Package 'fedstatAPIr'

July 19, 2021

Title Unofficial API for Fedstat (EMISS) for Automatic and Efficient Data Queries

Version 0.1

Description An API for automatic data queries to the fedstat <https://www.fedstat.ru>, using a small set of functions with a common interface.

License MIT + file LICENSE

URL https://github.com/DenchPokepon/fedstatAPIr

BugReports https://github.com/DenchPokepon/fedstatAPIr/issues

Encoding UTF-8

RoxygenNote 7.1.1

Imports httr, rvest, jsonlite, stringr, dplyr, xml2, readsdmx, rsdmx, magrittr, utils, rlang, methods

NeedsCompilation no

Author Denis Krylov [aut, cre]

Maintainer Denis Krylov <deniskrylovvit@gmail.com>

Repository CRAN

Date/Publication 2021-07-19 07:30:06 UTC

R topics documented:

`fedstatAPIr-package` *Unofficial API for fedstat (EMISS) for automatic and efficient data queries*

Description

This package makes it fairly easy to automatically download filtered data from fedstat.ru, using a small set of functions with a common interface.

Author(s)

Maintainer: Denis Krylov <deniskrylovvit@gmail.com>

See Also

Useful links:

- https://github.com/DenchPokepon/fedstatAPIr
- Report bugs at https://github.com/DenchPokepon/fedstatAPIr/issues

---

`fedstat_data_ids_filter`
Filters `data_ids` *based on* `filters` *that are given in JSON format*

Description

Filters indicator `data_ids` with given `filters` taking into account possible `filters` specification errors and default filters.

`filters` should use `filter_field_title` in names and `filter_value_title` in values as they are presented on fedstat.ru. If for some reason the specified filters do not return the expected result, it is worth inspecting possible filter values in `data_ids` to see if the strings are defined correctly (e.g. encoding issues, mixing latin and cyrillic symbols)

`filter_value_title` currently supports the following special values:

1. asterix (*), it's alias for "select all possible filter values for this filter field"

Unspecified filters use asterix as a default (i.e. all possible filter values are selected and a warning is given)

Internally normalized `filter_field_title` and `filter_value_title` are used (all lowercase, removed extra whitespaces) to compare the equality of `data_ids` and `filters`

Usage

```
fedstat_data_ids_filter(data_ids, filters, disable_warnings = FALSE)
```

Arguments

`data_ids` data.frame, result of `fedstat_get_data_ids` with or without conjunction with `fedstat_get_data_ids_special_cases_handle`

`filters` JSON in R list form. The structure should be like this:

```
{
 "filter_field_title1": ["filter_value_title1", "filter_value_title2"],
 "filter_field_title2": ["filter_value_title1", "filter_value_title2"],
 ...
}
```

Where for example `filter_field_title1` could be a string "Year" with `filter_value_title1` equal to 2020 and `filter_field_title2` could be a string "OKATO" with `filter_value_title1` equal to "Russian Federation" Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

`disable_warnings` bool, enables or disables following warnings:

1. About non matched `filter_value_title` in `filters` and `filter_value_title` from `data_ids`;
2. About unspecified `filter_filed_title` in filters.

Value

data.frame, filtered `data_ids`

See Also

`fedstat_get_data_ids`, `fedstat_post_data_ids_filtered`

Examples

```
## Not run:
# Get data filters identificators for week prices
# standardize names for DVFO and extract week numbers
# filter the data_ids to get data for week 21 and 22 of 2021
# for all goods and services for Russian Federation
data_ids_filtered <- fedstat_get_data_ids("37426") %>%
  fedstat_get_data_ids_special_cases_handle(
    filter_value_title_alias_lookup_table = data.frame(
      "filter_value_title" = "Dalnevostochnyj federalnyj okrug ( s 03.11.2018)",
      "filter_value_title_alias" = "Dalnevostochnyj federalnyj okrug"
    )
  ) %>%
  fedstat_data_ids_filter(
    filters = list(
      "Territory" = "Russian Federation",
      "Year" = "2021",
      "Period" = c(21, 22),
      "Types of goods and services" = "*"
    )
```

```
  )

# In this example names for Far Eastern Federal District are latinized for CRAN
# Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

## End(Not run)
```

`fedstat_data_load_with_filters`

Download subset of indicator data from fedstat.ru by specifying filters in JSON

Description

This function is a wrapper for the other functions of the package to provide a simple one function API for fedstat.ru

There are two basic terms in this API: `filter_field` and `filter_value`

The filter field reflects the individual property of the data point. For example, Year, Region, Unit of measurement, etc. Each filter field has its own title (`filter_field_title`), it is simply a human-readable word or phrase (e.g. "Year", "Region") that reflects the essence of the property by which filtering takes place

The filter field reflects the individual property specific value of the data point. (e.g. 2021 for Year, "Russian Federation" for Region, etc.) It also has a title (`filter_value_title`) with the same purpose as `filter_field_title`

`filters` should use `filter_field_title` in names and `filter_value_title` in values as they are presented on fedstat.ru. If for some reason the specified filters do not return the expected result, it is worth using `fedstat_get_data_ids` separately and inspecting possible filter values in `data_ids` to see if the strings are defined correctly (e.g. encoding issues, mixing latin and cyrillic symbols)

`filter_value_title` currently supports the following special values:

1. asterix (*), alias for "select all possible filter values for this filter field"

Unspecified filters use asterix as a default (i.e. all possible filter values are selected and a warning is given)

Internally normalized `filter_field_title` and `filter_value_title` are used (all lowercase, removed extra whitespaces) to compare the equality of `data_ids` and `filters`

Usage

```
fedstat_data_load_with_filters(
  indicator_id,
  filters,
 filter_value_title_alias_lookup_table = data.frame(filter_value_title = character(),
    filter_value_title_alias = character()),
  disable_warnings = FALSE,
  httr_verbose = httr::verbose(data_out = FALSE)
)
```

Arguments

`indicator_id` character, indicator id/code from indicator URL. For example for indicator with URL https://www.fedstat.ru/indicator/37426 indicator id will be 37426

`filters` JSON in R list form. The structure should be like this:

```
{
 "filter_field_title1": ["filter_value_title1", "filter_value_title2"],
 "filter_field_title2": ["filter_value_title1", "filter_value_title2"],
  ...
}
```

Where for example `filter_field_title1` could be a string "Year" with `filter_value_title1` equal to 2020 and `filter_field_title2` could be a string "OKATO" with `filter_value_title1` equal to "Russian Federation" Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

`filter_value_title_alias_lookup_table`
data.frame with columns `filter_value_title` and `filter_value_title_alias`. Used to replace `filter_value_title` with standard forms of filter value titles. It is mainly used to set consistent names. For example, the Dalnevostochnyj federalnyj okrug (s 03.11.2018) (transliteration for CRAN) in `filter_value_title` can be simply replaced with `filter_value_title_alias` as Dalnevostochnyj federalnyj okrug. In this example in fact, these are two different entities (after the inclusion of Buryatia and Transbaikalia and before), but in most cases such a replacement is more convenient for loading data, since you need the whole time series. By default it's empty data.frame, e.g. no replacement for anything.

`disable_warnings`
bool, enables or disables following warnings:

1. About non matched `filter_value_title` in `filters` and `filter_value_title` from `data_ids`;
2. About unspecified `filter_filed_title` in filters.

`httr_verbose` `httr::verbose()` or NULL, outputs messages to the console about the processing of the request

Value

data.frame with filtered indicator data from fedstat.ru

See Also

`fedstat_get_data_ids`, `fedstat_get_data_ids_special_cases_handle`, `fedstat_data_ids_filter`, `fedstat_post`

Examples

```
## Not run:
# Download weekly goods and services prices data for week 21 and 22 of 2021
# for all goods and services for Russian Federation
data <- fedstat_data_load_with_filters(
  indicator_id = "37426",
```

```
  filters = list(
    "Territory" = "Russian Federation",
    "Year" = "2021",
    "Period" = c(21, 22),
    "Types of goods and services" = "*"
  )
)
# Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

## End(Not run)
```

`fedstat_get_data_ids` *Get data filters ids from fedstat.ru indicator web page*

Description

To query data from fedstat we need to POST some filters in form of filter numeric identificators. Most filters don't have some rule from which their ids can be generated based on filters titles and values. It seems like these ids are just indexes in the fedstat inner database. So in order to get the data, we first need to get the ids of the filter values by parsing specific part of java script source code on indicator web page.

Usage

```
fedstat_get_data_ids(
  indicator_id,
  timeout_seconds = 180,
  retry_max_times = 3,
  httr_verbose = httr::verbose(data_out = FALSE)
)
```

Arguments

`indicator_id`	character, indicator id/code from indicator URL. For example for indicator with URL https://www.fedstat.ru/indicator/37426 indicator id will be 37426
`timeout_seconds`	numeric, maximum time before a new GET request is tried
`retry_max_times`	numeric, maximum number of tries to GET `data_ids`
`httr_verbose`	`httr::verbose()` or NULL, outputs messages to the console about the processing of the request

Details

It is known that the fedstat lags quite often. Sometimes site never responds at all. This is especially true for the most popular indicators web pages. In this regard, by default, a GET request is sent 3 times with a timeout of 180 seconds and with initially small, but growing exponentially, pauses between requests.

As a rule, requests to the indicator web page take much longer than requests to get the data itself. A POST request for data is sent to a single URL https://www.fedstat.ru/indicator/data.do?format=(excel or sdmx) for all indicators and is often quite fast. In this regard, for many indicators, it makes sense to cache `data_ids` to increase the speed of data download. This is not possible for all data, for example, for weekly prices, each new week adds a new filter (new week), the id of which can only be found on the indicator web page. But for most data (e.g. monthly frequency), time filters are trivial. There are 12 months in total with unique ids that do not change and year ids that match their values (that is, `filter_value_id` = `filter_value`, in other words 2020 = 2020)

Correct filter_field_object_ids are needed to get data. For the sdmx format, these ids do not change anything, except for the standard data sorting, but their incorrect specification will lead either to incomplete data loading or to no data at all. For the excel format, these ids determine the form of data presentation, as in the data preview on the fedstat site. For now only default filter_field_object_ids are used, which are parsed from java script source code on indicator web page. In theory, it is possible to let the user specify filter_field_object_ids himself, but this will add unnecessary complexity and room for errors on the user side.

Value

data.frame with all character type columns:

1. filter_field_id - id for filter field;
2. filter_field_title - filter field title string representation;
3. filter_value_id - id for filter field value;
4. filter_value_title - filter field value title string representation;
5. filter_field_object_ids - special strings that define the location of the filters fields. It can take the following values: lineObjectIds (filters in lines), columnObjectIds (filters in columns), filterObjectIds (hidden filters for all data);
6. filter_field_object_ids_order - sorting for `filter_field_object_ids`, determines the order of the filters fields.

See Also

`fedstat_get_data_ids_special_cases_handle`, `fedstat_data_ids_filter`, `fedstat_post_data_ids_filtered`

Examples

```
## Not run:
# Get data filters identificators for week prices
data_ids <- fedstat_get_indicator_data_ids("37426")

## End(Not run)
```

`fedstat_get_data_ids_special_cases_handle` *Handle special cases strings for* `filter_value_title` *col in* `data_ids`

Description

Handles special cases strings in `filter_value_title` to standardize them. Currently does only 2 things:

1. Replaces `filter_value_title` with given aliases in `filter_value_title_alias_lookup_table`;
2. Replaces week period titles with week numbers to filter by it instead of a complex unstandardized "period" string and adds a new column `filter_value_title_week` for weekly data for using original week title if needed;

Function will be supplemented with new methods for processing special cases as they are found

Usage

```
fedstat_get_data_ids_special_cases_handle(
  data_ids,
 filter_value_title_alias_lookup_table = data.frame(filter_value_title = character(),
    filter_value_title_alias = character())
)
```

Arguments

`data_ids`	data.frame, result of `fedstat_get_data_ids`
`filter_value_title_alias_lookup_table`	data.frame with columns `filter_value_title` and `filter_value_title_alias`. Used to replace `filter_value_title` with standard forms of filter value titles. It is mainly used to set consistent names. For example, the Dalnevostochnyj federalnyj okrug (s 03.11.2018) (transliteration for CRAN) in `filter_value_title` can be simply replaced with `filter_value_title_alias` as Dalnevostochnyj federalnyj okrug. In this example in fact, these are two different entities (after the inclusion of Buryatia and Transbaikalia and before), but in most cases such a replacement is more convenient for loading data, since you need the whole time series. By default it's empty data.frame, e.g. no replacement for anything.

Value

data.frame, data_ids with replaced by aliases `filter_value_title` column and a new column `filter_value_title_week`, which consists only of NA in case of non-weekly data

See Also

`fedstat_get_data_ids`

Examples

```
## Not run:
# Get data filters identificators for week prices
# and standardize names for DVFO and extract week numbers
data_ids_special_cases_handled <- fedstat_get_data_ids("37426") %>%
  fedstat_get_indicator_data_ids_special_cases_handle(
    filter_value_title_alias_lookup_table = data.frame(
      "filter_value_title" = "Dalnevostochnyj federalnyj okrug ( s 03.11.2018)",
      "filter_value_title_alias" = "Dalnevostochnyj federalnyj okrug"
    )
  )
# In this example names for Far Eastern Federal District are latinized for CRAN

## End(Not run)
```

---

fedstat_java_script_data_ids_parse_to_json

Get data ids from java script source

Description

Get data ids from java script source

Usage

```
fedstat_java_script_data_ids_parse_to_json(java_script_source_code)
```

Arguments

java_script_source_code	character, java script source code with data ids

Value

json in form of list with data ids

---

fedstat_java_script_default_data_ids_object_ids_parse_to_json

Get default data ids object ids from java script source

Description

Get default data ids object ids from java script source

Usage

```
fedstat_java_script_default_data_ids_object_ids_parse_to_json(
  java_script_source_code
)
```

Arguments

java_script_source_code	character, java script source code with data ids and default object ids in it

Value

json in form of list with 3 character vectors for lineObjectIds, columnObjectIds, filterObjectIds, which consist of `filters_id`

---

fedstat_parse_sdmx_to_table

Parse sdmx raw bytes to data.frame

Description

Parses sdmx raw bytes received in response to POST request. This function is a wrapper around `readsdmx::read_sdmx` and `rsdmx::readSDMX`, in addition to reading data, automatically adds columns with values from lookup tables

Usage

```
fedstat_parse_sdmx_to_table(data_raw)
```

Arguments

data_raw	sdmx raw bytes

Value

data.frame

See Also

fedstat_parse_sdmx_to_table

Examples

```
## Not run:
# Get data filters identificators for week prices
# standardize names for DVFO and extract week numbers
# filter the data_ids to get data for week 21 and 22 of 2021
# for all goods and services for Russian Federation
# POST filters and download data in sdmx format
# Parse raw sdmx to data.frame
data <- fedstat_get_data_ids("37426") %>%
  fedstat_get_data_ids_special_cases_handle(
    filter_value_title_alias_lookup_table = data.frame(
      "filter_value_title" = "Dalnevostochnyj federalnyj okrug ( s 03.11.2018)",
      "filter_value_title_alias" = "Dalnevostochnyj federalnyj okrug"
    )
  ) %>%
  fedstat_data_ids_filter(
    filters = list(
      "Territory" = "Russian Federation",
      "Year" = "2021",
      "Period" = c(21, 22),
      "Types of goods and services" = "*"
    )
  ) %>%
  fedstat_post_data_ids_filtered() %>%
  fedstat_parse_sdmx_to_table()

# In this example names for DVFO are latinized for CRAN
# Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

## End(Not run)
```

`fedstat_post_data_ids_filtered`

Post data filters ids to fedstat.ru and download specified subset of data

---

Description

Creates a request body from `data_ids` and sends it to https://www.fedstat.ru/indicator/data.do?format=`data_format`. Gets an sdmx or excel with data in binary format.

sdmx raw bytes can be passed to `fedstat_parse_sdmx_to_table` to create a data.frame or to `rawToChar` and `writeLines` to create an xml file

excel raw bytes can be passed to `writeBin` to create an xls file

Usage

```
fedstat_post_data_ids_filtered(
  data_ids,
  data_format = c("sdmx", "excel"),
```

```
  timeout_seconds = 180,
  retry_max_times = 3,
  httr_verbose = httr::verbose(data_out = FALSE)
)
```

Arguments

data_ids	data.frame, can be a result of fedstat_get_data_ids or fedstat_get_data_ids_special_cases_hand to download all available data, or a result of fedstat_data_ids_filter to download subset of available data
data_format	string, one of sdmx, excel
timeout_seconds	numeric, maximum time before a new POST request is tried
retry_max_times	numeric, maximum number of tries to POST data_ids
httr_verbose	httr::verbose() or NULL, outputs messages to the console about the processing of the request

Value

raw bytes (sdmx or excel)

See Also

fedstat_parse_sdmx_to_table

Examples

```
## Not run:
# Get data filters identificators for week prices
# standardize names for DVFO and extract week numbers
# filter the data_ids to get data for week 21 and 22 of 2021
# for all goods and services for Russian Federation
# POST filters and download data in sdmx format
data <- fedstat_get_data_ids("37426") %>%
  fedstat_get_data_ids_special_cases_handle(
    filter_value_title_alias_lookup_table = data.frame(
      "filter_value_title" = "Dalnevostochnyj federalnyj okrug ( s 03.11.2018)",
      "filter_value_title_alias" = "Dalnevostochnyj federalnyj okrug"
    )
  ) %>%
  fedstat_data_ids_filter(
    filters = list(
      "Territory" = "Russian Federation",
      "Year" = "2021",
      "Period" = c(21, 22),
      "Types of goods and services" = "*"
    )
  ) %>%
  fedstat_post_data_ids_filtered()
```

```
# In this example names for DVFO are latinized for CRAN
# Not actual filter field titles and filter values titles because of ASCII requirement for CRAN

## End(Not run)
```

Index