Universidad Politécnica de Chiapas

DSI - PI

José Gerardo Rodríguez Velasco 16322

Reporte de implementación API Map Matching de MapBox

18 de noviembre de 2020, Tuxtla Gutiérrez Chiapas.

Configuración

- HTML 5
- CSS 3
- JS ECMAScript
- MapBox Access Token
- Mapbox GL JS
- Mapbox GL Draw Plugin
- Mapbox Map Matching API
- jQuery
- Sublime Text

Introducción

La API Mapbox Map Matching es posible de trazar rutas personalizadas permitiendo diferentes posibilidades de enrutamiento. Ajusta las coordenadas a la red de carreteras de OpenStreetMap para devolver indicaciones paso a paso para las rutas que produce.

Objetivos

- ### Objetivo principal

  Aplicación web que permitirá a sus usuarios crear rutas de entrega.

- ### Objetivos específicos

- El usuario marcará la ruta en el mapa utilizando la herramienta de Mapbox GL Draw plugin.
- La aplicación enviará coordenadas a la API Map Matching para generar indicaciones paso a paso para la nueva ruta de entrega.
- La aplicación trazará la ruta con indicaciones para la nueva ruta de entrega.

Implementación

1. Crear mapa y barra de indicaciones
2. Agregar herramientas de dibujo.
3. Agregar la API de Map Matching.
4. Dibujar la ruta de correspondencia de mapas.
5. Mostrar las direcciones paso a paso.
6. Agregar método eliminar ruta.

1. Implementar scripts en <head>, funcionalidad y estilos de Mapbox, jQuery que permite usar Ajax para analizar peticiones a la API Map Matching.

```
<head>
    <meta charset="utf-8">
    <title>Rutas de entrega </title>
    <meta name='viewport' content='initial-scale=1,maximum-scale=1,user-scalable=no' />

    <!-- Importar Mapbox GL JS  -->
    <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v1.12.0/mapbox-gl.js'></script>
    <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v1.12.0/mapbox-gl.css' rel='stylesheet' />
    <!-- Importar jQuery -->
    <script src='https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js'></script>
    <!-- Impotar Mapbox GL Draw plugin-->
    <script src='https://api.mapbox.com/mapbox-gl-js/plugins/mapbox-gl-draw/v1.0.9/mapbox-gl-draw.js'></script>
    <link rel='stylesheet' href='https://api.mapbox.com/mapbox-gl-js/plugins/mapbox-gl-draw/v1.0.9/mapbox-gl-draw.css' type='text/css' />
    <!-- Impotar Hoja de estilos-->
    <link rel="stylesheet" href="style.css">
</head>
```

1. Crear contenedor del mapa y barra de indicaciones

```
<body>

    <div id="map"></div>
    <div class="info-box">
          <div id="info">
            <p>Dibuja tu ruta usando las herramientas de nodos, ubicados en la parte derecha, toca de nuevo el punto final para mostrar
            ruta.</p>
          </div>
          <div id="directions"></div>
        </div>
```

1. Crear estilos para los contendores.

```css
body {
  margin: 0;
  padding: 0;
}

#map {
  position: absolute;
  top: 0;
  bottom: 0;
  width: 100%;
}

.info-box {
  position: absolute;
  margin: 20px;
  width: 25%;
  top: 0;
  bottom: 40%;
  padding: 20px;
  background-color: rgba(255, 255, 255, 0.9);
  overflow-y: scroll;
  font-family: sans-serif;
  font-size: 0.8em;
  line-height: 2em;
}

#info {
  font-size: 16px;
  font-weight: bold;
}
```

1. Crear script para el Access token, y configuración del mapa.

```
<script>

//Mapbox access token
mapboxgl.accessToken ='pk.eyJ1Ijoicm9kdmVsMTIzIiwiYSI6ImNqdDdqNGh6dDBzajQ0OXIxNHl5aDR1ODcifQ.8yJaK5Y0Au2YyKxnUvMHNA';
var map = new mapboxgl.Map({
  container: 'map',
  style: 'mapbox://styles/mapbox/streets-v11',  //especifica la version del estilo del mapa
  center: [-93.1075127,16.7534462],     //especificar posicion coordenadas
  zoom: 14,     //zoom inicial del mapa
});
```

2. Implementar la herramienta para dibujar líneas y eliminarlas.

```
var draw = new MapboxDraw({
  //  Muestra solo las herramientas de dibujar y eliminar
  displayControlsDefault: false,
  controls: {
    line_string: true,
    trash: true
},
styles: [
// establece el estilo de linea para las coordenas introducidas por el usuario
        {
          "id": "gl-draw-line",
          "type": "line",
          "filter": ["all", ["==", "$type", "LineString"],
            ["!=", "mode", "static"]
          ],
          "layout": {
            "line-cap": "round",
            "line-join": "round"
          },
          "paint": {
            "line-color": "#438EE4",
            "line-dasharray": [0.2, 2],
            "line-width": 4,
            "line-opacity": 0.7
          }
        },
        // Estilo del contorno de los puntos
        {
          "id": "gl-draw-polygon-and-line-vertex-halo-active",
          "type": "circle",
          "filter": ["all", ["==", "meta", "vertex"],
            ["==", "$type", "Point"],
            ["!=", "mode", "static"]
          ],
          "paint": {
            "circle-radius": 12,
            "circle-color": "#FFF"
          }
        },
        // Estilo de los puntos
        {
          "id": "gl-draw-polygon-and-line-vertex-active",
          "type": "circle",
          "filter": ["all", ["==", "meta", "vertex"],
            ["==", "$type", "Point"],
            ["!=", "mode", "static"]
          ],
          "paint": {
            "circle-radius":8,
            "circle-color": "#438EE4",
          }
        },
    ]
    });

    // Agrega las herramientas al mapa
    map.addControl(draw);
```

3. La API requiere dos parámetros: **Profile** debe usar la consulta y **coordinates**, debe coincidir con la red de caminos y rutas de OpenStreetMap. Profile puede ser walking, cycling, driving o driving-traffic, Coordinate son pares de coordenadas.

```
https://api.mapbox.com/matching/v5/mapbox/{profile}/{coordinates}.json?access_token=YOUR_MAPBOX_ACCESS_TOKEN
```

18 de noviembre de 2020, Tuxtla Gutiérrez Chiapas.

La API Map Matching también acepta otros parámetros opcionales que se utilizan para personalizar la consulta. Para esta aplicación, utilizará dos parámetros opcionales:

- Radiuses: agregar un radio, para definir la precisión en que debe coincidir con las coordenadas de entrada, no siempre representan una ubicación en la red de carreteras de OpenStreetMap.
- Steps: Opción predeterminada (True) para devolver las indicaciones paso a paso.

3. Agregar las funciones updateRoute y getMatch.

   - updateRoute: tomará las coordenadas de la línea generada por el usuario y las formateará para que se puedan utilizar en una consulta de correspondencia de mapas.
   - getMatch: construirá la cadena de consulta y utilizará Ajax para realizar la llamada a la API Map Matching.

```
//Funcion para tomar coordenadas de la linea generada por el usuario
function updateRoute() {
      var profile = "cycling";
      // Obtener coordenadas que se dibujaron en el mapa
      var data = draw.getAll();
      var lastFeature = data.features.length - 1;
      var coords = data.features[lastFeature].geometry.coordinates;
      // Formato de coordenadas
      var newCoords = coords.join(';')
      var radius = [];      //radio para cada par de coordenads
      coords.forEach(element => {
        radius.push(10);
      });
      getMatch(newCoords, radius, profile);
}

//Construye una cadena de consulta mediante Ajax para realizar una consulta al API Map Matching
//Funcion para realizar una consulta al API Map Matching
function getMatch(coordinates, radius, profile) {
      var radiuses = radius.join(';')
      // Crea el query
      var query = 'https://api.mapbox.com/matching/v5/mapbox/' + profile + '/' + coordinates + '?geometries=geojson&radiuses=' + radiuses +
          '&steps=true&access_token=' + mapboxgl.accessToken;

      $.ajax({
        method: 'GET',
        url: query
      }).done(function(data) {
        // Obtiene las coordenadas de la peticion
        var coords = data.matchings[0].geometry;
        console.log(coords);
         addRoute(coords);
         getInstructions(data.matchings[0]);
      });
}
```

Para llamar las funciones cuando el usuario dibuja o actualiza una línea en el mapa agregar

```
map.on('draw.create', updateRoute);
map.on('draw.update', updateRoute);
map.on('draw.delete', removeRoute);
```

4. Crear función addRoute que toma las coordenadas devueltas por la API para agregarlas al mapa como una nueva capa.
   Usar la función addRoute dentro de la función getMatch.

```
//Funcion para dibujar la ruta con las cooredendas devueltas por el API, agregandolas como una nueva capa en el mapa
function addRoute(coords) {
      // si hay una ruta cargada, eliminar
      if (map.getSource('route')) {
        map.removeLayer('route')
        map.removeSource('route')
      } else {  //agregar nueva capa al mapa
        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": "#03AA46",
            "line-width": 8,
            "line-opacity": 0.8
          }
        });
      };
}
```

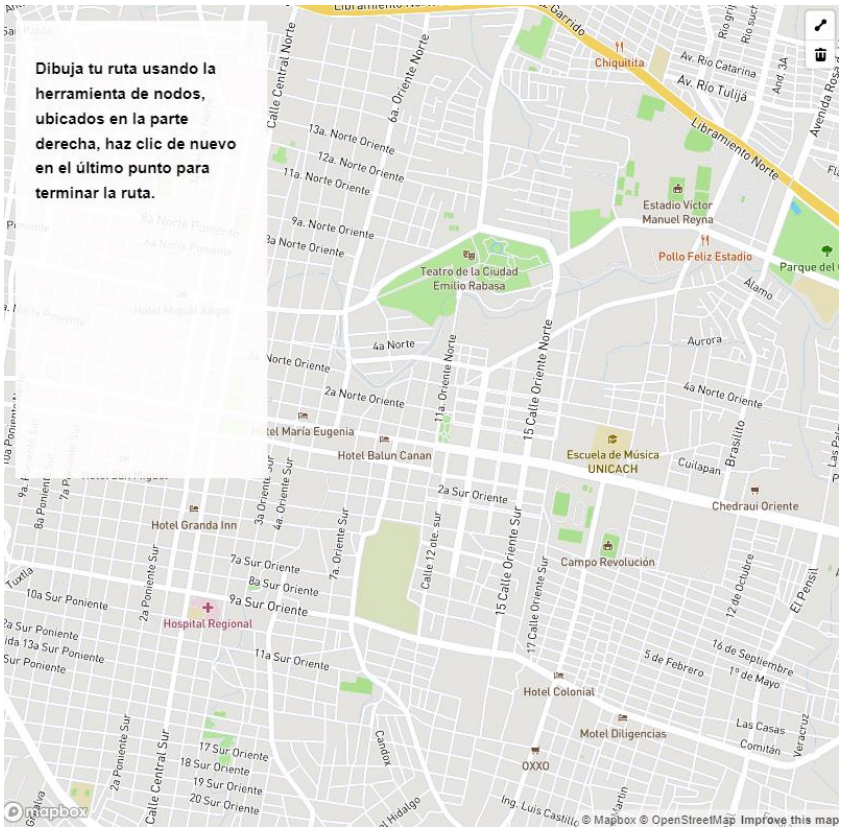
5. Crear función getInstructions para mostrar las indicaciones paso a paso en la barra lateral.
   llamar getInstructions dentro de la función getMach para que pueda acceder a los datos devueltos por la API Map Matching.

```
//funcion para agregar las indicaciones paso a paso
function getInstructions(data) {
      // id del contenedor de direcciones
      var directions = document.getElementById('directions');
      var legs = data.legs;
      var tripDirections = [];
      // salida de las instrucciones para cada paso de cada tramo en el la ruta
      for (var i = 0; i < legs.length; i++) {
        var steps = legs[i].steps;
        for (var j = 0; j < steps.length; j++) {
          tripDirections.push('<br><li>' + steps[j].maneuver.instruction) + '</li>';
        }
      }
      directions.innerHTML = '<br><h2>Trip duration: ' + Math.floor(data.duration / 60) + ' min.</h2>' + tripDirections;
}
```

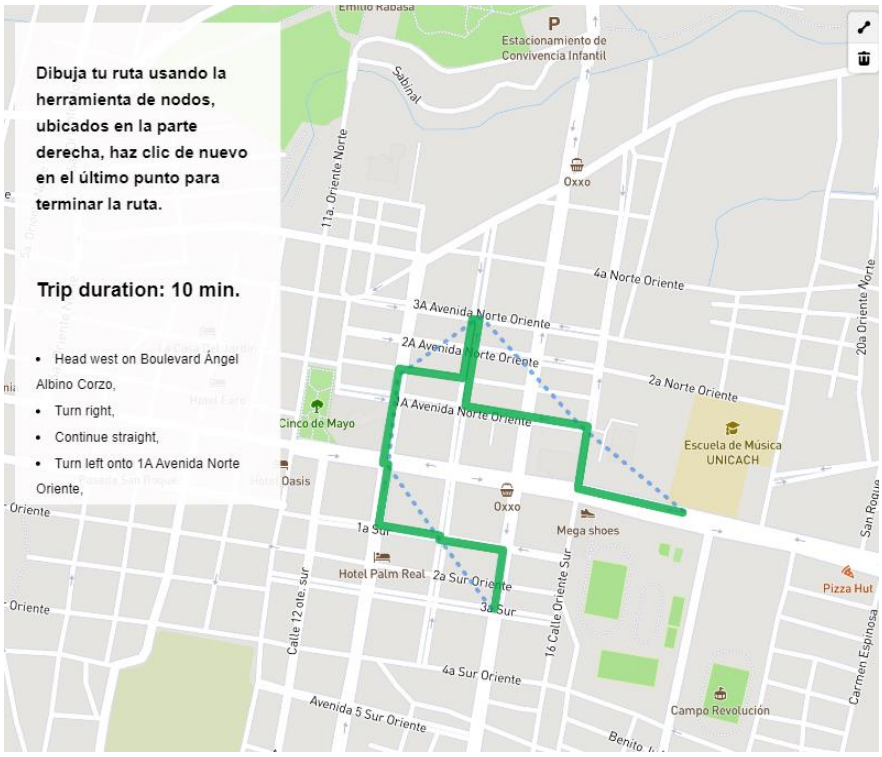
6. Agregar función que permita eliminar una ruta.

```
//Funcion para eliminar ruta
function removeRoute() {
    //si el usuario hace clic en eliminar, elimina la capa si existe
    if (map.getSource('route')) {
        map.removeLayer('route');
        map.removeSource('route');
    } else {
        return;
    }
}
```

Resultados



Mapa Ejemplo 1



Mapa Ejemplo 2

Referencias

jQuery. (2020). Obtenido de https://jquery.com/

Mapbox. (02 de Septiembre de 2020). Obtenido de Mabbox GL Draw Plugin: https://github.com/mapbox/mapbox-gl-draw

MapBox Access Token. (2020). Obtenido de https://account.mapbox.com/

Mapbox.docs. (2020). Obtenido de https://docs.mapbox.com/api/navigation/#map-matching

MaxBox GL JS. (13 de Agoato de 2020). Obtenido de https://docs.mapbox.com/mapbox-gl-js/api/

Sublime Text. (2020). Obtenido de https://www.sublimetext.com/

18 de noviembre de 2020, Tuxtla Gutiérrez Chiapas.