Show me the data!

Week12: Web Scraping

Big Data & Social Analysis

R

Instructors: Chung-pei Pien

ZU1942001/266868001/Z23937001/ZM1941001

International College of INNOVATION
National Chengchi University
國立政治大學創新國際學院

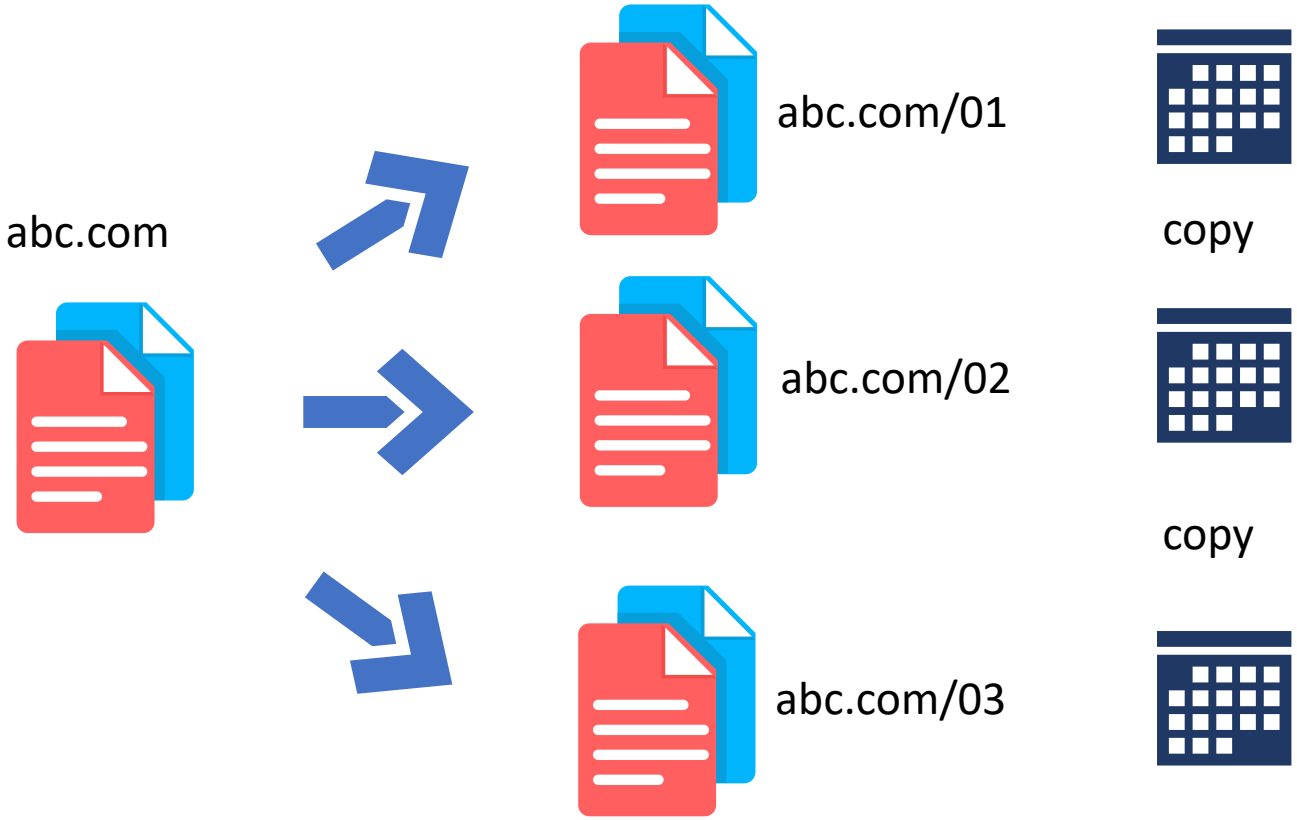
CONTENTS

1. Introduction for Web Scraping
2. Web Scraping Basics
3. Scrape Training Site
4. Practice
5. Project Proposal

Introduction for Web Scraping

01 Introduction of Web Scraping

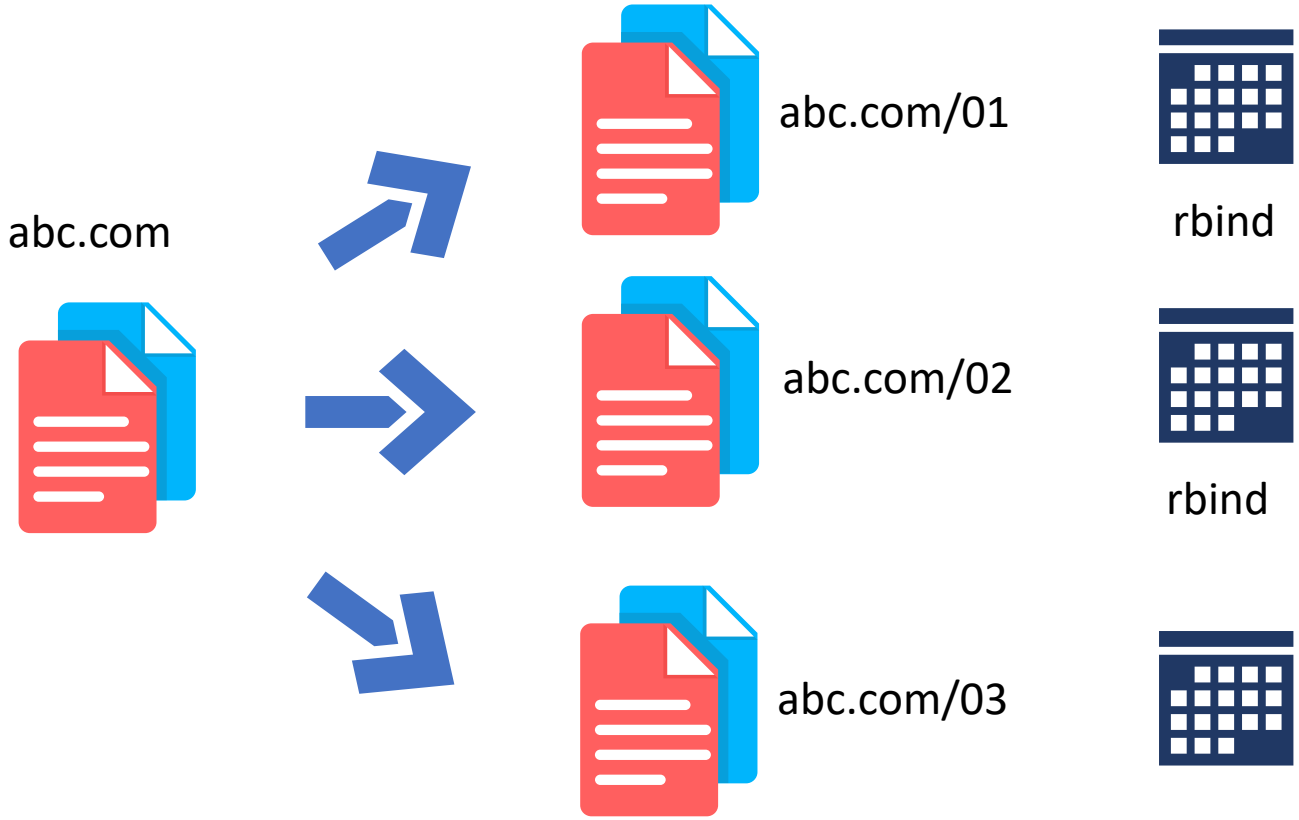
Web scraping is a coding skill of automation of manual copy and paste.



01 Introduction of Web Scraping

Web scraping is to deeply analyze a website:

- **Find a rule to enter subpages**
- **Get subpages' data and merge them**



01 Introduction of Web Scraping

01 Introduction of Web Scraping

Common Anti–Scraping Techniques

Setting Up Robots.txt

Filtering Requests By User Agent

Blacklisting IP Addresses

Showing A Captcha

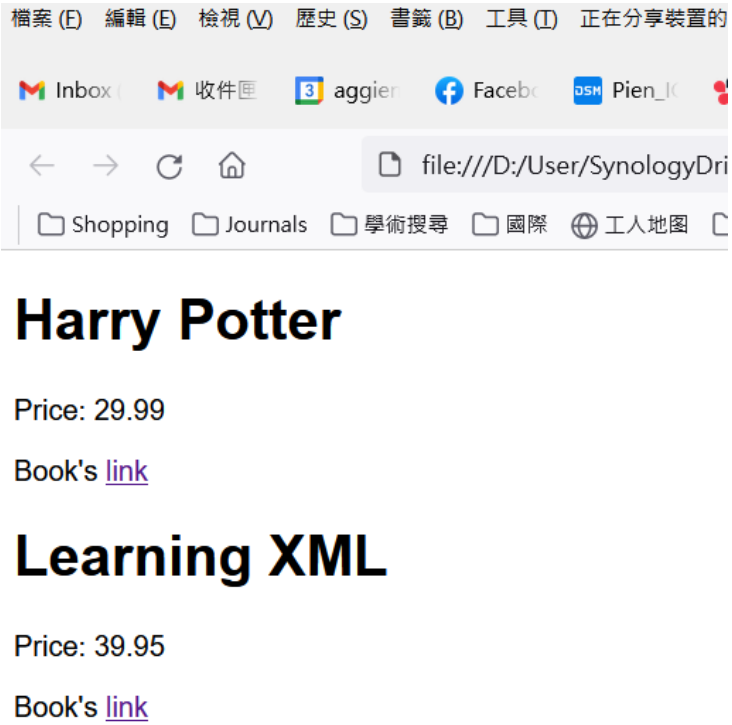
Honeypots

Grimes 2022

Website Engineers Hate You

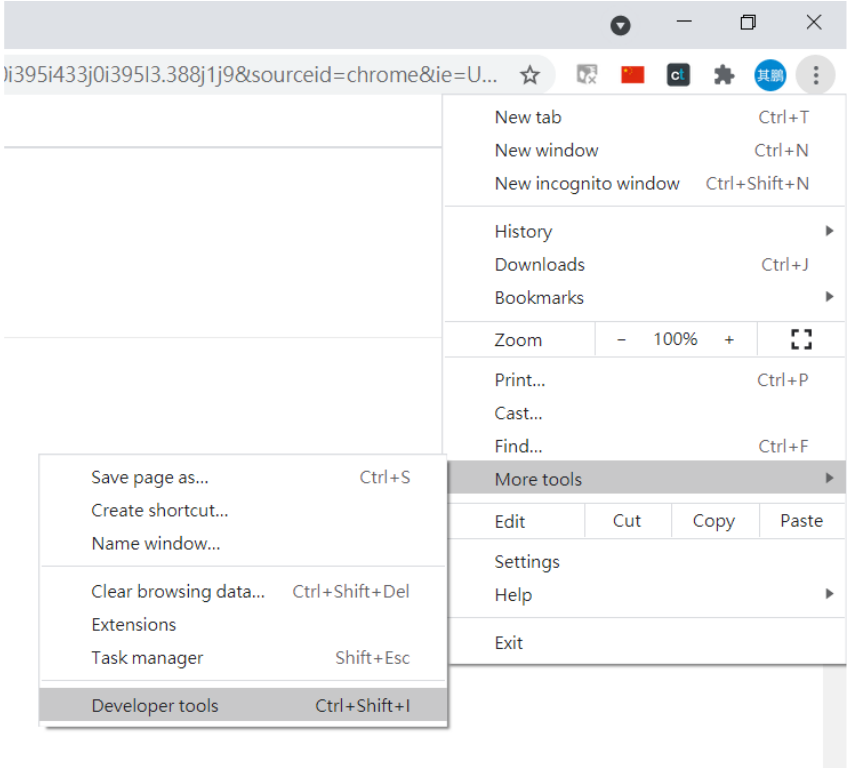
Web Scraping Basics

Download book.zip, then unzip and open it



	title	price	link
1	Harry Potter	29.99	https://www.amazon.com/
2	Learning XML	39.95	https://www.amazon.com/

Open Chrome
Click Developer tools



02 Web Scraping Basics

Developer tools allow you to monitor all actions when you browse a webpage and show original html codes.

```
<!DOCTYPE html>
<html>
  ▶ <head>…</head>
  ▼ <body> == $0
      <h1>Harry Potter</h1>
      <p>Price: 29.99</p>
    ▼ <p>
        "Book's "
        <a href="https://www.amazon.com/">link</a>
      </p>
      <h1>Learning XML</h1>
      <p>Price: 39.95</p>
    ▼ <p>
        "Book's "
        <a href="https://www.amazon.com/">link</a>
      </p>
    </body>
</html>
```

02 Web Scraping Basics

1. getURL() to get webpages
2. htmlParse() to parse html codes
3. xpathSApply() to appoint the data you want

XPath also called as XML Path is a language to query XML documents. It is an important strategy to locate elements in webpages (Vaidya 2021).

02 Web Scraping Basics

Select current node

Select attributes

Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc

Attribute Name

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

//: It is used to select the current node.
tagname: It is the name of the tag of a particular node.
@: It is used to select attribute.
Attribute: It is the name of the attribute of the node.
Value: It is the value of the attribute

02 Web Scraping Basics

parse <- htmlParse("book.html", encoding = "UTF-8")

```
> parse
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>What is the DOM ?</title>
</head>
<body>
    <h1>Harry Potter</h1>
    <p>Price: 29.99</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

    <h1>Learning XML</h1>
    <p>Price: 39.95</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

</body>
</html>
```

Select current node | Select attributes | Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc | Attribute Name

Book title:

xpathSApply(doc = parse, path = "//h1", fun = xmlValue)

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

02 Web Scraping Basics

parse <- htmlParse("book.html", encoding = "UTF-8")

```
> parse
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>What is the DOM ?</title>
</head>
<body>
    <h1>Harry Potter</h1>
    <p>Price: 29.99</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

    <h1>Learning XML</h1>
    <p>Price: 39.95</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

</body>
</html>
```

Select current node

Select attributes

Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc

Attribute Name

Price:

xpathSApply(doc = parse, path = "//p", fun = xmlValue)

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

02 Web Scraping Basics

parse <- htmlParse("book.html", encoding = "UTF-8")

```
> parse
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>What is the DOM ?</title>
</head>
<body>
    <h1>Harry Potter</h1>
    <p>Price: 29.99</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

    <h1>Learning XML</h1>
    <p>Price: 39.95</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

</body>
</html>
```

Select current node | Select attributes | Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc | Attribute Name

Price:

xpathSApply(parse, "//p[contains(text(), 'Price')]", xmlValue)

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

02 Web Scraping Basics

parse <- htmlParse("book.html", encoding = "UTF-8")

```
> parse
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>What is the DOM ?</title>
</head>
<body>
    <h1>Harry Potter</h1>
    <p>Price: 29.99</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

    <h1>Learning XML</h1>
    <p>Price: 39.95</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

</body>
</html>
```

Select current node | Select attributes | Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc | Attribute Name

link:

xpathSApply(parse, "//a", xmlValue)

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

02 Web Scraping Basics

parse <- htmlParse("book.html", encoding = "UTF-8")

```
> parse
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>What is the DOM ?</title>
</head>
<body>
    <h1>Harry Potter</h1>
    <p>Price: 29.99</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

    <h1>Learning XML</h1>
    <p>Price: 39.95</p>
    <p>Book's <a href="https://www.amazon.com/">link</a></p>

</body>
</html>
```

Select current node | Select attributes | Value of an attribute

```
xpathSApply(doc, "//tagname[@Attribute = 'Value']", function)
```

Tagname like input, div, img etc | Attribute Name

link:

xpathSApply(parse, "//a", xmlGetAttr, 'href')

Function	Return
xmlValue	Content
xmlName	Name of Tag
xmlAttrs	All Attributes
xmlGetAttr	Get Attributes
xmlChildren	Subnode
xmlSize	Node size

02 Web Scraping Basics

```
title <- xpathSApply(parse, "//h1", xmlValue)

price <- xpathSApply(parse, "//p[contains(text(), 'Price')]", xmlValue)

price <- substr(price, regexpr("[0-9]", price), nchar(price))

link <- xpathSApply(parse, "//a", xmlGetAttr, 'href')

booklist <- data.frame(title = title, price = price, link = link)
```

	title	price	link
1	Harry Potter	29.99	https://www.amazon.com/
2	Learning XML	39.95	https://www.amazon.com/

Scrape Training Site

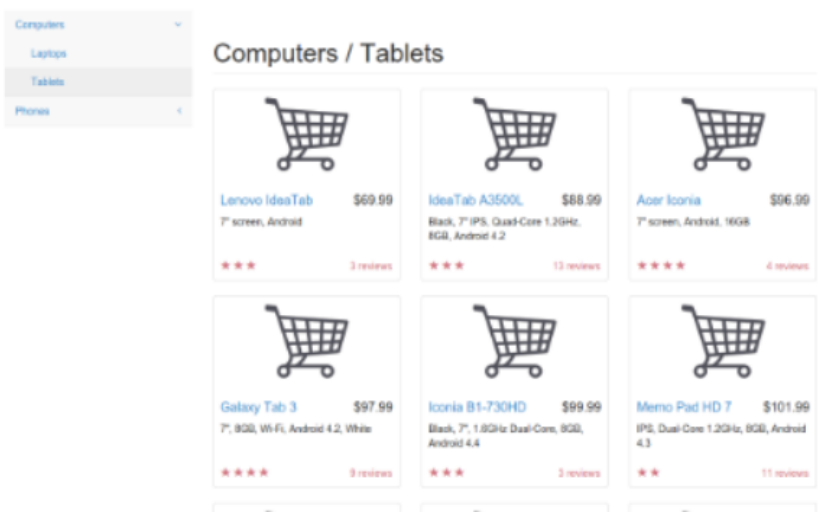
03 Scrape Training Site

https://webscraper.io/test-sites

Please click the first one and Phones.

Test Sites

Here are some sites that you can use for training while learning how to use the W

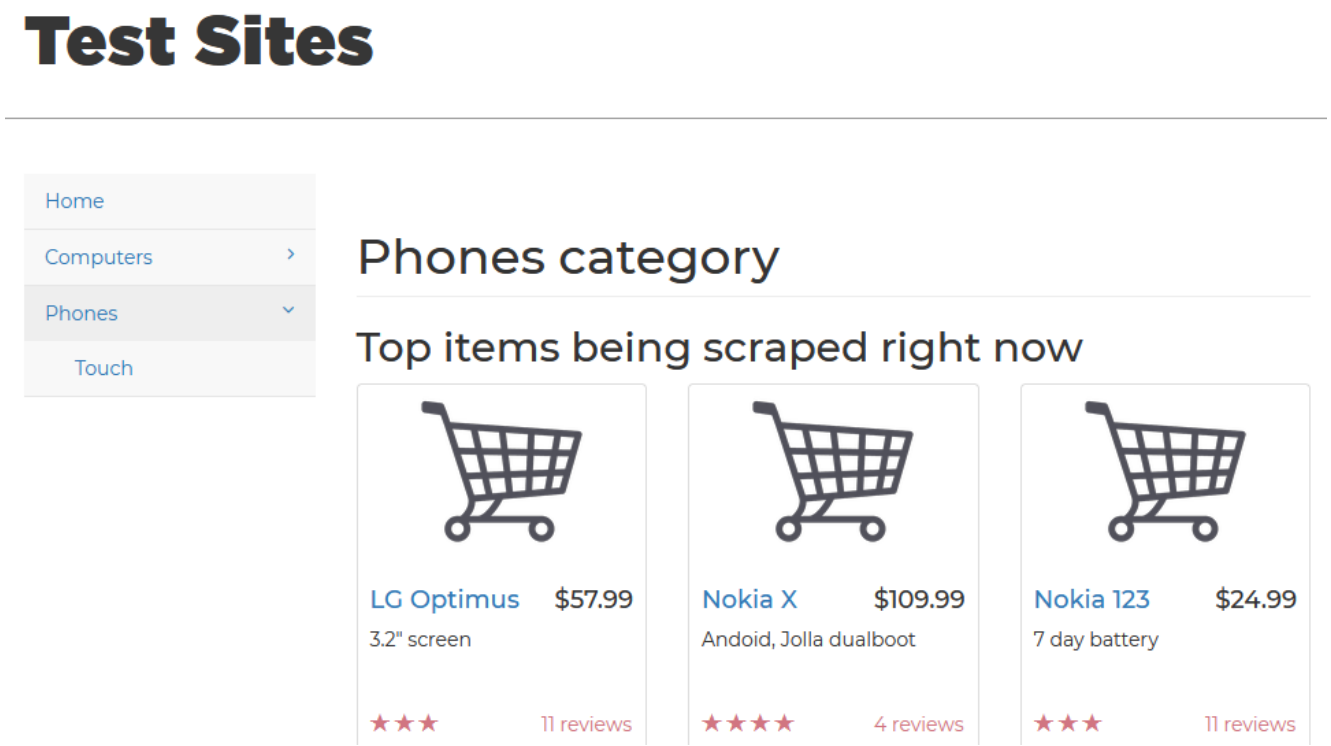


E-commerce site

E-commerce site with multiple cate
items are loaded in one page.

03 Scrape Training Site

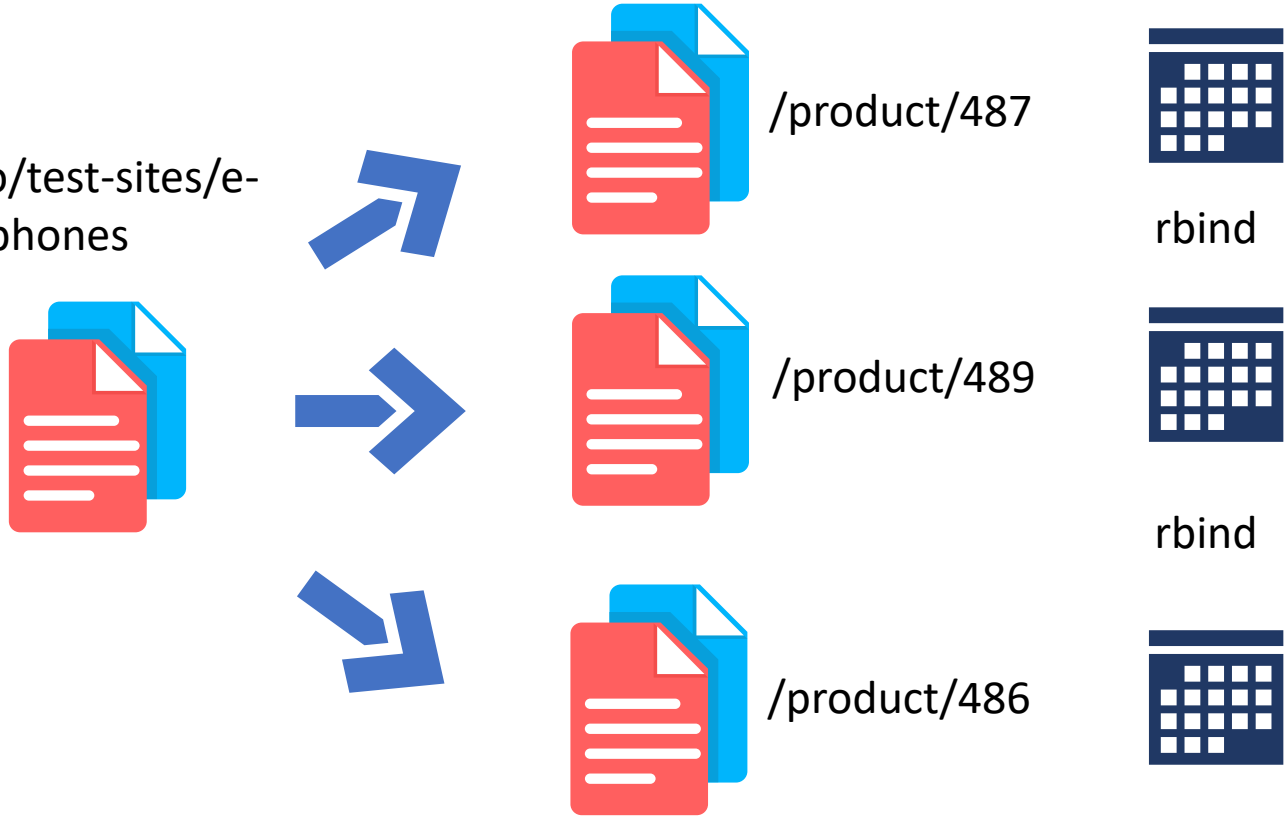
Our task is to go to every item's page and scrape the pages' information.



03 Scrape Training Site

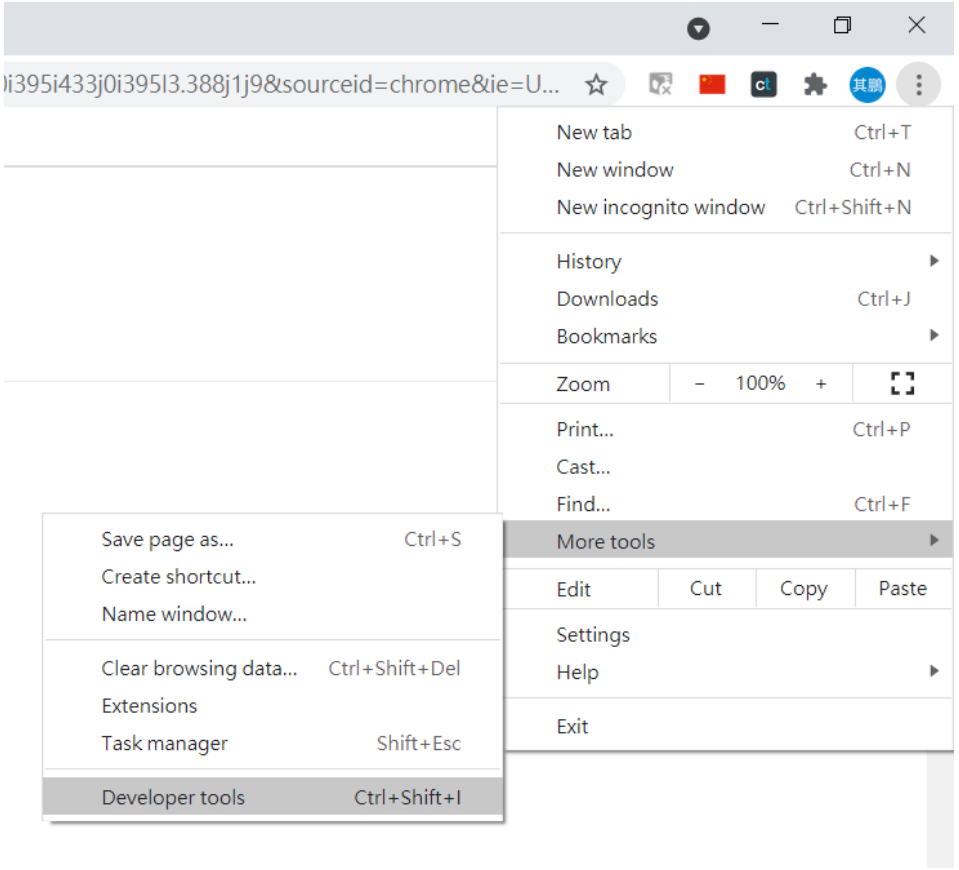
Go to every item's page and scrape pages' information

https://webscraper.io/test-sites/e-commerce/allinone/phones



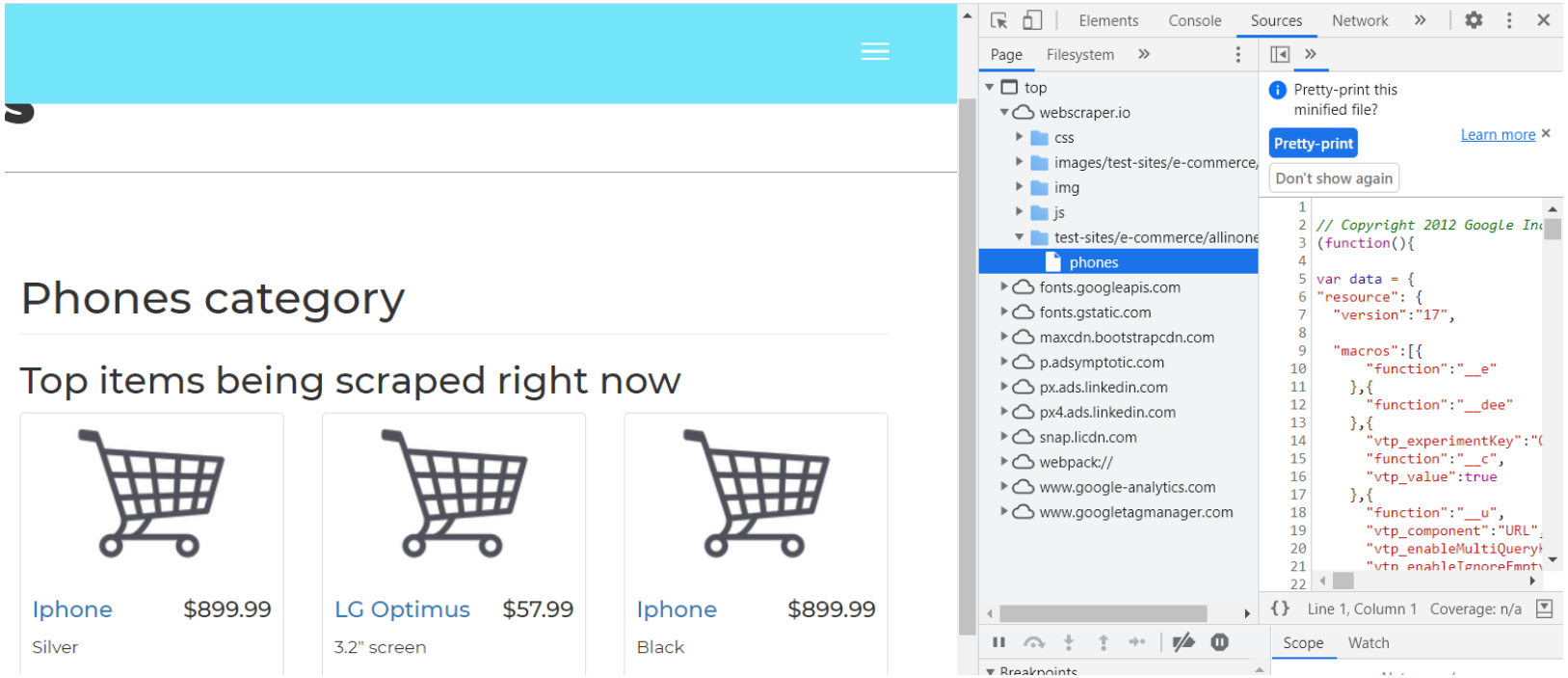
03 Scrape Training Site

Open Chrome
Click Developer tools



03 Scrape Training Site

Developer tools allow you to monitor all actions when you browse a webpage and show original html codes.



03 Scrape Training Site

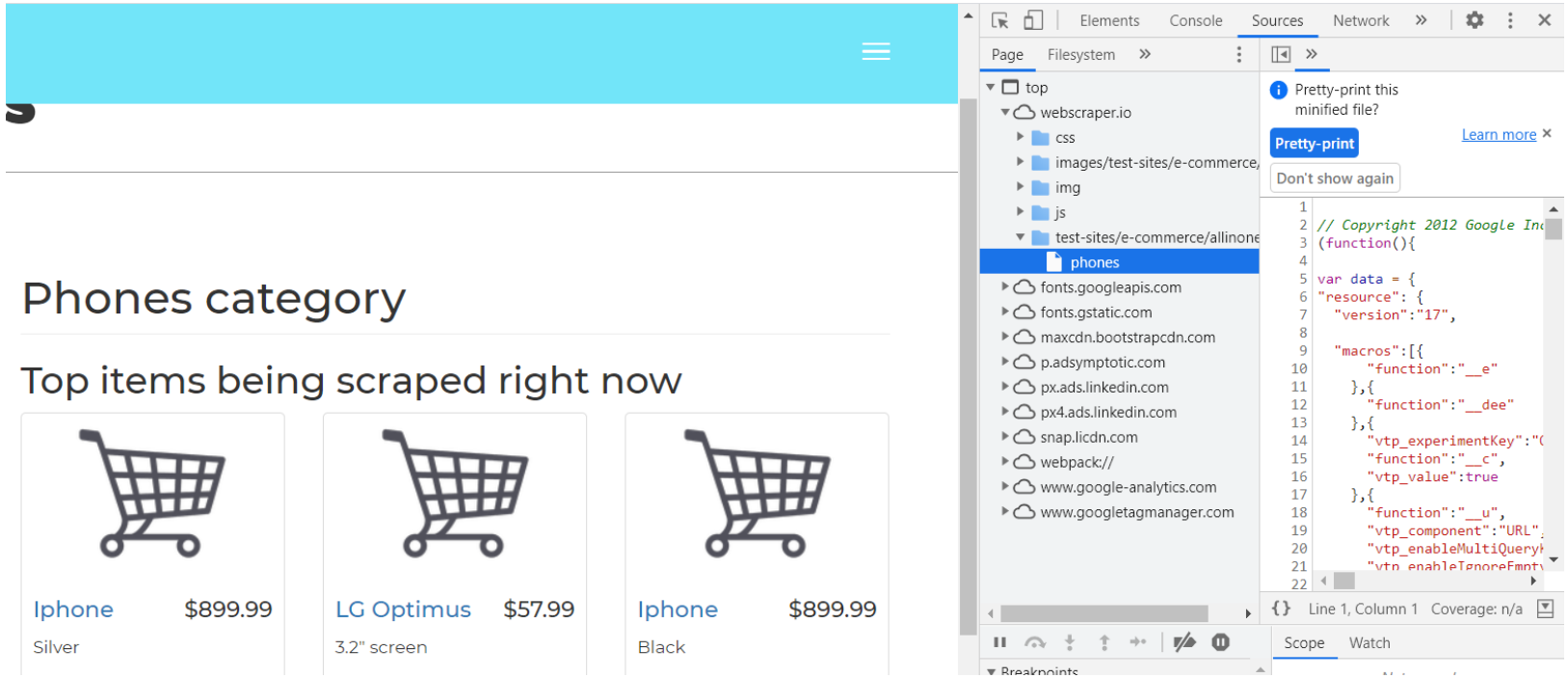
Identify what kinds of information we need:

1. **Items' names**
2. **Links to subpages**
3. **Prices**
4. **Description**
5. **Rates**
6. **Reviews**

03 Scrape Training Site

Is wanted information in the primary html file?

If the answer is yes, congrat, it's a static page. You can scrape the data very fast.



03 Scrape Training Site

Again, Please follow the steps:

1. **getURL() to get webpages**
2. **htmlParse() to parse html codes**
3. **xpathSApply() to appoint the data you want**

03 Scrape Training Site

```
testhtml <-
        getURL("https://webscraper.io/test-
        sites/e-commerce/allinone/phones")

phone_parse <- htmlParse(testhtml,
         encoding = "UTF-8")

phone_parse
```

03 Scrape Training Site

1. Items' names

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

```
title <- xpathSApply(phone_parse,
                     "//h4/a[@class='title']", xmlValue)
```

03 Scrape Training Site

2. Links to subpages

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

```
link <- xpathSApply(xpathSApply(phone_parse,
                                "//h4/a[@href]", xmlGetAttr, 'href')
```

03 Scrape Training Site

4. Description

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

descri <- xpathSApply(phone_parse,
"//p[@class='description']", xmlValue)

03 Scrape Training Site

3. Prices

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

price <- xpathSApply(phone_parse,
"//h4[@class='pull-right price']", xmlValue)

03 Scrape Training Site

5. Rates

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

```
rating <-  xpathSApply(phone_parse,
                       "//p[@data-rating]", xmlGetAttr, 'data-rating')
```

03 Scrape Training Site

6. Reviews

```
<div class="thumbnail">
  <img class="img-responsive" alt="item"
       src="/images/test-sites/e-commerce/items/cart2.png">
  <div class="caption">
    <h4 class="pull-right price">$489.99</h4>
    <h4>
                <a href="/test-sites/e-commerce/allinone/product/504"
                class="title" title="Galaxy Note">Galaxy Note</a>
            </h4>
    <p class="description">12.2&quot;, 32GB, WiFi, Android 4.4, White</p>

  </div>
  <div class="ratings">
    <p class="pull-right">9 reviews</p>
    <p data-rating="3">
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
              <span class="glyphicon glyphicon-star"></span>
            </p>
  </div>
</div>
```

review <- xpathSApply(phone_parse,
"//p[@class='pull-right']", xmlValue)

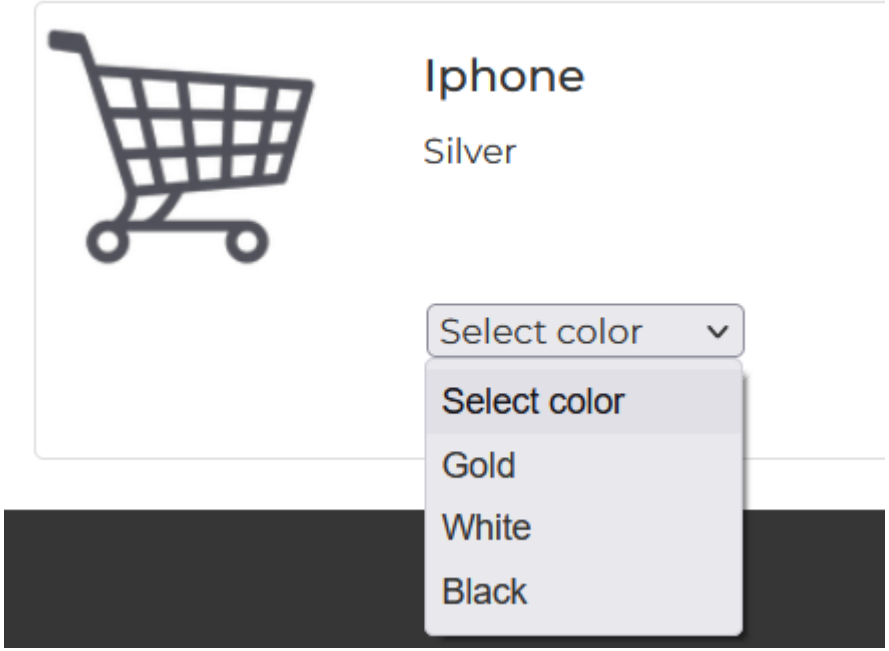
03 Scrape Training Site

```
phonelist <- data.frame(title = title, description = descri,
                        price = price, rate = rating,
                        review = review, link = link)
```

	title	description	price	rate	review	link
1	Sony Xperia	GPS, waterproof	$118.99	1	6 reviews	/test-sites/e-commerce/allinone/product/490
2	Iphone	White	$899.99	1	10 reviews	/test-sites/e-commerce/allinone/product/492
3	Iphone	Black	$899.99	1	1 reviews	/test-sites/e-commerce/allinone/product/494

03 Scrape Training Site

We want to enter every subpages to get color data.



03 Scrape Training Site

phonelist$link

```
sub <- getURL(paste0("https://webscraper.io/", phonelist$link[1]))

sub_parse <- htmlParse(sub, encoding = "UTF-8")

color <- xpathSApply(sub_parse, "//select[@aria
                                -label='color']//option[@value]", xmlValue)
```

```
▼<div class="col-lg-10">
  ▶<div class="caption">…</div>
  ▼<div class="dropdown">
    ▼<select aria-label="color">
        <option value>Select color</option>
        <option value="Gold">Gold</option>
        <option value="White">White</option>
        <option value="Black">Black</option>
      </select>
    </div>
```

03 Scrape Training Site

How to save your color data?

color <- color[2:length(color)]

```
> color
[1] "Gold"  "White" "Black"
```

color <- toString(color)

```
> color
[1] "Gold, White, Black"
```

temp <- data.frame(color = color)

	color
1	Gold, White, Black

03 Scrape Training Site

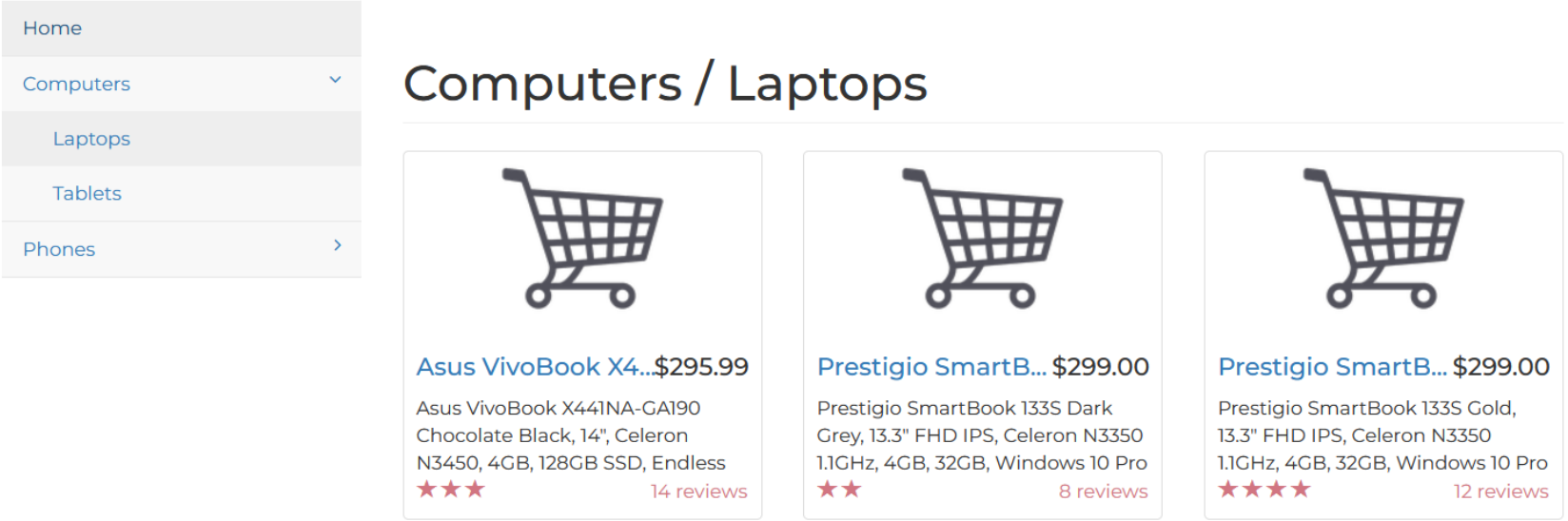
```
color_df <- data.frame()

for (i in 1:3) {

  sub <- getURL(paste0("https://webscraper.io/", phonelist$link[i]))

  sub_parse <- htmlParse(sub, encoding = "UTF-8")

  color <- xpathSApply(sub_parse,
                       "//select[@aria-label='color']//option[@value]", xmlValue)

  color <- color[2:length(color)]

  color <- toString(color)

  temp <- data.frame(color = color)

  color_df <- rbind(color_df, temp)
}

phonelist <- cbind(phonelist, color_df)
```

Practice

04 Practice

Click computer and Laptops, then scrape information from these items and subpages of items.

https://webscraper.io/test-sites/e-commerce/allinone/computers/laptops



Project Proposal

05 Evaluation

Items	Points	%
Project Proposal	150	11.54%

- **Every team will be required to present your project proposal in 5-10 slides in the Week 13 for the final presentation and report.**

- **I uploaded the project proposal guideline. You should follow the guideline to present your proposals.**

05 Evaluation

Items	Points	%
Project Proposal	150	11.54%

	Team Points	PM bonus	Writers bonus
A+	143	5	3
A	138	4	3
A-	135	3	3
B+	127	3	1.5
B	123	3	1.5
B-	120	1.5	1.5
C+	112	0	0
C	105	0	0