# Package 'RblDataLicense'

January 20, 2025

**Type** Package

**Title** R Interface to 'Bloomberg Data License'

**Version** 0.2.6

**Description** R interface to access prices and market data with the
'Bloomberg Data License' service from
<https://www.bloomberg.com/professional/product/data-license/>.
As a prerequisite, a valid Data License from 'Bloomberg' is needed
together with the corresponding SFTP credentials and whitelisting
of the IP from which accessing the service.
This software and its author are in no way affiliated,
endorsed, or approved by 'Bloomberg' or any of its affiliates.
'Bloomberg' is a registered trademark.

**License** GPL-3

**URL** <https://rbldatalicense.eguidotti.com>

**BugReports** <https://github.com/eguidotti/RblDataLicense/issues>

**Depends** RCurl, xts, methods

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**RoxygenNote** 7.2.3

**NeedsCompilation** no

**Author** Emanuele Guidotti [aut, cre] (<https://orcid.org/0000-0002-8961-6623>)

**Maintainer** Emanuele Guidotti <emanuele.guidotti@usi.ch>

**Repository** CRAN

**Date/Publication** 2024-08-22 07:20:02 UTC

# Contents

---

RblConnect                          *Create an SFTP connection to Bloomberg Data License*

---

**Description**

An sftp connection to Bloomberg Datalicense is established. On some Linux systems, this may not work out of the box, as libcurl does not natively support sftp. In that case, you need to compile curl with SFTP support. See here for details: http://askubuntu.com/questions/195545/how-to-enable-sftp-support-in-curl

**Usage**

```
RblConnect(
  user,
  pw,
  host = "sftp.bloomberg.com",
  port = "22",
  protocol = "sftp",
  verbose = TRUE
)
```

**Arguments**

| | |
|---|---|
| user | The account number assigned by Bloomberg |
| pw | The password assigned by Bloomberg |
| host | The connection host |
| port | The connection port |
| protocol | The connection protocol |
| verbose | logical. Should print extra information on failure? |

**Value**

logical. Is the connection succesful?

## Examples

```
## Not run:
# These are dummy credentials. Replace with the credentials received from Bloomberg
RblConnect(user = 'dl000000', pw = '0000000000000000')

## End(Not run)
```

---

RblDownload                    *Download a file from Bloomberg*

---

## Description

Download a generic file from Bloomberg. Use [RblFiles](#) to list the files available at Bloomberg. If the file is not available yet (i.e. a request file has just been uploaded), the function waits until the response file is there or the timeout is reached

## Usage

```
RblDownload(file, frequency = 60, timeout = 3600, verbose = TRUE)
```

## Arguments

| | |
|---|---|
| file | character string representing the file to download |
| frequency | the polling frequency to check if file is available at Bloomberg |
| timeout | the timeout in seconds |
| verbose | logical. Should R report extra information on progress? |

## Value

character string. Path to the downloaded file. NULL on failure

## Examples

```
## Not run:
# Run RblConnect first

# Build a request file to download the daily closing prices of
#  EURO STOXX Index from 2005-01-01 to 2015-12-31.
RblRequest <-
    RblRequestBuilder(
        header = c(FIRMNAME = RblUser(),
                   PROGRAMNAME = 'gethistory',
                   DATERANGE = '20050101|20151231'),
        fields = c('PX_LAST'),
        identifiers = c('SXXE Index')
    )
```

```
# Upload the request file
req <- RblUpload(RblRequest)

# Download the response file
out <- RblDownload(req$out)
out

## End(Not run)
```

---

RblFiles                   *List files available at Bloomberg*

---

## Description

Retrieve files available at Bloomberg.

## Usage

```
RblFiles()
```

## Value

Vector of character strings representing the filenames available at Bloomberg

## Examples

```
## Not run:
# Run RblConnect first
RblFiles()

## End(Not run)
```

---

RblParse                   *Parse Bloomberg response file and import data*

---

## Description

Parse Bloomberg local response files and import the data in R. The PROGRAMNAME in use is
auto detected: 'getdata' and 'gethistory' are supported.

## Usage

```
RblParse(file, auto.assign = FALSE, env = parent.frame(), verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `file` | character string representing the local file to parse (see [`RblDownload`](#)) |
| `auto.assign` | logical. Should results be loaded to env when using PROGRAMNAME=gethistory? |
| `env` | where to create objects if auto.assign = TRUE |
| `verbose` | logical. Should R report extra information on progress? |

## Value

**PROGRAMNAME=getdata** data.frame containing identifiers (rows) and fields (columns). NULL on failure.

**PROGRAMNAME=gethistory** list of xts objects. If *auto.assign*=TRUE the xts objects are loaded in *env* and the object names are returned. NULL on failure.

## Examples

```
## Not run:
# Run RblConnect first

# Build a request file to download the daily closing prices of
#  EURO STOXX Index from 2005-01-01 to 2015-12-31.
RblRequest <-
    RblRequestBuilder(
        header = c(FIRMNAME = RblUser(),
                    PROGRAMNAME = 'gethistory',
                    DATERANGE = '20050101|20151231'),
        fields = c('PX_LAST'),
        identifiers = c('SXXE Index')
    )

# Upload the request file
req <- RblUpload(RblRequest)

# Download the response file
out <- RblDownload(req$out)

# Import the data
data <- RblParse(out)
str(data)

## End(Not run)
```

---

| | |
|---|---|
| RblQuery | *Query Bloomberg and Fetch Data* |

---

## Description

The function provides a high level interface to Bloomberg Datalicense 'getdata' and 'gethistory' programs.

**Usage**

```
RblQuery(
  identifiers,
  fields,
  from = NULL,
  to = Sys.Date(),
  auto.assign = FALSE,
  env = parent.frame(),
  category = c(),
  add_headers = c(),
  overrides = c(),
  limit = 5,
  split = 100,
  frequency = 60,
  timeout = 3600,
  filename = format(Sys.time(), "%m%d%H%M%S"),
  verbose = TRUE
)
```

**Arguments**

| | |
|---|---|
| identifiers | vector of Bloomberg identifiers. E.g. c('SXXE Index', 'SX5E Index') |
| fields | vector of Bloomberg fields. E.g. c('PX_LAST', 'PX_CLOSE', 'PX_OPEN', 'PX_HIGH', 'PX_LOW') |
| from | date or string (format YYYY-MM-DD). Start time for the 'gethistory' request. If not provided, a 'getdata' request will be made |
| to | date or string (format YYYY-MM-DD). End time for the 'gethistory' request. Ignored if *from* is not provided |
| auto.assign | logical. Should results be loaded to env? Ignored if *from* is not provided |
| env | where to create objects if auto.assign = TRUE |
| category | vector of Data License categories to enable. E.g. c('SECMASTER', 'PRIC-ING', 'FUNDAMENTALS'). WARNING! Each DL category is billed separately, so check your DL license carefully! |
| add_headers | named vector of additional headers. E.g. c(PROGRAMFLAG = 'oneshot') |
| overrides | named vector of Bloomberg overrides. E.g. c('END_DT' = '20100101') |
| limit | prevent requesting data for more than this amout of identifiers. This is done to help you keeping your budget under control. Data License is billing based on the amout of instruments you request, so check your DL license carefully before increasing this limit. |
| split | maximum number of identifiers to process at once. Requests are split to avoid memory leaks. |
| frequency | the polling frequency to check if the response file is available at Bloomberg |
| timeout | the timeout in seconds |
| filename | name assigned to the remote file. Only alphanumeric characters are allowed. Invalid characters are removed. |
| verbose | logical. Should R report extra information on progress? |

## Details

The following routine to query Bloomberg is implemented:

**Build the request file** see [RblRequestBuilder](#)
**Upload the request file** see [RblUpload](#)
**Download the response file** see [RblDownload](#)
**Parse the response file** see [RblParse](#)

## Value

A list with components

**req** List of characters representing each of the request files uploaded to Bloomberg
**out** List of characters representing each of the response file downloaded from Bloomberg
**data** Return of [RblParse](#)

## Examples

```
## Not run:
# Run RblConnect first
data <- RblQuery(fields = c('PX_LAST', 'PX_OPEN', 'PX_HIGH', 'PX_LOW'),
                 identifiers = c('SXXE Index', "SX5E Index"),
                 from = '2005-01-01')
str(data)

## End(Not run)
```

---

RblRequestBuilder            *Build a request file to query Bloomberg*

---

## Description

The request file is generated according to Bloomberg Data License Documentation. Refer to the Documentation to build the request.

## Usage

```
RblRequestBuilder(header, fields, identifiers, overrides = c())
```

## Arguments

| | |
|---|---|
| header | named vector of headers. E.g. c(FIRMNAME = RblUser(), PROGRAMNAME = 'getdata') |
| fields | vector of Bloomberg fields. E.g. c('PX_LAST', 'PX_OPEN', 'PX_HIGH', 'PX_LOW') |
| identifiers | vector of Bloomberg identifiers. E.g. c('SXXE Index', 'SX5E Index') |
| overrides | named vector of Bloomberg overrides. E.g. c('END_DT' = '20100101') |

## Value

character string representing the request file. Upload it to query Bloomberg (see `RblUpload`)

## Examples

```
## Not run:
# Run RblConnect first

# Build a request file to download the daily closing prices of
#  EURO STOXX Index from 2005-01-01 to 2015-12-31.
RblRequest <-
    RblRequestBuilder(
        header = c(FIRMNAME = RblUser(),
                    PROGRAMNAME = 'gethistory',
                    DATERANGE = '20050101|20151231'),
        fields = c('PX_LAST'),
        identifiers = c('SXXE Index')
    )
RblRequest

## End(Not run)
```

---

RblUpload                          *Upload a request file to Bloomberg*

---

## Description

Upload a request file to query Bloomberg. A response file will be generated by Bloomberg. The request file can be user-defined following the Bloomberg Data License documentation or generated with the `RblRequestBuilder` function. The response file needs to be downloaded (see `RblDownload`) and parsed (see `RblParse`) to import the data in R.

## Usage

```
RblUpload(
  RblRequest,
  filename = format(Sys.time(), "%m%d%H%M%S"),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| RblRequest | character string representing the request file according to Bloomberg Datalicense documentation. Can be generated with the `RblRequestBuilder` function |
| filename | name assigned to the remote file. Only alphanumeric characters and underscores are allowed. Invalid characters are removed. |
| verbose | logical. Should R report extra information on progress? |

## Value

A list with components

**req** the request filename

**out** the response filename

## Examples

```
## Not run:
# Run RblConnect first

# Build a request file to download the daily closing prices of
#  EURO STOXX Index from 2005-01-01 to 2015-12-31.
RblRequest <-
    RblRequestBuilder(
        header = c(FIRMNAME = RblUser(),
                   PROGRAMNAME = 'gethistory',
                   DATERANGE = '20050101|20151231'),
        fields = c('PX_LAST'),
        identifiers = c('SXXE Index')
    )

# Upload the request file
req <- RblUpload(RblRequest)
req

## End(Not run)
```

---

RblUrl                          *Get Bloomberg connection Url*

---

## Description

Retrieve the Bloomberg connection Url.

## Usage

```
RblUrl()
```

## Value

Url string

**Examples**

```
## Not run:
# Run RblConnect first
RblUrl()

## End(Not run)
```

---

RblUser                         *Get Bloomberg User*

---

**Description**

Retrieve the Bloomberg User.

**Usage**

```
RblUser()
```

**Value**

User string

**Examples**

```
## Not run:
# Run RblConnect first
RblUser()

## End(Not run)
```

# Index