# Package 'rtsdata'

October 14, 2022

**Type** Package

**Title** R Time Series Intelligent Data Storage

**Version** 0.1.3

**Description** A tool that allows to download and save historical time series data
for future use offline. The intelligent updating functionality will only download
the new available information; thus, saving you time and Internet bandwidth.
It will only re-download the full data-set if any inconsistencies are detected.
This package supports following data provides: 'Yahoo' (<https://finance.yahoo.com>),
'FRED' (<https://fred.stlouisfed.org>), 'Quandl' (<https://data.nasdaq.com>),
'AlphaVantage' (<https://www.alphavantage.co>), 'Tiingo' (<https://www.tiingo.com>).

**License** MIT + file LICENSE

**Depends** xts

**Imports** quantmod, zoo, Quandl, anytime, data.table, mongolite, brotli,
curl

**Suggests** RQuantLib

**URL** https://bitbucket.org/rtsvizteam/rtsdata

**BugReports** https://bitbucket.org/rtsvizteam/rtsdata/issues

**LazyLoad** yes

**RoxygenNote** 7.1.2

**NeedsCompilation** no

**Author** RTSVizTeam [aut, cph],
Irina Kapler [cre]

**Maintainer** Irina Kapler <irkapler@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-10-29 06:30:02 UTC

## R topics documented:

---

`ds.default.location` *Default location to save data*

---

## Description

The following waterfall is used to lookup the default location: 1. options 2. environment 3. default option

Good practice is not to store this setting inside the script files. Add options(RTSDATA_DB='mongodb://localhost') line to the .Rprofile to use 'mongodb://localhost' database.

## Usage

```
ds.default.location()

ds.default.database()
```

## Details

Good practice is not to store this setting inside the script files. Add options(RTSDATA_FOLDER='C:/Data') line to the .Rprofile to use 'C:/Data' folder.

## Value

default location to save data

default database to save data

## Examples

```
# Default location to save data
ds.default.location()
```

## ds.functionality.default *Default Functionality*

### Description

Default functionality configuration

### Usage

```
ds.functionality.default(
  check.update = TRUE,
  update.required.fn = update.required
)
```

### Arguments

| | |
|---|---|
| check.update | flag to check for updates, **defaults to TRUE** |
| update.required.fn | function to check if update is required given stored historical data, **defaults to update.required**. The update.required function takes last update stamp, current date/time, holiday calendar name. |

### Value

list with options

### Examples

```
# disable check for updates for the 'yahoo' data source
register.data.source(src = 'yahoo', functionality = ds.functionality.default(FALSE))
```

## ds.get.url *Load data from URL*

### Description

Load data from URL

### Usage

```
ds.get.url(
  url,
  h = curl::new_handle(),
  useragent = "Mozilla/5.0 (Windows NT 6.1; Win64; rv:62.0) Gecko/20100101",
  referer = NULL
)
```

### Arguments

| | |
|---|---|
| url | url |
| h | curl handle |
| useragent | user agent |
| referer | referer |

### Examples

```
ds.get.url('https://finance.yahoo.com/')
```

---

## ds.getSymbol.yahoo *Get quotes from Yahoo Finance*

---

### Description

Download historical data from Yahoo Finance using ‘getSymbols.yahoo‘ function from ‘quantmod‘ package.

Download historical data from FRED using ‘get_fred_series‘ function from ‘alfred‘ package.

Download historical data from Quandl using ‘Quandl‘ function from ‘Quandl‘ package.

Download historical data from AlphaVantage using ‘getSymbols.av‘ function from ‘quantmod‘ package.

Download historical data from Tiingo using ‘getSymbols.tiingo‘ function from ‘quantmod‘ package.

Generate fake stock data for use in rtsdata examples

### Usage

```
ds.getSymbol.yahoo(Symbol, from = "1900-01-01", to = Sys.Date())

ds.getSymbol.FRED(Symbol, from = "1900-01-01", to = Sys.Date())

ds.getSymbol.Quandl(Symbol, from = "1900-01-01", to = Sys.Date())

ds.getSymbol.av(Symbol, from = "1900-01-01", to = Sys.Date())

ds.getSymbol.tiingo(Symbol, from = "1900-01-01", to = Sys.Date())

ds.getSymbol.fake.stock.data(Symbol, from = "1900-01-01", to = Sys.Date())
```

### Arguments

| | |
|---|---|
| Symbol | symbol |
| from | start date, expected in yyyy-mm-dd format, **defaults to 1900-01-01** |
| to | end date, expected in yyyy-mm-dd format, **defaults to today's date** |

### Details

Quandl recommends getting an API key Add following code options(Quandl.api_key = api_key) to your .Rprofile file

You need an API key from www.alphavantage.co Add following code options(getSymbols.av.Default = api_key) to your .Rprofile file

You need an API key from api.tiingo.com Add following code options(getSymbols.av.Default = api_key) to your .Rprofile file

### Value

xts object with data

### Examples

```
# get sample of the fake stock data
ds.getSymbol.fake.stock.data('dummy', from = '2018-02-01', to = '2018-02-13')
```

---

## ds.load.csv *Read csv*

---

### Description

Read csv

### Usage

```
ds.load.csv(filename, sep = ",", ...)
```

### Arguments

| | |
|---|---|
| filename | CSV filename |
| sep | delimiter |
| ... | other parameters |

### Examples

```
# generate csv file
filename = file.path(tempdir(), 'dummy.csv')
cat('x1,x2,x3\n1,2,3\n', file = filename)
ds.load.csv(filename)
```

## ds.storage.database *MongoDB GridFS Storage model*

### Description

MongoDB GridFS Storage model

### Usage

```
ds.storage.database(url = ds.default.database(), db = "data_storage")
```

### Arguments

| | |
|---|---|
| url | address of the mongodb server in mongo connection string URI format, **defaults to ds.default.database database**.<br>For local mongodb server, use 'mongodb://localhost' URI. For local authenticated mongodb server, use 'mongodb://user:password@localhost' URI. |
| db | name of database, **defaults to 'data_storage'** |

### Value

list with storage options

### Examples

```
# change the 'yahoo' data source to use MongoDB to store historical data
# register.data.source(src = 'yahoo', storage = ds.storage.database())
```

## ds.storage.file.csv *CSV file Storage model*

### Description

CSV file Storage model

### Usage

```
ds.storage.file.csv(
  location = ds.default.location(),
  extension = "csv",
  date.format = "%Y-%m-%d",
  custom.folder = FALSE
)
```

### Arguments

| | |
|---|---|
| location | storage location, **defaults to ds.default.location folder** |
| extension | file extension, **defaults to 'csv'** |
| date.format | date format, **defaults to "%Y-%m-%d"** use **"%Y-%m-%d %H:%M:%S"** for storing intra day data |
| custom.folder | custom folder flag, **defaults to False** if flag is False **default**, the data is stored at the "location\src_extnsion" folder. if flag is True, the data is stored at the **location** folder. |

### Value

list with storage options

### Examples

```
# change the 'yahoo' data source to use CSV files to store historical data
register.data.source(src = 'yahoo', storage = ds.storage.file.csv())
```

---

## ds.storage.file.csv.load

*Load data from CSV file into 'xts' object*

---

### Description

Load data from CSV file into 'xts' object

### Usage

```
ds.storage.file.csv.load(file, date.col = NULL, date.format = "%Y-%m-%d")
```

### Arguments

| | |
|---|---|
| file | CSV file |
| date.col | date column |
| date.format | date format |

### Value

xts object with loaded data

### Examples

```
# get sample of the fake stock data
data = ds.getSymbol.fake.stock.data('dummy', from = '2018-02-01', to = '2018-02-13')
filename = file.path(tempdir(), 'dummy.csv')
ds.storage.file.csv.save(data, filename)
ds.storage.file.csv.load(filename)
```

---

`ds.storage.file.csv.save`

*Save 'xts' object into CSV file*

---

## Description

Save 'xts' object into CSV file

## Usage

```
ds.storage.file.csv.save(ds.data, file, date.format = "%Y-%m-%d")
```

## Arguments

| | |
|---|---|
| `ds.data` | 'xts' object |
| `file` | filename to save 'xts' object |
| `date.format` | date format |

## Value

nothing

## Examples

```
# get sample of the fake stock data
data = ds.getSymbol.fake.stock.data('dummy', from = '2018-02-01', to = '2018-02-13')
filename = file.path(tempdir(), 'dummy.csv')
ds.storage.file.csv.save(data, filename)
```

---

`ds.storage.file.exists`

*Check if file exists with historical data for given ticker*

---

## Description

Check if file exists with historical data for given ticker

## Usage

```
ds.storage.file.exists(t, s)
```

## Arguments

| | |
|---|---|
| `t` | ticker |
| `s` | storage model |

### Value

boolean indicating if file exists with historical data for given ticker

### Examples

```
ds.storage.file.exists('dummy', ds.storage.file.rdata())
```

---

## ds.storage.file.rdata *Rdata file Storage model*

---

### Description

Rdata file Storage model

### Usage

```
ds.storage.file.rdata(
  location = ds.default.location(),
  extension = "Rdata",
  custom.folder = FALSE
)
```

### Arguments

| | |
|---|---|
| location | storage location, **defaults to ds.default.location folder** |
| extension | file extension, **defaults to 'Rdata'** |
| custom.folder | custom folder flag, **defaults to False** if flag is False **default**, the data is stored at the "location\src_extnsion" folder. if flag is True, the data is stored at the **location** folder. |

### Value

list with storage options

### Examples

```
# change the 'yahoo' data source to use Rdata files to store historical data
register.data.source(src = 'yahoo', storage = ds.storage.file.rdata())
```

## ds.storage.file.ticker *File with historical data for given ticker*

### Description

File with historical data for given ticker

### Usage

```
ds.storage.file.ticker(t, s)
```

### Arguments

| | |
|---|---|
| t | ticker |
| s | storage model |

### Value

filename with historical data for given ticker

### Examples

```
ds.storage.file.ticker('dummy', ds.storage.file.rdata())
```

## getSymbols *Download historical data*

### Description

Overwrite the getSymbols function from 'quantmod' package to efficiently load historical data

### Usage

```
getSymbols(
  Symbols = NULL,
  env = parent.frame(),
  reload.Symbols = FALSE,
  verbose = FALSE,
  warnings = TRUE,
  src = "yahoo",
  symbol.lookup = TRUE,
  auto.assign = TRUE,
  from = "1990-01-01",
  to = Sys.time(),
```

```
  calendar = NULL,
  check.update = NULL,
  full.update = NULL
)
```

## Arguments

| | |
|---|---|
| Symbols | list symbols to download historical data |
| env | environment to hold historical data, **defaults to parent.frame()** |
| reload.Symbols | flag, not used, inherited from the getSymbols function from 'quantmod' package, **defaults to FALSE** |
| verbose | flag, inherited from the getSymbols function from 'quantmod' package, **defaults to FALSE** |
| warnings | flag, not used, inherited from the getSymbols function from 'quantmod' package, **defaults to TRUE** |
| src | source of historical data, **defaults to 'yahoo'** |
| symbol.lookup | flag, not used, inherited from the getSymbols function from 'quantmod' package, **defaults to TRUE** |
| auto.assign | flag to store data in the given environment, **defaults to TRUE** |
| from | start date, expected in yyyy-mm-dd format, **defaults to 1900-01-01** |
| to | end date, expected in yyyy-mm-dd format, **defaults to today's date** |
| calendar | RQuantLib's holiday calendar, for example: calendar = 'UnitedStates/NYSE', **defaults to NULL** |
| check.update | flag to check for updates, **defaults to NULL** |
| full.update | flag to force full update, **defaults to NULL** |

## Value

xts object with data

## Examples

```
 # small toy example

 # register data source to generate fake stock data for use in rtsdata examples
 register.data.source(src = 'sample', data = ds.getSymbol.fake.stock.data)

 # Full Update till '2018-02-13'
 data = getSymbols('test', src = 'sample', from = '2018-01-01', to = '2018-02-13',
auto.assign=FALSE, verbose=TRUE)

 # No updated needed, data is loaded from file
 data = getSymbols('test', src = 'sample', from = '2018-01-01', to = '2018-02-13',
auto.assign=FALSE, verbose=TRUE)

 # Incremental update from '2018-02-13' till today
 data = getSymbols('test', src = 'sample', from = '2018-01-01',
```

```
auto.assign=FALSE, verbose=TRUE)

 # No updated needed, data is loaded from file
 data = getSymbols('test', src = 'sample', from = '2018-01-01',
auto.assign=FALSE, verbose=TRUE)

# data is stored in the 'sample_Rdata' folder at the following location
ds.default.location()


 ds.getSymbol.yahoo('AAPL',from='2018-02-13')
```

---

`register.data.source` *Data Sources*

---

## Description

List available data sources and Register new ones

## Usage

```
register.data.source(
  src = "yahoo",
  data = ds.getSymbol.yahoo,
  storage = ds.storage.file.rdata(),
  functionality = ds.functionality.default(),
  overwrite = TRUE
)

data.sources()
```

## Arguments

| | |
|---|---|
| `src` | data source name, **defaults to 'yahoo'** |
| `data` | data source to download historical data, function must take Symbol, from, to parameters, **defaults to ds.getSymbol.yahoo** |
| `storage` | storage model configuration, **defaults to ds.storage.file.rdata(src)** |
| `functionality` | functionality configuration, **defaults to ds.functionality.default()** |
| `overwrite` | flag to overwrite data source if already registered in the list of plugins, **defaults to True** |

## Value

None

## Examples

```
# register data source to generate fake stock data for use in rtsdata examples
register.data.source(src = 'sample', data = ds.getSymbol.fake.stock.data)

# print allregistered data sources
names(data.sources())
```

---

rtsdata *'rtsdata' - Efficient Data Storage system for R Time Series.*

---

## Description

The 'rtsdata' package simplifies the management of Time Series in R. This package overwrites the 'getSymbols' function from 'quantmod' package to allow for minimal changes to get started. The 'rtsdata' package provides functionality to **download** and **store** historical time series.

The **download** functionality will intelligently update historical data as needed. The incremental data is downloaded first to updated historical data. The full history is **only** downloaded if incremental data is not consistent. I.e. the last saved record is different from the first downloaded record.

The following download plugins are currently available: * Yahoo Finance - based on 'quantmod' package. * FRED - based on 'quantmod' package. * Quandl - based on 'Quandl' package. Quandl recommends getting an API key. Add following code options(Quandl.api_key = api_key) to your .Rprofile file. * AlphaVantage(av) - based on 'quantmod' package. You need an API key from www.alphavantage.co. Add following code options(getSymbols.av.Default = api_key) to your .Rprofile file. * Tiingo - based on 'quantmod' package You need an API key from api.tiingo.com. Add following code options(getSymbols.av.Default = api_key) to your .Rprofile file.

The download functionality plugins are easily created. The user needs to provide a function to download historical data with ticker, start, and end dates parameters to create new download plugin.

The **storage** functionality provides a consistent interface to store historical time series. The following storage plugins are currently available: * Rdata - store historical time series data in the Rdata files. * CSV - store historical time series data in the CSV files. The CSV storage is not efficient because CSV files will have to be parsed every time the data is loaded. The advantage of this format is ease of access to the stored historical data by external programs. For example the CSV files can be opened in Notepad or Excel. * MongoDB - store historical time series data in the MongoDB GridFS system. The MongoDB storage provides optional authentication. The MongoDB storage functionality is currently only available in the development version at bitbucket.

The storage functionality plugins are easily created. The user needs to provide a functions to load and save data to create new storage plugin.

## Examples

```
# small toy example

# register data source to generate fake stock data for use in rtsdata examples
register.data.source(src = 'sample', data = ds.getSymbol.fake.stock.data)
```

```
 # Full Update till '2018-02-13'
 data = getSymbols('test', src = 'sample', from = '2018-01-01', to = '2018-02-13',
auto.assign=FALSE, verbose=TRUE)

 # No updated needed, data is loaded from file
 data = getSymbols('test', src = 'sample', from = '2018-01-01', to = '2018-02-13',
auto.assign=FALSE, verbose=TRUE)

 # Incremental update from '2018-02-13' till today
 data = getSymbols('test', src = 'sample', from = '2018-01-01',
auto.assign=FALSE, verbose=TRUE)

 # No updated needed, data is loaded from file
 data = getSymbols('test', src = 'sample', from = '2018-01-01',
auto.assign=FALSE, verbose=TRUE)

# data is stored in the 'sample_Rdata' folder at the following location
ds.default.location()
```

# Index