



# Lab 6 – Google Map

KUAN-TING LAI

2020/10/24

# Use Google Map in Your APP

---

- Mark specific positions
- Sets of line segments (Polylines)
- Enclosed segments (Polygons)
- Bitmap graphics anchored to specific positions on the map (Ground Overlays).
- Sets of images which are displayed on top of the base map tiles (Tile Overlays).



# Get API Key

---

- Register an account at “cloud.google.com”
- APIs & Services -> Credentials

# Select APIs & Services -> Credentials

The screenshot shows the Google Cloud Platform dashboard with the following navigation path highlighted:

- Google Cloud Platform
- My Project
- DASHBOARD
- ACTIVITY
- Home
- Pins appear here
- Marketplace
- Billing
- APIs & Services (circled in red)
- Support
- IAM & admin
- Getting started
- Security
- COMPUTE
- App Engine
- Compute Engine
- Kubernetes Engine
- Cloud Functions
- STORAGE
- Riposte

The "APIs & Services" menu item is circled in red. Within the "APIs & Services" menu, the "Credentials" option is also circled in red.

The main dashboard area displays the following sections:

- Project info**: Shows Project name (My Project), Project ID (liquid-polygon-652), and Object number (9986274705).
- API APIs**: A chart titled "Requests (requests/sec)" showing data over time. A single data point is visible: api/request\_count:consumed\_api:REDUCE\_SUM(liquid-polygon-652): 0.017 at 10:00 AM.
- Resources**: States "This project has no resources".
- Trace**: States "No trace data from the past 7 days". Includes a link to "Get started with Stackdriver Trace".
- Getting Started**: Includes a link to "Enable APIs and get credentials like keys".
- Google Cloud Platform status**: Shows "All services normal". Includes a link to "Go to Cloud status dashboard".
- Error Reporting**: States "No sign of any errors. Have you set up Error Reporting?". Includes a link to "Learn how to set up Error Reporting".
- News**: Lists recent news items:
  - Scripting with gcloud: a beginner's guide to automating GCP tasks (2 days ago)
  - Serverless from the ground up: Adding a user interface with Google Sheets (Part 2) (2 days ago)
  - At HackerOne, automatic updates in Chrome OS reduce worries about security on Pixelbooks (3 days ago)Includes a link to "Read all news".

# Credentials

- AlzaSyABNsVHTjg8JRmMWTA3oQRxmP-MDEhBG0s

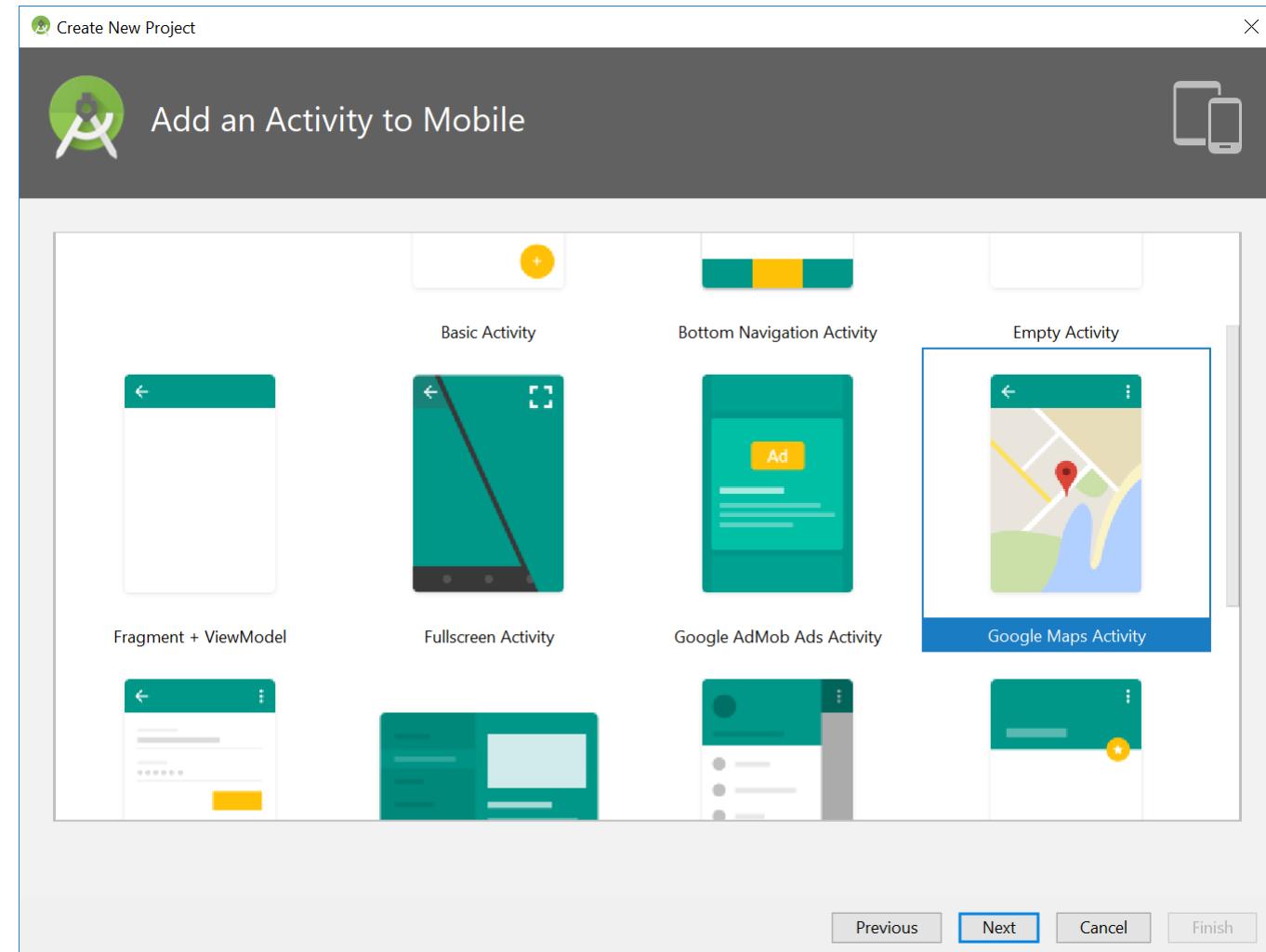
The screenshot shows the Google Cloud Platform interface for managing credentials. The top navigation bar includes the 'Google Cloud Platform' logo, a 'My Project' dropdown, a search bar, and various status icons. On the left, a sidebar menu under 'API' lists 'Dashboard', 'Library', and 'Credentials'. The main content area is titled 'Credentials' and contains three sections: 'Credentials', 'OAuth consent screen', and 'Domain verification'. A prominent blue button labeled 'Create credentials' is highlighted with a red oval. Below it, a note says 'Create credentials to access your enabled APIs. Refer to the API documentation for details.' The 'API keys' section lists one entry: 'API key 1' created on 'Oct 27, 2018' with 'None' restrictions. The 'OAuth 2.0 client IDs' section lists one entry: 'Service account client 1' created on 'Jul 26, 2014' as a 'Service account client'.

Name	Creation date	Restrictions	Key
API key 1	Oct 27, 2018	None	[REDACTED]

Name	Creation date	Type	Client ID
Service account client 1	Jul 26, 2014	Service account client	269386274705-jeq1e7k2cogutgosjg8lvhrof3oms.apps.googleusercontent.com

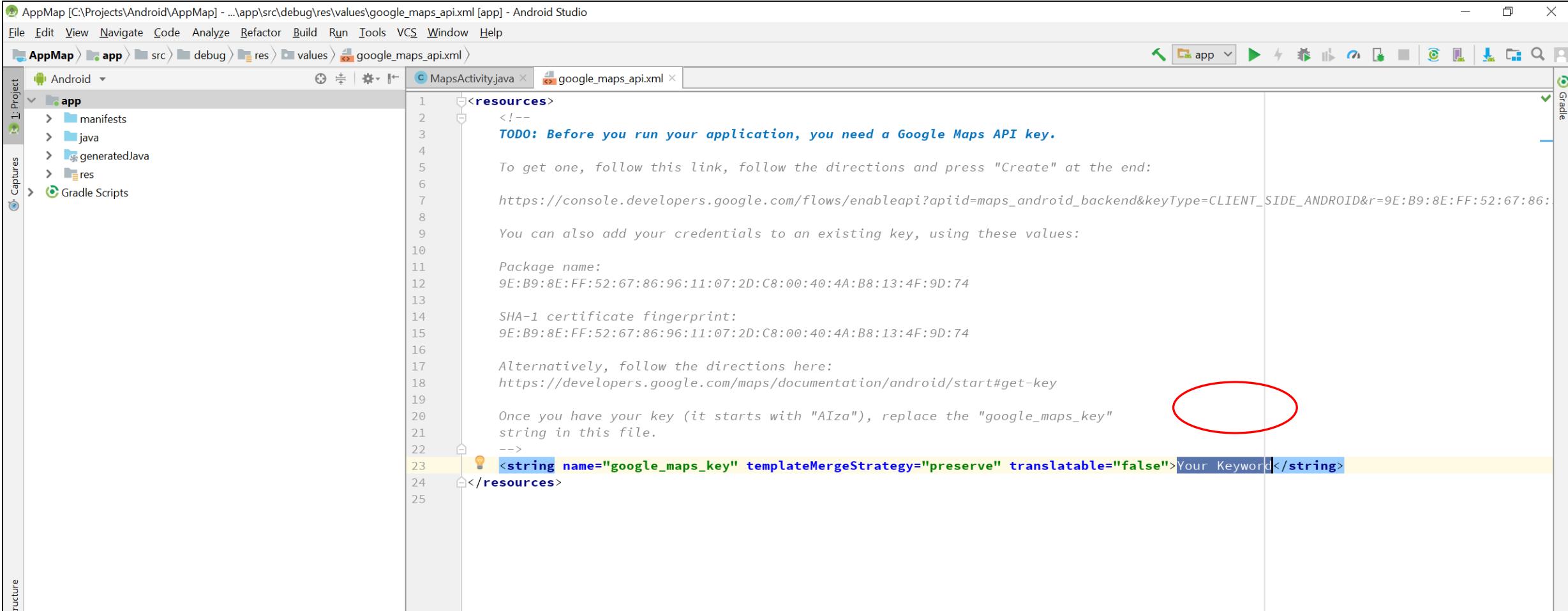
# Create Google Maps Project

- Create new project called “AppMap”
- Use default settings
- Select “Google Maps Activity”



# Paste Your Key in google\_maps\_api.xml

- app/res/google\_maps\_api.xml



AppMap [C:\Projects\Android\AppMap] - ...\\app\\src\\debug\\res\\values\\google\_maps\_api.xml [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools Window Help

AppMap app src debug res values google\_maps\_api.xml

MapsActivity.java x google\_maps\_api.xml x

1 <resources>

2 <!--

3 TODO: Before you run your application, you need a Google Maps API key.

4

5 To get one, follow this link, follow the directions and press "Create" at the end:

6

7 [https://console.developers.google.com/flows/enableapi?apiid=maps\\_android\\_backend&keyType=CLIENT\\_SIDE\\_ANDROID&r=9E:B9:8E:FF:52:67:86](https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID&r=9E:B9:8E:FF:52:67:86)

8

9 You can also add your credentials to an existing key, using these values:

10

11 Package name:  
9E:B9:8E:FF:52:67:86:96:11:07:2D:C8:00:40:4A:B8:13:4F:9D:74

12

13 SHA-1 certificate fingerprint:  
9E:B9:8E:FF:52:67:86:96:11:07:2D:C8:00:40:4A:B8:13:4F:9D:74

14

15 Alternatively, follow the directions here:  
<https://developers.google.com/maps/documentation/android/start#get-key>

16

17 Once you have your key (it starts with "AIza"), replace the "google\_maps\_key"

18 string in this file.

19 -->

20 <string name="google\_maps\_key" templateMergeStrategy="preserve" translatable="false">Your Keyword</string>

21

22 </resources>

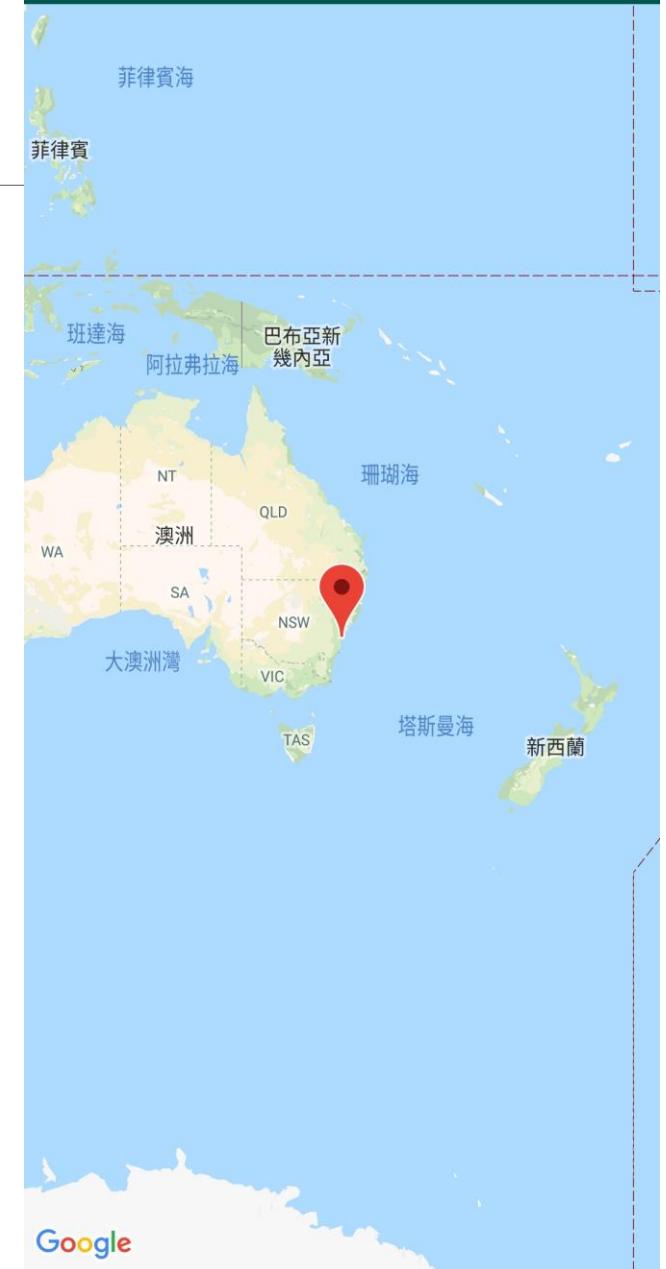
23

24

25

# Compile

- The default location is set at Sydney, let's change to NTUT



# Set the Position to NTUT

---

- (Latitude, Longitude) = (25.067060, 121.380580)

```
@Override  
public void onMapReady(GoogleMap googleMap) {  
    mMap = googleMap;  
  
    // Add a marker in NTUT and move the camera  
    LatLng ntut = new LatLng(25.0422329, 121.5354974);  
    mMap.addMarker(new MarkerOptions().position(ntut).title("Marker in National Taipei  
University of Technology"));  
    mMap.moveCamera(CameraUpdateFactory.newLatLng(ntut));  
}
```

# Compile

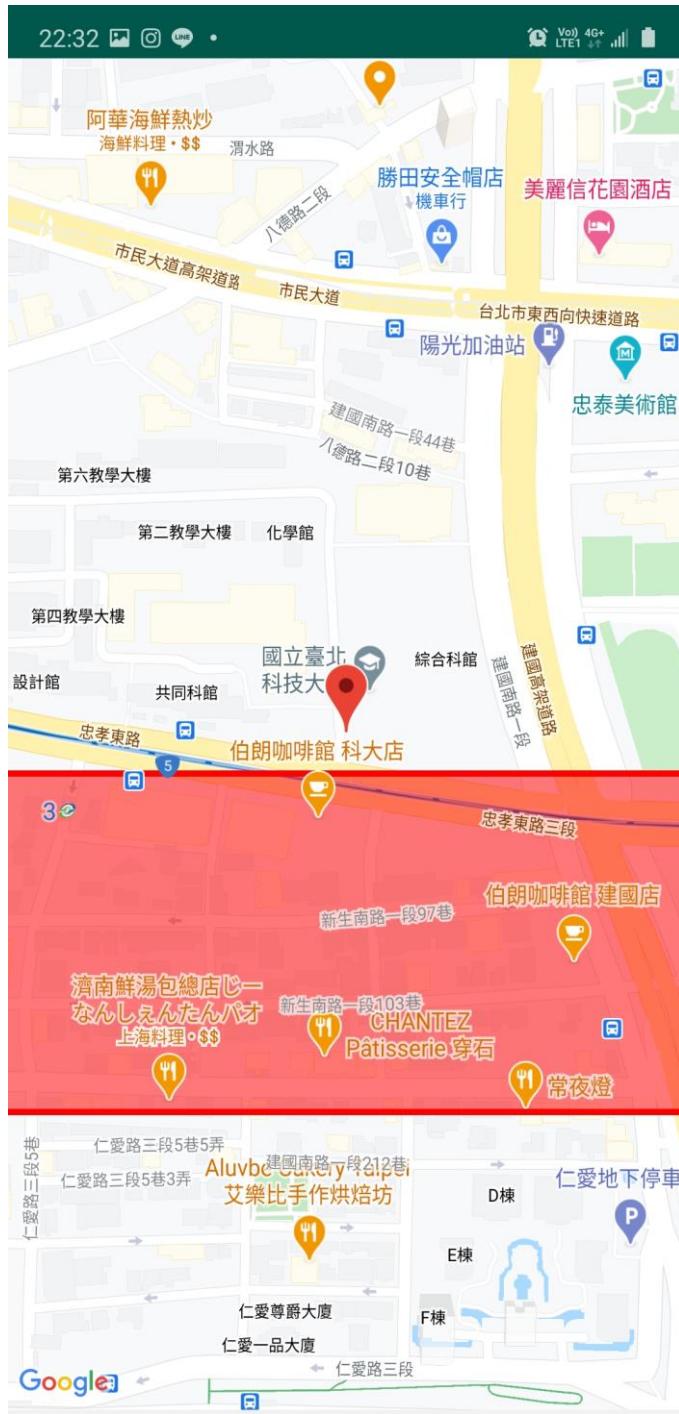
- The location is set to NTUT



# Add Polygon and Set Zoom Level

```
@Override  
public void onMapReady(GoogleMap googleMap) {  
    mMap = googleMap;  
  
    LatLng ntut = new LatLng(25.0422329, 121.5354974);  
    mMap.addMarker(new MarkerOptions().position(ntut).title("Marker in  
National Taipei University of Technology"));  
    mMap.moveCamera(CameraUpdateFactory.newLatLng(ntut));  
    mMap.animateCamera( CameraUpdateFactory.zoomTo( 17.0f ) );  
  
    Polygon polygon = mMap.addPolygon(new PolygonOptions()  
        .add(new LatLng(25.042, 121.538),  
              new LatLng(25.040, 121.538),  
              new LatLng(25.040, 121.532),  
              new LatLng(25.042, 121.532))  
        .strokeColor(Color.RED)  
        .fillColor(0x88FF0000));  
}
```

# Results



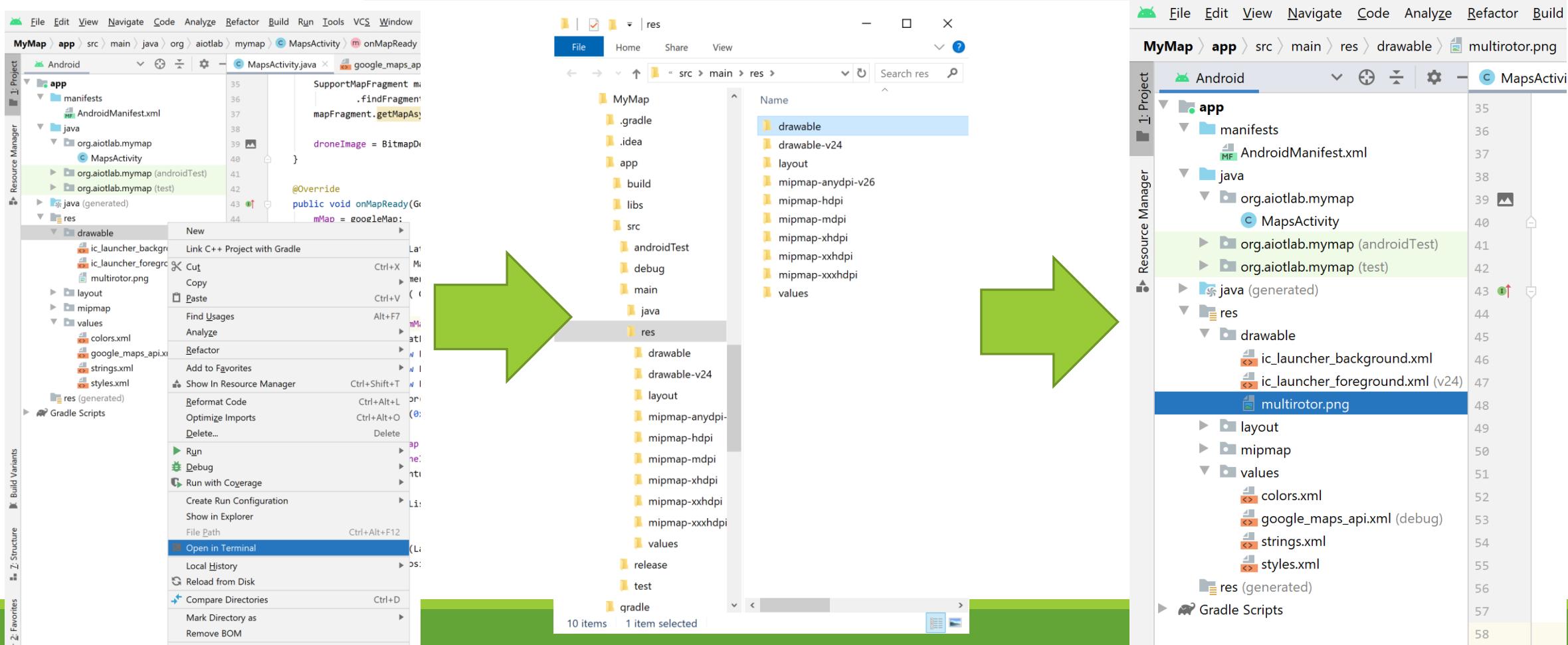
# Adding an Image on the Map (groundOverlay)

---



# Copy an Image to “res/drawable/”

- Open folder of “res/drawable” => Paste an image to the folder
  - Drone image: <https://www.pngwing.com/en/free-png-muxkx>



# groundOverlay

- Load Bitmap from R.drawable via BitmapDescriptor in `onCreate()`

```
public class MapsActivity extends FragmentActivity implements OnMapClickListener,  
OnMapReadyCallback {  
    private GoogleMap mMap;  
    BitmapDescriptor droneImage = null;  
    private GroundOverlay groundOverlay;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        .....  
        droneImage = BitmapDescriptorFactory.fromResource(R.drawable.multirotor);  
    }  
  
    @Override  
    public void onMapReady(GoogleMap googleMap) {  
        mMap = googleMap;  
        .....  
        groundOverlay = mMap.addGroundOverlay(new GroundOverlayOptions()  
            .image(droneImage).anchor(0.5f, 0.5f)  
            .position(ntut, 120));  
    }  
}
```

# Moving the Drone to the Clicked Location

---

```
public class MapsActivity extends FragmentActivity implements OnMapClickListener, OnMapReadyCallback {  
    private GoogleMap mMap;  
    BitmapDescriptor droneImage = null;  
    private GroundOverlay groundOverlay;  
  
    @Override  
    public void onMapReady(GoogleMap googleMap) {  
        mMap = googleMap;  
        .....  
        .....  
        mMap.setOnMapClickListener(this);  
    }  
  
    public void onMapClick(LatLng point) {  
        groundOverlay.setPosition(point);  
    }  
}
```

# Final Result



# More Google Map Examples

- <https://github.com/googlemaps/android-samples>

1. Get a Maps API key
2. Create a file in the root directory called secure.properties (this file should NOT be under version control to protect your API key)
3. Add a single line to secure.properties that looks like MAPS\_API\_KEY=YOUR\_API\_KEY, where YOUR\_API\_KEY is the API key you obtained in the first step
4. Build and run

The screenshot shows a mobile application interface for 'Google Maps API Demos'. At the top, there's a blue header bar with the time '14:13' and various status icons. Below the header is a dark blue navigation bar with the title 'Google Maps API Demos' in white. The main content area consists of a list of map features, each represented by a card with a title and a brief description. The features listed are: Basic Map (Launches a map), Camera (Demonstrates camera functions), Camera Clamping (Demonstrates how to constrain the camera to specific boundaries and zoom levels), Circles (Demonstrates how to add Circles to a map), Events (Demonstrates event handling), Ground Overlays (Demonstrates how to add a GroundOverlay to a map), Indoor (Demonstrates how to use the Indoor API), Layers (Demonstrates the different map layers), Lite Mode (Demonstrates some features on a map in lite mode), Lite Mode List (Demonstrates using maps in lite mode in a RecyclerView using LinearLayoutManager and GridLayoutManager), Location Source Demo (Demonstrates how to use a custom location source), Map In Pager (Demonstrates how to add a map to a ViewPager), Markers (Demonstrates how to add Markers to a map), and Marker Close Info Window on Retap (Demonstrates how to close the info window when the currently). Each card has a thin gray border.

- Basic Map  
Launches a map.
- Camera  
Demonstrates camera functions.
- Camera Clamping  
Demonstrates how to constrain the camera to specific boundaries and zoom levels.
- Circles  
Demonstrates how to add Circles to a map.
- Events  
Demonstrates event handling.
- Ground Overlays  
Demonstrates how to add a GroundOverlay to a map.
- Indoor  
Demonstrates how to use the Indoor API.
- Layers  
Demonstrates the different map layers.
- Lite Mode  
Demonstrates some features on a map in lite mode.
- Lite Mode List  
Demonstrates using maps in lite mode in a RecyclerView using LinearLayoutManager and GridLayoutManager.
- Location Source Demo  
Demonstrates how to use a custom location source.
- Map In Pager  
Demonstrates how to add a map to a ViewPager.
- Markers  
Demonstrates how to add Markers to a map.
- Marker Close Info Window on Retap  
Demonstrates how to close the info window when the currently