// ======= ======= ======= init ======= ======= =======

// ======= ======= ======= init ======= ======= =======

// ======= ======= ======= init ======= ======= =======

// Express is both a framework and a command line tool.

var fs = require("fs");

var url = require("url");

var http = require("http");

var https = require('https');

var express = require("express");

var bodyParser = require("body-parser");

var querystring = require("querystring");

var router = express.Router();

var app = express();

app.use(bodyParser.json())

// ======= init =======

var centerLat = 38.8990;

var centerLng = -77.0354;

var port = process.env.VCAP\_APP\_PORT || 4000;

// ======= listen =======

app.listen(port, function(){

console.log("\n-- app.listen:", port);

});

var root = app.use("/", express.static(\_\_dirname + '/public'));

// ======= ======= ======= serve ======= ======= =======

// ======= ======= ======= serve ======= ======= =======

// ======= ======= ======= serve ======= ======= =======

// ======= proxy =======

var proxy = app.post('/proxy', function(req, res) {

var pathname = url.parse(req.url).pathname;

var query = url.parse(req.url).query;

var id = querystring.parse(query)['id'];

var remoteHost = req.body.remoteHost;

var remotePath = req.body.remotePath;

// ======= GET options =======

var options = {

host: remoteHost,

path: remotePath,

method: 'GET'

};

// ======= http.request =======

var proxyRequest = http.request(options, function (proxyResponse) {

if (proxyResponse) {

console.log("\*\*\*\*\*\*\* proxyResponse: OK");

} else {

console.log("\*\*\*\*\*\*\* proxyResponse: NG");

}

proxyResponse.pipe(res, {

end: true

});

});

req.pipe(proxyRequest, {

end: true

});

});

// ======= ======= ======= script.js ======= ======= =======

// ======= ======= ======= script.js ======= ======= =======

// ======= ======= ======= script.js ======= ======= =======

// ======= constants =======

var loopCount = 0;

// ======= display =======

var defaultDisplay = {

loopCount: 0,

allRegions: false,

regionsArray: ["AL", "AR", "DC", "MO", "PG"],

tractStyle: {

"fill": "red",

"stroke": "#1A3742",

"stroke-width": 1 },

bufferStyle: { "fill": "#56B6DB", "stroke": "#1A3742", "stroke-width": 2 },

// ======= toggleClearAll =======

toggleClearAll: function() {

var clearFlag = false;

$.each(app.state.selRegions, function(region, selected) {

if (selected == true) { clearFlag = true; }

});

clearFlag ? $('#clearAll-r > p').text('clear') : $('#clearAll-r > p').text('all');

}

};

// ======= map =======

var defaultMap = {

mapEl: document.getElementById("map"),

mapStyle: "mapbox.light",

centerLat: 38.8990,

centerLng: -77.0354,

zoom: 11,

zoomControl: false

}

// ======= state =======

var defaultState = {

selRegions: { AL: false, AR: false, DC: false, MO: false, PG: false },

AL: {

selLanes: { lanes:true, paths:false, trails:false },

selBuffers: { ft500:false, ft1000:false, ft2500:false, ft5280:false },

laneData:[],

laneLayers:[],

bufferLayers: { ft500:null, ft1000:null, ft2500:null, ft5280:null },

bikeLaneStyle: { color: "#004529", weight: 2 },

regionColor: "#004529"

}, AR: { ... }, DC: { ... }, MO: { ... }, PG: { ... },

}

// ======= regions =======

var regions = {

AL: {

id: "AL\_4",

name: "Alexandria",

box: { NW: [null, null], SE: [null, null] },

laneFiles: {

lanes: "VA\_Alexandria\_Bike.geojson",

paths: null,

trails: null },

bufferFiles: {

ft500:"VA\_Alexandria\_Bike\_Buffer\_500ft.geojson",

ft1000:"VA\_Alexandria\_Bike\_Buffer\_1000ft.geojson",

ft2500:"VA\_Alexandria\_Bike\_Buffer\_2500ft.geojson",

ft5280:"VA\_Alexandria\_Bike\_Buffer\_5280ft.geojson" }

}, AR: { ... }, DC: { ... }, MO: { ... }, PG: { ... },}

// ======= app =======

var app;

app = {

map: defaultMap,

state: defaultState,

display: defaultDisplay,

geocoder: null,

activeMap: null,

// ======= initialize =======

initialize: function() {

app.activeMap = app.initMap();

app.activateMap();

app.activateMenu();

},

// ======= activateMap =======

activateMap: function() {

// ======= mouse location =======

app.activeMap.on("mousemove", function (e) {

document.getElementById("lat").innerHTML = (e.latlng.lat).toFixed(4);

document.getElementById("lng").innerHTML = (e.latlng.lng).toFixed(4);

});

// ======= mouse location =======

app.activeMap.on("click", function (e) {

var latLng = [JSON.stringify(e.latlng.lat), JSON.stringify(e.latlng.lng)];

!$("#startLoc").val() ?

app.addRouteMarker("start", latLng) :

app.addRouteMarker("end", latLng);

== DEV functions ======= ======= =======

app.getCensusData(latLng);

});

},

// ======= getCensusData =======

getCensusData: function(latLng) {

var url = "/proxy";

var lat = latLng[0];

var lng = latLng[1];

// == WORKING

var apiParams = {

remoteHost: "geocoding.geo.census.gov",

remotePath: "/geocoder/geographies/coordinates?x=" + lng +

"&y=" + lat + "&benchmark=4&vintage=4&format=json"

};

app.censusAjaxQueue(url, apiParams);

},

// ======= censusAjaxQueue =======

censusAjaxQueue: function(url, apiParams) {

var censusData = null;

var self = this;

$.ajax({

url: url,

data: JSON.stringify(apiParams),

method: "POST",

dataType: "json",

contentType: "application/json"

}).done(function(jsonData){

self.displayData(jsonData);

self.getBoundaryData(jsonData.result.geographies["Census Tracts"][0].GEOID);

}).fail(function(){

console.log("\*\*\* ajax fail T \*\*\*");

});

},

// ======= getBoundaryData =======

getBoundaryData: function(GEOID) {

var url = "GeojsonFiles/" + "tl\_2016\_11\_tract.geojson";

var nextFeature, nextFeatureId, targetFeature;

$.ajax({

url: url,

method: "GET",

dataType: "json"

}).done(function(jsonData){

$(jsonData.features).each(function(index, feature) {

nextFeatureId = feature.properties.GEOID;

if (nextFeatureId == GEOID) {

targetFeature = feature;

return false;

}

})

var censusTract = L.geoJson(targetFeature, {

style: style

}).addTo(app.activeMap);

function style(feature) {

return {

weight: 2,

opacity: 1,

color: "#004529",

fillOpacity: 0.3,

fillColor: '#ff0000'

};

}

}).fail(function(){

console.log("\*\*\* ajax fail T \*\*\*");

});

},

// ======= displayData =======

displayData: function(jsonData) {

var dataString = jsonData.result.geographies["Census Tracts"][0].NAME +

" geoid " + jsonData.result.geographies["Census Tracts"][0].GEOID;

$("#data").text(dataString);

},

// ======= activateMenu =======

activateMenu: function() {

var menu = document.getElementById("menu");

var dragBar = document.getElementById("dragBar");

var offset = { x: 0, y: 0 };

// == menu filter functions

$('.region, .buffer').on('click', function(e) {

event.stopPropagation();

app.toggleFilterState(e.currentTarget);

app.toggleFilterData(e.currentTarget);

});

$('#clearAll').on('click', function(e) {

event.stopPropagation();

app.clearSelectAll(e.currentTarget);

});

$('.region, .buffer, .start, .end, #clearAll-r, #dragBar').on('mouseenter', function(e) {

event.stopPropagation();

updateHoverText(e.currentTarget, "enter");

});

$('.region, .buffer, .start, .end, #clearAll-r, #dragBar').on('mouseleave', function(e) {

event.stopPropagation();

updateHoverText(e.currentTarget, "leave");

});

// == menu drag functions

$('#dragBar').on('mousedown', function(e) {

initDrag(e);

});

function initDrag(e){

offset.x = e.clientX - menu.offsetLeft;

offset.y = e.clientY - menu.offsetTop;

window.addEventListener('mousemove', menuMove, true);

window.addEventListener('mouseup', mouseUp, true);

}

function menuMove(e){

menu.style.position = 'absolute';

var top = e.clientY - offset.y;

var left = e.clientX - offset.x;

menu.style.top = top + 'px';

menu.style.left = left + 'px';

}

function mouseUp() {

window.removeEventListener('mousemove', menuMove, true);

}

// ======= updateHoverText =======

function updateHoverText(hoverEl, enterLeave) {

var hoverText;

var region = $(hoverEl).parents().eq(1).attr('id');

if (enterLeave == "enter") {

if ($(hoverEl).hasClass('buffer')) {

if (app.state.selRegions[region]) {

$(hoverEl).addClass("entered");

if ($(hoverEl).hasClass('ft500')) {

hoverText = "500 ft buffer";

} else if ($(hoverEl).hasClass('ft1000')) {

hoverText = "1000 ft buffer";

} else if ($(hoverEl).hasClass('ft2500')) {

hoverText = "2500 ft buffer";

} else if ($(hoverEl).hasClass('ft5280')) {

hoverText = "1 mile buffer";

}

}

} else {

if ($(hoverEl).hasClass('region')) {

hoverText = regions[$(hoverEl).closest('tr').attr('id')].name;

} else if ($(hoverEl).hasClass('start')) {

hoverText = "map click start loction";

} else if ($(hoverEl).hasClass('end')) {

hoverText = "map click end loction";

} else if (($(hoverEl).attr('id') == 'clearAll-r')&&($(hoverEl).text() == 'all')) {

hoverText = "select all regions";

} else if (($(hoverEl).attr('id') == 'clearAll-r')&&($(hoverEl).text() == 'clear')) {

hoverText = "clear all regions/buffers";

} else if ($(hoverEl).attr('id') == 'dragBar') {

hoverText = "draggable";

}

}

} else {

if ($(hoverEl).hasClass('buffer')) {

if (app.state.selRegions[region]) {

$(hoverEl).removeClass("entered");

}

}

hoverText = ".";

}

$("#hoverText").text(hoverText);

}

},

// ======= toggleFilterState =======

toggleFilterState: function(toggleEl) {

var region = $(toggleEl).parents().eq(1).attr('id');

var buffer = null;

// == update region filter

if ($(toggleEl).hasClass('region')) {

// == toggle region state

app.state.selRegions[region] ?

app.state.selRegions[region] = false :

app.state.selRegions[region] = true;

// == toggle region element (and buffer elements if deselecting)

app.state.selRegions[region] ?

$(toggleEl).addClass('selected') :

updateBufferState(toggleEl);

// == toggle buffer element visibility if selecting

if (app.state.selRegions[region]) {

var bufferEls = $(toggleEl).parents().eq(1).children();

bufferEls.each(function(index, bufferEl) {

if ($(bufferEl).hasClass('td-b')) {

$(bufferEl).children().css('visibility', 'visible');

}

})

};

};

// == identify selected buffer

if ($(toggleEl).hasClass('ft500')) { buffer = 'ft500' };

if ($(toggleEl).hasClass('ft1000')) { buffer = 'ft1000' };

if ($(toggleEl).hasClass('ft2500')) { buffer = 'ft2500' };

if ($(toggleEl).hasClass('ft5280')) { buffer = 'ft5280' };

// == update buffer filter

if (buffer) {

if (app.state.selRegions[region]) {

// == toggle buffer state

app.state[region].selBuffers[buffer] ?

app.state[region].selBuffers[buffer] = false :

app.state[region].selBuffers[buffer] = true;

// == toggle buffer filter elements

app.state[region].selBuffers[buffer] ?

$(toggleEl).addClass('selected') :

$(toggleEl).removeClass('selected');

}

}

// ======= updateBufferState =======

function updateBufferState(toggleEl) {

// == deselect region filter element

$(toggleEl).removeClass('selected');

// == deselect buffer filter elements for region

var bufferEls = $(toggleEl).parents().eq(1).children();

bufferEls.each(function(index, bufferEl) {

if ($(bufferEl).hasClass('td-b')) {

$(bufferEl).children().removeClass('selected');

$(bufferEl).children().css('visibility', 'hidden');

}

});

app.state[$(toggleEl).parents().eq(1).attr('id')].selBuffers.ft500 = false;

app.state[$(toggleEl).parents().eq(1).attr('id')].selBuffers.ft1000 = false;

app.state[$(toggleEl).parents().eq(1).attr('id')].selBuffers.ft2500 = false;

app.state[$(toggleEl).parents().eq(1).attr('id')].selBuffers.ft5280 = false;

}

// == update clear/all button text

app.display.toggleClearAll();

},

// ======= toggleFilterData =======

toggleFilterData: function(toggleEl) {

var laneFeatures = null;

var bufferFeatures = null;

var region = null;

var buffer = null;

// == region filter selection

if ($(toggleEl).hasClass('region')) {

region = $(toggleEl).parents().eq(1).attr('id');

if (app.state.selRegions[region] && (app.state[region].laneData.length == 0)) {

app.state[region].laneData = app.makeLanesArray(region);

if (app.state[region].laneData) {

var url = "bikelanes/" + app.state[region].laneData[0];

app.laneAjaxQueue(url, region, null);

}

} else {

app.state[region].laneData = [];

removeBufferLayers(region);

}

}

// == identift buffer selection

if ($(toggleEl).hasClass('ft500')) { buffer = 'ft500' };

if ($(toggleEl).hasClass('ft1000')) { buffer = 'ft1000' };

if ($(toggleEl).hasClass('ft2500')) { buffer = 'ft2500' };

if ($(toggleEl).hasClass('ft5280')) { buffer = 'ft5280' };

// == buffer filter selection

if (buffer) {

var selRegion = $(toggleEl).parents().eq(1).attr('id');

app.state[selRegion].selBuffers[buffer] ?

getBufferData(selRegion, buffer) :

clearBufferData(selRegion, buffer);

// ======= getBufferData =======

function getBufferData(region, buffer) {

var url = "buffers/" + regions[region].bufferFiles[buffer];

app.bufferAjaxQueue(url, region, buffer);

}

// ======= clearBufferData =======

function clearBufferData(region, buffer) {

var bufferFeature = app.state[region].bufferLayers[buffer];

if (bufferFeature) {

app.activeMap.removeLayer(bufferFeature);

}

app.state[region].bufferLayers[buffer] = null;

}

}

// ======= removeBufferLayers =======

function removeBufferLayers(region) {

if (!app.state.selRegions[region]) {

app.state[region].laneLayers.forEach(function (laneData) {

app.activeMap.removeLayer(laneData);

})

app.state[region].laneLayers = [];

if (app.state[region].bufferLayers['ft500']) {

app.activeMap.removeLayer(app.state[region].bufferLayers['ft500']);

app.state[region].bufferLayers['ft500'] = null;

}

if (app.state[region].bufferLayers['ft1000']) {

app.activeMap.removeLayer(app.state[region].bufferLayers['ft1000']);

app.state[region].bufferLayers['ft1000'] = null;

}

if (app.state[region].bufferLayers['ft2500']) {

app.activeMap.removeLayer(app.state[region].bufferLayers['ft2500']);

app.state[region].bufferLayers['ft2500'] = null;

}

if (app.state[region].bufferLayers['ft5280']) {

app.activeMap.removeLayer(app.state[region].bufferLayers['ft5280']);

app.state[region].bufferLayers['ft5280'] = null;

}

}

}

},

// ======= makeLanesArray =======

makeLanesArray: function(region) {

var pathDataArray = [];

if (regions[region].laneFiles.lanes) {

var laneFile = regions[region].laneFiles.lanes;

laneFile ? pathDataArray.push(laneFile) : console.log("NO LANES");

}

if (regions[region].laneFiles.paths) {

var pathFile = regions[region].laneFiles.paths;

pathFile ? pathDataArray.push(pathFile) : console.log("NO PATHS");

}

if (regions[region].laneFiles.trails) {

var trailFile = regions[region].laneFiles.trails;

trailFile ? pathDataArray.push(trailFile) : console.log("NO TRAILS");

}

return pathDataArray; // array of lane/path/trail geojson files

},

// ======= clearSelectAll =======

clearSelectAll: function(toggleEl) {

// == select or clear all regions

var clearFlag = false;

$.each(app.state.selRegions, function(region, selected) {

if (selected == true) {

clearFlag = true;

}

});

clearFlag ? clearAll() : selectAll();

// ======= selectAll =======

function selectAll() {

// == show buffer elements

var bufferEls = $('.td-b').children('.buffer');

bufferEls.each(function(index, bufferEl) {

$(bufferEl).css('visibility', 'visible');

});

app.display.allRegions = true;

// == update clear/all button text

$('#clearAll-r > p').text('clear');

// == select all region elements

var regionEls = $('div.region');

$(regionEls).each(function(index, regionEl) {

if (!$(regionEl).hasClass('selected')) { $(regionEl).addClass('selected') };

});

// == set selected state for all regions

$.each(app.state.selRegions, function(region, selected) {

if (!selected) {

app.state.selRegions[region] = true;

}

app.state[region].laneData = app.makeLanesArray(region);

});

var url = "bikelanes/" + regions[app.display.regionsArray[0]].laneFiles.lanes;

app.laneAjaxQueue(url, app.display.regionsArray[0]);

}

// ======= clearAll =======

function clearAll() {

// == update clear/all button text

$('#clearAll-r > p').text('all');

// == hide buffer elements

var bufferEls = $('.td-b').children('.buffer');

bufferEls.each(function(index, bufferEl) {

$(bufferEl).css('visibility', 'hidden');

});

// == update region elements

var regionEls = $('div.region');

$(regionEls).each(function(index, regionEl) {

if ($(regionEl).hasClass('selected')) { $(regionEl).removeClass('selected') };

});

// == update buffer elements

var bufferEls = $('div.buffer');

$(bufferEls).each(function(index, bufferEl) {

if ($(bufferEl).hasClass('selected')) { $(bufferEl).removeClass('selected') };

if ($(bufferEl).hasClass('td-b')) {

$(bufferEl).children().css('visibility', 'hidden');

}

});

// == clear lane and buffer states and data layers

$.each(app.state.selRegions, function(region, selected) {

if (selected) {

app.state.selRegions[region] = false;

app.state[region].laneLayers.forEach(function (laneData) {

app.activeMap.removeLayer(laneData);

})

app.state[region].laneData = [];

app.state[region].laneLayers = [];

$.each(app.state[region].selBuffers, function(buffer, state) {

buffer = false;

});

$.each(app.state[region].bufferLayers, function(buffer, bufferData) {

if (bufferData) {

app.activeMap.removeLayer(bufferData);

bufferData = null;

}

});

}

});

// == restore regions array

app.display.regionsArray = ["AL", "AR", "DC", "MO", "PG"];

}

},

// ======= laneAjaxQueue =======

laneAjaxQueue: function(url, region) {

var laneData = null;

$.ajax({

url: url,

method: "GET",

dataType: "text"

}).done(function(jsonData){

var parsedJson = $.parseJSON(jsonData);

var laneFeatures = L.mapbox.featureLayer(parsedJson).addTo(app.activeMap);

app.state[region].laneLayers.push(laneFeatures);

laneFeatures.setStyle(app.state[region].bikeLaneStyle);

if ((app.display.loopCount < app.state[region].laneData.length - 1) &&

(app.state[region].laneData.length != 1)) {

app.display.loopCount++;

laneData = app.state[region].laneData[app.display.loopCount];

url = "bikelanes/" + laneData;

app.laneAjaxQueue(url, region);

} else {

app.display.loopCount = 0;

if (app.display.allRegions) {

if (app.display.regionsArray.length > 0) {

region = app.display.regionsArray[0];

laneData = app.state[region].laneData[app.display.loopCount];

app.display.regionsArray.shift();

if (laneData) {

url = "bikelanes/" + laneData;

app.laneAjaxQueue(url, region);

} else {

console.log("=== ALERT: Missing data for", region);

}

} else {

app.display.regionsArray = ["AL", "AR", "DC", "MO", "PG"];

}

}

}

}).fail(function(){

console.log("\*\*\* ajax fail T \*\*\*");

});

},

// ======= bufferAjaxQueue =======

bufferAjaxQueue: function(url, region, buffer) {

$.ajax({

url: url,

method: "GET",

dataType: "text"

}).done(function(jsonData){

var parsedJson = $.parseJSON(jsonData);

var bufferFeatures = L.mapbox.featureLayer(parsedJson).addTo(app.activeMap);

app.state[region].bufferLayers[buffer] = bufferFeatures;

bufferFeatures.setStyle(app.display.bufferStyle);

}).fail(function(){

console.log("\*\*\* ajax fail T \*\*\*");

});

},

// ======= initMap =======

initMap: function () {

L.mapbox.accessToken = "pk.eyJ1IjoiYWx1bHNoIiwiYSI6ImY0NDBj...KEPsfuC4j54XDT3VA";

var map = L.mapbox.map(

app.map.mapEl,

app.map.mapStyle,

{ zoomControl: false })

.setView([app.map.centerLat, app.map.centerLng], app.map.zoom);

new L.Control.Zoom({ position: "bottomright" }).addTo(map);

return map;

},

// ======= addRouteMarker =======

addRouteMarker: function(startEnd, latLng, data) {

// == geocoding request format: /geocoding/v5/{mode}/{query}.json

var baseUrl = "https://api.mapbox.com/geocoding/v5/mapbox.places/" + latLng[1] + "," + latLng[0]

var token = "pk.eyJ1IjoiYWx1bHNoIiw...uC4j54XDT3VA";

var url = baseUrl + ".json?access\_token=" + token;

// == get directions for selected places

$.ajax({

url: url,

method: "GET",

dataType: "text"

}).done(function(jsonData){

var parsedJson = $.parseJSON(jsonData);

updateRouteData(parsedJson);

makeRouteMarker(parsedJson);

}).fail(function(){

console.log("\*\*\* ajax fail \*\*\*");

});

// ======= updateRouteData =======

function updateRouteData(jsonData) {

var startLoc = $('.start').text();

var endLoc = $('.end').text();

if (startLoc == "\xa0") {

$('.start').text(jsonData.features[0].text);

} else {

if (endLoc == "\xa0") {

$('.end').text(jsonData.features[0].text);

} else {

$('.start').text(jsonData.features[0].text);

$('.end').text('\xa0');

clearRouteMarkers();

}

}

}

// ======= clearRouteMarkers =======

function clearRouteMarkers(jsonData) {

$.each(app.map.markersGroup, function(index, marker) {

app.activeMap.removeLayer(marker);

});

app.map.markersGroup = [];

}

// ======= makeRouteMarker =======

function makeRouteMarker(jsonData) {

var latLng = jsonData.features[0].center;

var lat = latLng[0];

var lng = latLng[1];

var placeMarker = L.circle([latLng[1], latLng[0]], 200).addTo(app.activeMap);

app.map.markersGroup.push(placeMarker);

}

}

}

app.initialize();

// ======= ======= ======= hackForGood ======= ======= =======

// ======= ======= ======= hackForGood ======= ======= =======

// ======= ======= ======= hackForGood ======= ======= =======

app.initialize();

mapboxgl.accessToken = 'pk.eyJ1IjoiYWx1bHNoIiwiYSI6Im...n0.pngboKEPsfuC4j54XDT3VA';

var lanesUrl = 'https://raw.githubusercontent.com/dcfemtech/hackforgood-waba-map/master/bikelanes/';

var buffersUrl = 'https://raw.githubusercontent.com/dcfemtech/hackforgood-waba-map/master/buffers/';

// ======= regions =======

var regions = {

AL: {

id: 'AL\_4',

name: 'Alexandria',

box: {

NW: [null, null],

SE: [null, null]

},

laneFiles: {

lanes: lanesUrl + 'VA\_Alexandria.geojson',

paths: null,

trails: null

},

bufferFiles: {

ft500: buffersUrl + 'VA\_Alexandria\_Bike\_Buffer\_500ft.geojson',

ft1000: buffersUrl + 'VA\_Alexandria\_Bike\_Buffer\_1000ft.geojson',

ft2500: buffersUrl + 'VA\_Alexandria\_Bike\_Buffer\_2500ft.geojson',

ft5280: buffersUrl + 'VA\_Alexandria\_Bike\_Buffer\_5280ft.geojson'

}

},

AR: {

id: 'AR\_3',

name: 'Arlington',

box: {

NW: [null, null],

SE: [null, null]

},

laneFiles: {

lanes: lanesUrl + 'VA\_Arlington.geojson',

paths: null,

trails: null

},

bufferFiles: {

ft500: buffersUrl + 'VA\_Arlington\_Bike\_Buffer\_500ft.geojson',

ft1000: buffersUrl + 'VA\_Arlington\_Bike\_Buffer\_1000ft.geojson',

ft2500: buffersUrl + 'VA\_Arlington\_Bike\_Buffer\_2500ft.geojson',

ft5280: buffersUrl + 'VA\_Arlington\_Bike\_Buffer\_5280ft.geojson'

}

},

DC: {

id: 'DC\_0',

name: 'District of Columbia',

box: {

NW: [null, null],

SE: [null, null]

},

laneFiles: {

lanes: lanesUrl + 'DC\_Washington.geojson',

paths: null,

trails: null

},

bufferFiles: {

ft500: buffersUrl + 'DC\_Bike\_Buffer\_500ft.geojson',

ft1000: buffersUrl + 'DC\_Bike\_Buffer\_1000ft.geojson',

ft2500: buffersUrl + 'DC\_Bike\_Buffer\_2500ft.geojson',

ft5280: buffersUrl + 'DC\_Bike\_Buffer\_5280ft.geojson'

}

},

MO: {

id: 'MO\_1',

name: 'Montgomery County',

box: {

NW: [null, null],

SE: [null, null]

},

laneFiles: {

lanes: lanesUrl + 'MD\_MontgomeryCounty.geojson',

paths: null,

trails: null

},

bufferFiles: {

ft500: buffersUrl + 'MD\_MontgomeryCounty\_Bikeways\_Buffer\_500ft.geojson',

ft1000: buffersUrl + 'MD\_MontgomeryCounty\_Bikeways\_Buffer\_1000ft.geojson',

ft2500: buffersUrl + 'MD\_MontgomeryCounty\_Bikeways\_Buffer\_2500ft.geojson',

ft5280: buffersUrl + 'MD\_MontgomeryCounty\_Bikeways\_Buffer\_5280ft.geojson'

}

},

PG: {

id: 'PG\_2',

name: 'Prince George\'s County',

box: {

NW: [null, null],

SE: [null, null]

},

laneFiles: {

lanes: lanesUrl + 'MD\_PrinceGeorgesCounty.geojson',

paths: null,

trails: null

},

bufferFiles: {

ft500: null,

ft1000: null,

ft2500: null,

ft5280: null

}

},

}

// ======= initialize map =======

var map = new mapboxgl.Map({

container: 'map',

style: 'mapbox://styles/mapbox/light-v9',

center: [-77.0354, 38.8990],

zoom: 11

});

map.addControl(new mapboxgl.Navigation({

'position': 'bottom-right'

}));

// ======= directions =======

var directions = new mapboxgl.Directions({

position: 'top-left',

profile: 'cycling'

});

map.addControl(directions);

// ======= add bike lanes =======

function addLanes(REGION) {

map.addSource(REGION + 'lanes-src', {

type: 'geojson',

data: regions[REGION].laneFiles.lanes

});

map.addLayer({

id: REGION + 'lanes-layer',

type: 'line',

source: REGION + 'lanes-src',

layout: {

'line-join': 'round',

'line-cap': 'round',

'visibility': 'visible'

},

paint: {

'line-color': 'green',

'line-width': 2

}

});

}

// ======= add buffer =======

function addBuffer(REGION, FEET) {

map.addSource(REGION + FEET + 'buffers-src', {

type: 'geojson',

data: regions[REGION].bufferFiles[FEET]

});

map.addLayer({

id: REGION + FEET + 'buffers-layer',

type: 'fill',

source: REGION + FEET + 'buffers-src',

layout: {

visibility: 'none',

},

paint: {

'fill-outline-color': '#1A3742',

'fill-color': '#56B6DB',

'fill-opacity': 0.5

}

});

}

// ======= make layer visibile or invisible =======

function toggleLayerVisibility(LAYER) {

var vis = map.getLayoutProperty(LAYER, 'visibility');

if (vis == 'none') {

map.setLayoutProperty(LAYER, 'visibility', 'visible');

} else {

map.setLayoutProperty(LAYER, 'visibility', 'none');

}

}

// ======= initialize map layers =======

map.on('load', function () {

for (r in regions) {

addLanes(r);

if (regions[r].bufferFiles.ft500) {

addBuffer(r, 'ft500');

}

//console.log(regions[r].id + 'ft500' + 'buffers-layer')

if (regions[r].bufferFiles.ft1000) {

addBuffer(r, 'ft1000');

}

if (regions[r].bufferFiles.ft2500) {

addBuffer(r, 'ft2500');

}

if (regions[r].bufferFiles.ft5280) {

addBuffer(r, 'ft5280');

}

}

});

// ======= get coordinates under the mouse =======

map.on('mousemove', function (e) {

document.getElementById('lat').innerHTML = JSON.stringify(e.lngLat['lat']);

document.getElementById('lng').innerHTML = JSON.stringify(e.lngLat['lng']);

});

// ======= toggle buffers visibile or invisible =======

$('.buffer').on('click', function () {

toggleLayerVisibility($(this).parent().parent().attr("id") + $(this).attr("class").split(' ')[1] + 'buffers-layer');

$(this).toggleClass('selected');

});

// ======= toggle lanes visibile or invisible =======

$('.label-r').on('click', function () {

if ($(this).text() == "all") {

for (r in regions) {

map.setLayoutProperty(r + 'lanes-layer', 'visibility', 'none');

}

$('.region').removeClass('selected');

} else {

toggleLayerVisibility($(this).text() + 'lanes-layer');

}

$(this).parent().toggleClass('selected');

});