COMP47250

Data Collection & APIs

Mark Matthews PhD

UCD School of Computer Science

Summer 2019

Outline

- Dataset Considerations
- Data Formats
  - Plain Text, CSV, XML, HTML, JSON
- Web Scraping
- Online Web APIs
  - Twitter APIs
  - Google Maps APIs
  - YouTube API
  - Facebook API
- Other APIs and Resources

Dataset Considerations

- **Accessibility:** Is the data publicly accessible? Is an account required for authentication? Do specific terms of service apply?
- **Ethics:** Are there any privacy or ethical considerations in using, aggregating or republishing the data? Are you working with any identifiable personal data?
- **Scale:** How much data is available? Is historic data available? Is the data delivered in batches or streaming in real-time? What are the storage requirements?
- **Pre-processing:** What is the format of the data? What automated pre-processing techniques should be applied? Does the data contain missing values? Is manual cleaning required?
- **Verification:** Is the data reliable or potentially inaccurate/ misleading? Is automated or manual curation required to verify the accuracy of the data?

Data Formats: Plain Text

- Data is generated in different domains (e.g. finance, bioinformatics, engineering), and stored in many different formats.
- **Plain text:** Data is often stored as plain text (.txt) with no specific mark-up or metadata to explain its structure.
- When dealing with raw text, we may often want to tokenize the text to extract words and phrases.

```
GCHQ intelligence agency joins Twitter

The UK's intelligence agency GCHQ has set up an official
account on Twitter.

It said it had taken the step as part of an ongoing
strategy to be more accessible and open about the work it
does for the UK.

The first tweet that it sent from the @GCHQ account was
just two words: "Hello, world."

It said it would be sending messages about its history as
well as languages, maths, the outcomes of missions and
technology.
```

```
f = open("news.txt", "r")
lines = f.readlines()
f.close()
for line in lines:
  line = line.strip()
  if len(line) > 0:
    words = line.split(" ")
    print(words)
```

```
['GCHQ', 'intelligence',
'agency', 'joins', 'Twitter']
...
```

Data Formats: CSV

- The CSV ("Comma Separated Values") file format is often used to exchange tabular data between different applications (e.g. Excel).
- Essentially a plain text file where values are split by a comma separator. Alternatively can be tab or space separated.

```
Player,Team,Total Goals,Penalties,Home,Away
J Vardy,Leicester City,19,4,11,8
H Kane,Tottenham,16,4,7,9
R Lukaku,Everton,16,1,8,8
O Ighalo,Watford,15,0,8,7
S Aguero,Manchester City,14,1,10,4
R Mahrez,Leicester City,14,4,4,10
O Giroud,Arsenal,12,0,4,8
D Costa,Chelsea,10,0,7,3
J Defoe,Sunderland,10,0,3,7
G Wijnaldum,Newcastle Utd,9,0,9,0
```

Player	Team	Total Goals	Penalties	Home	Away
J Vardy	Leicester City	19	4	11	8
H Kane	Tottenham	16	4	7	9
R Lukaku	Everton	16	1	8	8
O Ighalo	Watford	15	0	8	7
S Aguero	Manchester City	14	1	10	4
R Mahrez	Leicester City	14	4	4	10
O Giroud	Arsenal	12	0	4	8
D Costa	Chelsea	10	0	7	3
J Defoe	Sunderland	10	0	3	7
G Wijnaldum	Newcastle Utd	9	0	9	0

```
import pandas
df = pd.read_csv("goal_scorers.csv")
```

```
df.to_csv("results.csv")
```

```
df = pd.read_csv("data.csv",sep=" ")
```

```
df.to_csv("results.csv",sep="\t")
```

Many Python packages can read and parse CSV data

e.g. Pandas http://pandas.pydata.org/

Data Formats: JSON

- **JSON (JavaScript Object Notation)**: a lightweight format which is becoming increasingly popular for online data exchange. Based originally on the JavaScript language and (relatively) easy for humans to read and write - http://www.json.org

- JSON files are built from two different structures:

  1. Object: Collection of name/value pairs - like a Python dictionary.
     Begins with `{` and ends with `}`
     Each name is followed by `:` (colon) and the name/value pairs are separated by commas

  2. Array: A list of values separated by commas - like a Python list.
     Begins with `[` and ends with `]`

- Values can be of the following types:
  - String in double quotes; a number; true/false; null; object; array.

Data Formats: JSON

- Object: Collection of name/value pairs, separated by commas - like a Python dictionary.

```
{ "firstname":"Alison", "age":30, "car":"BMW" }
```

```
{
   "name": "School of Computer Science",
   "link": "http://www.cs.ucd.ie"
}
```

- Array: A list of values separated by commas - like a Python list. The values can have the same or different types.

```
[ "Ford", "BMW", "Fiat", "Mercedes" ]
```

```
[ 1123, 353, 412, 99.0, "Dublin", false, true ]
```

- We can mix Objects and Arrays:

```
{ "codes" : [5,11,31,41] }
```

An array inside an object.

```
[ { "name":"Alison" }, { "name":"John"} ]
```

Two objects in an array.

Data Formats: JSON

- The outmost value in a JSON file can be an array or an object.

Top level value is an array, which itself contains two objects.

```
[
  {
    "id" : "978-1933988177",
    "category" : ["book","paperback"],
    "name" : "Lucene in Action",
    "author" : "Michael McCandless",
    "genre" : "technology",
    "price" : 30.50,
    "pages" : 475
  },
  {
    "id" : "978-1857995879",
    "category" : ["book","paperback"],
    "name" : "Sophie's World",
    "author" : "Jostein Gaarder",
    "sequence_i" : 1,
    "genre" : "fiction",
    "price" : 3.07,
    "pages" : 64
  }
]
```

Top level value is an object, which contains an array of objects.

```
{
   "students": [{
      "name": "John",
      "age": "23",
      "city": "Dublin"
   }, {
      "name": "Sarah",
      "age": "28",
      "city": "Cork"
   }, {
      "name": "Peter",
      "age": "32",
      "city": "Galway"
   }, {
      "name": "Alice",
      "age": "28",
      "city": "London"
   }]
}
```

Data Formats: JSON

- Note that whitespace does not matter in JSON, it only makes it data more readable for humans.

```
[
  {
    "id" : "978-1933988177",
    "category" : ["book","paperback"],
    "name" : "Lucene in Action",
    "author" : "Michael McCandless",
    "genre" : "technology",
    "price" : 30.50,
    "pages" : 475
  },
  {
    "id" : "978-1857995879",
    "category" : ["book","paperback"],
    "name" : "Sophie's World",
    "author" : "Jostein Gaarder",
    "sequence_i" : 1,
    "genre" : "fiction",
    "price" : 3.07,
    "pages" : 64
  }
]
```

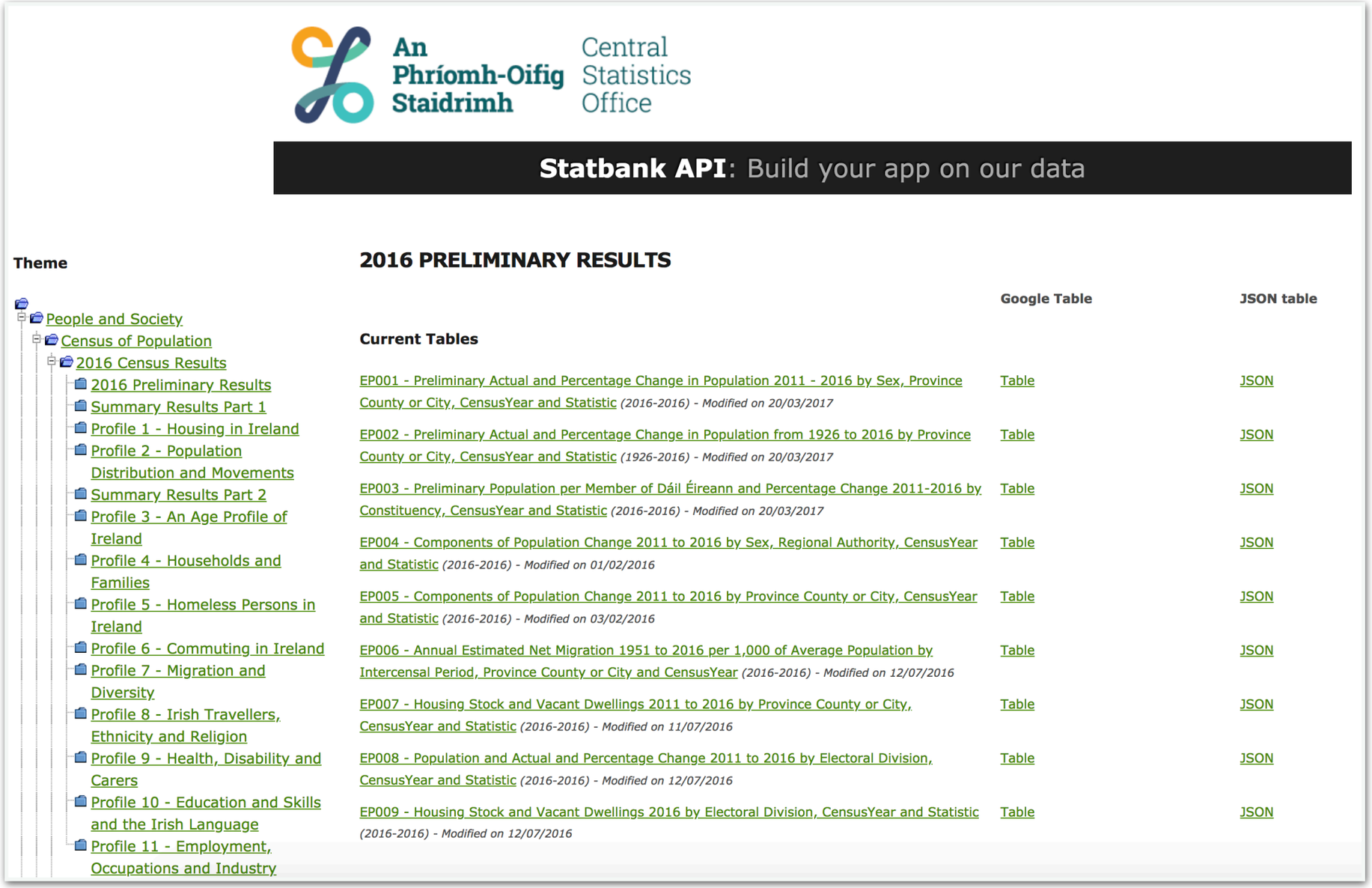
```
[ { "id" : "978-1933988177", "category" : ["book","paperback"],
"name" : "Lucene in Action", "author" : "Michael McCandless",
"genre" : "technology", "price" : 30.50, "pages" : 475 }, { "id"
: "978-1857995879", "category" : ["book","paperback"], "name" :
"Sophie's World", "author" : "Jostein Gaarder", "sequence_i" :
1, "genre" : "fiction", "price" : 3.07, "pages" : 64 } ]
```

```
{"students": [{"name": "John","age":
"23","city": "Dublin"}, {"name":
"Sarah","age": "28","city": "Cork"}, {"name":
"Peter","age": "32","city": "Galway"},
{"name": "Alice","age": "28","city":
"London"}]}
```

```
{
    "students": [{
        "name": "John",
        "age": "23",
        "city": "Dublin"
    }, {
        "name": "Sarah",
        "age": "28",
        "city": "Cork"
    }, {
        "name": "Peter",
        "age": "32",
        "city": "Galway"
    }, {
        "name": "Alice",
        "age": "28",
        "city": "London"
    }]
}
```

Example: CSO Data

- The Irish Central Statistics Office (CSO) provides information on a range of subjects in JSON format.



Example: CSO Data

- The Irish Central Statistics Office (CSO) provides information on a range of subjects in JSON format.

Example: Household composition by Age, Sex, Location and Year


```
        },
        "id": ["Age Group", "Sex", "Location", "Year", "Statistic"],
        "size": [8, 3, 3, 1, 1],
        "role": {
            "time": ["Year"],
            "metric": ["Statistic"]
        }
    },
    "label": "Household composition by  Age Group, Sex, Location and Year",
    "source": "Central Statistics Office, Ireland",
    "updated": "2018-05-22T11:09:32Z",
    "value": [2.73, 2.69, 2.82, 1.35, 1.32, 1.41, 1.38, 1.37, 1.41, 0.20, 0.22, 0.17, 0.11, 0.12, 0.09, 0.10, 0.10,
}
```


http://www.cso.ie/webserviceclient/DatasetListing.aspx

Data Formats: XML

- XML: a markup language used to describe data in a structured way using tags. Tags are not predefined, but are user defined. Many schemas exist that recommend standardised usage of tags.
- Tags must be opened `<tag>` and closed `</tag>`.
- Tags can contain values as plain text or other nested tags.

```
<note>
   <to>Alice</to>
   <from>John</from>
   <subject>Reminder</subject>
   <body>Remember to buy milk!</body>
</note>
```

Tags are opened and closed
e.g. `<note>...</note>`

We can "nest" tags inside other tags to create a hierarchical structure.

```
<notes>
   <note>
      <to>Alice</to>
      <from>John</from>
      <subject>Reminder</subject>
      <body>Remember to buy milk!</body>
   </note>
   <note>
      <to>John</to>
      <from>Alice</from>
      <subject>Shopping</subject>
      <body>I forgot the milk!</body>
   </note>
</notes>
```

Data Formats: XML

- Tags can also have zero or more name/value attribute pairs, which add extra information to a tag.

XML files often (but not always) contain a header line

Tags are opened and closed e.g. <book>...</book>

Values can be contained within tags.

Tags can have attributes e.g. currency="eur"

File: products.xml

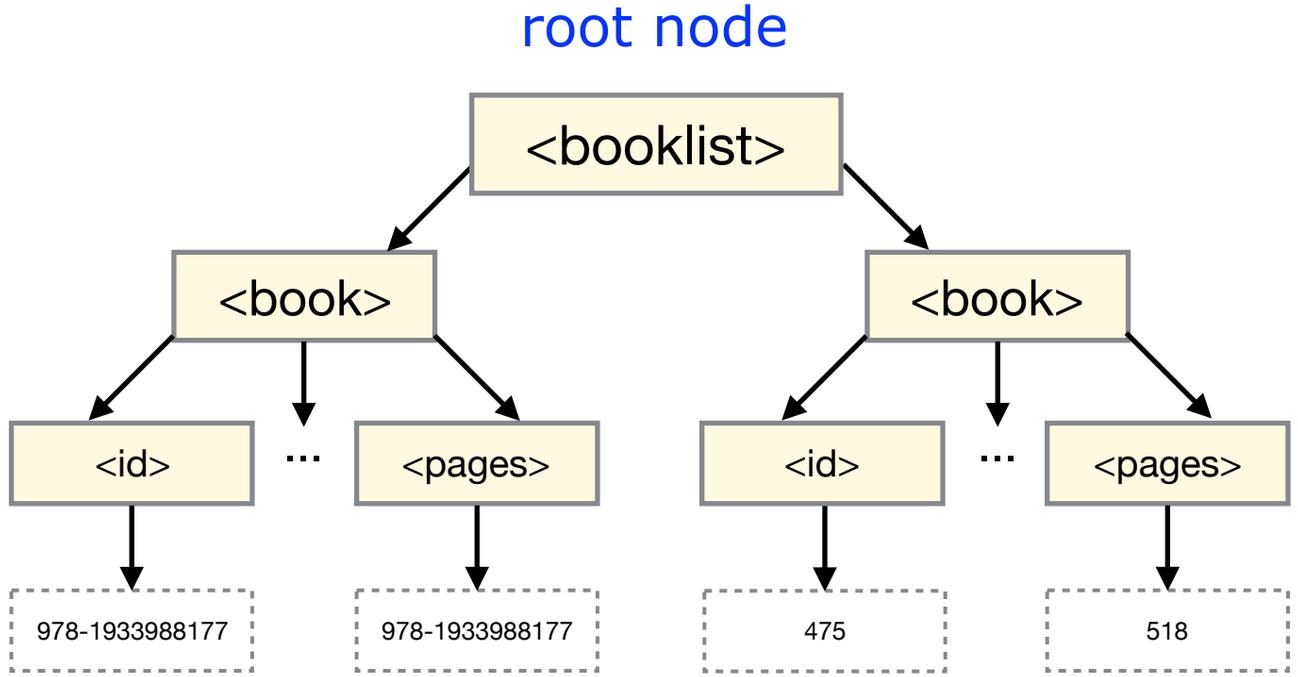
```
<?xml version="1.0" encoding="UTF-8" ?>
<booklist>
   <book>
      <id>978-1933988177</id>
      <category>paperback</category>
      <name>Lucene in Action</name>
      <author>Michael McCandless</author>
      <genre>technology</genre>
      <price currency="eur">30.5</price>
      <pages>475</pages>
   </book>
   <book>
      <id>978-1857995879</id>
      <category>paperback</category>
      <name>Sophie's World</name>
      <author>Jostein Gaarder</author>
      <genre>fiction</genre>
      <price currency="eur">3.07</price>
      <pages>518</pages>
   </book>
</booklist>
```

Data Formats: XML

- XML is an inherently hierarchical data format, and the most natural way to represent it is with a tree with nodes at different levels.
- The document itself is the root node of the tree. Other nodes are child nodes. If they contain nodes themselves, they are parent nodes. The lowest level of the tree contains leaf nodes.

File: products.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<booklist>
    <book>
        <id>978-1933988177</id>
        <category>paperback</category>
        <name>Lucene in Action</name>
        <author>Michael McCandless</author>
        <genre>technology</genre>
        <price>30.5</price>
        <pages>475</pages>
    </book>
    <book>
        <id>978-1857995879</id>
        <category>paperback</category>
        <name>Sophie's World</name>
        <author>Jostein Gaarder</author>
        <genre>fiction</genre>
        <price>3.07</price>
        <pages>518</pages>
    </book>
</booklist>
```



leaf nodes of this document contain values

Several built-in Python modules for XML: e.g. xml.etree.ElementTree

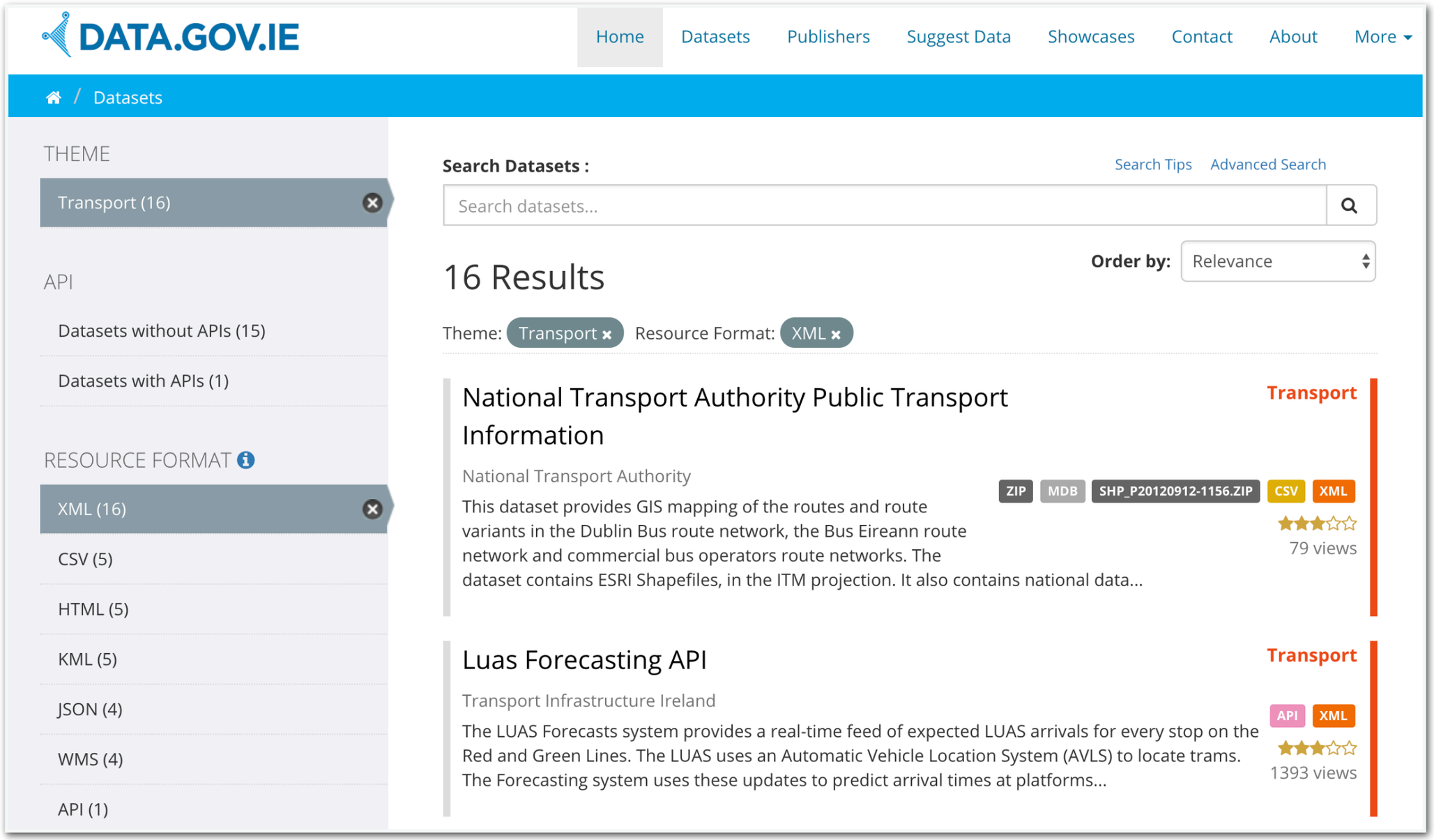
Example: RSS Feeds

- A common use of XML is for news syndication, where frequently updated RSS feeds list the latest articles on a given site.
  e.g. http://www.rte.ie/news/rss/business-headlines.xml

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet title="XSL_formatting" type="text/xsl" href="/rss/rss_convert.xsl"?>
<rss version="2.0" xmlns:atom="http://www.w3.org/2005/Atom">
    <channel>
        <title>RTÉ News - Business Headlines</title>
        <link>http://www.rte.ie/news/business/</link>
        <item>
            <title>Drumm dismisses legal team</title>
            <link>http://www.rte.ie/news/2016/0223/770341-david-drumm/</link>
            <description>Former Anglo Irish Bank CEO David Drumm has dismissed his legal team in his continued appeal
             against a failed bid to be declared bankrupt in the US.</description>
            <pubDate>Wed, 24 Feb 2016 07:34:37 +0000</pubDate>
            <guid isPermaLink="true">http://www.rte.ie/news/2016/0223/770341-david-drumm/</guid>
            <enclosure url="http://img.rasset.ie/000b6adf-144.jpg" type="image/jpeg" length="4094" />
        </item>
        <item>
            <title>Oil prices extend losses on Saudi comments</title>
            <link>http://www.rte.ie/news/business/2016/0224/770390-oil-prices/</link>
            <description>Oil prices fell further in Asia trade today after OPEC kingpin Saudi Arabia shut the door on an
             output cut to ease the global crude supply glut, touting only a freeze in production.</description>
            <pubDate>Wed, 24 Feb 2016 07:41:19 +0000</pubDate>
            <guid isPermaLink="true">http://www.rte.ie/news/business/2016/0224/770390-oil-prices/</guid>
            <enclosure url="http://img.rasset.ie/00036f00-144.jpg" type="image/jpeg" length="4094" />
        </item>
    </channel>
</rss>
```

Example: Ireland Open Data Portal

- The Ireland Open Data Portal provides a range of Irish Public Sector data in open and reusable formats, like XML.



Example: Ireland Open Data Portal

- The Ireland Open Data Portal provides a range of Irish Public Sector data in open and reusable formats, like XML.

Example: National Transport Authority Public Transport Information

```
<?xml version="1.0" encoding="UTF-8"?>
<NationalPublicTransportGazetteer CreationDateTime="2011-01-04T08:03:19" ModificationDateTime="2011-01-04T08:03:19" Modification="revi
        <Regions>
                <Region CreationDateTime="2011-01-18T11:54:31" ModificationDateTime="2011-01-18T11:54:31" RevisionNumber="0">
                        <RegionCode>CON</RegionCode>
                        <Name xml:lang="EN">Connaught</Name>
                        <Country>Eire</Country>
                        <AdministrativeAreas>
                                <AdministrativeArea CreationDateTime="2011-01-18T11:54:31" ModificationDateTime="2011-01-18T11:54:31"
                        <AdministrativeAreas>
                                <AdministrativeArea CreationDateTime="2011-01-18T11:54:31" ModificationDateTime="2011-01-18T11:54:31"
                                        <AdministrativeAreaCode>846</AdministrativeAreaCode>
                                        <AtcoAreaCode>846</AtcoAreaCode>
                                        <Name>Galway City</Name>
                                        <National>false</National>
                                </AdministrativeArea>
                                <AdministrativeArea CreationDateTime="2011-01-18T11:54:31" ModificationDateTime="2011-01-18T11:54:31"
                                        <AdministrativeAreaCode>847</AdministrativeAreaCode>
                                        <AtcoAreaCode>847</AtcoAreaCode>
                                        <Name>Galway County</Name>
                                        <National>false</National>
                                </AdministrativeArea>
                                <AdministrativeArea CreationDateTime="2011-01-18T11:54:31" ModificationDateTime="2011-01-18T11:54:31"
                                        <AdministrativeAreaCode>848</AdministrativeAreaCode>
                                        <AtcoAreaCode>848</AtcoAreaCode>
                                        <Name>Leitrim</Name>
                                        <National>false</National>
                                </AdministrativeArea>
```

https://data.gov.ie/dataset

Data Formats: HTML

- **HTML** is also a markup language similar to XML. Key differences:
  - XML describes data, HTML describes data and presentation.
  - In practice HTML is usually badly-written and invalid.

File: products.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<booklist>
    <book>
        <id>978-1933988177</id>
        <category>paperback</category>
        <name>Lucene in Action</name>
        <author>Michael McCandless</author>
        <genre>technology</genre>
        <price>30.5</price>
        <pages>475</pages>
    </book>
    <book>
        <id>978-1857995879</id>
        <category>paperback</category>
        <name>Sophie's World</name>
        <author>Jostein Gaarder</author>
        <genre>fiction</genre>
        <price>3.07</price>
        <pages>518</pages>
    </book>
</booklist>
```

File: products.html (in browser)

Book List

Lucene in Action

- *Author:* Michael McCandless
- *Genre:* Technology
- *Price:* €30.50
- Paperback, 475 pages

Sophie's World

- *Author:* Sophie's World
- *Genre:* Fiction
- *Price:* €3.07
- Paperback, 518 pages

Data Formats: HTML

- HTML also consists of opening `<tag>` and closing `</tag>`, with other tags and tested enclosed within tags.
- Basic page structure:
  - `<html>...</html>`: Root tag
    - `<head>...</head>`: Page metadata
    - `<body>...</body>`: Page content

File: products.html

```
<html>
    <head>
        <title>Book List</title>
    </head>
    <body>
        <h2>Book List</h2>
        <hr>
        <h3><font color="red">Lucene in Action</font></h3>
        <ul>
            <li><i>Author:</i> Michael McCandless</li>
            <li><i>Genre:</i> Technology</genre></li>
            <li><i>Price:</i> &euro;30.50</li>
            <li>Paperback, 475 pages</li>
        </ul>
        <hr>
        <h3><font color="blue">Sophie's World</font></h3>
        <ul>
            <li><i>Author:</i> Sophie's World</li>
            <li><i>Genre:</i> Fiction</genre></li>
            <li><i>Price:</i> &euro;3.07</li>
            <li>Paperback, 518 pages</li>
        </ul>
        <hr>
    </body>
</html>
```

Book List

Lucene in Action

- *Author:* Michael McCandless
- *Genre:* Technology
- *Price:* €30.50
- Paperback, 475 pages

Sophie's World

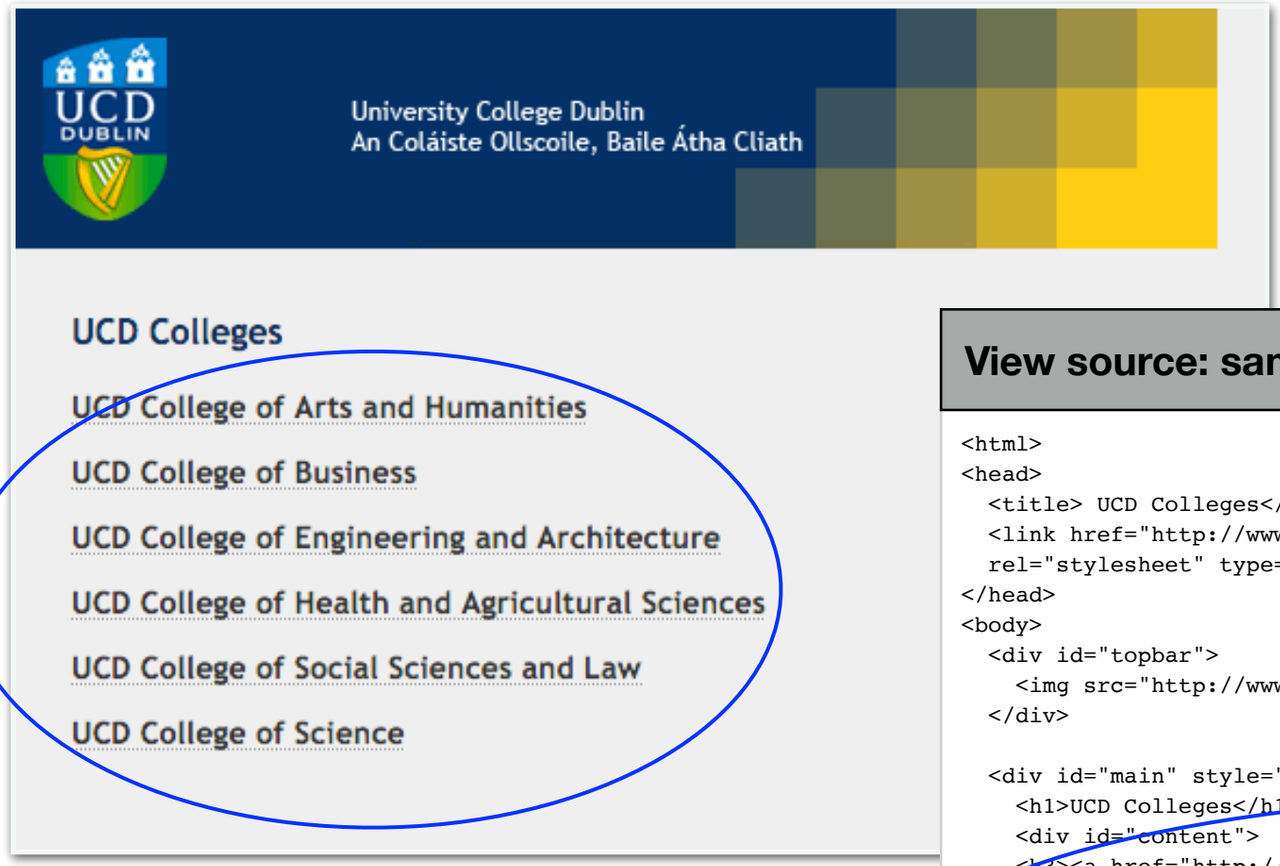
- *Author:* Sophie's World
- *Genre:* Fiction
- *Price:* €3.07
- Paperback, 518 pages

Web Scraping

- Web scraping: Technique used to extract data from web sites using a tool that acts as a web browser.
- **Basic steps:**
  1. *Data identification*: Identify and study the structure of the web pages containing the required data.
  2. *Data collection*: Download these web pages in the same way as a web browser does. This may even simulate a user logging in to obtain access.
  3. *Data extraction*: Parse web pages to extract data.
  4. *Data cleaning*: Clean and reformat the data to make it usable.
- **The "rules" of web scraping:**
  - Check a site's terms and conditions before you scrape their pages.
  - Do not hammer a site with too many automated requests.
  - Sites often change their layout, so scrapers often break and need to be re-written.

Web Data Identification

Example: Want to extract list of UCD college names from the page http://mlg.ucd.ie/modules/COMP41680/sample1.html



View source: sample1.html

```
<html>
<head>
  <title> UCD Colleges</title>
  <link href="http://www.ucd.ie/stylecss/sub/subpage_dropdown.css"
  rel="stylesheet" type="text/css"/>
</head>
<body>
  <div id="topbar">
    <img src="http://www.ucd.ie/images/subpage/1_1.gif">
  </div>

  <div id="main" style="margin: 20px;">
    <h1>UCD Colleges</h1>
    <div id="content">
    <h3><a href="http://www.ucd.ie/artshumanities/">UCD College of Arts and Humanities</a></h3>
    <h3><a href="http://www.ucd.ie/business/">UCD College of Business</a></h3>
    <h3><a href="http://www.ucd.ie/eacollege">UCD College of Engineering and Architecture</a></h3>
    <h3><a href="http://www.ucd.ie/chas/">UCD College of Health and Agricultural Sciences</a></h3>
    <h3><a href="http://www.ucd.ie/socscilaw/">UCD College of Social Sciences and Law</a></h3>
    <h3><a href="http://www.ucd.ie/science">UCD College of Science</a></h3>
    </div>
  </div>
</body>
</html>
```

View HTML source in browser to identify part of page with data of interest

Web Data Collection

- The built-in Python urllib.request module has functions which help in downloading content from HTTP URLs using minimal code:

```
import urllib.request
link = "http://mlg.ucd.ie/modules/COMP41680/sample1.html"
response = urllib.request.urlopen(link)
html = response.read().decode()
```

Fetch the HTML code from the web page

```
<html>
<head>
  <title> UCD Colleges</title>
  <link href="http://www.ucd.ie/stylecss/sub/subpage_dropdown.css"
  rel="stylesheet" type="text/css"/>
</head>
<body>
  <div id="topbar">
    <img src="http://www.ucd.ie/images/subpage/1_1.gif">
  </div>

  <div id="main" style="margin: 20px;">
    <h1>UCD Colleges</h1>
    <div id="content">
    <h3><a href="http://www.ucd.ie/artshumanities/">UCD College of Arts and Humanities</a></h3>
    <h3><a href="http://www.ucd.ie/business/">UCD College of Business</a></h3>
    <h3><a href="http://www.ucd.ie/eacollege">UCD College of Engineering and Architecture</a></h3>
    <h3><a href="http://www.ucd.ie/chas/">UCD College of Health and Agricultural Sciences</a></h3>
    <h3><a href="http://www.ucd.ie/socscilaw/">UCD College of Social Sciences and Law</a></h3>
    <h3><a href="http://www.ucd.ie/science">UCD College of Science</a></h3>
    </div>
  </div>
</body>
</html>
```

- For more intensive data collection tasks, consider using the Python **Requests** module: https://3.python-requests.org/

Web Data Extraction

- Many ways to parse HTML pages in Python. The third-party **Beautiful Soup** package is useful for working with badly written HTML pages: http://www.crummy.com/software/BeautifulSoup

```
<html>
<head>
  <title> UCD Colleges</title>
  <link href="http://www.ucd.ie/stylecss/sub/subpage_dropdown.css"
  rel="stylesheet" type="text/css"/>
</head>
<body>
  <div id="topbar">
    <img src="http://www.ucd.ie/images/subpage/1_1.gif">
  </div>

  <div id="main" style="margin: 20px;">
    <h1>UCD Colleges</h1>
    <div id="content">
    <h3><a href="http://www.ucd.ie/artshumanities/">UCD College of Arts and Humanities</a></h3>
    <h3><a href="http://www.ucd.ie/business/">UCD College of Business</a></h3>
    <h3><a href="http://www.ucd.ie/eacollege">UCD College of Engineering and Architecture</a></h3>
    <h3><a href="http://www.ucd.ie/chas/">UCD College of Health and Agricultural Sciences</a></h3>
    <h3><a href="http://www.ucd.ie/socscilaw/">UCD College of Social Sciences and Law</a></h3>
    <h3><a href="http://www.ucd.ie/science">UCD College of Science</a></h3>
    </div>
  </div>
</body>
</html>
```

In our example, we want to extract the text in the <h3> tags.

We can use BeautifulSoup to find all these tags and get the text between them.

```
import bs4
parser = bs4.BeautifulSoup(html,"html.parser")
for match in parser.find_all("h3"):
    text = match.get_text()
    print(text)
```

```
UCD College of Arts and Humanities
UCD College of Business
UCD College of Engineering and Architecture
UCD College of Health and Agricultural Sciences
UCD College of Social Sciences and Law
UCD College of Science
```

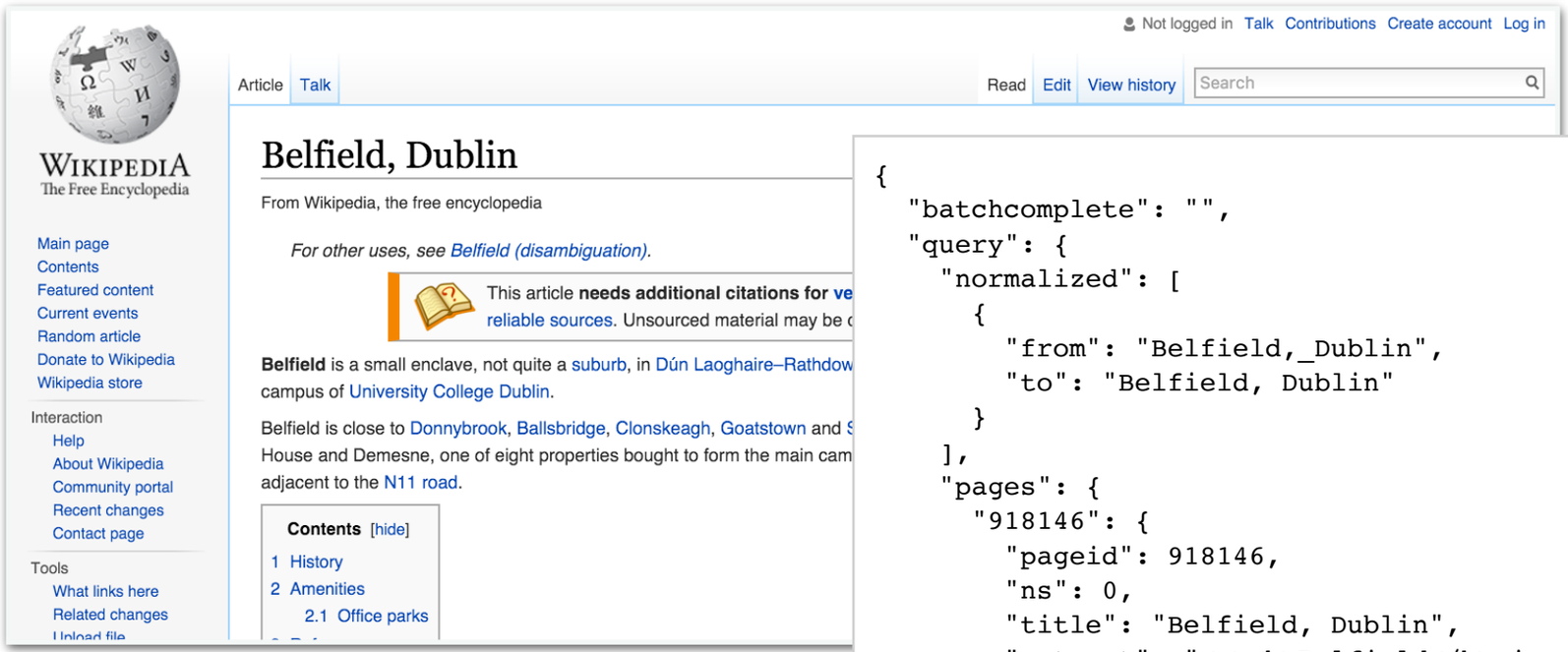
Web APIs

- Instead of manually scraping HTML data, some web sites provide a convenient "official" way of retrieving their data.
- Web application programming interface (API): A set of HTTP request messages with a definition of the expected structure of the response. API calls can return responses in a variety of formats. Sometimes plain text, but more commonly JSON or XML.
- Web APIs have an endpoint - a URL from which we retrieve the data.
- Many Open APIs exist, others are private/paid/proprietary.
- Some Open APIs have third-party Python packages which provide functions that "wrap" the queries and responses. In other cases, HTTP calls will need to be made and responses parsed manually.
- API Restrictions:
  - Many APIs have **rate limits**. Can only make limited number of calls from a given IP address/account, in a fixed amount of time.
  - Some APIs require **authentication** - i.e. first need to register a user account, retrieve an authentication token/password.

Web APIs: Data Collection

- Wikipedia provides a simple Web API for retrieving page content in JSON format, endpoint is https://en.wikipedia.org/w/api.php
- Example query URL for the Wikipedia page "Belfield,_Dublin"

  https://en.wikipedia.org/w/api.php?format=json&action=query&prop=extracts&exintro=true&titles=Belfield,_Dublin



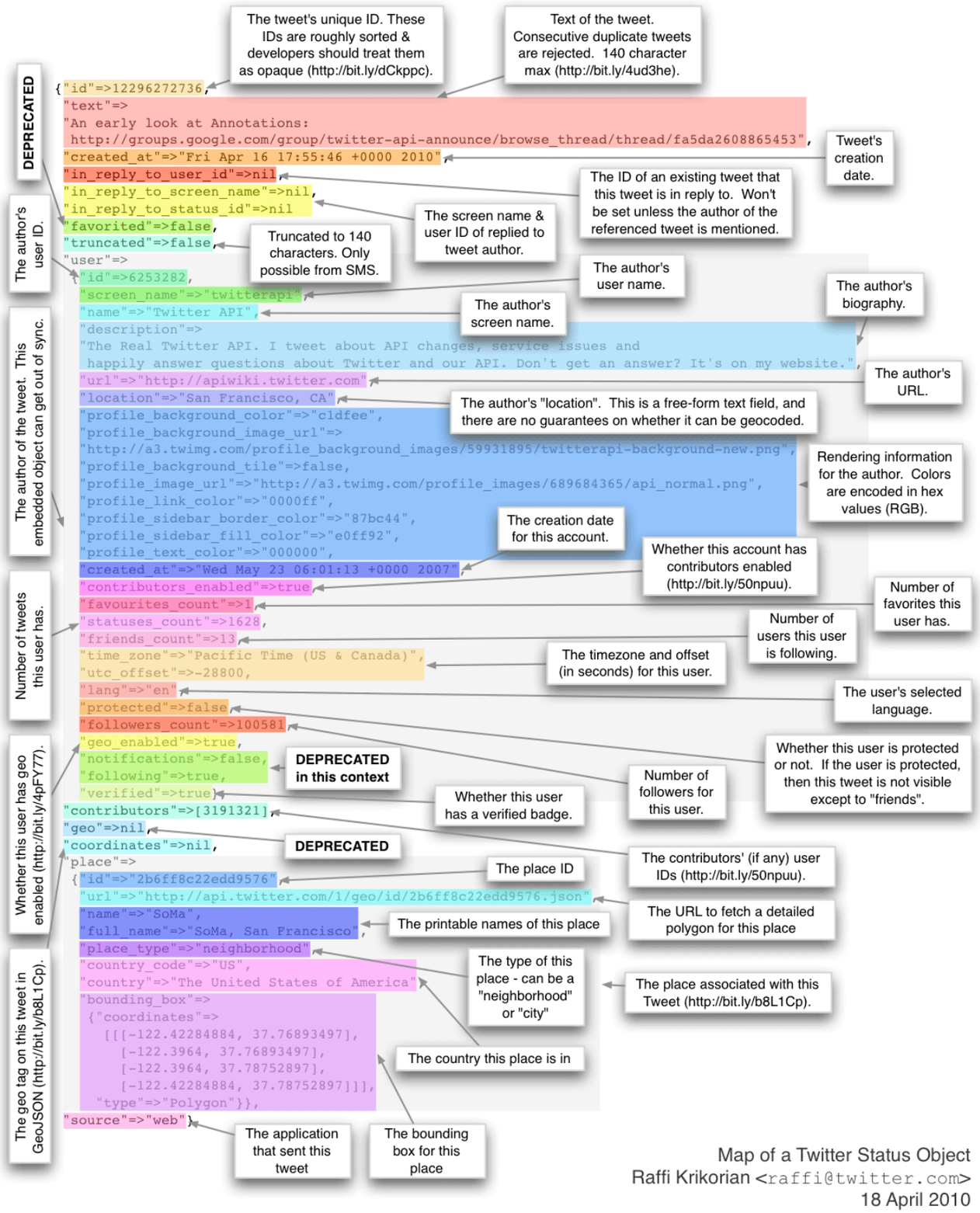

```
{
  "batchcomplete": "",
  "query": {
    "normalized": [
      {
        "from": "Belfield,_Dublin",
        "to": "Belfield, Dublin"
      }
    ],
    "pages": {
      "918146": {
        "pageid": 918146,
        "ns": 0,
        "title": "Belfield, Dublin",
        "extract": "<p><b>Belfield</b> is a small enclave, not quite a suburb, in
Dún Laoghaire–Rathdown, Ireland. It is synonymous with the main campus of
University College Dublin.</p>\n<p>Belfield is close to Donnybrook, Ballsbridge,
Clonskeagh, Goatstown and Stillorgan and takes its name from Belfield House and
Demesne, one of eight properties bought to form the main campus of University
College Dublin. It is adjacent to the N11 road.</p>\n<p></p>"
      }
    }
  }
}
```

Twitter APIs

- Twitter provides a number of APIs for collecting data:
  1. **Streaming API**: Access a sample of the stream of all public tweets flowing through Twitter. https://dev.twitter.com/streaming
  2. **Search API**: Access a relevant set of recent tweets based on a user-specified search query. https://dev.twitter.com/rest/public/search
  3. **REST APIs**: Programmatic access to read and write Twitter data - e.g. posts tweets, access follower information, trending topics. https://dev.twitter.com/rest/public
- To access the APIs, you need a Twitter account and credentials (i.e. API key, API secret, Access token and Access token secret) from the Twitter developer site. See https://dev.twitter.com/oauth
- Note that rate limits apply on all Twitter APIs.

Twitter APIs

- Twitter API requests are made to access points on twitter.com
- The results are returned in JSON format.
- Even a result for a single tweet can contain a significant amount of data...



Twitter APIs

- Since the JSON data returned by Twitter is often long and complex, many wrapper packages exist for accessing the APIs.
  e.g. For Python, the **Tweepy** Package: http://www.tweepy.org
- **Search API** example. Note the limit for the Search API is 180 requests per 15 minute time window.

```
CONSUMER_KEY = "kq6vOjtH0UDydaxbHduFwyzdK"
CONSUMER_SECRET = "RyTEJrYfDwQQggHGqkU6brYDlYmvRu8YZV"
ACCESS_KEY = "2744628678-7M86AJnrHIQMd6zEpOA04hgmqHzxm"
ACCESS_SECRET = "G82dp6QfdDukOXxTYvhOwtIwcQprrjOAKnOe"

import tweepy
auth = tweepy.auth.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
api = tweepy.API(auth)
```

Authenticate here. Replace keys with your OAuth details

```
results = api.search(q="Dublin", count=5)
for tweet in results:
   print(tweet.text)
```

Retrieve 5 most recent tweets containing "Dublin"

```
#job alert.. https://t.co/y5ovHiYOOR || Enterprise Inside Sales Account Manager - Dublin Dublin-Dublin
IKEA to open second store in Dublin https://t.co/N13Pzy1UjX https://t.co/fPC5gdUSiS
Setting Out Engineer (834), Dublin, Competitive  #job #jobs #hiring #ConstructionJobs https://t.co/5rwPMFUUlp
Demolition Works Underway to Allow for Construction of €9m Office Development at Charlemont Place, Dublin 2
Just commented on @thejournal_ie: A second Ikea store is coming to Dublin this summer
```

Twitter APIs

- Tweepy allows access to the **Streaming API**, to download tweets in real time via a persistent connection. These can be filtered by keyword or location.
- The stream will not terminate unless the connection is closed (e.g. you kill your Python script).

```
import tweepy
auth = tweepy.auth.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
api = tweepy.API(auth)
```

Authenticate your account first

```
class MyListener(tweepy.StreamListener):
   def on_status(self, tweet):
      print( "Got a new tweet:", tweet.text)
```

Code to handle new tweets as they arrive

```
listener = MyListener()
stream = tweepy.Stream(auth = api.auth, listener=MyListener())
stream.filter(track=['#bigdata'])
```

Start listening to the stream, but only handle tweets containing #bigdata

```
Got a new tweet: RT @jose_garde: Figures Don’t Lie, but Liars Can Figure #BigData #analytics
Got a new tweet: RT @headcountguru: Three Unexpected Uses for 3D Printing in Big Data #bigdata
Got a new tweet: RT @jose_garde: Is predictive policing the law-enforcement tactic of the future? #BigData #analytics
```

Twitter APIs

- Detailed information about Twitter users can be retrieved via Tweepy's wrapper for the **REST API**, including:
  - Basic profile information.
  - A user's 3200 most recent tweets.
  - A user's set of friends and followers.
  - A user's liked tweets.

```
import tweepy
auth = tweepy.auth.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
api = tweepy.API(auth)
```

Authenticate your account first

```
user = api.get_user( "ManUtd" )
print("Name:", user.name)
print("Location:", user.location)
print("Friend Count:", user.friends_count)
print("Follower Count:", user.followers_count)
print("Tweet Count:", user.statuses_count)
```

Retrieve profile information for the user @ManUtd

```
Name: Manchester United
Location: Manchester, England
Friend Count: 89
Follower Count: 7664118
Tweet Count: 25651
```

Google Maps APIs

- Google provides range of APIs for location-based services:
  1. **Geocoding API**: Convert addresses and place-names to latitude-longitude coordinates.
  2. **Directions API**: Return multi-part directions for a series of locations for different modes of transport.
  3. **Distance API**: Retrieve duration and distance values based on the recommended route between start and end points.
  4. **Places API**: Large database of information about establishments, geographic locations, or prominent points of interest.
- Using the APIs requires a Google account and a Google Maps API Key. Create one at https://console.developers.google.com
- Wrapper libraries exist for the web services. e.g. For Python: https://github.com/googlemaps/google-maps-services-python

Google Maps APIs

- Example use of **Geocoding API** to convert an incomplete address to a full address with coordinates.

```
KEY = "AsdfdsSyCDoY-Bf3oasdxsEaFtRHzh0a-Ct4z"
gmaps = googlemaps.Client(key=KEY)
```

Replace with your developer key

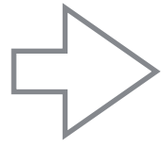
```
address = "Aviva Stadium, Dublin, Ireland"
results = gmaps.geocode(address)
for r in results:
   print(r["formatted_address"])
   print(r["geometry"]["location"])
```

Search for address may produce multiple matches

```
Aviva Stadium, 62 Lansdowne Rd, Dublin Southside, Dublin 4, Ireland
{'lng': -6.2261824, 'lat': 53.33483039}
```

- Example use of **Places API** to find open restaurants near the specified coordinates (or for reverse geocoding).

```
mylocation = (53.33483039,-6.2261824)
response = gmaps.places("restaurant", location=mylocation,
radius=100, open_now=True)
for r in response["results"]:
   print(r["name"])
```



```
Juniors
Farmer Browns
The Chop House
MAIA Cafe Restaurant
Ravi's Kitchen
...
```

YouTube API

- The YouTube Data API v3 provides access to YouTube data, such as videos, playlists, and channels. Using the APIs requires a Google account and a YouTube Data API Key.
  Create one at https://console.developers.google.com
- Many Google services such as YouTube can be accessed via the Python API Client Library package:
  https://developers.google.com/api-client-library/python

```
from apiclient.discovery import build
KEY = "AsdfdsSyCDoY-Bf3oasdxsEaFtRHzh0a-Ct4z"
youtube = build("youtube", "v3", developerKey=KEY)
```

Replace with your developer key

```
yt = youtube.search()
response = yt.list(q="UCD", part="snippet", type="video",
maxResults=5).execute()
for item in response["items"]:
  print(item["snippet"]["title"])
```

Fetch data for 5 top videos matching the query "UCD"

```
UCD, Dublin campus tour with student ambassadors
Room Tour | University College Dublin 2015
UCD Choral Scholars - The Gartan Mother's Lullaby (Official Video)
Free and Easy - UCD Choral Scholars & UCD Ad Astra Ensemble
DCU vs. UCD - Epic Rap Battles of COLLEGES
```

Display title of each video in the results

Facebook API

- The Graph API is the main way to read and write to the Facebook social graph: https://developers.facebook.com/docs/graph-api
- Access requires registering an App associated with your Facebook account, then generating an OAuth access token - this is unique to the combination of a logged in user and the Facebook App that makes the request.
- The API includes support for:
  - Posting to Facebook on behalf of a user.
  - Accessing data on friend's pages and friend lists.
  - Accessing comments on public Facebook pages.
  - Checking the number of lines across pages.
  - Searching for users or pages by name.
- Python wrapper for Graph API: https://github.com/mobolic/facebook-sdk

API Resources

Sample Public APIs:

- Google Translate API - https://developers.google.com/translate/
- World Bank APIs - http://data.worldbank.org/developers
- Guardian Content API - http://open-platform.theguardian.com/
- New York Times Developer Network - http://developer.nytimes.com/
- Weather Underground API - https://www.wunderground.com/weather/api/
- Flickr API - https://www.flickr.com/services/api/
- Reddit API - https://www.reddit.com/dev/api/
- Foursquare API - https://developer.foursquare.com/
- OpenStreetMap API - https://www.openstreetmap.org

Sample Web API Lists:

- https://github.com/toddmotto/public-apis
- http://www.programmableweb.com/apis/directory
- http://www.computersciencezone.org/50-most-useful-apis-for-developers

Advice Slip API

ADVICE SLIP JSON API

Advice Slip JSON API

Random advice

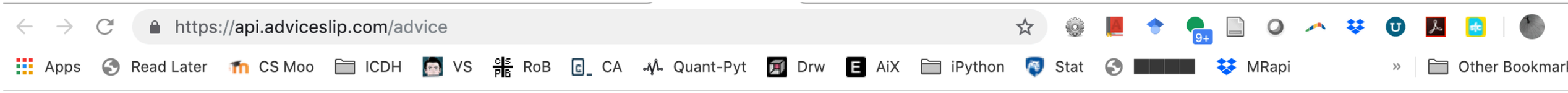
HTTP Method	GET
URL	https://api.adviceslip.com/advice
Description	Returns a random advice slip as a slip object.
Parameters	`callback` string To define your own callback function name and return the JSON in a function wrapper (as JSONP), add the parameter callback with your desired name as the value.

Advice by ID

HTTP Method	GET
URL	https://api.adviceslip.com/advice/{slip_id}
Description	If an advice slip is found with the corresponding {slip_id}, a slip object is returned.
Parameters	`callback` string To define your own callback function name and return the JSON in a function wrapper (as JSONP), add the parameter callback with your desired name as the value.

Searching advice

HTTP Method	GET
URL	https://api.adviceslip.com/advice/search/{query}
Description	If an advice slip is found, containing the corresponding search term in {query}, an array of slip objects is returned inside a search object.
Parameters	`callback` string To define your own callback function name and return the JSON in a function wrapper (as JSONP), add the parameter callback with your desired name as the value.



{"slip": {"advice":"Be brave. Even if you're not, pretend to be. No one can tell the difference.","slip_id":"199"}}