Package 'tidyBdE'

December 20, 2024

Title Download Data from Bank of Spain

Version 0.3.8

Description Tools to download data series from 'Banco de España' ('BdE') on 'tibble' format. 'Banco de España' is the national central bank and, within the framework of the Single Supervisory Mechanism ('SSM'), the supervisor of the Spanish banking system along with the European Central Bank. This package is in no way sponsored endorsed or administered by 'Banco de España'.

License GPL (>= 3)

URL https://ropenspain.github.io/tidyBdE/, https://github.com/rOpenSpain/tidyBdE

BugReports https://github.com/rOpenSpain/tidyBdE/issues

Depends R (>= 3.6.0)

Imports dplyr (>= 0.7.0), ggplot2 (>= 3.5.0), readr (>= 1.0.0), scales (>= 1.1.0), tibble (>= 3.0.0), tidyr, utils

Suggests knitr, lifecycle, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/Needs/coverage covr

Config/Needs/website cpp11, devtools, progress, reactable, remotes, styler, tidyverse, ropenspain/rostemplate

Config/testthat/edition 3

Config/testthat/parallel true

Copyright See file inst/COPYRIGHTS

Encoding UTF-8

RoxygenNote 7.3.2

X-schema.org-applicationCategory Macroeconomics

X-schema.org-isPartOf https://ropenspain.es/

X-schema.org-keywords api, bde, cran, ggplot2, macroeconomics, r, r-package, ropenspain, rstats, series-data, spain

NeedsCompilation no

Author Diego H. Herrero [aut, cre, cph]
(<https://orcid.org/0000-0001-8457-4658>)

Maintainer Diego H. Herrero <dev.dieghernan@gmail.com>

Repository CRAN

Date/Publication 2024-12-20 12:40:02 UTC

Contents

`bde_catalog_load` *Load BdE catalogs*

Description

Load the time-series catalogs provided by BdE.

Usage

```
bde_catalog_load(
  catalog = c("ALL", "BE", "SI", "TC", "TI", "PB"),
  parse_dates = TRUE,
  cache_dir = NULL,
  update_cache = FALSE,
  verbose = FALSE
)
```

Arguments

`catalog`	A single value indicating the catalogs to be updated or `"ALL"` as a shorthand. See **Details**.
`parse_dates`	Logical. If `TRUE` the dates would be parsed using `bde_parse_dates()`.
`cache_dir`	A path to a cache directory. The directory can also be set via options with `options(bde_cache_dir = "path/to/dir")`.

update_cache	Logical. If TRUE the requested file would be updated on the cache_dir.
verbose	Logical TRUE or FALSE, display information useful for debugging.

Details

Accepted values for catalog are:

CODE	PUBLICATION	UPDATE FREQUENCY	FREQUENCY
"BE"	Statistical Bulletin	Daily	Monthly
"SI"	Summary Indicators	Daily	Daily
"TC"	Exchange Rates	Daily	Daily
"TI"	Interest Rates	Daily	Daily
"PB"	Bank Lending Survey	Quarterly	Quarterly

Use "ALL" as a shorthand for updating all the catalogs at a glance.

If the requested catalog is not cached bde_catalog_update() is invoked.

Value

A tibble object.

Source

Time-series bulk data download.

See Also

Other catalog: bde_catalog_search(), bde_catalog_update()

Examples

```
bde_catalog_load("TI", verbose = TRUE)
```

---

bde_catalog_search *Search BdE catalogs*

Description

Search for keywords on the time-series catalogs.

Usage

```
bde_catalog_search(pattern, ...)
```

Arguments

pattern	regex pattern to search See **Details** and **Examples**.
...	Arguments passed on to bde_catalog_load
	catalog A single value indicating the catalogs to be updated or "ALL" as a shorthand. See **Details**.
	parse_dates Logical. If TRUE the dates would be parsed using bde_parse_dates().
	update_cache Logical. If TRUE the requested file would be updated on the cache_dir.
	cache_dir A path to a cache directory. The directory can also be set via options with options(bde_cache_dir = "path/to/dir").
	verbose Logical TRUE or FALSE, display information useful for debugging.

Details

Note that BdE files are only provided in Spanish, for the time being. Therefore search terms should be provided in Spanish as well in order to get search results.

This function uses base::grep() function for finding matches on the catalogs. You can pass regular expressions to broaden the search.

Value

A tibble object with the results of the query.

See Also

bde_catalog_load(), base::regex

Other catalog: bde_catalog_load(), bde_catalog_update()

Examples

```
# Simple search (needs to be in Spanish)
# !! PIB [es] == GDP [en]

bde_catalog_search("PIB")

# More complex - Single
bde_catalog_search("Francia(.*)PIB")

# Even more complex - Double
bde_catalog_search("Francia(.*)PIB|Italia(.*)PIB|Alemania(.*)PIB")


# Search a sequential code: Exact match
# Note that this series (sequential code) appears on several tables

bde_catalog_search("^3779313$")
```

`bde_catalog_update` *Update BdE catalogs*

Description

Update the time-series catalogs provided by BdE.

Usage

```
bde_catalog_update(
  catalog = c("ALL", "BE", "SI", "TC", "TI", "PB"),
  cache_dir = NULL,
  verbose = FALSE
)
```

Arguments

`catalog`	A vector of characters indicating the catalogs to be updated or `"ALL"` as a shorthand. See **Details**.
`cache_dir`	A path to a cache directory. The directory can also be set via options with `options(bde_cache_dir = "path/to/dir")`.
`verbose`	Logical `TRUE` or `FALSE`, display information useful for debugging.

Details

Accepted values for `catalog` are:

CODE	PUBLICATION	UPDATE FREQUENCY	FREQUENCY
`"BE"`	Statistical Bulletin	Daily	Monthly
`"SI"`	Summary Indicators	Daily	Daily
`"TC"`	Exchange Rates	Daily	Daily
`"TI"`	Interest Rates	Daily	Daily
`"PB"`	Bank Lending Survey	Quarterly	Quarterly

Use `"ALL"` as a shorthand for updating all the catalogs at a glance.

Value

None. Downloads the catalog file(s) to the local machine.

Source

Time-series bulk data download.

See Also

Other catalog: `bde_catalog_load()`, `bde_catalog_search()`

Examples

```
bde_catalog_update("TI", verbose = TRUE)
```

---

bde_indicators *Relevant Indicators of Spain*

Description

Set of helper functions for downloading some of the most relevant macroeconomic indicators of Spain.

Usage

```
bde_ind_gdp_var(series_label = "GDP_YoY", ...)

bde_ind_unemployment_rate(series_label = "Unemployment_Rate", ...)

bde_ind_euribor_12m_monthly(series_label = "Euribor_12M_Monthly", ...)

bde_ind_euribor_12m_daily(series_label = "Euribor_12M_Daily", ...)

bde_ind_cpi_var(series_label = "Consumer_price_index_YoY", ...)

bde_ind_ibex_monthly(series_label = "IBEX_index_month", ...)

bde_ind_ibex_daily(series_label = "IBEX_index_day", ...)

bde_ind_gdp_quarterly(series_label = "GDP_quarterly_value", ...)

bde_ind_population(series_label = "Population_Spain", ...)
```

Arguments

`series_label`	Optional. Character vector or value. Allows to specify a custom label for the series extracted. It should have the same length than `series_code`.
`...`	Arguments passed on to `bde_series_load`
	`out_format` Defines if the format must be returned as a "long" dataset or a "wide" dataset. Possible values are `"wide"` or `"long"`. See **Value** for Details and Section **Examples**.
	`parse_numeric` Logical. If `TRUE` the columns would be parsed to double (numeric) values. See **Note**.
	`extract_metadata` Logical `TRUE/FALSE`. On `TRUE` the output is the metadata of the requested series.

parse_dates Logical. If TRUE the dates would be parsed using bde_parse_dates().

update_cache Logical. If TRUE the requested file would be updated on the cache_dir.

cache_dir A path to a cache directory. The directory can also be set via options with options(bde_cache_dir = "path/to/dir").

verbose Logical TRUE or FALSE, display information useful for debugging.

Details

This functions are convenient wrappers of bde_series_load() referencing specific series. Use verbose = TRUE, extract_metadata = TRUE options to see the specification and the source.

Value

A tibble with the required series.

See Also

bde_series_load(), bde_catalog_search()

Examples

```
bde_ind_gdp_var()
```

---

bde_parse_dates *Parse dates*

Description

This function is tailored for the date formatting used on this package, so it may fail if it is used for another datasets. See **Examples** for checking which formats would be considered.

Date Formats:

FREQUENCY	FORMAT
Daily / Business day	DD MMMMYYYY
Monthly	MMM YYYY
Quarterly	MMM YYYY, where MMM is the first or the last month of the quarter, depending on the value of its v
Half-yearly	MMM YYYY, where MMM is the first or the last month of the halfyear period, depending on the valu
Annual	YYYY

Usage

```
bde_parse_dates(dates_to_parse)
```

Arguments

`dates_to_parse` Dates to parse

Details

Tries to parse strings representing dates using `as.Date()`

Value

A `Date` object.

See Also

`as.Date()`

Examples

```
# Formats parsed
would_parse <- c(
  "02 FEB2019", "15 ABR 1890", "MAR 2020", "ENE2020",
  "2020", "12-1993", "01-02-2014", "01/02/1990"
)

parsed_ok <- bde_parse_dates(would_parse)

class(parsed_ok)

tibble::tibble(raw = would_parse, parsed = parsed_ok)

#-----------------------------------

# Formats not admitted
wont_parse <- c("JAN2001", "2010-01-12", "01 APR 2017", "01/31/1990")

parsed_fail <- bde_parse_dates(wont_parse)

class(parsed_fail)

tibble::tibble(raw = wont_parse, parsed = parsed_fail)
```

`bde_series_full_load` *Load BdE full time-series files*

Description

Load a full time-series file provided by BdE.

Usage

```
bde_series_full_load(
  series_csv,
  parse_dates = TRUE,
  parse_numeric = TRUE,
  cache_dir = NULL,
  update_cache = FALSE,
  verbose = FALSE,
  extract_metadata = FALSE
)
```

Arguments

series_csv	csv file of a series, as defined in the field Nombre del archivo con los valores de la serie of the corresponding catalog. See bde_catalog_load().
parse_dates	Logical. If TRUE the dates would be parsed using bde_parse_dates().
parse_numeric	Logical. If TRUE the columns would be parsed to double (numeric) values. See **Note**.
cache_dir	A path to a cache directory. The directory can also be set via options with options(bde_cache_dir = "path/to/dir").
update_cache	Logical. If TRUE the requested file would be updated on the cache_dir.
verbose	Logical TRUE or FALSE, display information useful for debugging.
extract_metadata	Logical TRUE/FALSE. On TRUE the output is the metadata of the requested series.

Details

About BdE file naming:

The series name is a positional code showing the location of the table. For example, table **be_6_1** represents the Table 1, Chapter 6 of the Statistical Bulletin ("BE"). Although it is a unique value, it is subject to change (i.e. a new table is inserted before).

For that reason, the function bde_series_load() is more suitable for extracting specific time-series.

Value

A tibble with a field Date and the alias of the fields series as described on the catalogs. See bde_catalog_load().

Note

This function tries to coerce the columns to numbers. For some series a warning may be displayed if the parser fails. You can override the default behavior with parse_numeric = FALSE

See Also

Other series: bde_series_load()

Examples

```
# Metadata
bde_series_full_load("TI_1_1.csv", extract_metadata = TRUE)

# Data
bde_series_full_load("TI_1_1.csv")
```

`bde_series_load` *Load a single BdE time-series*

Description

The series alias is a positional code showing the location (column and/or row) of the series in the table. However, although it is unique, it is not a good candidate to be used as the series ID, as it is subject to change. If a series changes position in the table, its alias will also change.

To ensure series can still be identified, even after these changes, they are assigned a sequential number (`series_code` on this function) which will remain unchanged throughout the series' lifetime.

Note that a single series could be used on different tables, so it can have several aliases. If you need to search by alias it is recommended to use `bde_series_full_load()`.

Usage

```
bde_series_load(
  series_code,
  series_label = NULL,
  out_format = "wide",
  parse_dates = TRUE,
  parse_numeric = TRUE,
  cache_dir = NULL,
  update_cache = FALSE,
  verbose = FALSE,
  extract_metadata = FALSE
)
```

Arguments

`series_code`	a numeric (or coercible with `base::as.double()` value or vector with time-series code(s), as defined in the field `Número secuencial` of the corresponding series. See `bde_catalog_load()`.
`series_label`	Optional. Character vector or value. Allows to specify a custom label for the series extracted. It should have the same length than `series_code`.
`out_format`	Defines if the format must be returned as a "long" dataset or a "wide" dataset. Possible values are `"wide"` or `"long"`. See **Value** for Details and Section **Examples**.

parse_dates	Logical. If TRUE the dates would be parsed using bde_parse_dates().
parse_numeric	Logical. If TRUE the columns would be parsed to double (numeric) values. See **Note**.
cache_dir	A path to a cache directory. The directory can also be set via options with options(bde_cache_dir = "path/to/dir").
update_cache	Logical. If TRUE the requested file would be updated on the cache_dir.
verbose	Logical TRUE or FALSE, display information useful for debugging.
extract_metadata	Logical TRUE/FALSE. On TRUE the output is the metadata of the requested series.

Details

Load a single time-series provided by BdE.

Value

A tibble with a field Date and :

- With out_format = "wide" each series is presented in a separate column with the name defined by series_label.
- With out_format = "long" the tibble would have two more columns, serie_name with the labels of each series and serie_value with the value of the series.

"wide" format is more suitable for exporting to a .csv file while "long" format is more suitable for producing plots with ggplot2::ggplot(). See also tidyr::pivot_longer() and tidyr::pivot_wider().

Note

This function tries to coerce the columns to numbers. For some series a warning may be displayed if the parser fails. You can override the default behavior with parse_numeric = FALSE

See Also

bde_catalog_load(), bde_catalog_search(), bde_indicators()

Other series: bde_series_full_load()

Examples

```
# Metadata
bde_series_load(573234, verbose = TRUE, extract_metadata = TRUE)

# Data
bde_series_load(573234, extract_metadata = FALSE)

# Vectorized
bde_series_load(c(573234, 573214),
  series_label = c("US/EUR", "GBP/EUR"),
  extract_metadata = TRUE
```

```
)

wide <- bde_series_load(c(573234, 573214),
  series_label = c("US/EUR", "GBP/EUR")
)

# Wide format
wide


# Long format
long <- bde_series_load(c(573234, 573214),
  series_label = c("US/EUR", "GBP/EUR"),
  out_format = "long"
)

long


# Use with ggplot
library(ggplot2)


ggplot(long, aes(Date, serie_value)) +
  geom_line(aes(group = serie_name, color = serie_name)) +
  scale_color_bde_d() +
  theme_tidybde()
```

---

bde_tidy_palettes *BdE color palettes*

Description

Custom palettes based on the publications of BdE. These are manual palettes with a maximum of 6 colors.

Usage

```
bde_tidy_palettes(
  n = 6,
  palette = c("bde_vivid_pal", "bde_rose_pal", "bde_qual_pal"),
  alpha = NULL,
  rev = FALSE
)
```

Arguments

n	The number of colors (>= 1) to be in the palette.
palette	A valid palette name.
alpha	An alpha-transparency level in the range [0,1] (0 means transparent and 1 means opaque). A missing, i.e., alpha = NULL, does not add opacity codes ("FF") to the individual color hex codes. See ggplot2::alpha().
rev	Logical indicating whether the ordering of the colors should be reversed.

Value

A vector of colors.

See Also

Other bde_plot: scales_bde, theme_tidybde()

Examples

```
# BdE vivid pal
scales::show_col(bde_tidy_palettes(palette = "bde_vivid_pal"),
  labels = FALSE
)

# BdE rose pal
scales::show_col(bde_tidy_palettes(palette = "bde_rose_pal"), labels = FALSE)

# BdE qual pal
scales::show_col(bde_tidy_palettes(palette = "bde_qual_pal"), labels = FALSE)
```

---

scales_bde *BdE scales for* R*hrefhttps://CRAN.R-project.org/package=ggplot2*ggplot2

Description

Scales to be used with the **ggplot2** package. Discrete palettes are named as scale_*_bde_d while continuous palettes are named scale_*_bde_c.

Usage

```
scale_color_bde_d(
  palette = c("bde_vivid_pal", "bde_rose_pal", "bde_qual_pal"),
  alpha = NULL,
  rev = FALSE,
  ...
)
```

```
scale_fill_bde_d(
  palette = c("bde_vivid_pal", "bde_rose_pal", "bde_qual_pal"),
  alpha = NULL,
  rev = FALSE,
  ...
)

scale_color_bde_c(
  palette = c("bde_rose_pal", "bde_vivid_pal", "bde_qual_pal"),
  alpha = NULL,
  rev = FALSE,
  guide = "colorbar",
  ...
)

scale_fill_bde_c(
  palette = c("bde_rose_pal", "bde_vivid_pal", "bde_qual_pal"),
  alpha = NULL,
  rev = FALSE,
  guide = "colorbar",
  ...
)
```

Arguments

palette	Name of the BdE palette to apply. See bde_tidy_palettes() for details.
alpha	An alpha-transparency level in the range [0,1] (0 means transparent and 1 means opaque). A missing, i.e., alpha = NULL, does not add opacity codes ("FF") to the individual color hex codes. See ggplot2::alpha().
rev	Logical indicating whether the ordering of the colors should be reversed.
...	Further arguments of ggplot2::discrete_scale() or ggplot2::continuous_scale().
guide	A function used to create a guide or its name. See guides() for more information.

Value

A **ggplot2** color scale.

See Also

ggplot2::discrete_scale(), ggplot2::continuous_scale()

Other bde_plot: bde_tidy_palettes(), theme_tidybde()

Examples

```
library(ggplot2)

set.seed(596)
```

```
txsamp <- subset(
  txhousing,
  city %in% c(
    "Houston", "Fort Worth",
    "San Antonio", "Dallas", "Austin"
  )
)

ggplot(txsamp, aes(x = sales, y = median)) +
  geom_point(aes(colour = city)) +
  scale_color_bde_d() +
  theme_minimal()


ggplot(txsamp, aes(x = sales, y = median)) +
  geom_point(aes(colour = city)) +
  scale_color_bde_d("bde_qual_pal") +
  theme_minimal()
```

---

theme_tidybde *BdE theme* R*hrefhttps://CRAN.R-project.org/package=ggplot2***ggplot2**

Description

A custom **ggplot2** theme based on the publications of BdE.

Usage

```
theme_tidybde(...)
```

Arguments

`...`	Arguments passed on to `ggplot2::theme_classic`
	`base_size` base font size, given in pts.
	`base_family` base font family
	`base_line_size` base size for line elements
	`base_rect_size` base size for rect elements

Details

Theme based on `ggplot2::theme_classic()`.

Value

A **ggplot2** `theme()`.

See Also

ggplot2::theme_classic()

Other bde_plot: bde_tidy_palettes(), scales_bde

Examples

```
library(ggplot2)
library(dplyr)
library(tidyr)

series_TC <- bde_series_full_load("TC_1_1.csv")

# If download was OK then plot
if (nrow(series_TC) > 0) {
  series_TC <- series_TC[c(1, 2)]

  series_TC_pivot <- series_TC %>%
    filter(
      Date >= "2020-01-01" & Date <= "2020-12-31",
      !is.na(series_TC[[2]])
    )

  names(series_TC_pivot) <- c("x", "y")

  ggplot(series_TC_pivot, aes(x = x, y = y)) +
    geom_line(linewidth = 0.8, color = bde_tidy_palettes(n = 1)) +
    labs(
      title = "Title",
      subtitle = "Some metric",
      caption = "Bank of Spain"
    ) +
    theme_tidybde()
}
```

Index