

모바일프로그래밍기초

- 안드로이드(Android) -

○안드로이드 기능/실습 4

- 액티비티 - 데이터 전달

○안드로이드 기능/실습 5

- 액티비티 - 데이터 전달(결과 리턴)

□ 안드로이드 기능/실습 4 액티비티 - 데이터전달

□ 안드로이드 기능 / 실습 - 4

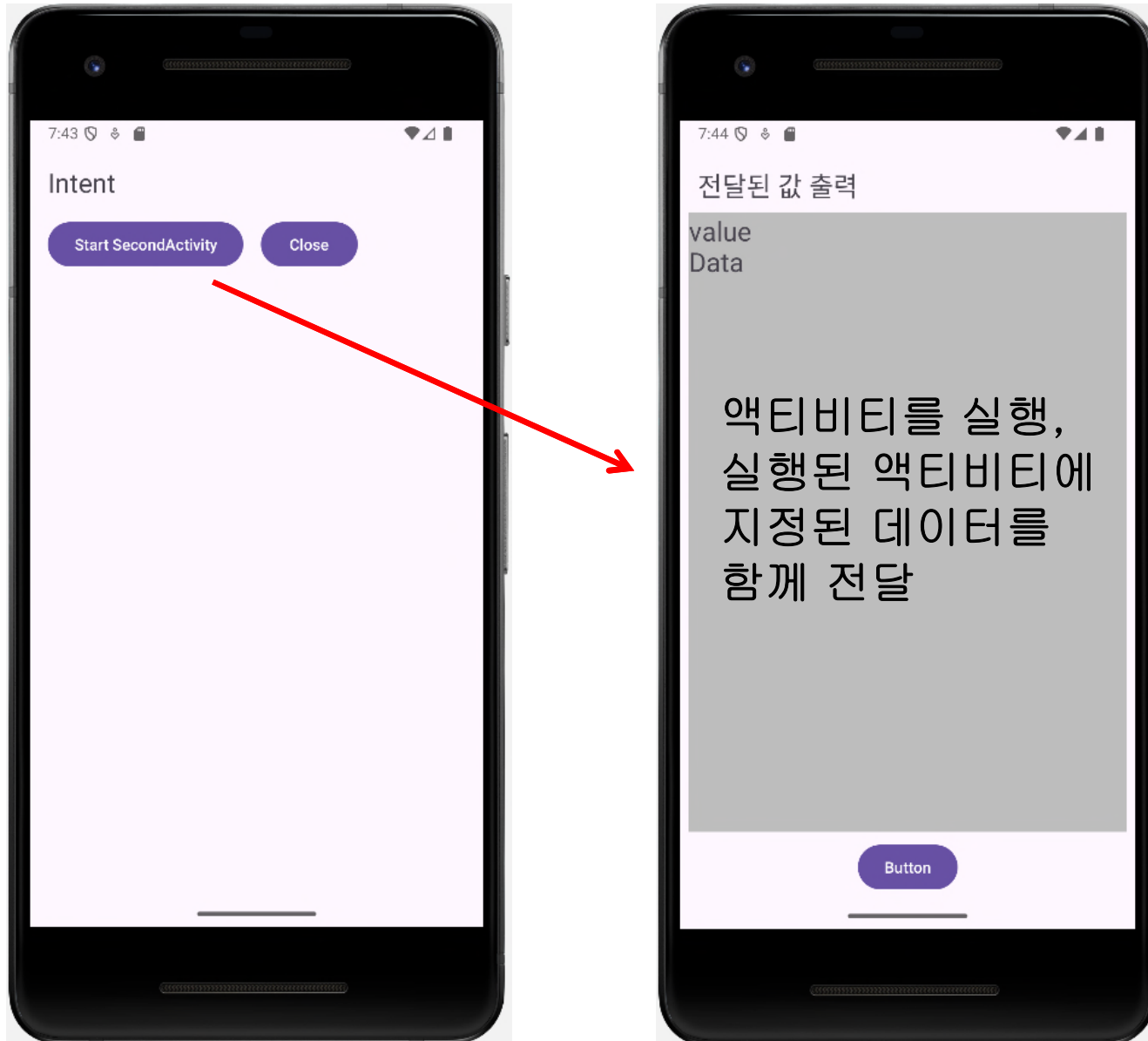
○ 목적

- 다른 액티비티로 데이터를 전달하는 과정 습득
 - ❖ Intent 객체 활용
 - ❖ 데이터 전달 과정 습득

○ 기능/실습 내용

- Intent를 이용하여 데이터 전달 과정 확인
- 데이터 전달, 데이터 구분 과정 확인

□ 실행 모습(*startActivity* - *putExtra*)



□ 해야할 일의 순서는? / 준비해야 할 내용

○액티비티 준비

- MainActivity.java
- SecondActivity.java

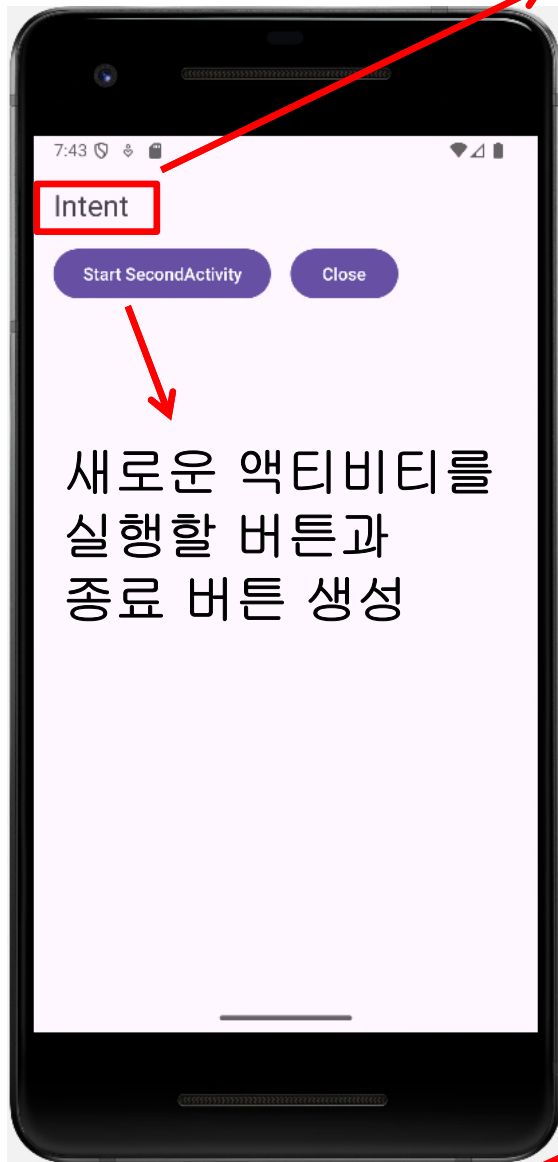
○레이아웃 준비

- activity_main.xml
- activity_second.xml

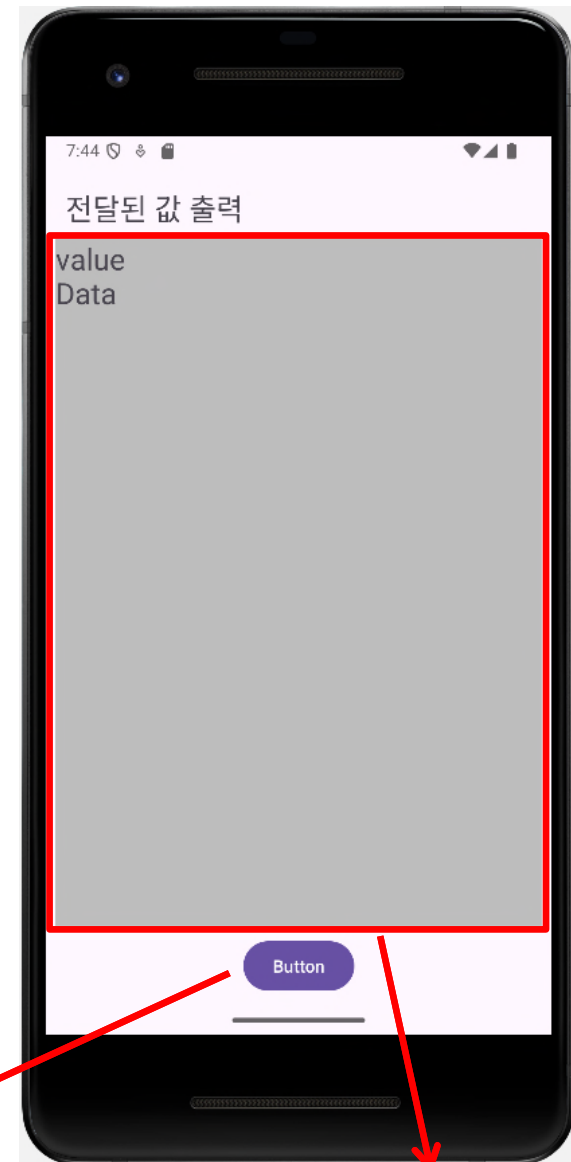
○파일명은 개인별 원하는 이름으로 생성

□ 레이아웃 구성

텍스트 사이즈 확대(텍스트 표시 없어도 됨)



종료 버튼

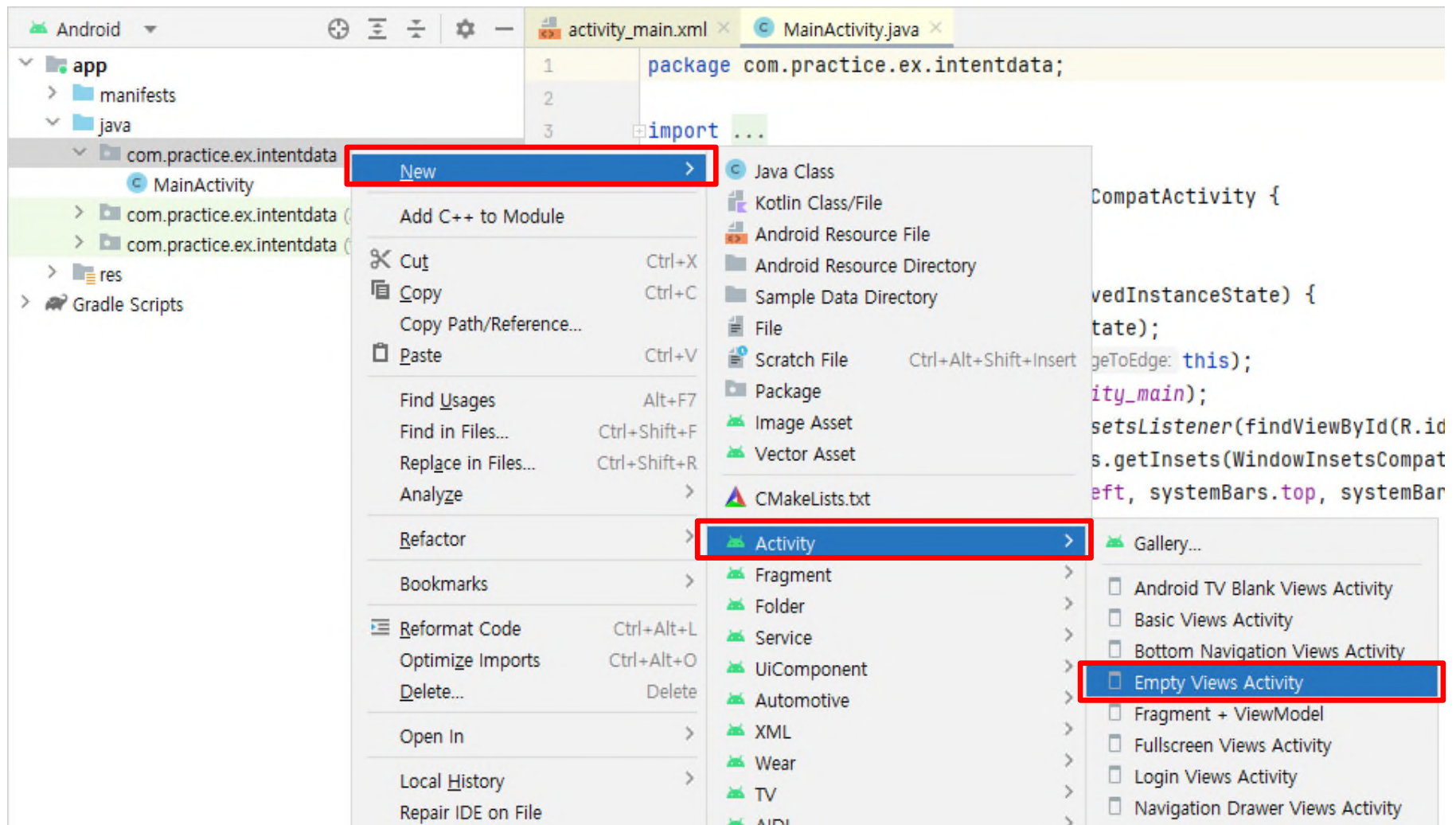


전달된 데이터 출력창 (TextView)
색상 변경 (RGB 예, #FFBDBDBD or #BDBDBD)

□ 액티비티/레이아웃 준비

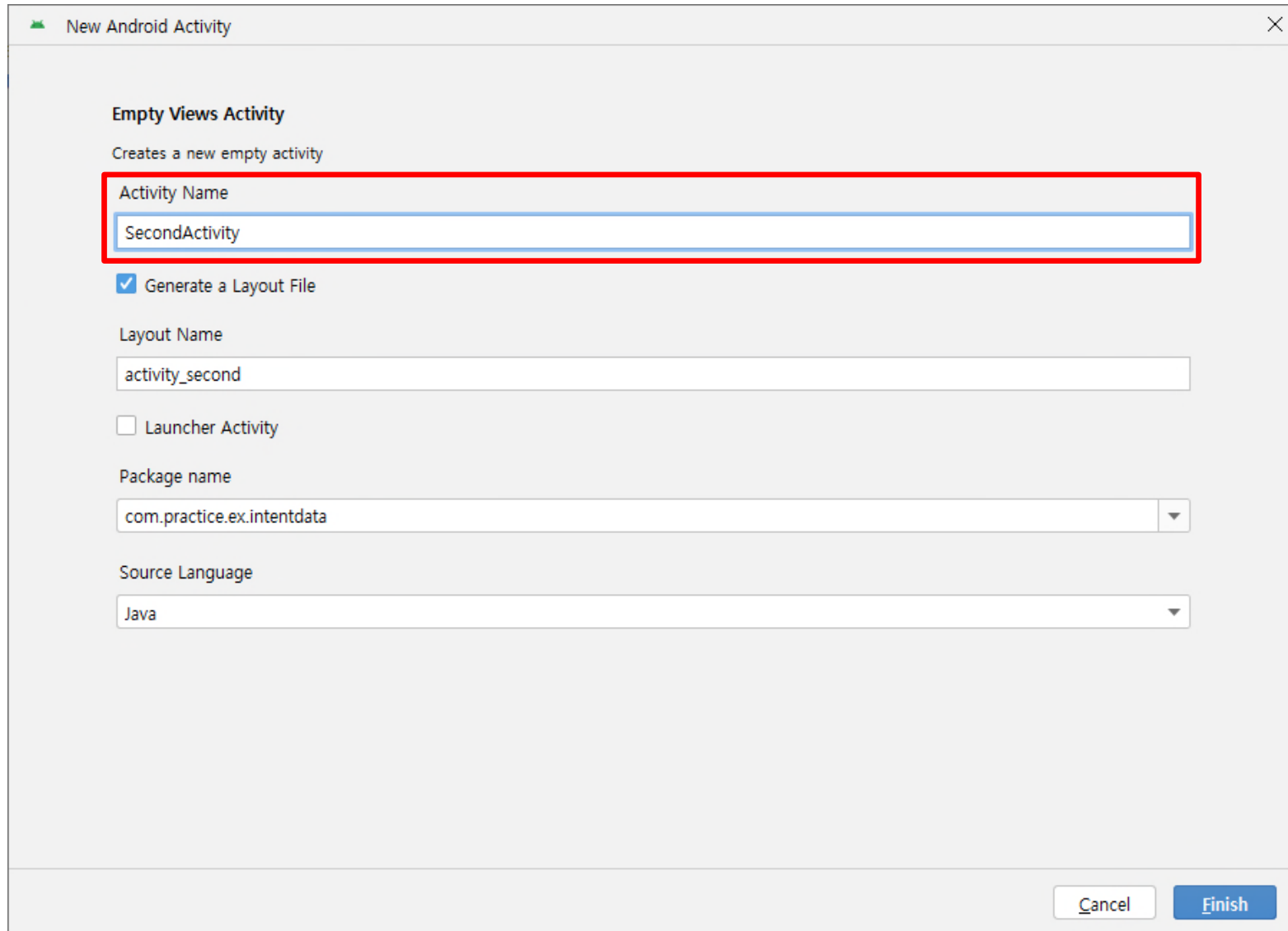
○필요한 액티비티 및 레이아웃 추가

□ New → Activity → Empty Views Activity 선택



□ 액티비티/레이아웃 준비

○ Activity Name 입력(수정)



New Android Activity

Empty Views Activity

Creates a new empty activity

Activity Name

SecondActivity

☒ Generate a Layout File

Layout Name

activity_second

☐ Launcher Activity

Package name

com.practice.ex.intentdata

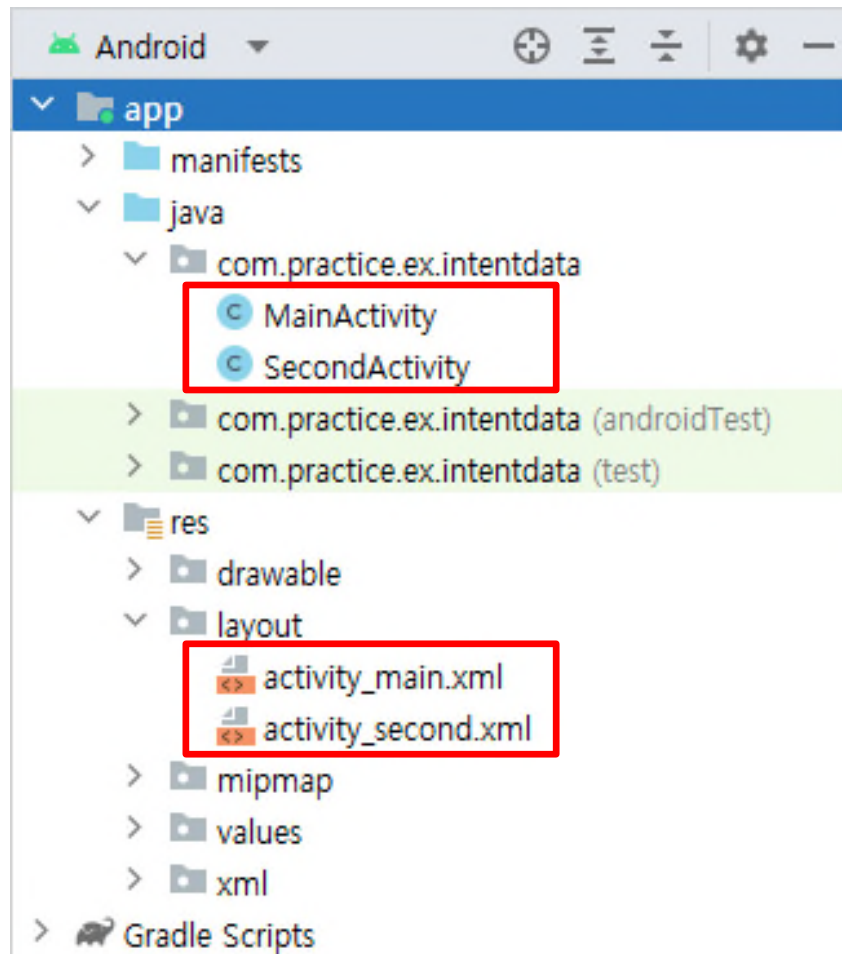
Source Language

Java

Cancel Finish

□ 액티비티/레이아웃 준비

○ 생성 결과



□ *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">

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

        <activity
            android:name=".SecondActivity"
            android:exported="false" />

        <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>
```

추가된 액티비티 정보 확인

- Label 등을 추가 정보를 자유롭게 추가

□ 레이아웃 구성 : *activity_main.xml*

- TextView 1개와 Button 2개로 구성
 - 속성 확인
- 상단 TextView는 단순 표기용
 - 글씨 크기는 24sp
- 버튼 2개의 역할
 - 액티비티 실행
 - 종료
- ConstraintLayout으로 구성



□ 레이아웃 구성 : *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/textView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:text="Intent"
        android:textSize="24sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

글씨 크기 속성
android:textSize=" " ”
– 단위 필수 입력
– 글씨 크기 단위는 sp 사용

□ 레이아웃 구성 : *activity_main.xml*

```
<Button
    android:id="@+id/button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="16dp"
    android:text="Start SecondActivity"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView01" />

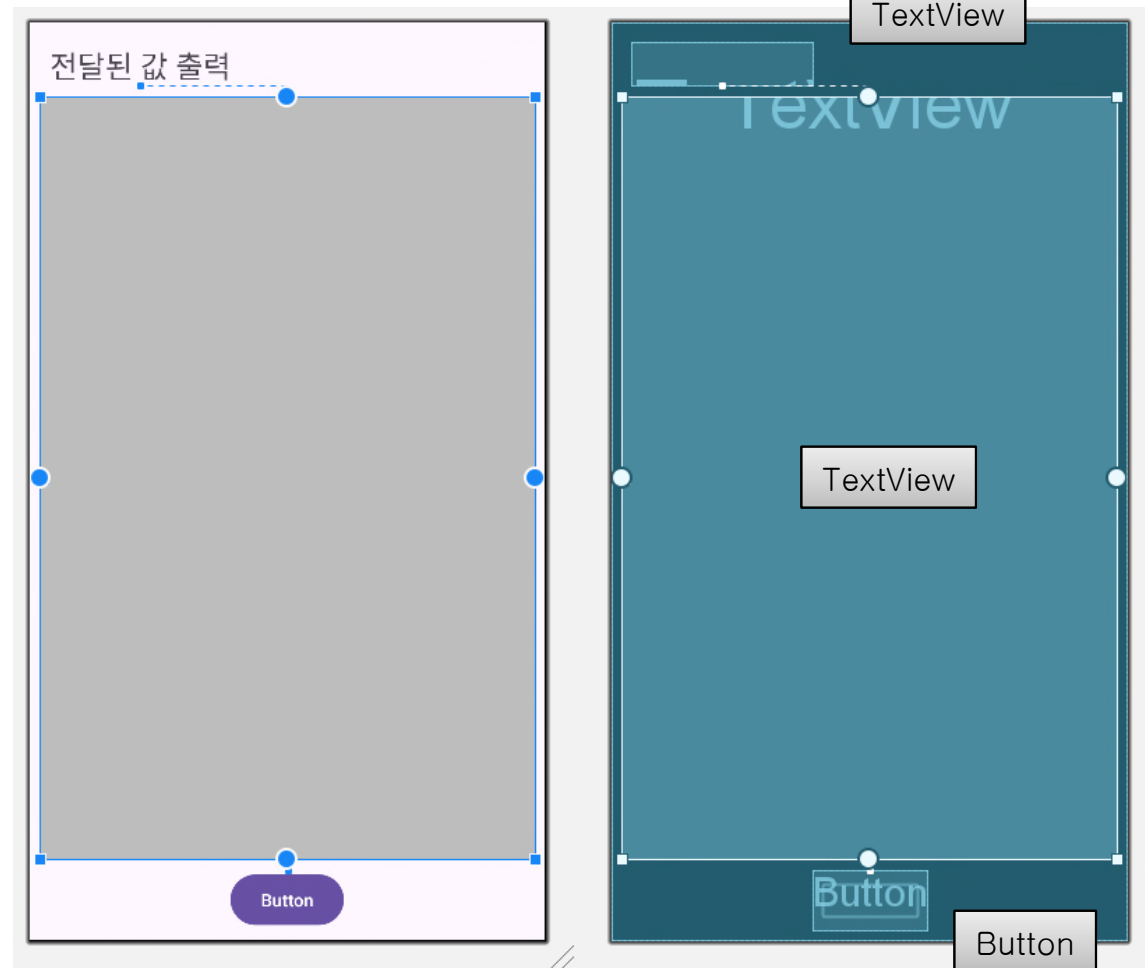
<Button
    android:id="@+id/button02"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:text="Close"
    app:layout_constraintStart_toEndOf="@+id/button01"
    app:layout_constraintTop_toTopOf="@+id/button01" />

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

□ 레이아웃 구성 : *activity_second.xml*

○ 2개의 TextView와 1개의 Button으로 구성

- 첫 번째 TextView는
단순 표기용
- 두 번째 TextView는
받은 데이터 표시용
 - ❖ 색상 변경
 - ❖ 크기 조절



- 버튼은 액티비티 종료를 위해 존재

□ 레이아웃 구성 : *activity_second.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=".SecondActivity">

    <TextView
        android:id="@+id/textView02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:text="전달된 값 출력"
        android:textSize="24sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```


□ 레이아웃 구성 : *activity_second.xml*

```
<Button
    android:id="@+id/button03"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:text="Button"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent" />

<TextView
    android:id="@+id/result_textView"
    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:background="#FFBDBDBD"
    android:textSize="24sp"
    app:layout_constraintBottom_toTopOf="@+id/button03"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView02" />

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

배경색 지정
android:background=" " "

□ 코드 분석 / 설명 - MainActivity

```
package com.practice.ex.intentdata;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

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 implements View.OnClickListener {

    Button btn01, btn02; 2 usages

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

□ 코드 분석 / 설명 - MainActivity

```
btn01 = (Button) findViewById(R.id.button01);  
btn02 = (Button) findViewById(R.id.button02);  
  
btn01.setOnClickListener(this);  
btn02.setOnClickListener(this);
```

버튼에 대한 클릭 이벤트 처리를 위해서
해당 버튼 레이아웃과 연결 / 리스너에 등록

@Override

```
public void onClick(View view) {  
    if(view.getId() == R.id.button01) {  
        Intent intent01 = new Intent( packageContext: MainActivity.this, SecondActivity.class);  
        intent01.putExtra( name: "key01", value: "value\nData");  
        startActivity(intent01);  
    }  
    if(view.getId() == R.id.button02) {  
        finish();  
    }  
}
```

Intent 객체에 putExtra() 메소드를 이용하여
전달할 데이터 기록

- putExtra() - 특정 데이터를 추가하여 전달할 때 이용
(이름(키), 값)으로 구성

□ 코드 분석 / 설명 - *SecondActivity*

```
package com.practice.ex.intentdata;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

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 SecondActivity extends AppCompatActivity implements View.OnClickListener {

    Button btn01; 2 usages

    TextView result_text; 2 usages

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

□ 코드 분석 / 설명 - *SecondActivity*

```
Intent data_receive;  
data_receive = getIntent();  
String temp01 = data_receive.getStringExtra( name: "key01");
```

넘어온 데이터를 네임(키) 값으로 찾아 확인

```
result_text = (TextView) findViewById(R.id.result_textView);
```

```
result_text.setText(temp01);
```

받은 값을 하단의 TextView에 출력

```
btn01 = (Button) findViewById(R.id.button03);  
btn01.setOnClickListener(this);
```

```
}  
  
@Override  
public void onClick(View view) {  
    finish();  
}  
}
```

□ 안드로이드 기능 / 실습 - 5

액티비티 - 데이터 전달(결과 리턴)

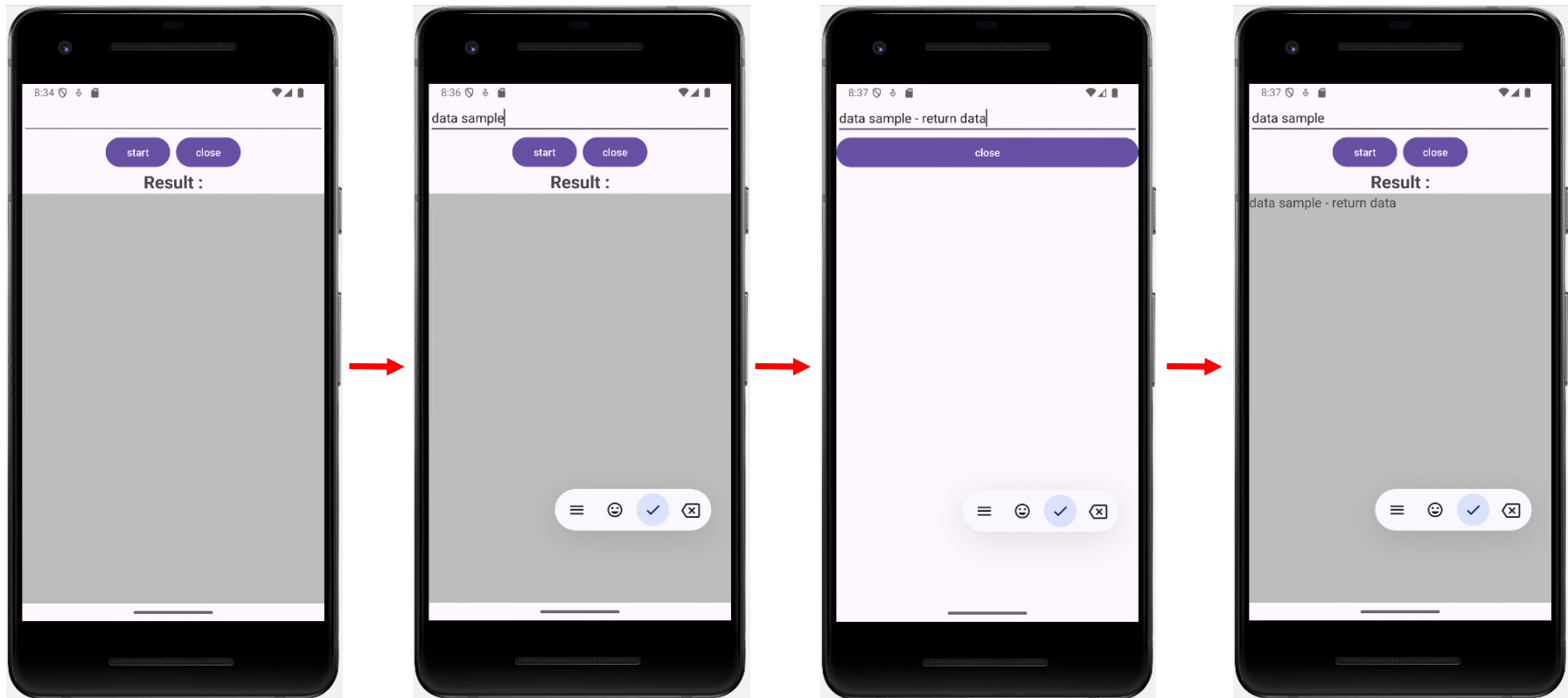
○ 목적

- 액티비티에서 실행한 또 다른 액티비티로부터 결과를 되돌려 받아 처리하는 과정 습득
 - ❖ 액티비티가 새로운 액티비티를 실행하면서 데이터를 전달한 후, 실행된 액티비티로부터 다시 데이터(결과)를 전달 받아야 하는 경우

○ 기능/실습 내용

- 처리 결과를 돌려받기 위한 데이터 전달 과정 확인
- 돌아온 결과 처리 과정 습득
- 레이아웃 부분 - LinearLayout으로 구성
 - ❖ LinearLayout에 대한 기초 사용법 습득

□ 실행 모습



실행된 액티비티가 종료되면, 해당 종료한 액티비티에서
가지고 있는 값을 해당 액티비티를 실행한 액티비티로 되돌려 주는
과정을 통해서 데이터 리턴 처리 과정을 확인

□ 실행 확인



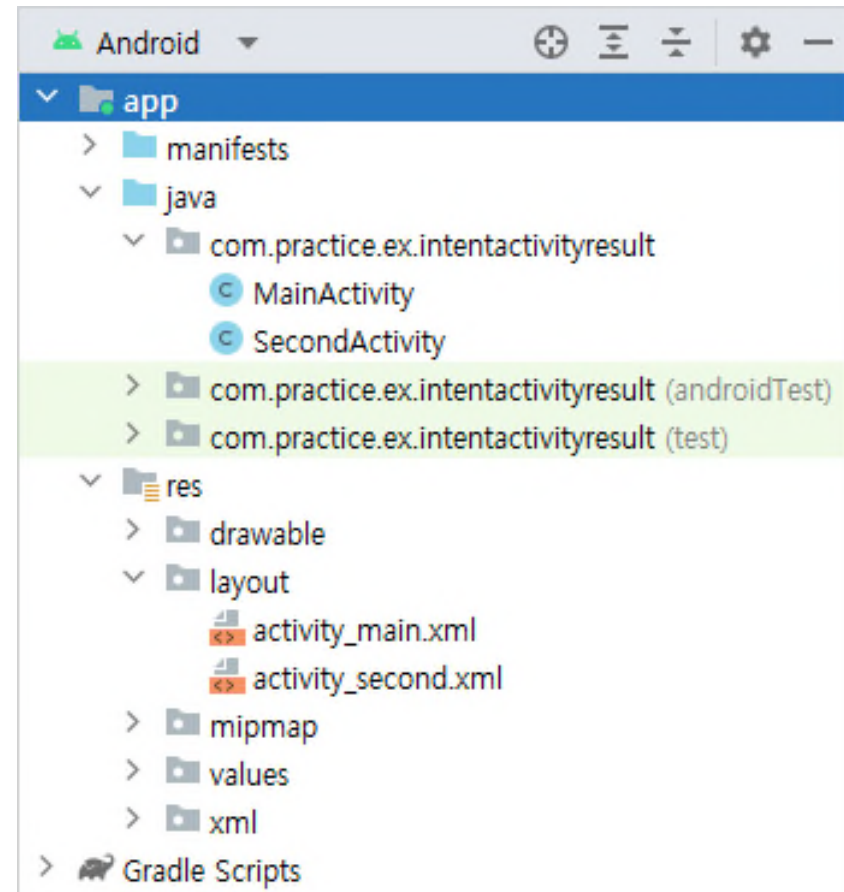
□ 준비해야 할 내용

○ 액티비티 준비

- MainActivity.java
- SecondActivity.java

○ 레이아웃 준비

- activity_main.xml
- activity_second.xml



□ *AndroidManifest.xml*

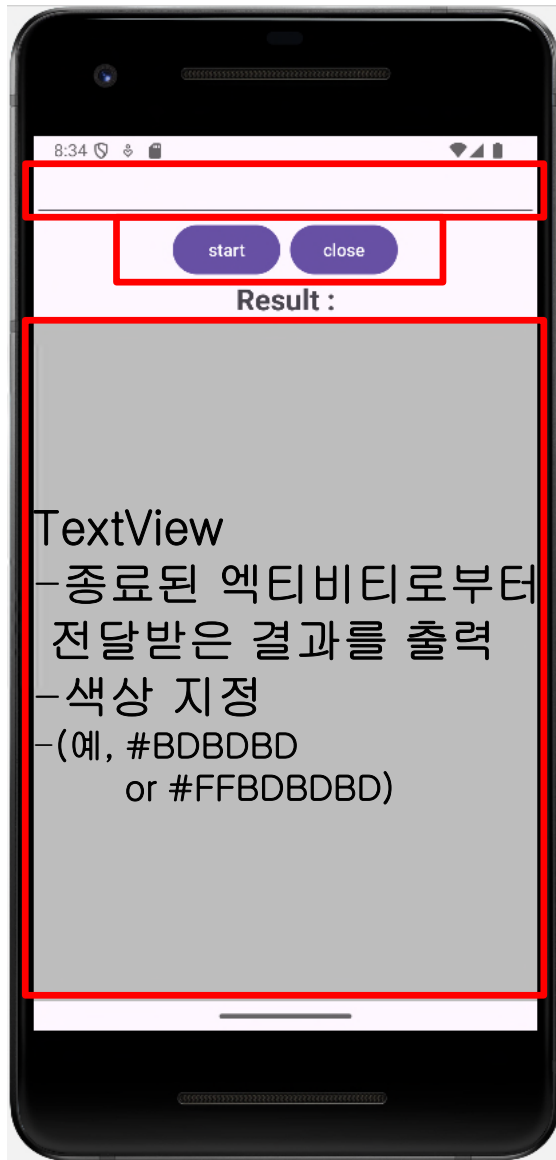
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

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

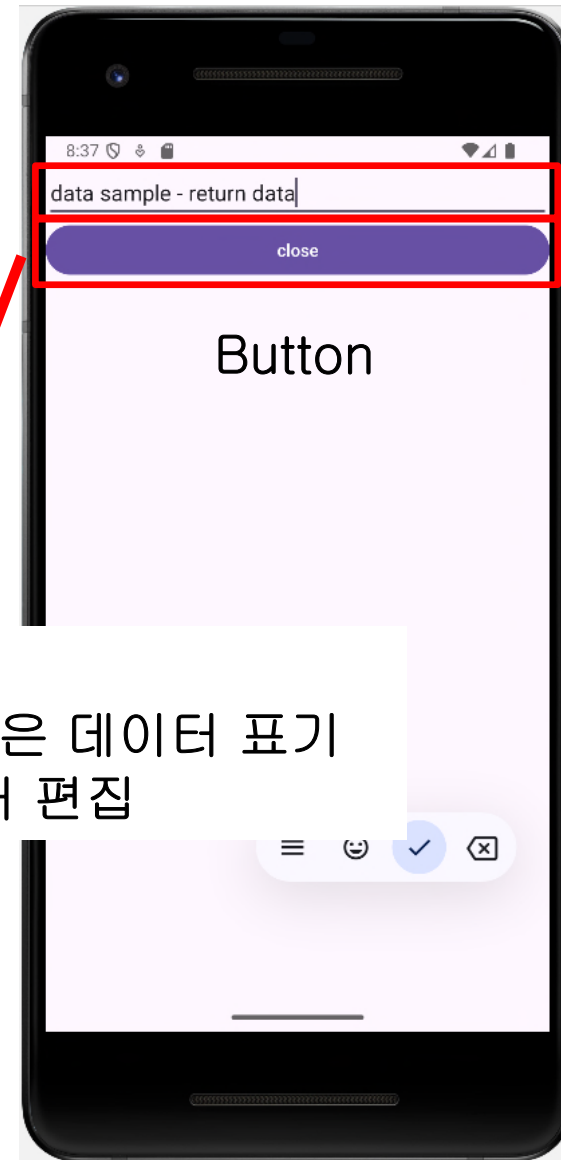
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

□ 레이아웃 구성



EditText
Button



EditText
- 전달받은 데이터 표기
및 데이터 편집

□ 레이아웃 구성 : *activity_main.xml*

○LinearLayout

- 순서대로 한방향으로 나열해주는 역할
- android:orientation
 - ❖ “vertical”
 - ❖ “horizontal”

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout
```

```
    xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    xmlns:tools="http://schemas.android.com/tools"
```

```
    android:id="@+id/main"
```

```
    android:layout_width="match_parent"
```

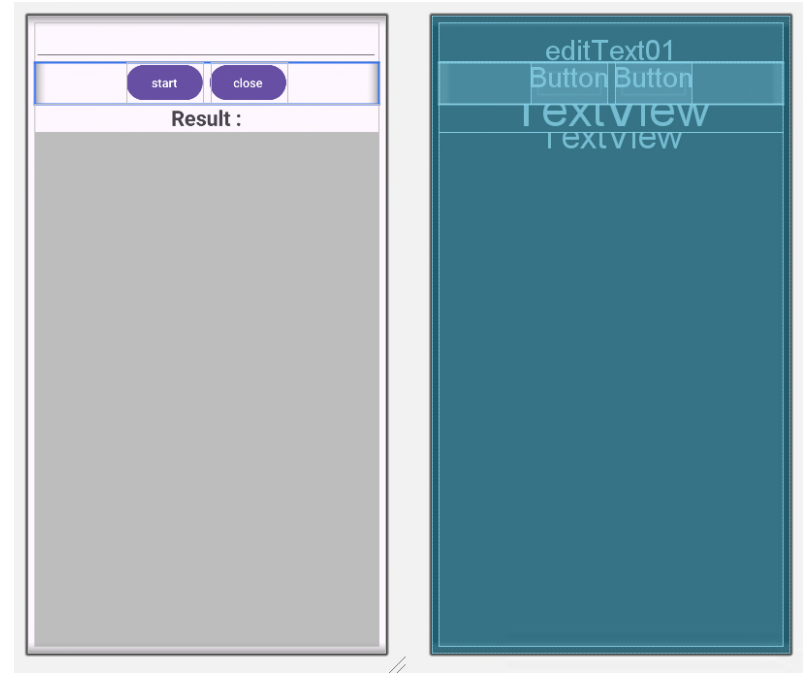
```
    android:layout_height="match_parent"
```

```
    android:orientation="vertical"
```

```
    android:weightSum="1"
```

```
    android:padding="8dp"
```

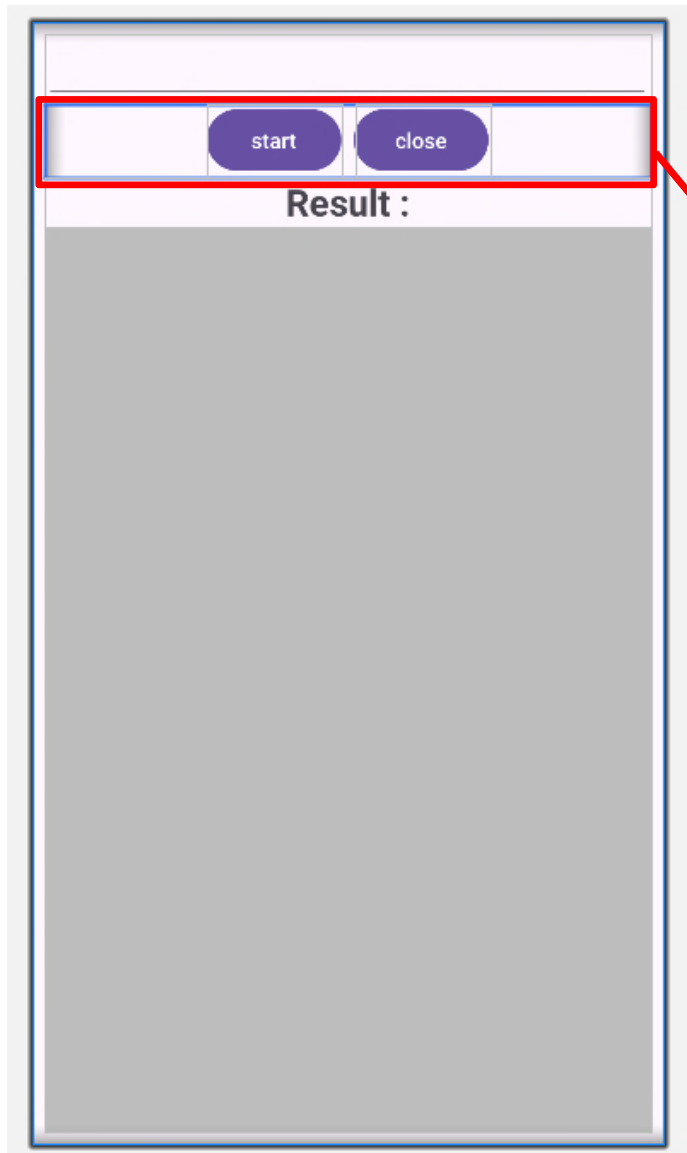
```
    tools:context=".MainActivity">
```



← orientation : 배치 방향 결정

weightSum 속성은 여기서 사용하지 않음
LinearLayout 설명을 위해 기술한 것임

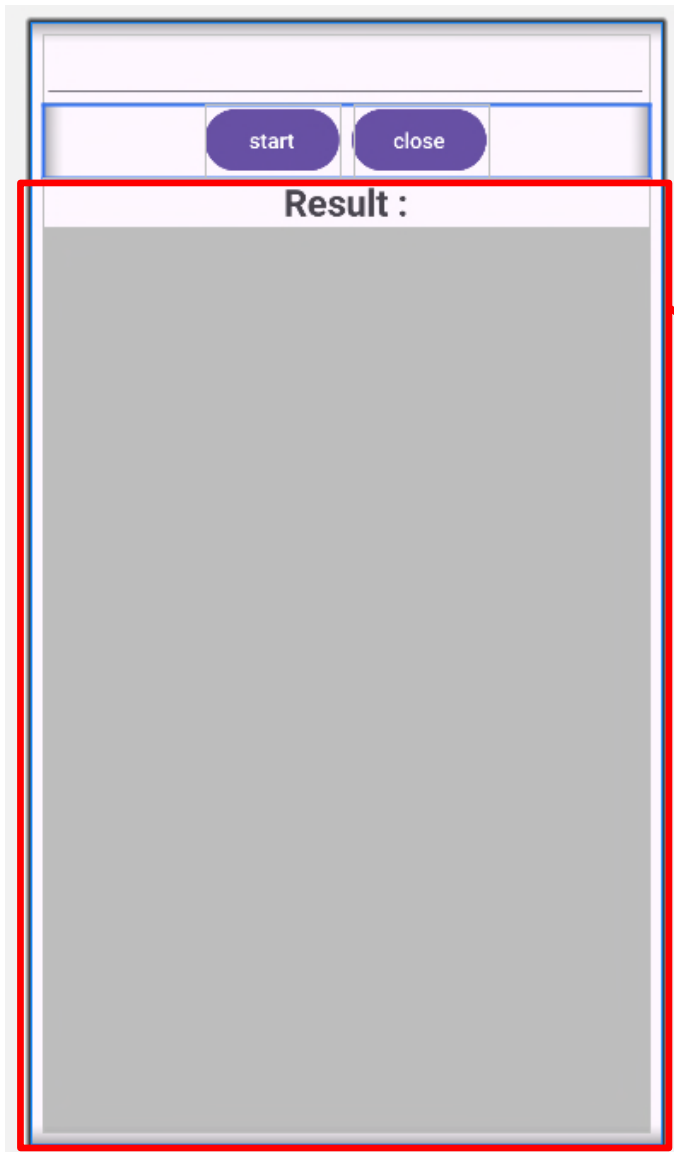
□ 레이아웃 구성 : *activity_main.xml*



```
1 <EditText
2     android:id="@+id/editText01"
3     android:inputType="text"
4     android:layout_width="match_parent"
5     android:layout_height="wrap_content" />
6
7 <LinearLayout
8     android:layout_width="match_parent"
9     android:layout_height="wrap_content"
10    android:gravity="center"
11    android:orientation="horizontal">
12
13    <Button
14        android:id="@+id/button01"
15        android:layout_width="wrap_content"
16        android:layout_height="wrap_content"
17        android:layout_marginEnd="8dp"
18        android:text="start" />
19
20    <Button
21        android:id="@+id/button02"
22        android:layout_width="wrap_content"
23        android:layout_height="wrap_content"
24        android:text="close" />
25 </LinearLayout>
```

배치 방향의 변경이 필요한 경우

□ 레이아웃 구성 : *activity_main.xml*



```
<TextView
    android:id="@+id/textView01"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Result :"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="bold" />

<TextView
    android:id="@+id/textView02"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFBDBDBD"
    android:textSize="18sp" />

</LinearLayout>
```

LinearLayout 구성을 위해 알아야 하는 기본 속성
android:orientation = ...
android:weightSum = ...
android:layout_weight = ...
...

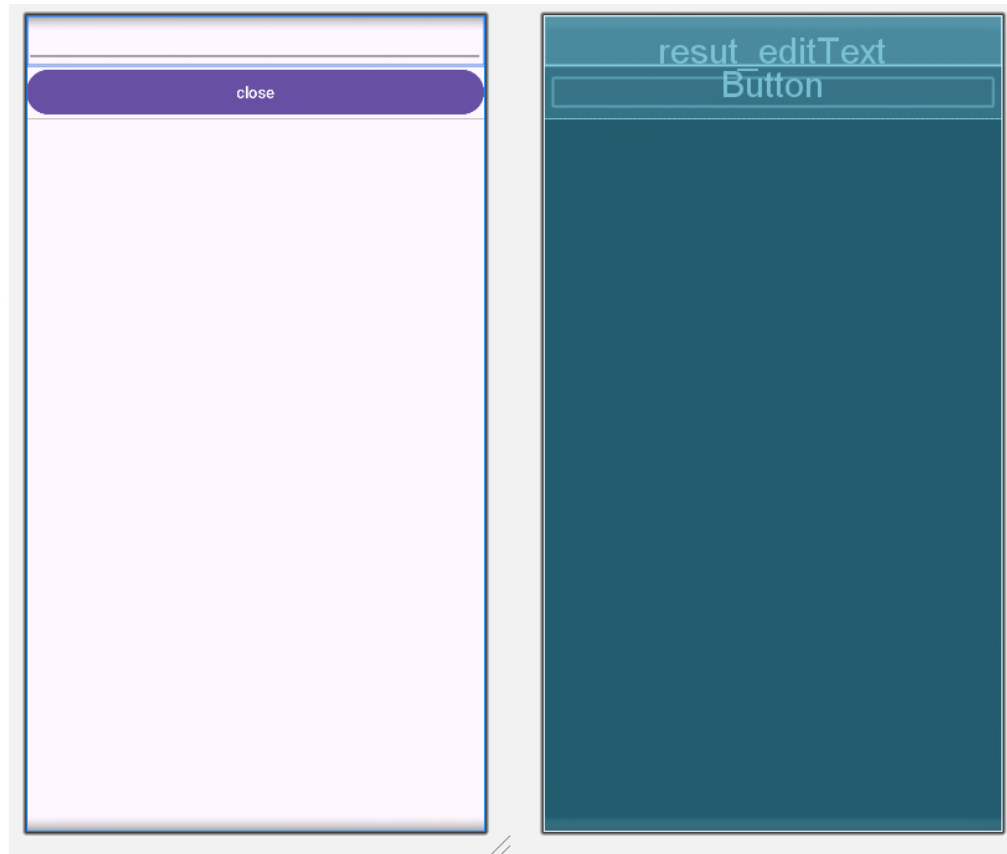
□ 레이아웃 구성 : *activity_second.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".SecondActivity">

    <EditText
        android:id="@+id/resut_editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="text" />

    <Button
        android:id="@+id/button03"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="close" />

</LinearLayout>
```



□ 코드 분석 / 설명 - MainActivity

```
package com.practice.ex.intentactivityresult;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.activity.EdgeToEdge;
import androidx.activity.result.ActivityResult;
import androidx.activity.result.ActivityResultCallback;
import androidx.activity.result.ActivityResultLauncher;
import androidx.activity.result.contract.ActivityResultContracts;
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 implements View.OnClickListener {

    private ActivityResultLauncher<Intent> resultLauncher; 2 usages

    EditText input_edit; 2 usages
    Button btn01, btn02; 2 usages
    TextView result_text; 3 usages
```

데이터 리턴을 처리하기 위한
ActivityResultLauncher 선언

□ 코드 분석 / 설명 - MainActivity

```
@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) -> {...});

    input_edit = (EditText)findViewById(R.id.editText01);

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

    btn02 = (Button)findViewById(R.id.button02);
    btn02.setOnClickListener(this);

    result_text = (TextView) findViewById(R.id.textView02);
}
```

레이아웃 연결 및
버튼 클릭 이벤트 처리를 위한 리스너 등록

□ 코드 분석 / 설명 - MainActivity

```
resultLauncher = registerForActivityResult(  
    new ActivityResultContracts.StartActivityForResult(),  
    new ActivityResultCallback<ActivityResult>() {  
        @Override  
        public void onActivityResult(ActivityResult o) {  
            if(o.getResultCode() == RESULT_OK) {  
                Intent data_intent = o.getData();  
                String data_result = data_intent.getExtras().getString( key: "ResultData");  
                result_text.setText(data_result);  
            }  
            if(o.getResultCode() != RESULT_OK) {  
                result_text.setText("No Data");  
            }  
        }  
    }  
);
```

onCreate 메소드 내에 콜백함수를 작성(등록)
- registerForActivityResult()를 통해 콜백을 등록

□ 코드 분석 / 설명 - MainActivity

```
@Override
public void onClick(View view) {
    if(view.getId() == R.id.button01) {
        String s = input_edit.getText().toString();
        Intent intent01 = new Intent( packageContext: MainActivity.this, SecondActivity.class);
        intent01.putExtra( name: "SendData", s);
        resultLauncher.launch(intent01);
    }
    if(view.getId() == R.id.button02) {
        finish();
    }
}
```



ActivityResultLauncher 객체의 launcher 메소드를
이용하여 액티비티를 실행
- 실행 액티비티 종료시 결과값을 받음

□ 코드 분석 / 설명 - *SecondActivity*

```
package com.practice.ex.intentactivityresult;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

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 SecondActivity extends AppCompatActivity implements View.OnClickListener {

    EditText editText01; 3 usages
    Button btn01; 2 usages
    String receive, result; 3 usages

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


□ 코드 분석 / 설명 - *SecondActivity*

```
receive = getIntent().getStringExtra( name: "SendData");  
if(receive != null) {  
    editText01 = (EditText)findViewById(R.id.resut_editText);  
    editText01.setText(receive);  
}
```

```
btn01 = (Button)findViewById(R.id.button03);  
btn01.setOnClickListener(this);
```

버튼처리를 위한 리스너 등록

Intent에서 넘어온 데이터를 확인
데이터를 EditText에 출력

@Override

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

```
    result = editText01.getText().toString();  
    if(result.length() != 0) {  
        Intent intent01 = new Intent();  
        intent01.putExtra( name: "ResultData", result);  
        setResult(RESULT_OK, intent01);  
    } else {  
        setResult(RESULT_CANCELED);  
    }  
}
```

```
    finish();  
}
```

setResult() - 리턴할 결과 설정
- 수행 결과(성공, 실패 등)와 데이터 결과