LabW09 – Location Access

Objectives:

1. Understand how to use GPS and Google Map

Tasks:

1. Use Google Map v2
2. Get GPS location

Homework2:

Add more functions to your ToDoList application

Worth 7.5 marks

Due in the lab of Week 11

Task 1: Use Google Map v2

Follow the tutorial at https://developers.google.com/maps/documentation/android-api/start

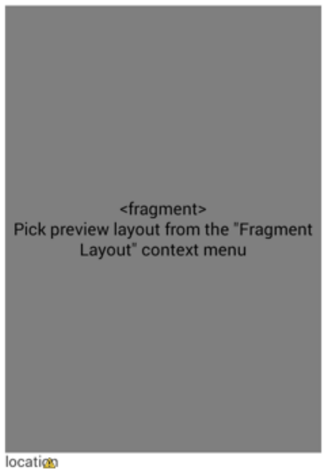
1. Add Google Play Services into Android workspace
2. Get Google Maps API key from Google API console. Notice that the "package name" is the same of the one defined in your **AndroidManifest.xml**
3. Make Google Play Services as a library project of your own project
4. Make sure all required permissions are set in the manifest.
5. Use the code provided to show a Google Map with a marker in your project.

You can get some simple snippets to customise your map from the top section of: https://developers.google.com/maps/documentation/android/

Task 2: Get GPS location

You need to get Task 1 done first.

1. Copy the **activity_week09.xml** (in lab files) into your Task 1 project's res/layout folder. It only contains a TextView at the bottom to show the latitude and longitude, and a Google



Map Fragment above it to cover the rest of the screen.

2. In your Activity, create variables to reference the textview and the location client.

```
    GoogleApiClient mGoogleApiClient;

    TextView locationTextView;

    private Location mLastLocation;
```

3. In onCreate(), use the layout activity_week09, and initialise the location client. Also initialise the Google Map (you should have done it in Task 1).

```
super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_maps);

        locationTextView = (TextView)findViewById(R.id.textView);

        setUpMapIfNeeded();

}
```

4. Make sure your activity implements the Location-related Callbacks. We will add the required methods in the next step.

```
public class ActivityXXXXX extends FragmentActivity implements
GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener
```

5. Add the required callback methods.

```
@Override

    @Override
    public void onConnected(Bundle connectionHint) {
        Location mCurrentLocation = mLocationClient.getLastLocation();
        Location mCurrentLocation = LocationServices.FusedLocationApi.getLastLocation(
                mGoogleApiClient);
        double lat = mCurrentLocation.getLatitude();
        double lon = mCurrentLocation.getLongitude();
        String msg = "Current Location: " +
                Double.toString(lat) + "," +
                Double.toString(lon);
        locationTextView.setText(msg);
        //update google map, move it to current location
        CameraPosition cameraPosition = new CameraPosition.Builder().target(new
LatLng(lat,lon)).zoom(16).build();
        mMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));

        MarkerOptions marker = new MarkerOptions().position(new LatLng(lat, lon)) // create marker

                .title("I am here");
        mMap.addMarker(marker); // adding marker

    }

    @Override
    public void onConnectionSuspended(int i) {
        locationTextView.setText("Connection suspended.");
    }

    @Override
    public void onConnectionFailed(ConnectionResult connectionResult) {
        locationTextView.setText("Connection failed.");
    }


    private void setUpMapIfNeeded() {
        // Do a null check to confirm that we have not already instantiated the map.
        if (mMap == null) {
            // Try to obtain the map from the SupportMapFragment.
            mMap = ((SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.map))
                    .getMap();
            // Check if we were successful in obtaining the map.
            if (mMap != null) {
                setUpMap();
            }
        }
    }

    private void setUpMap() {
        mMap.addMarker(new MarkerOptions().position(new LatLng(0, 0)).title("Marker"));

        mGoogleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API)
                .build();
        mGoogleApiClient.connect();
        locationTextView.setText("Detecting location...");
    }
```

6. Run it, preferably on a real phone.

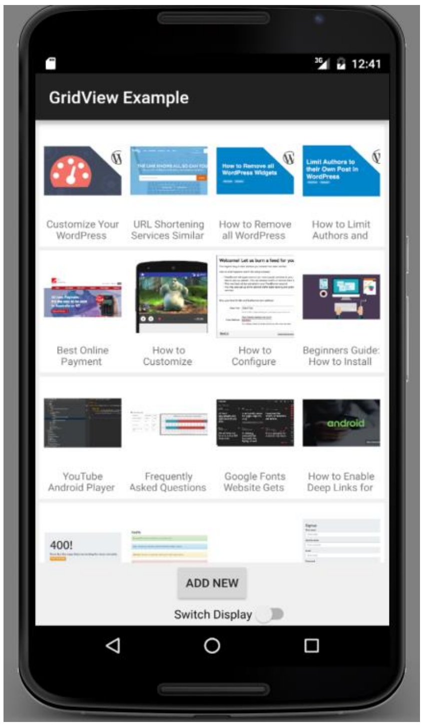
   If you run it on the emulator, follow the section “**Setup Emulator**” in: https://github.com/thecodepath/android_guides/wiki/Google-Maps-Fragment-Guide

   The final code will be the similar to that in **ActivityWeek09.java** (in lab files)

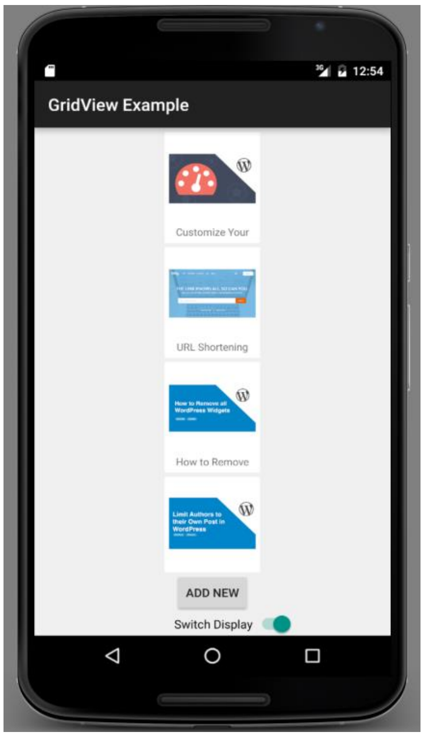
Homework2: [7.5 marks, due in the lab Week 11]

In this assignment, you need to add more functions (i.e. camera, map display, user account login) to your ToDoList application. Please notice that You have to finish Homework1 first.

1) The main view of your app should contain a **GridView** to display the ToDoList and captured images for each TodoItem, a “**ADD NEW**” button to add new TodoItem, and a **Switch** Display button to switch display mode (either a GridView or a ListView. The latter view mode is based on the distance to current location) [2.5 marks].


GridView Example
Customize Your WordPress
URL Shortening Services Similar
How to Remove all WordPress
How to Limit Authors and
Best Online Payment
How to Customize
How to Configure
Beginners Guide: How to Install
YouTube Android Player
Frequently Asked Questions
Google Fonts Website Gets
How to Enable Deep Links for
ADD NEW
Switch Display

GridView



ListView

2) Once user click the “**ADD NEW**” button in the main view, the app will change to a new view to add a new TodoItem. Each TodoItem should

consist with **text** (from text input), **image** (captured from camera), as well as recorded **location information** [2.5 marks].

Hint: you only need to store the photo path in the local DB, rather than add the entire image as a binary file to local DB.

3) Your app also should be able to connect to Facebook account [2.5 marks]:

- Once you launch the app for the first time, the app will pop up a dialog that ask user to login with the Facebook account.
- Load and display the ToDoList once user login the Facebook account.