






## 지오로케이션 사용하기

### 현재 위치 정보 알아보기

Element					
Geolocation	Yes	Yes	Yes	9.0+	Yes

geolocation API 는 사용자가 원할 경우, 웹 어플리케이션에서 사용자의 위치 정보를 제공합니다. 사생활 침해의 가능성 때문에, 위치정보를 제공하기 위해서는 사용자의 동의를 받아야 합니다.

#### geolocation 객체

geolocation API 는 `navigator.geolocation` 객체를 통해 사용할 수 있습니다.

해당 객체가 존재하는 경우 지오로케이션 서비스가 가능합니다.

지오로케이션 사용 가능 여부는 다음과 같이 테스트 할 수 있습니다.

```
if ("geolocation" in navigator) {  
    /* 지오로케이션 사용 가능 */  
} else {  
    /* 지오로케이션 사용 불가능 */  
}
```

**참고 :** 파이어폭스 24 나 그 이전 버전에서 navigator 의 "geolocation" 객체는 API 를 사용할 수 없더라도 언제나 true 값을 리턴합니다. 이 부분은 파이어폭스 25 에서 표준에 맞게 수정되었습니다. ([bug 884921](#)).

#### 현재 위치 가져오기

사용자의 현재 위치를 얻기 위해서는 `getCurrentPosition()` 메서드를 호출하면 됩니다.

사용자 위치를 탐지하는 비동기 요청을 초기화하고, 하드웨어에 사용자의 최신 위치 정보를 요청합니다.

위치가 확인되면 정의된 콜백 함수가 실행됩니다.

선택적으로 에러가 발생할 때 실행될 두번째 콜백함수를 지정할 수 있습니다.

세번째 선택적인 파라미터는 옵션 객체인데, 위치값이 반환된 최대 시간과 요청을 대기할 시간, 그리고 높은 정확도를 사용할 지 여부를 지정합니다.

**참고 :** 기본적으로, `getCurrentPosition()` 은 최대한 빨리 응답이 오겠지만 정확도는 낮습니다. 정확도에 개의치 않고 빠른 응답을 필요로 하는 경우 유용합니다.

예를 들어, GPS 가 있는 장치는 고정된 GPS 를 얻기 위해서 일분 이상의 시간이 필요해서 `getCurrentPosition()` 는 덜 정확한 정보(IP 위치 또는 wifi)를 반환합니다.

```
navigator.geolocation.getCurrentPosition(function(position) {  
    do_something(position.coords.latitude, position.coords.longitude);  
});
```

위의 예제는 사용자 위치가 확인되면 `do_something()` 함수를 실행합니다.

## 현재 위치 확인

```
1 <script type="text/javascript">  
2   window.onload=function(){  
3     //위치정보를 확인할 수 있는 브라우저인지 확인  
4     if(navigator.geolocation == undefined){  
5       alert("위치 정보 기능을 지원하지 않습니다!")  
6       return;  
7     }  
8   }  
9  
10  //현재 위치 정보 알아보는 메소드  
11  function showData(){  
12    navigator.geolocation.getCurrentPosition(success, fail); //현재 위치 정보를 조사하고 성공  
13    실패 했을시 호출되는 함수 설정  
14  }  
15  
16  function success(position) { //성공시  
17    log("위치정보 확인 성공!");  
18    for(var property in position.coords) { //반복문 돌면서 출력  
19      log("Key 값:"+property+" 정보:"+position.coords[property]);  
20    }  
21    var lat=position.coords["latitude"];  
22    var lon=position.coords["longitude"];  
23    var  
24    url="http://maps.googleapis.com/maps/api/geocode/json?latlng="+lat+","+lon+"&sensor=false"  
25;  
26    //location.href = url;//페이지 이동하기  
27  }  
28  
29  //실패시
```

```

30 function fail(err){
31   switch (err.code){
32     case err.PERMISSION_DENIED:
33       msg = "사용자 거부";
34       break;
35
36     case err.PERMISSION_UNAVAILABLE:
37       msg = "지리정보를 얻을 수 없음";
38       break;
39
40     case err.TIMEOUT:
41       msg = "시간초과";
42       break;
43
44     case err.UNKNOWN_ERROR:
45       msg = "알 수 없는 오류 발생";
46       break;
47   }
48   log(msg);
49 }
50
51 function log(msg){
52   var console = document.getElementById("console");
53   console.innerHTML += msg+"<br/>";
54 }
55</script>
56
   <button onclick="showData()">현재 위치 확인</button>

```

```

<div id="console" style="width:500px; border:5px; font-size:20px"></div>

```

## 현재 위치 갱신하기

만약 위치 정보가 변경(장치가 움직였거나 좀 더 정확한 위치 정보가 도착했을 때)되었다면 갱신된 위치 정보로 호출되는 콜백 함수를 지정할 수 있습니다.

**watchPosition()** 함수를 사용하면 되는데, **getCurrentPosition()** 함수와 파라미터가 같습니다.

콜백 함수는 사용자가 움직여서 사용자의 위치가 변경되거나, 다른 기술을 통해 정확도가 높은 정보가 제공되었을 때 여러 번 호출됩니다.

예러 콜백 함수는 **getCurrentPosition()** 함수와 같이 선택적이며 정상적인 결과를 반환 할 수 없을 때 마다 호출됩니다.

**참고** : `watchPosition()` 를 사용할 때 `getCurrentPosition()` 을 먼저 호출하지 않고 사용할 수 있습니다.

```
var watchID = navigator.geolocation.watchPosition(function(position) {  
    do_something(position.coords.latitude, position.coords.longitude);  
});
```

`watchPosition()` 메서드는 숫자로 된 ID 를 반환하며, watcher 를 식별하는데 사용합니다.  
사용자 위치 갱신을 할 필요가 없을 때 `clearPosition()` 에 매개변수로 넘겨줍니다.

```
navigator.geolocation.clearWatch(watchID);
```

## 미세 조정

`getCurrentPosition()` 과 `watchPosition()` 은 둘다 success 콜백, 선택적으로 사용할 수 있는 에러 콜백과 PositionOptions 객체를 매개변수로 받습니다.

PositionOptions 객체는 아래의 프로퍼티를 갖는 자바스크립트 객체입니다 :

### enableHighAccuracy

최대한 정확도를 높게 받을 것인지를 지시하는 불리언 값입니다.

true 값을 지정하면 장치에서 정확도가 높은 위치 정보를 제공 할 수 있으면 제공해 줍니다.  
이것은 응답 시간이 느려지거나 전력소비(예:모바일 장치의 GPS 칩)가 늘어 나니까 주의해야 합니다.

반대로 false 를 지정하면 장치는 적은 전력을 사용하든 빠른 응답을 하든 자원을 적절히 사용할 자유를 갖게 됩니다.

### timeout

위치 값을 장치로부터 받을 때까지 최대한 대기할 시간을 양의 정수(천분의 일초 단위)로 지정합니다.

기본값은 [Infinity](#) 이며, `getCurrentPosition()` 는 위치값이 가능 할때 까지 반환하지 않습니다.

### maximumAge

캐시된 위치 값을 반환 받아도 되는 최대한의 시간을 천분의 일초 단위로 양의 정수로 나타냅니다.

0 을 지정하면 캐시된 위치를 사용하지 말라는 의미이고, 반드시 실제 현재 위치를 수집하고자 하는 것을 나타냅니다.

[Infinity](#) 를 지정하면 장치는 캐시된 위치 정보 중 언제 수집되었는지 개의치 않고 반환 합니다.

`watchPosition` 함수의 사용은 아래와 같습니다.:

```
function geo_success(position) {
```

```

    do_something(position.coords.latitude, position.coords.longitude);
}

function geo_error() {
    alert("위치 정보를 사용할 수 없습니다.");
}

var geo_options = {
    enableHighAccuracy: true,
    maximumAge          : 30000,
    timeout              : 27000
};

var wpid = navigator.geolocation.watchPosition(geo_success, geo_error, geo_options);

```

watchPosition 을 사용하는 데모: <http://www.thedotproduct.org/experiments/geo/>

## 위치 객체

사용자의 위치는 Position 객체로 표현되며 Coordinates 객체를 참조합니다.

Position 객체는 아래의 프로퍼티를 갖는 자바스크립트 객체입니다 :

### coords

[Coordinates](#) 객체, 현재의 위치를 정의하는 객체.

### timestamp

위치를 수집한 시간을 표현하는 숫자

Coordinates 객체는 아래의 프로퍼티를 갖는 자바스크립트 객체입니다 :

### latitude

위도, 소수점을 포함하는 숫자.

### longitude

경도, 소수점을 포함하는 숫자.

### altitude

고도, 해수면을 기준으로 미터 단위, null 일 수 도 있습니다.

### accuracy

미터로 위도와 경도의 정확도를 나타내는 숫자.

### altitudeAccuracy

미터로 고도의 정확도를 나타내는 숫자, null 일 수 도 있습니다.

### heading

장치가 움직이는 방향을 나타내는 숫자. 이 값은 정북에서 벗어난 각을 나타냅니다.

0 도는 정북향을 나타내며 방향은 시계방향(동쪽은 90 도이고, 서쪽은 270 도라는 의미)입니다.

speed 값이 0 이면 heading 값은 NaN 이 됩니다.

장치에서 heading 정보를 제공 할 수 없을 때 이 값은 null 이 됩니다.

### speed

장치의 속도를 나타내며, 초당 미터 값을 숫자로 나타냅니다. 이 값은 null 일 수도 있습니다.

## 에러 처리

getCurrentPosition() 또는 watchPosition()를 호출할 때 전달한 에러 콜백 함수에는 PositionError 객체가 첫 번째 파라미터로 전달됩니다.

```
function errorCallback(error) {  
    alert('ERROR(' + error.code + '): ' + error.message);  
};
```

PositionError 객체는 아래의 프러퍼티를 갖는 자바스크립트 객체입니다 :

### code

에러 코드를 나타내는 숫자입니다.

- (1) 권한 없음(permission denied),
- (2) 위치확인 불가(position unavailable),
- (3) 시간초과(timeout) 중에 하나의 값이 됩니다.

### message

사람이 읽을 수 있는 에러 메시지를 문자열로 표혀합니다.

## Geolocation 라이브 예제

HTML Content

```
<p><button onclick="geoFindMe()">현 위치</button></p>  
<div id="out"></div>
```

JavaScript Content

```
function geoFindMe() {  
    var output = document.getElementById("out");
```

```

if (!navigator.geolocation){
    output.innerHTML = "<p>사용자의 브라우저는 지오로케이션을 지원하지 않습니다.</p>";
    return;
}

function success(position) {
    var latitude = position.coords.latitude;
    var longitude = position.coords.longitude;

    output.innerHTML = '<p>위도 : ' + latitude + '° <br>경도 : ' + longitude + '°</p>';

    var img = new Image();
    img.src = "http://maps.googleapis.com/maps/api/staticmap?center=" + latitude + "," +
longitude + "&zoom=13&size=300x300&sensor=false";

    output.appendChild(img);
};

function error() {
    output.innerHTML = "사용자의 위치를 찾을 수 없습니다.";
};

output.innerHTML = "<p>Locating...</p>";

navigator.geolocation.getCurrentPosition(success, error);
}

```

## 권한 허락

addons.mozilla.org 에 등록되는 어떤 부가기능(Add-on)도 지오로케이션 데이터를 사용하기 위전에는 명시적인 권한 허락이 필요합니다.

아래의 함수는 웹페이지에 권한을 요청하는 프롬프트를 표시할 것입니다.

가능하다면 사용자의 응답은 pref 파라미터를 통해 프리퍼런스에 저장될 것입니다.

콜백 파라미터에 전달된 함수는 사용자의 응답을 가리키는 불리언 값을 가지고 호출 됩니다.

만약 true 값이 전달 되었다면 부가기능(Add-on)은 지오로케이션 정보에 접근 할 수 있습니다.

```

function prompt(window, pref, message, callback) {
    let branch = Components.classes["@mozilla.org/preferences-service;1"]
        .getService(Components.interfaces.nsIPrefBranch);

```

```

if (branch.getPrefType(pref) === branch.PREF_STRING) {
    switch (branch.getCharPref(pref)) {
        case "always":
            return callback(true);
        case "never":
            return callback(false);
    }
}

```

```

let done = false;

```

```

function remember(value, result) {
    return function() {
        done = true;
        branch.setCharPref(pref, value);
        callback(result);
    }
}

```

```

let self = window.PopupNotifications.show(
    window.gBrowser.selectedBrowser,
    "geolocation",
    message,
    "geo-notification-icon",
    {
        label: "Share Location",
        accessKey: "S",
        callback: function(notification) {
            done = true;
            callback(true);
        }
    }, [
        {
            label: "Always Share",
            accessKey: "A",
            callback: remember("always", true)
        },
        {
            label: "Never Share",

```



```

        accessKey: "N",
        callback: remember("never", false)
    }
}, {
    eventCallback: function(event) {
        if (event === "dismissed") {
            if (!done) callback(false);
            done = true;
            window.PopupNotifications.remove(self);
        }
    },
    persistWhileVisible: true
});
}

```

```

prompt(window,
    "extensions.foo-addon.allowGeolocation",
    "Foo Add-on wants to know your location.",
    function callback(allowed) { alert(allowed); });

```