Web Advanced: Javascript APIs

“We will learn JavaScript properly. Then, we will learn useful design patterns. Then we will pick up useful tools to understand the modern world of coding.”

FALL 2022

HELLO.

jaink@newschool.edu

https://canvas.newschool.edu/courses/1661668

https://replit.com/@jaink/pgte-5505-f22

https://NewSchool.zoom.us/j/91939750510?pwd=dE5tM1dzeUlpelNlQTJYUUVBY003UT09

https://github.com/kujain/F22-5505_Javascript

INTRODUCTIONS

Why Learn Coding?

DON'T LEARN

- Learning curve/mental blockade.
- Too much specialization.
- Apps available to helps auto-generate code.
- Coding standards and patterns are constantly evolving.

DO LEARN

- Better understanding of the process needed to build.
- Better understanding of limits.
- Create more efficient and responsible design.

QUIZ

What does a Program look like?

➔ Let’s Compare Code written in different languages...

MACHINE LANGUAGE

```
01001000 01100101 01101100
01101100 01101111 00100000
01010111 01101111 01110010
01101100 01100100
```

OUTPUTS: **HELLO WORLD**

C++

```cpp
#include <iostream>
using namespace std;

int main()  {
   float length, width, area;
   cout << "Enter The Length: ";
   cin >> length;
   cout << "Enter The Width: ";
   cin >> width;
   area = length*width;
   cout <<"Answer is : "<< area << endl;
   return 0;
}
```

JAVA

```
public static int fctl(int n)
{  int result = 1;
      for(int i = 2; i <= n; i++)
         result *= i;
      return result;
}


factl(10)
```

PHP

```php
<?php

class Vegetable {
   var $veg;
   var $color;

   function __construct($veg, $color="green") {
       $this->veg = $veg;
       $this->color = $color;
   }

   function get_name() {
       return $this->veg;
   }

   function what_color() {
       return $this->color;
   }
} // end of class Vegetable

$Veg = new Vegetable( "tomato", "red");
echo $Veg->get_name() . " is " . $Veg->what_color();

?>
```

P5

```
function setup() {
  let d = 70;
  let p1 = d;
  let p2 = p1 + d;
  let p3 = p2 + d;
  let p4 = p3 + d;

  createCanvas(720, 400);
  background(0);
  noSmooth();

  translate(140, 0);

  // Draw
  stroke(150);
  line(p3, p3, p2, p3);
  line(p2, p3, p2, p2);
  line(p2, p2, p3, p2);
  line(p3, p2, p3, p3);
}
```

RUBY

```ruby
items = [ 'Mark', 12, 'goobers', 18.45 ]
for stuff in items
    print stuff, " "
end

print "\n"
```

JAVASCRIPT

```
let score = 75;          // Score
let msg;                 // Message

if (score >= 50) {
  msg = 'Congratulations!';
  msg += ' Proceed to the next round.';
  let el =
document.getElementById('answer');
  el.textContent = msg;
}

<div class="var"
id="answer">'Congratulations….</div>
```

Why Javascript?

In the Beginning…

Mocha? Java?

The Browser Wars

The AJAX revolution

The Standards War

Beyond the Browser

Javascript...Python...C#...R

What Can Javascript do?

Generative
http://color-wander.surge.sh/

Practical
https://usecubes.com/design

Informative
http://www.histography.io/

Apps
http://ubereats.com

https://www.facebook.com/

Entertainment
https://www.netflix.com/

3D
http://alteredqualia.com/three/examples/webgl_city.html

Quick List of Features

- Written to enable both-way interaction in web browsers
- Interpretive: compiled at runtime
- Always backward-compatible by design
- Loose type declaration: makes it flexible and confusing at the same time
- Has functions that can be used as variable objects
- Allows both functional and object-oriented programming
- Many ways to approach asynchronous events
- Many ways to use design patterns
- Many popular frameworks: jQuery, Angular, Vue, React
- Isomorphic - can be used in frontend and servers

Syllabus

- ➔ Syntax and Constructs
- ➔ Document Object Model
- ➔ Forms and AJAX
- ➔ Classes and Object Oriented Programming
- ➔ Functional Programming
- ➔ Modules and DevOps
- ➔ Web/HTML APIs
- ➔ DevOps Workflows
- ➔ Advanced: Frameworks(Vue/React)
- ➔ JS in the Backend: Nodejs
- ➔ Final Project Development

Tools of the Trade

➔ Text Editors

Sublime Text:	https://www.sublimetext.com/
Atom:	https://atom.io/
MS Visual Studio	https://visualstudio.microsoft.com/vs/mac/
Chrome DevTools:	https://developer.chrome.com/devtools

and more…

➔ Browsers (latest versions)

Chrome:	https://www.google.com/chrome/
Firefox:	https://www.mozilla.org/en-US/firefox/
Safari:	OSX only

➔ Debugger & Tools

Built in Browser Developer Console (Fn + F12)

Patterns Reference: https://jstherightway.org/

➔ Automators

NPM, Babel, Gulp
(will be discussed during DevOps session)

Creating a Basic HTML Template

https://replit.com/@jaink/pgte-5501-f22

```
<!DOCTYPE html>

<html>

  <head>
    <meta charset="utf-8">
    <meta name="viewport"
content="width=device-width">

    <title>The Parsons Web Project</title>
    <meta name="description" content="Fall 2022
Class">
    <meta name="author" content="Parsons Faculty">
    <link rel="stylesheet" href="css/styles.css">
  </head>

  <body>

    <header>This is the header</header>

    <section>This is Section 1</section>

    <section>
      <button id="button">Click me</button>
    </section>

    <!-- script always before closing body tag -->
    <script src="js/scripts.js"></script>
  </body>

</html>
```

Our First Javascript Code

➔ **Hello World!**

```
console.log(‘Hello’);
```

➔ **Using vars with Hello World!**

```
let greeting_container;
// assign greeting to variable
greeting_container = “Hello”;
console.log(greeting_container);
```

➔ **Generate an Alert**

```
alert(‘Greetings ’ +
greeting_container);
```

➔ **Update the Document**

```
document.write('<p>’ +
greeting_container + ’</p>');
```

Inline vs External

➔ **INLINE:**

```
  <body>

    <header>This is the header</header>

    <section>This is Section 1</section>

    <section>
      <button id="button">Click me</button>
    </section>

    <!-- script always before closing body tag -->
    <script>
         console.log('Hello');
         // more stuff
   </script>
  </body>
</html>
```

➔ **EXTERNAL:**

```
     </section>

    <!-- script always before closing body tag -->
    <script src="js/scripts.js"></script>
  </body>
```

Our Second Javascript Code

➔ **Event Listener**

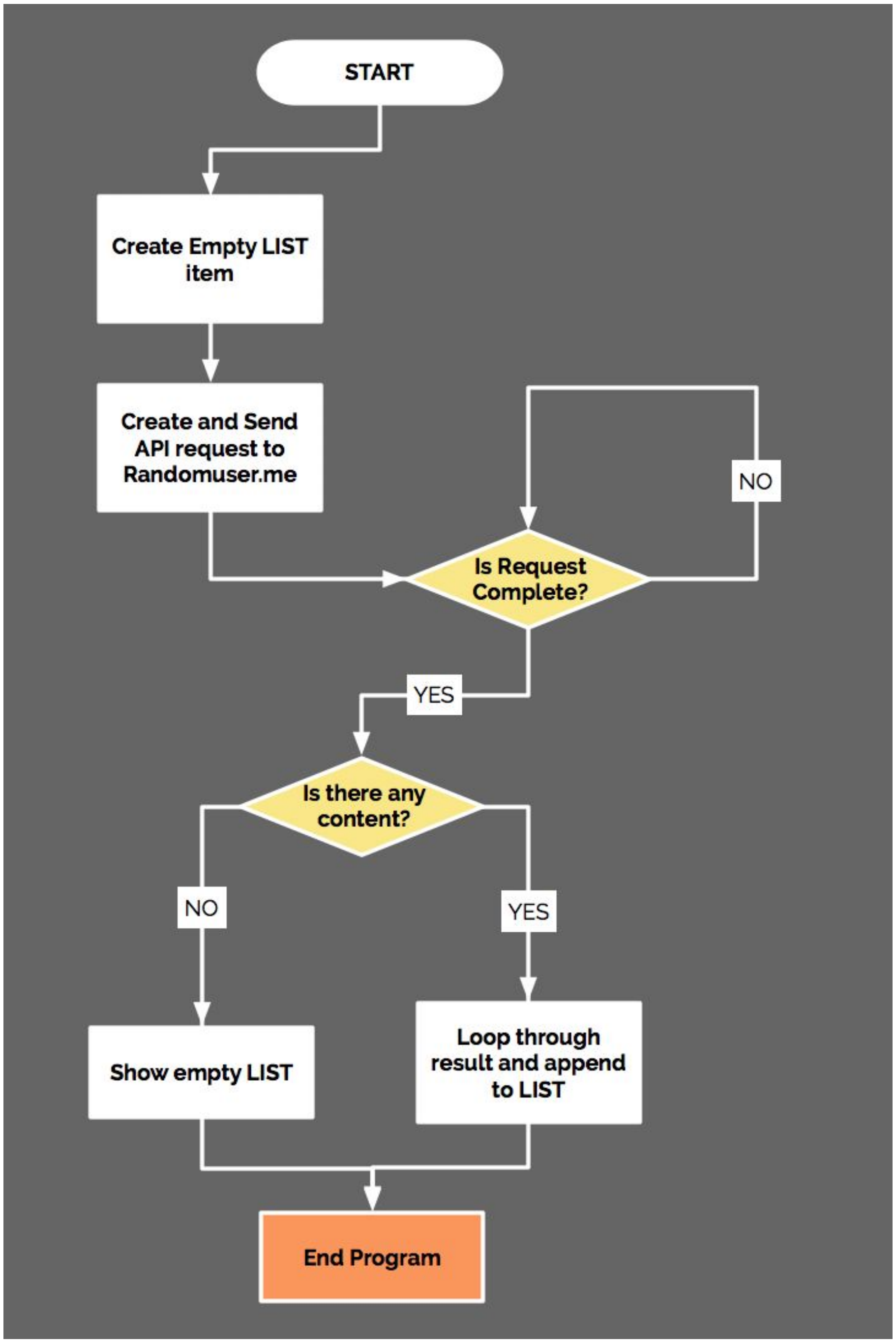
```
/* event listener to change body
background */

const btn =
document.getElementById('button');


const rainbow =
['red','orange','yellow','green','blue','
rebeccapurple','violet'];

function change() {

  document.body.style.background =
rainbow[Math.floor(7*Math.random())];

}

btn.addEventListener('click', change);
```

Our Third Javascript Code

➔ DOM Manipulation

```javascript
/* Simple DOM Manipulation example */
const now = new Date();
const hours = now.getHours();

document.write(`It's now: ${hours}. <br><br>`);
let bgColor = "lightorange";

if (hours > 17 && hours < 20){
  bgColor = "orange";
}
else if (hours > 19 && hours < 22){
  bgColor = "orangered";
}
else if (hours > 21 || hours < 5){
  bgColor = "#C0C0C0";
}
else if (hours > 8 && hours < 18){
  bgColor = "lightblue";
}
else if (hours > 6 && hours < 9){
  bgColor = "skyblue";
}
else if (hours > 4 && hours < 7){
  bgColor = "steelblue";
}
else {
  bgColor = "white";
}

document.body.style.backgroundColor = bgColor;
```

Our 4th Javascript task - flow



Our 4th Javascript Code

➔ Connect with API using AJAX

➔ `API endpoint: https://randomuser.me`

```
const ul = document.createElement('ul');
const url = 'https://randomuser.me/api/?results=10';
const xhr = new XMLHttpRequest();
xhr.onerror = function() { // only triggers on error
     alert(`Oops - we cannot not do this!`);
};
xhr.onload = function() {
     if (xhr.status == 200) {
        let authors = JSON.parse(xhr.responseText); // Get
results

        for (key in authors.results) { // loop through the
results
          let author = authors.results[key]; //assign current row
to author var
          let li = document.createElement('li'), //  Create the
elements we need
              img = document.createElement('img'),
              span = document.createElement('span');
          img.src = author.picture.medium;  // Add the source of
the image to be the src of the img element
          span.innerHTML = author.name.first + ' ' +
author.name.last; // Make the HTML of our span to be the first
and last name of our author
          li.appendChild(img); // Append img element back to
containing li
          li.appendChild(span); // Append span element back to
containing li
          ul.appendChild(li); // Append li element back to
containing ul
          document.body.append(ul); //Append the new ul to body
        }
    }
}

xhr.open('GET', url, true);
xhr.send(null);
```

Our 4th Javascript Code (alternative)

➔ Connect with API using Fetch API

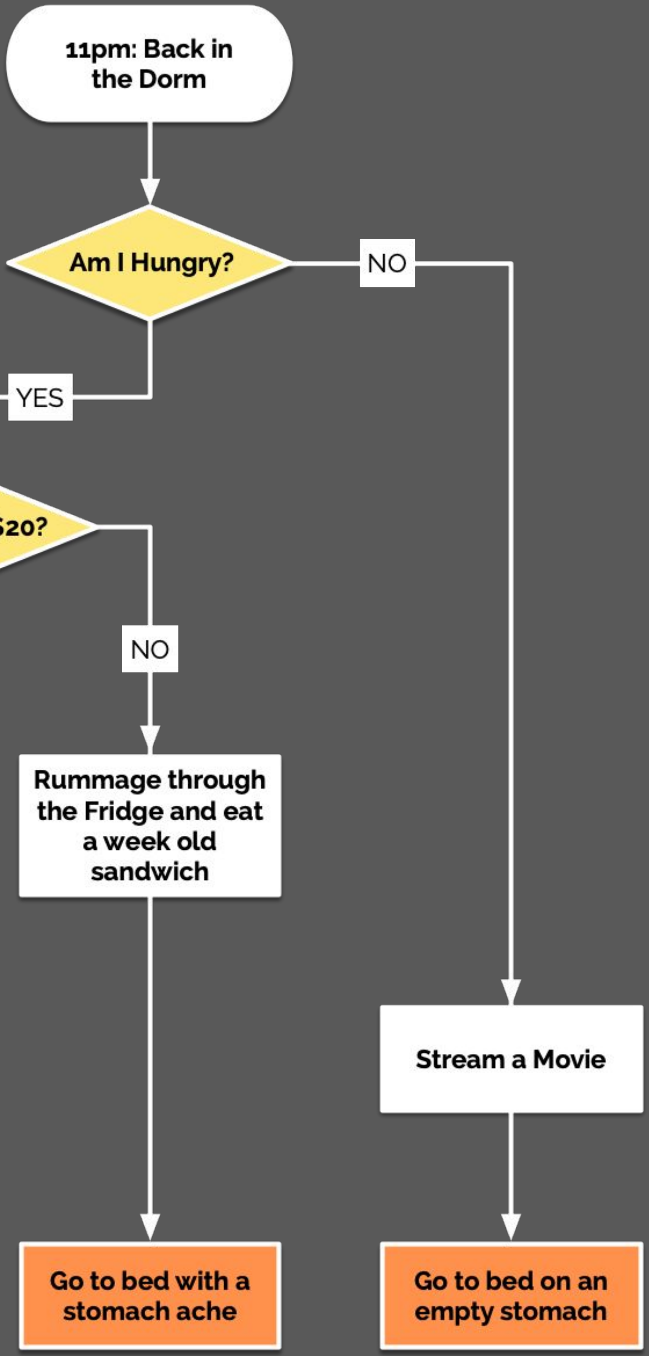
API endpoint: https://randomuser.me

```
const ul = document.createElement('ul');
const url = 'https://randomuser.me/api/?results=10';

fetch(url)
    .then((resp) => resp.json())
    .then(function(data) {

        console.log(data);

        let authors = data.results; // Get the results
        authors.forEach(function(author) { // Map through the
results and for each run the code below
          let li = document.createElement('li'), //  Create the
elements we need
              img = document.createElement('img'),
              span = document.createElement('span');
          img.src = author.picture.medium;  // Add the source of
the image to be the src of the img element
          span.innerHTML = `${author.name.first}
${author.name.last}`; // Make the HTML of our span to be the
first and last name of our author
          li.appendChild(img); // Append all our elements
          li.appendChild(span);
          ul.appendChild(li);
        })

        document.body.append(ul);
    })
    .catch(function(error) {
        console.log(error);
    });
```

Assignment: Decision Trees

Find a regular online or physical activity or task that requires some interaction and decision-making. Write the process flow out in detail.

eg. Dining Options back in the Dorm:



Next Class

- ➔ **Javascript Structure**
- ➔ **Javascript Syntax:**

  Data types: strings, numbers, variables, arrays

  Operators

  Conditional logic

  Loops