How to Get Data: Web Scraping Using R

Data is of high importance, yes. But where to get this data from? One of the most convenient sources is… Of course, the internet!

Web scraping allows data geeks to acquire easy-to-access and real-time data. And thanks to highly tailored technologies it is so convenient and fast.

I share how it is easy using R’s ‘’rvest’’ library. With just a few lines of code, we are able to gather the data, format it, and save it into a CSV file.

Let’s get to work!

**Libraries**

First, we need the required libraries:

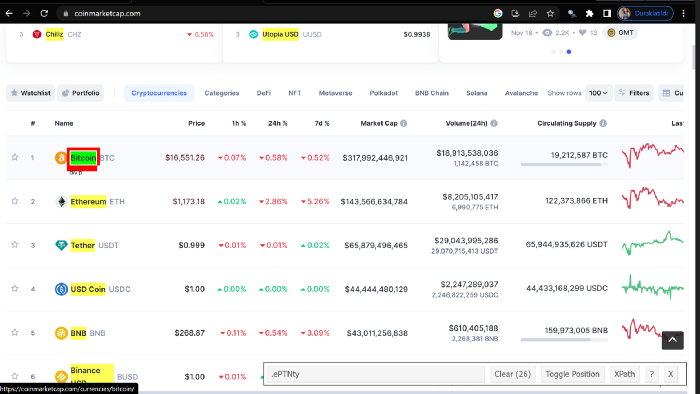
* *rvest*
* *data.table*
* *readr*

To scrape the webpage, we use the rvest library. And the data.table is required to create a data frame out of scraped values. Lastly, we will use readr to save our data in CSV format.

rm(list = ls())  
library(rvest)  
library(data.table)  
library(readr)

After importing all the required libraries, we can start scraping!

I chose [https://coinmarketcap.com](https://coinmarketcap.com/). To detect the elements easily, I use the SelectorGadget extension of Chrome. Give it a try :)



**Creating The Scraper Function**

After detecting all the elements that I want to see in my data frame, I created the function.

url <- ‘https://coinmarketcap.com'

getPage <- function(url) {  
 page <- read\_html(url)  
   
 coin <- page %>% html\_nodes(‘.cmc-link .ePTNty’) %>% html\_text()  
   
 price <- substring(page %>% html\_nodes(‘.hEduBL a’) %>% html\_text(),2)  
   
 capitalization <- substring(page %>% html\_nodes(‘.dKgvPU’) %>% html\_text(),2)  
   
 volume <- substring(page %>% html\_nodes(‘.font\_weight\_500’) %>% html\_text(),2)  
   
 supply <- sub(‘ .\*’,’’,page %>% html\_nodes(‘.izpQLd’) %>% html\_text())  
   
 link <- paste0(url,page %>% html\_nodes(‘.LCOyB a’) %>% html\_attr(‘href’))  
   
 df <- data.frame(‘coin’=coin, ‘price’=price, ‘capitalization’=capitalization,  
 ‘volume’=volume, ‘supply’=supply,’link’=link)  
   
 return(df)  
}

1. We read the page using rvest’s read\_html function.
2. We set relevant class names as parameters to html\_nodes function. To be able to retrieve text, we pipe the node into the html\_text() function. In case you are retrieving a URL, use html\_attr(‘href’).
3. After the retrieval of the values, we can convert them into a data frameusing the data.frame function of data.table.
4. We are ready to return the data!

That simple. We can get a bunch of information with only a single function.

**Scrape Multiple Pages**

And…. If you want to get data from multiple pages, you can do that with a single function. Use lapply.

urls <- paste0(url,’/?page=’,1:10)

coins <- rbindlist(lapply(urls, getPage))

1. Create a list of URLs. You should check the URL structure of the webpage.
2. Apply the getPage function to each individual element using lapply, and convert the list into a data frame. (rbindlist is the magic function of data.table)

**Writing to a CSV**

Yes, we are ready. Let’s save our data into a CSV.

With readr’s write\_csv. You can save it very easily.

write\_csv(coins, ‘coins.csv’)

Thank you for reading. Happy scraping! :)