

# ECON446 : Applications of Cloud Computing and Big Data

Week 1, Lecture 1 | Web-Scraping with BeautifulSoup

Sam Borghese

Tuesday, April 7th, 2024

1. What is Web Scraping?
2. How Websites work
3. Reading HTML
4. HTML in the wild
5. BeautifulSoup and Requests

# What is Web Scraping?

# “Automatic Data Gathering from the Internet”

Alternatively called, Web Harvesting or Data Mining

Formal Definition : Gathering data through the internet through any means other than an API

Pulling information from a URL to be used elsewhere

The screenshot shows a Wikipedia page titled "List of highest-grossing films in the United States and Canada". The page content discusses the list of highest-grossing films in North America, noting that it includes films from both the United States and Canada. It provides a table of the top 10 films, including their rank, title, initial gross (unadjusted), lifetime gross (unadjusted), lifetime gross (adjusted), and year of release. The table is as follows:

Rank	Title	Initial gross (unadjusted)	Lifetime gross (unadjusted)	Lifetime gross (adjusted)	Year
1	Star Wars: The Force Awakens	\$936,662,225	\$936,662,225	\$1,060,713,951	2015
2	Avengers: Endgame	\$858,373,000	\$858,373,000	\$934,680,275	2019

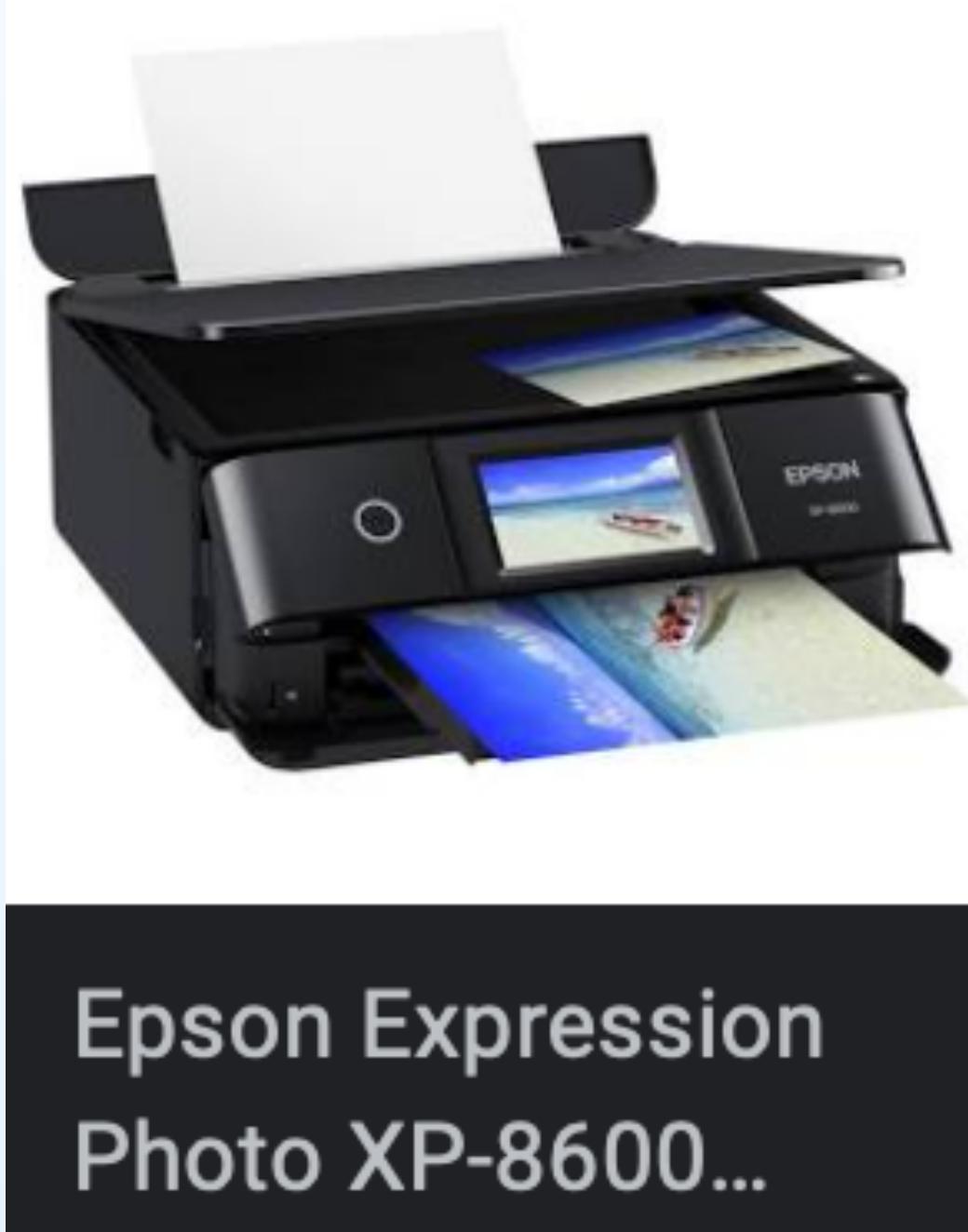
Webscraping

Rank	Title	Initial gross n(unadjusted)	Unadjusted	Adjusted	Year
0	Nan	Nan	Nan	Nan	Nan
1	1.0	Star Wars: The Force Awakens	\$936,662,225	\$936,662,225	\$1,060,713,951
2	2.0	Avengers: Endgame	\$858,373,000	\$858,373,000	\$934,680,275
3	3.0	Spider-Man: No Way Home	\$814,115,070	\$814,115,070	\$814,115,070
4	4.0	Avatar	\$749,766,139	\$785,221,649	\$979,415,544
5	5.0	Top Gun: Maverick	\$718,732,821	\$718,732,821	\$718,732,821
6	6.0	Black Panther	\$700,426,566	\$700,426,566	\$748,305,211
7	7.0	Avatar: The Way of Water	\$682,044,777	\$682,044,777	\$682,044,777
8	8.0	Avengers: Infinity War	\$678,815,482	\$678,815,482	\$710,507,470
9	9.0	Titanic	\$600,788,188	\$674,276,105	\$1,344,908,752
10	10.0	Jurassic World	\$652,270,625	\$653,406,625	\$753,616,100
					2015.0

# WS Use Case Example I.)

Imagine you work for a company that sells Printers

Task : Find all competitors prices for these types of printers



# WS Use Case Example I.)

Task : Find all competitors prices for these types of printers

Ads · Shop Epson Expression Photo XP-8600

Image	Name	Store	Price	Rating
	Epson Expression Photo XP-8600 All-in-One Printer	Adorama	\$199.99	★★★★★ (82)
	Epson Expression Photo XP-8600 All-in-One Printer	B&H Photo-Video...	\$249.99	★★★★★ (82)
	Expression Photo Small-in-One Print...	Dell	\$249.99	★★★★★ (82)
	Epson Expression Photo XP-970 All-in-One Printer	B&H Photo-Video...	\$299.99	★★★★★ (238)
	Epson Expression Photo XP-970 All-in-One Printer	Dell	\$299.99	★★★★★ (238)
	Epson Expression Photo Xp-8600	eBay	\$199.00	Used
	Epson Expression Photo Small-in-On...	Staples	\$299.99	★★★★★ (238)

Web Scraping script

Adorama	B&H Photo an...	Dell	B&H Photo an...	Dell	ebay	Staples
\$199.99	\$249.99	\$249.99	\$299.99	\$299.99	\$199.99	\$299.99

## WS Use Case Example 2.)

Task : Find which competitors have the best web advertising (SEO) for certain key words

# WS Use Case Example 2.)

Task : Find which competitors have the best web advertising (SEO) for certain key words

Google search results for 'printer':

- Printers, Ink & Toner - Best Buy**  
Compare and shop a wide variety of **printer** and ink options from Best Buy, including laser, inkjet, laser, photo, and portable **printers**.  
All Printers · Printer Accessories · Printer Packages · All-in-One Printer
- Home & Office Printers | Amazon.com**  
Results 1 - 12 of 4000+ — Shop for inkjet, laser, wireless, all-in-one, and more best-selling **printers** on Amazon.com for the home, office, or business.  
Inkjet Computer Printer Ink · Computer Printer Ink & Toner · Inkjet Printers
- Printers - Walmart.com**  
Shop for **Printers** in **Printers** & Supplies. Buy products such as Canon PIXMA MG2522 Wired All-in-One Color Inkjet **Printer** at Walmart and save.

Web Scraping script

Printers	Ink
<a href="https://www.bestbuy.com">bestbuy.com</a>	<a href="https://www.dell.com">dell.com</a>
<a href="https://www.amazon.com">amazon.com</a>	<a href="https://www.hp.com">hp.com</a>
<a href="https://www.walmart.com">walmart.com</a>	<a href="https://www.staples.com">staples.com</a>

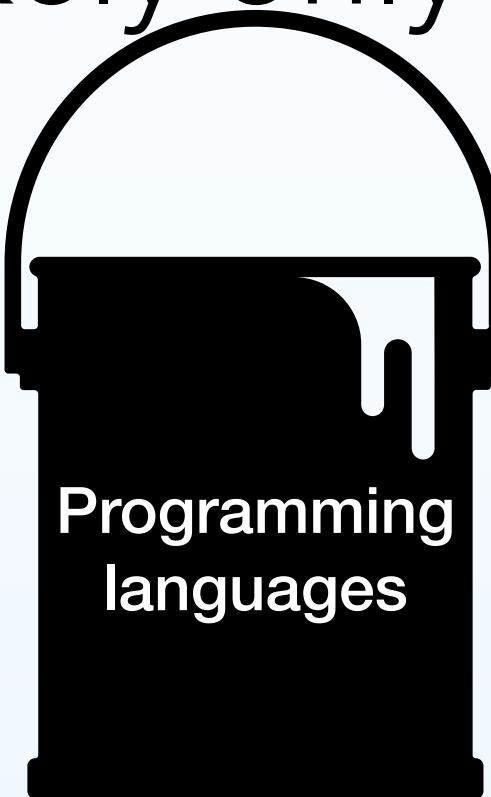
# Use Cases of Web Scraping

- Scraping stock prices into an app API
- Scraping data from YellowPages to generate leads
- Scraping data from a store locator to create a list of business locations
- Scraping product data from sites like Amazon or eBay for competitor analysis
- Scraping sports stats for betting or fantasy leagues
- Scraping site data before a website migration
- Scraping product details for comparison shopping
- Scraping financial data for market research and insights
- Scraping News and Social media for sentiment Analysis

# How Websites Work

# Programming Languages vs. Markup Languages

You have likely only been introduced to Programming Languages



- Contains a set of instructions used to produce various outputs
- Compiled by an interpreter/compiler
- Examples: C, C++, Java, Python
- Used to give instructions to a computer



- Used to annotate that is syntactically distinguishable from the text
- Interpreted by a browser
- Examples: HTML and XML
- Used to present information

# HTML

## Hypertext Mark-up Language

The language used to write code to generate website content

Says how to display information on a webpage

---

```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
```

# Browsers

Browsers act as interpreters and make the HTML code readable.

HTML Code

```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1 id='site_title'>446 Sample Webscraping Site</h1>
    <hr><hr>
    <div class="article">
      <h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
      <p>Some items in you may want to scrape</p>
    </div>
    <hr><hr>
    <div class="article">
      <h2><a href="article_2.html">This is Article 2 with Article Class</a></h2>
      <p>Some useful stuff in the article you may want to scrape</p>
    </div>
    <hr><hr>

    <div class='footer'>
      <p>The Footer </p>
    </div>

    <script src="js/vendor/modernizr-3.5.0.min.js"></script>
    <script src="js/plugins.js"></script>
    <script src="js/main.js"></script>
  </body>
</html>
```

Browser

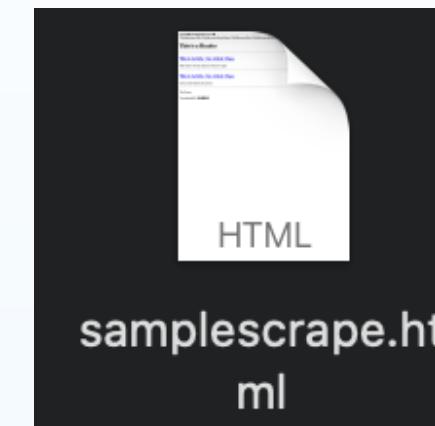


Display to User

A screenshot of a web browser window titled "446 Sample Webscraping Site". The page content includes two articles: "This is Article 1 with an Article Class" and "This is Article 2 with Article Class", each with some descriptive text. At the bottom, there is a footer section with the text "The Footer". The browser's address bar shows the URL "s/windows/Desktop/CourseMaterials/Materials/441B/Code/samplescrape.html".

# HTML has to be in a browser to be viewable

Browsers can open local files or web-based files



A screenshot of a Mac OS X desktop environment showing a web browser window. The browser's address bar is highlighted with a red box and contains the URL `s/windows/Desktop/CourseMaterials/Materials/441B/Code/samplescrape.html`. The main content area of the browser shows the following:

**446 Sample Webscraping Site**

---

**This is Article 1 with an Article Class**

Some items in you may want to scrape

---

**This is Article 2 with Article Class**

Some useful stuff in the article you may want to scrape

---

The Footer

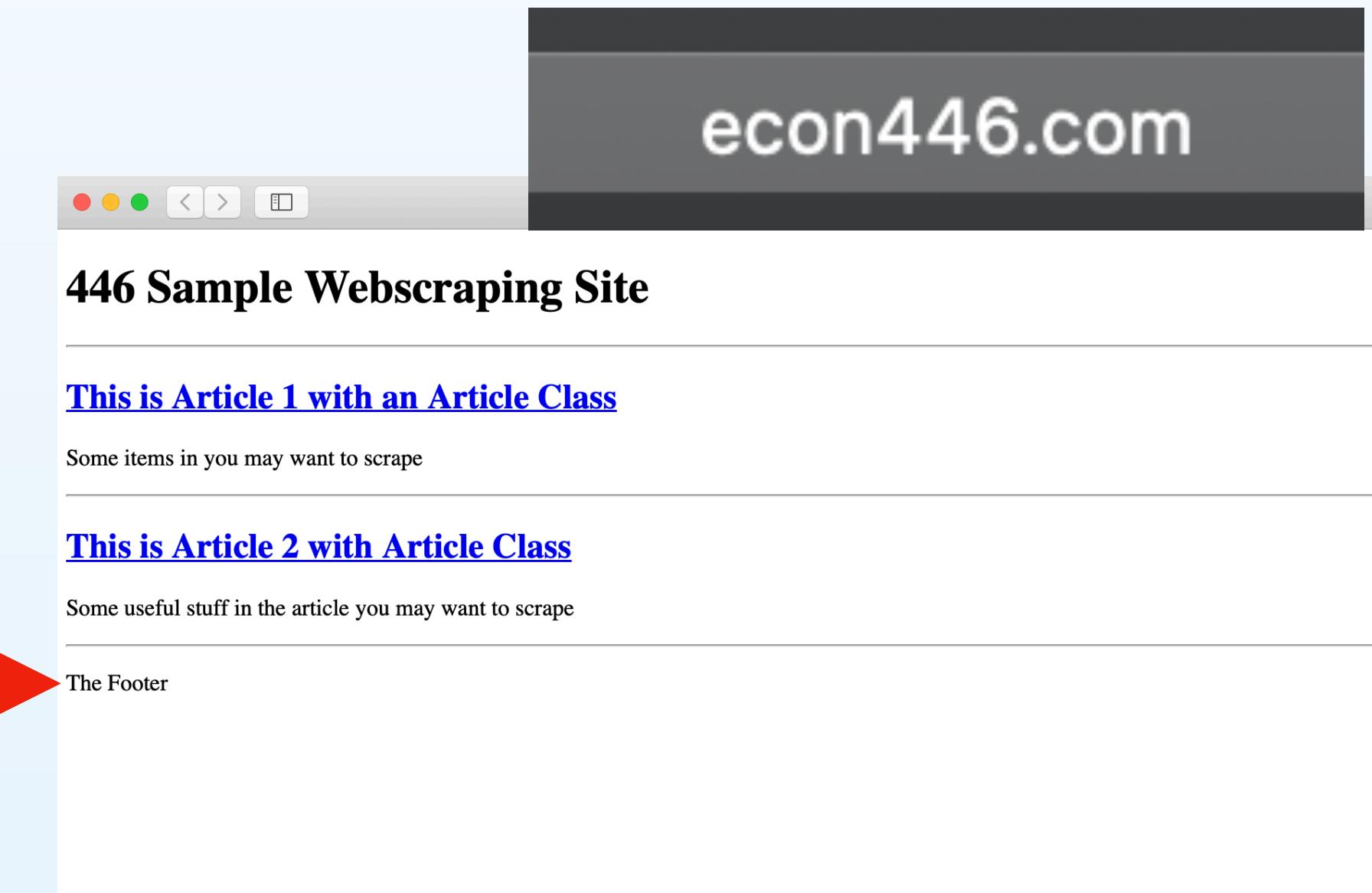
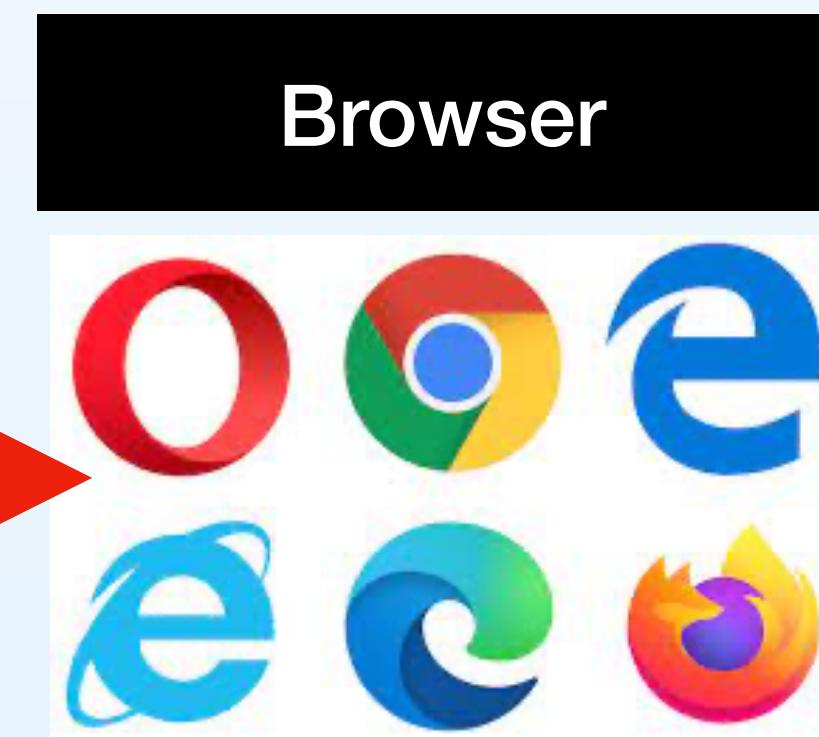
A large red arrow points from the text "Local HTML file opened in my browser" to the browser window.

Local HTML file  
opened in my browser

# Web servers

A web address locates you a companies server that holds the HTML code to be sent your browser

Formal Definition : On the hardware side, a web **server** is a computer that stores web **server** software and a **website's** component files.



# HTTP : Hypertext Transfer Protocol

Websites are requests sent through HTTP and get HTML code back



# Reading HTML

# Reading HTML

HTML uses tags

Opening Tag

<head>

Closing Tag

</head>

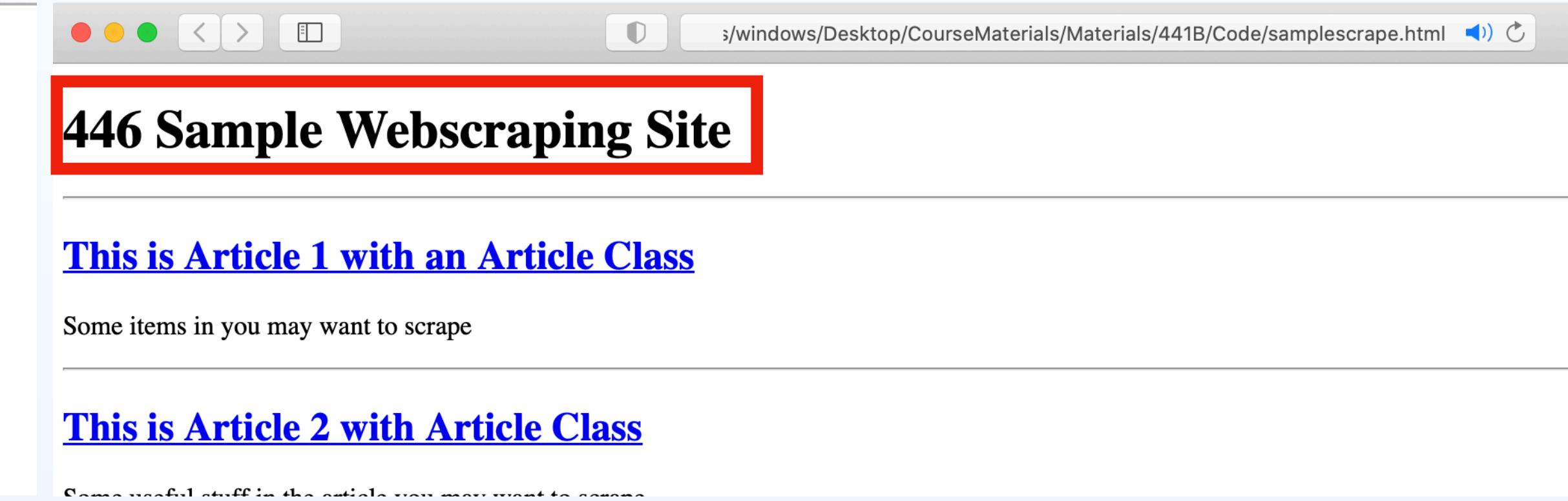
```
<!doctype html>
<html class="no-js" lang="">
  <head>
    </head>
</html>
```

Nested Tag

First Tag you always write is the <html> tag

# Locating Relevant information inside HTML

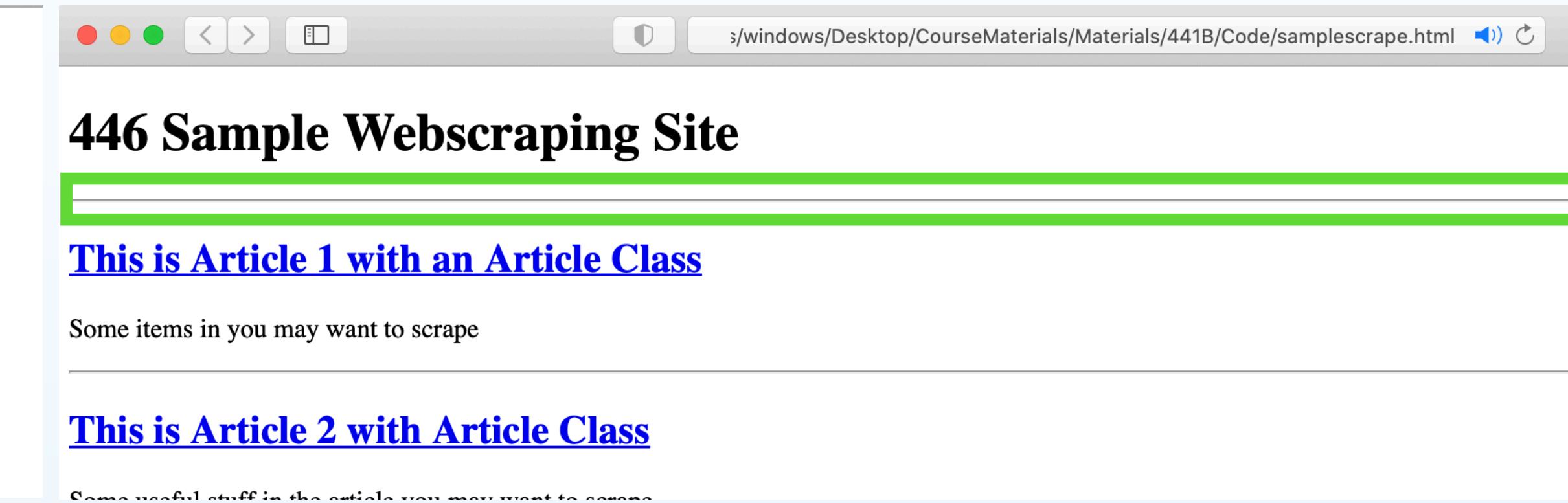
```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1 id='site_title'>446 Sample Webscraping Site</h1>
    <hr></hr>
    <div class="article">
```



h1,h2,h3,h4,h5 are headers of descending sizes

# Locating Relevant information inside HTML

```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1 id='site_title'>446 Sample Webscraping Site</h1>
    <hr></hr>
    <div class="article">
```



A horizontal line across the screen

# Locating Relevant information inside HTML

```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1 id='site_title'>446 Sample Webscraping Site</h1>
    <hr></hr>
    <div class="article">
      <h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
      <p>Some items in you may want to scrape</p>
    </div>
    <hr></hr>
    <div class="article">
      <h2><a href="article_2.html">This is Article 2 with Article Class</a></h2>
      <p>Some useful stuff in the article you may want to scrape</p>
    </div>
    <hr></hr>
    <div class='footer'>
      <p>The Footer </p>
    </div>

    <script src="js/vendor/modernizr-3.5.0.min.js"></script>
    <script src="js/plugins.js"></script>
    <script src="js/main.js"></script>
  </body>
</html>
```

446 Sample Webscraping Site

This is Article 1 with an Article Class

Some items in you may want to scrape

This is Article 2 with Article Class

Some useful stuff in the article you may want to scrape

The Footer

Href's are hyper links that bring you to another webpage

# Hrefs, Anchor Tags and Attributes

## Attributes

```
<div class='footer'>  
    <p>The Footer </p>  
</div>
```

Class attribute here is footer for the div tag

## Anchor Tags

Wraps text using `<a>` to input a hyper link

```
<h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
```

## Hrefs

```
<a href="article_1.html">
```

An attribute of anchor tags that is a link to another page

# Relative vs. Absolute Hyperlinks

Root Domain Name (Zero-level domain)

<https://en.wikipedia.org/>

Absolute  
Hyperlinks

**December 31:** Saint Sylvester's Day (Western Christianity)

- 1225 – Lý Chiêu Hoàng, the only



```
▼<div id="mp-otd">
  ▼<p>
    ▼<b>
      <a href="www.en.wikipedia.org/wiki/Dece
          mber_31" title="December 31">December
          31</a> == $0
    </b>
```

Relative  
Hyperlinks

**December 31:** Saint Sylvester's Day (Western Christianity)

- 1225 – Lý Chiêu



```
▼<div id="mp-otd">
  ▼<p>
    ▼<b>
      <a href="/wiki/December_31" title="Dece
          mber 31">December 31</a> == $0
    </b>
```

# Relative vs. Absolute Hyperlinks

## Relative Hyperlinks

**December 31:** Saint Sylvester's Day (Western Christianity)

- 1225 – Lý Chiêu



```
▼<div id="mp-otd">
  ▼<p>
    ▼<b>
      <a href="/wiki/December_31" title="Dece
          mber 31">December 31</a> == $0
```

Relative hyperlinks add to the root domain

<https://en.wikipedia.org/>

# Reference for Tags

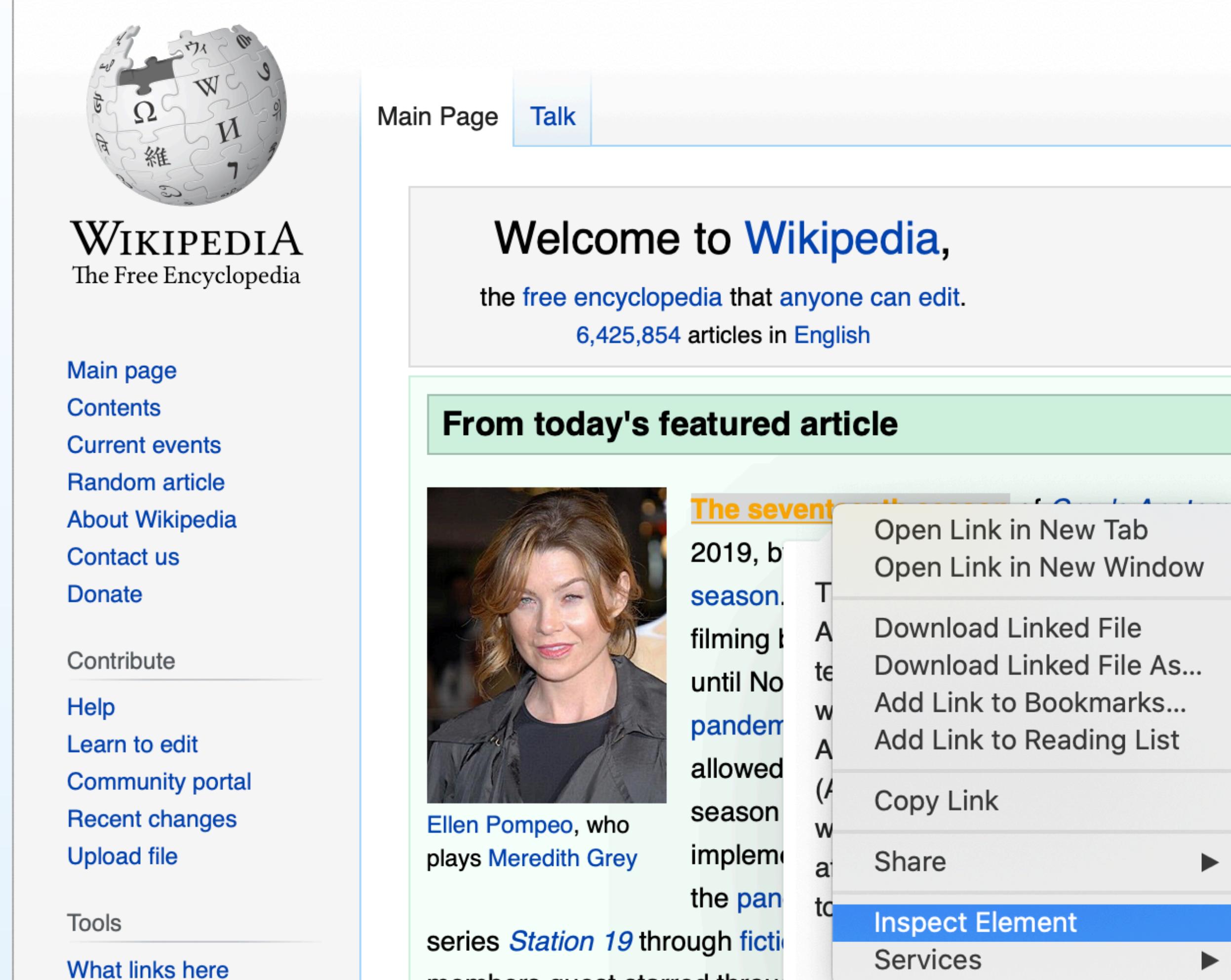
<https://www.w3schools.com/TAGS/default.asp>

# Practice

Write HTML that will display the words “Go Home” as a header and when you click the text it will take you to Apples Home page

# HTML in the wild

# Inspect Element



The screenshot shows the Wikipedia homepage. At the top, there is a navigation bar with 'Main Page' and 'Talk' buttons. Below this, a large banner reads 'Welcome to Wikipedia, the free encyclopedia that anyone can edit. 6,425,854 articles in English'. A green box labeled 'From today's featured article' contains a photo of Ellen Pompeo and some text about her role in 'Station 19'. A context menu is open over the text 'The seventh season...'. The menu items are: Open Link in New Tab, Open Link in New Window, Download Linked File, Download Linked File As..., Add Link to Bookmarks..., Add Link to Reading List, Copy Link, Share, Inspect Element (which is highlighted in blue), and Services.

The HTML for any piece of a website can be located by right clicking and inspecting a given element

# Inspect Element

The screenshot shows the Wikipedia homepage with the developer tools 'Elements' tab open. A yellow highlight box surrounds the text "The seventeenth season" in the "From today's featured article" section. Below the highlight, the browser's developer tools show the corresponding HTML code:

```
> <div id="mp-topbanner" class="mp-bordered">...</div>
<table role="presentation" id="mp-upper">
  <tbody>
    <tr>
      <td id="mp-left" class="MainPageBG mp-bordered">
        <h2 id="mp-tfa-h2" class="mp-h2">...</h2>
        <div id="mp-tfa">
          <div id="mp-tfa-img" style="float: left; margin: 0.5em 0.9em 0.4em 0em;">...</div>
          <p>
            <b>
              <a href="/wiki/Grey%27s_Anatomy_(season_17)" title="Grey's Anatomy (season 17)">The
              seventeenth season</a> = $0
            </b>
            " of "
          </p>
        </div>
      </td>
      <td id="mp-right" class="MainPageBG mp-bordered">
        ...
      </td>
    </tr>
  </tbody>
</table>
```

The highlighted section will correspond to the highlighted HTML code.

# Get Inspect Element

For Safari

- 1. The primary step is to enable the Developer menu. To do so, open the Safari browser, click on Safari -> Preferences.**
  - 2. Click on Advanced. Check the Show Develop menu in menu bar checkbox. ...**
  - 3. The Inspect Element feature is now enabled.**
- 1. Chrome is built in**

# Web scraping process

1.) What are you trying to locate? (Titles, Prices, links, photos, etc.)

2.) What tags and attributes are associated with this information?

The screenshot shows the Wikipedia homepage with the developer tools element inspector open. The 'Elements' tab is selected, displaying the HTML structure of the page. The current selection is on a link to 'Grey's Anatomy (season 17)'. The page content includes a banner for 'Grey's Anatomy', a news section, and a sidebar with categories like 'The arts' and 'History'.

```
<div id="mp-topbanner" class="mp-bordered">...</div>
<table role="presentation" id="mp-upper">
  <tbody>
    <tr>
      <td id="mp-left" class="MainPageBG mp-bordered">
        <h2 id="mp-tfa-h2" class="mp-h2">...</h2>
        <div id="mp-tfa">
          <div id="mp-tfa-img" style="float: left; margin: 0.5em 0.9em 0.4em 0em;">...</div>
          <p>
            <b><a href="/wiki/Grey%27s_Anatomy_(season_17)" title="Grey's Anatomy (season 17)">The seventeenth season</a></b>
            " of "
          </p>
        </div>
      </td>
    </tr>
  </tbody>
</table>
```

# Web scraping process Example I.)

1.) Locate Links

2.) <a> anchor tag.  
Info is inside of the  
href attribute.

Main page  
Contents  
Current events  
Random article  
About Wikipedia  
Contact us  
Donate  
  
Contribute  
Help  
Learn to edit

From today's featured article  
a 164px x 16px Role link

The seventeenth season of *Grey's Anatomy*, 2019, by ABC, as part of a double renewal season. For the 2020–2021 U.S. broadcast season. Filming began in September 2020 while the pandemic was still ongoing, and concluded on November 12, 2020, both delayed as a result of the COVID-19 pandemic. The impact of the pandemic on filming was significant, with many episodes being shot in different locations and some scenes being re-shot due to safety concerns.

Elements Console Sources N

E > E > E > E > E > E > E > E > E > E td#mp-left.MainPage... > E div#mp-topbanner

```
> <div id="mp-topbanner" class="mp-bordered">...</div>
  <table role="presentation" id="mp-upper">
    <tbody>
      <tr>
        <td id="mp-left" class="MainPageBG mp-bordered">
          <h2 id="mp-tfa-h2" class="mp-h2">...</h2>
          <div id="mp-tfa">
            <div id="mp-tfa-img" style="float: left; margin: 0 10px 0 0;">
              <p>
                <b>
                  <a href="/wiki/Grey%27s_Anatomy_(season_17)">seventeenth season</a> = $0
                </b>
                " of "
              </p>
            </div>
          </div>
        </td>
      </tr>
    </tbody>
  </table>
</div>
```

# Web scraping process Example 2.)

1.) Locate the writing blurbs

2.) <p> paragraph tag. All of the text inside of the p tag



The screenshot shows a web page with a header "From today's" and a thumbnail image of a football team huddle. Below the image is a paragraph of text describing the 2010 PapaJohns.com Bowl game.

The 2010 PapaJohns.com Bowl was a postseason bowl game between the college football teams South Carolina Gamecocks and Connecticut Huskies on January 2, 2010, at Legion Field in Birmingham, Alabama; it ended in a 20–7 victory for Connecticut. Both teams had a 7–5 regular season record. Connecticut's tumultuous season had seen a victory at Notre Dame, and the murder of Jasper Howard, their cornerback. Connecticut scored twice in the first quarter: on a one-

file-height="720">  
"/a>  
"/div>  
"/div>  
...  
▼ <p> == \$0  
"The "  
▶ <b>...</b>  
" was a postseason "  
<a href="/wiki/Bowl\_game" title="Bowl game">bowl game</a>  
" between the "  
<a href="/wiki/College\_football" title="College football">college football</a>  
" teams "  
<a href="/wiki/South\_Carolina\_Gamecocks\_football" title="South Carolina Gamecocks football">South Carolina Gamecocks</a>  
" and "  
<a href="/wiki/UConn\_Huskies\_football" title="UConn Huskies football">UConn Huskies</a>  
... ut table#mp-upper tbody tr td#mp-left.MainPageBG.mp-bordered div#mp-tfa p ...  
Styles Computed Layout Event Listeners DOM Breakpoints Properties »  
Filter :hov .cls + □  
element.style {  
}  
.vector-body p { margin: 0.5em 0; } load.php?la...in=vector:1  
p { margin: 0.4em 0 0.5em 0; } load.php?la...in=vector:1  
p { display: block; margin-block-start: 1em; margin-block-end: 1em; margin-inline-start: 0px; margin-inline-end: 0px; } user agent stylesheet  
Inherited from td#mp-left.MainPageBG.mp-b...

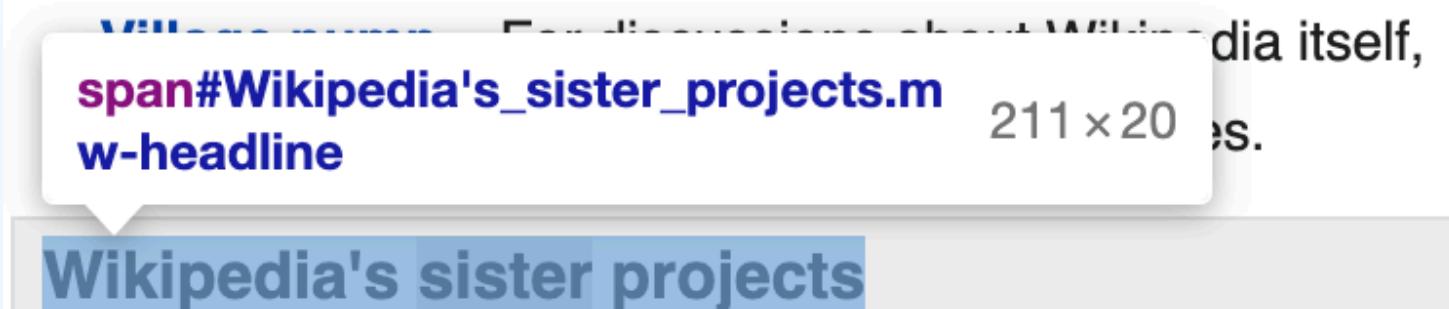
# Web scraping process Example 3.)

1.) Locate the headlines

2.) <span> span tag.

Attribute of class =  
“mw-headline”

- **Reference desk** – Serving as virtual librarians, Wikipedia volunteers tackle your questions on a wide range of subjects.
- **Site news** – Announcements, updates, articles and press releases on Wikipedia and the Wikimedia Foundation.
- **Teahouse** – To ask your first basic questions about contributing to Wikipedia.



```
h2 id="mp-sister" class="mp-h2">
</span>
Wikipedia's sister projects</span>
== $0
/h2>


...


h2 id="mp-lang" class="mp-h2">...


...


iv>
-
PP limit report
...
span#Wikipedia\s_sister_projects.mw-headline ...

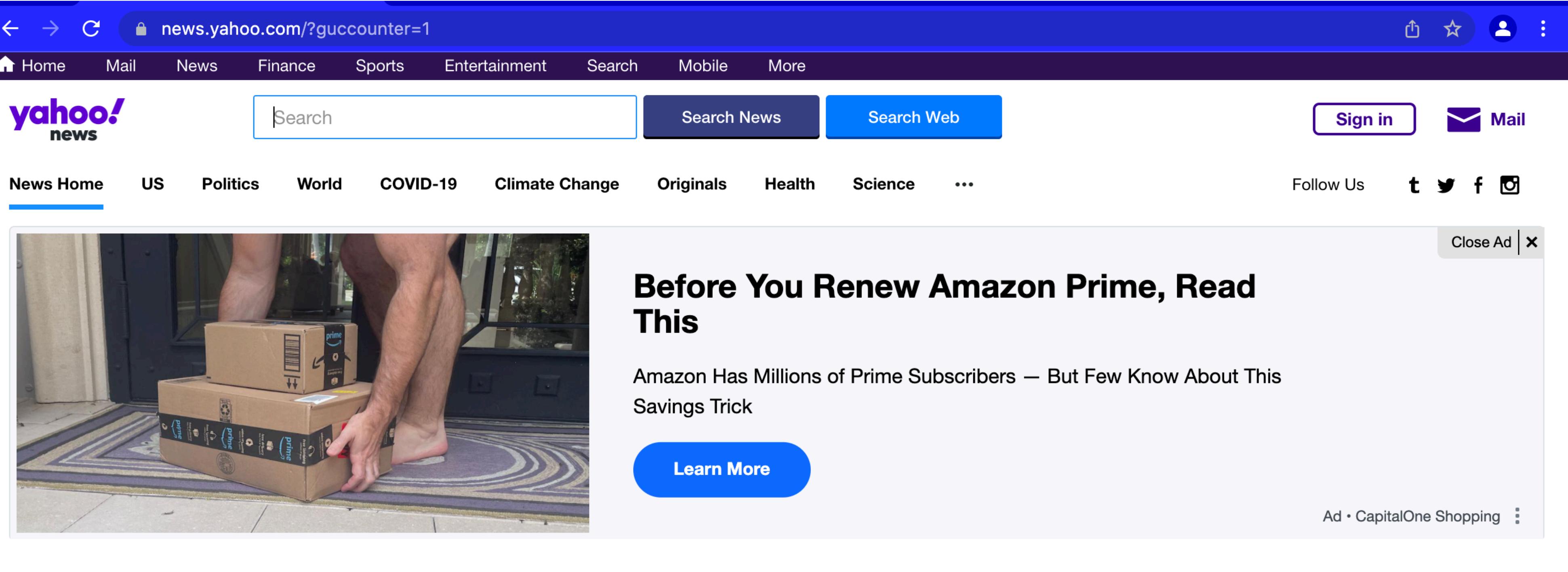
```

Styles    Computed    Layout    Event Listeners    »

# BeautifulSoup and Requests

# Example used in this Section

## Web scraping Yahoo News



The screenshot shows a web browser displaying the Yahoo News homepage at [news.yahoo.com/?guccounter=1](https://news.yahoo.com/?guccounter=1). The page features the Yahoo logo and navigation links for Home, Mail, News, Finance, Sports, Entertainment, Search, Mobile, and More. Below the header is a search bar with options for "Search", "Search News", and "Search Web". On the right side, there are "Sign in" and "Mail" buttons, along with social media links for Twitter, Facebook, and Instagram.

The main content area displays a news article titled "Before You Renew Amazon Prime, Read This" with the subtitle "Amazon Has Millions of Prime Subscribers — But Few Know About This Savings Trick". A blue "Learn More" button is located below the article summary. To the left of the article is a photograph of a person's legs and feet as they step over two stacked Amazon Prime delivery boxes on a porch.

A small "Close Ad" button is visible in the top right corner of the ad area. At the bottom right of the ad, it says "Ad • CapitalOne Shopping".

# Requests

Python Package allows for Python pull HTML code like a browser



```
import requests
```

```
URL = "https://news.yahoo.com"  
page = requests.get(URL)
```

A requests object has many attributes

```
print(type(page))
```

```
<class 'requests.models.Response'>
```

# Requests

Get Attributes and methods of an object

Python Attribute:  
Value/object  
associated with  
an object

`dir(page)`

Python Method:  
Function  
associated with  
an object

```
'apparent_encoding',
'close',
'connection',
['content'],
'cookies',
'elapsed',
'encoding',
'headers',
'history',
'is_permanent_redirect',
'is_redirect',
'iter_content',
'iter_lines',
['json'],
'links',
'next',
'ok',
'raise_for_status',
'raw',
'reason',
'request',
'status_code',
['text'],
['url'])
```

How to call an attribute

```
page.url
'https://news.yahoo.com/'
```

# Requests : Side Note

## Fully Private variables

```
['__attrs__',
 '__bool__',
 '__class__',
 '__delattr__',
 '__dict__',
 '__dir__',
 '__doc__',
 '__enter__',
 '__eq__',
 '__exit__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getstate__']
```

Attributes, Methods and variables that have double underscores  
I.e. \_\_<name>\_\_

Are for the software engineers to use to build the user methods,

# Beautiful Soup

Beautifulsoup

Package for parsing HTML code

```
from bs4 import BeautifulSoup
```

```
soup = BeautifulSoup(page.content, 'html.parser')
```

```
print(type(soup))
```

```
<class 'bs4.BeautifulSoup'>
```

Same “page” from requests

# Beautiful Soup

# BeautifulSoup

# Prettify : Makes HTML code readable

```
print(page.content)
```

b'<!DOCTYPE html><html id="atomic" class="NoJs desktop" lang="en-US"><head><script>window.performance && window.performance.mark('PageStart');</script><meta charset="utf-8"/><meta name="msapplication-TileImage" content="https://guce.yahoo.com/rz/p/yahoo\_frontpage\_en-US\_s\_f\_w\_bestfit\_frontpage.png"/><meta content="A9862C0E6E1BE95BCE0BF3D0298FD58B"/><meta name="referrer" content="origin-when-cross-origin"/><meta name="theme-color" content="#400090"/><meta name="twitter:site" content="@YahooNews"/><meta name="apple-mobile-web-app-capable" content="yes"/><meta name="apple-itunes-app" content="app-id=304158842"/><meta name="apple-mobile-web-app-status-bar-style" content="black-translucent"/><meta content="guce.yahoo.com"/><meta name="twitter:description" content="Get breaking news stories and headlines from Yahoo! News. Get breaking news stories and in-depth photos."/><meta name="twitter:image" content="https://s.yimg.com/ yahoo/default\_logo.png"/><meta name="twitter:image:src" content="https://api.v2/social/images/yahoo\_default\_logo.png"/><meta name="twitter:title" content="Yahoo News - Latest News & Headlines"/><meta name="description" content="Get breaking news stories and headlines from Yahoo! News. Get breaking news stories and in-depth photos."/>

```
soup = BeautifulSoup(page.content, 'html.parser')
```

```
print(soup.prettify())

<!DOCTYPE html>
<html class="NoJs desktop" id="atomic" lang="en-US">
    <head prefix="og: http://ogp.me/ns#">
        <script>
            window.performance && window.performance.mark && window.pe
        ;
        </script>
        <meta charset="utf-8"/>
        <meta content="#6e329d" name="msapplication-TileColor"/>
        <meta content="https://s.yimg.com/rz/p/yahoo_frontpage_en-l
ng" name="msapplication-TileImage"/>
        <meta content="A9862C0E6E1BE95BCE0BF3D0298FD58B" name="msva
        <meta content="unsafe-url" name="referrer"/>
        <meta content="#400090" name="theme-color"/>
        <meta content="on" name="twitter:dnt"/>
        <meta content="@YahooNews" name="twitter:site"/>
        <meta content="Yahoo News - Latest News & Headlines" na
        <meta content="summary" name="twitter:card"/>
```

# Beautiful Soup

Beautifulsoup

## Find Method

Use the HTML Tag to locate the first instance of that tag

```
soup.find('h3')
```

```
<h3 class="M(0) Mend(9px) Mt(4px) Fz(12px) LineClamp(2,2.6em) LineClamp(3,4em)--md1100 T  
(70%) Start(2px) Td(u):h" data-reactid="32">Authorities find hundreds of dumped Amazon p  
ackages</h3>
```

```
type(soup.find('h3'))
```

```
bs4.element.Tag
```

```
dir(soup.find('h3'))
previous_sibling ,
previous_siblings ,
recursiveChildGenerator ,
renderContents ,
replaceWith ,
replaceWithChildren ,
replace_with ,
replace_with_children ,
select ,
select_one ,
setup ,
smooth ,
sourceline ,
sourcepos ,
string ,
strings ,
stripped_strings ,
text ,
unwrap ,
wrap ]
```

# Beautiful Soup

## Find Method

```
dir(soup.find('h3'))  
[  
    'previous_sibling',  
    'previous_siblings',  
    'recursiveChildGenerator',  
    'renderContents',  
    'replaceWith',  
    'replaceWithChildren',  
    'replace_with',  
    'replace_with_children',  
    'select',  
    'select_one',  
    'setup',  
    'smooth',  
    'sourceline',  
    'sourcepos',  
    'string',  
    'strings',  
    'stripped_strings',  
    'text',  
    'unwrap',  
    'wrap']
```

Beautifulsoup

Most likely method you will encounter is `text`

```
soup.find('h3').text
```

```
'Authorities find hundreds of dumped Amazon packages'
```

# Beautiful Soup

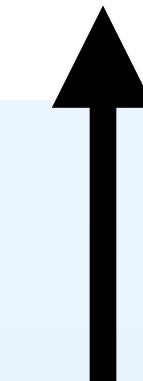
Beautifulsoup

## Findall Method

Let's you be able to take all of the tags you are searching for

```
soup.findAll('h3')
```

```
[<h3 class="M(0) Mend(9px) Mt(4px) Fz(12px) LineClamp(2,2.6em) LineClamp(3,4em)--md1100  
T(70%) Start(2px) Td(u):h" data-reactid="32">Authorities find hundreds of dumped Amazon  
packages</h3>,  
 <h3 class="M(0) Mend(9px) Mt(4px) Fz(12px) LineClamp(2,2.6em) LineClamp(3,4em)--md1100  
T(70%) Start(2px) Td(u):h" data-reactid="37">Trump's plans for the future are becoming c  
learer</h3>,
```



A list of bs tag objects

# Beautiful Soup

Beautifulsoup

## Findall Method

If we wanted to get the text from every tag  
=> loop through list and pull out text attribute

These two statements  
are equivalent

```
titles = soup.find_all('h3')

texts = []
for title in titles:
    texts.append(title.text)

df = pd.DataFrame(texts)
```

```
titles = soup.find_all('h3')
df = pd.DataFrame([title.text for title in titles])
```

# Beautiful Soup

Beautifulsoup

## Findall Loop Output

df

0

- 0 Authorities find hundreds of dumped Amazon pac...
- 1 Trump's plans for the future are becoming clearer
- 2 'People just have no sense of how bad this pro...
- 3 'Sex for lies' false testimony kept man in ja...
- 4 Biden's climate declaration 'a slap in the face'
- 5 'River Dave' arrested after returning to live ...
- 6 Florida deputies who shared an infant child to...
- 7 Rita Ora Is Manifesting A Better Year By Rocki...

What'd we do here?

Pulled all the  
headlines into a  
dataframe

# Methods: Side Notes

A method is a function associated with a specific objects

Get some basic documentation of a method  
Run “*object.method*” without the soft brackets

?soup.find

**Signature:** soup.find(name=None, attrs={}, recursive=True, text=None, \*\*kwargs)  
**Docstring:**

Look in the children of this PageElement and find the first PageElement that matches the given criteria.

All `find_*` methods take a common set of arguments. See the online documentation for detailed explanations.

:param name: A filter on tag name.  
:param attrs: A dictionary of filters on attribute values.  
:param recursive: If this is True, `find()` will perform a recursive search of this PageElement's children. Otherwise, only the direct children will be considered.  
:param limit: Stop looking after finding this many results.

# Parent Children

## HTML Tags are set up in a hierarchical tree

```
<!doctype html>
<html class="no-js" lang="">
  <head>
    <title>446 Heading of the Webpage</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1 id='site_title'>446 Sample Webscraping Site</h1>
    <hr></hr>
    <div class="article">
      <h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
      <p>Some items in you may want to scrape</p>
    </div>
    <hr></hr>
    <div class="article">
      <h2><a href="article_2.html">This is Article 2 with Article Class</a></h2>
      <p>Some useful stuff in the article you may want to scrape</p>
    </div>
    <hr></hr>

    <div class='footer'>
      <p>The Footer </p>
    </div>

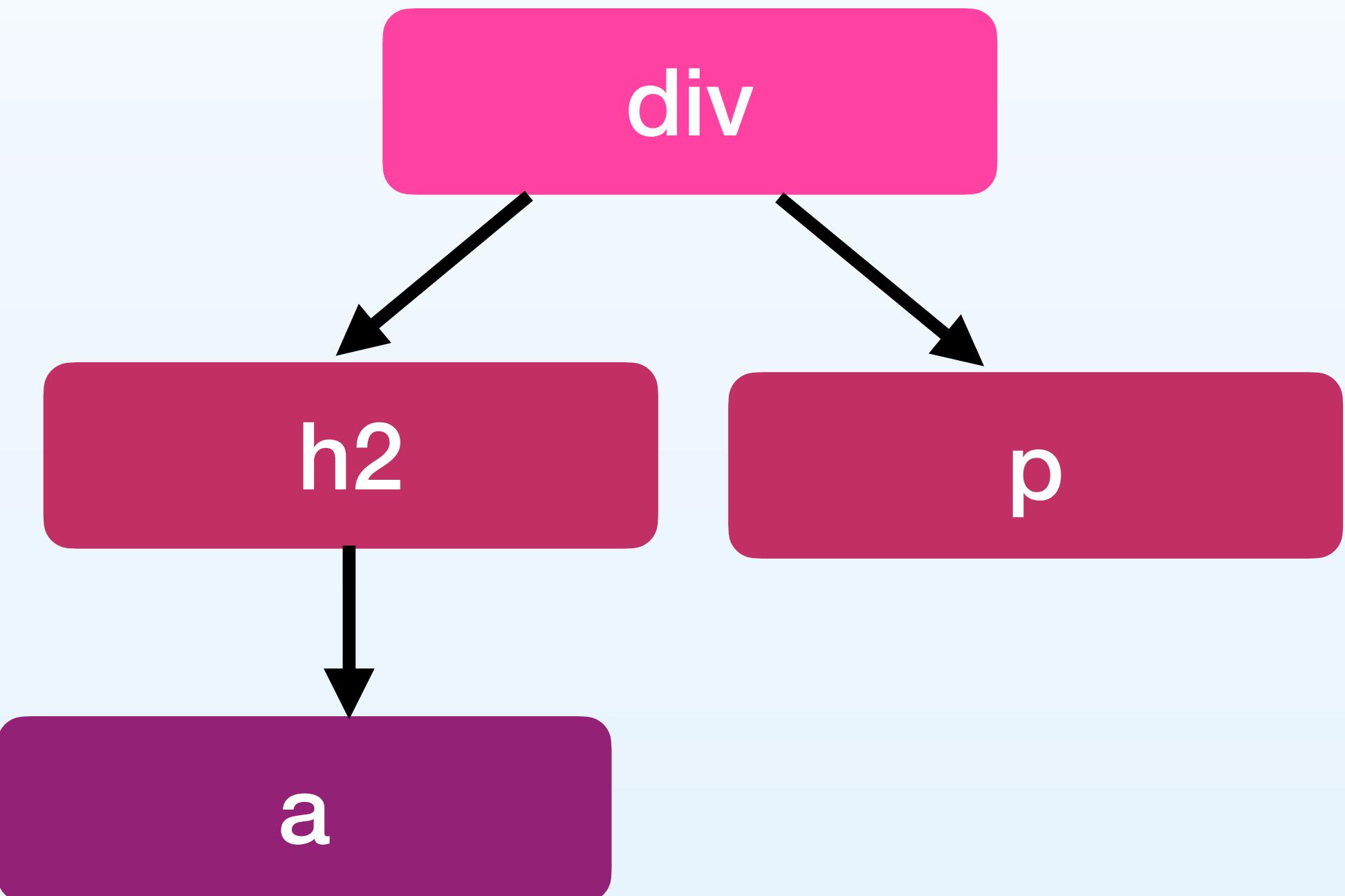
    <script src="js/vendor/modernizr-3.5.0.min.js"></script>
    <script src="js/plugins.js"></script>
    <script src="js/main.js"></script>
  </body>
</html>
```

# Parent Child

There are relationships of tags inside a tree that allow you to navigate the HTML

```
<div class="article">
  <h2>
    <a href="article_2.html">
      This is Article 2 with Article Class
    </a>
  </h2>
  <p>
    Some useful stuff in the article you may want to scrape
  </p>
</div>
```

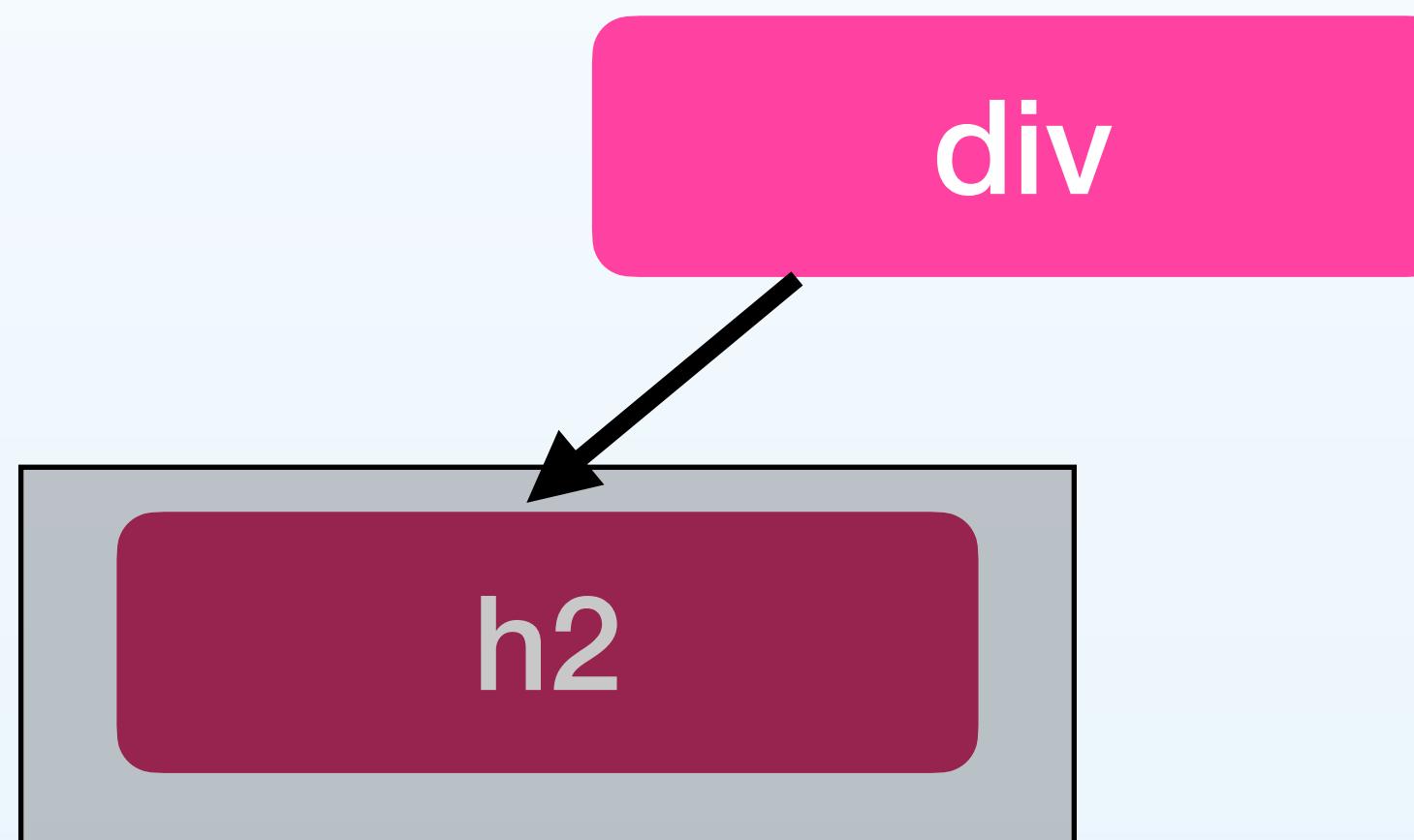
```
soup.div
<div class="article">
  <h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
  <p>Some items in you may want to scrape</p>
</div>
```



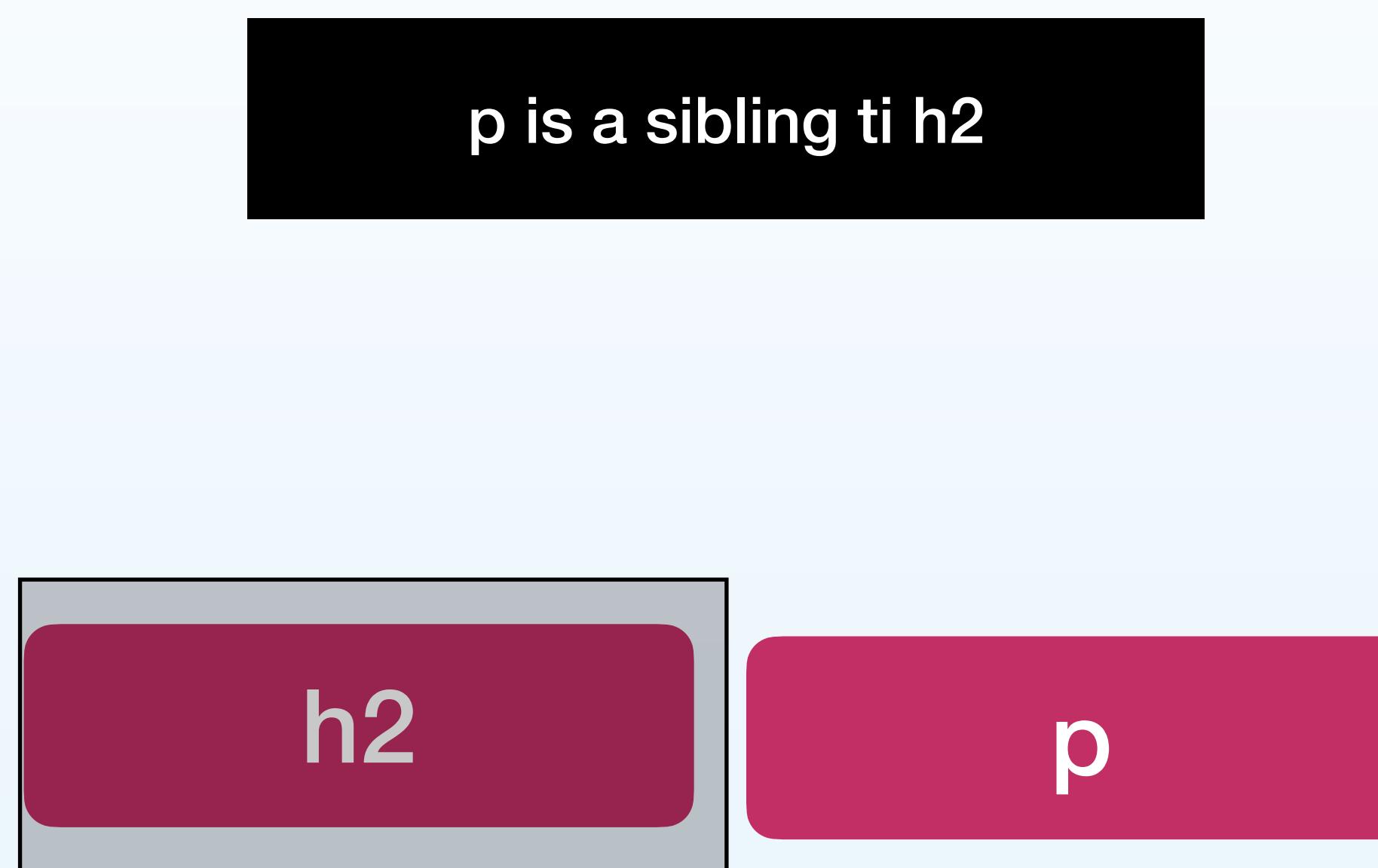
# Relationships

There are relationships of tags inside a tree that allow you to navigate the HTML

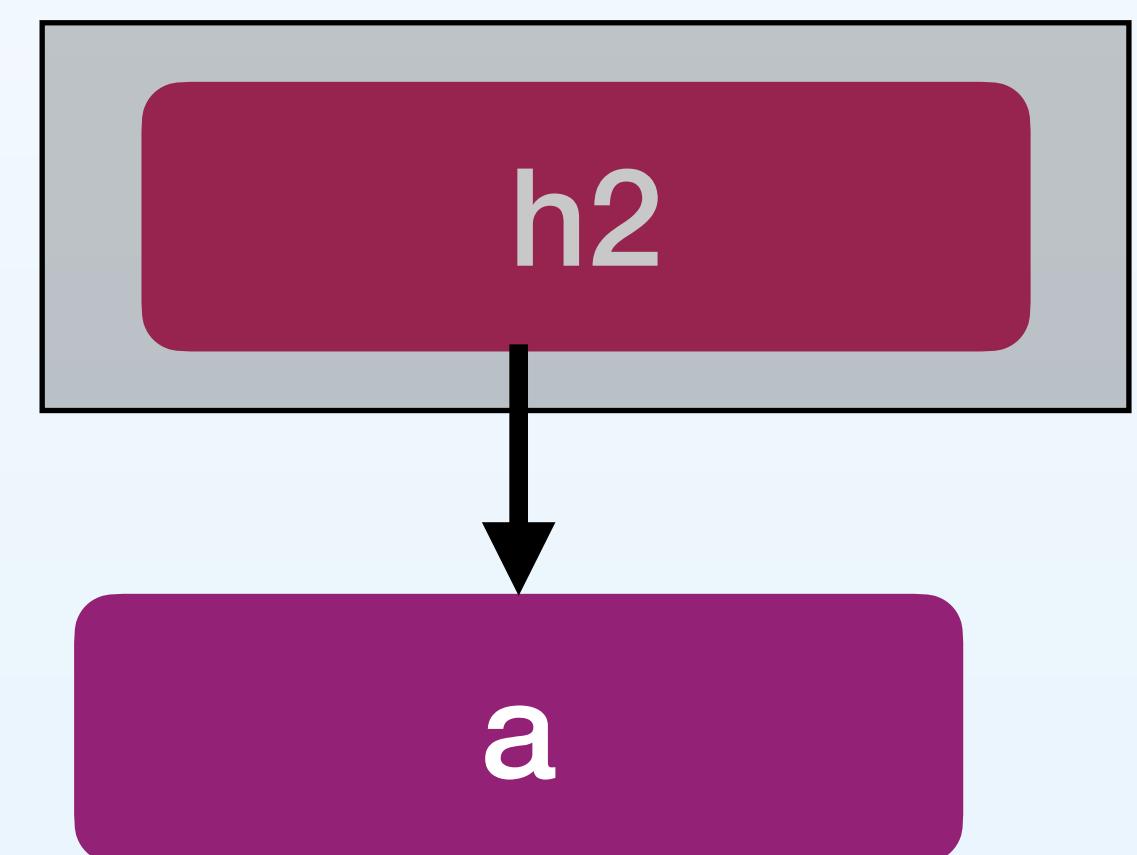
Div is a parent to h2



p is a sibling to h2



a is a child of h2



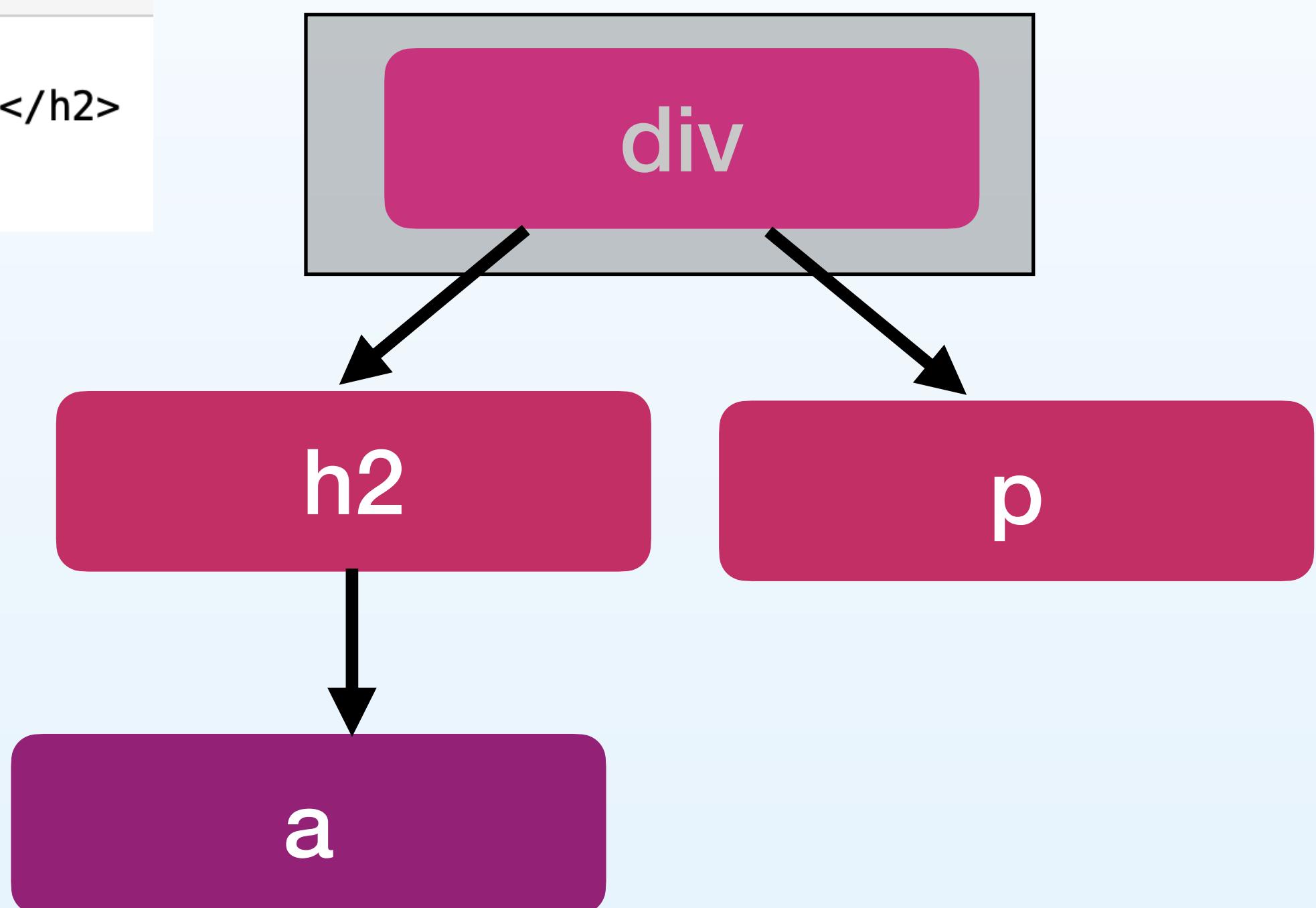
# Tag Object

Saving a tag object allows you to move through Parents and Children

```
soup.div
```

```
<div class="article">
<h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
<p>Some items in you may want to scrape</p>
</div>
```

```
tag = soup.div
```



# Example fo Tag Methods

Get First Child

```
tag.findChild()
```

```
<h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
```

Get Next Child

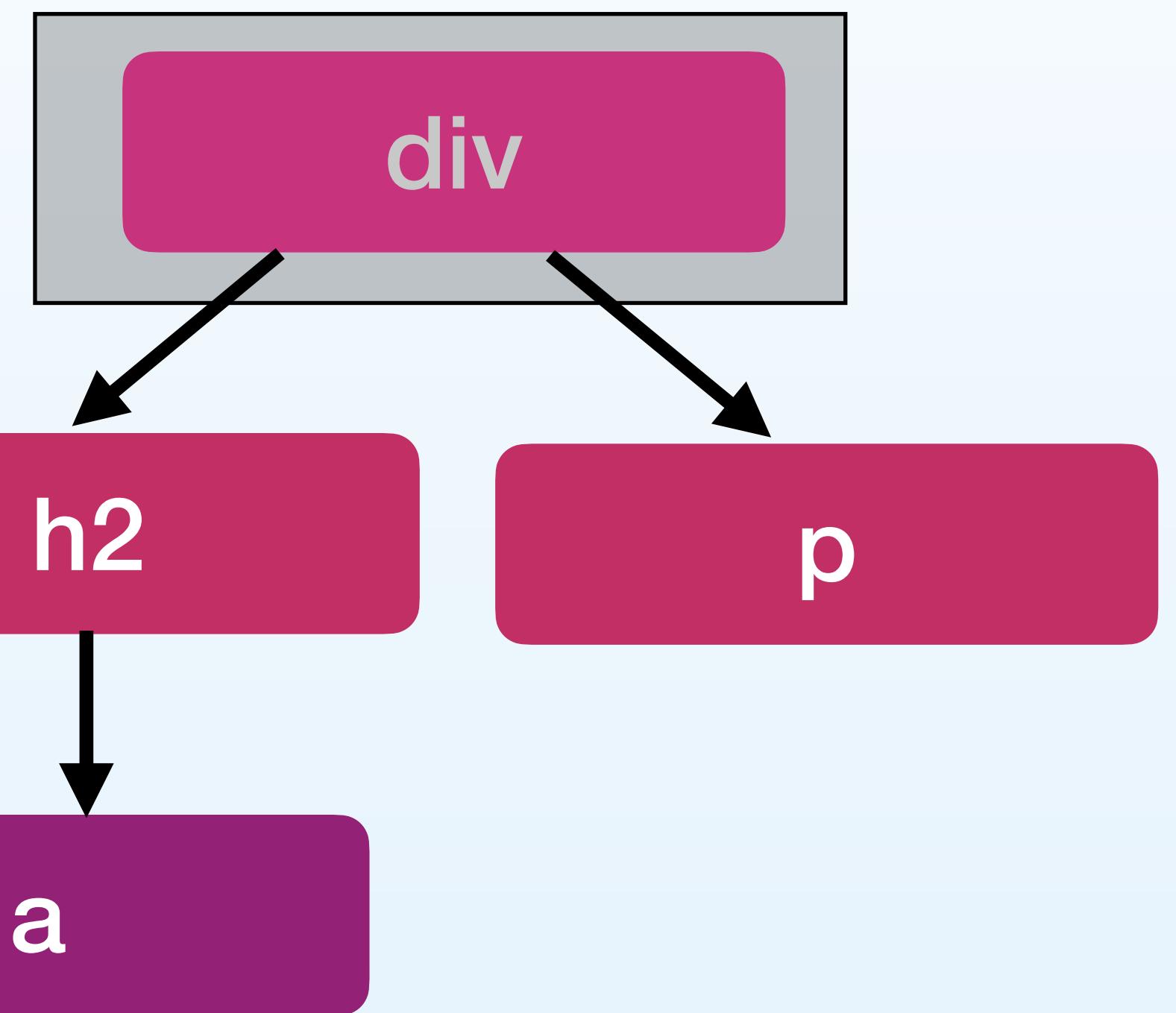
```
tag.findChild().next_sibling
```

```
<p>Some items in you may want to scrape</p>
```

Get a Parent

```
tag.findChild().parent
```

```
<div class="article">
<h2><a href="article_1.html">-
<p>Some items in you may want
</div>
```



# Example fo Tag Methods #2

Find a child with a specific tag

```
tag.findChild("p")
```

```
<p>Some items in you may want to scrape</p>
```

Get all tags lower in the tree

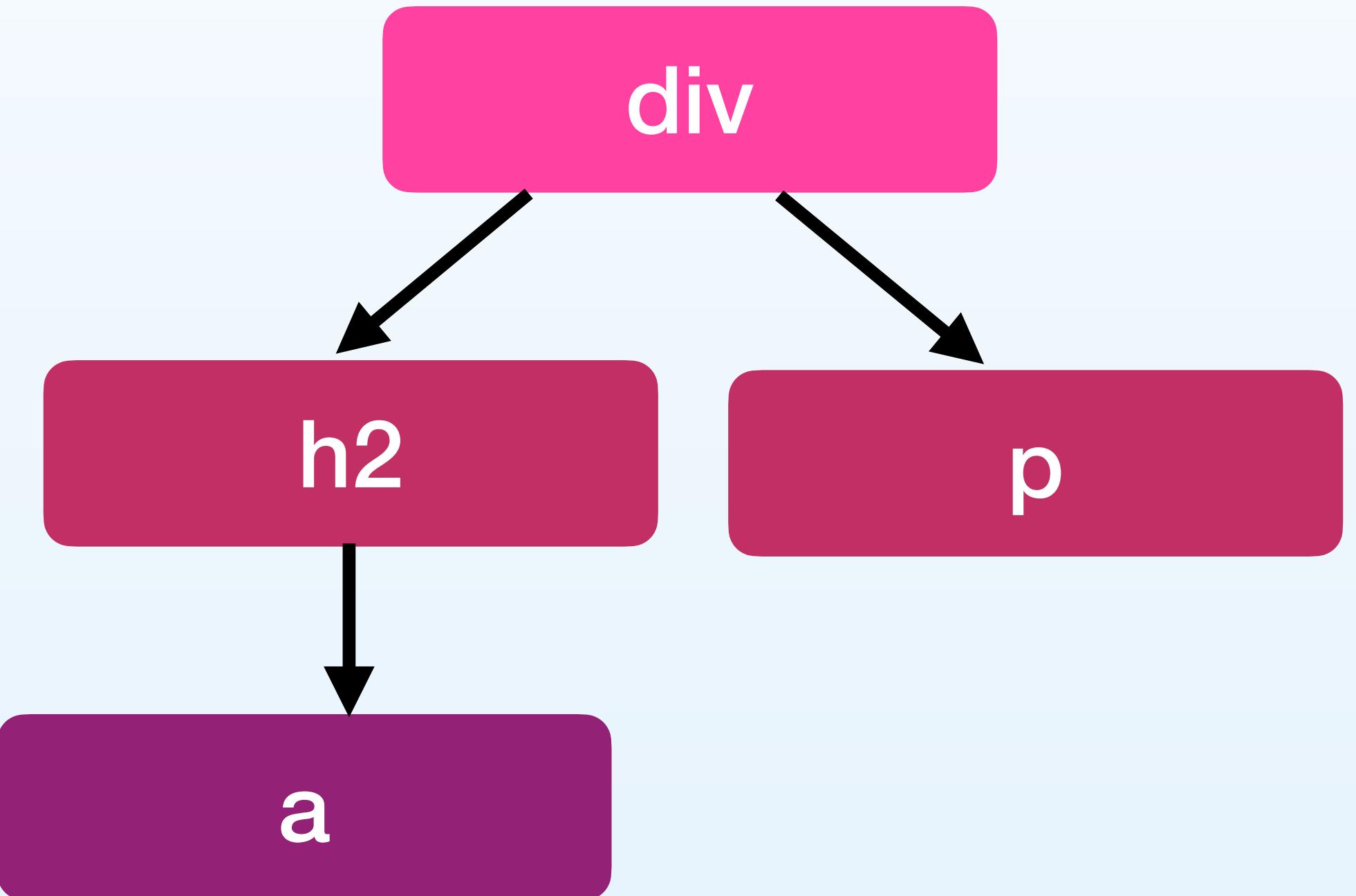
```
tag.descendants
```

```
<generator object Tag.descendants at 0x7f9c84d7f580>
```

Get attributes of a Tag

```
tag.attrs
```

```
{'class': ['article']}
```



# Common case necessary for this type of navigation

Inside of a single object we want multiple pieces of info



U.S. Miami Herald

## Florida deputies who shared an infant child took their own lives, their sheriff said

St. Lucie County Sheriff Ken Mascara announced the deaths of Deputy Clayton Osteen and Deputy Victoria Pacheco on Tuesday via social media. Mascara wrote "it is impossible for us...



...

0 1

- |   |                                                   |
|---|---------------------------------------------------|
| 0 | Authorities find hundreds of dumped Amazon pac... |
| 1 | Trump assembles a sort of shadow political party  |
| 2 | Man served 37 years on testimony from false wi... |
| 3 | Podcasts spread election misinformation ahead ... |
| 4 | Biden's climate declaration 'a slap in the face'  |

# Common case necessary for this type of navigation

Inside of a single object we want multiple pieces of info

The screenshot shows a news article from the U.S. Miami Herald. At the top, there is a large image of a white sheriff's patrol car. Below the image, the headline reads: "Florida deputies who shared an infant child took their own lives, their sheriff...". A sub-headline below the main one states: "St. Lucie County Sheriff Ken Mascara announced the deaths of Deputy Clayton Osteen and Deputy Victoria Pacheco on...".

Two arrows point from specific code snippets at the bottom to different parts of the article:

- An arrow points from the code snippet `div.FI(start).Pos(r).Mt(2px).W(26.5%).Maw(220px)` to the image of the sheriff's car.
- An arrow points from the code snippet `a.js-content-viewer.wafer-caas.Fw(b).Fz(20px).Lh(23px).Fz(17px)--sm1024.Lh(19px)...` to the headline text.

The code snippets are part of a larger DOM structure shown in the developer tools:

- The top snippet is associated with the headline area: `<div class="Cf"> == $0`
- The left snippet is associated with the image area: `<div class="Fl(start) Pos(r) Mt(2px) W(26.5%) Maw(220px)">...</div> == $0`
- The right snippet is associated with the link text: `<a class="js-content-viewer wafer-caas Fw(b) Fz(20px) Lh(23px) Fz(17px)--sm1024 Lh(19px)" href="/florida-deputies-shared-infant-child-160026496.html" data-uuid="d40e4b56-37b9-3983-9d18-fd8544d4afb1" data-wf-caas-prefetch="1"> == $0`

# I.) Locate the tag

## 2.) Locate the Children associated with the tag

```
print(soup.find("div", attrs = {"class" : "Cf"}).prettify())  
  
<div class="Cf" data-reactid="55">  
  <div class="Fl(start) Pos(r) Mt(2px) W(26.5%) Maw(220px)" data-reactid="56">  
    <div class="H(0) 0v(h) Bdrs(2px)" data-reactid="57" style="padding-bottom:56%;">  
        
    </div>  
  </div>
```

```
tag = soup.find("div", attrs = {"class" : "Cf"})
```

```
tag.findChild("img").attrs["src"]
```

```
'https://s.yimg.com/uu/api/res/1.2/KK2_HJ4lnkagg76sDQa8TQ--~B/Zmk9c3RyaW07aD0xMjM7cT04MDt3PTIyMDthcHBpZD15dGFjaHlvbg--/https://s.yimg.com/uu/api/res/1.2/UBi0c9kA9Q9NJ1h0njjymA--~B/aD0z0DM103c9NTc1MjthcHBpZD15dGFjaHlvbg--/https://media.zenfs.com/en/ap.org/b0a120c4ab657177620dd797e4a61777.cf.jpg'
```

# Headers Brief

# Sites are getting more and more weary of Automation and their data being used by competitors

We will cover in depth next week but for todays example know “User-Agents” can solve robot blocking issues

```
<h1>
  Please verify you are a human
</h1>
<p>
  Click "I am not a robot" to continue
</p>
... . . . . .
```

```
headers = {
  'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36'}
```

```
page = requests.get(URL, headers=headers)
```

# In Class Example

Theory : Rental price fluctuations are an indicator of a recession/booming economy

Write a function to web scrape the front page of “Trulia” Housing and get a distribution of low/high end housing costs