Lecture 15: Location & Maps
IN721: Design and Development of Applications for Mobile Devices
Semester One, 2020

Kaiako: Grayson Orr

Te Kura Matatini ki Otago, Ōtepoti, Aotearoa

Friday, 8 May

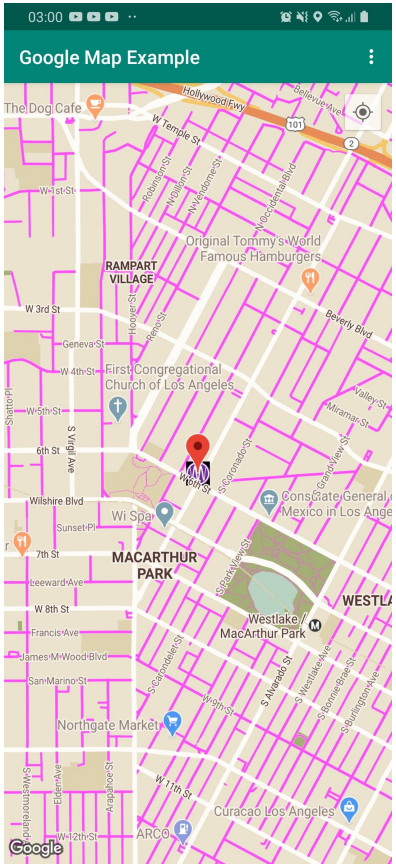
Lecture 14: Camera Topics

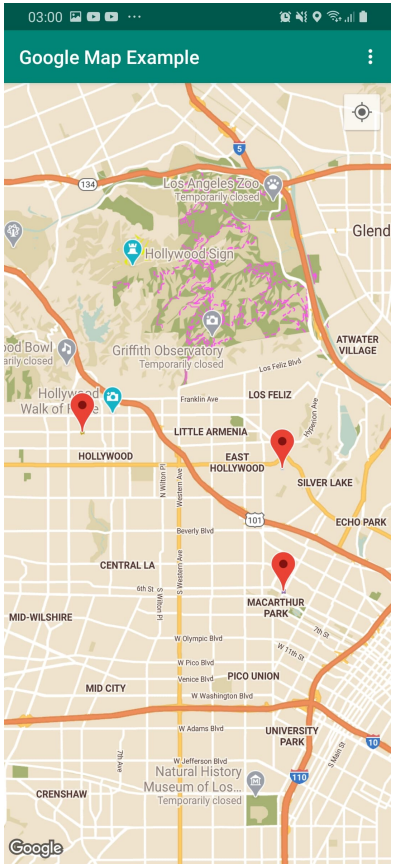
- Camera API

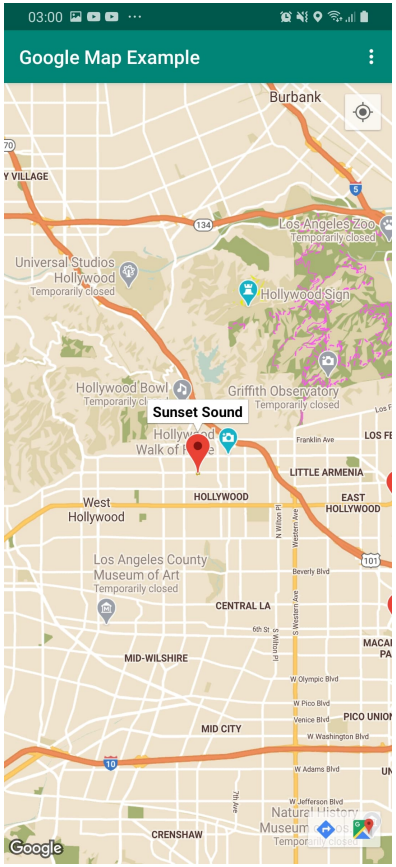
Lecture 15: Location & Maps Topics

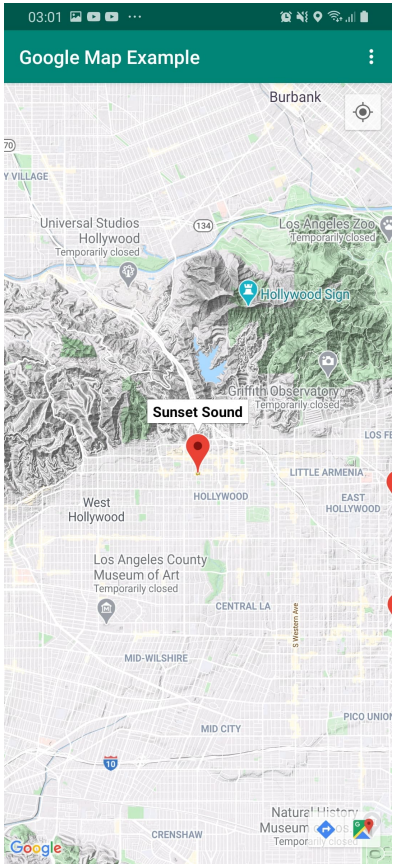
- Location
- Google Maps

Today's Practical - Part One

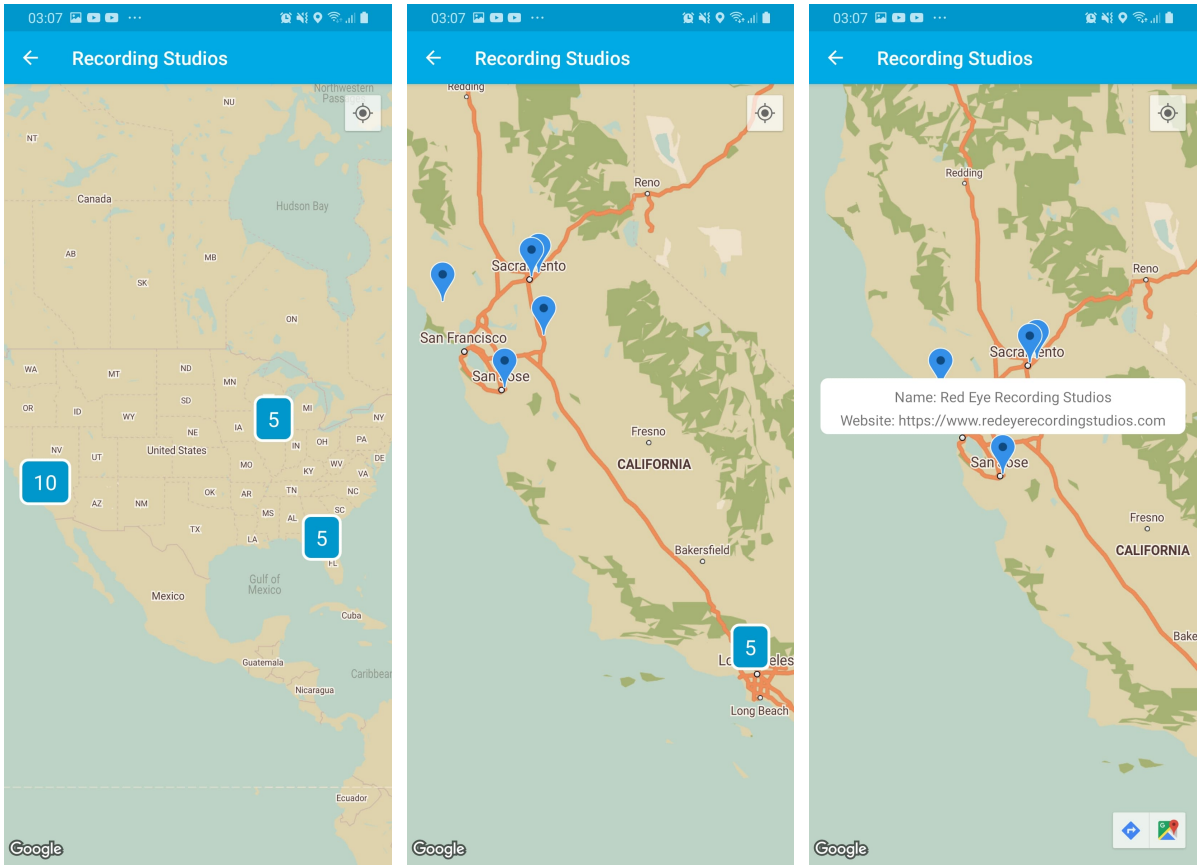




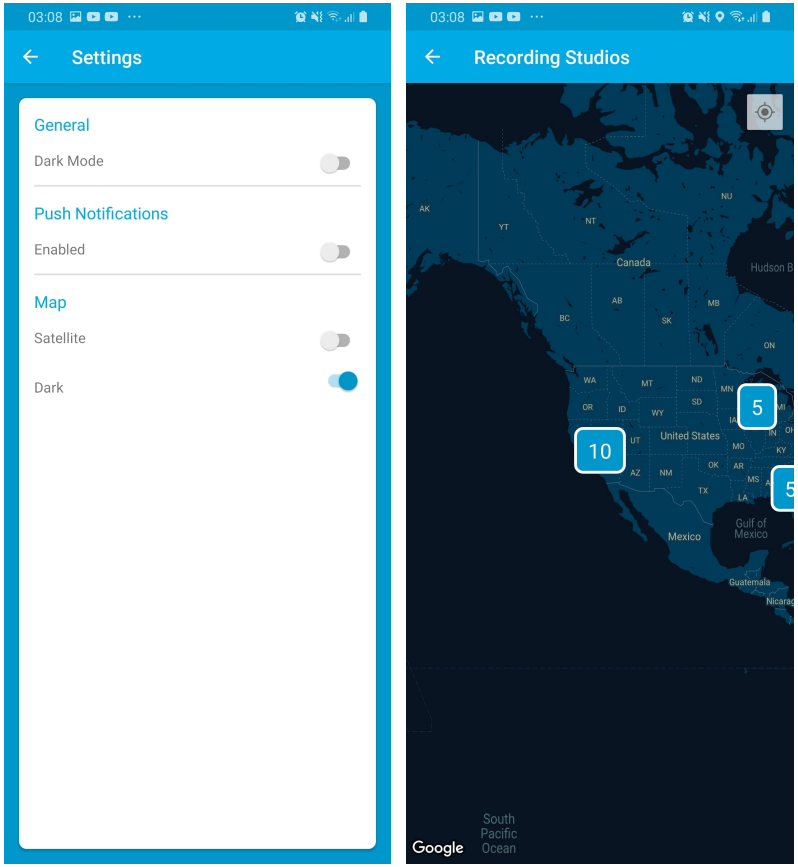




Today's Practical - Part Two (Research)



Today's Practical - Part Two (Research)



Location - Build Gradle

- Add the following dependencies to build.gradle (Module):
  - implementation ‘com.google.android.gms:play-services-maps:17.0.0‘
  - implementation ‘com.google.maps.android:android-maps-utils:0.5‘

Location - MapsActivity

- Create a new class called MapsActivity which extends AppCompatActivity or BaseActivity & OnMapReadyCallback
- getMapAsync - initialises the maps system & view

```
class MapsActivity : AppCompatActivity(), OnMapReadyCallback {

    private lateinit var map: GoogleMap

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_maps)
        val mapFragment: SupportMapFragment = supportFragmentManager
            .findFragmentById(R.id.map) as SupportMapFragment
        mapFragment.getMapAsync(this@MapsActivity)
    }
```

Location - Activity Layout

- activity_maps.xml
- Simplest way to place a map in an application is using a fragment

```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    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="graysono.com.maps.MapsActivity" />
```

Location - API Key

- Store this safely...
- Create a google_maps_api.xml (debug) in the values resource directory

```
<resources>
    <string name="google_maps_key" templateMergeStrategy="preserve"
        translatable="false">AIzaSyBOm2SY3mY9Awm8qmxoIMWT236M3bGZB8k</string>
</resources>
```

Location - AndroidManifest

- Declare permissions - ACCESS_FINE_LOCATION
- API key for Google Maps

```xml
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="graysono.com.maps">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Google Map Example"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyBOm2SY3mY9Awm8qmxoIMWT236M3bGZB8k" />

        <activity
            android:name="graysono.com.maps.MapsActivity"
            android:label="Google Map Example">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Location - OnMapReady

- Override function - OnMapReady
- Reference to Google Map

```
override fun onMapReady(googleMap: GoogleMap) {
    map = googleMap

    val data: ArrayList<MapsData> = arrayListOf(
        MapsData( name: "Sunset Sound", LatLng(34.0977818, -118.3349175), R.drawable.sunset_sound),
        MapsData( name: "MIX Recording Studio", LatLng(34.0628191, -118.281333), R.drawable.mix_recording_studio),
        MapsData( name: "Silverlake Recording Studios", LatLng(34.089849, -118.281676), R.drawable.silverlake_recording_studios)
    )

    val zoomLevel = 15f
    map.moveCamera(CameraUpdateFactory.newLatLngZoom(data[1].location, zoomLevel))

    for (d: MapsData in data) {
        map.addMarker(
            MarkerOptions()
                .position(d.location)
                .title(d.name)
        )
        val overlaySize = 100f
        val googleOverlay: GroundOverlayOptions = GroundOverlayOptions()
            .image(BitmapDescriptorFactory.fromResource(d.drawable))
            .position(d.location, overlaySize)
        map.addGroundOverlay(googleOverlay)
    }

    setMapLongClick(map)
    setPointOfInterest(map)
    setMapStyle(map)
    enableMyLocation()
}
```

Location - Menu

- Menu containing four items
- Change the map's type
  - Normal
  - Hybrid
  - Satellite
  - Terrain

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    val inflater: MenuInflater = menuInflater
    inflater.inflate(R.menu.map_options, menu)
    return true
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    when (item.itemId) {
        R.id.normal_map -> map.mapType = GoogleMap.MAP_TYPE_NORMAL
        R.id.hybrid_map -> map.mapType = GoogleMap.MAP_TYPE_HYBRID
        R.id.satellite_map -> map.mapType = GoogleMap.MAP_TYPE_SATELLITE
        R.id.terrain_map -> map.mapType = GoogleMap.MAP_TYPE_TERRAIN
    }
    return true
}
```

Location

- Private function - setMapLongClick
- Reference to Google Map
- setOnMapLongClickListener
  - Add a new marker to the map
  - Add marker options - position, title, snippet & icon

```
private fun setMapLongClick(map: GoogleMap) {
    map.setOnMapLongClickListener { latLng ->
        val snippet: String =
            "Latitude: " + latLng.latitude + ", " + "Longitude: " + latLng.longitude
        map.addMarker(
            MarkerOptions()
                .position(latLng)
                .title(getString("Dropped Pin"))
                .snippet(snippet)
                .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_BLUE))
        )
    }
}
```

Location

- Private function - setPointOfInterest
- Reference to Google Map
- Displays each MapsData object's name in an info window
- Default marker colour - red

```
private fun setPointOfInterest(map: GoogleMap) {
    map.setOnPoiClickListener { poi ->
        val poiMarker: Marker = map.addMarker(
            MarkerOptions()
                .position(poi.latLng)
                .title(poi.name)
        )
        poiMarker.showInfoWindow()
    }
}
```

Location - Map Style

- Private function - setMapStyle
- Reference to Google Map
- Loads raw resource style from raw resource directory
- JSON file containing map styling information
- Error checking - loading & missing resource style
- Resource - https://mapstyle.withgoogle.com/

```
private fun setMapStyle(map: GoogleMap) {
    try {
        val success: Boolean = map.setMapStyle(
            MapStyleOptions.loadRawResourceStyle(
                this,
                R.raw.map_style
            )
        )
        if (!success) {
            Toast.makeText( context: this@MapsActivity,  text: "Error loading map style.", Toast.LENGTH_LONG)
                .show()
        }
    } catch (e: Resources.NotFoundException) {
        Toast.makeText( context: this@MapsActivity,  text: "Map style not found.", Toast.LENGTH_LONG).show()
    }
}
```

Location - Permissions

- Private function - isPermissionGranted
- Check if location has been granted by the user to the given package
  - Returns a boolean value

```
private fun isPermissionGranted(): Boolean {
    return ContextCompat.checkSelfPermission(
        context: this@MapsActivity,
        Manifest.permission.ACCESS_FINE_LOCATION
    ) == PackageManager.PERMISSION_GRANTED
}
```

Location - Permissions

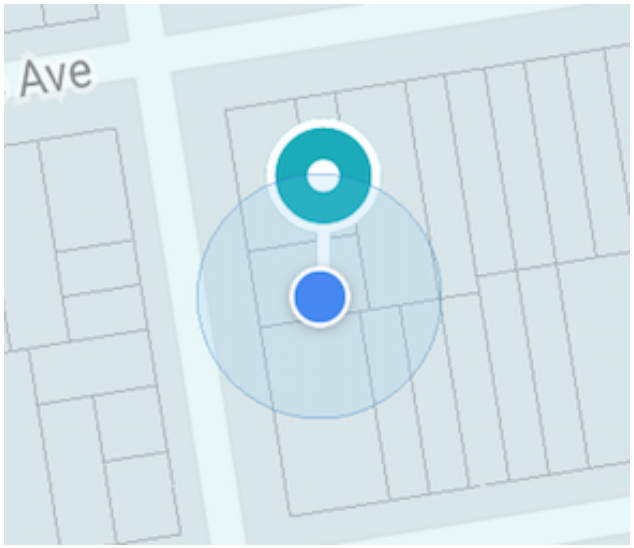
- Private function - isRequestPermissionResult
- Check if the request code = REQUEST_CODE_PERMISSION
- Enable location if location has been granted to the given package

```
override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<String>,
    grantResults: IntArray
) {
    if (requestCode == REQUEST_LOCATION_PERMISSION) {
        if (grantResults.contains(PackageManager.PERMISSION_GRANTED)) {
            enableMyLocation()
        }
    }
}
```

Location - Permissions

- Private function - enableMyLocation
- Get the status of my-location layer - set it to true
  - The little blue icon
- If the user has not granted permission, request permission

```
private fun enableMyLocation() {
    if (isPermissionGranted()) {
        map.isMyLocationEnabled = true
    } else {
        ActivityCompat.requestPermissions(
            activity: this@MapsActivity,
            arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
            REQUEST_LOCATION_PERMISSION
        )
    }
}
```



Location - MapsData

- Create a class called MapsData
- MapsData object contains a name, latitude/longitude & drawable

```
class MapsData (
    var name: String,
    var location: LatLng,
    var drawable: Int
)
```

Practical

- Series of tasks covering today's lecture
- Worth 3% of your final mark for the Design and Development of Applications for Mobile Devices course
- Deadline: Friday, 12 June at 5pm