The Interactive Maps JavaScript API lets you customize maps with your own content and imagery for display on web pages and mobile devices. The Maps JavaScript API features different basic map types (base map, traffic, hybrid etc.) which you can modify using layers and styles, controls and events, and various services and libraries.

#### **Map Scripts:**

Script use for loading map tiles is very similar in both MapmyIndia & Google APIs. If you are already using Google APIs, you will find the following mechanism very similar and can easily integrate the map loading scripts in your source code:

#### 1. MapmyIndia:

<script src="https://apis.mapmyindia.com/advancedmaps/v1/{your API key}/map\_load?v=01"></script>

#### 2. Google:

<script src="https://maps.googleapis.com/maps/api/js?key=<lic key>&callback=initMap"async
defer></script>

## **Map Initialization:**

Map initialization is comparatively a simpler task in MapmyIndia, which uses just three lines to code.

The following code snippet is used for initializing map loading for respective map providers:

#### 1. MapmyIndia:

```
<! DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"lang="en" xml:lang="en">
<head>
<title>MapmyIndia Maps</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
// Style
<style type="text/css">
#map {
height: 400px; width: 100%;
}
```

```
</style>
// Map Scriptlet
<script
src="https://apis.mapmyindia.com/advancedmaps/v1/your_map_key_here/map_load?v=0.1">
</script>
</head>
<body><h3>My MapmyIndia Maps Demo</h3>
// Map container div
<div id="map"></div>
// Script to initialize map
<script>
var map = new MapmyIndia.Map('map', {center: [28.549948, 77.268241], zoomControl: true,
hybrid: true});
</script></body>
</html>
2. Google:
<!DOCTYPE html>
<html><head><style>
//style
   #map {
height: 400px;
width: 100%;
</style>
</head>
<body>
<h3>My Google Maps Demo</h3>
```

```
// Map container Div
<div id="map"></div>
//Script to initialize map
<script>
functioninitMap() {
varuluru = {lat: 28.549948, lng: 77.268241};
var map = new google.maps.Map(document.getElementById('map'), {
zoom: 4,
center: uluru
    });
</script>
//map scriptlet
<script async defer
  src="https://maps.googleapis.com/maps/api/js?key=<lic key>&callback=initMap">
</script>
</body>
</html>
```

## Map Marker:

Map makers are very similarly codified with both map providers. Any developer can easily modify his or her code to migrate to MapmyIndia.

The code snippet for markers are:

# 1. MapmyIndia:

For plotting marker on mapmyindia map user need to use **L.marker** method.

```
var marker = L.marker([28.549948, 77.268241]).addTo(map);
```

### 2. Google:

For plotting marker on google map user need to use new google.maps.Marker method.

```
var marker = new google.maps.Marker({
  position: {lat: 28.549948, lng: 77.268241},
  map: map
});
```

## **Map Custom Marker:**

Custom markers are easy to code and migrate to MapmyIndia APIs if developer is already familiar with Google APIs:

#### 1. MapmyIndia:

```
var icons = {
icon: 'icon.png' //icon
     };
// method for plotting marker
var marker = new google.maps.Marker({
position: new google.maps.LatLng(-33.91818154739766, 151.2346203981781),
```

```
icon: icons,
map: map
});
```

#### Marker Info Window:

Following is a code snippet for marker info window in both map providers. One look at the code snippet demonstrates how easy creating an info window is within MapmyIndia Maps.

#### 1. MapmyIndia:

User need to use **bindpopup**() method to bind data with marker and **openpopup**() method to open infowindow.

```
var marker = L.marker([28.61, 77.23]).addTo(map).bindPopup('Hello World!!') .openPopup();
```

## 2. Google:

```
// infowindowcontructor
varinfowindow = new google.maps.InfoWindow({
    content: "Hello World!!"
});

// Marker Plotting
var marker = new google.maps.Marker({
    position: {lat: 28.549948, lng: 77.268241},
    map: map,
    title: 'Uluru (Ayers Rock)'
});
marker.addListener('click', function() {
    infowindow.open(map, marker); // Open InfoWindow
})
```

## PolyLine:

Creating a polyline means specifying a series of coordinates as an array of LatLng objects.

Following are code snippets from each map providers comparing the implementations in each:

#### 1. MapmyIndia:

```
var map = new MapmyIndia.Map('map', {center: [28.549948, 77.268241], zoomControl: true,
hybrid: true});
/*array of wgs points*/
var pts = [
    [28.535405, 77.258151],
    [28.534327, 77.260082],
    [28.535802, 77.258945],
    [ 28.535881, 77.257610]
        1;
Var polyParam = {
color: "red", //color of the polyline
weight: 4, // width of polyline in pixels
opacity: 0.5 //opacity of the polyline in %/100, here it is 50%
        };
Var poly = new L.Polyline(pts, polyParam);/*polyline with given points and red color is
created*/
map.addLayer(poly);
        /*** zoom the map to the polyline**/
map.fitBounds(poly.getBounds());
```

```
// This example creates a 2-pixel-wide red polyline showing the path of
// the first trans-Pacific flight between Oakland, CA, and Brisbane,
// Australia which was made by Charles Kingsford Smith.

functioninitMap(){
  var map =newgoogle.maps.Map(document.getElementById('map'),{
    zoom:3,
```

```
center:{lat:0,lng:-180},
 mapTypeId:'terrain'
});
varflightPlanCoordinates=[
 {lat:37.772,lng:-122.214},
 {lat:21.291,lng:-157.821},
 {lat:-18.142,lng:178.431},
 {lat:-27.467,lng:153.027}
varflightPath=newgoogle.maps.Polyline({
 path:flightPlanCoordinates,
 geodesic:true, // ellipsoidal path(curve)
 strokeColor: '#FF0000',
 strokeOpacity:1.0,
 strokeWeight:2
});
flightPath.setMap(map);
```

# Polygon:

A polygon represents an area enclosed by a closed path (or loop), which is defined by a series of coordinates.

Following is a comparison of implementation of the same in both map providers:

# 1. MapmyIndia:

```
var map = new MapmyIndia.Map('map', {center: [28.549948, 77.268241], zoomControl: true, hybrid: true});
/*Array of WGS location object*/
pts = [[
      [28.535405, 77.258151],[ 28.534327, 77.260082],[ 28.535802, 77.258945],[ 28.535881, 77.257610],
      [ 28.545645, 77.257546]]];
Var poly = new L.polygon(
```

```
pts, // set of latLng
{
color: "green", //boundry color of polygon
fillcolor: "green", // color of polygon
weight: 4, // width of polygon in pixels
opacity: "0.5" //opacity of the polyline in %/100, here it is 50%
}); /*creates a polygon with color:green*/
map.fitBounds(poly.getBounds()); // Bounds polygon to map
```

```
// This example creates a simple polygon representing the Bermuda Triangle.
functioninitMap(){
 var map =newgoogle.maps.Map(document.getElementById('map'),{
  center:{lat:24.886,lng:-70.268},
  mapTypeld:'terrain'
});
// Define the LatLng coordinates for the polygon's path.
 vartriangleCoords=[
 {lat:25.774,lng:-80.190},
 {lat:18.466,lng:-66.118},
  {lat:32.321,lng:-64.757},
 {lat:25.774,lng:-80.190}
];
// Construct the polygon.
 varbermudaTriangle=newgoogle.maps.Polygon({
  paths:triangleCoords,
  strokeColor: '#FF0000',
  strokeOpacity:0.8,
  strokeWeight:2,
  fillColor: '#FF0000',
 fillOpacity:0.35
 });
```

```
bermudaTriangle.setMap(map);
}
```

#### Circle:

Following is the comparison of implementing circle in MapmyIndia vs the implementation in Google.

#### 1. MapmyIndia:

```
var map = new MapmyIndia.Map('map', {center: [28.549948, 77.268241], zoomControl: true, hybrid: true});
var currentDiameter = L.circle([lat, lon], {
    color: 'pink', //boundry color of circle
    fillColor: '#FFC0CB', // color of circle
    fillOpacity: 0.5, //opacity of the polyline in %/100, here it is 50%
    radius: "5" // radius of circle
        });
currentDiameter.addTo(map); // add circle to map
map.fitBounds(currentDiameter.getBounds()); // Bound to map
```

```
functioninitMap(){
    // Create the map.
    var map =newgoogle.maps.Map(document.getElementById('map'),{
        zoom:4,
        center:{lat:37.090,Ing:-95.712},
        mapTypeId:'terrain'
    });

// Construct the circle for each value in citymap.
    // Add the circle for this city to the map.
    varcityCircle=newgoogle.maps.Circle({
        strokeColor:'#FF0000',
        strokeOpacity:0.8,
        strokeWeight:2,
```

```
fillColor:'#FF0000',
fillOpacity:0.35,
map: map,
center:{lat:25.774,lng:-80.190},
radius:radius
});
}
```

#### Marker Cluster:

#### MapmyIndia:

The Mapmylndia basic cluster provides basic implementation of the leaflet's cluster functionality. To set the basic cluster up below is the file composition

```
k rel="stylesheet" href="css/MarkerCluster.css"/>
<link rel="stylesheet" href="css/MarkerCluster.Default.css" />
<script src="js/leaflet.markercluster-src.js"></script>
map=new MapmyIndia.Map('map-container',{ zoomControl: true,hybrid:true,zoom:10});/*map
initialized*/
vartest num markers = 100;
        //function to generate random markers//
vargenerate markers = function(num markers) {
var bounds = map.getBounds();
varmap sw = bounds.getSouthWest();
varmap ne = bounds.getNorthEast();
varing span = map ne.ing - map sw.ing;
varlat span = map ne.lat - map sw.lat;
var markers = [];
markerClusters = L.markerClusterGroup({chunkedLoading: true, maxClusterRadius: 100,
disableClusteringAtZoom: 26, singleMarkerMode: false, addRegionToolTips: false});
// by default the Cluster group some defaults:
//showCoverageOnHover: When you mouse over a cluster it shows the bounds of its markers.
```

//zoomToBoundsOnClick: When you click a cluster we zoom to its bounds.

//spiderfyOnMaxZoom: When you click a cluster at the bottom zoom level we spiderfy it so you can see all of its markers. (Note: the spiderfy occurs at the current zoom level if all items within the cluster are still clustered at the maximum zoom level or at zoom specified by disableClusteringAtZoom option)

//removeOutsideVisibleBounds: Clusters and markers too far from the viewport are removed from the map for performance.spiderLegPolylineOptions

```
functioninitMap(){
  var map =newgoogle.maps.Map(document.getElementById('map'),{
   zoom:3,
   center:{lat:-28.024,lng:140.887}
  });
```

```
// Create an array of alphabetical characters used to label the markers.
    var labels ='ABCDEFGHIJKLMNOPQRSTUVWXYZ':
    // Add some markers to the map.
    // Note: The code uses the JavaScript Array.prototype.map() method to
    // create an array of markers based on a given "locations" array.
    // The map() method here has nothing to do with the Google Maps API.
    var markers =locations.map(function(location,i){
     returnnewgoogle.maps.Marker({
      position: location,
     label: labels[i%labels.length]
     });
    });
    // Add a marker clusterer to manage the markers.
    varmarkerCluster=newMarkerClusterer(map, markers,
{imagePath: https://developers.google.com/maps/documentation/javascript/examples/marker
clusterer/m'});
   var locations =[
    {lat:-31.563910,lng:147.154312},
    {lat:-33.718234,lng:150.363181},
    {lat:-33.727111,lng:150.371124},
    {lat:-33.848588,lng:151.209834},
    {lat:-33.851702,lng:151.216968},
    {lat:-34.671264,lng:150.863657},
    {lat:-35.304724,lng:148.662905},
    {lat:-36.817685,lng:175.699196},
    {lat:-36.828611,lng:175.790222},
    {lat:-37.750000,lng:145.116667},
    {lat:-37.759859,lng:145.128708},
    {lat:-37.765015,lng:145.133858},
    {lat:-37.770104,lng:145.143299},
    {lat:-37.773700,lng:145.145187},
```

{lat:-37.774785,lng:145.137978}, {lat:-37.819616,lng:144.968119}, {lat:-38.330766,lng:144.695692}, {lat:-39.927193,lng:175.053218}, {lat:-41.330162,lng:174.865694}, {lat:-42.734358,lng:147.439506}, {lat:-42.734358,lng:147.501315},

```
{lat:-42.735258,lng:147.438000},
 {lat:-43.999792,lng:170.463352}
]
</script>
```

<scriptsrc="https://developers.google.com/maps/documentation/javascript/examples/markerc lusterer/markerclusterer.js">

## **Decorated polyline:**

#### MapmyIndia:

MapmyIndia also provide feature for creating decorated polyline. In case user wants to show moving vehicle/marker by using following code snippet.

```
<script src="js/leaflet.polylineDecorator.js"></script> //Script need to include
<div id="map"></div>
    <script>
      var map = null;
      var poly = [];
      var decorator;
      var line;
      var center = new L.LatLng(28.549948, 77.268241);
      var interval = 0;
      var pp = [[28.56731, 77.346355], [28.567074, 77.346022], [28.566584, 77.345378],
[28.565689, 77.344144], [28.563503, 77.341312], [28.563277, 77.341076], [28.562636,
77.340186],
        [28.561986, 77.339328], [28.561788, 77.338942], [28.561675, 77.338695],
[28.561524, 77.33833], [28.56142, 77.338094], [28.561166, 77.337472], [28.560817,
77.336925],
        [28.560515, 77.335981], [28.560458, 77.33582], [28.560204, 77.33509], [28.559818,
77.33406], [28.558046, 77.329629], [28.557518, 77.328427], [28.556698, 77.326292],
        [28.55617, 77.325123], [28.555473, 77.323878], [28.555256, 77.323739], [28.555105,
77.323675], [28.554973, 77.323664], [28.554813, 77.323685], [28.554681, 77.323749],
```

```
[28.554577, 77.323824], [28.554464, 77.323963], [28.554407, 77.324124],
[28.554388, 77.324274], [28.554397, 77.324424], [28.554435, 77.324585], [28.55451,
77.324724],
        [28.554642, 77.324853], [28.555198, 77.325261], [28.555462, 77.3254], [28.555632,
77.325443], [28.55599, 77.325368], [28.55681, 77.324295], [28.557922, 77.322879],
        [28.558516, 77.321892], [28.55878, 77.321388], [28.559289, 77.320412], [28.55944,
77.320122], [28.560062, 77.318867], [28.560486, 77.31818], [28.56107, 77.317622],
        [28.561503, 77.317407], [28.562408, 77.316989], [28.564575, 77.315466],
[28.569183, 77.312097], [28.570464, 77.311088], [28.570831, 77.310798], [28.573921,
77.308481],
        [28.575551, 77.307354], [28.579329, 77.304554], [28.581308, 77.303127], [28.58177,
77.302784], [28.581949, 77.302655], [28.582637, 77.302129], [28.583014, 77.301839],
        [28.583532, 77.301453], [28.584022, 77.301088], [28.58486, 77.30023], [28.585934,
77.298621], [28.58616, 77.298396], [28.587036, 77.297742], [28.588685, 77.296916]];
      window.onload = function () {
        map = new MapmyIndia.Map('map', {
          center: center,
          editable: true.
          zoomControl: true,
          hybrid: true
        });
        //draw polyline
        drawCarMarkerOnPolyline();
drawCarMarkerOnPolyline : function(){
        removePolyline();
        var offset = 0; //intial offset value
```

var w = 14, h = 33;

//Polyline css

var linecss = {

```
color: '#234FB6', // color of line
           weight: 3, // width of line
           opacity: 1 // opacity of line
         };
        line = L.polyline(pp, linecss).addTo(map); //add polyline on map
         decorator = L.polylineDecorator(line).addTo(map); //create a polyline decorator
instance.
        //offset and repeat can be each defined as a number, in pixels, or in percentage of the
line's length, as a string
        interval = window.setInterval(function () {
           decorator.setPatterns([{
               offset: offset + '%', //Offset value for first pattern symbol, from the start point of
the line. Default is 0.
               repeat: 0, //repeat pattern at every x offset. 0 means no repeat.
               //Symbol type.
               symbol: L.Symbol.marker({
                  rotate: true, //move marker along the line. false value may cause the custom
marker to shift away from a curved polyline. Default is false.
                  markerOptions: {
                    icon: L.icon({
                      iconUrl: 'images/car.png',
                      iconAnchor: [w / 2, h / 2], //Handles the marker anchor point. For a
correct anchor point [ImageWidth/2,ImageHeight/2]
                      iconSize: [14, 33]
                    })
               })
```

```
]);
          if ((offset += 0.03) > 100) //Sets offset. Smaller the value smoother the movement.
             offset = 0;
        }, 10); //Time in ms. Increases/decreases the speed of the marker movement on
decrement/increment of 1 respectively. values should not be less than 1.
         poly.push(line);
         poly.push(decorator);
         map.fitBounds(line.getBounds());
</script>
2. Google:
<script>
   // This example adds an animated symbol to a polyline.
   function initMap() {
    var map = new google.maps.Map(document.getElementById('map'), {
     center: {lat: 20.291, lng: 153.027},
     zoom: 6,
     mapTypeId: 'terrain'
    });
    // Define the symbol, using one of the predefined paths ('CIRCLE')
    // supplied by the Google Maps JavaScript API.
    var lineSymbol = {
     path: google.maps.SymbolPath.CIRCLE,
     scale: 8,
     strokeColor: '#393'
    };
    // Create the polyline and add the symbol to it via the 'icons' property.
    var line = new google.maps.Polyline({
     path: [{lat: 22.291, lng: 153.027}, {lat: 18.291, lng: 153.027}],
     icons: [{
      icon: lineSymbol,
      offset: '100%'
```

```
}],
   map: map
  });
  animateCircle(line);
 // Use the DOM setInterval() function to change the offset of the symbol
 // at fixed intervals.
 function animateCircle(line) {
   var count = 0;
   window.setInterval(function() {
    count = (count + 1) \% 200;
    var icons = line.get('icons');
    icons[0].offset = (count / 2) + '%';
    line.set('icons', icons);
  }, 20);
</script>
<script async defer
src="https://maps.googleapis.com/maps/api/js?key="YOUR_API_KEY" &callback=initMap">
</script>
```

#### Advance cluster:

#### MapmyIndia:

MapmyIndia Advanced cluster can handle large data sets with impressive performance. Alongside this, the advanced cluster also provides some cool tweaks that can suit your use case scenarios. Let's go through the implementation.

```
<script src="js/PruneCluster.js"></script>
</div id="map-container"></div>
<script>

var map = null;
    var mark = [];
```

```
var markers = [];
      var center = new L.LatLng(28.7041, 77.1025);
      var markerCluster = new PruneClusterForLeaflet(); //Create cluster object.
      var size = 50000; //set no. of markers in cluster.
      window.onload = function () {
         map = new MapmyIndia.Map('map-container', {
           zoomControl: true,
           hybrid: true,
           zoom: 15,
           center: center
        });
        drawCluster(); //call drawCluster()
      };
function drawCluster() {
   markerCluster.Cluster.showExactCount = false; //false will display rounded off marker's
count in cluster, true will display exact count. Default true.
        markerCluster.Cluster.Size = 150; //set the area of cluster. Higher number means more
markers merged. Default 120.
        for (var i = 0; i < size; i++) {
          var marker = new PruneCluster.Marker(center.lat + (Math.random() - 0.5) *
Math.random() * 0.00001 * size,
               center.lng + (Math.random() - 0.5) * Math.random() * 0.00002 * size); //create
prune cluster marker
           marker.data.idx = i; //set custom data in marker.data object;
           markers.push(marker); //add marker in custom array
           markerCluster.RegisterMarker(marker); //Register marker as a member of cluster
group.
         //set custom marker icon
```

# Migration

To MapmyIndia from Google

```
markerCluster.PrepareLeafletMarker = function (marker, data) {
    marker.setlcon(L.icon({
        iconUrl: "images/location.png",
        iconSize: [21, 32]
    }));
    //tooltip on mouseover
    marker.bindTooltip("Marker no. " + data.idx);
};
    map.addLayer(markerCluster);
    //fit map bound
    markerCluster.FitBounds();

    //ProcessView function must be called everytime cluster changes/updates.
    markerCluster.ProcessView();
}
</script>
```