**Lab - Google Maps en Angular (II)**

**Uso de MapMarker**

Con el mapa en su sitio, podemos empezar a añadir marcadores. Esto se hace utilizando el componente MapMarker. Para añadir un marcador, asegúrese de que el marcador se añade dentro de la etiqueta google-map, de lo contrario no se mostrará.

Al igual que el MapControl, las opciones más utilizadas se pueden establecer directamente con las propiedades @Input(), pero también es posible hacer uso del conjunto completo de opciones del MapMarker.

Se modificará el archivo **app.component.html** para agregar la siguiente etiqueta:

<map-marker

\*ngFor="let marker of markers"

[position]="marker.position"

[label]="marker.label"

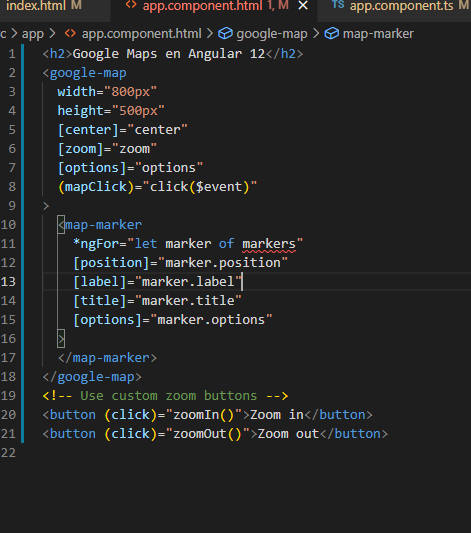
[title]="marker.title"

[options]="marker.options"

>

</map-marker>

Esta etiqueta se muestra dentro de la etiqueta google-maps:



A continuación, agregaremos este método, dentro de *app.component.ts*:

Declarar la variable:

markers = [] as any;

y luego se declara el siguiente método:

addMarker() {

this.markers.push({

position: {

lat: this.center.lat + ((Math.random() - 0.5) \* 2) / 10,

lng: this.center.lng + ((Math.random() - 0.5) \* 2) / 10,

},

label: {

color: 'red',

text: 'Marker label ' + (this.markers.length + 1),

},

title: 'Marker title ' + (this.markers.length + 1),

options: { animation: google.maps.Animation.BOUNCE },

})

}

Para que este método funcione, se agregara un botón que activara esta operación:

<button (click)="addMarker()">Add Marker</button>

En el laboratorio anterior, se definió el método *logCenter*(), el cual permitirá mostrar la posición del centro del mapa.

Con estos datos, y colocando un \*ngFor para manejar el despliegue de los datos de los marcadores, el archivo **app.component.html** quedara asi:

<h2>Google Maps en Angular 12</h2>

<google-map

  width="800px"

  height="500px"

  [center]="center"

  [zoom]="zoom"

  [options]="options"

  (mapClick)="click($event)"

>

  <map-marker

    \*ngFor="let marker of markers"

    [position]="marker.position"

    [label]="marker.label"

    [title]="marker.title"

    [options]="marker.options"

  >

  </map-marker>

</google-map>

<div  \*ngFor="let marker of markers">

  <li>{{marker | json}}</li>

</div>

<!-- Use custom zoom buttons -->

<button (click)="zoomIn()">Zoom in</button>

<button (click)="zoomOut()">Zoom out</button>

<button (click)="addMarker()">Add Marker</button>

En el caso de **app.component.ts**, con las modificaciones realizadas, el código quedara asi:

import { Component, OnInit, ViewChild } from '@angular/core';

import {GoogleMap, MapInfoWindow, MapMarker} from '@angular/google-maps';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.css']

})

export class AppComponent implements OnInit{

  @ViewChild(GoogleMap, { static: false }) map: GoogleMap

  markers = [] as any;

  title = 'Como usar el Componente Google Maps de Angular';

  ngOnInit(): void {

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

      this.center = {

        lat: position.coords.latitude,

        lng: position.coords.longitude,

      }

      console.log("position: "+position.coords.latitude+" "+position.coords.longitude)

      setTimeout(() => {

        this.addMarker();

      }, 1000)

    })

  }

  // Configuración de Google Maps

  center: google.maps.LatLngLiteral;

  zoom = 15;

  options: google.maps.MapOptions = {

    mapTypeId: 'hybrid',

    zoomControl: false,

    scrollwheel: false,

    disableDoubleClickZoom: true,

    maxZoom: 15,

    minZoom: 8,

  }

  zoomIn() {

    if (this.zoom < this.options.maxZoom)

         this.zoom++

  }

  zoomOut() {

    if (this.zoom > this.options.minZoom)

         this.zoom--

  }

  click(event: google.maps.MapMouseEvent) {

    console.log(event)

    this.logCenter();

  }

  addMarker() {

    this.markers.push({

      position: {

        lat: this.center.lat + ((Math.random() - 0.5) \* 2) / 10,

        lng: this.center.lng + ((Math.random() - 0.5) \* 2) / 10,

      },

      label: {

        color: 'red',

        text: 'Etiqueta ' + (this.markers.length + 1),

      },

      title: 'Titulo ' + (this.markers.length + 1),

      options: { animation: google.maps.Animation.BOUNCE },

    })

    console.log("marcador colocado "+this.markers)

  }

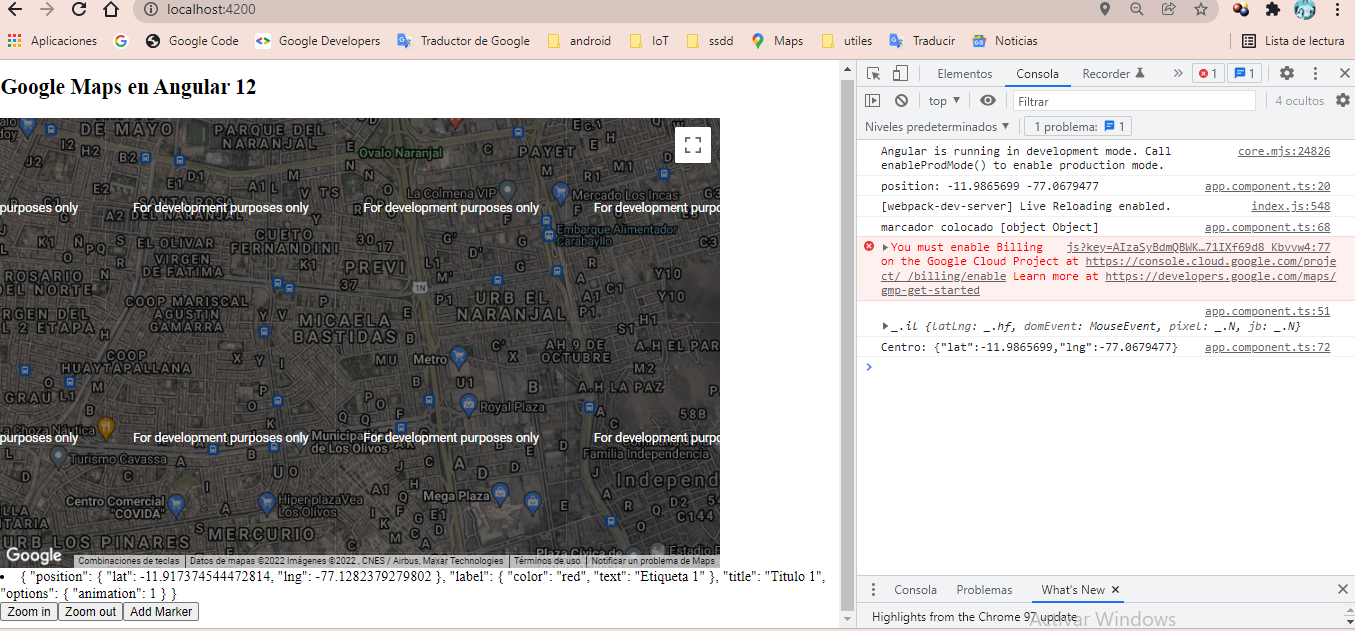
  logCenter() {

    console.log("Centro: "+JSON.stringify(this.map.getCenter()))

  }

}

Al ejecutarse se observará el resultado siguiente:



**MAPINFOWINDOW**

El último componente es MapInfoWindow, se puede utilizar para abrir una ventana emergente de un marcador. Para mostrar la ventana emergente tenemos que añadir el componente dentro de la plantilla de google-map.

<map-info-window>Hello Google Maps</map-info-window>

Lo anterior no hace nada, para que aparezca tenemos que abrir la ventana de información cuando hagamos clic en un marcador. Vinculamos el método mapClick() al marcador y pasamos la referencia del marcador al método openInfo para abrir la ventana de información.

<map-marker

#markerElem

\*ngFor="let marker of markers"

[position]="marker.position"

[label]="marker.label"

[title]="marker.title"

[options]="marker.options"

(mapClick)="openInfo(markerElem)"

>

</map-marker>

Por último, también tenemos que añadir una referencia al componente MapInfoWindow dentro de nuestro componente, podemos hacerlo utilizando el decorador @ViewChild. Al tener la referencia a la ventana de información y al marcador, podemos abrir la ventana de información utilizando el método infoWindow.open().

export class AppComponent implements OnInit {

@ViewChild(MapInfoWindow, { static: false }) infoWindow: MapInfoWindow

----

infoContent = ''

---

openInfo(marker: MapMarker, content:string) {

this.infoContent = content

this.infoWindow.open(marker)

}

}

**Contenido dinámico**

Tener contenido estático dentro de la ventana de información es un poco aburrido. Para proporcionar contenido dinámico dentro de la ventana de información, podemos crear una propiedad de cadena dentro del componente, esto se ve de la siguiente manera.

Usar ViewChildren, usar el índice en el bucle for de la plantilla, pasar el índice cuando se haga clic en el marcador, hacer un bucle a través de todos los ViewChildren usando un índice dentro de un forEach (el bucle for con el índice no abrirá la ventana por alguna razón), entonces si el índice del bucle coincide con el índice pasado, abrir esa ventana.

<map-info-window>{{ infoContent }}</map-info-window>

Y se coloca en el archivo ts seria:

export class AppComponent implements OnInit {

@ViewChildren(MapInfoWindow) infoWindowsView: QueryList<MapInfoWindow>;

infoContent = ''

---

openInfo(marker: MapMarker, windowIndex: number) {

/// stores the current index in forEach

let curIdx = 0;

this.infoWindowsView.forEach((window: MapInfoWindow) => {

if (windowIndex === curIdx) {

window.open(marker);

curIdx++;

} else {

curIdx++;

}

});

}

}

Al final de este proceso los 2 archivos quedan de la siguiente manera:

**app.component.html**

<h2>Google Maps en Angular 12</h2>

<google-map

  width="800px"

  height="500px"

  [center]="center"

  [zoom]="zoom"

  [options]="options"

  (mapClick)="click($event)"

>

  <map-marker

    \*ngFor="let marker of markers;let i = index"

    [position]="marker.position"

    [label]="marker.label"

    [title]="marker.title"

    [options]="marker.options"

    (mapClick)="openInfo(marker, i)"

  >

  <map-info-window>{{ infoContent }}</map-info-window>

  </map-marker>

</google-map>

<div  \*ngFor="let marker of markers">

  <li>{{marker | json}}</li>

</div>

<!-- Use custom zoom buttons -->

<button (click)="zoomIn()">Zoom in</button>

<button (click)="zoomOut()">Zoom out</button>

<button (click)="addMarker()">Add Marker</button>

**app.component.ts**

import { Component, OnInit, QueryList, ViewChild, ViewChildren } from '@angular/core';

import {GoogleMap, MapInfoWindow, MapMarker} from '@angular/google-maps';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.css']

})

export class AppComponent implements OnInit{

  @ViewChild(GoogleMap, { static: false }) map: GoogleMap

  //@ViewChild(MapInfoWindow, { static: false }) infoWindow: MapInfoWindow

  @ViewChildren(MapInfoWindow) infoWindowsView: QueryList<MapInfoWindow>;

  markers = [] as any;

  infoContent = ''

  title = 'Como usar el Componente Google Maps de Angular';

  ngOnInit(): void {

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

      this.center = {

        lat: position.coords.latitude,

        lng: position.coords.longitude,

      }

      console.log("position: "+position.coords.latitude+" "+position.coords.longitude)

      setTimeout(() => {

        this.addMarker();

      }, 1000)

    })

  }

  // Configuración de Google Maps

  center: google.maps.LatLngLiteral;

  zoom = 15;

  options: google.maps.MapOptions = {

    mapTypeId: 'hybrid',

    zoomControl: false,

    scrollwheel: false,

    disableDoubleClickZoom: true,

    maxZoom: 15,

    minZoom: 8,

  }

  zoomIn() {

    if (this.zoom < this.options.maxZoom)

         this.zoom++

  }

  zoomOut() {

    if (this.zoom > this.options.minZoom)

         this.zoom--

  }

  click(event: google.maps.MapMouseEvent) {

    console.log(event)

    this.logCenter();

  }

  addMarker() {

    this.markers.push({

      position: {

        lat: this.center.lat + ((Math.random() - 0.5) \* 2) / 10,

        lng: this.center.lng + ((Math.random() - 0.5) \* 2) / 10,

      },

      label: {

        color: 'red',

        text: 'Etiqueta ' + (this.markers.length + 1),

      },

      title: 'Titulo ' + (this.markers.length + 1),

      options: { animation: google.maps.Animation.BOUNCE },

    })

    console.log("marcador colocado "+this.markers)

  }

  logCenter() {

    console.log("Centro: "+JSON.stringify(this.map.getCenter()))

  }

  openInfo(marker: MapMarker, windowIndex: number) {

    /// stores the current index in forEach

    let curIdx = 0;

    this.infoWindowsView.forEach((window: MapInfoWindow) => {

      if (windowIndex === curIdx) {

        window.open(marker);

        curIdx++;

      } else {

        curIdx++;

      }

    });

  }

}