

# **Data Collecting**

## **– Web scraping from public websites**

Jung PARK, PhD  
Research Fellow in Data Science

Version 2018 Aug 09

- Web scraping from public websites
  - How to collect data from websites
  - 10K reports from SEC.gov
  - EU Open Data
  - Open data Swiss
- Collecting data using API
  - Tweeter
  - Facebook
- Database in IMD library
  - Bloomberg
  - Thomson One
  - Datastream
  - Factiva: news media
- IoT (Internet of Things)
  - Smart building
  - Wearable devices
  - Web traffic

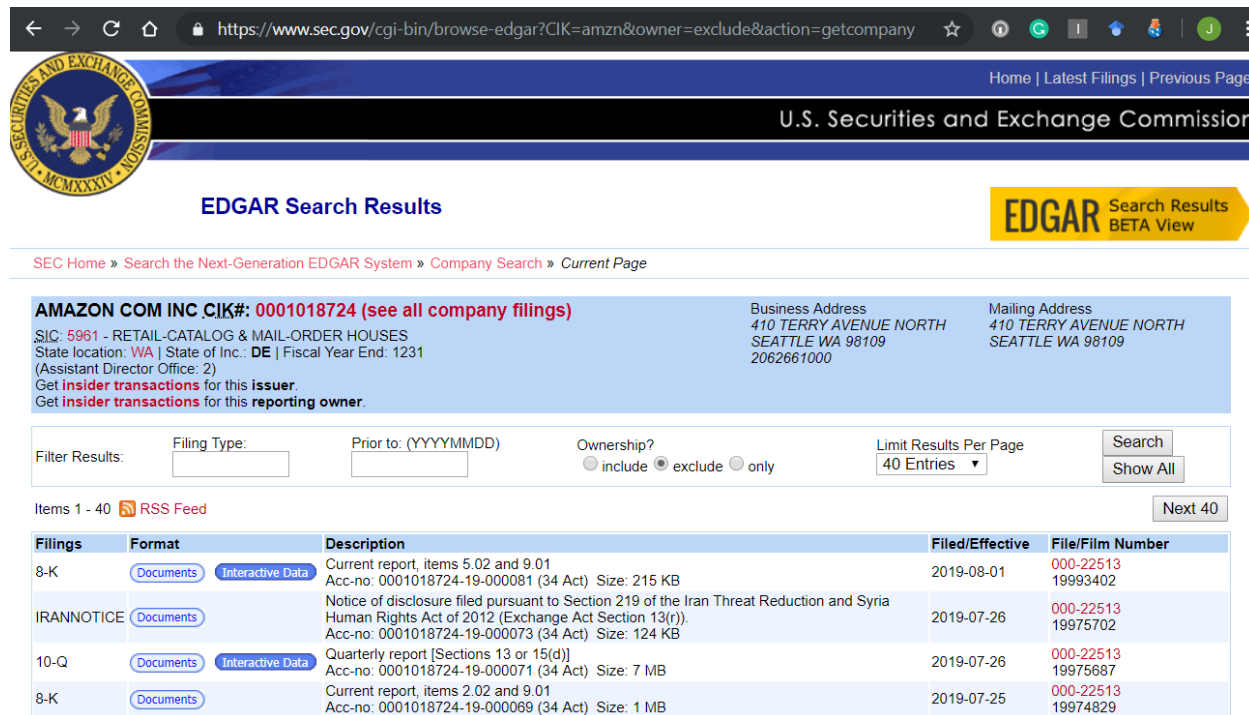
- Level 1. Manual collection
- Level 2. Using a code to download multiple files
- Level 3. Using API (Application Programming Interface)
- Level 4. Using a code for Web scraping

- Data from public websites
  - SEC.gov
  - EU Open Data
  - Open data Swiss
  - London Data Store
  - Wikipedia
  
- Data from commercial websites
  - Tweeter
  - Amazon

# Level 1: manual collection

## ex. Company filings from SEC.gov

- <https://www.sec.gov/edgar/searchedgar/companysearch.html>
- Mouse right-click and download individual files



The screenshot shows the SEC.gov Edgar Search Results page for Amazon.com Inc. The page header includes the SEC logo and the text "U.S. Securities and Exchange Commission". The search results are for "AMAZON COM INC" with CIK# 0001018724. The page displays various filing details, including the filing type, format, description, and the date the filing was effective. The table below summarizes the filings shown on the page.

Filings	Format	Description	Filed/Effective	File/Film Number
8-K	Documents	Current report, items 5.02 and 9.01 Acc-no: 0001018724-19-000081 (34 Act) Size: 215 KB	2019-08-01	000-22513 19993402
IRANNOTICE	Documents	Notice of disclosure filed pursuant to Section 219 of the Iran Threat Reduction and Syria Human Rights Act of 2012 (Exchange Act Section 13(r)). Acc-no: 0001018724-19-000073 (34 Act) Size: 124 KB	2019-07-26	000-22513 19975702
10-Q	Documents	Quarterly report [Sections 13 or 15(d)] Acc-no: 0001018724-19-000071 (34 Act) Size: 7 MB	2019-07-26	000-22513 19975687
8-K	Documents	Current report, items 2.02 and 9.01 Acc-no: 0001018724-19-000069 (34 Act) Size: 1 MB	2019-07-25	000-22513 19974829

## Level 2: Using a code to download multiple files

### ex. Company filings from SEC.gov

```
import requests
def download(url):
    path = url.split("/")[-1]
    r = requests.get(url)
    open(path, 'wb').write(r.content)

urls =
['https://www.sec.gov/Archives/edgar/data/320193/0000
320193-19-000026.txt',
'https://www.sec.gov/Archives/edgar/data/320193/00011
93125-19-004664.txt']

for url in urls:
    download(url)
```

- Define a function for downloading
- Use the same file name obtained from url
- Get the file indicated by the url
- Write the content of the file indicated by the url as the name defined in path
- Set url addresses (ex. from SEC.gov)
- Call the download function repeatedly

## Level 2: Using a code to download multiple files ex. Company filings from SEC.gov

SEC.gov explains how to access files from their server:

<https://www.sec.gov/edgar/searchedgar/accessing-edgar-data.htm>

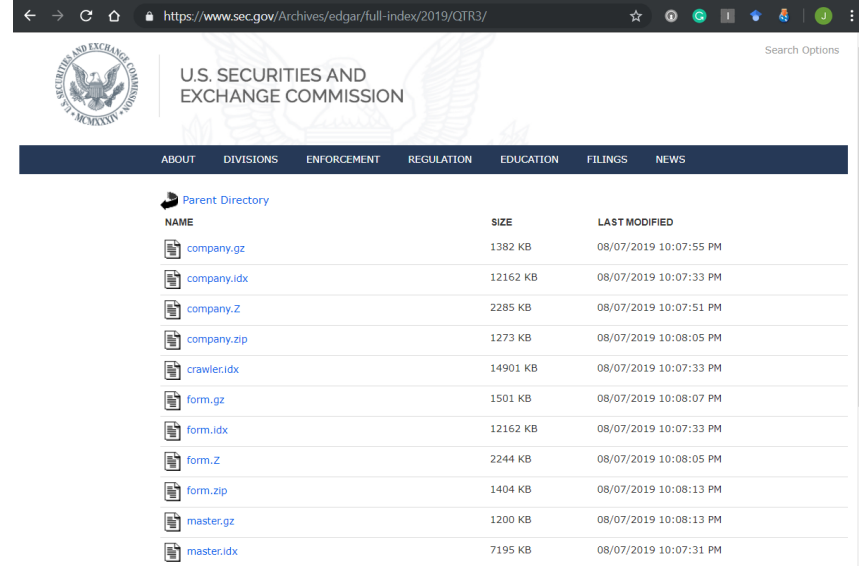
For example, we can use the file locations written in the master index provided by SEC.gov:

<https://www.sec.gov/Archives/edgar/full-index/2019/QTR1/master.idx>

master.idx












Description: Master Index of EDGAR Dissemination FeedLast Data Received: March 31, 2019  
Comments: webmaster@sec.gov Anonymous FTP: ftp://ftp.sec.gov/edgar/Cloud HTTP:  
<https://www.sec.gov/Archives/CIK/Company Name/Form Type/Date Filed/Filename>

1000045|NICHOLAS FINANCIAL INC|10-Q|2019-02-14|edgar/data/1000045/0001193125-19-039489.txt  
1000045|NICHOLAS FINANCIAL INC|4|2019-01-15|edgar/data/1000045/0001357521-19-000001.txt  
1000045|NICHOLAS FINANCIAL INC|4|2019-02-19|edgar/data/1000045/0001357521-19-000002.txt  
1000045|NICHOLAS FINANCIAL INC|4|2019-03-15|edgar/data/1000045/0001357521-19-000003.txt  
1000045|NICHOLAS FINANCIAL INC|8-K|2019-02-01|edgar/data/1000045/0001193125-19-024617.txt  
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-04|edgar/data/1000045/0001104659-19-005360.txt  
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-08|edgar/data/1000045/0001258897-19-001312.txt  
1000045|NICHOLAS FINANCIAL INC|SC 13G/A|2019-02-11|edgar/data/1000045/0001019056-19-000082.txt



U.S. SECURITIES AND EXCHANGE COMMISSION

Search Options

NAME	SIZE	LAST MODIFIED
 <a href="#">company.gz</a>	1382 KB	08/07/2019 10:07:55 PM
 <a href="#">company.idx</a>	12162 KB	08/07/2019 10:07:33 PM
 <a href="#">company.Z</a>	2285 KB	08/07/2019 10:07:51 PM
 <a href="#">company.zip</a>	1273 KB	08/07/2019 10:08:05 PM
 <a href="#">crawler.idx</a>	14901 KB	08/07/2019 10:07:33 PM
 <a href="#">form.gz</a>	1501 KB	08/07/2019 10:08:07 PM
 <a href="#">form.idx</a>	12162 KB	08/07/2019 10:07:33 PM
 <a href="#">form.Z</a>	2244 KB	08/07/2019 10:08:05 PM
 <a href="#">form.zip</a>	1404 KB	08/07/2019 10:08:13 PM
 <a href="#">master.gz</a>	1200 KB	08/07/2019 10:08:13 PM
 <a href="#">master.idx</a>	7195 KB	08/07/2019 10:07:31 PM

## Level 3. Using API (Application Programming Interface)

```
# Construct a vector of 2 URLs
urls <- c("http://httpbin.org/status/404", "http://httpbin.org/status/301")
for(url in urls){
  # Send a GET request to url
  result <- GET(url, user_agent("my@email.address this is a test"))

  # Check request_result
  if(http_error(result)){
    warning('The request failed')
  } else {
    content(result)
  }

  # Delay for 5 seconds between requests
  Sys.sleep(5)
}

# Create list with nationality and country elements
query_params <- list(nationality = "americans", country =
"antigua")
# Make parameter-based call to httpbin, with
query_paramsparemeter_response <-
GET("https://httpbin.org/get", query = query_params)
# Print parameter_responseparemeter_response

# Construct a directory-based API URL to `http://swapi.co/api`,
# looking for person `1` in `people`directory_url <-
paste("http://swapi.co/api", "people", "1", sep = '/')# Make a GET
call with itresult <- GET(directory_url)
```



## Level 3. Using API (Application Programming Interface)

### ex. R code for collecting table information from Wikipedia

```
library(httr)
library(rvest)
library(xml2)

get_infobox <- function(title){
  base_url <- "https://en.wikipedia.org/w/api.php"

  # Change "Hadley Wickham" to title
  query_params <- list(action = "parse",
                        page = title,
                        format = "xml")

  resp <- GET(url = base_url, query = query_params)
  resp_xml <- content(resp)

  page_html <- read_html(xml_text(resp_xml))
  infobox_element <- html_node(x = page_html, css = ".infobox")
  page_name <- html_node(x = infobox_element, css = ".fn")
  page_title <- html_text(page_name)

  wiki_table <- html_table(infobox_element)
  colnames(wiki_table) <- c("key", "value")
  cleaned_table <- subset(wiki_table, !wiki_table$key == "")
  name_df <- data.frame(key = "Full name", value = page_title)
  wiki_table <- rbind(name_df, cleaned_table)

  wiki_table
}

# Test get_infobox with "Hadley Wickham"
get_infobox(title = "Hadley Wickham")
```

## Level 4. Using a code for Web scraping

It is possible to extract texts from any websites using R and Python.

[to be updated]

## 10K and proxy statements from SEC.gov and context analysis

- Abraham Lu at IMD Global Board Center ([abraham.lu@imd.org](mailto:abraham.lu@imd.org))

## Webscribing codes

<https://www.datacamp.com/courses/working-with-web-data-in-r>