Angular 2 y Google Maps

Instalación

Desde NPM, instalar en nuestro proyecto el modulo para que se pueda importar en el app.module.ts

```
npm install angular2-google-maps --save
```

En el modulo

```
import { AgmCoreModule } from 'angular2-google-maps/core';
```

En los imports, hay que añadir la API KEY que se genera desde aquí

```
AgmCoreModule.forRoot({
      apiKey: 'YOUR_KEY'
    })
```

La página official de google maps está disponible en castellano

Ejemplos

Desde este link del autor del módulo se puede ver un uso de las principales funciones de trabajar con los mapas, como cargar/actualizar marcadores, evento clic, draggable, etc.

Geolocalización

Html5 dispone de acceso al geolocalizador de un dispositivo, ya sea por ip (PC) o por gelocalización de un dispositivo móvil.

Llamar al método

En el ejemplo, solicita gelocalizarse cuando el componente está cargado

```
ngOnInit(){
       if(navigator.geolocation){
      navigator.geolocation.getCurrentPosition(this.setPosition.bind(this));
      };
   }
```

El cual solicita una llamada al método SetPosition, recoge la latitud y longitud y los establece como marcador inicial en el mapa.

```
setPosition(position){
      this.location = position.coords;
      console.log(position.coords);
      this.lat=position.coords.latitude;
      this.lng=position.coords.longitude;
      this.markers.push({
      lat: this.lat,
      lng: this.lng
```

```
    });
      }
```

Mas interacciones

```
// Importar el núcleo de Angular
import {Component, OnInit} from '@angular/core';
// Decorador component, indicamos en que etiqueta se va a cargar la plantilla
@Component({
    selector: 'contacto',
    templateUrl: 'app/view/contacto.html',
    })

// Clase del componente donde iran los datos y funcionalidades
export class ContactoComponent {
       public titulo:string = "Contacto";
       public geo:any;
       public location:any;
  public MouseEvent:any;
  public markers:any[];
  constructor(){
    this.markers = [
    {
      lat: 51.673858,
      lng: 7.815982,
      label: 'A',
      draggable: true
    },
    {
      lat: 51.373858,
      lng: 7.215982,
      label: 'B',
      draggable: false
    },
    {
      lat: 51.723858,
      lng: 7.895982,
      label: 'C',
      draggable: true
    }
  ]

  }
       // google maps zoom level
  zoom: number = 12;

  // initial center position for the map
  lat: number = 51.673858;
  lng: number = 7.815982;

  setPosition(position){
      this.location = position.coords;
      console.log(position.coords.latitude);
      this.lat=position.coords.latitude;
      this.lng=position.coords.longitude;
      this.markers.push({
      lat: position.coords.latitude,
      lng: position.coords.longitude
    });
```

```
  }


  clickedMarker(label: string, index: number) {
    console.log(`clicked the marker: ${label || index}`)
  }

  mapClicked($event: any) {
    this.markers.push({
      lat: $event.coords.lat,
      lng: $event.coords.lng
    });
  }

  markerDragEnd(m: any, $event: MouseEvent) {
    console.log('dragEnd', m, $event);
  }


  ngOnInit(){
       if(navigator.geolocation){
      navigator.geolocation.getCurrentPosition(this.setPosition.bind(this));
      };
   }
}
```

El template

```
<h2>{{titulo}}</h2>
<sebm-google-map
      [latitude]="lat"
      [longitude]="lng"
      [zoom]="zoom"
      [disableDefaultUI]="false"
      [zoomControl]="false"
      (mapClick)="mapClicked($event)">

      <sebm-google-map-marker
          *ngFor="let m of markers; let i = index"
          (markerClick)="clickedMarker(m.label, i)"
          [latitude]="m.lat"
          [longitude]="m.lng"
          [label]="m.label"
          [markerDraggable]="m.draggable"
          (dragEnd)="markerDragEnd(m, $event)">
        <sebm-google-map-info-window>
          <strong>InfoWindow content</strong>
        </sebm-google-map-info-window>
      </sebm-google-map-marker>
 </sebm-google-map>
```

Mapas con Ionic2

Se puede usar tanto en ionic 2 como en Angular 2, es el método standard para usar google maps sin necesidad de instalar nada previamente llamando a la librería directamente en el index.html de la App, aquí veremos un ejemplo que sirve para cualquier otro APIi que se quiera usar en TypeScript usando una librería externa declarada en el index.html principal.

En index.html se declara el uso del API, para entornos de producción es necesario declarar un API añadiendo ?key=<key_api>, es recomendable también ponerla en entorno de desarrollo ya que pueda saltar el warning en la consola si se usa muchas veces seguidas.

```
<script src="http://maps.google.com/maps/api/js"></script>
```

Usando de base una plantilla en blanco de ionic 2, el home.ts quedaría.

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

import { NavController } from 'ionic-angular';

declare var google;

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {
       @ViewChild('map') mapElement: ElementRef;
  map: any;

  constructor(public navCtrl: NavController) {

  }

   ionViewDidLoad(){
    this.loadMap();
  }

  loadMap(){

    let latLng = new google.maps.LatLng(-34.9290, 138.6010);

    let mapOptions = {
      center: latLng,
      zoom: 15,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    }

    this.map = new google.maps.Map(this.mapElement.nativeElement,
mapOptions);

  }

}
```

Con geolocalización

```
ionic plugin add cordova-plugin-geolocation
```

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

import { NavController } from 'ionic-angular';
import { Geolocation } from 'ionic-native';

//Necesario para poder crear mapas, es la forma de declara una variable de
uso javascript dentro de typeScript
declare var google;

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {
       @ViewChild('map') mapElement: ElementRef;
  map: any;

  constructor(public navCtrl: NavController) {

  }

   ionViewDidLoad(){
    this.loadMap();
  }

  loadMap(){

    Geolocation.getCurrentPosition().then((position) => {

      let latLng = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude);

      let mapOptions = {
        center: latLng,
        zoom: 15,
        mapTypeId: google.maps.MapTypeId.ROADMAP
      }

      this.map = new google.maps.Map(this.mapElement.nativeElement,
mapOptions);

    }, (err) => {
      console.log(err);
    });
  }
}
```

Plantilla

#map es el atributo que se declara para el @ViewChild('map') mapElement: ElementRef;

```
<div #map id="map" style="height: 300px;width: 300px"></div>
```

Añadir Marcadores

Añadir las funciones

```
addMarker(){

        let marker = new google.maps.Marker({
          map: this.map,
          animation: google.maps.Animation.DROP,
          position: this.map.getCenter()
        });

        let content = "<h4>Information!</h4>";

        this.addInfoWindow(marker, content);

            }
      addInfoWindow(marker, content){

        let infoWindow = new google.maps.InfoWindow({
          content: content
        });

        google.maps.event.addListener(marker, 'click', () => {
          infoWindow.open(this.map, marker);
        });

      }
```

En el template

```
<button ion-button block (click)="addMarker()">Añadir Marker</button>
```

Geolocalización por dirección postal

Basados en el [ejemplo que ofrece el propio google] todo se basa en la declaración del objeto de google *let geocoder = new google.maps.Geocoder();* Se le manda la dirección por variable address, geolocaliza y añade marcador en el mapa. En el ejemplo la variable address está definida estática pero podría ser un dataBinding de un elemento de formulario (ngModel)

```
      geocodeAddress() {
              let geocoder = new google.maps.Geocoder();
              var address = 'Carrer dels madrazo, 27, barcelona';
              geocoder.geocode({'address': address}, (results, status) => {
                if (status === google.maps.GeocoderStatus.OK) {
                  this.map.setCenter(results[0].geometry.location);
                  var marker = new google.maps.Marker({
                    map: this.map,
                    position: results[0].geometry.location
                  });
                } else {
                  alert('Geocode was not successful for the following
reason: ' + status);
                }
              });
      }
```