

# Introduction to Web Scraping in R

Retrieve your own Data



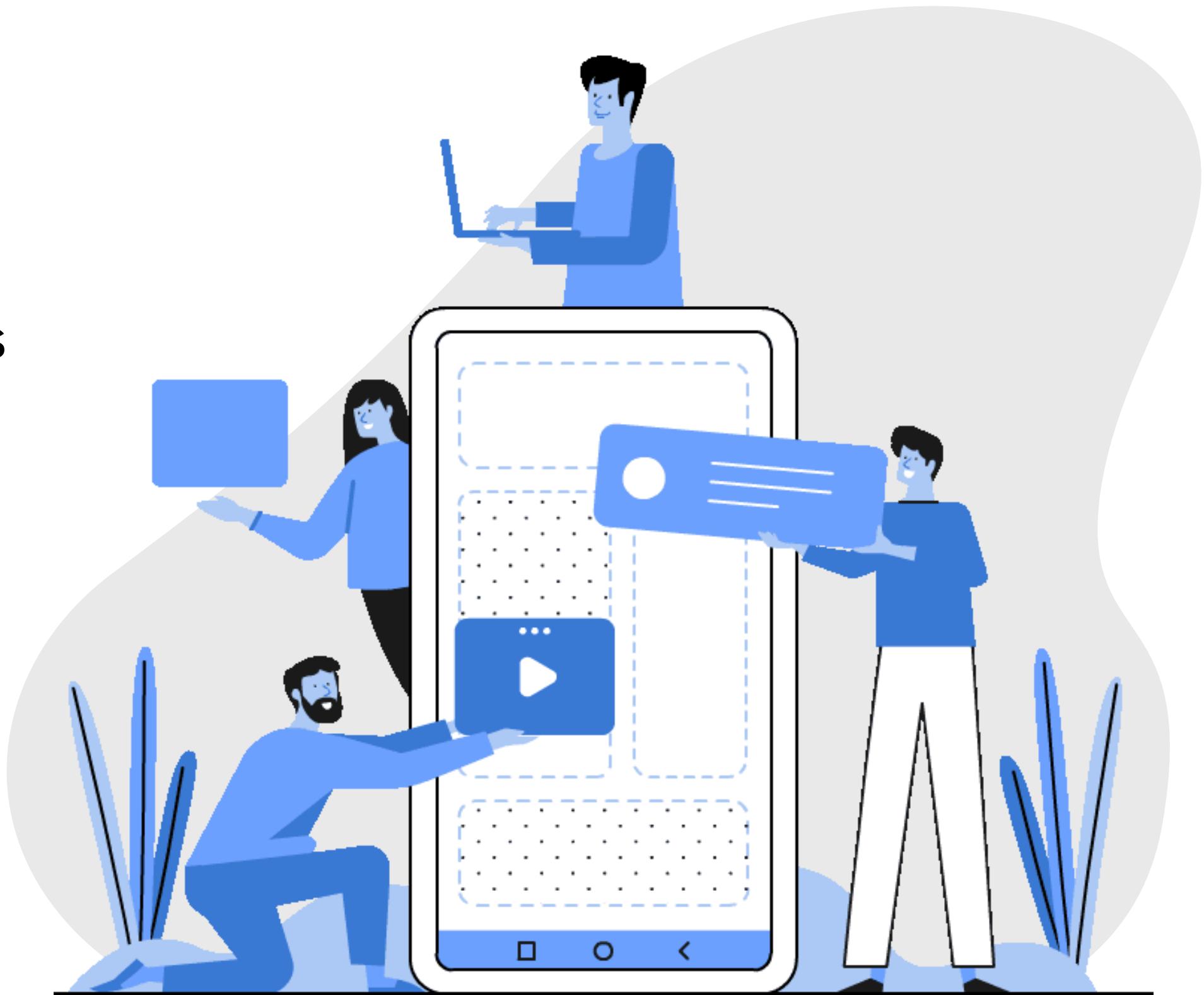


# WEB SCRAPING

- Web scraping adalah penggunaan perangkat lunak untuk mengekstrak informasi dari situs web.
- Mengubah internet menjadi sumber data potensial untuk berbagai proyek penelitian

# Manfaat WEB SCRAPING

- Dapat mengunduh banyak file dari situs web dengan cepat
- Dapat menyimpan konten dari seluruh pengguna internet untuk mengklasifikasikan persepsi atau sentimen mereka
- Dapat terdiri dari beberapa konten arsip yang mungkin telah "hilang" dari web



# Tujuan

Pada perkuliahan ini, akan diperkenalkan WEB SCRAPING sebagai alternatif pengumpulan data dari situs web maupun sosial media (twitter, instagram, facebook, dll) dalam Data Science. Kita akan mempelajari beberapa teknik untuk mengunduh teks atau file dengan cepat dari halaman web yang terstruktur.





# **Langkah**

# **WEB SCRAPING**

**berbasis teks dari situs web**



1. Identifikasi informasi di internet yang ingin Anda gunakan.
2. Jika informasi disimpan di lebih dari satu halaman web, cari tahu cara navigasi secara otomatis antar halaman web.  
Best case scenario : Anda memiliki halaman direktori atau URL yang memiliki pola konsisten  
misal : [www.somewebsite.com/year/month/day.html](http://www.somewebsite.com/year/month/day.html).

### 3. Temukan fitur di situs web yang menandai informasi yang ingin Anda ekstrak.



Langkah ini dapat dilakukan dengan melakukan identifikasi HTML untuk menemukan elemen yang Anda inginkan dan / atau mengidentifikasi semacam pola di situs web.

### 4. Tulis skrip untuk mengekstrak, memformat, dan menyimpan informasi yang Anda inginkan



5. Ulangi langkah 2-3, terapkan skrip yang telah dibuat pada langkah 4 untuk masing-masing situs.
6. Lakukan analisis pada data Anda yang Anda peroleh!

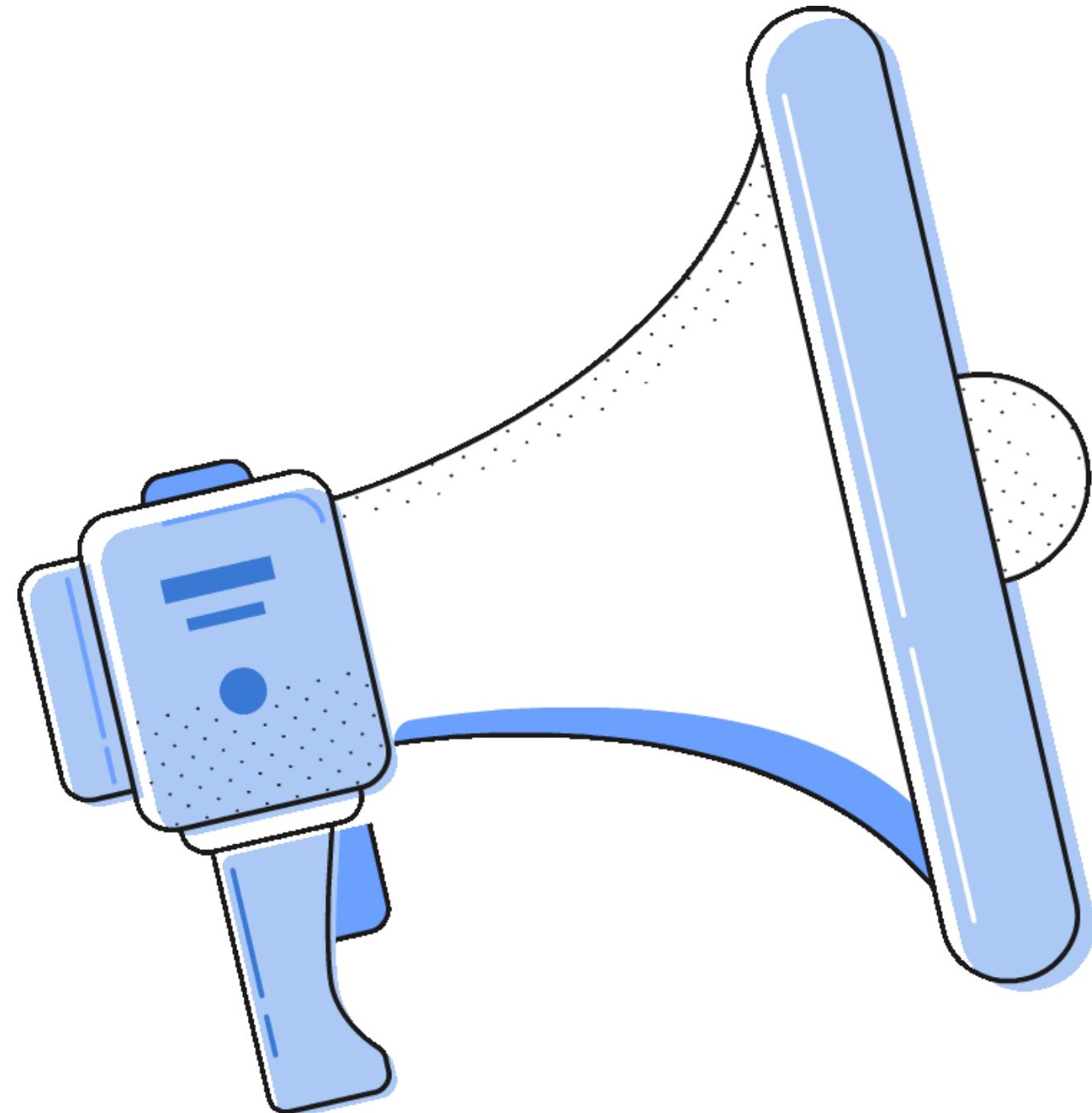
# Target Pembelajaran

Pembahasan kali ini akan memberikan perkenalan teori yang berfokus pada langkah 3 dan 4, yang merupakan bagian inti dari WEB SCRAPING, dan mencoba memahami contoh-contoh kode/script WEB SCRAPING serta mengimplementasikannya



Cara yang lebih sederhana dalam WEB SCRAPING adalah dengan menggunakan Application Programming Interfaces (APIs) yang disediakan beberapa situs web. API pada dasarnya memberi Anda cara sederhana untuk melakukan query pada database dan mengembalikan data yang Anda minta dalam format yang rapi (biasanya JSON atau XML).





# API adalah alternatif yang cukup baik

tetapi tidak semua situs memberikan  
akses yang "terbuka" bagi API nya,  
sehingga API bukan merupakan fokus  
utama dalam pembelajaran ini.

# Identifikasi HTML untuk ekstraksi informasi dengan WEB SCRAPING





Hypertext Markup Language - atau HTML - adalah sistem standar untuk menulis halaman web. Strukturnya cukup sederhana, dan mengidentifikasi isinya adalah salah satu langkah penting untuk menghasilkan WEB SCRAPING yang sukses

# Basically, what a website looks like...

```
<!DOCTYPE html>
<html>

    <head>
        <title>This is the title of the webpage</title>
    </head>

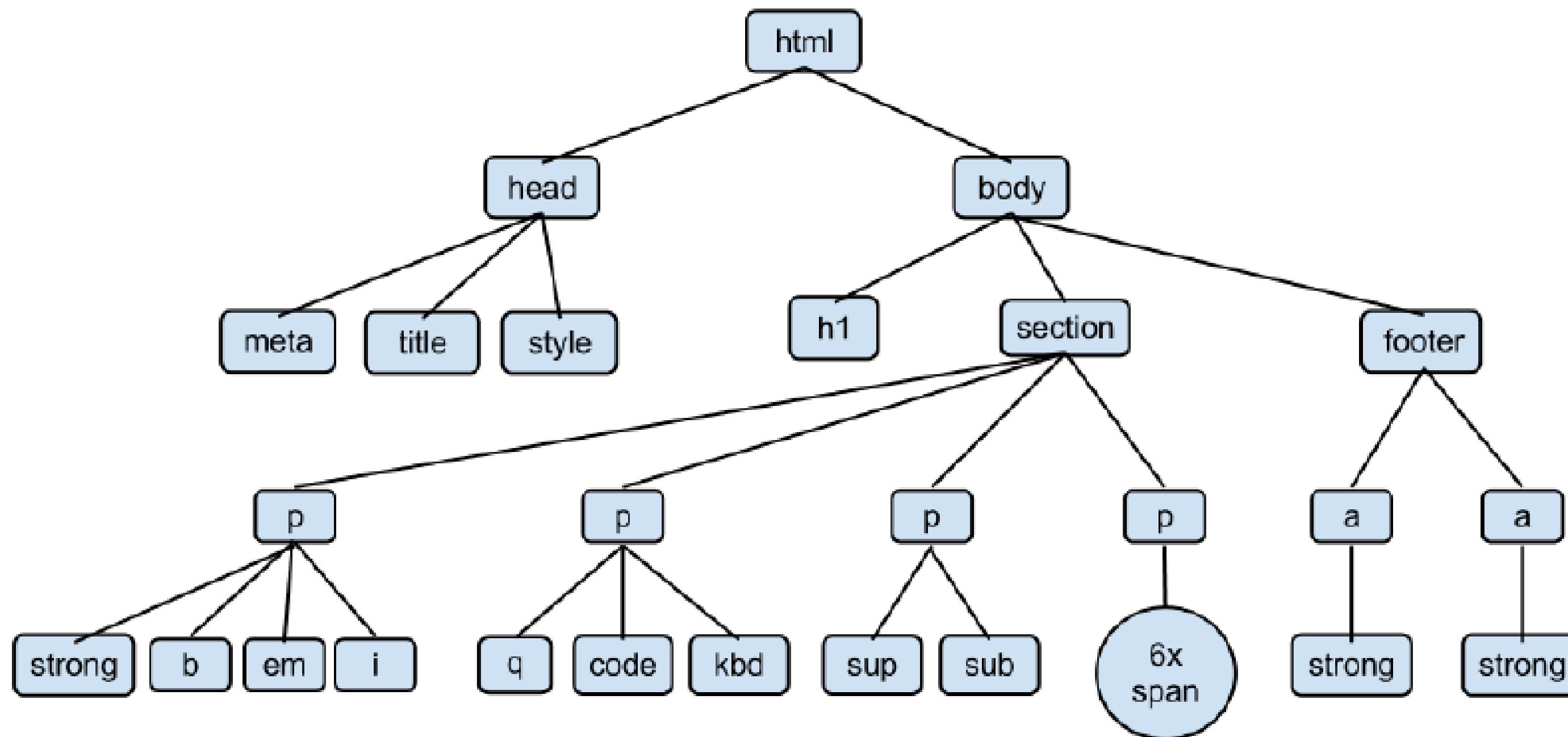
    <body>
        <h1>This is a heading</h1>
        <p class="notThisOne">This is a paragraph</p>
        <p class="thisOne">This is another paragraph with a different class!</p>

        <div id="myDivID">
            <p class="divGraf">
                This is a paragraph inside a division, along with a
                <a href="http://stanford.edu">a link</a>.
            </p>
        </div>
    </body>

</html>
```

\*\* Melihat HTML di Chrome : View -> Developer -> View Source.

# Visual representation of an HTML tree



# Mengingat kembali tentang HTML ..

01

**Tiap elemen pada website terdiri dari sekumpulan kode/script yang mendefinisikan bentuk elemen tersebut agar dapat dikenali browser**

Tag dibuka dengan tanda kurung segitiga `<tag>` dan ditutup dengan garis miring di dalam tanda kurung segitiga `</tag>`.

# Mengingat kembali tentang HTML ..

02

## **Informasi tambahan pada tag**

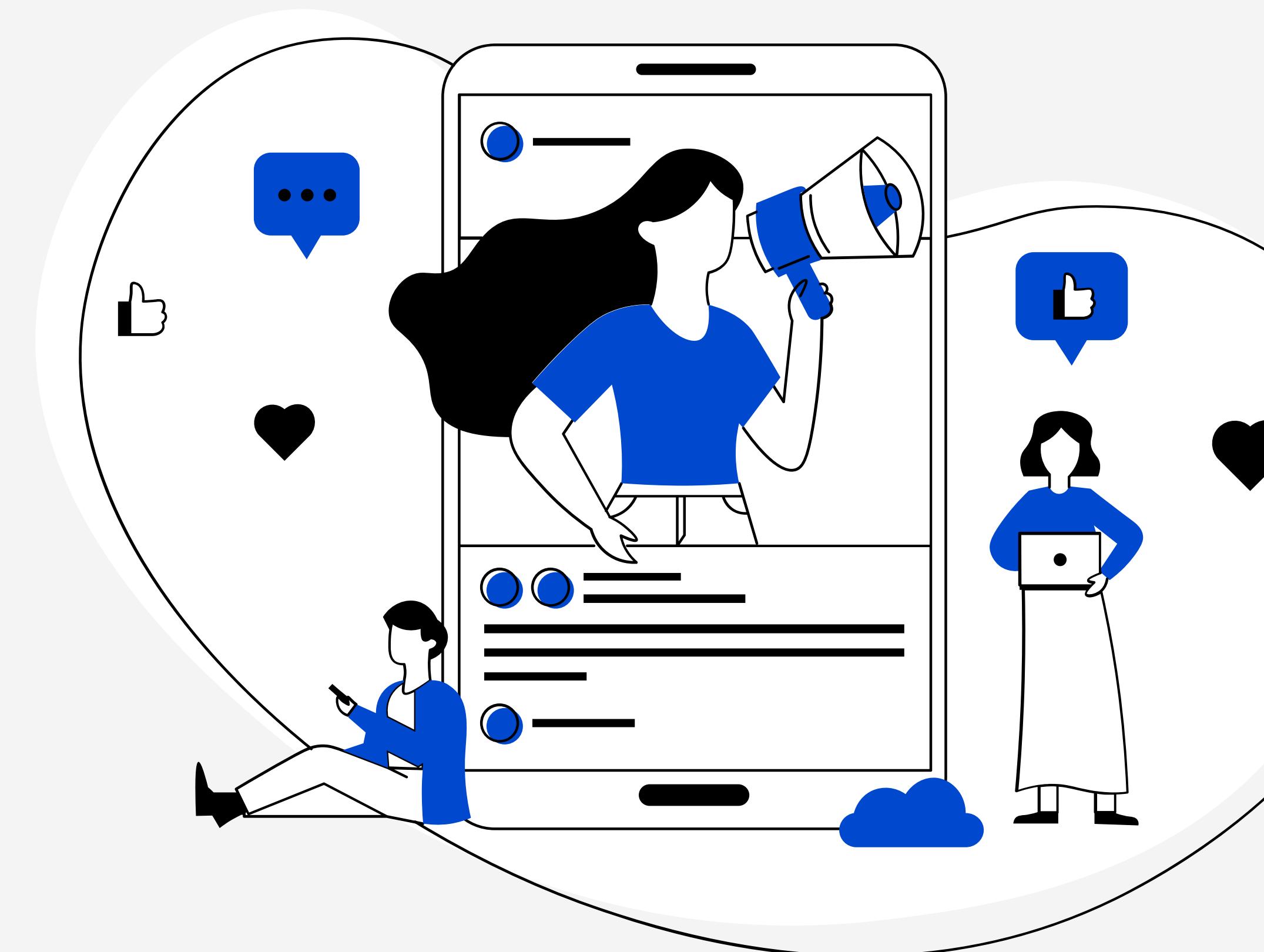
tag sering memiliki informasi tambahan, seperti informasi tentang kelas.

03

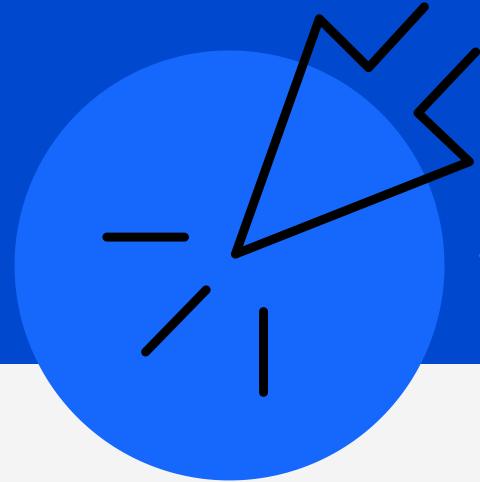
## **Nested element**

Tiap elemen-elemen dalam halaman biasanya "bersarang" di dalam elemen-elemen lain.

# Tools untuk WEB SCRAPING



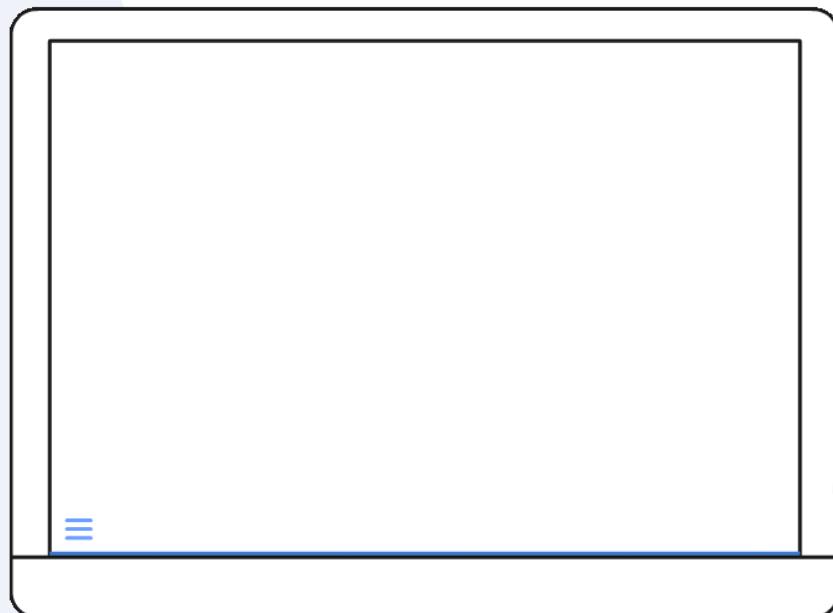
# RVEST



**Alur kerjanya adalah sebagai berikut:**

- 1** Membaca halaman web menggunakan fungsi `read_html ()`. Fungsi ini akan mengunduh HTML dan menyimpannya sehingga rvest dapat menavigasi HTML tersebut..
- 2** Pilih elemen yang Anda inginkan menggunakan fungsi `html_nodes ()`. Fungsi ini akan mengambil objek HTML (dari `read_html`) dengan CSS atau Xpath dan menyimpan semua elemen yang sesuai dengan yang diinginkan.
- 3** Ekstrak komponen node yang telah Anda pilih menggunakan fungsi seperti `html_tag ()` (nama tag), `html_text ()` (semua teks di dalam tag), `html_attr ()` (konten atribut tunggal) dan `htmlAttrs ()` (semua atribut)

# Regular Expressions



Seringkali Anda akan melihat pola dalam teks yang ingin Anda eksploitasi. Misalnya, variabel baru yang diikuti tanda titik dua yang muncul setelah satu kata di baris baru.

Ekspresi reguler (atau regex) adalah bahasa yang secara tepat dapat mendefinisikan pola-pola semacam contoh diatas.

Regex sangat penting untuk pembuatan web dan analisis teks.

# Regular Expressions



beberapa perintah regex yang dapat Anda gunakan :

## 1 grep(pattern, string)

Perintah ini mengambil vektor string dan mengembalikan vektor indeks string yang cocok dengan pola

```
string = c("this is", "a string", "vector", "this")
grep("this", string)

## [1] 1 4
```

# Regular Expressions



beberapa perintah regex yang dapat Anda gunakan :

2

## **grepl(pattern, string)**

Perintah ini mengambil vektor string dengan panjang n sebagai input dan mengembalikan vektor logis dengan panjang n yang mengatakan apakah string cocok dengan pola.

```
grepl("this", string)
```

```
## [1] TRUE FALSE FALSE TRUE
```

# Regular Expressions



beberapa perintah regex yang dapat Anda gunakan :

## 3 **gsub(pattern, replacement, string)**

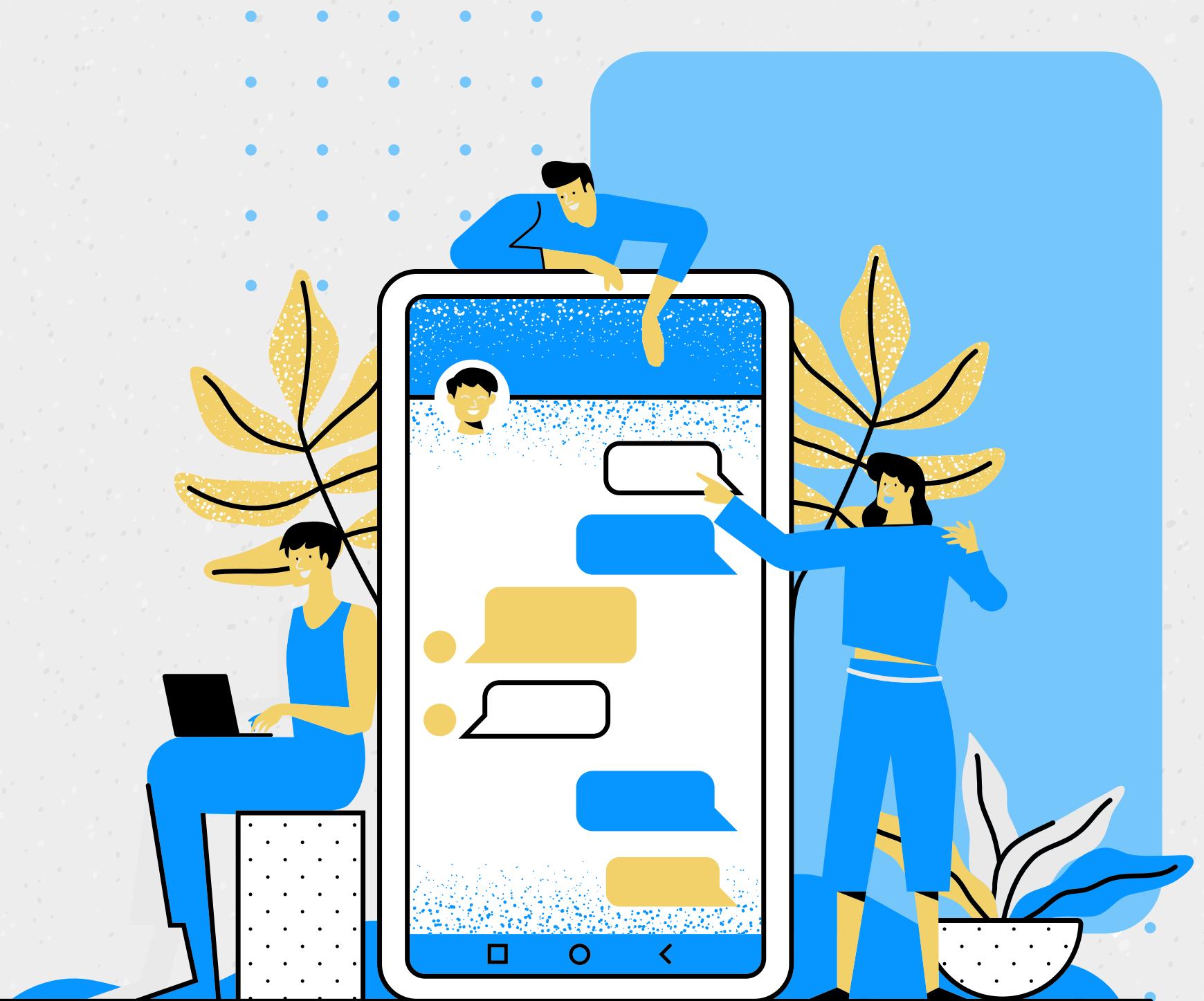
Perintah ini menemukan semua instance pola dalam string dan menggantinya dengan pola penggantian baru yang diinginkan.

```
gsub(pattern="is", replacement="WTF", string)
```

```
## [1] "thWTF WTF" "a string" "vector" "thWTF"
```

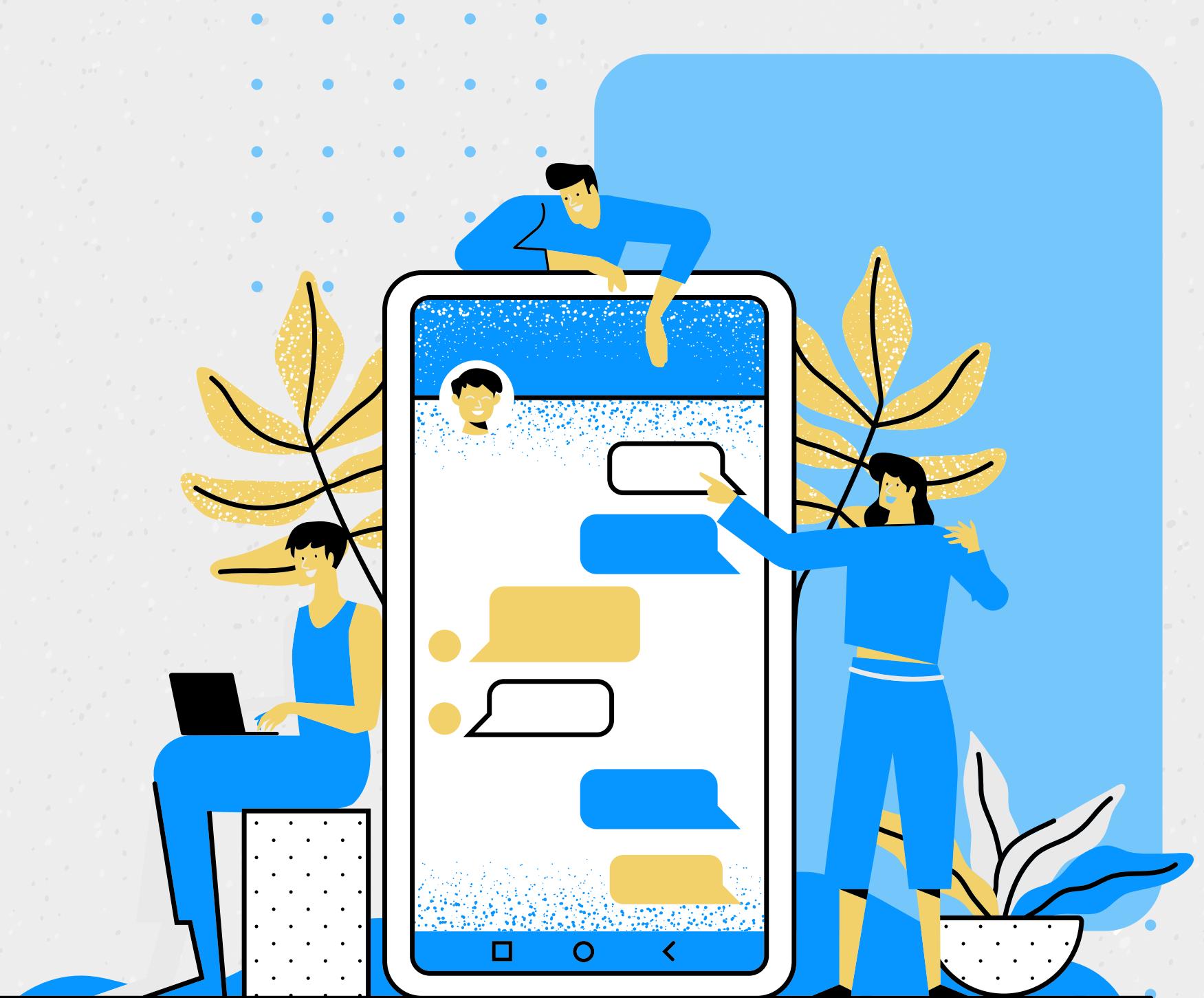
# Contoh Sederhana

WEB SCRAPING dengan RVEST



Kita akan melihat seperti apa bentuk situs web contoh di RVEST. Langkah-langkahnya adalah sebagai berikut:

1. Membaca dalam HTML.
2. Memilih semua paragraf, lalu pilih elemen dengan kelas "thisOne," dan kemudian memilih elemen dengan ID "myDivID."
3. Mengekstrak beberapa teks dan tautannya.



```
## First, load required packages (or install if they're not already)
pkgs = c("rvest", "magrittr", "httr", "stringr")
for (pkg in pkgs){ if (!require(pkg, character.only = T)){ install.packages(pkg)
  library(pkg)
}
}
```

```
## Read my example html with read_html()
silly_webpage = read_html("http://stanford.edu/~wpmarble/webscraping_tutorial/html/silly_webpage.html")

# get paragraphs (css selector "p")
my_paragraphs = html_nodes(silly_webpage, "p")
my_paragraphs
## {xml_node[set (3)]}
## [1] <p class="notThisOne">This is a paragraph</p>
## [2] <p class="thisOne">This is another paragraph with a different class! ...
## [3] <p class="divGraf"> \n This is a paragraph inside a division, ...

# get elements with class "thisOne" -- use a period to denote class
thisone_elements = html_nodes(silly_webpage, ".thisone")
thisone_elements
## {xml_node[set (1)]}
## [1] <p class="thisOne">This is another paragraph with a different class! ...

# get elements with id "myDivID" -- use a hashtag to denote id
myDivID_elements = html_nodes(silly_webpage, "#myDivID")
myDivID_elements
## {xml_node[set (1)]}
## [1] <div id="myDivID"> \n <p class="divGraf"> \n This is a p ...

# extract text from myDivID_elements
myDivID_text = html_text(myDivID_elements)
myDivID_text
## [1] " \n \n This is a paragraph inside a division, along with a \n a link.\n

# extract links from myDivID_elements. first i extract all the "a" nodes (as in a href="website.com")
# and then extract the "href" attribute from those nodes
myDivID_link = html_nodes(myDivID_elements, "a") %>% html_attr("href")
myDivID_link
## [1] "http://stanford.edu"
```

Pada contoh diatas digunakan penyeleksi CSS (kelas dan ID) untuk mengekstrak node dari HTML. Beberapa yang perlu diperhatikan adalah :

- Untuk memilih kelas, Anda memberi tanda titik di depan nama kelas -  
html\_nodes (silly\_webpage, ".thisOne").
- Untuk memilih ID, letakkan tagar di depan ID yang Anda inginkan -  
html\_nodes (silly\_webpage, "#myDivID").



# **TUGAS INDIVIDU**

**Temukan contoh script / kode  
WEB SCRAPING menggunakan R.  
Pahami tiap baris kodennya dan coba lakukan  
WEB SCRAPING dengan script yang Anda peroleh.**

Kumpulkan tugas dalam bentuk .rar (terdiri dari 3 file) :

1. File .pdf yang berisi:  
Penjelasan sesuai pemahaman dari tiap fungsi/baris script.
2. File .csv yang berisi:  
Hasil WEB SCRAPING yang Anda lakukan dalam bentuk  
dataset.
3. Script R (cantumkan sumber nya dalam script)

\*\* Jika ingin memodifikasi / membuat sendiri script sesuai kemampuan diperbolehkan. Metode WEB SCRAPING bebas.

**KELAS - A**

[bit.ly/DSA-TUGAS4](http://bit.ly/DSA-TUGAS4)

**KELAS - B**

[bit.ly/DSB-TUGAS4](http://bit.ly/DSB-TUGAS4)

**KELAS - C**

[bit.ly/DSC-TUGAS4](http://bit.ly/DSC-TUGAS4)

**DEADLINE :  
SABTU, 11 APRIL 2020 JAM 24:00**

# PERHATIKAN..

---

Dataset hasil WEB SCRAPING akan digunakan pada tugas selanjutnya.. seluruh mahasiswa diharapkan dapat mengerjakan tugas - tugas yang diberikan selama perkuliahan online.....

Project akhir kelas DITIADAKAN, diganti dengan mini project berkelanjutan di beberapa pertemuan perkuliahan online.