Communicating on the Web - Oct 28

JSON

JavaScript Object Notation is the most used for server to client communication.

The media type used for JSON is **application/json**.

JSON is structured with key and value pairs.

- Number - `{"age" : 34}`
- String - `{"name" : "Joseph"}`
- Array - `{"fruits" : ["Apple","Banana","Strawberry"]}`
- Boolean - `{"Enable" : true}`
- Null - `{"genre" : null}`
- Object
  `{"user" : {"name" : "Joseph", "age" : 34, "genre" : null}}`

Example
Build a JSON file for the data below (hint: think of this as a user object).

Name	Id	Telephone number	Hobbies	Lessons
Christopher	7464839	+1457938740	Read, Sing, Dance	Name:HTML, ID:1; Name:Javascript, ID:2; Name:API; ID:3

`{"name" : "Christopher", "Id" : 7464839, "Telephone number" : “+1457938740”, “Hobbies” : [“Read, Sing, Dance”], “Lessons” : [{“Name” : “HTML”, “ID”:1}, {“Name” : “Javascript”, “ID”:2}, {“Name” : “API”, “ID” : 3}]}`

Accessing APIs with fetch()

The fetch() method is used to request data from a server for any type of API that returns the data in JSON or XML format.

```
let result = fetch("https://www.boredapi.com/api/activity");
result.then(res => res.json())
      .then(d => { console.log(d)});
```

Fetching data with options

```
user = {
   "name": "Christopher Miller",
   "age": "35",
   "salary": "4321"};

let options = {
   method: 'POST',
   headers: {'Content-Type':'application/json;charset=utf-8'},
   body: JSON.stringify(user)};

let result = fetch("https://dummy.restapiexample.com/api/v1/create", options);
   result.then(res => res.json())
   .then(d => {console.log(d)});
```

Exercise 2

Complete the code below using this API and embed the id “imageAPI” in the tag.

```
let response = fetch("https://dog.ceo/api/breeds/image/random");
    response.then(res => res.json())
            .then(d => {document.getElementById("imageAPI").src =
d.message});
```

Leaflet

Leaflet is an open-source JavaScript library for interactive maps.
Documentation: https://leafletjs.com/reference.html

Example:

```
const map = L.map('map').setView([43.096214, -79.037743], 13);
L.tileLayer('http://{s}.google.com/vt/lyrs=m&x={x}&y={y}&z={z}', {
maxZoom: 20, subdomains: ['mt0', 'mt1', 'mt2', 'mt3'], attribution:
'Copyright Google Maps' }).addTo(map);
```

Most tile providers (ex OpenStreetMap, Google Maps) require an attribution. Check the copyright notice of the tiles provider before publishing your page in production.

Markers, Circles & Polygons

```
//marker
var marker = L.marker([43.096214, -79.037743]).addTo(map);

//circle
var circle = L.circle([43.08, -79.08], {radius: 600}).addTo(map);

// create a orange polygon from an array of LatLng points
var latlngs = [[43.09, -79.07],[43.08, -79.06],[43.1, -79.05]];
var polygon = L.polygon(latlngs, {color: 'orage', fillOpacity: 0.8,
weight: 6}).addTo(map);
```

Popups

```
marker.bindPopup("Testing a popup.").openPopup();
circle.bindPopup("<b>Testing!</b>I am a circle.");
polygon.bindPopup("Testing a polygon.");
```

Events

```
function onMapClick(e) { alert("You clicked the map at " + e.latlng); }
map.on('click', onMapClick);
```

Google Maps API

Google maps has extended functionality, so for large projects with heavy mapping needs, Google maps could offer more solutions than Leaflet.
Documentation: https://developers.google.com/maps/documentation

Start by setting up a project in Google cloud.
You can then enable the **Maps Static API** on your project.
You do need a credit card to activate billing, so if you do this for a customer, use the customer's google account and the customer's credit card.

API Keys

APIs & Services > Credentials

You should **restrict your API Keys** to protect them from unwanted requests.

> The in-class exercise has you restrict the key to your own IP address, but when launching a web project, you want to restrict the key to the domain name that will be running the API. Alternatively, you may be able to restrict the key to the fixed IP address of the server that is hosting the website.

Script on your webpage to call the APIs attached to your API Key:

In the header:

```
<script
src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=ini
tMap&v=weekly" defer></script>
```

Loading a Map on your page

```
<script>
function initMap() { const coordinates = { lat: 40.689, lng: -74.044 };
const map = new google.maps.Map(document.getElementById("map"), { zoom:
13, center: coordinates, });

//set the marker
const marker = new google.maps.Marker({ position: coordinates, map: map,
}); }

window.initMap = initMap;
</script>
```

Markers, Circles & Polygons

```
//set the marker
const marker = new google.maps.Marker({ position: coordinates, map: map,
}); }

//set the circle
const circle = new google.maps.Circle({
      strokeColor: "blue",
      strokeOpacity: 0.8,
      strokeWeight: 2,
      fillColor: "#FFF",
      fillOpacity: 0.5, map,
      center: coordinates,
      radius: 600, });

//set the rectangle
const rectangle = new google.maps.Rectangle({
    strokeColor: "#FF0000",
    strokeOpacity: 0.8,
    strokeWeight: 2,
    fillColor: "#FF0000",
    fillOpacity: 0.35,
    map,
    bounds: {
      north: 37.429,
      south: 37.415,
      east: -122.073,
      west: -122.091,
    },
  });
```

Events

```
// inside the initMap() fucntion
map.addListener("click", (e) => { alert("You clicked the map at " +
JSON.stringify(e.latLng.toJSON(), null, 2)); });
```

Geocoding services

convert addresses into geographic coordinates

Structure

```
{ address: string, location: LatLng, placeId: string, bounds:
LatLngBounds, componentRestrictions: GeocoderComponentRestrictions,
region: string }
```

Geocoder Result Structure

```
results[]: { types[]: string, formatted_address: string,
address_components[]: { short_name: string, long_name: string,
postcode_localities[]: string, types[]: string }, partial_match: boolean,
place_id: string, postcode_localities[]: string, geometry: { location:
LatLng, location_type: GeocoderLocationType viewport: LatLngBounds,
bounds: LatLngBounds } }
```

Status codes:

- “OK”
- “ZERO_RESULTS”
- “OVER_QUERY_LIMIT”
- “REQUEST_DENIED”
- “INVALID_REQUEST”
- “UNKNOWN_ERROR”
- “ERROR”

Example

```
var geocoder; var map;

function initMap() {
      geocoder = new google.maps.Geocoder();
      const coordinates = { lat: 37.422040, lng: -122.082810 };
      map = new google.maps.Map(document.getElementById("map"), { zoom:
13, center: coordinates, }); }

function getCoordinates() {
      var address = document.getElementById('address').value;
      geocoder.geocode({ 'address': address }, function (results, status)
{ if (status == 'OK') { map.setCenter(results[0].geometry.location); var
marker = new google.maps.Marker({ map: map, position:
results[0].geometry.location }); } else { alert('Geocode was not
successful for the following reason: ' + status); } }); }
```

Directions Services

Required parameters are origin, destination and travelMode.

Travel Modes:

- DRIVING (Default)
- BICYCLING
- TRANSIT
- WALKING

Status Codes:

- OK
- NOT_FOUND
- ZERO_RESULTS
- MAX_WAYPOINTS_EXCEEDED
- MAX_ROUTE_LENGTH_EXCEEDED
- INVALID_REQUEST
- OVER_QUERY_LIMIT
- REQUEST_DENIED
- UNKNOWN_ERROR

Example on page 4 of **communicating-on-the-web--lesson-3-2---sat---oct-28**

```
<head><title>Hello Google Maps</title></head>
<body>
<div id="map" style="width: 600px; height: 400px;"></div>
    <div>
        <label>Origin</label>
        <input id="origin" type="textbox">
        <label>Destination</label>
        <input id="destination" type="textbox">
        <input type="button" value="Search Route" onclick="calcRoute()">
    </div>
<script
src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap&v=weekly" defer></script>
<script>
        var directionsService;
        var directionsRenderer;
        var map;
```

```
        // Initialize and add the map
        function initMap() {
            directionsService = new google.maps.DirectionsService();
            directionsRenderer = new google.maps.DirectionsRenderer();
            const coordinates = { lat: 37.422040, lng: -122.082810 };

            map = new google.maps.Map(document.getElementById("map"), {
                zoom: 13,
                center: coordinates,
            });
            directionsRenderer.setMap(map);
        }

        function calcRoute() {
            var start = document.getElementById('origin').value;
            var end = document.getElementById('destination').value;
            var request = {
                origin: start,
                destination: end,
                travelMode: 'DRIVING'
            };
            directionsService.route(request, function (result, status) {
                if (status == 'OK') {
                    directionsRenderer.setDirections(result);
                } else { alert("An unexpected error occurred")}
            });
        }
        </script>
        </body>
</html>
```