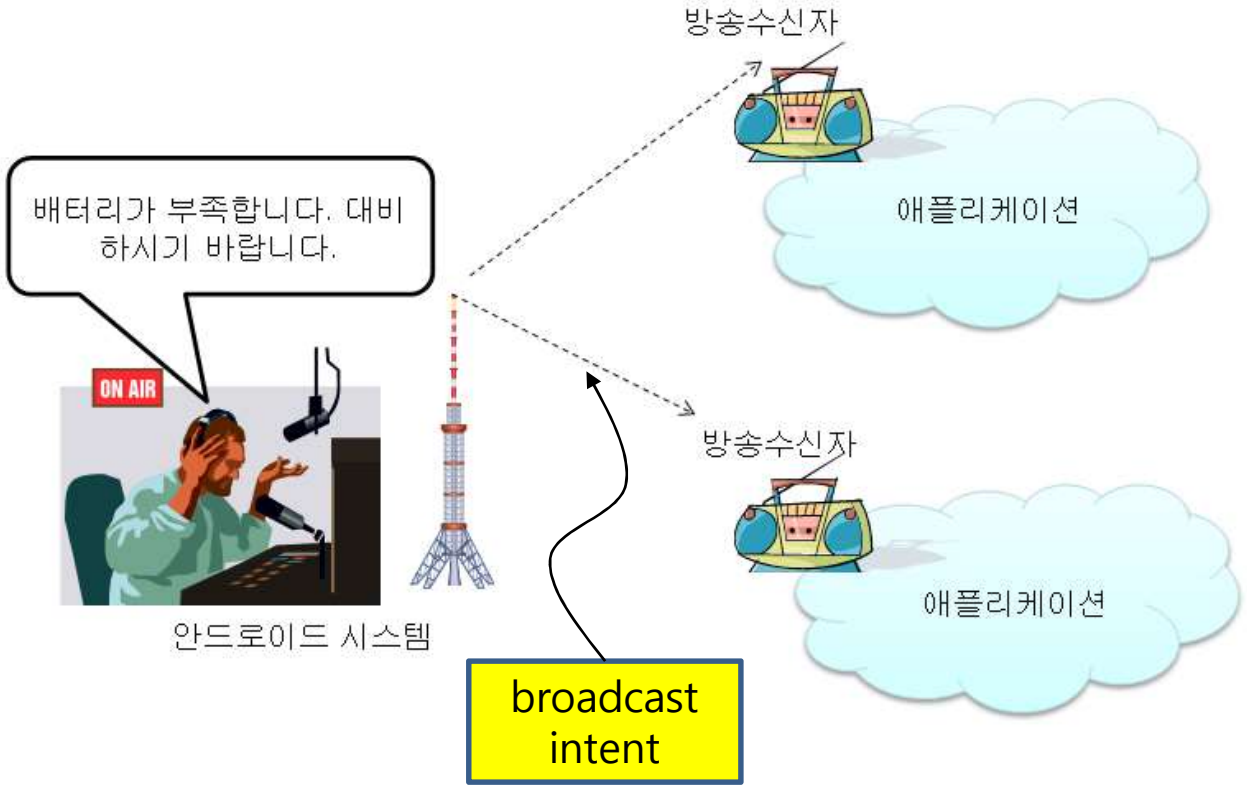
Broadcast Receiver

Mobile Software
2019 Fall

Broadcast Receiver(방송 수신자)

- Android 장치에서는 많은 이벤트들이 발생하는데,
  - 이 이벤트를 수신하는 컴포넌트가 방송 수신자



Broadcast Receiver란?

- 방송(broadcast)이란?
  - battery가 부족하거나 SD card 연결과 같은
  - 이벤트 발생 사실을 모든 응용 프로그램에게 전달.

- 방송 메시지는 누가 보낼 수 있나?
  - 대부분 Android 시스템이 보내지만, 일반 App 도 보낼 수 있음
    - **방송 메시지는 intent**

- Broadcast Receiver는 ?
  - 방송 메시지를 수신하고, 이를 처리
    - Intent filter를 사용하여 메시지 수신 의사를 밝혀야 함
  - 수신할 경우 **짧은 시간(5초 이내) 동안 background에서 처리**
    - **여러 BR에게 동시 전파되므로 각 BR이 system resource를 많이 사용하지 않도록 제한**
  - 사용자 인터페이스는 없음

broadcast 메시지 종류

액션	설명
ACTION_TIME_TICK	1분마다 보내진다.
ACTION_TIME_CHANGED	현재 시각 설정
ACTION_TIMEZONE_CHANGED	시간대 변경
ACTION_BOOT_COMPLETED	부트 완료
ACTION_PACKAGE_ADDED	패키지 추가
ACTION_PACKAGE_CHANGED	패키지 변경
ACTION_PACKAGE_REMOVED	패키지 삭제
ACTION_MEDIA_MOUNTED	외부 저장 장치 마운트 완료
ACTION_MEDIA_REMOVED	외부 저장 장치 제거
ACTION_BATTERY_CHANGED	배터리 상태 변경
ACTION_BATTERY_LOW	배터리 저충전
ACTION_POWER_CONNECTED	전원 연결
ACTION_POWER_DISCONNECTED	전원 연결 해제
ACTION_SHUTDOWN	파워 오프

Type of Broadcasts

- **sendBroadcast** ( )
  - 비동기이며 순서 없이 Broadcast 메시지 발송
  - 방송 수신자로부터 어떤 데이터도 전달받을 수 없음.
- **sendStickyBroadcast ( )**
  - 방송 수신자가 한 번 처리한 broadcast intent는 없어짐.
  - "sticky": 처리된 뒤에도 사라지지 않고 시스템에 남아 있는 broadcast intent
    - **registerReceiver** ( ) ➔ intent에 포함된 데이터를 access할 수 있음
    - **removeStickyBroadcast** ( ) ➔ sticky intent를 없앨 수 있음
- **sendOrderedBroadcast** ( )
  - 방송 수신자로부터 결과를 전달받고 싶을 때 사용
  - 정해진 순서에 따라 순서대로 방송 수신자에게 메시지 전달
    - Intent-filter의 **android: priority** 속성으로 수신 우선 순위 지정

Broadcast Intent 구현

- **Broadcast Intent 객체 생성**
  - action 정의 (필수)
  - category 정의 (선택) : **addCategory** ( )
  - data 추가 : **putExtra** ( ) → key-value pair

3.0 이전

```
val intent = Intent()
intent.action = "edu.ourincheon.sendbroadcast"
intent.putExtra("MyData", 100)
sendBroadcast(intent)
```

3.0 이후

```
val intent = Intent()
intent.action = "edu.ourincheon.sendbroadcast"
intent.flags = Intent.FLAG_INCLUDE_STOPPED_PACKAGES
intent.putExtra("MyData", 100)
sendBroadcast(intent)
```

잠깐! stopped packages

Broadcast intent 객체는 action string 을 반드시 포함해야 함
→ 객체를 식별하기 위한 고유 문자열
→ 주로 package 이름을 사용

```
val intent = Intent()
intent.action = "edu.ourincheon.sendbroadcast"
intent.flags = Intent.FLAG_INCLUDE_STOPPED_PACKAGES
intent.putExtra("MyData", 100)
sendBroadcast(intent)
```

3.0 이상 버전에서는 stopped app.의 컴포넌트는 이 broadcast Intent를 수신할 수 없다.
→ Stopped app.이란? 방금 설치되어 이전에 launch된 적이 없거나 user가 실행 중단시킨 app.

flag 설정이 필요!

Broadcast Receiver 구현

- **BroadcastReceiver** 클래스로부터 상속 받음
  - **onReceive** 메소드를 재정의

void onReceive (Context context, Intent intent)

- **Context** : BR이 실행되는 context
- **Intent** : 수신된 방송 메시지

```
class MyReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context, intent: Intent) {
        // This method is called
        // when the BroadcastReceiver is receiving an Intent broadcast.
        val message = "Broadcast intent detected " + intent.action
        Toast.makeText(context, message,
            Toast.LENGTH_LONG).show()
    }
```

Manifest file

AndroidManifest.xml

```
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.ourincheon.sendbroadcast">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="SendBroadcast"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">
        </receiver>

        <activity android:name=".MainActivity"...>
    </application>

</manifest>
```

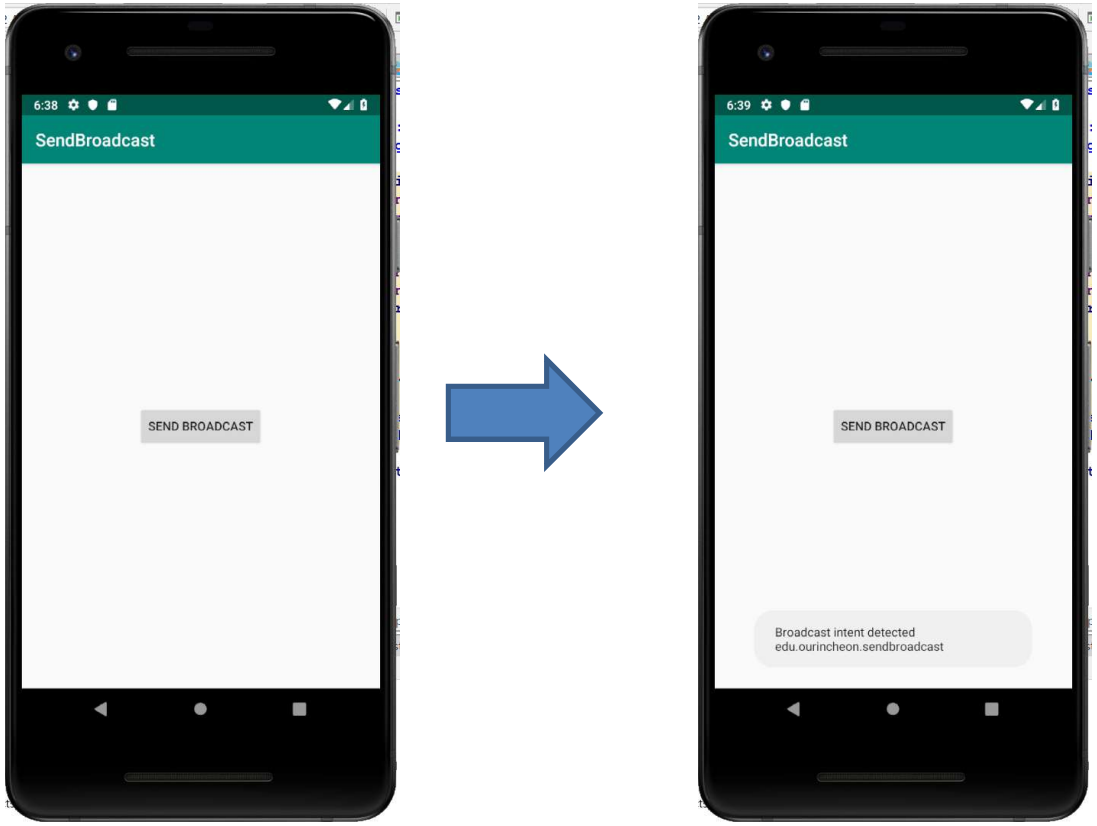
실습 준비

- 새 프로젝트 생성
  - **Activity** : **Empty Activity**
  - **Application name** : **SendBroadcast**
  - **Minimum API level : API 26** (Oreo)
  - **Activity name** : **MainActivity.kt** (자동 생성)
  - **Layout name**
    - **activity_main.xml** (자동 생성)

- 자동 생성된 레이아웃 XML 파일은 1개
  - **activity_main.xml** 의 root layout은 **ConstraintLayout**

실습 1: App.이 직접 방송

- MainActivity.kt, MyReceiver.kt 파일 2개 구현
- MainActivity.kt가 메시지를 broadcast
  - MyReceiver.kt 가 이 메시지를 수신 ➔ Toast 창 띄움



실습 1: App이 방송 - Layout

activity_main.xml

```
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="broadcastIntent"
        android:text="Send Broadcast"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

실습 1: App이 방송 - Activity

MainActivity.kt

```
class MainActivity : AppCompatActivity() {

    var receiver: BroadcastReceiver? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        configureReceiver()
    }

    private fun configureReceiver() {
        val filter = IntentFilter()
        filter.addAction("edu.ourincheon.sendbroadcast")
        receiver = MyReceiver()
        registerReceiver(receiver, filter)
    }

    fun broadcastIntent(view: View) {
        val intent = Intent()
        intent.action = "edu.ourincheon.sendbroadcast"
        intent.flags = Intent.FLAG_INCLUDE_STOPPED_PACKAGES
        sendBroadcast(intent)
    }
}
```

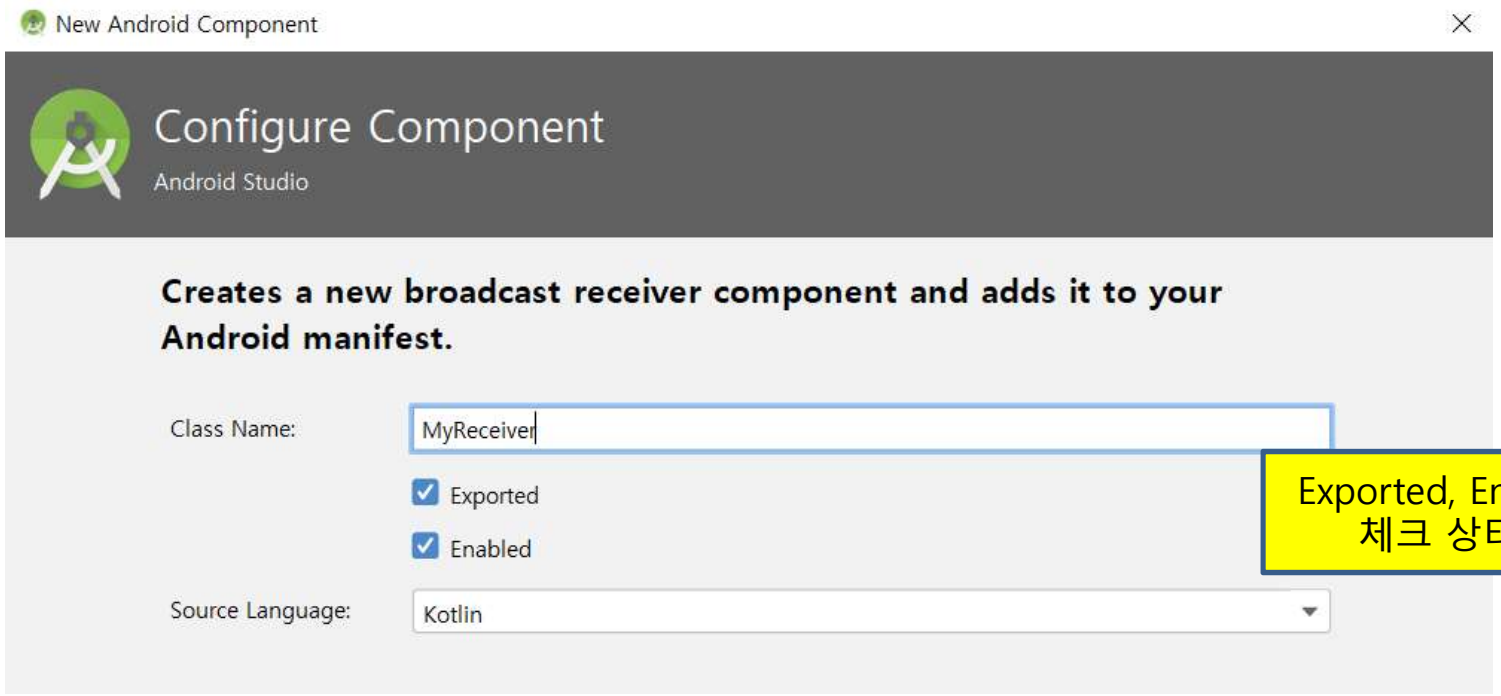
실습 1: App이 방송 - Receiver

MyReceiver.kt

```
class MyReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context, intent: Intent) {
        // This method is called
        // when the BroadcastReceiver is receiving an Intent broadcast.
        val message = "Broadcast intent detected " + intent.action
        Toast.makeText(context, message,
            Toast.LENGTH_LONG).show()
    }
}
```

Broadcast Receiver 파일 만들기

패키지 이름 > 오른쪽 버튼 > **New** > **Other** > Broadcast Receiver



Exported, Enabled 속성은 체크 상태로 둘 것!

실습 1: App이 방송 - Manifest

AndroidManifest.xml

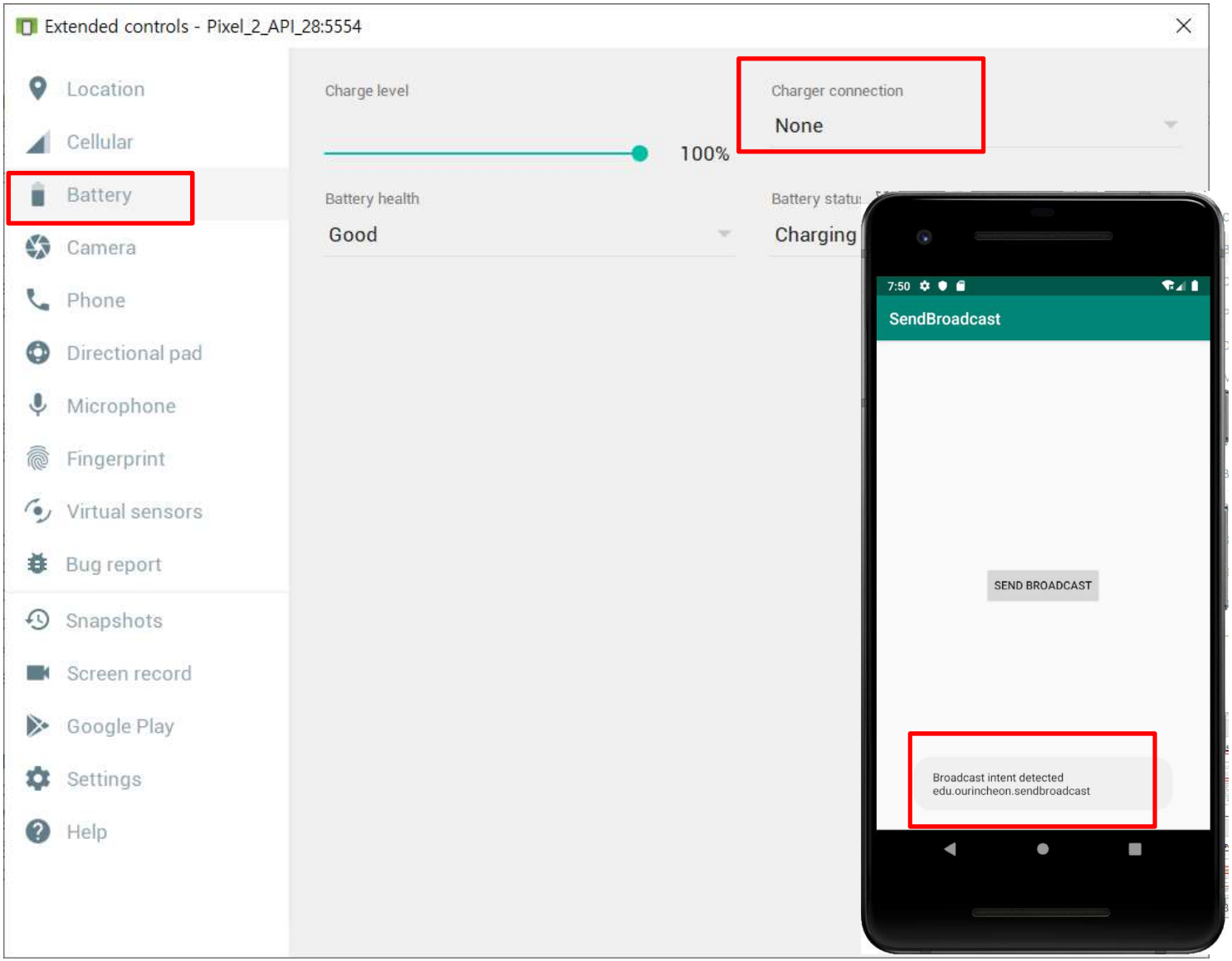
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.ourincheon.ch12_project">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Ch12_Project"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

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

        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="com.ourincheon.ch12_project.BR_INTENT" />
            </intent-filter>

        </receiver>
    </application>

</manifest>
```

실습 1-2: 방송 메시지 수신

MainActivity.kt

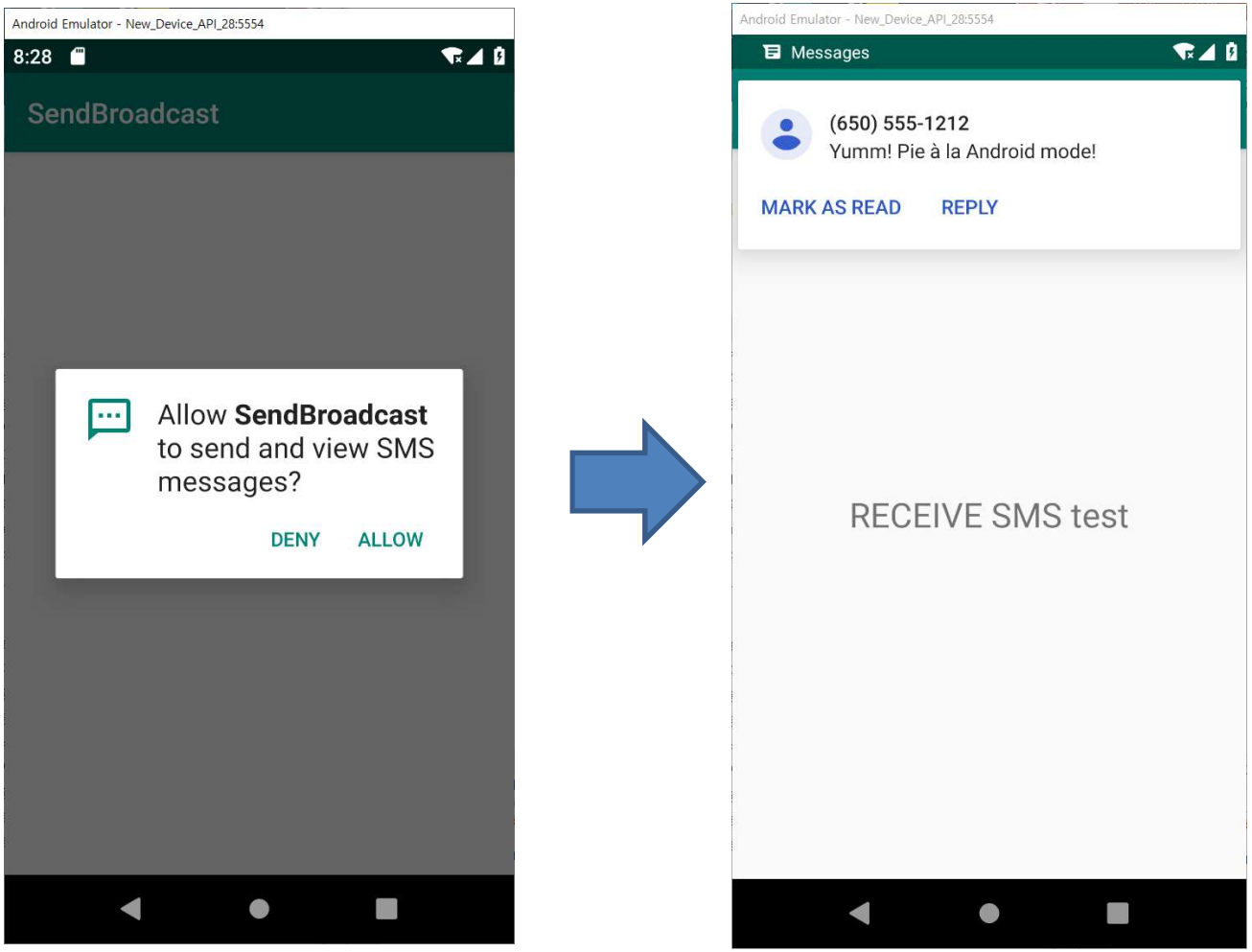
```
class MainActivity : AppCompatActivity() {

    var receiver: BroadcastReceiver? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    override fun onResume() {
        super.onResume()
        configureReceiver()
    }

    override fun onPause() {
        super.onPause()
        unregisterReceiver(receiver)
    }

    private fun configureReceiver() {
        val filter = IntentFilter()
        filter.addAction("edu.ourincheon.sendbroadcast")
        filter.addAction("android.intent.action.ACTION_POWER_DISCONNECTED")
        receiver = MyReceiver()
        registerReceiver(receiver, filter)
    }

    fun broadcastIntent(view: View) {...}
}
```

Emulator에서 전원이 연결 안 된 상황 테스트



실습 2: SMS 수신

- 문자 메시지를 수신하면, 이를 화면에 출력
- Marshmallow(6.0) 이상일 경우 permission을 요청하고 승낙을 받아야 함.



실습 2: SMS 수신 – Manifest

AndroidManifest.xml

```
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.ourincheon.sendbroadcast">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="SendBroadcast"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>

        <activity android:name=".MainActivity"...>
    </application>

</manifest>
```

실습 2: SMS 수신 – Activity (1/2)

MainActivity.kt

```
class MainActivity : AppCompatActivity() {

    private val TAG = "PermissionDemo"
    private val RECEIVE_SMS_CODE = 101


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        setupPermissions()
    }

    private fun setupPermissions() {
        val permission = ContextCompat.checkSelfPermission(this,
            Manifest.permission.RECEIVE_SMS)

        if (permission != PackageManager.PERMISSION_GRANTED) {
            Log.i(TAG, "Permission to RECEIVE SMS denied")
            makeRequest()
        }
    }

    private fun makeRequest() {...}

    override fun onRequestPermissionsResult(
        requestCode: Int,
        permissions: Array<out String>,
        grantResults: IntArray
    ) {...}
}
```

실습 2: SMS 수신 – Activity (2/2)

MainActivity.kt

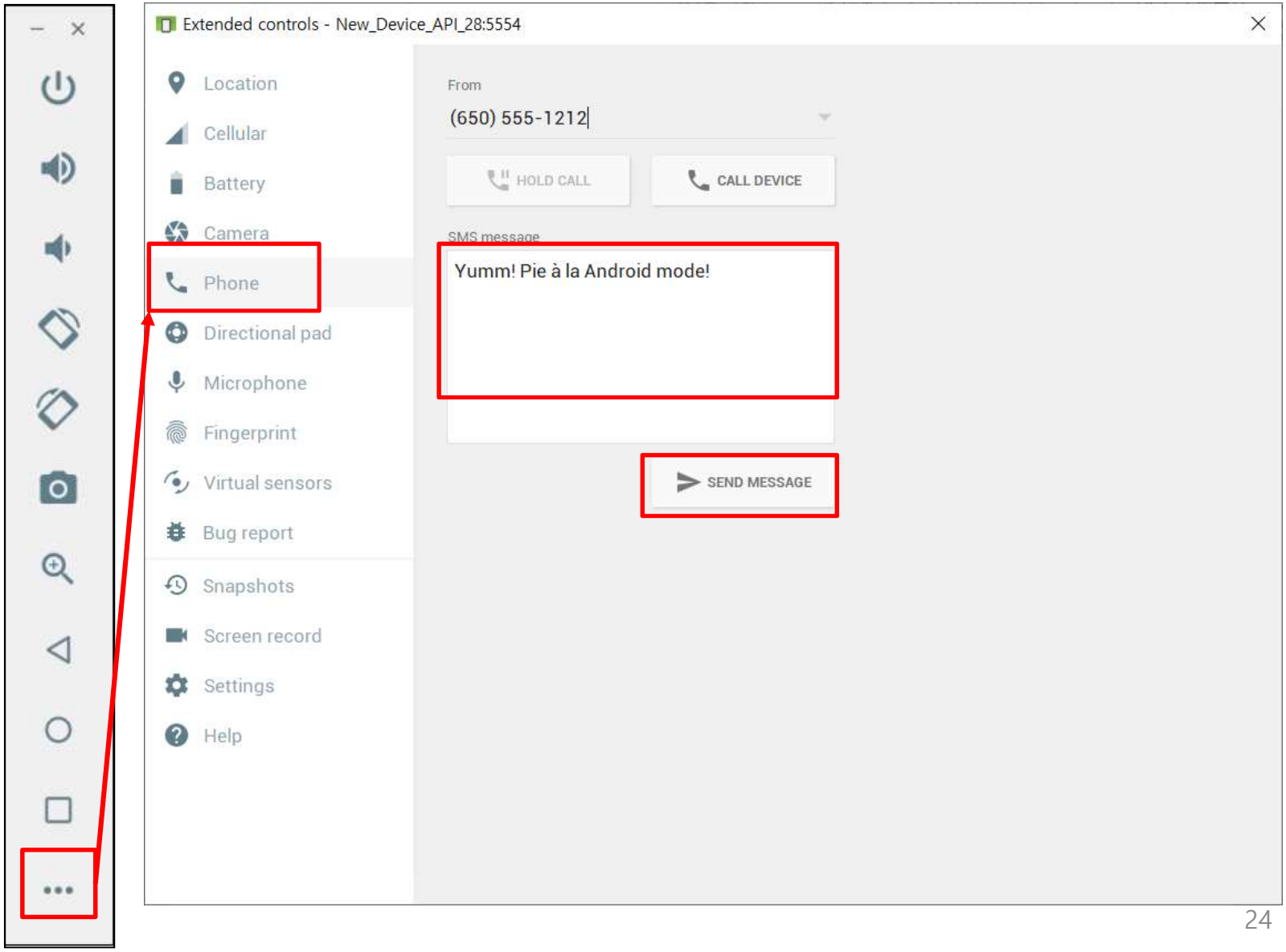
```
override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<out String>,
    grantResults: IntArray
) {
    when (requestCode) {
        RECEIVE_SMS_CODE -> {
            if (grantResults.isEmpty() ||
                grantResults[0] != PackageManager.PERMISSION_GRANTED) {
                Log.i(TAG, "Permission has been denied by user")
            } else {
                Log.i(TAG, "Permission has been granted by user")
            }
        }
    }
}
```

실습 2: SMS 수신 - Receiver

MyReceiver.kt

```
class MyReceiver:BroadcastReceiver() {

    override fun onReceive(context: Context, intent: Intent) {

        if (intent.action == "android.provider.Telephony.SMS_RECEIVED") {
            val bundle: Bundle? = intent.extras
            if (bundle != null) {
                var msgs:Array<android.telephony.SmsMessage>? =
                    Telephony.Sms.Intents.getMessagesFromIntent(intent)
                if (msgs != null) {
                    for (i in msgs.indices) {
                        var currentSMS: android.telephony.SmsMessage
                                = msgs[i]
                        val senderNo = currentSMS.originatingAddress
                        val message = currentSMS.displayMessageBody
                        Toast.makeText(
                            context,
                            "senderNum: " + senderNo +
                                    " :\n message: " + message,
                            Toast.LENGTH_LONG
                        ).show()
                    }
                }
            }
        }
    }
}
```

Emulator에서 문자 메시지 테스트



Broadcast Receiver 등록과 실행

- BR(*Broadcast Receiver*)은 생성된 직후부터 broadcast 메시지 발생 여부를 감시
  - Manifest 파일에 BR을 등록

- Android platform은 단말에 설치된 모든 응용 프로그램의 manifest 파일을 검색
  - 해당 BR을 호출하여 실행
    - BR의 실행 여부와는 상관없음

Activity가 실행 중에만 BR을 동작하게 하려면?

- Manifest 파일에 등록하는 대신 프로그램에서 아래 메소드를 호출하여 직접 BR을 등록하고 해지

  – BR 등록
    - **onResume** 메소드에서 등록

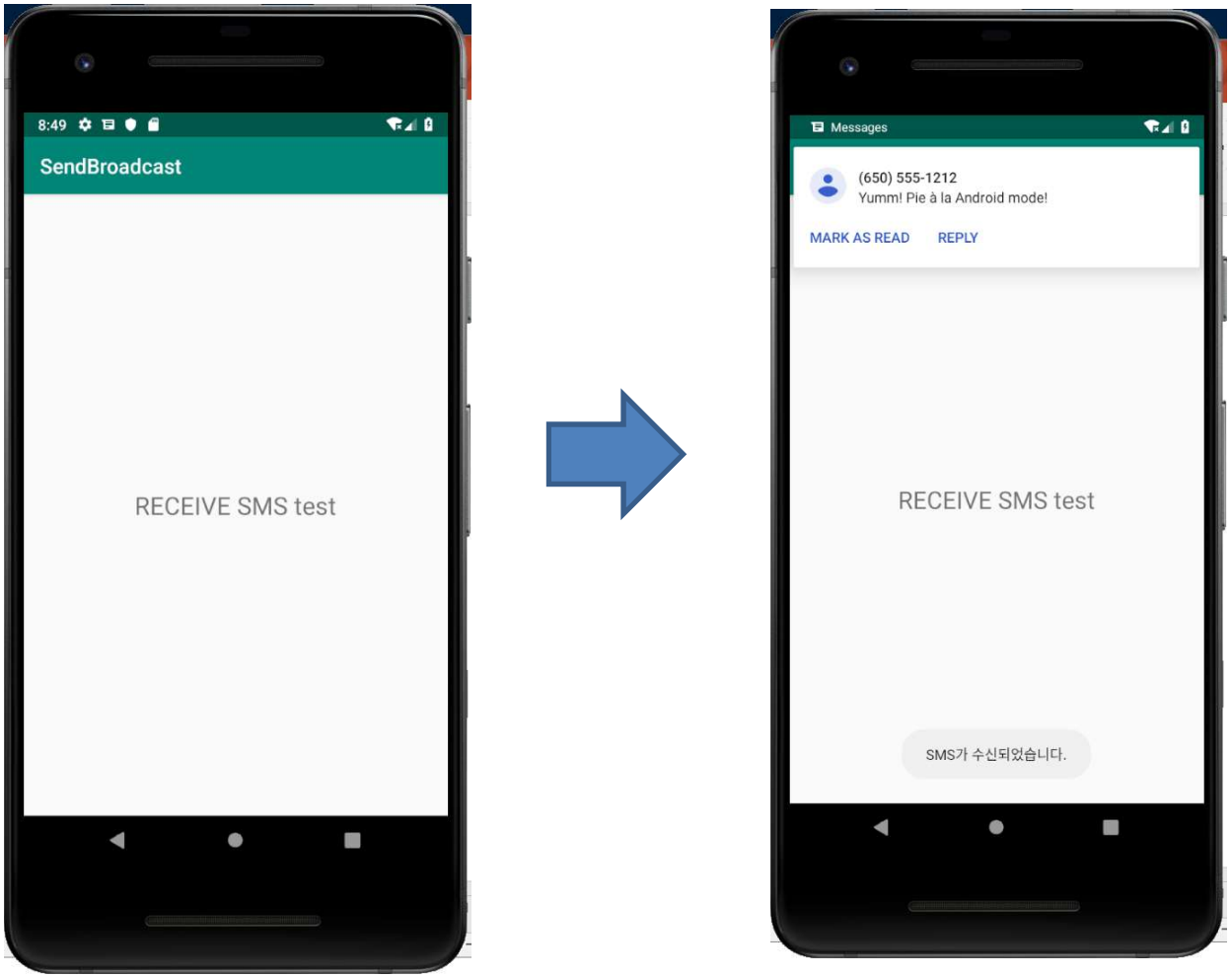
    intent registerReceiver (BroadcastRecevier receiver, IntentFilter filter)

  – BR 등록 해제
    - **onPause** 메소드에서 해제

    void unregisterReceiver (BroadcastRecevier receiver)

실습 3: DynamicBR

- 문자 메시지를 수신하면, 이를 화면에 출력



실습 3: DynamicBR – Activity

MainActivity.kt

```
class MainActivity : AppCompatActivity() {

    private val TAG = "PermissionDemo"
    private val RECEIVE_SMS_CODE = 101
    var receiver: BroadcastReceiver? = null

    class MyReceiver:BroadcastReceiver() {

        override fun onReceive(context: Context, intent: Intent) {

            Toast.makeText(context, "SMS가 수신되었습니다.",
                Toast.LENGTH_LONG).show()
        }
    }

    override fun onCreate(savedInstanceState: Bundle?) {...}

    override fun onResume() {
        super.onResume()
        val filter = IntentFilter()
        filter.addAction("android.provider.Telephony.SMS_RECEIVED")
        receiver = MyReceiver()
        registerReceiver(receiver, filter)
    }

    override fun onPause() {
        super.onPause()
        unregisterReceiver(receiver)
    }
```

실습 3: DynamicBR – Manifest

AndroidManifest.xml

```
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.ourincheon.sendbroadcast">

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

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

        <activity android:name=".MainActivity"...>
    </application>

</manifest>
```