5COSC023W - MOBILE APPLICATION DEVELOPMENT
Lecture 9: Maps, Location and Runtime Permissions

Dr Dimitris C. Dracopoulos

Get the Last Known Location

Steps (set up):

1. Create a new Android Studio project using the "*Google Maps Activity*" template.
2. Obtain an Google Maps API (for usage with Maps) and insert it in the `res/values/` (follow the instructions in the same file for how to obtain the API key).
3. Add a **dependency** of Google Play location services by adding the following line in the `build.gradle` module file (where `XX.X.X` is the latest version for Google Play Services:

```
implementation 'com.google.android.gms:play-services-location:XX.X.X'
```

4. Add the `ACCESS_FINE_LOCATION` and `ACCESS_COARSE_LOCATION` permissions in the manifest file of the project:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
```

Get the Last Known Location (cont'ed)

Steps (Kotlin code):

1. Check if the permission is granted by the user, otherwise request the permission by calling `ActivityCompat.requestPermissions`.
2. Implement in your activity the `onRequestPemissionsResult()` callback method which will receive the permissions result.
3. Create a `FusedLocationProviderClient` object:

```
mFusedLocationClient =
        LocationServices.getFusedLocationProviderClient(this);
```

4. Call `getLastLocation()` (or access the `lastLocation` property) on the `FusedLocationProviderClient` object returning a `Task` object.
5. Call `addOnSuccessListener()` method on the task and pass it an object which implements the `OnSuccessListener<Location>` interface.

How to Receive Location Updates

Steps (Kotlin code):

1. Create a `LocationRequest` object containing the requirements of the request (update frequency, accuracy).
2. Create a `LocationCallback` as part of the activity and override its `onLocationResult()` method which is called periodically with the location updates.
3. Call `requestLocationUpdates()` on the `FusedLocationProviderClient` object and pass it the `LocationRequest` and the `LocationCallBack` objects.

The Location and Maps Code

The code for the maps and location application developed in the lecture can be found in the following link:

`https://dracopd.users.ecs.westminster.ac.uk/DOCUM/courses/5cosc023w/location_maps_app.zip`

The Location and Maps Application

The layout file `activity_main.xml` of the main activity:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:textSize="24sp"
        android:hint="Location"/>
```

The layout of the main activity (cont'ed)

```
    <Button
        android:id="@+id/bt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Get Location"
        android:textSize="24sp"/>
    <Button
        android:id="@+id/bt2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Show Map"
        android:textSize="24sp"/>
</LinearLayout>
```

The layout of the maps activity

```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MapsActivity" />
```

The Main activity

```
package com.example.mapstest


import android.content.Intent
import android.content.pm.PackageManager
import android.location.Address
import android.location.Geocoder
import android.location.Location
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Looper
import android.widget.Button
import android.widget.TextView
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.android.gms.location.*
import java.util.*

class MainActivity : AppCompatActivity() {
    lateinit var tv: TextView
    lateinit var fusedLocationProviderClient: FusedLocationProviderClient
    var lastLocation: Location? = null

    lateinit var locationCallback: LocationCallback
```

The Main activity (cont'ed)

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    tv = findViewById(R.id.tv)
    val bt = findViewById<Button>(R.id.bt)
    bt.setOnClickListener {
        getLocation()
    }

    val sMap = findViewById<Button>(R.id.bt2)
    sMap.setOnClickListener {
        showMap()
    }

    fusedLocationProviderClient =
            LocationServices.getFusedLocationProviderClient(this)

    locationCallback = object: LocationCallback() {
        override fun onLocationResult(p0: LocationResult) {
            super.onLocationResult(p0)
            lastLocation = p0.lastLocation
            displayLastLocation()
        }
    }
}
```

The Main activity (cont'ed)

```
fun displayLastLocation() {
    if (lastLocation != null) {
        var address: Address? = null

        var geocoder: Geocoder = Geocoder(this, Locale.getDefault())
        var addresses: List<Address> =
            geocoder.getFromLocation(lastLocation!!.latitude,
                                     lastLocation!!.longitude,
                                     1) as List<Address>

        if (addresses != null && addresses.size > 0) {
            address = addresses[0]
        }

        tv.setText("Latitude: " + lastLocation!!.latitude + ",
                                  Longitude: " + lastLocation!!.longitude)
        if (address != null)
            tv.append(address.toString())
    }
    else
        tv.setText("Location is not available")
}
```

The Main activity (cont'ed)

```
fun getLocation() {
  if (ContextCompat.checkSelfPermission(this,
                  android.Manifest.permission.ACCESS_FINE_LOCATION)
                   != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this,
           arrayOf(android.Manifest.permission.ACCESS_FINE_LOCATION), 1)
    }
    else {
        var locationRequest =
             LocationRequest.Builder(Priority.PRIORITY_HIGH_ACCURACY,
                                     10000L).build()
        fusedLocationProviderClient.requestLocationUpdates(locationRequest,
                              locationCallback, Looper.getMainLooper())
        getLastLocation()
    }
}
```

The Main activity (cont'ed)

```
    override fun onRequestPermissionsResult(
        requestCode: Int,
        permissions: Array<out String>,
        grantResults: IntArray
    ) {
      super.onRequestPermissionsResult(requestCode, permissions,
                                       grantResults)
      if (grantResults.size > 0 && grantResults[0] ==
                                PackageManager.PERMISSION_GRANTED)
          getLastLocation()
    }

    fun getLastLocation() {
        fusedLocationProviderClient.lastLocation.addOnSuccessListener {
                location -> displayLastLocation()
        }
    }

    fun showMap() {
        var i = Intent(this, MapsActivity::class.java)
        i.putExtra("latitude", lastLocation?.latitude)
        i.putExtra("longitude", lastLocation?.longitude)
        startActivity(i)
    }
}
```

The Map activity

```
package com.example.mapstest

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

import com.google.android.gms.maps.CameraUpdateFactory
import com.google.android.gms.maps.GoogleMap
import com.google.android.gms.maps.OnMapReadyCallback
import com.google.android.gms.maps.SupportMapFragment
import com.google.android.gms.maps.model.LatLng
import com.google.android.gms.maps.model.MarkerOptions
import com.example.mapstest.databinding.ActivityMapsBinding

class MapsActivity : AppCompatActivity(), OnMapReadyCallback {

    private lateinit var mMap: GoogleMap
    private lateinit var binding: ActivityMapsBinding
```

The Map activity (cont'ed)

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)

    binding = ActivityMapsBinding.inflate(layoutInflater)
    setContentView(binding.root)

    // Obtain the SupportMapFragment and get notified when the map is ready
    val mapFragment = supportFragmentManager
        .findFragmentById(R.id.map) as SupportMapFragment
    mapFragment.getMapAsync(this)
}
```

The Map activity (cont'ed)

```
/**
     * Manipulates the map once available.
     * This callback is triggered when the map is ready to be used.
     */
    override fun onMapReady(googleMap: GoogleMap) {
        mMap = googleMap

        // Add a marker in Sydney and move the camera
        //val location = LatLng(51.5209479,-0.142197)
        val location = LatLng(intent.getDoubleExtra("latitude", 0.0),
            intent.getDoubleExtra("longitude", 0.0))

        mMap.addMarker(MarkerOptions().position(location).title(
                                        "Marker for last Location"))
        //mMap.moveCamera(CameraUpdateFactory.newLatLng(location))
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(location, 10.0f))
    }
}
```

Applications Developed in this Module

1. Lottery app.
2. Lost Dogs - Notify owners by email for their lost dog based on recognising the dog image.
3. Identify the dog breed based on random dog images.
4. The Memory game - Highlighting squares in a grid for a few seconds and challenge the user to recall the hidden squares.
5. The Tic Tac Toe Game (the Computer player attacks and defends in a logical manner)
6. Employee management system in a database.
7. The Book Finder app (retrieve details of a book from Internet)
8. The Weather App
9. Shopping management (add products and calculate their total cost by adding them to a database
10. Display the current (last) location of a user in a map.
11. Coctails app (display recipe and picture of a cocktail by searching in Internet).

Coursework apps:

1. Dice game.
2. Meals retrieval and search from Internet.