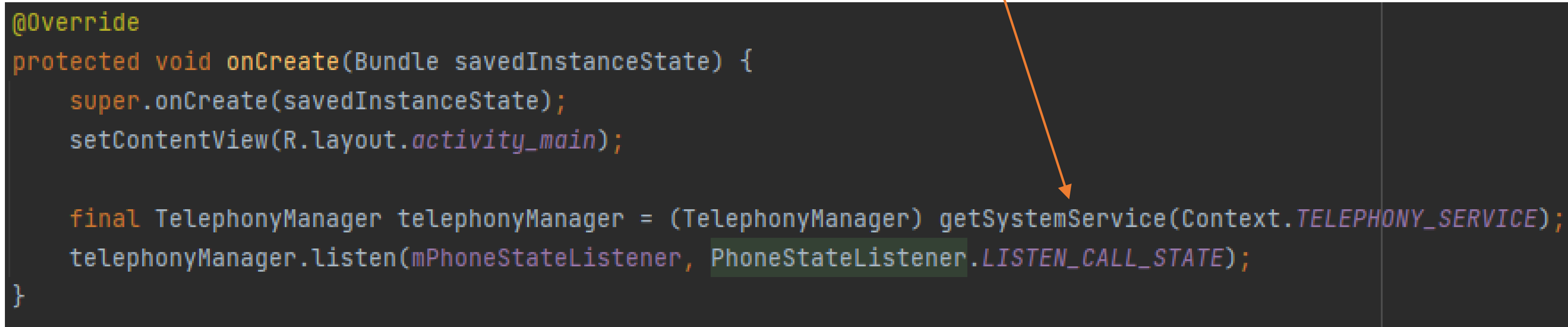
Recognizing incoming calls

Dr. Charles Yu

- Demo1: PhoneStateListener
  - Easy. No need to request user’s permission in the runtime.
  - Deprecated
- Demo2: IncomingCallStatus
  - BroadcastReceiver is used to listen to system events
  - Complicated
  - It needs to request user’s permission in the runtime
  - I need to put a note here, the way to request user’s permission is deprecated
    - The approach is around the year of 2018~2020

PhoneStateListener

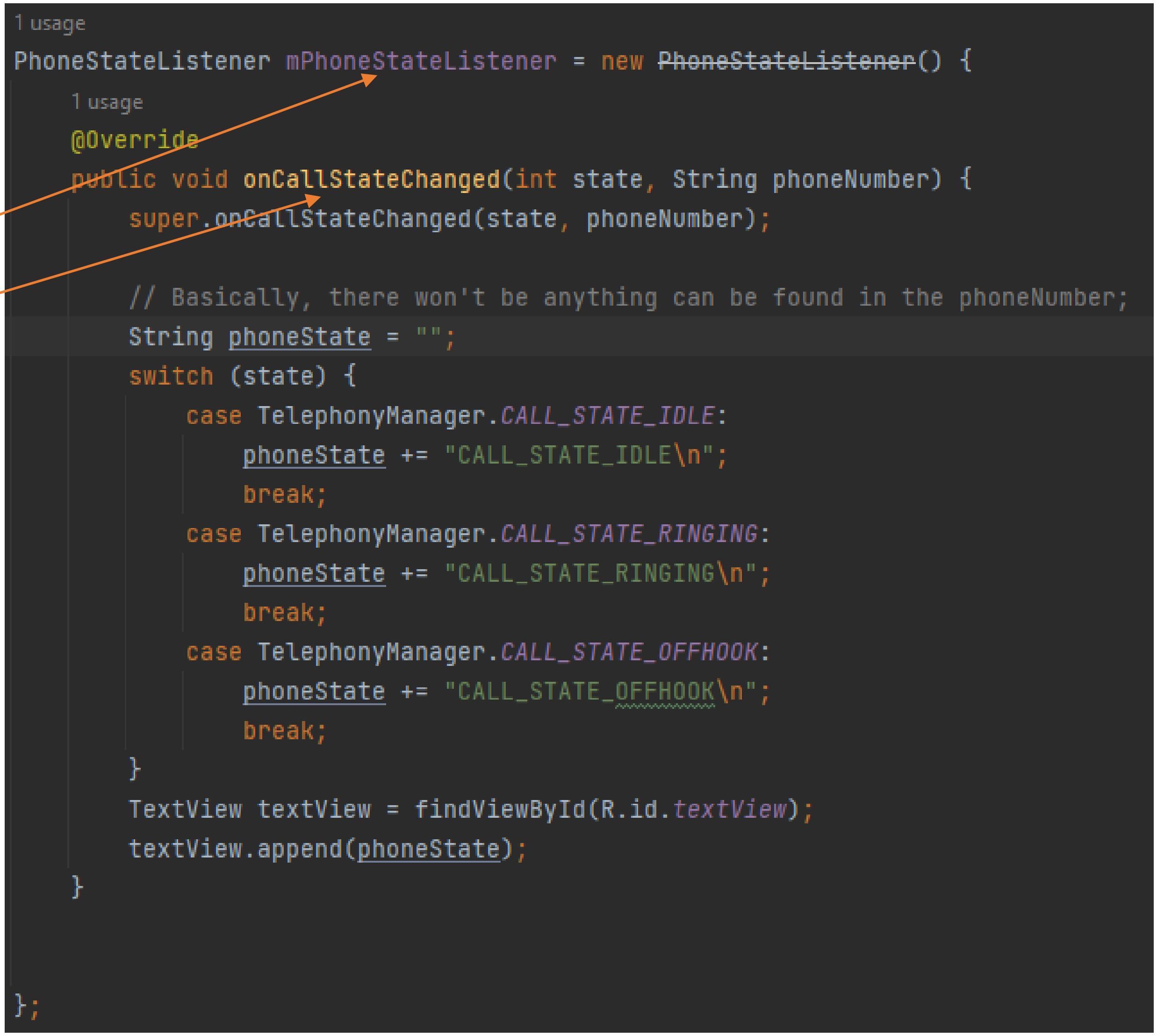
- You will need to import this “android.telephony.PhoneStateListener;”
  - Deprecated
  - It is still runnable
  - No need to ask end user for access rights in the run time.
  - No need to specify access rights in the AndroidMenifest.xml
  - In the following demo, everything is in MainActivity.java
- [Demo1] PhoneStateListener
  - Step1: Get in instance of TelephonyManager by calling getSystemService() in the onCreate()



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    final TelephonyManager telephonyManager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
    telephonyManager.listen(mPhoneStateListener, PhoneStateListener.LISTEN_CALL_STATE);
}
```

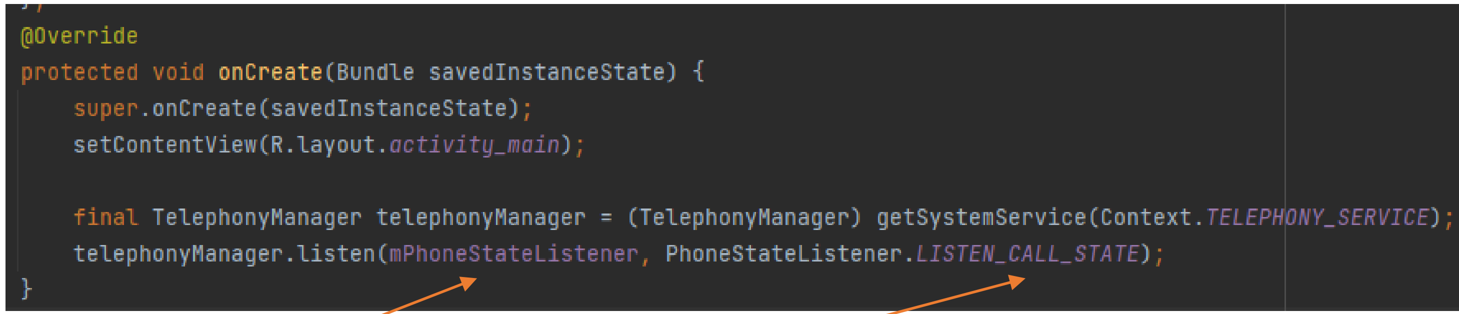
PhoneStateListener

- Step2:
  - “New” an instance mPhoneStateListener from PhoneStateListener and override (implement) the onCallStateChanged() with a case-switch
  - Append the displaying status to a string and paste the string onto a TextView



```
1 usage
PhoneStateListener mPhoneStateListener = new ~~PhoneStateListener~~() {
    1 usage
    @Override
    public void onCallStateChanged(int state, String phoneNumber) {
        super.onCallStateChanged(state, phoneNumber);

        // Basically, there won't be anything can be found in the phoneNumber;
        String phoneState = "";
        switch (state) {
            case TelephonyManager.CALL_STATE_IDLE:
                phoneState += "CALL_STATE_IDLE\n";
                break;
            case TelephonyManager.CALL_STATE_RINGING:
                phoneState += "CALL_STATE_RINGING\n";
                break;
            case TelephonyManager.CALL_STATE_OFFHOOK:
                phoneState += "CALL_STATE_OFFHOOK\n";
                break;
        }
        TextView textView = findViewById(R.id.textView);
        textView.append(phoneState);
    }


};
```

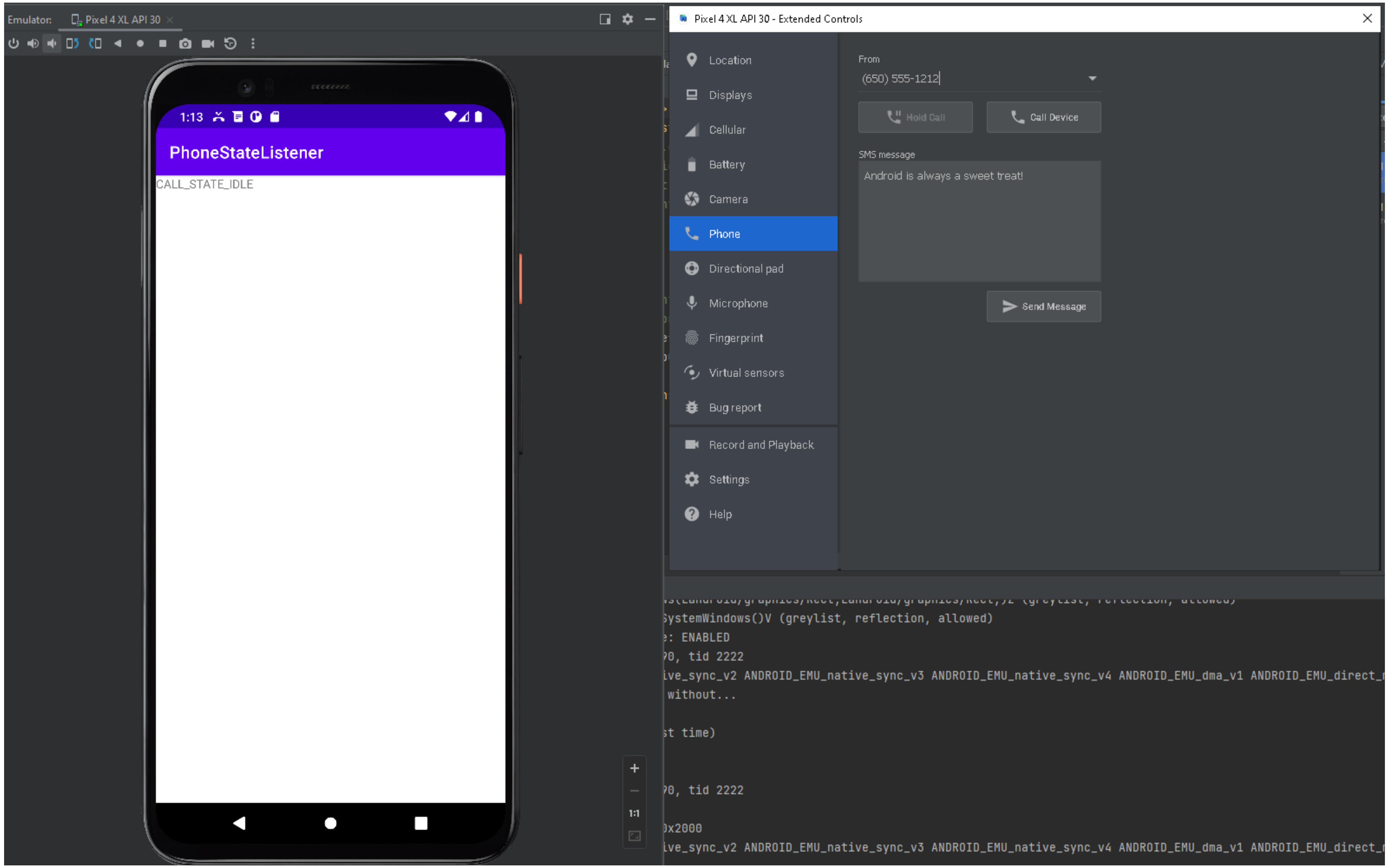
PhoneStateListener



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    final TelephonyManager telephonyManager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
    telephonyManager.listen(mPhoneStateListener, PhoneStateListener.LISTEN_CALL_STATE);
}
```

- Step3:
  - Go back to the onCreate() and use the TelephonyManager to listen to the call state
  - The 1st parameter is the new created instance "mPhoneStateListener"
  - The 2nd parameter is the "call state". This is the state the TelephonyManager about to listen.
- Basically, there is nothing special in the AndroidManifest.xml or activity_main.xml

PhoneStateListener



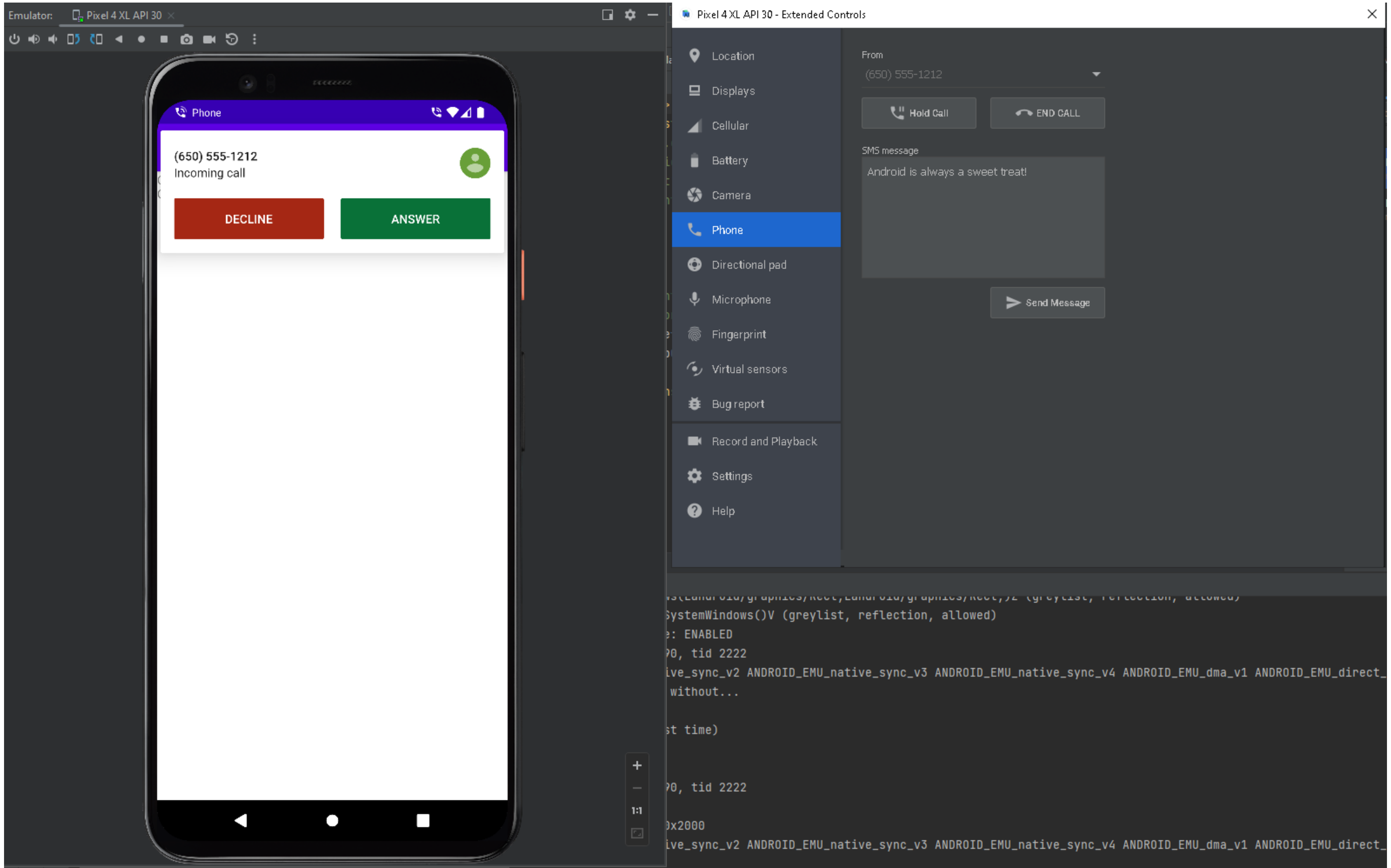
CALL_STATE_IDLE

From
(650) 555-1212

SMS message
Android is always a sweet treat!

```
SystemWindows()V (greylist, reflection, allowed)
e: ENABLED
90, tid 2222
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
 without...

st time)


90, tid 2222

0x2000
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
```

PhoneStateListener



Phone

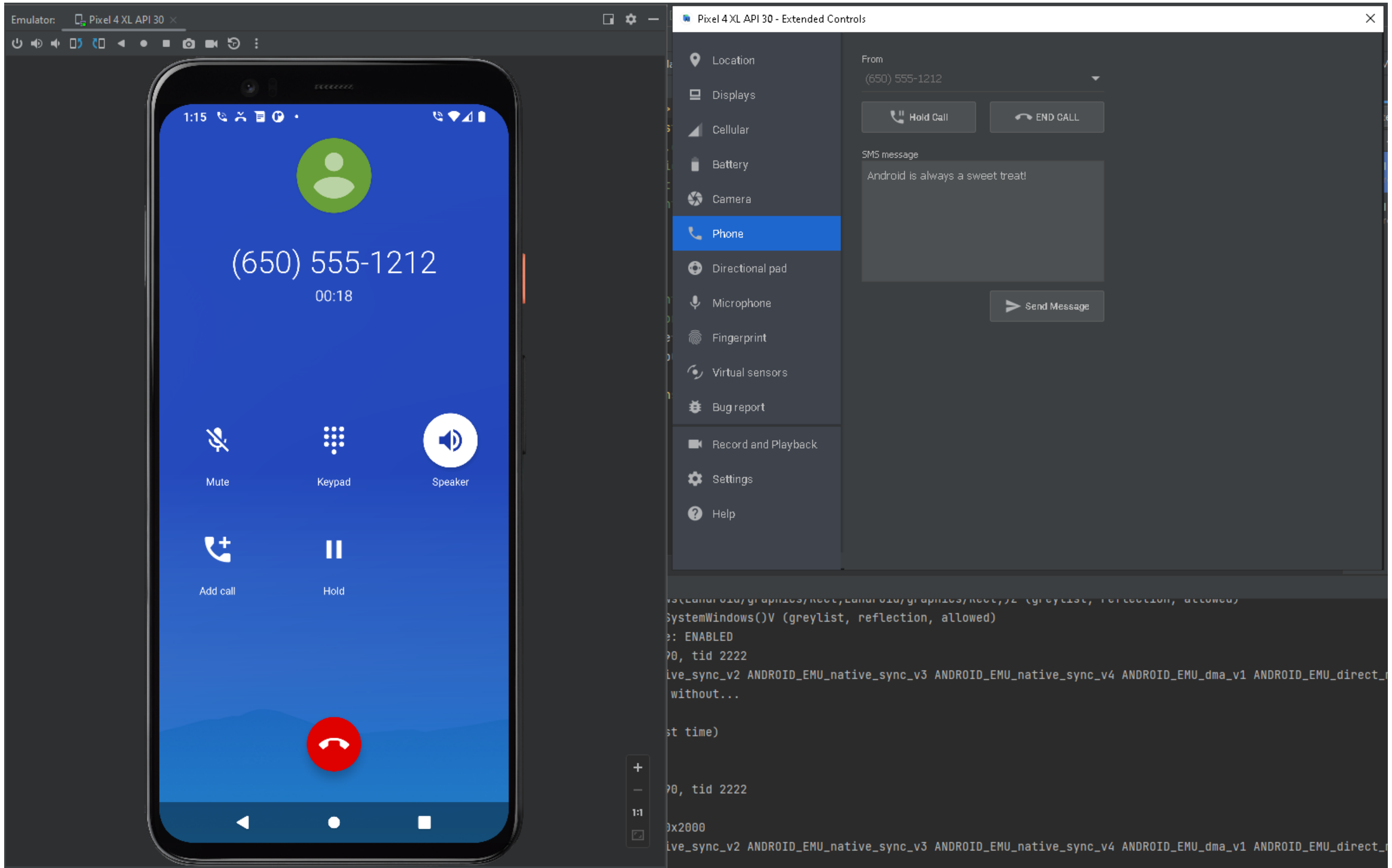
(650) 555-1212
Incoming call

From: (650) 555-1212

SMS message: Android is always a sweet treat!

```
ws(Landroid/graphics/Rect;Landroid/graphics/Rect;)Z (greylist, reflection, allowed)
SystemWindows()V (greylist, reflection, allowed)
e: ENABLED
90, tid 2222
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
 without...

st time)


90, tid 2222

0x2000
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
```

PhoneStateListener



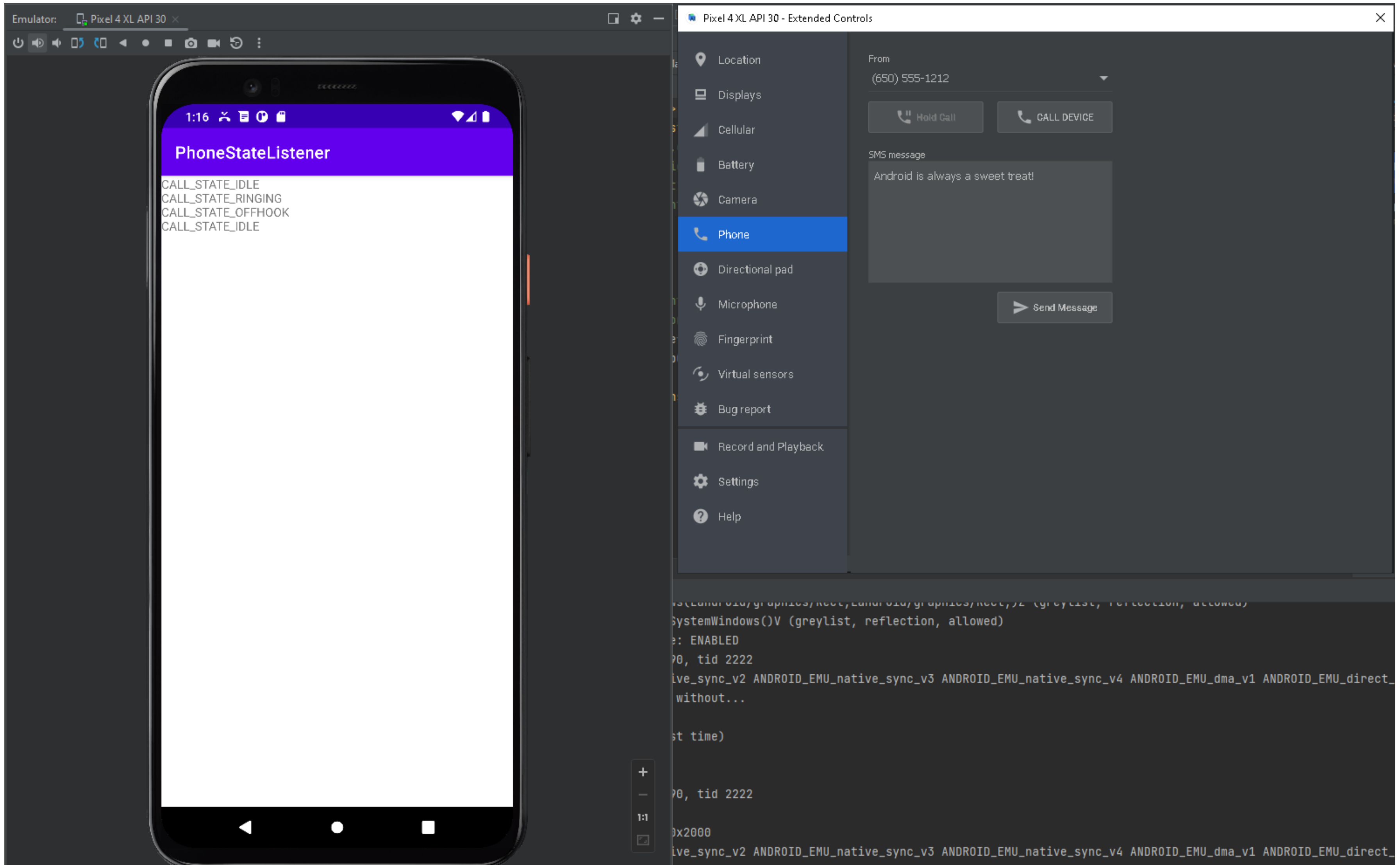
(650) 555-1212
00:18

From
(650) 555-1212

SMS message
Android is always a sweet treat!

```
SystemWindows()V (greylist, reflection, allowed)
e: ENABLED
90, tid 2222
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
 without...

st time)


90, tid 2222

0x2000
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
```

PhoneStateListener



PhoneStateListener

```
CALL_STATE_IDLE
CALL_STATE_RINGING
CALL_STATE_OFFHOOK
CALL_STATE_IDLE
```

From: (650) 555-1212

SMS message: Android is always a sweet treat!

```
s(Landroid/graphics/Rect;Landroid/graphics/Rect;)Z (greylist, reflection, allowed)
SystemWindows()V (greylist, reflection, allowed)
e: ENABLED
90, tid 2222
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
 without...

st time)


90, tid 2222

0x2000
ive_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_
```

IncomingCallStatus

- A very obvious change from the previous demo is that,
  - This one ask for runtime permissions from user
  - Check the video for runtime permissions (Android Kotlin language)
    - https://youtu.be/x38dYUm7tCY
    - Official Document
      - https://developer.android.com/training/permissions/requesting
- [Demo2] IncomingCallStatus
  - Check the AndroidManifest.xml in the next page

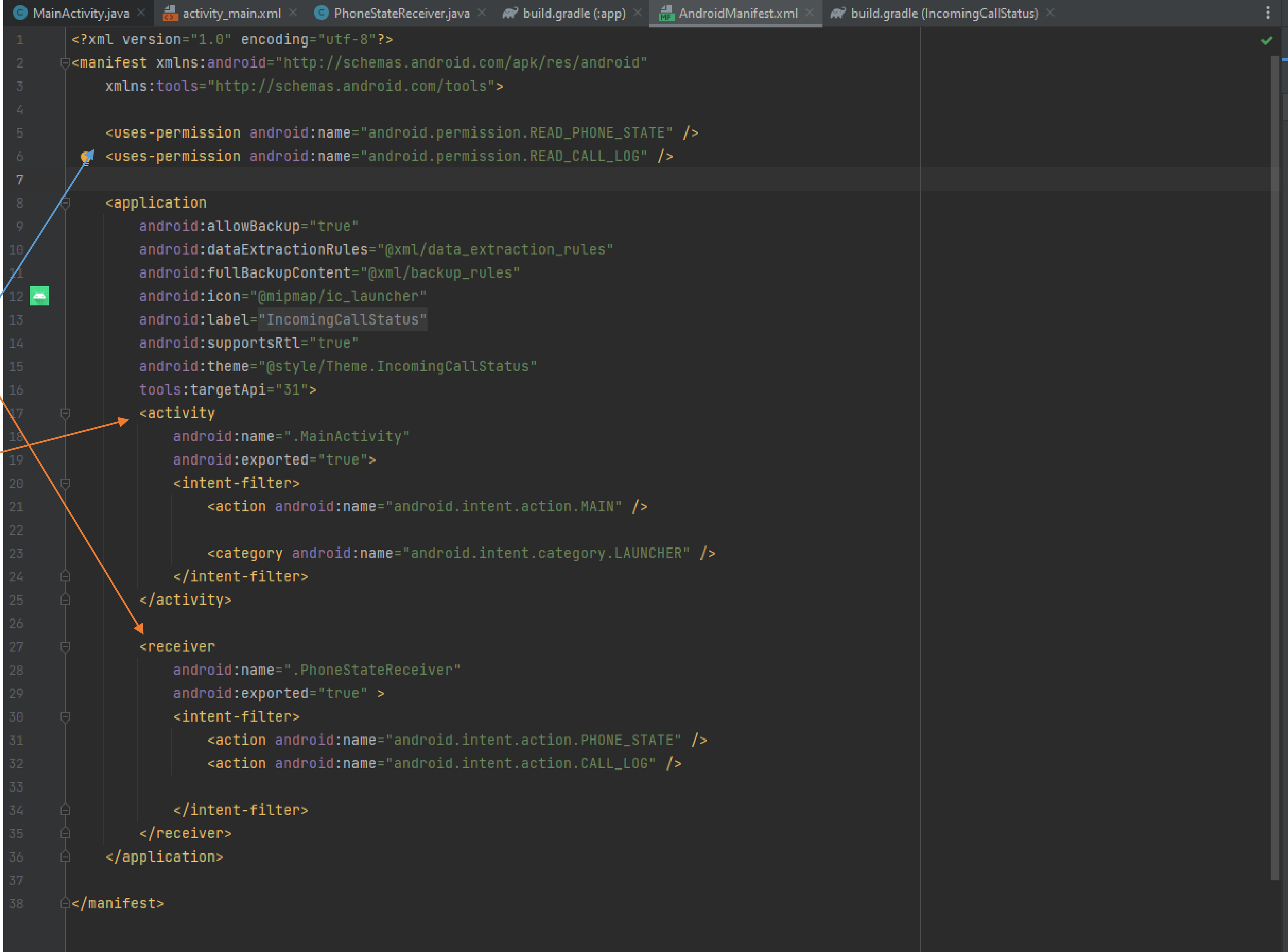
Incoming Call Status

One broadcast receiver

- Listen to system events, including the events from TelephonyManager
- Two actions are user defined action.

One activity

Two permissions



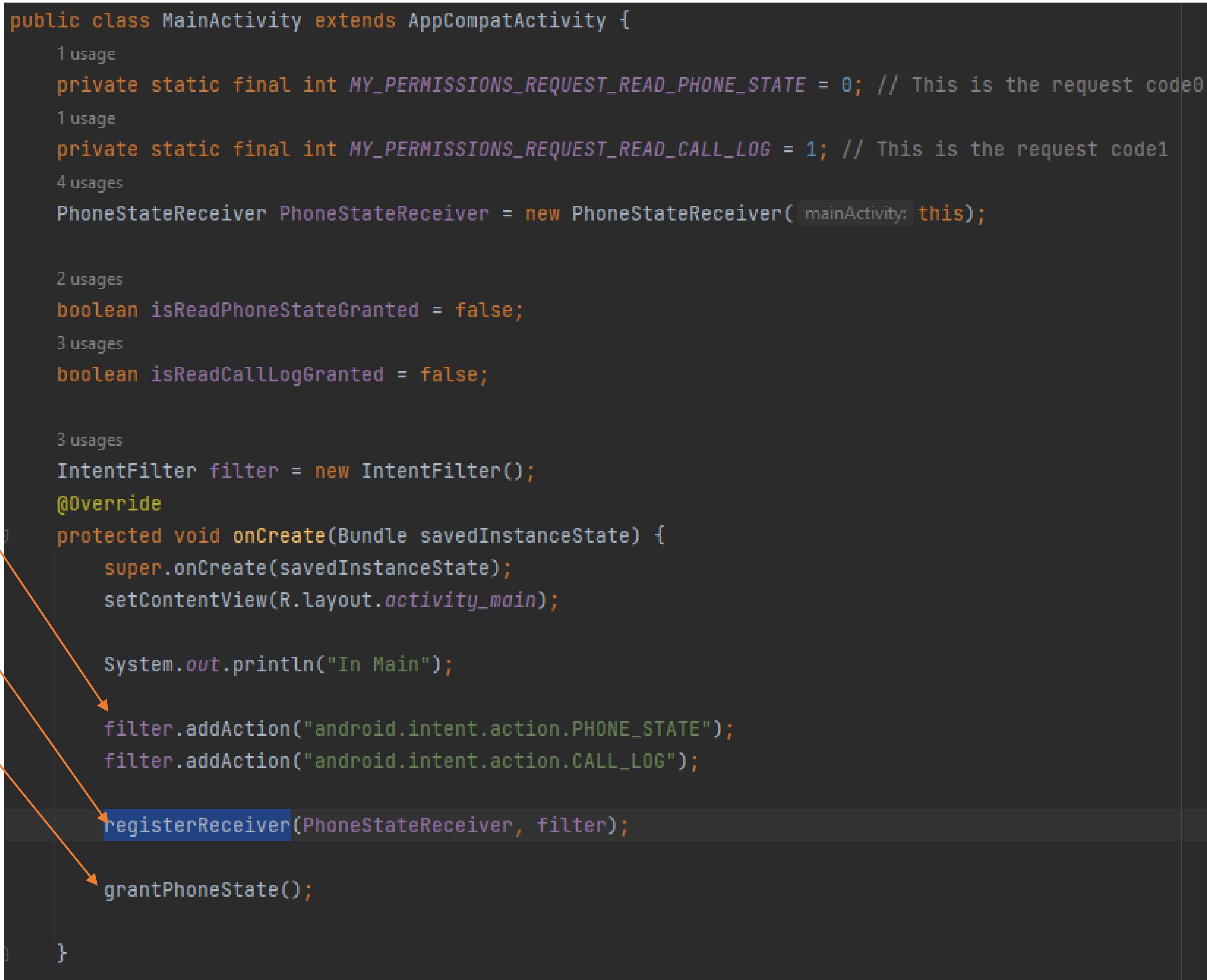
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

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

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="IncomingCallStatus"
        android:supportsRtl="true"
        android:theme="@style/Theme.IncomingCallStatus"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

        <receiver
            android:name=".PhoneStateReceiver"
            android:exported="true" >
            <intent-filter>
                <action android:name="android.intent.action.PHONE_STATE" />
                <action android:name="android.intent.action.CALL_LOG" />

            </intent-filter>
        </receiver>
    </application>

</manifest>
```

IncomingCallStatus

- MainActivity
  - A call to registerReceiver() is used in the MainActivity. It accepts 2 arguments
    - A broadcast receiver
    - An intent filter
      - 2 actions are recorded in the AndroidManifest.xml (previous page)
  - Then, we need to grant the READ_PHONE_STATE by calling grantPhoneState(). This one is defined by us for asking the run-time access rights
  - (see the next page)

Incoming Call Status(MainActivity)

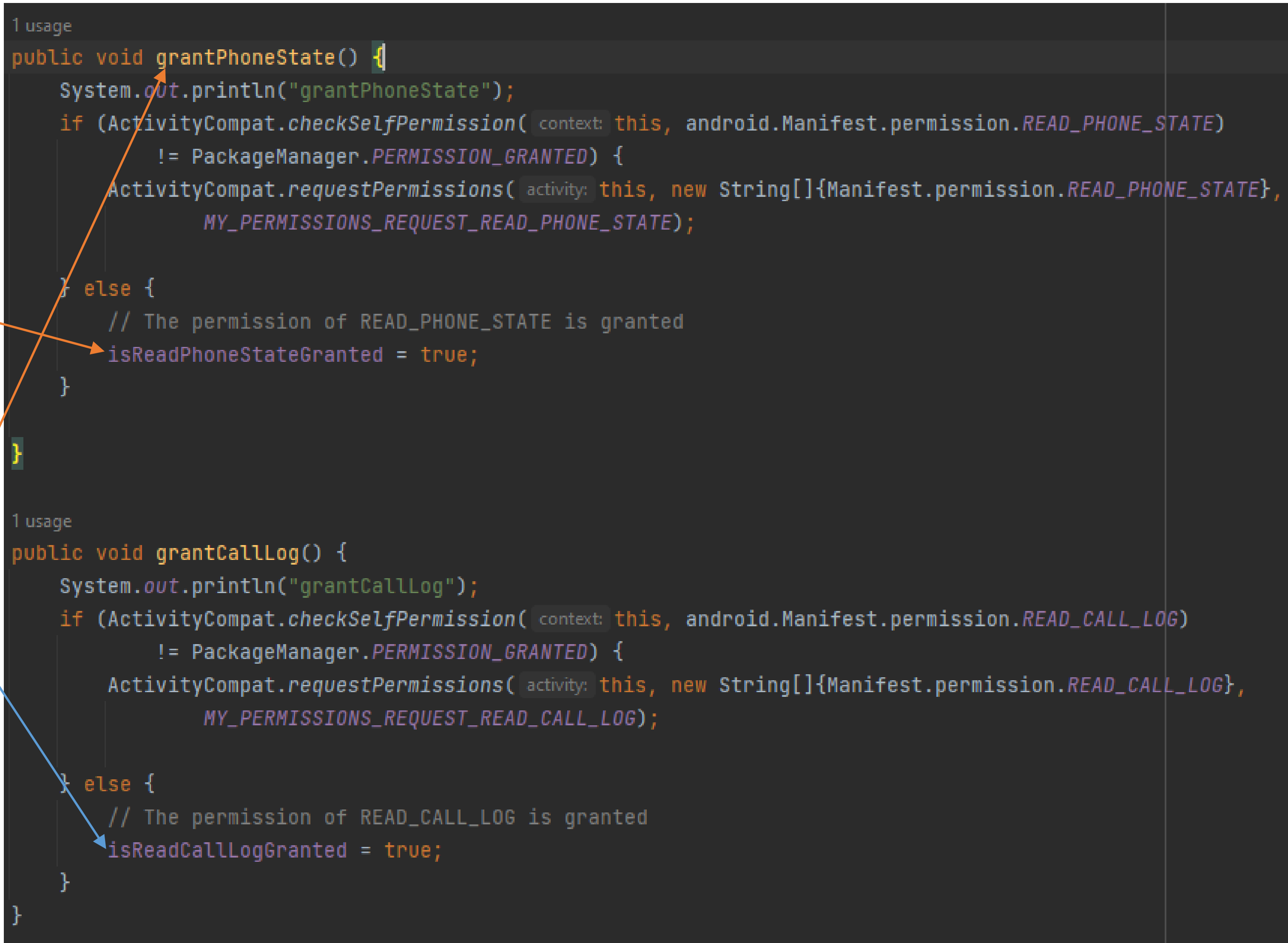
- Intent filter
- Register the Broadcast Receiver
- Setup the runtime permissions --- READ_PHONE_STATE



```
public class MainActivity extends AppCompatActivity {
    1 usage
    private static final int MY_PERMISSIONS_REQUEST_READ_PHONE_STATE = 0; // This is the request code0
    1 usage
    private static final int MY_PERMISSIONS_REQUEST_READ_CALL_LOG = 1; // This is the request code1
    4 usages
    PhoneStateReceiver PhoneStateReceiver = new PhoneStateReceiver( mainActivity: this);

    2 usages
    boolean isReadPhoneStateGranted = false;
    3 usages
    boolean isReadCallLogGranted = false;

    3 usages
    IntentFilter filter = new IntentFilter();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        System.out.println("In Main");

        filter.addAction("android.intent.action.PHONE_STATE");
        filter.addAction("android.intent.action.CALL_LOG");

        registerReceiver(PhoneStateReceiver, filter);

        grantPhoneState();

    }
```

Incoming Call Status(MainActivity)

- This is the way to ask runtime permission (old style)
- There are 2 boolean flags and the purpose of this is to keep tracking the status of user’s permission, in case I ask for the permission again
    - isReadPhoneStateGranted
    - isReadCallLogGranted
- This one is called in the onCreate()



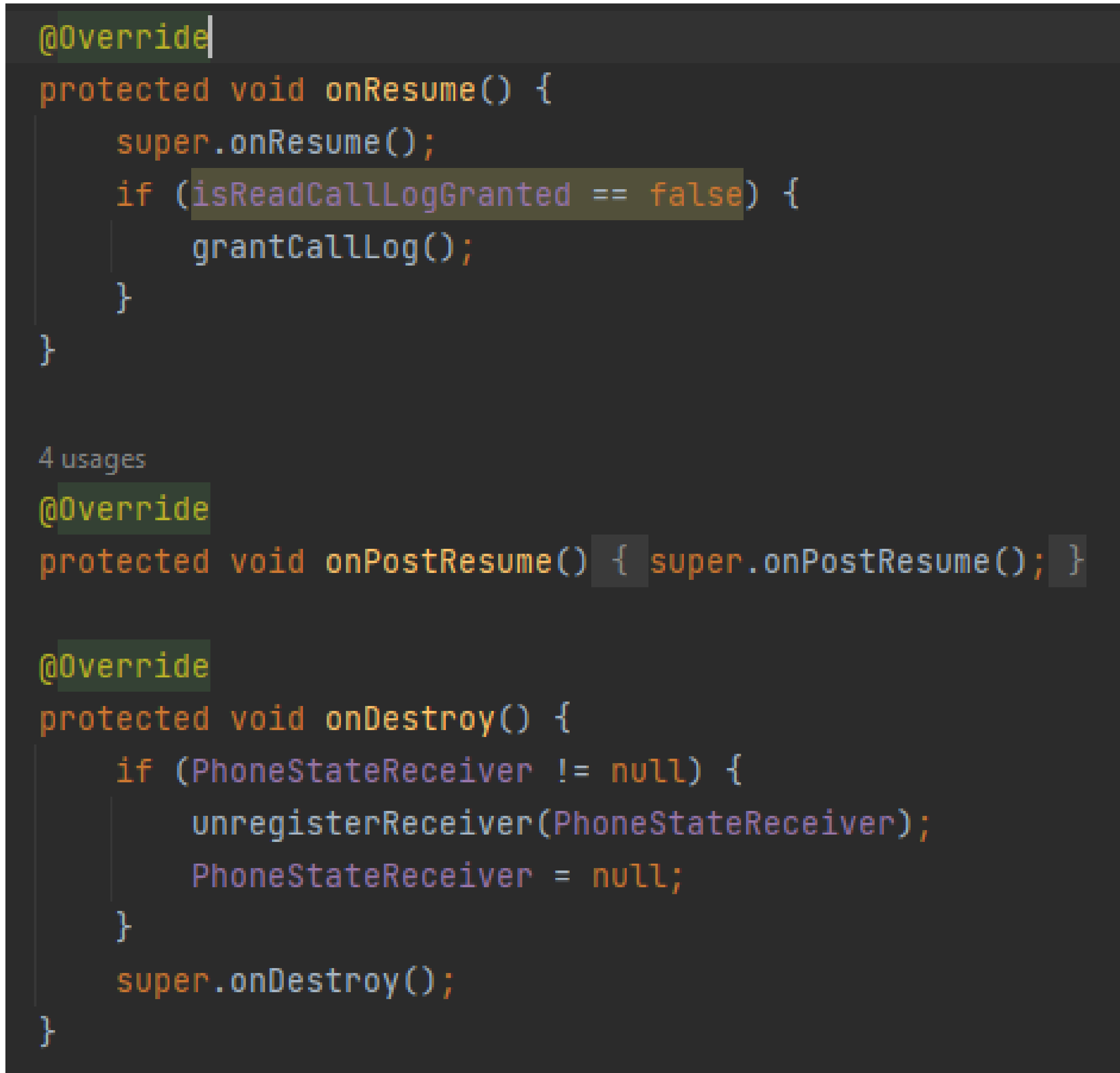
```
1 usage
public void grantPhoneState() {
    System.out.println("grantPhoneState");
    if (ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.READ_PHONE_STATE)
            != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions( activity: this, new String[]{Manifest.permission.READ_PHONE_STATE},
                MY_PERMISSIONS_REQUEST_READ_PHONE_STATE);

    } else {
        // The permission of READ_PHONE_STATE is granted
        isReadPhoneStateGranted = true;
    }

}

1 usage
public void grantCallLog() {
    System.out.println("grantCallLog");
    if (ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.READ_CALL_LOG)
            != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions( activity: this, new String[]{Manifest.permission.READ_CALL_LOG},
                MY_PERMISSIONS_REQUEST_READ_CALL_LOG);

    } else {
        // The permission of READ_CALL_LOG is granted
        isReadCallLogGranted = true;
    }
}
```

Incoming Call Status(MainActivity)

- So far, I only ask for the access right grantPhoneState() in onCreate()
- But I call the grantCallLog() in the onResume()
- The way I'm asking for the access rights is old style. It can only ask for ONE access right per time.
- That's why I need to ask that 2 times. (old style)
- Also, I unregister the Broadcast receiver in the onDestroy()

IncomingCallStatus(MainActivity)

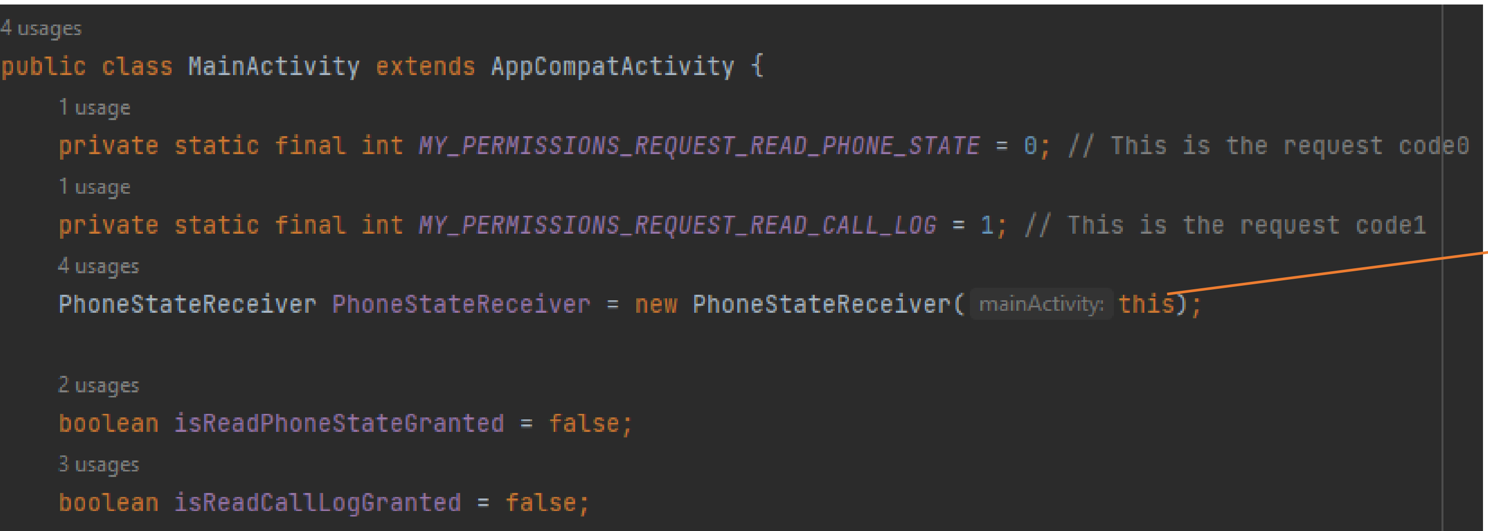
- In the old style for access right requests, it can only ask one type of the access right in one time



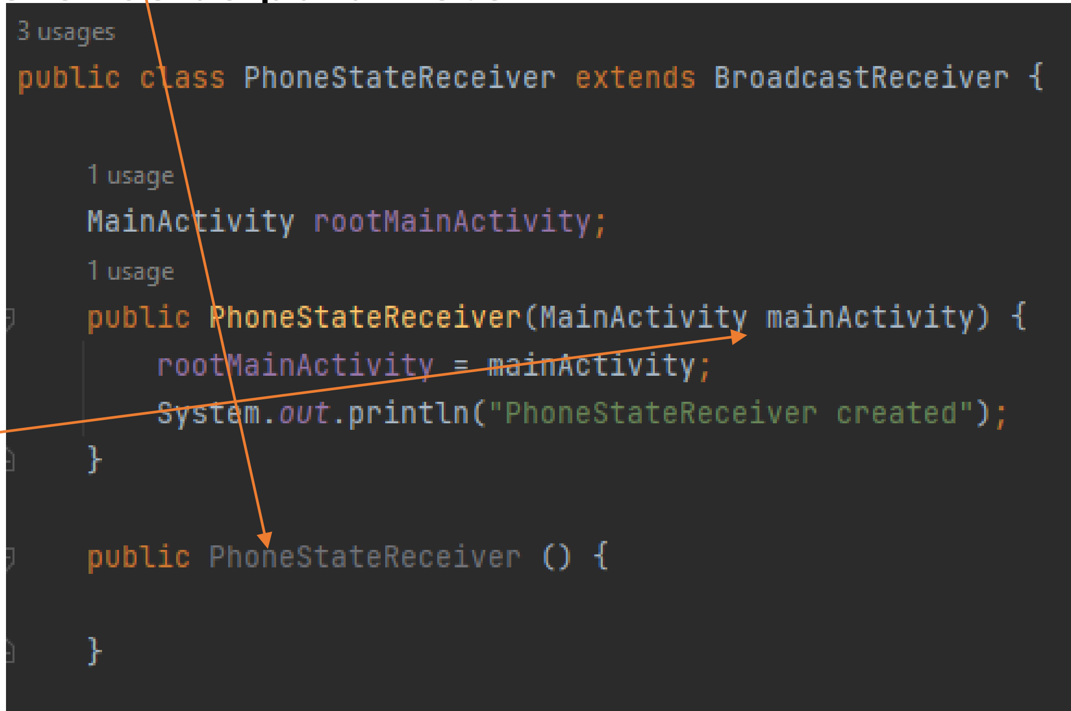
```
@Override
protected void onResume() {
    super.onResume();
    if (isReadCallLogGranted == false) {
        grantCallLog();
    }
}

4 usages
@Override
protected void onPostResume() { super.onPostResume(); }

@Override
protected void onDestroy() {
    if (PhoneStateReceiver != null) {
        unregisterReceiver(PhoneStateReceiver);
        PhoneStateReceiver = null;
    }
    super.onDestroy();
}
```

IncomingCallStatus(PhoneStateReceiver)

- The 2nd constructor is a default constructor.
  - It seems required. Otherwise, I will get warning message.
- The 1st constructor is required by the API
  - It needs original Activity to pass its “this” reference as parameter



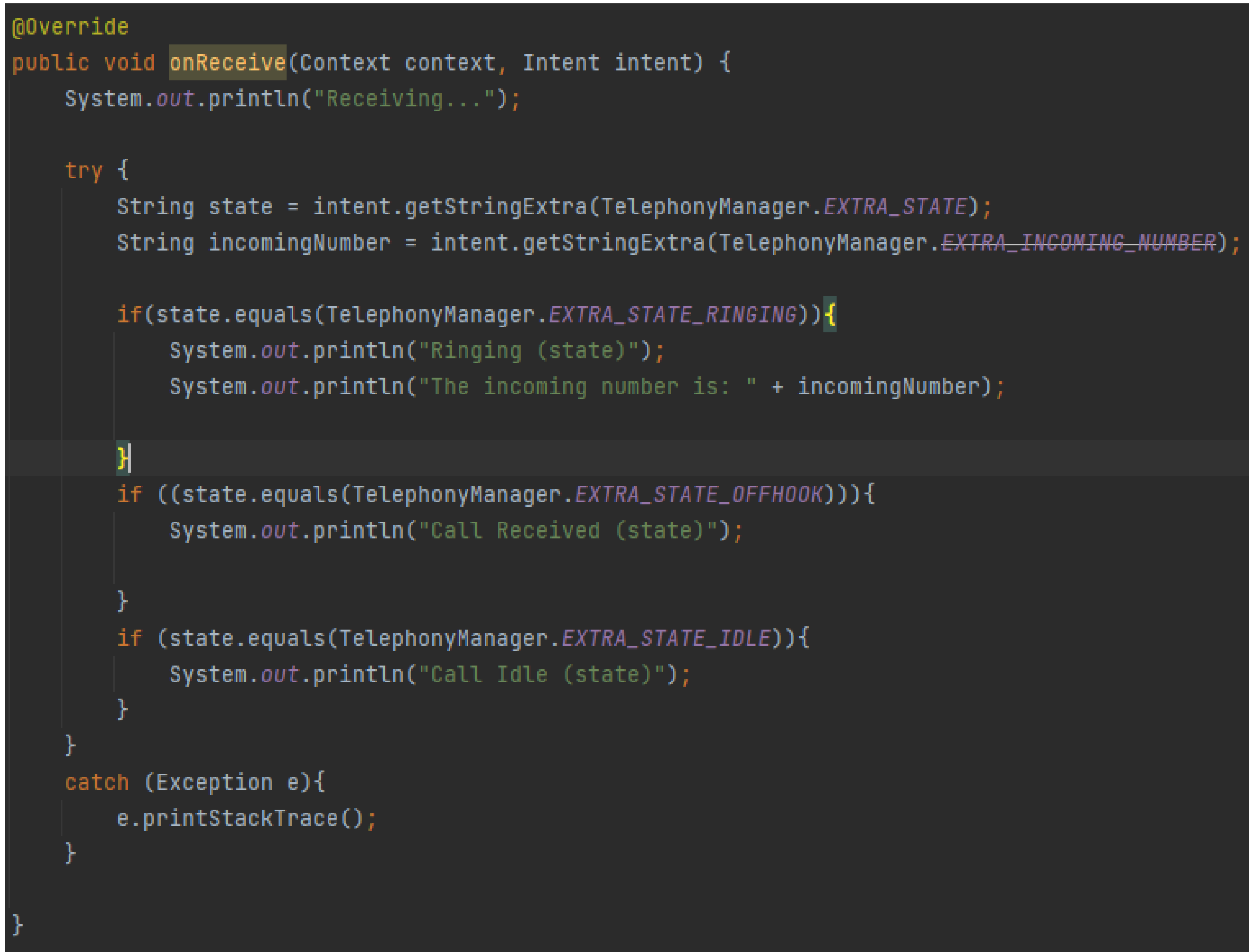
```
4 usages
public class MainActivity extends AppCompatActivity {
    1 usage
    private static final int MY_PERMISSIONS_REQUEST_READ_PHONE_STATE = 0; // This is the request code0
    1 usage
    private static final int MY_PERMISSIONS_REQUEST_READ_CALL_LOG = 1; // This is the request code1
    4 usages
    PhoneStateReceiver PhoneStateReceiver = new PhoneStateReceiver( mainActivity: this);

    2 usages
    boolean isReadPhoneStateGranted = false;
    3 usages
    boolean isReadCallLogGranted = false;
```



```
3 usages
public class PhoneStateReceiver extends BroadcastReceiver {

    1 usage
    MainActivity rootMainActivity;
    1 usage
    public PhoneStateReceiver(MainActivity mainActivity) {
        rootMainActivity = mainActivity;
        System.out.println("PhoneStateReceiver created");
    }

    public PhoneStateReceiver () {

    }
```

IncomingCallStatus(PhoneStateReceiver)

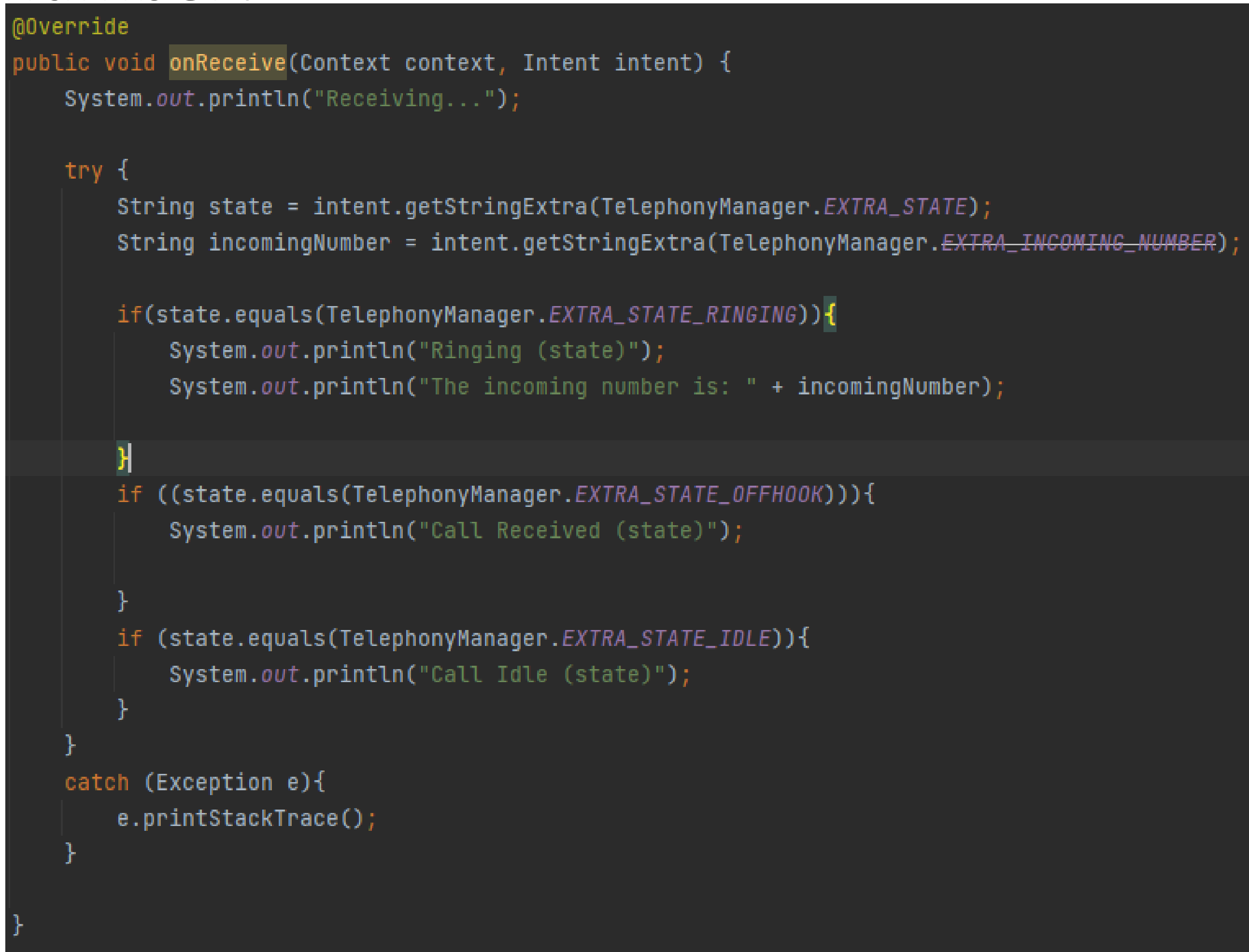
- It looks **very similar** to our previously introduced case-switch
- PhoneStateReceiver extends BroadcastReceiver
- BroadcastReceiver is a thing, once it is registered, it can **keep listening to system events**



```java
@Override
public void onReceive(Context context, Intent intent) {
    System.out.println("Receiving...");

    try {
        String state = intent.getStringExtra(TelephonyManager.EXTRA_STATE);
        String incomingNumber = intent.getStringExtra(TelephonyManager.~~EXTRA_INCOMING_NUMBER~~);

        if(state.equals(TelephonyManager.EXTRA_STATE_RINGING)){
            System.out.println("Ringing (state)");
            System.out.println("The incoming number is: " + incomingNumber);

        }
        if ((state.equals(TelephonyManager.EXTRA_STATE_OFFHOOK))){
            System.out.println("Call Received (state)");

        }
        if (state.equals(TelephonyManager.EXTRA_STATE_IDLE)){
            System.out.println("Call Idle (state)");
        }
    }
    catch (Exception e){
        e.printStackTrace();
    }

}
```

IncomingCallStatus(PhoneStateReceiver)

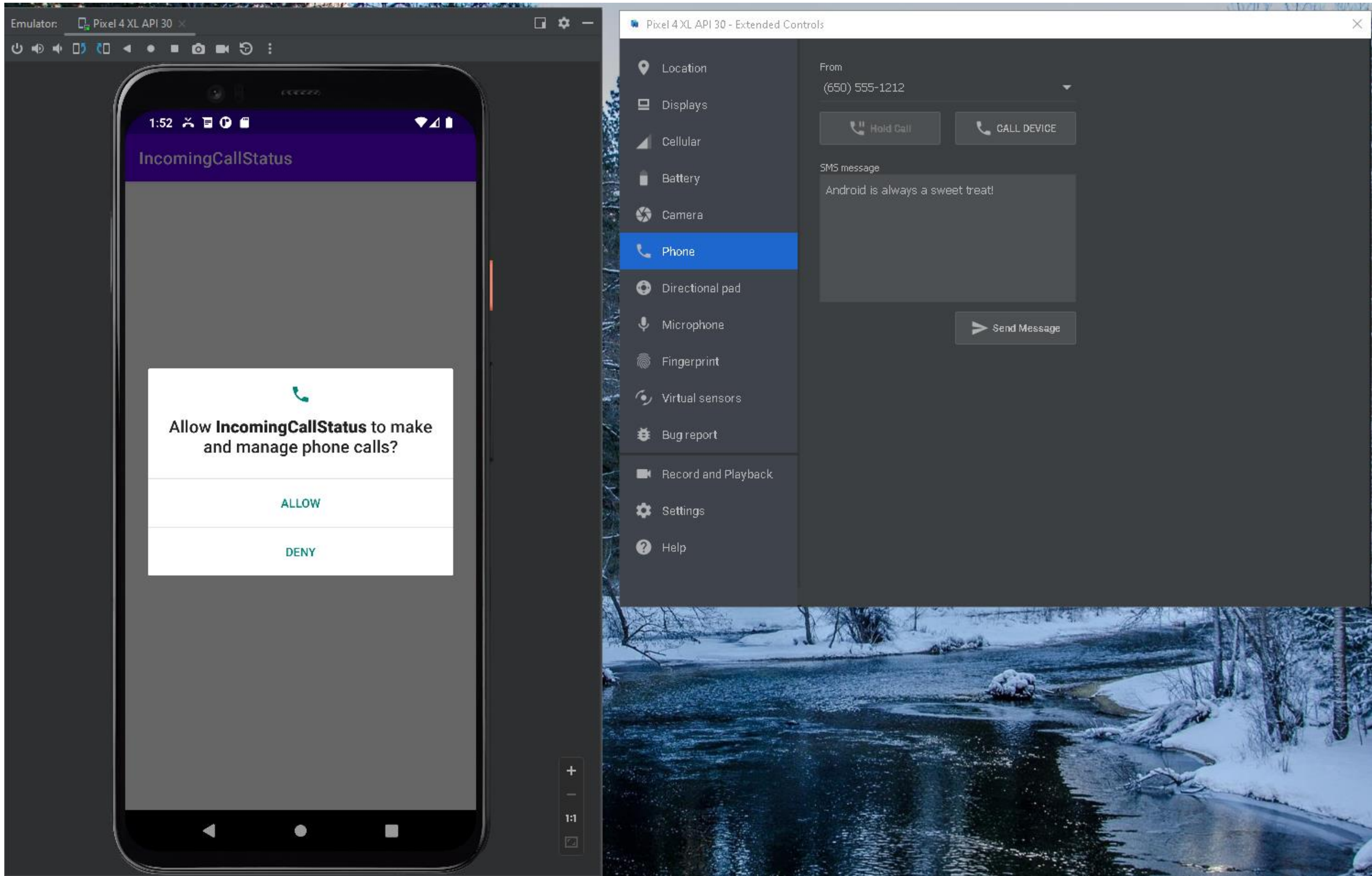
- How do I get the incoming phone number?
  - TelephonyManager.EXTRA_INCOMING_NUMBER



```
@Override
public void onReceive(Context context, Intent intent) {
    System.out.println("Receiving...");

    try {
        String state = intent.getStringExtra(TelephonyManager.EXTRA_STATE);
        String incomingNumber = intent.getStringExtra(TelephonyManager.~~EXTRA_INCOMING_NUMBER~~);

        if(state.equals(TelephonyManager.EXTRA_STATE_RINGING)){
            System.out.println("Ringing (state)");
            System.out.println("The incoming number is: " + incomingNumber);

        }
        if ((state.equals(TelephonyManager.EXTRA_STATE_OFFHOOK))){
            System.out.println("Call Received (state)");

        }
        if (state.equals(TelephonyManager.EXTRA_STATE_IDLE)){
            System.out.println("Call Idle (state)");
        }
    }
    catch (Exception e){
        e.printStackTrace();
    }

}
```

IncomingCallStatus

- grantPhoneState()



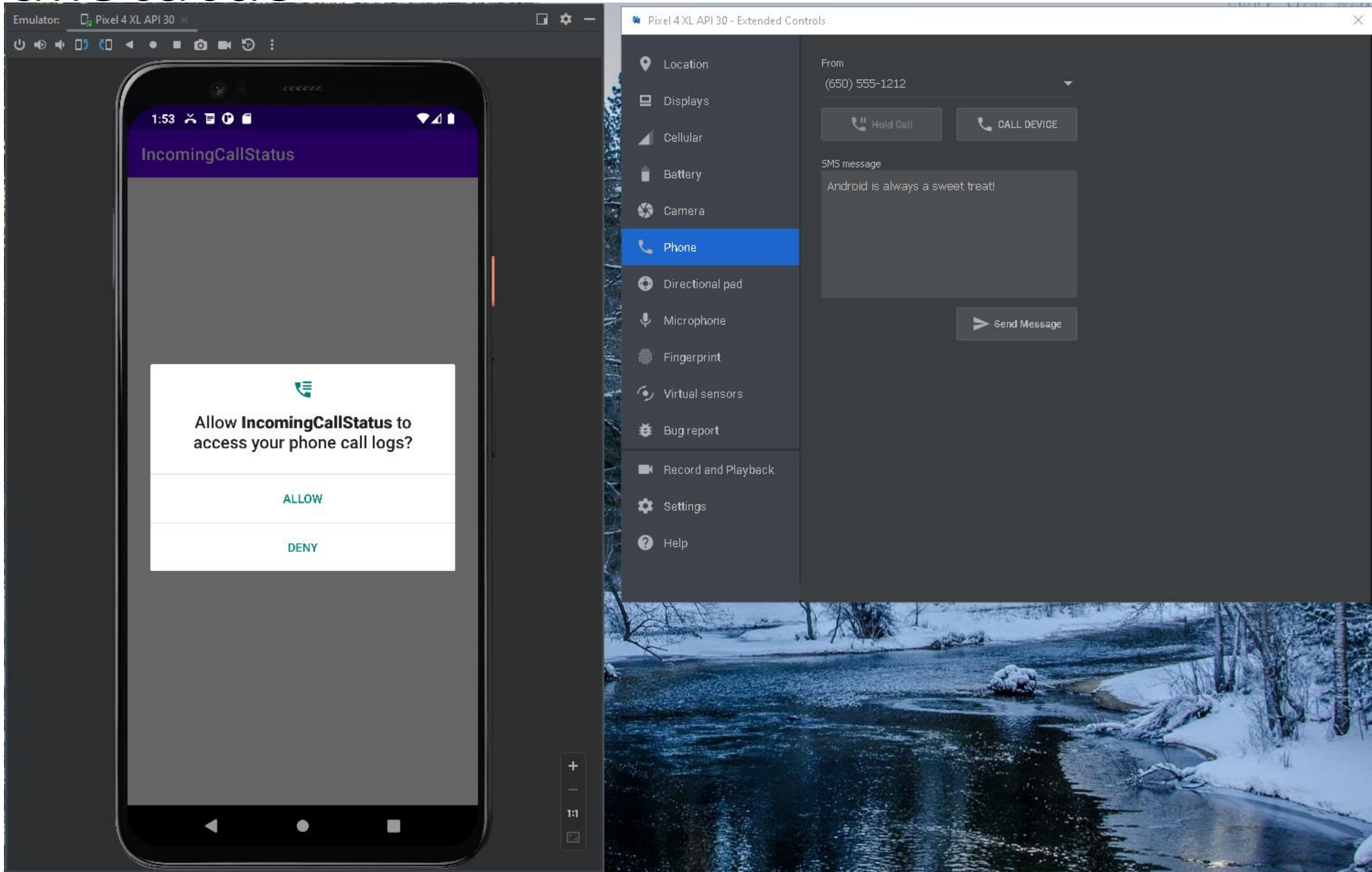
Allow **IncomingCallStatus** to make and manage phone calls?

From
(650) 555-1212

SMS message
Android is always a sweet treat!

IncomingCallStatus

- grantCallLog()



Allow **IncomingCallStatus** to access your phone call logs?

ALLOW

DENY

From

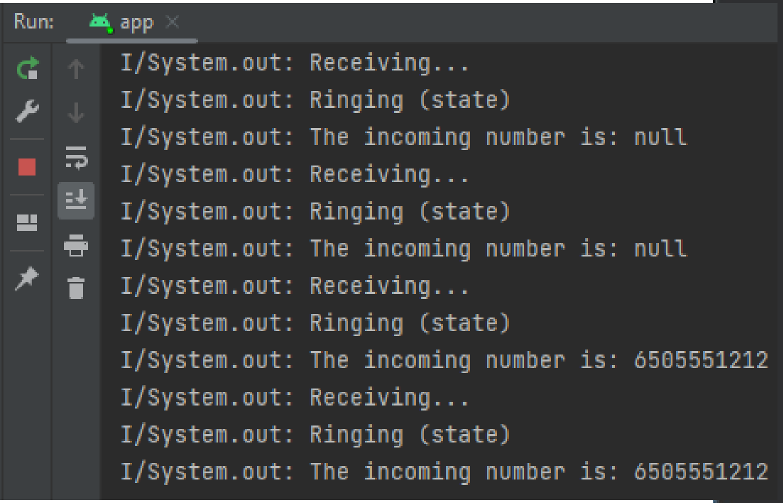
(650) 555-1212

SMS message

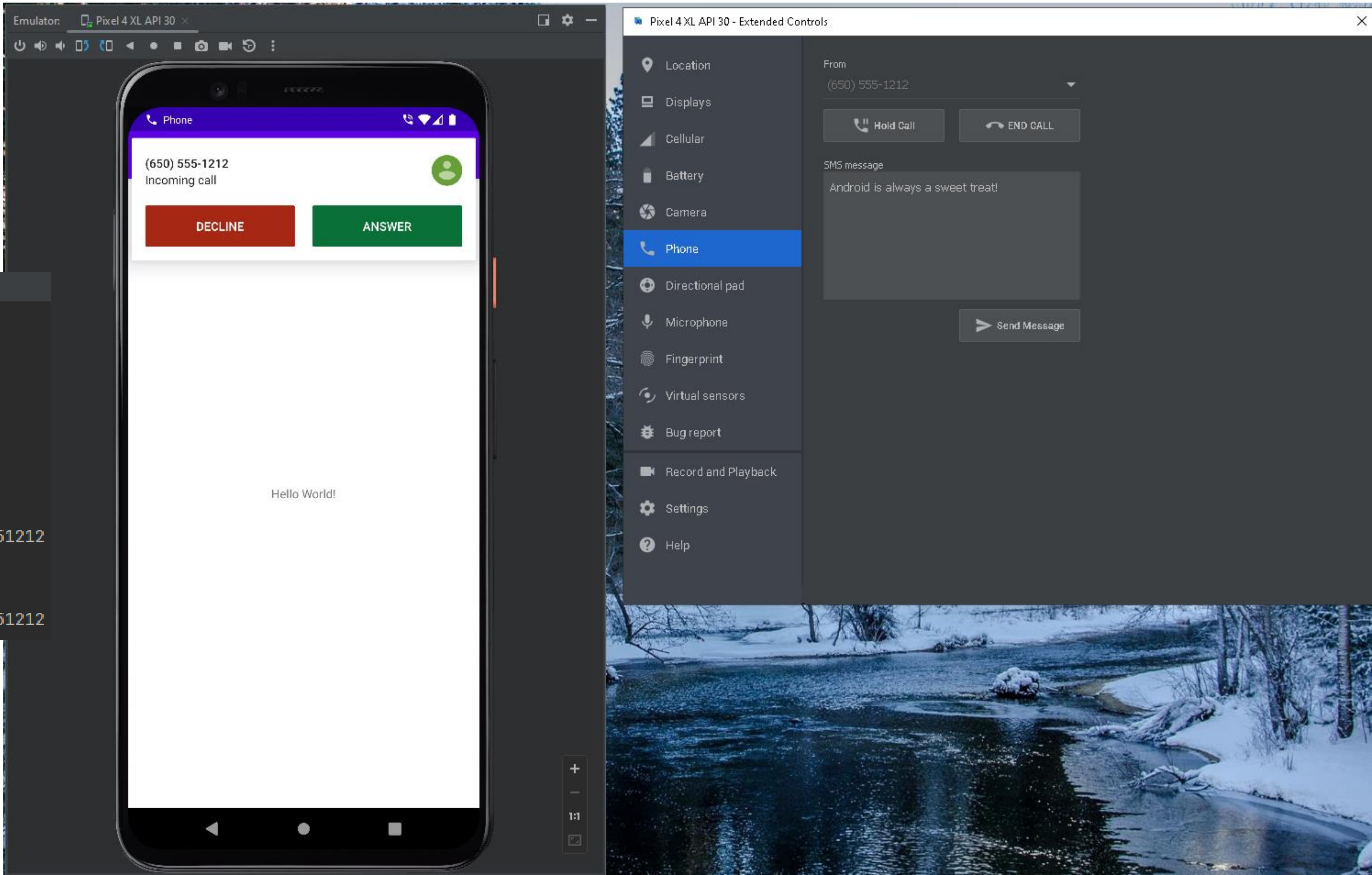
Android is always a sweet treat!

IncomingCallStatus

- RHS: Calling
- LHS: Ringing



```
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: null
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: null
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: 6505551212
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: 6505551212
```

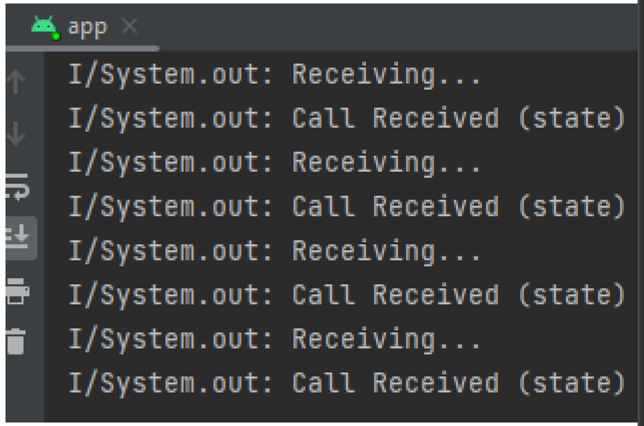


From: (650) 555-1212

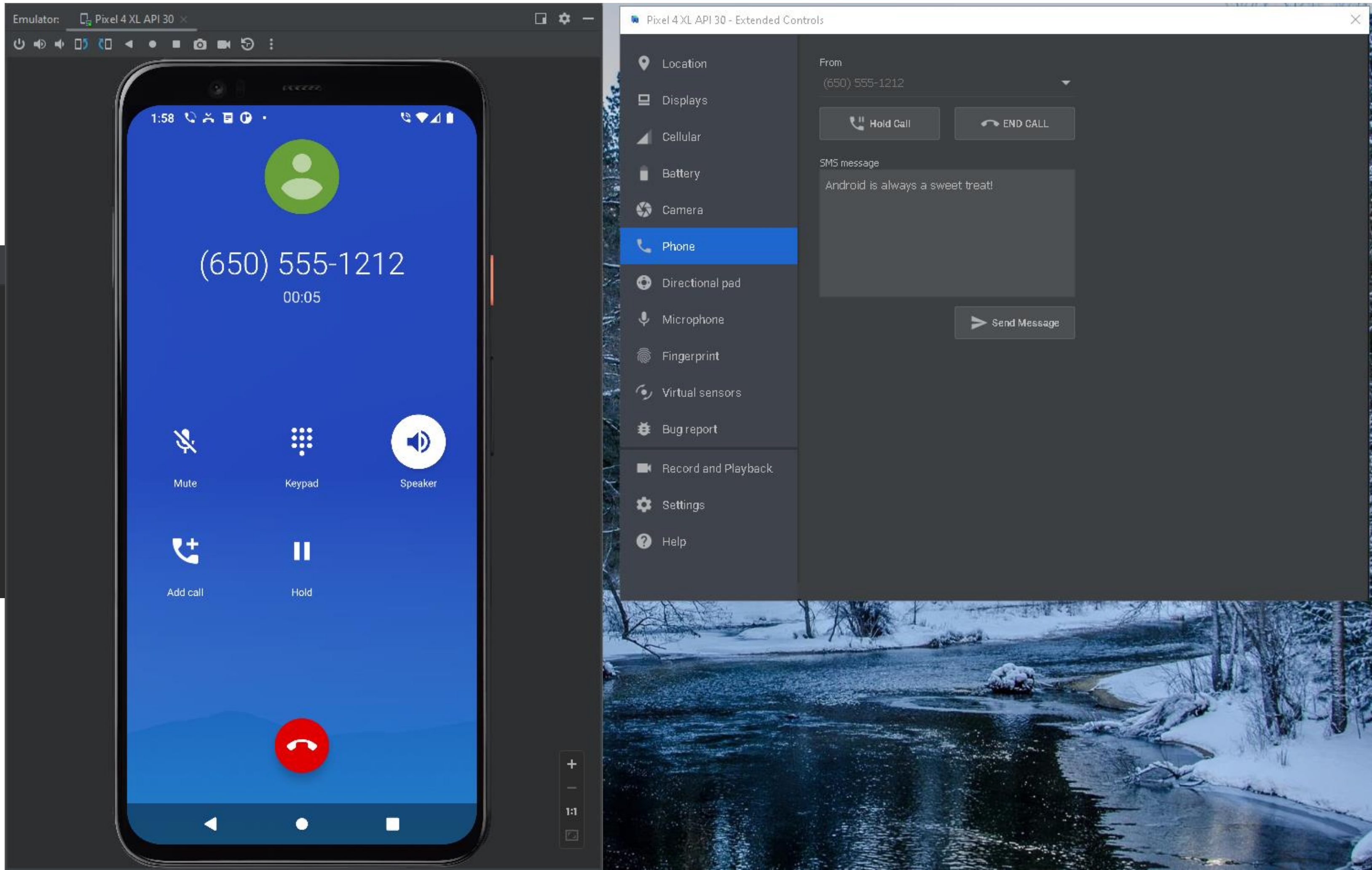
SMS message: Android is always a sweet treat!

IncomingCallStatus

- If I click the [Answer]…
- LHS: Call Received



```
I/System.out: Receiving...
I/System.out: Call Received (state)
I/System.out: Receiving...
I/System.out: Call Received (state)
I/System.out: Receiving...
I/System.out: Call Received (state)
I/System.out: Receiving...
I/System.out: Call Received (state)
```

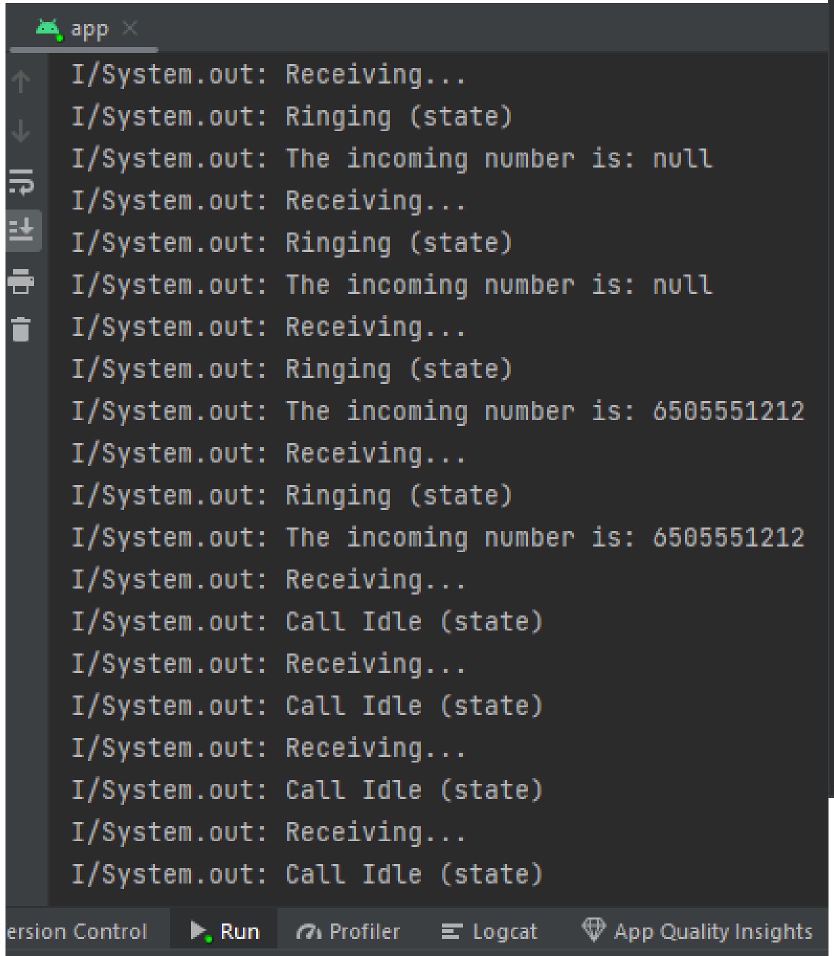


IncomingCallStatus

- What if I finished the call with someone?
  - Ringing → Call Received → Idle
  - Because you finished and hang up.

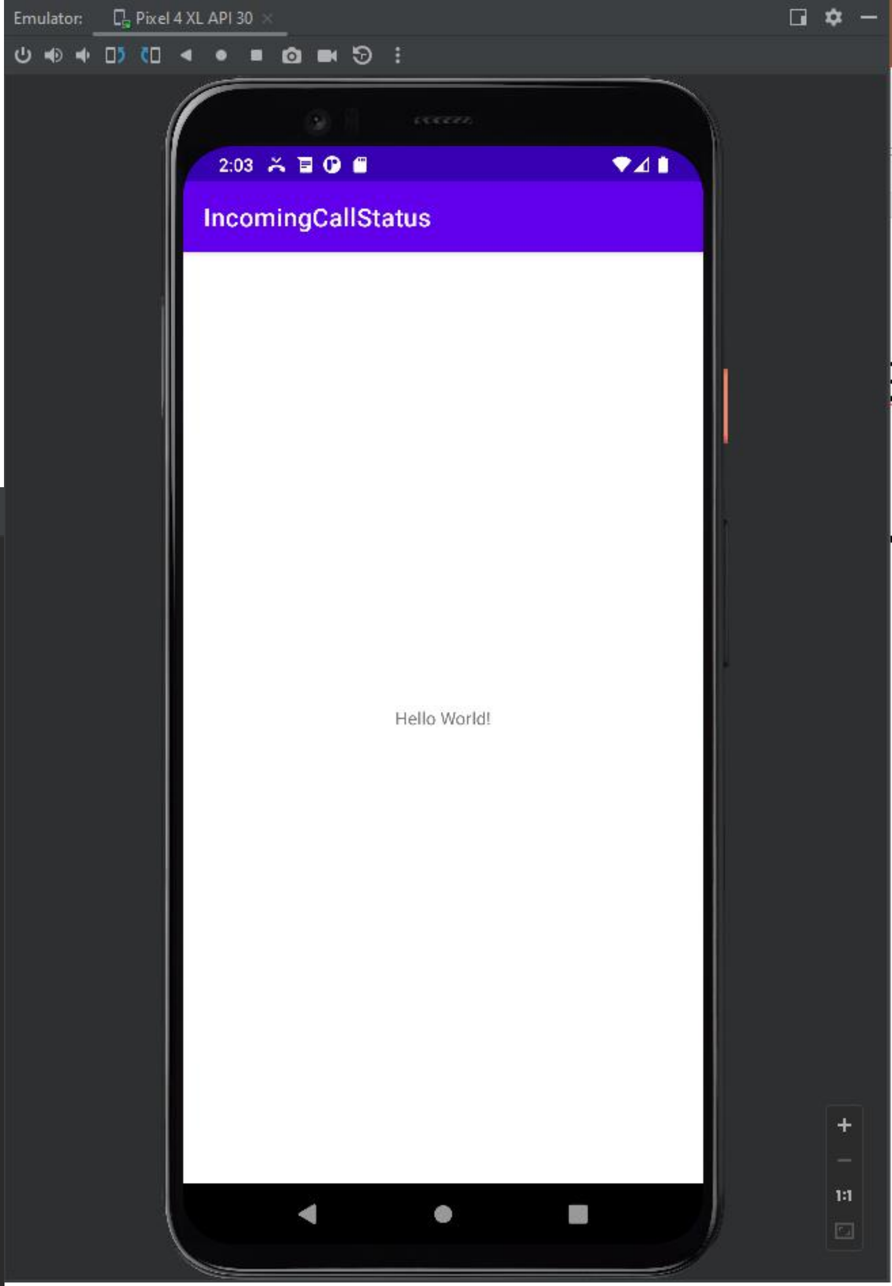
IncomingCallStatus

- If I click the [Decline]…
- Phone will go back to “Idle” from “Ringing” quickly



```
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: null
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: null
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: 6505551212
I/System.out: Receiving...
I/System.out: Ringing (state)
I/System.out: The incoming number is: 6505551212
I/System.out: Receiving...
I/System.out: Call Idle (state)
I/System.out: Receiving...
I/System.out: Call Idle (state)
I/System.out: Receiving...
I/System.out: Call Idle (state)
I/System.out: Receiving...
I/System.out: Call Idle (state)
```

app
ersion Control
Run
Profiler
Logcat
App Quality Insights



Emulator:
Pixel 4 XL API 30
2:03
IncomingCallStatus
Hello World!