get set up for today’s workshop

3. Sit with a Partner
Introduction to Web Scraping with Python

NaLette Brodnax
Indiana University Bloomington
School of Public & Environmental Affairs
Department of Political Science
Center of Excellence for Women in Technology
CEWiT addresses the global need to increase participation of women at all stages of their involvement in technology related fields. Faculty, staff, alumnae and student alliances hold events, host professional seminars, and give IU women opportunities to build a community. We started a conversation about women in technology, ensuring all women have a seat at the table in every technology venture.

CONNECT WITH US

cewit.indiana.edu

iucewit

@IU_CEWiT

iucewit

CONNECT WITH US
next up:

**Python Brown Bags**
Every Tuesday in October at noon

**Introducing to Using APIs with Python**
Friday, November 4<sup>th</sup>
Friday, November 11<sup>th</sup>
my goals

• Get you involved
• Help you scrape some data
your goals

Please take the quick poll at: bit.ly/wimscape930
what is web scraping?

Web scraping is a set of techniques for extracting information from the web and transforming it into structured data that we can store and analyze.
1. static webpage
2. dynamic webpage
3. application programming interface (api)
when should we scrape the web?

Web scraping should help you be more efficient
part 1: getting started
Go to www.python.org
getting the tools

Integrated Development Environment (IDE) → Output

Command Line + Text Editor + Interpreter → Output

Text Editor + Interpreter → Output

Interpreter → Output
installing the tools on your machine

Download Python 3.5
https://www.python.org/downloads/

→ Includes IDLE, an IDE with a text editor and interpreter
→ Includes pip, Python’s standard package manager

Install the necessary libraries **from the command line:**

```
$ pip3 install --upgrade pip
$ pip3 install requests
$ pip3 install beautifulsoup4
```
python development environment

Command Line ➔ Terminal (Bash)/Powershell
- Interact with computer’s operating system
- Manage Python installation
- Access the Python Interpreter
- Execute a program without the Interpreter
Python development environment

IDE ➔ IDLE

Interact with Python

Write programs in a separate screen: **File ➔ New File**
Run the program in the interpreter: **F5 or Run ➔ Run Module**
create a new script

```python
print("Hello, world.")
```

Save the program/script.  
Run the program in the interpreter: **F5** or **Run → Run Module**

**Optional:** Download all the code for today’s workshop  
From the Command Line: 

```bash
git clone https://github.com/nmbrodnax/wim-workshop.git
```
data types: sequences

**String**—ordered sequence of characters

```
‘happy’
```

**List**—ordered sequence of items

```
[‘Leia’, ‘Rey’, ‘Maz’]
```

**Dictionary**—unordered sequence of key-value pairs

```
{‘name’: ‘Kylo’, ‘side’: ‘dark’}
```
functions v. methods

**Function**—named block of code that can accept any number of arguments

```python
my_string = 'aBcDe'
print(my_string)
```

**Method**—a function with a built-in parameter for the object being acted on

```python
print(my_string.lower())
```
def function_name(argument1, argument2, ...):
    first command
    second command
    return output

def say_hello(name_string):
    print('Hello, ' + str(name_string) + '!')
    return None

say_hello('NaLette')
part 2: let’s scrape the web!
the page we will scrape

Please open the following in Firefox or Chrome:
Review • Access • Parse • Transform • store
NAME: Pricing the C's of Diamond Stones
TYPE: Observational Regression Analysis Data
SIZE: 308 observations, 5 variables

The article associated with this dataset appears in the Journal of Statistics Education, Volume 9, Number 2 (July 2001).

SUBMITTED BY:
Singfat Chu
Faculty of Business Administration
National University of Singapore
10 Kent Ridge Crescent
Singapore 119260
jbachucl@nus.sg
JSE Data Archive

4cdata.txt (the basic data file)
4c1data.txt (includes indicator or "dummy" variables)
4c.txt (the documentation file)

NAME: Pricing the C's of Diamond Stones
TYPE: Observational Regression Analysis Data
SIZE: 308 observations, 5 variables
<html>

<!-- Last updated 1/4/13 by JGG-->

<head></head>

<body vlink="blue" link="blue" bgcolor="#FFFFDF" alink="blue">

<table>

<tbody>

<tr></tr>
<tr></tr>
<tr>

<td valign="TOP"></td>
<td valign="TOP">
<table cellpadding="5" bgcolor="navy" align="CENTER"></table>
</td>
</tr>

</tbody>
</table>

<center></center>
<hr />

<a href="v9n2/4cdata.txt">4cdata.txt</a>

(the basic data file)
<br/>

<a href="v9n2/4c1data.txt">4c1data.txt</a>

(includes indicator or "dummy" variables)
html elements

tag begins

\[ <a \ href="v9n2/4cdata.txt">4cdata.txt</a> \]

tag ends

attribute

content

what the browser displays: 4cdata.txt
let’s take a 5-minute break!
Import statements allow you to add functions.

```python
import requests
import bs4
import csv

webpage = 'http://www.amstat.org/...'
server_response = requests.get(webpage)
```

This tells Python to use the get() function from the requests library.
Check every instance of the `<a>` html tag to get the url and filename.

Save the info for each link in its own dictionary inside the list.
what is python doing?

1. Create an empty list: `link_info_list = []`
2. Find all the html chunks with ‘a’ tags: `soup.find_all('a')`
3. Go to the first tag in the list:

```html
<a href="v9n2/4cdata.txt">4cdata.txt</a>
```

4. Assign the url to a variable called `link` and the text to a variable called `name`

5. If the last three letters in the value assigned to the name variable are ‘txt’, proceed. (If not, go to the next tag.)
what is python doing?

6. Save the two variables as values in a dictionary
7. Add the dictionary to the list:

```python
link_info_list = [{
    'link': 'v9n2/4cdata.txt',
    'name': '4cdata.txt'
}]
```

8. Repeat steps 3 through 7 until all tags have been checked
transform

host = 'http://www.amstat.org/publications/jse/'
for dataset in link_info_list[:3]:
    url = host + dataset['link']
    data_response = requests.get(url)
    if data_response.text[:5] == 'NAME:
        dataset['type'] = 'doc'
    else:
        dataset['type'] = 'dat'
what is python doing?

1. Build the address for the link and assign it to the `url` variable:
```
url = 'http://www.amstat.org/publications/jse/v9n2/4cdata.txt'
```

2. Using the requests library, retrieve the web page information
3. If the text on the webpage starts with `NAME:`', add a new key `type` to the link's dictionary with the value `doc`
4. If not, add a new key `type` to the link's dictionary with the value `dat`
def download_to_txt(file_name, data):
    with open(file_name, 'w') as txtfile:
        txtfile.writelines(data)
def strip_extension(file_name):
    i = -1
    while i < -1:
        if file_name[i] == '.':
            break
        else:
            i -= 1  # this is the same as i = i - 1
    return file_name[:i]

Note: We need to do something with the return value, e.g.,

stripped1 = strip_extension('my_file.txt')
for dataset in link_info_list[:3]:
    url = host + dataset['link']
    data_response = requests.get(url)

    description = strip_extension(dataset['name'])
    filename = description + '_' + dataset['type'] + '.txt'

    download_to_text(filename, data_response.text)
with open('data_links.csv', 'w') as csvfile:
    fieldnames = ['link', 'name', 'type']
    writer = csv.DictWriter(csvfile, fieldnames)
    writer.writeheader()
    for link in link_info_list:
        writer.writerow(link)
    print('Links added: ' + str(len(link_info_list)))
run your web scraper!

From IDLE:
Run the program in the interpreter: F5 or Run → Run Module

OR

From the Command Line:
$ python3 wim_web_scraper.py
Review • Access • Parse • Transform • stORe
Thank you!

Please complete the poll at [bit.ly/wimscape930](bit.ly/wimscape930) so we can send you the workshop materials and evaluation email: [nbrodnax@indiana.edu](mailto:nbrodnax@indiana.edu)

linkedin: [nalettebrodnax](https://www.linkedin.com/in/nalettebrodnax)
github: [nmbrodnax](https://github.com/nbrodnax)
twitter: [@nbrodnax](https://twitter.com/nbrodnax)