Web scraping part I

Programming for Statistical Science

Shawn Santo

Supplementary materials

Full video lecture available in Zoom Cloud Recordings

Additional resources

- SelectorGadget Vignette
- `rvest` website

Recap

Summary of packages

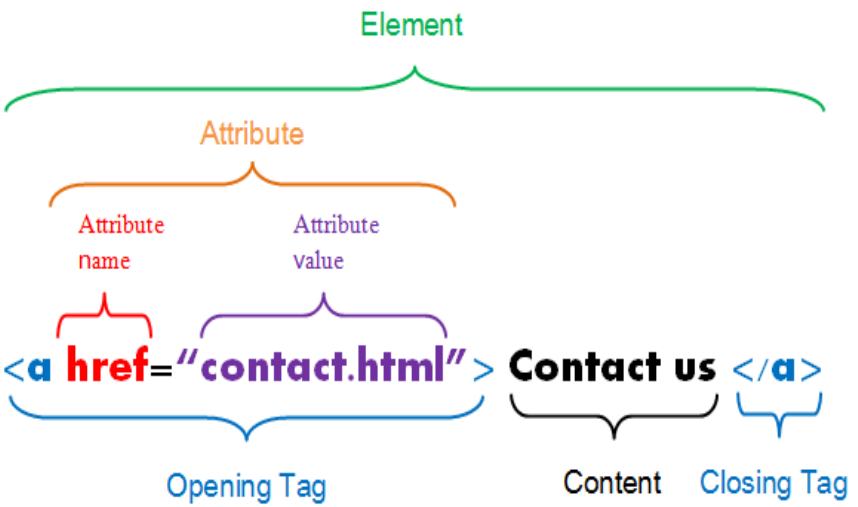
Task	Package	Cheat sheet
Visualize data	`ggplot2`	https://github.com/rstudio/cheatsheets/raw/master/data-visualization-2.1.pdf
Wrangle data frames	`dplyr`	https://github.com/rstudio/cheatsheets/raw/master/data-transformation.pdf
Reshape data frames	`tidyr`	https://github.com/rstudio/cheatsheets/raw/master/data-import.pdf
Iterate	`purrr`	https://github.com/rstudio/cheatsheets/raw/master/purrr.pdf
Text manipulation	`stringr`	https://github.com/rstudio/cheatsheets/raw/master/strings.pdf
Manipulate factors	`forcats`	https://github.com/rstudio/cheatsheets/raw/master/factors.pdf
Manipulate dates	`lubridate`	https://github.com/rstudio/cheatsheets/raw/master/lubridate.pdf
Spatial data	`sf`	https://github.com/rstudio/cheatsheets/raw/master/sf.pdf

You don't need to memorize every function in these packages. Just know where you need to look when you come across a specific problem.

HTML

Hypertext Markup Language

- HTML describes the structure of a web page; your browser interprets the structure and contents and displays the results.
- The basic building blocks include elements, tags, and attributes.
  - an element is a component of an HTML document
  - elements contain tags (start and end tag)
  - attributes provide additional information about HTML elements



Simple HTML document

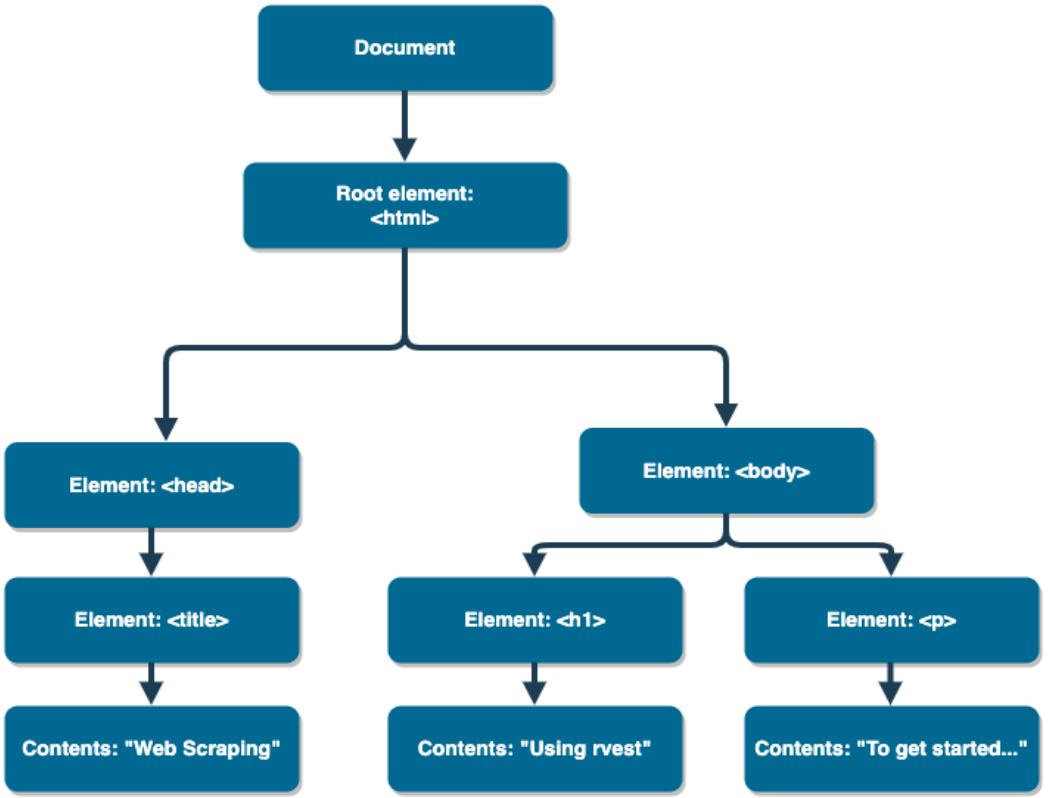
```
<html>
<head>
<title>Web Scraping</title>
</head>
<body>

<h1>Using rvest</h1>
<p>To get started...</p>

</body>
</html>
```

We can visualize this in a tree-like structure.

HTML tree-like structure



If we have access to an HTML document, then how can we easily extract information?

Package `rvest`

Package `rvest`

`rvest` is a package authored by Hadley Wickham that makes basic processing and manipulation of HTML data easy.

```
library(rvest)
```

Core functions:

Function	Description
`xml2::read_html()`	read HTML from a character string or connection
`html_nodes()`	select specified nodes from the HTML document using CSS selectors
`html_table()`	parse an HTML table into a data frame
`html_text()`	extract tag pairs' content
`html_name()`	extract tags' names
`html_attrs()`	extract all of each tag's attributes
`html_attr()`	extract tags' attribute value by name

HTML in R

```
simple_html <- "<html>
  <head>
    <title>Web Scraping</title>
  </head>
  <body>

    <h1>Using rvest</h1>
    <p>To get started...</p>

  </body>
</html>"
```

```
simple_html
```

```
#> [1] "<html>\n  <head>\n    <title>Web Scraping</title>\n  </head>\n  <body>\n
```

```
html_doc <- read_html(simple_html)
attributes(html_doc)
```

```
#> $names
#> [1] "node" "doc"
#>
#> $class
#> [1] "xml_document" "xml_node"
```

```
html_doc
```

```
#> {html_document}
#> <html>
#> [1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8
#> [2] <body>\n  \n    <h1>Using rvest</h1>\n    <p>To get started...</p>\n  \n
```

CSS selectors

To extract components out of HTML documents use `html_nodes()` and CSS selectors. In CSS, selectors are patterns used to select elements you want to style.

- CSS stands for Cascading Style Sheets.
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media.
- CSS can be added to HTML elements in 3 ways:
    - Inline - by using the style attribute in HTML elements
    - Internal - by using a style element in the head section
    - External - by using an external CSS file

More on CSS

Selector	Example	Description
element	`p`	Select all <p> elements
element element	`div p`	Select all <p> elements inside a element
element>element	`div > p`	Select all <p> elements with as a parent
.class	`.title`	Select all elements with class="title"
#id	`#name`	Select all elements with id="name"
[attribute]	`[class]`	Select all elements with a class attribute
[attribute=value]	`[class=title]`	Select all elements with class="title"

For more CSS selector references click here.

Fortunately, we can determine the necessary CSS selectors we need via the point-and-click tool selector gadget. More on this in a moment.

Example

URL: https://raysnotebook.info/ows/schedules/The%20Whole%20Shebang.html

```
<html lang=en>
<head>
   <title>Rays Notebook: Open Water Swims 2020 — The Whole Shebang</title>
</head>
<body>
<main class=schedule>
<h1>The Whole Shebang</h1>

<p>This schedule lists every swim in the database. 383 events.</p>

<table class=schedule>
<thead><tr><th>Date</th><th>Location</th><th>Name</th><th>Distance</th><th>More</th></tr></thea
<tbody>

<tr id=January>
<td class=date>Jan 12, Sun</td>
<td class=where>
   <a class=mapq href="http://www.google.com/maps/?q=27.865501,-82.631997">Petersburg, FL</a>
   <span class=more>
   Gandy Beach, Gandy Blvd N St, Petersburg, FL
   </span>
</td>
<td class=name><a href="http://tampabayfrogman.com/">Tampa Bay Frogman</a></td>
<td class=distance>5 km</td>
<td class=more><span class=time>7:15 AM</span>, Old Tampa Bay.</td>
</tr>
</body>
</html>
```

Suppose we want to extract and parse the information highlighted below in yellow.

```
<html lang=en>
<head>
   <title>Rays Notebook: Open Water Swims 2020 — The Whole Shebang</title>
</head>
<body>
<main class=schedule>
<h1>The Whole Shebang</h1>

<p>This schedule lists every swim in the database. 383 events.</p>

<table class=schedule>
<thead><tr><th>Date</th><th>Location</th><th>Name</th><th>Distance</th><th>More</th></tr></thea
<tbody>

<tr id=January>
<td class=date>Jan 12, Sun</td>
<td class=where>
   <a class=mapq href="http://www.google.com/maps/?q=27.865501,-82.631997">Petersburg, FL</a>
   <span class=more>
   Gandy Beach, Gandy Blvd N St, Petersburg, FL
   </span>
</td>
<td class=name><a href="http://tampabayfrogman.com/">Tampa Bay Frogman</a></td>
<td class=distance>5 km</td>
<td class=more><span class=time>7:15 AM</span>, Old Tampa Bay.</td>
</tr>
</body>
</html>
```

Step 1

Save the HTML as a character object named `html_swim`.

```
html_swim <- "<html lang=en> ... </body></html>"
```

Step 2

To extract all <p> elements:

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "p")
```

```
#> {xml_nodeset (1)}
#> [1] <p>This schedule lists every swim in the database. 383 events.</p>
```

Step 3

To extract the contents between the tags:

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "p") %>%
  html_text()
```

```
#> [1] "This schedule lists every swim in the database. 383 events."
```

Suppose we want to extract and parse pieces of the information highlighted below in yellow.

```
<html lang=en>
<head>
   <title>Rays Notebook: Open Water Swims 2020 — The Whole Shebang</title>
</head>
<body>
<main class=schedule>
<h1>The Whole Shebang</h1>

<p>This schedule lists every swim in the database. 383 events.</p>

<table class=schedule>
<thead><tr><th>Date</th><th>Location</th><th>Name</th><th>Distance</th><th>More</th></tr></thea
<tbody>

<tr id=January>
<td class=date>Jan 12, Sun</td>
<td class=where>
   <a class=mapq href="http://www.google.com/maps/?q=27.865501,-82.631997">Petersburg, FL</a>
   <span class=more>
   Gandy Beach, Gandy Blvd N St, Petersburg, FL
   </span>
</td>
<td class=name><a href="http://tampabayfrogman.com/">Tampa Bay Frogman</a></td>
<td class=distance>5 km</td>
<td class=more><span class=time>7:15 AM</span>, Old Tampa Bay.</td>
</tr>
</body>
</html>
```

To select all elements with `class="where"`:

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "[class=where]")
```

```
#> {xml_nodeset (1)}
#> [1] <td class="where">\n    <a class="mapq" href="http://www.google.com/maps/?
```

To extract the text:

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "[class=where]") %>%
  html_text()
```

```
#> [1] "\n    Petersburg, FL\n    \n    Gandy Beach, Gandy Blvd N St, Petersburg, FL
```

To extract the attributes:

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "[class=where]") %>%
  html_attrs()
```

```
#> [[1]]
#>    class
#> "where"
```

Suppose we want to extract and parse the information highlighted below in yellow.

```
<html lang=en>
<head>
   <title>Rays Notebook: Open Water Swims 2020 — The Whole Shebang</title>
</head>
<body>
<main class=schedule>
<h1>The Whole Shebang</h1>

<p>This schedule lists every swim in the database. 383 events.</p>

<table class=schedule>
<thead><tr><th>Date</th><th>Location</th><th>Name</th><th>Distance</th><th>More</th></tr></thea
<tbody>

<tr id=January>
<td class=date>Jan 12, Sun</td>
<td class=where>
   <a class=mapq href="http://www.google.com/maps/?q=27.865501,-82.631997">Petersburg, FL</a>
   <span class=more>
   Gandy Beach, Gandy Blvd N St, Petersburg, FL
   </span>
</td>
<td class=name><a href="http://tampabayfrogman.com/">Tampa Bay Frogman</a></td>
<td class=distance>5 km</td>
<td class=more><span class=time>7:15 AM</span>, Old Tampa Bay.</td>
</tr>
</body>
</html>
```

To extract the links (those with an `href` attribute):

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "[href]")
```

```
#> {xml_nodeset (2)}
#> [1] <a class="mapq" href="http://www.google.com/maps/?q=27.865501,-82.631997"
#> [2] <a href="http://tampabayfrogman.com/">Tampa Bay Frogman</a>
```

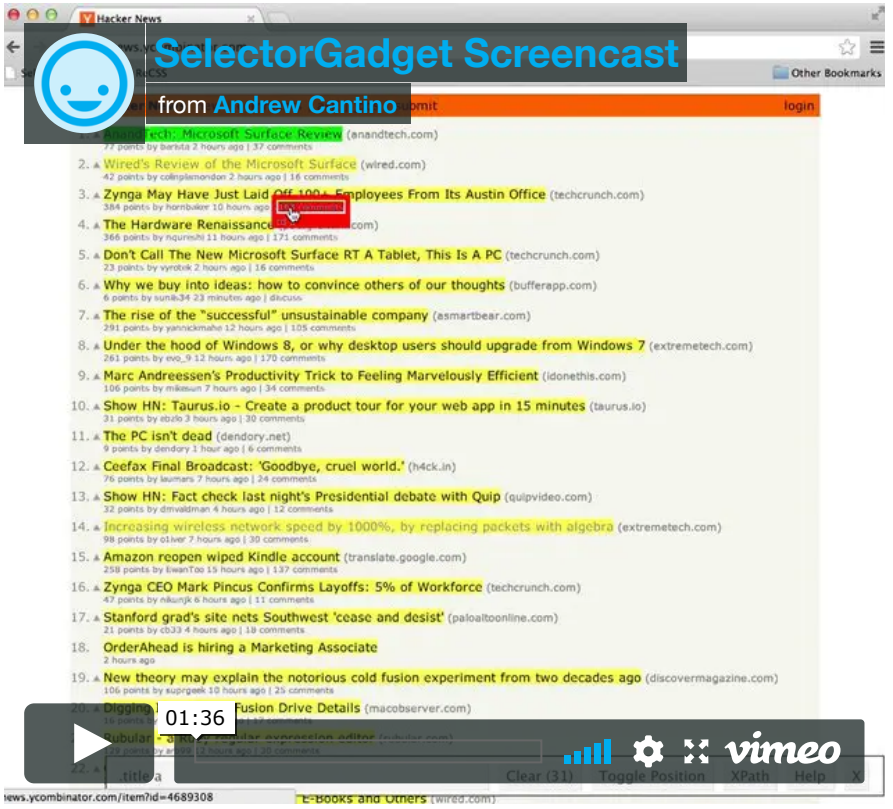
To get only the URLs (value of the `href` attribute):

```
html_swim %>%
  read_html() %>%
  html_nodes(css = "[href]") %>%
  html_attr("href")
```

```
#> [1] "http://www.google.com/maps/?q=27.865501,-82.631997"
#> [2] "http://tampabayfrogman.com/"
```

SelectorGadget

SelectorGadget makes identifying the CSS selector you need by easily clicking on items on a webpage.



Live demo

Exercise

Go to http://books.toscrape.com/catalogue/page-1.html and scrape the first five pages of data on books with regards to their

1. title
2. price
3. star rating

Organize your results in a neatly formatted tibble similar to below.

```
# A tibble: 100 x 3
   title                                                price rating
   <chr>                                                <chr> <chr>
 1 A Light in the Attic                                 £51.… Three
 2 Tipping the Velvet                                   £53.… One
 3 Soumission                                           £50.… One
 4 Sharp Objects                                        £47.… Four
 5 Sapiens: A Brief History of Humankind                £54.… Five
 6 The Requiem Red                                      £22.… One
 7 The Dirty Little Secrets of Getting Your Dream J…    £33.… Four
 8 The Coming Woman: A Novel Based on the Life of t…    £17.… Three
 9 The Boys in the Boat: Nine Americans and Their E…    £22.… Four
10 The Black Maria                                      £52.… One
# … with 90 more rows
```

References

1. Easily Harvest (Scrape) Web Pages. (2020). Rvest.tidyverse.org. https://rvest.tidyverse.org/.

2. W3Schools Online Web Tutorials. (2020). W3schools.com. https://www.w3schools.com/.

3. SelectorGadget: point and click CSS selectors. (2020). Selectorgadget.com. https://selectorgadget.com/.