

모바일프로그래밍

- 안드로이드(Android) -

○안드로이드 기능/실습 - 21

- Broadcast Receiver

○안드로이드 기능/실습 - 22

- 태스크, 인텐트 플래그

○안드로이드 기능/실습 - 23

- Notifications

□ 안드로이드 기능/실습 - 21 **- BROADCAST RECEIVER**

○브로드캐스트 리시버(Broadcast receiver)

- 시스템에 변화가 생겼을 때 이를 알려주는 방식
- 시스템 서비스를 위해 시스템 이벤트를 전역으로 방송할 때 이를 이용
- 애플리케이션들이 항상 방송에 주의를 기울이고 있으면 휴대 단말에서 발생하는 여러 가지 변화에 대한 제어가 가능
- 서비스나 알림과 비슷한 부류
- 방송을 수신하는 애플리케이션은 브로드캐스트 리시버를 갖고 있고 방송 수신을 대기
- 이것은 사용자와 직접 상호작용하지 않음
- 브로드캐스트 리시버는 사용자 인터페이스를 보여주지 않지만, 수신한 정보에 응답하는 액티비티를 시작하거나, 사용자에게 알려주기 위한 알림 매니저를 사용할 수 있음

□ 브로드캐스트 액션

액션	설명
ACTION_BATTERY_CHANGED	배터리 잔량 충전도가 변화했을 때의 통지
ACTION_BOOT_COMPLETED	장치가 부팅 절차를 완료할 때 한번 발생 RECEIVE_BOOT_COMPLETED 권한 필요
ACTION_DATE_CHANGED	날짜가 바뀔 때 통지
ACTION_TIMEZONE_CHANGED	시간대가 바뀔 때마다
ACTION_TIME_CHANGED	시각이 바뀔 때 통지
ACTION_CAMERA_BUTTON	카메라 버튼이 클릭할 때 발생
ACTION_MEDIA_BUTTON	미디어 버튼을 클릭할 때 발생
ACTION_MEDIA_EJECT	외부 미디어를 꺼내기로 결정할 경우 발생
ACTION_MEDIA_MOUNTED & ACTION_MEDIA_UNMOUNTED	저장 미디어가 장치에 성공적으로 추가되거나 제거될 때 마다 방송
ACTION_SCREEN_OFF & ACTION_SCREEN_ON	화면이 꺼지거나 켜질 때 통지

□ 브로드캐스트 리시버

○BroadcastReceiver 클래스를 확장한 서브 클래스 형태로 구현

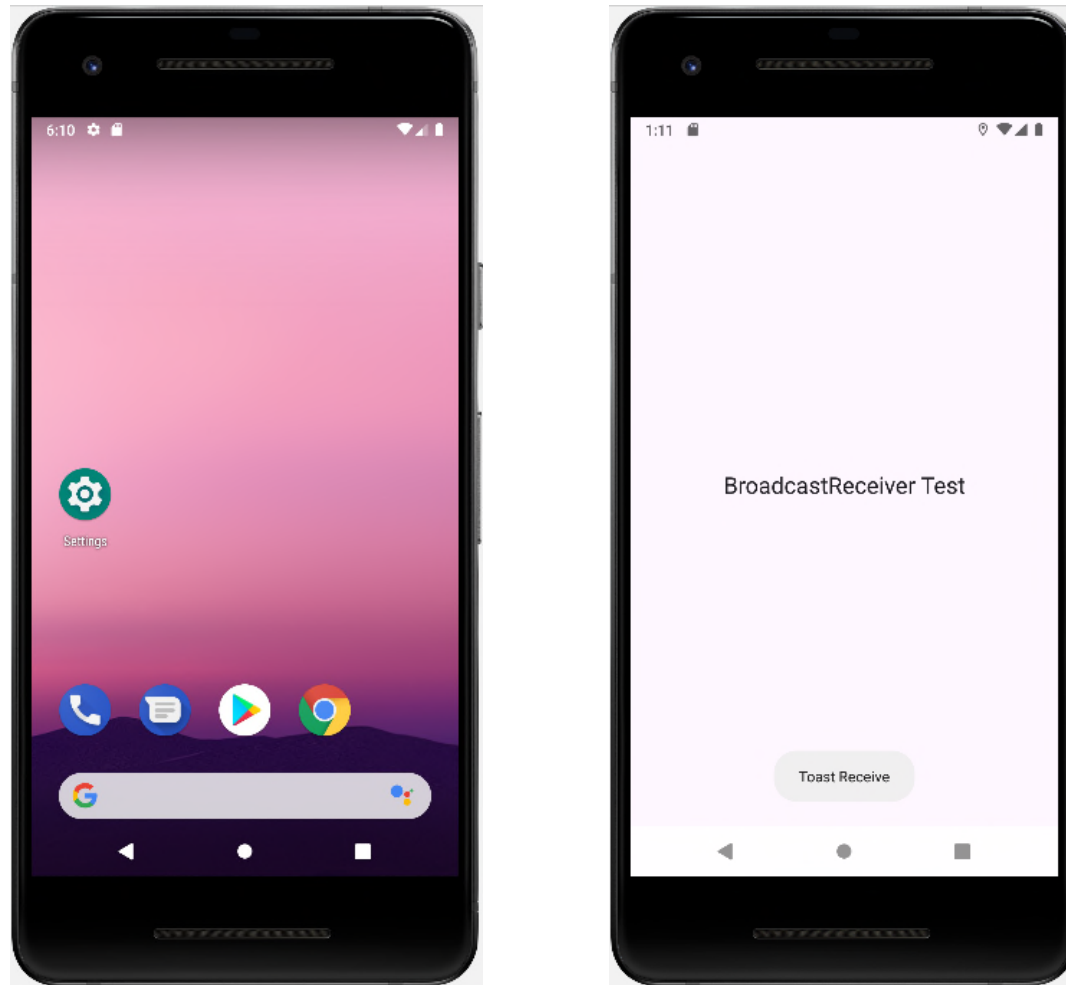
- onReceive() 오버라이딩

○매니페스트 파일에 등록 필요

- Receiver 엘리먼트를 application 엘리먼트에 추가
- Intent-filter 엘리먼트를 receiver 엘리먼트에 포함시켜 인텐트 대상이 되는 컴포넌트를 알려주어야 함
- 안드로이드 버전 상승에 따라 정적 등록과 동적 등록으로 나누어 짐

□ 실행 모습

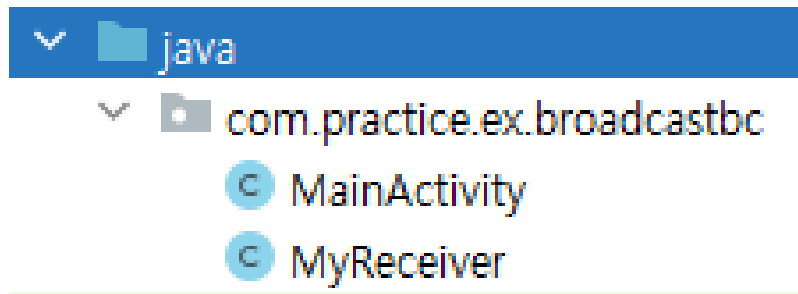
- 해당 예제는 단순히 브로드캐스트리시버 개념과 구현 과정을 설명하기 위함 → 안드로이드9 버전으로 실행
 - 변경사항은 실습으로 진행



□ 코드 분석 - 구성

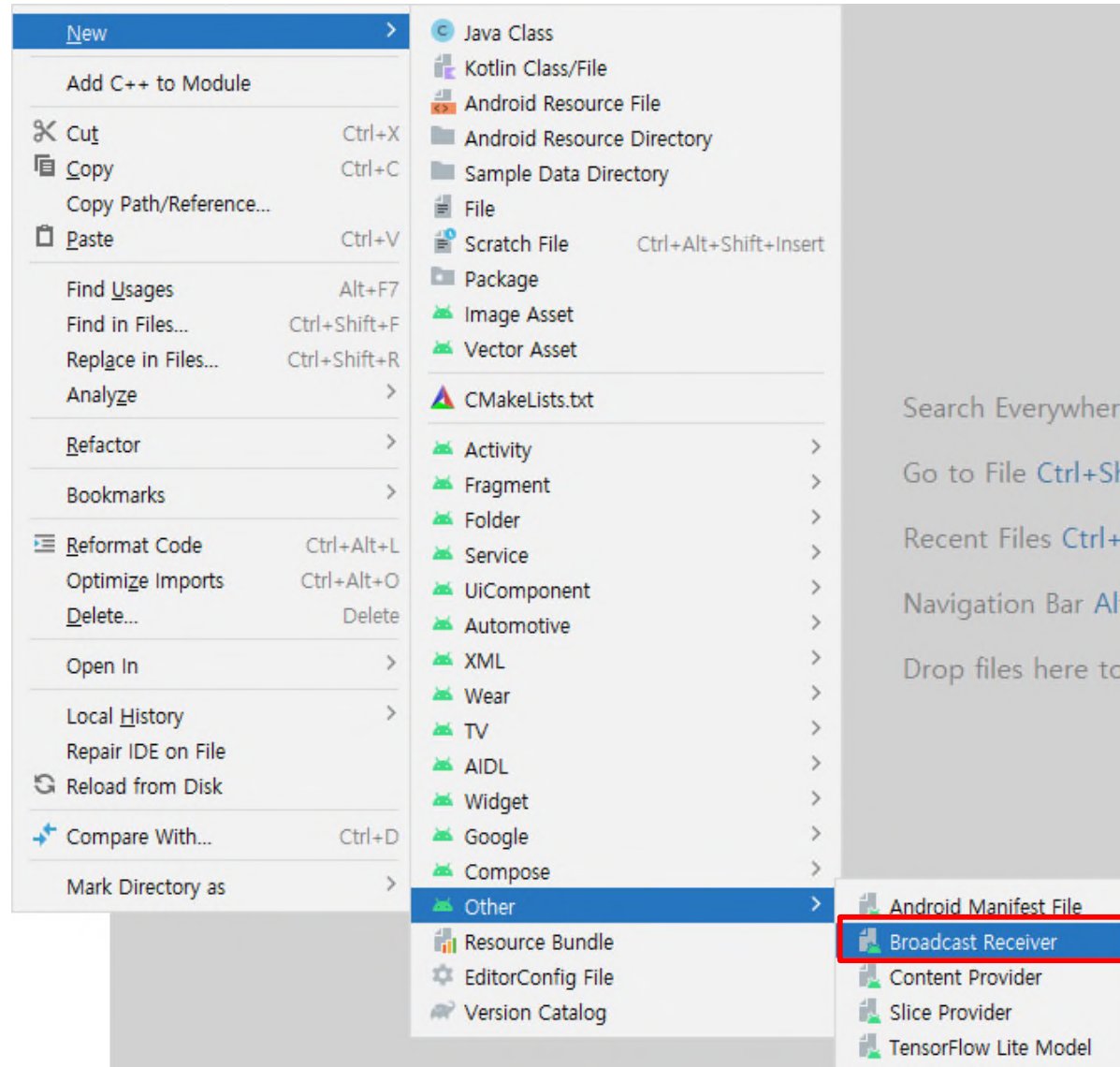
○수신된 메시지를 실행할 액티비티와 브로드캐스트리시버 구현 부분으로 별도 구성

- Broadcast Receiver 추가는 메뉴에서 Broadcast Receiver를 선택하여 추가



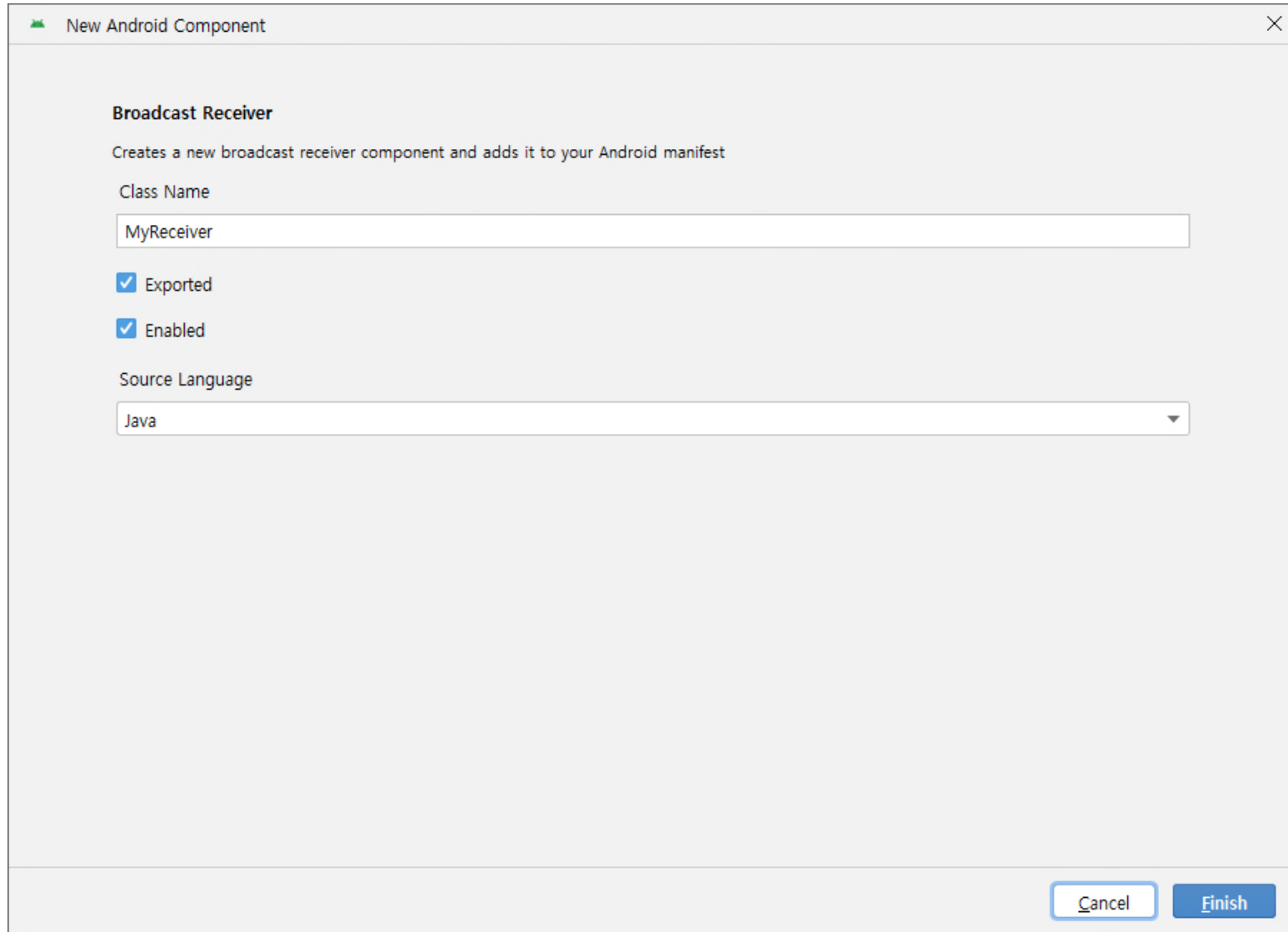
□ Broadcast Receiver 추가 과정

○ 코드에서 Other → Broadcast Receiver 선택



□ *Broadcast Receiver* 추가 과정

○ 생성할 이름 작성



New Android Component

Broadcast Receiver

Creates a new broadcast receiver component and adds it to your Android manifest

Class Name

MyReceiver

☒ Exported

☒ Enabled

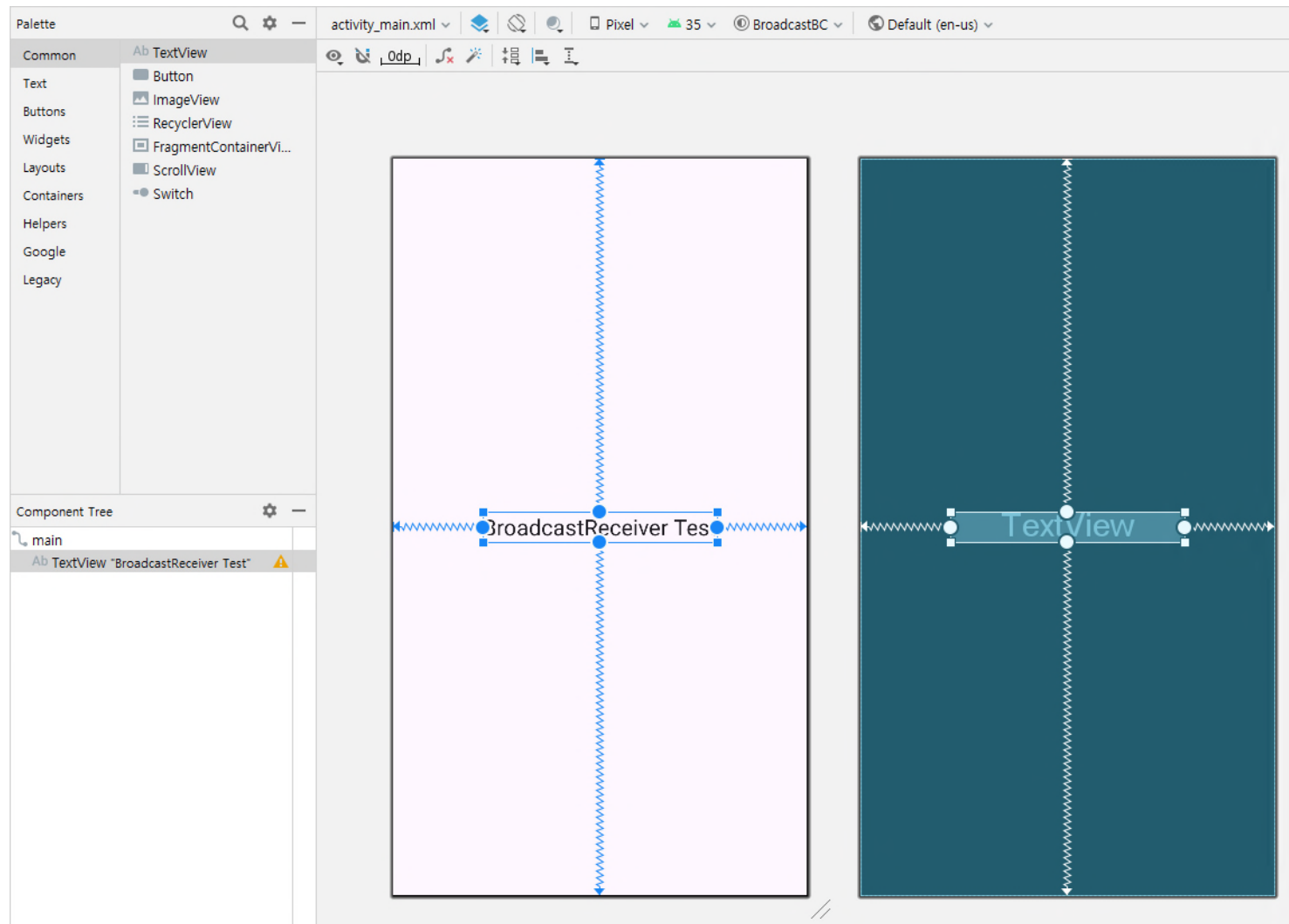
Source Language

Java

Cancel Finish

□ 레이아웃 구성

- 브로드캐스트 리시버를 테스트하기 위한 코드이기 때문에 UI는 TextView 1개로 간단히 구성



□ 레이아웃

```
<?xml version="1.0" encoding="utf-8"?>
<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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="BroadcastReceiver Test"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

여기서는 텍스트 속성을
android:textAppearance를
이용하였음

□ AndroidManifest.xml

```
<?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.RECEIVE_BOOT_COMPLETED"/>
```

부팅 완료 시점을 알아내기 위한 권한 설정

```
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/BroadcastBC"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.BroadcastBC"
    tools:targetApi="31">
```

리시버 등록 과정을 수행

```
<receiver
    android:name=".MyReceiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED"/>
    </intent-filter>
</receiver>
```

<개별 공부해야 할 사항>

-인텐트필터

- 인텐트필터는 앱컴포넌트가 받고자 하는 인텐트를 정의하는 수단

❑ *AndroidManifest.xml*

```
}      <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>
|  </application>
|
|</manifest>
```


□ 코드 분석 / 설명 - MainActivity

```
package com.practice.ex.broadcastbc;

import android.os.Bundle;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
            return insets;
        });
    }
}
```

- 다양한 형태로 구현 가능
- 여기서는 Broadcast Receiver를 이용하는 방법만 소개하기 때문에 Activity에 별도의 코드 작성 없음

□ 코드 분석 / 설명 - MyReceiver

```
package com.examples.broadcastbc;
```

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.widget.Toast;
```

```
1 usage
```

```
public class MyReceiver extends BroadcastReceiver {
```

```
    @Override
```

```
    public void onReceive(Context context, Intent intent) {
```

```
        // TODO: This method is called when the BroadcastReceiver is receiving
        // an Intent broadcast.
```

```
        if(intent.getAction().equals(Intent.ACTION_BOOT_COMPLETED)) {
```

```
            Toast.makeText(context, text: "Toast Receive", Toast.LENGTH_LONG).show();
```

```
            Intent intent01 = new Intent(context, MainActivity.class);
```

```
            intent01.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK
                               | Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

```
            context.startActivity(intent01);
```

<개별 공부해야 할 사항>

-인텐트 플래그

• 인텐트플래그는 액티비티의 실행, 관리에 대한 정보 등을

설정할 수 있고 이를 통해 액티비티 스택, 인텐트 상태 등을 컨트롤 할 수 있음

해당 예제는 안드로이드 10부터는 실행되지 않음

- 백그라운드에서 액티비티를 실행 X, 알림 기능을 이용하여 처리하여야 함
- 브로드캐스트 리시버 동작 과정을 보여주기 위한 단순 예제임

원하는 브로드캐스트 메시지가 도착하면 호출
애플리케이션 Context를 사용할 수 있으며
Intent를 통해 동작을 구분할 수 있음

등록한 브로드캐스트리시버 액션이 여러 개인 경우 구분이 필요

인텐트 플래그를 이용하여 상황에 맞게 동작하도록 설정

- 여기서는 여러 설정을 동시에 적용하는 방법을 소개하기

위해서 여러 플래그를 작성

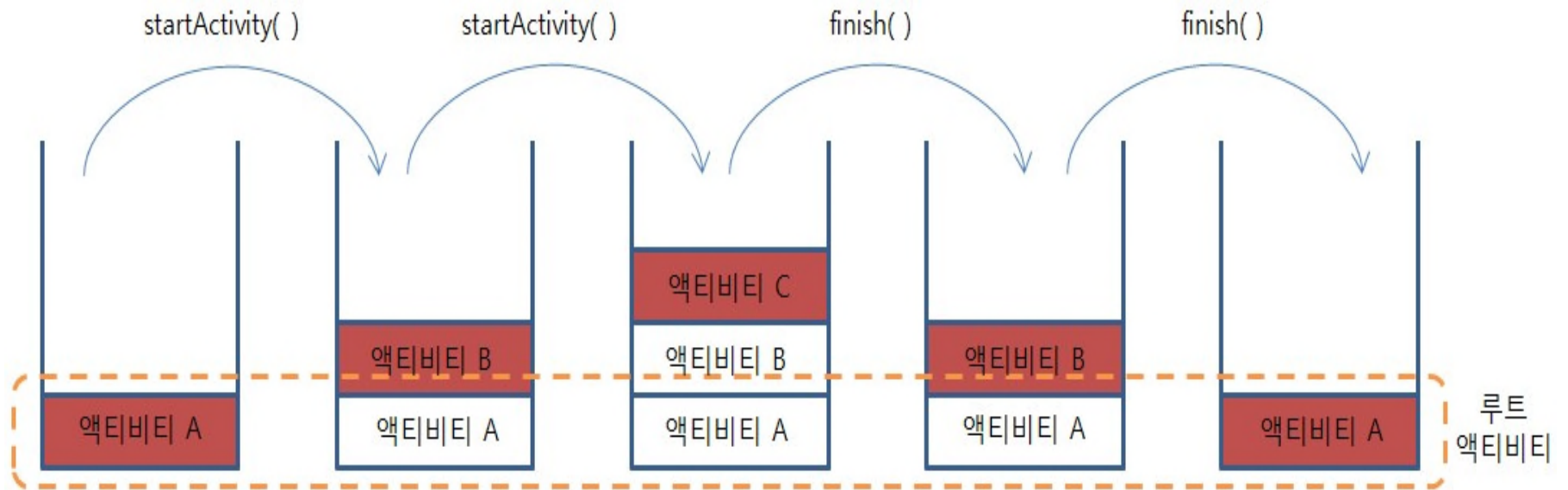
□ 안드로이드 기능/실습 - 22 **- 태스크, 인텐트 플래그**

○ 액티비티와 태스크

- 액티비티: 같은 애플리케이션 내에 존재하는 액티비티 뿐만 아니라 다른 애플리케이션 내에 존재하는 액티비티까지 **호출 가능**
예) 서적 관리 애플리케이션 = 서적 관리 일반 기능 + 특정 기능 (바코드 스캔)
- 한 애플리케이션에서 다른 애플리케이션의 컴포넌트를 거의 자유 자재로 접근 가능
→ 파일의 구성 면에서는 애플리케이션의 경계가 뚜렷함, 실제로 애플리케이션의 실행 면에서는 애플리케이션간 경계가 없음
- 각 컴포넌트들, 특히 화면에 표시되면서 사용자와 상호작용하는 액티비티는 애플리케이션 단위보다 태스크(Task) 단위로 관리
- 태스크(Task): 사용자가 실질적으로 “하나의 애플리케이션처럼” 느끼는 **액티비티들의 집합**임.

□ 액티비티 스택

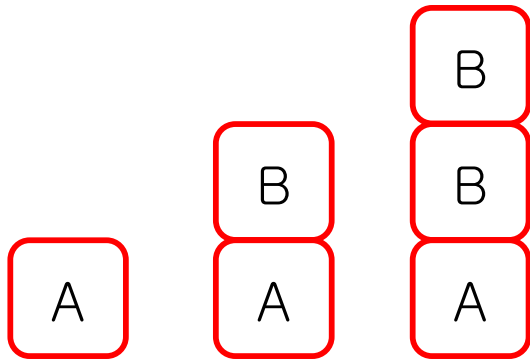
○ 액티비티 스택



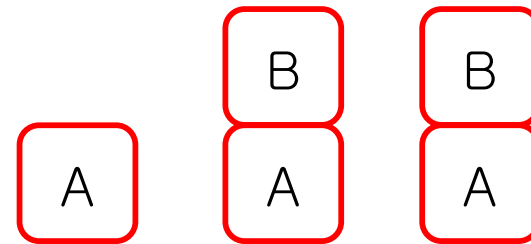
□ 액티비티 플래그

○ FLAG_ACTIVITY_SINGLE_TOP

- 재호출 하는 경우, 액티비티를 재사용



NO_FLAG

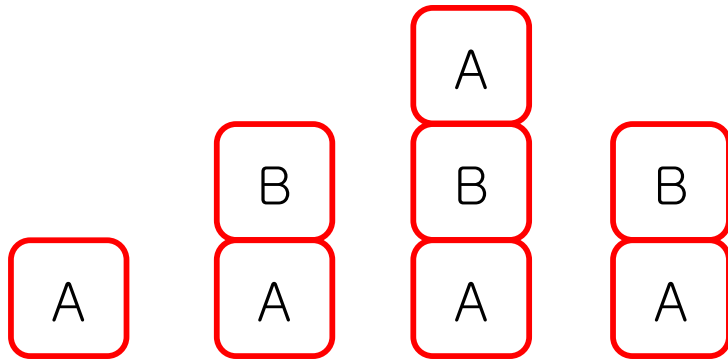


FLAG_ACTIVITY_SINGLE_TOP

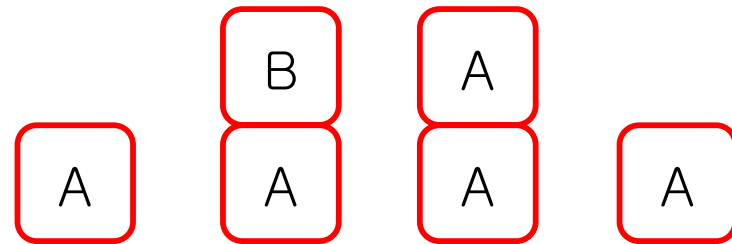
□ 액티비티 플래그

○ FLAG_ACTIVITY_NO_HISTORY

- 액티비티 사용 기록을 남기지 않음



NO_FLAG

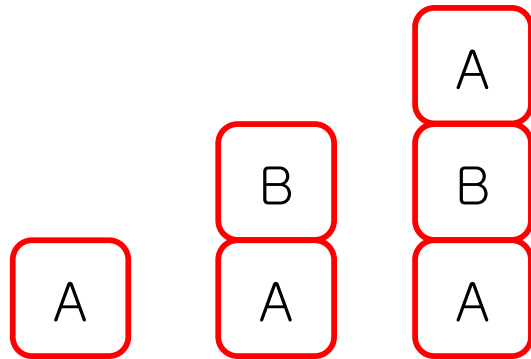


FLAG_ACTIVITY_NO_HISTORY

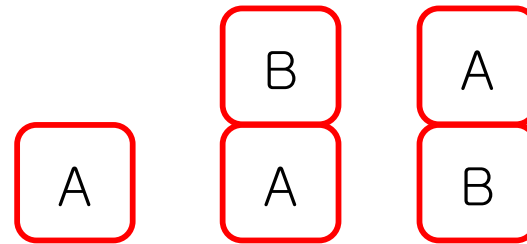
□ 액티비티 플래그

○ FLAG_ACTIVITY_REORDER_TO_FRONT

- 기존 액티비티를 가장 앞으로 가지고 옴



NO_FLAG

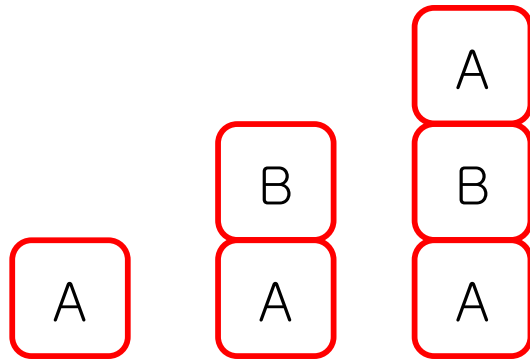


FLAG_ACTIVITY_REORDER_TO_FRONT

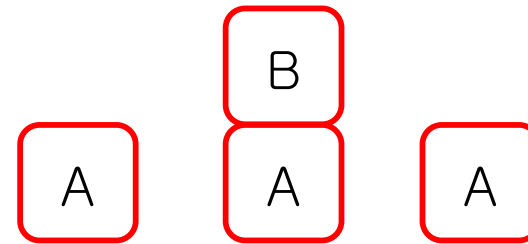
□ 액티비티 플래그

○ FLAG_ACTIVITY_CLEAR_TOP

- 액티비티가 존재하는 경우, 상위 액티비티를 모두 제거



NO_FLAG



FLAG_ACTIVITY_CLEAR_TOP

□ 안드로이드 기능/실습 - 23 **- NOTIFICATIONS**

□ 알림 (Notification)

○알림(Notification)

- 시스템이 사용자에게 무엇인가를 알릴 때 사용
- SMS, 약속시간 ...

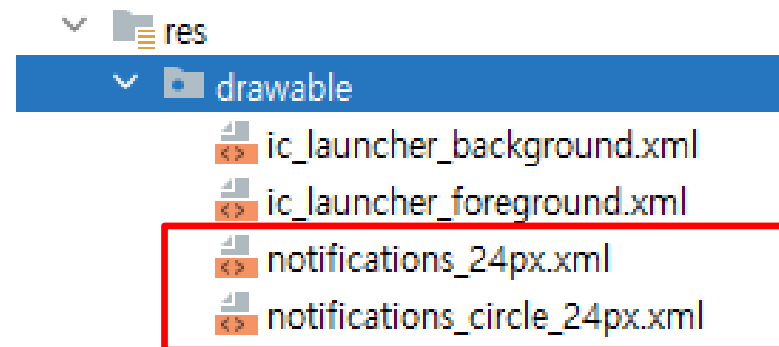
○NotificationManager 클래스

- 알림 메시지를 사용하기 위해 필요
- getSystemService() 메소드의 NOTIFICATION_SERVICE를 지정 함으로써 얻을 수 있음
- 알림 매니저를 보내기 위해 notify() 메소드에 Notification 클래스의 인스턴스를 매개변수로 전달하여 호출
- 알림 메시지를 표시하는데 필요한 각종 정보를 담고 있음
- 재생하는 오디오 타입, 소리, 진동, 점멸의 기본 통지 방법, 등... 다양한 필드가 있어 여러 기능을 갖는 알림 서비스를 만들 수 있음

□ 이미지 준비

○알림 표시에 사용할 이미지 준비

- 5.0 이후에는 알파값이 있는 단색 아이콘이어야 정상 표현
- 여기서는 테스트만을 진행하기 때문에 mipmap 이미지, 또는 기타 다른 이미지를 사용
- 수업 자료로 이용한 이미지는 자료실 참고
- 알림 기능의 경우, 안드로이드8(Oreo)과 안드로이드13에서 동작 코드 관련 변경이 있음



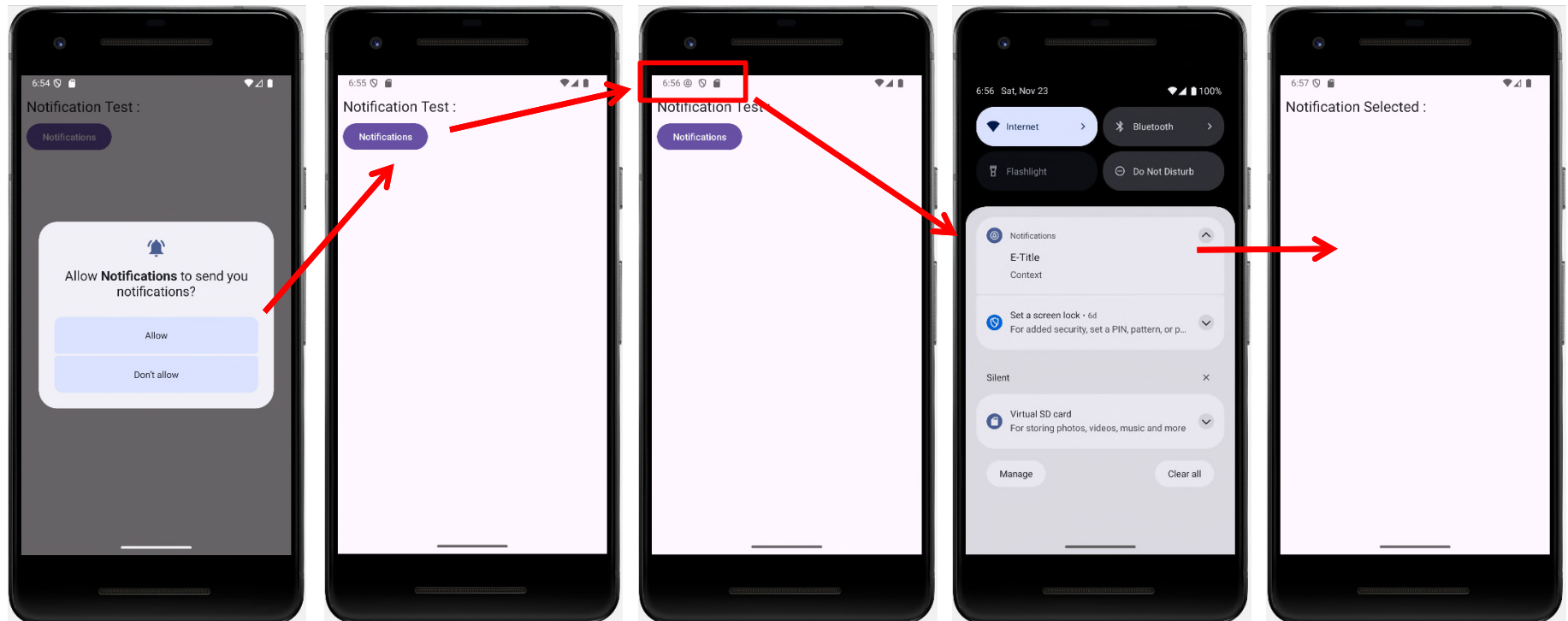
□ 실행 모습

○여기서는 버전에 따라 알림 아이콘을 별도로 설정

□ 안드로이드 버전8 미만에서 수행한 경우

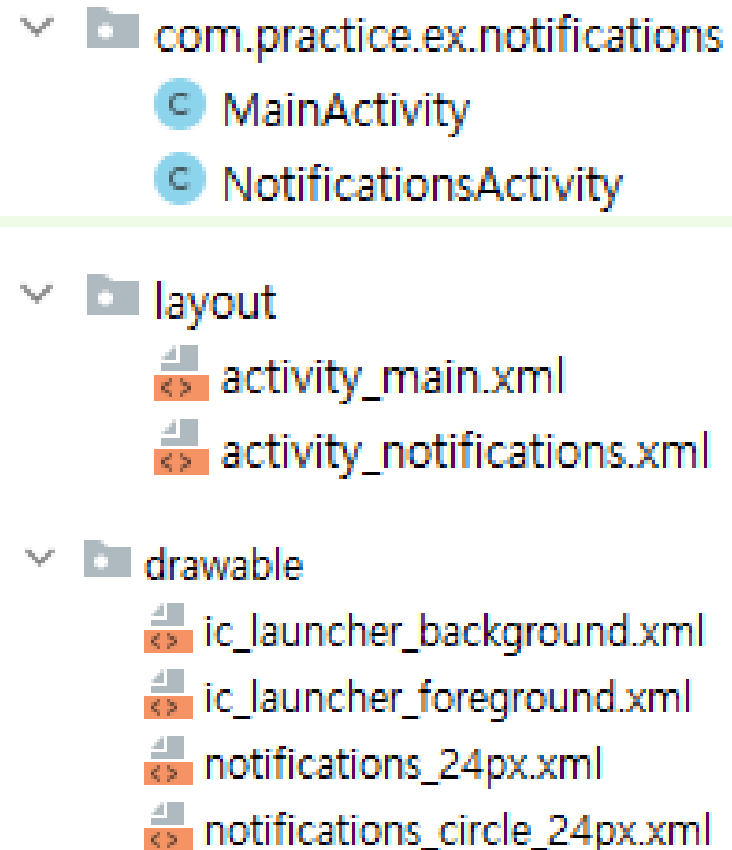


□ 안드로이드 버전8 이상에서 수행한 경우



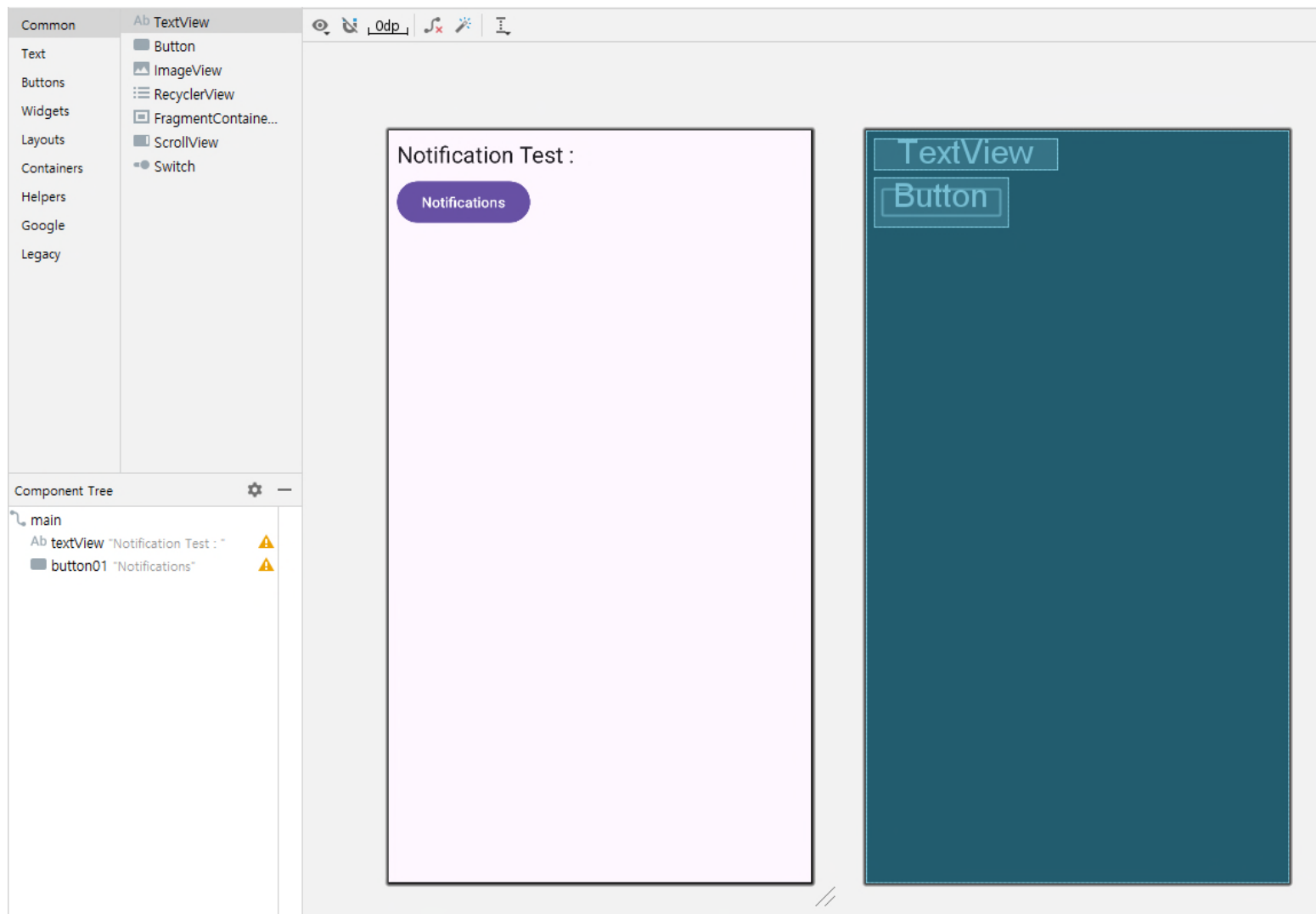
□ 프로젝트 구조

- Notifications을 생성할 Activity 1개
- 발생한 알림(Notifications)을 선택할 때, 해당 정보를 표시할 Activity 1개
- 각 Activity에 대한 레이아웃
- Notifications에 사용할 이미지 1개 이상
 - 기존 drawable 파일 이용 또는 생성한 이미지 활용



□ 레이아웃 구성 - *activity_main.xml*

○ Notification 기능을 확인하기 위함으로 해당 기능을 발생시키기 위한 버튼 1개로 구성



□ 레이아웃 - *activity_main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Notification Test : "
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

□ 레이아웃 - *activity_main.xml*

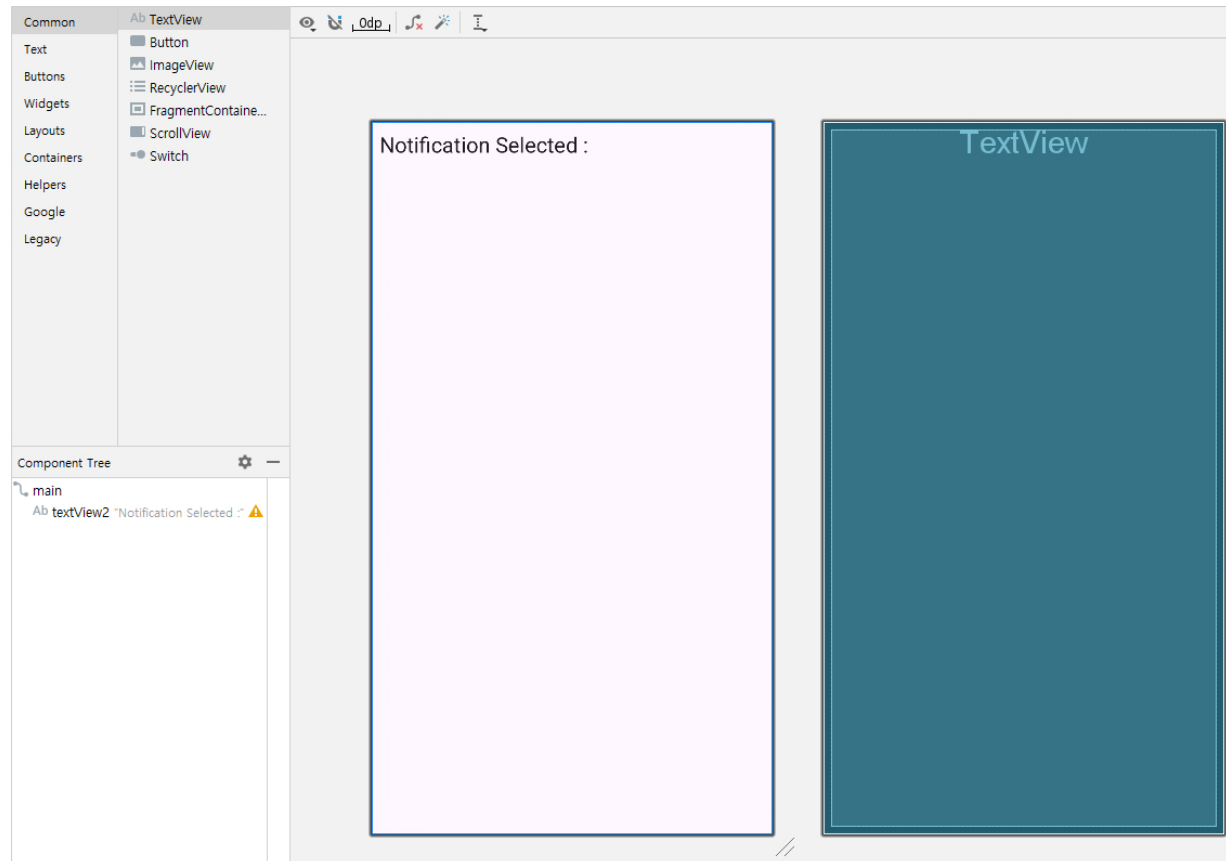
```
    <Button
        android:id="@+id/button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Notifications"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView" />

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

□ 레이아웃 구성 - *activity_notifications.xml*

○ Notification을 선택하면 연결할 액티비티

- 액티비티 연결 방법만 소개, 데이터 전달 표시 등, 필요에 따라 화면 구성
- 여기서는 연결 방법만 소개하기 때문에 텍스트뷰 1개로 단순하게 구성



○ Notification을 선택하면 연결할 액티비티

□ 액티비티 연결 방법만 소개, 필요에 따라 화면 구성

```
<?xml version="1.0" encoding="utf-8"?>
<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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".NotificationsActivity">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="Notification Selected :"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

AndroidManifest.xml

```
<?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.POST_NOTIFICATIONS"/>
```

Notification 기능 활성화를 위한 권한
– API33에서 추가

```
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="Notifications"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Notifications"
    tools:targetApi="31">
    <activity
        android:name=".NotificationsActivity"
        android:exported="false" />
```

AndroidManifest.xml

```
}      <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>
|  </application>
|
|</manifest>
```

□ 코드 설명 / 분석 - MainActivity

```
package com.practice.ex.notifications;

import android.Manifest;
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
```

□ 코드 설명 / 분석 - MainActivity

```
private static final int NOTIFY_1 = 0x1001; 1 usage
private NotificationManager notifier = null; 3 usages
Notification notify = new Notification(); 3 usages
Button btn01; 2 usages
```

@Override

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);
    setContentView(R.layout.activity_main);
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {...});
```

```
    btn01 = (Button) findViewById(R.id.button01);
    btn01.setOnClickListener(this);
```

Notification 표시를 할 객체 생성

```
    notifier = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
```

```
    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
        ActivityCompat.requestPermissions( activity: this,
            new String[]{Manifest.permission.POST_NOTIFICATIONS}, MODE_PRIVATE);
    }
```

Notification 기능을 관련 권한 요청

□ 코드 설명 / 분석 - MainActivity

@Override

권한 허용 여부 확인 - 필요한 내용 추가 구현

```
public void onClick(View v) {
```

```
    if(ActivityCompat.checkSelfPermission( context: this,
        Manifest.permission.POST_NOTIFICATIONS) != PackageManager.PERMISSION_GRANTED) {
        // example :
        // ActivityCompat. ---
    }
```

```
String expandedTitle = "E-Title";
String expandedText = "Context";
```

확장 상태바 제목, 확장 상태바에 표시될 내용

확장 텍스트가 클릭된 경우, 액티비티를 동작 시키기 위한
인텐트 설정으로 알림을 드래그 했을 때 실행할 액티비티 설정

```
Intent toLaunch = new Intent( packageContext: MainActivity.this, NotificationsActivity.class);
```

```
PendingIntent intentBack = PendingIntent.getActivity( context: MainActivity.this,
    requestCode: 0, toLaunch,
    flags: PendingIntent.FLAG_IMMUTABLE|PendingIntent.FLAG_UPDATE_CURRENT);
```

```
if(Build.VERSION.SDK_INT < Build.VERSION_CODES.O) {
```

현재 버전 확인

```
    notify = new Notification.Builder(getApplication())
        .setContentTitle(expandedTitle)
        .setContentText(expandedText)
        .setSmallIcon(R.drawable.notifications_24px)
        .setAutoCancel(true)
        .setContentIntent(intentBack)
        .setWhen(System.currentTimeMillis())
        .build();
```

- Notification 생성
- 확장 상태바 제목
- 확장 상태바 표시 내용
- 아이콘
- 확인시 사라지게 함
- 선택시 동작할 인텐트 지정
- 설정한 값을 기반으로 생성

- (5.0 이후 이미지 단색)

□ 코드 설명 / 분석 - MainActivity

- 현재 버전 확인 : Notification의 경우, 8.0 이상에서는 동작 코드에 채널 정보를 추가해야 함

```
if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
```

```
    NotificationChannel channelMessage = new NotificationChannel( id: "channel_id",  
        name: "channel_name", NotificationManager.IMPORTANCE_DEFAULT);  
    channelMessage.setDescription("channel description");  
    channelMessage.enableLights(true);  
    channelMessage.setLightColor(Color.GREEN);  
    channelMessage.enableVibration(true);  
    channelMessage.setVibrationPattern(new long[]{100, 200, 100, 200});  
    channelMessage.setLockscreenVisibility(Notification.VISIBILITY_PRIVATE);  
    notifier.createNotificationChannel(channelMessage);
```

채널 정보 생성
- 중요도 설정 등을
진행할 수 있음

```
    notify = new Notification.Builder(getApplication(), channelId: "channel_id")  
        .setContentTitle(expandedTitle)  
        .setContentText(expandedText)  
        .setSmallIcon(R.drawable.notifications_circle_24px)  
        .setAutoCancel(true)  
        .setContentIntent(intentBack)  
        .setWhen(System.currentTimeMillis())  
        .build();
```

채널정보를 포함하여
Notification 생성

NotificationManager를 통한 Notification 실행

```
    notifier.notify(NOTIFY_1, notify);
```

Notification 클래스는 재생하는 오디오 타입, 소리, 진동 LED의 표시 색, 재생하는 사운드의 URI, 등의 다양한 필드가 있어 다양한 기능을 갖는 알림을 만들 수 있음

* 안드로이드 버전에 따라 구현에 차이가 있음

□ 코드 설명 / 분석 - NotificationsActivity

```
package com.practice.ex.notifications;

import android.os.Bundle;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class NotificationsActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_notifications);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {...});
    }
}
```

○여기서는 액티비티 연결된 상태만 확인하기 때문에 추가적인 코드를 작성하지 않음

□ 개별 공부해야 할 사항

❖ PendingIntent

- PendingIntent는 Activity/Service 등과 같은 특정 Component가 Intent를 생성한 후에 해당 Intent를 바로 사용하는 대신에 다른 Component가 해당 Intent를 사용 할 수 있도록 할 때 사용하는 클래스