GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 4: WEB SCRAPING AND COMMAND LINE

Paul Burkard

07/02/2015

LAST TIME:

I. WHAT ARE NOSQL DATABASES?

II. WHY PYTHON?

III. WHY PANDAS?

WEB SCRAPING:

I. INTRO TO WEB SCRAPING

HANDS-ON: WEB SCRAPING EXERCISES

COMMAND LINE:

II: INTRO TO COMMAND LINE

HANDS-ON: COMMAND LINE EXERCISES

- What is Web Scraping?
  - How do we do it in Python?
  - What are HTML and XML?
  - How do we work with web APIs in Python?
- What is the Unix Command Line?
  - What are some common command line commands and operations?
  - How can we incorporate the command line into our workflow?

I. INTRO TO WEB SCRAPING

*Q: What is **Web Scraping**?*

A: Retrieving data from a website in a format suitable for analysis

- *Most involves parsing **HTML**, or occasionally **XML***
- *Alternatively many websites offer public **APIs** (**A**pplication **P**rogram **I**nterface) with open methods for common data retrieval operations*
- *Websites often contain rich data, but also mountains of extraneous content that we need to wade through to get the stuff that we want*

Web Scraping in Python

- *We will use **BeautifulSoup***
- *Other options:*
  - ***Scrapy**, **lxml**, **HTQL**, **Mechanize***

ASIDE: HTML

*Q: What is **HTML**?*

*A: **HTML** is a markup language for describing web documents*

- *HTML stands for **H**yper **T**ext **M**arkup **L**anguage*
- *A markup language is a set of **markup tags***
- *HTML documents are described by **HTML tags***
- *Each HTML tag **describes** different document **content***

Sample HTML snippet

```
<!DOCTYPE html>
<html>
<head>

</head>
<body>

<table style="width:100%">
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
</table>

</body>
</html>
```

*Q: How is **HTML** used?*

A:

- ***Designers** use it to create webpages*
- ***Browsers** interpret the **HTML markup** to display the webpages*
- *Different **HTML tags** can provide many different types of **content***
  - *Headers, spacing, tables, audio, images, video, links, etc.*
- *Here is a sufficient **HTML Tutorial***

ASIDE: XML

*Q: What is **XML**?*

*A: **XML** is a markup language for describing data*

- ***XML** stands for E**X**tensible **M**arkup **L**anguage*
- ***XML** is a **markup language** much like **HTML***
- ***XML** was designed to **describe/carry data**, not to display data (HTML)*
- ***XML** tags are not predefined. You must define your own tags*
- *Here is a sufficient **XML Tutorial***

Sample XML snippet

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

ASIDE: WEB APIs

*Q: What is an **API**?*

A: When an application allows access to certain programmatic functions to interact with its system

- ***API** stands for **A**pplication **P**rogram **I**nterface*
- ***Web applications** with APIs allow users to access them by hitting **specific URLs** with the appropriate **HTTP Requests***
- *Results are returned in various prescribed data formats, commonly **JSON***

Sample Yelp Web API call

```
def search(term, location):
    """Query the Search API by a search term and location.
    Args:
        term (str): The search term passed to the API.
        location (str): The search location passed to the API.
    Returns:
        dict: The JSON response from the request.
    """

    url_params = {
        'term': term.replace(' ', '+'),
        'location': location.replace(' ', '+'),
        'limit': SEARCH_LIMIT
    }
    return request(API_HOST, SEARCH_PATH, url_params=url_params)
```

Some Public Web APIs:

- *Yelp*
- *Facebook*
- *Twitter*
- *ESPN*
- *StubHub*
- *EchoNest*
- *Spotify*
- ***Many more!***

ASIDE: JSON

*Q: What is **JSON**?*

*A: **JSON** is a syntax for storing and exchanging data*

- ***JSON** stands for **J**avaScript **O**bject **N**otation*
- *Many programming languages (including Python) contain easy functions for converting JSON into usable objects*
- *JSON is "**self-describing**" and **easy to understand***
- *Doesn't require as strict schema structure as XML*

Sample JSON snippet

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

*Q: How is **JSON** used with Web APIs?*

A:

- ***Users** make appropriate Web API calls*
- ***Web Applications** return **results** of queries in **JSON***
- ***JSON** is converted into **programming objects** to be manipulated*
- *Here is a sufficient JSON Tutorial*

HANDS ON: WEB SCRAPING

II. INTRO TO COMMAND LINE

*Q: What is the **UNIX Command Line**?*

*A: It's what you're using in the **Terminal**!*

- *Commands for navigating a UNIX-based (MacOSX, Linux) operating system*
  - *Navigating the filesystem, operating on files, viewing files, system info and stats, built-in functions, etc*
- *Programming language in its own right*
  - *You can write **scripts**, **functions** etc*
  - *There are many of these "**shell**" languages, but we will use **Bash***
- *Sometimes quicker for simple data manipulation than Python, R, etc*

EXERCISE: WORKING AT THE COMMAND LINE

KEY OBJECTIVES

- Navigate the filesystem
- Create, move, copy, and delete files & directories
- View & search files
- Edit & interact with files
- Combine steps
- Learn more

TOOLS

- ls, cd
- cat, touch, mv, cp, mkdir, rm, rmdir

- head, tail, less, cat, grep
- vim, tr, sort, uniq, wc
- pipe (|)
- man, apropos

NOTE

Being comfortable at the command line can make your life much easier!

HANDS ON: COMMAND LINE