CSS Locators

WEB SCRAPING IN PYTHON

Thomas Laetsch
Data Scientist, NYU

Rosetta CSStone

- `/` replace by `>` (except first character)
  - **XPath:** `/html/body/div`
  - **CSS Locator:** `html > body > div`
- `//` replaced by a blank space (except first character)
  - **XPath:** `//div/span//p`
  - **CSS Locator:** `div > span p`
- `[N]` replaced by `:nth-of-type(N)`
  - **XPath:** `//div/p[2]`
  - **CSS Locator:** `div > p:nth-of-type(2)`

Rosetta CSStone

XPATH

```
xpath = '/html/body//div/p[2]'
```

CSS

```
css = 'html > body div > p:nth-of-type(2)'
```

Attributes in CSS

- To find an element by class, use a period `.`
  - Example: `p.class-1` selects all paragraph elements belonging to `class-1`
- To find an element by id, use a pound sign `#`
  - Example: `div#uid` selects the `div` element with `id` equal to `uid`

Attributes in CSS

Select paragraph elements within class `class1` :

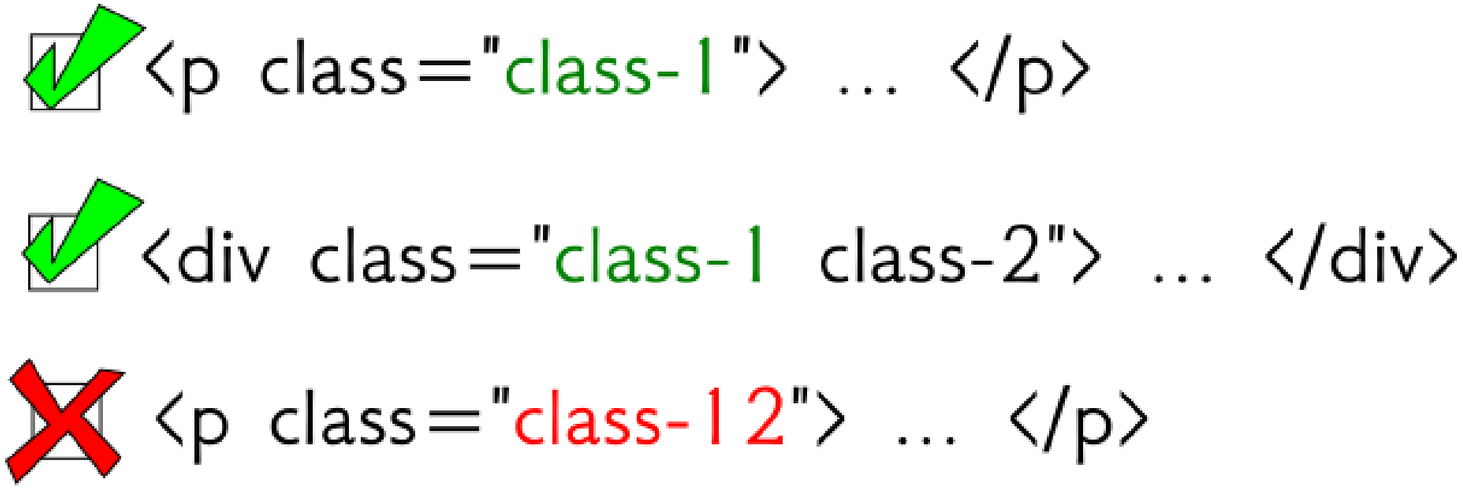
```
css_locator = 'div#uid > p.class1'
```

Select all elements whose class attribute belongs to `class1` :

```
css_locator = '.class1'
```

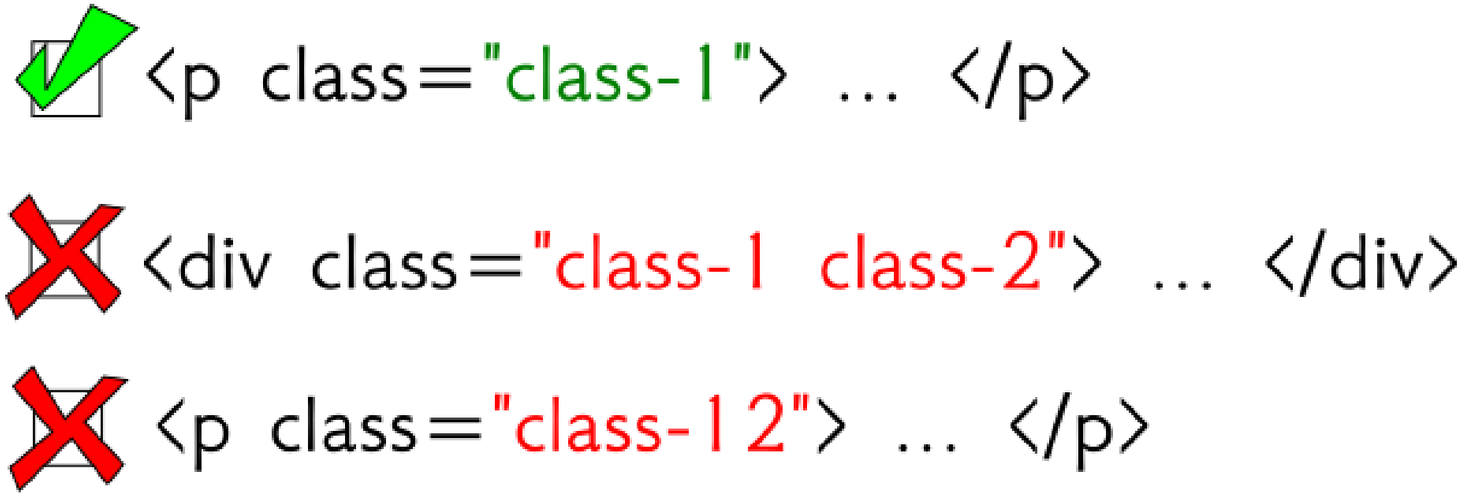
Class Status

```
css = '.class1'
```



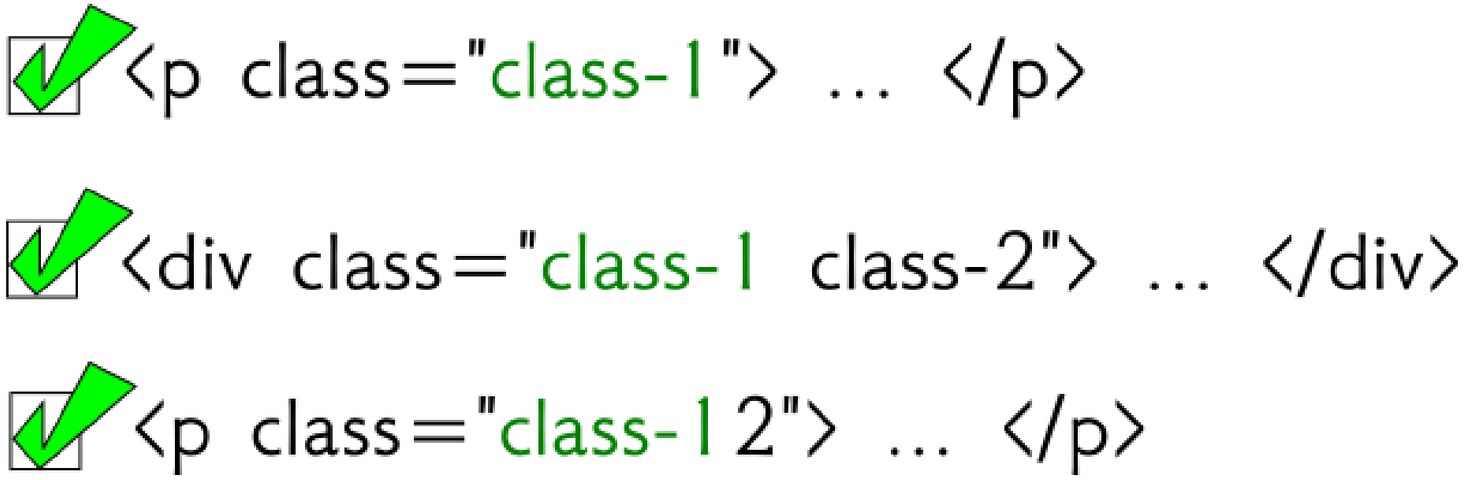
Class Status

```
xpath = '//*[@class="class1"]'
```



Class Status

```
xpath = '//*[contains(@class,"class1")]'
```



Selectors with CSS

```
from scrapy import Selector

html = '''
<html>
  <body>
    <div class="hello datacamp">
      <p>Hello World!</p>
    </div>
    <p>Enjoy DataCamp!</p>
  </body>
</html>
'''
sel = Selector( text = html )
```

```
>>> sel.css("div > p")
out: [<Selector xpath='...' data='<p>Hello World!</p>'>]

>>> sel.css("div > p").extract()
out: [ '<p>Hello World!</p>' ]
```

C(SS) You Soon!

WEB SCRAPING IN PYTHON

Attribute and Text Selection

WEB SCRAPING IN PYTHON

Thomas Laetsch
Data Scientist, NYU

datacamp

You Must have Guts to use your Colon

- Using XPath: `<xpath-to-element>/@attr-name`

```
xpath = '//div[@id="uid"]/a/@href'
```

- Using CSS Locator: `<css-to-element>::attr(attr-name)`

```
css_locator = 'div#uid > a::attr(href)'
```

Text Extraction

```
<p id="p-example">
  Hello world!
  Try <a href="http://www.datacamp.com">DataCamp</a> today!
</p>
```

- In XPath use `text()`

```
sel.xpath('//p[@id="p-example"]/text()').extract()
# result: ['\n Hello world!\n Try ', ' today!\n']
```

```
sel.xpath('//p[@id="p-example"]//text()').extract()
# result: ['\n Hello world!\n Try ', 'DataCamp', ' today!\n']
```

Text Extraction

```
<p id="p-example">
  Hello world!
  Try <a href="http://www.datacamp.com">DataCamp</a> today!
</p>
```

- For CSS Locator, use `::text`

```
sel.css('p#p-example::text').extract()
# result: ['\n Hello world!\n Try ', ' today!\n']
```

```
sel.css('p#p-example ::text').extract()
# result: ['\n Hello world!\n Try ', 'DataCamp', ' today!\n']
```

Scoping the Colon

WEB SCRAPING IN PYTHON

Getting Ready to Crawl

WEB SCRAPING IN PYTHON

Thomas Laetsch
Data Scientist, NYU

datacamp

Let's Respond

Selector vs Response:

- The Response **has all the tools we learned with Selectors:**
  - `xpath` and `css` methods followed by `extract` and `extract_first` methods.
- The Response also **keeps track of the url** where the HTML code was loaded from.
- The Response **helps us move from one site to another**, so that we can "crawl" the web while scraping.

What We Know!

- `xpath` method works like a Selector

```
response.xpath( '//div/span[@class="bio"]' )
```

- `css` method works like a Selector

```
response.css( 'div > span.bio' )
```

- Chaining works like a Selector

```
response.xpath('//div').css('span.bio')
```

- Data extraction works like a Selector

```
response.xpath('//div').css('span.bio').extract()
response.xpath('//div').css('span.bio').extract_first()
```

What We Don't Know

- The `response` keeps track of the URL within the response url variable.

```
response.url
>>> 'http://www.DataCamp.com/courses/all'
```

- The `response` lets us "follow" a new link with the `follow()` method

```
# next_url is the string path of the next url we want to scrape
response.follow( next_url )
```

- We'll learn more about `follow` later.

In Response

WEB SCRAPING IN PYTHON

Scraping For Reals

WEB SCRAPING IN PYTHON

Thomas Laetsch
Data Scientist, NYU

datacamp

DataCamp Site

https://www.datacamp.com/courses/all

What's the Div, Yo?

```
# response loaded with HTML from https://www.datacamp.com/courses/all
```

```
course_divs = response.css('div.course-block')
```

```
print( len(course_divs) )
>>> 185
```

Inspecting course-block

```
first_div = course_divs[0]
children = first_div.xpath('./*')
print( len(children) )
>>> 3
```

The first child

```
first_div = course_divs[0]
children = first_div.xpath('./*')
```

```
first_child = children[0]
print( first_child.extract() )
>>> <a class=... />
```

The second child

```
first_div = course_divs[0]
children = first_div.xpath('./*')
```

```
second_child = children[1]
print( second_child.extract() )
>>> <div class=... />
```

The forgotten child

```
first_div = course_divs[0]
children = first_div.xpath('./*')
```

```
third_child = children[2]
print( third_child.extract() )
>>> <span class=... />
```

Listful

- In one CSS Locator

```
links = response.css('div.course-block > a::attr(href)').extract()
```

- Stepwise

```
# step 1: course blocks
course_divs = response.css('div.course-block')
# step 2: hyperlink elements
hrefs = course_divs.xpath('./a/@href')
# step 3: extract the links
links = hrefs.extract()
```

Get Schooled

```
for l in links:
    print( l )

>>> /courses/free-introduction-to-r
>>> /courses/data-table-data-manipulation-r-tutorial
>>> /courses/dplyr-data-manipulation-r-tutorial
>>> /courses/ggvis-data-visualization-r-tutorial
>>> /courses/reporting-with-r-markdown
>>> /courses/intermediate-r
...
```

Links Achieved

WEB SCRAPING IN PYTHON