[301] Web 1

Tyler Caraza-Harter

Learning Objectives Today

Network basics

- IP addresses
- host/domain names
- client/server and request/response

HTTP basics

- URLs
- GET/POST/etc
- headers
- status codes

Requests modules

- downloading data with requests.get
- remote calls with requests.post

Learning Objectives Today

Motivation

Networking Basics

HTTP (Hypertext Transfer Protocol)

Requests Module

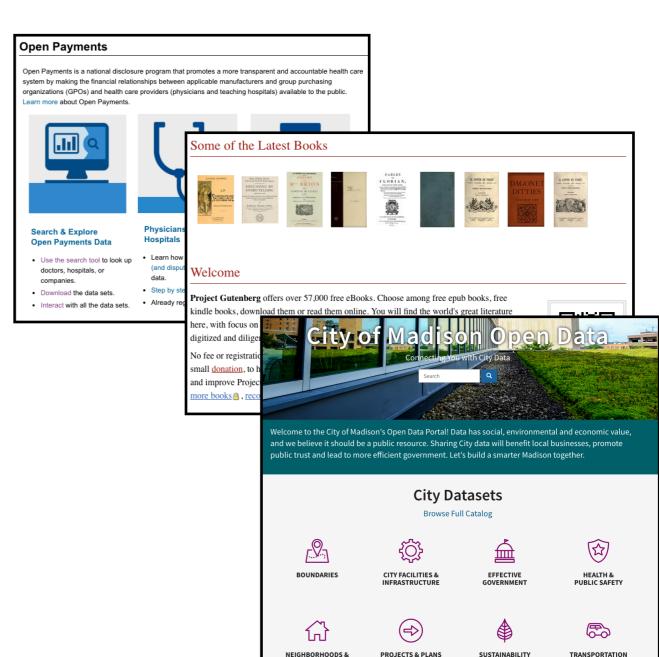
Data Science and the Internet

There are tons of online sources of data

Examples: https://tyler.caraza-harter.com/cs301/fall18/datasets.html

Wide range of topics

- healthcare
- roads and city planning
- astronomy
- population
- business
- entertainment
- education
- etc



Data Science and the Internet

There are tons of online sources of data

Examples: https://tyler.caraza-harter.com/cs301/fall18/datasets.html

Wide range of topics

- healthcare
- roads and city planning
- astronomy
- population
- business
- entertainment
- education
- etc

Open Payments Open Payments is a national disclosure program that promotes a more transparent and accountable health ca system by making the financial relationships between applicable manufacturers and group purchasing Some of the Latest Books Search & Explore **Open Payments Data** Use the search tool to look up doctors, hospitals, or Welcome companies. Step by Project Gutenberg offers over 57,000 free eBooks. Choose among free epub books, free and improve Proje nore books A. r and we believe it should be a public resource. Sharing City data will benefit local businesses, promote **City Datasets** Browse Full Catalog 〇 BOUNDARIES CITY FACILITIES & INFRASTRUCTURE EFFECTIVE HEAITH & (➪)

SUSTAINABILITY

Why not just download data by hand?

Motivation 1: too much data

What if you're analyzing language trends over time?

- Dataset: Project Gutenberg has 57K free books
- Too much work to download one by one



Welcome

Project Gutenberg offers over 57,000 free eBooks. Choose among free epub books, free kindle books, download them or read them online. You will find the world's great literature here, with focus on older works for which copyright has expired. Thousands of volunteers digitized and diligently proofread the eBooks, for enjoyment and education.

No fee or registration is required. If you find Project Gutenberg useful, please consider a small <u>donation</u>, to help Project Gutenberg digitize more books, maintain our online presence, and improve Project Gutenberg programs and offerings. Other ways to help include <u>digitizing</u> more books A, recording audio books A, or reporting errors.

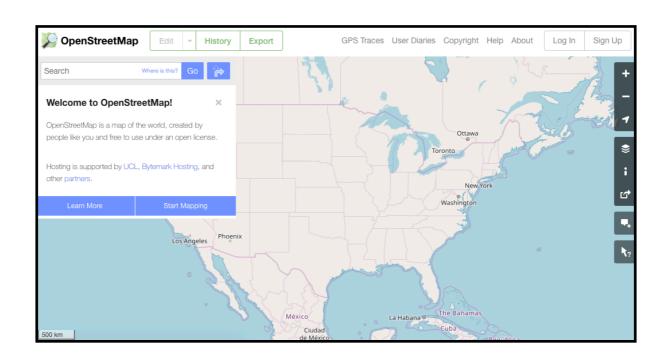


Motivation 2: data doesn't always come in files

Many datasets are difficult to download complete

Instead, you can make function calls to servers (we'll learn how) to grab specific data

- Dataset: OpenStreetMap
- You issue calls to get specific data:
 - 1. specify latitude/longitude rectangle
 - 2. specify structures of interest (e.g., bike paths)



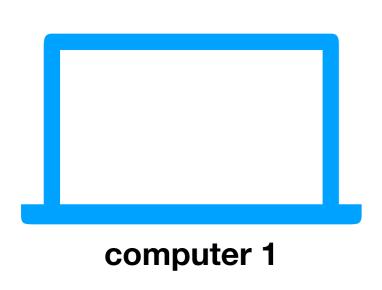
Learning Objectives Today

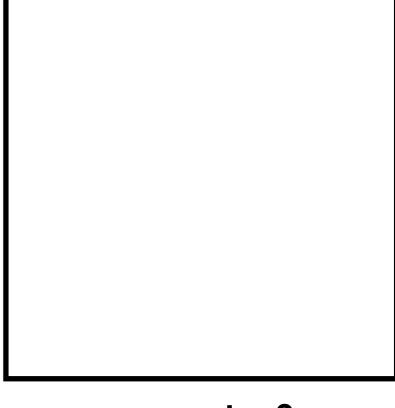
Motivation

Networking Basics

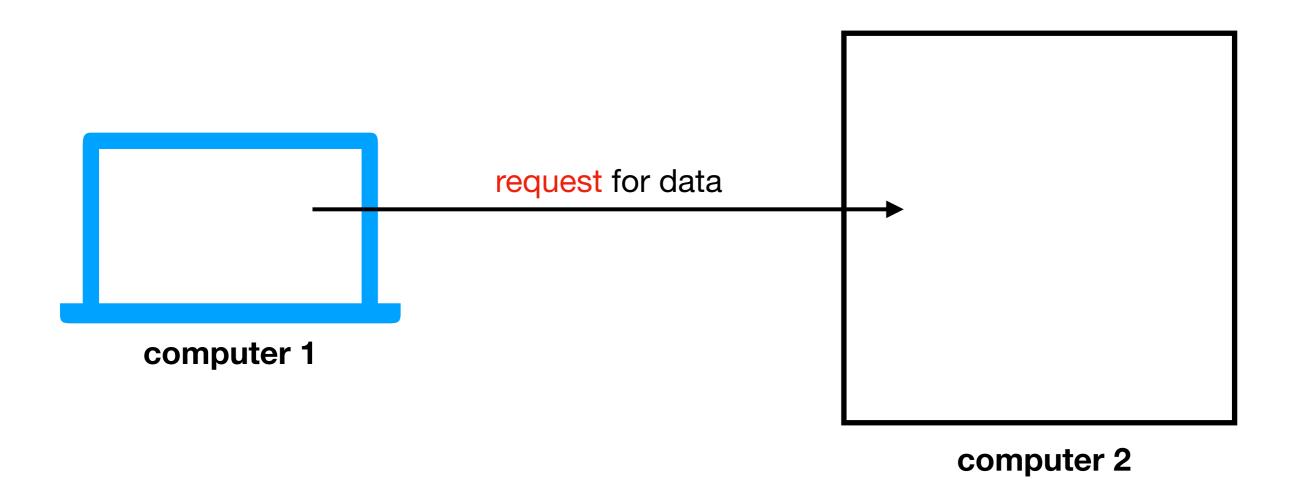
HTTP (Hypertext Transfer Protocol)

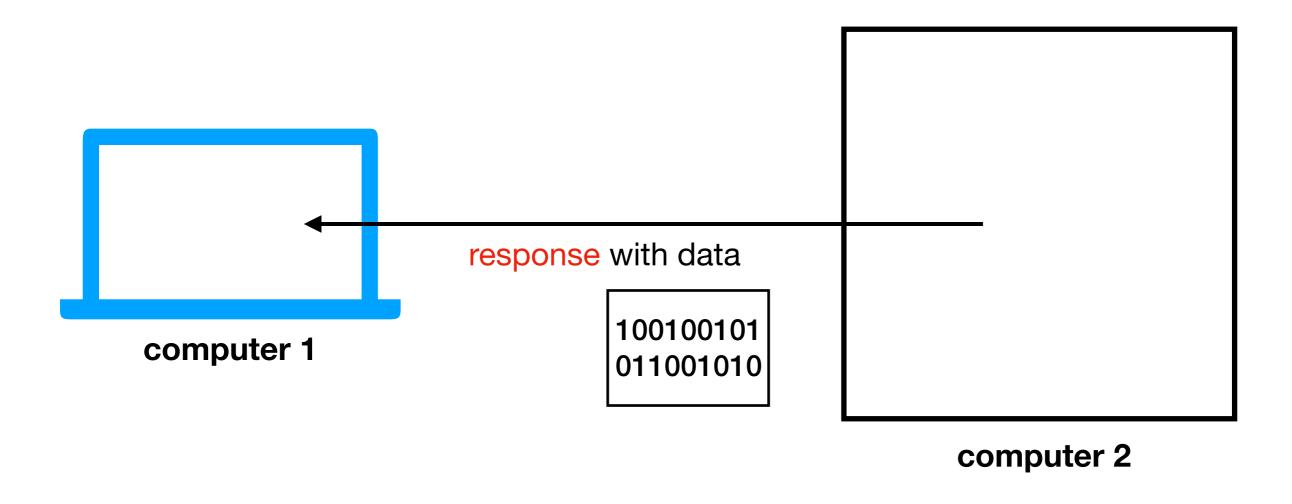
Requests Module

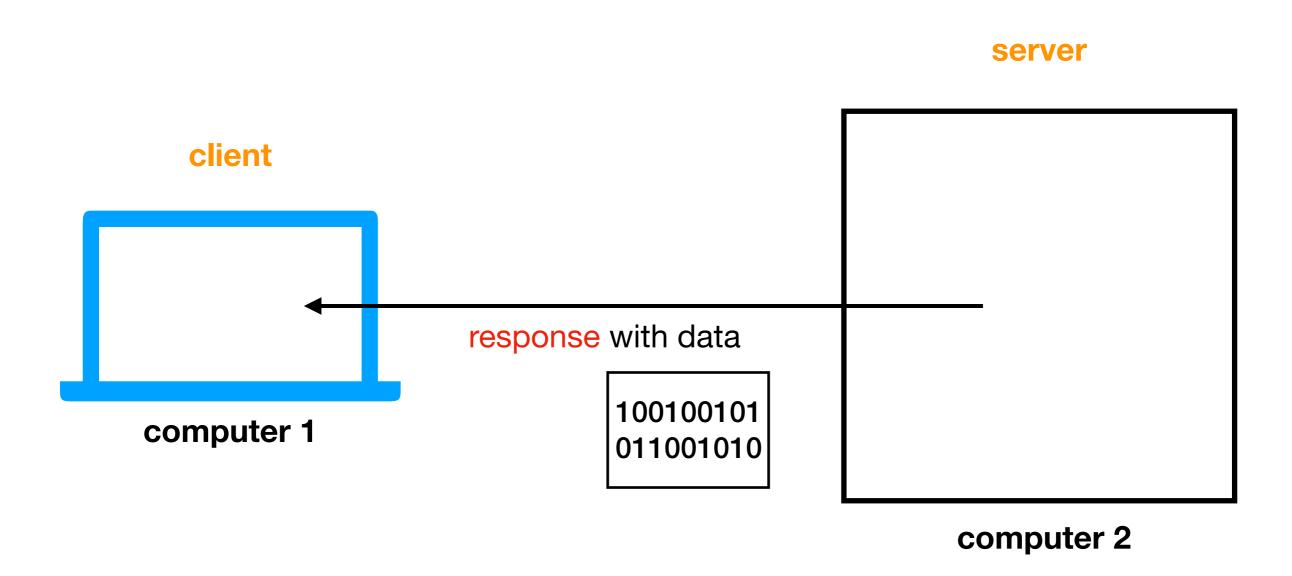


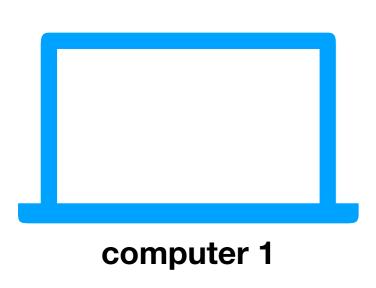


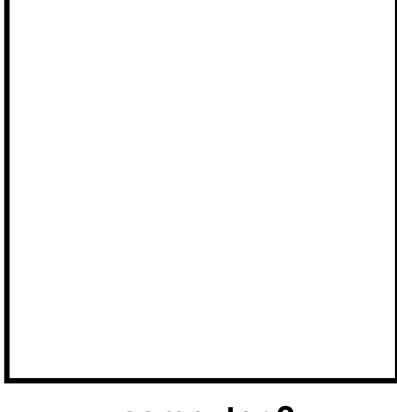
computer 2







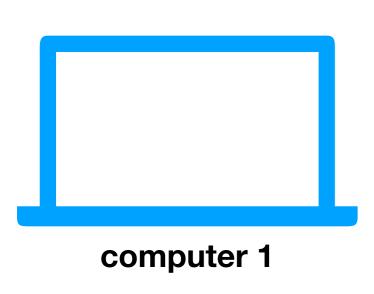


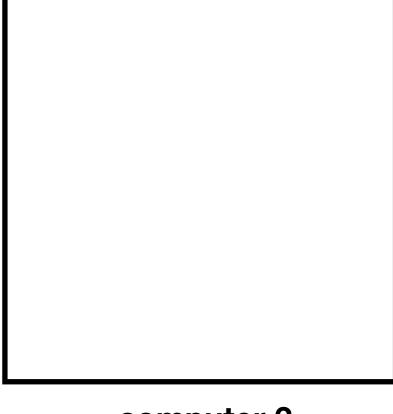


computer 2

Challenge: there are millions of computers.

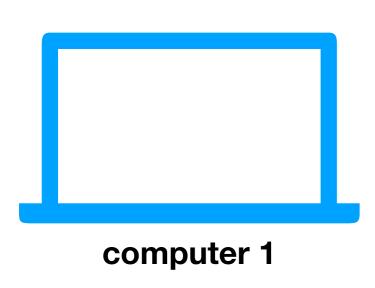
How do we indicate which machine should get our request?

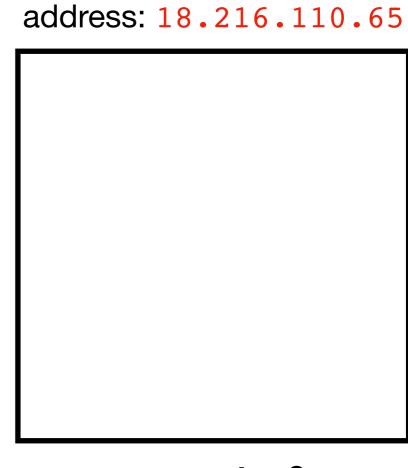




computer 2

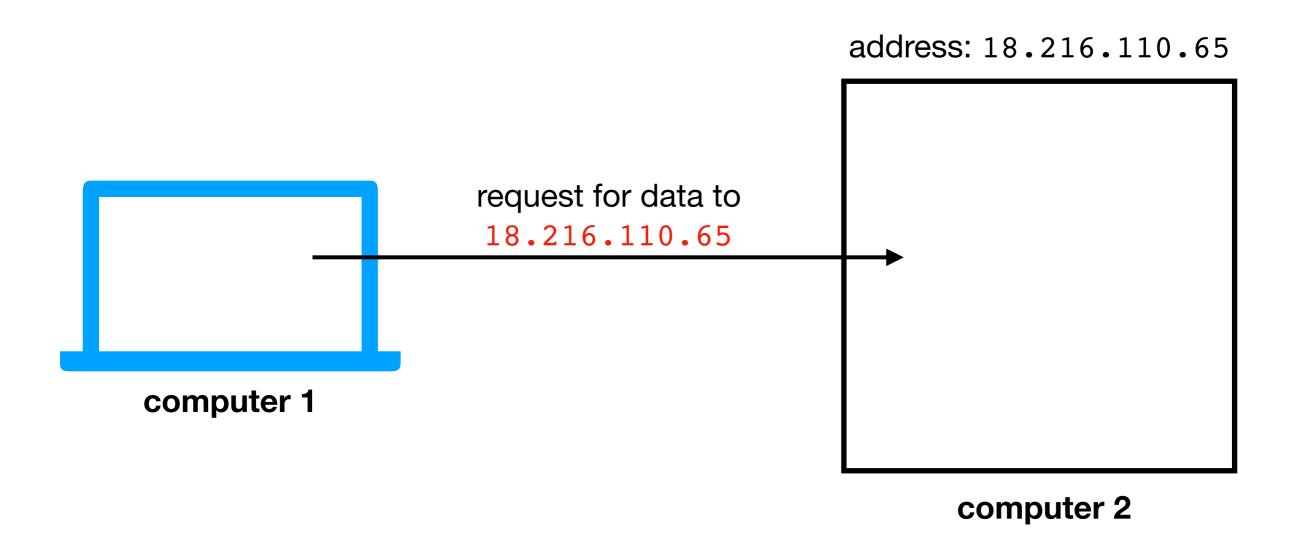
Solution: every machine* has an IP address (Internet Protocol). Requests are sent to a specific IP address.



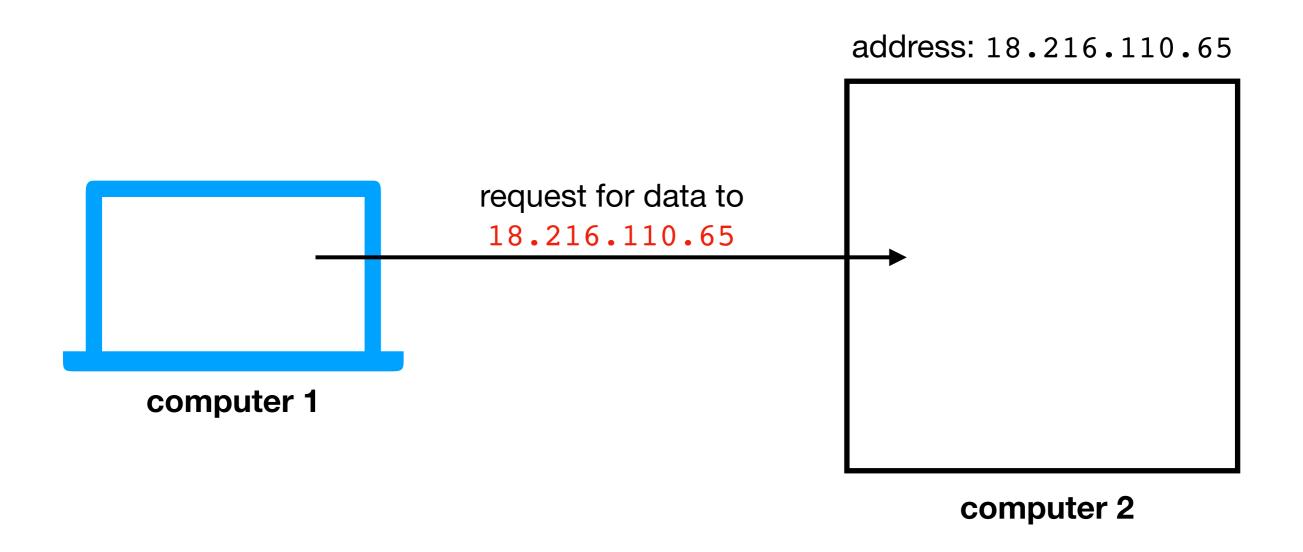


computer 2

Solution: every machine* has an IP address (Internet Protocol). Requests are sent to a specific IP address.



Solution: every machine* has an IP address (Internet Protocol). Requests are sent to a specific IP address.



Challenge: it's hard to remember IP addresses.

Imagine you had to type a number instead of www.google.com!

Domain Names

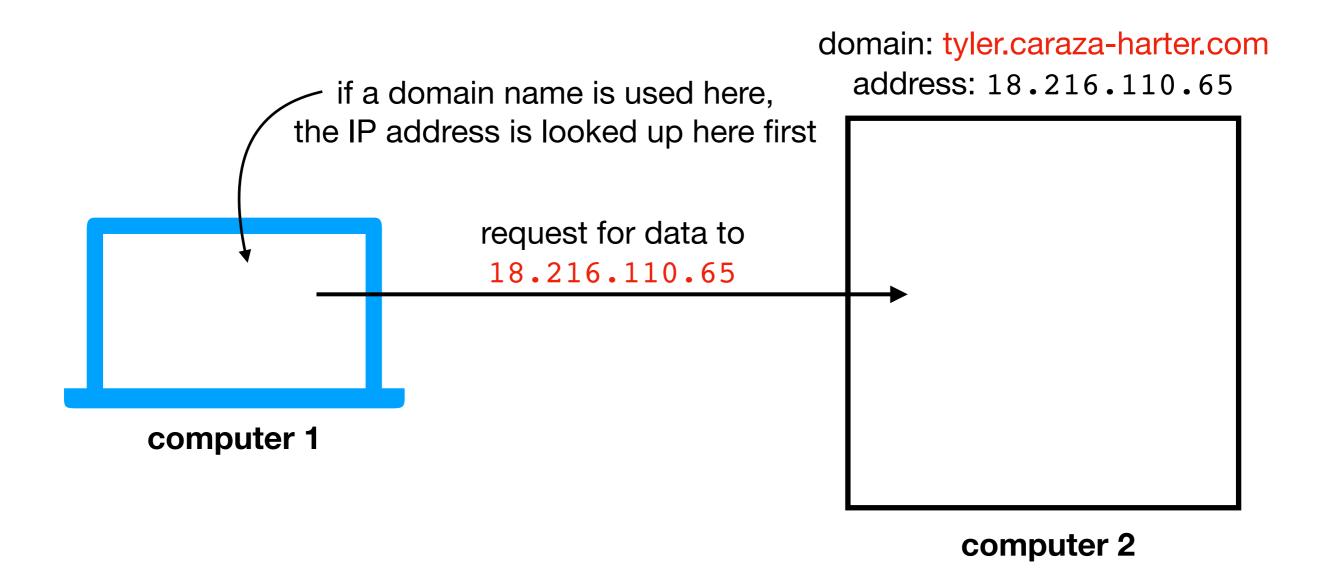
request for data to
18.216.110.65

computer 1

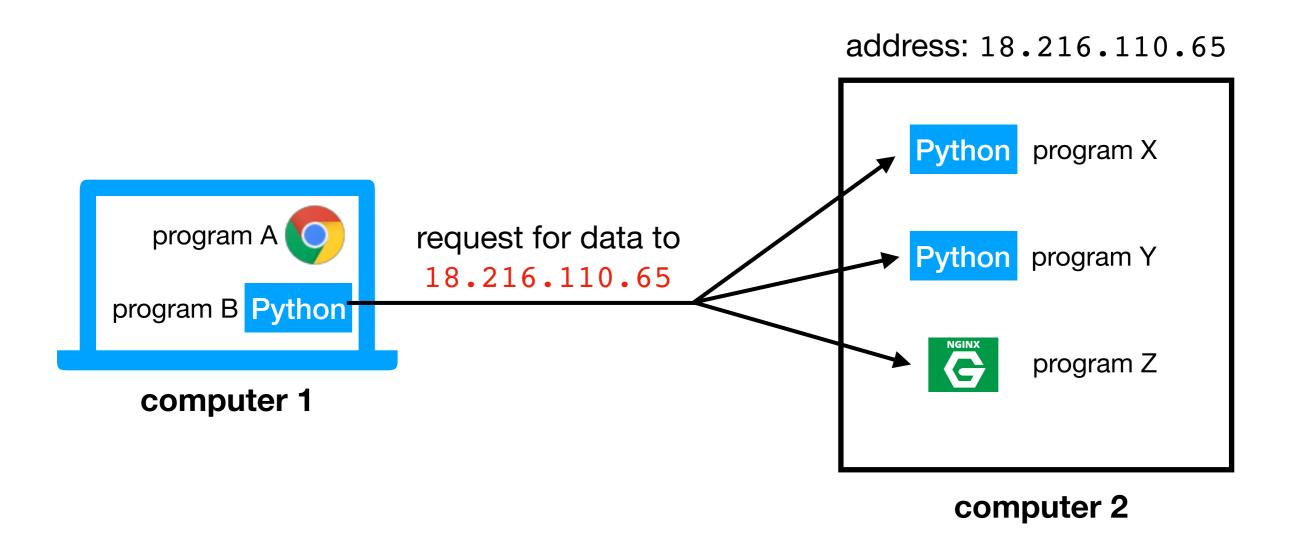
Solution: use "nicknames" (called domain names) for IP addresses of machines that serve data

computer 2

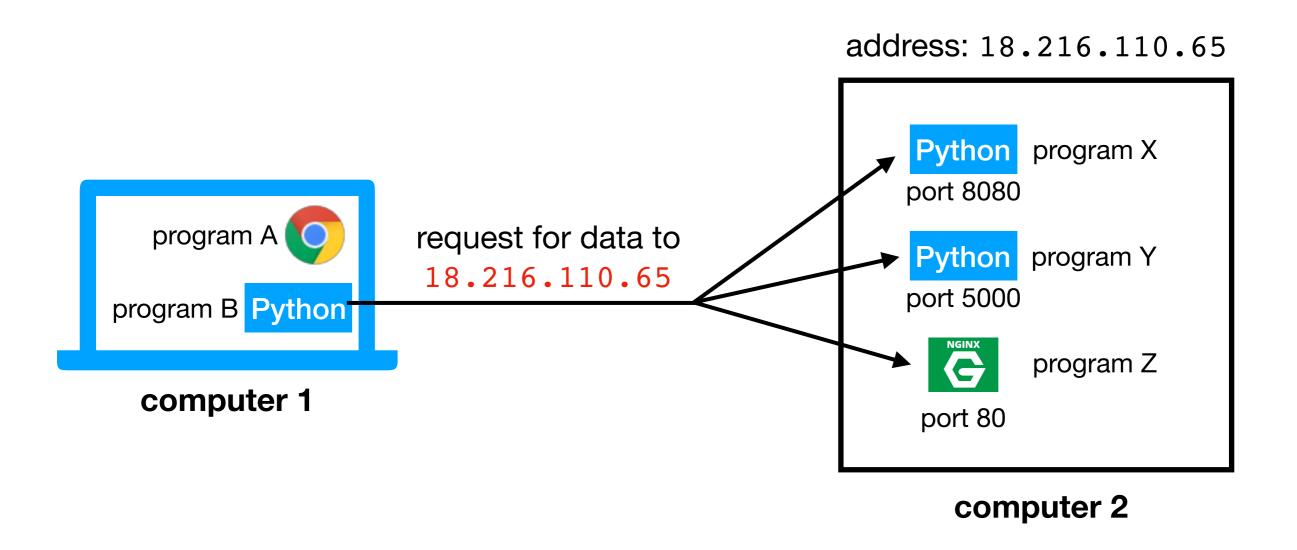
Domain Names



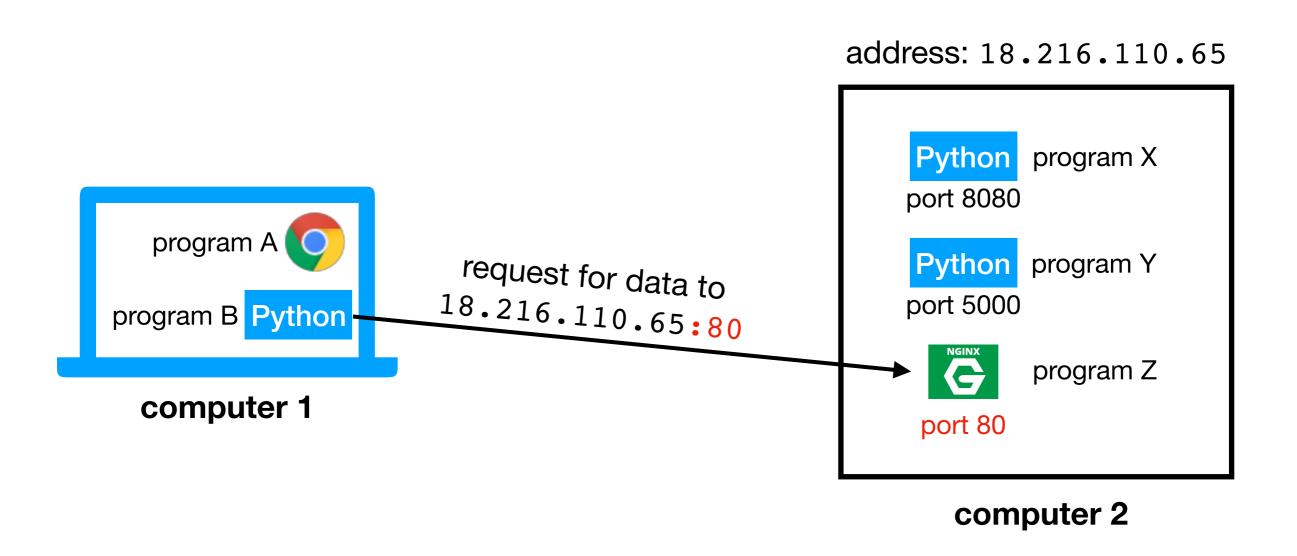
Solution: use "nicknames" (called domain names) for IP addresses of machines that serve data



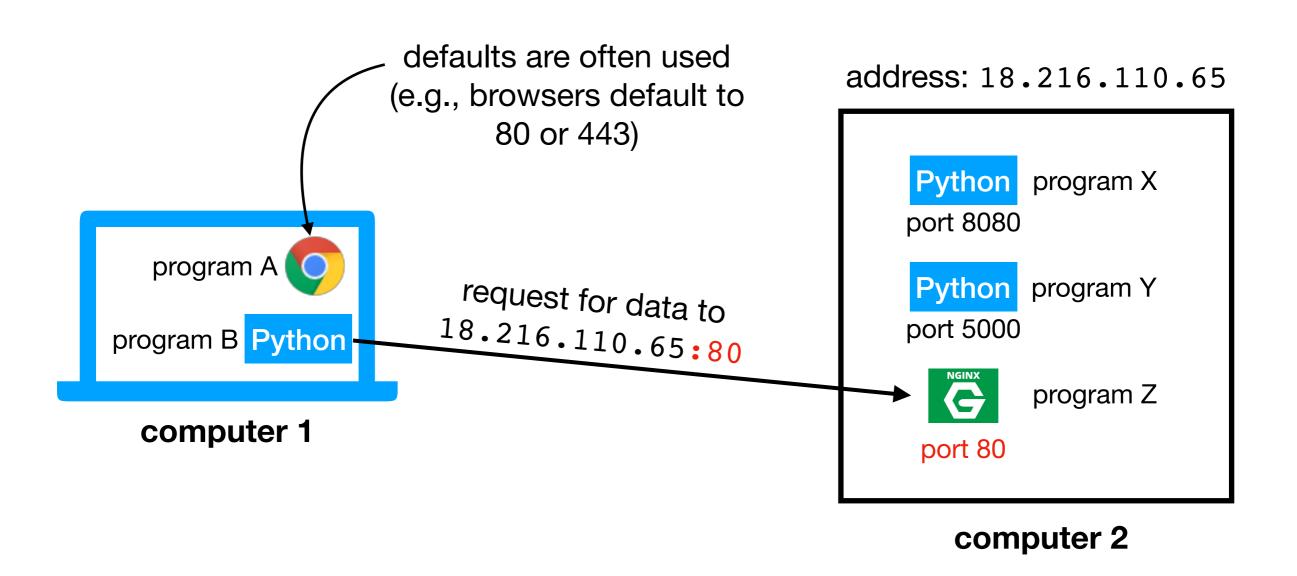
Challenge: there may be multiple programs running on each computer. How do we get the messages to the right program?



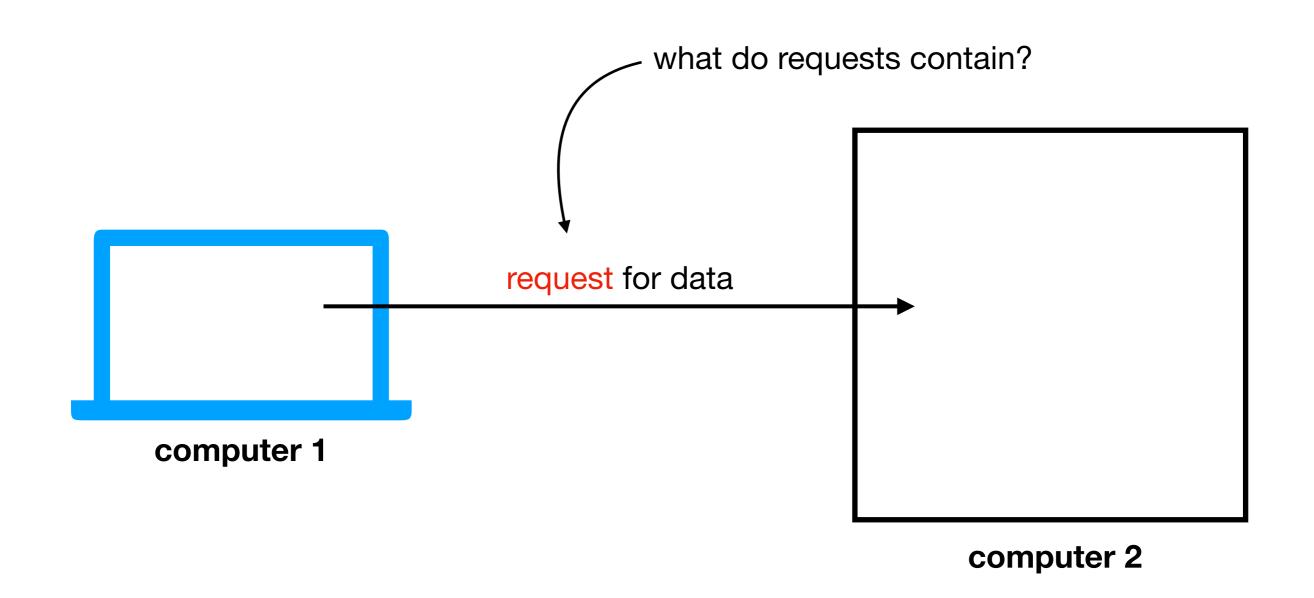
Solution: give each program a unique ID (called a "port number")

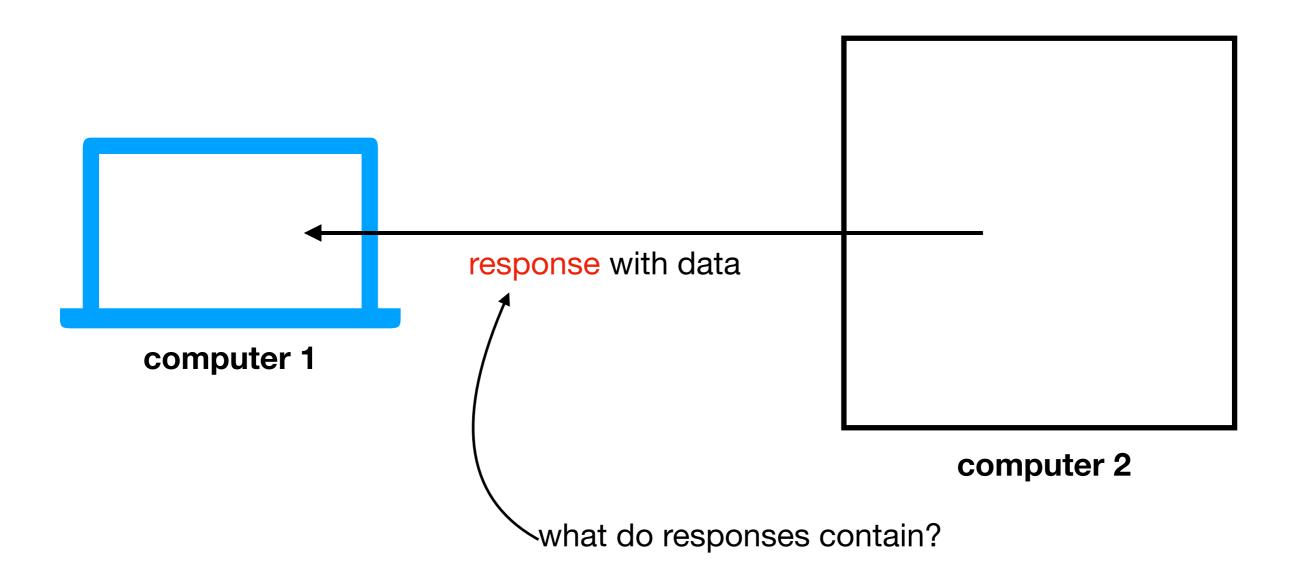


Solution: specify port number in request

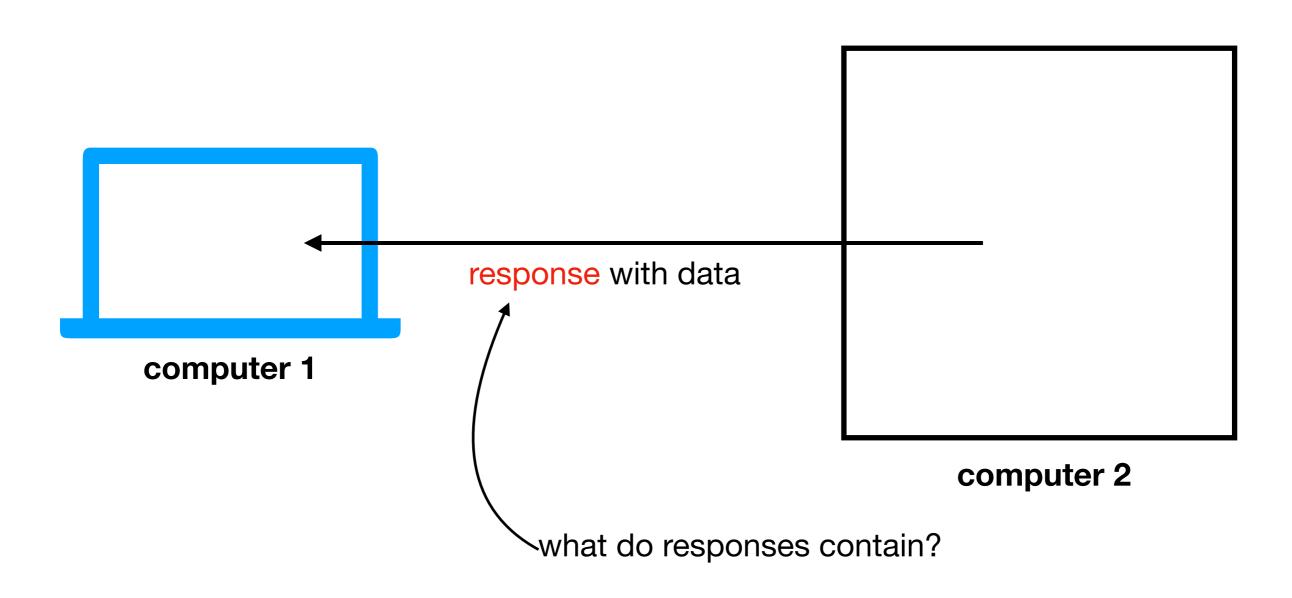


Solution: specify port number in request





depends on application! (video chat, web browsing, etc)
we'll only consider web applications for this semester



Learning Objectives Today

Motivation

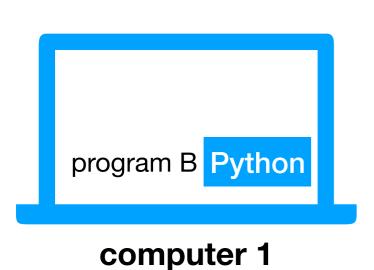
Networking Basics

HTTP (Hypertext Transfer Protocol)

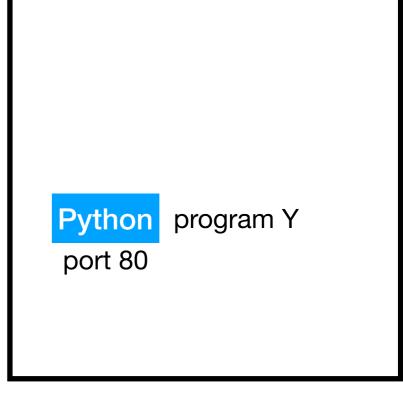
Requests Module

Protocol for communicating web data

• downloading a specific webpage, image, etc



domain: <u>example.com</u> address: 12.34.56.78



computer 2

Note: we won't talk about HTTPS today, which is HTTP with encryption

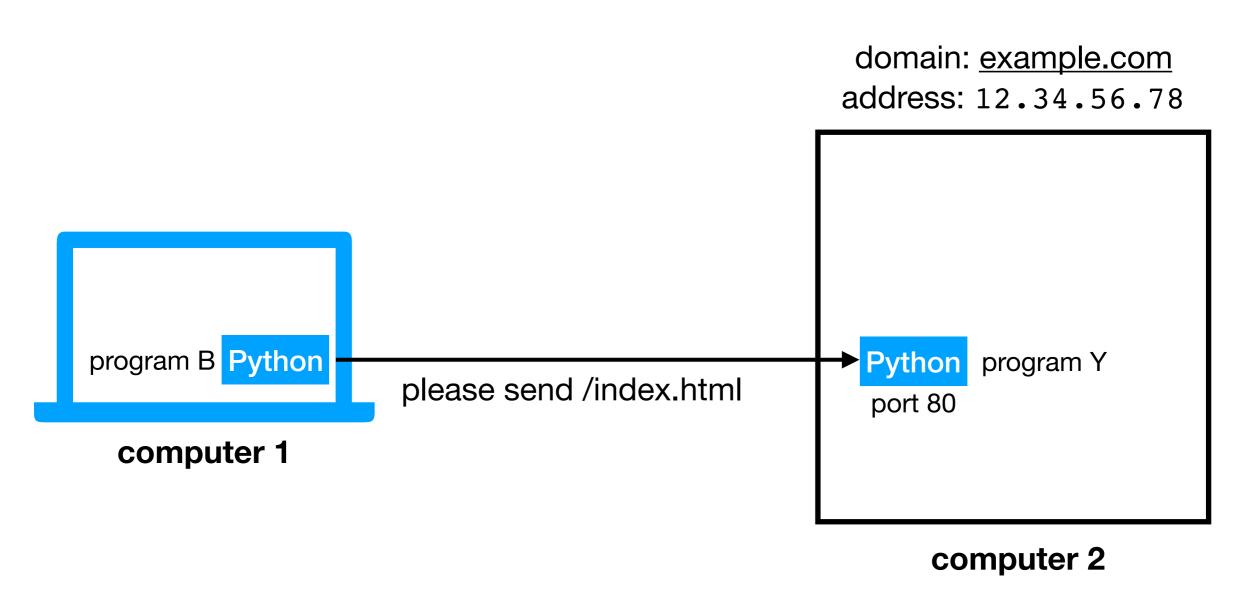
Protocol for communicating web data

downloading a specific webpage, image, etc

domain: example.com address: 12.34.56.78 program B Python Python program Y please send home page port 80 computer 1 computer 2

Protocol for communicating web data

downloading a specific webpage, image, etc



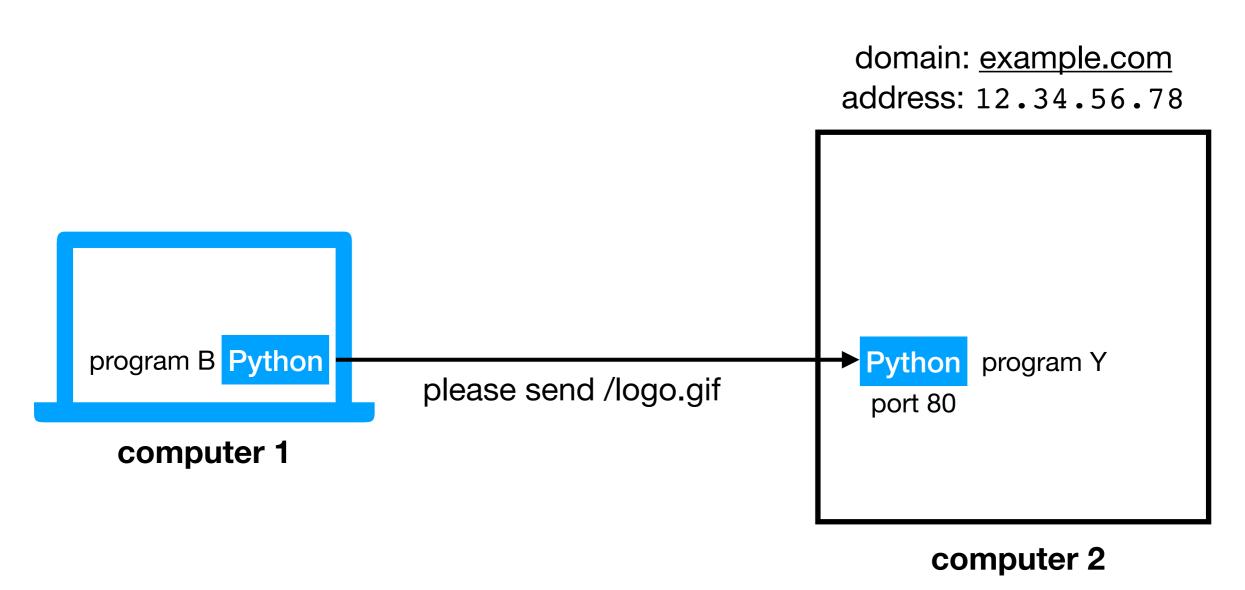
Protocol for communicating web data

downloading a specific webpage, image, etc

domain: example.com address: 12.34.56.78 program B Python Python program Y please send /about.html port 80 computer 1 computer 2

Protocol for communicating web data

downloading a specific webpage, image, etc



Protocol for communicating web data

downloading a specific webpage, image, etc

domain: <u>example.com</u> address: 12.34.56.78



Note we need three things:

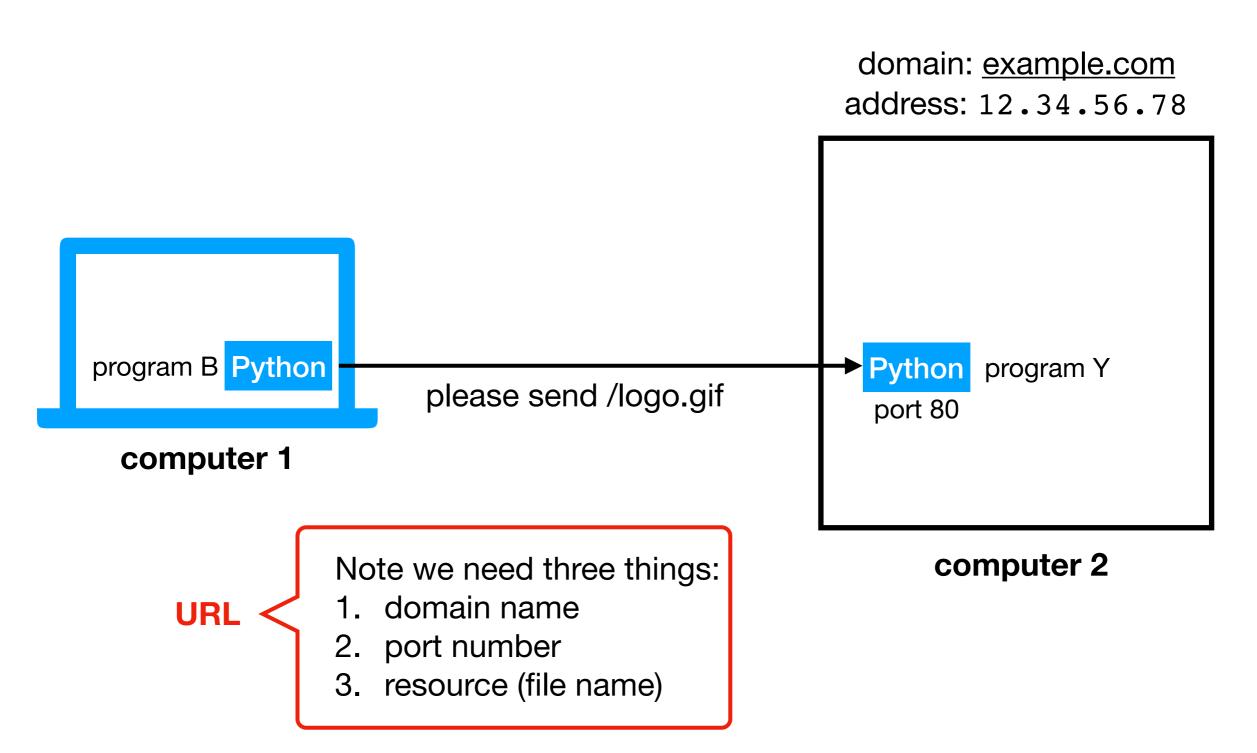
- 1. domain name
- 2. port number
- 3. resource (file name)

computer 2

HTTP

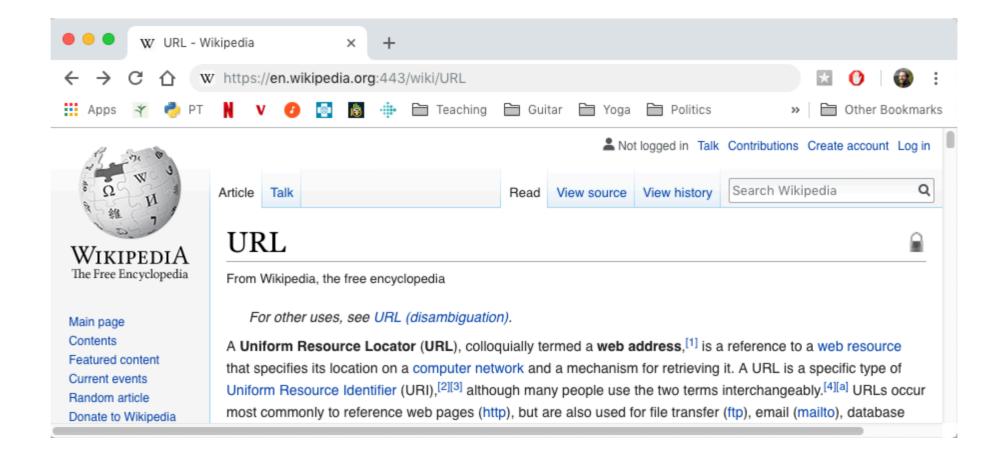
Protocol for communicating web data

downloading a specific webpage, image, etc



URL

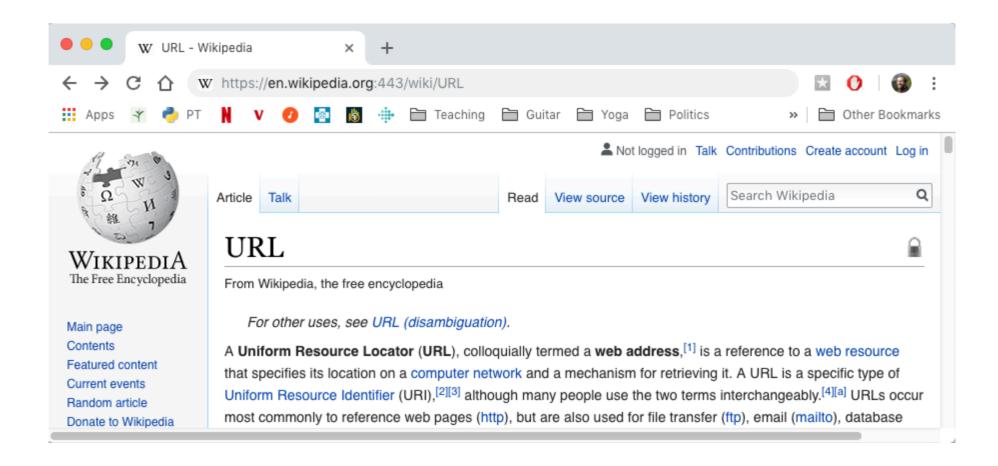
https://en.wikipedia.org:443/wiki/URL



- domain name
- 2. port number
- 3. resource (file name)

domain name

https://en.wikipedia.org:443/wiki/URL

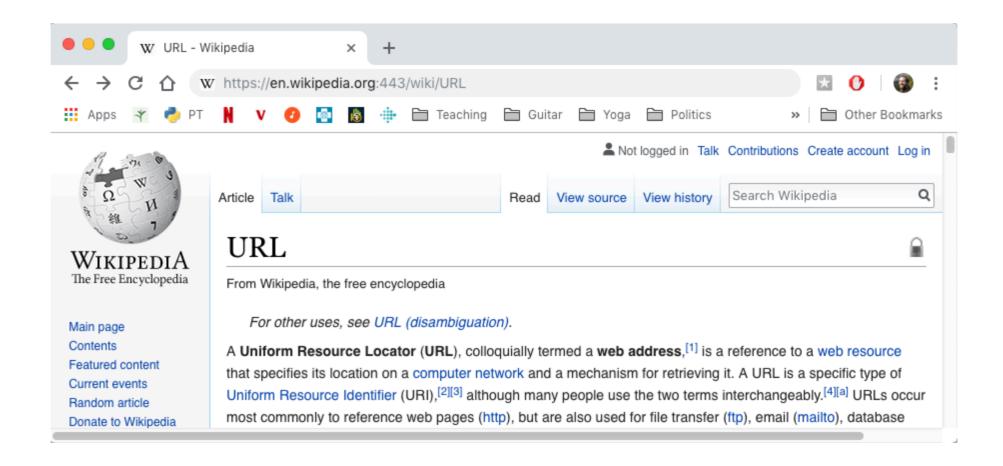


URL <

- domain name
- 2. port number
- 3. resource (file name)

domain name

https://en.wikipedia.org:443/wiki/URL port



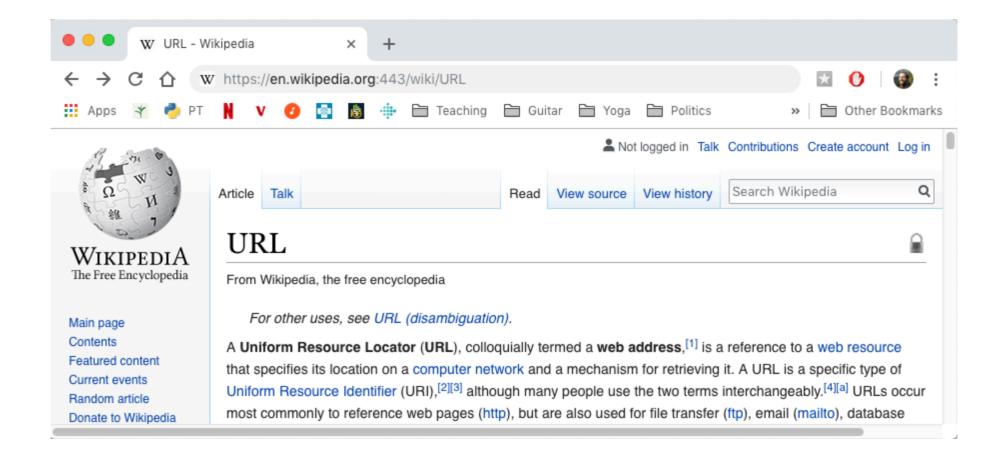
URL <

- domain name
- 2. port number
- 3. resource (file name)

domain name

resource

https://en.wikipedia.org:443/wiki/URL port



URL

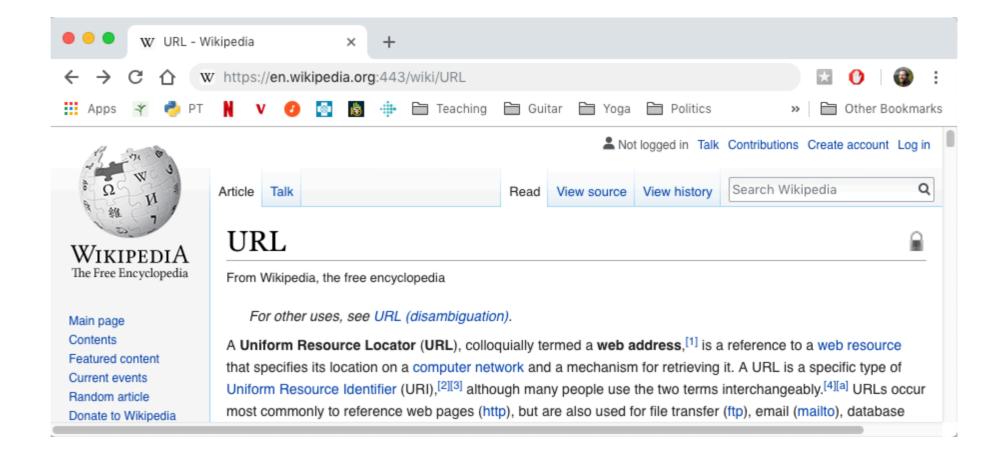
- domain name
- 2. port number
- 3. resource (file name)

domain name

resource

https://en.wikipedia.orc/wiki/URL

port would have defaulted to 443 if not specified



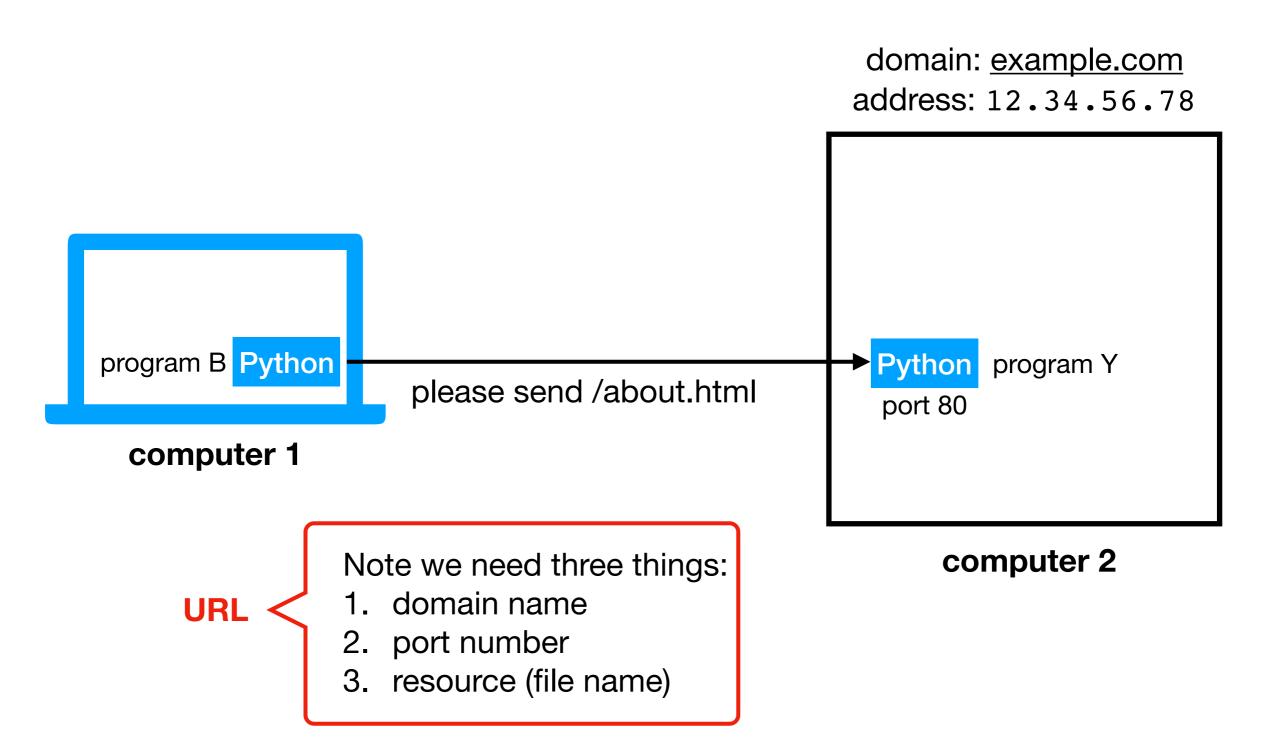
URL

- domain name
- 2. port number
- 3. resource (file name)

HTTP

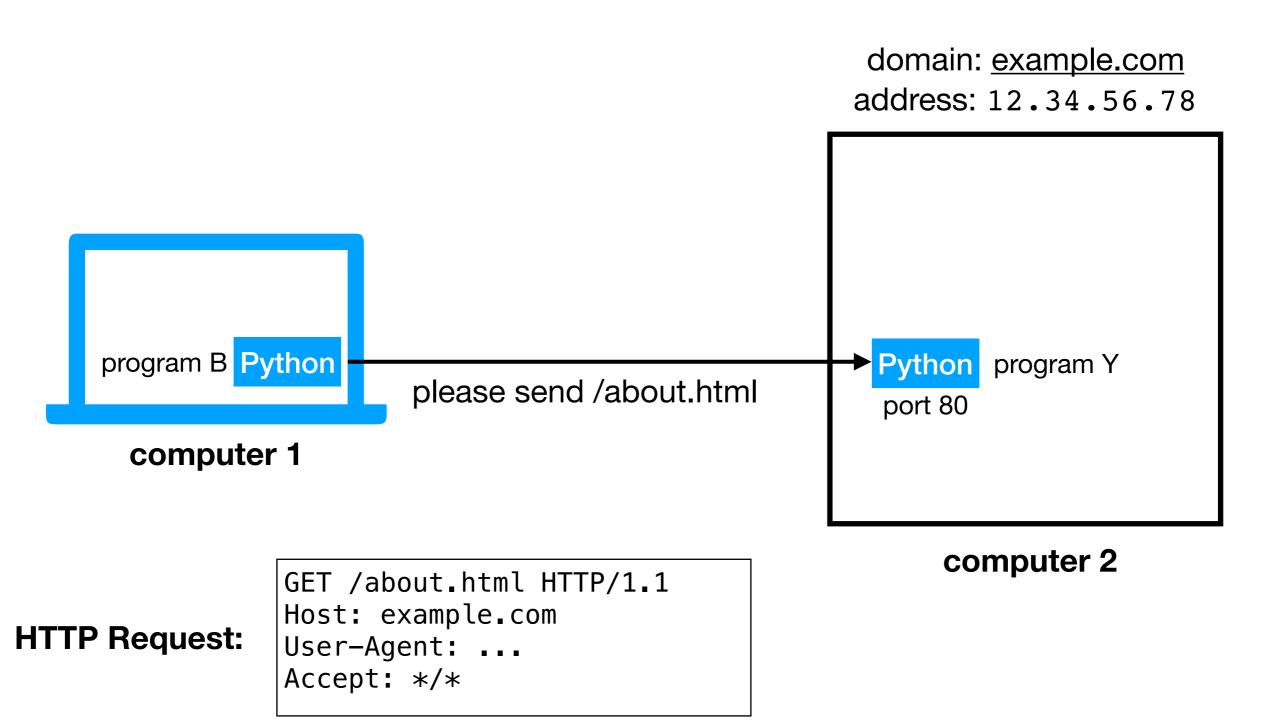
Protocol for communicating web data

• downloading a specific webpage, image, etc



Protocol for communicating web data

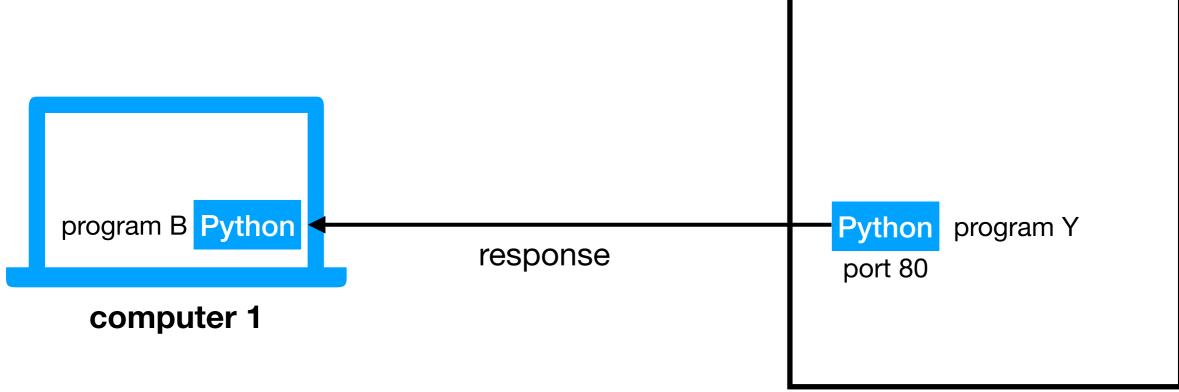
downloading a specific webpage, image, etc



Protocol for communicating web data

• downloading a specific webpage, image, etc

domain: <u>example.com</u> address: 12.34.56.78



HTTP Response:

HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 74
Server: Werkzeug/0.14.1 Python/3.6.6
Date: Sun, 11 Nov 2018 17:00:29 GMT
all the contents

computer 2

Request and Response

we want the about.html page

```
HTTP Request: GET /about.html HTTP/1.1
Host: example.com
User-Agent: ...
Accept: */*
```

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 74
Server: Werkzeug/0.14.1 Python/3.6.6
Date: Sun, 11 Nov 2018 17:00:29 GMT

data in about.html

all the contents
```

There are **LOTS** of details here we don't care about right now

Request and Response

we want the about.html page GET /about.html HTTP/1.1 Host: example.com **HTTP Request:** User-Agent: ... Accept: */* status code. 200 is good. 404, 500, others are various errors or other more complicated states HTTP/1.0 200 OK Content-Type: text/html; charset=utf-8 Content-Length: 74 Server: Werkzeug/0.14.1 Python/3.6.6 **HTTP Response:** Date: Sun, 11 Nov 2018 17:00:29 GMT all the contents data in about.html

There are **LOTS** of details here we don't care about right now

method. GET is simple download. POST means we are uploading data as part of our request. We we want the about.html page wont talk about others today. GET /about.html HTTP/1.1 Host: example.com **HTTP Request:** User-Agent: ... Accept: */* status code. 200 is good. 404, 500, others are various errors or other more complicated states HTTP/1.0 200 OK Content-Type: text/html; charset=utf-8 Content-Length: 74 Server: Werkzeug/0.14.1 Python/3.6.6 **HTTP Response:** Date: Sun, 11 Nov 2018 17:00:29 GMT all the contents data in about.html

There are **LOTS** of details here we don't care about right now

Learning Objectives Today

Motivation

Networking Basics

HTTP (Hypertext Transfer Protocol)

Requests Module