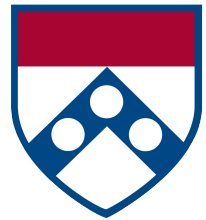
Wharton PhD Tech Camp

Session 4

Alex Miller

Ph.D. Student, Information Systems

Wharton, OID



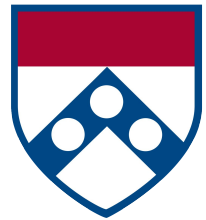
Goal for today

- pip modules in Python
- Primer on data types
- Learn how to use APIs
  - REST APIs
  - Language-specific libraries

Tools to look up on your own time

- Working with matrix-type data in Python:
  - Pandas, Numpy
- Python Notebooks:
  - Jupyter
- Accessing remote files with macOS:
  - Cyberduck
- Recommended plain-text editor:
  - Sublime Text 3
- (Will link when posted)

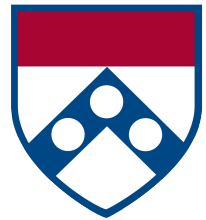
Python Modules



Python Modules

- Also called packages, libraries, gems (esp. in other languages)
- Allows anyone to extend the basic functionality of a language
  - Means they don't come bundled with Python
- pip is the de-facto package installer for Python
  - On Windows: can also use easy_install, but not recommended
  - Whenever you install Python, make sure you install pip!
    - Many IDEs will handle this for you
- Make sure you can install packages on your machine
  - Test with "`pip --version`" from the command line
  - In Thonny, go the the [[[insert]]] menu

Data



Data Collection Workflow

- From unstructured data to structured data

Lorem ipsum dolor
ist segat molur

```
<html>
    <h1>Some
    text</h1>
</html>
```

```
{
  “name”: “Twitter”,
  “date_founded”: 2006,
  “description”: “yadayada”
}
```

	Name	Age	Sex	Score
0	Alex	26	M	4.5
1	Jill	29	F	4.6
2	Max	23	M	3.9

Raw, unstructured information

Semi-structured data

Most structured-data

Data markup you should be familiar with

- XML/HTML (the entire web)
- JSON
  - Probably the most common data-storage/transfer formats
- CSV
  - Typically row/column based
  - Lots of other similar formats (fixed width, .tsv, etc.)

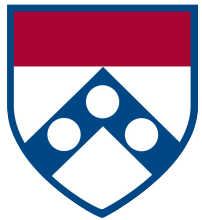
- Formats I recommend you not use:
  - Proprietary ones
  - .xlsx, .dta, pickle, etc.

JSON

- Simply document with a set of keys and values
- More flexible than row/value (i.e., matrix) data structure
- Same variable types/formatting as Javascript

```
{
   “name”: “Alex”,
   “age”: 26,
   “department”: “OID”,
   “interests”: [“hiking”, “astronomy”, “the cyber”]
}
```

The Internet, in Brief



How the internet works, in brief

- Computers transferring data through wires and radio waves
- Note that the internet is not the web
  - The web is all the stuff you can see, connected by hyperlinks
  - The internet is the infrastructure that allows two computers to communicate to each other

Why this matters

- Most computers and programs talk to each other using APIs
  - API = Application Programming Interface
- APIs are the *intended* way for computers to communicate
  - The web = for humans
  - APIs = for programs
- Alex’s first rule of web scraping:
  - Avoid web scraping
  - Scraping is a last resort
  - Use APIs if they exist!

Data collection checklist

- If you want some data from somewhere:
  - Ask your colleagues
  - Do a Google search
    - Use the word “dataset” in your query
    - data.world is interesting
  - Ask the library
    - Check for *databases*
  - Check for language-specific API
  - Check for REST API
  - Scrape

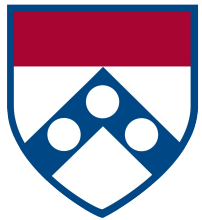
Types of API

- REST API
  - Can be accessed using any HTTP-driven request mechanism
  - Doesn’t matter what language you are using
- Language-specific APIs
  - Packages/libraries/modules that are desinged to make it easy to get data from a particular source into your programming environment
    - Twitter - Python
    - Google Trends - R
  - Either built by data source (if well funded) or hacked together by fellow researchers

Example Python API

- Clearbit
  - Company information database/service
- Note that under the hood is just a basic REST API

The Nitty Gritty



Web Requests

- Any HTTP client makes what are called “requests” when it wants to fetch any data/webpage
- There are two important types of HTTP requests
- GET
  - URL, Headers
- POST
  - URL, Headers, Body/Payload
- Other types:
  - HEAD, OPTIONS, PUT

Server Response

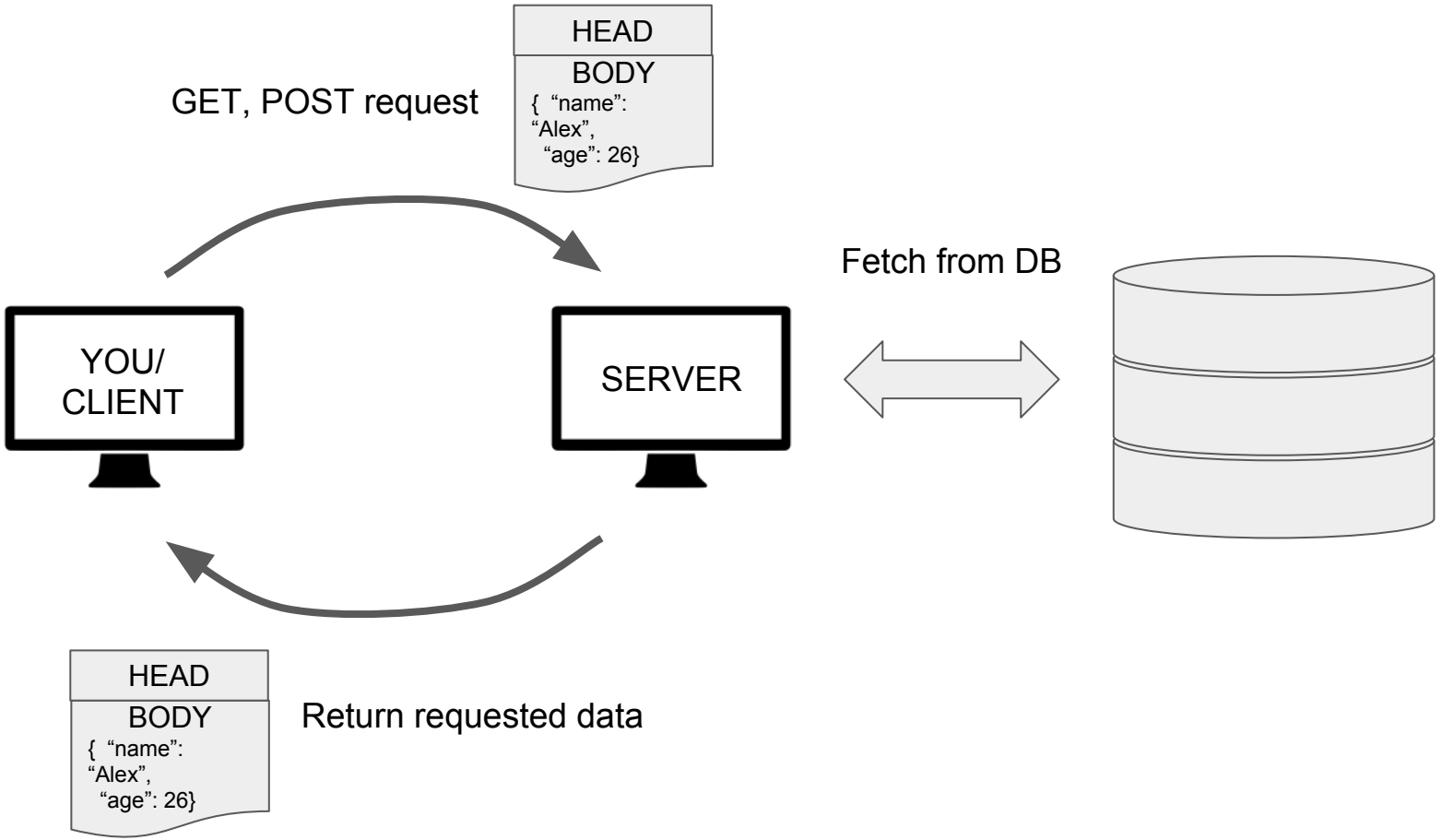
- When a server (i.e., someone else’s computer) receives an HTTP request, if it is properly configured, it will return a request object with its own:
  - Status Code
  - Header
  - Body

Status Codes

- 200: All Clear!
  - Other 2xx are usually fine too
- 3xx: Redirection
- 4xx: You messed up
  - 404: File Not Found
  - 403: Unauthorized (need to authenticate somehow)
- 5xx: They messed up
  - 503: Server totally failed

- Consult Google if you get a code that you don’t understand
  - “420 HTTP response code”

What's Going On



Determine the API's Query Format

- You need to know how to communicate with a particular REST APIs
- Basic GET REST API example
  - https://en.wikipedia.org/api/rest_v1/page/summary/Barack_Obama
- URL parameters:
  - http://domain.com/?key1=value1&key2=value2
  - REST API with URL parameters:
    - http://samples.openweathermap.org/data/2.5/forecast/daily?q=London,UK&appid=0521d43e1c977533f9492c26987e620c

POST Requests

- You must use a programmatic interface to make POST requests
- Main use is the POST request can have its own body that contains data to be processed by the server
- Most APIs you come across will likely be based on GET protocol, but don’t be confused when you see documentation mentioning POST

- In Python (pretty easy):
  - r = requests.post(rl, payload=data, headers=headers)

Demonstration

- Python
  - Primary mechanism for making HTTP requests is the `requests` module (v > 3.x.x)
  - Go over basic example
- Exercise
  - https://github.com/alexmill/techcamp_week1/blob/master/session4.md