Working with APIs

Goals

- Define what an API is
- Compare and contrast different kinds of APIs
- Understand the limitations
- Use Terminal and GUI clients for making HTTP requests

APIs

What Is An API?

A set of clearly defined methods of communication between various components.

An API may be for a web-based system, operating system, database system, computer hardware, or software library.

APIs You Have Used

Web APIs https://developer.mozilla.org/en-US/docs/Web/API <https://developer.mozilla.org/en-US/docs/Web/API>

The jQuery API https://api.jquery.com/ <https://api.jquery.com/>

Bootstrap API https://getbootstrap.com/docs/4.1/getting-started/javascript/ <https://getbootstrap.com/docs/4.1/getting-started/javascript/>

Third Party APIs

Companies will provide access to their data (sometimes not for free)

- Twitter API, give me all tweets that mention “ice cream”
- Facebook API, send me the current user’s profile picture
- Weather API, what is the weather in Missoula Montana?
- Reddit API, what is the current top post?
- GooglePlaces API, what gas stations are near the user?
- Yelp API, give me 10 restaurants in the zipcode 94110

Data Formats

- When we browse on the web, we make HTTP requests and get HTML back.
- APIs don't respond with HTML.
  - HTML contains info about page structure. APIs respond with data, not structure.
- They use different data formats like XML and JSON.
  - These are still text based formats—remember, HTTP is text based!

XML

Syntactically similar to HTML, but does not describe presentation like HTML, and many of the tags are custom.

```
<person>
  <name>Elie</name>
  <favoriteColor>purple</favoriteColor>
  <city>San Francisco</city>
</person>
```

JSON

JSON stands for **JavaScript Object Notation**.

JSON looks similar to JS objects, but all the keys must be "double-quoted".

```
{
  "person": {
    "name": "Elie",
    "favoriteColor": "purple",
    "city": "San Francisco",
    "favoriteNumber": -97,
    "interests": ["CEOing", "eating Mediterranean food"],
    "futureDreams": null
  }
}
```

A JSON payload must be sent as a string over HTTP requests.

To convert JavaScript object to JSON string:

```
JSON.stringify(myObject)   // "...string of JSON..."
```

To convert JSON string to JavaScript object:

```
JSON.parse(jsonString)   // {prop: value, ...}
```

Most libraries do this for you.

JSON vs XML

We'll primarily use JSON: it's easier to parse & works great with JavaScript!

JSON is also the contemporary standard for most RESTful APIs.

API Security

AJAX & Same Origin Policy

Many APIs can be used with AJAX

Some cannot unless the JS app is from the "same origin"

This is to prevent subtle security issues.

Same Origin Policy

- Critical security mechanism that restricts how a document or script loaded from one origin can interact with a resource from another origin.
- It helps to isolate potentially malicious documents, reducing possible attacks
- It is **very** restrictive

What constitutes a "different" origin?

- Different domain
- Different protocol
- Different port

CORS

You can't use AJAX if the API requires the same origin

But the backend API server can opt-in using "CORS"

Curl

curl is used in command lines or scripts to transfer data.

Open source & comes with OSX—so it's easy to use right out of the box

Making a request using Curl

We do it in the Terminal!

Simplest & most common request/operation made using HTTP is to GET a URL:

```
$ curl https://curl.haxx.se
```

This will return the entire HTML document that that URL holds.

```
$ curl https://api.github.com/users/elie
```

This will return a JSON response from the Github API

Flags with Curl

- `-d` or `--data` to send information to a server

  `-d  '{"username":"xyz","password":"xyz"}'`
- `-X` or `--request` to specify HTTP verb (`-X POST`)
- `-H` or `--header` to specify additional headers

  `-H "Content-Type: application/json"`

Example of a larger request

```
curl --header "Content-Type: application/json" \
  --request POST \
  --data '{"username":"xyz","password":"xyz"}' \
  https://myapplication.com/login
```

When to use Curl

- When you are making a simple HTTP(S) request
- When you don't have any other option
- When you're doing scripting
- You will also see it in almost all API documentation for examples

Insomnia

A GUI for making HTTP requests.

https://insomnia.rest/ <https://insomnia.rest/>

Insomnia vs Curl

- You can save previous HTTP requests
- It's easier to write complex HTTP requests with many headers/long data fields

Practicing with Insomnia

If you want extra practice, check out ***https://jsonplaceholder.typicode.com/***