

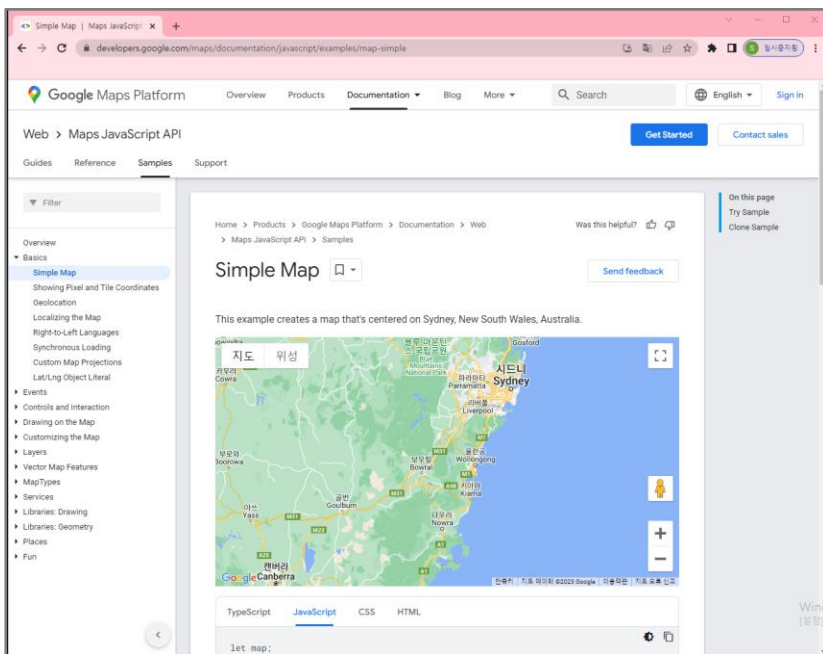
Making Your HTML Location Aware: Geolocation

AIzaSyCIwcnmj5PcfmncGWA-0UCdLiZVQ1tDhZ4

Geolocation API

▶ Geolocation API

- 페이지를 실행하는 브라우저가 탑재된 디바이스의 위치 정보를 확인
- 모바일웹 애플리케이션에 유용하게 사용
- GPS가 없는 디바이스일 경우 네트워크 정보로 대략적인 위치 추측



구글검색 : 구글 개발자

<https://developers.google.com/maps/documentation/javascript/tutorial>
https://www.w3schools.com/graphics/google_maps_intro.asp



구글검색 : 카카오개발자

<https://apis.map.kakao.com/web/sample/>

Location, Location, Location

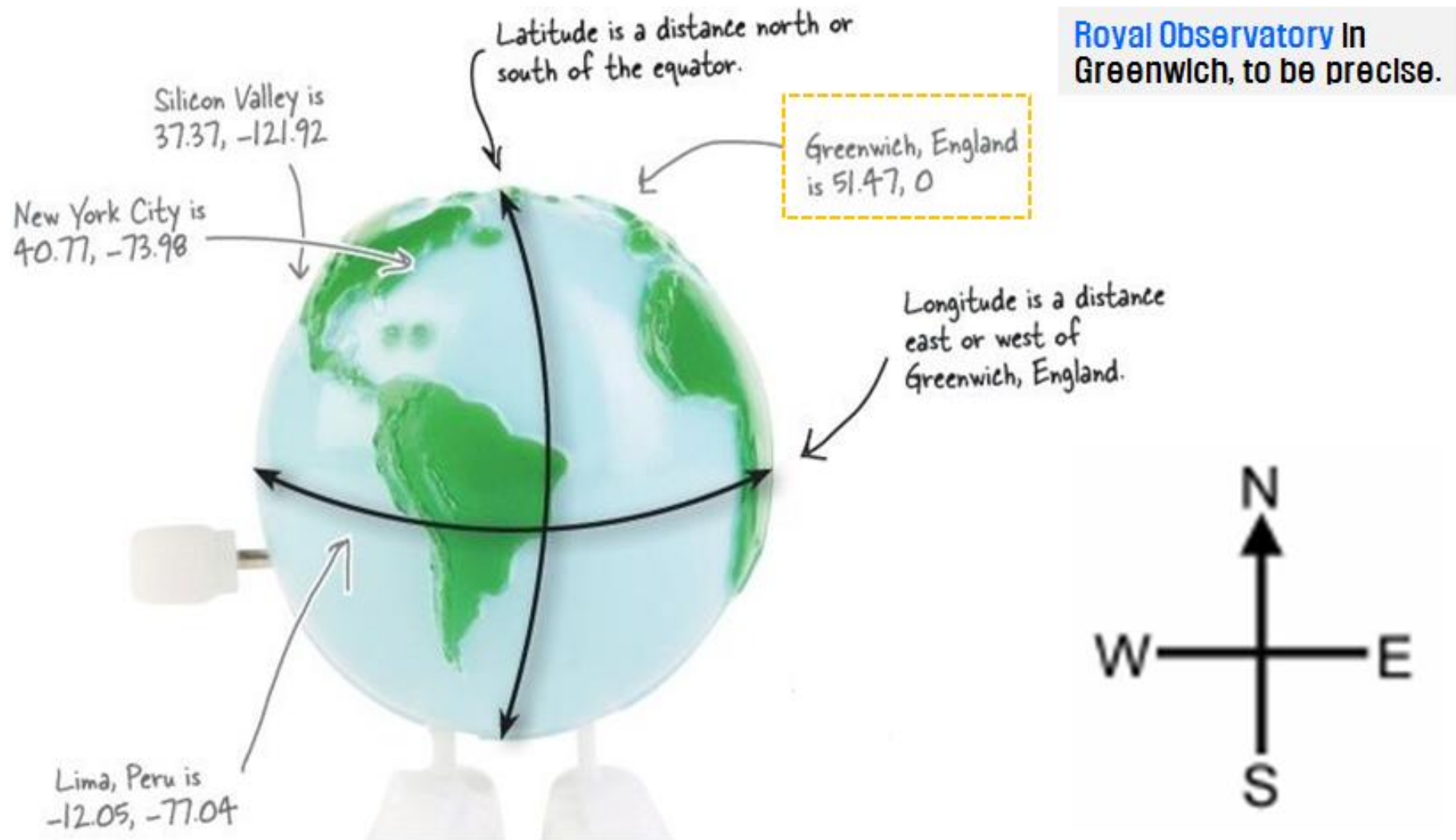
- 앱 사용자의 위치를 아는 것은 웹 경험(web experience)에 많은 것을 추가할 수 있다:
 - 사용자에게 방향을 알려 준다거나
 - 이동할 수 있는 장소에 관해 제시한다거나
 - 비가 오고 있으니 실내 활동을 제안한다거나 등등위치정보(location information)를 이용할 수 있는 방법은 무한하다
- HTML5의 **Geolocation JavaScript-based API**를 이용하면 웹 페이지에서 위치정보를 쉽게 접근(access)할 수 있다

시작하기에 앞서 **위치**에 관해 알아야 할 몇 가지 사항이 있다

위도(latitude)와 경도(longitude)

- 현재의 위치를 알기 위해서는 지구표면에 대한 좌표계(coordinate system)가 필요하다.
- 대표 좌표계: 위도(latitude)와 경도(longitude)
- 위도(latitude)는 적도(equator)를 기준으로 남북으로 떨어진 거리이고
- 경도(longitude)는 영국(Greenwich, England)의 그리니치를 기준으로 동서로 떨어진 거리이다.
- 위도(latitude)의 남쪽과 경도(longitude)의 서쪽 좌표는 음수 값으로 표현 된다.
- Geolocation API : 좌표계를 이용하여 언제 어디에 있던 현재위치에 대한 좌표제공

Lat(위도) and Long(경도) of it...



- 위도(latitude)와 경도(longitude)

Latitude / Longitude Closeup(위도경도 자세히 보기)

- 위도와 경도를 (47°38'34", 122°32'32")처럼 도/분/초로 표시하거나 (47.64, -122.54)처럼 십진수형태로 표시할 수 있다.
- 지오로케이션 API는 항상 십진수 형태로 좌표를 알려줍니다.

*“With the **Geolocation API** we always use **decimal** values. ”*

- 도/분/초를 십진수로 변환하고 싶다면 아래의 함수를 사용한다.

Decimal로 만드는 공식

```
function degreesToDecimal(degrees, minutes, seconds) {  
    return degrees + (minutes/60.0) + (seconds/3600.0);  
}
```

Also notice that longitude(경도) **West** and latitude(위도) **South** are represented by **negative(음수)** values.

How **Geolocation API** determines your location

- **데스크탑 브라우저**에서도 위치를 파악(aware)할 수 있다
- GPS 장치가 없는데도 데스크탑 브라우저가 어떻게 **위치를 결정**할 수 있을까?
- 모든 장치의 브라우저들은 현재 위치를 결정하기 위해 **몇 가지 방법을 사용**하고 있다:
장치마다 **정확도**가 다르다

How the **Geolocation** API determines your location

IP Address

- IP 주소 기반 위치 정보는 **외부 데이터베이스**를 **이용**하여 IP주소를 물리적 위치에 매핑시킨다
- 장점은 어느 곳에서든지 동작된다는 점이다

우리 사무실에는 최신기기라
곤 하나도 없어요. Ip 주소와
위치를 연결하는데 가끔식은
꽤 정확 하더군요



How **Geolocation** API determines your location

GPS

- **인공위성**(satellites)에 기반하여 극히 정확한 위치 정보를 제공한다
- 위치 데이터는 고도정보(altitude), 속도정보(speed), 헤드정보(heading)를 포함할 수 있다
- 그러나 하늘을 볼 수 있어야만 하고 위치 찾는데 오래 걸린다



How the **Geolocation** API determines your location

Cell Phone

- **Cell phone 삼각측정법**(triangulation)은 하나 이상의 **셀폰 타워**(cell phone tower)와의 거리에 기반하여 현재 위치를 계산한다
 - 타워가 많을수록 더 정확하다
- 매우 정확하다
- GPS와는 달리 **Indoor**에서도 위치를 파악할 수 있다
- GPS보다 빠르다

구식 핸드폰도
좋아좋아~



How the **Geolocation** API determines your location

무선 통신을 이용해서
노트북을 들고 커피숍을
 옮겨 다니면서 작업하니 편해요
무선통신장치를 이용한
삼각측량법으로
제 위치를 알 수 있죠.



WiFi

- **와이파이 포지셔닝**(WiFi positioning): 하나 이상의 **액세스 포인트**(Access Point, AP)를 이용하여 위치를 **삼각측량**한다
- 매우 정확하다
- Indoors에서 동작한다
- 빠르다



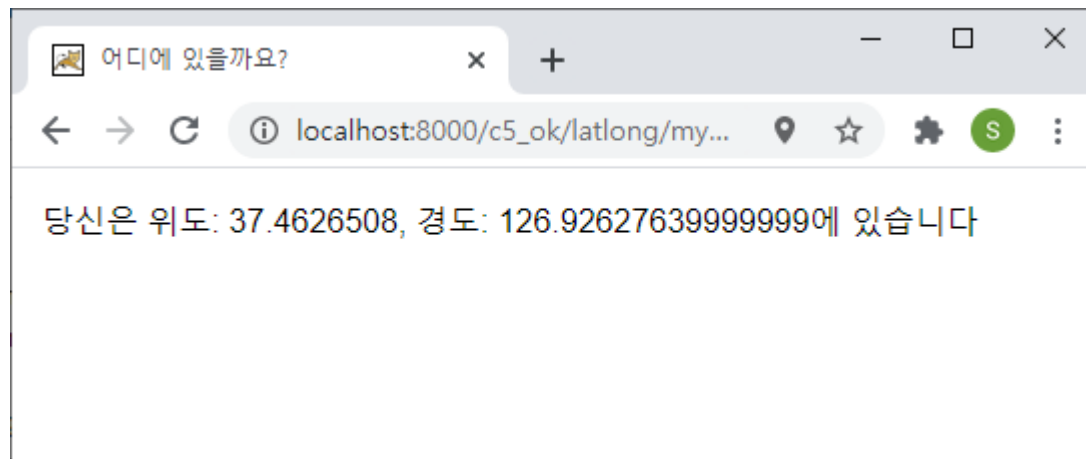
알 수 없습니다,
You' re not.
그럼 위치를
파악하는 방법은?

You're not.

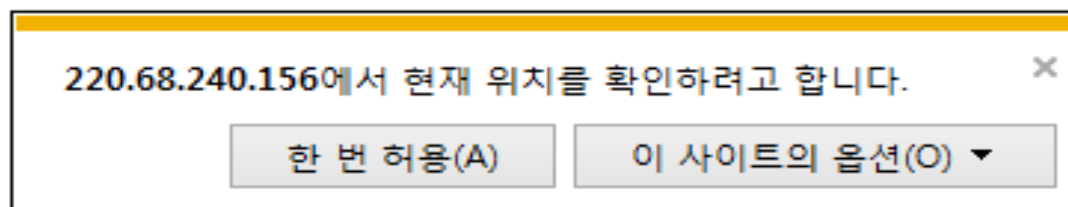
- 브라우저 구현방법(browser implementation)에 따라 위치 결정 방법이 결정된다
- 브라우저는 위치 결정을 위해 앞에서 제시된 방법 중의 어느 것이라도 사용할 수 있다
- 사실 스마트 브라우저는 처음에는 셀론 삼각측량법을 사용해서 개략적인 위치 파악을 한 후 WiFi나 GPS를 이용하여 더 정확한 위치정보를 제공한다
- 개발자는 위치 결정 방법에 관하여 걱정할 필요가 없다
- 대신 위치의 정확도(accuracy)에 집중할 필요가 있다
 - 정확도에 따라 위치 정보를 얼마나 유용하게 사용할 수 있는지가 결정됩니다.
 - 채널 고정! 잠시 후에 정확도에 관해 더 알아보겠습니다!

<실습1> 여러분은 어디에 있나요? – location

나의 위치(latitude, longitude) 가져오기



Your location will go here.



http://localhost:8000/c5_ok/latlong/myLoc.html

<실습1-1> 그런데 여러분은 어디에 있나요? – location

자, 우리는 우리가 어디에 있는지 알고 있다. 그러나 [브라우저](#)가 우리의 위치를 어떻게 생각하고 있는지 살펴 보자

```
1  <!doctype html>                                     myLoc.html
2  <html>
3  <head>
4  <title>어디에 있을까요?</title>
5  <meta charset="utf-8">
6  <script src="myLoc.js"></script>
7  <link rel="stylesheet" href="myLoc.css">
8  </head>
9  <body>
10 <div id="location">
11 여러분의위치가 여기에 나타날 것입니다.
12 </div>
13 </body>
14 </html>
```

Does this browser support it? – 브라우저 지원여부

- **geolocation** code를 작성하기 위한 첫 번째 확인 사항:
 - 브라우저에서 geolocation이 지원될 때만 navigator 객체에 geolocation 프로퍼티를 가진다
 - geolocation 프로퍼티가 존재하는지 알아보려면?

```
if (navigator.geolocation) {  
    ...  
} else {  
    alert("Oops, no geolocation support");  
}
```


navigator.geolocation 사용

- `navigator.geolocation` **프로퍼티**는 완전한 Geolocation API를 포함하는 객체이다.
 - `navigator.geolocation` 의 메인 메소드가 `getCurrentPosition`으로 브라우저의 위치를 가져 온다.

`getCurrentPosition(successHandler, errorHandler, options)`



```
navigator.geolocation.getCurrentPosition(  
    displayLocation,  
    displayError);
```

<실습1-2> getMyLocation()

```
function getMyLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(
            displayLocation,
            displayError);
    }
    else {
        alert("이런, 지오로케이션이 제공되지 않네요");
    }
}
```

myLoc.js
//추가1

<실습1-3> displayLocation() - successHandler

```
function displayLocation(position) {  
    var latitude = position.coords.latitude;  
    var longitude = position.coords.longitude;  
  
    var div = document.getElementById("location");  
    div.innerHTML = "당신은 위도: "  
        + latitude + ", 경도: " + longitude + "에 있습니다";  
}
```

myLoc.js
//추가2

<실습1-4> displayError() - errorHandler

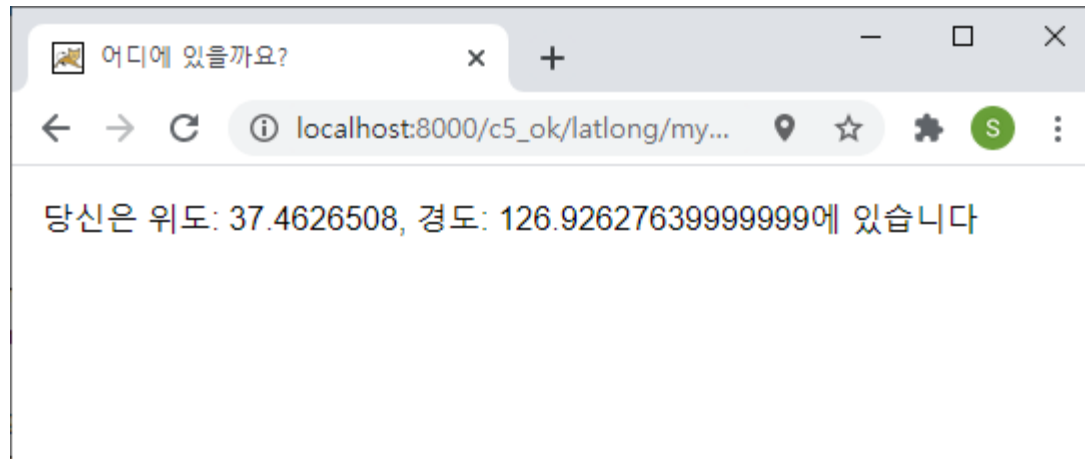
```
function displayError(error) {  
    var errorTypes = {  
        0: "알려지지 않은 에러",  
        1: "사용자가 권한 거부",  
        2: "위치를 찾을 수 없음",  
        3: "요청 응답 시간 초과"  
    };  
    var errorMessage = errorTypes[error.code];  
    if (error.code == 0 || error.code == 2) {  
        errorMessage = errorMessage + " " + error.message;  
    }  
    var div = document.getElementById("location");  
    div.innerHTML = errorMessage;  
}
```

myLoc.js
//추가3

작동과정



<실습1-5> 실행화면 - latlong



http://localhost:8000/c5_ok/latlong/myLoc.html

<실습2> 거리계산 위한 코드 작성 - distance

위치의 비밀을 밝혀 봅시다...

Wickedly Smart HQ (본부)에서 위치 정보를 작성하는 비밀을 알려드릴까요?

- ✓ Needs the [HQ coordinates\(좌표\)](#)
- ✓ Needs to know [how to calculate distance between two coordinates](#)

처음으로 또다른 <div>를 만들어 보자:

```
<body>
  <div id="location">
    Your location will go here.
  </div>
  <div id="distance">
    Distance from WickedlySmart HQ will go here.
  </div>
</body>
</html>
```



Wickedlysmart HQ와의 거리가 여기에 나타날 것입니다.

<실습2-1> 거리계산한 결과를 보여주기 위한 영역

myLoc.html

<!--추가1-->

```
1  <!doctype html>
2  <html>
3  <head>
4  <title>어디에 있을까요?</title>
5  <meta charset="utf-8">
6  <script src="myLoc.js"></script>
7  <link rel="stylesheet" href="myLoc.css">
8  </head>
9  <body>
10
11  <div id="location">
12  여러분의 위치가 여기에 나타날 것입니다.
13  </div>
14
15  <div id="distance">
16  WickedlySmart HQ (위키들리스마트 본부) 와의 거리가 여기에 나타날 것입니다.
17  </div>
18
19  </body>
20  </html>
```


WickedlySmart HQ(본부)의 좌표

두 지점간의 거리를 계산하는 함수가 확보 됐으니 wickedly smart본부에서 우리의 위치를 정의하여 봅시다. 우선 WickedlySmart HQ(본부)의 좌표를 알아봅시다

```
var ourCoords = {  
    latitude: 47.624851,  
    longitude: -122.52099  
};
```

Wickedly smart
본부 좌표



거리 계산을 위한 코드 작성하기 - distance

이제 코드를 작성하여 봅시다. 해야 할 일은 Computer Distance함수에 우리와 여러분의 위치좌표를 넘기는 것 뿐입니다.

```
function displayLocation(position) {  
    var latitude = position.coords.latitude;  
    var longitude = position.coords.longitude;  
  
    var div = document.getElementById("location");  
    div.innerHTML = "You are at Latitude: " + latitude + ", Longitude: " + longitude;  
  
    var km = computeDistance(position.coords, ourCoords);  
    var distance = document.getElementById("distance");  
    distance.innerHTML = "You are " + km + " km from the WickedlySmart HQ";  
}
```



나의 좌표



Wickedly smart좌표

<실습2-3> 거리 계산을 위한 코드 작성하기 – distance

```
var ourCoords = {  
  latitude: 47.624851,  
  longitude: -122.52099  
};
```

myLoc.js
//추가1

```
var km = computeDistance(position.coords, ourCoords);  
var distance = document.getElementById("distance");  
distance.innerHTML = "당신은 WickedlySmart HQ와 " + km + "km 떨어져 있습니다";
```

myLoc.js
//추가2

<실습2-2> **computing distance**– Some Ready Bake Code – 준비된 코드

```
// 구면 코사인 법칙으로 두 위도/경도 지점의 거리를 구함
//
function computeDistance(startCoords, destCoords) {
    var startLatRads = degreesToRadians(startCoords.latitude);
    var startLongRads = degreesToRadians(startCoords.longitude);
    var destLatRads = degreesToRadians(destCoords.latitude);
    var destLongRads = degreesToRadians(destCoords.longitude);

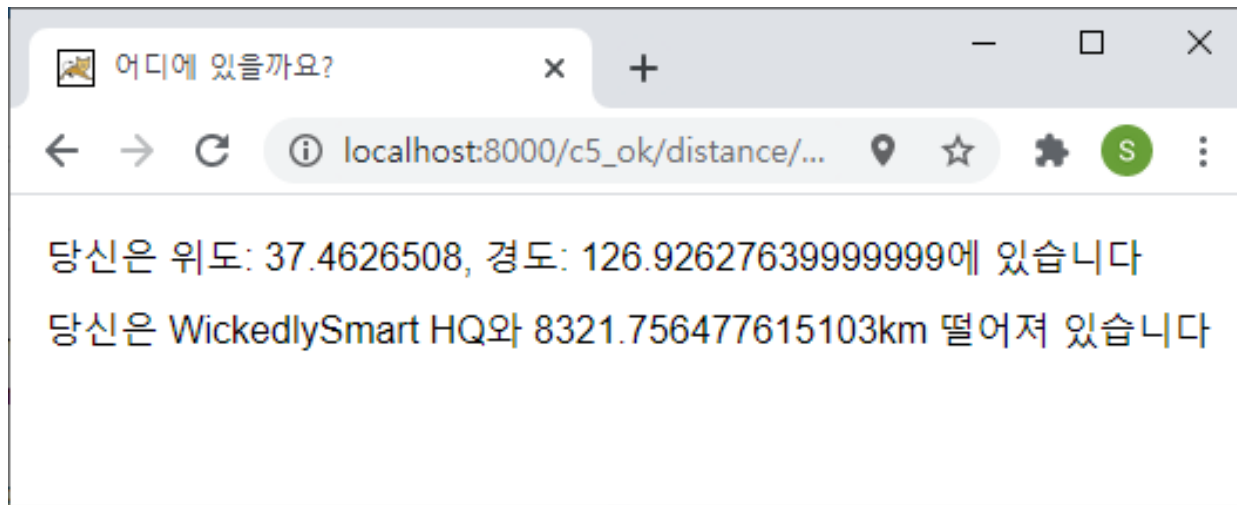
    var Radius = 6371; // 지구의 반경 (km)
    var distance = Math.acos(Math.sin(startLatRads) * Math.sin(destLatRads) +
                             Math.cos(startLatRads) * Math.cos(destLatRads) *
                             Math.cos(startLongRads - destLongRads)) * Radius;
    return distance;
}

function degreesToRadians(degrees) {
    radians = (degrees * Math.PI)/180;
    return radians;
}
```

myLoc.js
//추가3

<실습2-4> 실행화면 - distance

위치 정보와 거리계산 보여주기



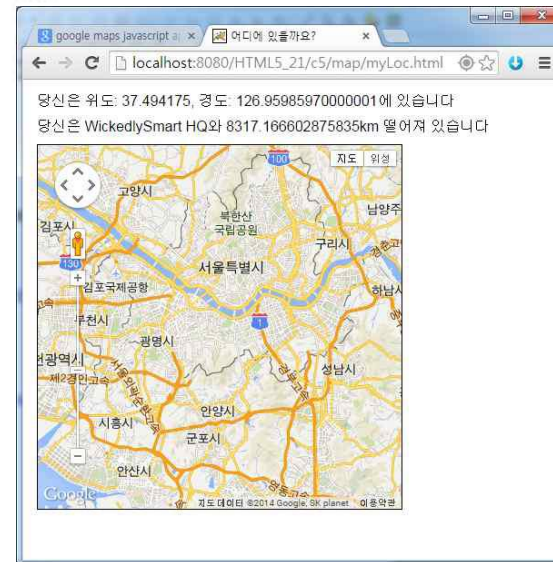
http://localhost:8000/c5_ok/distance/myLoc.html

<실습3> 페이지에 구글지도 추가 – 사전작업

- Geolocation API는 매우 간단합니다.
 - 분명히 구글지도는 HTML5 명세서에 포함되어 있지는 않지만 HTML5와 궁합이 잘 맞으며
 - 여기서는 구글지도는 변형하지 않고 지오로케이션 API와 통합하는 방법을 소개
- 만약에 구글지도를 사용하려면 HTML 문서의 head 부분에 아래 문장을 추가:
- Maps javascript API key : AIzaSyA7nnoG8FxFJ8CcMFYcE_isUIH-GUFvu4jA <= 각자 만들기

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

This is the location of the
Google Maps JavaScript API



<실습3-1> 페이지에 구글 지도 추가하기 - map

- 구글지도 API를 연결 했으니, 자바스크립트 모든 구글지도의 기능을 사용할 수 있다.
 - 구글지도를 놓을 자리가 필요
 - 구글지도를 포함할 요소 하나를 정의

```
<scriptmyLoc.html  
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCIwcnmj5PcfmncGWA-  
OUCdLiZVQ1tDhZ4&callback=initMap&v=weekly"></script>  
<!--추가1-->
```

또는 카드를 등록하여 개인의 KEY 를 사용하세요

```
<div id="map">  
</div>  
myLoc.html  
<!--추가2-->
```

지도 생성을 위한 준비 작업 1

- Needs two things:
 - `latitude/longitude`
 - `options` that describe how we want the map created
- 먼저 위도/경도에 대해 살펴보자:
 - Google API는 **위도/경도** 를 객체 자체에 번들로 제공 – 묶음 제공
- 위도/경도 객체 생성을 위해 Google이 제공하는 **생성자**(constructor) 사용:

```
var googleLatAndLong = new google.maps.LatLng(latitude, longitude);
```

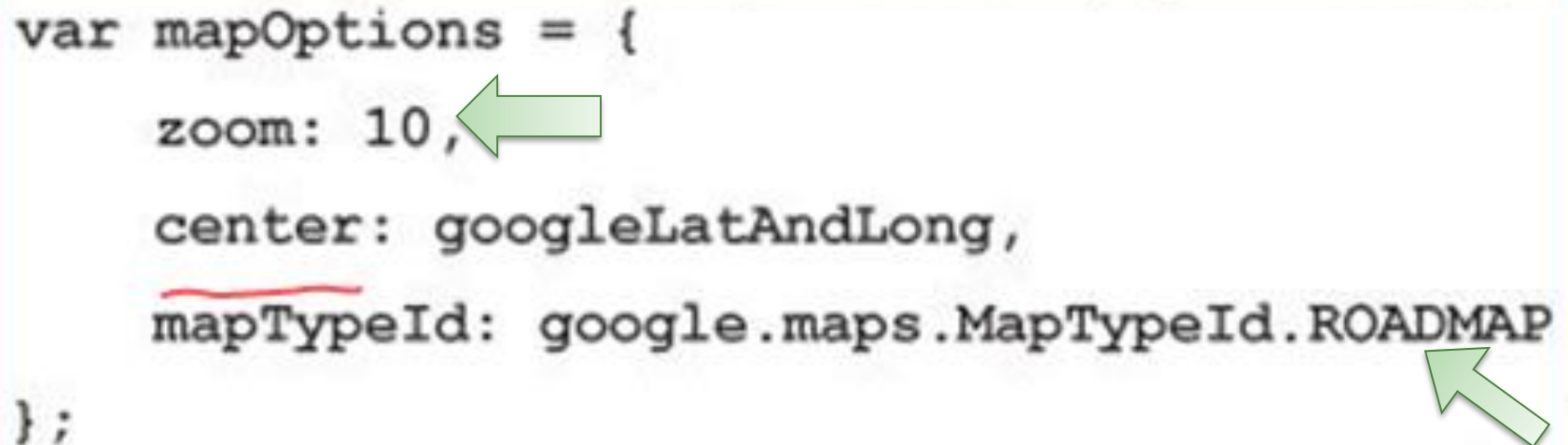
**google.maps는 Google Maps API의
모든 메소드 앞에 붙는다**

지도 생성을 위한 준비 작업 2

- Google 맵이 어떻게 생성되어야 하는지를 제어할 수 있게 설정할 수 있는 옵션 제공

Options을 만들어 보자:

```
var mapOptions = {  
    zoom: 10,  
    center: googleLatAndLong,  
    mapTypeId: google.maps.MapTypeId.ROADMAP  
};
```

A diagram showing a code block for mapOptions. The code is: var mapOptions = { zoom: 10, center: googleLatAndLong, mapTypeId: google.maps.MapTypeId.ROADMAP };. There are two green arrows: one pointing to the value 10 in the zoom property, and another pointing to the value google.maps.MapTypeId.ROADMAP in the mapTypeId property. The word 'mapTypeId' is underlined with a red line.

<실습3-2> 지도를 보여줍니다 – showMap() 만들기

showMap, 이라는 함수에 이것들을 모두 담아봅시다:

```
function showMap(coords) {  
    var googleLatAndLong = new google.maps.LatLng(coords.latitude,  
                                                    coords.longitude);  
  
    var mapOptions = {  
        zoom: 10,  
        center: googleLatAndLong,  
        mapTypeId: google.maps.MapTypeId.ROADMAP  
    };  
    var mapDiv = document.getElementById("map");  
    map = new google.maps.Map(mapDiv, mapOptions);  
}
```

myLoc.js
//추가1

Google API

<실습3-3> 지도를 보여줍니다 – displayLocation() 만들기

- **displayLocation** 함수를 수정합니다.

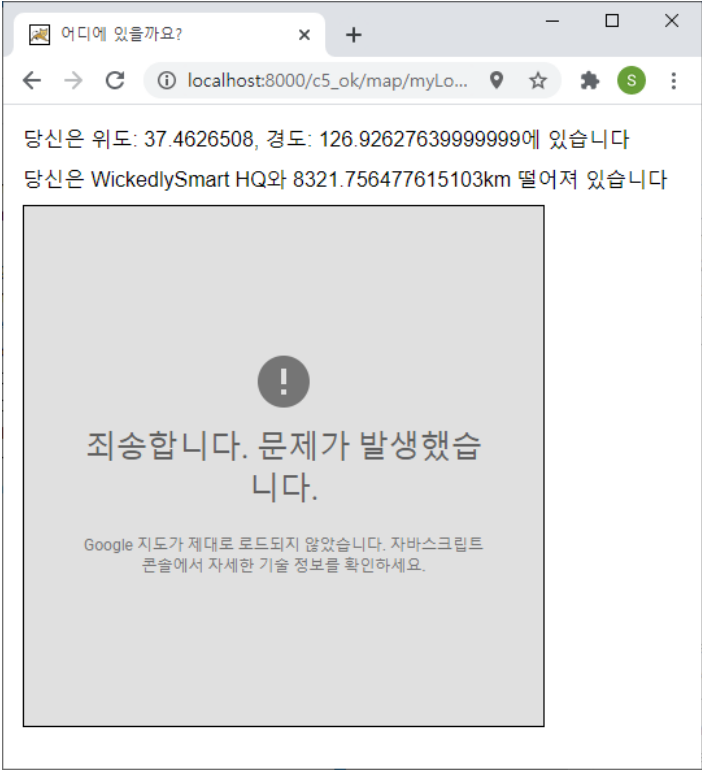
```
function displayLocation(position) {  
    var latitude = position.coords.latitude;  
    var longitude = position.coords.longitude;  
  
    var div = document.getElementById("location");  
    div.innerHTML = "당신은 위도: " + latitude + ", 경도: " + longitude + "에 있습니다";  
  
    var km = computeDistance(position.coords, ourCoords);  
    var distance = document.getElementById("distance");  
    distance.innerHTML = "당신은 WickedlySmart HQ와 " + km + "km 떨어져 있습니다";  
  
    showMap(position.coords);  
}
```

myLoc.js
//추가2

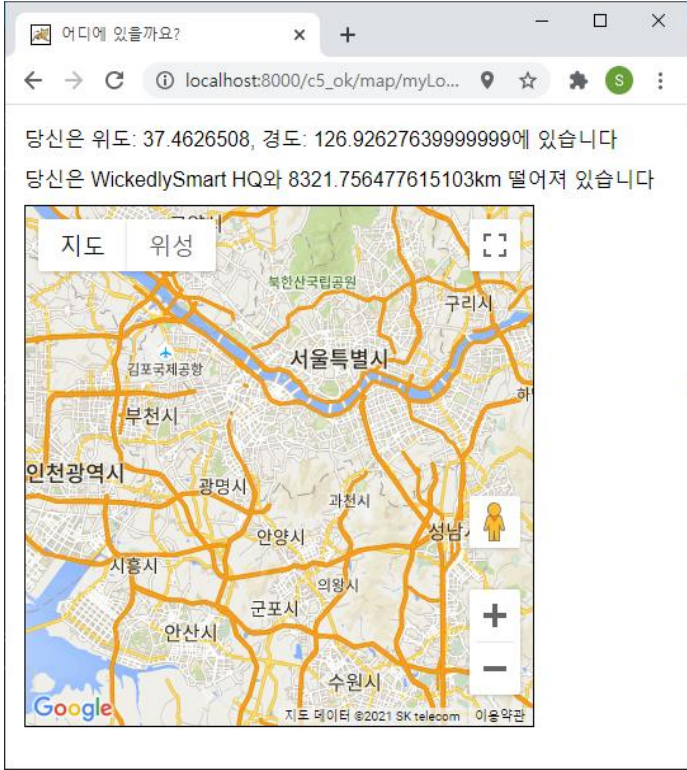
```
3 var map = null; //추가3
```

<실습3-4> 실행화면 - 페이지에 구글 지도 추가

키에 문제가 있는 경우



정상 지도 추가



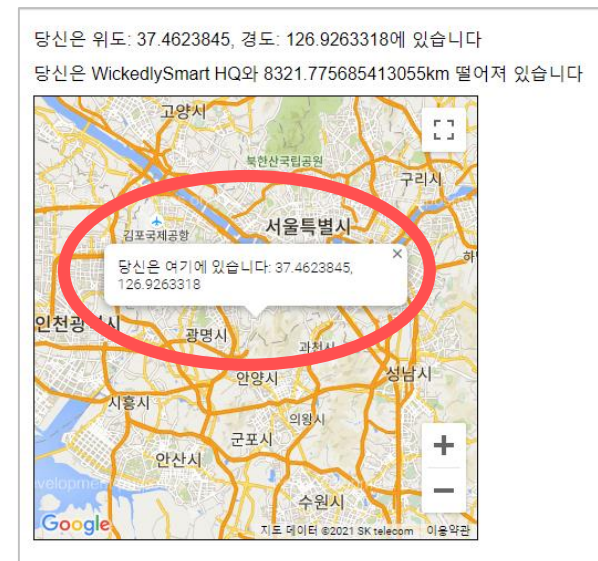
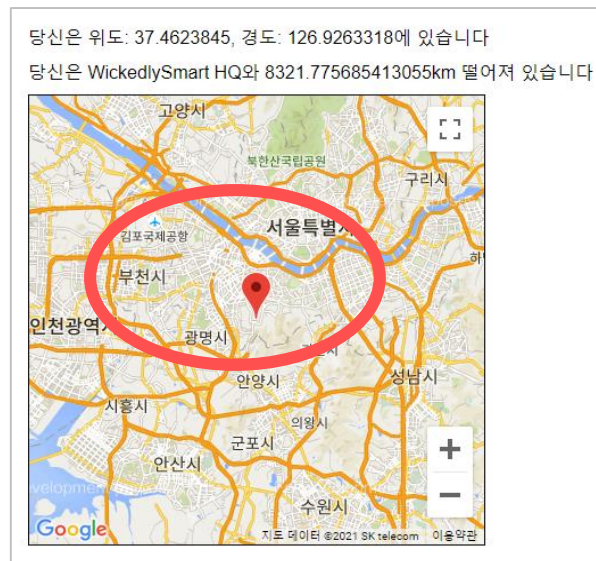
http://localhost:8000/c5_ok/map/myLoc.html

<실습4> 지도 위에 핀을 꽂아 봅시다.

팝업창과 마커를 추가하는 코드가 필요합니다.

- `marker`,
- `information window`
- `handler` for the click event on the marker

구글 지도에서 임의의 지점을 검색하면 빨간색 핀이 검색 지점을 가리키는 있는 것을 볼 수 있습니다.



<실습4-1> addMarker라는 새 함수를 만드는 것부터..

1. addMarker라는 새 함수를 만드는 것부터 시작해보죠, 구글 API를 사용해서
마커 객체를 생성

```
function addMarker(map, latlong, title, content) {  
  var markerOptions = {  
    position: latlong,  
    map: map,  
    title: title,  
    clickable: true  
  };  
  var marker = new google.maps.Marker(markerOptions);
```

1

```
    var infoWindowOptions = {  
      content: content,  
      position: latlong  
    };  
  
    var infoWindow = new google.maps.InfoWindow(infoWindowOptions);  
  
    google.maps.event.addListener(marker, 'click', function() {  
      infoWindow.open(map);  
    });  
  }
```

2

myLoc.js
//추가1

2. Google API 로 새로운 InfoWindow 객체를 생성

<실습4-2> showMap에서 addMarker함수를 호출하는 일

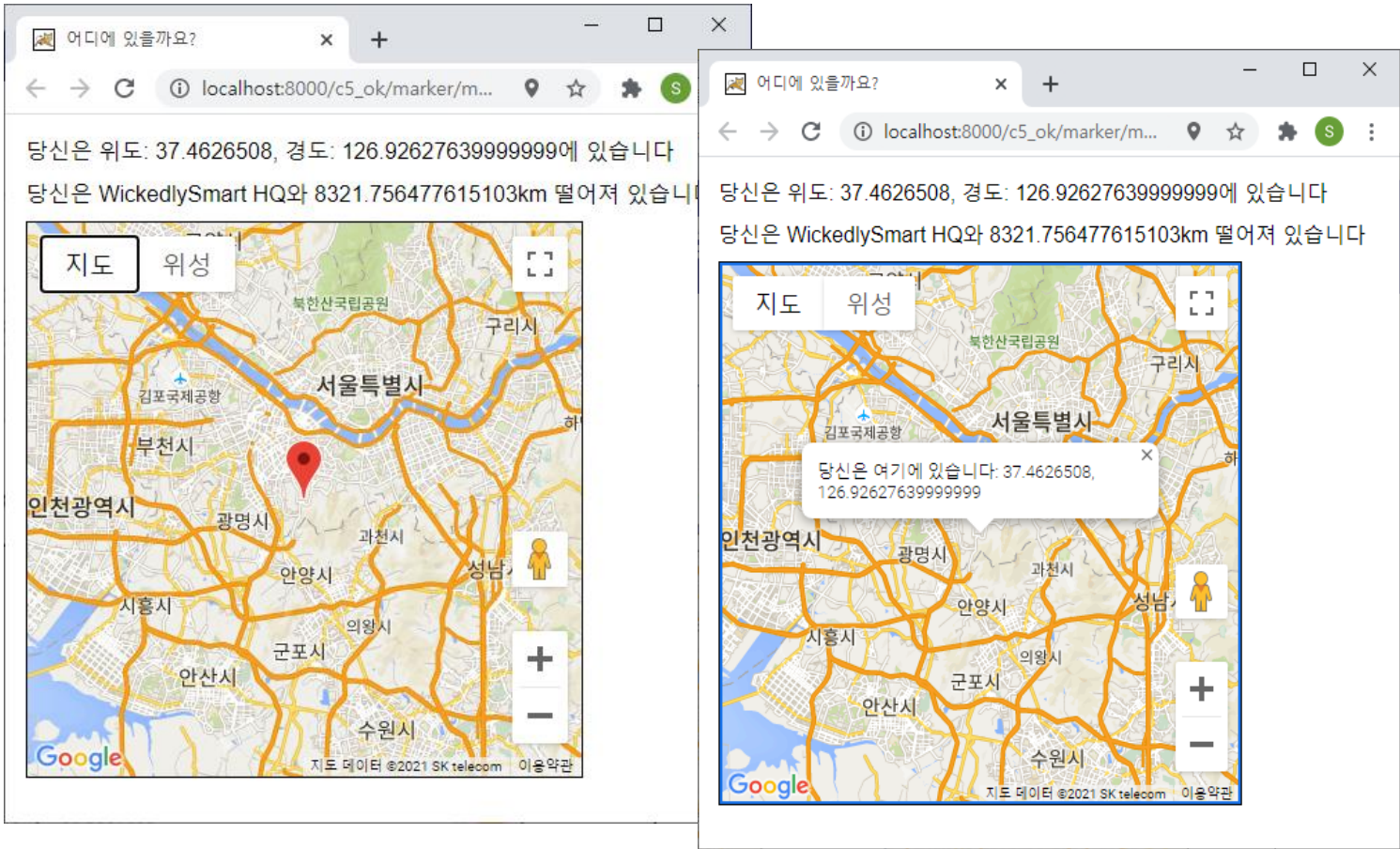
3. showMap에서 addMarker함수를 호출, showMap 함수 맨끝에 추가

```
function showMap(coords) {  
    var googleLatAndLong = new google.maps.LatLng(coords.latitude, coords.longitude);  
    var mapOptions = {  
        zoom: 10,  
        center: googleLatAndLong,  
        mapTypeId: google.maps.MapTypeId.ROADMAP  
    };  
    var mapDiv = document.getElementById("map");  
    map = new google.maps.Map(mapDiv, mapOptions);  
  
    // 사용자 마커를 추가  
    var title = "당신의 위치";  
    var content = "당신은 여기에 있습니다: " +  
        coords.latitude + ", " + coords.longitude;  
    addMarker(map, googleLatAndLong, title, content);  
  
}
```

myLoc.js
//추가2

<실습4-3> 지도위에 핀을 꽂아봅시다 - 실행해보기

지도위에 마커표시 하기



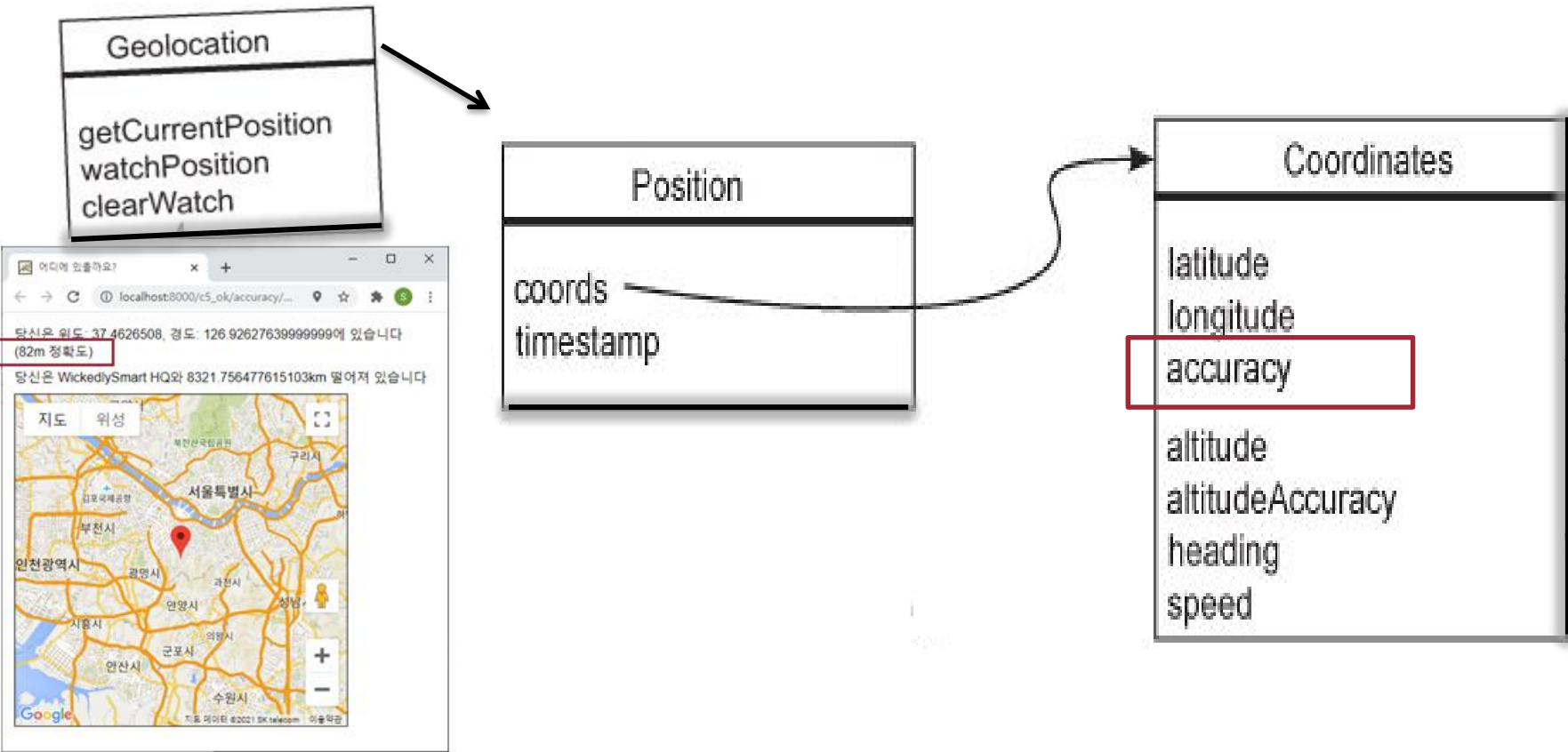
http://localhost:8000/c5_ok/marker/myLoc.html

<실습5> Geolocation API를 살펴보자

getCurrentPosition은 Position and Coordinates object를 가진다.

getCurrentPosition(successHandler, errorHandler, positionOptions)

```
navigator.geolocation.getCurrentPosition(displayLocation, displayError);
```



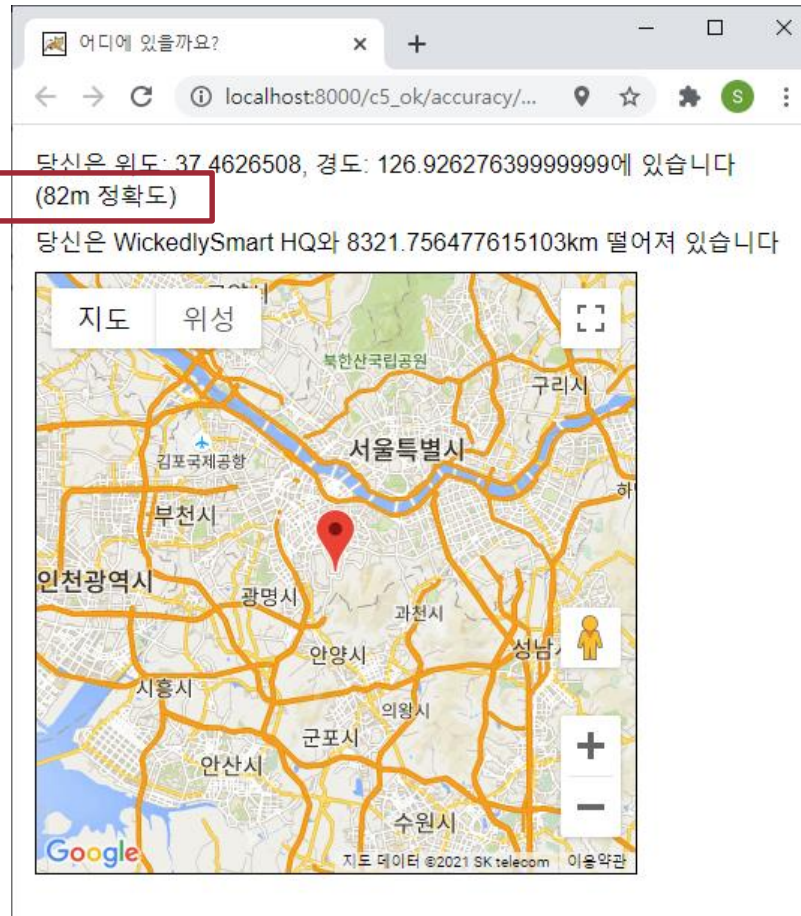
Accuracy, 정확도에 대하여 얘기하여 볼까요?

- 위치를 찾는 일은 과학적인 측정 작업은 아닙니다. 브라우저가 사용하는 방법에 따라 부, 도시 도시내의 구역만 알 수도 있습니다. 성능이 우수한 장치에는 10미터 오차로 여러분의 위치, 속도, 목적지, 고도까지도 알 수 있습니다.
- 위치 정도를 줄 때 신뢰수준 95% 내에서 미터 단위로 정확도까지 제공해 줄 수 있습니다.
- `displayLocation` 함수에 적용하여 봅시다.

```
29      div.innerHTML += " (" + position.coords.accuracy + "m 정확도) ";
```

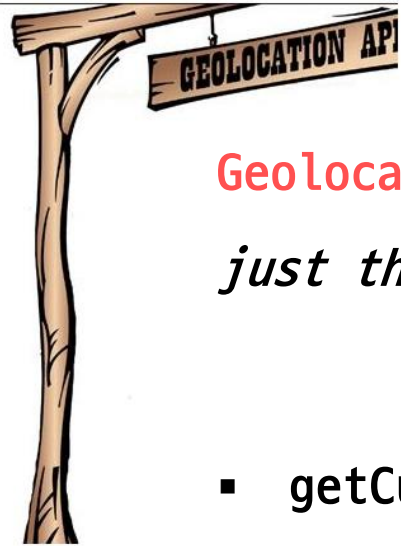
myLoc.js
//추가1

<실습5-1> 정확도(Accuracy) Test 실행해보기



http://localhost:8000/c5_ok/accuracy/myLoc.html

한편 Geolocation API 에서는..



Geolocation API is actually really simple, having *just three* methods:

- `getCurrentPosition`
- `watchPosition`
- `clearWatch`

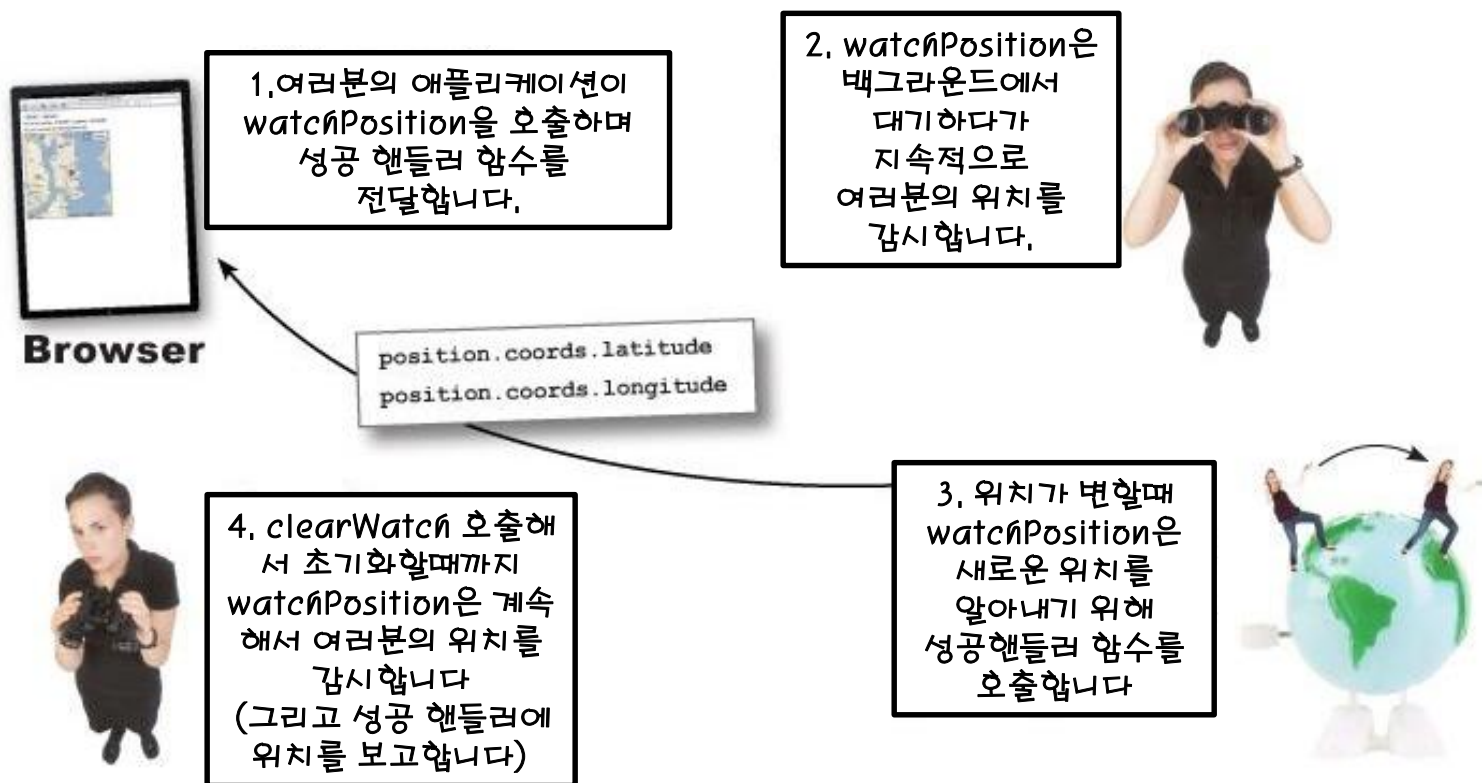
“Wherever you go, there you are”

어디를 가더라도 거기엔 여러분이 있습니다.

여러분의 이동을 추적하는 방법에 대하여 알아보시다

Geolocation API has a **watchPosition** method

- watchPosition 메서드는 실제로 getCurrentPosition 메서드와 흡사하지만 다르게 동작
- 위치가 변할 때 마다 매번 여러분의 성공 핸들러를 반복적으로 호출



애플리케이션 준비 작업

왜 두개의 버튼이 필요할거죠?

Two reasons:

1. 사용자들은 계속해서 자신들의 위치가 추적되는 것보다는 이를 제어하길 원합니다.
2. 위치를 추적하는 일은 모바일 장치에서 전력소모를 극대화 해서 배터리가 빨리 소진되는 결과를 낳습니다.



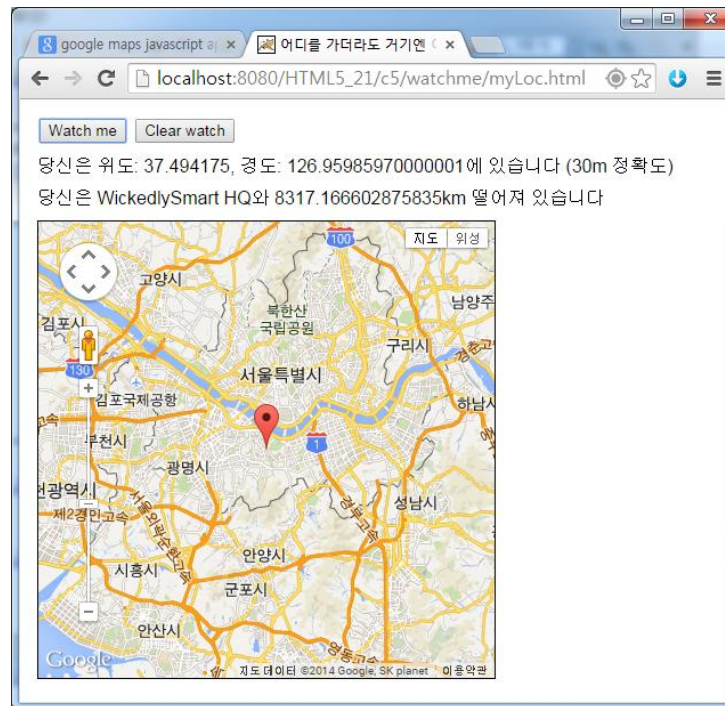
So, first, we'll update the HTML to add a form and two buttons:

- one to start watching your position -watchLocation은 watch 버튼용
- one to stop- ClearWatch은 clear 버튼용

<실습6> watchPosition – 애플리케이션

```
12 <form>
13 <input type="button" id="watch" value="Watch me">
14 <input type="button" id="clearWatch" value="Clear watch">
15 </form>
```

myLoc.html
//추가1



<실습6-1> 이전 코드 재작업

- 이제 두버튼에 대한 클릭 이벤트 핸들러를 추가해야 합니다:
 - ✓ `watchLocation` for the watch button
 - ✓ `clearWatch` for the clear button

```
12 function getLocation() {  
13     if (navigator.geolocation) {  
14         var watchButton = document.getElementById("watch");  
15         watchButton.onclick = watchLocation;  
16         var clearWatchButton = document.getElementById("clearWatch");  
17         clearWatchButton.onclick = clearWatch;  
18     }  
19     else {  
20         alert("이런, 지오로케이션이 제공되지 않네요");  
21     }  
22 }
```

myLoc.js
//추가1

<실습6-2> 이전코드 작업 – watchlocation 핸들러 작성하기

Writing the watchLocation handler

- `geolocation.watchPosition` method는 their position를 watching하는데 사용
- a `success handler` and an `error handler` 두개의 파라미터를 가진다.
- `watchId` (global variable)를 선언하여 준다.

//나중에 추적을 종료할 때 사용

```
3 var watchId = null; myLoc.js
```

//추가2

// 사용자 위치를 보는 코드

```
124 function watchLocation() {  
125     watchId = navigator.geolocation.watchPosition(  
126         displayLocation,  
127         displayError); myLoc.js  
128 } //추가3
```

<실습6-3> 이전코드 작업 – clearWatch 핸들러 작성하기

Writing the clearWatch handler

- 이제 추적작업을 종료하는 handler를 작성
- watchId를 가져다가 geolocation.clearWatch method에 전달

```
130 function clearWatch() {  
131     if (watchId) {  
132         navigator.geolocation.clearWatch(watchId);  
133         watchId = null;  
134     }  
135 }
```

myLoc.js
//추가4

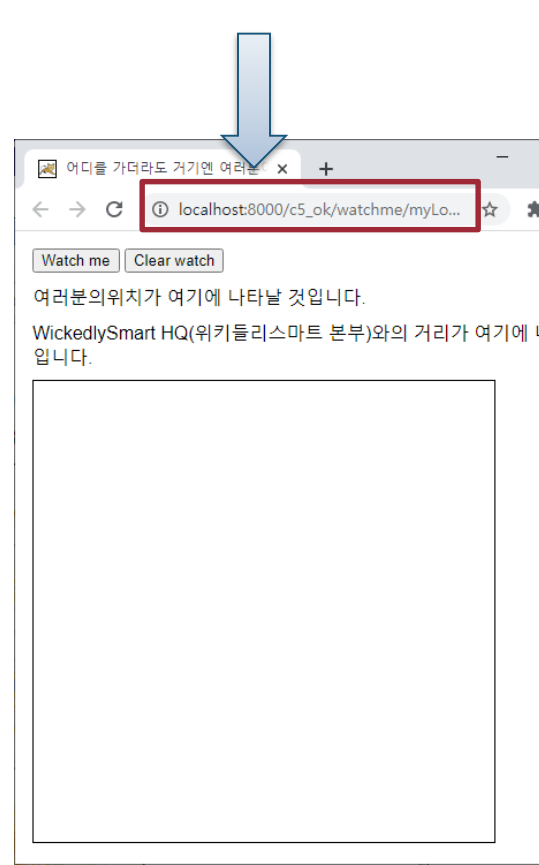
<실습6-4> 이전코드 작업 – displaylocation을 수정

We still need to make a small update to `displayLocation`...

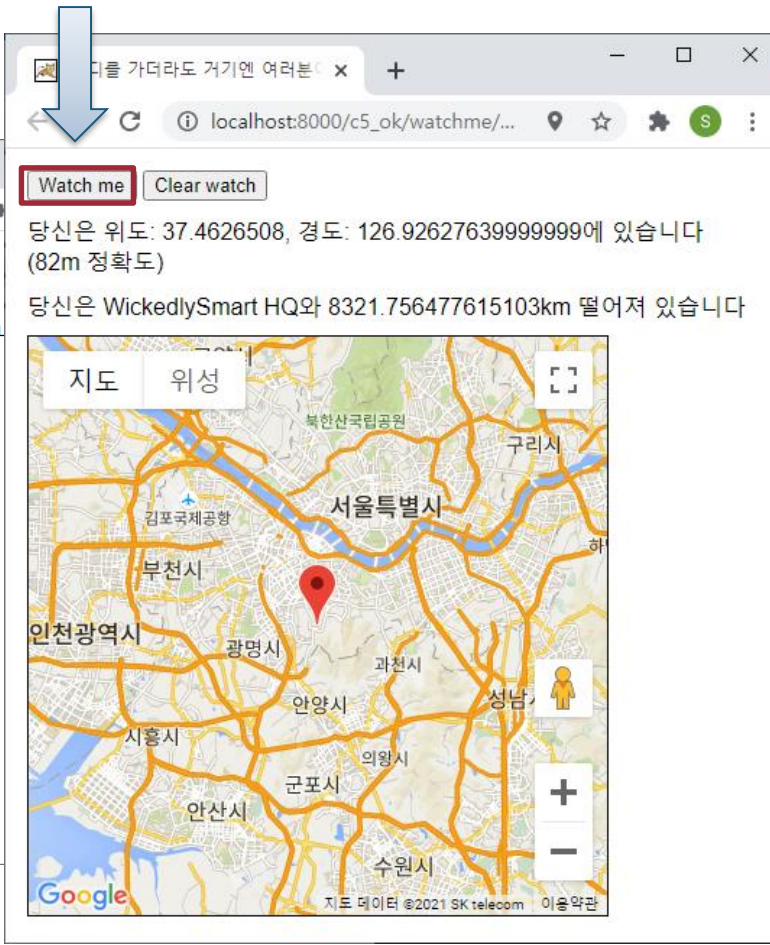
- `showMap`을 한번만 호출하기 위해서는 이미 지도가 만들어졌는지 체크해 보고, 만들어지지 않았을 경우에만 `showMap`을 호출한다.

```
36         if (map == null) {                               myLoc.js
37             showMap(position.coords);                     //추가5
38         }
```

<실습6-5> watchPosition -이동할 시간 실행해보기



여러분의 위치가 여기에 나타날 것입니다.
WickedlySmart HQ(위키들리스마트 본부)와의 거리가 여기에 나타납니다.



Watch me Clear watch

당신은 위도: 37.4626508, 경도: 126.92627639999999에 있습니다 (82m 정확도)
당신은 WickedlySmart HQ와 8321.756477615103km 떨어져 있습니다

지도 위성

서울특별시

김포국제공항

부천시

인천광역시

광명시

과천시

안양시

시흥시

안산시

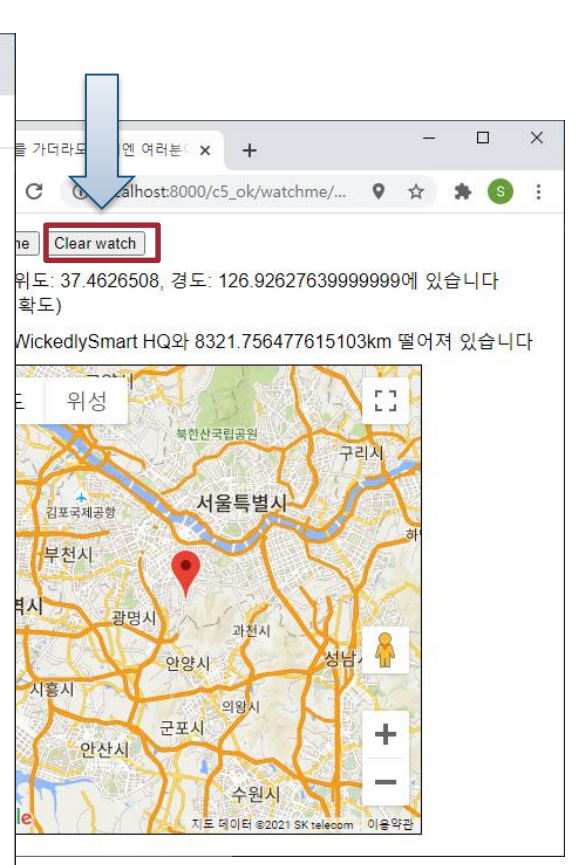
군포시

의왕시

수원시

성남시

지도 데이터 ©2021 SK telecom 이용 약관



Clear watch

위도: 37.4626508, 경도: 126.92627639999999에 있습니다 (확도)
WickedlySmart HQ와 8321.756477615103km 떨어져 있습니다

위성

서울특별시

김포국제공항

부천시

광명시

과천시

안양시

시흥시

안산시

군포시

의왕시

수원시

성남시

지도 데이터 ©2021 SK telecom 이용 약관

http://localhost:8000/c5_ok/watchme/myLoc.html

몇가지 옵션을 더 알아야 합니다

- **세 번째** parameter of `getCurrentPosition`: `positionOptions`
 - Geolocation이 계산하는 방식을 조정할 수 있습니다.

```
var positionOptions = {  
    enableHighAccuracy: false,  
    timeout: Infinity,  
    maximumAge: 0  
}
```

Can we talk about your accuracy, again?

다시 한번 정확도를 얘기해 봅시다.



- Geolocation API has an **accuracy** property
- 정밀도(accuracy)는 브라우저 구현 방식에 따라 달라질 수 있다
- 정밀도와 전력소비 간에 약간의 거래(deal)가 있을 수 있다
- 높은 정밀도를 요구하지 않는 경우라면 빠르고 **적은 전력소비**도 줄일 수 있다.
 - 예: 앱 사용자가 '서울'에 있다
- 그러나 사용자가 위치해 있는 거리(street)를 알 필요가 있다면 그 정보를 얻기 위해 API는 GPS를 켜고 많은 전력소비를 감수해야 한다
- **enableHighAccuracy** 옵션을 이용하면 얻을 수 있는 가장 정확한 위치를 얻을 수 있다
- 그러나 명심해야 할 사항은 이 옵션은 브라우저가 더 정확한 위치 정보를 제공할 수 있음을 **보장해 주지 않는다**는 사실이다

timeouts 와 maximum age의 세계

■ timeout:

- 브라우저가 사용자의 위치를 결정하는 소요되는 시간
- 브라우저가 timeout에 명시된 밀리세컨드 시간 범위내에서 새 위치를 결정하지 못하면 에러 핸들러가 호출
- 기본값은 Infinity

```
var positionOptions = {  
    enableHighAccuracy: false,  
    timeout: Infinity,  
    maximumAge: 0  
}
```

■ maximumAge:

- 브라우저가 예전 위치 정보를 얼마나 오래 보유하고 있는지를 나타냄
- 브라우저가 위치를 결정하는데 60초 걸렸고, maximumAge값이 90초 일때 getCurrentPosition을 호출하면 새 위치가 아닌 기존의 위치, 이미가지고 있는 위치를 반환할 것, - 90초 동안은 유효
- 하지만 maximumAge값이 30초로 설정되어 있다면 브라우저는 새로운 위치를 찾을 겁니다.- 30초안에 못찾았는데 또 새로 찾은 위치 채택
- 내 위치를 결정하는 브라우저의 재계산 주기를 결정
- 위치의 최대수명 값, 적절하게 조절 필요

Options을 설정하는 방법

- 높은 정확도를 원하고 위치에 최대 수명값을 60초(60000밀리세컨드)로 설정한다고 하면 다음과 같이 옵션을 설정하면 된다.

```
var options = {enableHighAccuracy: true, maximumAge: 60000};
```

그리고 나서 이 옵션을 `getCurrentPosition`이나 `watchPosition`에 전달한다

```
navigator.geolocation.getCurrentPosition(  
    displayLocation,  
    displayError,  
    options);
```

혹은 다음과 같이 인라인 형식으로 직접 입력할 수도 있다.

```
navigator.geolocation.getCurrentPosition(  
    displayLocation,  
    displayError,  
    {enableHighAccuracy: true, maximumAge: 60000});
```


진단 시험 주행 테스트

```
function watchLocation() {  
    watchId = navigator.geolocation.watchPosition(  
        displayLocation,  
        displayError,  
        {timeout:5000});  
}
```

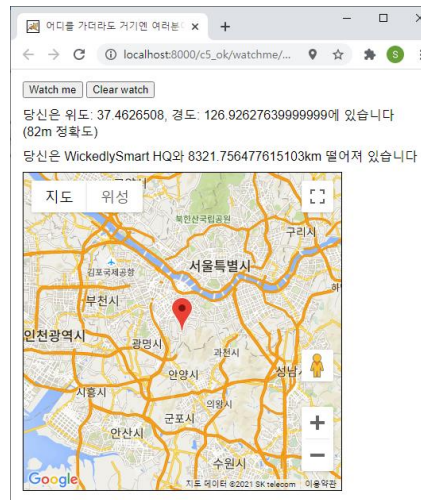
tip

- 전에 테스트를 하면서 기다리고 기다렸건만 아무 반응이 없었던 적이 있었나요?
- 아마도 timeout 값이 무한대로 설정되었기 때문일겁니다. 다시 말해, 브라우저가 특정 에러 조건을 만나지 않았기 때문에 위치를 수집하기 위해 계속 기다린 것이죠. 이제 이런 문제를 해결하는 방법을 알고 있으니, timeout값을 설정함으로써 지오로케이션 API를 좀 더 편리한 도구로 만들어 봅시다. 여기에 어떻게 하는지 나와 있군요.
- Timeout값을 5000밀리세컨드(5초)로 설정하면 브라우저가 계속해서 위치를 얻어 오려하는 것을 막을 수 있습니다. 옵션값을 변경하면서 테스트해 보세요.

<실습7> 새 위치를 가져올 때마다 새 마커를 표시

- 이동함에 따라 현재위치를 지도 정 중앙에 표시하는 함수를 만들고
- 새로운 위치를 가져올 때마다 새 마커를 표시해 봅시다.

```
133 function scrollMapToPosition(coords) {myLoc.js  
134     var latitude = coords.latitude;//추가1  
135     var longitude = coords.longitude;  
136  
137     var latlong = new google.maps.LatLng(latitude, longitude);  
138     map.panTo(latlong);  
139  
140     // 새 마커 추가  
141     addMarker(map, latlong, "Your new location", "You moved to: " +  
142         latitude + ", " + longitude);  
143 }
```

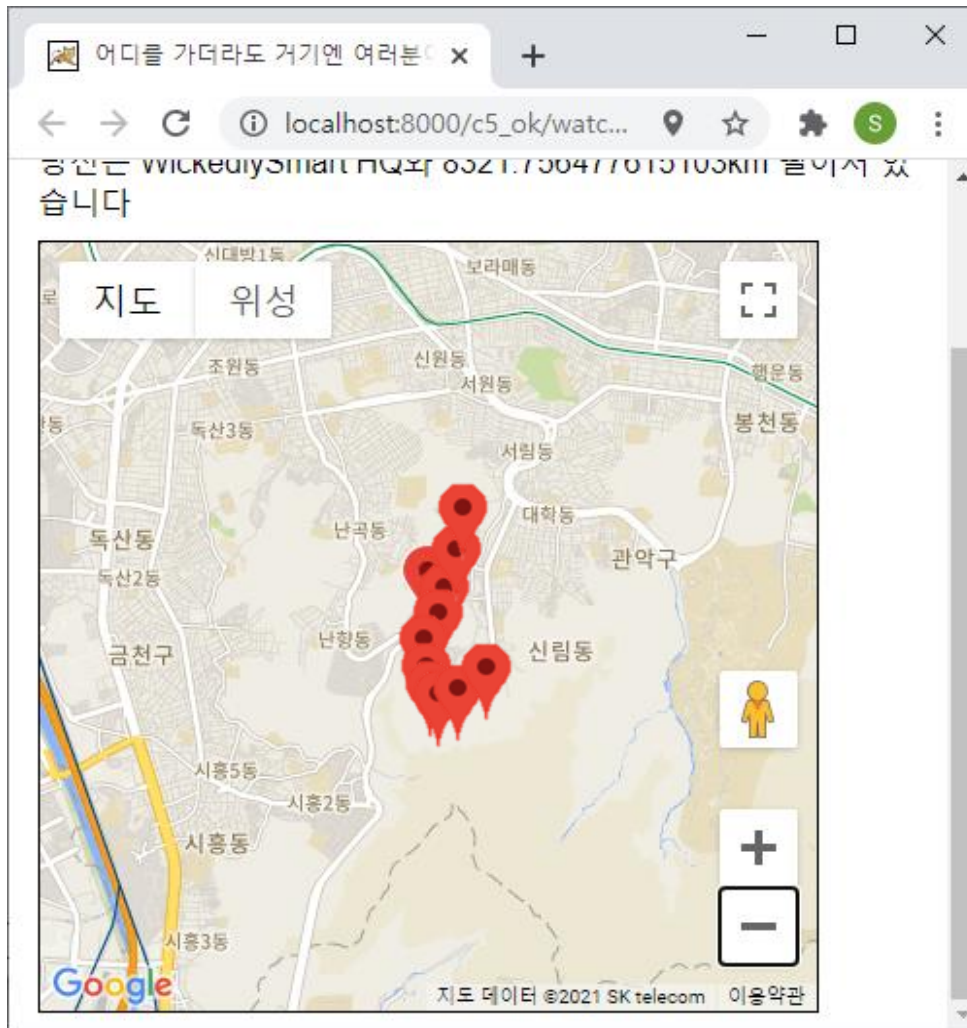


<실습7-1> 새로운 함수 통합하기

- **Update** the **displayLocation** function to call **scrollMapToPosition** each time your position changes

```
36     if (map == null) {                                myLoc.js
37         showMap(position.coords) ;                    //추가2
38     }
39     else {
40         scrollMapToPosition(position.coords) ;
41     }
```

<실습7-2> 새 위치마다 새 마커 표시 실행해보기



이동하여 위치추적이
되는지 확인하고
잘 사용하자!



http://localhost:8000/c5_ok/watchmepan/myLoc.html

Map_jquery> index.html

```
<head>
<title>3300윤선희_Map Tracker</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="shortcut icon" href="./logo.png"> <!-- sizes="192*192" -->
<link rel="apple-touch-icon" href="./logo_apple.png"> <!-- sizes="180x180" -->
<meta name="mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-capable" content="yes">

<link rel="stylesheet" href="https://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css" />
<script src="https://code.jquery.com/jquery-1.11.1.min.js"></script>
<script src="https://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js"></script>

<script src="https://maps.google.com/maps/api/js?key=Maps javascript API key"></script>
</script>
<script src="m_myLoc.js"></script>
<link rel="stylesheet" href="m_myLoc.css">

</head>
```

로고만들기

모바일폰에서는 주소창이 나타나지 않도록 만들기

jquery design

제인키 입력 : 빈칸없어!!!

- Pt에서 복사하여 사용하면 보이지 않는 문자가 포함되어 실행이 안되는 경우 발생함!!!

```
<body>
```

```
<div data-role="header"  
style="background:green; color:yellow; text-align:center;  
text-align:middle; height:25px; font-size:15pt; padding:20px;">  
Map Tracker</div>
```

```
<form>
```

```
<input type="button" id="watch" value="Watch me">
```


```
<input type="button" id="clearWatch" value="Clear watch">
```

```
</form>
```

```
....
```

작업순서

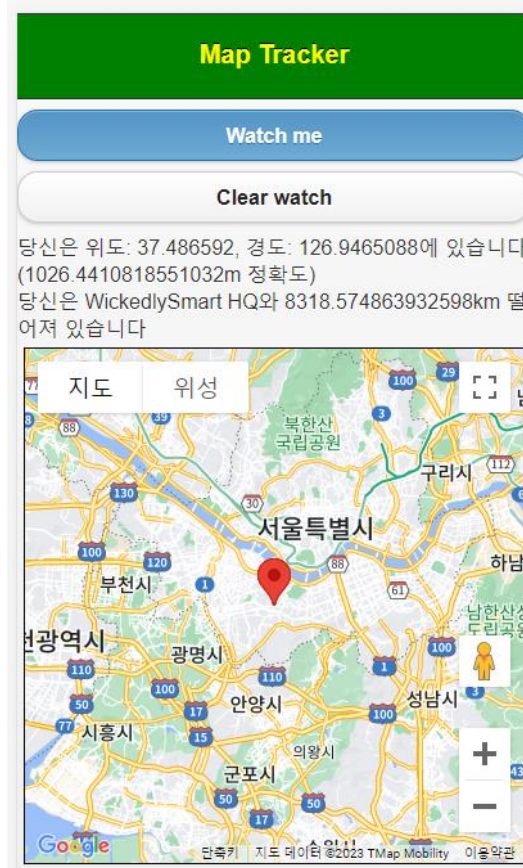
- 1. <link rel="shortcut icon" href="/logo.png"> 입력 후 https로 모두 수정 (GetCurrent method는 구글 정책에서 https만 되도록 바뀜)
- 2. 로그 그림(logo.png)을 완성 후 html, css, js와 동일 디렉토리 위치
- 3. 로컬실행에서 그림(logo.png) 변경 된 것을 확인
- 4. heroku에 maptracker app 이름(sunnylocation)을 만든 후 프로그램(logo.png포함)을 올린다
- 메인 프로그램을 index.html로 바꾼다
- 5. https://sunnylocation.herokuapp.com
- 6. 홈화면 추가
 - 1)안드로이드Mobile phone – url 입력 후 메뉴[⋮] – 홈화면추가 – 이름 붙이기(Map Tracker)
 - 2)아이폰Mobile phone – 사파리브라우저(나침판)-url 입력 후 -가운데 밑에 버튼(📎) 누른 후- 홈화면추가

 올리기전 꼭 확인해야 할것들..

- title:3300학번Map tracker
- 위도경도:서울역
- https 확인
- Width: auto;
- 쿠키 삭제



실행화면 – netlify 의 결과



<https://mapjquery3300.netlify.app/>

핸드폰 쿠키 삭제

Chrome 앱의 경우

- Android 스마트폰 또는 태블릿에서 Chrome 앱을 엽니다.
- 오른쪽 상단에서 더보기 를 탭합니다.
- 방문 기록 인터넷 사용 기록 삭제를 탭합니다.
- 상단에서 기간을 선택합니다. ...
- '쿠키 및 사이트 데이터'와 '캐시된 이미지 또는 파일' 옆의 체크박스를 선택합니다.

iPhone, iPad 또는 iPod touch에서...

- 방문 기록과 쿠키를 삭제하려면 설정 > Safari로 이동하여 '방문 기록 및 웹 사이트 데이터 지우기'를 탭합니다.

8m_map_bootstrap> index.html

```
<head>
  <title>윤선희_어디를 가더라도 거기엔 여러분이 있습니다</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, height=device-height, initial-scale=1">
  <link rel="apple-touch-icon" sizes="180x180" href="imgs/location.png" />
  <link rel="icon" type="image/png" href="imgs/location.png" sizes="192x192"/>
  <meta name="mobile-web-app-capable" content="yes">
  <meta name="apple-mobile-web-app-capable" content="yes">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

  <script src="https://maps.googleapis.com/maps/api/js?key=Maps javascript API key"></script>
  <script src="myLoc.js"></script>
  <link rel="stylesheet" href="m_myLoc.css">
</head>
```

로고만들기

모바일폰에서는 주소창이 나타나지 않도록 만들기

개인키 입력: 빈칸없이!!!

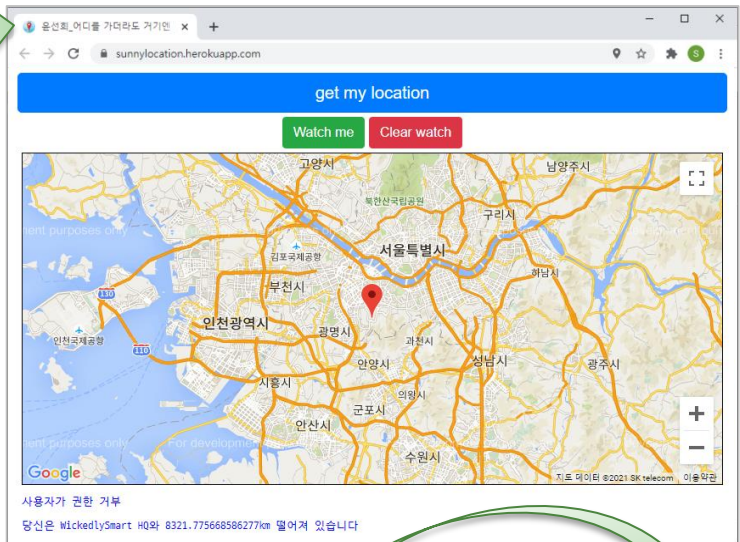
bootstrap design

- Pt에서 복사하여 사용하면 보이지 않는 문자가 포함되어 실행이 안되는 경우 발생함!!!

<https://mapboot3300.netlify.app/>

Map tracker

- heroku 사이트만들기- 학생 개인별로 다르게 만들기
- 로고만들기
- bootstrap 적용하기
- 모바일 화면 캡쳐하기



로고



모바일 실행화면



홈화면 추가

정리1 – Geolocation API 개요

▶ Geolocation API

- 페이지를 실행하는 브라우저가 탑재된 디바이스의 위치 정보를 확인
- 모바일웹 애플리케이션에 유용하게 사용
- GPS가 없는 디바이스일 경우 네트워크 정보로 대략적인 위치 추측

▶ Geolocation

- `window.navigator.geolocation` 속성으로 정의
- `getCurrentPosition(successCallback, errorCallback, options)`
 - 현재 위치 정보를 비동기적으로 한번 확인
 - `successCallback` : 위치 정보 확인 성공시 호출
(인자로 위치정보를 나타내는 `Position` 객체가 지정됨)
 - `errorCallback` : 에러 발생 시 호출
(인자로 에러정보를 나타내는 `PositionError` 객체가 지정됨)
 - `options` : 옵션(`PositionOptions` 타입)
- `watchPosition(successCallback, errorCallback, options)`
 - 현재 위치 정보를 계속 확인
 - `watchId`(정수형 값)가 반환
- `clearWatch(watchId)`
 - 위치 확인을 중지

정리2 – 위치정보 확인 성공 시 처리

▶ 콜백 함수의 인자

- 위치 정보 확인 성공 시 `getCurrentPosition()` 메소드에 첫번째 인자로 지정한 `displayLocation` 콜백 함수가 호출
- 콜백 함수의 인자로 `Position` 객체가 지정

▶ `Position` -> `function displayLocation(position)`

- 위치 정보와 시간 정보를 가지고 있는 객체
- `Position`의 속성
 - `coords` : 위/경도 등의 위치 정보가 저장된 `Coordinates` 객체
 - `timestamp` : 위치 정보를 얻은 시각(1970/01/01 부터의 밀리세컨드)

▶ `Coordinates`

- 속성
 - `latitude` : 위도
 - `longitude` : 경도
 - `altitude` : 고도
 - `accuracy` : 위/경도의 오차(미터 단위)
 - `altitudeAccuracy` : 고도의 오차(미터 단위)
 - `heading` : 디바이스의 진행 방향(북쪽을 기준으로 시계방향의 각도값)
 - `speed` : 디바이스의 진행 속도(미터/초)

정리3 – 위치정보 확인 실패 시 처리

▶ 콜백 함수의 인자

- 위치 정보 확인 실패 시 `getCurrentPosition()` 메소드에 두번째 인자로 지정한 콜백 함수가 호출
- 콜백 함수의 인자로 `PositionError` 객체가 지정

▶ `PositionError`

- 속성
 - `code(0)` : 에러 코드. `PositionError`에 다음의 상수로 정의
 - `PERMISSION_DENIED(1)` : 사용자가 동의하지 않음
 - `PERMISSION_UNAVAILABLE(2)` : 네트워크 문제나 GPS 문제로 위치정보 확인 불가
 - `TIMEOUT(3)` : 지정 시간 초과
 - `message` : 에러 메시지

```
var errorTypes = {  
  0: "알려지지 않은 에러",  
  1: "사용자가 권한 거부",  
  2: "위치를 찾을 수 없음",  
  3: "요청 응답 시간 초과"
```

정리4 – getCurrentPosition()의 옵션

▶ 옵션 객체

- `getCurrentPosition()` 메소드에 세번째 인자로 지정한 옵션
 - `enableHighAccuracy` : 정확도 높은 위치 정보 요청(GPS>기지국>IP)
 - `timeout` : 시간제한(밀리 세컨드). 시간 초과시 에러발생
 - `maximumAge` : 위치 정보의 유효 기간 설정(밀리 세컨드). 0일 경우 항상 새로운 위치 정보 확인

Q & A

