



## Contents

- [1 Introduction](#)
- [2 Using JavaScript Location API to Update the User Position on a Map](#)
- [3 Allow user to reach out and touch](#)
- [4 Emulator Support](#)
- [5 Download Widget](#)
- [6 See also](#)

## Introduction

In this article we are going to explain how to display your current GPS position on a Map inside your WRT widget. Enclosed in this article you will also find a full implemented widget and the respective source code to download.

The main objective behind this article is to show how easy and fast you can use maps to improve the user experience. We are also going to provide some tips for you to create a smooth user experience. At the time of this writing only Google Maps provide a public API to use maps on web and mobile.

## Using JavaScript Location API to Update the User Position on a Map

The web runtime widget triggers *onload* when the document is finished loading. The contents of *onload* is one or more JavaScript commands.

```
...
<body onload="javascript:init();" onunload="GUnload();" >
  <div id="container">
    <div id="mapbody">
      <div id="map"></div>
    </div>
  </div>
</body>
...
```

You can customize the marker by using the following javascript code.

```
var youIcon = new GIcon(G_DEFAULT_ICON);
youIcon.image = "http://i2tecnologia.com.br/fuelup/you-ico.png";
youIcon.iconSize = new GSize(49, 49);
```

The *init* function creates a new map and setup the application.

```
...
function init()
{
    if (GBrowserIsCompatible())
```

## Using\_built-in\_GPS\_and\_JavaScript\_to\_display\_your\_current\_position\_on\_Google\_Maps

```
{
    map = new GMap2(document.getElementById('map'));
    map.addControl(new GLargeMapControl());

    setup();
    getLocation();
}
...

```



The security prompt ask for user authorization

The Location Service API allows widgets to retrieve information about the physical location of a device and to perform calculations based on location information. The API is integrated into WRT through the device object.

To use the Location Service API, your widget must first create a service object for it using the *device.getServiceObject()* method. Use *Service.Location* to identify the service provider and *ILocation* to identify the supported interface:

```
function setup()
{
    try {
        so = device.getServiceObject("Service.Location", "ILocation");
    }
    catch (e) {
        alert('(006) Error ::setup ' + e);
    }
}
```

The *getLocation* function check for errors and use the *Trace* method to retrieve periodic updates about the

## Using built-in GPS and JavaScript to display your current position on Google Maps

current location of the device based on a predefined update interval. Since this is an asynchronous method you have to define a callback function to receive data.

```
function getLocation() {
  try {
    var updateoptions = new Object();
    updateoptions.PartialUpdates = false;

    var criteria = new Object();
    criteria.LocationInformationClass = "GenericLocationInfo";
    criteria.Updateoptions = updateoptions;

    var result = so.ILocation.Trace(criteria, callbackLocation);
    var errCode = result.ErrorCode;
    if(errCode) {
      alert("(005) GPS Error: " + errCode + " " + result.ErrorMessage);
    }
  }
  catch (e) {
    alert("(004) ::getLocation error: " + e);
  }
}
```

The following callback function check for errors and call *createUserMarker* function to plot the mark on screen.

```
function callbackLocation(transId, eventCode, result)
{
  var errCode = result.ErrorCode;
  if (errCode) {
    alert("(003) GPS Error: " + errCode + " " + result.ErrorMessage);
  }
  else {
    createUserMarker(result.ReturnValue.Latitude,
                     result.ReturnValue.Longitude,
                     "You");
  }
}
```

The *createUserMarker* function check if the user mark was already created if so it only updates the user mark position on the screen.

```
function createUserMarker(aLatitude, aLongitude, name)
{
  if(userMarker == null) {
    try {
      userPoint = new GLatLng(aLatitude, aLongitude);
      userMarker = new GMarker(userPoint, markerUserOptions);

      var html = '<strong>' + name + '</strong>';

      GEvent.addListener(userMarker, 'click', function() {
        userMarker.openInfoWindowHtml(html);
      });

      map.setCenter(userPoint, 15);
      map.addOverlay(userMarker);
    } catch(e) {
      alert("(001) Error: " + e);
    }
  }
}
```

## Using built-in GPS and JavaScript to display your current position on Google Maps

```
} else {  
  try {  
    userPoint = new GLatLng(aLatitude, aLongitude);  
    userMarker.setLatLng(userPoint);  
    map.setCenter(userPoint, 15);  
  } catch(e) {  
    alert("(002) Error: " + e);  
  }  
}  
}  
}
```



Widget User Interface

## Allow user to reach out and touch

By providing a map view to your apps users will be able to find what they're looking for in the real world. Studies have shown that many people get in troubles when using paper maps, and in a tiny screen it's even worse, so when writing location-aware widgets pay attention to the following topics:

- Be careful of user's privacy
- Enhance user experience through connecting the physical environment and digital words
- Provide a graphical interface which allows the user to experiment and explore the application
- Use graphical symbols to represent an information
- Determine if standard symbols (e.g. food, gas symbols) exist before creating your own
- Use graphical symbols that are easy to touch

Allow user to reach out and touch

Using\_built-in\_GPS\_and\_JavaScript\_to\_display\_your\_current\_position\_on\_Google\_Maps

- Use graphical symbols to provide valuable information around the users current location
- Provide a user interface that any user initiated command should be immediately reversible or cancelable

## Emulator Support

The S60 5th Edition SDK emulator provides full support for the Location Service API.

GPS data is simulated by using Position System plugins (PSY) as dummy providers. Dummy providers are available on the emulator by default.

## Download Widget

[Download MyLocation widget](#)

## See also

- [High performance Widgets: Combine your JavaScripts and CSS in external Files](#)
- [High performance Widgets: Optimize your JavaScript](#)

--Felipe Andrade 19:36, 5 June 2009 (EEST)