unit_localities
Search for unit localities

Description
Search for a given phrase in unit locality names.

Usage
search_unit_localities(
  name,
  year = NULL,
  sort = c("id", "-id", "name", "-name"),
  lang = c("pl", "en"),
  ...
)

Arguments
name A phrase to search.
year A vector of years. If NULL (default) returns data for all available years.
sort A type of sorting, "id" (default), "-id", "name", "-name"
lang A language of returned data, "pl" (default), "en"
... Other arguments passed on to GET. For example a proxy parameters, see details.

Details
To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).

Value
A dataset as a tibble.

Examples
# search_unit_localities("wro")
search_variables  Search for variable codes

Description

Search for given phrase in variable names

Usage

```r
search_variables(
  name,
  subjectId = NULL,
  level = NULL,
  year = NULL,
  sort = c("id", "-id", "subjectId", "-subjectId"),
  lang = c("pl", "en"),
  ...
)
```

Arguments

- **name**: A phrase to search.
- **subjectId**: A subject id code. If not specified returns all top level subjects. Use `search_subjects` or `get_subjects` to get subject id.
- **level**: A number from 0 to 6, filters the returned unit by its level. If NULL (default) no level filters apply. Use `get_levels` to find more info.
- **year**: A vector of years. If NULL (default) returns data for all available years.
- **sort**: A type of sorting, "id" (default), "-id", "name", "-name"
- **lang**: A language of returned data, "pl" (default), "en"
- **...**: Other arguments passed on to `GET`. For example a proxy parameters, see details.

Details

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = http::use_proxy(url, port, username, password))`.

Value

A dataset as a tibble.

Examples

```r
# search_variables("samochody")
# search_variables("cars", lang = "en")
```
subject_info

Information about subject

Description

Retrieve information about subject

Usage

subject_info(subjectId, lang = c("pl", "en"), ...)

Arguments

subjectId A subject id code. If not specified returns all top level subjects. Use search_subjects or get_subjects to find subject codes.

lang A language of returned data, "pl" (default), "en"

... Other arguments passed on to GET. For example a proxy parameters, see details.

Details

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).

Value

A named list.

Examples

# subject_info("G7")

summary.bdl

Summarize bdl data frame

Description

Prints brief summary with basic statistical functions like mean, standard deviation, variance, min and max for bdl data frame.

Usage

## S3 method for class 'bdl'  
summary(object, ...)
Arguments

object bdl data frame to summarise

... other arguments ignored (for compatibility with generic)

Examples

# df <- get_data_by_variable(varId = "3643")
# summary(df)

unit_info Information about unit

Description

Retrieve information about unit

Usage

unit_info(unitId, lang = c("pl", "en"), ...)

Arguments

unitId A 12 character NUTS id code of interested unit. Use search_units or get_units to find unit id code.
lang A language of returned data, "pl" (default), "en"

... Other arguments passed on to GET. For example a proxy parameters, see details.

Details

To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = http::use_proxy(url, port, username, password)).

Value

A named list.

Examples

# unit_info("030210106062")
unit_locality_info Information about unit locality

Description
Retrieve information about unit locality

Usage
unit_locality_info(unitId, lang = c("pl", "en"), ...)

Arguments
unitId A 12 character NUTS id with 7 characters locality individual id, separated by dash. Use search_unit_localities or get_unit_localities to find unit id code.
lang A language of returned data, "pl" (default), "en"
... Other arguments passed on to GET. For example a proxy parameters, see details.

Details
To use a proxy to connect, a use_proxy can be passed to GET. For example get_request(id, filters, config = httr::use_proxy(url, port, username, password)).

Value
A named list.

Examples

# unit_locality_info("030210106062-0189782")

variable_info Information about variable

Description
Retrieve information about variable.

Usage
variable_info(varId, lang = c("pl", "en"), ...)
variable_info

Arguments

- **varId**: A vector of variable Id's. Use `search_variables` or `get_variables` to find variable id code.
- **lang**: A language of returned data, "pl" (default), "en"
- **...**: Other arguments passed on to `GET`. For example a proxy parameters, see details.

Details

To use a proxy to connect, a `use_proxy` can be passed to `GET`. For example `get_request(id, filters, config = http::use_proxy(url, port, username, password))`.

Value

A named list.

Examples

```r
# variable_info("420")
```
Index

* database
  get_data_by_unit, 7
  get_data_by_unit_locality, 8
  get_data_by_variable, 9
  get_data_by_variable_locality, 10
  get_panel_data, 12
  get_request, 13

* info
  attribute_info, 2
  subject_info, 25
  unit_info, 26
  unit_locality_info, 27
  variable_info, 27

* levels
  get_aggregates, 5
  get_attributes, 6
  get_levels, 11

* localities
  get_unit_localities, 16
  search_unit_localities, 23
  unit_locality_info, 27

* search
  search_subjects, 21

* search
  get_subjects, 14
  get_unit_localities, 16
  get_units, 15
  get_variables, 17
  search_unit_localities, 23
  search_units, 22
  search_variables, 24

* subjects
  get_subjects, 14
  search_subjects, 21
  search_variables, 24
  subject_info, 25

* units
  get_unit_localities, 16
  get_units, 15

search_unit_localities, 23
search_units, 22
unit_info, 26
unit_locality_info, 27

* utilities
  get_aggregates, 5
  get_attributes, 6
  get_data_by_unit, 7
  get_data_by_unit_locality, 8
  get_data_by_variable, 9
  get_data_by_variable_locality, 10
  get_levels, 11
  get_panel_data, 12
  get_request, 13
  get_subjects, 14
  get_units, 15
  get_variables, 17
  search_subjects, 21
  search_unit_localities, 23
  search_units, 22
  search_variables, 24

* variables
  attribute_info, 2
  get_variables, 17
  variable_info, 27

attribute_info, 2

bdl, 3
bdl-package (bdl), 3

generate_map, 4
GET, 2–28
get_aggregates, 4, 5, 7, 9, 18, 19, 21
get_attributes, 2, 6
get_data_by_unit, 7
get_data_by_unit_locality, 8, 13
get_data_by_variable, 9
get_data_by_variable_locality, 10
get_levels, 4, 9, 11, 15, 17–20, 22, 24
get_panel_data, 12
get_request, 7, 8, 10–13, 13
get_subjects, 14, 17, 24, 25
get_unit_localities, 8, 16, 27
get_units, 4, 7, 9, 10, 12, 15, 18–20, 26
get_variables, 4, 7–10, 12, 17, 18–20, 28

line_plot, 18

pie_plot, 19

scatter_2var_plot, 20
search_subjects, 14, 17, 21, 24, 25
search_unit_localities, 8, 23, 27
search_units, 4, 7, 9, 10, 12, 15, 16, 18–20, 22, 26
search_variables, 4, 7–10, 12, 18–20, 24, 28
subject_info, 25
summary.bdl, 25

unit_info, 26
unit_locality_info, 27
use_proxy, 3, 5–8, 10–18, 20–28

variable_info, 27