MapmyIndia

Migration Guide Maps-iOS

Table of Contents

Add a Map

MapmyIndia:

```
let mapView = MapmyIndiaMapView(frame: view.bounds)
mapView.autoresizingMask = [.flexibleWidth, .flexibleHeight]
mapView.styleURL = url
view.addSubview(mapView)
```

Apple:

```
let mapView = MKMapView(frame: view.bounds)
mapView.autoresizingMask = [.flexibleWidth, .flexibleHeight]
self.view.addSubview(mapView)
```

Google:

```
let camera = GMSCameraPosition.camera(withLatitude: 28.550667, longitude:
77.268959, zoom: 10)
mapView = GMSMapView.map(withFrame: CGRect.zero, camera: camera)
self.view = mapView
```

Set Map Center

MapmyIndia:

```
mapView.centerCoordinate = CLLocationCoordinate2D(latitude: 28.551438,
longitude: 77.265119)
```

Apple:

```
mapView.centerCoordinate = CLLocationCoordinate2D(latitude: 28.551438,
longitude: 77.265119)
```

Google:

Map center can be set by setting camera property of map instance.

```
mapView.camera = GMSCameraPosition.camera(withTarget:
CLLocationCoordinate2D(latitude: 28.551438, longitude: 77.265119), zoom:
18)
```

Set Zoom Level

MapmyIndia:

```
mapView.zoomLevel = 15
```

Apple:

Zoom can be set by creation a region around a coordinate with distance and set on map:

```
mapView.setRegion(MKCoordinateRegionMakeWithDistance(CLLocationCoordinate2D
(latitude: 28.551438, longitude: 77.265119), 1000, 1000), animated: false)
```

Google:

Zoom level can be set using two ways either using animate function

```
mapView.animate(toZoom: 18)
```

or

update camera property of map, which accepts an instance of GMSCameraPosition.

```
mapView.camera = GMSCameraPosition.camera(withTarget:
CLLocationCoordinate2D(latitude: 28.551438, longitude: 77.265119), zoom:
18)
```

Show User Location and Track

MapmyIndia:

User location on map can be show by setting showsUserLocation property of map instance to true.

```
mapView.showsUserLocation = true
```

And to track user location, use enum property userTrackingMode of map instance whose value can be none, follow, followWithCourse.

```
mapView.userTrackingMode = .follow
```

Apple:

User location on map can be show by setting showsUserLocation property of map instance to true.

```
mapView.showsUserLocation = true
```

And to track user location, use enum property userTrackingMode of map instance whose value can be none, follow, followWithCourse.

```
mapView.userTrackingMode = .follow
```

Google:

User location on map can be show by setting isMyLocationEnabled property of map to true but Your app must prompt the user for consent to use location services. To do this, include the NSLocationWhenInUseUsageDescription key in the Info.plist file for the app, and set the value of each key to a string that describes how the app intends to use location data.

```
mapView.isMyLocationEnabled = true
```

And to track user location, logic need to implement on application side using CLLocationManager as described below.

First create an instance of CLLocationManager in your ViewController at global scope

```
var locationManager = CLLocationManager()
```

And then add below code in viewDidLoad method of ViewController:

```
locationManager.delegate = self
locationManager.requestWhenInUseAuthorization()
locationManager.desiredAccuracy = kCLLocationAccuracyBest
locationManager.startUpdatingLocation()
```

Use below delegateMethod of CLLocationManagerDelegate to continuously change map position on change of user location.

```
func locationManager(_ manager: CLLocationManager, didUpdateLocations
locations: [CLLocation]) {
        let userLocation = locations.last

        let camera = GMSCameraPosition.camera(withLatitude:
userLocation!.coordinate.latitude,
                                              longitude:
userLocation!.coordinate.longitude, zoom: 13.0)
        mapView.camera = camera
    }
```

Add Annotation/Marker

MapmyIndia:

An annotation can be added on map by creating an instance of MGLPointAnnotation and add that in instance of map using method “addAnnotation”.

```
let annotation = MGLPointAnnotation()
annotation.coordinate = CLLocationCoordinate2D(latitude: 28.551438,
longitude: 77.265119)
annotation.title = "Title 1"
annotation.subtitle = "Subtitle 1"
mapView.addAnnotation(annotation)
```

Apple:

An annotation can be added on map by creating an instance of MKPointAnnotation and add that in instance of map using method “addAnnotation”.

```
let annotation = MKPointAnnotation()
annotation.coordinate = CLLocationCoordinate2D(latitude: 28.551438,
longitude: 77.265119)
annotation.title = "Title 1"
annotation.subtitle = "Subtitle 1"
mapView.addAnnotation(annotation)
```

Google:

An annotation can be added on map by creating an instance of GMSMarker and add that in instance of map by setting marker’s map property to map instance.

```
let marker = GMSMarker()
marker.position = CLLocationCoordinate2D(latitude: 28.551438, longitude:
77.265119)
marker.title = "Title 1"
marker.snippet = "Subtitle 1"
marker.map = mapView
```

Custom Annotation/Marker

MapmyIndia:

To change image or view for default marker use delegate methods of protocol MGLMapViewDelegate.

Either you can override whole view of marker by using below method of delegate:

```
func mapView(_ mapView: MGLMapView, viewFor annotation: MGLAnnotation) ->
MGLAnnotationView? {
            let annotationView = MGLAnnotationView()
            annotationView.bounds = CGRect(x: 0, y: 0, width: 20, height:
20)
            annotationView.backgroundColor = UIColor.blue
            return annotationView
    }
```

or you can override image of marker by using below method of delegate:

```
func mapView(_ mapView: MGLMapView, imageFor annotation: MGLAnnotation) ->
MGLAnnotationImage? {
        let reuseIdentifier = "iconImage"
        var annotationImage =
mapView.dequeueReusableAnnotationImage(withIdentifier: reuseIdentifier)

        if annotationImage == nil {
            let image = UIImage(named: "MapMarker")!
            annotationImage = MGLAnnotationImage(image: image,
reuseIdentifier: reuseIdentifier)
        }

        return annotationImage
    }
```

Note: Use above methods according to your requirement If you want to change image or view on condition basis.

Apple:

To change view for default marker use delegate methods of protocol MKMapViewDelegate.

```
func mapView(_ mapView: MKMapView, viewFor annotation: MKAnnotation) ->
MKAnnotationView? {
        let reuseIdentifier = "pin"
        var annotationView =
mapView.dequeueReusableAnnotationView(withIdentifier: reuseIdentifier)

        if annotationView == nil {
            annotationView = MKAnnotationView(annotation: annotation,
reuseIdentifier: reuseIdentifier)
            annotationView?.canShowCallout = true
        } else {
```

```
            annotationView?.annotation = annotation
        }
        annotationView?.image = UIImage(named: "ActiveMapMarker")

        return annotationView
    }
```

Google:

To change default icon of marker set an image to its icon property.

```
let marker = GMSMarker()
marker.position = CLLocationCoordinate2D(latitude: 28.551438, longitude:
77.265119)
marker.title = "Title 1"
marker.snippet = "Subtitle 1"
marker.map = mapView
markerSquirt.icon = UIImage(named: “ActiveMapMarker")
```

You can also change entire view of marker by setting its iconView property.

```
markerSquirt.iconView = UIImageView(image: UIImage(named:
"ActiveMapMarker"))
```

Show Info Window/Callout

MapmyIndia:

To enable/disable Info window, use below delegate method of MGLMapViewDelegate protocol which returns a boolean value.

```
func mapView(_ mapView: MGLMapView, annotationCanShowCallout annotation:
MGLAnnotation) -> Bool {
        return true
    }
```

Apple:

To enable/disable Info window, First Annotation view has to be overridden as shown in example for Custom Annotation above and use property canShowCallout to enable or disable.

```
annotationView.canShowCallout = true
```

Google:

To enable/disable Info window, use below delegate method of GMSMapViewDelegate protocol. Return true if this delegate handled the tap event, which prevents the map from performing its default selection behavior, and NO if the map should continue with its default selection behavior.

```
func mapView(_ mapView: GMSMapView, didTap marker: GMSMarker) -> Bool {
        return true
    }
```

Custom Info Window/Callout

MapmyIndia:

To change default view for callout of annotation use delegate method of MGLMapViewDelegate protocol and return custom object of MGLCalloutView. To create custom MGLCalloutView create a class which is extended from MGLCalloutView and UIView classes.

```
func mapView(_ mapView: MGLMapView, calloutViewFor annotation:
MGLAnnotation) -> MGLCalloutView? {
      return CustomCalloutView(representedObject: annotation)
    }
```

Apple:

To change default view for callout of annotation use delegate method of MKMapViewDelegate protocol and rand return custom object of MKAnnotationView. To create custom object create a class which is extended from MKAnnotationView.

```
func mapView(_ mapView: MKMapView, viewFor annotation: MKAnnotation) ->
MKAnnotationView? {
      if annotation is MKUserLocation { return nil }
      var annotationView =
mapView.dequeueReusableAnnotationView(withIdentifier:
kPersonWishListAnnotationName)
      if annotationView == nil {
      annotationView = PersonWishListAnnotationView(annotation: annotation,
reuseIdentifier:  kPersonWishListAnnotationName)
      (annotationView as!
PersonWishListAnnotationView).personDetailDelegate = self
      } else {
      annotationView!.annotation = annotation
      }
      return annotationView
}
```

A sample to described this can be found here. And full sample can be downloaded here.

Google:

To change default view for callout of annotation use delegate method of GMSMapViewDelegate protocol and return custom object of UIView.

```
func mapView(_ mapView: GMSMapView, markerInfoWindow marker: GMSMarker) ->
UIView? {
        let view = UIView(frame: CGRect.init(x: 0, y: 0, width: 200,
height: 70))
        view.backgroundColor = UIColor.white
        view.layer.cornerRadius = 6

        let label1 = UILabel(frame: CGRect.init(x: 8, y: 8, width:
view.frame.size.width - 16, height: 15))
```

```
        label1.text = "Hi there!"
        view.addSubview(label1)

        let label2 = UILabel(frame: CGRect.init(x: label1.frame.origin.x,
y: label1.frame.origin.y + label2.frame.size.height + 3, width:
view.frame.size.width - 16, height: 15))
        label2.text = "I am a custom info window."
        label2.font = UIFont.systemFont(ofSize: 14, weight:
UIFontWeightLight)
        view.addSubview(label2)

        return view
    }
```

Add a Polyline

MapmyIndia:

To show a polyline on map create an instance of MGLPolyline and add that object to instance of map using method “addAnnotation”.

```
var coordinates = [
                CLLocationCoordinate2D(latitude: 28.550834, longitude:
77.268918),
                CLLocationCoordinate2D(latitude: 28.551059, longitude:
77.268890),
                CLLocationCoordinate2D(latitude: 28.550938, longitude:
77.267641),
                CLLocationCoordinate2D(latitude: 28.551764, longitude:
77.267575),
                CLLocationCoordinate2D(latitude: 28.552068, longitude:
77.267599),
                CLLocationCoordinate2D(latitude: 28.553836, longitude:
77.267450),
                ]
            let polyline = MGLPolyline(coordinates: &coordinates, count:
UInt(coordinates.count))
            mapView.addAnnotation(polyline)
```

After adding polyline it can be styled using different method of delegate MGLMapViewDelegate as described below.

```
func mapView(_ mapView: MGLMapView, strokeColorForShapeAnnotation
annotation: MGLShape) -> UIColor {
        if annotation is MGLPolyline {
            return .red
        }
        return mapView.tintColor
    }

func mapView(_ mapView: MGLMapView, lineWidthForPolylineAnnotation
annotation: MGLPolyline) -> CGFloat {
        return 10.0
    }
func mapView(_ mapView: MGLMapView, alphaForShapeAnnotation annotation:
MGLShape) -> CGFloat {
        if annotation is MGLPolyline {
            return 0.5
        }
        return 1.0
    }
```

Apple:

To show a polyline on map create an instance of MKPolyline and add that object to instance of map using method “add”.

```
let coordinates = [
```

```
                CLLocationCoordinate2D(latitude: 28.550834, longitude:
77.268918),
                CLLocationCoordinate2D(latitude: 28.551059, longitude:
77.268890),
                CLLocationCoordinate2D(latitude: 28.550938, longitude:
77.267641),
                CLLocationCoordinate2D(latitude: 28.551764, longitude:
77.267575),
                CLLocationCoordinate2D(latitude: 28.552068, longitude:
77.267599),
                CLLocationCoordinate2D(latitude: 28.553836, longitude:
77.267450),
                ]
            let polyline = MKPolyline(coordinates: coordinates, count:
coordinates.count)

            mapView.add(polyline)
```

After adding polyline its renderer has to be handled in delegate method of MKMapViewDelegate. Where you can also style that according to requirement.

```
func mapView(_ mapView: MKMapView, rendererFor overlay: MKOverlay) ->
MKOverlayRenderer {
        if overlay is MKPolyline {
            let polylineRenderer = MKPolylineRenderer(overlay: overlay)
            polylineRenderer.strokeColor =
UIColor.red.withAlphaComponent(0.5)
            polylineRenderer.lineWidth = 5
            return polylineRenderer
        }

        return MKOverlayRenderer(overlay: overlay)
    }
```

Google:

To show a polyline on map create an instance of GMSPolyline by passing an instance of GMSMutablePath and add that in instance of map by setting marker's map property to map instance.

```
let path = GMSMutablePath()
path.add(CLLocationCoordinate2D(latitude: 28.550834, longitude: 77.268918))
path.add(CLLocationCoordinate2D(latitude: 28.551059, longitude: 77.268890))
path.add(CLLocationCoordinate2D(latitude: 28.550938, longitude: 77.267641))
path.add(CLLocationCoordinate2D(latitude: 28.551764, longitude: 77.267575))
path.add(CLLocationCoordinate2D(latitude: 28.552068, longitude: 77.267599))
path.add(CLLocationCoordinate2D(latitude: 28.553836, longitude: 77.267450))
let rectangle = GMSPolyline(path: path)
rectangle.strokeColor = UIColor.red.withAlphaComponent(0.5)
rectangle.strokeWidth = 5
rectangle.map = mapView
```

Add a Polygon

MapmyIndia:

To show a polygon on map create an instance of MGLPolygon and add that object to instance of map using method “addAnnotation”.

```
let coordinates = [
                CLLocationCoordinate2D(latitude: 28.550834, longitude:
                    77.268918),
                CLLocationCoordinate2D(latitude: 28.551059, longitude:
                    77.268890),
                CLLocationCoordinate2D(latitude: 28.550938, longitude:
                    77.267641),
                CLLocationCoordinate2D(latitude: 28.551764, longitude:
                    77.267575),
                CLLocationCoordinate2D(latitude: 28.552068, longitude:
                    77.267599),
                CLLocationCoordinate2D(latitude: 28.553836, longitude:
                    77.267450),
                ]
            let polygon = MGLPolygon(coordinates: &coordinates, count:
                UInt(coordinates.count))
            mapView.addAnnotation(polygon)
```

After adding polygon it can be styled using different method of delegate MGLMapViewDelegate as described below.

```
func mapView(_ mapView: MGLMapView, strokeColorForShapeAnnotation
annotation: MGLShape) -> UIColor {
        // Give our polyline a unique color by checking
        if annotation is MGLPolygon {
            return .green
        }
        return mapView.tintColor
    }

    func mapView(_ mapView: MGLMapView, alphaForShapeAnnotation annotation:
MGLShape) -> CGFloat {
        // Set the alpha for all shape annotations to 1 (full opacity)
        if annotation is MGLPolygon {
            return 0.8
        }
        return 1.0
    }
```

Apple:

After adding polygon its renderer has to be handled in delegate method of MKMapViewDelegate. Where you can also style that according to requirement.

```
let coordinates = [
                CLLocationCoordinate2D(latitude: 28.550834, longitude:
77.268918),
```

```
                CLLocationCoordinate2D(latitude: 28.551059, longitude:
77.268890),
                CLLocationCoordinate2D(latitude: 28.550938, longitude:
77.267641),
                CLLocationCoordinate2D(latitude: 28.551764, longitude:
77.267575),
                CLLocationCoordinate2D(latitude: 28.552068, longitude:
77.267599),
                CLLocationCoordinate2D(latitude: 28.553836, longitude:
77.267450),
                ]
            let polygon = MKPolygon(coordinates: coordinates, count:
coordinates.count)

            mapView.add(polygon)
```

After adding polygon it can be styled using delegate method MKMapViewDelegate as described below.

```
func mapView(_ mapView: MKMapView, rendererFor overlay: MKOverlay) ->
MKOverlayRenderer {
        if overlay is MKPolygon {
            let polygonRenderer = MKPolygonRenderer(overlay: overlay)
            polygonRenderer.strokeColor =
UIColor.green.withAlphaComponent(0.8)
            polygonRenderer.fillColor =
UIColor.gray.withAlphaComponent(0.8)
            polygonRenderer.lineWidth = 10
            return polygonRenderer
        }
        return MKOverlayRenderer(overlay: overlay)
    }
```

Google:

To show a polyline on map create an instance of GMSPolygon by passing an instance of GMSMutablePath and add that in instance of map by setting marker's map property to map instance.

```
let path = GMSMutablePath()
path.add(CLLocationCoordinate2D(latitude: 28.550834, longitude: 77.268918))
path.add(CLLocationCoordinate2D(latitude: 28.551059, longitude: 77.268890))
path.add(CLLocationCoordinate2D(latitude: 28.550938, longitude: 77.267641))
path.add(CLLocationCoordinate2D(latitude: 28.551764, longitude: 77.267575))
path.add(CLLocationCoordinate2D(latitude: 28.552068, longitude: 77.267599))
path.add(CLLocationCoordinate2D(latitude: 28.553836, longitude: 77.267450))
let rectangle = GMSPolygon(path: path)
rectangle.strokeColor = UIColor.red.withAlphaComponent(0.5)
rectangle.strokeWidth = 5
rectangle.fillColor = UIColor.gray.withAlphaComponent(0.8)
rectangle.map = mapView
```

Remove Annotations/Polyline/Polygon

MapmyIndia:

To remove a particular marker or shape from map use “removeAnnotation” method of map instance.

```
mapView.removeAnnotation(shape)
```

and to remove many markers or shapes from map use “removeAnnotations”

```
mapView.removeAnnotations(shape)
```

Apple:

To remove a particular marker or shape from map use “removeAnnotation” method of map instance.

```
mapView.removeAnnotation(shape)
```

and to remove many markers or shapes from map use “removeAnnotations”

```
mapView.removeAnnotations(shape)
```

Google:

To remove a particular marker from marker set its map property to nil.

```
marker.map = nil
```

To Clear all annotations and other shape from map use clear function of map instance.

```
mapView.clear()
```