JavaScript で位置情報取得

y-yagi　著

2020-02-29 版　　発行

はじめに

本書の内容

本書は、JavaScript で位置情報にアクセスするための API について説明した本です。

ネイティブで作られた位置情報取得アプリを、筆者が Web サービスに移植しようとした際に調査した内容をまとめています。1 章では関連する API についての紹介を、2 章では 1 章で紹介した API の使用方法について記載しています。

JavaScript で位置情報サービスを作ってみたい、と思っている方の参考になれば幸いです。

対象読者

本書は、既に Web サービスの開発や、JavaScript を使った事がある人を対象としています。JavaScript 自体や関連ツールの説明、Web サービス開発に関する基本的な事項についての説明は端折っています。また、記載されている情報は執筆時点 (2020 年 2 月) の情報です。最新の情報と異なる可能性があります。予めご了承ください。

免責事項

本書に記載された内容は、情報の提供のみを目的としています。したがって、本書を用いた開発、製作、運用は、必ずご自身の責任と判断によって行ってください。これらの情報による開発、製作、運用の結果について、著者はいかなる責任も負いません。また、本書に記載されいてる情報は、執筆時点でのものです。時間経過によって情報が古くなっている可能性がありますのであらかじめご了承

ください。

第 1 章

JavaScript で出来ること

まずはじめに、一番大事な JavaScript で出来ることついて整理してみたいと思います。

1.1 Geolocation API

ガラケー時代に位置情報サービスを作られたことがある方はキャリア毎に異なる実装が必要になり辛い思いをしたことかもしれませんが、流石に今はそんなことをする必要はなくなっています。W3C[*1]が位置情報を取得するための「Geolocation API」[*2]という API を定義しており、この API を使うことで、同一のコードで異なる OS・ブラウザで位置情報を取得出来るようになっています。

メソッド

Geolocation API では、下記 3 つのメソッドが定義されています。

Geolocation.getCurrentPosition()

デバイスの現在位置を取得するメソッド。

Geolocation.watchPosition()

デバイスの位置が変化する度に呼び出されるコールバック関数を登録するメソッド。

Geolocation.clearWatch()

[*1] 「World Wide Web Consortium」の略称で、Web 技術の標準化を行う非営利団体の名称。

[*2] https://w3c.github.io/geolocation-api/

watchPosition() で登録した関数を解除するメソッド。

上記 API を使用することで、現在位置の取得や、現在位置が変更になった際に何か処理を行う、ということが出来るようになっています。API の具体的な使い方については次章で説明します。

精度

位置情報の精度は、GPS による取得なのか、Wi-Fi アクセスポイントなのか、はたまた IP アドレスなのかによって異なります。

Geolocation API では、位置情報を取得する際に何の値を使用するかを明確に指定する事は出来ません。代わりに、API に `enableHighAccuracy` というオプションを使用することで、より正確な値を取得するよう指定することが出来るようになっています。デフォルトは無効になっているので、精度の高い情報が必要な場合、このオプションを指定する必要があります[*3]。

サポート状況

Geolocation API は比較的古くからある API であり、例えば Google Chrome ではバージョン 4 から、Safari ではバージョン 5 からサポートしています。おそらく通常利用されるブラウザでは問題ないかと思います。もし特殊なブラウザをサポートする必要がある場合、Can I use[*4]でサポート状況を確認してください。

なお、Geolocation API は HTTPS 環境でしか動作しない、という制限があります。注意してください。

PC での使用

スマートフォンであれば、前節で説明した Geolocation API を使用して期待通りに位置情報を取得出来ると思います。では PC ではどうしょうか。GPS レシーバをつけた PC で、位置情報を正確に取得出来るか、というと、残念ながら執筆時点では難しいようです。

たとえば Windows の場合、GPS レシーバでから位置情報を取得する倍、OS

[*3] 精度が高い情報を取得出来るようになるかわりに、測位結果の取得に時間がかかったり、消費電力が高まる可能性があります。

[*4] https://caniuse.com/#feat=geolocation

が提供している Windows.Devices.Geolocation[*5]という API を使う必要があるのですが、Geolocation API でこの API を使用しているかどうかはブラウザ依存です。

たとえば Chrome の場合、まだこの API を使用しての実装はされていません。そのため、GPS 機器がある場合でも、GPS 精度の位置情報は取得出来ません[*6]ので注意してください。

問題点

Geolocation API により位置情報の取得はできそうです。しかしこの API はフォアグラウンドでの使用しか想定されていません。そのため、バックグラウンド[*7]で動作して位置情報を取得し続けることや、特定のエリアに入った時に何かアクションをする[*8]、ということが出来ません。

他にも、例えば API のインターフェイスに関する問題があります。Geolocation API は比較的昔に仕様が策定されたという事もあり、類似の他の API と Geolocation API とインターフェイスが異なってしまってしまっています。

当然、これらの問題はそのまま、というわけではなく、問題を解決するための対応が行われています。次節ではその対応について見ていきます。

1.2 Geolocation Sensor

前節で説明した Geolocation API の問題を解決し、更にセキュリティやプライバシーについても配慮した新しい API である、「Geolocation Sensor API」[*9]の策定が行われています。

"行われています"と記載したとおり、現在進行中で対応が進んでいる状態です。残念ながらまだ草案であり、ブラウザでも当然実装はされていません。そのため、Geolocation API であげた問題点については、現在対応するすべはありません。

[*5] https://docs.microsoft.com/en-us/uwp/api/Windows.Devices.Geolocation

[*6] Chrome のバグトラッキングシステムに要望は登録されているので、いずれ実装される可能性はあります。https://bugs.chromium.org/p/chromium/issues/detail?id=968883

[*7] 要は Service Workers で Geolocation API を使用することが出来ません。

[*8] https://w3c.github.io/geofencing-api/

[*9] https://w3c.github.io/geolocation-sensor

Geolocation Sensor の今後

Geolocation Sensor は「Devices and Sensors Working Group」[*10]というワーキンググループで管理されています。名前の通りデバイスとセンサーに関する API を策定するワーキンググループで、加速度を取得する API や、バッテリーの状態を取得する API の作成を行っています。

その「Devices and Sensors Working Group」のロードマップ[*11]によると、2020 の Q2 に Candidate Recommendation(勧告候補) になる予定のようです。勧告候補になると諮問委員会への実装依頼が行われる為、もしかしたら 2020 年中には試せるような状態になるかもしれません。

1.3 まとめ

現状をまとめると、

- フォアグラウンドでの位置情報の取得は出来る
- バックグラウンドの取得は (まだ) 出来ない
- ジオフェンス用の API は (まだ) ない

という状態です。ちなみに筆者はバックグラウンドでの処理を行いたかったのでしたが駄目でした。残念。

ちなみに、"JavaScript"で"ジオフェンス"について調査すると、Geofencing API[*12]という API について情報にヒットすることがあります。これは正式に廃止された仕様であり、使用出来ることはありませんので注意してください。

[*10] https://www.w3.org/das/
[*11] https://www.w3.org/das/roadmap
[*12] https://w3c.github.io/geofencing-api/

第 2 章

チュートリアル

ここでは、1 章で説明した API を実際に使用してみたいと思います。1 章で説明したとおり、Geolocation Sensor はまだ使用出来ない為、ここでは Geolocation API を使用したサンプルのみ記載しています。

https://github.com/y-yagi/gps-with-js-example に本章で説明したコードがありますので、合わせてご参照ください。

2.1 Geolocation.getCurrentPosition

まずは、`Geolocation.getCurrentPosition` という現在位置を取得するメソッドについて説明します。早速コードです。

リスト 2.1: getCurrentPosition_v1

```
 1: <!DOCTYPE html>
 2: <html>
 3:   <head>
 4:     <script>
 5:     function getLocation() {
 6:       if (navigator.geolocation) {
 7:         navigator.geolocation.getCurrentPosition(showPosition);
 8:       } else {
 9:         alert("Geolocation is not supported by this browser.");
10:       }
11:     }
12:
13:     function showPosition(position) {
14:       const pos = document.getElementById("pos");
```

```
15:       pos.innerHTML = "Latitude: " + position.coords.latitude +
16:         "<br>Longitude: " + position.coords.longitude;
17:     }
18:     </script>
19:   </head>
20:   <body>
21:     <button onclick="getLocation()">現在位置を表示</button>
22:     <p id="pos"></p>
23:   </body>
24: </html>
```

ボタンを押すと現在位置の緯度経度を表示するだけのコードです。コードを見るだけで何をしているかわかりそうですが、簡単に解説します。

ボタンを押下すると、getLocation 関数が実行されます。getLocation 関数ではまず、はじめに、navigator.geolocation を使用して Geolocation API が使用出来るかどうかを確認しています。

Geolocation API が使用出来る事を確認したら、Geolocation.getCurrentPosition を呼び出します。Geolocation.getCurrentPosition には、位置情報を取得出来た場合に実行するコールバック関数を指定します。Geolocation.getCurrentPosition には他にも引数を指定出来ますが、これは後ほど説明します。

無事位置情報が取得出来たら、Geolocation.getCurrentPosition に指定した showPosition 関数が実行されます。コールバック関数には、GeolocationPosition オブジェクトが渡されます。

GeolocationPosition オブジェクトは、現在位置と位置情報が取得された時間を保持しており、coords プロパティを使用して現在位置を取得して、その値を表示しています。なお、coords はは GeolocationCoordinates オブジェクトを返します。今回は緯度経度だけ使用していますが、他にも高度などの情報も取得することが出来ます。

以上です。簡単ですね。

オプションを指定

続けて、Geolocation.getCurrentPosition にオプションを指定する方法について説明します。とはいっても殆ど前節のコードと同じです。diff だけ記載し

ます。

リスト 2.2: getCurrentPosition_v2

```
 1:      <script>
 2:      function getLocation() {
 3:        if (navigator.geolocation) {
 4: -        navigator.geolocation.getCurrentPosition(showPosition);
 5: +        navigator.geolocation.getCurrentPosition(showPosition,
 6:            error, { enableHighAccuracy: true });
 7:        } else {
 8:          alert("Geolocation is not supported by this browser.");
 9:        }
10:
11:        const button = document.getElementById("try");
12:      }
13: +
14: +    function error(err) {
15: +      console.warn(`Error: (${err.code}): ${err.message}`);
16: +    }
```

Geolocation.getCurrentPosition 呼び出し時の引数を追加しています。

第一引数には位置情報が取得出来た際に実行するコールバック関数を指定しましたが、第二引数には位置情報の取得に失敗した際に実行するコールバック関数を指定します。たとえば、位置情報を取得するにはユーザの許可が必要ですが、ユーザが位置情報の取得を許可しなかった場合このコールバック関数が実行されます。

位置情報の取得に失敗した場合、コールバック関数にはエラー情報を保持した GeolocationPositionError オブジェクトが渡されます。GeolocationPositionError からはエラーコード (code) とエラーメッセージ (message) が取得出来ます。

上記コードでは、位置情報の取得に失敗時に実行する error 関数でエラーコードとエラーメッセージをコンソールに出力するようにしています。エラーメッセージはエラーの詳細についての文字列が取得出来ますが、ロケールについては定義されいていないので注意してください。

最後に第三引数についてです。第三引数には、Geolocation.getCurrentPosition の挙動を指定するためのオプションを指定出来ます。

オプションには PositionOptions オブジェクトを指定します。PositionOptions オブジェクトには以下の三つの値を指定出来ます。

enableHighAccuracy

高精度の結果を取得するかどうかを指定。true を指定した場合、高精度な位置情報を取得するようになる。デフォルトは false。

timeout

位置情報の取得にかかる時間の上限を指定。デフォルトは Infinity で、位置情報が得られるまで待つ。

maximumAge

キャッシュされた位置情報の有効期限を指定。0 を指定するとキャッシュを使用しない。@<tt>Infinity を指定した場合、必ずキャッシュを使用する。デフォルトは 0。

今回のサンプルコードでは enableHighAccuracy を指定して、高精度の結果が取得するようにしています。これにより、例えば GPS 機器が内蔵されているスマートフォンの場合、より正確な結果が取得されるようになっています。

これで Geolocation.getCurrentPosition についての説明は終わりです。ここで実装したサンプルは、https://y-yagi.github.io/gps-with-js-example/getCurrentPosition.html で実際にブラウザからお試し頂けます。

2.2 Geolocation.watchPosition

続いて、Geolocation.watchPosition について です。Geolocation.watchPosition に指定する引数は、Geolocation.getCurrentPosition は同じで、位置情報を取得出来た場合に実行するコールバック関数、位置情報の取得に失敗した際に実行するコールバック関数、挙動を指定するためのオプションです。オプションに指定出来る値も Geolocation.getCurrentPosition と同じです。

Geolocation.getCurrentPosition が一度だけ位置情報を取得するのに対して、Geolocation.watchPosition は機器の位置が変化する度に実行されます。

この監視を解除するには、Geolocation.clearWatch を使用する必要があります。Geolocation.getCurrentPosition は戻り値として ID を返すように

なって、その ID を Geolocation.clearWatch にすると処理が解除されます。

API の説明も終わったところでサンプルです。

リスト 2.3: watchPosition

```
 1: <!DOCTYPE html>
 2: <html>
 3:   <head>
 4:     <script>
 5:     function start() {
 6:       if (navigator.geolocation) {
 7:         const id = navigator.geolocation.watchPosition(showPosition,
 8:           error, { enableHighAccuracy: true });
 9:         document.getElementById("start").disabled = true;
10:         document.getElementById("stop").dataset.watchId = id;
11:         document.getElementById("stop").disabled = false;
12:       } else {
13:         alert("Geolocation is not supported by this browser.");
14:       }
15:     }
16:
17:     function showPosition(position) {
18:       const pos = document.getElementById("pos");
19:       pos.innerHTML = "Latitude: " + position.coords.latitude +
20:         "<br>Longitude: " + position.coords.longitude;
21:     }
22:
23:     function error(err) {
24:       console.warn(`Error: (${err.code}): ${err.message}`);
25:     }
26:
27:     function stop() {
28:       const id = document.getElementById("stop").dataset.watchId;
29:       navigator.geolocation.clearWatch(id)
30:       document.getElementById("stop").disabled = true;
31:       document.getElementById("start").disabled = false;
32:     }
33:     </script>
34:   </head>
35:   <body>
36:     <button onclick="start()" id="start">開始</button>
37:     <button onclick="stop()" id="stop" disabled>停止</button>
38:     <p id="pos"></p>
39:   </body>
40: </html>
```

開始ボタンを押下すると Geolocation.watchPosition を呼び出して位置情報の取得を開始します。Geolocation.watchPosition の戻り値を**停止**ボタンのデータ属性に設定し、**停止**ボタン押下時に Geolocation.clearWatch に渡すようにしています。

これで位置情報が変わる度に表示される緯度経度の値が更新されるようになりました。とはいえ、PC で開発している場合、実際に位置情報を変えるのは難しいです。

そんなときの為に、Google Chrome では、Chrome DevTools のセンサーエミュレーションで位置情報データのオーバーライドが出来るようになっています。

Chrome DevTools のメインメニュー -> [More Tools] -> [Sensors] の [Geolocation] で値を変更出来ます。

Geolocation.watchPosition が実際に動作するかを確認したい場合にご利用ください。なお、設定方法の詳細は、https://developers.google.com/web/tools/chrome-devtools/device-mode/device-input-and-sensors をご参照ください。

JavaScript で位置情報取得

2020 年 2 月 29 日　初版第 1 刷　発行

著　者　y-yagi