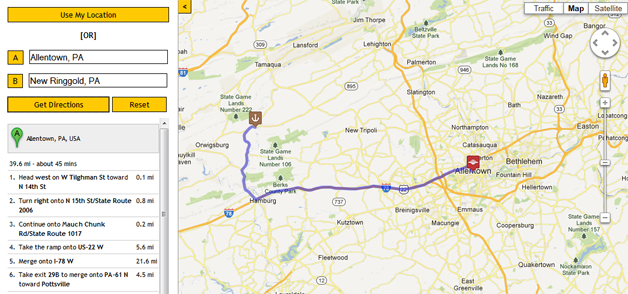
**Custom Directions Panel with Google Maps API v3**

[4 Comments](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#respond)Posted by [thewebstorebyg](http://thewebstorebyg.wordpress.com/author/thewebstorebyg/" \o "Posts by thewebstorebyg) on January 11, 2013

I’ve been working a lot on Google Maps API V3 these days and most recently worked on the get directions. So I thought I will put up a small tutorial which covers the below:

* [Google Maps like Directions Panel with Toggle option](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#custom)
* [Custom Direction Markers](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#markers)
* [Autocomplete Integration (Google Places API)](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#auto)
* [Traffic Layer Integration (Adding Custom Control To Map)](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#traffic)
* [HTML 5 GeoLocation, The Google Geocoding API Integration](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#geo)
* [Download the complete code](http://thewebstorebyg.wordpress.com/2013/01/11/custom-directions-panel-with-google-maps-api-v3/#download)

[](http://thewebstorebyg.files.wordpress.com/2013/01/full_map.png)  
**Prerequisites:** Knowledge on XHTML / HTML5, CSS, Good understanding of JS / jQuery is required.

**For Map Basics refer:** [Google Maps API Integration](http://bit.ly/11lTeqq)

**Custom Directions Panel or Google Maps Like Direction Panel:**

**Directions Setup Basic:**

DirectionsService() is an asynchronous service comes along with the Google Maps API V3 which will generate the directions results upon providing the appropriate direction requests(source and destination). This service will generate a JSON as output, to render this JSON we can use the default rendering option DirectionsRenderer() provided in the API or we can also use the routes object to manually render each and every detail. I’ve used the DirectionsRenderer() option in this example.

Here is a glance at the code for the above two services. These needs to be setup once the map is initialized.

directionsSetup = function() {

directionsService = new google.maps.DirectionsService();

directionsDisplay = new google.maps.DirectionsRenderer({

suppressMarkers: true

});

directionsDisplay.setMap(map);

directionsDisplay.setPanel($Selectors.dirSteps[0]);

}, // direstionsSetup Ends

**Get Directions:**

Also, in the above code I’ve passed the option "suppressMarkers: true" in the DirectionsRenderer service which will suppress the default map marker images provided by Google and allows us to use the custom marker images.

The below function helps to generate the directions by taking the source and destination values from the input fields in the HTML or geolocation and passes the values to directions service, on success it will generate a response.

The div where the response needs to be rendered should be defined using:directionsDisplay.setPanel($Selectors.dirSteps[0])

Along with the setPanel we also need setDiretions to display the response in the div.  
directionsDisplay.setDirections(response)

The response contains route object which will contain all the direction details like start / end position coordinates, waypoints and lot of other details.

**Tip:** If you want to completely customize the look and feel of the direction steps you need to use the **route object** to extract the details like, position, distance, route text etc,. and style them accordingly. If you need more details on this just leave a comment I can help you with whatever I know.

directionsRender = function(source, destination) {

var request = {

origin: source,

destination: destination,

provideRouteAlternatives: false,

travelMode: google.maps.DirectionsTravelMode.DRIVING

};

directionsService.route(request, function(response, status) {

if (status == google.maps.DirectionsStatus.OK) {

directionsDisplay.setDirections(response);

var \_route = response.routes[0].legs[0];

pinA = new google.maps.Marker({

position: \_route.start\_location,

map: map,

icon: markerA

}),

pinB = new google.maps.Marker({

position: \_route.end\_location,

map: map,

icon: markerB

});

}

});

}, // directionsRender Ends

A detailed documentation on the DirectionsService is available [here](https://developers.google.com/maps/documentation/javascript/directions" \t "_blank).

**Custom Direction Markers**

To customize the makers on the map we need to define them using the below code:

**Defining Custom Marker Images:**

var markerA = new google.maps.MarkerImage('m1.png',

new google.maps.Size(24, 27),

new google.maps.Point(0, 0),

new google.maps.Point(12, 27));

var markerB = new google.maps.MarkerImage('m2.png',

new google.maps.Size(24, 28),

new google.maps.Point(0, 0),

new google.maps.Point(12, 28))

**Replacing the Markers on the Map:**

Once the marker images are defined we can simple map then to the A and B markers on the map using the Marker object like shown below:

Here I have used the route object and manually setup the start and end markers.  
**Important:** If you don’t use the below code you will not see any markers on the Map at all.

var \_route = response.routes[0].legs[0];

pinA = new google.maps.Marker({

position: \_route.start\_location,

map: map,

icon: markerA

});

pinB = new google.maps.Marker({

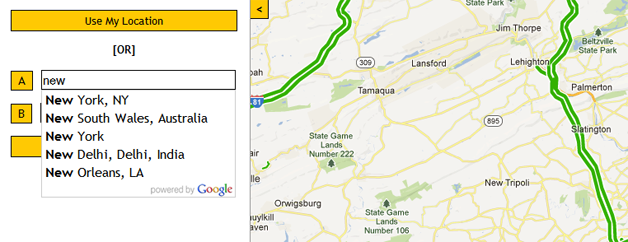
position: \_route.end\_location,

map: map,

icon: markerB

});

**Autocomplete Integration (Google Places API):**

[](http://thewebstorebyg.files.wordpress.com/2013/01/auto_complete.png)

Autocomplete or the Google Places API is the most easiest to integrate. This is a really powerful feature and you can do a whole bunch of things with it.

Here is a glace at the code. It’s very simple and straight forward all you need to do is pass the input field.

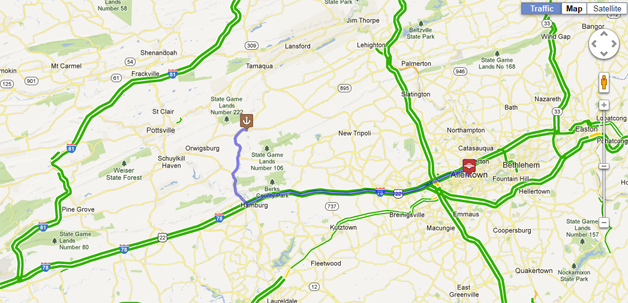
autoCompleteSetup = function() {

autoSrc = new google.maps.places.Autocomplete($Selectors.dirSrc[0]);

autoDest = new google.maps.places.Autocomplete($Selectors.dirDst[0]);

}, // autoCompleteSetup Ends

**Traffic Layer Integration (Adding Custom Control To Map)**

[](http://thewebstorebyg.files.wordpress.com/2013/01/traffic_layer.png)

There is no straight forward parameter / code for traffic layer integration. You need to setup a custom map control and bind it with an event to enable and disable the traffic layer.

Good thing is it’s not that complicated at all. Have a look at the code below:

trafficSetup = function() {

// Creating a Custom Control and appending it to the map

var controlDiv = document.createElement('div'),

controlUI = document.createElement('div'),

trafficLayer = new google.maps.TrafficLayer();

jQuery(controlDiv).addClass('gmap-control-container').addClass('gmnoprint');

jQuery(controlUI).text('Traffic').addClass('gmap-control');

jQuery(controlDiv).append(controlUI);

// Traffic Btn Click Event

google.maps.event.addDomListener(controlUI, 'click', function() {

if (typeof trafficLayer.getMap() == 'undefined' || trafficLayer.getMap() === null) {

jQuery(controlUI).addClass('gmap-control-active');

trafficLayer.setMap(map);

} else {

trafficLayer.setMap(null);

jQuery(controlUI).removeClass('gmap-control-active');

}

});

map.controls[google.maps.ControlPosition.TOP\_RIGHT].push(controlDiv);

}, // trafficSetup Ends

Learn more about Custom Map Controls [Here](https://developers.google.com/maps/documentation/javascript/controls" \l "CustomControls" \t "_blank).

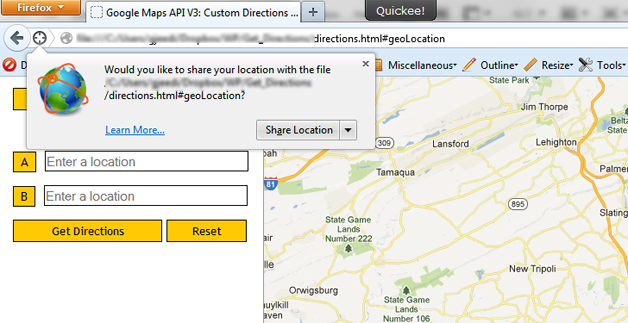
var trafficLayer = new google.maps.TrafficLayer();

trafficLayer.setMap(map);

The above code will setup the traffic layer on top of the map. But once the traffic layer is set up you don’t have any direct option to disable it. Hence I have a custom control in place which helps me to easily enable and disable the traffic layer.

I’ve learned all about traffic integration from this [tutorial](http://pietervogelaar.nl/google-maps-api-v3-traffic-toggle-button/#.UPAf4qGAfeS). Pieter Vogelaar the author of the tutorial also included the code to display details of the traffic legends.

**HTML 5 GeoLocation, The Google Geocoding API Integration**

[](http://thewebstorebyg.files.wordpress.com/2013/01/geo_location.png)

The geolocation API lets you share your location(latitude and longitude) once the user allows the api to track the location we can pass the coordinates to the Google Geocoding API and get a formatted address.

**Invoking the HTML 5 Geolocation:**

$Selectors.useGPSBtn.on('click', function(e) {

if (navigator.geolocation) {

navigator.geolocation.getCurrentPosition(function(position) {

fetchAddress(position);

});

}

});

**Using the Google Geocoding API:**

fetchAddress = function(p) {

var Position = new google.maps.LatLng(p.coords.latitude, p.coords.longitude),

Locater = new google.maps.Geocoder();

Locater.geocode({'latLng': Position}, function (results, status) {

if (status == google.maps.GeocoderStatus.OK) {

var \_r = results[0];

$Selectors.dirSrc.val(\_r.formatted\_address);

}

});

} // fetchAddress Ends

This tutorial is still a work in progress. My intention is to just give a basic idea on how we can put things together. Any feedback is welcome. Please comment or [email](mailto:thewebstorebyg@gmail.com) me for any questions.

[Click Here](https://www.box.com/s/7i81zmnk4s6k85b4zzkt) to download the complete code.

Good Luck!