Web Scraping

Gathering Data from the Web

Web Scraping!

You may want to…

download all videos from a website;
download all news articles from a media platform;
download all academic papers from a journal;
download all tweets/weibo of a specific person.

You may need to spend days and nights downloading these data manually, and you can easily make a lot of mistakes.

Web Scraping!

In today's class, we are going to learn about webscraping.
In Chinese, it is called 网络爬虫.

What is webscraping?
Using tools to gather data you can see on a webpage.
Almost anything you see on a website can be scraped.

It can be done with python, R,… We are doing it on R.

WHAT IS
WEB
SCRAPING?

Learning about HTML

HTML: HyperText Markup Language.

Websites are written on the HTML language.

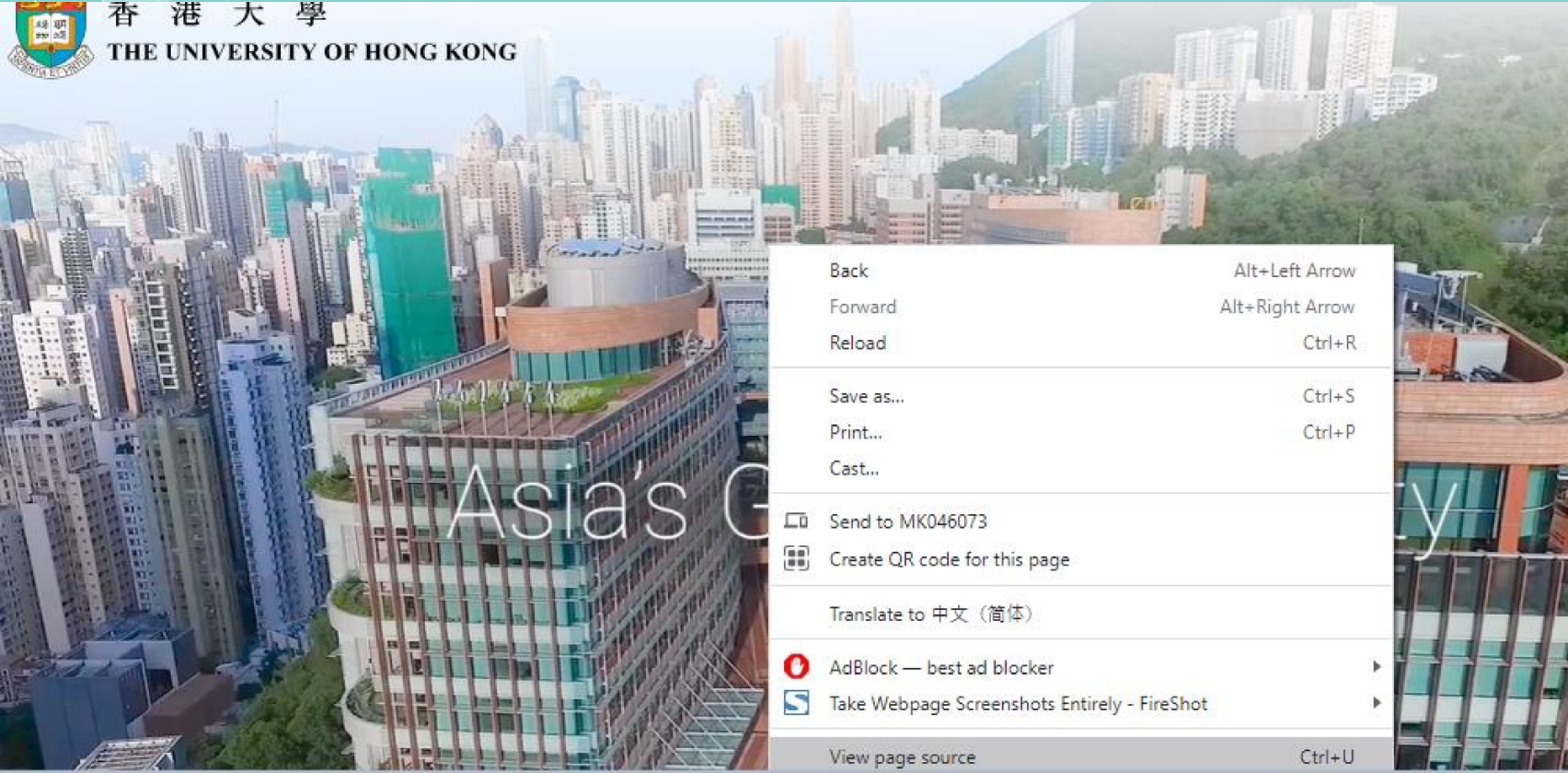
Webscraping is based on reading and interpreting the HTML of a webpage.

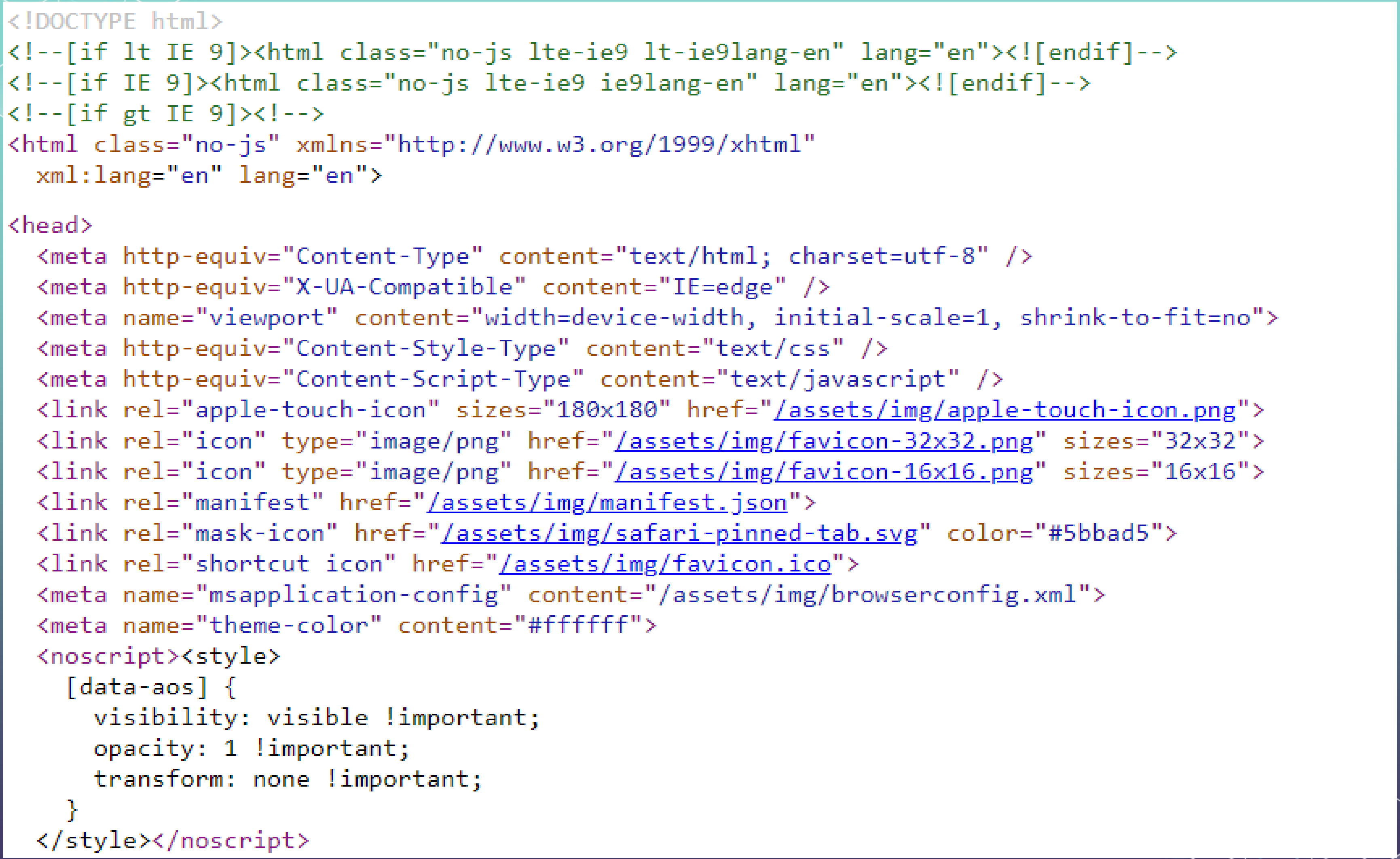
But how to find the HTML of a webpage?

Learning about HTML

Please use Chrome as your browser.

If you are not using Chrome, please download and install one now.


香 港 大 學
THE UNIVERSITY OF HONG KONG
Asia's G
y
Back
Alt+Left Arrow
Forward
Alt+Right Arrow
Reload
Ctrl+R
Save as...
Ctrl+S
Print...
Ctrl+P
Cast...
Send to MK046073
Create QR code for this page
Translate to 中文（简体）
AdBlock — best ad blocker
Take Webpage Screenshots Entirely - FireShot
View page source
Ctrl+U

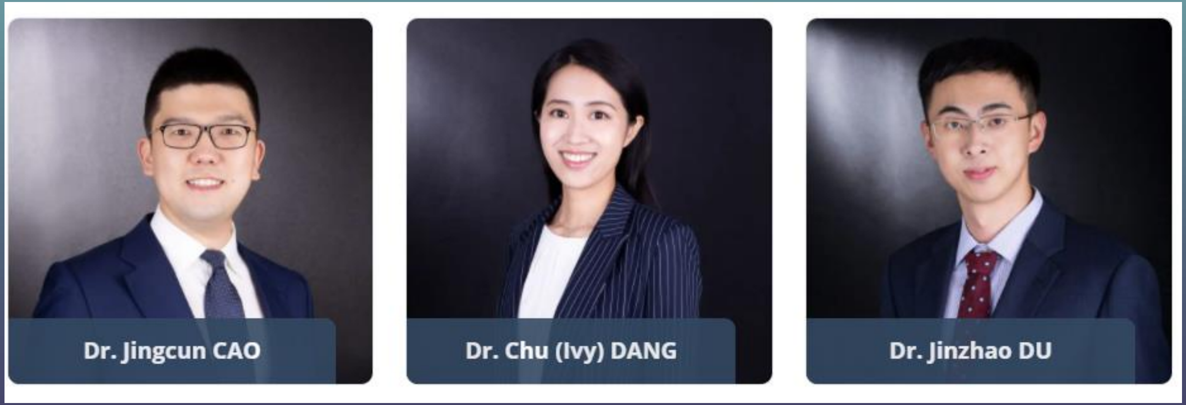


```
<!DOCTYPE html>
<!--[if lt IE 9]><html class="no-js lte-ie9 lt-ie9lang-en" lang="en"><![endif]-->
<!--[if IE 9]><html class="no-js lte-ie9 ie9lang-en" lang="en"><![endif]-->
<!--[if gt IE 9]><!-->
<html class="no-js" xmlns="http://www.w3.org/1999/xhtml"
  xml:lang="en" lang="en">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="Content-Style-Type" content="text/css" />
  <meta http-equiv="Content-Script-Type" content="text/javascript" />
  <link rel="apple-touch-icon" sizes="180x180" href="/assets/img/apple-touch-icon.png">
  <link rel="icon" type="image/png" href="/assets/img/favicon-32x32.png" sizes="32x32">
  <link rel="icon" type="image/png" href="/assets/img/favicon-16x16.png" sizes="16x16">
  <link rel="manifest" href="/assets/img/manifest.json">
  <link rel="mask-icon" href="/assets/img/safari-pinned-tab.svg" color="#5bbad5">
  <link rel="shortcut icon" href="/assets/img/favicon.ico">
  <meta name="msapplication-config" content="/assets/img/browserconfig.xml">
  <meta name="theme-color" content="#ffffff">
  <noscript><style>
    [data-aos] {
      visibility: visible !important;
      opacity: 1 !important;
      transform: none !important;
    }
  </style></noscript>
```

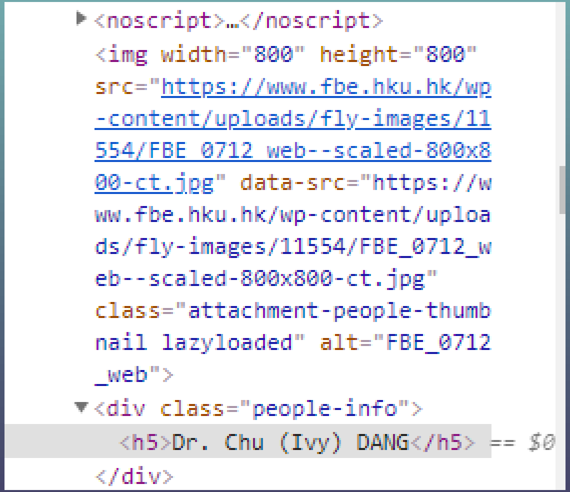
Learning about HTML

The data you want to scrape appears in certain place of the HTML. For example, suppose that you want to scrape data from the HKU marketing faculty webpage:



Learning about HTML

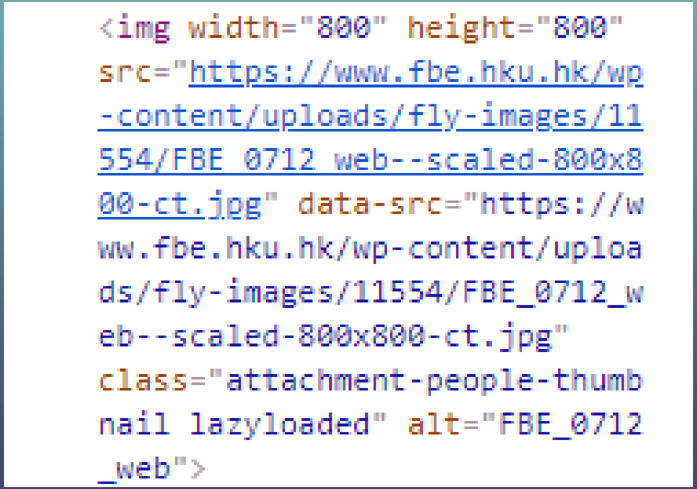
You can find the name and images of the professors from the HTML file:



```
▶<noscript>…</noscript>
 <img width="800" height="800"
 src="https://www.fbe.hku.hk/wp
 -content/uploads/fly-images/11
 554/FBE_0712_web--scaled-800x8
 00-ct.jpg" data-src="https://w
 ww.fbe.hku.hk/wp-content/uploa
 ds/fly-images/11554/FBE_0712_w
 eb--scaled-800x800-ct.jpg"
 class="attachment-people-thumb
 nail lazyloaded" alt="FBE_0712
 _web">
▼<div class="people-info">
   <h5>Dr. Chu (Ivy) DANG</h5> == $0
 </div>
```

Learning about HTML

For example, it provides you with the link to their profile photos:



```
<img width="800" height="800"
src="https://www.fbe.hku.hk/wp
-content/uploads/fly-images/11
554/FBE_0712_web--scaled-800x8
00-ct.jpg" data-src="https://w
ww.fbe.hku.hk/wp-content/uploa
ds/fly-images/11554/FBE_0712_w
eb--scaled-800x800-ct.jpg"
class="attachment-people-thumb
nail lazyloaded" alt="FBE_0712
_web">
```

Webscraping

Suppose that you want to download the names of each individual marketing faculty, what should you do?

First, you need to get the HTML for the webpage.

Second, you need to analyze the HTML to get the desired information --- this is much more difficult.

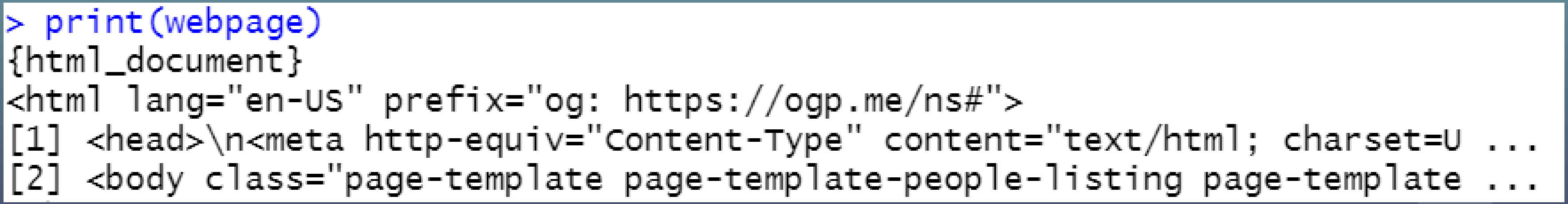
Webscraping

```
install.packages("rvest")
library(rvest)

url =
"https://www.fbe.hku.hk/people/faculty?pg=1&s
taff_type=faculty&subject_area=marketing&trac
k=all"
webpage = read_html(url, encoding = "UTF-8")
print(webpage)
```

Webscraping

Now, you get the HTML source file here. The next thing you need to do it to understand the HTML file, which is very challenging.



```
> print(webpage)
{html_document}
<html lang="en-US" prefix="og: https://ogp.me/ns#">
[1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=U ...
[2] <body class="page-template page-template-people-listing page-template ...
```

Webscraping

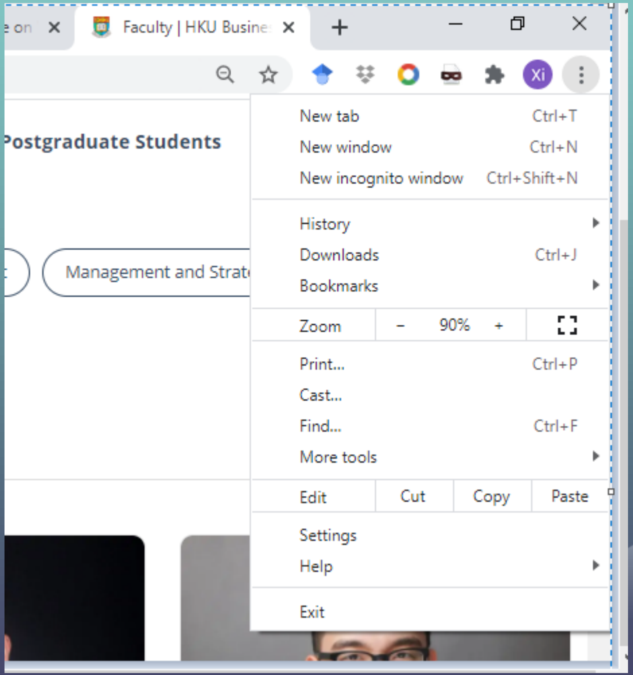
To better understand the HTML code, you are strongly recommended to use Chrome as your browser.

Chrome allows you to check the HTML code in a convenient matter.

Check HTML with Chrome

Open the webpage in your Chrome browser.

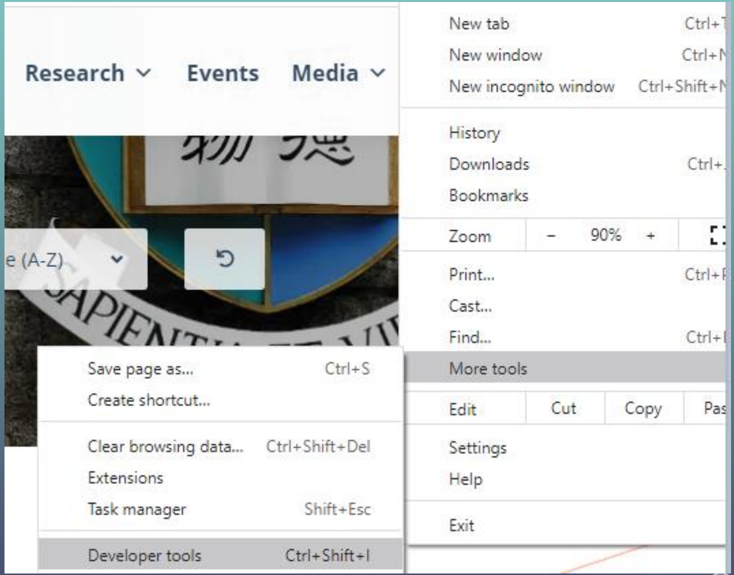
Click the upper right Chome setting button of your browser and you will be directed here.



Check HTML with Chrome

Choose “More tools”…

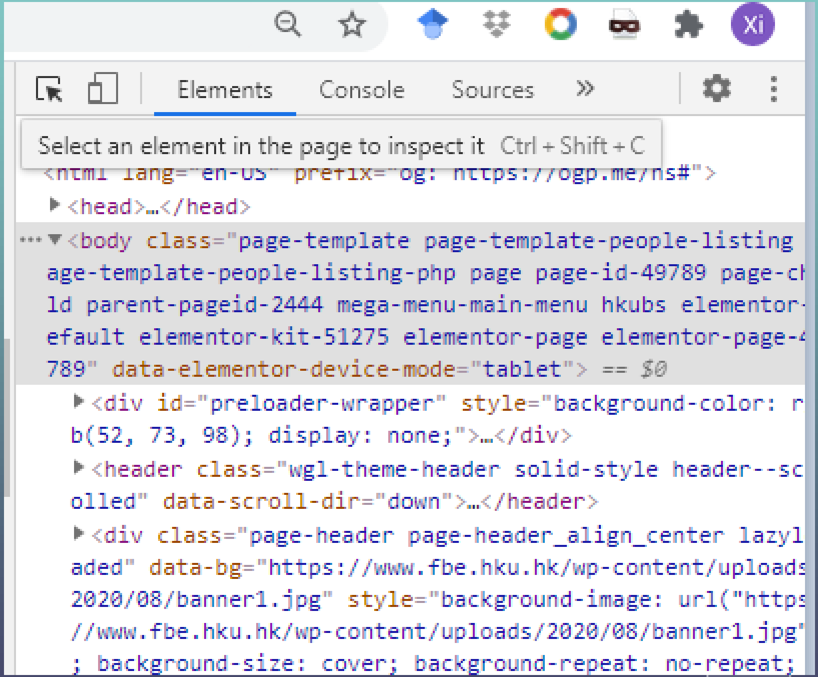
Choose “Developer tools”…



Check HTML with Chrome

Click the button and you will get to “select an element in the page to inspect it”.

Alternatively, use “Ctrl + Shift + C”


Elements
Console
Sources
Select an element in the page to inspect it Ctrl + Shift + C

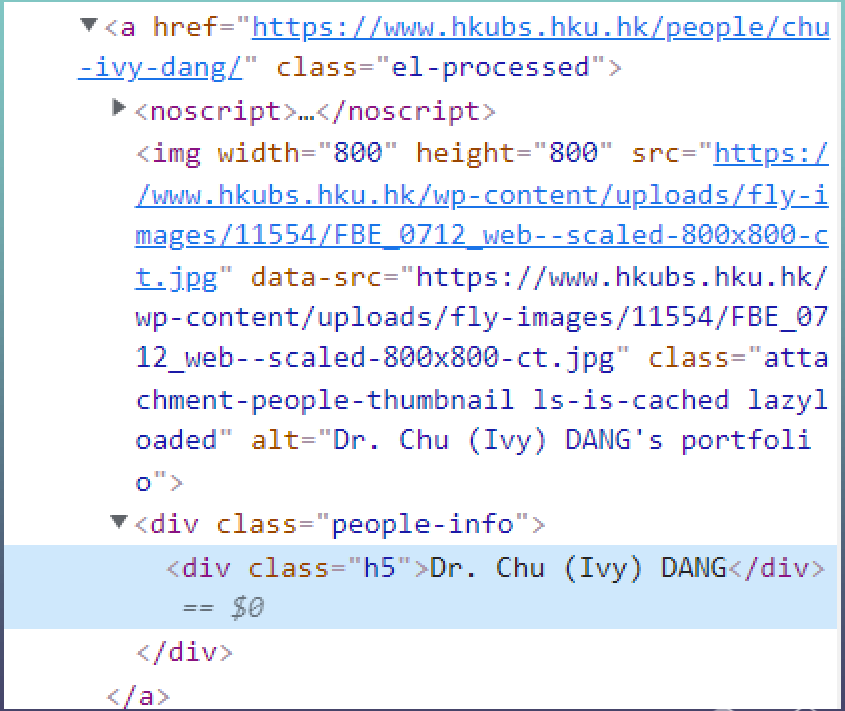
```
<html lang="en-US" prefix="og: https://ogp.me/ns#">
 ▶<head>…</head>
…▼<body class="page-template page-template-people-listing
  age-template-people-listing-php page page-id-49789 page-ch
  ld parent-pageid-2444 mega-menu-main-menu hkubs elementor-
  efault elementor-kit-51275 elementor-page elementor-page-4
  789" data-elementor-device-mode="tablet"> == $0
   ▶<div id="preloader-wrapper" style="background-color: r
   b(52, 73, 98); display: none;">…</div>
   ▶<header class="wgl-theme-header solid-style header--sc
   olled" data-scroll-dir="down">…</header>
   ▶<div class="page-header page-header_align_center lazyl
   aded" data-bg="https://www.fbe.hku.hk/wp-content/uploads
   2020/08/banner1.jpg" style="background-image: url("https
   //www.fbe.hku.hk/wp-content/uploads/2020/08/banner1.jpg"
   ; background-size: cover; background-repeat: no-repeat;
```

Check HTML with Chrome

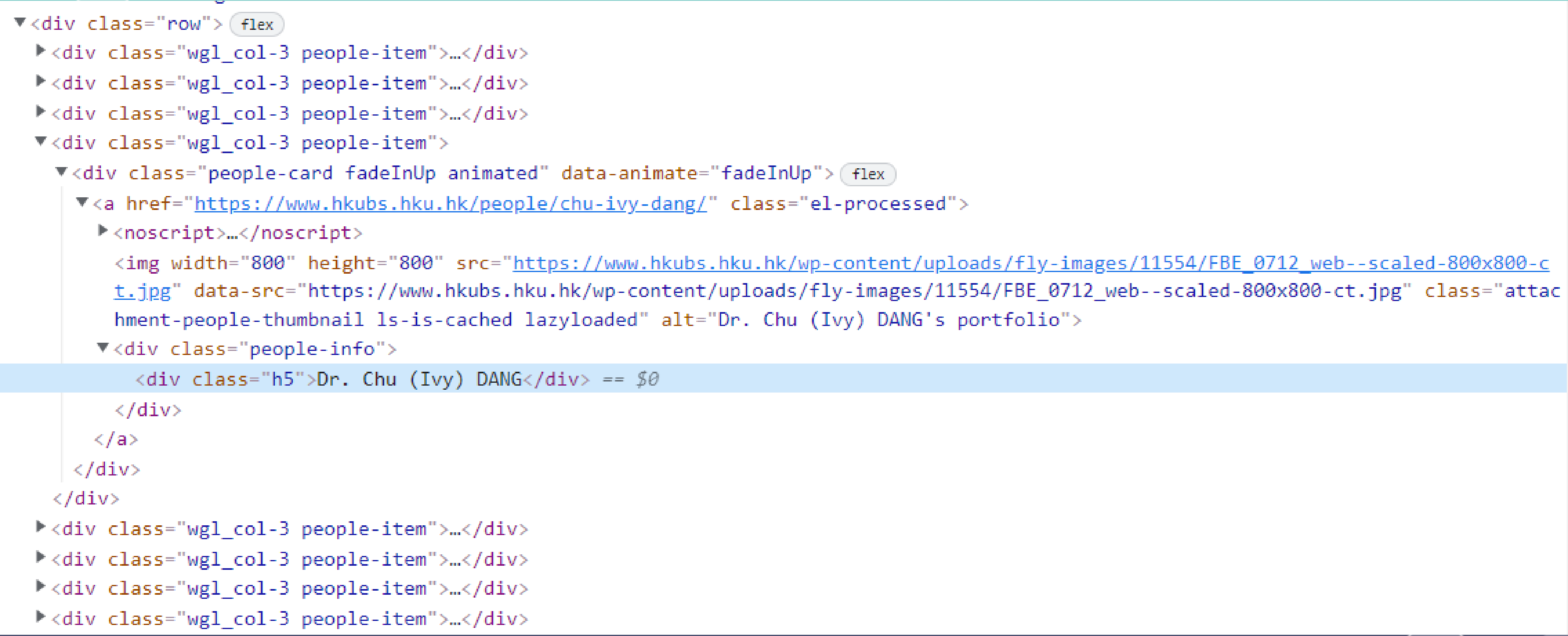
Take Prof. Dang's information as an example.

You can see her name appears here in the HTML code.

But what does this mean?



```
▼<a href="https://www.hkubs.hku.hk/people/chu-ivy-dang/" class="el-processed">
  ▶<noscript>…</noscript>
   <img width="800" height="800" src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/11554/FBE_0712_web--scaled-800x800-ct.jpg" data-src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/11554/FBE_0712_web--scaled-800x800-ct.jpg" class="attachment-people-thumbnail ls-is-cached lazyloaded" alt="Dr. Chu (Ivy) DANG's portfolio">
  ▼<div class="people-info">
     <div class="h5">Dr. Chu (Ivy) DANG</div> == $0
   </div>
 </a>
```



```
▼<div class="row"> flex
  ▶<div class="wgl_col-3 people-item">…</div>
  ▶<div class="wgl_col-3 people-item">…</div>
  ▶<div class="wgl_col-3 people-item">…</div>
  ▼<div class="wgl_col-3 people-item">
    ▼<div class="people-card fadeInUp animated" data-animate="fadeInUp"> flex
      ▼<a href="https://www.hkubs.hku.hk/people/chu-ivy-dang/" class="el-processed">
        ▶<noscript>…</noscript>
         <img width="800" height="800" src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/11554/FBE_0712_web--scaled-800x800-c
         t.jpg" data-src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/11554/FBE_0712_web--scaled-800x800-ct.jpg" class="attac
         hment-people-thumbnail ls-is-cached lazyloaded" alt="Dr. Chu (Ivy) DANG's portfolio">
        ▼<div class="people-info">
           <div class="h5">Dr. Chu (Ivy) DANG</div> == $0
         </div>
       </a>
     </div>
   </div>
  ▶<div class="wgl_col-3 people-item">…</div>
  ▶<div class="wgl_col-3 people-item">…</div>
  ▶<div class="wgl_col-3 people-item">…</div>
  ▶<div class="wgl_col-3 people-item">…</div>
```

WHAT IS HTML

WHAT DOES IT DO & WHAT IS IT USED FOR?

UNDERSTANDING HTML

Here, the name information is within an “div” node.

And this node belongs to a “div” node.

This “div” node further belongs to another “a” node.

And so on….

We call this is “path”: …div/div/div/a/div/div

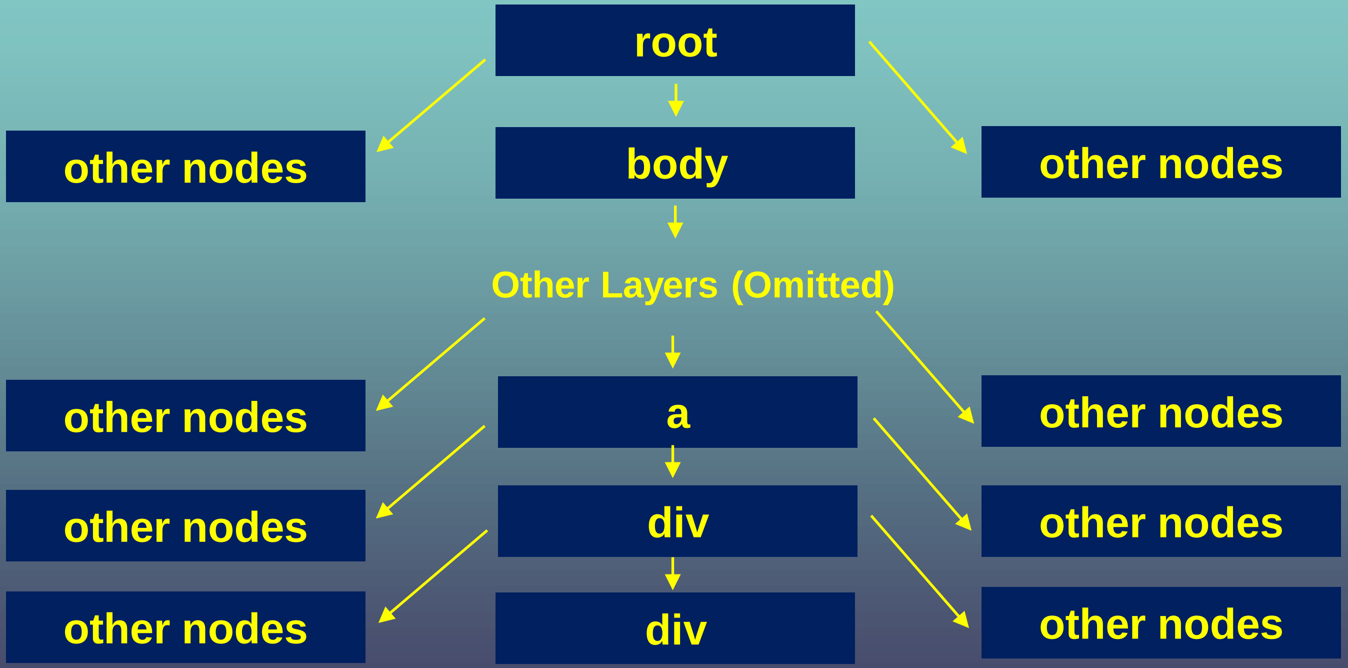
UNDERSTANDING HTML

You can see that we have various types of nodes, including "div", "a", and "img". You may wonder, "what do these types mean?"

Here, these types are called "tag". For example, an "img" tag is used to mark up an image in the HTML language.

For detailed information, check here.

UNDERSTANDING HTML



UNDERSTANDING HTML

This is something like your home address:

We have something like…
Country/Province/City/District/Street/Building/Floor/Room

The path helps us locate nodes and find the content of the nodes.

UNDERSTANDING HTML

However, unlike your home address, here each node does not have its name.

For example, we know it is an "div" node (not an "a" node) but there may be multiple "div" nodes.

My building is in a street (not an avenue or road) but there may be multiple streets here.

UNDERSTANDING HTML

Let’s get all “h5” nodes. This can be done by running this:

```
nodes <- html_nodes(webpage,xpath = '//div')
```

You can see that in total we have 274 “div” nodes.

```
print(length(nodes))
```

UNDERSTANDING HTML

We want to make the path more accurate to pin down to the “div” nodes that we are interested in. That is, we want to remove other unrelated “div” nodes.

We can do this by putting more restrictions on the path.

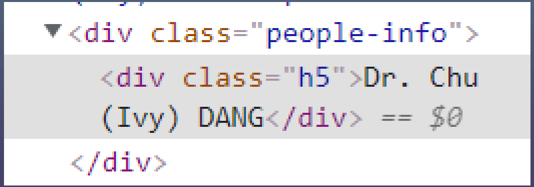
UNDERSTANDING HTML

Tags, Attributes and Elements

UNDERSTANDING HTML

```
nodes <- html_nodes(webpage,xpath =
'//div/div')
```

Here we restrict the parent of the "div" node must also be a "div" node. Now, we have 219 nodes --- still too many unrelated nodes.

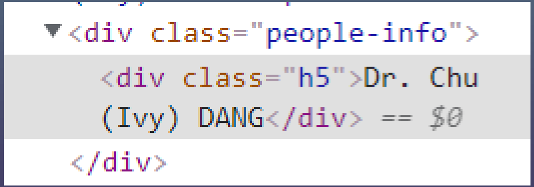


```
▼<div class="people-info">
    <div class="h5">Dr. Chu
    (Ivy) DANG</div> == $0
  </div>
```

UNDERSTANDING HTML

```
nodes <- html_nodes(webpage,xpath =
'//div[@class="people-info"]/div')
```

Here we restrict the parent of the “div” node must also be a “div” node. Moreover, the its parent node must have a class attribute will is called “people-info.”



```
▼<div class="people-info">
   <div class="h5">Dr. Chu
   (Ivy) DANG</div> == $0
 </div>
```

UNDERSTANDING HTML

Now, we only have 15 div nodes selected. These are actually all HKU marketing faculties. Let us print their names:

```
nodes <- html_nodes(webpage,xpath =
'//div[@class="people-info"]/div')
for (node in nodes)
  print(html_text(node))
```

UNDERSTANDING HTML

You can also use other refinement to select the nodes that you are looking for. For example, the following codes work as well:

```
nodes <- html_nodes(webpage,xpath =
'//div[@class="h5"]')
for (node in nodes)
  print(html_text(node))
```

Exercise

Great! You know have a sense of how to scrape data from the web. It is very preliminary, and you will need a lot more exercises. Let us try the following exercise.

Scraping Exercise

In marketing, the most premier academic journal is *Marketing Science*. It covers the latest, most important progress in the marketing community. Let us try to scrape data from it.

Exercise

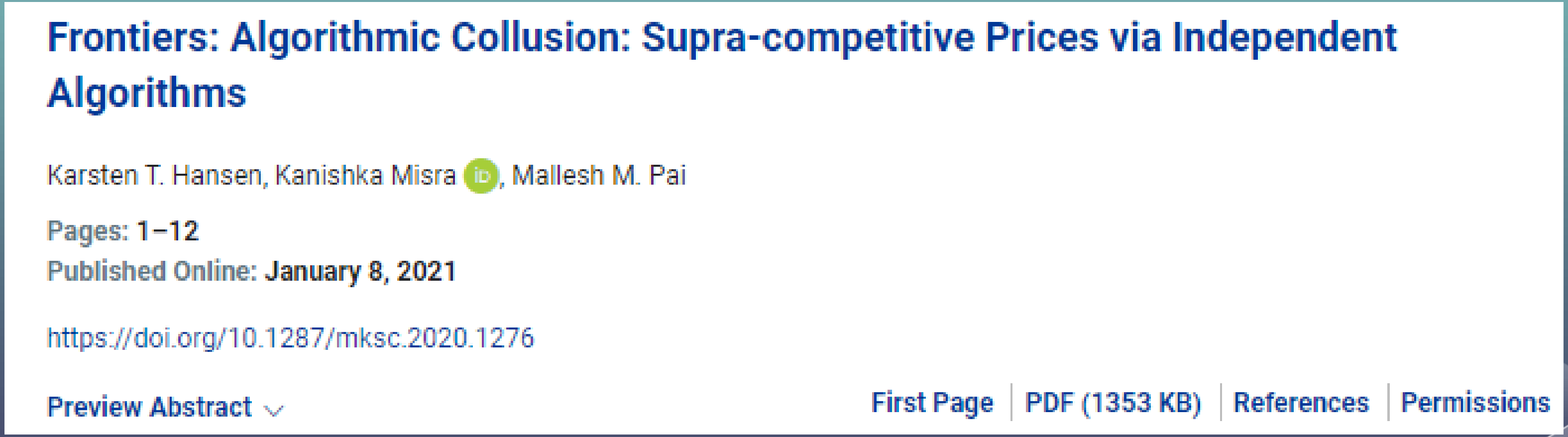
Each year, Marketing Science publishes 6 issues. We take its first issue in 2021 as an example. The URL for the issue is here:

https://pubsonline.informs.org/toc/mksc/40/1

You can see 10 articles in this issue. We want to download the information about these 10 articles.

Exercise

Let’s try to scrape the title information:



Frontiers: Algorithmic Collusion: Supra-competitive Prices via Independent Algorithms

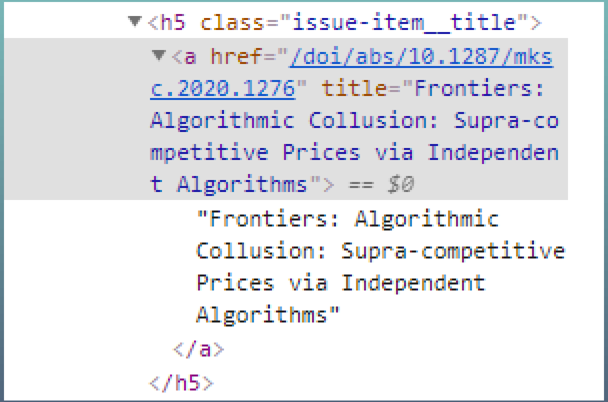
Karsten T. Hansen, Kanishka Misra, Mallesh M. Pai

Pages: **1–12**
Published Online: **January 8, 2021**

https://doi.org/10.1287/mksc.2020.1276

Exercise

First, let us scrape the titles. We must understand the corresponding HTML code to scrape the data.



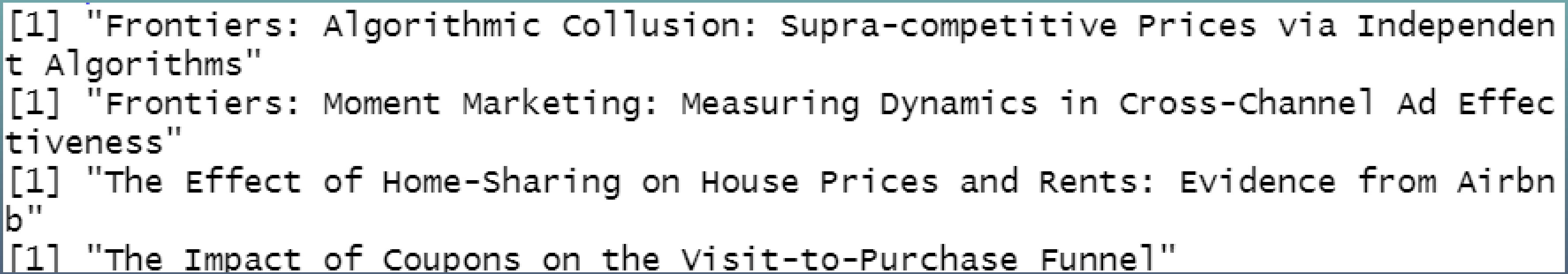
```
▼<h5 class="issue-item__title">
  ▼<a href="/doi/abs/10.1287/mksc.2020.1276" title="Frontiers: Algorithmic Collusion: Supra-competitive Prices via Independent Algorithms"> == $0
      "Frontiers: Algorithmic
      Collusion: Supra-competitive
      Prices via Independent
      Algorithms"
    </a>
  </h5>
```

Exercise

```r
library(rvest)
url =
"https://pubsonline.informs.org/toc/mksc/40/1"
webpage = read_html(url, encoding = "UTF-8")
nodes <- html_nodes(webpage,xpath =
'//h5[@class="issue-item__title"]/a')
for (node in nodes)
  print(html_text(node))
```

Exercise

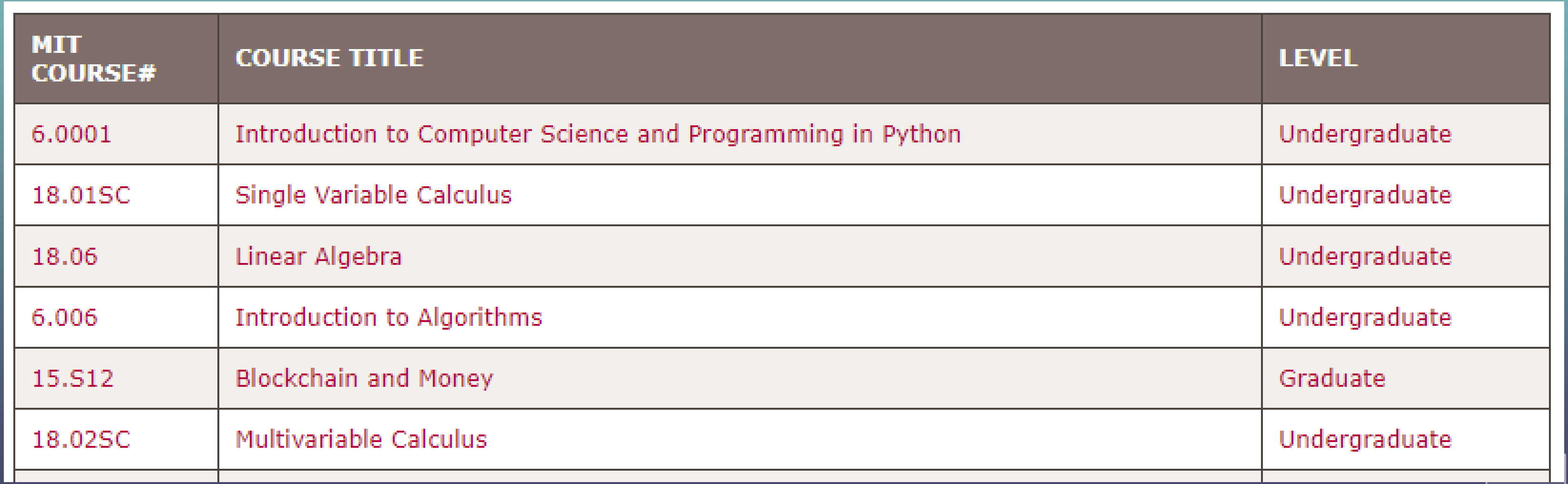
Now, we are done!



```
[1] "Frontiers: Algorithmic Collusion: Supra-competitive Prices via Independen
t Algorithms"
[1] "Frontiers: Moment Marketing: Measuring Dynamics in Cross-Channel Ad Effec
tiveness"
[1] "The Effect of Home-Sharing on House Prices and Rents: Evidence from Airbn
b"
[1] "The Impact of Coupons on the Visit-to-Purchase Funnel"
```

Exercise #2

Now, let us visit the MIT Open course website here: https://ocw.mit.edu/courses/most-visited-courses/



MIT COURSE#	COURSE TITLE	LEVEL
6.0001	Introduction to Computer Science and Programming in Python	Undergraduate
18.01SC	Single Variable Calculus	Undergraduate
18.06	Linear Algebra	Undergraduate
6.006	Introduction to Algorithms	Undergraduate
15.S12	Blockchain and Money	Graduate
18.02SC	Multivariable Calculus	Undergraduate

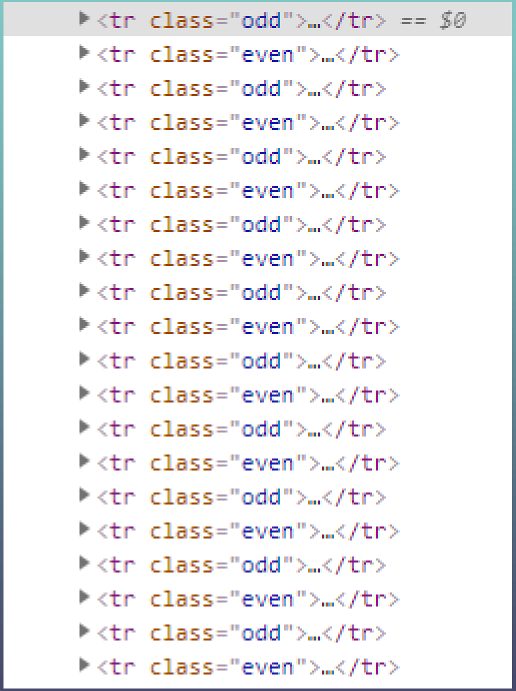
Exercise #2

In this exercise, we attempt to scrape the course code, course name, and course level for each course listed on the website. For example, the information we are scraping for the first course is 6.0001, “Introduction to Computer Science and Programming in Python”, “Undergraduate”.

Try this exercise yourself!

Exercise #2

First, we identify the root of each individual course. We need to inspect the HTML code first.



```
▶<tr class="odd">…</tr> == $0
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
▶<tr class="odd">…</tr>
▶<tr class="even">…</tr>
```

Exercise #2

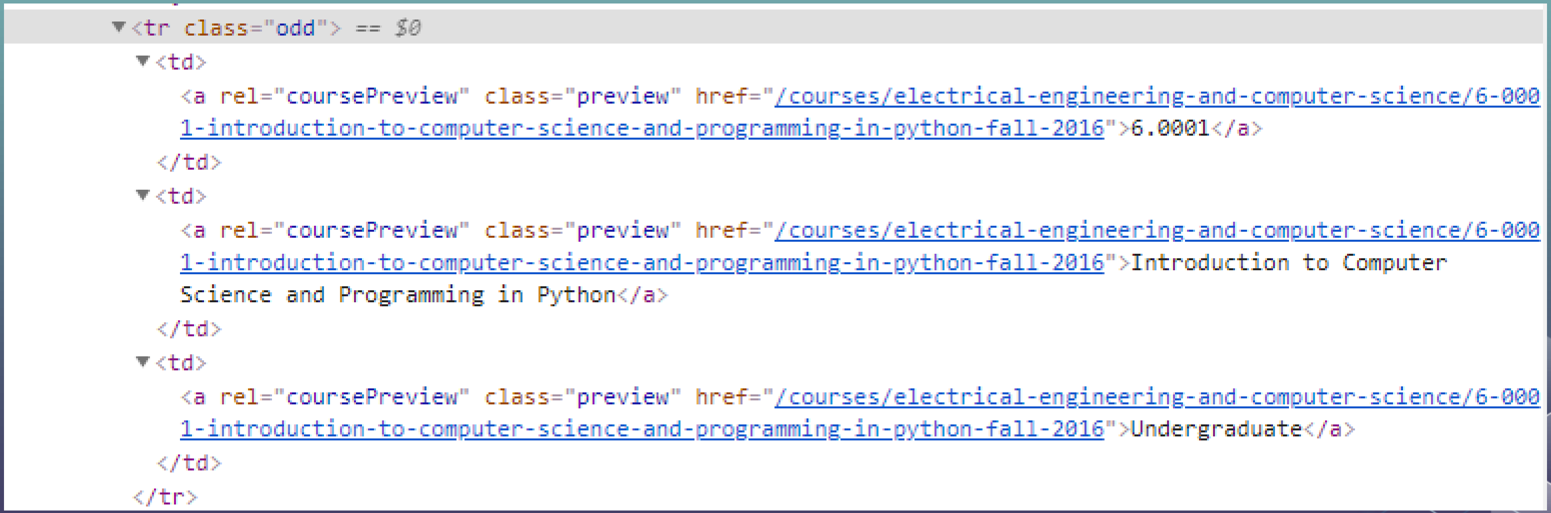
You can see that each class is represented by a “tr” node.

Interestingly, the 1st, 3rd, 5th … classes have a class attribute “odd”.

For the 2nd, 4th , 6th … classes, they have a class attribute “even”.

Exercise #2

Then, we need to analyze the course HTML code to understand how we could extract the course information (e.g., course code and course title).



```
▼<tr class="odd"> == $0
  ▼<td>
      <a rel="coursePreview" class="preview" href="/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016">6.0001</a>
    </td>
  ▼<td>
      <a rel="coursePreview" class="preview" href="/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016">Introduction to Computer Science and Programming in Python</a>
    </td>
  ▼<td>
      <a rel="coursePreview" class="preview" href="/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016">Undergraduate</a>
    </td>
  </tr>
```

Exercise #2

You can see that, interestingly, the course code, course title, and course level are all represented by an "a" node. Moreover, they share the same "rel" attribute (coursePreview) and the same "class" attribute (preview).

This means we can extract all information with the same code (good news)!

Exercise #2

You can use the following code to select the “tr” nodes whose class attribute is either “odd” or “even”:

```
url = "https://ocw.mit.edu/courses/most-visited-courses/"
webpage = read_html(url, encoding = "UTF-8")
course_nodes <- html_nodes(webpage,xpath = '//tr[@class="odd" or @class="even"]/td/a')
for (node in course_nodes)
  print(html_text(node))
```

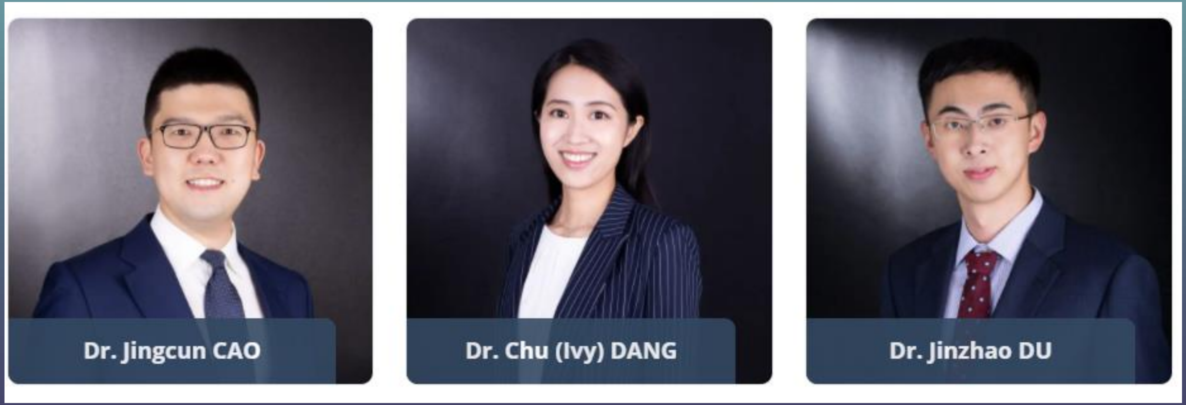
Downloading Images

Previously, we have discussed how to scrape text information from a website using a web scraper.

Now, let us consider scraping images from the web.

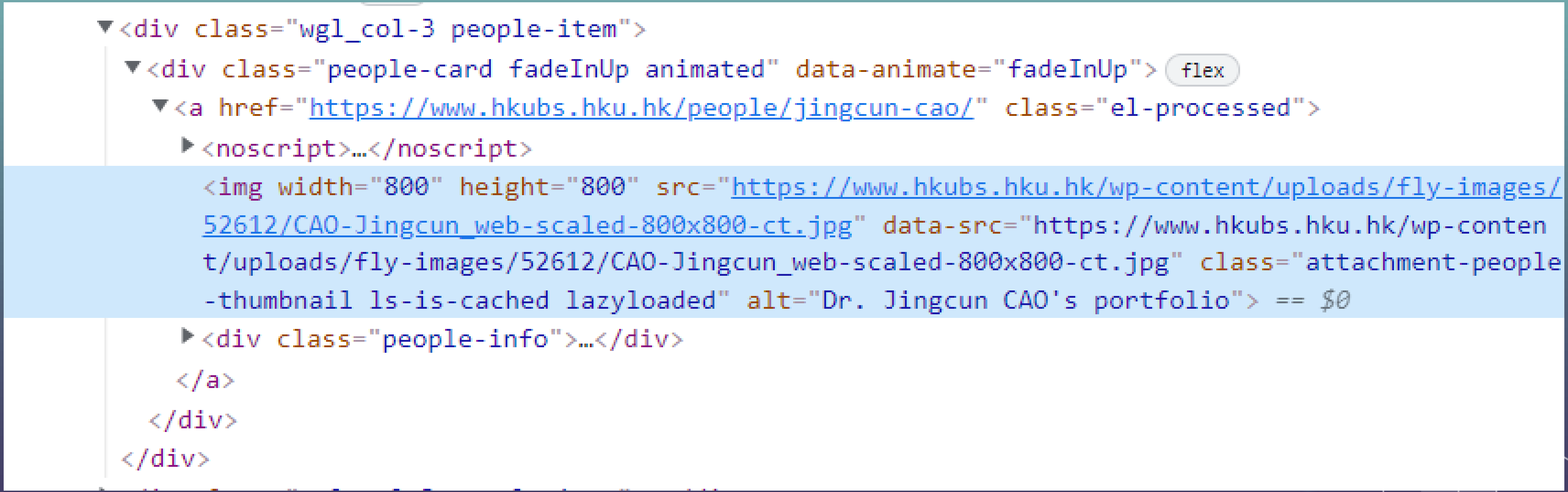
Scraping Images

Let us go back to the HKU marketing faculty webpage:



Scraping Images

You can find a link to each photo (in "src" or "data-src" attribute):



```
▼<div class="wgl_col-3 people-item">
  ▼<div class="people-card fadeInUp animated" data-animate="fadeInUp"> flex
    ▼<a href="https://www.hkubs.hku.hk/people/jingcun-cao/" class="el-processed">
      ▶<noscript>…</noscript>
       <img width="800" height="800" src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/
       52612/CAO-Jingcun_web-scaled-800x800-ct.jpg" data-src="https://www.hkubs.hku.hk/wp-conten
       t/uploads/fly-images/52612/CAO-Jingcun_web-scaled-800x800-ct.jpg" class="attachment-people
       -thumbnail ls-is-cached lazyloaded" alt="Dr. Jingcun CAO's portfolio"> == $0
      ▶<div class="people-info">…</div>
     </a>
   </div>
 </div>
```

Scraping Images

If you get the link, you will have access to the photo:

https://www.hkubs.hku.hk/wp-content/uploads/fly-images/52612/CAO-Jingcun_web-scaled-800x800-ct.jpg

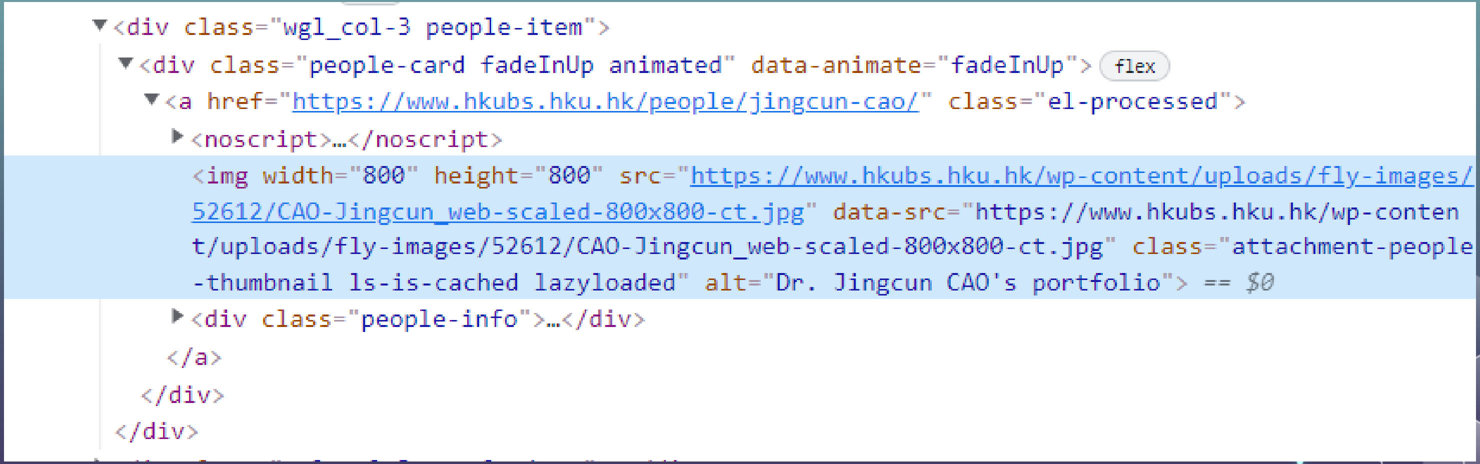
So, our first step to get the link information.

Scraping Images

```r
url =
"https://www.fbe.hku.hk/people/faculty?pg=1&
staff_type=faculty&subject_area=marketing&tr
ack=all"
webpage = read_html(url, encoding = "UTF-8")
image_nodes <- html_nodes(webpage,xpath =
'//div/a/img[@width="800"]')
print(length(image_nodes))
```

Scraping Images

But that's not enough. We not only want to get the nodes, but also need the link to each of the nodes. The link appears in the "src" or "data-src" attribute.



```
▼<div class="wgl_col-3 people-item">
  ▼<div class="people-card fadeInUp animated" data-animate="fadeInUp"> flex
    ▼<a href="https://www.hkubs.hku.hk/people/jingcun-cao/" class="el-processed">
      ▶<noscript>…</noscript>
       <img width="800" height="800" src="https://www.hkubs.hku.hk/wp-content/uploads/fly-images/
       52612/CAO-Jingcun_web-scaled-800x800-ct.jpg" data-src="https://www.hkubs.hku.hk/wp-conten
       t/uploads/fly-images/52612/CAO-Jingcun_web-scaled-800x800-ct.jpg" class="attachment-people
       -thumbnail ls-is-cached lazyloaded" alt="Dr. Jingcun CAO's portfolio"> == $0
      ▶<div class="people-info">…</div>
     </a>
   </div>
 </div>
```

Scraping Images

But that's not enough. We not only want to get the nodes, but also need the link to each of the nodes. The link appears in the "src" or "data-src" attribute.

```
image_nodes <- html_nodes(webpage,xpath =
'//div/a/img[@width="800"]')
for (image in image_nodes)
{
  photourl <- html_attr(image, "data-src")
  print(photourl)
}
```

Downloading Images

```
number = 1
for (image in image_nodes)
{
    photourl <- html_attr(image, "data-src")
    print(photourl)
    download.file(photourl,
paste0(toString(number),'_HKU_Photo.jpg'),
mode = 'wb')
    number = number + 1
}
```

Static vs. Dynamic Websites

The Difference Between STATIC & DYNAMIC Websites

DON MUDALIGE | WWW.DONWEBSOLUTIONS.COM | (510) 314-3172

Dynamic Websites

What we learned in today’s class works well for static websites. But it does not work equally well on dynamic websites. If you want to scrape data from a dynamic website, you may need to use some more advanced tools.

Dynamic Websites

If you want to scrape data from a dynamic website, there is a tool called “selenium”. We also have a packaged called “RSelenium” in R.

The selenium tool allows your scraper to visit a webpage like a human-being. That is, if you write a scraper with selenium, your scraper will also be able to scroll down your pages, click buttons, enter your password, etc.

Dynamic Websites

For example, you can also write a program to log in to the 12306 (China railway) website to book a rail ticket for you (i.e., 抢票).

You can also write a program to log in to your Moodle account first and then scrape data from the website.