**Traffic management using IOT**

**Create a WebApp For Traffic Management Using IOT**

# CODING:

**getCurrentPosition() method:**

The getCurrentPosition(successCallback, errorCallback, options) method steps are:

If the current settings object's relevant global object's associated Document is not fully active:

1. [Call back with error](https://www.w3.org/TR/geolocation/) errorCallback and [POSITION\_UNAVAILABLE](https://www.w3.org/TR/geolocation/).
2. Terminate this algorithm.
3. In parallel, [request a position](https://www.w3.org/TR/geolocation/) passing successCallback, errorCallback, and options.

# // A one-shot position request:

navigator.geolocation.getCurrentPosition(position => {

const { latitude, longitude } = position.coords;

*// Show a map centered at latitude / longitude.*

});

# watchPosition() method:

The watchPosition(successCallback, errorCallback, options) method steps are:

If the current settings object's relevant global object's associated Document is not fully active:

1. [Call back with error](https://www.w3.org/TR/geolocation/)

passing errorCallback and [POSITION\_UNAVAILABLE](https://www.w3.org/TR/geolocation/).

1. Return 0.
2. Let watchId be an implementation-defined [unsigned long](https://webidl.spec.whatwg.org/) that is greater than zero.
3. Append watchId to this's [[[watchIDs]]](https://www.w3.org/TR/geolocation/).
4. In parallel, [request a](https://www.w3.org/TR/geolocation/)

position passing successCallback, errorCallback, options, and watchId.

1. Return watchId.

# Watching a position for repeated updates:

const watchId = navigator.geolocation.watchPosition(position => {

const { latitude, longitude } = position.coords;

*// Show a map centered at latitude / longitude.*

});

# clearWatch() method:

When clearWatch**()** is invoked, the user agent MUST:

1) Remove watchId from this's [[[watchIDs]]](https://www.w3.org/TR/geolocation/).

# Using clearWatch():

const watchId = navigator.geolocation.watchPosition(

position => console.log(position)

);

function buttonClickHandler() {

*// Cancel the updates when the user clicks a button.*

navigator.geolocation.clearWatch(watchId);

}

# A HTML button that when pressed stops watching the position.

<**button** onclick="buttonClickHandler()">

Stop watching location

</**button**>

# Handling errors:

*// Request repeated updates.*const watchId = navigator.geolocation.watchPosition(

scrollMap, handleError

);

function scrollMap(position) {

const { latitude, longitude } = position.coords;

*// Scroll map to latitude / longitude.*

}

function handleError(error) {

*// Display error based on the error code.*

const { code } = error;

switch (code) {

case GeolocationPositionError.TIMEOUT:

*// Handle timeout.*

break;

case GeolocationPositionError.PERMISSION\_DENIED:

*// User denied the request.*

break;

case GeolocationPositionError.POSITION\_UNAVAILABLE:

*// Position not available.*

break;

}

}

# Getting cached position:

navigator.geolocation.getCurrentPosition(

successCallback,

console.error,

{ maximumAge: 600\_000 }

);

function successCallback(position) {

*// By using the 'maximumAge' member above, the position*

*// object is guaranteed to be at most 10 minutes old.*

}

# Timing out a position request:

*// Request a position. We are only willing to wait 10// seconds for it.*

navigator.geolocation.getCurrentPosition(

successCallback,

errorCallback,

{ timeout: 10\_000 }

);

function successCallback(position) {

*// Request finished in under 10 seconds...*

}

function errorCallback(error) {

switch (error.code) {

case GeolocationPositionError.TIMEOUT:

*// We didn't get it in a timely fashion.*

doFallback();

*// Acquire a new position object,*

*// as long as it takes.*

navigator.geolocation.getCurrentPosition(

successCallback, errorCallback

);

break;

case "...": *// treat the other error cases.*

}

}

function doFallback() {}

# Enabling the Geolocation API in an iframe:

<**iframe**

src="https://third-party.com"

allow="geolocation">

</**iframe**>

# Permissions Policy over HTTP:

Permissions-Policy: geolocation=()

# PositionOptions dictionary:

[PositionOptions](https://www.w3.org/TR/geolocation/) {

boolean [enableHighAccuracy](https://www.w3.org/TR/geolocation/) = false;

[Clamp] unsigned long [timeout](https://www.w3.org/TR/geolocation/) = 0xFFFFFFFF; [Clamp] unsigned long [maximumAge](https://www.w3.org/TR/geolocation/) = 0;

};