GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/JS-SF-10-resources` repo to your computer
2. Open the `10-ajax-apis/starter-code` folder in your code editor

JAVASCRIPT DEVELOPMENT

AJAX & APIS

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Identify all the HTTP verbs & their uses.
- Describe APIs and how to make calls and consume API data.
- Access public APIs and get information back.
- Implement an Ajax request with vanilla JS.
- Create an Ajax request using jQuery.
- Reiterate the benefits of separation of concerns – API vs. Client.

AGENDA

- APIs
- HTTP
- Ajax using Fetch
- Ajax & jQuery
- Separation of concerns

WEEKLY OVERVIEW

WEEK 6	Ajax & APIs / Asynchronous JavaScript & Callbacks
WEEK 7	Advanced APIs / Project 2 Lab
WEEK 8	Closures & the module pattern / CRUD & Firebase

EXIT TICKET QUESTIONS

1. I don't think I could have actually written any of those examples by myself if I was starting from scratch. I think I could use some practice starting with nothing and having to build up the HTML, CSS, and JS.
2. What is the use of template literal?
3. Should I focus on vanilla JS or jQuery for DOM manipulation?

AJAX & APIS

ACTIVITY

EXERCISE

TYPE OF EXERCISE

- Individual/Partner

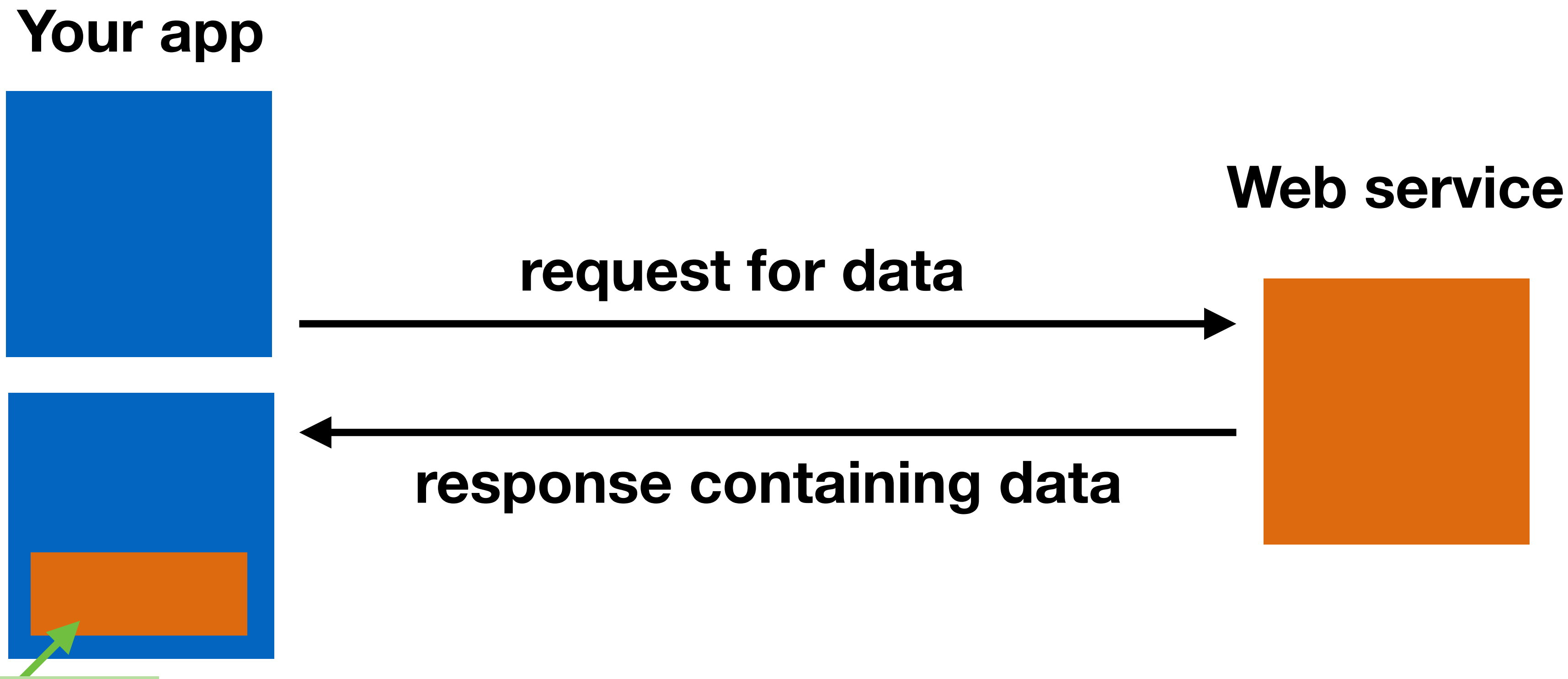
TIMING

3 min

1. Think about how you could use one or more sources of web data in an app.
2. Write a description or sketch a schematic of your app on your desk.

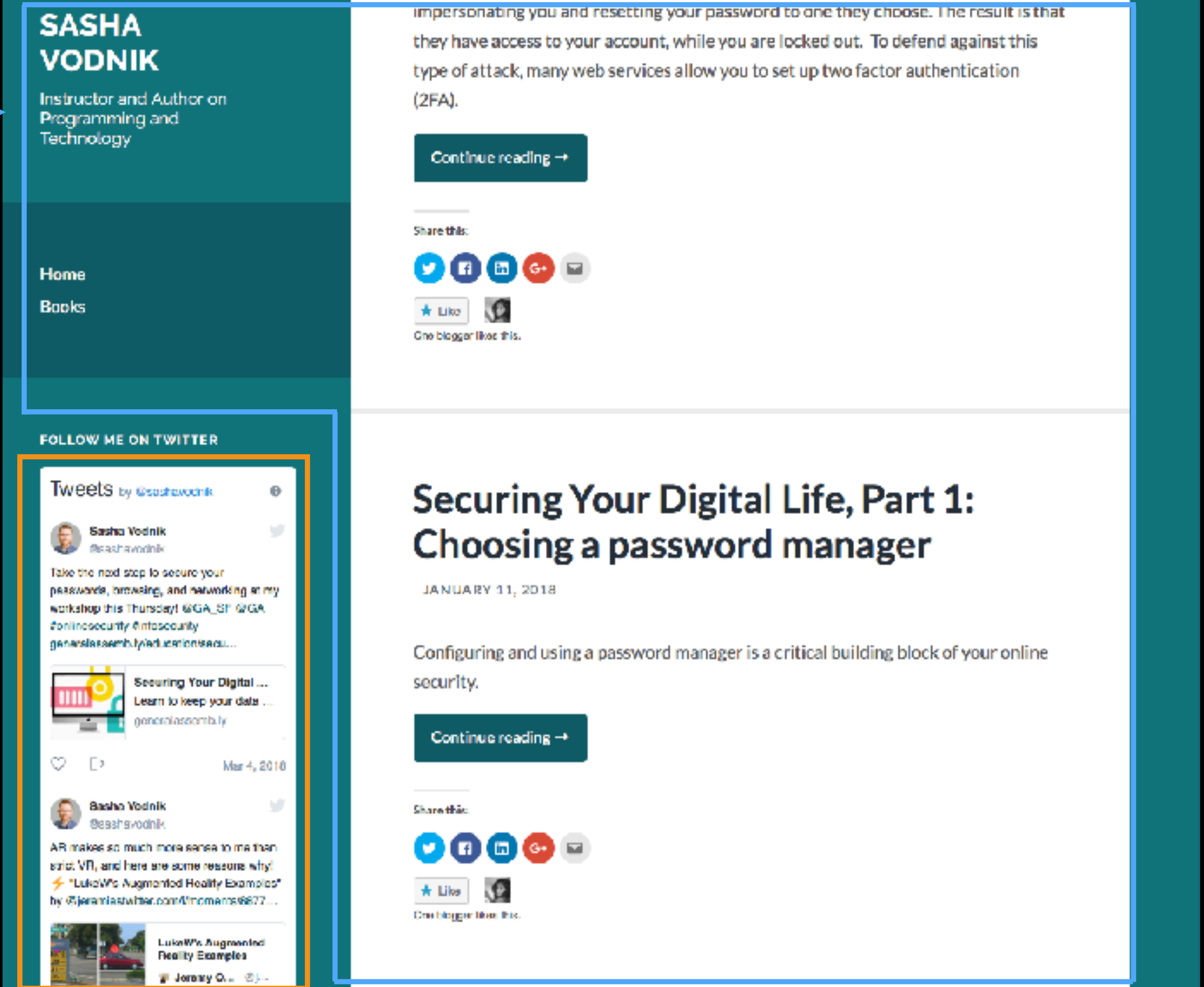
APIs

WEB SERVICES



your app can now incorporate data from the web service my website content

Content from Twitter added using Twitter API



SASHA VODNIK

Instructor and Author on Programming and Technology

Home

Books

impersonating you and resetting your password to one they choose. The result is that they have access to your account, while you are locked out. To defend against this type of attack, many web services allow you to set up two factor authentication (2FA).

Continue reading →

Share this:

★ Like

One blogger likes this.

FOLLOW ME ON TWITTER

Tweets by @sashavodnik

Sasha Vodnik @sashavodnik

Take the next step to secure your passwords, browsing, and networking at my workshop this Thursday! @GA_SF @GA #onlinesecurity #infosecurity generalassemb.ly/education/secu…

Securing Your Digital … Learn to keep your data … generalassemb.ly

Mar 4, 2018

Sasha Vodnik @sashavodnik

AR makes so much more sense to me than strict VR, and here are some reasons why! ⚡ "LukeW's Augmented Reality Examples" by @jeremiastwitter.com/i/moments/9677…

LukeW's Augmented Reality Examples

Jeremy O…

Securing Your Digital Life, Part 1: Choosing a password manager

JANUARY 11, 2018

Configuring and using a password manager is a critical building block of your online security.

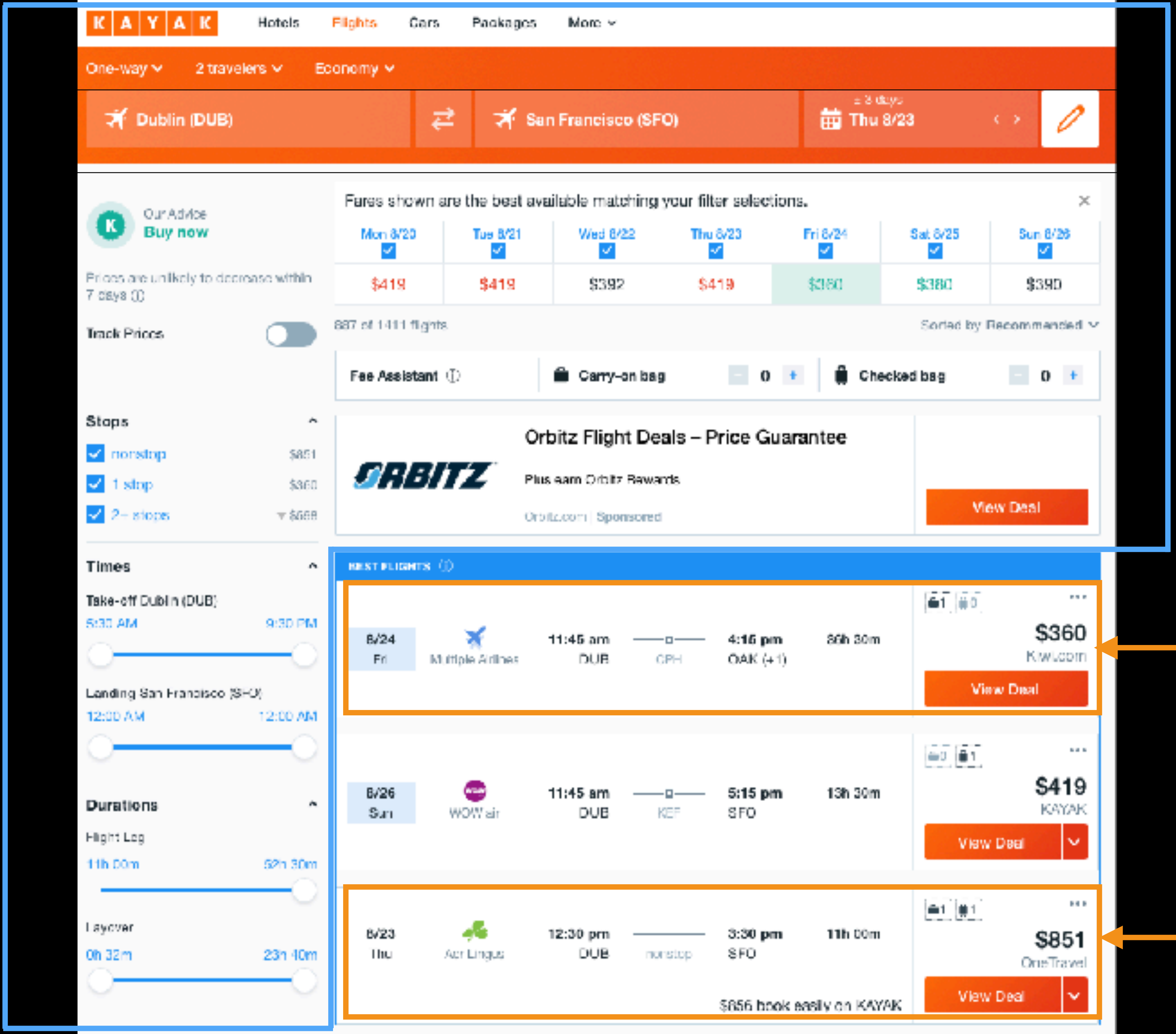
Continue reading →

Share this:

★ Like

One blogger likes this.

Kayak website content



KAYAK Hotels Flights Cars Packages More

One-way 2 travelers Economy

Dublin (DUB) ⇄ San Francisco (SFO) ± 3 days Thu 8/23

Our Advice
Buy now

Prices are unlikely to decrease within 7 days

Track Prices

Stops
☑ nonstop $851
☑ 1 stop $360
☑ 2+ stops $668

Times
Take-off Dublin (DUB)
5:30 AM 9:30 PM
Landing San Francisco (SFO)
12:00 AM 12:00 AM

Durations
Flight Leg
11h 00m 52h 30m
Layover
0h 32m 23h 40m

Fares shown are the best available matching your filter selections. ×

Mon 8/20 ☑	Tue 8/21 ☑	Wed 8/22 ☑	Thu 8/23 ☑	Fri 8/24 ☑	Sat 8/25 ☑	Sun 8/26 ☑
$419	$419	$392	$419	$360	$380	$390

887 of 1411 flights Sorted by Recommended

Fee Assistant ⓘ Carry-on bag − 0 + Checked bag − 0 +

ORBITZ Orbitz Flight Deals – Price Guarantee
Plus earn Orbitz Rewards
Orbitz.com | Sponsored
View Deal

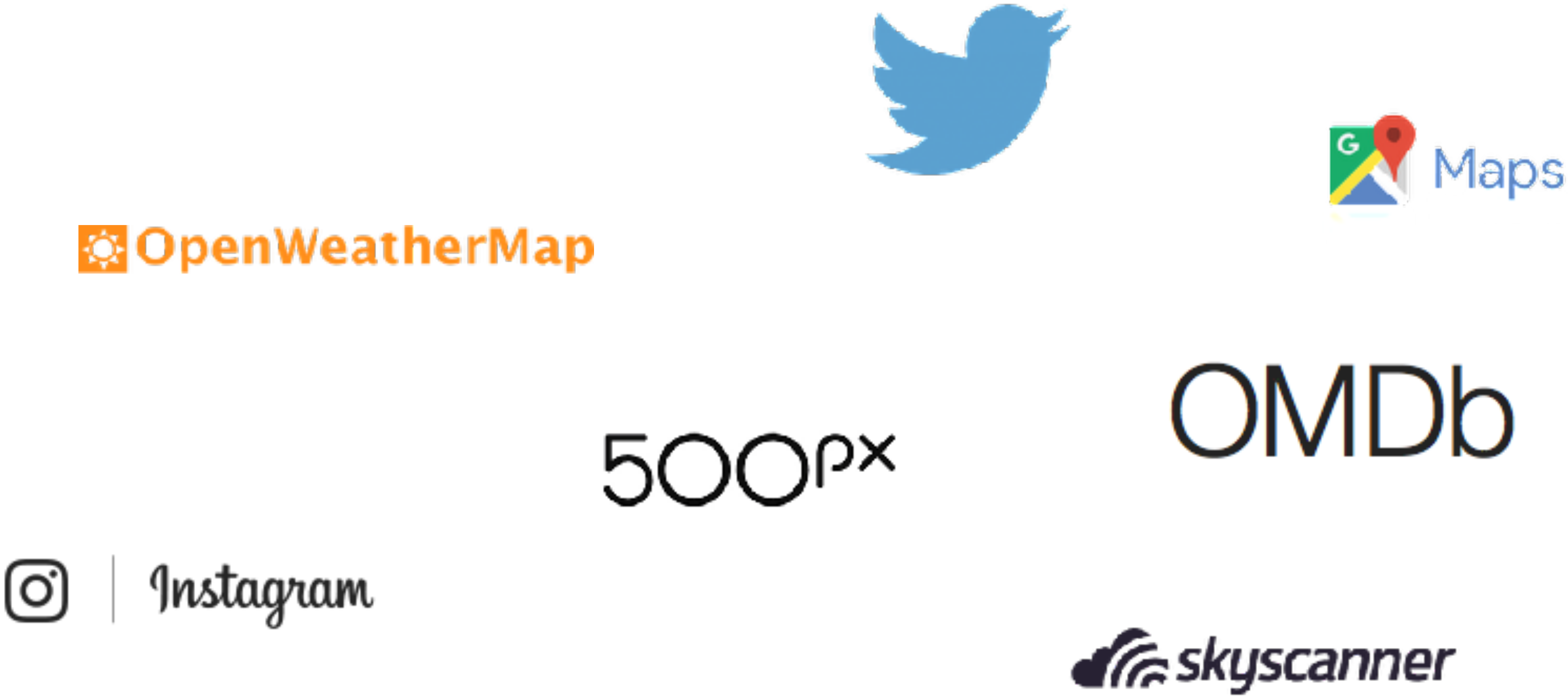
BEST FLIGHTS ⓘ

8/24 Fri	Multiple Airlines	11:45 am DUB	CPH	4:15 pm OAK (+1)	20h 30m	1 0 $360 Kiwi.com	View Deal
8/26 Sun	WOW air	11:45 am DUB	KEF	5:15 pm SFO	13h 30m	0 1 $419 KAYAK	View Deal
8/23 Thu	Aer Lingus	12:30 pm DUB	nonstop	3:30 pm SFO	11h 00m	1 1 $851 OneTravel $856 book easily on KAYAK	View Deal

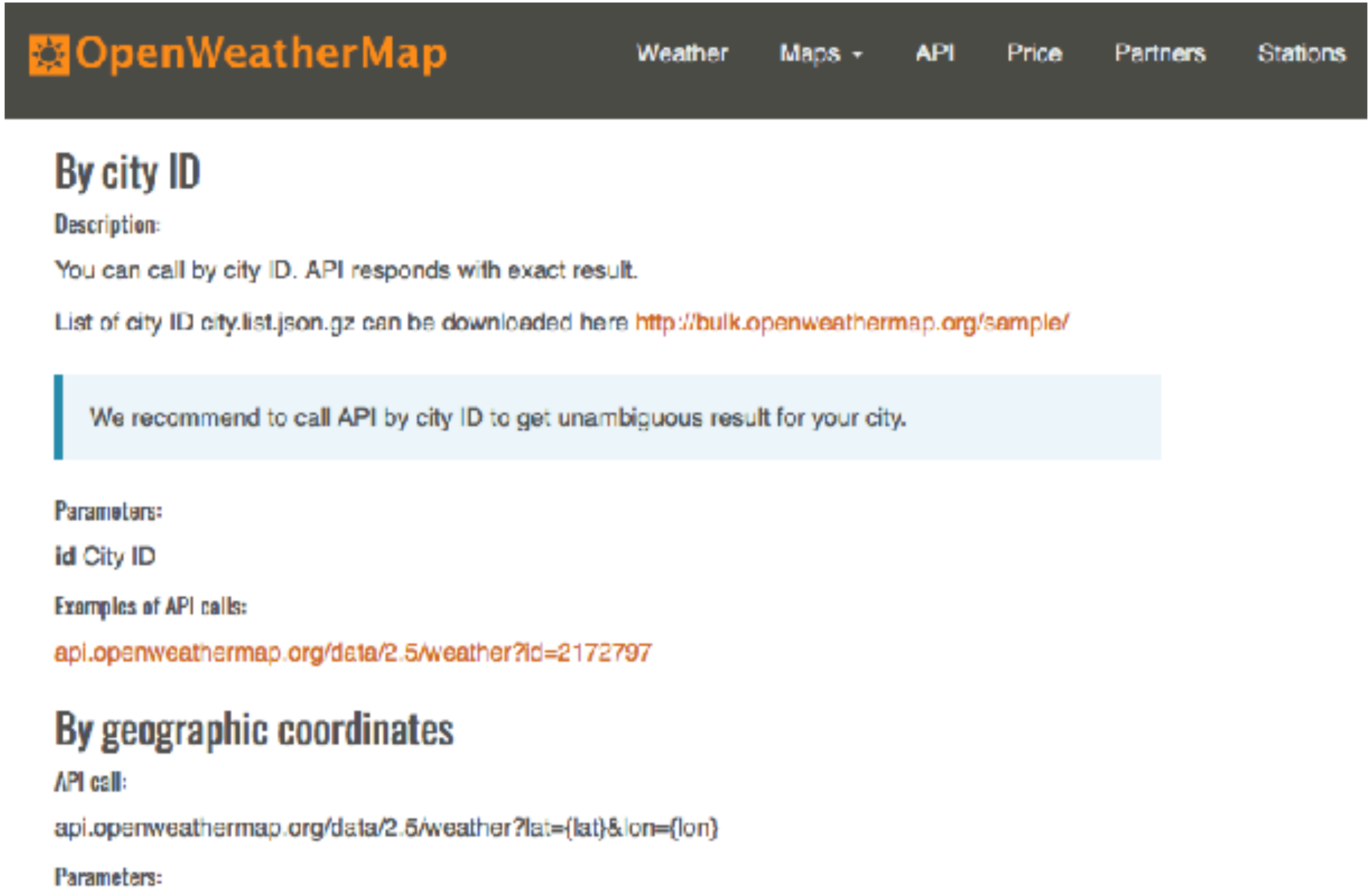
Content from kiwi.com using API

Content from OneTravel using API

WEB SERVICES



API = application programming interface



By city ID

Description:

You can call by city ID. API responds with exact result.

List of city ID city.list.json.gz can be downloaded here http://bulk.openweathermap.org/sample/

> We recommend to call API by city ID to get unambiguous result for your city.

Parameters:

id City ID

Examples of API calls:

api.openweathermap.org/data/2.5/weather?id=2172797

By geographic coordinates

API call:

api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}

Parameters:

APIS IN THE REAL WORLD

- Most APIs are unique, like separate languages
- APIs for
  - devices (iPhone)
  - operating systems (macOS)
  - JavaScript libraries (jQuery API)
  - services (Slack)

WEB SERVICES

Maps

OpenWeatherMap

500px

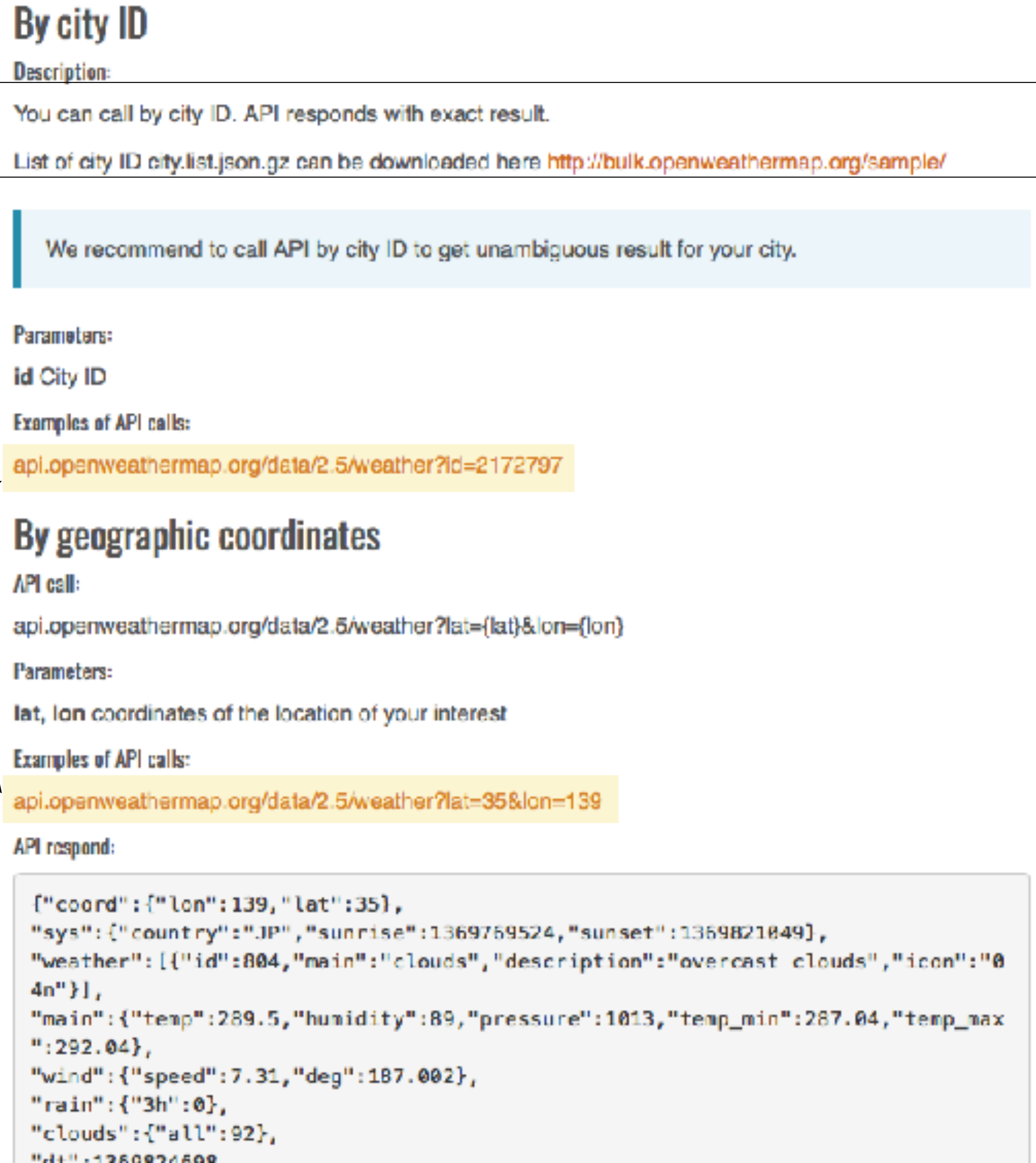
OMDb

Instagram

skyscanner

ENDPOINTS

- Addresses (URLs) that return data (JSON) instead of markup (HTML)



By city ID

Description:

You can call by city ID. API responds with exact result.

List of city ID city.list.json.gz can be downloaded here http://bulk.openweathermap.org/sample/

> We recommend to call API by city ID to get unambiguous result for your city.

Parameters:

id City ID

Examples of API calls:

api.openweathermap.org/data/2.5/weather?id=2172797

By geographic coordinates

API call:

api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}

Parameters:

lat, lon coordinates of the location of your interest

Examples of API calls:

api.openweathermap.org/data/2.5/weather?lat=35&lon=139

API respond:

```
{"coord":{"lon":139,"lat":35},
"sys":{"country":"JP","sunrise":1369769524,"sunset":1369821049},
"weather":[{"id":804,"main":"clouds","description":"overcast clouds","icon":"0
4n"}],
"main":{"temp":289.5,"humidity":89,"pressure":1013,"temp_min":287.04,"temp_max
":292.04},
"wind":{"speed":7.31,"deg":187.002},
"rain":{"3h":0},
"clouds":{"all":92},
"dt":1369824698
```

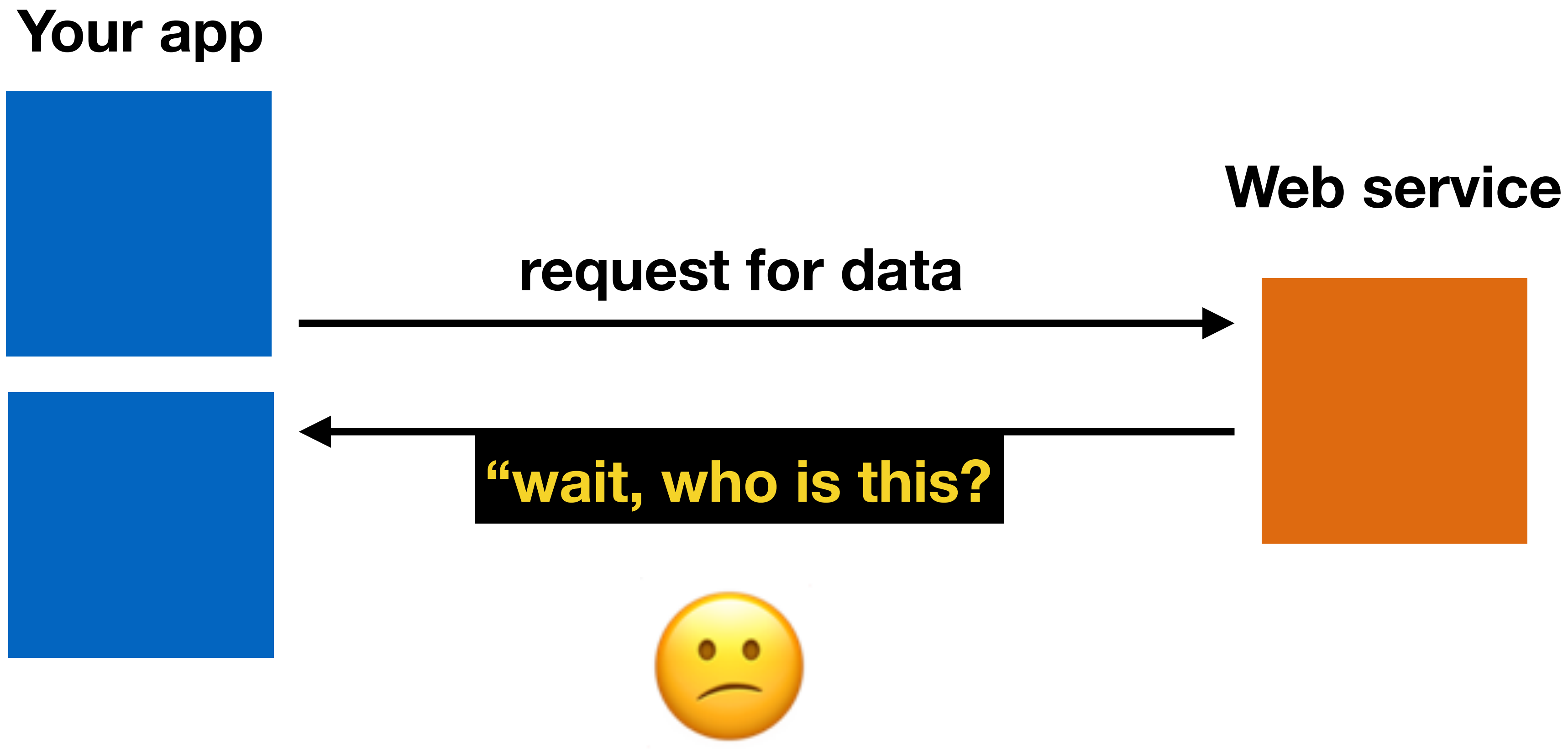
WHAT WE NEED TO KNOW TO USE AN API

TERMS OF SERVICE

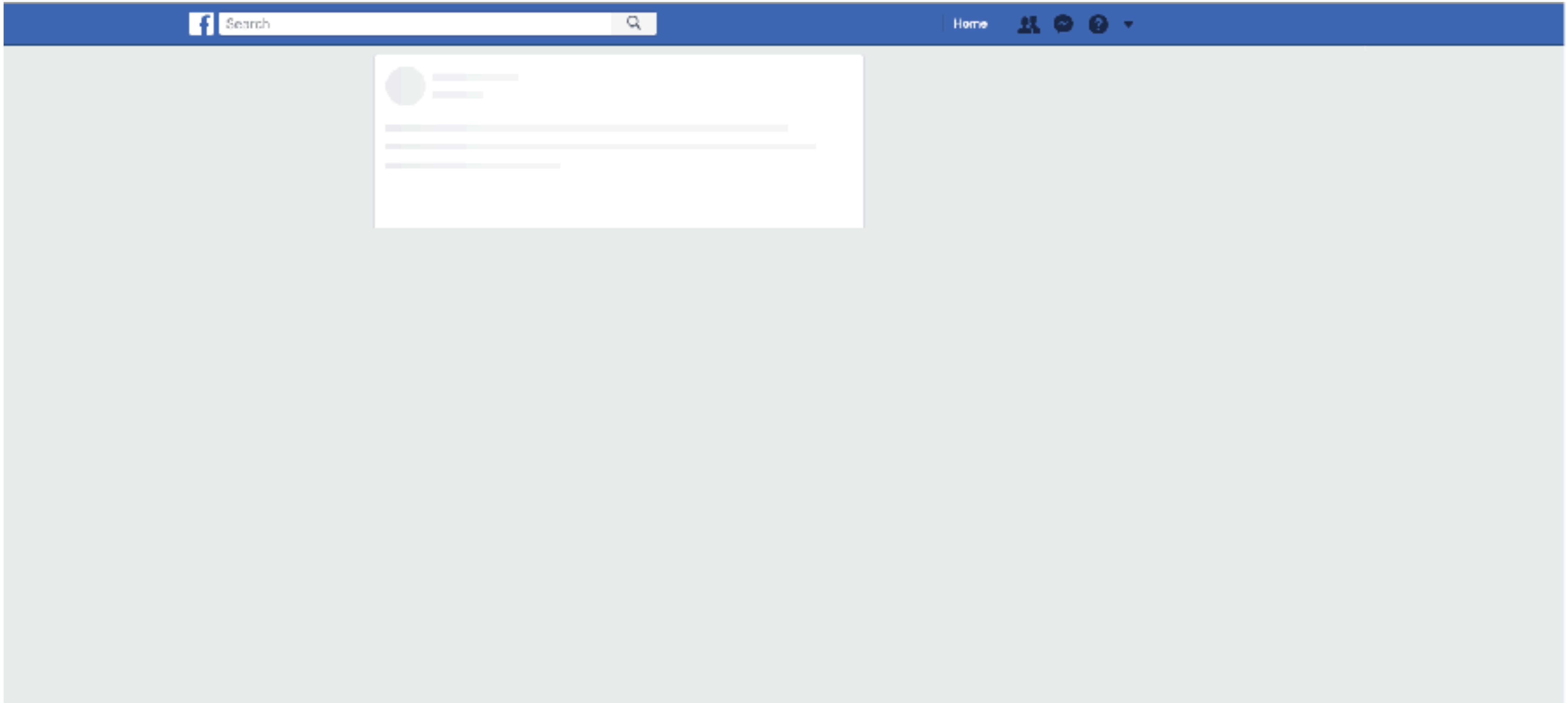
HOW TO MAKE A REQUEST

HOW TO UNDERSTAND RESPONSE

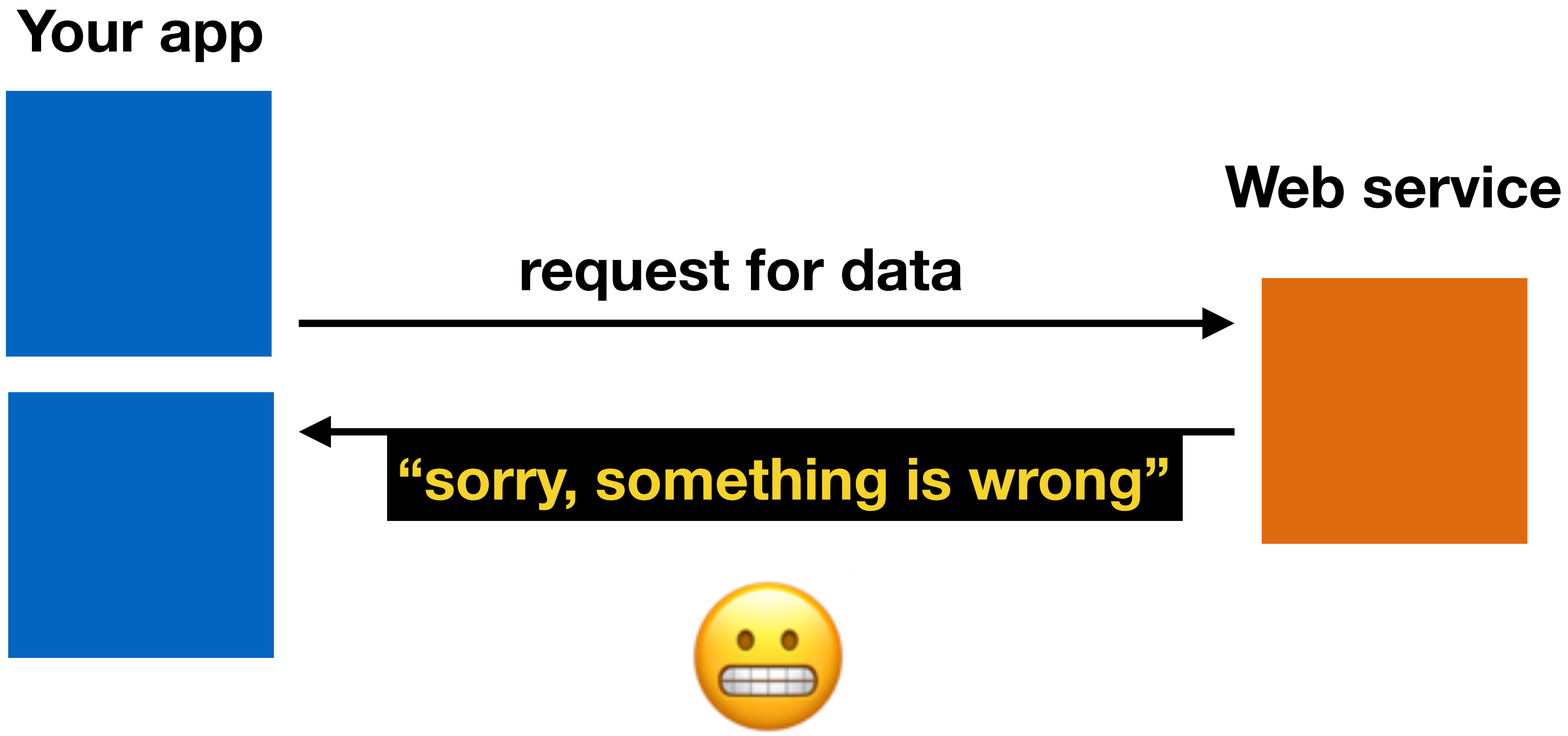
AN API MIGHT REQUIRE AUTHENTICATION



YOUR APP MIGHT EXPERIENCE A DELAYED RESPONSE



YOUR REQUEST MAY RESULT IN AN ERROR



Your app

Web service

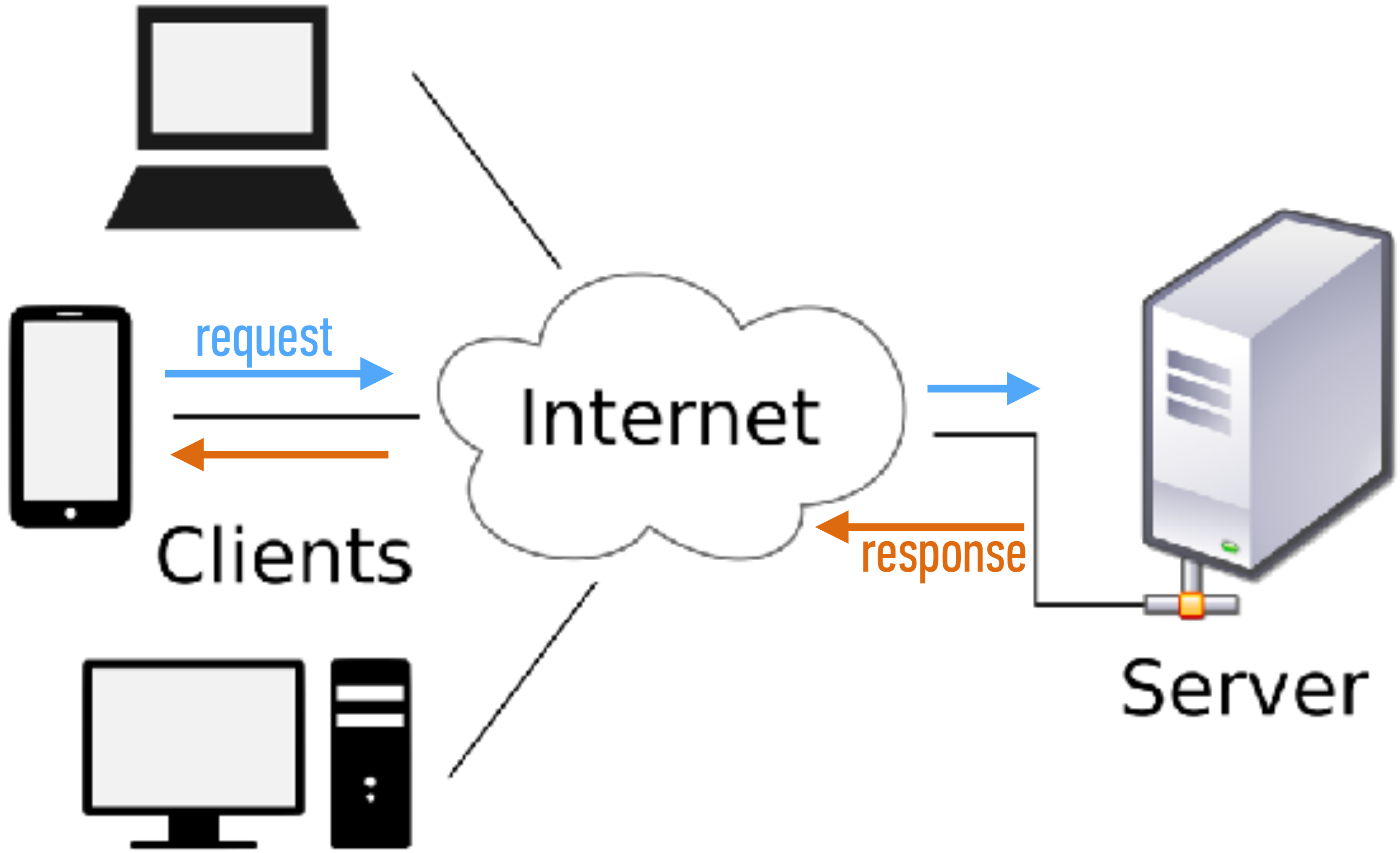
request for data

“sorry, something is wrong”

😬

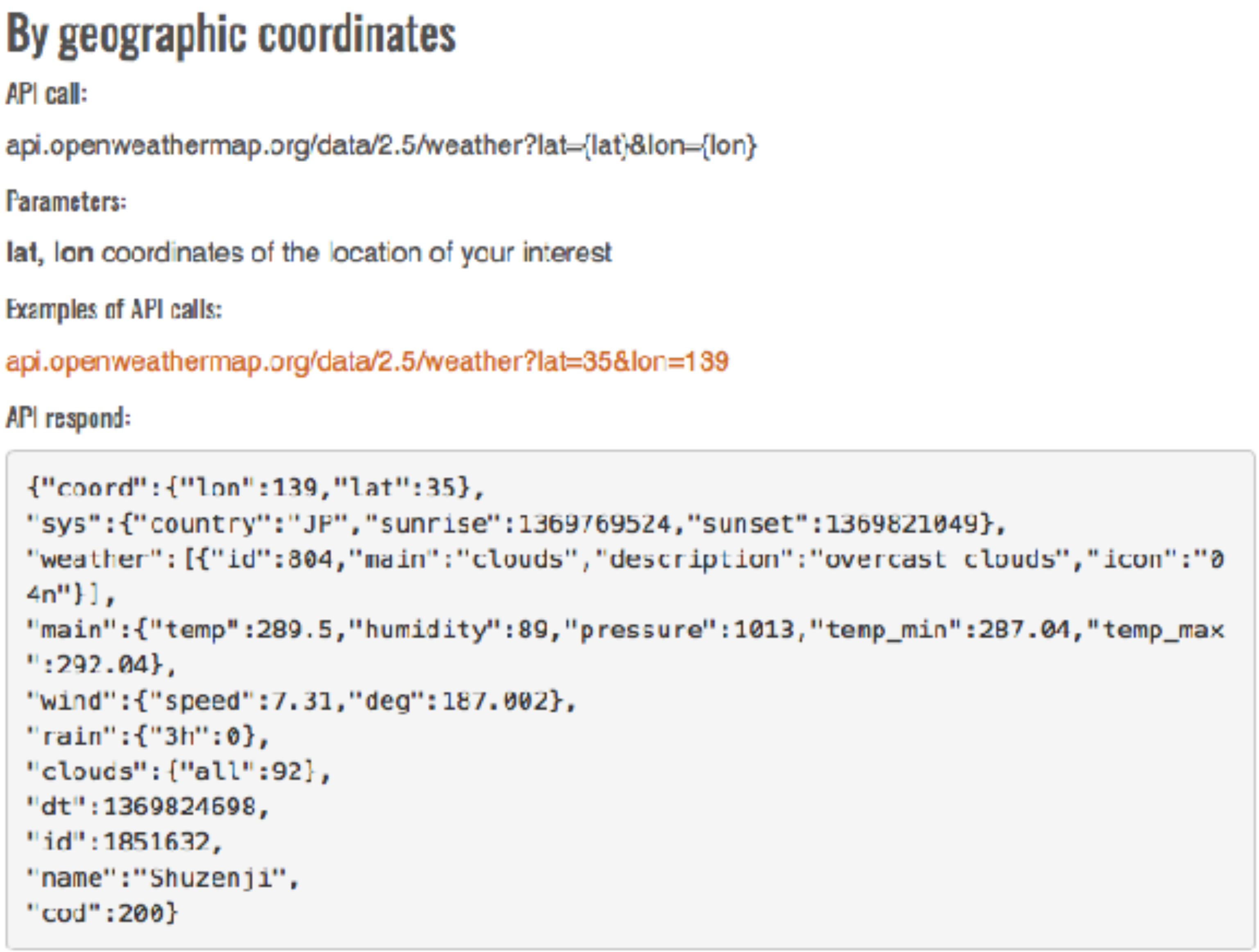
REST (representational state transfer)

- architectural style of web applications
- transfers a representation of the state of a resource between the server and the client



RESTful API

- adheres to REST architecture
- uses
  - a base URL
  - an Internet media type (such as JSON)
  - standard HTTP methods



By geographic coordinates

API call:

api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}

Parameters:

lat, lon coordinates of the location of your interest

Examples of API calls:

api.openweathermap.org/data/2.5/weather?lat=35&lon=139

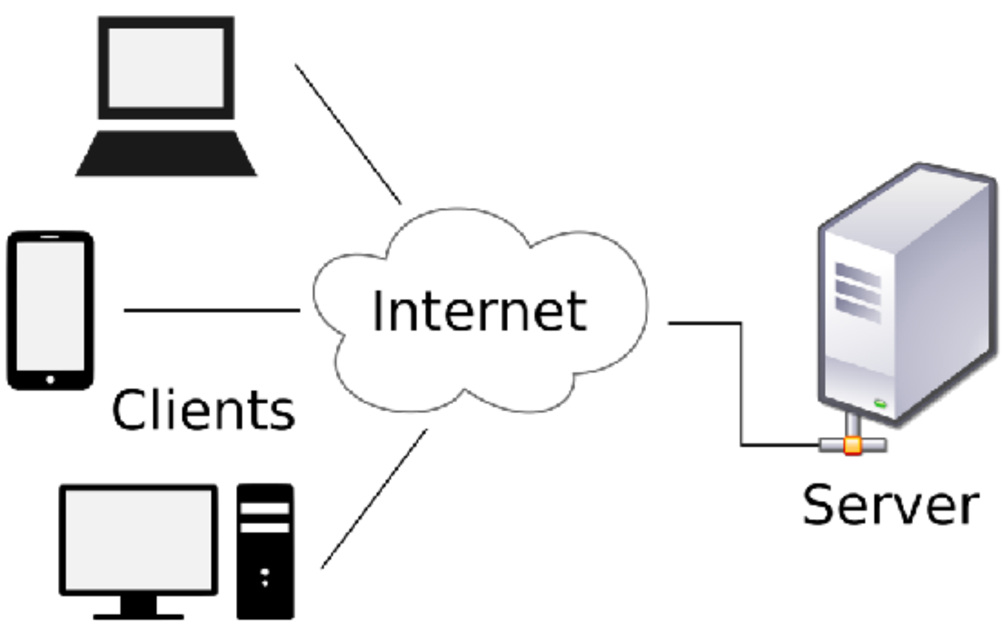
API respond:

```
{"coord":{"lon":139,"lat":35},
"sys":{"country":"JP","sunrise":1369769524,"sunset":1369821049},
"weather":[{"id":804,"main":"clouds","description":"overcast clouds","icon":"0
4n"}],
"main":{"temp":289.5,"humidity":89,"pressure":1013,"temp_min":287.04,"temp_max
":292.04},
"wind":{"speed":7.31,"deg":187.002},
"rain":{"3h":0},
"clouds":{"all":92},
"dt":1369824698,
"id":1851632,
"name":"Shuzenji",
"cod":200}
```

HTTP

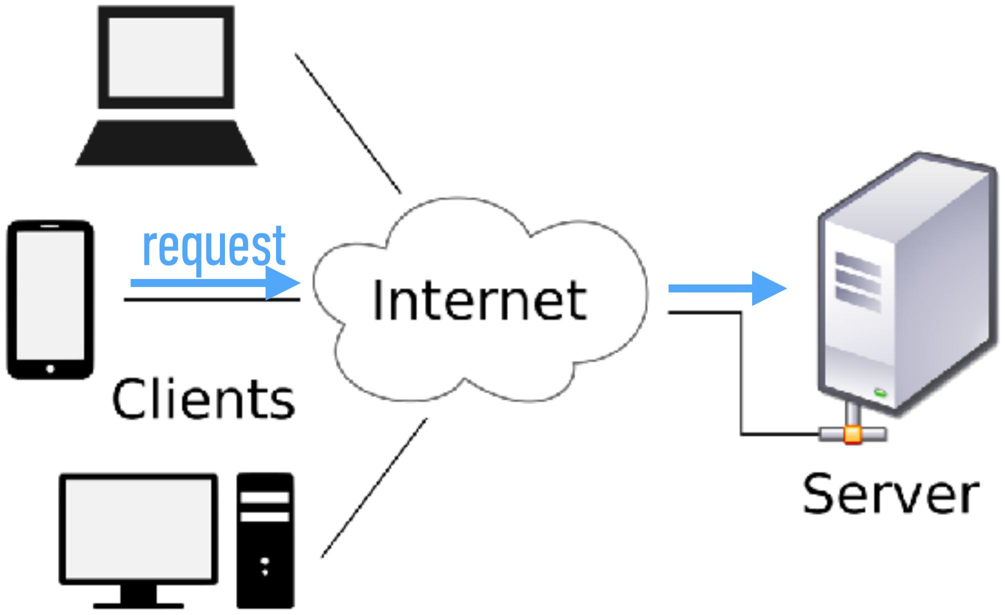
HTTP (hypertext transfer protocol)

- System of rules for how web pages are transmitted between computers
- Defines the format of messages passed between HTTP clients and HTTP servers



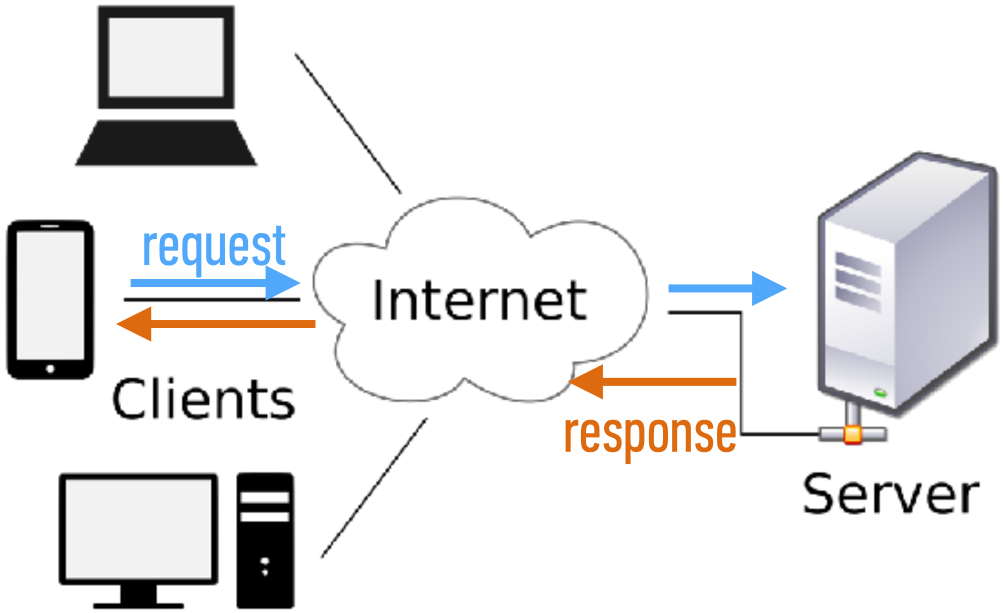
HTTP (hypertext transfer protocol)

- A client sends a **request** to a server.

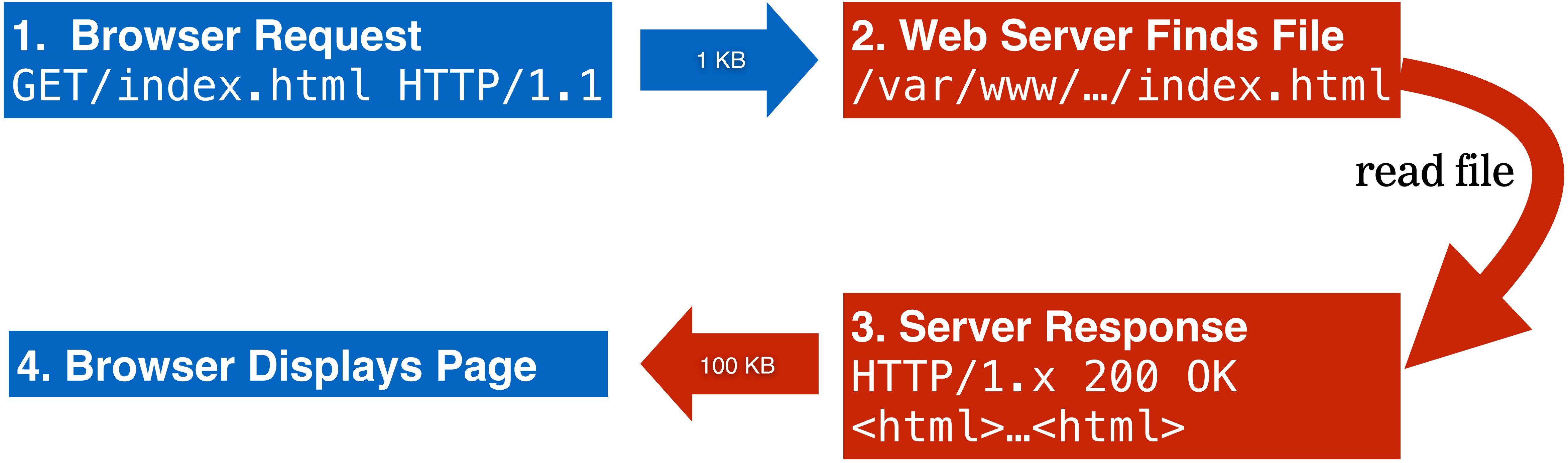


HTTP (hypertext transfer protocol)

- A server sends a **response** back to a client.



HTTP REQUEST AND RESPONSE



1. Browser Request
`GET/index.html HTTP/1.1`

2. Web Server Finds File
`/var/www/…/index.html`

3. Server Response
`HTTP/1.x 200 OK`
`<html>…<html>`

4. Browser Displays Page

HTTP (hypertext transfer protocol)

HTTP client

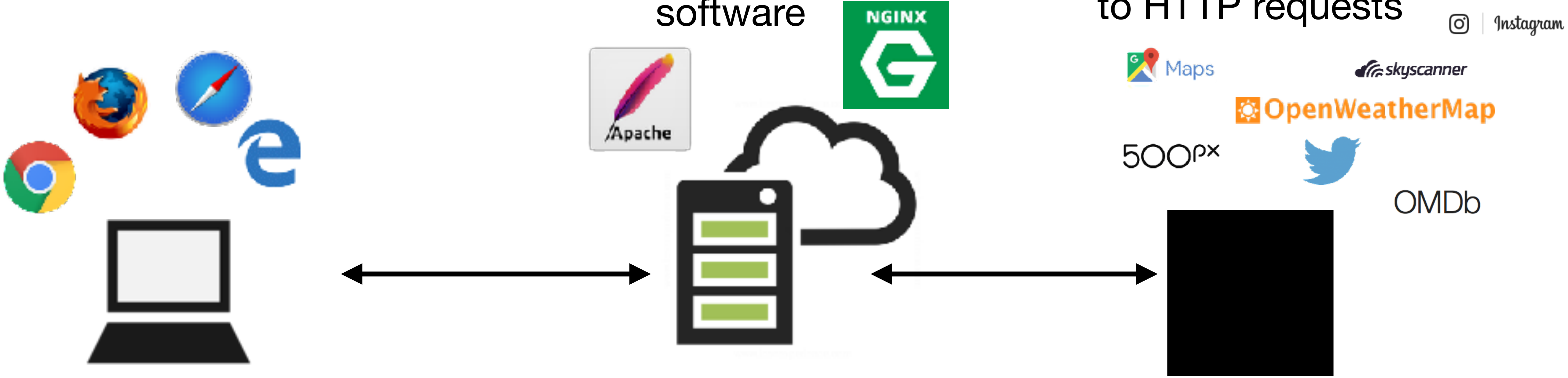
web browser

HTTP server

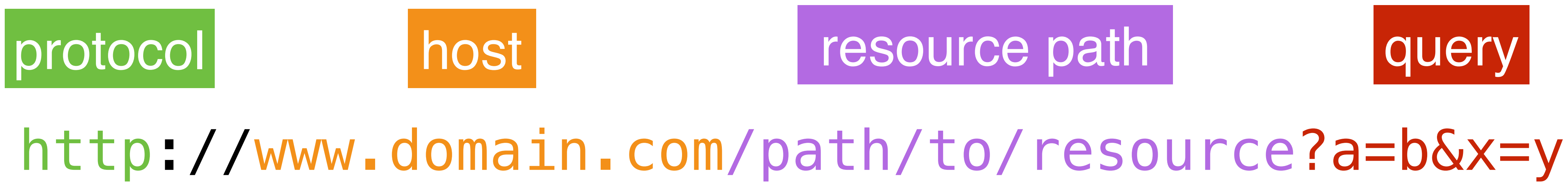
web server software

Web service

app that responds to HTTP requests



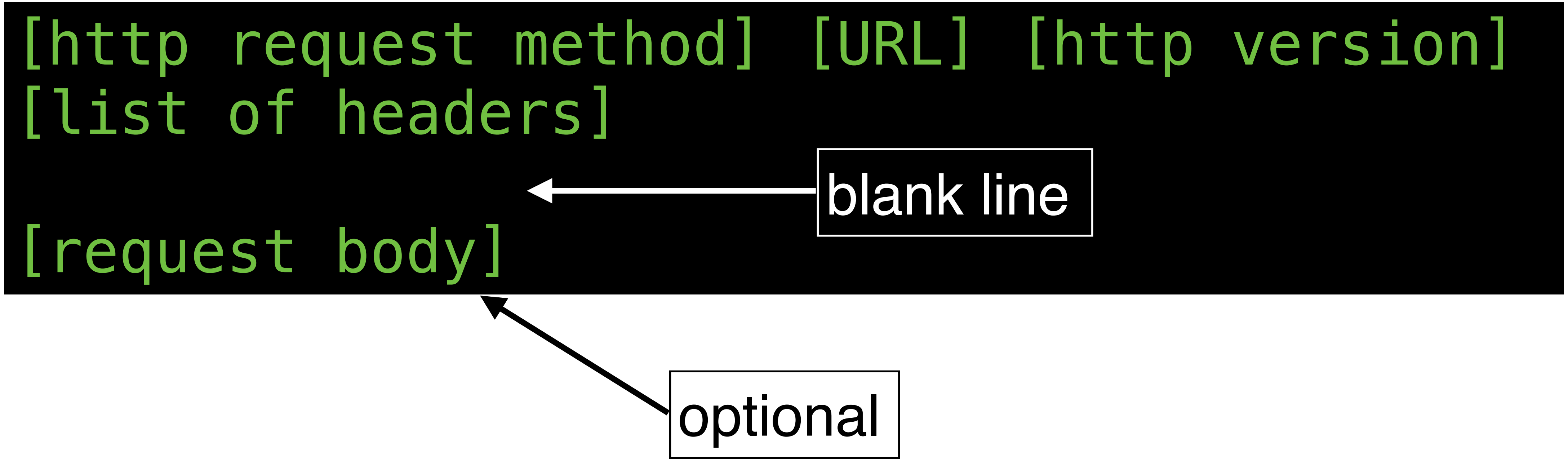
HTTP REQUESTS IN EVERYDAY LIFE



protocol host resource path query

`http://www.domain.com/path/to/resource?a=b&x=y`

HTTP REQUEST STRUCTURE



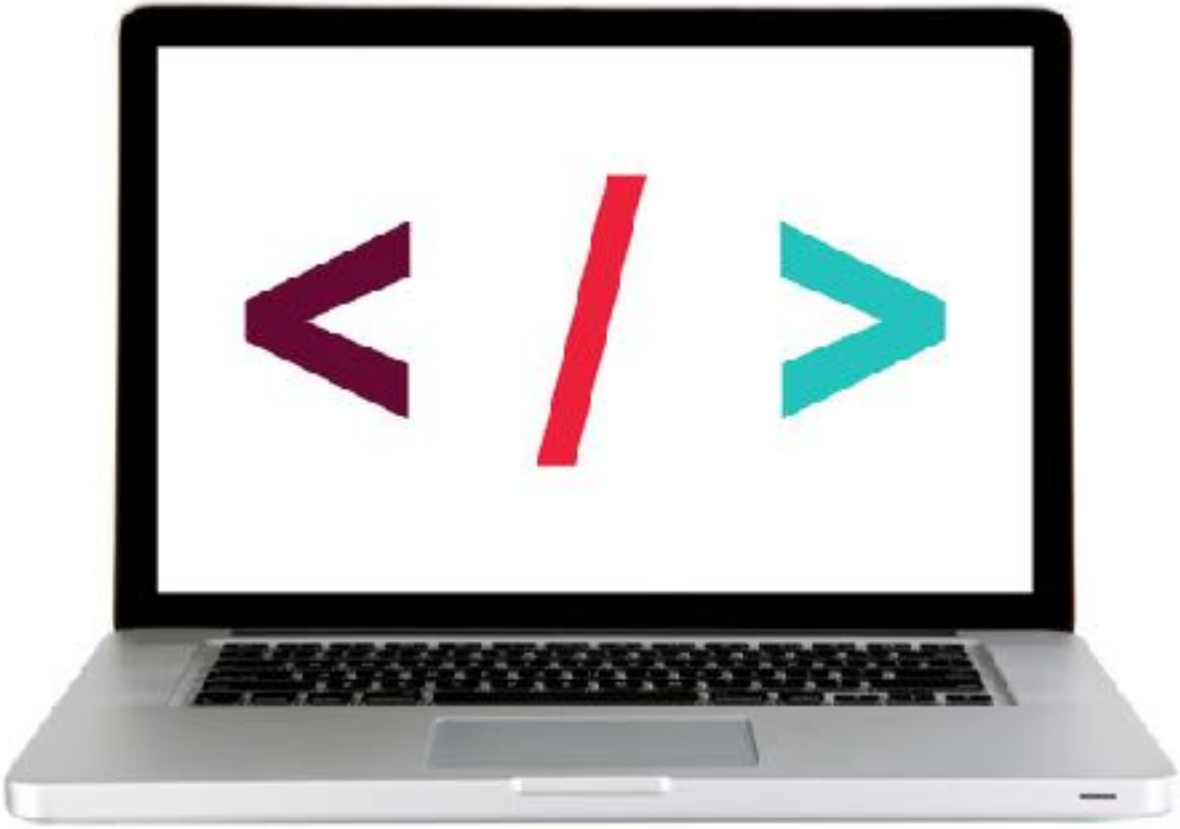
```
[http request method] [URL] [http version]
[list of headers]

[request body]
```

HTTP REQUEST METHODS (“HTTP VERBS”)

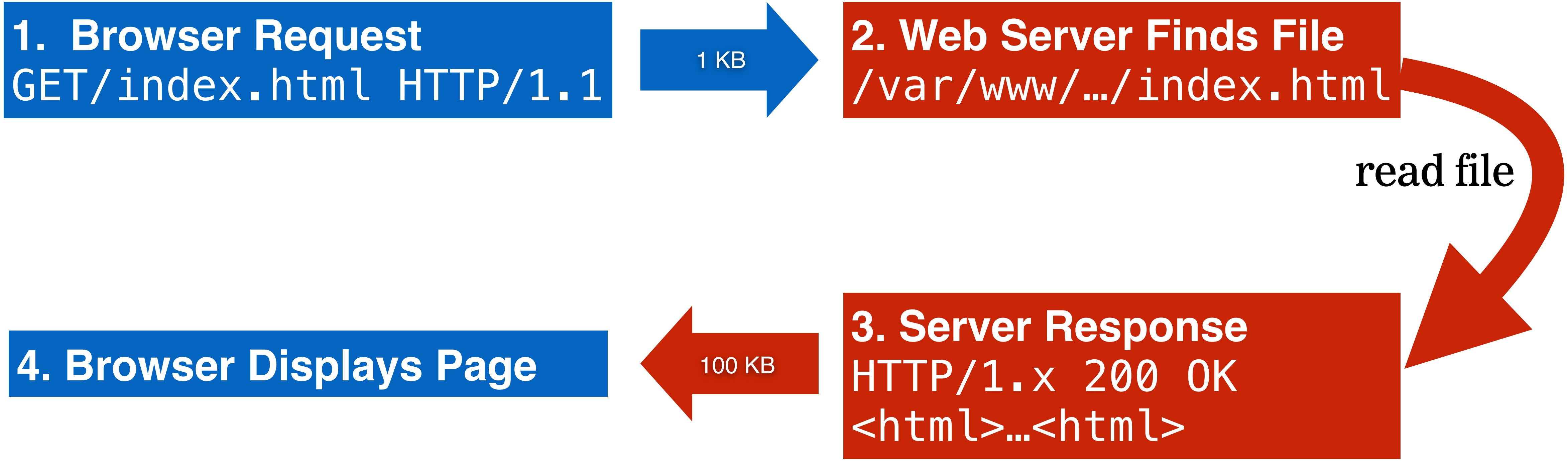
GET	Retrieve a resource
POST	Create a resource
PATCH	Update an existing resource (use instead of PUT, which replaces)
DELETE	Delete a resource
HEAD	Retrieve the headers for a resource

Most widely used

LET'S TAKE A CLOSER LOOK



HTTP REQUEST AND RESPONSE



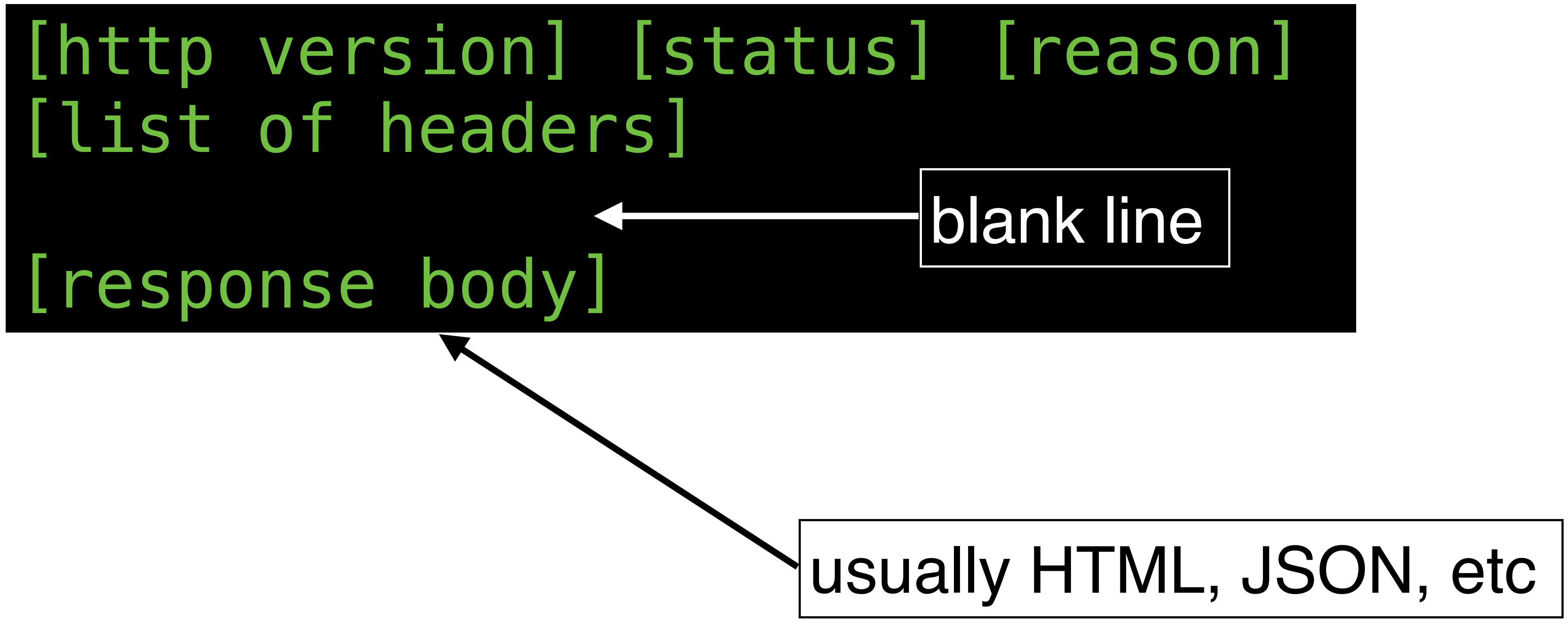
1. Browser Request
`GET/index.html HTTP/1.1`

2. Web Server Finds File
`/var/www/…/index.html`

3. Server Response
`HTTP/1.x 200 OK`
`<html>…<html>`

4. Browser Displays Page

HTTP RESPONSE STRUCTURE

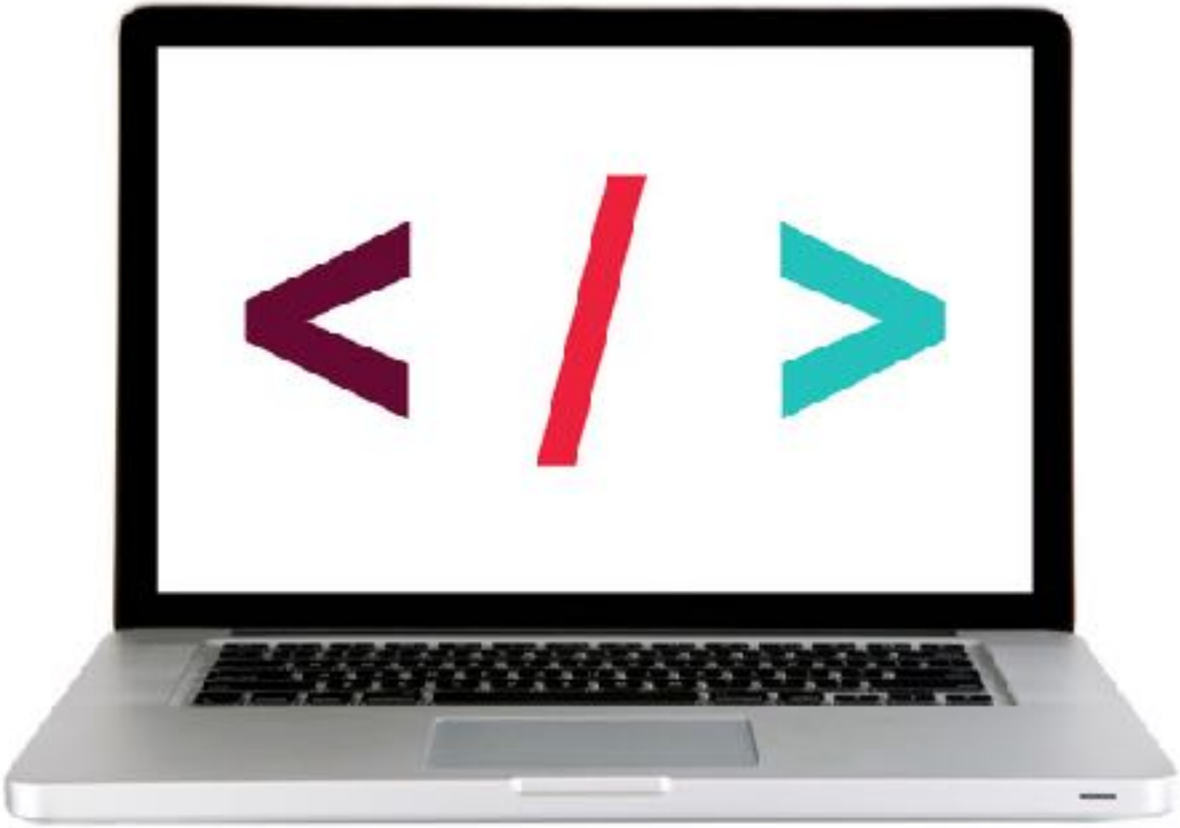


```
[http version] [status] [reason]
[list of headers]

[response body]
```

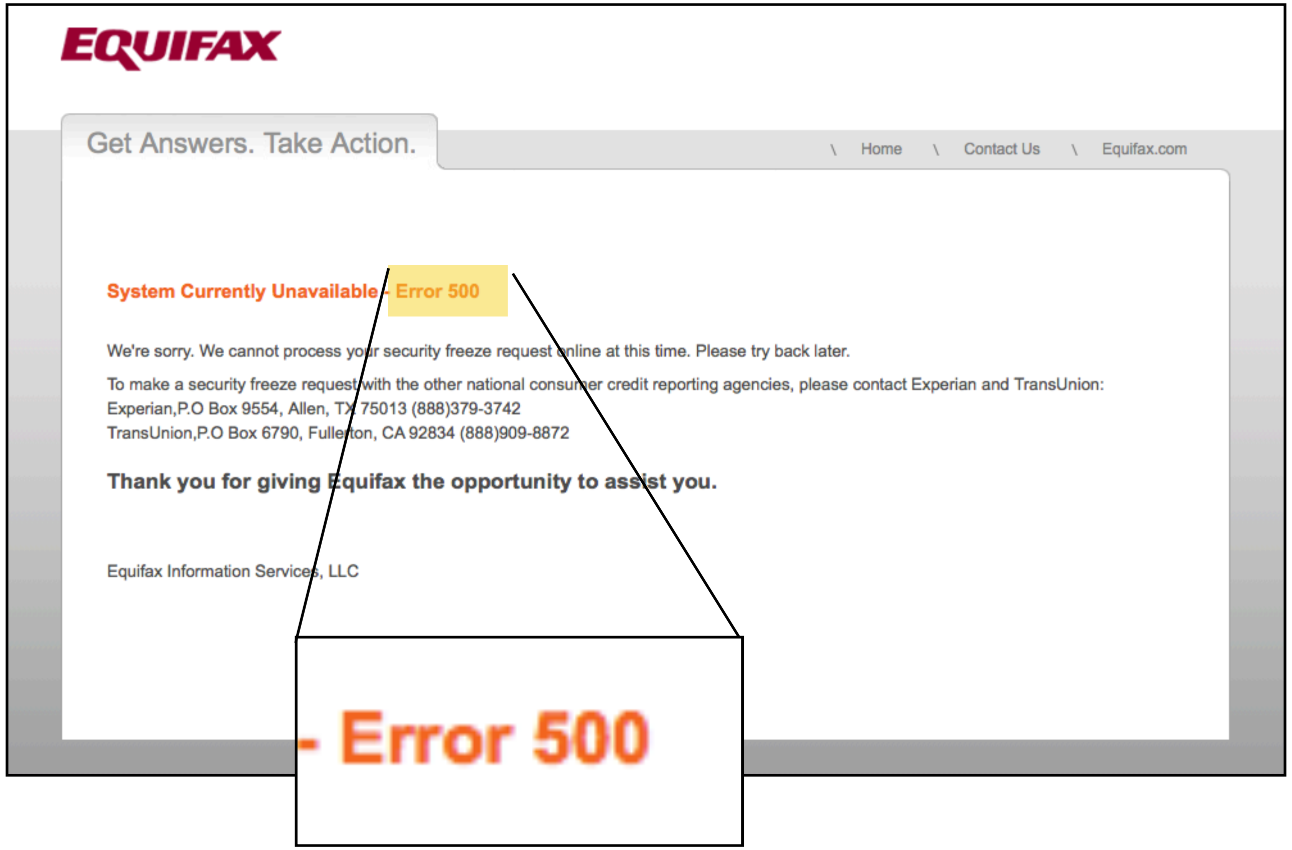
blank line

usually HTML, JSON, etc

LET'S TAKE A CLOSER LOOK



HTTP STATUS CODES



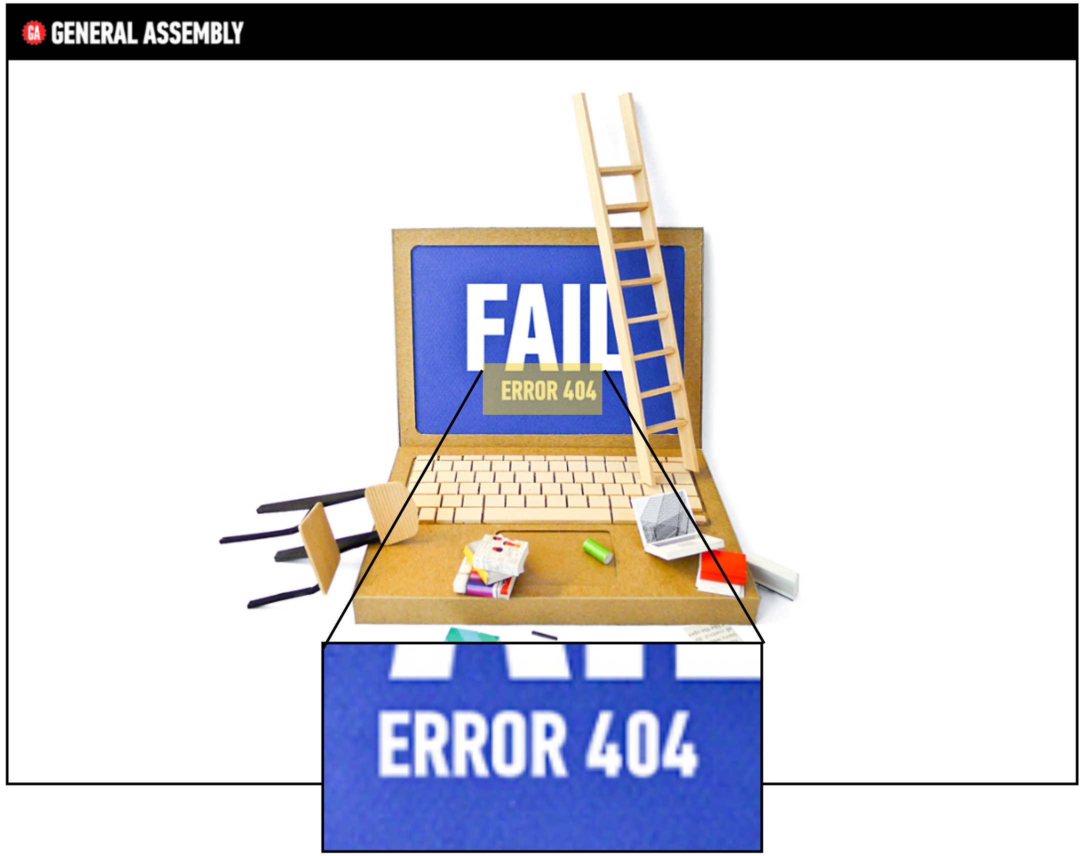
System Currently Unavailable - Error 500

We're sorry. We cannot process your security freeze request online at this time. Please try back later.

To make a security freeze request with the other national consumer credit reporting agencies, please contact Experian and TransUnion:
Experian,P.O Box 9554, Allen, TX 75013 (888)379-3742
TransUnion,P.O Box 6790, Fullerton, CA 92834 (888)909-8872

Thank you for giving Equifax the opportunity to assist you.

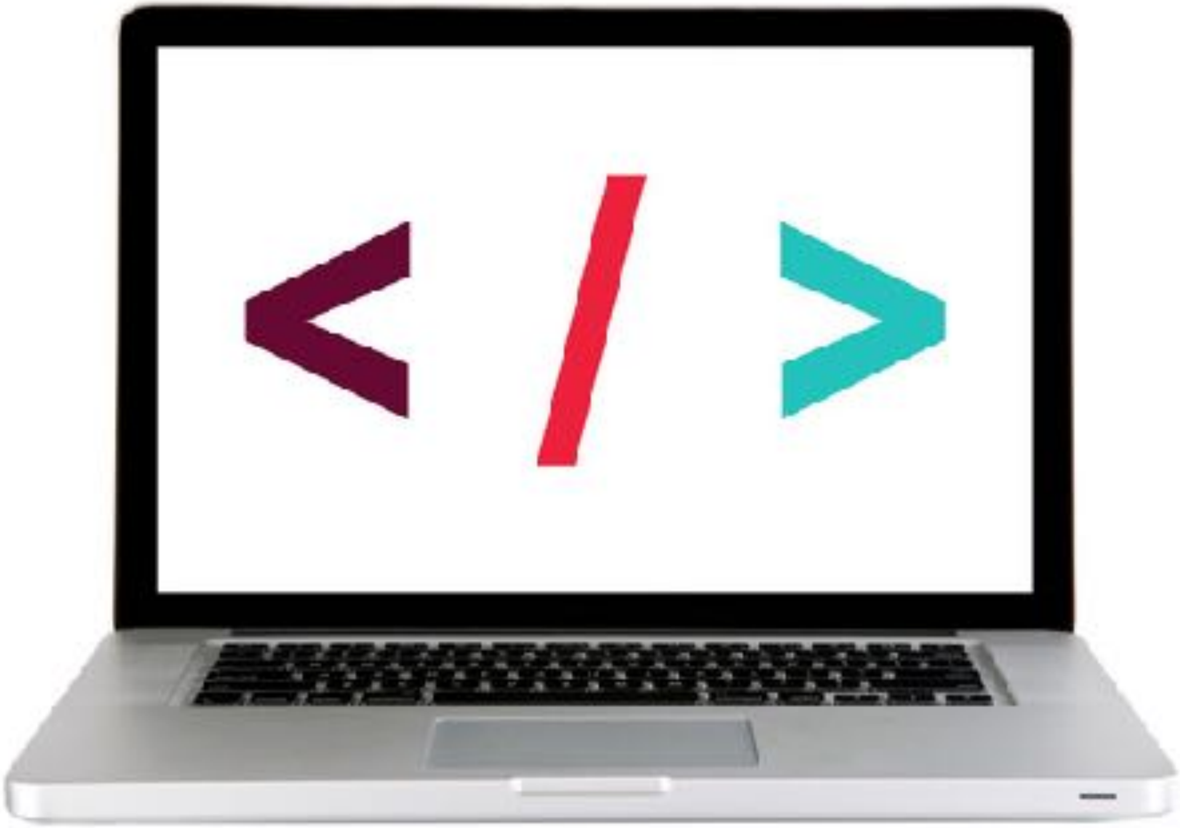
Equifax Information Services, LLC



HTTP STATUS CODES

`200`	Okay
`301`	Moved permanently
`302`	Moved temporarily
`400`	Bad request
`403`	Forbidden
`404`	Not found
`500`	Internal server error

LET'S TAKE A CLOSER LOOK



Ajax

Ajax

Asynchronous

JavaScript

And

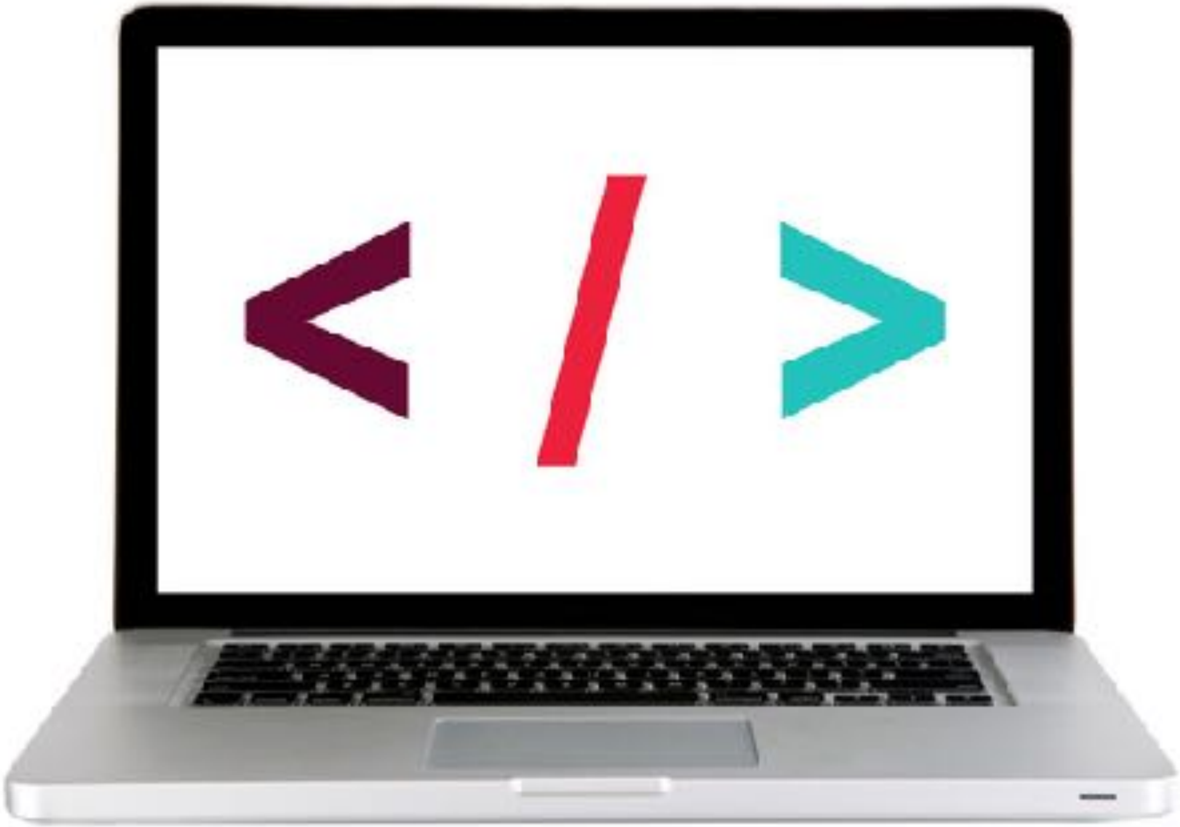
XML **or JSON!**

Ajax in vanilla JS

Fetch = Ajax requests in vanilla JavaScript

```
fetch(url).then(function(response) {
  // check if request was successful
}).then(function(response) {
  // do something with the response
});
```

LET'S TAKE A CLOSER LOOK



EXERCISE - CREATING AN AJAX REQUEST

EXERCISE

LOCATION

- starter-code > 1-fetch-ajax-exercise

TIMING

5 min

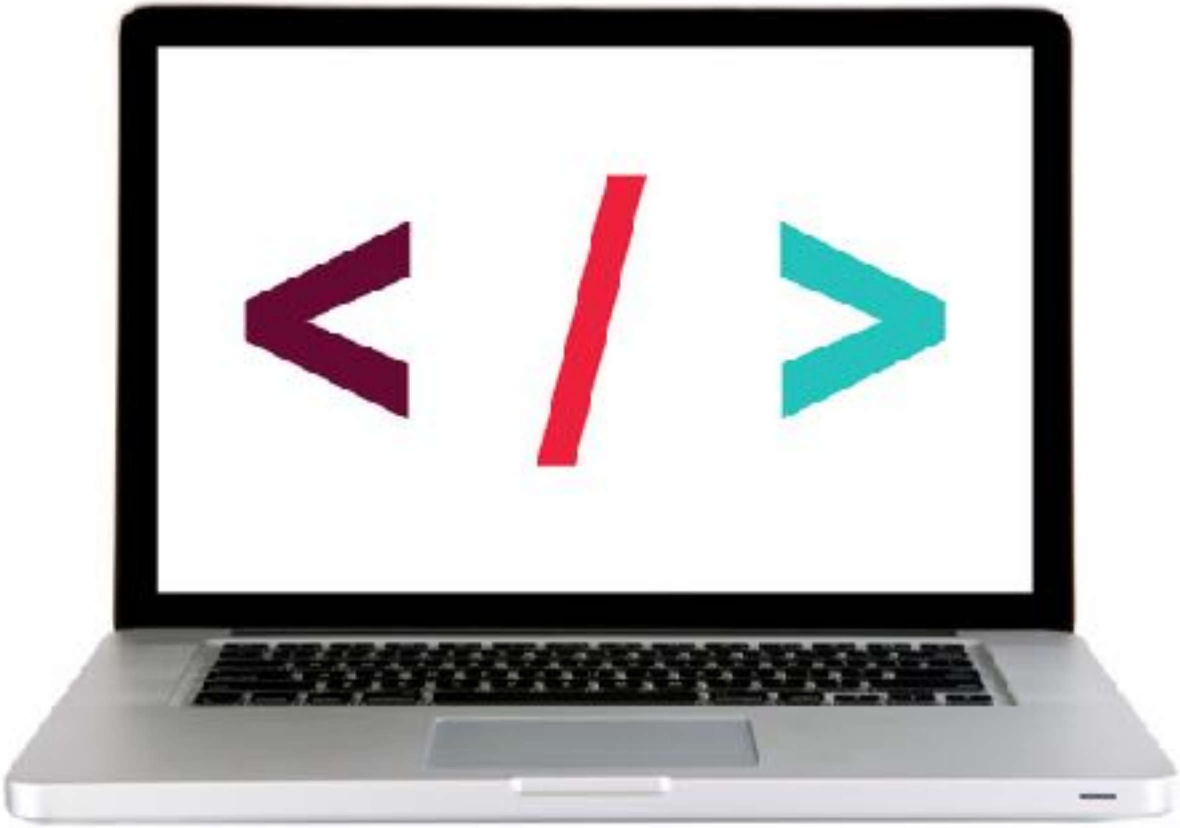
1. Copy the code from the codealong to the main.js file.
2. Change the URL to the one shown in the instructions.
3. Verify that a new set of results is shown in the console.
4. BONUS: Customize the error message to display the text of the HTTP status message.
   (Hint: look at https://developer.mozilla.org/en-US/docs/Web/API/Response/statusText)
5. BONUS: Refactor the code to work with user interaction. In the index.html file there is a "Get Health Data" button. Modify your code so data is only requested and logged to the console after a user clicks the button.

jQuery Ajax

Using Ajax with jQuery

method	description
`$.get()`	loads data from a server using an HTTP GET request
`$.ajax()`	performs an Ajax request based on parameters you specify

LET'S TAKE A CLOSER LOOK

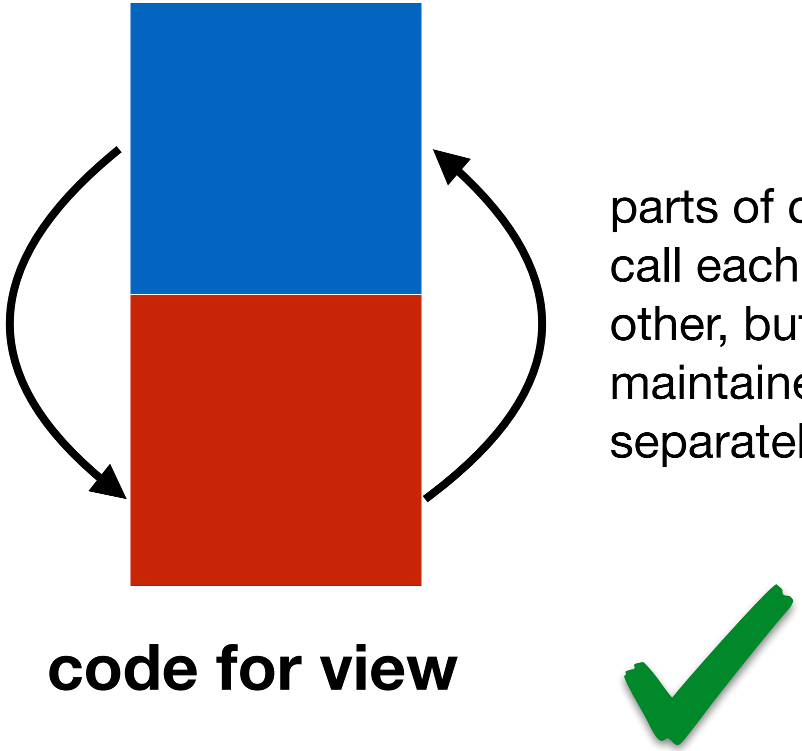


Code organization

SEPARATION OF CONCERNS

code for data and view intermingled

✗



code for data

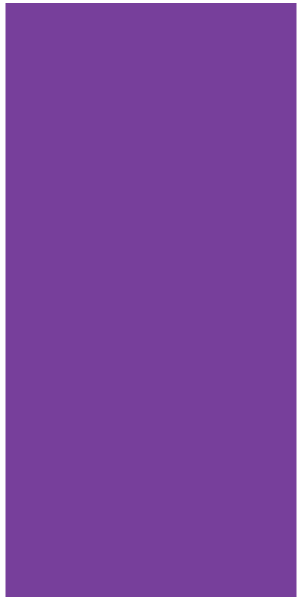
parts of code call each other, but are maintained separately

code for view

✓

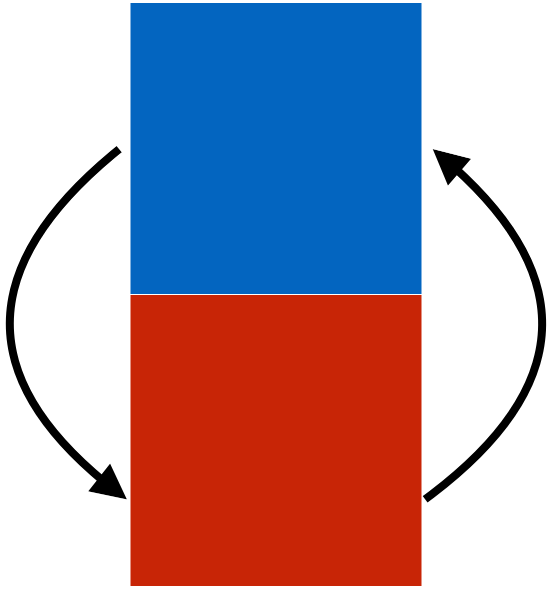
SEPARATION OF CONCERNS – HTTP

code for client
and for HTTP
requests
intermingled



✗

code for client

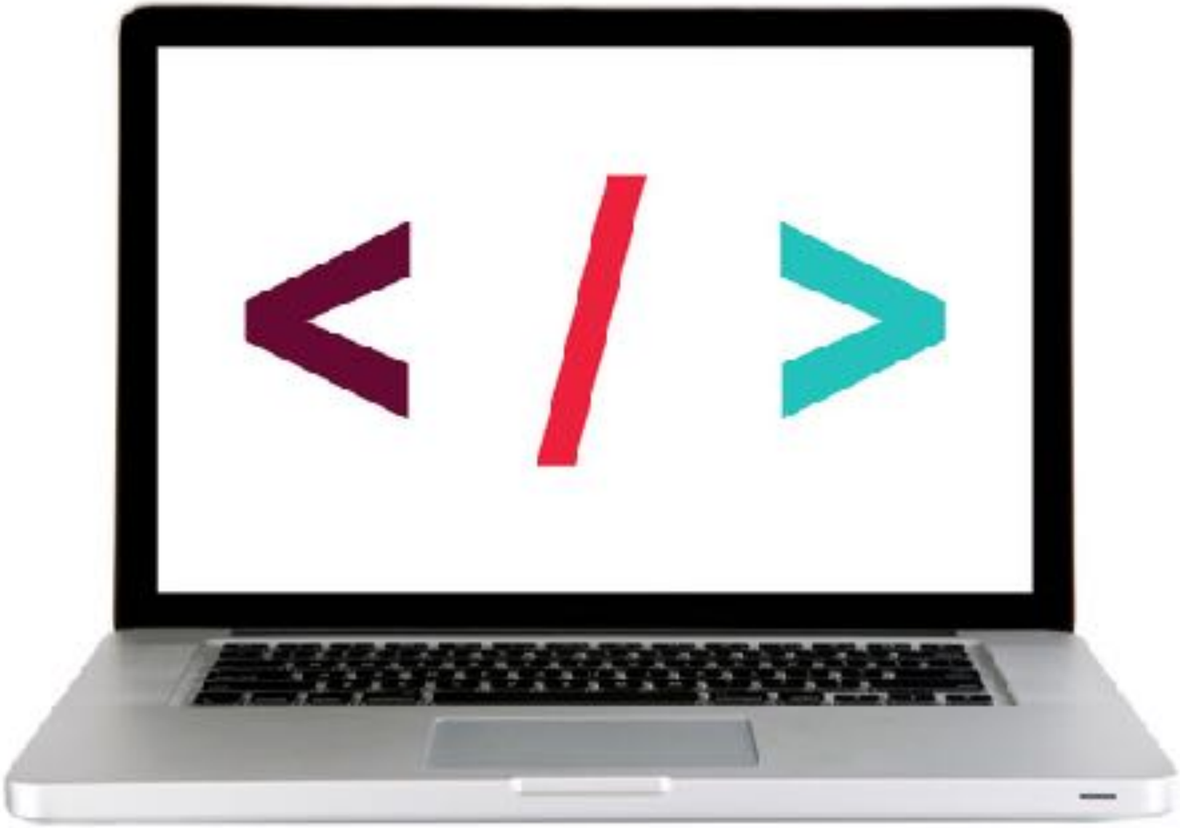


code for HTTP

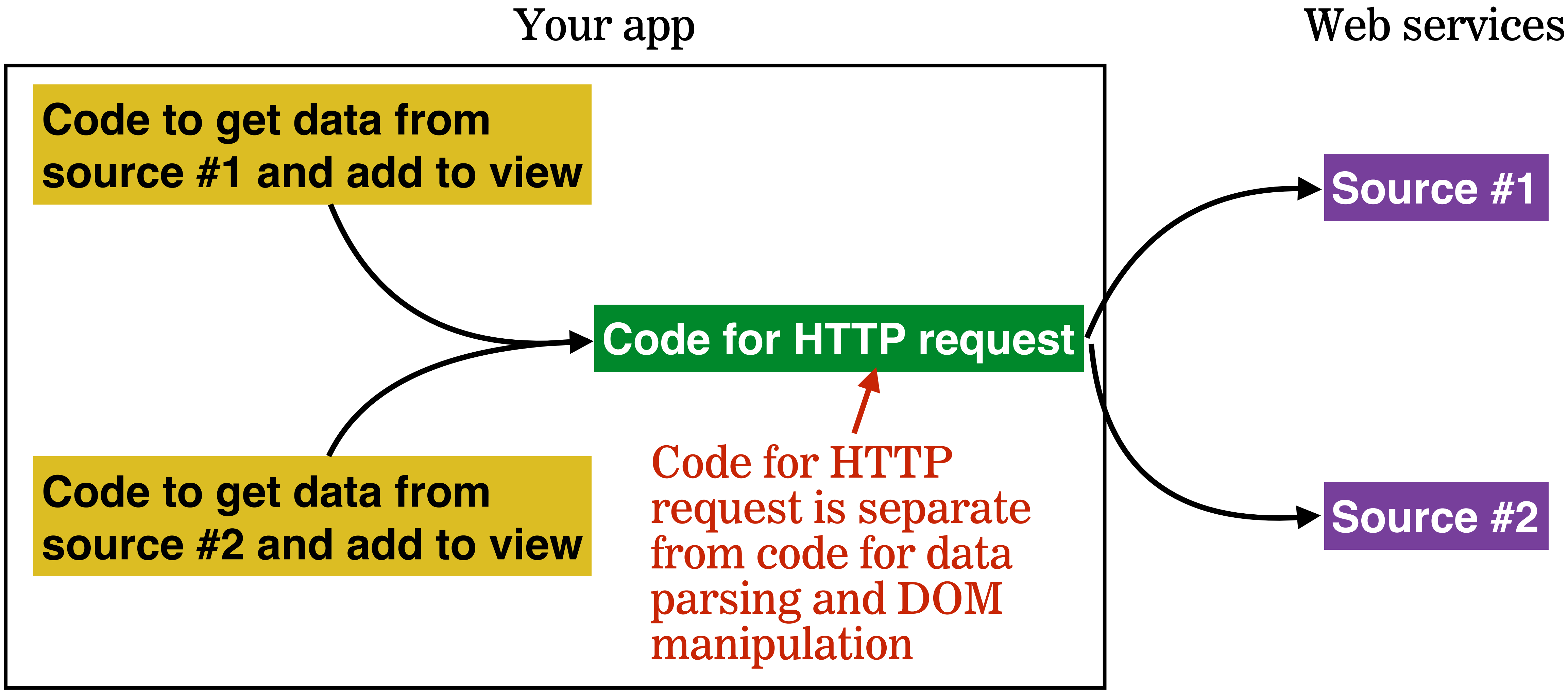
parts of code
call each
other, but are
maintained
separately

✓

LET'S TAKE A CLOSER LOOK



CREATING DRY CODE FOR HTTP REQUESTS


Your app
Web services
Code to get data from source #1 and add to view
Code to get data from source #2 and add to view
Code for HTTP request
Source #1
Source #2

Code for HTTP request is separate from code for data parsing and DOM manipulation

LAB — JQUERY AJAX

OBJECTIVE

- Create an Ajax request using jQuery.

LOCATION

- `starter-code > 4-jquery-ajax-exercise`

EXECUTION

until 9:20

1. Open index.html in your editor and familiarize yourself with the structure and contents of the file.
2. Open main.js in your editor and follow the instructions.

Exit Tickets!

(Class #10)

LEARNING OBJECTIVES – REVIEW

- Use event delegation to manage dynamic content in jQuery.
- Identify all the HTTP verbs & their uses.
- Describe APIs and how to make calls and consume API data.
- Access public APIs and get information back.
- Implement an Ajax request with vanilla JS.
- Create an Ajax request using jQuery.
- Reiterate the benefits of separation of concerns – API vs. Client.

NEXT CLASS PREVIEW

Asynchronous JavaScript and Callbacks

- Pass functions as arguments to functions that expect them.
- Write functions that take other functions as arguments.
- Return functions from functions.

Q&A