

Web scraping 2

Advanced Crime Analysis UCL

Bennett Kleinberg

21 Jan 2019



Getting data from the Internet

Webscraping 2

Today

- “Real” webscraping: basics of a webpage
- Access webpage in the wild
- Retrieve wild data
- Download data through webscraping

APIs: Pros & Cons

Pro

- easily to access
- nicely documentation
- **works even if website changes**

Cons

- quota limits (\$ \$ \$)
- under the platforms' control
- only for few platforms

Don't let the data determine your research!

COOL

But what about:

Due to the lapse in appropriations, Department of Justice websites will not be regularly updated. Please refer to the Department of Justice's [Facebook page](#).

[←](#) [→](#) [G](#) [↑](#) <https://www.fbi.gov/wanted>

MOST WANTED

Ten Most Wanted | Fugitives | Terrorism | Kidnapping/Missing Persons | Seeking Info | Parental Kidnapping

Ten Most Wanted
RAFAEL CARO-
QUINTERO
HASAN IZZ-AL-DIN

Most Wanted Terrorists
JOEL BARRAGAN
CRIMINAL ENTERPRISE INVESTIGATIONS

Crimes Against Children
LUIS TEJADA



National Crime Agency (GB) | <https://www.missingpersons.police.uk/en-gb/case->



Bureau Reference: 18-009282
Location London
Gender Male
Date found 21 September 2018
Age 30 - 40
Ethnicity White European

[View case details](#)

Bureau Reference:

18-006195

Location

Dunsden

Gender

Male

Date found

15 May 2018

Age

16 - 100

Ethnicity

White European

[View case details](#)



No APIs

- incels.me
- stormfront
- 4chan
- **APIs are restrictive!**

... what about:
Your research question → no API?



Main problem:

Really ‘juicy’ data of the Internet vs APIs

“Real” webscraping: basics of a webpage

- # Three elements of a webpage
1. Structure
 2. Behaviour
 3. Style

Three elements of a webpage

1. Structure
2. Behaviour
 - JavaScript (!= Java)
 - user interaction
 - examples: alerts, popups, server-interaction
3. Style

Three elements of a webpage

1. Structure

2. Behaviour

3. Style

- CSS (Cascading Style Sheets)
 - formatting, design, responsiveness
 - examples: submit buttons, app interfaces

Three elements of a webpage

1. Structure
 - HTML (hypertext markup language)
 - structured with <tags>
 - contains the pure content of the webpage
2. Behaviour
3. Style

For now: HTML

The very basics of HTML:

Raw architecture of a webpage

```
<!DOCTYPE html>
<html>
<body>
HERE COMES THE VISIBLE PART ! 
</body>
</html>
```

Note: Every tags `< >` is closed `< />`. Content is contained within the tag.

HTML basics

Ways to put content in the `<body> ... </body>` tag:

- headings: `<h1>I'm a heading at level 1</h1>`

```
I'm a heading at level 1
```

file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

Content in the body tag

- paragraphs: <p>This is a paragraph</p>

The screenshot shows a presentation slide with a black border. At the top left, there is a navigation bar with icons for back, forward, search, and refresh. To the right of the navigation bar is a URL bar containing the text "file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html". The main content area contains the following text:

I'm a heading at level 1

I'm the first paragraph
I'm the second paragraph
I'm the third paragraph

Content in the body tag

- images:



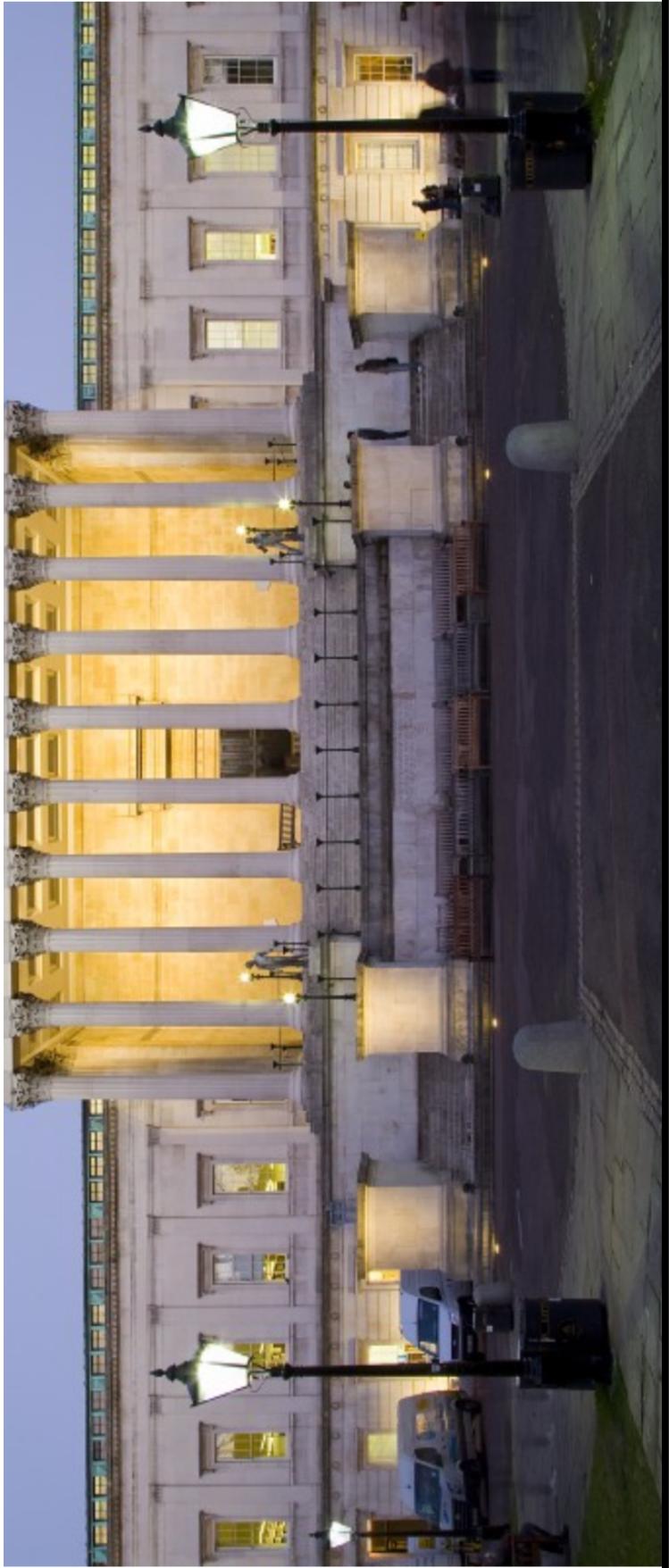
I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph





Content in the body tag

- links:

```
<a href="https://www.ucl.ac.uk/">Click here</a>
```

The screenshot shows a simple HTML page with the following structure:

```
<html>
  <head>
    <meta charset="UTF-8">
    <title>A Simple HTML Document</title>
  </head>
  <body>
    <h1>I'm a heading at level 1</h1>
    <p>I'm the first paragraph</p>
    <p>I'm the second paragraph</p>
    <p>I'm the third paragraph</p>
    <a href="https://www.ucl.ac.uk/">Click here to go to UCL's website</a>
  </body>
</html>
```

The page contains an **H1** heading with the text "I'm a heading at level 1". Below it are three regular paragraphs with the text "I'm the first paragraph", "I'm the second paragraph", and "I'm the third paragraph". At the bottom of the page is a link "Click here to go to UCL's website". The browser interface includes a back button, forward button, search bar, and address bar.

Content in the body tag

- tables

```
<table>
<tr>
<th>Departments</th>
<th>Location</th>
</tr>
<tr>
<td>Dept. of Security and Crime Science</td>
<td>Division of Psychology and Language Sciences</td>
</tr>
<tr>
<td>35 Tavistock Square</td>
<td>26 Bedford Way</td>
</tr>
</table>
```

Html <table> . . . </table>

◀ → ⌂ ⌂ ⓘ file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph

[Click here to go to UCL's website](#)

Departments

Dept. of Security and Crime Science
Division of Psychology and Language Sciences
35 Tavistock Square
26 Bedford Way

Location

Content in the body tag

- lists

```
<ul>
  <li>Terrorism</li>
  <li>Cyber Crime</li>
  <li>Data Science</li>
</ul>
```



I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph

[Click here to go to UCI's website](#)

Departments

Dept. of Security and Crime Science
Division of Psychology and Language Sciences
35 Tavistock Square
26 Bedford Way

Location

- Terrorism
- Cyber Crime
- Data Science

HTML basics

Elements (can) have IDs:

```
<p id='paragraph1'>This is a paragraph</p>

```

Same for tables, links, etc.

Every element can have an ID.

You need unique IDs! Two elements cannot have the same ID.

HTML basics

Common elements (can) have CLASSES:

```
<p id="paragraph1" class="paragraph_class">I am the first paragraph</p>
<p class="paragraph_class">I am the second paragraph</p>
<p class="paragraph_class">I am the third paragraph</p>
```

Multiple elements can have the same class.

Now what?

Web scraping logic

If all webpages are built in this structure...

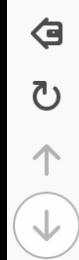
... then we could access this structure programmatically.

But where do I find that structure?

Is it just “there”?

YES!.

How to see the html structure?

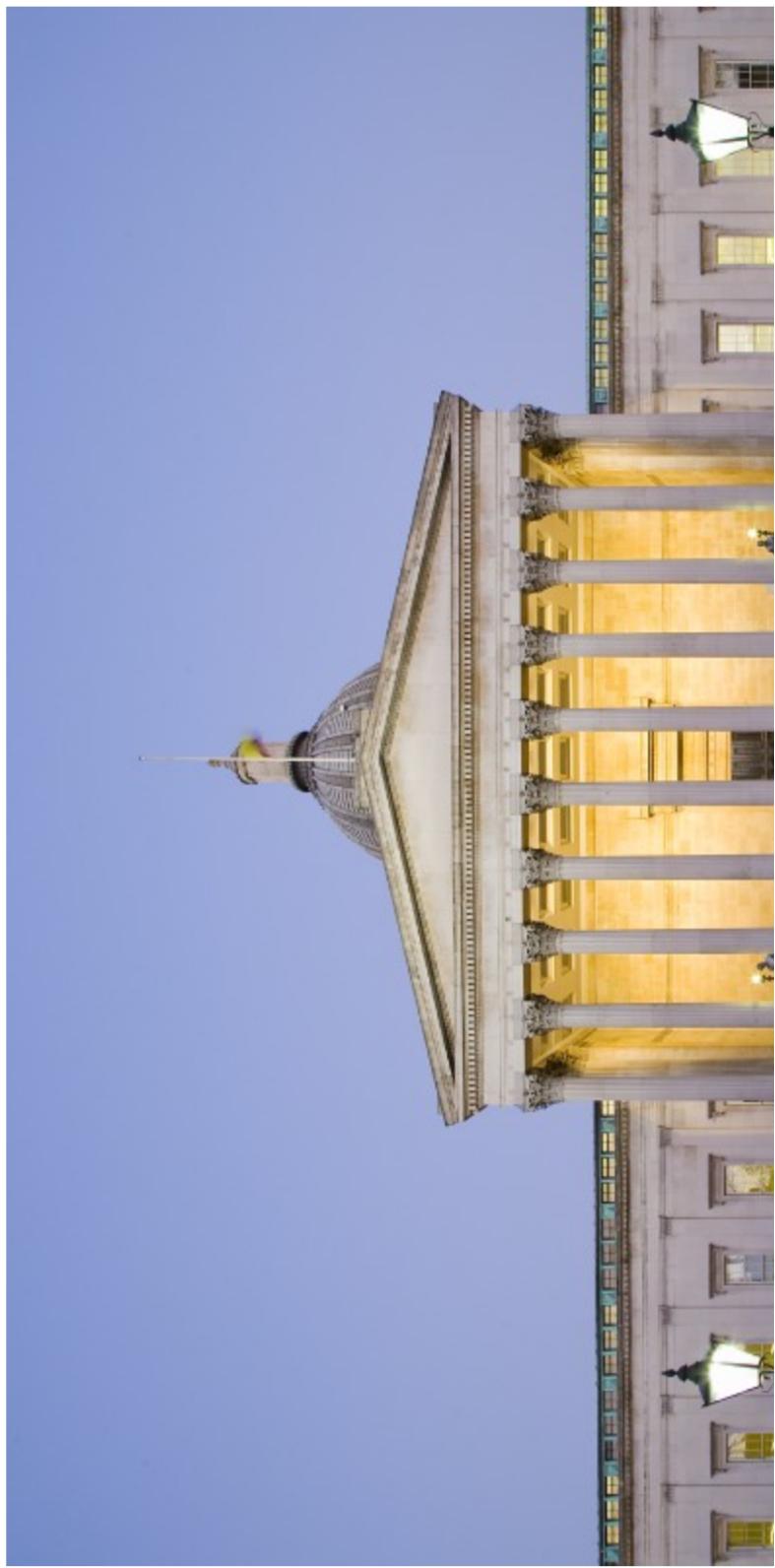


I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph







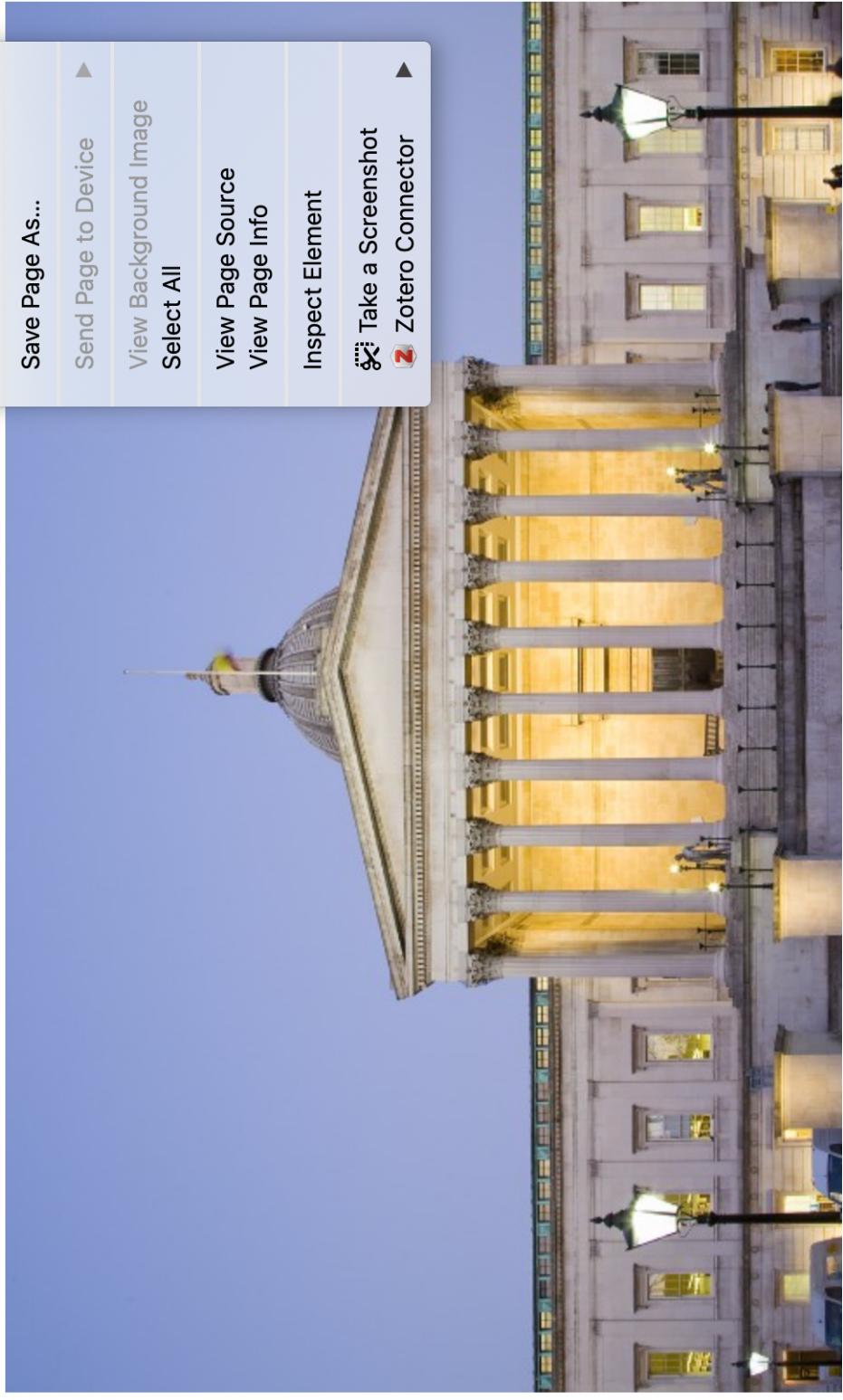
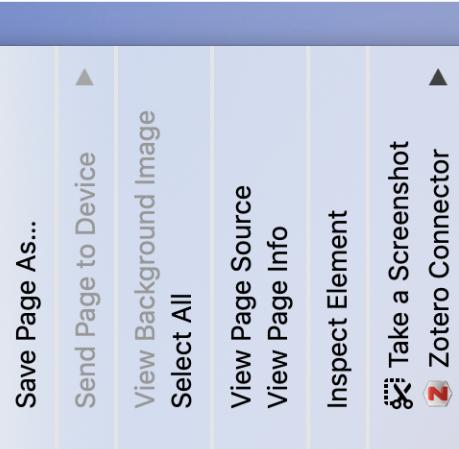
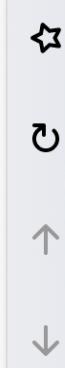
① file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph





◀ → ⌂ ⌂ ⌂

ⓘ file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

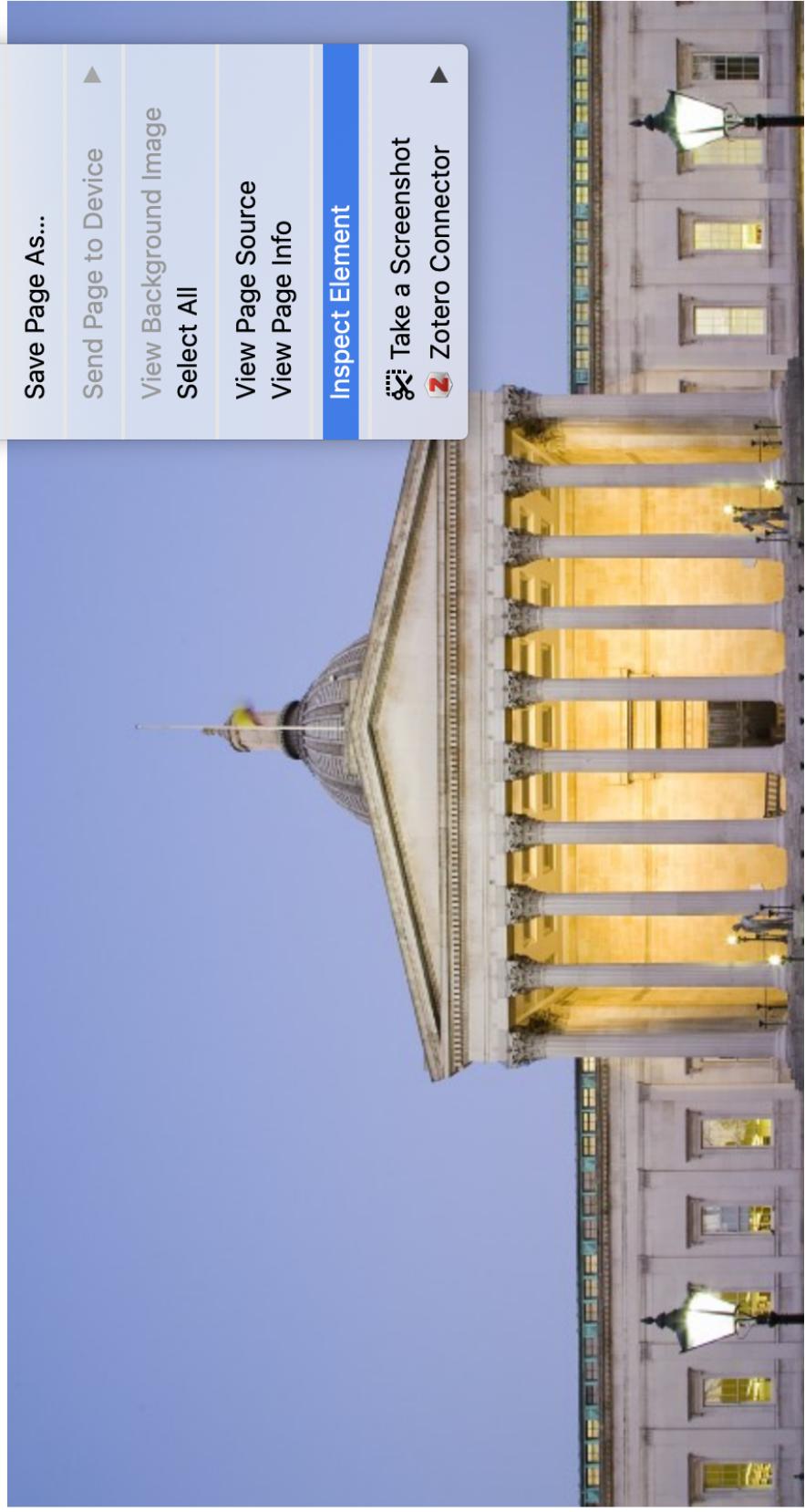
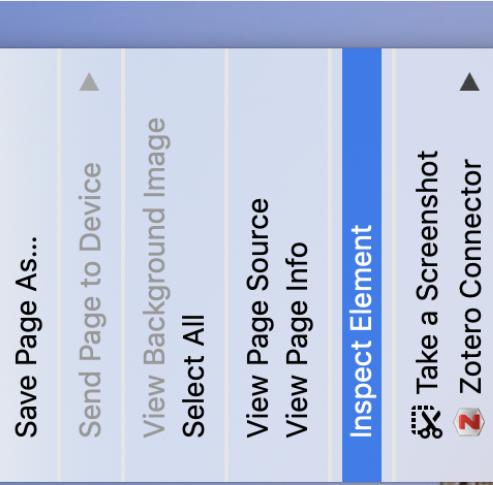
I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph

◀ → ⌂ ⌂ ⌂





html gr_ > body

```
</DOCTYPE html>
<html class="gr__"> event
<head></head>
<body data-gr-c-s-loaded="true">
  <h1>I'm a heading at level 1</h1>
  <p id="paragraph1" class="paragraph_class">I'm the first paragraph</p>
  <p class="paragraph_class">I'm the second paragraph</p>
  <p class="paragraph_class">I'm the third paragraph</p>
  
  <br>
  <a href="https://www.ucl.ac.uk/">Click here to go to UCL's website</a>
  <br>
  <br>
  ▲ <table> ... </table>
  ▲ <ul> ... </ul>
  </body>
</html>
```

```
▼ <table>
  ▼ <tbody>
    ▶ <tr>
      <th>Departments</th>
      <th>Location</th>
    </tr>
    ▶ <tr>
      <td>Dept. of Security and Crime Science</td>
      <td>Division of Psychology and Language Sciences</td>
    </tr>
    ▶ <tr> ...
    </tbody>
</table>
```

Example 1: Missing persons

Order By

Date Found - Most Recent First ▾

Sort

div Case | 403 × 390.05

Bureau Reference:
18-009282

Location
London

Gender
Male

Date found
21 September 2018

Age
30 - 40

Ethnicity
White European

View case details

```
<div id="PageHeader">...</div>
<div class="Container">
  <div id="PageContent">
    <div class="Indent">
      <div class="row">
        ::before
        <div class="col-sm-8 mobileMargin">
          <div class="Breadcrumb">...</div>
          <h1>Case Search</h1>
          <form class="CaseSearch Existing" action="/en-gb/case-search/" method="post">...</form>
          <event>
            <div class="col-sm-8 mobileMargin">
              <div class="row">
                <div class="col-sm-4">
                  <h2>...</h2>
                  <p>...</p>
                  <p>...</p>
                  <p class="Pagination">...</p>
                  <form class="form-in-line" action="/en-gb/case-search/p24442">
                    <div class="form-group">
                      <label for="orderBy">Order By</label>
                      <select id="orderBy" class="form-control" name="orderBy">...</select>
                    </div>
                    <input class="btn btn-default" type="submit" value="Sort">
                  </form>
                  <br>
                  <div class="UnindentGrid">
                    ::before
                    <div class="Case">...</div>
                    ::after
                  </div>
                  <p class="Pagination">...</p>
                </div>
                <div class="col-sm-4">...</div>
              </div>
            </div>
          </event>
        </div>
      </div>
    </div>
  </div>

```

Example 1: Missing persons

Sort



18-009282

Bureau Reference:

18-009282

Location

London

Gender

Male

Date found

21 September 2018

Age

30 - 40

Ethnicity

White European

View case details

```
<div>
  <div>
    <div><h1>Case Search</h1></div>
    <form>
      <input type="text" value="Case Search Existing" name="q" />
      <input type="submit" value="Search" />
    </form>
  </div>
  <div>
    <h2>Case Search Existing</h2>
    <div>
      <div>
        <h3>Bureau Reference:</h3>
        <p>18-009282</p>
        <img alt="Placeholder for a missing person image" data-bbox="365 855 515 930"/>
      </div>
      <div>
        <h3>Location</h3>
        <p>London</p>
      </div>
      <div>
        <h3>Gender</h3>
        <p>Male</p>
      </div>
      <div>
        <h3>Date found</h3>
        <p>21 September 2018</p>
      </div>
      <div>
        <h3>Age</h3>
        <p>30 - 40</p>
      </div>
      <div>
        <h3>Ethnicity</h3>
        <p>White European</p>
      </div>
      <div>
        <a href="#">View case details
      </div>
    </div>
  </div>
</div>
```

Example 2: FBI most wanted

Webscraping in a nutshell

1. understand the structure of a webpage
2. exploit that structure for web-scraping

Webscraping in practice

Case for today: Missing persons FBI

<https://www.fbi.gov/wanted/kidnap>

Explore the target page

Aims

1. Get a list of all names
2. Store the bio information
3. Extract the “details” description
4. Download the poster

Getting started

Set up your workspace first:

```
library(rvest)
```

```
## Loading required package: xml2
```

```
target_url = 'https://www.fbi.gov/wanted/kidnap'
```

Remember...

- understanding the structure of a webpage
- exploiting that structure for web-scraping

1. Get a list of all names
 - understanding the structure of a webpage*
 - <https://www.fbi.gov/wanted/kidnap>

1. Get a list of all names *exploiting that structure for web-scraping*

Access the full html page (snapshot-mode):

```
target_page = read_html(target_url)  
target_page
```

```
## {xml_document}  
## <html lang="en" data-gridsystem="bs3">  
## [1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=...</head>  
## [2] <body id="visual-portal-wrapper" class="... portal-type-folder site-...</body>
```

1. Get a list of all names

Key here: look for the `<h3>` heading with class `title`:

```
target_page %>%  
html_nodes('h3.title')
```

```
## {xml_nodeset(40)}  
## [1] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/al  
## [2] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/al  
## [3] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/je  
## [4] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/f  
## [5] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/m  
## [6] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/m  
## [7] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/a  
## [8] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/j  
## [9] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/i  
## [10] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/c  
## [11] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/l  
## [12] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/k  
## [13] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/b  
## [14] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/j  
## [15] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/s  
## [16] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/a  
## [17] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [18] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [19] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [20] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [21] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [22] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [23] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [24] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [25] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [26] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [27] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [28] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [29] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [30] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [31] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [32] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [33] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [34] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [35] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [36] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [37] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [38] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [39] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n  
## [40] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/n
```

```
#note: equivalent to num_nodes(target_page, no_true )
```

1. Get a list of all names

A closer look:

```
all_titles = target_page %>%
  html_nodes('h3.title')
```

```
all_titles[1]
```

```
## {xml_nodeset (1)}
## [1] <h3 class="title">\n<a href="https://www.fbi.gov/wanted/kidnap/ar-
```

It's the text of the ` tag.`

1. Get a list of all names

So what we want is:

1. Access the full html page
2. Search all h3 headings with class “title”
3. Find all `<a>` tags (= links)
4. Extract the text

1. Get a list of all names

1. Access the full html page `read_html(target_url)`

2. Search all h3 headings with class “title”

`html_nodes('h3.title')`

3. Find all `<a> tags (= links)` `html_nodes('a')`

4. Extract the text `html_text()`

1. Get a list of all names

Combined:

```
all_names = target_page %>%
  html_nodes('h3.title') %>%
  html_nodes('a') %>%
  html_text()
```

1. Get a list of all names

```
all_names
```

```
## [ 1 ] "ARIANNA FITTS"
## [ 2 ] "ANGELA MAE MEEKER"
## [ 3 ] "JENNIFER LYNN MARCUM"
## [ 4 ] "FELIX BATISTA"
## [ 5 ] "MARK HIMEBAUGH"
## [ 6 ] "MICHAELA JOY GARECHT"
## [ 7 ] "ASHLEY SUMMERS"
## [ 8 ] "JAYME CLOSS"
## [ 9 ] "ILENE BETH MISHELOFF"
## [ 10 ] "CARLA VICENTINI"
## [ 11 ] "LISA IRWIN"
## [ 12 ] "KYRON RICHARD HORMAN"
## [ 13 ] "BIANCA LEBRON"
## [ 14 ] "JABEZ SPANN"
## [ 15 ] "SHANNA GENELLE PEOPLES"
## [ 16 ] "ABBY LYNN PATTERSON"
## [ 17 ] "RUOCHEN LIAO"
## [ 18 ] "JOSETTA KECHARA STEPPA GARCIA"
```

2. Store the bio information
understanding the structure of a webpage
<https://www.fbi.gov/wanted/kidnap>

2. Store the bio information

We know: there's a table with class
wanted-person-description that contains the data we want.

But: we need to access each 'kidnapped' person!?

For-loops to the rescue...

2. Store the bio information *exploiting that structure for web-scraping*

So what we want is:

1. Access the full html page
2. Search all h3 headings with class “title”
3. Find all **<a > tags (= links)**
4. Extract the **text** actual link
5. Access that page
6. Extract the table with class **wanted-person-description**

2. Store the bio information

```
all_persons_links = target_page %>%
  html_nodes('h3.title') %>%
  html_nodes('a') %>%
  html_attr('href')

all_persons_links
```

```
## [1] "https://www.fbi.gov/wanted/kidnap/arianna-fitts"
## [2] "https://www.fbi.gov/wanted/kidnap/angela-mae-meeker"
## [3] "https://www.fbi.gov/wanted/kidnap/jennifer-lynn-marcum"
## [4] "https://www.fbi.gov/wanted/kidnap/felix-batista"
## [5] "https://www.fbi.gov/wanted/kidnap/mark-himebaugh"
## [6] "https://www.fbi.gov/wanted/kidnap/michaela-joy-garecht"
## [7] "https://www.fbi.gov/wanted/kidnap/ashley-summers"
## [8] "https://www.fbi.gov/wanted/kidnap/jayne-closs"
## [9] "https://www.fbi.gov/wanted/kidnap/ilene-beth-misheloff"
## [10] "https://www.fbi.gov/wanted/kidnap/carla-vicentini"
## [11] "https://www.fbi.gov/wanted/kidnap/lisa-irwin"
## [12] "https://www.fbi.gov/wanted/kidnap/kyron-richard-horman"
## [13] "https://www.fbi.gov/wanted/kidnap/bianca-lebron"
## [14] "https://www.fbi.gov/wanted/kidnap/jabbez-spann"
## [15] "https://www.fbi.gov/wanted/kidnap/shanna-genelle-peoples"
## [16] "https://www.fbi.gov/wanted/kidnap/abby-lynn-patterson"
## [17] "https://www.fbi.gov/wanted/kidnap/ruochen-liao"
## [18] "https://www.fbi.gov/wanted/kidnap/joshua-kochaba-sierra-marcia"
```

2. Store the bio information

Now what?

```
for(i in all_persons_links) {  
  print(i)  
}
```

```
## [1] "https://www.fbi.gov/wanted/kidnap/arianna-fitts"  
## [1] "https://www.fbi.gov/wanted/kidnap/angela-mae-meeker"  
## [1] "https://www.fbi.gov/wanted/kidnap/jennifer-lynn-marcum"  
## [1] "https://www.fbi.gov/wanted/kidnap/felix-batista"  
## [1] "https://www.fbi.gov/wanted/kidnap/mark-himebaugh"  
## [1] "https://www.fbi.gov/wanted/kidnap/michaela-joy-garecht"  
## [1] "https://www.fbi.gov/wanted/kidnap/ashley-summers"  
## [1] "https://www.fbi.gov/wanted/kidnap/jayme-closs"  
## [1] "https://www.fbi.gov/wanted/kidnap/ilene-beth-misheloff"  
## [1] "https://www.fbi.gov/wanted/kidnap/carla-vicentini"  
## [1] "https://www.fbi.gov/wanted/kidnap/lisa-irwin"  
## [1] "https://www.fbi.gov/wanted/kidnap/kyron-richard-horman"  
## [1] "https://www.fbi.gov/wanted/kidnap/bianca-lebron"  
## [1] "https://www.fbi.gov/wanted/kidnap/jabbez-spann"  
## [1] "https://www.fbi.gov/wanted/kidnap/shanna-genelle-peoples"  
## [1] "https://www.fbi.gov/wanted/kidnap/abby-lynn-patterson"  
## [1] "https://www.fbi.gov/wanted/kidnap/ruochen-liao"  
## [1] "https://www.fbi.gov/wanted/kidnap/ioshua-sierra-garcia"
```

2. Store the bio information

Before you write a loop...

```
arianna = all_persons_links[1]
temp_target_url = arianna
temp_target_page = read_html(temp_target_url)
```

Single-case proof.

2. Store the bio information

```
description = temp_target_page %>%
  html_nodes('table.wanted-person-description') %>%
  html_table()

description
```

```
## [1]
## 1 Date(s) of Birth Used
## 2 Place of Birth
## 3 Hair
## 4 Eyes
## 5 Height 2'0" (at time of disappearance)
## 6 Weight 45 pounds (at time of disappearance)
## 7 Sex Female
## 8 Race Black
## 9 Nationality American
```

2. Store the bio information

What we need for the for-loop:

1. do this for each link
2. store it somewhere (easiest: in a list)
3. log progress

2. Store the bio information

```
list_for_data = list()
for(i in all_persons_links) {
  print(paste('Accessing:', i))
  temp_target_url = i
  temp_target_page = read_html(temp_target_url)
  description = temp_target_page %>%
    html_nodes('table.wanted-person-description') %>%
    html_table()
  index_of_i = which(i == all_persons_links)
  list_for_data[[index_of_i]] = description
  print('--- NEXT ---')
}
```

```
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ariana-fitts"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/angela-mae-meeker"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/jennifer-lynn-marcus"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/felix-batista"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/mark-himebaugh"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/michaela-joy-garech"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ashley-summers"
```

```
## [1] "--- NEXT ---"  
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/jayme-closs"  
## [1] "--- NEXT ---"  
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ilene-beth-misheff"
```

2. Store the bio information

Now we have a list of tables.

Each table contains the details of one missing person:

```
# thirteenth element in the list  
list_for_data[[13]]
```

	X1	X2
## 1	Date(s) of Birth	Used
## 2	Hair	Long Dark Brown
## 3	Eyes	Hazel
## 4	Height	4'11"
## 5	Weight	115 pounds
## 6	Sex	Female
## 7	Race	White (Hispanic)

3. Extract the “details” description
understanding the structure of a webpage
<https://www.fbi.gov/wanted/kidnap>

3. Extract the “details” description

We want: the text of the `<div>` with class
`wanted-person-details`

exploiting that structure for web-scraping

So what we want is:

1. Access the full html page
2. Search all `h3` headings with class “title”
3. Find all `<a>` tags (= links)
4. Extract the `text` actual link
5. Access that page
6. Extract the ~~table with class~~
`wanted-person-description` `<div>` with class
`wanted-person-details`

3. Extract the “details” description

Start with single-case proof:

```
arianna = all_persons_links[1]
temp_target_url = arianna
temp_target_page = read_html(temp_target_url)

person_details = temp_target_page %>%
  html_nodes('div.wanted-person-details') %>%
  html_text()
```

3. Extract the “details” description

```
person_details
```

```
#> [1] "\nDetails:\nArianna Fitts was reported missing from the San Fran-
```

3. Extract the “details” description

Putting it in a loop:

```
list_for_person_details = list()
for(i in all_persons_links) {
  print(paste('Accessing: ', i))
  temp_target_url = i
  temp_target_page = read_html(temp_target_url)
  person_details = temp_target_page %>%
    html_nodes('div.wanted-person-details') %>%
    html_text()
  index_of_i = which(i == all_persons_links)
  list_for_person_details[[index_of_i]] = person_details
  print('--- NEXT ---')
}
```

```
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ariana-fitts"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/angela-mae-meeker"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/jennifer-lynn-marcus"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/felix-batista"
## [1] "--- NEXT ---"
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/mark-himebaugh"
## [1] "--- NEXT ---"
```

```
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/michaela-joy-garech-  
## [1] "---- NEXT ----"  
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ashley-summers"  
## [1] "---- NEXT ----"  
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/jayme-closs"  
## [1] "---- NEXT ----"  
## [1] "Accessing: https://www.fbi.gov/wanted/kidnap/ilene-beth-mishelof"
```

3. Extract the “details” description

```
#look at the 20th element  
list_for_person_details[[20]]  
  
## [1] "\nDetails:\nErica Nicole Hunt was last seen on July 4, 2016, nea:
```

4. Download the poster
understanding the structure of a webpage
<https://www.fbi.gov/wanted/kidnap>

4. Download the poster

What do we want?

understanding the structure of a webpage

Each kidnapped person has a ‘download link’...

<https://www.fbi.gov/wanted/kidnap/ariana-fitts/download.pdf>

Don’t overthink it!

4. Download the poster

Compare these two:

```
https://www.fbi.gov/wanted/kidnap/arianna-fitts/download.pdf
```

```
arianna
```

```
## [1] "https://www.fbi.gov/wanted/kidnap/arianna-fitts"
```

Notice something?

4. Download the poster

We can just ‘work around’ this:

```
download_url_arianna = paste(tolower(arianna), '/download.pdf', sep="")  
download_url_arianna
```

```
#> [1] "https://www.fbi.gov/wanted/kidnap/ariana-fitts/download.pdf"
```

<https://www.fbi.gov/wanted/kidnap/ariana-fitts/download.pdf>

4. Download the poster

Make use of R's vectorised structure:

```
all_download_links = paste(toupper(all_persons_links), '/download.pdf',  
all_download_links)
```

```
## [1] "https://www.fbi.gov/wanted/kidnap/arianna-fitts/download.pdf"  
## [2] "https://www.fbi.gov/wanted/kidnap/angela-mae-meeker/download.pdf"  
## [3] "https://www.fbi.gov/wanted/kidnap/jennifer-lynn-marcum/download.pdf"  
## [4] "https://www.fbi.gov/wanted/kidnap/felix-batista/download.pdf"  
## [5] "https://www.fbi.gov/wanted/kidnap/mark-himebaugh/download.pdf"  
## [6] "https://www.fbi.gov/wanted/kidnap/michaela-joy-garecht/download.pdf"  
## [7] "https://www.fbi.gov/wanted/kidnap/ashley-summers/download.pdf"  
## [8] "https://www.fbi.gov/wanted/kidnap/jayme-closs/download.pdf"  
## [9] "https://www.fbi.gov/wanted/kidnap/ilene-beth-misheloff/download.pdf"  
## [10] "https://www.fbi.gov/wanted/kidnap/carla-vicentini/download.pdf"  
## [11] "https://www.fbi.gov/wanted/kidnap/lisa-irwin/download.pdf"  
## [12] "https://www.fbi.gov/wanted/kidnap/kyron-richard-horman/download.pdf"  
## [13] "https://www.fbi.gov/wanted/kidnap/bianca-lebron/download.pdf"  
## [14] "https://www.fbi.gov/wanted/kidnap/jabez-spann/download.pdf"  
## [15] "https://www.fbi.gov/wanted/kidnap/shanna-genelle-peoples/download.pdf"  
## [16] "https://www.fbi.gov/wanted/kidnap/abby-lynn-patterson/download.pdf"  
## [17] "https://www.fbi.gov/wanted/kidnap/ruochen-liao/download.pdf"  
## [18] "https://www.fbi.gov/wanted/kidnap/rochana-sierra-garcia/
```

4. Download the poster

Now:

1. access each
2. “download” (= write) the file
 - needs a filename on your computer

4. Download the poster

Create filenames:

```
library(stringr)
file_names = paste(all_names, '.pdf', sep="")
```

```
## [ 1 ] "ARIANNA FITTS.pdf"
## [ 2 ] "ANGELA MAE MEEKER.pdf"
## [ 3 ] "JENNIFER LYNN MARCUM.pdf"
## [ 4 ] "FELIX BATISTA.pdf"
## [ 5 ] "MARK HIMEBAUGH.pdf"
## [ 6 ] "MICHAELA JOY GARECHT.pdf"
## [ 7 ] "ASHLEY SUMMERS.pdf"
## [ 8 ] "JAYME CLOSS.pdf"
## [ 9 ] "ILENE BETH MISHELOFF.pdf"
## [10 ] "CARLA VICENTINI.pdf"
## [11 ] "LISA IRWIN.pdf"
## [12 ] "KYRON RICHARD HORMAN.pdf"
## [13 ] "BIANCA LEBRON.pdf"
## [14 ] "JABEZ SPANN.pdf"
## [15 ] "SHANNA GENELLE PEOPLES.pdf"
## [16 ] "ABBY LYNN PATTERSON.pdf"
## [17 ] "RUOCHEN LIAO.pdf"
## [18 ] "JOSHUA KEISHARA STEPPA GARCIA.pdf"
```

4. Download the poster

Refine filenames:

```
refined_file_names = tolower(str_replace_all(string = file_names, pattern =
```

```
## [1] "arianna_fitts.pdf"
## [2] "angela_mae_meeker.pdf"
## [3] "jennifer_lynn_marcum.pdf"
## [4] "felix_batista.pdf"
## [5] "mark_himebaugh.pdf"
## [6] "michaela_joy_garecht.pdf"
## [7] "ashley_summers.pdf"
## [8] "jayme_closs.pdf"
## [9] "ilene_beth_misheloff.pdf"
## [10] "carla_vicentini.pdf"
## [11] "lisa_irwin.pdf"
## [12] "kyron_richard_horman.pdf"
## [13] "bianca_lebron.pdf"
## [14] "jabez_spann.pdf"
## [15] "shanna_genelle_peoples.pdf"
## [16] "abby_lynn_patterson.pdf"
## [17] "ruochen_liao.pdf"
## [18] "joshua_keshaba_sierra_garcia.pdf"
```

4. Download the poster

Download each pdf from the url and use the refined filenames

```
for(i in 1:length(all_download_links)) {  
  print(paste('Accessing URL:', all_download_links[i], sep=""))  
  download.file(url = all_download_links[i]  
                , destfile = refined_file_names[i]  
                , mode = "wb")  
}
```

```
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/arianna-fitts/d  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/angela-mae-meek  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/jennifer-lynn-m  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/felix-batista/d  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/mark-himebaugh/  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/michaela-joy-gai  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/ashley-summers/  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/jayme-closs/down  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/ilene-beth-mish  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/carla-vicentini  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/kyron-richard-h  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/bianca-lebron/d  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/jabez-spann/dow  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/shanna-genelle-l  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/abby-lynn-patte  
## [1] "Accessing URL: https://www.fbi.gov/wanted/kidnap/ruochen-liao/do
```

Notes on webscraping

- highly customisable (= juicy data)
- basically: “anything goes”
- can be unstable/sensitive to html changes

What this means

Same idea, different code details:

The screenshot shows a search results page for "TV receiver" on eBay. The page layout includes a header with navigation links like "Sell | My eBay" and "Help & Contact". Below the header, there are sections for "Categories" (All, Consumer Electronics, TV, Video & Home Audio, TV & Video), "Satellite TV Receivers" (listing a product), "Cable TV Boxes" (listing a product), "DVRs, Hard Drive Recorders" (listing a product), "TVs" (listing a product), "DVD & Blu-ray Players" (listing a product), "Computers/Tablets & Networking" (listing a product), "eBay Motors" (listing a product), and "Show More ▾" (link). On the right side, there are filters for "Type" (Analog, HD Digital, Standard Digital, Not Specified), "Brand" (DIRECTV, DISH Network, Dreambox, Humax), and "Feedback" (3 Watching). The bottom of the page shows the eBay footer with links like "Feedback", "Pseudo-elements", "Layout", "Computed", "Animations", "Fonts", "Filter Styles", "Background-color", and "Browser styles". The entire page is framed by a large black border.

What this means

Same idea, different code details:

```
<div>
  <div>
    <div>
      <div>View case details</div>
      <div>Bureau: Bureau</div>
      <div>Reference: Reference</div>
      <div>Location: Location</div>
      <div>Gender: Gender</div>
      <div>Date found: Date found</div>
      <div>Age: Age</div>
      <div>Ethnicity: Ethnicity</div>
      <div>View case details</div>
    </div>
  </div>
</div>
```

```
<div>
  <div>
    <div>
      <div>View case details</div>
      <div>Bureau: Bureau</div>
      <div>Reference: Reference</div>
      <div>Location: Location</div>
      <div>Gender: Gender</div>
      <div>Date found: Date found</div>
      <div>Age: Age</div>
      <div>Ethnicity: Ethnicity</div>
      <div>View case details</div>
    </div>
  </div>
</div>
```

The screenshot shows two rows of case details. The first row has a blue background and the second has a yellow background. Each row contains a 'View case details' button, followed by fields for Bureau, Reference, Location, Gender, Date found, Age, and Ethnicity, and another 'View case details' button at the bottom. There are also thumbnail images of people in each row.

HTML structure (highlighted with blue and yellow boxes):

```
<div>
  <div>
    <div>
      <div>View case details</div>
      <div>Bureau: Bureau</div>
      <div>Reference: Reference</div>
      <div>Location: Location</div>
      <div>Gender: Gender</div>
      <div>Date found: Date found</div>
      <div>Age: Age</div>
      <div>Ethnicity: Ethnicity</div>
      <div>View case details</div>
    </div>
  </div>
</div>
```

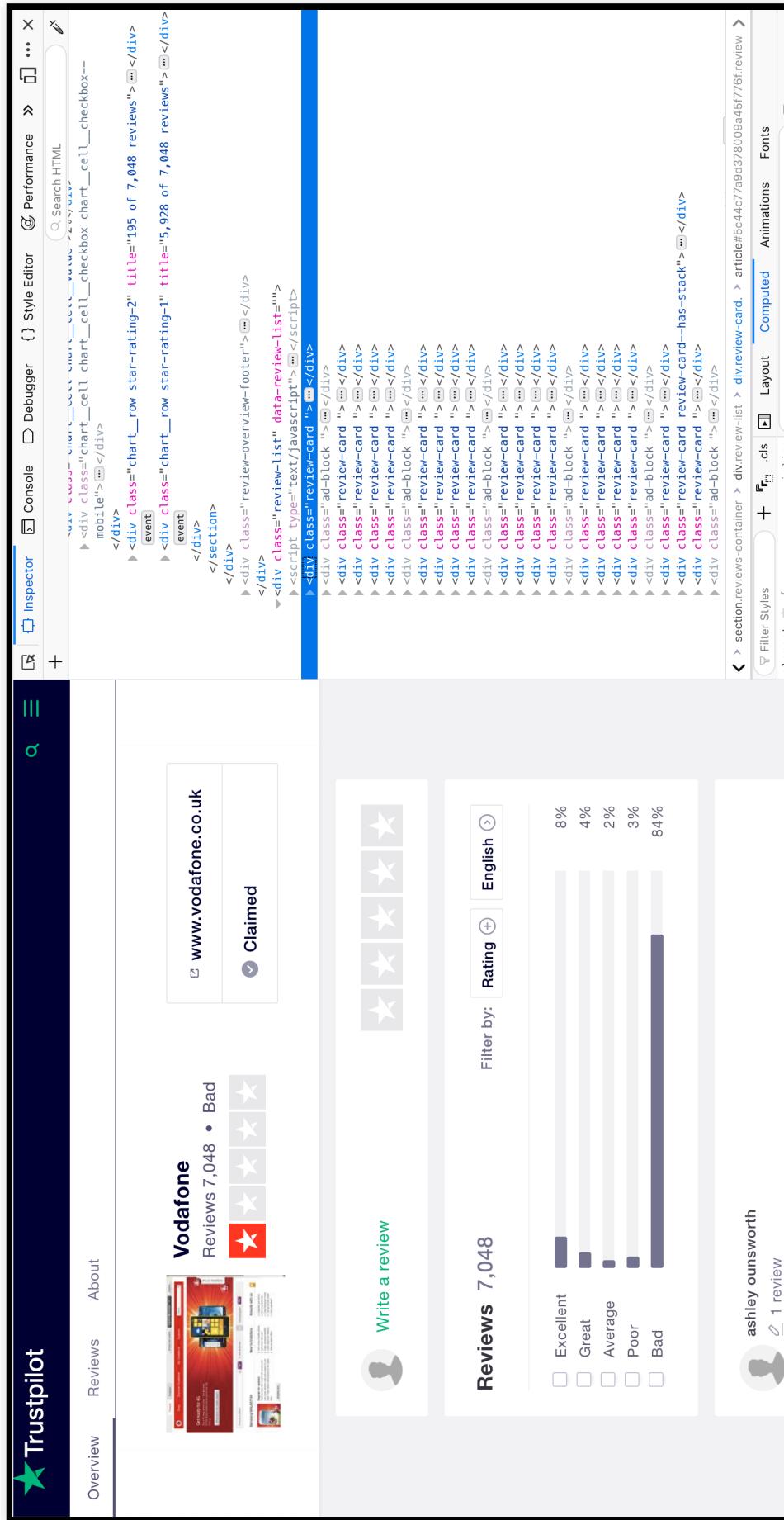
Code editor view (highlighted with blue and yellow boxes):

```
<div>
  <div>
    <div>
      <div>View case details</div>
      <div>Bureau: Bureau</div>
      <div>Reference: Reference</div>
      <div>Location: Location</div>
      <div>Gender: Gender</div>
      <div>Date found: Date found</div>
      <div>Age: Age</div>
      <div>Ethnicity: Ethnicity</div>
      <div>View case details</div>
    </div>
  </div>
</div>
```

The code editor view shows the same HTML structure as the screenshot, with the entire row highlighted in blue or yellow. The code is annotated with various CSS classes and JavaScript events, such as 'Search HTML', 'method="post"', 'action="/en-gb/case-search/p24442"', and various 'Case' and 'Modal' classes.

What this means

Same idea, different code details:



The screenshot shows a Trustpilot review page for Vodafone. At the top, there's a navigation bar with links for Overview, Reviews, and About. Below the navigation is a search bar and a 'Style Editor' toolbar.

The main content area displays a 4-star rating (4.0 out of 5), 7,048 reviews, and a 'Bad' category. A 'Claimed' badge is visible. Below this, there's a 'Write a review' button and a 'Reviews 7,048' section with a star distribution chart.

The 'Style Editor' toolbar at the top has several tabs: Inspector, Console, Debugger, Style Editor, Performance, and more. The 'Inspector' tab is active, showing the DOM structure of the page. The DOM tree starts with a `<div>` element with the class `chart_cell_checkbox chart_cell_checkbox_mobile`. It contains `<div>` elements for 'star-rating-2' and 'star-rating-1'. The 'star-rating-2' div has a title of '195 of 7,048 reviews' and the 'star-rating-1' div has a title of '5,928 of 7,048 reviews'.

The bottom right corner of the screenshot shows a user profile for 'ashley ownsworth' with 1 review.

What this means

Same idea, different code details:

Name: (NA) - NA
Email: Email Seller
Location: Michigan
Website: NA

We have available one tame hand raised baby female Kinkajou, perfect as an educational ambassador or pet. We are USDA licensed, serious and educated inquiries only please.

[View Details](#)



Adult Kinkajous

Name: CIG EXOTICS [View Profile](#)
Posted: 1/20/2019
Phone: (NA) - NA
Email: Email Seller
Location: Michigan
Website: NA

We have available several unrelated adult Kinkajou pairs and trios. These would make an excellent breeding project or zoo exhibit. These are not tame and not pets, we are USDA licensed.

[View Details](#)

HTML Body Div #header Div Div Collect Div Row Div Row Div 5U12u\$(medium)

Styles Event Listeners DOM Breakpoints Properties Accessibility

Filter : hover .cls + ↻

⋮ Console Animations What's New X

RECAP

- Always: problem first, never the method first!
- Method follows problem!
- HTML structure key to ‘real’ webscraping
- Webscraping:
 - understanding the structure of a webpage
 - exploiting that structure for web-scraping
- principle is always the same: understand + exploit the html structure

Outlook

Tutorial: APIs + Webscraping in R

Homework: Do the readings for today (blogposts/guides)

Next week: Text data 1

END