HTML5 Geolocation

What is Geolocation?

The HTML5 geolocation feature lets you find out the geographic coordinates (latitude and longitude numbers) of the current location of your website's visitor.

This feature is helpful for providing better browsing experience to the site visitor. For example, you can return the search results that are physically close to the user's location.

Finding a Visitor's Coordinates

Getting the position information of the site visitor using the HTML5 geolocation API is fairly simple. It utilizes the three methods that are packed into the `navigator.geolocation` object — `getCurrentPosition()`, `watchPosition()` and `clearWatch()`.

The following is a simple example of geolocation that displays your current position. But, first you need to agree to let the browser tell the web server about your position.

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Get Current Position</title>
<script>
    function showPosition() {
        if(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(function(position) {
                var positionInfo = "Your current position is (" + "Latitude: " +
position.coords.latitude + ", " + "Longitude: " + position.coords.longitude + ")";
                document.getElementById("result").innerHTML = positionInfo;
            });
        } else {
            alert("Sorry, your browser does not support HTML5 geolocation.");
        }
    }
</script>
</head>
<body>
    <div id="result">
        <!--Position information will be inserted here-->
    </div>
    <button type="button" onclick="showPosition();">Show Position</button>
</body>
</html>
```

Note: The web browsers won't share the visitor location with a web page unless the visitor gives it explicit permission. The geolocation standard makes it an official rule to get user permission for every website that wants location data.

Dealing with Errors and Rejections

There may be a situation when a user does not want to share his location data with you. To deal with such situations, you can supply two functions when you call the `getCurrentLocation()` function.

The first function is called if your geolocation attempt is successful, while the second is called if your geolocation attempt ends in failure. Let's check out an example:

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Handling Geolocation Errors</title>
<script>
    // Set up global variable
    var result;

    function showPosition() {
        // Store the element where the page displays the result
        result = document.getElementById("result");

        // If geolocation is available, try to get the visitor's position
        if(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
            result.innerHTML = "Getting the position information...";
        } else {
            alert("Sorry, your browser does not support HTML5 geolocation.");
        }
    };

    // Define callback function for successful attempt
    function successCallback(position) {
        result.innerHTML = "Your current position is (" + "Latitude: " +
position.coords.latitude + ", " + "Longitude: " + position.coords.longitude + ")";
    }

    // Define callback function for failed attempt
    function errorCallback(error) {
        if(error.code == 1) {
            result.innerHTML = "You've decided not to share your position, but it's
OK. We won't ask you again.";
        } else if(error.code == 2) {
            result.innerHTML = "The network is down or the positioning service can't
be reached.";
        } else if(error.code == 3) {
            result.innerHTML = "The attempt timed out before it could get the
location data.";
        } else {
            result.innerHTML = "Geolocation failed due to unknown error.";
        }
    }
</script>
</head>
<body>
    <div id="result">
        <!--Position information will be inserted here-->
    </div>
    <button type="button" onclick="showPosition();">Show Position</button>
</body>
</html>
```

Showing Location on Google Map

You can do very interesting things with geolocation data, like showing the user location on Google map. The following example will show your current location on Google map based the latitude and longitude data retrieved through the HTML5 geolocation feature.

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Using the Google Maps</title>
<script>
    function showPosition() {
        navigator.geolocation.getCurrentPosition(showMap);
    }

    function showMap(position) {
        // Get location data
        var latlong = position.coords.latitude + "," + position.coords.longitude;

        // Set Google map source url
        var mapLink = "https://maps.googleapis.com/maps/api/staticmap?
center="+latlong+"&zoom=16&size=400x300&output=embed";

        // Create and insert Google map
        document.getElementById("embedMap").innerHTML = "<img alt='Map Holder'
src='"+ mapLink +"'>";
    }
</script>
</head>
<body>
    <button type="button" onclick="showPosition();">Show My Position on Google
Map</button>
    <div id="embedMap">
        <!--Google map will be embedded here-->
    </div>
</body>
</html>
```

The above example will simply show the location on the Google map using a static image. However, you can also create interactive Google maps with dragging, zoom in/out and other features that you have come across in your real life. Let's take a look at the following example:

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Using the Google Maps</title>
<script src="https://maps.google.com/maps/api/js?sensor=false"></script>
<script>
function showPosition() {
    if(navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showMap, showError);
    } else {
        alert("Sorry, your browser does not support HTML5 geolocation.");
```

```
    }
}

// Define callback function for successful attempt
function showMap(position) {
    // Get location data
    lat = position.coords.latitude;
    long = position.coords.longitude;
    var latlong = new google.maps.LatLng(lat, long);

    var myOptions = {
        center: latlong,
        zoom: 16,
        mapTypeControl: true,
        navigationControlOptions: {
            style:google.maps.NavigationControlStyle.SMALL
        }
    }

    var map = new google.maps.Map(document.getElementById("embedMap"), myOptions);
    var marker = new google.maps.Marker({ position:latlong, map:map, title:"You are
here!" });
}

// Define callback function for failed attempt
function showError(error) {
    if(error.code == 1) {
        result.innerHTML = "You've decided not to share your position, but it's OK.
We won't ask you again.";
    } else if(error.code == 2) {
        result.innerHTML = "The network is down or the positioning service can't be
reached.";
    } else if(error.code == 3) {
        result.innerHTML = "The attempt timed out before it could get the location
data.";
    } else {
        result.innerHTML = "Geolocation failed due to unknown error.";
    }
}
</script>
</head>
<body>
    <button type="button" onclick="showPosition();">Show My Position on Google
Map</button>
    <div id="embedMap" style="width: 400px; height: 300px;">
        <!--Google map will be embedded here-->
    </div>
</body>
</html>
```

Please check out the following URL to learn more about the Google Maps Javascript API: https://developers.google.com/maps/documentation/javascript/reference.

Monitoring the Visitor's Movement

All the examples we've used so far have relied on the `getCurrentPosition()` method. However, the geolocation object has another method `watchPosition()` that allow you to track the visitor's movement by

returning the updated position as the location changes.

The `watchPosition()` has the same input parameters as `getCurrentPosition()`.
However, `watchPosition()` may trigger the success function multiple times — when it gets the location for the first time, and again, whenever it detects a new position. Let's see how this works:

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Watching Position</title>
<script>
    // Set global variable
    var watchID;

    function showPosition() {
        if(navigator.geolocation) {
            watchID = navigator.geolocation.watchPosition(successCallback);
        } else {
            alert("Sorry, your browser does not support HTML5 geolocation.");
        }
    }

    function successCallback(position) {
        toggleWatchBtn.innerHTML = "Stop Watching";

        // Check position has been changed or not before doing anything
        if(prevLat != position.coords.latitude || prevLong !=
position.coords.longitude){

            // Set previous location
            var prevLat = position.coords.latitude;
            var prevLong = position.coords.longitude;

            // Get current position
            var positionInfo = "Your current position is (" + "Latitude: " +
position.coords.latitude + ", " + "Longitude: " + position.coords.longitude + ")";
            document.getElementById("result").innerHTML = positionInfo;

        }

    }

    function startWatch() {
        var result = document.getElementById("result");

        var toggleWatchBtn = document.getElementById("toggleWatchBtn");

        toggleWatchBtn.onclick = function() {
            if(watchID) {
                toggleWatchBtn.innerHTML = "Start Watching";
                navigator.geolocation.clearWatch(watchID);
                watchID = false;
            } else {
                toggleWatchBtn.innerHTML = "Aquiring Geo Location...";
                showPosition();
            }
        }
    }
```

```
    // Initialise the whole system (above)
    window.onload = startWatch;
</script>
</head>
<body>
    <button type="button" id="toggleWatchBtn">Start Watching</button>
    <div id="result">
        <!--Position information will be inserted here-->
    </div>
</body>
</html>
```