JMAPS.

BY

Jashan

# About Project

Jmaps is a Navigation website which is inspired by Google maps. JMaps is made using vanilla Javascript and CSS. Jmaps uses Material UI for its UI components and some Libraries like MapBox ( For displaying Map and getting routes data from its API ) & RXJS for Data flow throughout all HTML elements.

JMAPS uses **DOM manipulation, Classes, EventListeners, Class Toggling & DOM property changing.**

I used CSS variables in the project which makes it easier for me to implement Dark mode in JMaps.

I tried to divide my project into modules and each module serves its own purpose. There are 4 javascript files in this project. initMaps.js (To serve features of MapBox ), location.js (To serve Algola location to search address), dom.js (To serve dom manipulation ) and root.js (It connects all of the modules together and connects links itself with index.html).

JMaps is Licensed by me - ©Jashanpreet Singh

Jmaps is live on - <https://this-is-jassa.github.io/jMaps/>

-CODE-

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="index.css">

*<!-- RXJX -->*

<script src="https://unpkg.com/@reactivex/rxjs@5.1.0/dist/global/Rx.js"></script>

*<!-- MAPS -->*

<link href='https://api.mapbox.com/mapbox-gl-js/v1.10.1/mapbox-gl.css' rel='stylesheet' />

<script src='https://api.mapbox.com/mapbox-gl-js/v1.10.1/mapbox-gl.js'></script>

<script src="https://cdn.jsdelivr.net/npm/places.js@1.19.0"></script>

*<!-- Material UI -->*

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

<link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.indigo-pink.min.css">

<script defer src="https://code.getmdl.io/1.3.0/material.min.js"></script>

<script type="module" defer src="js/root.js"></script>

<title>JMaps</title>

</head>

<body id="body" class="dark">

<div id="map">

</div>

<div class="demo-card-wide mdl-card mdl-shadow--2dp card" id="sidebar">

<div id="sidebarWrapper">

<button id="navBtn" class="mdl-button mdl-js-button mdl-button--fab mdl-button--mini-fab mdl-button--colored">

<i class="material-icons">arrow\_back</i>

</button>

<label class="mdl-switch mdl-js-switch mdl-js-ripple-effect" for="themeToggle">

<input type="checkbox" id="themeToggle" class="mdl-switch\_\_input" checked>

<span class="mdl-switch\_\_label" style="color: var(--color);">Dark</span>

</label>

<h2 id="heading">JMaps.</h2>

<form id="searchForm">

<input class="input" type="search" id="address-input1" placeholder="Starting Point" required />

<input class="input" type="search" id="address-input2" placeholder="Destination" required />

*<!-- FAB button -->*

<div style="text-align: center;">

<button id="searchBtn" type="submit"

class="mdl-button mdl-js-button mdl-button--raised mdl-js-ripple-effect">

<i class="material-icons">near\_me</i>

Navigate

</button>

</div>

*<!-- <button id="searchBtn" type="submit" class="btn btn-primary btn-sm mb-3 px-3">Search</button> -->*

</form>

<button id="reset" class="mdl-button mdl-js-button mdl-button--raised mdl-js-ripple-effect">

<i class="material-icons">delete\_outline</i>

</button>

</div>

</div>

<div id="navigate">

<div id="progressBar"></div>

<div class="demo-card-wide mdl-card mdl-shadow--2dp"

style="position: relative; background-color: var(--primary); color: var(--color);">

<div class="mdl-card\_\_title">

<h2 class="mdl-card\_\_title-text" style="color: var(--color);"><span id="title"> </span> <i id="sign"

class="material-icons"></i></h2>

</div>

<div style="border-radius: 10px; background-color: black; color: white; margin-top: 7px; position: absolute; bottom: 0;"

class="mdl-card\_\_supporting-text" id="discription">

</div>

</div>

</div>

</body>

</html>

CSS

*@import* url('https://fonts.googleapis.com/css2?family=Nunito:wght@400;600&display=swap');

:root{

--sidebar\_width: 500px

}

.light{

--primary: rgb(243, 243, 243);

--danger: rgb(119, 46, 46);

--cprimary: rgb(50, 110, 201);

--color: #000000;

}

.dark {

--primary: rgb(24, 24, 24);

--danger: rgb(119, 46, 46);

--cprimary: rgb(50, 110, 201);

--color: #ffff;

}

body{

width: 100vw;

height: 100vh;

letter-spacing: 0.6px;

}

.input{

font-family: 'Nunito', sans-serif;

}

#heading {

text-align: center;

color: var(--color);

font-family: 'Nunito', sans-serif;

font-weight: bold;

}

#sidebarWrapper{

overflow: visible;

position: relative;

}

#navBtn {

position: absolute;

right: -30px;

top: 0;

*/\* transform: rotate(180deg); \*/*

}

#map {

z-index: 0;

position: absolute;

bottom: 0;

width: 100vw;

height: 100vh;

}

#sidebar{

z-index: 0;

position: absolute;

height: auto;

overflow: visible;

left: calc( (-1 \* var(--sidebar\_width)) + 30px );

width: var(--sidebar\_width);

background-color: var(--primary);

transition: left 0.2s;

}

#navBtn {

transition: 0.7s;

}

#sidebar:hover{

left: 0px;

}

#sidebar:hover #navBtn {

transform: rotate(180deg);

}

#navigate {

position: absolute;

bottom: -200px;

left: 30px;

background-color: var(--primary);

}

#progressBar {

height: 10px;

width: 0%;

background-color: rgb(122, 174, 230);

border-top-left-radius: 4px;

border-top-right-radius: 4px;

}

.card {

background-color: var(--primary);

padding: 10px;

}

#searchBtn {

margin: 15px;

background-color: var(--cprimary);

letter-spacing: 1px;

color: white;

font-family: 'Nunito', sans-serif;

}

#reset {

margin: 10px 200px;

transform: scale(0.8);

background-color: var(--danger);

letter-spacing: 1px;

color: white;

font-family: 'Nunito', sans-serif;

}

input {

margin: 10px 5px;

}

.navigation\_animation{

animation-name: navigate;

animation-iteration-count: infinite;

animation-direction: alternate-reverse;

}

.progressBar\_animation {

animation-name: progress;

animation-iteration-count: infinite;

animation-direction: alternate-reverse;

}

*@keyframes* sidebar {

from { left: calc( (-1 \* var(--sidebar\_width)) + 40px ); }

to{left: calc( (-1 \* var(--sidebar\_width)) + 20px );}

}

*@keyframes* navigate {

0% {bottom: -200px;}

20% {bottom: 20px;}

90% {bottom: 20px;}

100% {bottom: -200px;}

}

*@keyframes* progress {

from {width: 0%;}

to {width: 100%;}

}

Root.js

*// #########################################################*

*// ##################### IMPORTS ##########################*

*// #######################################################*

*import* *InitMap* *from* './initMaps.js';

*import* *Location* *from* './location.js';

*import* *DOM* *from* './dom.js';

*// #########################################################*

*// ##################### Global Variables ##########################*

*// #######################################################*

const $ = (s, o = document) => o.querySelector(s);

let mapObj = new InitMap()

let location = new Location();

let dom = new DOM();

let jmap = mapObj.map;

const initialLocation = {

startLocation: [0, 0],

endLocation: [0, 0],

pointer: [-79.3871, 43.6426], *// Indicated where to fly,*

route: {

draw: false,

coord: {},

steps: [],

duration: 0,

distance: 0

}

}

*// #########################################################*

*// ##################### Observbles ##########################*

*// #######################################################*

let $location = new Rx.BehaviorSubject(initialLocation)

*// #########################################################*

*// ##################### Subscriptions ##########################*

*// #######################################################*

$location.subscribe(res => {

jmap.flyTo(

{

center: [...res.pointer]

});

*if* (res.route.draw) {

mapObj.drawRoute(res.route.coord);

dom.showSteps([...res.route.steps]);

}

});

*// #########################################################*

*// ##################### Events ##########################*

*// #######################################################*

location.input1.on('change', (e) => setLatLong(e, true))

location.input2.on('change', (e) => setLatLong(e, false));

$('#searchForm').addEventListener('submit', searchRoute);

$('#reset').addEventListener('click', reset)

$('#themeToggle').addEventListener('click', changeTheme)

*// #########################################################*

*// ##################### Helper Functions ##########################*

*// #######################################################*

function setLatLong(e, inStartLocation = true) {

const { lat, lng } = e.suggestion.latlng;

let LocationData = { ...$location.getValue() };

(inStartLocation)? LocationData.startLocation = [lng, lat]: LocationData.endLocation = [lng, lat];

LocationData.pointer = [lng, lat];

$location.next({ ...LocationData });

}

function searchRoute(event) {

event.preventDefault();

let LocationData = { ...$location.getValue() };

const newCoords = LocationData.startLocation.join(',') + ';' + LocationData.endLocation.join(',');

getRoute(newCoords);

}

function getRoute(e) {

var url = 'https://api.mapbox.com/directions/v5/mapbox/driving/' + e

+ '?geometries=geojson&steps=true&access\_token=' + mapboxgl.accessToken;

var req = new XMLHttpRequest();

req.responseType = 'json';

req.open('GET', url, true);

req.onload = function () {

var jsonResponse = req.response;

var distance = Math.ceil(jsonResponse.routes[0].distance \* 0.001);

var duration = jsonResponse.routes[0].duration / 60;

var steps = jsonResponse.routes[0].legs[0].steps;

var coords = jsonResponse.routes[0].geometry;

console.log(coords);

console.log(distance)

console.log(duration)

console.log(steps)

let LocationData = { ...$location.getValue() };

LocationData.route.draw = true;

LocationData.route.coord = coords;

LocationData.route.steps.push(...steps);

LocationData.route.duration = duration;

LocationData.route.distance = distance;

LocationData.pointer = [...LocationData.startLocation];

$location.next({ ...LocationData });

};

req.send();

}

function reset() {

let locationData = { ...$location.getValue() }

locationData.route.draw = false;

dom.reset();

mapObj.reset();

$location.next(locationData);

}

function changeTheme() {

const isDark = $('#themeToggle').checked;

dom.changeTheme(isDark);

*if* (isDark) {

*return* jmap.setStyle('mapbox://styles/mapbox/' + 'dark-v10');

}

jmap.setStyle('mapbox://styles/mapbox/' + 'outdoors-v11');

}

Dom.js

*export* *default* class DOM {

*// #########################################################*

*// ##################### Global Variables ##########################*

*// #######################################################*

AnimationTime = 3000;

$dataflow;

$ = (s, o = document) => o.querySelector(s);

steps = [];

directionsIcon = {

left: 'arrow\_back',

right: 'arrow\_forward',

"slight right": 'call\_made',

"slight left": 'arrow\_back'

}

$interval = new Rx.Observable.interval(this.AnimationTime);

constructor() { }

*// #########################################################*

*// ##################### Helper Functions ##########################*

*// #######################################################*

showSteps(steps = []) {

*try* {

this.steps = [...steps];

this.$dataflow.unsubscribe();

this.animateNavigation(false)

} *catch* (err) { }

*finally* {

this.$dataflow = new Rx.BehaviorSubject(steps).switchMap(

() => {

*return* this.$interval

}

).subscribe(value => {

this.animateNavigation(true, this.steps[value].maneuver.instruction, this.steps[value].name, this.directionsIcon[this.steps[value].maneuver.modifier]);

*if* (value === this.steps.length - 1) {

this.animateNavigation(false);

this.$dataflow.unsubscribe();

}

})

}

}

changeTheme(isDark) {

const x = document.getElementsByTagName("BODY")[0];

*if* (isDark) {

x.classList.remove('light');

x.classList.add('dark');

*return*;

}

x.classList.remove('dark');

x.classList.add('light');

}

reset() {

*try* {

this.$dataflow.unsubscribe();

this.animateNavigation(false);

} *catch* (err) { }

*finally* {

this.$('#address-input1').value = '';

this.$('#address-input2').value = '';

}

}

*// Animating the Notifiction Popup*

animateNavigation(add = true, title = 'Loading', discription = 'Loading', sign = '', minutes = ' Loading') {

*if* (add) {

this.$('#navigate').classList.add('navigation\_animation');

this.$('#progressBar').classList.add('progressBar\_animation');

this.$('#navigate').style.animationDuration = (this.AnimationTime / 1000) + 's';

this.$('#progressBar').style.animationDuration = (this.AnimationTime / 1000) + 's';

this.$('#title').innerHTML = title;

this.$('#discription').innerHTML = discription;

this.$('#sign').innerHTML = sign;

} *else* {

this.$('#navigate').classList.remove('navigation\_animation');

this.$('#progressBar').classList.remove('progressBar\_animation');

}

}

}

initMaps.js

*import* *DOM* *from* './dom.js'

*export* *default* class InitMap {

*// #########################################################*

*// ##################### Gobal Varibles ##########################*

*// #######################################################*

map;

constructor() {

mapboxgl.accessToken = 'pk.eyJ1IjoiZmFraHIiLCJhIjoiY2pseXc0djE0MHBibzN2b2h4MzVoZjk4aSJ9.ImbyLtfsfSsR\_yyBluR8yQ';

this.\_init();

}

*// #########################################################*

*// ##################### Helper Functions ##########################*

*// #######################################################*

*// Initialise the map By setting necessory parameters*

\_init() {

this.map = new mapboxgl.Map({

container: 'map',

style: 'mapbox://styles/mapbox/dark-v10',

center: [-79.3871, 43.6426],

zoom: 17.5,

bearing: 0,

pitch: 30

});

}

*// Delete Current Route*

reset() {

*if* (this.map.getSource('route')) {

this.map.removeLayer('route');

this.map.removeSource('route')

}

}

*// Draw Up the line on map according to the array of cords*

drawRoute(coords) {

*// check if the route is already loaded*

this.reset();

this.map.addLayer({

"id": "route",

"type": "line",

"source": {

"type": "geojson",

"data": {

"type": "Feature",

"properties": {},

"geometry": coords

}

},

"layout": {

"line-join": "round",

"line-cap": "round"

},

"paint": {

"line-color": "#4264FB",

"line-width": 10,

"line-opacity": 1

}

});

}

}

Location.js

*export* *default* class Location {

*// #########################################################*

*// ##################### Global Variables ##########################*

*// #######################################################*

input1;

input2;

constructor() {

this.\_init()

}

*// #########################################################*

*// ##################### Helper Functions ##########################*

*// #######################################################*

*// Initializing the Form*

\_init() {

const apiKey = 'f5ee4a045ea9d2b9d41174f7812f7f51';

const appID = 'plJI4ZSOG6E6';

this.input1 = places({

appId: appID,

apiKey: apiKey,

container: document.querySelector('#address-input1'),

templates: {

value: function (suggestion) {

*return* suggestion.name;

}

}

}).configure({

type: 'address'

});

this.input2 = places({

appId: appID,

apiKey: apiKey,

container: document.querySelector('#address-input2'),

templates: {

value: function (suggestion) {

*return* suggestion.name;

}

}

}).configure({

type: 'address'

});

}

*// Get the currrent location of user but I am not using it anywhere yet (It is giving inacurate results)*

getCurrentLoc() {

*if* (!navigator.geolocation) alert("Geolocation is not supported by this browser."); *return*;

navigator.geolocation.getCurrentPosition((position) => {

*return* [position.coords.longitude, position.coords.latitude]

})

}

OUTPUT







