Tesi - Geolocation API

Daniele Rigon - 857319

30 luglio 2018

Indice

1 Overview

La Geolocation API viene utilizzata per ottenere la posizione geografica di un utente. Poiché questo può compromettere la privacy la posizione non è disponibile a meno che l'utente non la approvi: 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.

2 Specifiche

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 e disponibile */
3  } else {
4  /* la geolocalizzazione non e disponibile */
5  }
```

2.2 Metodi

Ci sono solamente tre metodi a disposizione: getCurrentPosition, watchPosition e clearWatch. I primi due sono utili a ottenere la posizione corrente mentre il terzo serve ad annullare la ricerca della posizione corrente. La differenza tra i primi due va ricercata nella loro periodicità, mentre il primo metodo fornisce il dato una sola volta, il secondo si attiva automaticamente ogni qualvolta 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);
```

2.2.1 GetCurrentPosition

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

L'esempio chiama la funzione dosomething() quando la posizione viene calcolata. Un esempio concreto potrebbe essere il seguente:

```
/*  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 sara impostato a null.*/
function success(position) {
  document.getElementById('latitude').innerHTML = position.coords.latitude;
  document.getElementById('longitude').innerHTML = position.coords.longitude;
  document.getElementById('position-accuracy').innerHTML = position.coords.accuracy;

  document.getElementById('altitude').innerHTML = position.coords.altitude ? position.
    coords.altitude :
    'unavailable';
  document.getElementById('altitude-accuracy').innerHTML = position.coords.
    altitudeAccuracy ? position.coords.altitudeAccuracy :
    'unavailable';
  document.getElementById('heading').innerHTML = position.coords.heading ? position.
    coords.heading :
    'unavailable';
  document.getElementById('speed').innerHTML = position.coords.speed ? position.coords.
    speed :
    'unavailable';
}
```

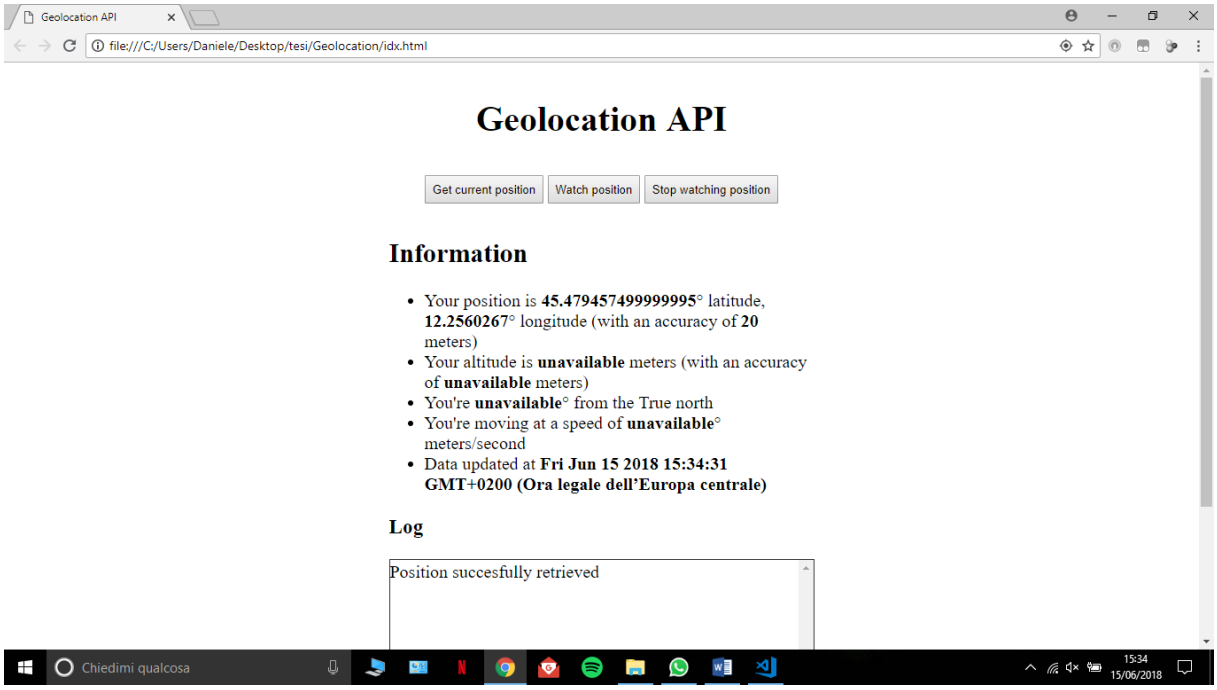

Figura 1: Prova info Geolocation.

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:

```
  message.opzInCasoDiErrore = function(error){
    alert( "Errore " + error.code + ": " + error.message);
  }
```

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().

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

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.

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

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

```
<script>
var id_watch = null;
/*some code*/
watchId = navigator.geolocation.watchPosition(success, error, positionOptions);
// ...
</script>
```

2.2.3 ClearPosition

Viene usato il metodo clearWatch() per annullare il monitoraggio della posizione.

```
navigator.geolocation.clearWatch(watchId);
/*Il numero ID e quello restituito dal metodo watchPosition() descritto precedentemente*/
```

3 Problemi sicurezza/privacy

Uno dei principali problemi con l'API di geolocalizzazione è rappresentato dagli attacchi di cross-site scripting (XSS) dovuti al fatto che gli oggetti per tracciare le coordinate (latitudine e longitudine) risiedono all'interno del DOM, il quale è accessibile con JavaScript. La vulnerabilità XSS potrebbe essere sfruttata per determinare la posizione della vittima e potrebbe verificarsi l'invasione della privacy. A causa del fatto che gli utenti si fidano del sito web, questi si fidano anche della richiesta e condividono la loro posizione. La cosa peggiore è che se l'utente non disabilita il tracciamento il browser continuerà a esporre la posizione della vittima all'attaccante. Supponiamo che un utente malintenzionato abbia rilevato una vulnerabilità XSS in un sito Web; tutto ciò che dovra fare è fare in modo che la vittima esegua il seguente codice JavaScript per rubare la posizione. Il codice utilizza le proprietà DOM cords.latitude e cords.langitude per determinare rispettivamente la latitudine / longitudine e le memorizza in una variabile. Successivamente il codice JavaScript invia i dati al dominio dell'attaccante, in modo diverso a seconda di come è stato configurata la richiesta.

```
<script>
  function showPosition(position){
    var pos="Latitude: " + position.coords.latitude + "<br>Longitude: " + position.coords
    .longitude;
    document.getElementById("mydiv").innerHTML = pos;
  }
  function getLocation(){
    navigator.geolocation.getCurrentPosition(showPosition);
  }
  getLocation();
</script>
```

4 Supporto compatibilità web

manca mobile, fare meglio desktop, vedi payment

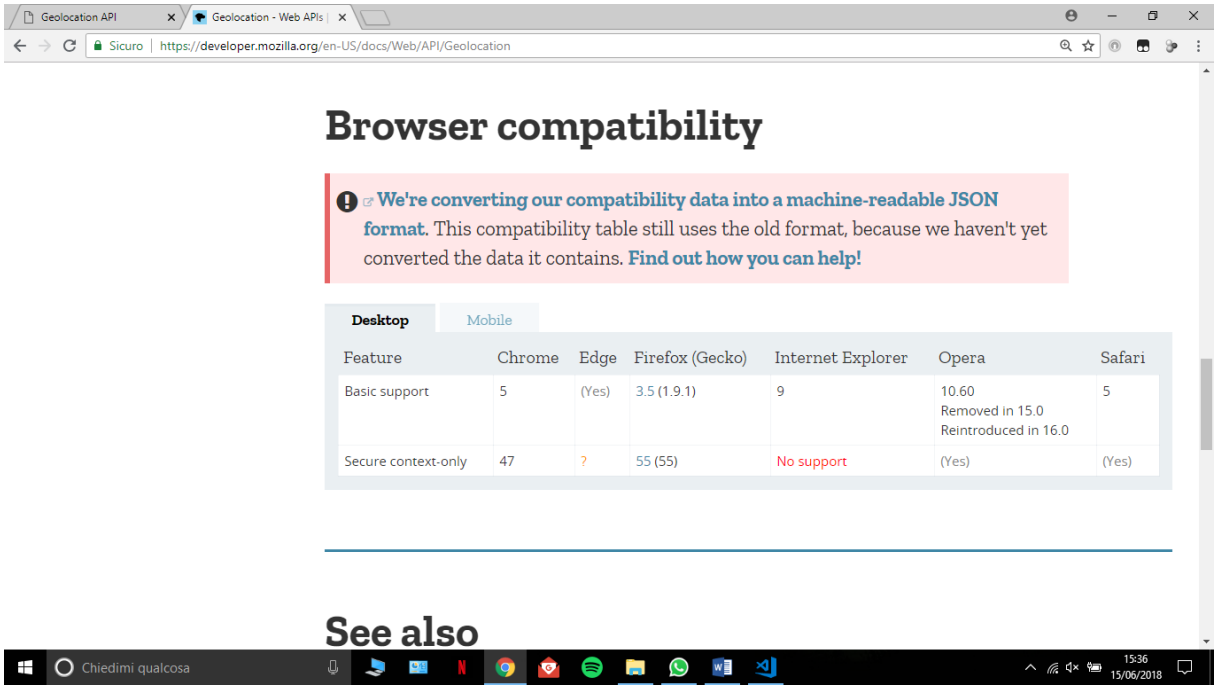

Figura 2: Browser compatibility