2d Assignment for the course Social Media Ecosystems

Georgios Akritidis
840703-R171
ga222ey@student.lnu.se
Department of Media Technology
Linnaeus University, Vaxjo

19 December, 2016

Abstract

This assignment is the second for the course *Social Media Ecosystems* and aims to make the student capable of implementing APIs for social media tools. Specifically, for this project I had to implement a mash up which would extend an already made web project. The theme of the assignment is about informations related to the city Vaxjo that the visitor of the web page can obtain. The information regards mainly places that a tourist can visit like restaurants and coffee shops but also videos and articles that are published in social media.

INTRODUCTION

For this web project my aim was to implement 3 APIs that would be added to the mash up that we have been given. The functions of the APIs as I mentioned give informations about the facilities of Vaxjo, thus this project can be seen also as tourist guide as well. For this assignment I used the most popular and often used APIs because it is easier to find tutorials to develop the code. Specifically, I made use of Google Places API to provide the location of the restaurants and the coffee shops of the city, YouTube Data API to display videos related to Vaxjo and Wikipedia API to display articles that are published. Of course, the visitor of the page can search for anything else, by typing another keyword in the appropriate field.

In this document, I will briefly describe the methodology that I followed, provide part of the code I used and give references of the sources I used to implement the project.

METHODOLOGY

General Implementation

The structure of the web project was designed and developed from scratch since I embedded the code that was available to us in a new template. The idea was to display the widgets that were offered to us in the front page and the rest that I would create in separated pages. The new widgets can be navigated through the main task bar that is located in the top of the page.

The main programming languages that I used are HTML, Javascript and Jquery. For the shaping of the page I used CSS and Bootstrap. Since I had to combine a variety of social media I create web api keys for all the APIs I made use of. Flickr, Wolfram, Google, Wunderground are some of social media tools that I created an API secret key.

During the construction of the site I was uploading the data on my localhost in order to save time and effort. In the end, I uploaded my files in the web host www.000webhostapp.com with the domain name ecosystem.000webhostapp.com.

Limitations

The first problem that I encountered when making the assignment was to find the perfect host for the project. I prefer to use the host x10hosting.com but this specific host does not provide solutions for anyone that is located in Greece, so I had to exclude it from my list of available hosts and move to the one that I aforementioned.

Another limitation that I faced, regarded the Google Places API. Specifically, for the implementation of the map that displays the restaurants in Vaxjo, I had to deal with the problem that Google supports the display of maximum 20 markers. Although, I made great efforts to find a solution for this restriction I did not manage to resolve the problem. So, in my application the visitor can see only the 20 restaurants that are closest to the coordinates that I inserted. Nevertheless, I modified the code with another feature that Google offers, so that the viewer can see up to 200 spots. Yet, also for this, there is another limitation which is that the viewer can see only the name of the place and not more details like the address of the id of the place. I used this new method for the display of the cafeterias.

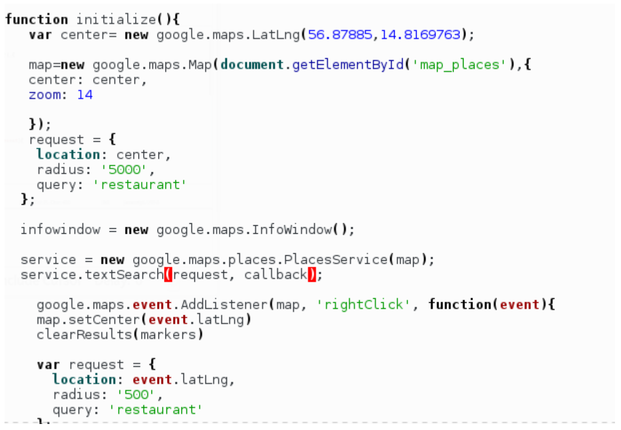
For the YouTube Data API the only thing I could not overcome is to display the buttons *'Previous'* and *'Next'* in the bottom of the page when the visitor clicks the link Media in the taskbar. Despite this, the visitor can type in the field the keyword that he wishes and afterwards a variety of YouTube videos will be displayed with the buttons available.

Finally, for the Wikipedia API there is a minor problem which is that after a new search the new links are being added with the previous ones in the same array. This mean practically, that after many searches a long list of links would be created.

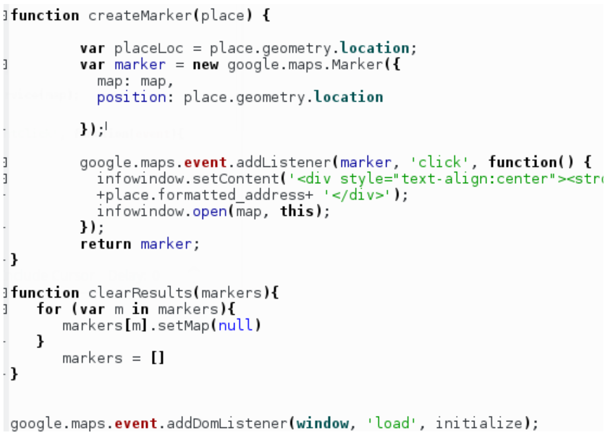
PRESENTATION

In this section I will present the template of my site as well as the structure of my code. Since I do not have much experience with server-side programming, I restricted my code to client-side functions.

First of all, I will present the Google Place API that I used for the displaying of the restaurants by giving you pictures of the code and the google map. Initially, I had to create a method (initialize) to define the coordinates and the parameters of the API and afterwards another function to produce the markers.

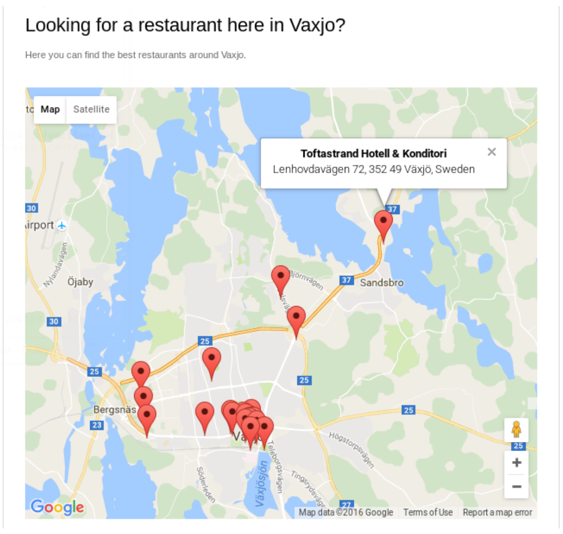


```
function initialize(){
   var center= new google.maps.LatLng(56.87885,14.8169763);

   map=new google.maps.Map(document.getElementById('map_places'),{
   center: center,
   zoom: 14

   });
   request = {
    location: center,
    radius: '5000',
    query: 'restaurant'
  };

  infowindow = new google.maps.InfoWindow();

  service = new google.maps.places.PlacesService(map);
  service.textSearch(request, callback);

    google.maps.event.AddListener(map, 'rightClick', function(event){
    map.setCenter(event.latLng)
    clearResults(markers)

    var request = {
      location: event.latLng,
      radius: '500',
      query: 'restaurant'
```



```
function createMarker(place) {

        var placeLoc = place.geometry.location;
        var marker = new google.maps.Marker({
          map: map,
          position: place.geometry.location

        });

        google.maps.event.addListener(marker, 'click', function() {
          infowindow.setContent('<div style="text-align:center"><str
          +place.formatted_address+ '</div>');
          infowindow.open(map, this);
        });
        return marker;
}

function clearResults(markers){
   for (var m in markers){
      markers[m].setMap(null)
   }
      markers = []
}


google.maps.event.addDomListener(window, 'load', initialize);
```

The result of this code can be viewed in the picture that is depicted below. As you can see, there are 20 markers which represent restaurants in the center of Vaxjo.



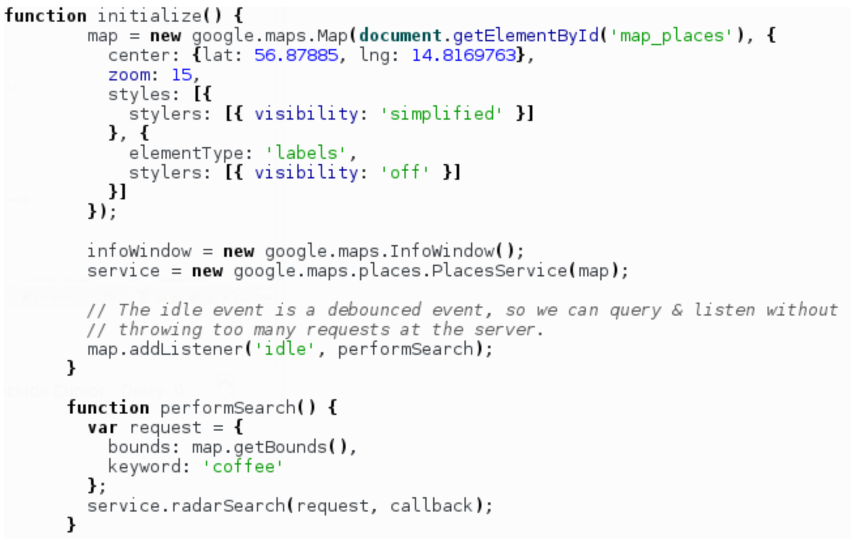
Looking for a restaurant here in Vaxjo?

Here you can find the best restaurants around Vaxjo.

Toftastrand Hotell & Konditori
Lenhovdavägen 72, 352 49 Växjö, Sweden

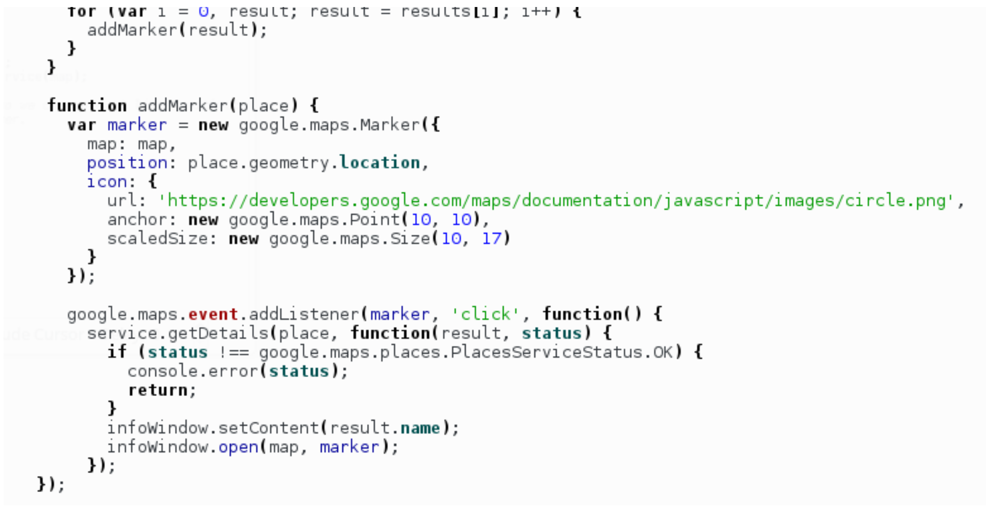
I will demonstrate now part of the code I used for the implementation of the cafeterias google places map. In this case, the number of spots that are displayed in the map, is increased to 200 hundreds.

As in the previous example, I created two main functions; the function initialize() and the function addmarker(). You can see below part of the code for each method.



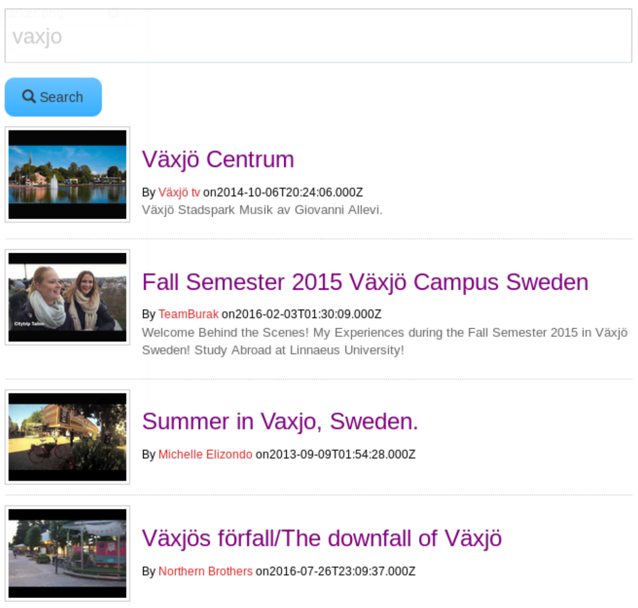
```
function initialize() {
        map = new google.maps.Map(document.getElementById('map_places'), {
          center: {lat: 56.87885, lng: 14.8169763},
          zoom: 15,
          styles: [{
            stylers: [{ visibility: 'simplified' }]
          }, {
            elementType: 'labels',
            stylers: [{ visibility: 'off' }]
          }]
        });

        infoWindow = new google.maps.InfoWindow();
        service = new google.maps.places.PlacesService(map);

        // The idle event is a debounced event, so we can query & listen without
        // throwing too many requests at the server.
        map.addListener('idle', performSearch);
      }

      function performSearch() {
        var request = {
          bounds: map.getBounds(),
          keyword: 'coffee'
        };
        service.radarSearch(request, callback);
      }
```

The cafeterias are projected with red circles and if the viewer clicks on it, he can see the name of the coffee shop but not its address.



```
    for (var i = 0, result; result = results[i]; i++) {
      addMarker(result);
    }
  }

  function addMarker(place) {
    var marker = new google.maps.Marker({
      map: map,
      position: place.geometry.location,
      icon: {
        url: 'https://developers.google.com/maps/documentation/javascript/images/circle.png',
        anchor: new google.maps.Point(10, 10),
        scaledSize: new google.maps.Size(10, 17)
      }
    });

    google.maps.event.addListener(marker, 'click', function() {
      service.getDetails(place, function(result, status) {
        if (status !== google.maps.places.PlacesServiceStatus.OK) {
          console.error(status);
          return;
        }
        infoWindow.setContent(result.name);
        infoWindow.open(map, marker);
      });
});
```

For the implementation of the YouTube Data API, I had to create an API key which I did in the google developers console. I used this key because, otherwise, I couldn't display the videos online but only on my localhost. Also, I created two buttons in order for the viewer to navigate back and forwards. Below there is a picture of what my page looks like.


vaxjo

Search

Växjö Centrum

By Växjö tv on2014-10-06T20:24:06.000Z
Växjö Stadspark Musik av Giovanni Allevi.

Fall Semester 2015 Växjö Campus Sweden

By TeamBurak on2016-02-03T01:30:09.000Z
Welcome Behind the Scenes! My Experiences during the Fall Semester 2015 in Växjö Sweden! Study Abroad at Linnaeus University!

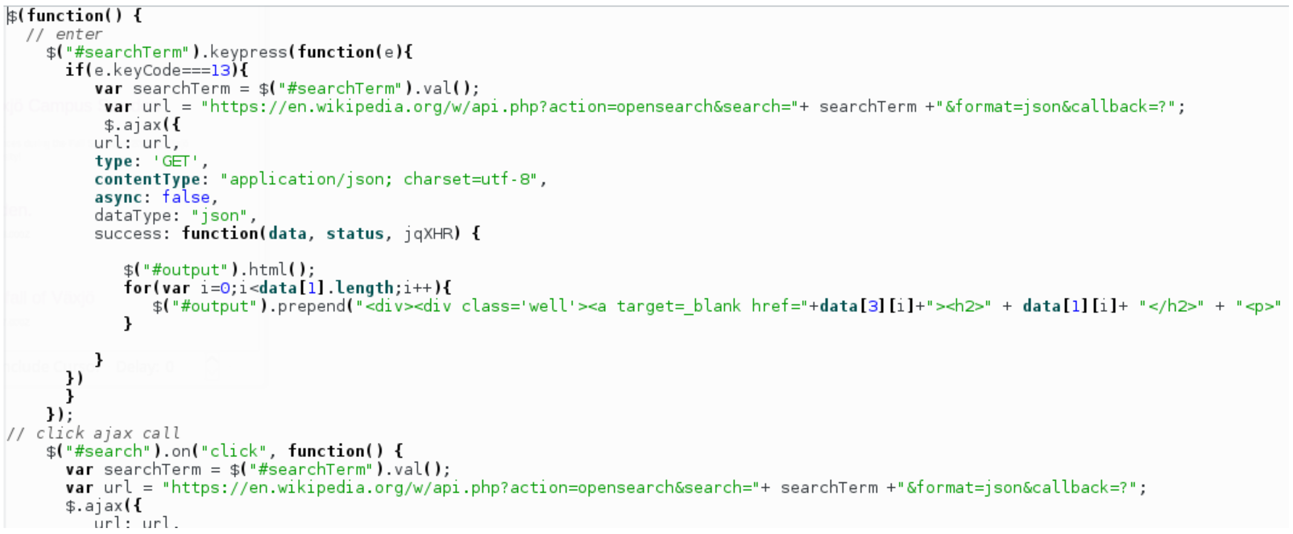
Summer in Vaxjo, Sweden.

By Michelle Elizondo on2013-09-09T01:54:28.000Z

Växjös förfall/The downfall of Växjö

By Northern Brothers on2016-07-26T23:09:37.000Z

Finally, for the Wikipedia API which was more simple I wrote two functions which firstly search for the data and then project it appropriately. Part of the code can be viewed below.



```
$(function() {
  // enter
    $("#searchTerm").keypress(function(e){
      if(e.keyCode===13){
         var searchTerm = $("#searchTerm").val();
          var url = "https://en.wikipedia.org/w/api.php?action=opensearch&search="+ searchTerm +"&format=json&callback=?";
          $.ajax({
         url: url,
         type: 'GET',
         contentType: "application/json; charset=utf-8",
         async: false,
         dataType: "json",
         success: function(data, status, jqXHR) {

            $("#output").html();
            for(var i=0;i<data[1].length;i++){
               $("#output").prepend("<div><div class='well'><a target=_blank href="+data[3][i]+"><h2>" + data[1][i]+ "</h2>" + "<p>"
            }

         }
      })
      }
    });
// click ajax call
    $("#search").on("click", function() {
      var searchTerm = $("#searchTerm").val();
      var url = "https://en.wikipedia.org/w/api.php?action=opensearch&search="+ searchTerm +"&format=json&callback=?";
      $.ajax({
         url: url,
```

CONCLUSIONS

This assignment has been rather enlightening to me since I obtained a broader view of how the world of social media is. There is a wide range of social media tools that an ambitious developer can use in order to produce a project that will not be only useful to the visitor but also lucrative to the developer. Of course, to achieve a goal like this, a lot of effort is required in order to improve the programming skills and to obtain the knowledge that is needed to use this tools effectively.

In the future, I plan to expand my work even further so that I can enhance more features and more complicated functions. My primary goal is to develop server-side widgets from several API developers such as YELP or SKYSCANNER.

REFERENCES

1. https://developers.google.com/maps/documentation

2. https://developers.google.com/youtube/v3/

3. https://www.mediawiki.org/wiki/API:Main_page

4. https://www.youtube.com/watch?v=-vH2eZAM30s

5. https://www.youtube.com/watch?v=O4yAHlE3HNs&t=768s