HTML5 Geolocation API lets you share your location with your favorite web sites. A Javascript can capture your latitude and longitude and can be sent to backend web server and do fancy location-aware things like finding local businesses or showing your location on a map.

Today most of the browsers and mobile devices support Geolocation API. The geolocation APIs work with a new property of the global navigator object ie. Geolocation object which can be created as follows:

```javascript
var geolocation = navigator.geolocation;
```

The geolocation object is a service object that allows widgets to retrieve information about the geographic location of the device.

**Geolocation Methods**

The geolocation object provides the following methods –

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCurrentPosition</td>
<td>This method retrieves the current geographic location of the user.</td>
</tr>
<tr>
<td>watchPosition</td>
<td>This method retrieves periodic updates about the current geographic location of the device.</td>
</tr>
<tr>
<td>clearWatch</td>
<td>This method cancels an ongoing watchPosition call.</td>
</tr>
</tbody>
</table>

**Example**

Following is a sample code to use any of the above method –

```javascript
function getLocation() {
    var geolocation = navigator.geolocation;
    geolocation.getCurrentPosition(showLocation, errorHandler);
}
```

Here showLocation and errorHandler are callback methods which would be used to get actual position as explained in next section and to handle errors if there is any.

**Location Properties**

Geolocation methods getCurrentPosition and getPositionUsingMethodName specify the callback method that retrieves the location information. These methods are called asynchronously with an object `Position` which stores the complete location information.

The `Position` object specifies the current geographic location of the device. The location is expressed as a set of geographic coordinates together with information about heading and speed.

The following table describes the properties of the Position object. For the optional properties if the system cannot provide a value, the value of the property is set to null.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
Specifies the geographic location of the device. The location is expressed as a set of geographic coordinates together with information about heading and speed.

**coords.latitude** Number

Specifies the latitude estimate in decimal degrees. The value range is [-90.00, +90.00].

**coords.longitude** Number

Specifies the longitude estimate in decimal degrees. The value range is [-180.00, +180.00].

**coords.altitude** Number

[Optional] Specifies the altitude estimate in meters above the WGS 84 ellipsoid.

**coords.accuracy** Number

[Optional] Specifies the accuracy of the latitude and longitude estimates in meters.

**coords.altitudeAccuracy** Number

[Optional] Specifies the accuracy of the altitude estimate in meters.

**coords.heading** Number

[Optional] Specifies the device's current direction of movement in degrees counting clockwise relative to true north.

**coords.speed** Number

[Optional] Specifies the device's current ground speed in meters per second.

**timestamp** date

Specifies the time when the location information was retrieved and the Position object created.

**Example**

Following is a sample code which makes use of Position object. Here showLocation method is a callback method −

```javascript
function showLocation( position ) {
    var latitude = position.coords.latitude;
    var longitude = position.coords.longitude;
    ...
}
```

**Handling Errors**

Geolocation is complicated, and it is very much required to catch any error and handle it gracefully.

The geolocations methods `getCurrentPosition` and `watchPosition` make use of an error handler callback method which gives `PositionError` object. This object has following two properties −
The following table describes the possible error codes returned in the PositionError object.

<table>
<thead>
<tr>
<th>Code</th>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>UNKNOWN_ERROR</td>
<td>The method failed to retrieve the location of the device due to an unknown error.</td>
</tr>
<tr>
<td>1</td>
<td>PERMISSION_DENIED</td>
<td>The method failed to retrieve the location of the device because the application does not have permission to use the Location Service.</td>
</tr>
<tr>
<td>2</td>
<td>POSITION_UNAVAILABLE</td>
<td>The location of the device could not be determined.</td>
</tr>
<tr>
<td>3</td>
<td>TIMEOUT</td>
<td>The method was unable to retrieve the location information within the specified maximum timeout interval.</td>
</tr>
</tbody>
</table>

Example

Following is a sample code which makes use of PositionError object. Here errorHandler method is a callback method –

```javascript
function errorHandler(err) {
    if (err.code == 1) {
        // access is denied
    }
    ...
}
```

Position Options

Following is the actual syntax of getCurrentPosition method –

```javascript
ggetCurrentPosition(callback, ErrorCallback, options)
```

Here third argument is the `PositionOptions` object which specifies a set of options for retrieving the geographic location of the device.

Following are the options which can be specified as third argument –

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableHighAccuracy</td>
<td>Boolean</td>
<td>Specifies whether the widget wants to receive the most accurate location...</td>
</tr>
</tbody>
</table>
accurate location estimate possible. By default this is false.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Number</td>
<td>The timeout property is the number of milliseconds your web application is willing to wait for a position.</td>
</tr>
<tr>
<td>maximumAge</td>
<td>Number</td>
<td>Specifies the expiry time in milliseconds for cached location information.</td>
</tr>
</tbody>
</table>

**Example**

Following is a sample code which shows how to use above mentioned methods –

```javascript
function getLocation() {
    var geolocation = navigator.geolocation;
    geolocation.getCurrentPosition(showLocation, errorHandler, {maximumAge: 75000});
}
```