LA TROBE
UNIVERSITY

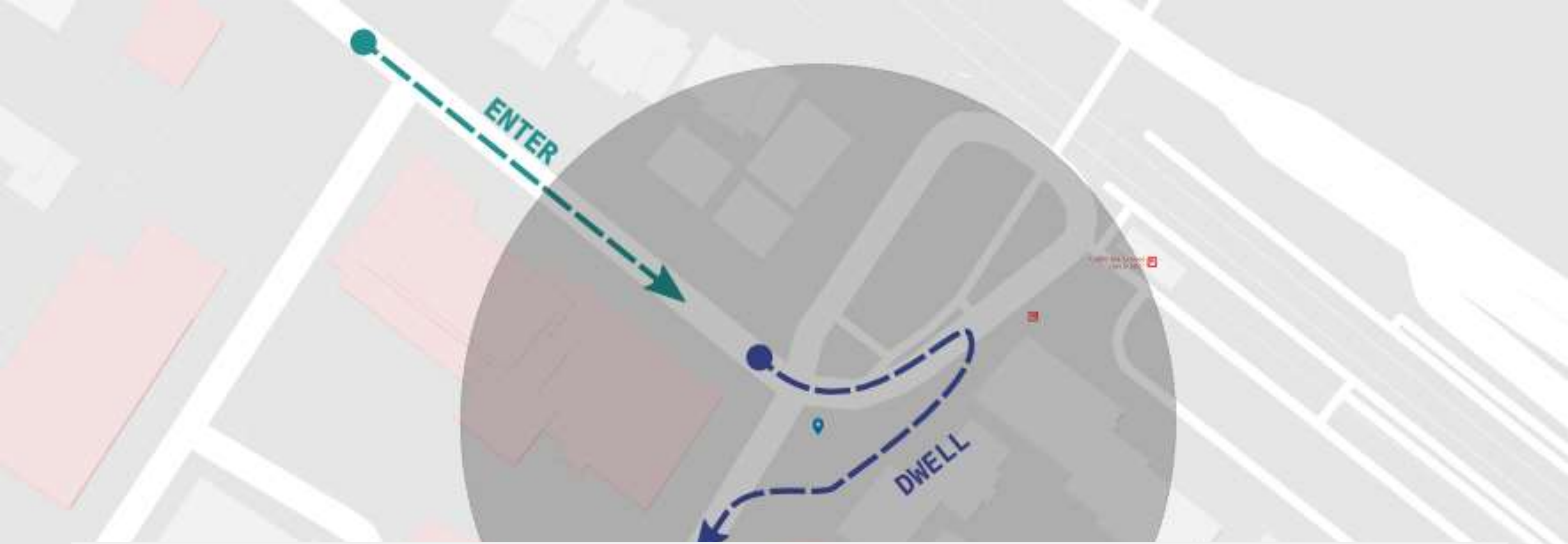
All kinds of clever

CSE2MAD

Mobile Application Development
Lecture 7 Part 1

Overview

- Location and GPS Part II
- Sensing in Android
- Camera
- Audio
- Image Processing
- Telephony



Geofencing & Proximity Alerts Continued

- Awareness of the user's current location with awareness of the user's proximity to locations that may be of interest.
- Can register to be informed when your location is within some distance of a given location (e.g., of a landmark)

https://developers.google.com/location-context/geofencing https://developer.android.com/training/location/geofencing

Set up for Geofence Monitoring

1. Handle Dangerous Permissions

- ACCESS_FINE_LOCATION
- ACCESS_BACKGROUND_LOCATION if your app targets Android 10 (API level 29) or higher

2. Using geofences requires the play-services-location library added to your project. To do that, open the build.gradle file for the app module and add the following dependency:

```
implementation 'com.google.android.gms:play-services-location:17.0.0'
```

3. Obtain a Google Maps API key, if you are using a Google maps activity just click the link in the google_maps_api.xml file otherwise manually create one as detailed here;

https://developers.google.com/maps/documentation/android-sdk/get-api-key

https://developer.android.com/training/location/geofencing

Set up for Geofence Monitoring

4. If you want to use a BroadcastReceiver to listen for geofence transitions, add an element specifying the service name. This element must be a child of the <application> element:

```
<application
  android:allowBackup="true">
  ...
  <receiver android:name=".GeofenceBroadcastReceiver"/>
<application/>
```

5. To access the location APIs, you need to create an instance of the Geofencing client.

```
private GeofencingClient geofencingClient;

@Override
public void onCreate(Bundle savedInstanceState) {
    // ...
    geofencingClient = LocationServices.getGeofencingClient(this);
}
```

https://developer.android.com/training/location/geofencing

Create the Geofence object and initial triggers

6. Build Geofence object

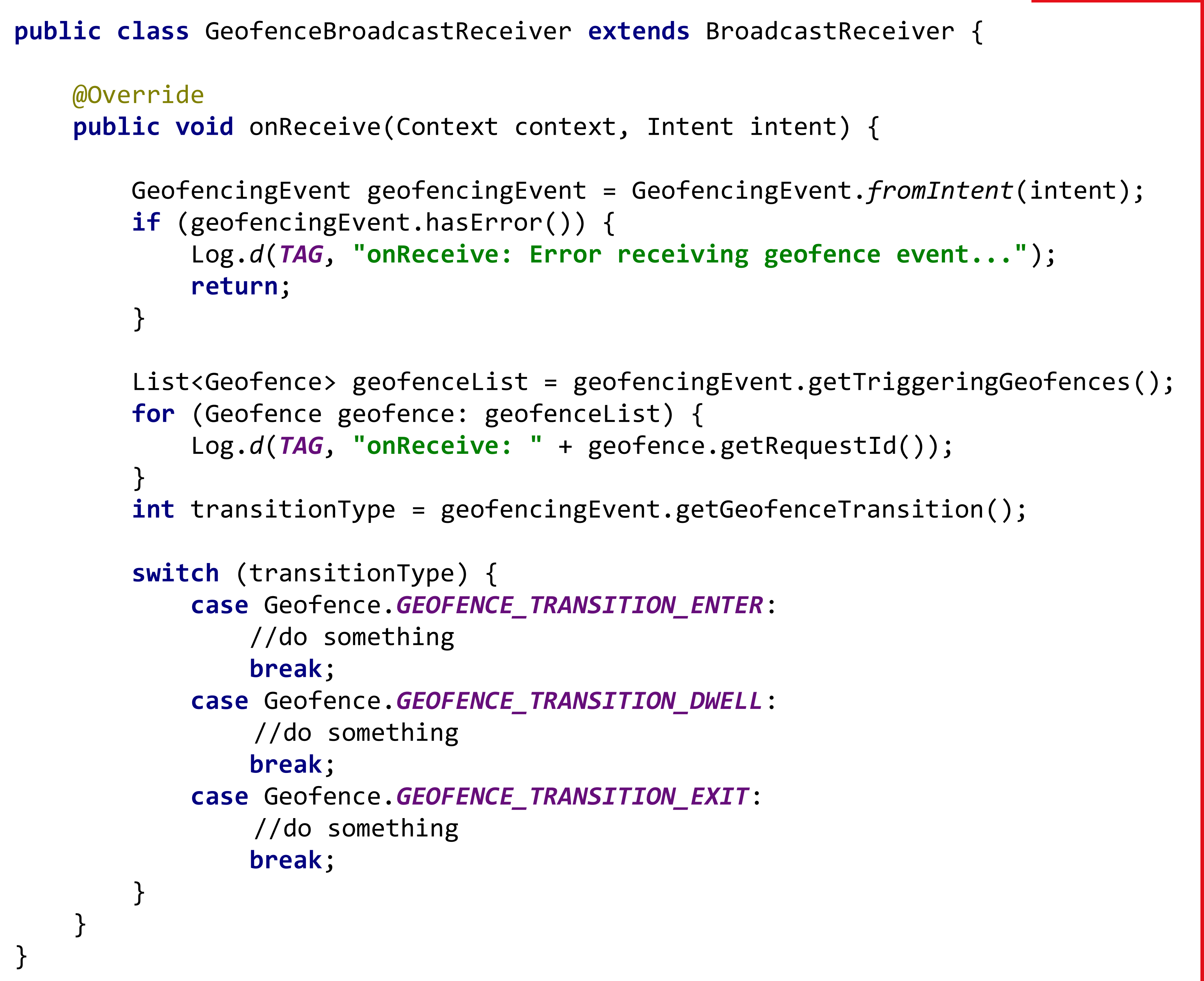
```
Geofence geofence = new Geofence.Builder()
        .setRequestId(“GeoFenceDemo1")
        .setCircularRegion(myLat, myLong, GEOFENCE_RADIUS_IN_METERS)
        .setExpirationDuration(GEOFENCE_EXPIRATION_IN_MILLISECONDS)
        .setTransitionTypes(Geofence.GEOFENCE_TRANSITION_ENTER |
                Geofence.GEOFENCE_TRANSITION_EXIT |
                Geofence.GEOFENCE_TRANSITION_DWELL)
        .setLoiteringDelay(1)
        .build();
```

7. Make GeofencingRequest with geofence object

```
public GeofencingRequest getGeofencingRequest(Geofence geofence) {
    return new GeofencingRequest.Builder()
            .addGeofence(geofence)
            .setInitialTrigger(GeofencingRequest.INITIAL_TRIGGER_ENTER)
            .build();
}
```

https://developer.android.com/training/location/geofencing

8. Define a broadcast receiver for geofence transition



```
public class GeofenceBroadcastReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {

        GeofencingEvent geofencingEvent = GeofencingEvent.fromIntent(intent);
        if (geofencingEvent.hasError()) {
            Log.d(TAG, "onReceive: Error receiving geofence event...");
            return;
        }

        List<Geofence> geofenceList = geofencingEvent.getTriggeringGeofences();
        for (Geofence geofence: geofenceList) {
            Log.d(TAG, "onReceive: " + geofence.getRequestId());
        }
        int transitionType = geofencingEvent.getGeofenceTransition();

        switch (transitionType) {
            case Geofence.GEOFENCE_TRANSITION_ENTER:
                //do something
                break;
            case Geofence.GEOFENCE_TRANSITION_DWELL:
                //do something
                break;
            case Geofence.GEOFENCE_TRANSITION_EXIT:
                //do something
                break;
        }
    }
}
```

Start a Broadcast Receiver with a Pending Intent

9. define a PendingIntent that starts a BroadcastReceiver

```
private PendingIntent getGeofencePendingIntent() {
    // Reuse the PendingIntent if we already have it.
    if (geofencePendingIntent != null) {
        return geofencePendingIntent;
    }
    Intent intent = new Intent(this, GeofenceBroadcastReceiver.class);
    // We use FLAG_UPDATE_CURRENT so that we get the same pending intent back when
    // calling addGeofences() and removeGeofences().
    geofencePendingIntent = PendingIntent.getBroadcast(this, 0, intent, PendingIntent.
            FLAG_UPDATE_CURRENT);
    return geofencePendingIntent;
}
```

https://developer.android.com/training/location/geofencing https://developer.android.com/reference/android/app/PendingIntent

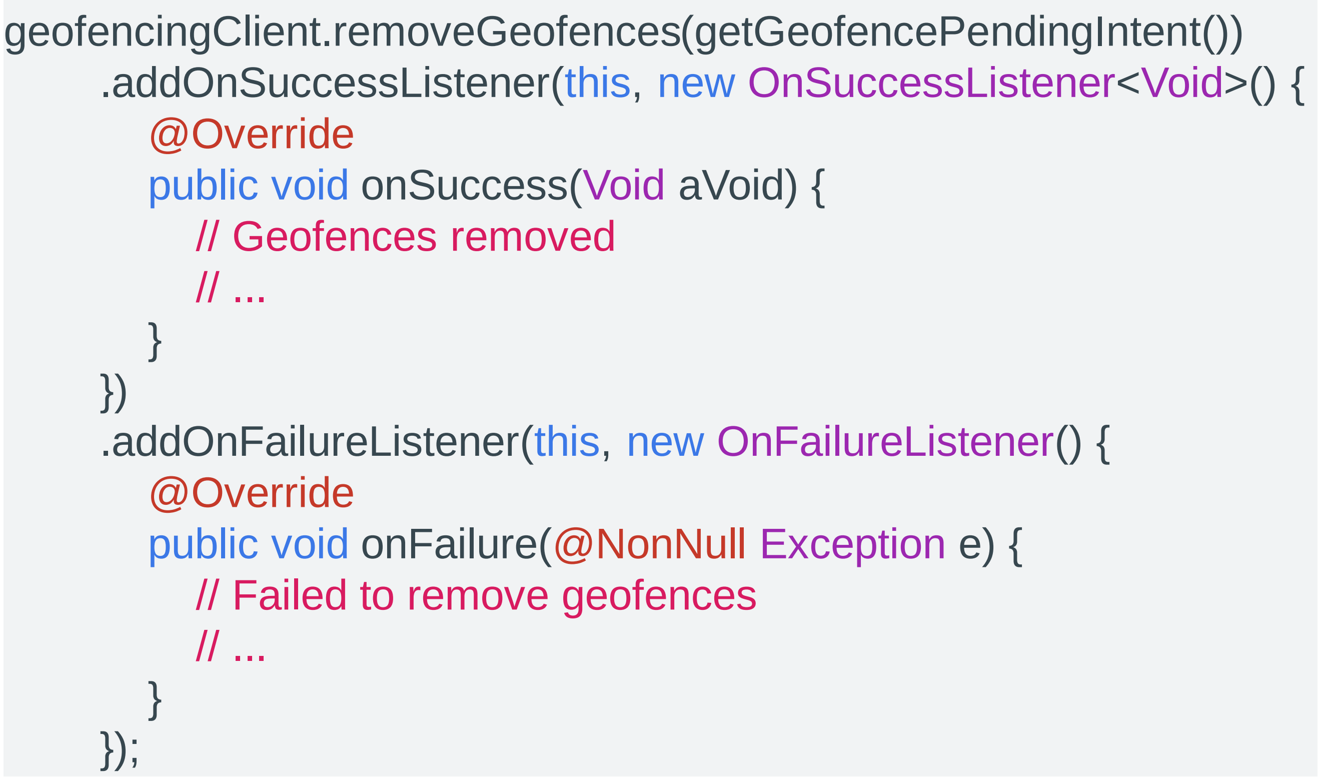
Add Geofence

10. Add the geofence

```
geofencingClient.addGeofences(getGeofencingRequest(), getGeofencePendingIntent())
        .addOnSuccessListener(this, new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {
                // Geofences added
                // ...
            }
        })
        .addOnFailureListener(this, new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                // Failed to add geofences
                // ...
            }
        });
```

https://developer.android.com/training/location/geofencing

Remove Geofence



```
geofencingClient.removeGeofences(getGeofencePendingIntent())
        .addOnSuccessListener(this, new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {
                // Geofences removed
                // ...
            }
        })
        .addOnFailureListener(this, new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                // Failed to remove geofences
                // ...
            }
        });
```

https://developer.android.com/training/location/geofencing

Geofence example

https://github.com/latrobe-cs-educator/CSE2MAD_Lecture7_GeofenceDemo

Geocoder

Geocoding is the process of converting addresses (like a street address) into geographic coordinates (like latitude and longitude), which you can use to place markers on a map, or position the map.

```
String locationName = "La Trobe University, Bundoora VIC 3086";
geoAddresses = geocoder.getFromLocationName(locationName, 2);
```

Reverse geocoding is the process of converting geographic coordinates into a human-readable address.

```
revGeoAddresses = geocoder.getFromLocation(latitude, longitude, 5);
```

https://developer.android.com/reference/android/location/Geocoder

Geocoding example

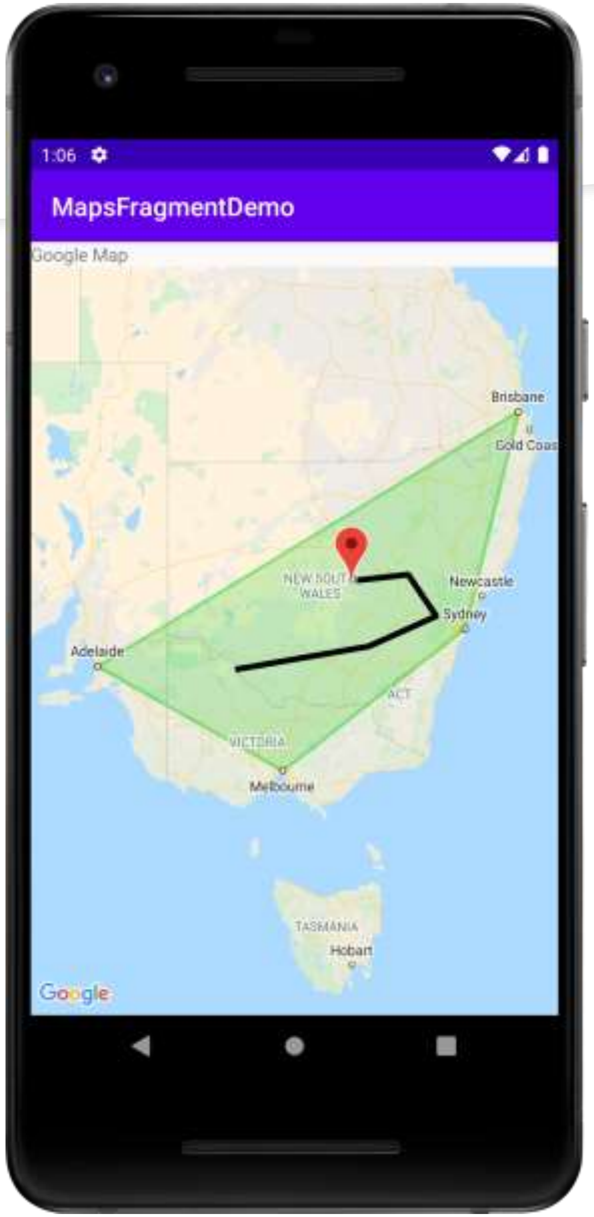
https://github.com/latrobe-cs-educator/CSE2MAD_Lecture7_GeoCodeDemo

Google Map Objects

Maps are represented in the API by the GoogleMap and MapFragment classes.

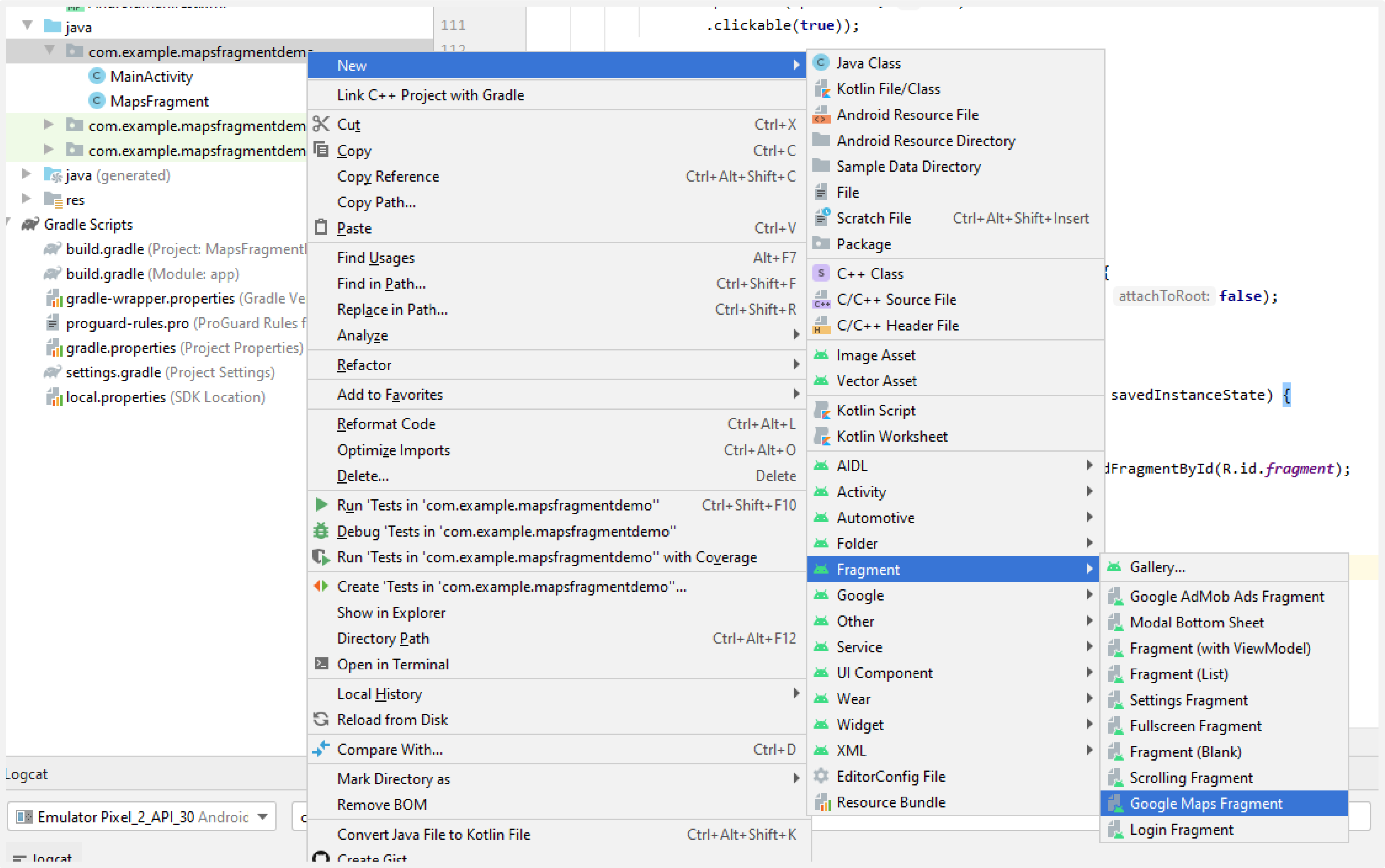
The basic steps for adding a map are:

1. (You only need to do this step once.) Follow the steps in the project configuration guide to get the API, obtain a key and add the required attributes to your Android manifest.
2. Add a Fragment object to the Activity that will handle the map. The easiest way to do this is to add a <fragment> element to the layout file for the Activity.
3. Implement the OnMapReadyCallback interface and use the onMapReady(GoogleMap) callback method to get a handle to the GoogleMap object. The GoogleMap object is the internal representation of the map itself. To set the view options for a map, you modify its GoogleMap object.
4. Call getMapAsync() on the fragment to register the callback.



https://developers.google.com/maps/documentation/android-sdk/map

Adding a Google Maps Fragment

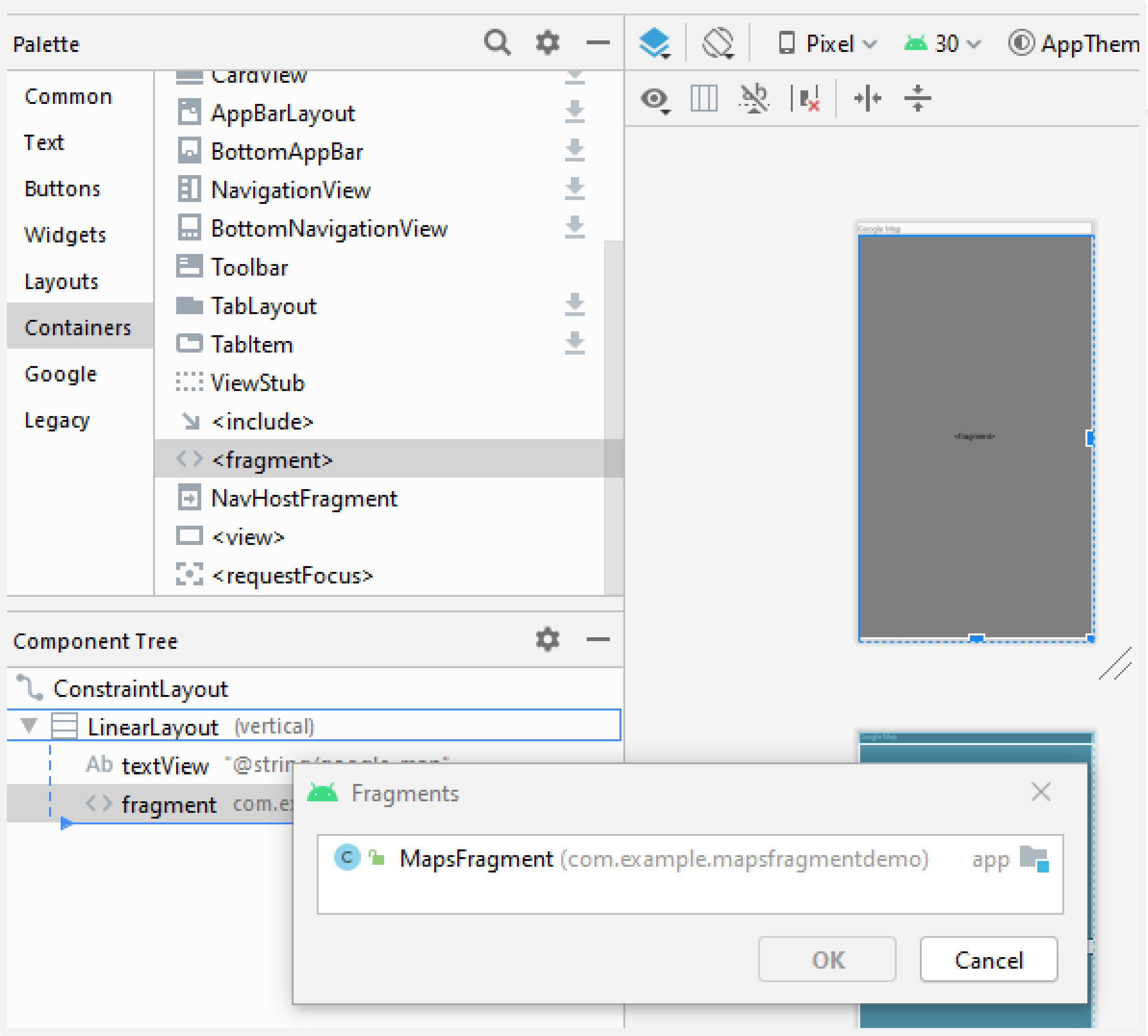

java
com.example.mapsfragmentdem
MainActivity
MapsFragment
com.example.mapsfragmentdem
com.example.mapsfragmentdem
java (generated)
res
Gradle Scripts
build.gradle (Project: MapsFragment
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Ve
proguard-rules.pro (ProGuard Rules f
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
Logcat
Emulator Pixel_2_API_30 Android
logcat
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy Ctrl+C
Copy Reference Ctrl+Alt+Shift+C
Copy Path...
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Run 'Tests in 'com.example.mapsfragmentdemo'' Ctrl+Shift+F10
Debug 'Tests in 'com.example.mapsfragmentdemo''
Run 'Tests in 'com.example.mapsfragmentdemo'' with Coverage
Create 'Tests in 'com.example.mapsfragmentdemo''...
Show in Explorer
Directory Path Ctrl+Alt+F12
Open in Terminal
Local History
Reload from Disk
Compare With... Ctrl+D
Mark Directory as
Remove BOM
Convert Java File to Kotlin File Ctrl+Alt+Shift+K
Create Gist
Java Class
Kotlin File/Class
Android Resource File
Android Resource Directory
Sample Data Directory
File
Scratch File Ctrl+Alt+Shift+Insert
Package
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset
Kotlin Script
Kotlin Worksheet
AIDL
Activity
Automotive
Folder
Fragment
Google
Other
Service
UI Component
Wear
Widget
XML
EditorConfig File
Resource Bundle
Gallery...
Google AdMob Ads Fragment
Modal Bottom Sheet
Fragment (with ViewModel)
Fragment (List)
Settings Fragment
Fullscreen Fragment
Fragment (Blank)
Scrolling Fragment
Google Maps Fragment
Login Fragment

```
.clickable(true));
attachToRoot: false);
savedInstanceState) {
dFragmentById(R.id.fragment);
```

1. The above shortcut will generates your MapsFragment.java, fragment_maps.xml & google_maps_api.xml (where you put your API key) .

Adding a Google MapFragment

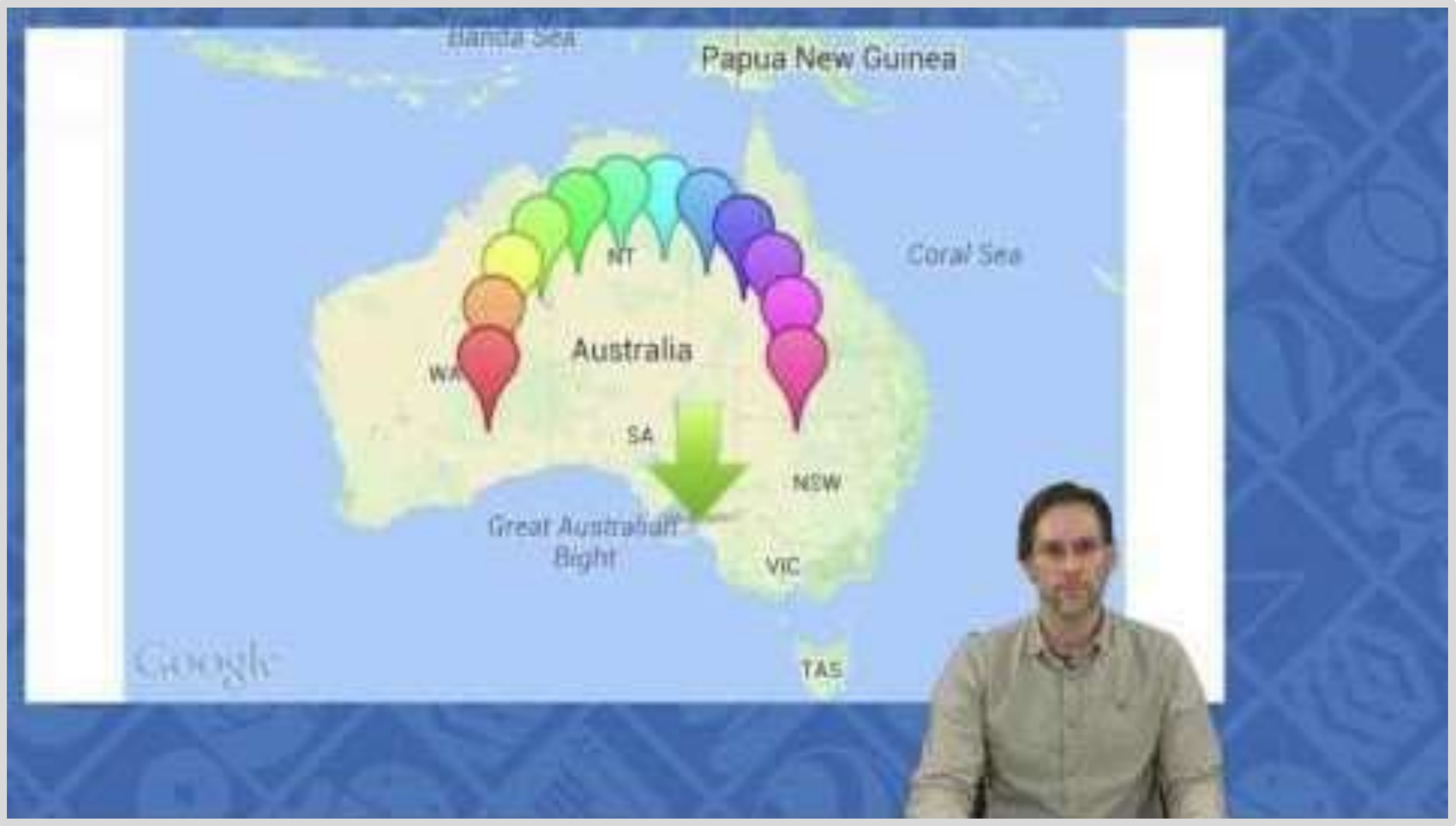
2. Then you drag the fragment into your Activity layout, selecting your MapsFragment when prompted.



3. And of course add google play services to your app level build.gradle file

```
implementation 'com.google.android.gms:play-services-maps:17.0.0'
```

Markers

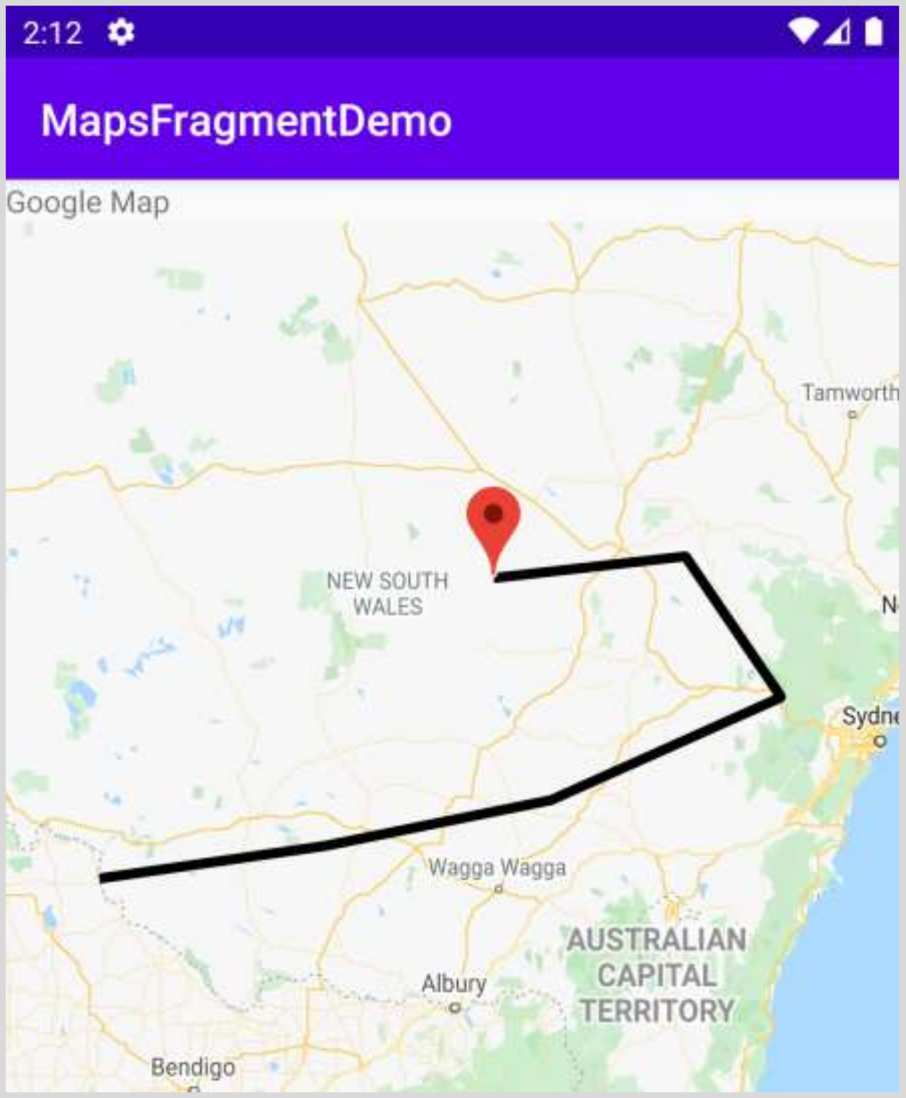


https://youtu.be/l3bw8Senjmg

Polylines

The Polyline class defines a set of connected line segments on the map. A Polyline object consists of a set of LatLng locations, and creates a series of line segments that connect those locations in an ordered sequence.

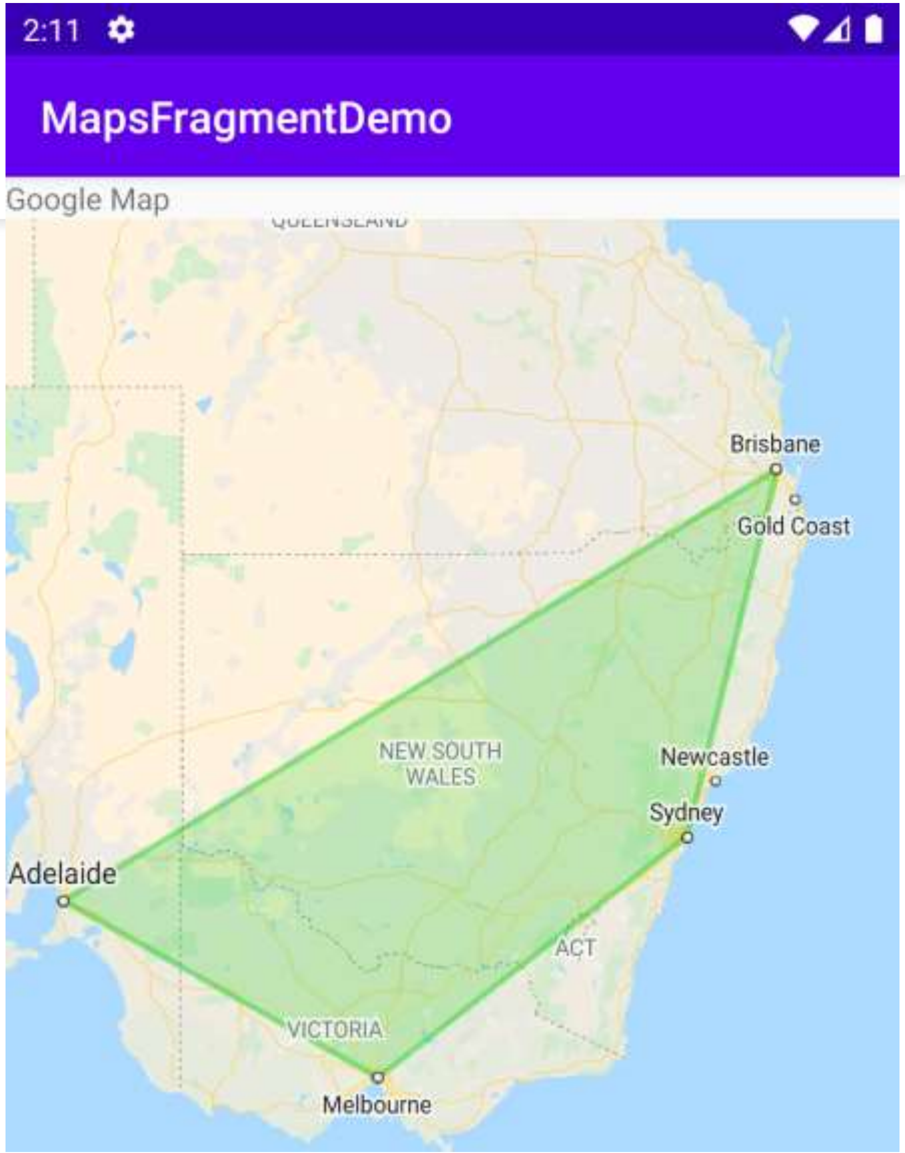
```
Polyline polyline1 = googleMap.addPolyline(new PolylineOptions()
        .clickable(true)
        .add(
                new LatLng(-35.016, 143.321),
                new LatLng(-34.747, 145.592),
                new LatLng(-34.364, 147.891),
                new LatLng(-33.501, 150.217),
                new LatLng(-32.306, 149.248),
                new LatLng(-32.491, 147.309)));
```



Polygons

Polygon objects are similar to Polyline objects in that they consist of a series of coordinates in an ordered sequence. However, instead of being open-ended, polygons are designed to define regions within a closed loop with the interior filled in.

```
// Add polygons to indicate areas on the map.
Polygon polygon1 = googleMap.addPolygon(new PolygonOptions()
        .clickable(true)
        .add(
                new LatLng(-27.457, 153.040),
                new LatLng(-33.852, 151.211),
                new LatLng(-37.813, 144.962),
                new LatLng(-34.928, 138.599)));

polygon1.setFillColor(argb(50, 0, 204, 0));
polygon1.setStrokeWidth(6);
polygon1.setStrokeColor(argb(100, 0, 204, 0));
```

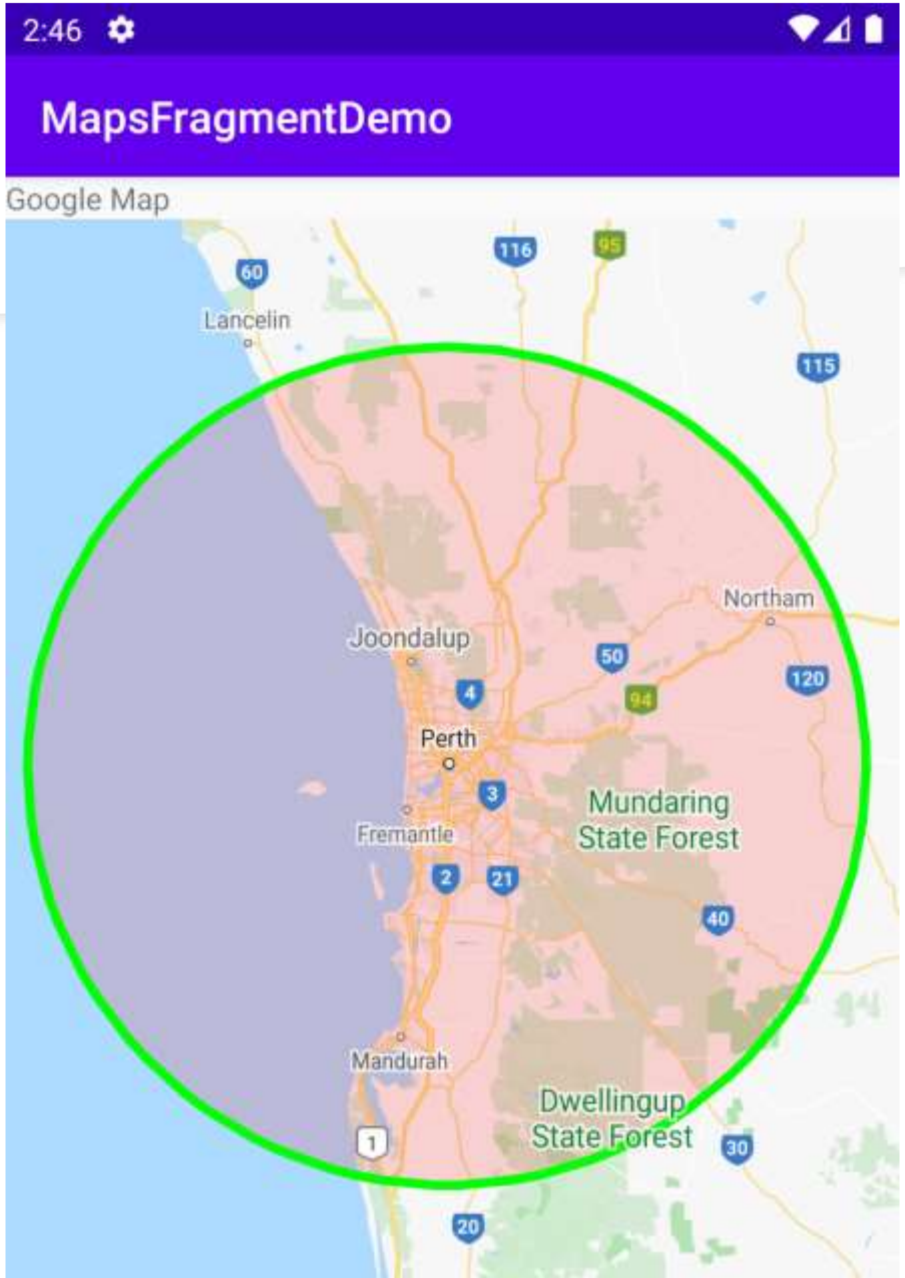


https://developers.google.com/maps/documentation/android-sdk/shapes

Circles

In addition to a generic Polygon class, the Maps API also includes specific classes for Circle objects, to simplify their construction.

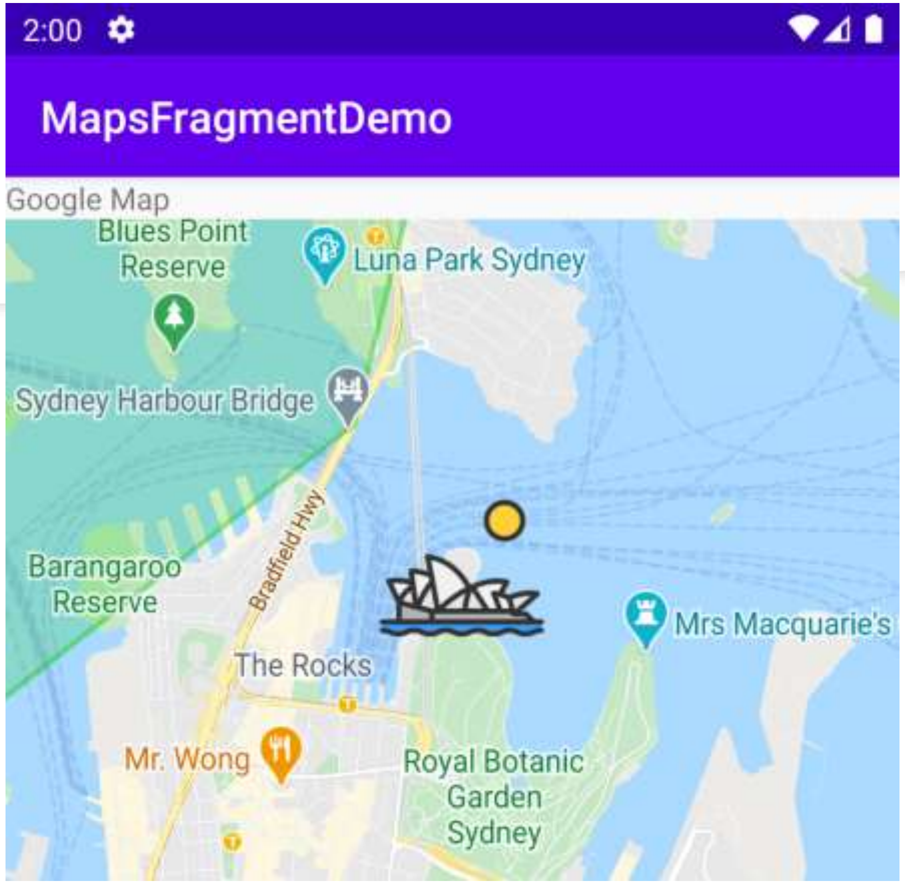
```
LatLng perth = new LatLng(-31.953570, 115.856978);
Circle circle = googleMap.addCircle(new CircleOptions()
        .center(perth)
        .radius(100000)
        .strokeWidth(10)
        .strokeColor(Color.GREEN)
        .fillColor(Color.argb(40, 255, 0, 0))
        .clickable(true));
```



https://developers.google.com/maps/documentation/android-sdk/shapes

Ground Overlays

Ground overlays are image overlays that are tied to latitude/longitude coordinates, so they move when you drag or zoom the map.

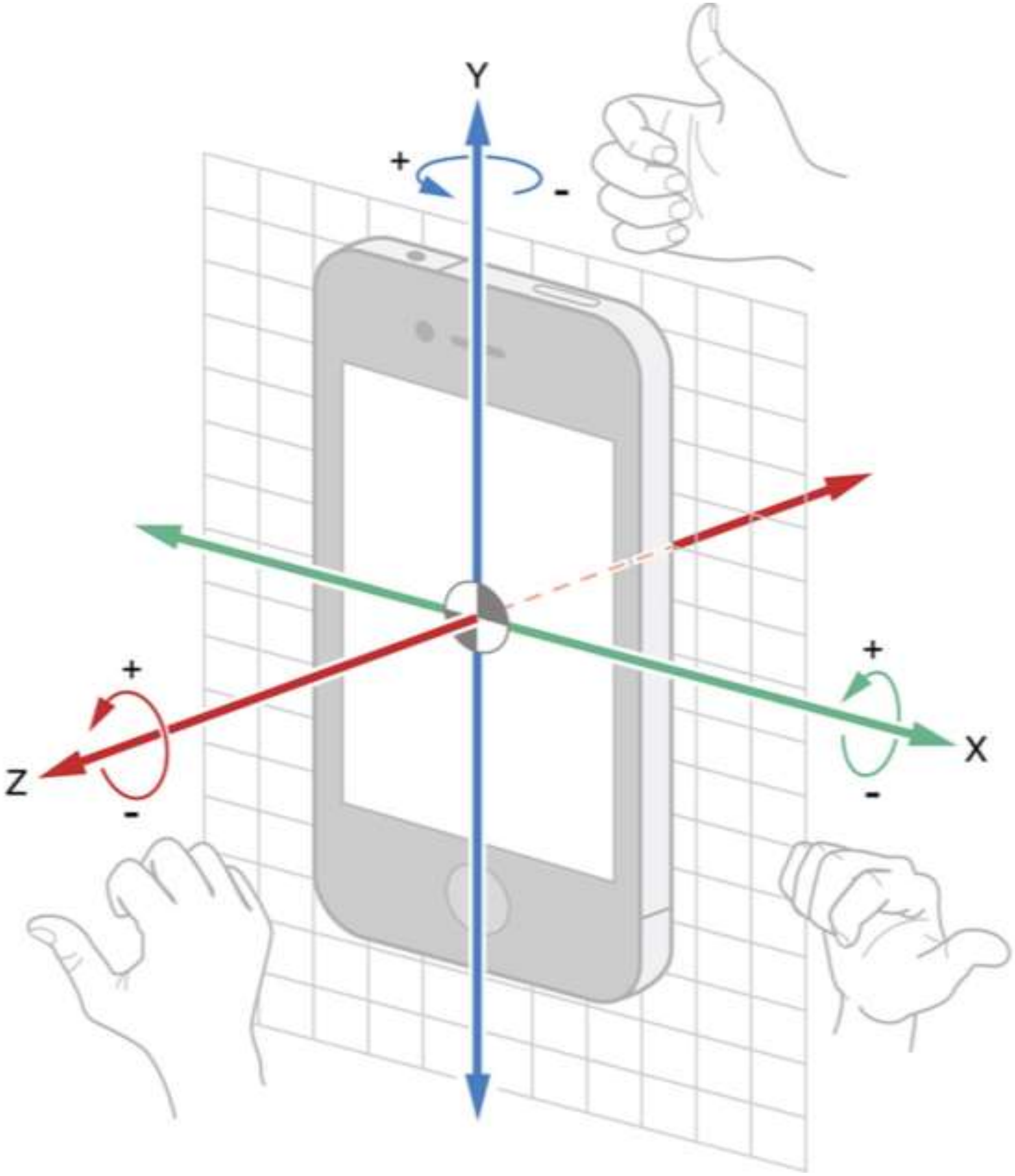


```
LatLng operahouse = new LatLng(-33.856732, 151.215227);
GroundOverlay sydneyGroundOverlay = googleMap.addGroundOverlay(new GroundOverlayOptions()
        .image(BitmapDescriptorFactory.fromResource(R.drawable.operahouse))
        .position(operahouse, 600f)
        .clickable(true));
sydneyGroundOverlay.setTag("Sydney");
```

https://developers.google.com/maps/documentation/android-sdk/groundoverlay

Map Fragment example

https://github.com/latrobe-cs-educator/CSE2MAD_Lecture7_MapsFragmentDemo



Android Sensing

Android Sensors Overview

Main 3 sensor categories in Android:

1. **Motion sensors**
   These sensors measure acceleration forces and rotational forces along three axes. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors.

2. **Environmental sensors**
   These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. This category includes barometers, photometers, and thermometers.

3. **Position sensors**
   These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers.

You can access sensors available on the device and acquire raw sensor data by using the Android sensor framework.

https://developer.android.com/guide/topics/sensors/sensors_overview

Android sensor framework: Relevant main classes

SensorManager
https://developer.android.com/reference/android/hardware/SensorManagerSensorEvent

Sensor
https://developer.android.com/reference/android/hardware/Sensor

SensorEvent
https://developer.android.com/reference/android/hardware/SensorEvent

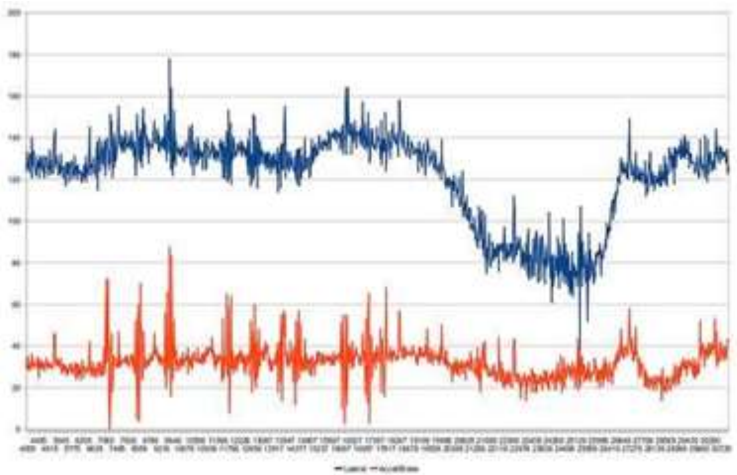
SensorEventListener
https://developer.android.com/reference/android/hardware/SensorEventListener

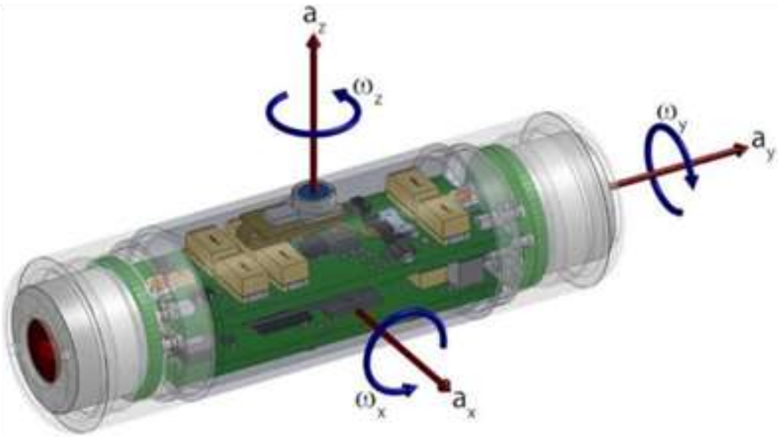
https://developers.google.com/maps/documentation/android-sdk/groundoverlay

Android sensor framework

In a typical application you use these sensor-related APIs to perform two basic tasks:

- Identify available sensors and their capabilities
- Monitor sensor events





https://developer.android.com/guide/topics/sensors/sensors_overview#sensor-framework

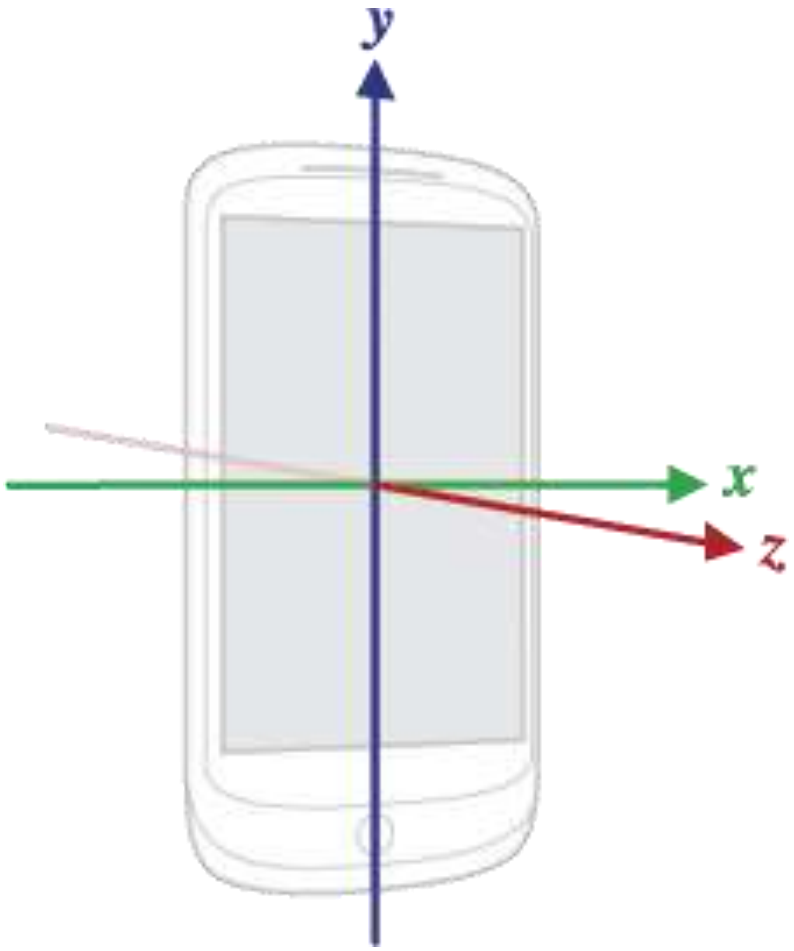
Sensor Coordinate System

Uses a standard 3-axis coordinate system

- This system is used by:
- Acceleration sensor
- Gravity sensor Gyroscope
- Linear acceleration sensor
- Geomagnetic field sensor

Axes are not swapped when the device's screen orientation changes

Similar to the behavior of the OpenGL coordinate system.



https://developer.android.com/guide/topics/sensors/sensors_overview#sensors-coords

Using the Accelerometer

An acceleration sensor measures the acceleration applied to the device, including the force of gravity. The following code shows you how to get an instance of the default acceleration sensor:

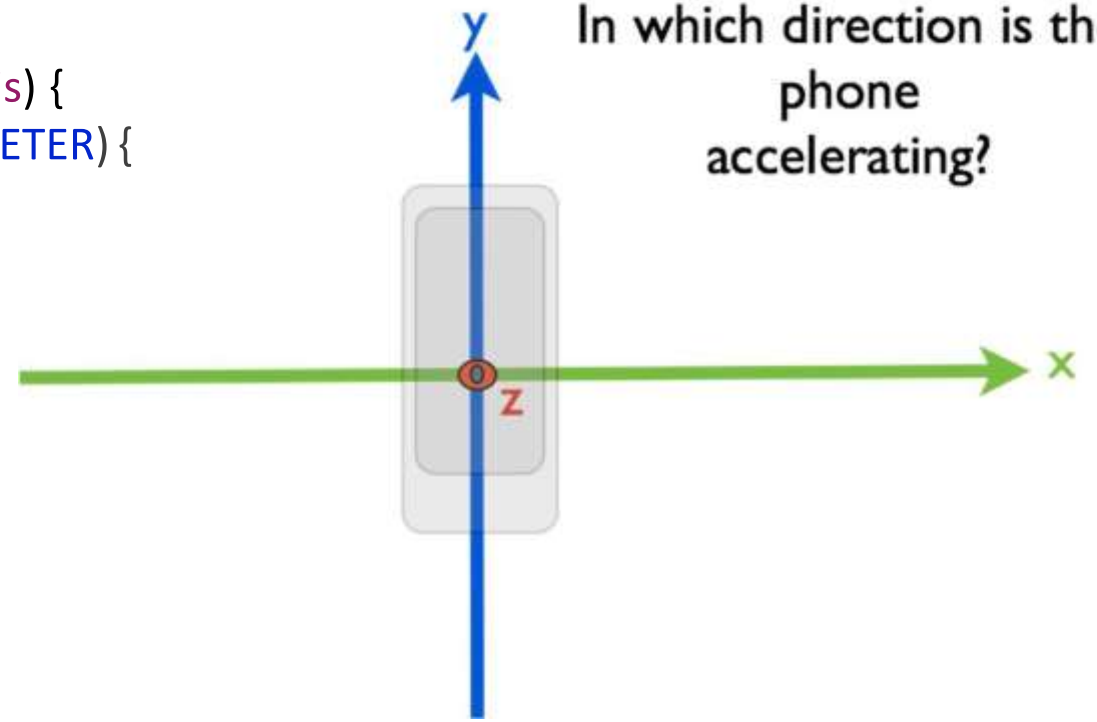
```
private SensorManager sensorManager;
private Sensor sensor;
  ...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

https://developer.android.com/guide/topics/sensors/sensors_overview#sensors-coords

Acceleration, measured via the Accelerometer

Accelerometer readings:

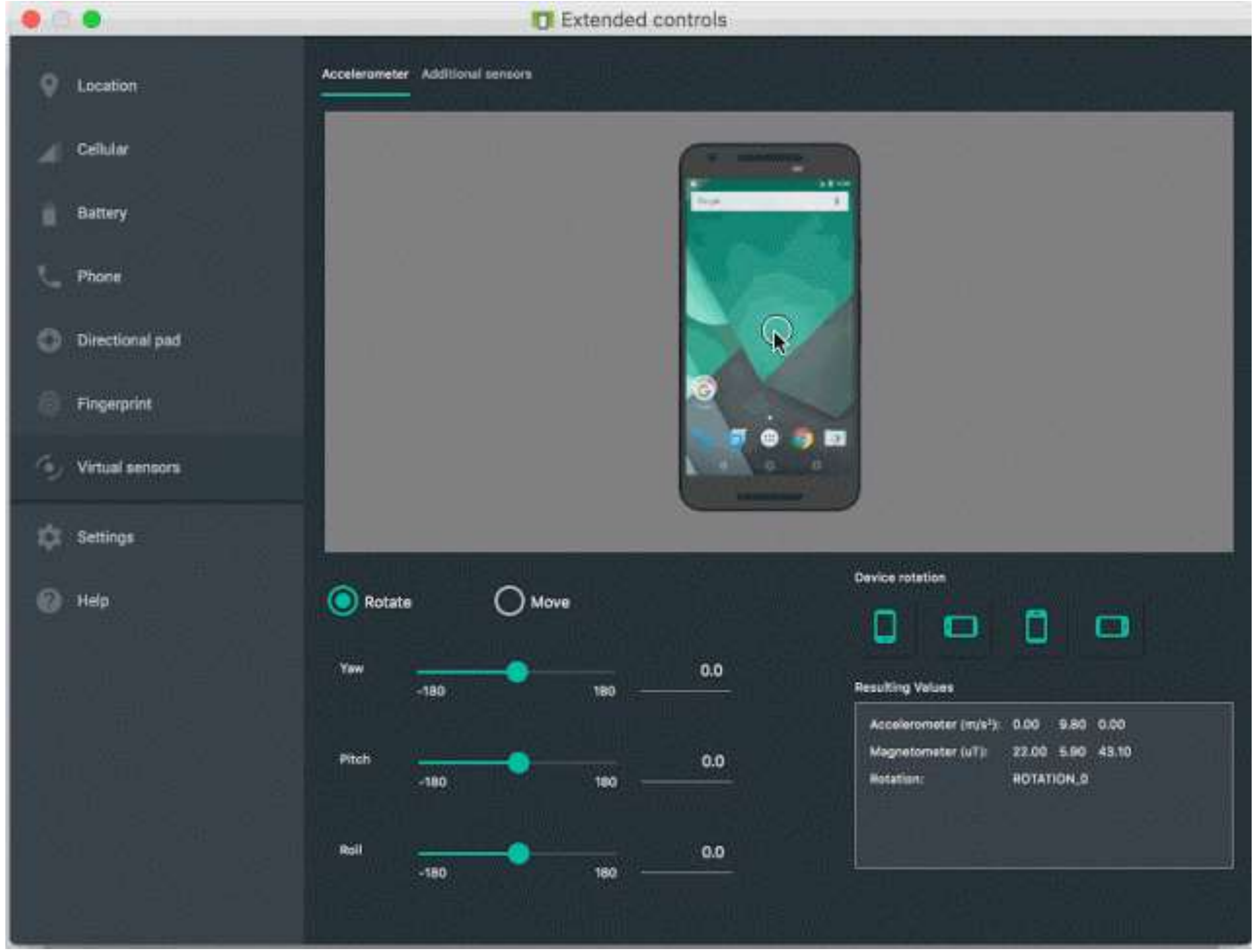
```
public void onSensorChanged(SensorEvent event) {  synchronized (this) {
                if (event.sensor.getType() ==  Sensor.TYPE_ACCELEROMETER) {
                  // acceleration of phone along x-axis
                  xValueTV.setText(Float.toString(event.values[0]));

                  // acceleration of phone along y-axis
                  yValueTV.setText(Float.toString(event.values[1]));
                  // acceleration of phone along z-axis
                  zValueTV.setText(Float.toString(event.values[2]));
                  }
      ..
      }
```



In which direction is the phone accelerating?

https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-accel

Test with the Android Emulator



● Rotate ○ Move

Yaw 0.0

Pitch 0.0

Roll 0.0

Resulting Values

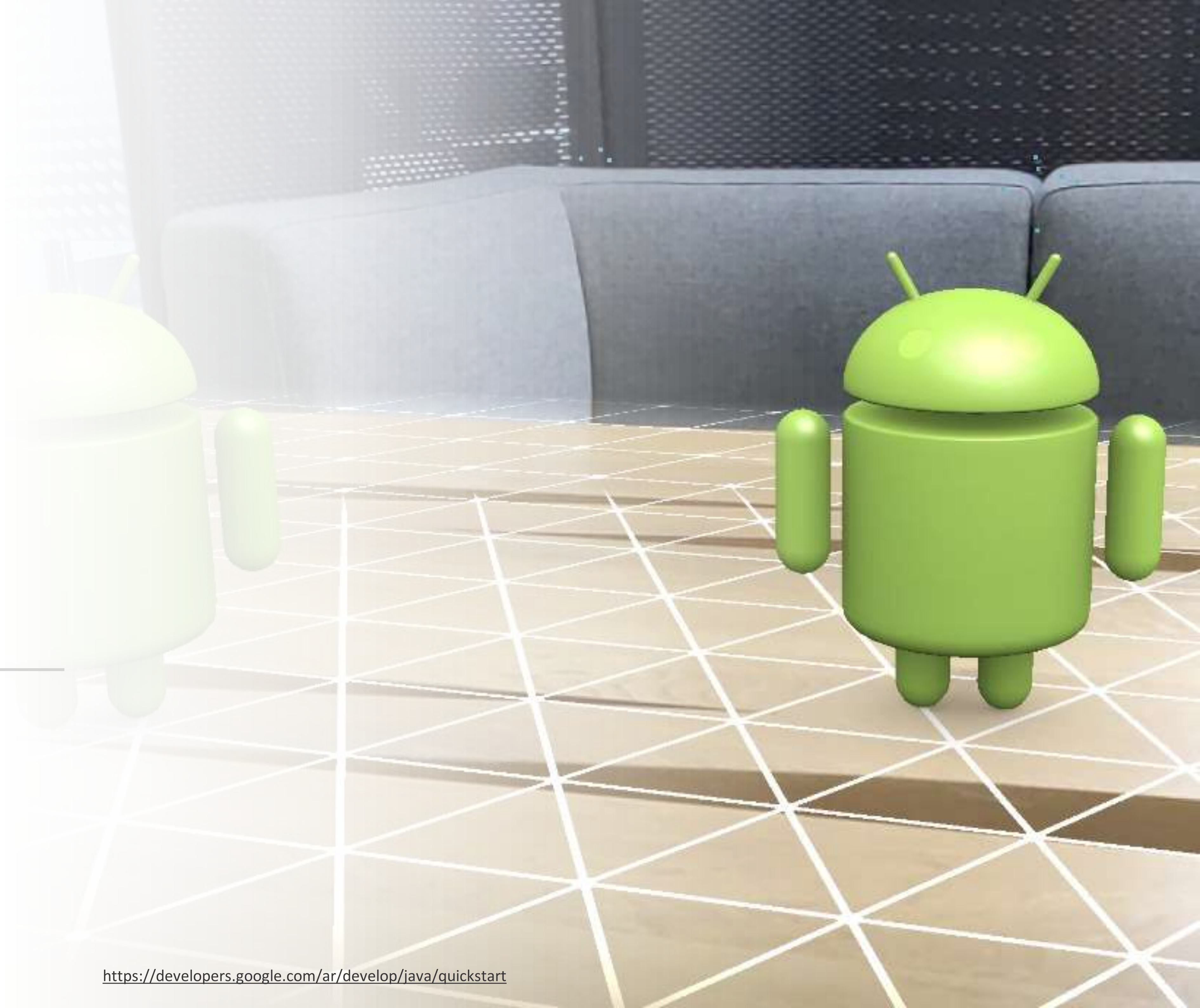
```
Accelerometer (m/s²):  0.00   9.80  0.00
Magnetometer (uT):     22.00  5.90  43.10
Rotation:              ROTATION_0
```

The Android Emulator includes a set of virtual sensor controls that allow you to test sensors such as accelerometer, ambient temperature, magnetometer, proximity, light, and more.

https://developer.android.com/guide/topics/sensors/sensors_overview#test-with-the-android-emulator

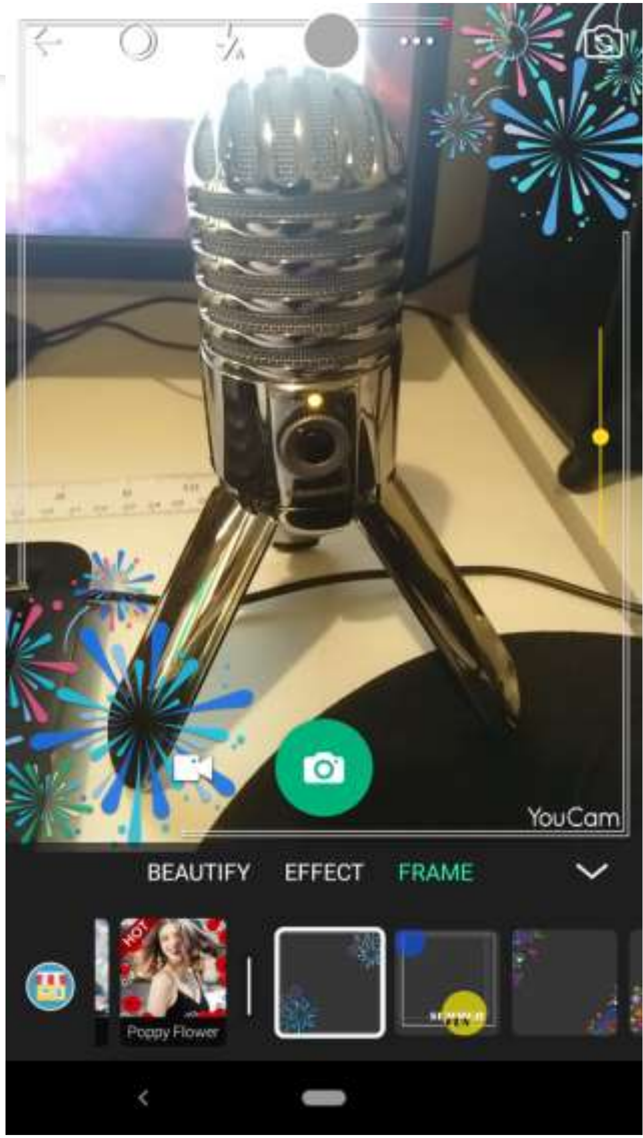
Using the Camera

- Augmented Reality
- Take Photos
- Record Video



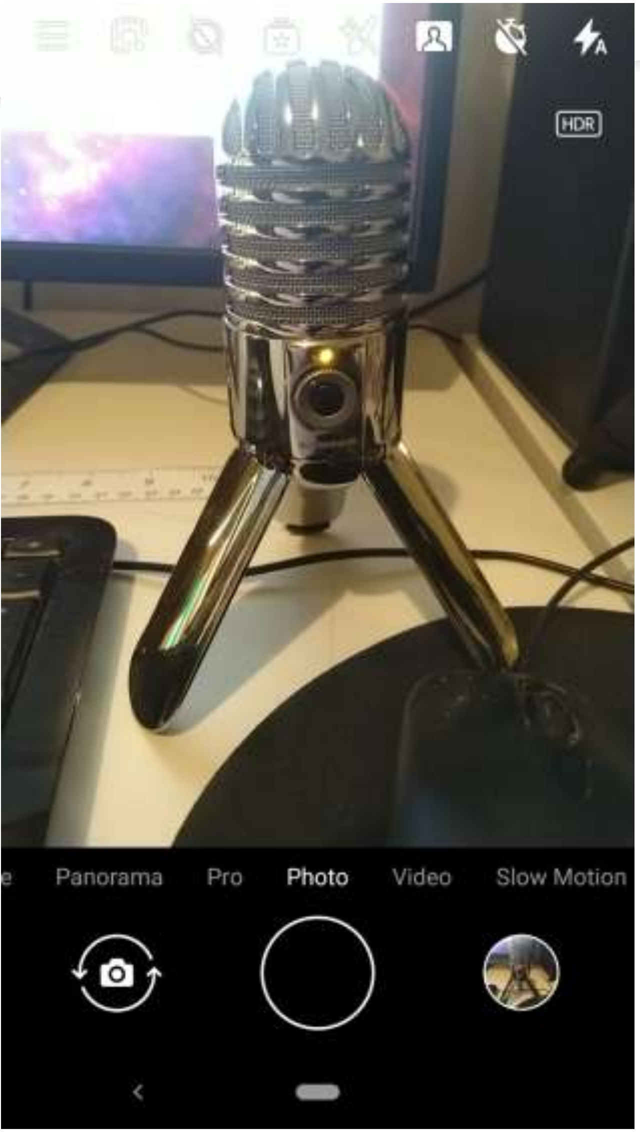
https://developers.google.com/ar/develop/java/quickstart

Write a Camera app from Scratch



Using the Camera

Use the existing Camera app



https://developer.android.com/guide/topics/media/camera

Using the Camera

Ask:

1. How important is the Camera functionality to your app?
   declare the camera requirement in the app manifest?

2. Where will you store the photos or videos generated by your camera app?
   Only accessed by your app? Or to be shared with other apps?

3. How will your application use the camera?
   Are you just interested in snapping a quick picture or video clip, or will your application provide a new way to use cameras?

4. When does your app interact with the camera?
   On Android 9 (API level 28) and later, apps running in the background cannot access the camera. Therefore, you should use the camera either when your app is in the foreground or as part of a foreground service.

https://developer.android.com/guide/topics/media/camera

Camera: Relevant Classes

PACKAGE: android.hardware.camera2 This is the new API
Camera: android.hardware.Camera deprecated in API level 21
SurfaceView: android.view.SurfaceView
MediaRecorder: android.media.MediaRecorder
Intent: android.content.Intent

https://developer.android.com/guide/topics/media/camera#basics

Using the Existing Camera app
- what to include in the Manifest

Manifest <uses-feature> :
If you’re just using an **existing** camera app, you don’t need permission.

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

But you will still need to indicate in the Manifest that your app is using the camera feature.

```
<uses-feature android:name="android.hardware.camera" />
```

https://developer.android.com/guide/topics/media/camera#basics

Using the Existing Camera app - Feature based filtering

```
<uses-feature android:name="android.hardware.camera"
android:required="true" />
```

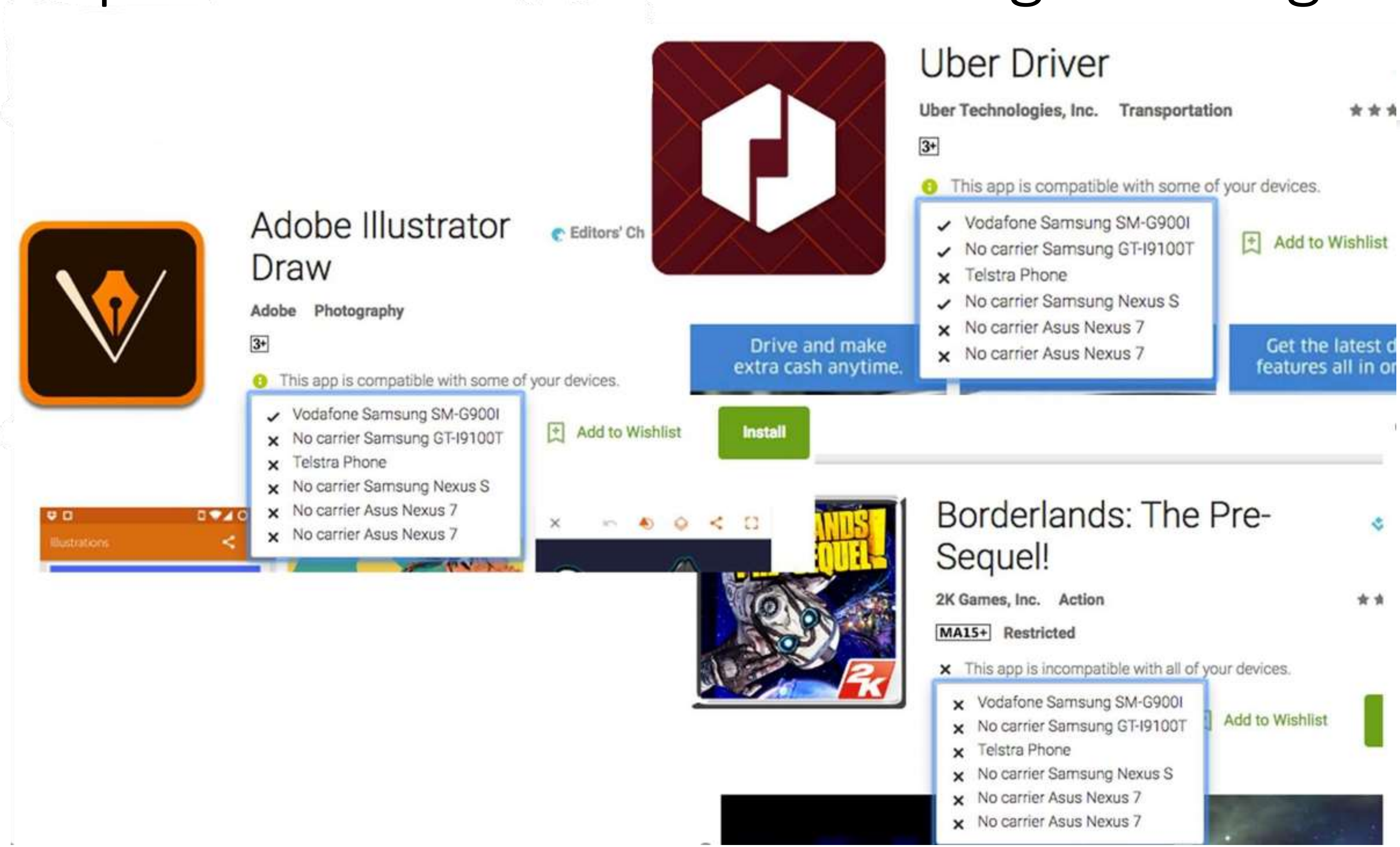
- If a feature is declared as being required, Google Play adds the feature to the list of required features for the application. It then filters the application from users on devices that do not provide that feature.

```
<uses-feature android:name="android.hardware.camera"
android:required=“false" />
```

- If required = false, Google Play does not add the feature to the list of required features. It's then your responsibility to check for the availability of the camera at runtime.
- If a feature is explicitly declared, but without an android:required attribute, Google Play assumes that the feature is required and sets up filtering on it.

https://developer.android.com/guide/topics/manifest/uses-feature-element.html#market-feature-%20%20filtering

Examples for feature-based filtering in Google Play



Adobe Illustrator Draw

Editors' Ch

Adobe Photography

3+

This app is compatible with some of your devices.

- ✓ Vodafone Samsung SM-G900I
- ✗ No carrier Samsung GT-I9100T
- ✗ Telstra Phone
- ✗ No carrier Samsung Nexus S
- ✗ No carrier Asus Nexus 7
- ✗ No carrier Asus Nexus 7

Uber Driver

Uber Technologies, Inc. Transportation

3+

This app is compatible with some of your devices.

- ✓ Vodafone Samsung SM-G900I
- ✓ No carrier Samsung GT-I9100T
- ✗ Telstra Phone
- ✓ No carrier Samsung Nexus S
- ✗ No carrier Asus Nexus 7
- ✗ No carrier Asus Nexus 7

Add to Wishlist
Drive and make extra cash anytime.
Get the latest d features all in or

Borderlands: The Pre-Sequel!

2K Games, Inc. Action

MA15+ Restricted

✗ This app is incompatible with all of your devices.

- ✗ Vodafone Samsung SM-G900I
- ✗ No carrier Samsung GT-I9100T
- ✗ Telstra Phone
- ✗ No carrier Samsung Nexus S
- ✗ No carrier Asus Nexus 7
- ✗ No carrier Asus Nexus 7

https://developer.android.com/guide/topics/manifest/uses-feature-element.html#market-feature-%20%20filtering

Using the Existing Camera app: Take a Photo

Use Intents to get other apps (eg: existing Camera app) to do work for our app!

build an Intent:

```
Intent pictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
```

Make sure the intent can be supported, and if yes, start the Intent

```
if (pictureIntent.resolveActivity(getPackageManager()) != null) {

   startActivityForResult(pictureIntent, REQUEST_CAPTURE_IMAGE);
```

https://developer.android.com/guide/topics/media/camera#basics

Using the Existing Camera app: Take a Photo

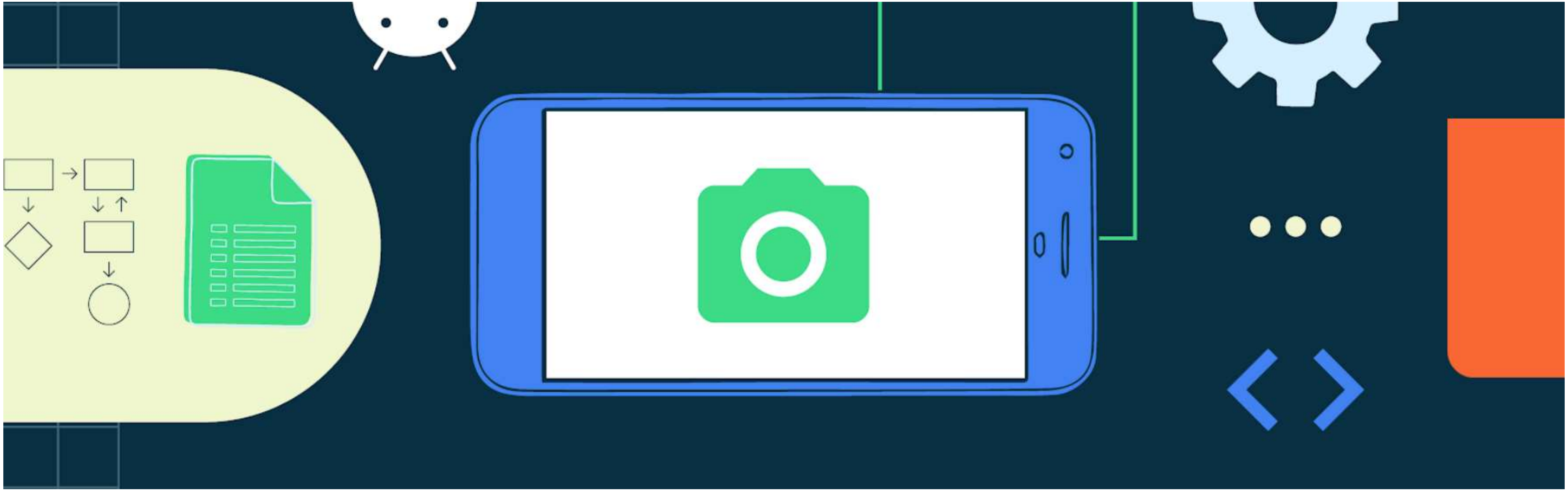
- Get the Thumbnail

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
        Bundle extras = data.getExtras();
        Bitmap imageBitmap = (Bitmap) extras.get("data");
        mImageView.setImageBitmap(imageBitmap);
    }
}
```

https://developer.android.com/guide/topics/media/camera#basics

Build a Custom Camera

- Camera (deprecated) API 21
- Camera2 (android.hardware.camera2)
- CameraX (Beta)



https://developer.android.com/reference/android/hardware/camera2/package-summary

Build a Camera with Android Camera 2

https://developer.android.com/reference/android/hardware/camera2/package-summary https://youtu.be/Xtp3tH27OFs

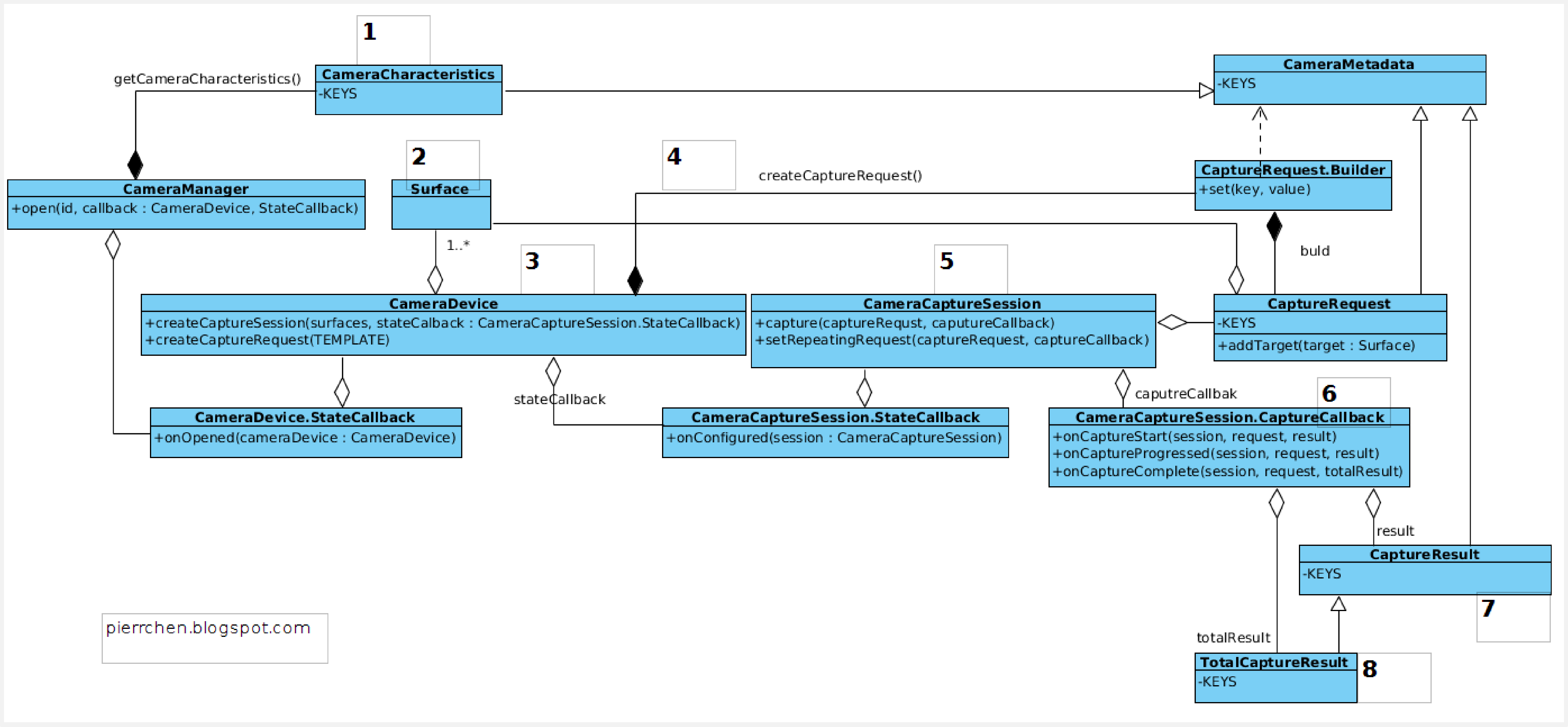
Build a Camera with Android Camera 2

Replaces the deprecated Camera class

1. Start from CameraManager: get all camera devices in the device, each with its own ID
2. Using the ID, get properties of the each camera device.
3. Setup the output targets. Eg: SurfaceView or SurfaceTexture for preview, ImageReader for still picture or MediaRecoder for video recording
4. Get a CameraDevice. CameraManager.open(ID)
5. Create a CaptureRequest from the CameraDevice. createCaptureRequest(CameraDevice.TEMPLATE_PREVIEW)
6. Create a CaptureRequestSession from the CameraDevice to capture or stream images. This capture session can be either a one-shot capture or an endlessly repeating one.
7. Construct a CaptureRequest, which defines all the capture parameters needed by a camera device to capture a single image. Submit this CaptureRequest to CaptureRequestSession.
8. Get the Capture Results.

https://inducesmile.com/android/android-camera2-api-example-tutorial/
https://pierrchen.blogspot.com/2015/01/android-camera2-api-explained.html
https://developer.android.com/reference/android/hardware/camera2/package-summary.html

Build a Camera with Android Camera 2



CameraCharacteristics
-KEYS

CameraMetadata
-KEYS

CameraManager
+open(id, callback : CameraDevice, StateCallback)

Surface

CaptureRequest.Builder
+set(key, value)

CameraDevice
+createCaptureSession(surfaces, stateCalback : CameraCaptureSession.StateCallback)
+createCaptureRequest(TEMPLATE)

CameraCaptureSession
+capture(captureRequst, caputureCallback)
+setRepeatingRequest(captureRequest, captureCallback)

CaptureRequest
-KEYS
+addTarget(target : Surface)

CameraDevice.StateCallback
+onOpened(cameraDevice : CameraDevice)

CameraCaptureSession.StateCallback
+onConfigured(session : CameraCaptureSession)

CameraCaptureSession.CaptureCallback
+onCaptureStart(session, request, result)
+onCaptureProgressed(session, request, result)
+onCaptureComplete(session, request, totalResult)

CaptureResult
-KEYS

TotalCaptureResult
-KEYS

https://pierrchen.blogspot.com/2015/01/android-camera2-api-explained.html

Intro to CameraX Beta

https://medium.com/@akhilbattula/android-camerax-java-example-aeee884f9102

https://youtu.be/QYkTXJ2TuiA

Android Audio

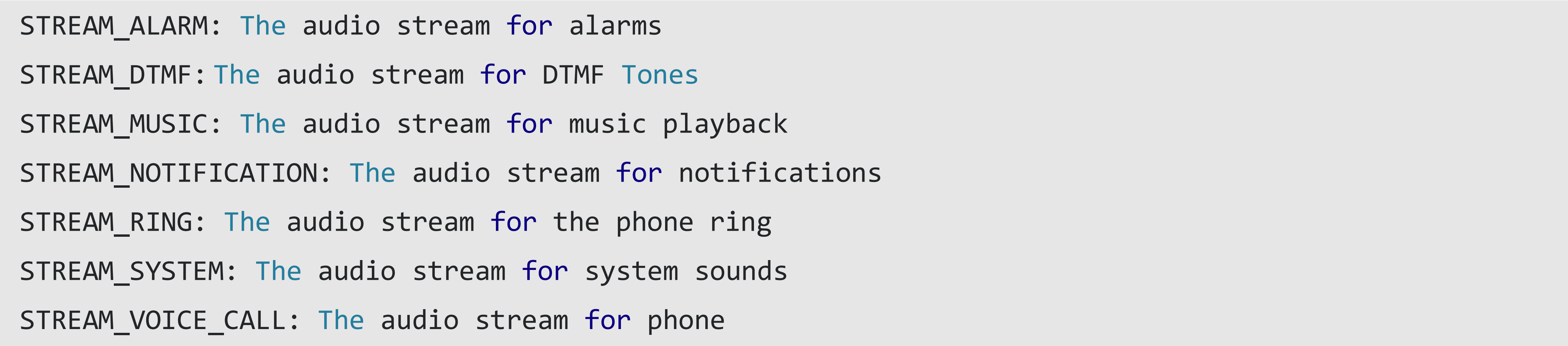
- start, stop, pause, resume playing of audio files.
- Manage audio focus
- Manage audio output hardware
- Android APIs:

  SoundPool

  MediaPlayer

Android Audio : Control App’s Volume and Playback

Identify Which Audio Stream to Use. Typical : STREAM_MUSIC



```
STREAM_ALARM: The audio stream for alarms
STREAM_DTMF:The audio stream for DTMF Tones
STREAM_MUSIC: The audio stream for music playback
STREAM_NOTIFICATION: The audio stream for notifications
STREAM_RING: The audio stream for the phone ring
STREAM_SYSTEM: The audio stream for system sounds
STREAM_VOICE_CALL: The audio stream for phone
```

Eg:
STREAM_MUSIC - sound will be produced through one audio device (phone speaker, earphone, bluetooth speaker or something else).

STREAM_RING - sound will be produced through all audio device connected to the phone.
This behaviour might be differed for each devices.

https://developer.android.com/guide/topics/media-apps/volume-and-earphones#volume-controls

Android Audio : Control App's Volume and Playback

Having identified the audio stream your application will be using, you should set it as the volume stream target

```
setVolumeControlStream(AudioManager.STREAM_MUSIC);
```

Set this inside onCreate()

Now, pressing volume keys affect the audio stream specified whenever the target activity or fragment is visible.

https://developer.android.com/reference/android/app/Activity#setVolumeControlStream(int)

Android Audio : Control App’s Volume and Playback

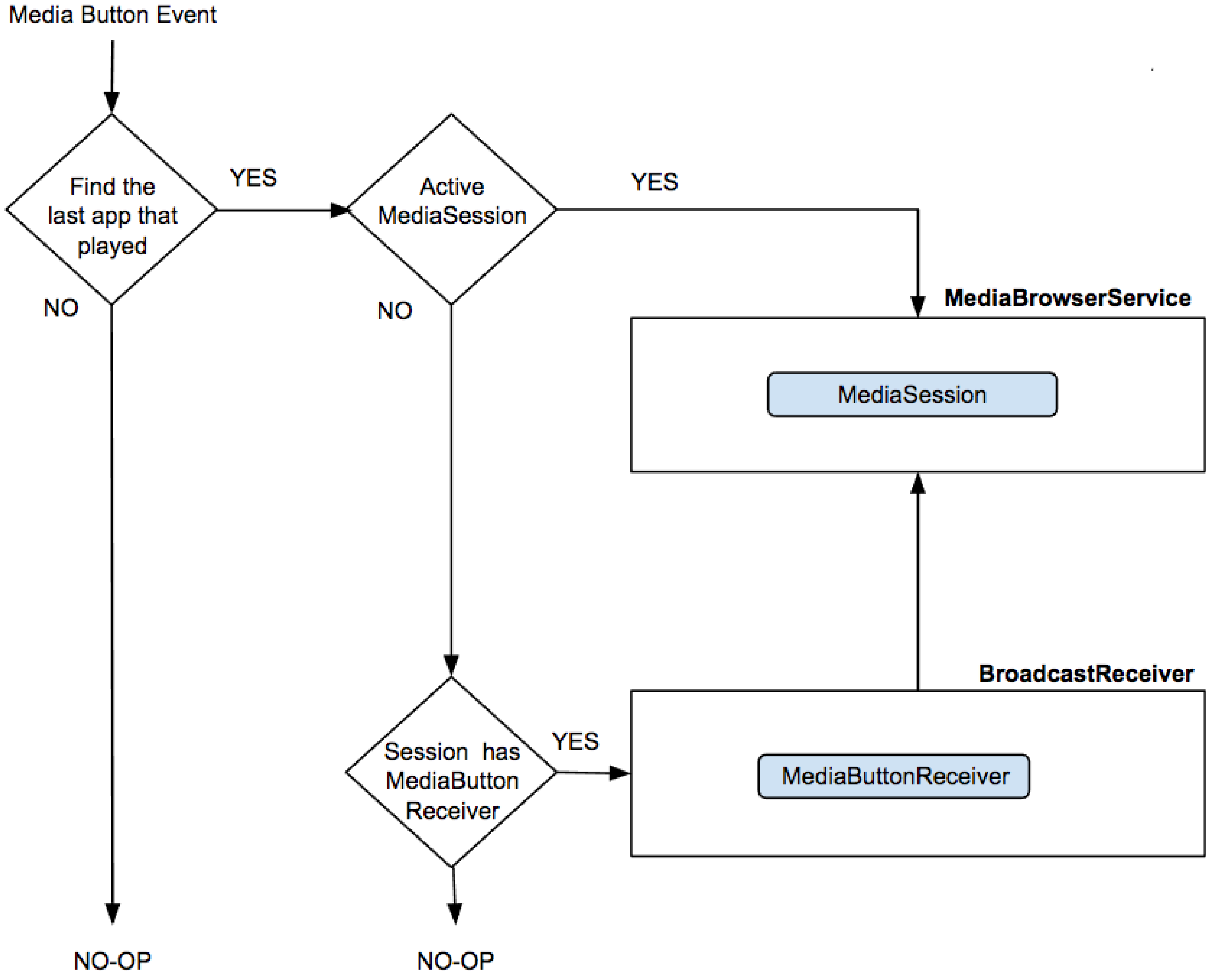
Media playback buttons (h/w): play, pause, stop, skip, previous

Apps should be able to handle media button events in three cases, in this order of priority:

1. When the app's UI activity is visible
2. When the UI activity is hidden and the app's media session is active
3. When the UI activity is hidden and app's media session is inactive and needs to be restarted

https://developer.android.com/guide/topics/media-apps/mediabuttons

If the foreground activity does not handle the event, Android will try to find a media session that can handle it.



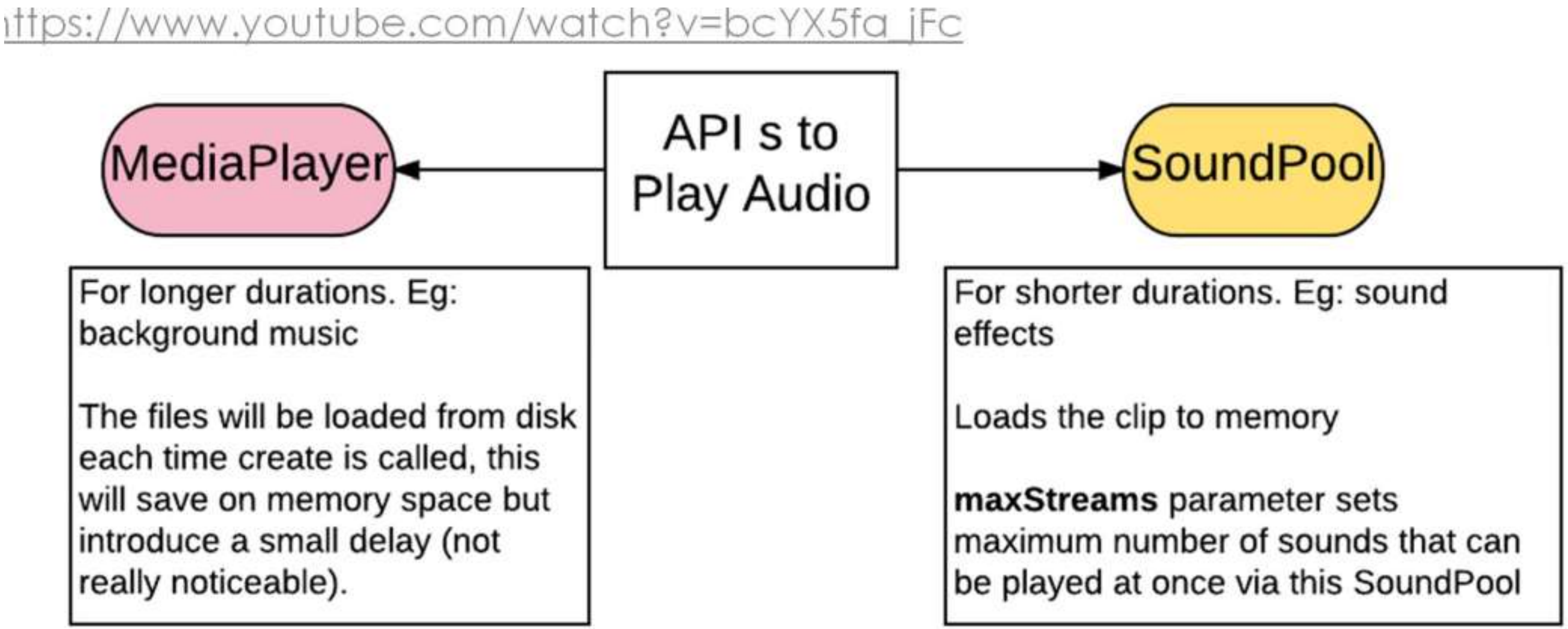
Android Audio: Resources

- res/raw directory
- assets/ directory
  - reference as file://andorid_asset in a URI
  - can share with other apps via URI (file provider)

- Store media in application local directory, pull from network / web and store locally
- Store and use media on SD card
- Stream via the Internet

https://www.vogella.com/tutorials/AndroidMedia/article.html

Android Audio: Resources



ıttps://www.youtube.com/watch?v=bcYX5fa_jFc

For longer durations. Eg: background music

The files will be loaded from disk each time create is called, this will save on memory space but introduce a small delay (not really noticeable).

For shorter durations. Eg: sound effects

Loads the clip to memory

maxStreams parameter sets maximum number of sounds that can be played at once via this SoundPool

https://developer.android.com/guide/topics/media/mediaplayer
https://developer.android.com/reference/android/media/SoundPool

Soundpool Demo

https://github.com/latrobe-cs-educator/CSE2MAD_Lecture7_SoundpoolDemo

Image Processing

- an image is basically an array of pixels, each pixel represented by a set of bytes
- http://developer.android.com/reference/android/graphics/Color.html
- operations on an image are operations with the pixel information

https://developer.android.com/reference/android/media/Image

https://en.wikipedia.org/wiki/The_Matrix

Image Processing: APIs

- OpenCV on Android

  http://opencv.org/platforms/android.html, uses JNI
- Android/Google APIs:
- android.media.FaceDetector class

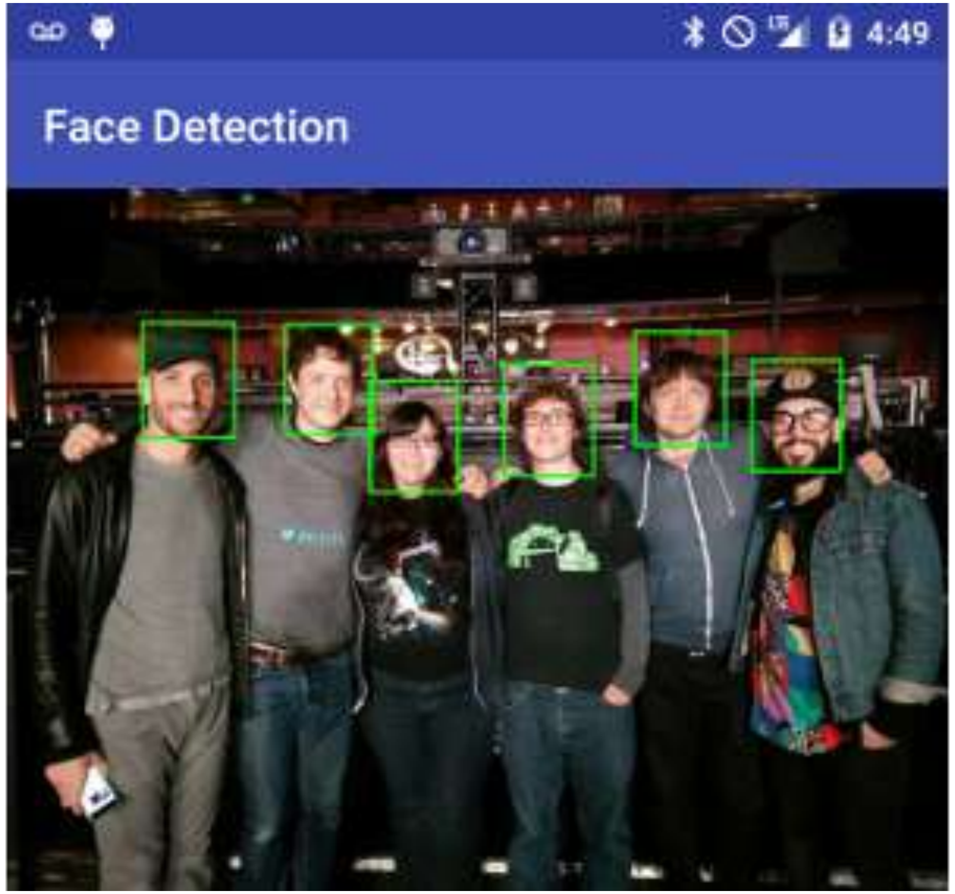
  http://developer.android.com/reference/android/media/FaceDetector.html
- Mobile Vision APIs in Google Play (newer)

  https://developers.google.com/vision/
- Face API: distinguish faces at different orientations and with different facial features facial expressions.
- Barcode Scanner API
- Text Recognition API

https://www.vogella.com/tutorials/AndroidMedia/article.html

Image Processing: android.media.FaceDetector

```
FaceDetector detector = new FaceDetector.Builder( getContext() )
        .setTrackingEnabled(false)
        .setLandmarkType(FaceDetector.ALL_LANDMARKS)
        .setMode(FaceDetector.FAST_MODE)
        .build();


if (!detector.isOperational()) {
    //Handle contingency
} else {
    Frame frame = new Frame.Builder().setBitmap(bitmap).build();
    mFaces = detector.detect(frame);
    detector.release();
}
invalidate();
```



https://code.tutsplus.com/tutorials/an-introduction-to-face-detection-on-android--cms-25212

Speech Processing

Text to Speech

Android has a text-to-speech engine (see https://developer.android.com/reference/android/speech/tts/TextToSpeech)

```
TextToSpeech ttsObject =new TextToSpeech(getApplicationContext(), new TextToSpeech.OnInitListener() {
    @Override
    public void onInit(int status) {
        if(status != TextToSpeech.ERROR) {
            ttsObject.setLanguage(Locale.ENGLISH);
        }
    }
});
```

```
ttsObject.speak(mQuestion, TextToSpeech.QUEUE_FLUSH, null, "Question");
```

Apps targeting Android 11 that use text-to-speech should declare TextToSpeech.Engine#INTENT_ACTION_TTS_SERVICE in the <queries> elements of their manifest:

Speech Processing

Speech to Text

Android also has a speech recognition engine, result is an array list of strings, representing what was recognized.

```java
private void startSpeechRecognizer() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    startActivityForResult(intent, REQUEST_SPEECH_RECOGNIZER);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    String response;
    if (requestCode == REQUEST_SPEECH_RECOGNIZER) {
        if (resultCode == RESULT_OK) {
            List<String> results = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS)
```

https://developer.android.com/reference/android/speech/SpeechRecognizer

Speech Demo

https://github.com/latrobe-cs-educator/CSE2MAD_Lecture7_SpeechRecognition

Telephony & SMS Overview

Telephony

- Initiating phone calls
- Reading the phone, network, data connectivity, and SIM states
- Monitoring changes to the phone, network, data connectivity

SMS

- Using Intents to send SMS and MMS messages
- Using the SMS Manager to send SMS Messages
- Handling incoming SMS messages

Telephony API Overview

The Android telephony APIs allows:

- Access the underlying telephone hardware stack Create your own dialer
- Integrate call handling and phone state monitoring

For security, you can’t create your own ‘‘in call’’ Activity

The screen that is displayed when an incoming call is received or an outgoing call has been placed.

https://developer.android.com/reference/android/telephony/package-summary

Telephony: Launch the dialer

Use Intent **Intent.ACTION_DIAL** to launch dialer activity.

- Specify the number to dial using the **tel:** schema as the data component of the Intent.
- Allows you to manage the call initialization (the default dialer asks the user to explicitly initiate the call).
- Doesn’t require any permissions
- The standard way applications should initiate calls

```
Intent intent = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:1234567"));
startActivity(intent);
```

https://developer.android.com/reference/android/telephony/package-summary
https://developer.android.com/reference/android/content/Intent#ACTION_DIAL

Telephony: Placing a call

Allows an application to make calls by sending an intent to the built-in dialer activity

```
Intent whoyougonnacall = new Intent(Intent.ACTION_CALL,
Uri.parse(“tel:555-2368”));  startActivity(whoyougonnacall);
```

Manifest

```
<uses-feature android:name=”android.hardware.telephony” android:required=”true”/>
```

```
<uses-permission android:name=”android.permission.CALL_PHONE”/>
```

https://developer.android.com/reference/android/Manifest.permission#CALL_PHONE

Telephony: the telephony manager

```
TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
```

Use TelephonyManager to get:

- the phone type (GSM or CDMA)
- unique ID (IMEI or MEID)
- software version
- number

Manifest

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

http://developer.android.com/reference/android/telephony/TelephonyManager.html

Telephony:Listen for phone states/calls # 1

```
PhoneStateListener callStateListener = new PhoneStateListener() {
  public void onCallStateChanged(int state, String incomingNumber) {
          String callStateStr = “Unknown”;
          switch (state) {
            case TelephonyManager.CALL_STATE_IDLE :
               callStateStr = “idle”; break;
            case TelephonyManager.CALL_STATE_OFFHOOK
               callStateStr = “offhook”; break;
            case TelephonyManager.CALL_STATE_RINGING
               callStateStr = “ringing. Incoming number is: “ + incomingNumber; break;
        default : break;
     }
     Toast.makeText(MyActivity.this, callStateStr, Toast.LENGTH_LONG).show();
  }
};
telephonyManager.listen(callStateListener, PhoneStateListener.LISTEN_CALL_STATE);
```

TelephonyManager.CALL_STATE_IDLE — When the phone is neither ringing nor in a call

TelephonyManager.CALL_STATE_RINGING — When the phone is ringing

TelephonyManager.CALL_STATE_OFFHOOK

Manifest

```
<uses-permission android:name=”android.permission.READ_PHONE_STATE”/>
```

https://developer.android.com/reference/android/telephony/PhoneStateListener

Telephony:Listen for phone states/calls # 2

```java
public class PhoneStateChangedReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        String phoneState = intent.getStringExtra(TelephonyManager.EXTRA_STATE);
        if (phoneState.equals(TelephonyManager.EXTRA_STATE_RINGING)) {
            String phoneNumber =
                    intent.getStringExtra(TelephonyManager.EXTRA_INCOMING_NUMBER);
            Toast.makeText(context, "Incoming Call From: " + phoneNumber,
                    Toast.LENGTH_LONG).show();
        }
    }
}
```

https://developer.android.com/reference/android/telephony/PhoneStateListener

Telephony: Listen for cell location change(s)

```
PhoneStateListener cellLocationListener = new PhoneStateListener() {
public void onCellLocationChanged(CellLocation location) {
   if (location instanceof GsmCellLocation) {
     GsmCellLocation gsmLocation = (GsmCellLocation)location;
     Toast.makeText(getApplicationContext(), String.valueOf(gsmLocation.getCid()),Toast.LENGTH_LONG).show();
   } else if (location instanceof CdmaCellLocation) {
   CdmaCellLocation cdmaLocation = (CdmaCellLocation)location;
     StringBuilder sb = new StringBuilder(); sb.append(cdmaLocation.getBaseStationId());
     sb.append("\n@"); sb.append(cdmaLocation.getBaseStationLatitude());  sb.append(cdmaLocation.getBaseStationLongitude());
     Toast.makeText(getApplicationContext(), sb.toString(), Toast.LENGTH_LONG).show();
    }
  }
};
telephonyManager.listen(cellLocationListener, PhoneStateListener.LISTEN_CELL_LOCATION);
```

Manifest

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

https://developer.android.com/reference/android/telephony/PhoneStateListener

Telephony: Tracking service changes & data connectivity

Can also listen for:

- phone service on/off
- emergency only
- data connectivity

```
TelephonyManager.listen(serviceStateListener,PhoneStateListener.LISTEN_SERVICE_STATE);

TelephonyManager.listen(dataStateListener,

PhoneStateListener.LISTEN_DATA_ACTIVITY |

PhoneStateListener.LISTEN_DATA_CONNECTION_STATE);
```

SMS/MMS Overview

- Can send/receive SMS/MMS
- Using SMSManager
- Use the SEND and SEND_TO actions in Intents

Telephony: Tracking service changes & data connectivity

Can also listen for:

- phone service on/off
- emergency only
- data connectivity

```
TelephonyManager.listen(serviceStateListener,PhoneStateListener.LISTEN_SERVICE_STATE);

TelephonyManager.listen(dataStateListener,
PhoneStateListener.LISTEN_DATA_ACTIVITY |
PhoneStateListener.LISTEN_DATA_CONNECTION_STATE);
```

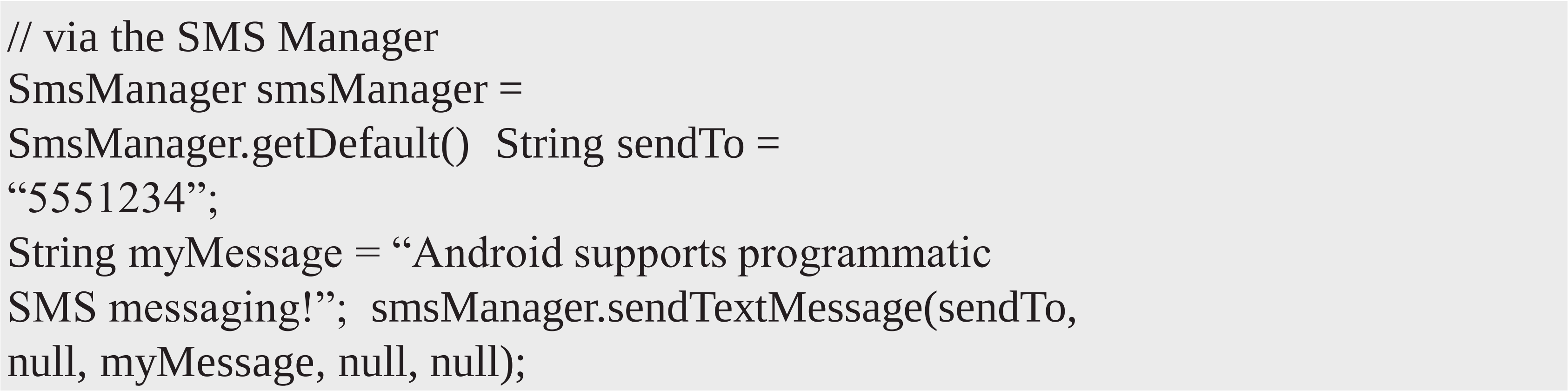
Sending SMS/MMS by Intent

```
// Normal SMS sending
Intent smsIntent = new Intent(Intent.ACTION_SENDTO,
Uri.parse(“sms:55512345”)); smsIntent.putExtra(“sms_body”, “Press send to
send me”);
startActivity(smsIntent);
```

```
// Get the URI of a piece of media to attach.
Uri attached_Uri = Uri.parse(“content://media/external/images/media/1”);

// Create a new MMS intent
Intent mmsIntent = new Intent(Intent.ACTION_SEND,
attached_Uri); mmsIntent.putExtra(“sms_body”, “Please see
the attached image”); mmsIntent.putExtra(“address”,
“07912355432”);
mmsIntent.putExtra(Intent.EXTRA_STREAM, attached_Uri);
mmsIntent.setType(“image/jpeg”);
startActivity(mmsIntent);
```

Such an intent is received and processed by a builtin SMS application

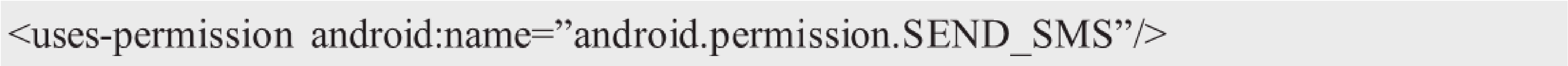
Sending SMS/MMS using SmsManager



```
// via the SMS Manager
SmsManager smsManager =
SmsManager.getDefault()  String sendTo =
“5551234”;
String myMessage = “Android supports programmatic
SMS messaging!”;  smsManager.sendTextMessage(sendTo,
null, myMessage, null, null);
```

One can set up a broadcast receiver to receive status of a message sent

Manifest



```
<uses-permission  android:name=”android.permission.SEND_SMS”/>
```

https://developer.android.com/reference/android/telephony/SmsManager

Listen for SMS messages

Broadcast Receiver

pdu = Protocol Data Unit
a Bundle object contains a set of messages

```
public class MySMSReceiver extends BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent intent) {
     Bundle bundle = intent.getExtras();
     if (bundle != null) {
        Object[] pdus = (Object[]) bundle.get(“pdus”);
        SmsMessage[] messages = new SmsMessage[pdus.length];
        for (int i = 0; i < pdus.length; i++)
          messages[i] = SmsMessage.createFromPdu((byte[]) pdus[i]);
        for (SmsMessage message : messages) {
          String msg = message.getMessageBody();
          long when = message.getTimestampMillis();
          String from = message.getOriginatingAddress();
          Toast.makeText(context, from + “ : “ + msg,
          Toast.LENGTH_LONG).show();
        }
    }
}
}
```

Manifest Permission

```
<uses-permission android:name=”android.permission.RECEIVE_SMS”/>
```

Manifest Intent Filter*

SMS-Received is one of the Implicit Broadcast Exceptions
So it can be declared in the manifest.

```
<receiver android:name=”MySMSReceiver”>
  <intent-filter>
    <action android:name=”android.provider.Telephony.SMS_RECEIVED”/>
  </intent-filter>
</receiver>
```

https://developer.android.com/guide/components/broadcast-exceptions