

# Package: rb3 (via r-universe)

August 26, 2024

**Title** Download and Parse Public Data Released by B3 Exchange

**Description** Download and parse public files released by B3 and convert them into useful formats and data structures common to data analysis practitioners.

**Version** 0.0.11

**License** MIT + file LICENSE

**Depends** R (>= 4.1.0),

**Imports** bizdays, stringr, proto, cli, readr, dplyr, rvest, httr, jsonlite, purrr, ascii, rlang, methods, yaml, digest, base64enc, XML, readxl, tidy

**Suggests** testthat, knitr, DT, miniUI, shiny, xtable, rmarkdown, ggplot2, covr, scales, magrittr, tibble, withr, vcr (>= 0.6.0)

**Collate** 'rb3-package.R' 'util.R' 'transmute.R' 'fields.R' 'handlers.R' 'marketdata.R' 'download-data.R' 'file.R' 'convert\_to.R' 'scraper-futures.R' 'scraper-yc.R' 'scraper-cotahist.R' 'scraper-indexes.R' 'scraper-company.R' 'addin-show-templates.R' 'addin-display-template.R' 'readers.R' 'downloaders.R' 'zzz.R'

**BugReports** <https://github.com/ropensci/rb3/issues>

**URL** <https://github.com/ropensci/rb3>, <http://ropensci.github.io/rb3/>

**VignetteBuilder** knitr

**RoxygenNote** 7.3.2

**Config/testthat/edition** 3

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Wilson Freitas [aut, cre], Marcelo Perlin [aut]

**Maintainer** Wilson Freitas <[wilson.freitas@gmail.com](mailto:wilson.freitas@gmail.com)>

**Repository** CRAN

**Date/Publication** 2024-07-13 23:00:02 UTC

## Contents

cachedir	2
clearcache	3
code2month	3
company_cash_dividends_get	4
company_info_get	5
company_stock_dividends_get	5
company_subscriptions_get	6
convert_to	7
cotahist-extracts	8
cotahist-options-superset	10
cotahist_get	11
display_template	12
download_marketdata	12
futures_get	13
indexes_get	14
indexes_last_update	15
indexreport_get	15
index_by_segment_get	16
index_comp_get	17
index_get	18
index_weights_get	19
maturity2date	19
read_marketdata	20
show_templates	21
yc_get	22
yc_superset	24
<b>Index</b>	<b>26</b>

---

cachedir	<i>Returns rb3 package cache directory</i>
----------	--

---

### Description

Returns rb3 package cache directory

### Usage

```
cachedir()
```

### Details

In order to set a default directory for cache, which is a good idea for those who want to increase data historically, the option `rb3.cachedir` can be set. Once it is set, the defined directory will be used as the default `cachedir`.

**Value**

a string with the file path of rb3 cache directory

**Examples**

```
cachedir()
```

---

clearcache	<i>Clear cache directory</i>
------------	------------------------------

---

**Description**

Clear cache directory

**Usage**

```
clearcache()
```

**Value**

Has no return

**Examples**

```
## Not run:  
clearcache()  
  
## End(Not run)
```

---

code2month	<i>Get month from maturity code</i>
------------	-------------------------------------

---

**Description**

Get the corresponding month for the string that represent maturities of futures contracts.

**Usage**

```
code2month(x)
```

**Arguments**

x a character with letters that represent the month of maturity of futures contracts.

**Value**

a vector of integers

## Examples

```
code2month(c("F", "G", "H", "J", "K", "M", "N", "Q", "U", "V", "X", "Z"))
code2month(c("JAN", "FEV", "MAR", "NOV", "DEZ"))
```

---

company\_cash\_dividends\_get

*Gets company's dividends in cash*

---

## Description

Gets a list of all dividends in cash paid by the company. *A cash dividend is a payment made by a company out of its earnings to investors in the form of cash.* (<https://www.investopedia.com/>)

## Usage

```
company_cash_dividends_get(code, cache_folder = cachedir(), do_cache = TRUE)
```

## Arguments

code	Represents the company, can be the stock symbol, like PETR4 or the first four characters PETR
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

## Details

The code parameter can be the stock symbol, but the returned data refers to the company, always. The returned data.frame has all company's symbols that paid dividends in cash.

## Value

data.frame with company information

## Examples

```
## Not run:
company_cash_dividends_get(c("PETR", "VALE", "MGLU"))

## End(Not run)
```

---

company_info_get	<i>Gets information about the company</i>
------------------	---

---

**Description**

Gets informations like sector, subsector, segment, total number of shares and many more.

**Usage**

```
company_info_get(code, cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

code	Represents the company, can be the stock symbol, like PETR4 or the first four characters PETR
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

**Details**

The code parameter can be the stock symbol, but the returned data refers to the company, always.

**Value**

data.frame with company information

**Examples**

```
## Not run:  
company_info_get(c("PETR", "VALE", "MGLU"))  
  
## End(Not run)
```

---

company_stock_dividends_get	<i>Gets company's stocks dividends</i>
-----------------------------	--

---

**Description**

Gets a list of all stocks dividends paid by the company. *A stock dividend is a payment to shareholders that consists of additional shares rather than cash.* (<https://www.investopedia.com/>)

**Usage**

```
company_stock_dividends_get(code, cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

code	Represents the company, can be the stock symbol, like PETR4 or the first four characters PETR
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

**Details**

The code parameter can be the stock symbol, but the returned data refers to the company, always. The returned data.frame has all company's symbols that paid dividends in stocks.

**Value**

data.frame with all stocks dividends

**Examples**

```
## Not run:
company_stock_dividends_get(c("PETR", "VALE", "MGLU"))

## End(Not run)
```

---

company\_subscriptions\_get

*Gets company's subscription rights*

---

**Description**

Gets a list of all company's subscription rights. *A subscription right is the right of existing shareholders in a company to retain an equal percentage ownership by subscribing to new stock issuances at or below market prices.* (<https://www.investopedia.com/>)

**Usage**

```
company_subscriptions_get(code, cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

code	Represents the company, can be the stock symbol, like PETR4 or the first four characters PETR
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

**Details**

The code parameter can be the stock symbol, but the returned data refers to the company, always. The returned data.frame has all company's symbols that have issued subscription rights.

**Value**

data.frame with company information

**Examples**

```
## Not run:  
company_subscriptions_get(c("PDGR", "VALE", "MGLU"))  
  
## End(Not run)
```

---

convert_to	<i>Converts B3 messy files to structured formats</i>
------------	--

---

**Description**

Convert B3 files to structured formats based on the template.

**Usage**

```
convert_to(  
  filename,  
  template = NULL,  
  parse_fields = TRUE,  
  format = "csv",  
  destdir = NULL  
)
```

**Arguments**

filename	a string containing a path for the file.
template	a string with the template name.
parse_fields	a logical indicating if the fields must be parsed.
format	output format
destdir	a string with destination directory to save converted file

**Value**

a string with the file path of generated file.

**See Also**

read\_marketdata

## Examples

```
## Not run:  
f <- system.file("extdata/Indic.txt", package = "rb3")  
res <- convert_to(f, output_format = "csv")  
res <- convert_to(f, output_format = "json")  
  
## End(Not run)
```

---

cotahist-extracts

*Extract data from COTAHIST dataset*

---

## Description

Extracts specific data from COTAHIST dataset: stocks, funds, BDRs, ETFs, UNITs, options on stocks, options on indexes, ...

## Usage

```
cotahist_equity_get(x)  
  
cotahist_bdrs_get(x)  
  
cotahist_units_get(x)  
  
cotahist_etfs_get(x)  
  
cotahist_fiis_get(x)  
  
cotahist_fidcs_get(x)  
  
cotahist_fiagros_get(x)  
  
cotahist_indexes_get(x)  
  
cotahist_equity_options_get(x)  
  
cotahist_index_options_get(x)  
  
cotahist_funds_options_get(x)  
  
cotahist_get_symbols(x, symbols)
```

## Arguments

x	COTAHIST dataset returned from cotahist_get.
symbols	list of symbols to extract market data from cotahist



**Value**

a data.frame with prices, volume, traded quantities informations

**Examples**

```
## Not run:
df <- cotahist_equity_get(x)

## End(Not run)
## Not run:
df <- cotahist_brds_get(x)

## End(Not run)
## Not run:
df <- cotahist_units_get(x)

## End(Not run)
## Not run:
df <- cotahist_etfs_get(x)

## End(Not run)
## Not run:
df <- cotahist_fiis_get(x)

## End(Not run)
## Not run:
df <- cotahist_fidcs_get(x)

## End(Not run)
## Not run:
df <- cotahist_fiagros_get(x)

## End(Not run)
## Not run:
df <- cotahist_indexes_get(x)

## End(Not run)
## Not run:
df <- cotahist_equity_options_get(x)

## End(Not run)
## Not run:
df <- cotahist_index_options_get(x)

## End(Not run)
## Not run:
df <- cotahist_funds_options_get(x)

## End(Not run)
## Not run:
df <- cotahist_get_symbols(x, c("BBDC4", "ITSA4", "JHSF3"))
```

```
## End(Not run)
```

---

```
cotahist-options-superset
```

*Extracts equity option superset of data*

---

## Description

Equity options superset is a dataframe that brings together all data regarding equities, equity options and interest rates. This data forms a complete set (superset) up and ready to run options models, implied volatility calculations and volatility models.

## Usage

```
cotahist_equity_options_superset(ch, yc)
```

```
cotahist_options_by_symbol_superset(symbol, ch, yc)
```

## Arguments

ch	cotahist data structure
yc	yield curve
symbol	character with the name of the stock

## Value

A dataframe with data of equities, equity options, and interest rates.

## Examples

```
## Not run:  
refdate <- Sys.Date() - 1  
ch <- cotahist_get(refdate, "daily")  
yc <- yc_get(refdate)  
ch_ss <- cotahist_equity_options_superset(ch, yc)  
petr4_ch_ss <- cotahist_options_by_symbol_superset("PETR4", ch, yc)  
  
## End(Not run)
```

---

cotahist_get	<i>Get COTAHIST data from B3</i>
--------------	----------------------------------

---

**Description**

Download COTAHIST file and parses it returning structured data into R objects.

**Usage**

```
cotahist_get(  
  refdate,  
  type = c("yearly", "monthly", "daily"),  
  cache_folder = cachedir(),  
  do_cache = TRUE  
)
```

**Arguments**

refdate	the reference date used to download the file. This reference date will be formatted as year/month/day according to the given type. Accepts ISO formatted date strings.
type	a string with yearly for all data of the given year, monthly for all data of the given month and daily for the given day.
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE) All valuable information is in the HistoricalPrices element of the returned list. Header and Trailer have informations regarding file generation. The HistoricalPrices element has a data.frame with data of many assets traded in the stock exchange: stocks, bdrs, funds, ETFs, equity options, forward contracts on equities and a few warrants due to some corporate events.

**Value**

a list with 3 data.frames: Header, HistoricalPrices, Trailer.

**Examples**

```
## Not run:  
# get all data to the year of 2001  
df_2001 <- cotahist_get("2001-01-01", "yearly")  
# get data of January of 2001  
df_200101 <- cotahist_get("2001-01-01", "monthly")  
# get data of 2001-01-02  
df_daily <- cotahist_get("2001-01-02", "daily")  
  
## End(Not run)
```

---

display\_template      *Display templates*

---

### Description

display\_template opens an **RStudio gadget** and **addin** that allows users to query for specific attributes of templates.

### Usage

```
display_template()
```

### Value

Addin has no return

### Examples

```
## Not run:
display_template()

## End(Not run)
```

---

download\_marketdata      *Download datasets*

---

### Description

Download datasets for a given template.

### Usage

```
download_marketdata(template, cache_folder = cachedir(), do_cache = TRUE, ...)
```

### Arguments

template	the template name
cache_folder	Location of cache folder (default = cachedir())
do_cache	a logical indicating if the existing file (previously downloaded) should be used or replaced.
...	additional arguments

### Value

a string with the file path of downloaded file or NULL if download fails.

This function downloads data sets for those templates that specifies a downloader attribute. If dest is not provided, cache\_folder is used and a file with template id is saved inside it.

**Examples**

```
## Not run:
fname <- download_marketdata("CDIIDI")

## End(Not run)
```

---

futures\_get

*Get futures prices from trading session settlements page*


---

**Description**

Scrape page [https://www.b3.com.br/en\\_us/market-data-and-indices/data-services/market-data/historical-data/derivatives/trading-session-settlements/](https://www.b3.com.br/en_us/market-data-and-indices/data-services/market-data/historical-data/derivatives/trading-session-settlements/) to get futures prices.

**Usage**

```
futures_mget(
  first_date = Sys.Date() - 5,
  last_date = Sys.Date(),
  by = 1,
  cache_folder = cachedir(),
  do_cache = TRUE
)

futures_get(refdate = Sys.Date(), cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

first_date	First date ("YYYY-MM-DD") to yc_mget multiple curves
last_date	Last date ("YYYY-MM-DD") to yc_mget multiple curves
by	Number of days in between fetched dates (default = 1) in yc_mget
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)
	futures_get returns the future contracts for the given date and futures_mget returns future contracts for multiple dates in a given range.
refdate	Specific date ("YYYY-MM-DD") to yc_get single curve

**Value**

data.frame with futures prices.

**Examples**

```
## Not run:
df <- futures_get("2022-04-18", "2022-04-22")

## End(Not run)
## Not run:
df_fut <- futures_get(Sys.Date())
head(df_fut)

## End(Not run)
```

---

indexes_get	<i>Get B3 indexes available</i>
-------------	---------------------------------

---

**Description**

Gets B3 indexes available.

**Usage**

```
indexes_get(cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

**Value**

a character vector with symbols of indexes available

**Examples**

```
## Not run:
indexes_get()

## End(Not run)
```

---

indexes\_last\_update     *Get the date of indexes composition last update*

---

**Description**

Gets the date where the indexes have been updated lastly.

**Usage**

```
indexes_last_update(cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

cache\_folder     Location of cache folder (default = cachedir())  
do\_cache         Whether to use cache or not (default = TRUE)

**Value**

the Date when the indexes have been updated

**Examples**

```
## Not run:  
indexes_last_update()  
  
## End(Not run)
```

---

indexreport\_get         *Fetches indexes data from B3*

---

**Description**

Downloads index data from B3 website [https://www.b3.com.br/pt\\_br/market-data-e-indices/servicos-de-dados/market-data/historico/boletins-diarios/pesquisa-por-pregao/pesquisa-por-pregao/](https://www.b3.com.br/pt_br/market-data-e-indices/servicos-de-dados/market-data/historico/boletins-diarios/pesquisa-por-pregao/pesquisa-por-pregao/).

**Usage**

```
indexreport_mget(  
  first_date = Sys.Date() - 5,  
  last_date = Sys.Date(),  
  by = 1,  
  cache_folder = cachedir(),  
  do_cache = TRUE  
)  
  
indexreport_get(  

```

```

    refdate = Sys.Date(),
    cache_folder = cachedir(),
    do_cache = TRUE
  )

```

### Arguments

first_date	First date ("YYYY-MM-DD") to yc_mget multiple curves
last_date	Last date ("YYYY-MM-DD") to yc_mget multiple curves
by	Number of days in between fetched dates (default = 1) in yc_mget
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)
refdate	Specific date ("YYYY-MM-DD") to yc_get single curve

### Details

indexreport\_get returns index data for the given date and indexreport\_mget returns index data for a given range of dates.

### Value

A dataframe with index data (OHLC, average and daily oscillation)

### Examples

```

## Not run:
df_ir <- indexreport_mget(Sys.Date() - 5, Sys.Date())
head(df_ir)

## End(Not run)
## Not run:
df_ir <- indexreport_get(Sys.Date())
head(df_ir)

## End(Not run)

```

---

index\_by\_segment\_get *Get B3 indexes available*

---

### Description

Gets B3 indexes available.

### Usage

```
index_by_segment_get(index_name, cache_folder = cachedir(), do_cache = TRUE)
```



**Arguments**

index\_name      a string with the index name  
 cache\_folder    Location of cache folder (default = cachedir())  
 do\_cache        Whether to use cache or not (default = TRUE)

**Value**

A dataframe with the index stocks, their weights, segments and positions.

**Examples**

```
## Not run:
index_by_segment_get("IBOV")

## End(Not run)
```

---

index_comp_get	<i>Get composition of B3 indexes</i>
----------------	--------------------------------------

---

**Description**

Gets the composition of listed B3 indexes.

**Usage**

```
index_comp_get(index_name, cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

index\_name      a string with the index name  
 cache\_folder    Location of cache folder (default = cachedir())  
 do\_cache        Whether to use cache or not (default = TRUE)

**Value**

a character vector with symbols that belong to the given index name

**Examples**

```
## Not run:
index_comp_get("IBOV")

## End(Not run)
```

---

index_get	<i>Get index historical data</i>
-----------	----------------------------------

---

## Description

Gets historical data from B3 indexes

## Usage

```
index_get(  
  index_name,  
  first_date,  
  last_date = Sys.Date(),  
  cache_folder = cachedir(),  
  do_cache = TRUE  
)
```

## Arguments

index_name	a string with the index name
first_date	First date
last_date	Last date
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

## Value

A data.frame/tibble with index data

## Examples

```
## Not run:  
index_get("IBOV", as.Date("1977-01-01"), as.Date("1999-12-31"))  
  
## End(Not run)
```

---

index_weights_get	<i>Get the assets weights of B3 indexes</i>
-------------------	---

---

**Description**

Gets the assets weights of B3 indexes.

**Usage**

```
index_weights_get(index_name, cache_folder = cachedir(), do_cache = TRUE)
```

**Arguments**

index_name	a string with the index name
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)

**Value**

data.frame with symbols that belong to the given index name with its weights and theoretical positions.

**Examples**

```
## Not run:  
index_weights_get("IBOV")  
  
## End(Not run)
```

---

maturity2date	<i>Get maturity date from maturity code</i>
---------------	---

---

**Description**

Get the corresponding maturity date for the three characters string that represent maturity of futures contracts.

**Usage**

```
maturity2date(x, expr = "first day", refdate = NULL)
```

**Arguments**

x	a character vector with three letters string that represent maturity of futures contracts.
expr	a string which indicates the day to use in maturity date. See bizdays::getdate for more details on this argument.
refdate	reference date to be passed. It is necessary to convert old maturities like JAN0, that can be Jan/2000 or Jan/2010. If refdate is greater than 2001-01-01 JAN0 is converted to Jan/2010, otherwise, Jan/2000.

**Value**

a Date vector with maturity dates

**Examples**

```
maturity2date(c("F22", "F23", "G23", "H23", "F45"), "first day")
maturity2date(c("F23", "K35"), "15th day")
maturity2date(c("AG02", "SET2"), "first day")
```

---

read_marketdata	<i>Read and parses files delivered by B3</i>
-----------------	--

---

**Description**

B3, and previously BMF&Bovespa, used to deliver many files with a diverse set of valuable data and informations that can be used to study of can be called of marketdata. There are files with informations about futures, option, interest rates, currency rates, bonds and many other subjects.

**Usage**

```
read_marketdata(
  filename,
  template = NULL,
  parse_fields = TRUE,
  do_cache = TRUE
)
```

**Arguments**

filename	a string containing a path for the file.
template	a string with the template name.
parse_fields	a logical indicating if the fields must be parsed.
do_cache	Whether to use cache or not (default = TRUE)

Each template has a default value for the filename, if the given file name equals one template filename attribute, the matched template is used to parse the file. Otherwise the template must be provided.

The function show\_templates can be used to view the available templates and their default filenames.

**Value**

data.frame of a list of data.frame containing data parsed from files.

**See Also**

show\_templates display\_template

**Examples**

```
## Not run:  
# Eletro.txt matches the filename of Eletro template  
path <- "Eletro.txt"  
df <- read_marketdata(path)  
path <- "Indic.txt"  
df <- read_marketdata(path, template = "Indic")  
path <- "PUWEB.TXT"  
df <- read_marketdata(path, template = "PUWEB")  
  
## End(Not run)
```

---

show_templates	<i>Show templates.</i>
----------------	------------------------

---

**Description**

display\_template opens an **RStudio gadget** and **addin** that allows users to view the available templates.

**Usage**

```
show_templates()
```

**Value**

Addin has no return

**Examples**

```
## Not run:  
show_templates()  
  
## End(Not run)
```

---

 yc\_get

*Fetches Yield Curve Data from B3*


---

### Description

Downloads yield curve data from B3 website <https://www2.bmf.com.br/pages/portal/bmfbovespa/lumis/lum-texas-referenciais-bmf-ptBR.asp>. Particularly, we import data for

- DI X Pre (yc\_get)
- Cupom limpo (yc\_usd\_get)
- DI x IPCA (yc\_ipca\_get)

### Usage

```
yc_mget(
  first_date = Sys.Date() - 5,
  last_date = Sys.Date(),
  by = 1,
  cache_folder = cachedir(),
  do_cache = TRUE
)
```

```
yc_get(refdate = Sys.Date(), cache_folder = cachedir(), do_cache = TRUE)
```

```
yc_ipca_mget(
  first_date = Sys.Date() - 5,
  last_date = Sys.Date(),
  by = 1,
  cache_folder = cachedir(),
  do_cache = TRUE
)
```

```
yc_ipca_get(refdate = Sys.Date(), cache_folder = cachedir(), do_cache = TRUE)
```

```
yc_usd_mget(
  first_date = Sys.Date() - 5,
  last_date = Sys.Date(),
  by = 1,
  cache_folder = cachedir(),
  do_cache = TRUE
)
```

```
yc_usd_get(refdate = Sys.Date(), cache_folder = cachedir(), do_cache = TRUE)
```

### Arguments

`first_date` First date ("YYYY-MM-DD") to `yc_mget` multiple curves

last_date	Last date ("YYYY-MM-DD") to yc_mget multiple curves
by	Number of days in between fetched dates (default = 1) in yc_mget
cache_folder	Location of cache folder (default = cachedir())
do_cache	Whether to use cache or not (default = TRUE)
refdate	Specific date ("YYYY-MM-DD") to yc_get single curve

## Details

See [https://www.b3.com.br/data/files/8B/F5/11/68/5391F61043E561F6AC094EA8/Manual\\_de\\_Curvas.pdf](https://www.b3.com.br/data/files/8B/F5/11/68/5391F61043E561F6AC094EA8/Manual_de_Curvas.pdf) for more details.

yc\_get returns the yield curve for the given date and yc\_mget returns multiple yield curves for a given range of dates.

yc\_ipca\_get returns the yield curve of real interest rates for the given date and yc\_ipca\_mget returns multiple yield curves of real interest rates for a given range of dates. These real interest rates consider IPCA as its inflation index.

yc\_usd\_get returns the yield curve of nominal interest rates for USD in Brazil for the given date and yc\_usd\_mget returns multiple yield curves of nominal interest rates for USD in Brazil for a given range of dates. These real interest rates consider IPCA as its inflation index.

## Value

A dataframe/tibble with yield curve data

## Examples

```
## Not run:
df_yc <- yc_mget(first_date = Sys.Date() - 5, last_date = Sys.Date())
head(df_yc)

## End(Not run)
## Not run:
df_yc <- yc_get(Sys.Date())
head(df_yc)

## End(Not run)
## Not run:
df_yc_ipca <- yc_ipca_mget(
  first_date = Sys.Date() - 5,
  last_date = Sys.Date()
)
head(df_yc_ipca)

## End(Not run)
## Not run:
df_yc_ipca <- yc_ipca_get(Sys.Date())
head(df_yc_ipca)

## End(Not run)
## Not run:
```

```
df_yc_usd <- yc_usd_mget(  
  first_date = Sys.Date() - 5,  
  last_date = Sys.Date()  
)  
head(df_yc_usd)  
  
## End(Not run)  
## Not run:  
df_yc_usd <- yc_usd_get(Sys.Date())  
head(df_yc_usd)  
  
## End(Not run)
```

---

yc\_superset

*Creates superset with yield curves and futures*

---

### Description

Creates superset with yield curves and future contracts indicating the terms that match with futures contracts maturities.

### Usage

```
yc_superset(yc, fut)  
  
yc_usd_superset(yc, fut)  
  
yc_ipca_superset(yc, fut)
```

### Arguments

yc	yield curve dataset
fut	futures dataset

### Value

A dataframe with yield curve flagged with futures maturities.

### Examples

```
## Not run:  
fut <- futures_get(Sys.Date() - 1)  
  
yc <- yc_get(Sys.Date() - 1)  
yc_superset(yc, fut)  
  
yc_usd <- yc_usd_get(Sys.Date() - 1)  
yc_usd_superset(yc_usd, fut)
```



```
yc_ipca <- yc_ipca_get(Sys.Date() - 1)
yc_ipca_superset(yc_ipca, fut)

## End(Not run)
```

# Index

cachedir, 2  
clearcache, 3  
code2month, 3  
company\_cash\_dividends\_get, 4  
company\_info\_get, 5  
company\_stock\_dividends\_get, 5  
company\_subscriptions\_get, 6  
convert\_to, 7  
cotahist-extracts, 8  
cotahist-options-superset, 10  
cotahist\_bdrs\_get (cotahist-extracts), 8  
cotahist\_equity\_get  
    (cotahist-extracts), 8  
cotahist\_equity\_options\_get  
    (cotahist-extracts), 8  
cotahist\_equity\_options\_superset  
    (cotahist-options-superset), 10  
cotahist\_etfs\_get (cotahist-extracts), 8  
cotahist\_fiagros\_get  
    (cotahist-extracts), 8  
cotahist\_fidcs\_get (cotahist-extracts),  
    8  
cotahist\_fiis\_get (cotahist-extracts), 8  
cotahist\_funds\_options\_get  
    (cotahist-extracts), 8  
cotahist\_get, 11  
cotahist\_get\_symbols  
    (cotahist-extracts), 8  
cotahist\_index\_options\_get  
    (cotahist-extracts), 8  
cotahist\_indexes\_get  
    (cotahist-extracts), 8  
cotahist\_options\_by\_symbol\_superset  
    (cotahist-options-superset), 10  
cotahist\_units\_get (cotahist-extracts),  
    8  
  
display\_template, 12  
download\_marketdata, 12  
  
futures\_get, 13  
futures\_mget (futures\_get), 13  
  
index\_by\_segment\_get, 16  
index\_comp\_get, 17  
index\_get, 18  
index\_weights\_get, 19  
indexes\_get, 14  
indexes\_last\_update, 15  
indexreport\_get, 15  
indexreport\_mget (indexreport\_get), 15  
  
maturity2date, 19  
  
read\_marketdata, 20  
  
show\_templates, 21  
  
yc\_get, 22  
yc\_ipca\_get (yc\_get), 22  
yc\_ipca\_mget (yc\_get), 22  
yc\_ipca\_superset (yc\_superset), 24  
yc\_mget (yc\_get), 22  
yc\_superset, 24  
yc\_usd\_get (yc\_get), 22  
yc\_usd\_mget (yc\_get), 22  
yc\_usd\_superset (yc\_superset), 24