# Package: Rapi (via r-universe)

September 14, 2024

**Type** Package

**Title** Interface for Multiple Data Providers 'EDDS' and 'FRED'

**Version** 1.0.5

**Date** 2024-04-06

**Maintainer** Sermet Pekin <sermet.pekin@gmail.com>

**URL** https://github.com/DataRapi/Rapi, https://DataRapi.github.io/Rapi/

**BugReports** https://github.com/DataRapi/Rapi/issues

**Description** Interface for multiple data sources, such as the 'EDDS' API <https://evds2.tcmb.gov.tr/index.php?/evds/userDocs> of the Central Bank of the Republic of Türkiye and the 'FRED' API <https://fred.stlouisfed.org/docs/api/fred/> of the Federal Reserve Bank. Both data providers require API keys for access, which users can easily obtain by creating accounts on their respective websites. The package provides caching ability with the selection of periods to increase the speed and efficiency of requests. It combines datasets requested from different sources, helping users when the data has common frequencies. While combining data frames whenever possible, it also keeps all requested data available as separate data frames to increase efficiency.

**License** MIT + file LICENSE

**LinkingTo** Rcpp

**Depends** R (>= 3.4.3), Rcpp

**Imports** crayon, digest, dplyr, httr, httr2, glue, jsonlite, lubridate, magrittr, purrr, rlang, rlist, stringr, tibble

**Suggests** writexl , devtools, testthat

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.1

**Encoding** UTF-8

**Config/testthat/edition** 3

**Repository** https://datarapi.r-universe.dev

**RemoteUrl** https://github.com/datarapi/rapi

**RemoteRef** HEAD

**RemoteSha** 58eba58d61994e7096868b0b220898a98a100320

# Contents



---

`change_cache_folder` *Sets the cache folder or changes it if it was already set to save caches.*

---

## Description

Sets the cache folder or changes it if it was already set to save caches.

## Usage

```
change_cache_folder(folder = null, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `folder` | Folder to set as a cache folder. The default value is NULL, which triggers the check_users_choice_if_cache function that provides some options to the user to use it as a cache folder, a temporary one, or disable caching. |
| `verbose` | Boolean. If TRUE, it provides information when the cache folder is set. Otherwise, it only prints a warning when there is an error. |

## Value

No return value, called for side effects

## Examples

```
change_cache_folder("my_cache_folder", verbose = TRUE)
```

---

`excel` *Creates an excel file from a data.frame or a list of data.frame or from Rapi_GETPREP object.*

---

## Description

The excel() function creates an excel file according to the object given. data.frame or List of data frame or Rapi_GETPREP object can be passed..

## Usage

```
excel(
  dfs = null,
  file_name = null,
  folder = null,
  .debug = FALSE,
  env = rlang::caller_env(),
  ...
)
```

## Arguments

| | |
|---|---|
| `dfs` | object or list of data frame to write |
| `file_name` | file name to save |
| `folder` | folder to save file |
| `.debug` | for internal use |
| `env` | environment |
| `...` | for future versions |

## Value

it returns object or list of data frame back

## Examples

```
## Not run:
excel(data.frame(a = 1:3), file_name = "test1.xlsx", folder = ".")

## End(Not run)
```

get_series *Requests data from multiple data sources.*

## Description

The get_series() function retrieves data from various sources, including the EDDS API and FRED API at this version. When multiple indexes are provided as a character vector or string template, the function individually requests each item from the corresponding sources, discerning the source from the item's format. The function combines data frames when there are common frequencies and returns both a combined data frame and individual data frames for each requested item.

## Usage

```
get_series(
  index = NULL,
  start_date = default_start_date(),
  end_date = default_end_date(),
  freq = NULL,
  cache = FALSE,
  na.remove = TRUE,
  verbose = NULL,
  ...,
  source = c("multi", "evds", "fred"),
  base = c("multi", "series", "table"),
  debug = FALSE
)
```

## Arguments

| | |
|---|---|
| index | A character vector or string representing the index to be retrieved. |
| start_date | Limits the start date of the data. |
| end_date | Limits the end date of the data. |
| freq | Frequency of the data (rarely needed). |
| cache | If FALSE, a new request will be made; if TRUE, cached data will be used. |
| na.remove | If TRUE, NA values are removed only if all columns are NA. |
| verbose | If TRUE, prints information during the process; if FALSE, silently does its job. default is NULL which implies applying default verbose option. If this function is called with a TRUE or FALSE value it changes global verbose option for Rapi package. If verbose option is FALSE it gives a warning only if something goes wrong. |
| ... | Additional parameters for future versions. |
| source | Source such as "evds" or "fred" for internal use at this version. |
| base | Table or series on the source for internal use at this version. |
| debug | Debug option for development. |

## Value

An S3 object, Rapi_GETPREP, which has generic functions such as print and excel. The `print` generic provides hints to the user on how to use requested data, such as creating output with the `excel` function or examining requested data in the global environment.

## Examples

```
## Not run:
o <- get_series(template_test())
excel(o)
object <- get_series("UNRATE", start_date = "2000/01/01", na.remove = TRUE)
excel(object)

## End(Not run)
```

---

`inn` *inn*

---

## Description

Checks if the second parameter includes the first one as a value, a column name, or a name.

## Usage

```
inn(x, table)
```

## Arguments

| | |
|---|---|
| `x` | Character to check if it exists in a vector or list. |
| `table` | List, data frame, or any vector. |

## Value

Logical value TRUE if it exists, FALSE if it does not.

## Examples

```
.check <- inn("a", list(a = 1:5))
```

## lag_df *lag_df*

### Description

The lag_df function creates additional columns based on a list of column names and lag sequences. This feature is beneficial for scenarios where you need varying lag selections for certain columns, allowing flexibility in specifying different lags for different columns or opting for no lag at all.

### Usage

```
lag_df(df, laglist)
```

### Arguments

| | |
|---|---|
| df | A data.frame or tibble. |
| laglist | A list of column names where each index corresponds to a column name and the associated value is the lag sequence. |

### Value

tibble

### Examples

```
df <- data.frame(a = 1:15, b = 2:16)
tb <- lag_df(df, laglist = list(a = 1:5, b = 1:3))
```

## lag_df2 *lag_df2*

### Description

The lag_df2 function creates additional columns based on a list of column names and lag sequences. This feature is beneficial for scenarios where you need varying lag selections for certain columns, allowing flexibility in specifying different lags for different columns or opting for no lag at all.

### Usage

```
lag_df2(df, laglist)
```

### Arguments

| | |
|---|---|
| df | A data.frame or tibble. |
| laglist | A list of column names where each index corresponds to a column name and the associated value is the lag sequence. |

## Value

data.frame

## Examples

```
df <- data.frame(a = 1:15, b = 2:16)
df2 <- lag_df2(df, laglist = list(a = 1:5, b = 1:3))
```

---

`print.Rapi_GETPREP` *print.Rapi_GETPREP Generic method for S3 Rapi_GETPREP object*

---

## Description

print.Rapi_GETPREP Generic method for S3 Rapi_GETPREP object

## Usage

```
## S3 method for class 'Rapi_GETPREP'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | S3 Rapi_GETPREP object |
| `...` | further arguments passed to or from other methods. |

## Value

S3 Rapi_GETPREP object

## Examples

```
## Not run:

obj <- get_series(template_test())
print(obj)

## End(Not run)
```

## remove_columns *Remove a column or columns from a data.frame.*

### Description

Remove a column or columns from a data.frame.

### Usage

```
remove_columns(df, column_names, verbose = FALSE)
```

### Arguments

| | |
|---|---|
| df | Data.frame or tibble. |
| column_names | Column name or column names as a character vector. |
| verbose | Boolean, provides extra information when removing a column. |

### Value

Data.frame.

### Examples

```
df <- remove_columns(cars, "speed")
```

## remove_na_safe *remove_na_safe*

### Description

This function removes rows from both ends of a data frame until it identifies a row where all columns have non-NA values. Starting from the beginning, it removes rows until it encounters a row with complete data at a specific row index (e.g., row 5). It then proceeds to remove rows from the end of the data frame, eliminating any rows with at least one NA value in any column. The process stops when it finds a row where all columns contain non-NA values, and the resulting data frame is returned.

### Usage

```
remove_na_safe(df , verbose = FALSE )
```

### Arguments

| | |
|---|---|
| df | data.frame to remove na rows from the beginning and from the end |
| verbose | give detailed info while removing NA values |

## Value

data.frame returns data.frame after removing rows if all columns are NA from the beginning and after

## Examples

```
df <- data.frame(
  a = c(NA, 2:7, NA),
  b = c(NA, NA, 5, NA, 12, NA, 8, 9)
)
df2 <- remove_na_safe(df)
```

---

# set_api_key *set_api_key*

---

## Description

set_api_key

## Usage

```
set_api_key(key = null, source_name = null, option = c("env", "file"), ...)
```

## Arguments

| | |
|---|---|
| key | api key of the source |
| source_name | evds or fred |
| option | choice of later usage. env or file should be given to save api key for later use. Default is env which saves api key as environment variable. if env default value is selected it will save api key as an environment variable if file was selected it will save api key to current folder. |
| ... | for future versions |

## Value

The function has no return value.

## Examples

```
## Not run:

set_api_key("ABCDEFGHIJKLMOP", "evds", "env")
set_api_key("ABCDEFGHIJKLMOP", "fred", "env")
set_api_key("ABCDEFGHIJKLMOP", "fred", "file")

## End(Not run)
```

## template_test *template_test creates a string template for testing and example purposes*

### Description

template_test creates a string template for testing and example purposes

### Usage

```
template_test()
```

### Value

a string template that includes ID examples from different sources

### Examples

```
template_test()
```

## verbose_off *Turn Off Verbose Mode*

### Description

This function turns off verbose mode, suppressing additional informational output. It is useful when you want to limit the amount of information displayed during the execution of certain operations.

### Usage

```
verbose_off()
```

### Details

Verbose mode is often used to provide detailed information about the progress of a function or operation. By calling `verbose_off`, you can disable this additional output.

The `options("Rapi_verbose" = FALSE)` line sets the verbose option to `FALSE`, silencing additional messages.

### Value

The function has no return value.

### See Also

`verbose_on`: Turn on verbose mode.

## Examples

```
verbose_off()
```

---

`verbose_on` *Turn On Verbose Mode*

---

## Description

This function turns on verbose mode, enabling additional informational output. It is useful when you want to receive detailed information about the progress of certain operations.

## Usage

```
verbose_on()
```

## Details

Verbose mode is designed to provide detailed information during the execution of a function or operation. By calling `verbose_on`, you can enable this additional output.

The `options("Rapi_verbose" = TRUE)` line sets the verbose option to `TRUE`, allowing functions to produce more detailed messages.

## Value

The function has no explicit return value.

## See Also

`verbose_off`: Turn off verbose mode.

## Examples

```
verbose_on()
```

`%inn%` *%inn%*

## Description

Checks if the second parameter includes the first one as a value, a column name, or a name.

## Usage

```
x %inn% table
```

## Arguments

| | |
|---|---|
| `x` | Character to check if it exists in a vector or list. |
| `table` | List, data frame, or any vector. |

## Value

Logical value TRUE if it exists, FALSE if it does not.

## Examples

```
.check <- "a" %inn% data.frame(a = 1:5)
```

# Index