Karta_2.0.js

Armend Azemi

September 2022

1 Introduction

The purpose of this document is to outline the code in the JavaScript file Karta_2.0.js, which is a file that implements GoogleMaps on the page. This file contains the necessary code to enable simple functionality, such as placing markers on the map, making HTML elements responsive to events that interact with the map, and more. Also, the line numbers on the code preview in this document do not match the line numbers in the actual file, all snippets start on line 1, regardless of their line number in the actual file.

All figures in this document are print screens of the JavaScript file using VisualStudio code, thus, the formating will look different if you use WM3's text-editor or if you are using a different theme on VisualStudio Code.

2 Global Variables

At the top of the file we have some global variables, in this section we will cover their meaning and usage.

```
1  //---------------------
2  // GLOBAL VARIABLES
3  //---------------------
4
5  // Google related variabels
6  let focusZoom = 17;
7  let cityZoom = 11;
8  let overviewZoom = 6;
9  var mainMap;
10
11 // DOM variables
12 let resellersParentDiv = document.querySelector('.resellers');
13 let resellerDivs = document.querySelectorAll('.reseller');
14 const searchBar = document.querySelector('#search-city');
15 const autoCompletionList = document.querySelector('.city-list-
       autocomplete');
16
17 // Global storage of all the resellers.
18 let resellersGlobal = [];
19 fetchResellers().then((data) => {resellersGlobal= data})
```

The first three variables in the code-preview above are related to the zoom amount on the map.

- `focusZoom` the zoom when user clicks on a specific reseller, either map marker or HTML element
- `cityZoom` is used for when user searches for a specific city, in which there are resellers available. Gives a city-overview perspective.
- `overviewZoom` this is the broadest overview zoom, and is used on different occasions:
    1. When the map first loads.
    2. If a user is zoomed in a specific marker on the map, and then clicks the marker/DOM element again.
    3. When transitioning between markers and cities.

The DOM element variables are pretty self-explanatory, so we won't dwell any deeper into their meaning. One thing to point out is that the structure of the DOM has to be the same.

- `resellers` (Container of all the reseller divs)
    - `reseller` (Individual reseller div, containing reseller information

The final global variable is `resellersGlobal` which is an `object` type. It contains all the data from the resellers. The data is retrieved from the WM3 list on the page, in this case the list is called 'Återförsäljare'. We will talk more about this once we get to the function `fetchResellers()`. This variable is used when we want access to the resellers information, thus, we will only fetch the data once from the page list.

3 initMap()

This section will cover the functionality of the function `initMap()`. It's important to notice that this function name is has to be indentical to the function name declared in the imported script tag from the Google Maps API, as shown below:

```
<script src="https://maps.googleapis.com/maps/api/js?key=
    YOUR_API_KEY&callback=initMap&v=weekly" defer></script>
```

Notice the key `callback=` and the value being `initMap`

```
async function initMap() {
console.log('InitMap Called....');
// Get the coordinates from the resellers, fetched from  'lists.
    aterforsaljre.lat_lng'.
let resellerCoordinates = await fetchResellersCoordinates();


// Set the map options, the center of the map and the zoom on load.
let mapOptions = {
center: new google.maps.LatLng('57.78594750386966', '
    14.162155747193367'),
zoom: overviewZoom
}
// Create a map and initilize it in the div '#map'
let map = new google.maps.Map(document.getElementById('map'),
    mapOptions);
mainMap = map

// Loop over all the coordinates and create markers for them on the
     map.
resellerCoordinates.forEach((item)=> {

// The coordinate are in 'xxxxxxxx, xxxxxx' format. Split and get
    Longitude, and Langtiude by themselves.
let longAndLang = item.split(',');
let long = longAndLang[0];
let lang = longAndLang[1];

// Create markers
let markerOptions = {
position: new google.maps.LatLng(long, lang),
map: map
}
let marker = new google.maps.Marker(markerOptions);


// Add EventListner to the marker. On click, center the map on the
    marker on zoom in.
marker.addListener('click', () => {
map.setCenter(marker.getPosition());
map.panTo(marker.getPosition());

// If already zoomed in, zoom out.
```

```
38 if (map.getZoom() == focusZoom - 1) {
39 smoothZoomOut(map, overviewZoom, map.getZoom());
40 }else {
41 smoothZoomIn(map, focusZoom, map.getZoom());
42 }
43 });
44
45 });
46
47 // Enable all the 'Reseller' divs click functionality.
48 makeResellerDivsClickableOnMap(map)
49 }
```

This function is responsible for initializing the map and populating it with the pin markers. We will skip the first 7 lines of the code preview above, since we will discuss the function `fetchResellerCoorinates()` in a later section.

On line 8 we declare `mapOptions` that has two values, `center` and `zoom` that are used by the Google Map, the centering of the map, and the zoom in which the map should be displayed.

On line 13 we create the `map` object and populate the DOM element with the id 'map', this is a must, and we also call the constructor with the `mapOptions` defined previously.

On line 17 we loop over all the coordinates of the resellers. This data is fetched from the page list with the function `fetchResellersCoordinates()`, which returns an string array. We then extract the Longitude and Latitude coordinates to construct `google.maps.Markers`. The `google.maps.Marker` takes two necessary values, the position of the marker and the map object.

The rest of the code is well documented, and we will discuss the functions in sections below.

4 smoothZoomIn() and smoothZoomOut()