Programming with Python: Beyond the Basics

How to Write a Web Scraper in Python

Set up

- Python 3.6+ installed
- An IDE for Python (PyCharm recommended)
- Course material downloaded and unzipped
- Resources downloaded (PDF slides and course reference sheet)
- Go to https://github.com/ariannedee/python-level-2 for the course material and step-by-step instructions

Today's schedule

- **Introduction, set-up, and review** (35 mins)
  - Break, Q&A
- **More concepts: dictionaries and exceptions** (25 min)
- **Reading and writing to files** (30 mins)
  - Break, Q&A
- **Scraper foundations** (40 mins)
  - Break, Q&A
- **Build a scraper** (50 mins)
- **Further discussion** (10 mins)

Break format

- 3 Breaks (10 mins each)
  - Step away or work through code
- Q&A (5 mins)
  - Use Q&A feature
- Use group chat throughout for questions that anyone can answer

Introduction

Poll (single choice)

How long have you been programming?

- Less than a week
- Less than a month
- Less than a year
- 1 - 3 years
- 3 - 10 years
- 10+ years

Poll (multi-choice)

Have you taken any of my other classes?

- Introduction to Python Programming (webinar)
- Introduction to Python LiveLessons (recorded video)
- Object-Oriented Programming in Python
- Rethinking REST: A hands-on guide to GraphQL
- None

Poll (multi choice)

- Why are you interested in this class
  - I’m a programmer in another language and want to learn Python
  - I am new to programming and am starting off with Python
  - I just want to build a web scraper
  - Other

Introduction

Installation

Set up

- Download the PDF of these slides and the Reference document (Resources widget)

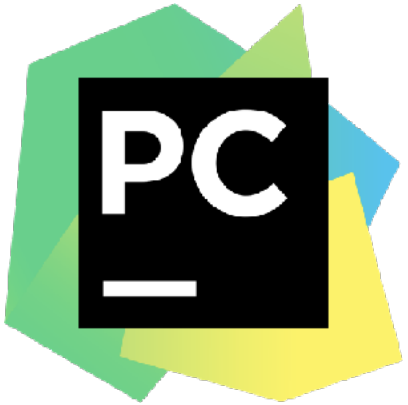
- Go to https://github.com/ariannedee/python-level-2 and follow the installation instructions in the Readme

  - Python 3.6+ installed
  - An IDE for Python (PyCharm recommended)
  - Course material downloaded and unzipped

Install links

- Install Python 3.8.x for your operating system
  - https://www.python.org/downloads/
- Download the free, community edition of PyCharm
  - https://www.jetbrains.com/pycharm/download/
- Download the code
  - https://github.com/ariannedee/python-level-2

PyCharm IDE



- Supports syntax and error highlighting for **Python**
- Integrated Terminal/Command Line
- Package installation without command line

Reviewing Python Basics

Functions, conditionals, lists, and for-loops

Conditionals

```
if temp <= 0:
    print("It’s freezing")
elif temp >= 100:
    print("It’s boiling")
else:
    print("It's alright")
```

Lists

```
populous_countries = ["China", "India", "USA", "Indonesia", "Brazil"]

populous_countries[2] = "United States"
populous_countries.append("Pakistan")
```

For Loops

```
for i in range(10):
    print(i * i)
```

While Loops

```
seconds_left = 3

while seconds_left > 0:
    print(seconds_left)
    seconds_left -= 1

print("Lift off!")
```

Functions

```
def square(num):
    return num * num

square(10)
```

Let’s do some practice

Syntax

- For certain keywords (e.g. `if`, `for`, `while`, `def`)
  - Use colon at end of line
  - Indent next line(s) to define a code block
    - 4-spaces (by convention)
    - All lines in block of code must be indented the same amount
    - Can be nested

Poll

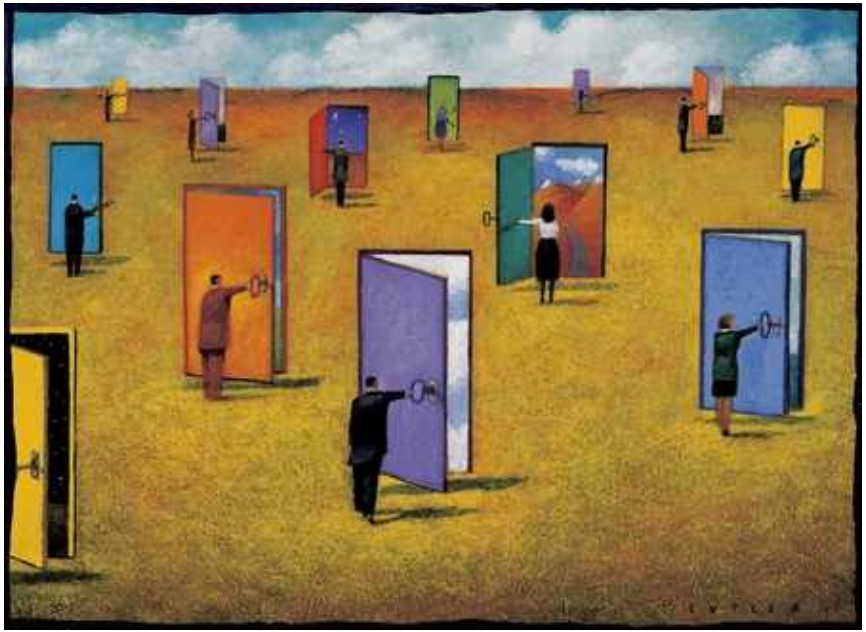
- How much of the review content do you feel comfortable with?
  - None of it - This is too advanced for me
  - Some of it - I'll be struggling
  - Most of it - I'll be following along
  - All of it - It was good to review
  - All of it and more - It was very basic

More Concepts

Dictionaries and exceptions

Dictionaries

Key-value relationship



Dictionary examples

- Key-value examples
  - Dictionary: word (key), definition (value)
  - Thesaurus: word (key), synonyms (value)
  - Phone book: name (key), phone number (value)

- Can also be used to store data about objects
  - User with keys: name, email, birthday, country
  - Book with keys: title, author, published year

Exceptions

```
# Request a number from the user
number = int(input("Enter a positive whole number: "))
```

Questions and break

Q&A widget

Reading and Writing to Files

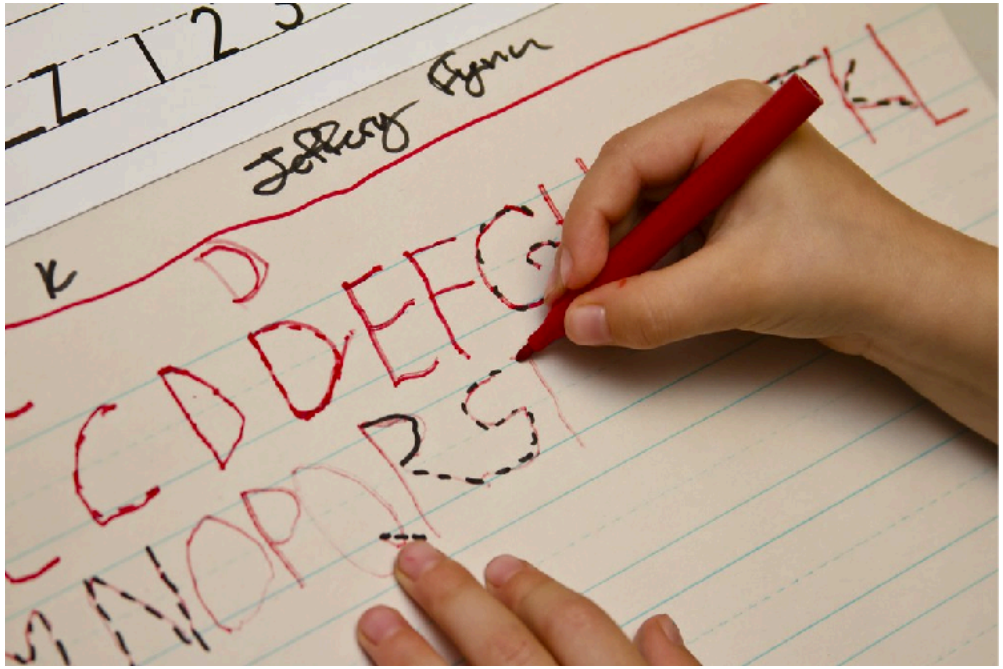
Reading from files

Sample data

Country list: https://gist.github.com/kalinchernev/486393efcca01623b18d

Comprehensive list: https://github.com/umpirsky/country-list

Writing to files



CSV files

Comma-separated values

Name, Age
Shehin, 23
Freddy, 85
Bob, 5
Gabriella, 62

Name	Age
Shehin	23
Freddy	85
Bob	5
Gabriella	62

Scraper Foundations

Installing external libraries

Pip

```
ariannedee$ pip install requests
```

Pip commands

- **Install package(s)**
  - `$ pip install`
- **Uninstall package**
  - `$ pip uninstall`
- **List all installed packages**
  - `$ pip list`

More advanced **pip** commands

- **Install specific package**
  - `$ pip install SomePackage` # latest version
  - `$ pip install SomePackage==1.0.4` # specific version
  - `$ pip install 'SomePackage>=1.0.4'` # minimum version
- **Upgrade package to newest version**
  - `$ pip install -U SomePackage`
- **Install with proxy**
  - `pip install --proxy proxy.server:port package`
  - or `--proxy [user:passwd@]proxy.server:port`

Pip with requirements.txt

- **Create text file with all installed packages + versions**
  - `$pip freeze > requirements.txt`

- **Install all packages + versions from text file**
  - `$ pip install -r requirements.txt`

Further reading

- Beginner tutorial
  - What is Pip? A Guide for New Pythonistas

- Using `requirements.txt` files to save your list of libraries
  - Why and how to make a requirements.txt

Scraper Foundations

Making HTTP Requests with the `requests` library

Requests

wikipedia.org

Scraper Foundations

Introduction to HTML

Poll

- Do you know HTML?
  - Not at all
  - A bit, and I would like to review it
  - A bit, but I don’t want to review it
  - Yes

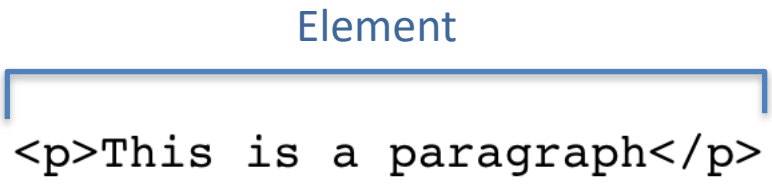
HTML page structure

```
<html>
  <body>
    <h1>This a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

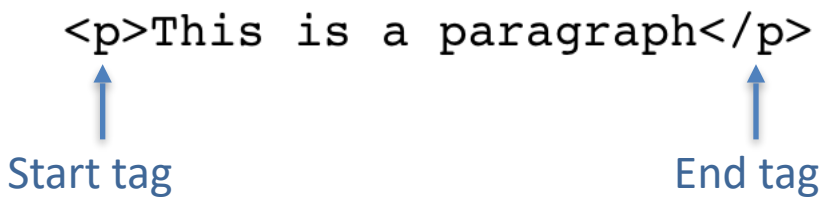
HTML elements

```
<p>This is a paragraph</p>
```

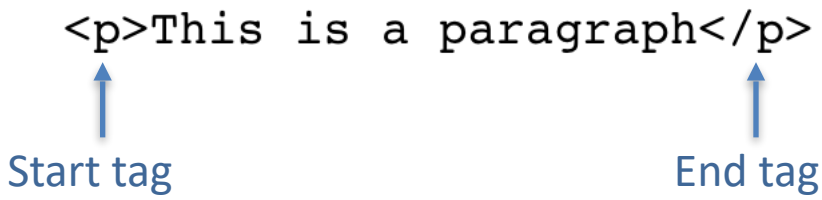
HTML elements



HTML elements



HTML elements

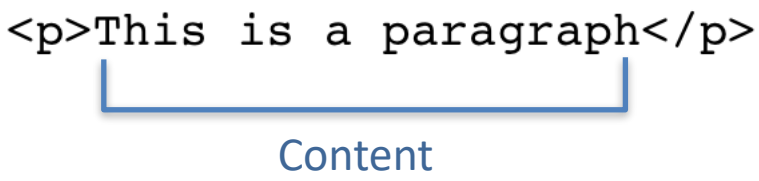


Tags start with "**<**" and end with "**>**"

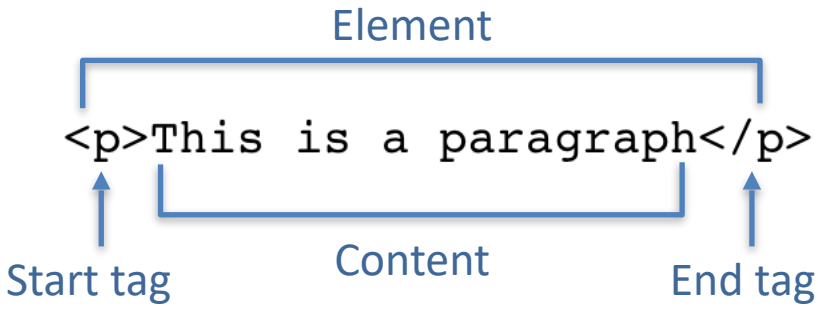
Tags have a name (e.g. **p** is for paragraph)

End tags have a "**/**" before the name

HTML elements



HTML elements



Common tags

- `p` - paragraph
- `div` - divider
- `a` - link (a.k.a anchor)
- `h1 … h6` - heading

- `table`
  - `th` - table header
  - `tr` - table row
  - `td` - table data

Nesting

```
<div>
  <p>This is a paragraph inside a div</p>
</div>
```

Nesting

```
<table>
  <tr>
    <th>Title</th>
    <th>Author</th>
  </tr>
  <tr>
    <td>Animal Farm</td>
    <td>George Orwell</td>
  </tr>
  <tr>
    <td>Pride and Prejudice</td>
    <td>Jane Austen</td>
  </tr>
</table>
```

Nesting

```
<table>
  <tr>
    <th>Title</th>
    <th>Author</th>
  </tr>
  <tr>
    <td>Animal Farm</td>
    <td>George Orwell</td>
  </tr>
  <tr>
    <td>Pride and Prejudice</td>
    <td>Jane Austen</td>
  </tr>
</table>
```

Nesting

```
<table>
<tr><th>Title</th><th>Author</th>
</tr><tr>
<td>Animal Farm</td><td>George Orwell</td>
</tr><tr>
<td>Pride and Prejudice</td><td>Jane Austen</td>
</tr></table>
```

HTML page structure

```
<html>
  <body>
    <h1>This a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Attributes

```
<a href="http://example.com">Visit website</a>
```

Attributes

```
<a href="http://example.com">Visit website</a>
```

Attributes are listed after the tag name, with an "**=**" after

The value of an attribute is in quotes after the "**=**"

If there are multiple attributes, they have a space between them

Common attributes

- `id`
  - A unique identifier for an element
- `class`
  - Often used for determining the styling of an object
  - E.g. Menu link has a class “active” so the styling is different for the current page
- `href`
  - The URL for a link (**h**ypertext **ref**erence)
  - Required for links ("`a`" tag)
- `src`
  - The source location of an image
  - Required for images ("`img`" tag)

Sample HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Member states of the United Nations - Wikipedia</title>
</head>
<body class="mediawiki">
<h1>Member states of the United Nations</h1>
<div id="bodyContent">
  <p>
    The current members and their dates of admission are listed below with their
    official designations used by the United Nations.
  </p>
  <table class="wikitable">
    <caption>UN member states
    </caption>
    <tbody>
    <tr>
      <th><a href="/wiki/Flag">Flag</a>
      </th>
      <th>
        Member state
      </th>
      <th>
```

Sample HTML

```
<tbody>
<tr>
  <th><a href="/wiki/Flag">Flag</a>
  </th>
  <th>
    Member state
  </th>
  <th>
    Date of admission
  </th>
  <th>
    See also
  </th>
</tr>
<tr>
  <td>
    <div>
      <span class="flagicon">
        <a href="/wiki/Afghanistan" title="Afghanistan">
          <img alt="Afghanistan" src="//upload.wikimedia.org/wikipedia/commons/thur
        </a>
      </span>
    </div>
  </td>
  <td>
    <a href="/wiki/Afghanistan" title="Afghanistan">Afghanistan</a>
  </td>
```

Scraping websites

Scraping data

Python scraper options

- **Beautiful Soup** - simple
- **lxml** - more technical, supports xml
- **Scrapy** - advanced features, full scraper capability
- **Selenium** - handles JavaScript and user events, slow
- **Requests-HTML** - simple, but not production-ready

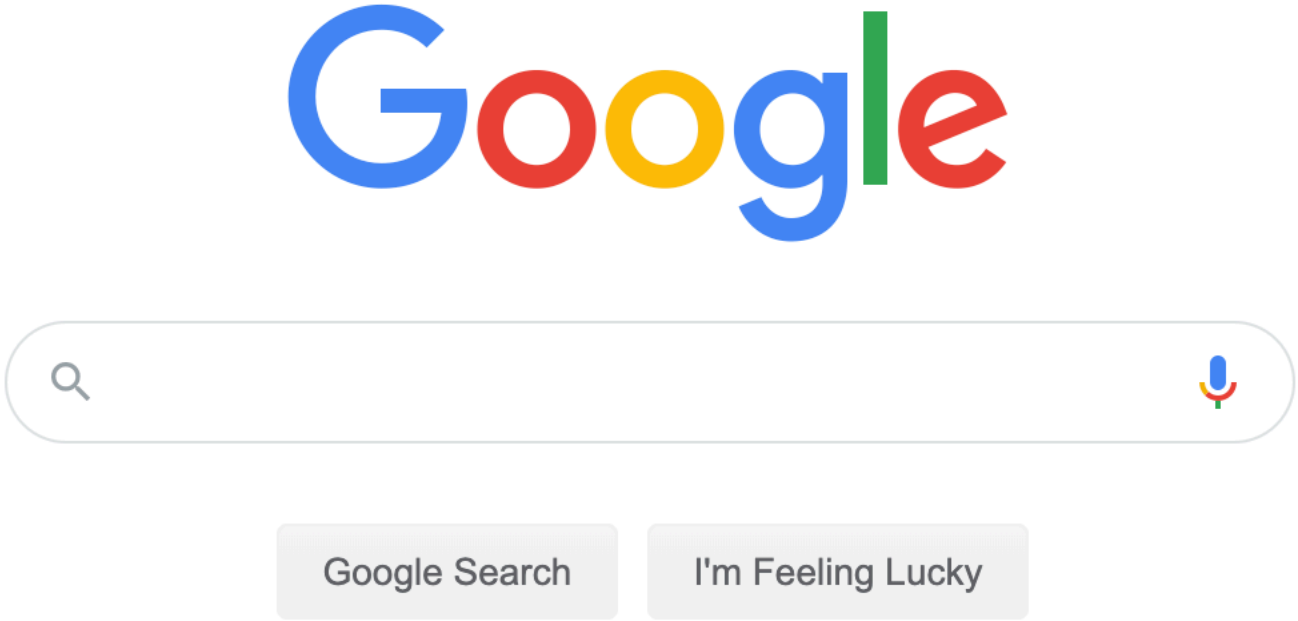
Beautiful Soup 4

You didn't write that awful page. You're just trying to get some data out of it. Beautiful Soup is here to help. Since 2004, it's been saving programmers hours or days of work on quick-turnaround screen scraping projects.

Install BeautifulSoup

```
ariannedee$ pip install beautifulsoup4
```

Practise: find the buttons



Pearson

Questions and break

Q&A widget

Project data

Countries in the United Nations

https://en.wikipedia.org/wiki/Member_states_of_the_United_Nations

UN member states

Flag	Member state[7][13][14]	Date of admission	See also
	Afghanistan	19 November 1946	*United Nations Assistance Mission in Afghanistan*
	Albania	14 December 1955	
	Algeria	8 October 1962	
	Andorra	28 July 1993	
	Angola	1 December 1976	
	Antigua and Barbuda	11 November 1981	
	Argentina	24 October 1945	
	Armenia	2 March 1992	Former member: *Union of Soviet Socialist Republics* (original member)
	Australia	1 November 1945	*Australia and the United Nations*
	Austria	14 December 1955	
	Azerbaijan	2 March 1992	Former member: *Union of Soviet Socialist Republics* (original member)
	Bahamas	18 September 1973	
	Bahrain	21 September 1971	

https://en.wikipedia.org/wiki/Member_states_of_the_United_Nations

Ethics

- Let the website know who you are and how to contact you
- Limit the rate at which you make requests
- Use public APIs instead of scraping when you can
- Read more: Ethics in Web Scraping - James Densmore

Wikipedia Terms of Use

4. Refraining from Certain Activities

Engaging in Disruptive and Illegal Misuse of Facilities

- Engaging in automated uses of the site that are abusive or disruptive of the services and have not been approved by the Wikimedia community;
- Disrupting the services by placing an undue burden on a Project website or the networks or servers connected with a Project website;
- Disrupting the services by inundating any of the Project websites with communications or other traffic that suggests no serious intent to use the Project website for its stated purpose;
- …

https://foundation.wikimedia.org/wiki/Terms_of_Use/en

Let’s code!

Project data

Countries in the United Nations

https://en.wikipedia.org/wiki/Member_states_of_the_United_Nations

Notes

- You might not be able to follow along the whole time
- Remember that you'll get a recording of this lesson in a day or two
- There are 3 versions of the solution file:
  - Retrieving all country names and printing to .txt
  - Countries and the date they joined the UN to .csv
  - Countries and more info from their country page

Wrapping up

Follow-up, feedback, etc.

More examples

- Authenticated sites (login required)
- REST APIs
- GraphQL APIs

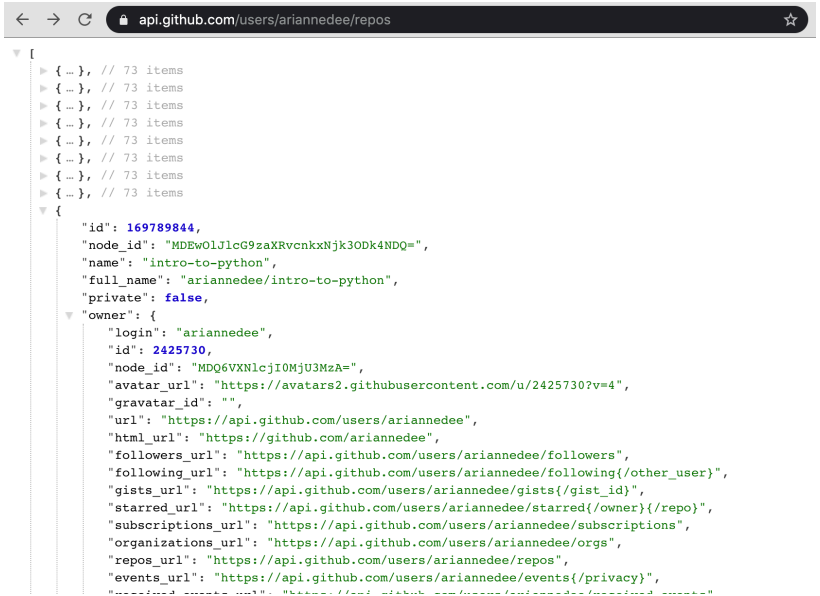
What is an API?

- Short for **A**pplication **P**rotocol **I**nterface
- A structured way of retrieving data without a graphical interface
- Can return multiple document types, but JSON is the most common
- REST APIs are the most common
  - **RE**presentational **S**tate **T**ransfer

REST APIs

URL: https://api.github.com/users/ariannedee/repos

Returns the list of repositories for a user



REST APIs

- Have a root API URL
  - e.g. https://api.github.com/
- Have a different endpoint for each resource you want to access
  - e.g. https://api.github.com/users
- Filter via URL or query parameters
  - e.g. https://api.github.com/users/ariannedee/repos?type=member

REST API resources

- Wikipedia
  - https://en.wikipedia.org/wiki/Representational_state_transfer
- Very basic intro video to APIs
  - https://www.youtube.com/watch?v=s7wmiS2mSXY
- What are RESTful APIs video
  - https://www.youtube.com/watch?v=SLwpqD8n3d0
- Tutorial
  - https://realpython.com/api-integration-in-python/

Other types of APIs

- GraphQL APIs
  - e.g. https://developer.github.com/v4/explorer
  - Documentation: https://developer.github.com/v4/
  - Short intro video: https://www.youtube.com/watch?v=zvZP0PVAdR0

- SOAP (Simple Object Access Protocol)
- RPC (Remote Procedure Call)

More web scraper ideas

- Other data gathering options:
  - weather, news, sports, stock prices, currency exchange rates, etc
- Custom notifications across different websites - jobs, classifieds, flights
- Analyze Twitter or other social media content

Advanced scraper ideas

- Create your own visualizations of scraped data over time
  - Create a “cron-job” to run your script every day and gather new data
  - Store that data either in a file or in a database (db)
    - SQLite is the simplest db
  - Create a new script that visualizes all of the data collected in a time frame

More scraper tutorials

- Follow the stock price
  - Free Code Camp tutorial - easy
- Trip Advisor reviews
  - Medium tutorial - Advanced
  - Use Selenium library to load JavaScript
- Advanced scraper tips and tricks
  - Codementor article

More courses by me, Arianne

Live Trainings

- **Introduction to Python Programming**
  - Beginner
- **Python Environments and Best Practices**
  - Beginner (recommended if you're new to programming)
  - link to next class on Feb 3
- **Object-Oriented Programming in Python**
  - Intermediate (recommended after this class)
  - link to next class on Jan 20

Videos

- **Introduction to Python LiveLessons** - link
  - Beginner
- **Rethinking REST LiveLessons: A hands-on guide to GraphQL** - link
  - Advanced and specific

Thanks!

Questions?

Email me at arianne.dee.studios@gmail.com