Upright Education

Programming Libraries In JavaScript

Software Development Bootcamp

Topic

JavaScript Libraries

What Are JavaScript Libraries?

JavaScript libraries are pre-written collections of JavaScript code that provide useful functions and tools to simplify development tasks.

Why Use JavaScript Libraries?

- **Reusable Code:** Libraries contain pre-written JS functions, methods and objects that you can use
- **Abstraction:** Libraries often abstract complex operations into simpler more manageable functions
- **Modularity:** Many modern JS libraries are modular, allowing you to import only the parts you need
- **Cross-browser Compatibility:** Libraries often handle browser inconsistencies, ensuring your code works across different browsers.

Examples Of Popular JS Libraries

- **React:** A library for building user interfaces with reusable components
- **Lodash:** Provides utility functions for common programming tasks
- **Moment.js:** Simplifies parsing, validating, manipulating and displaying dates and times
- **Axios:** A promise-based HTTP client for making API requests

How To Use Libraries

There are several ways to incorporate and use JavaScript libraries in your projects. The method you choose often depends on your development environment and project requirements.

- Today we'll practice by using the Browser-based approach to adding a library
- In the next unit we'll learn about adding libraries using NPM

Browser-based Approach CDN

- Use a Content Delivery Network (CDN) to include the library directly in your HTML file.
- Find the CDN link for your desired library and version. You can usually find this on the library's website
- Add a **`<script>`** tag in your HTML file with the CDN link
- Always include the library before your own scripts that use it.
- Quick and easy for small projects and prototypes

Order of Scripts

- HTML renders top to bottom, so script tags are processed in order.
- A script imported later has access to global variables from earlier files.
- Internal scripts go at the bottom of the body; external scripts in the head.

Topic

Using The LeafletJS Library For Mapping

Web Mapping

- Creating Maps is a complex process, so developers often use libraries to simplify the development process

- Today, we'll use LeafletJS, an open-source alternative to Google Maps.

What Is LeafletJS?

LeafletJS is an open-source JavaScript library for mobile-friendly interactive maps. It's designed with simplicity, performance, and usability in mind. LeafletJS allows developers to quickly create web mapping applications with just a few lines of code.

- Leaflet Documentation

Key Features Of LeafletJS

- **Lightweight**: The core of Leaflet is about 39 KB of JS, making it much smaller than many other mapping libraries.
- **Mobile-friendly**: Works well on both desktop and mobile platforms.
- **Extensive plugin ecosystem**: Offers a wide range of plugins to extend functionality.
- **Easy to use**: Has a simple, readable API and well-documented code.
- **Customizable**: Allows for extensive customization of map features and appearance.

When To Use LeafletJS

- Building web applications that require interactive maps
- Creating data visualizations with geographical components
- Developing location-based services or applications
- When you need a lightweight, mobile-friendly mapping solution

Importing LeafletJS

Add these tags to your HTML's head:

- <link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" integrity="sha512-xodZBNTC5n17Xt2atTPuE1HxjVMSvLVW9ocqUKLsCC5CXdbqCmblAshOMAS6/keqq/sMZMZ19scR4PsZChSR7A==" crossorigin=""/>
- <script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js" integrity="sha512-XQoYMqMTK8LvdxXYG3nZ448hOEQiglfqkJs1NOQV44cWnUrBc8PkAOcXy20w0vIaXaVUearIOBhiXZ5V3ynxwA==" crossorigin=""></script>

- Add these tags to your HTML's head:

```
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- Add link to your style.css file -->
  <link rel="stylesheet" href="style.css" />
  <!-- CSS link from leaflet -->
  <link
    rel="stylesheet"
    href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css"
    integrity="sha512-xodZBNTC5n17Xt2atTPuE1HxjVMSvLVW9ocqUKLsCC5CXdbqCmblAshOMAS6/keqq/sMZMZ19scR4PsZChSR7A=="
    crossorigin=""
  />
  <!-- JS link from leaflet -->
  <script
    src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"
    integrity="sha512-XQoYMqMTK8LvdxXYG3nZ448hOEQiglfqkJs1NOQV44cWnUrBc8PkAOcXy20w0vlaXaVUearIOBhiXZ5V3ynxwA=="
    crossorigin=""
  ></script>
  <title>Document</title>
</head>
```

Importing LeafletJS

The `L` Object

The `L` object is the core namespace in LeafletJS. It contains all of Leaflet's classes, methods, and properties. When you include Leaflet in your project, the `L` object becomes globally available, allowing you to access all of Leaflet's functionality.

- Documentation

Upright Education

Creating the Map

- **Need:** a container in HTML, the L object in JS, and a tileset.

index.html

```html
<body>
   <!-- Container for the map -->
   <div id="map"></div>
   <!-- Script tag to link our index.js file -->
   <script src="index.js"></script>
 </body>
```

style.css

```css
/* Set the height and width of the map container
*/
#map {
   height: 500px;
   width: 500px;
}
```

index.js

```javascript
// Setting up map view
let mymap = L.map("map").setView([51.505, -0.09], 13);
// Setting the particular tile
L.tileLayer("https://tile.openstreetmap.org/{z}/{x}/{y}.png", {
 maxZoom: 19,
 attribution:
   '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>',
}).addTo(mymap);
```

Adding Markers

- Markers add points to the map.
- Set marker text using .bindPopup:

```
// adding marker to the map
let marker = L.marker([51.5,
-0.09]).addTo(mymap);
// using bindPopup to add set marker text
marker.bindPopup("<b>Hello World</b><br>I
am a popup").openPopup();
// .remove() method removes the marker
from the map
marker.remove()
```

Upright Education

Adding Polygons

- Polygons are shaped regions on the map

```
// Creating polygon
let polygon = L.polygon(
 [
   [51.509, -0.08],
   [51.503, -0.06],
   [51.51, -0.047],
 ],
 { color: "red" }.addTo(mymap)
);
// .remove() removes polygon
polygon.remove()
```

Polyline

- A polyline is a line between two or more points:

```
// Adding line to map
let line = L.polyline([
 [45.51, -122.68],
 [37.77, -122.43],
 [34.04, -118.2],
]).addTo(mymap);
// removing line from map
line.remove()
```