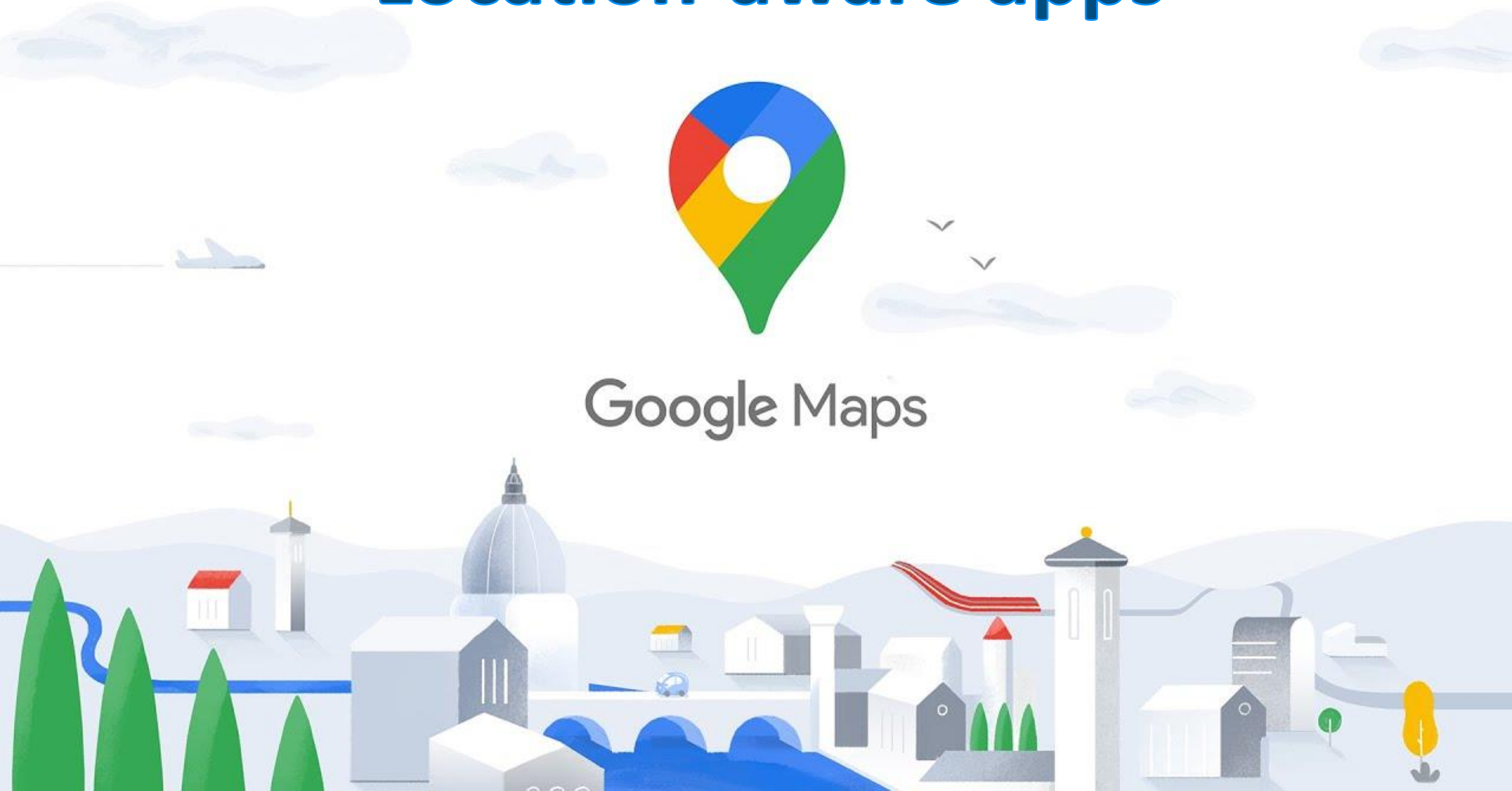
Camera, Google Maps, and Location-aware apps



Access Image Gallery and Camera from Android App

< Back

registerForActivityResult

- `registerForActivityResult` can be used to **launch** an activity from another app (e.g., launch take photo activity from the camera app), then register a callback to **handle the result** once it is dispatched by Android OS
    - `registerForActivityResult` takes an `ActivityResultContract` and an `Callback` and returns an `ActivityResultLauncher` which you'll use to launch the desired activity
    - Android offers many built-in contracts such as TakePicture, TakeVideo
    - Your app can start the camera app and receive the captured photo as a result using `registerForActivityResult(ActivityResultContracts.TakePicture())`
- **`registerForActivityResult`** required these dependencies:

```
// Kotlin extensions - activity-ktx & fragment-ktx
implementation "androidx.activity:activity-ktx:1.2.0-beta01"
implementation "androidx.fragment:fragment-ktx:1.3.0-beta01"
```

Take Picture

- Create **ActivityResultLauncher** using **registerForActivityResult** to **launch** the camera app, then register a callback to **handle** the taken image once the camera app is closed
  - The image taken is available at the location assigned to the `photoUri` variable

```
private lateinit var photoUri : Uri
takePhotoBtn.setOnClickListener {
    val takePicture = registerForActivityResult(ActivityResultContracts.TakePicture()) { isSuccess ->
        if (isSuccess) {
            println("Debug: taken photo location = $photoUri")
        }
    }

    try {
        val photoFile = createPhotoFile()
        val authority = BuildConfig.APPLICATION_ID + ".fileProvider"
        photoUri = FileProvider.getUriForFile(requireContext(), authority, photoFile);
        takePicture.launch(photoUri)
    } catch (e: Exception) {
        println("Debug: $e")
    }
}
```

Select photo/video from Gallery

- Create **ActivityResultLauncher** using **`registerForActivityResult`** to **launch** the gallery app to allow the user to select a photo/video, then register a callback to **handle** the selected media
    - The location of the selected image is available as a `uri` parameter accessible to the callback function

```
selectMediaBtn.setOnClickListener {
    val mediaPicker =
        registerForActivityResult(ActivityResultContracts.GetContent()) { uri ->
            println("Debug: Uri of selected media: $uri")
        }
    mediaPicker.launch("*/*")
}
```

Google Maps Platform

Google
Maps

< Back

Google Maps Platform Key Services

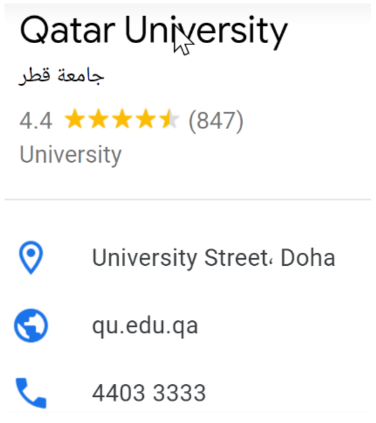
- **Maps**:
  - Apps can integrate customized and interactive maps, satellite imagery and Street View imagery

Qatar University
B-13 قطر
College of Engineering - Qatar University
كلية الهندسة - جامعة قطر
Lat: 25.37499, Long: 51.49066
Admission
Registration...
Pizza
Tennis Co
College of Engineering - Qatar University
كلية الهندسة - جامعة قطر

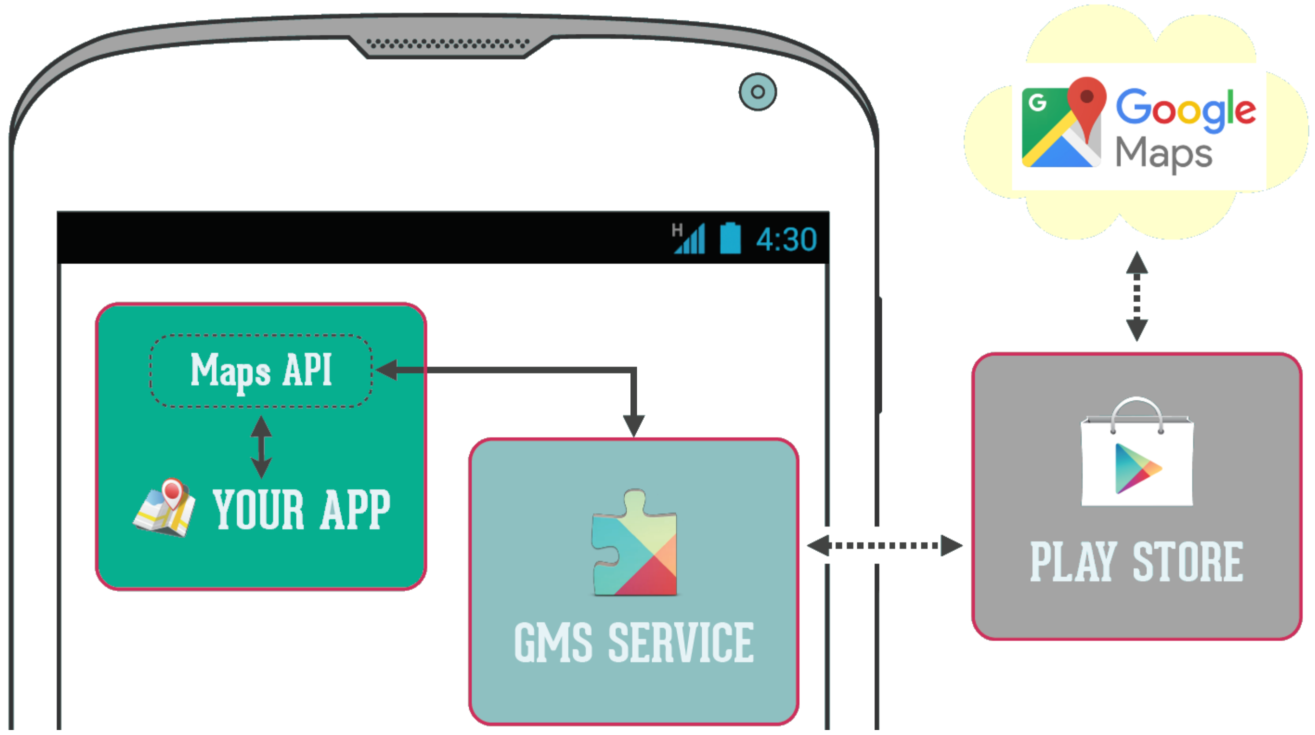
- **Routes**:
  - Allow users to find the best <u>route</u> to get from A to Z using public transport, biking, driving, or walking.
  - Compute travel times and distances
  - Real-time traffic updates about the selected route

17 minutes
38 minutes

- **Places**:
  - Users can search details about million **points of interest** around the world including place names, addresses, images, contact information and reviews.


Qatar University
جامعة قطر
4.4 ★★★★★ (847)
University
University Street، Doha
qu.edu.qa
4403 3333

Google Mobile Services (GMS)



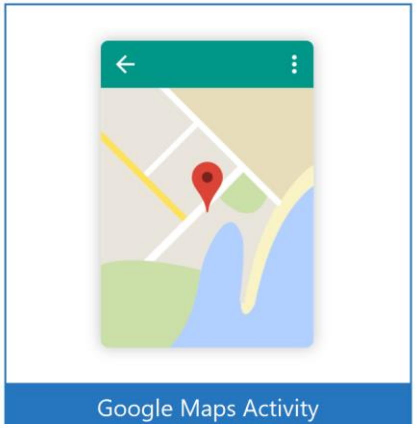
- Add these dependences to build.gradle

```
implementation 'com.google.android.gms:play-services-maps:17.0.0'
implementation 'com.google.android.gms:play-services-location:17.1.0'
// KTX for the Maps SDK for Android library
implementation 'com.google.maps.android:maps-ktx:2.2.0'
```

Typical Features in Location-aware App

- Visualise data in a **custom map**
- Get the device **geolocation** (latitude & longitude)
- Finding the geolocation of an address, known ad **Geocoding**
- Finding the address of a geolocation: **Reverse Geocoding**
- **Location tracking** as the user moves
- Getting alerts if user is in area of interest: **Geofencing**
- Tracking User Activity (e.g., walking, running, driving, etc.)

Add a Google Maps Activity



- Add **Google Maps** activity. This adds a layout having **SupportMapFragment** to display the map

```
<fragment
    android:id="@+id/mapFragment"
    android:name="com.google.android.gms.maps.SupportMapFragment" />
```

- It also adds file "res/values/**google_maps_api.xml**" that has the API key for Google Maps
  - This file has a Web link to allow generating an API key for Google Maps.
  - Paste the generate key in **google_maps_api.xml**

```
<string name="google_maps_key">AIzaSyC6f0s...</string>
```

Customize Map

Qatar University
B-13
Admission
Registration...
College of Engineering - Qatar University
كلية الهندسة - جامعة قطر
Lat: 25.37499, Long: 51.49066
Tennis Co
Pizza
ersity

- In the activity onCreate you can obtain the **mapFragment** from the activity layout

Await for the map to be ready then **customize** it such as:

  - Add marker
  - Add overlay (e.g., image over the map)
  - Change the zoom level
  - Handle events such as Point of Interest (PoI) click event

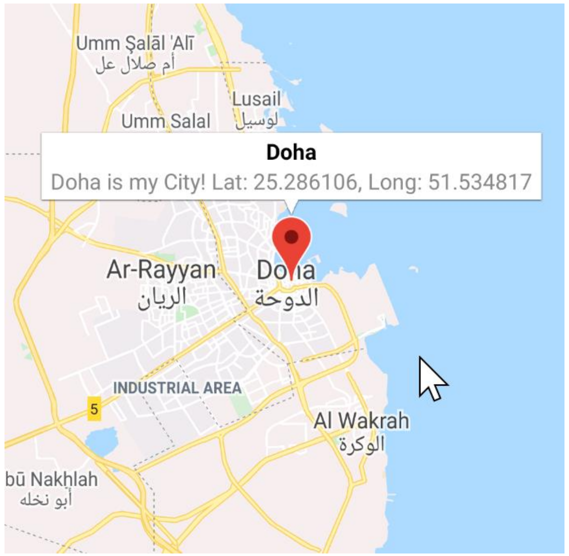
```
override fun onCreate(savedInstanceState: Bundle?) {
    ...
    // Obtain the mapFragment from the activity layout
    val mapFragment =
        supportFragmentManager.findFragmentById(R.id.mapFragment) as SupportMapFragment

    lifecycle.coroutineScope.launchWhenCreated {
        // Await for the map to be ready then customize it
        val googleMap = mapFragment.awaitMap()
        onMapReady(googleMap)
    }
}
```

Add Marker

- Marker identify a location on the map at a particular geo coordinates
  - When the marker is clicked an **info window** displays the marker's title and snippet text

```
val homeLatLng = LatLng(25.286106, 51.534817)
val markerTitle = "Doha"
// A Snippet is Additional text that's displayed below the title
val snippetText = "$markerTitle is my City! Lat: ${homeLatLng.latitude}, Long: ${homeLatLng.longitude}"
```

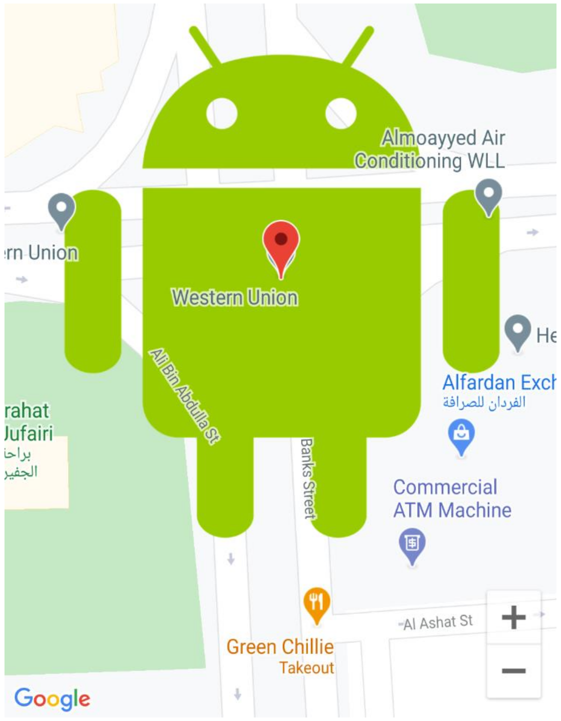
```
map.addMarker {
     position(homeLatLng)
     title(markerTitle)
     snippet(snippetText)
}
```


Doha
Doha is my City! Lat: 25.286106, Long: 51.534817

Add Overlay

- A ground **overlay** is an image that is displayed over the map at a particular geo coordinates
    - Unlike markers, ground overlays **size** and **orientation** changes when rotating, tilting or zooming the map

```
map.addGroundOverlay {
    position(homeLatLng, overlaySize)
    image(BitmapDescriptorFactory
        .fromResource(R.drawable.android))
}
```

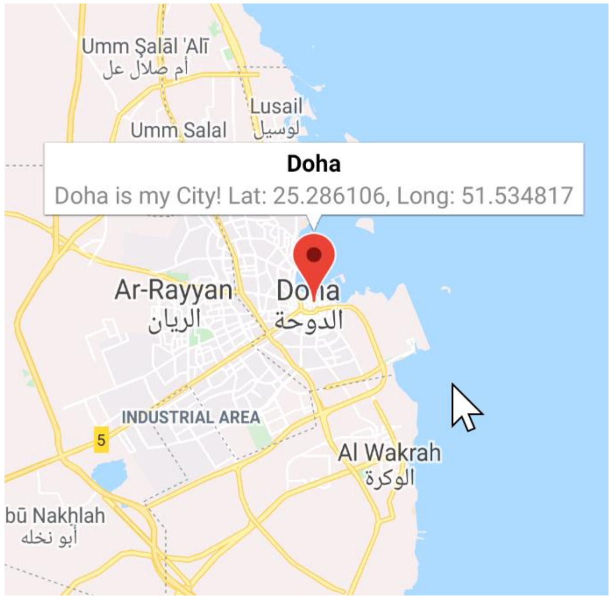


Map Zoom Level

- Move the map view to a particular **geo coordinates** and change the zoom level

```
map.moveCamera(
          CameraUpdateFactory.newLatLngZoom(latLng, zoomLevel))
```

- Zoom level values:
  - 1: World
  - 5: Continent
  - 10: City
  - 15: Streets
  - 20: Buildings



Doha
Doha is my City! Lat: 25.286106, Long: 51.534817

Zoom level 10

Other Map Customization

- Show the zoom controls

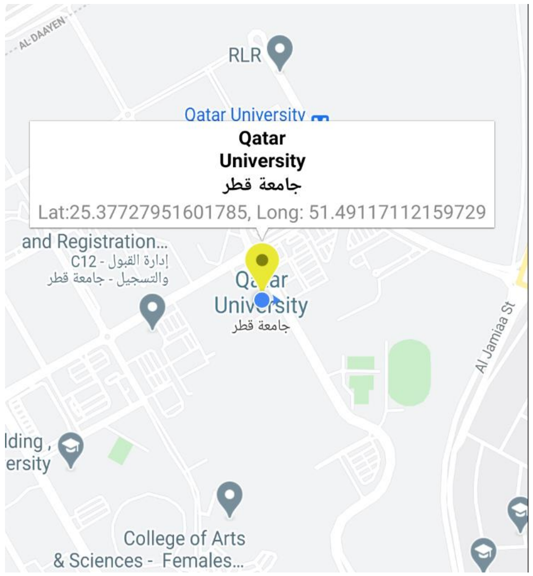
```
map.uiSettings.isZoomControlsEnabled = true
```

- Set the map type

```
map.mapType = GoogleMap.MAP_TYPE_NORMAL       // OR

map.mapType = GoogleMap.MAP_TYPE_HYBRID       // OR

map.mapType = GoogleMap.MAP_TYPE_SATELLITE    // OR

map.mapType = GoogleMap.MAP_TYPE_TERRAIN
```

Handle Point of Interest (PoI) click event

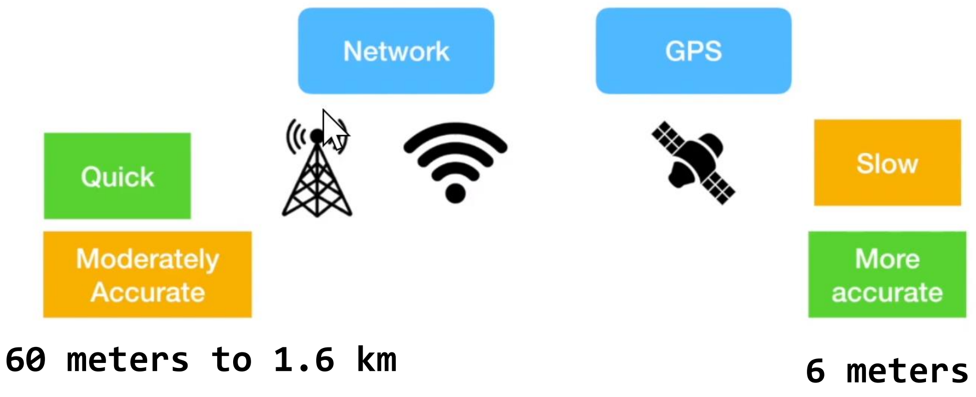
- If you want to respond to a user tapping on a PoI, you can use `map.setOnPoiClickListener`
  - poi parameter has the **placeId**, **name** and **geo coordinates** (i.e., `latitude` & `longitude`)


RLR
Qatar University
Qatar University جامعة قطر
Lat:25.37727951601785, Long: 51.49117112159729
and Registration... C12 - إدارة القبول والتسجيل - جامعة قطر
Qatar University جامعة قطر
Al Jamiaa St
College of Arts & Sciences - Females...

```
map.setOnPoiClickListener { poi ->
    println(">> Debug. Clicked PoI placeId: ${poi.placeId}. Name: ${poi.name}")
    // A Snippet is Additional text that's displayed below the title.
    val snippet = "Lat:${poi.latLng.latitude}, Long: ${poi.latLng.longitude}"

    poiMarker = map.addMarker {
        position(poi.latLng)
        title(poi.name)
        snippet(snippet)
        icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_YELLOW))
    }
    poiMarker.showInfoWindow()
}
```

Get User Location

- Request last known location of the user's device
  - Location is determined by the **LocationServices** using WiFi & Cellular Tower and/or GPS (Global Positioning System)



```
val fusedLocationClient =
    LocationServices.getFusedLocationProviderClient(appContext)
val lastLocation = fusedLocationClient.lastLocation.await()
lastLocation?.let {
        val currentLocation = "Lat: ${it.latitude} & Long: ${it.longitude}"
        println(">> Debug: $currentLocation")
    }
}
```

Request location updates

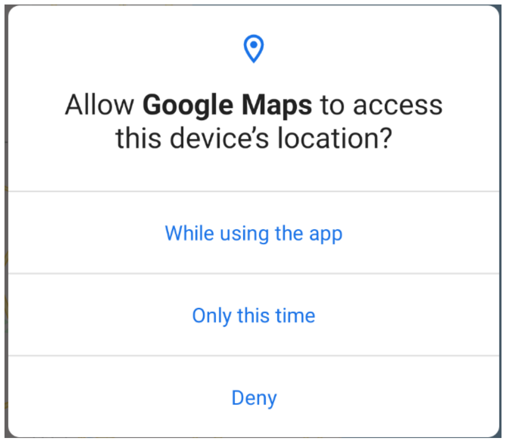
- To get the location (latitude and longitude) of the device at regular intervals you can use `fusedLocationClient.requestLocationUpdates`
    - The location provider invokes the `LocationCallback.onLocationResult()` on a regular interval. The incoming argument contains a list `Location` object containing the location's latitude and longitude

```
val locationRequest: LocationRequest = LocationRequest.create().apply {
     interval = 10000
     fastestInterval = 5000
     priority = LocationRequest.PRIORITY_HIGH_ACCURACY
}
fusedLocationClient.requestLocationUpdates(
     locationRequest, locationCallback, null)

private val locationCallback = object : LocationCallback() {
    override fun onLocationResult(locationResult: LocationResult?) {
        locationResult ?: return
        locationResult.locations.forEach {
                val currentLocation = "Lat: ${it.latitude} & Long: ${it.longitude}"
                println(">> Debug: $currentLocation")
            }
        }}}
```

Request Location Permission

- At runtime must ask for the permission to access the device's location using `registerForActivityResult( ActivityResultContracts.RequestPermission())`

```
// Register request permission callback, which handles the user's response to the system permission dialog
private val requestPermissionLauncher = registerForActivityResult(
           ActivityResultContracts.RequestPermission())
    // Callback for the result from requesting permission
    { isGranted: Boolean ->
        if (isGranted) {
            // Permission is granted. Enable My Location button on the map
            enableMyLocation()
        }
    }
...
// Ask for the permission to access the user’s device location
// The registered ActivityResultCallback gets the result of this request.
requestPermissionLauncher.launch(Manifest.permission.ACCESS_FINE_LOCATION)
```


Allow **Google Maps** to access this device’s location?

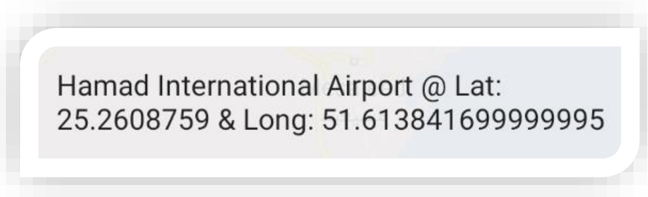
While using the app

Only this time

Deny

Geocoding

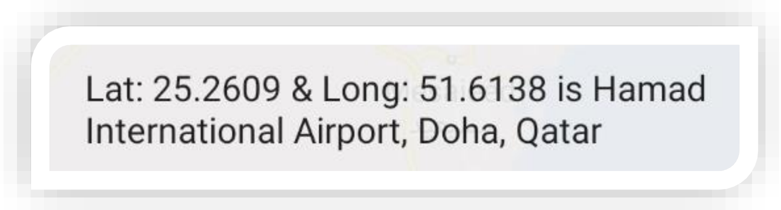
- **Geocoding** is the process of converting an address (e.g., location name or a street address) into geographic coordinates (lat, lng), which you can use to place markers on a map, or position the map


Hamad International Airport @ Lat: 25.2608759 & Long: 51.613841699999995

```kotlin
/*
    Geocoding = converting an address or location name (like a street address) into
    geographic coordinates (lat, lng)
*/
 private fun getGeoCoordinates(locationAddress: String): GeoLocation? {
     val geocoder = Geocoder(this)
     val coordinates = geocoder.getFromLocationName(locationAddress, 1)
     return if (coordinates != null && coordinates.size > 0) {
          val latitude = coordinates[0].latitude
          val longitude = coordinates[0].longitude
          GeoLocation(latitude, longitude)
     } else {
          null
     }
 }
```

Reverse Geocoding

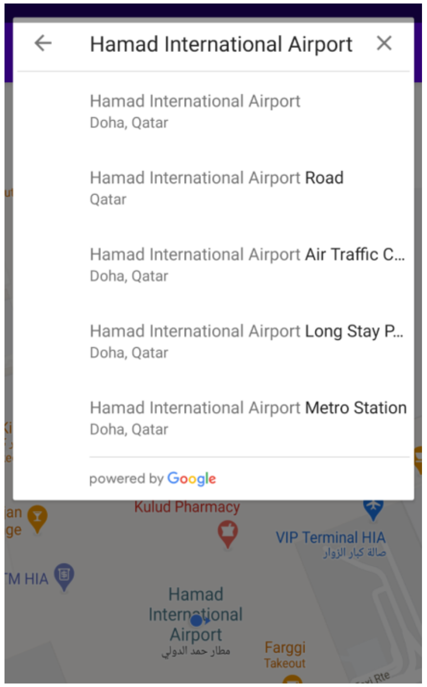
- **Reverse geocoding** is the process of converting geographic coordinates (lat, lng) into a human-readable address


Lat: 25.2609 & Long: 51.6138 is Hamad International Airport, Doha, Qatar

```
/*
    Reverse geocoding = converting geographic coordinates (lat, lng)
    into a human-readable address
*/
private fun getAddress(lat: Double, lng: Double): String {
    val geocoder = Geocoder(this)
    val addresses = geocoder.getFromLocation(lat, lng, 1)

    return if (addresses!= null && addresses.size > 0) {
        val address = addresses[0]?.getAddressLine(0) ?: ""
        //val city = addresses[0]?.locality ?: ""
        //val country = addresses[0]?.countryName ?: ""
        address
    } else {
        ""
    }
}
```

Places AutoComplete Search Box

- The AutoComplete Places Search Box returns place predictions in response to user search queries.
  - As the user types, the autocomplete service returns suggestions for places such as businesses, addresses, and points of interest.



Hamad International Airport

Hamad International Airport
Doha, Qatar

Hamad International Airport Road
Qatar

Hamad International Airport Air Traffic C...
Doha, Qatar

Hamad International Airport Long Stay P...
Doha, Qatar

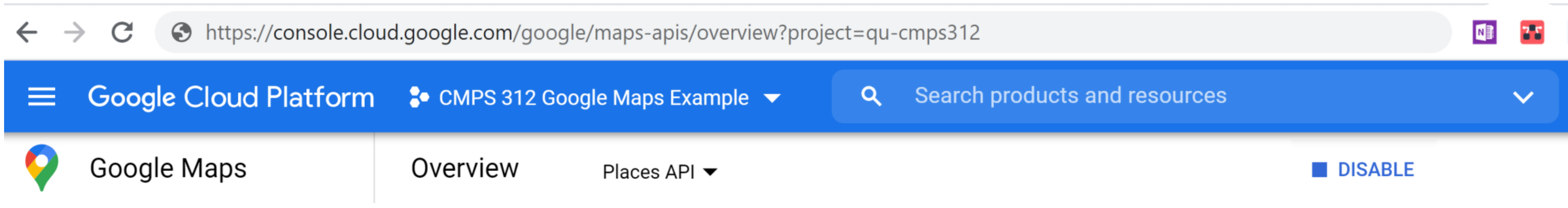
Hamad International Airport Metro Station
Doha, Qatar

powered by Google

```kotlin
private fun initPlacesAutocomplete() {
    if (!Places.isInitialized()) {
        val apiKey = getString(R.string.google_maps_key)
        Places.initialize(applicationContext, apiKey, Locale.US);
    }
    // Initialize the AutocompleteSupportFragment.
    val autocompleteFragment = supportFragmentManager.findFragmentById(R.id.autocompleteFragment)
            as AutocompleteSupportFragment

    // Specify the types of place data to return.
    autocompleteFragment.setPlaceFields(listOf(Place.Field.ID, Place.Field.NAME, Place.Field.LAT_LNG))

    // Set up a PlaceSelectionListener to handle the response.
    autocompleteFragment.setOnPlaceSelectedListener(object : PlaceSelectionListener {
        override fun onPlaceSelected(place: Place) {
            // Zoom the map to the location of the selected place
            map.moveCamera(CameraUpdateFactory.newLatLngZoom(place.latLng, 15F))
            println(">> Debug - Place: ${place.name}, ${place.id}")
        }

        override fun onError(status: Status) {
            println(">> Debug - An error occurred: $status")
        }
    })
}
```

Places API

- Need to enable billing and provide a Credit Card (will not be charged unless you enable auto-billing)
  - At https://console.cloud.google.com/google/maps-apis/start
  - For demo and testing the free $200 per month could be enough
- Need to enable the places API to be able to use it in your app


https://console.cloud.google.com/google/maps-apis/overview?project=qu-cmps312
Google Cloud Platform
CMPS 312 Google Maps Example
Search products and resources
Google Maps
Overview
Places API
DISABLE

Resources

- Android Google Maps Codelab
    - https://codelabs.developers.google.com/codelabs/advanced-android-kotlin-training-maps
- Google Maps Android samples
    - https://github.com/googlemaps/android-samples
- Receive location updates in Android with Kotlin Codelab
    - https://codelabs.developers.google.com/codelabs/while-in-use-location/
- Adding geofencing to your map Codelab
    - https://developer.android.com/codelabs/advanced-android-kotlin-training-geofencing