

Tesi - Geolocation API

Daniele Rigon - 857319

25 giugno 2018

Indice

1	Overview geolocation	2
2	Specifiche	2
2.1	Oggetto della geolocalizzazione	2
2.2	Metodi	2
2.2.1	GetCurrentPosition	2
2.2.2	WatchPosition	4
2.2.3	ClearPosition	5
2.3	Supporto compatibilità web	5
2.4	Problemi sicurezza/privacy	5
2.5	Come la geolocation api trova la posizione esatta	5
2.6	Test Result	5
2.7	Test Repository	5

1 Overview geolocation

Per quanto riguarda le Geolocation Api, su un dispositivo mobile avremo un set di coordinate provenienti dal sensore GPS, mentre su un portatile potremo usare il posizionamento legato all'ip della connessione internet. Le API per la geolocalizzazione permettono agli utenti di fornire la propria posizione alle applicazioni web. Per proteggere la privacy all'utente viene richiesta l'autorizzazione all'uso della posizione.

2 Specifiche

Esistono praticamente solo due metodi a disposizione, `getCurrentPosition` e `watchPosition` (+ `clearWatch`), entrambi utili a ottenere la posizione corrente. La differenza tra i due va ricercata nella loro periodicità, mentre il primo metodo fornisce il dato una sola volta, il secondo si attiva automaticamente ogniqualvolta la posizione cambi, o ogni tot intervallo di tempo. La sintassi per invocare questi metodi è la seguente:

```
1 navigator.geolocation.getCurrentPosition(inCasoDiSuccesso, opzInCasoDiErrore, opzioni);
2 navigator.geolocation.watchPosition(inCasoDiSuccesso, opzInCasoDiErrore, opzioni);
3
```

2.1 Oggetto della geolocalizzazione

Le API di geolocalizzazione sono pubblicate tramite l'oggetto `navigator.geolocation`. Se l'oggetto esiste, il servizio di geolocalizzazione è disponibile. Per testare l'esistenza di tale oggetto:

```
1 if ("geolocation" in navigator) {
2   /* la geolocalizzazione è disponibile */
3 } else {
4   /* la geolocalizzazione NON è disponibile */
5 }
6
```

2.2 Metodi

2.2.1 GetCurrentPosition

```
1 navigator.geolocation.getCurrentPosition(function(position) {
2   do_something(position.coords.latitude, position.coords.longitude);
3 });
4
```

L'esempio qui sopra chiama la funzione `dosomething()` quando la posizione viene calcolata. != La funzione invocata in caso di successo con tutte le info:

```

1  inCasoDiSuccesso = function(position){
2  alert( "Posizione delle: " + position.timestamp.getHours() + ":" +
3  position.timestamp.getMinutes() + "n" +
4  "Accuratezza delle coordinate: " + position.coords.accuracy + " mt; n" +
5  "Latitudine: " + position.coords.latitude + " gradi; n" +
6  "Longitudine: " + position.coords.longitude + "gradi; n" +
7  "Accuratezza dell'altezza: " + position.coords.altitudeAccuracy + " mt; n" +
8  "Altezza: " + position.coords.altitude + " mt; n" +
9  "Direzione: " + position.coords.heading + " gradin " +
10 "(0 = Nord, 90 = Ovest, 180 = Sud, 270 = Est);n" +
11 "Velocita: " + position.coords.speed + " m/s;"
12 );
13 }
14

```

Nel frammento di codice appena illustrato possiamo vedere tutte le informazioni estraibili dalla struttura Position. Chiaramente, a seconda del device sul quale viene effettuata l'interrogazione, non tutte saranno sempre disponibili, in tal caso il loro valore sarà impostato a null.

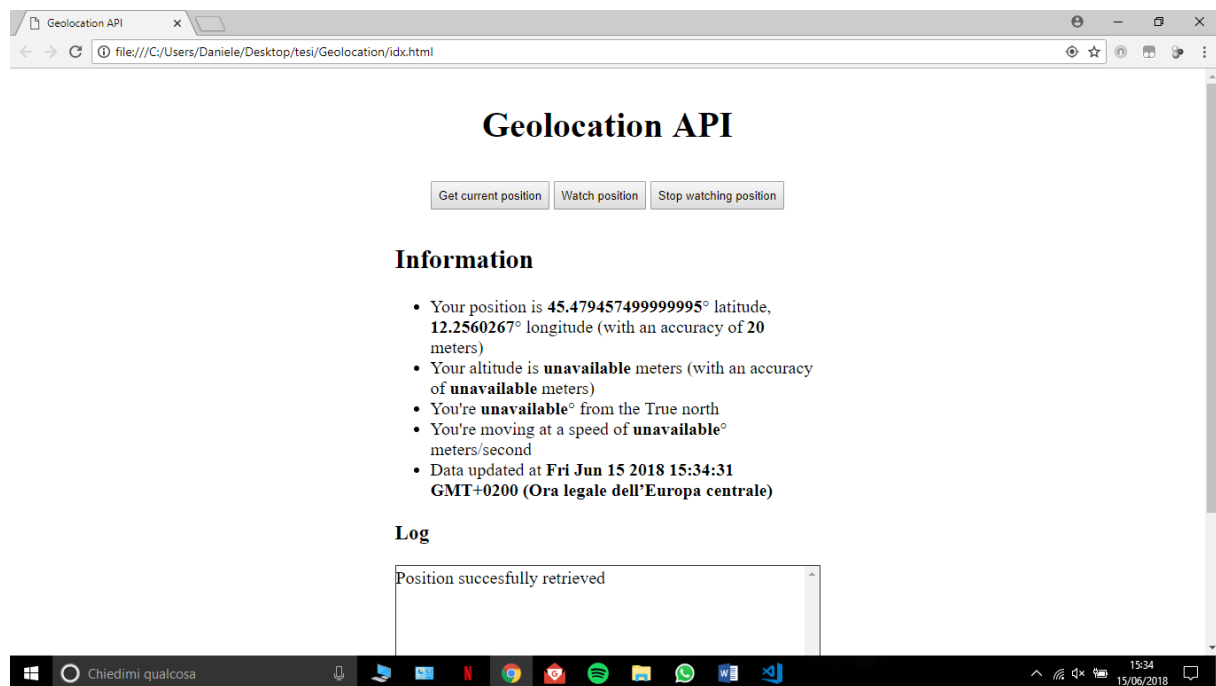


Figura 1: Prova info Geolocation.

In caso invece si verifichi un errore la funzione preposta deve accettare anch'essa un parametro, un oggetto di tipo PositionError contenente un codice di errore ed un messaggio ad uso di debug, ad esempio:

```

1  message.opzInCasoDiErrore = function(error){
2  alert( "Errore " + error.code + ": " + error.message);
3  }
4

```

2.2.2 WatchPosition

Se la posizione cambia (perché il dispositivo si sposta o perché viene calcolata una posizione più accurata), si può settare una funzione che viene chiamata quando la posizione attuale si aggiorna. Basta usare la funzione `watchPosition()`, che ha gli stessi parametri di input di `getCurrentPosition()`. Questa funzione viene chiamata più volte così da permettere al browser di sapere sempre la posizione del dispositivo. La funzione di errore è opzionale come lo era per `getCurrentPosition()`.

```
1  var watchID = navigator.geolocation.watchPosition(function(position) {
2      do_something(position.coords.latitude, position.coords.longitude);
3  });
4
```

Il metodo `watchPosition()` ritorna un ID numerico che può essere usato per identificare univocamente il controllo della posizione; Nota: si può usare questo valore insieme al metodo `clearWatch()` per fermare il controllo della posizione.

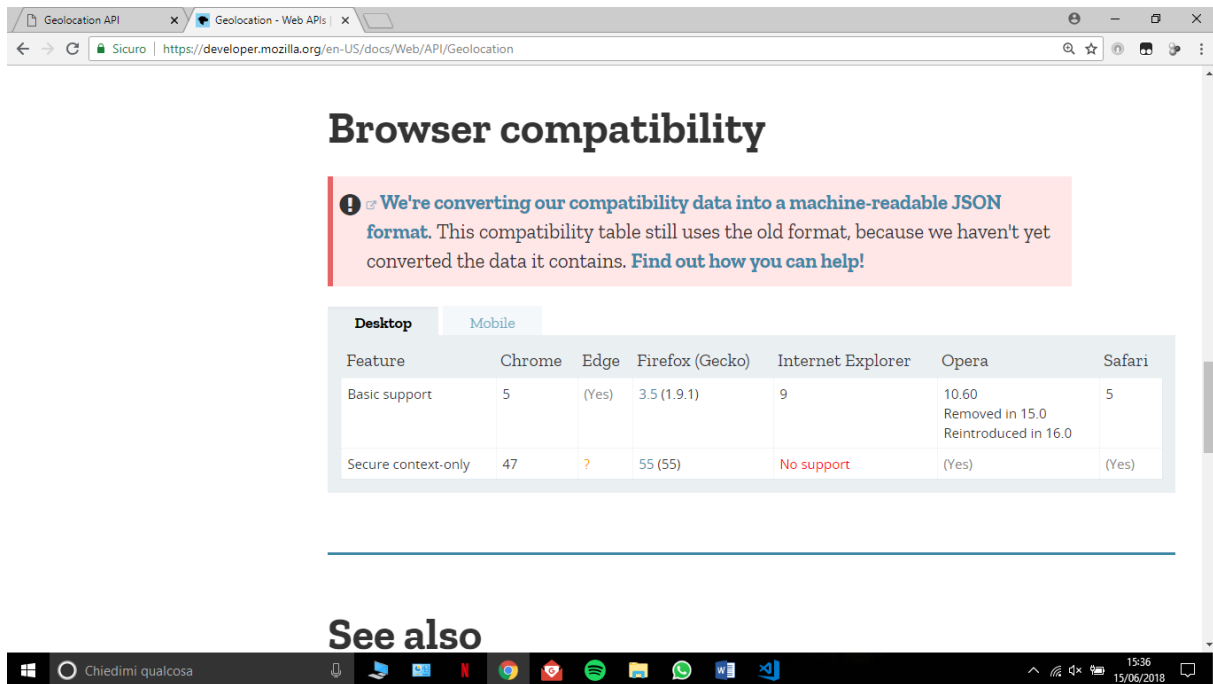
```
1  navigator.geolocation.clearWatch(watchID);
2
```

Infatti, un'invocazione del metodo `watchPosition` può essere successivamente interrotta utilizzando la funzione `clearWatch`, come nell'esempio seguente:

```
1  <!doctype html>
2  <html>
3  <head>
4  <title> Un esempio di watchPosition</title>
5  <script>
6  var id_watch = null;
7
8  inCasoDiSuccesso = function(position){
9      document.getElementById("posizione_corrente").insertAdjacentHTML('beforeend',
10         "<li> Lat: " + position.coords.latitude + ", Lon: " + position.coords.longitude + );
11         "</li>"
12     );
13 }
14
15 sospendiLaRicezione = function(){
16     navigator.geolocation.clearWatch(id_watch);
17 }
18
19 window.onload = function(){
20     id_watch = navigator.geolocation.watchPosition(inCasoDiSuccesso);
21 }
22 </script>
23 </head>
24 <body>
25 <h1>La tua posizione attuale</h1>
26 <menu type="toolbar">
27 <button type="button" onclick="sospendiLaRicezione()">
28 sospendi la ricezione di dati geospaziali
29 </button>
30 </menu>
31 <ul id="posizione_corrente">
32 </ul>
33 </body>
34 </html>
35
```

2.2.3 ClearPosition

2.3 Supporto compatibilità web



The screenshot shows the Mozilla Developer Network page for the Geolocation API. The page title is "Browser compatibility". A notice at the top states: "We're converting our compatibility data into a machine-readable JSON format. This compatibility table still uses the old format, because we haven't yet converted the data it contains. Find out how you can help!". Below the notice is a table with two tabs: "Desktop" (selected) and "Mobile". The table has columns for Feature, Chrome, Edge, Firefox (Gecko), Internet Explorer, Opera, and Safari. The rows show "Basic support" and "Secure context-only".

Feature	Chrome	Edge	Firefox (Gecko)	Internet Explorer	Opera	Safari
Basic support	5	(Yes)	3.5 (1.9.1)	9	10.60 Removed in 15.0 Reintroduced in 16.0	5
Secure context-only	47	?	55 (55)	No support	(Yes)	(Yes)

Figura 2: Browser compatibility

2.4 Problemi sicurezza/privacy

<https://developers.google.com/web/updates/2016/04/geolocation-on-secure-contexts-only>

2.5 Come la geolocation api trova la posizione esatta

<https://stackoverflow.com/questions/4213410/how-does-html5-geolocation-work>

2.6 Test Result

<https://wpt.fyi/results/geolocation-API>

2.7 Test Repository

<https://github.com/web-platform-tests/wpt/tree/master/geolocation-API>