ECON446 :Applications of Cloud Computing and Big Data

Week 1, Lecture 1 | Web-Scraping with Beautiful Soup

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

https://en.wikipedia.org/wiki/List_of_highest-grossing_films_in_the_United_States_and_Canada

WIKIPEDIA
The Free Encyclopedia

List of highest-grossing films in the United States and Canada

7 languages

Article Talk Tools

From Wikipedia, the free encyclopedia

The following is a **list of the highest-grossing films in the United States and Canada**, a market known in the film industry as the "North American box office",[1][2] and where "gross" is defined in US dollars.

Not adjusted for inflation [edit]

This is a list of the highest-grossing films in the U.S. and Canada,[3] a market known in the film industry as the **North American box office**, or as the **domestic box office** within the U.S. itself.[1][2] The chart is ranked by lifetime gross, and for comparison, the figures adjusted for the effects of inflation are also listed, using the U.S. consumer price index;[4] a film's earnings from its initial release are also included to provide a basis for comparison between films released around the same time.

† *Background shading indicates films playing in the week commencing 31 March 2023 in theaters around the world.*

Rank	Title	Initial gross (unadjusted)	Lifetime gross		Year
			Unadjusted	Adjusted	
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	NaN
1	1.0	Star Wars: The Force Awakens	$936,662,225	$936,662,225	$1,060,713,951	2015.0
2	2.0	Avengers: Endgame	$858,373,000	$858,373,000	$934,680,275	2019.0
3	3.0	Spider-Man: No Way Home	$814,115,070	$814,115,070	$814,115,070	2021.0
4	4.0	Avatar	$749,766,139	$785,221,649	$979,415,544	2009.0
5	5.0	Top Gun: Maverick	$718,732,821	$718,732,821	$718,732,821	2022.0
6	6.0	Black Panther	$700,426,566	$700,426,566	$748,305,211	2018.0
7	7.0	Avatar: The Way of Water	$682,044,777	$682,044,777	$682,044,777	2022.0
8	8.0	Avengers: Infinity War	$678,815,482	$678,815,482	$710,507,470	2018.0
9	9.0	Titanic	$600,788,188	$674,276,105	$1,344,908,752	1997.0
10	10.0	Jurassic World	$652,270,625	$653,406,625	$753,616,100	2015.0

WS Use Case Example 1.)

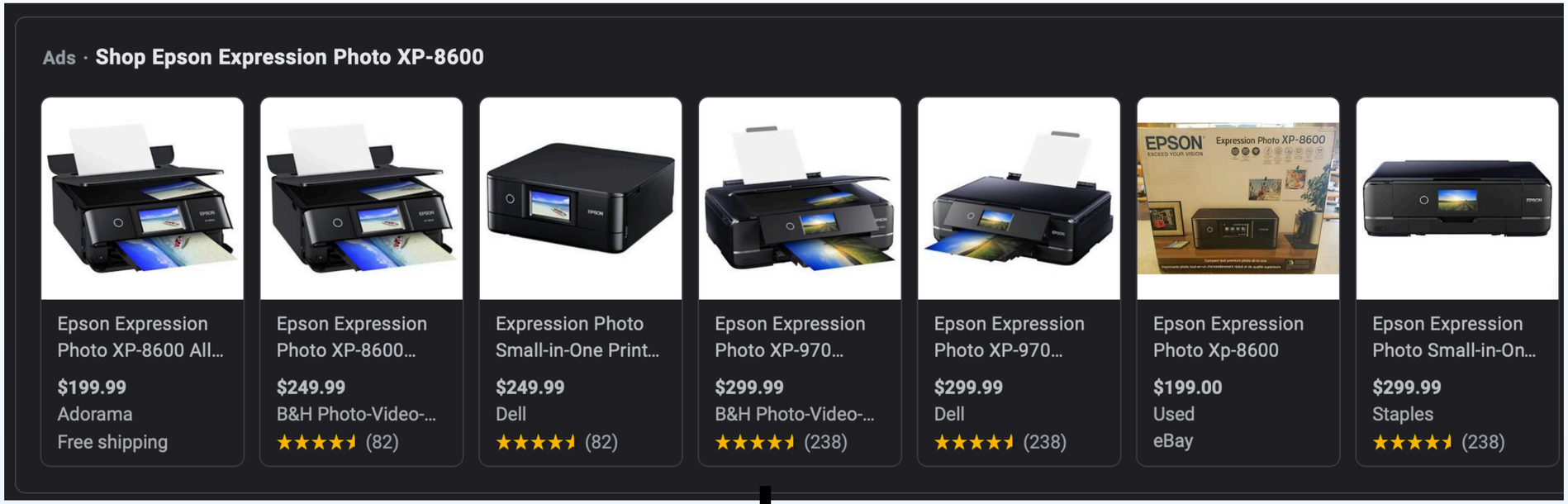
Imagine you work for a company that sells Printers

Task : Find all competitors prices for these types of printers

Epson Expression Photo XP-8600...

WS Use Case Example 1.)

Task : Find all competitors prices for these types of printers



Web Scraping script

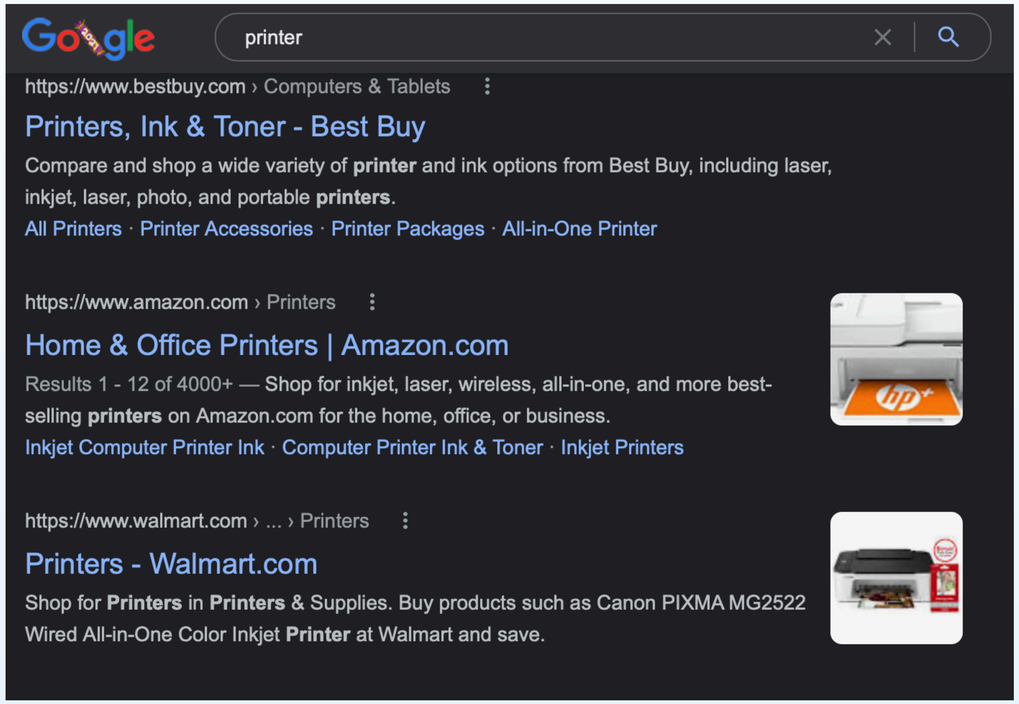
Adorama	B&H Photo an...	Dell	B&H Photo an...	Dell	ebay	Staples
$199.99	$249.99	$249.99	$299.99	$299.99	$199.99	$299.99

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



Web Scraping script

Printers	Ink
bestbuy.com	dell.com
amazon.com	hp.com
walmart.com	staples.com

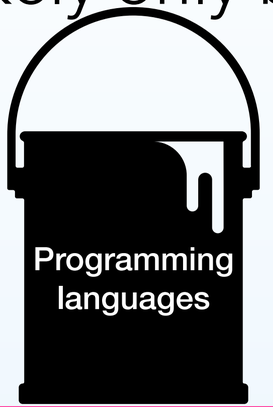
Use Cases of Web Scraping

- 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 intpretter/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

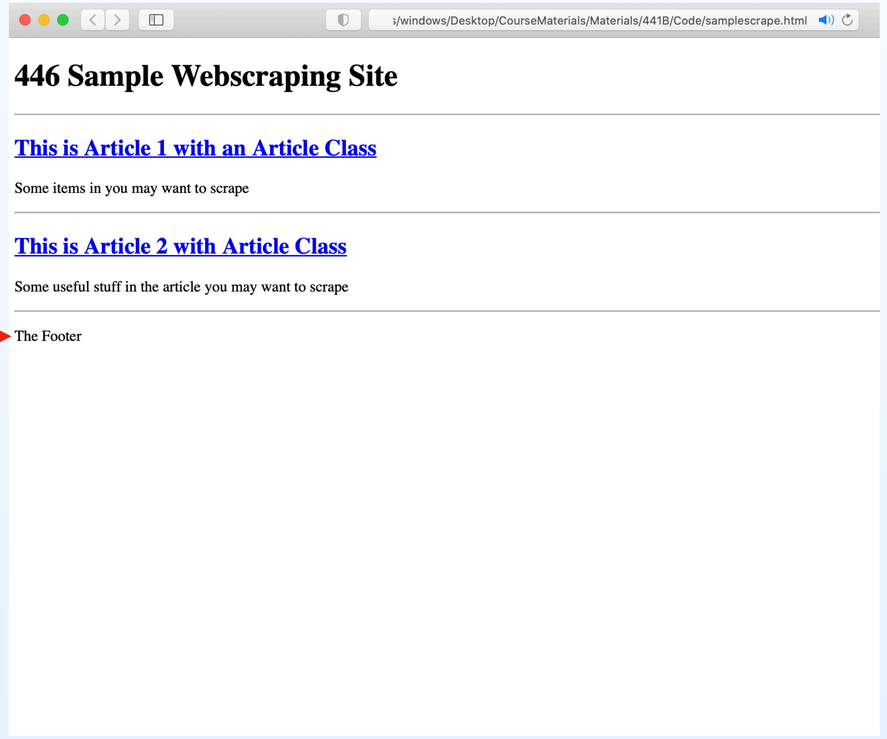
```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <title>446 Heading of the Webpage</title>
        <meta charset="utf-8">
        <link rel="stylesheet" href="css/normalize.css">
        <link rel="stylesheet" href="css/main.css">
    </head>
    <body>
        <h1 id='site_title'>446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

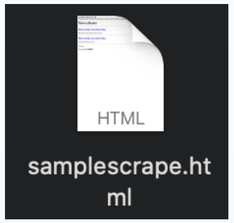
Browser

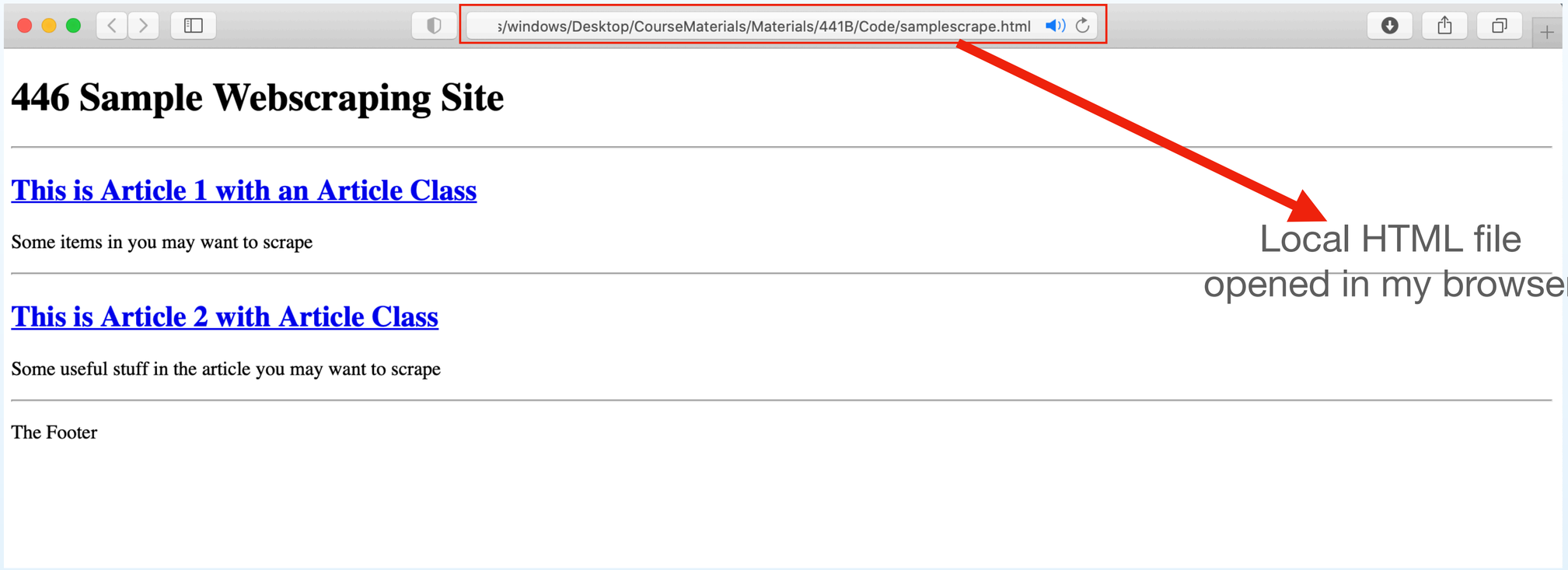
Display to User



HTML has to be in a browser to be viewable

Browsers can open local files or web-based files





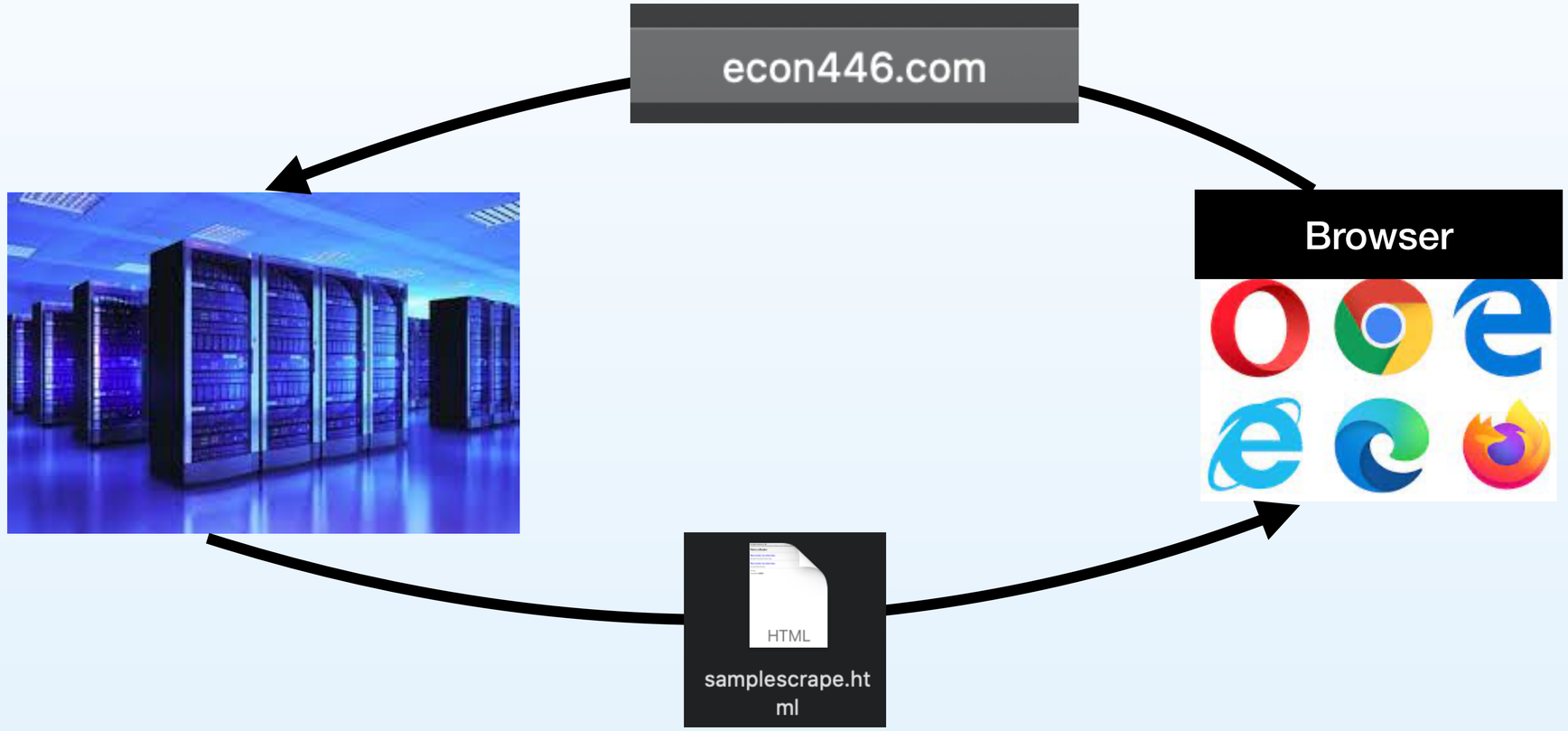
Web servers

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	Closing Tag
<head>	</head>

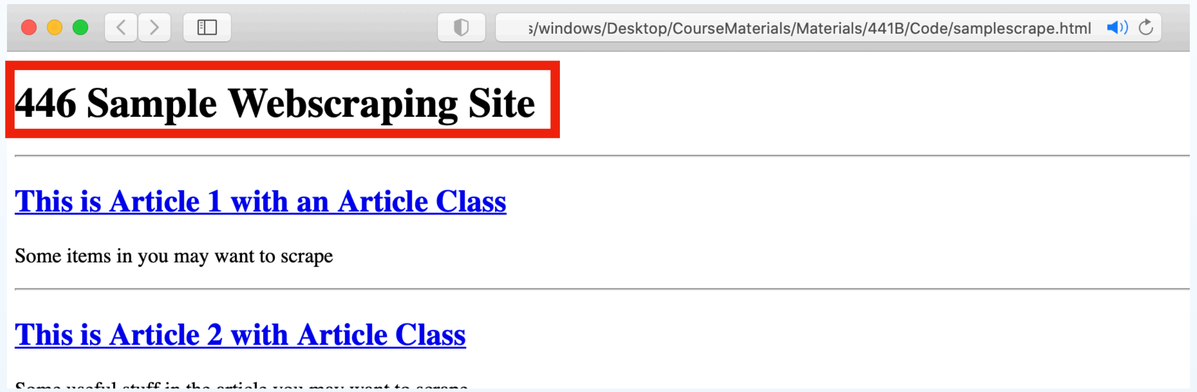
```
<!doctype html>
<html class="no-js" lang="">
    <head>
    </head>
</html>
```

Nested Tag

First Tag you always write is the <html> tag

Locating Relevant information inside HTML

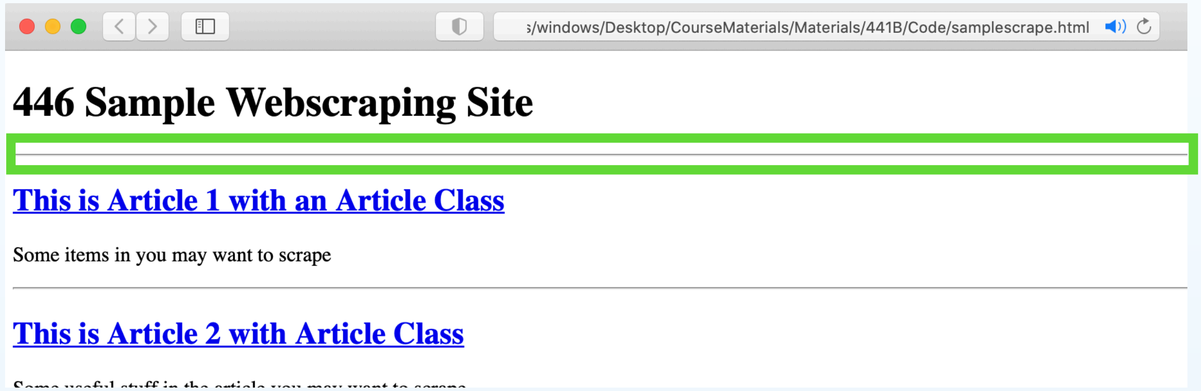
```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <title>446 Heading of the Webpage</title>
        <meta charset="utf-8">
        <link rel="stylesheet" href="css/normalize.css">
        <link rel="stylesheet" href="css/main.css">
    </head>
    <body>
        <h1 id='site_title'>446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



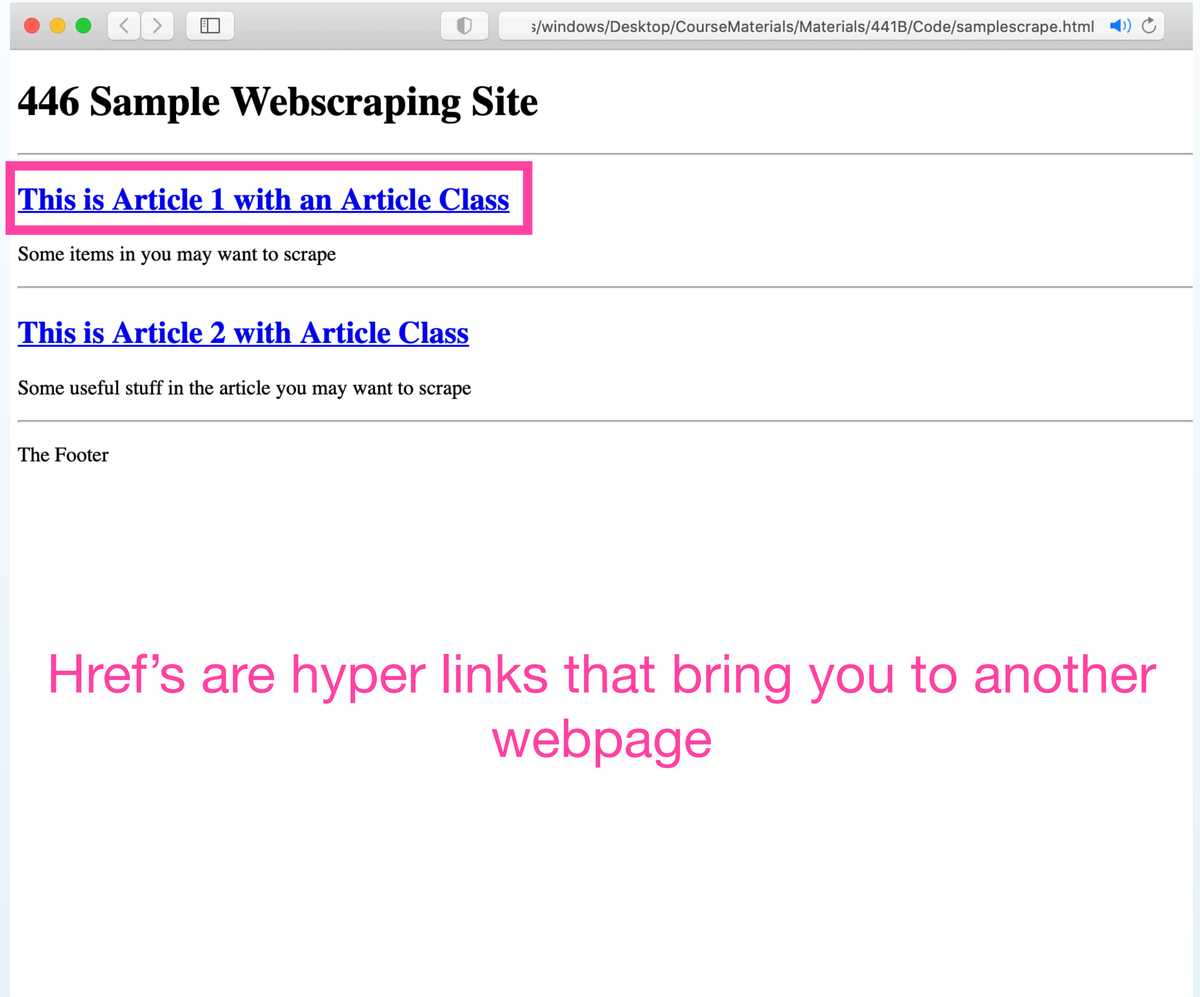
A horizontal line across the screen

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>
```



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

Relative Hyperlinks

Relative vs. Absolute Hyperlinks

Relative Hyperlinks


```
<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

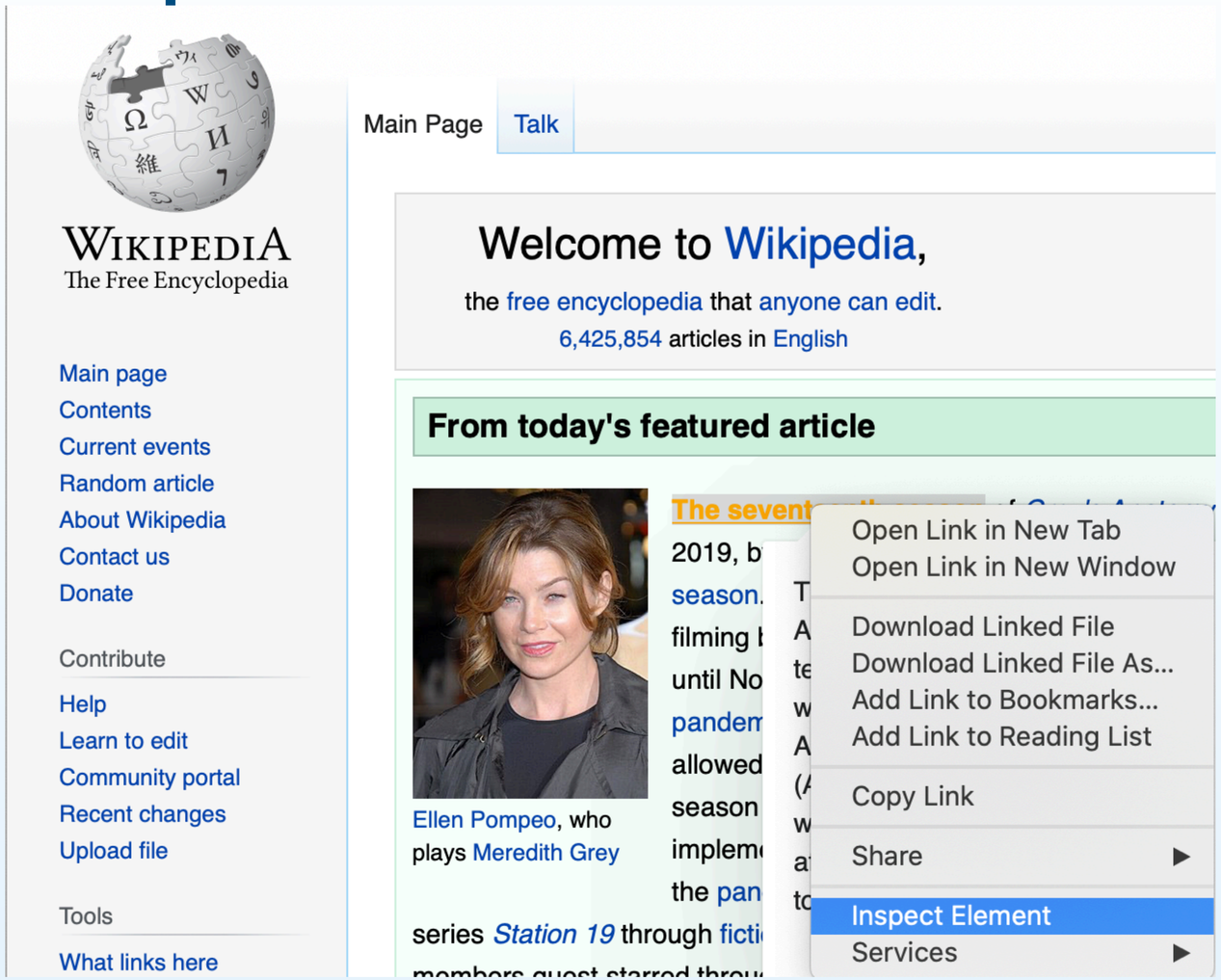
https://www.w3schools.com/TAGS/default.asp

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

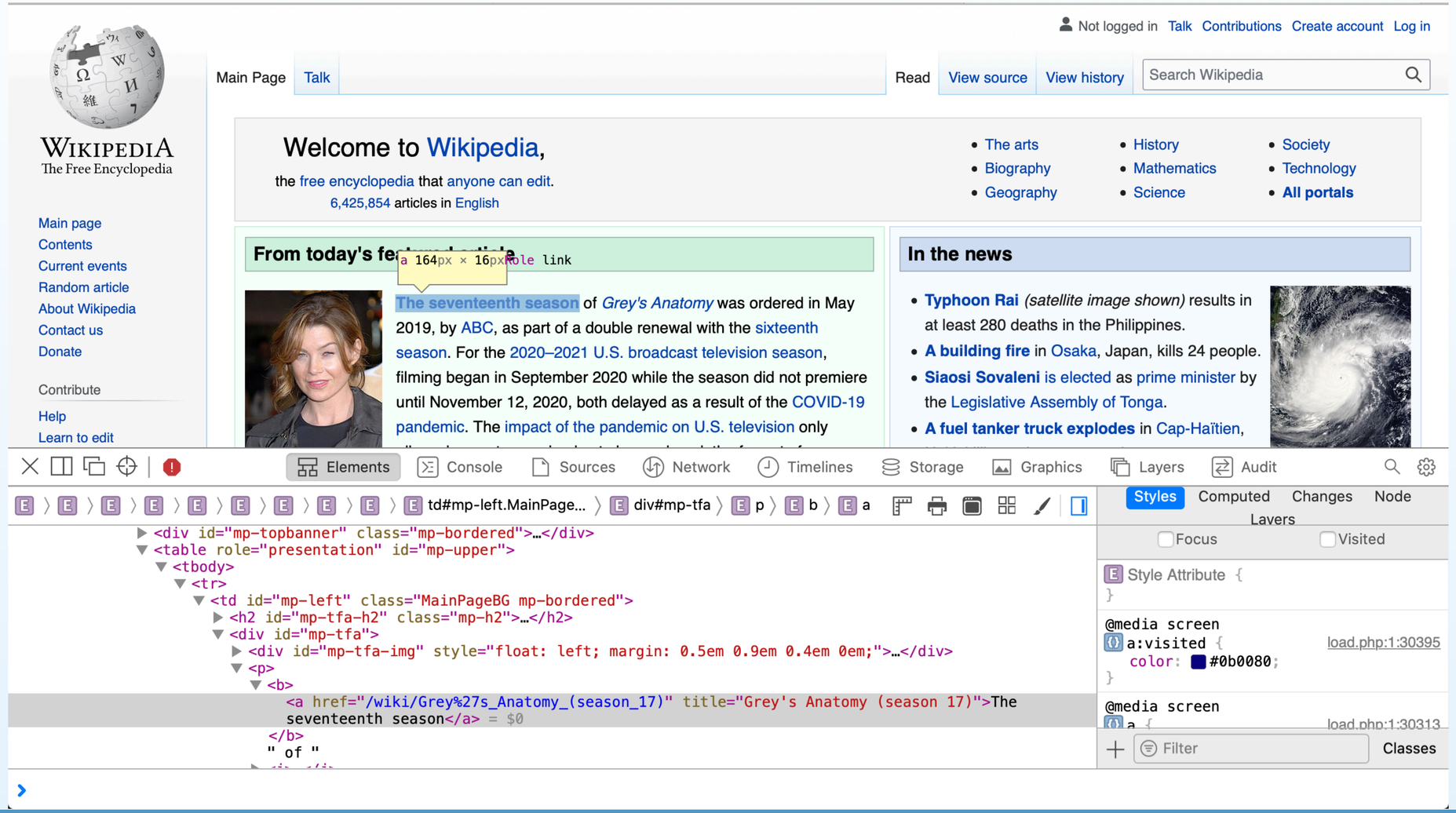
HTML in the wild

Inspect Element



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

Inspect Element



The highlighted section will correspond to the highlighted HTML code.

Get Inspect Element

For Safari

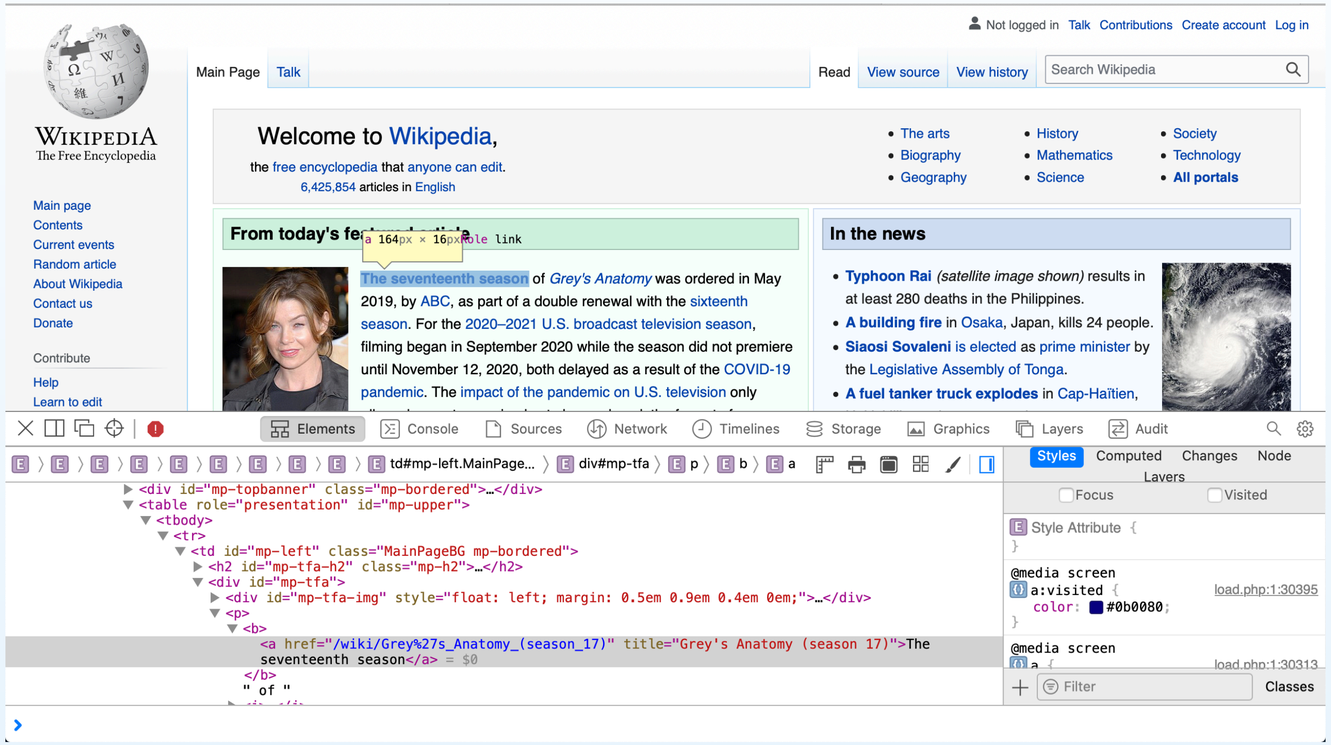
1. **The primary step is to enable the Developer menu. To do so, open the Safari browser, click on Safari -> Preferences.**
2. **Click on Advanced. Check the Show Develop menu in menu bar checkbox. ...**
3. **The Inspect Element feature is now enabled.**

1. **Chrome is built in**

Web scraping process

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

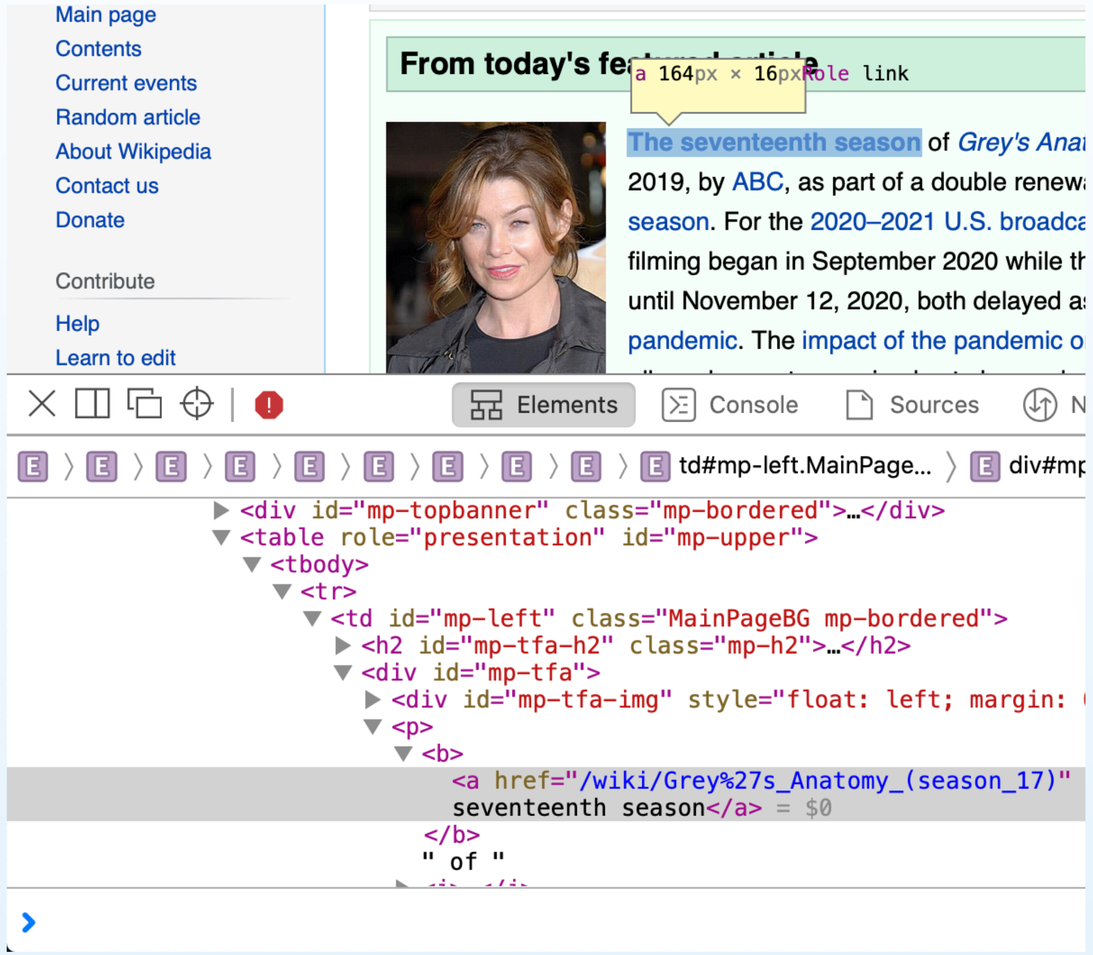
2.) What tags and attributes are associated with this information?



Web scraping process Example 1.)

1.) Locate Links

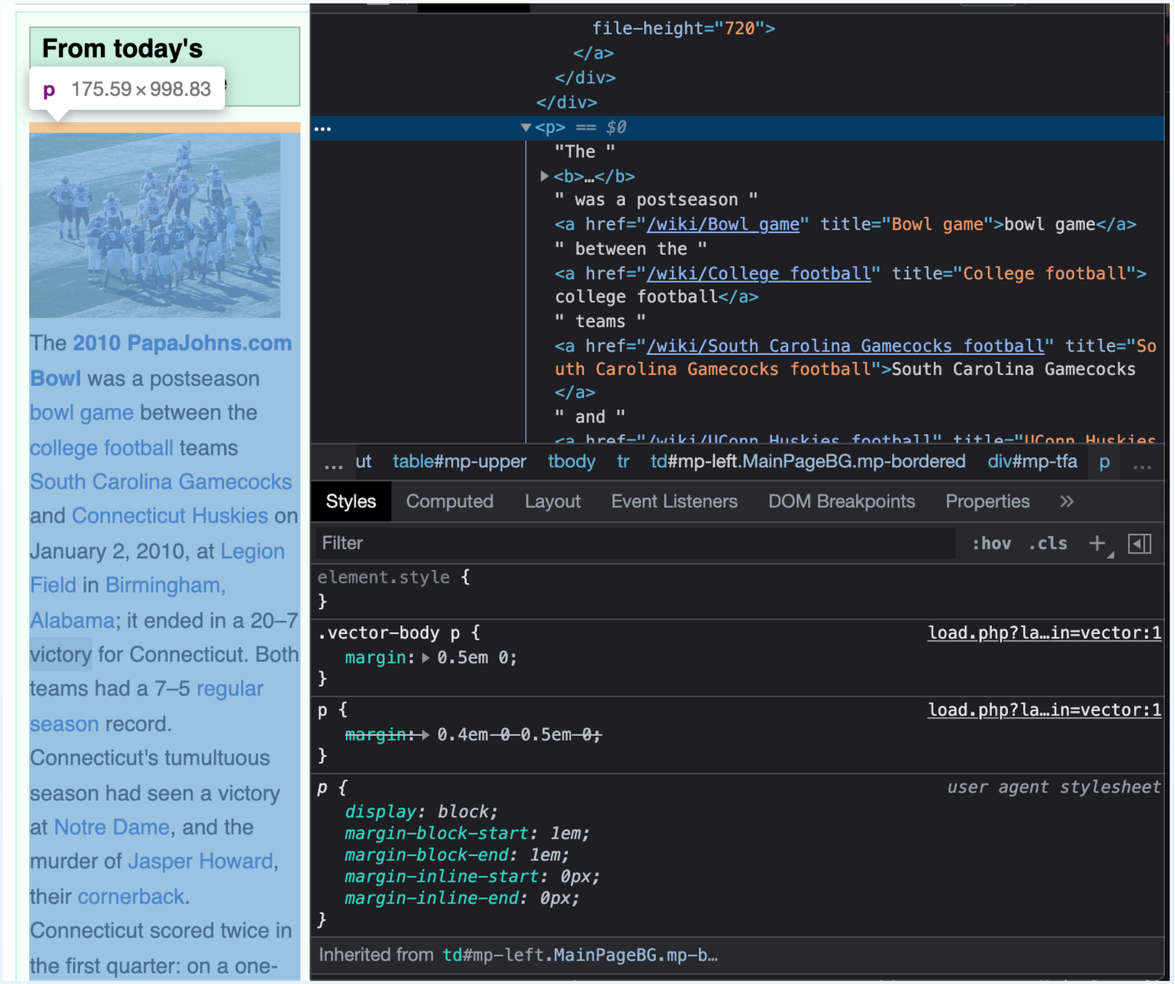
2.) <a> anchor tag. Info is inside of the href attribute.



Web scraping process Example 2.)

1.) Locate the writing blurbs

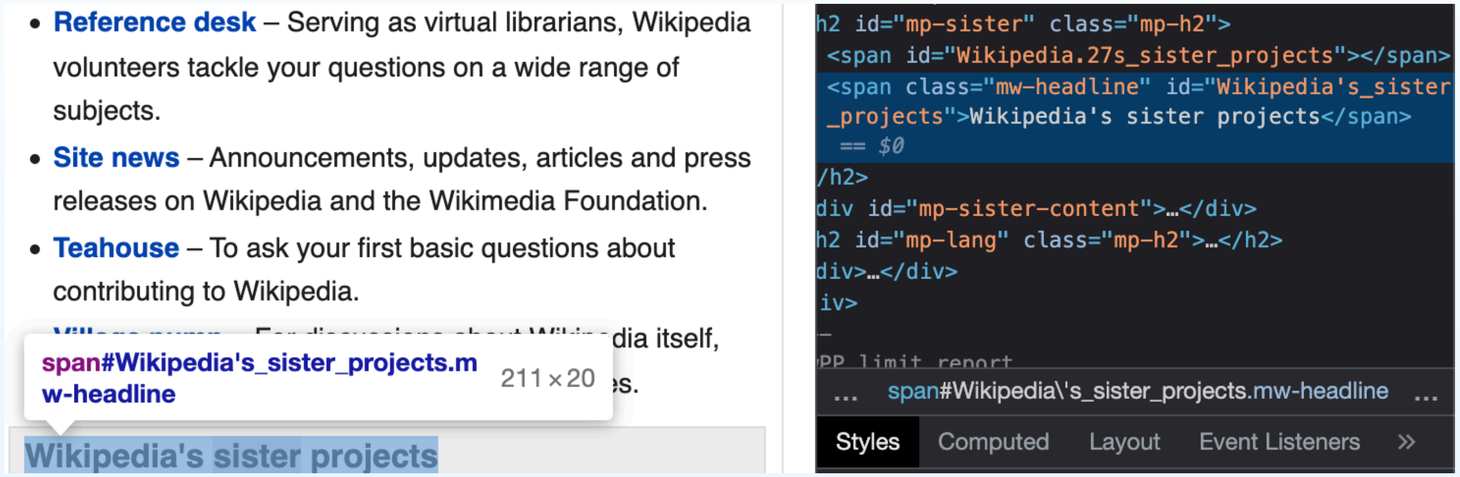
2.) <p> paragraph tag. All of the text inside of the p tag



Web scraping process Example 3.)

1.) Locate the headlines

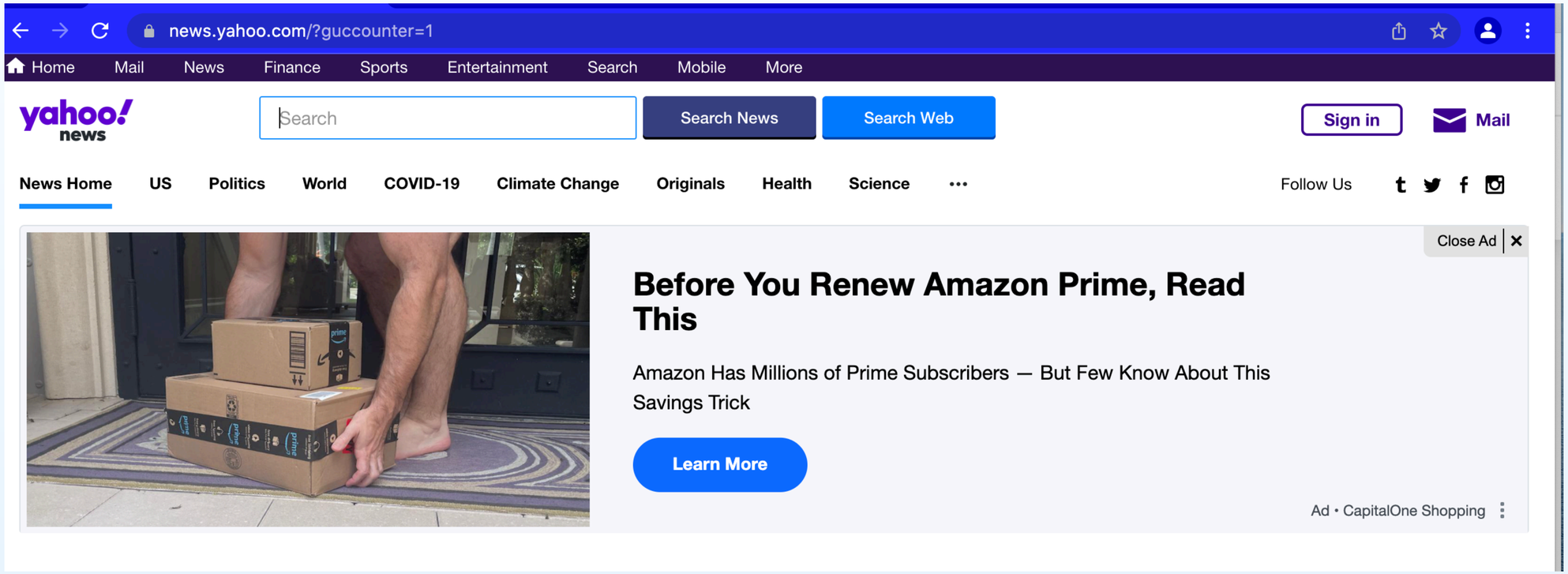
2.) span tag.
Attribute of class = “mw-headline”



BeautifulSoup and Requests

Example used in this Section

Web scraping Yahoo News



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

Python Method: Function associated with an object

```
dir(page)
```

```
'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

Package for parsing HTML code

Same “page” from requests

```
from bs4 import BeautifulSoup
```

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

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

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

Beautiful Soup

Prettify : Makes HTML code readable

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

```
b'<!DOCTYPE html><html id="atomic" class="NoJs desktop" lang=
ttp://ogp.me/ns#"><script>window.performance && window.perfor
mance.mark(\'PageStart\');</script><meta charset="utf-8"/><me
Color" content="#6e329d"/><meta name="msapplication-TileImage
om/rz/p/yahoo_frontpage_en-US_s_f_w_bestfit_frontpage.png"/><
ontent="A9862C0E6E1BE95BCE0BF3D0298FD58B"/><meta name="referr
meta name="theme-color" content="#400090"/><meta name="twitte
name="twitter:site" content="@YahooNews"/><meta name="applica
ews - Latest News &amp; Headlines"/><meta name="twitter:card"
ame="apple-itunes-app" content="app-id=304158842"/><meta name
ontent="guce.yahoo.com"/><meta name="twitter:description" con
eadlines from Yahoo! News. Get breaking news stories and in-d
nd photos."/><meta name="twitter:image" content="https://s.yi
es/yahoo_default_logo.png"/><meta name="twitter:image:src" co
v/apiv2/social/images/yahoo_default_logo.png"/><meta name="tw
News - Latest News &amp; Headlines"/><meta name="description"
nd headlines from Yahoo! News. Get breaking news stories and
```

```
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 &amp; Headlines" na
  <meta content="summary" name="twitter:card"/>
```

Beautiful Soup

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']
```

Most likely method you will encounter is text

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

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

Beautiful Soup

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

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

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>
```

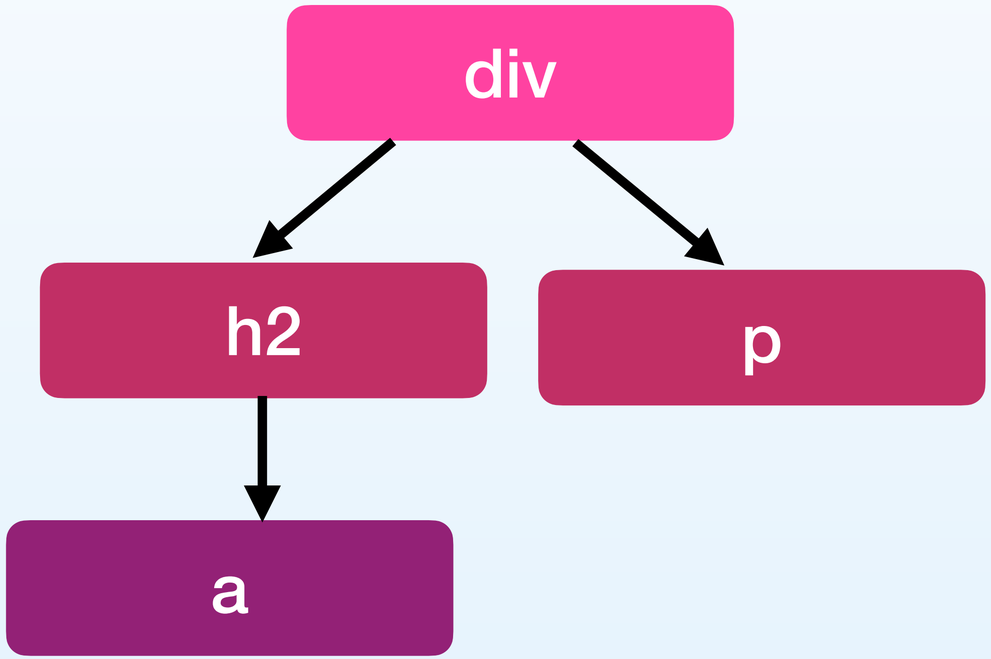
Parent Child

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
```

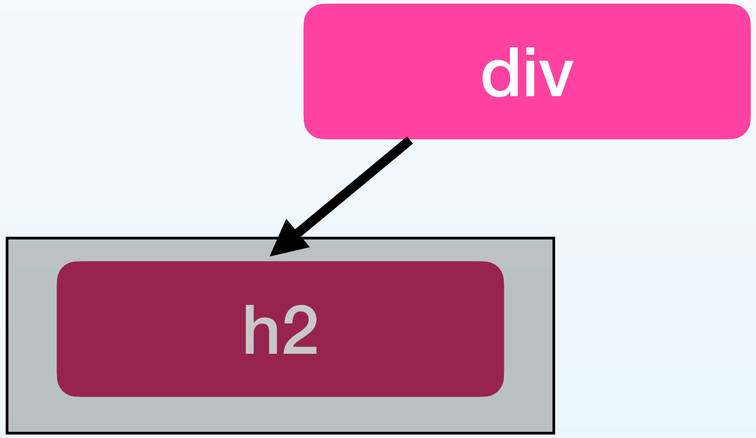
```
<div class="article">
<h2><a href="article_1.html">This is Article 1 with an Article Class</a></h2>
<p>Some items in you may want to scrape</p>
</div>
```



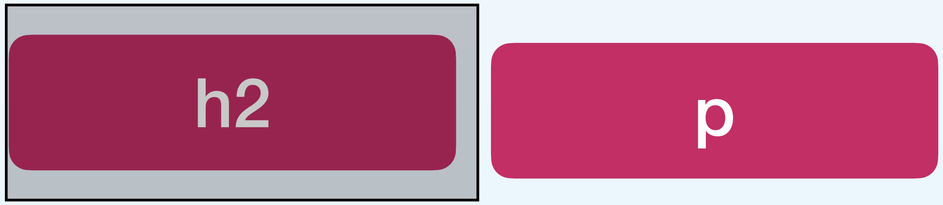
Relationships

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

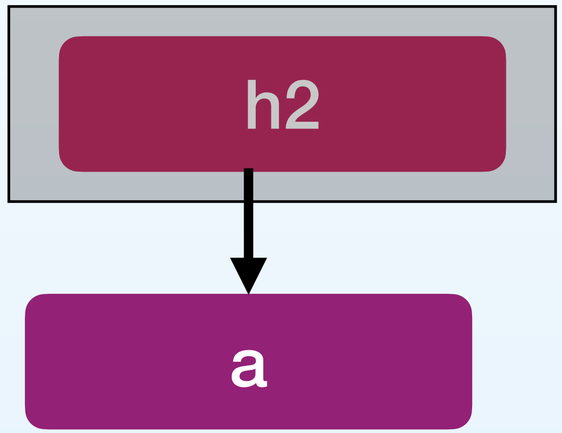
Div is a parent to h2



p is a sibling ti h2



a is a child of h2



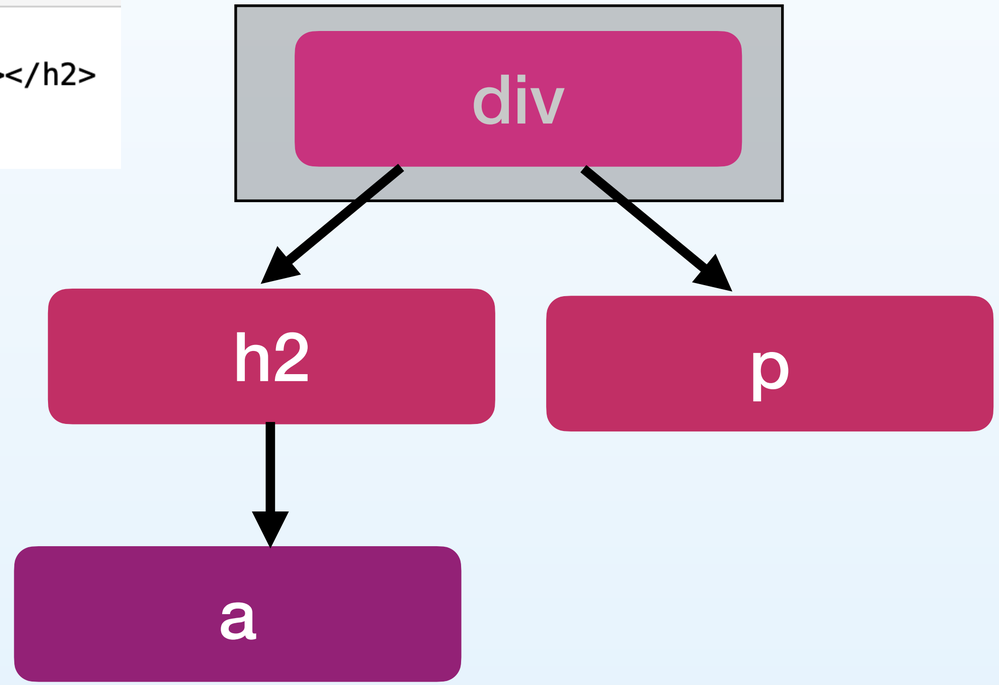
Tag Object

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

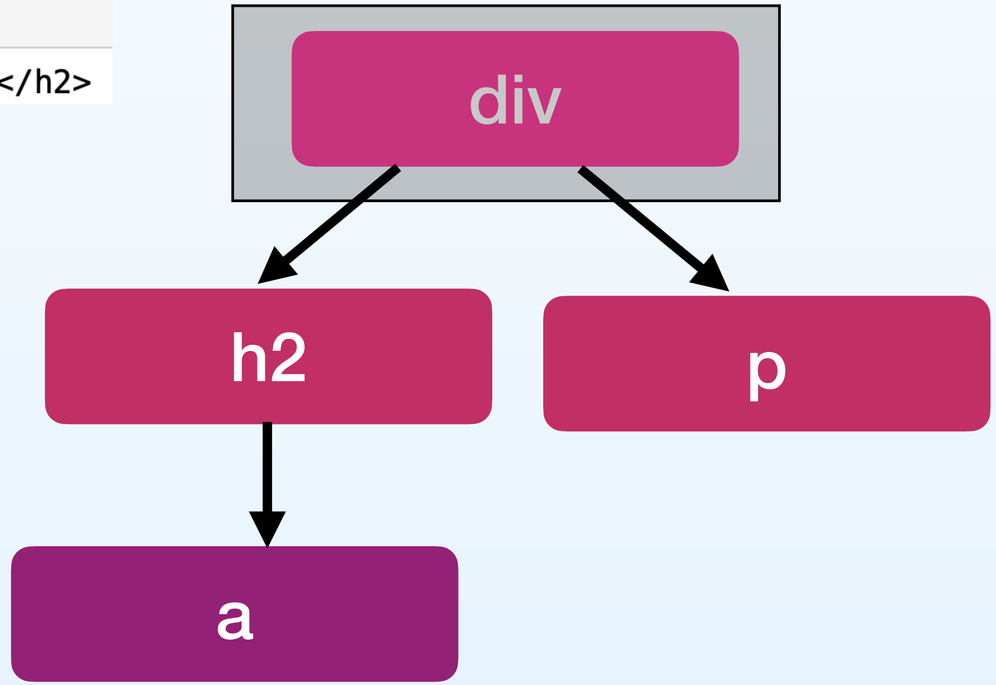
```
tag.findChild().next_sibling
```

```
<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

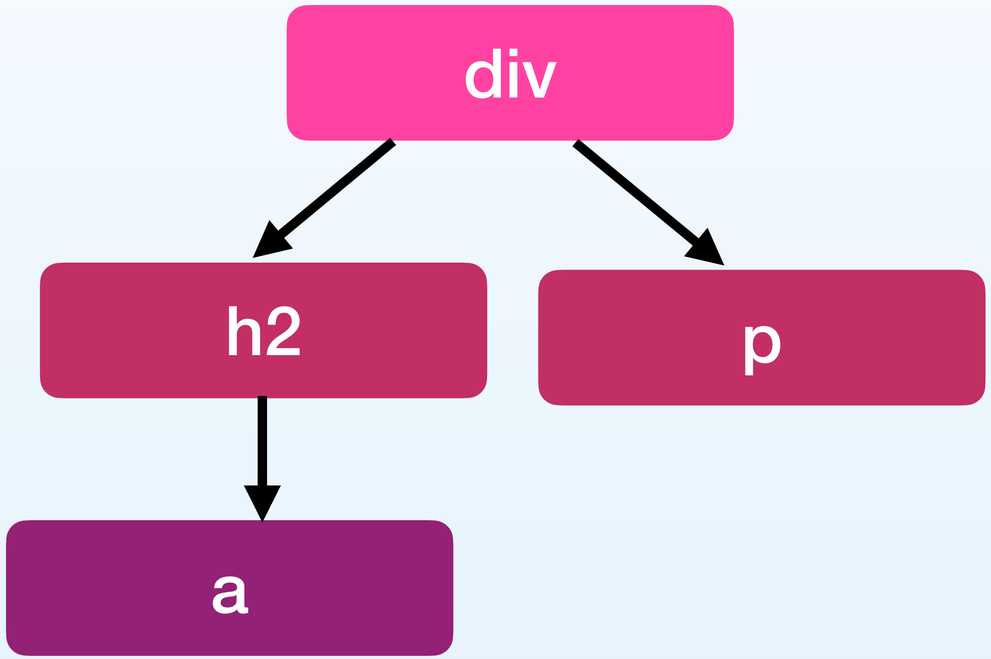
```
tag.descendants
```

```
<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

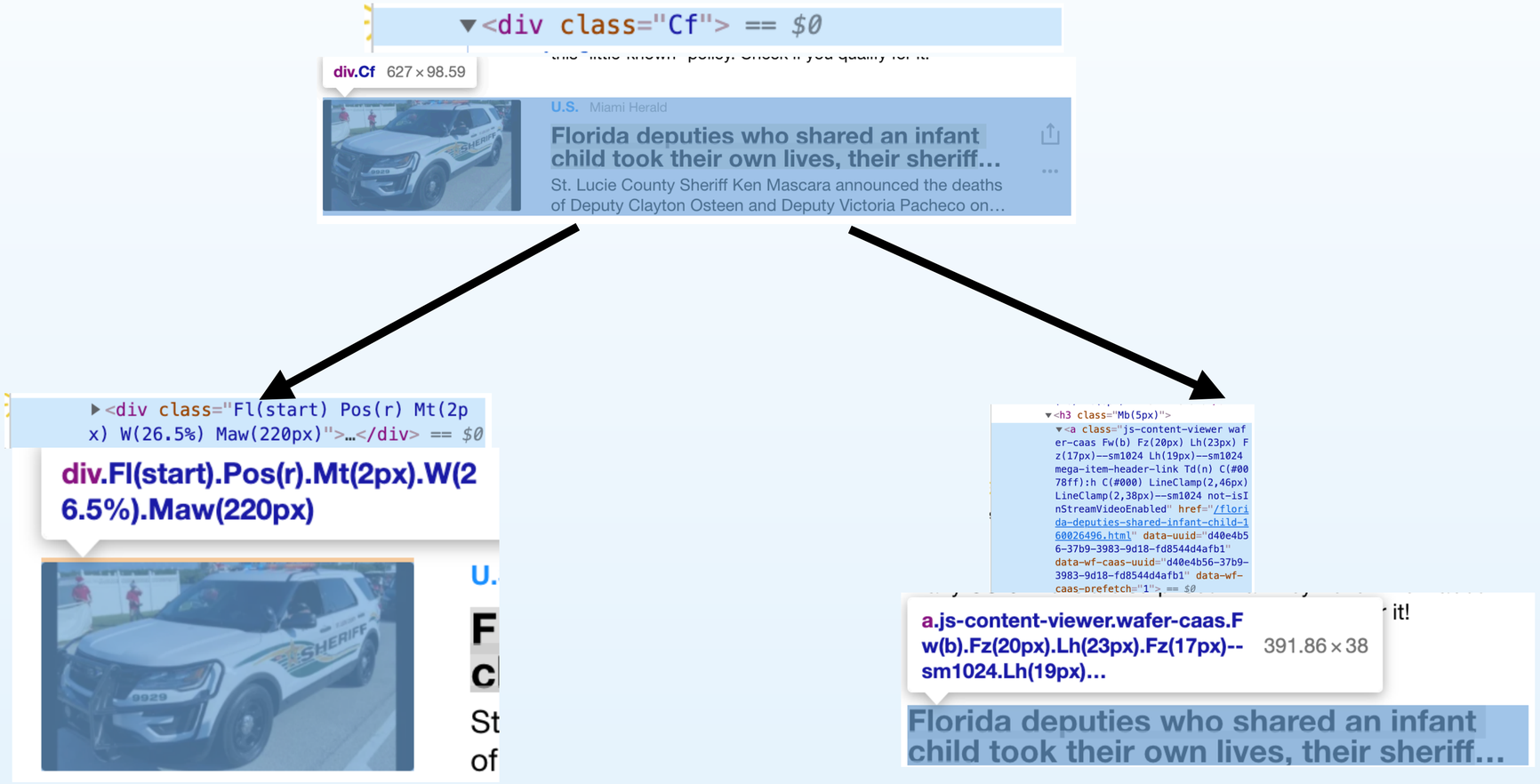
Florida deputies who shared an infant child took their own lives, their sheriff said

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...	img134.jpg
1	Trump assembles a sort of shadow political party	img123.jpg
2	Man served 37 years on testimony from false wi...	img85.jpg
3	Podcasts spread election misinformation ahead ...	img475.jpg
4	Biden's climate declaration 'a slap in the face'	img475.jpg

Common case necessary for this type of navigation

Inside of a single object we want multiple pieces of info



1.) 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) Ov(h) Bdrs(2px)" data-reactid="57" style="padding-bottom:56%;">
   <img alt="" class="W(100%) Trsdu(0s)! Bdrs(2px)" data-reactid="58" src="https://s.yim
g.com/uu/api/res/1.2/KK2_HJ4lnkagq76sDQa8TQ--~B/Zmk9c3RyaW07aD0xMjM7cT04MDt3PTIyMDthcHBp
ZD15dGFjaHlvbg--/https://s.yimg.com/uu/api/res/1.2/UBi0c9kA9Q9NJ1h0njjymA--~B/aD0zODM1O3
c9NTc1MjthcHBpZD15dGFjaHlvbg--/https://media.zenfs.com/en/ap.org/b0a120c4ab657177620dd79
7e4a61777.cf.jpg" srcset="https://s.yimg.com/uu/api/res/1.2/KK2_HJ4lnkagq76sDQa8TQ--~B/Z
mk9c3RyaW07aD0xMjM7cT04MDt3PTIyMDthcHBpZD15dGFjaHlvbg--/https://s.yimg.com/uu/api/res/1.
2/UBi0c9kA9Q9NJ1h0njjymA--~B/aD0zODM1O3c9NTc1MjthcHBpZD15dGFjaHlvbg--/https://media.zenf
s.com/en/ap.org/b0a120c4ab657177620dd797e4a61777.cf.jpg 1x,https://s.yimg.com/uu/api/res
/1.2/os3WnrM64IXalFhqn.3xBw--~B/Zmk9c3RyaW07aD0yNDY7cT04MDt3PTQ0MDthcHBpZD15dGFjaHlvbg--
/https://s.yimg.com/uu/api/res/1.2/UBi0c9kA9Q9NJ1h0njjymA--~B/aD0zODM1O3c9NTc1MjthcHBpZD
15dGFjaHlvbg--/https://media.zenfs.com/en/ap.org/b0a120c4ab657177620dd797e4a61777.cf.jpg
2x"/>
  </div>
 </div>
```

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

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

```
'https://s.yimg.com/uu/api/res/1.2/KK2_HJ4lnkagq76sDQa8TQ--~B/Zmk9c3RyaW07aD0xMjM7cT04MD
t3PTIyMDthcHBpZD15dGFjaHlvbg--/https://s.yimg.com/uu/api/res/1.2/UBi0c9kA9Q9NJ1h0njjymA-
-~B/aD0zODM1O3c9NTc1MjthcHBpZD15dGFjaHlvbg--/https://media.zenfs.com/en/ap.org/b0a120c4a
b657177620dd797e4a61777.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 (M
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