**Introduction**

In the [latest post](https://www.msperlin.com/blog/post/2019-10-16-new-package-getedgardata/) It was written about package GetEdgarData, which downloaded structured data from the SEC. I’ve been working on this project and soon realized that the available data at the SEC/DERA section is not complete. For example, all Q4 statements are missing. This seems to be the way all exchanges release the financial documents. I’ve found the same problem here in the Brazilian exchange.

It came to my attention that there is an alternative way of fetching corporate data and adjusted prices, the [SimFin project](https://simfin.com/). From its own website:

Our core goal is to make financial data as freely available as possible because we believe that having the right tools for investing/research shouldn't be the privilege of those that can afford to spend thousands of dollars per year on data.

The platform is free with a daily limit of 2000 api calls. This is not bad and should suffice for most users. If you need more calls, the [premium version](https://simfin.com/simfin-plus) is just 10 euros a month, a fraction of what other data vendors usually request.

Package simfinR, available in [Github](https://github.com/msperlin/simfinR/) and soon in CRAN, facilitates all calls to the simfin API. It first makes sure the requested data exists and only then calls the api. As usual, all api queries are saved locally using package memoise. This means that the second time you ask for a particular data about a company/year, the function will load a local copy, and will not call the web api.

Package GetEdgarData, however, will be discontinued. I’ll keep the files in [Github](https://github.com/msperlin/GetEdgarData/) but will no longer develop it. It takes a lot of time to write and maintain R packages, and I fell that simfinR has far more potential.

As mentioned before, both new packages, [GetQuandlData](https://www.msperlin.com/blog/post/2019-10-01-new-package-getquandldata/) and simfinR will be part of my next book, “Analyzing Financial and Economic Data with R”, which should be released in early 2020.

**Installation**

# not in CRAN yet (need to test it further)

#install.packages('simfinR')

# from Github

devtools::install\_github('msperlin/simfinR')

**Example 01 – Apples Quarterly Net Profit**

The first step in using simfinR is finding information about available companies:

library(simfinR)

library(tidyverse)

# You need to get your own api key at <https://simfin.com/>

my\_apy\_key <- readLines('~/Dropbox/.api\_key\_simfin.txt')

# get info

df\_info\_companies <- simfinR\_get\_available\_companies(my\_apy\_key)

# check it

glimpse(df\_info\_companies)

## Observations: 2,564

## Variables: 3

## $ simId 171401, 901704, 901866, 45730, 378251, 896477, 418866, 79…

## $ ticker "ZYXI", "ZYNE", "ZVO", "ZUMZ", "ZTS", "ZS", "ZNGA", "ZIOP…

## $ name "ZYNEX INC", "Zynerba Pharmaceuticals, Inc.", "Zovio Inc"…

We find information about 2564 companies. Digging deeper we find that the simfin id of Apple is 111052. Let’s use it to download the annual financial information since 2009.

id\_companies <- 111052 # id of APPLE INC

type\_statements <- 'pl' # profit/loss

periods = 'FY' # final year

years = 2009:2018

df\_fin\_FY <- simfinR\_get\_fin\_statements(id\_companies,

type\_statements = type\_statements,

periods = periods,

year = years,

api\_key = my\_apy\_key)

glimpse(df\_fin\_FY)

## Observations: 580

## Variables: 13

## $ company\_name "APPLE INC", "APPLE INC", "APPLE INC", "APPLE INC…

## $ company\_sector "Computer Hardware", "Computer Hardware", "Comput…

## $ type\_statement pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, p…

## $ period FY, FY, FY, FY, FY, FY, FY, FY, FY, FY, FY, FY, F…

## $ year 2009, 2009, 2009, 2009, 2009, 2009, 2009, 2009, 2…

## $ ref\_date 2009-12-31, 2009-12-31, 2009-12-31, 2009-12-31, …

## $ acc\_name "Revenue", "Sales & Services Revenue", "Financing…

## $ acc\_value 4.2905e+10, NA, NA, NA, -2.5683e+10, NA, NA, NA, …

## $ tid "1", "3", "5", "6", "2", "7", "8", "9", "4", "10"…

## $ uid "1", "0", "0", "0", "2", "0", "0", "0", "4", "10"…

## $ parent\_tid "4", "1", "1", "1", "4", "2", "2", "2", "19", "19…

## $ display\_level "0", "1", "1", "1", "0", "1", "1", "1", "0", "0",…

## $ check\_possible FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, …

And now we plot the results of the “Net Income” (profit/loss) for all years:

net\_income <- df\_fin\_FY %>%

filter(acc\_name == 'Net Income')

p <- ggplot(net\_income,

aes(x = ref\_date, y = acc\_value)) +

geom\_col() +

labs(title = 'Yearly Profit of APPLE INC',

x = '',

y = 'Yearly Profit',

subtitle = '',

caption = 'Data from simfin ') +

theme\_bw()

print(p)

Not bad!

We can also grab data for all quarters:

type\_statements <- 'pl' # profit/loss

periods = c('Q1', 'Q2', 'Q3', 'Q4') # final year

years = 2009:2018

df\_fin\_quarters <- simfinR\_get\_fin\_statements(id\_companies,

type\_statements = type\_statements,

periods = periods,

year = years,

api\_key = my\_apy\_key)

glimpse(df\_fin\_quarters)

## Observations: 2,320

## Variables: 13

## $ company\_name "APPLE INC", "APPLE INC", "APPLE INC", "APPLE INC…

## $ company\_sector "Computer Hardware", "Computer Hardware", "Comput…

## $ type\_statement pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, pl, p…

## $ period Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q…

## $ year 2009, 2009, 2009, 2009, 2009, 2009, 2009, 2009, 2…

## $ ref\_date 2009-03-31, 2009-03-31, 2009-03-31, 2009-03-31, …

## $ acc\_name "Revenue", "Sales & Services Revenue", "Financing…

## $ acc\_value 1.188e+10, NA, NA, NA, -7.373e+09, NA, NA, NA, 4.…

## $ tid "1", "3", "5", "6", "2", "7", "8", "9", "4", "10"…

## $ uid "1", "0", "0", "0", "2", "0", "0", "0", "4", "10"…

## $ parent\_tid "4", "1", "1", "1", "4", "2", "2", "2", "19", "19…

## $ display\_level "0", "1", "1", "1", "0", "1", "1", "1", "0", "0",…

## $ check\_possible FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, …

And plot the results:

net\_income <- df\_fin\_quarters %>%

filter(acc\_name == 'Net Income')

p <- ggplot(net\_income,

aes(x = period, y = acc\_value)) +

geom\_col() + facet\_grid(~year, scales = 'free') +

labs(title = 'Quarterly Profit of APPLE INC',

x = 'Quarters',

y = 'Net Profit') +

theme\_bw()

print(p)

Nice and impressive profit record. The first quarter (Q1) seems to present the best performance, probably due to end of year holidays.

**Example 02 – Quarterly Net Profit of Many Companies**

Package simfinR can also fetch information for many companies in a single call. Let’s run another example by selecting four random companies and creating the same previous graph:

set.seed(5)

my\_ids <- sample(df\_info\_companies$simId, 4)

type\_statements <- 'pl' # profit/loss

periods = 'FY' # final year

years = 2010:2018

df\_fin <- simfinR\_get\_fin\_statements(id\_companies = my\_ids,

type\_statements = type\_statements,

periods = periods,

year = years,

api\_key = my\_apy\_key)

net\_income <- df\_fin %>%

filter(acc\_name == 'Net Income')

p <- ggplot(net\_income,

aes(x = ref\_date, y = acc\_value)) +

geom\_col() +

labs(title = 'Annual Profit/Loss of Four Companies',

x = '',

y = 'Net Profit/Loss') +

facet\_wrap(~company\_name, scales = 'free\_y') +

theme\_bw()

print(p)

**Example 03: Fetching price data**

The simfin project also provides adjusted prices of stocks. Have a look:

set.seed(5)

my\_ids <- sample(df\_info\_companies$simId, 4)

type\_statements <- 'pl' # profit/loss

periods = 'FY' # final year

years = 2009:2018

df\_price <- simfinR\_get\_price\_data(id\_companies = my\_ids,

api\_key = my\_apy\_key)

p <- ggplot(df\_price,

aes(x = ref\_date, y = close\_adj)) +

geom\_line() +

labs(title = 'Adjusted stock prices for four companies',

x = '',

y = 'Adjusted Stock Prices') +

facet\_wrap(~company\_name, scales = 'free\_y') +

theme\_bw()

print(p)

As you can see, the data is comprehensive and should suffice for many different corporate finance research topics.