# Important Note!

Web scraping should be done in accordance with the website's terms of use. [1]

[1] https://www.crummy.com/software/BeautifulSoup/zine/

# Web Scraping

- Web scraping (web harvesting or web data extraction) is the term given for acquiring data from websites
- Scraping technologies must be tolerant to several artefacts of real-world data (so-called 'wrangling' will be covered in later lectures)
- Erroneous data makes parsing information difficult
- In the context of scraping data from the web, we will discuss:
  - BeautifulSoup [2]
  - scrapy [3]
  - Selenium [4]
- First we need to know a little about webpages, however.

[2] https://www.crummy.com/software/BeautifulSoup/
[3] https://scrapy.org
[4] https://www.selenium.dev/

# HTML Code Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>Scraping</title>
  </head>
  <body class="col-sm-12">
    <h1>section1</h1>
      <p>paragraph1</p>
      <p>paragraph2</p>
    <div class="col-sm-2">
      <h2>section2</h2>
      <p>paragraph3</p>
      <p>unclosed
    </div>
  </body>
</html>
```

- <!DOCTYPE html>: HTML documents must start with a type declaration
- HTML document is contained between <html>and </html>
- Meta and script declaration of the HTML document is between <head>and </head>
- Visible part of the HTML document is between <body>and </body>
- Title headings are defined with the <h1>to <h6>tags
- Section/division tags are often used to segment the source
- Paragraphs are defined with the <p>tag

# Interactive DOM Explorer

`https://javascript.info/dom-nodes`

- HTML cheat sheet:
  `https://html.com/wp-content/uploads/html-cheat-sheet.pdf`

# Summary of Beautiful Soup

- Beautiful Soup is a powerful library for parsing XML/HTML documents
- With this library, one can create information extraction engines that can run autonomously
- However, HTML scraping has many serious problems:
  - Selecting the 'correct' link to follow is a strong function of the ability to identify the link of interest based only on its HTML tags. This can be difficult in poorly designed websites.
  - If the layout or formatting of information on a webpage changes, it will become necessary to reconfigure the web scraping mechanism from scratch
- Tests for format changes should be defined

# Web APIs

Web APIs provide a portal for explicit data acquisition, and are generally less prone to the issues from HTML scraping since:

- Code is optimised for retrieval and not for visual layout/aesthetics
- Standard serialisation tools (e.g., JSON) are typically used
- The core items of interest have been extracted (e.g., dates, URLs)

# Examples of Web APIs

- `https://github.com/toddmotto/public-apis`
- Web APIs, although similar in principle, will have very different schemas
- However, writing code for APIs is generally simpler than parsing HTML, requires less maintenance, are often documented, and results in faster overall code

Example Task: acquire a list of recent posts from the 'technology' section of a newspaper

- `http://open-platform.theguardian.com/`
- `http://open-platform.theguardian.com/documentation/`

# RESTful APIs

- Representational state transfer (REST) or RESTful Web services are one way of providing interoperability between computer systems on the Internet [5]
- In most circumstances, API keys are required before data can be accessed
- RESTful APIs define a collection of resources
  `https://content.guardianapis.com/sections?api-key=test`
- For each resource, RESTful APIs additionally provide a list of items
  `https://content.guardianapis.com/technology?api-key=test`

[5] `https://en.wikipedia.org/wiki/Representational_state_transfer`

```
# Specify the arguments
args = {
  "section": "technology",
  "order-by": "newest",
  "api-key": "test",
  "page-size": "100"
}
# Construct the URL
base_url = create_guardian_url(args)

# Make the request and extract the source
response = json.loads(requests.get(url).text)
...
...

# Print the data that is available
print(response.keys())
# ["currentPage", "orderBy", "pageSize", "pages", "results",
#  "startIndex", "status", "total", "userTier"]
```

# Regular Expression

- Regular expressions are sequences of characters that define a search pattern
- In Python there are two main options for executing regular expressions:
  - `re.match`
  - `re.search`
- These two functions are similar, but with distinct differences

```
# ..attempts to match the RE pattern
# to the *start* of the string
re.match(pattern, string, flags=0)

# ..attempts to match the RE pattern
# to *any* point in the string
re.search(pattern, string, flags=0)
```

- RegEx tutorial: `https://developers.google.com/edu/python/regular-expressions`
- Cheat sheet: `https://www.dataquest.io/wp-content/uploads/2019/03/python-regular-expressions-cheat-sheet.pdf`

```python
import re

line = "Cats␣are␣smarter␣than␣dogs"
matchObj = re.match(r"(.*)␣are␣(.*?)␣than␣(.*)", line, re.I)

print("matchObj.group():", matchObj.group())
# "Cats are smarter than dogs"

print("matchObj.group(1):", matchObj.group(1))
# "Cats"

print("matchObj.group(2):", matchObj.group(2))
# "smarter"

print("matchObj.group(3):", matchObj.group(3))
# "dogs"
```

# Summary on APIs

- API-based querying is robust, reliable, well maintained and documented with a static schema
  - HTML-based web scraping is not
- Information extraction is not based on fickle naming conventions of tag attributes
- Since only content is acquired (and no *images*, *JavaScript* or *style* files) the bandwidth spent to acquire data is reduced significantly
  - This naturally lends itself to faster processing
- Since access is acquired through API keys, it is easy for the service provider to manage the bandwidth and throughput of its service as necessary
- Data structures being received from APIs are complex, and need specific serialisation tools to be built

# Resources for today's lecture

- Python tutorials: `https://docs.python.org/3/tutorial/`
- 'Python for Data Analysis' `http://shop.oreilly.com/product/0636920023784.do`
- JSON Verification: `http://jsonlint.com/`
- Serialisation techniques: `https://en.wikipedia.org/wiki/Comparison_of_data_serialization_formats`