

Android application 만들기

## 8. Activity와 Intent

Activity의 life cycle

단순 전환

단방향 데이터 전달, 양방향 데이터 전달

Intent 보내기

# Android App Components

안드로이드 앱의 기본 구성 요소.

시스템이나 사용자가 앱에 들어올 수 있는 진입점.

## Activity

사용자와 상호작용을 하는 진입점. GUI를 제공한다.

각 Activity는 독립되어 있으며 앱이 허용할 경우 외부에서도 Activity를 사용할 수 있다.

## Service

백그라운드에서 실행된다. GUI를 제공하지 않는다.

앱이 화면에 출력되지 않더라도 동작한다.

## Broadcast Receiver

시스템 또는 다른 앱이 전송하는 Broadcast를 받을 수 있으며 Broadcast를 통해 앱이 실행 될 수 있다.

GUI는 없지만 상태 표시줄에 알림을 표시할 수 있다.

## Content Provider

파일, SQLite 등의 데이터를 제공한다. 다른 앱이 사용할 수 있도록 적절한 API를 제공한다.

# Activity Lifecycle

## onCreate()

Activity를 생성할 때 호출됨. 필수. 이전에 저장한 데이터가 있다면 savedInstanceState라는 번들로 전달된다. 저장하고 싶은 데이터는 onSaveInstanceState를 override해서 저장한다.

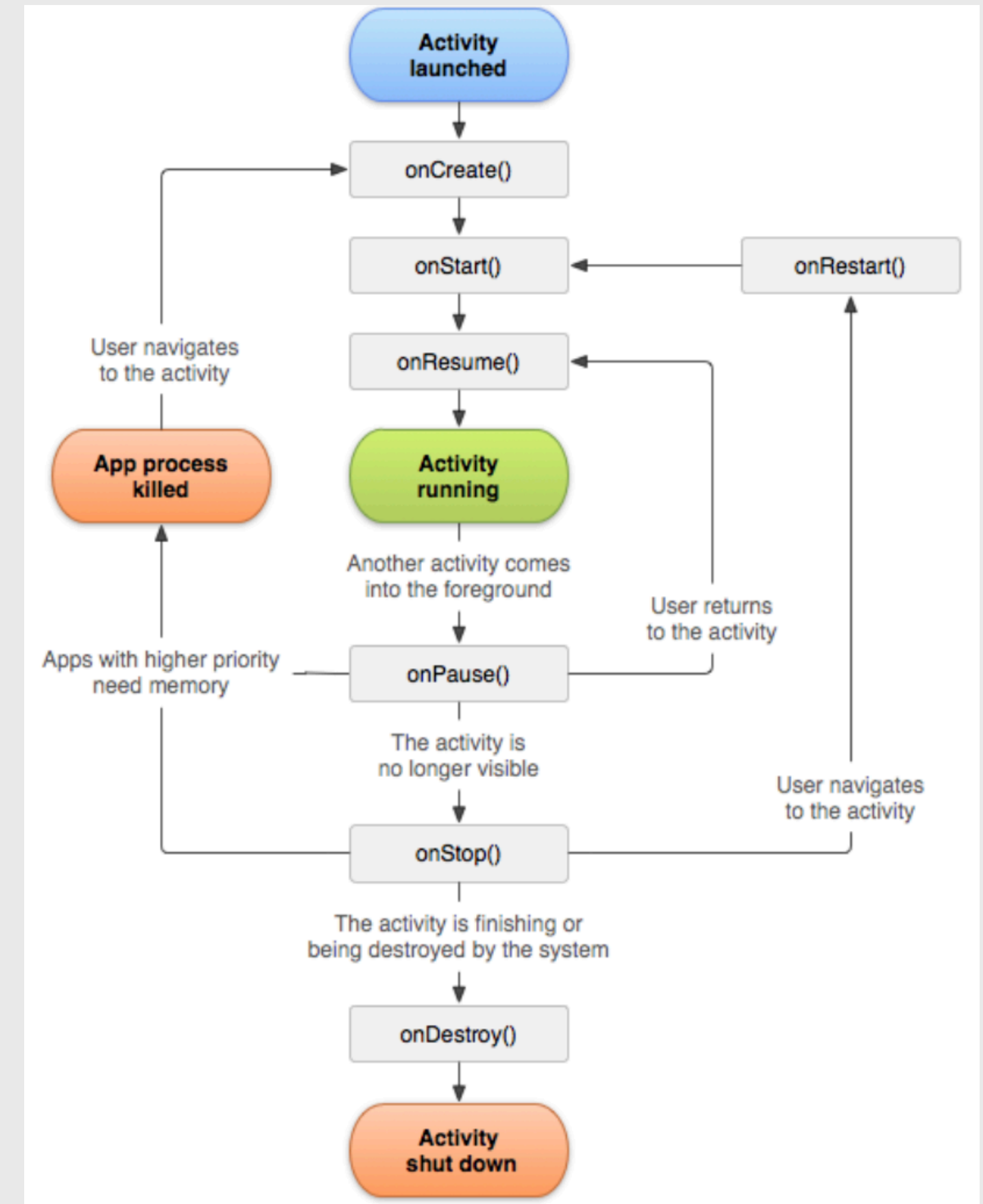
## onStart()

Activity가 사용자에게 보이는 시점.

## onResume()

Activity가 사용자와 상호작용을 할 수 있는 시점.

onPause -> onStop -> onDestroy는 반대로 동작



# Intent

android app components 중 Activity, Service, Broadcast Receiver를 부를 수 있는 메시지.

안드로이드의 컴포넌트는 직접 new MainActivity()와 같은 형태로 만들어 쓸 수 없으며 intent를 사용해 불러 쓸 수 있다.

불편한 점

- 아주 간단한 화면 전환도 intent 객체를 만들어야 한다.

편한 점

- 요청을 받은 Activity가 내 앱이 아니라도 동일한 사용법이다.
- Service, Broadcast Receiver 등을 동일한 사용법으로 불러 쓸 수 있다.

명시적 Intent: 대상 클래스를 지정하여 요청함. 즉 내 앱 내의 컴포넌트를 사용할 때 사용

묵시적 Intent: 내가 원하는 동작(Action)을 지정하여 요청함. 다른 앱이 공개한 컴포넌트를 불러 쓸 수 있다.

# 명시적 Intent

내 앱 내에 있는 컴포넌트를 지정하여 실행하고자 함.

Intent를 받아 처리해 주는 것은 시스템이므로 시스템에게 내 앱에는 이러이러한 컴포넌트가 있다는 사실을 알려야 함. 어디에?

## AndroidManifests.xml

```
manifest
```

```
  application
```

```
    activity
```

```
    service
```

```
    broadcast receiver
```

# Activity에게 필요한 것들

Java 코드

레이아웃 파일

AndroidManifests.xml의 항목

새 Activity를 만드는 방법

1. 위 3가지 항목을 각각 만들기
2. Android studio에서 제공하는 위저드를 이용해 만들기

# 실습: Widget 프로젝트에 Activity 추가하기

MainActivity.java가 포함된 패키지에서 마우스 우 클릭

New > Activity(메뉴 아래쪽) > Empty Activity

Activity Name: ConfirmActivity

FINISH 클릭 후

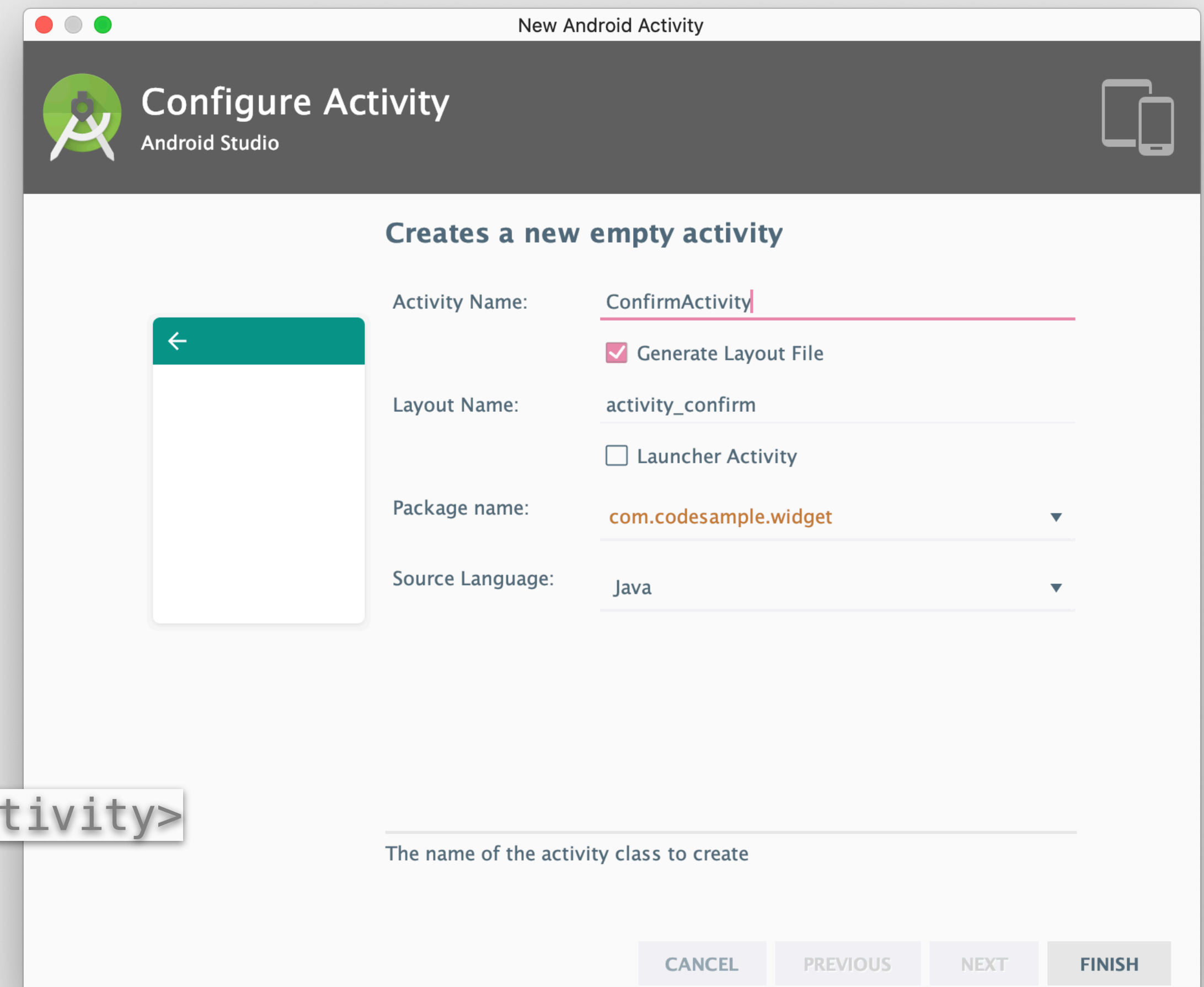
ConfirmActivity.java

res/layout/activity\_confirm.xml 추가 확인

AndroidManifests.xml에

태그가 추가된 것 확인

```
<activity android:name=".ConfirmActivity"></activity>
```



# 실습: Widget 프로젝트에 Activity 추가하기

ConfirmActivity의 onCreate 함수 수정

```
public class ConfirmActivity extends AppCompatActivity {  
    private ActivityConfirmBinding binding;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        binding=ActivityConfirmBinding.inflate(getLayoutInflater());  
        setContentView(binding.getRoot());  
    }  
}
```



# 실습: Activity 전환하기

MainActivity의 initWidget 함수에 다음 라인 추가하고 onClick 함수를 수정 후 버튼 동작 확인

```
binding.buttonApply.setOnClickListener(this);  
}
```

initWidget 함수 수정

```
@Override  
public void onClick(View v) {  
    if(v.getId()==R.id.buttonApply){  
        Intent intent = new Intent(this, ConfirmActivity.class);  
        startActivity(intent);  
    } else {  
        updateProgress();  
    }  
}
```

onClick 함수 수정

# Activity에 데이터 전달하기

Intent를 이용해 데이터를 전달할 수 있다.

MainActivity에서 ConfirmActivity에 데이터를 전달하려면 다음과 같이 Intent에 추가 데이터를 넣으면 된다.

`intent.putExtra(String name, Object data);`

MainActivity.java

```
@Override
public void onClick(View v) {
    if(v.getId()==R.id.buttonApply){
        Intent intent = new Intent(this, ConfirmActivity.class);
        intent.putExtra("name", binding.editTextName.getText().toString());
        intent.putExtra("phone", binding.editTextPhone.getText().toString());
        startActivity(intent);
    } else {
        updateProgress();
    }
}
```

# 전달 받은 데이터 꺼내기

데이터를 넣을 때 사용한 String parameter를 이용해 꺼낼 수 있다.

ConfirmActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding=ActivityConfirmBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());

    String name=getIntent().getStringExtra("name");
}
```

intent에 데이터를 넣을 때는 putExtra 함수로 통일 되지만 꺼낼 때는 데이터 타입에 따라 적절한 get 함수를 쓴다.

String과 같은 클래스 타입의 경우 null이 반환될 수 있으므로 이를 주의한다.

int, float와 같이 클래스가 아닌 타입의 경우 두 번째 파라미터로 default value를 넘겨야 한다.

# 실습: 전달 받은 데이터로 UI 업데이트하기

MainActivity에서 넘겨준 이름, 전화번호, 구분 세 가지 정보를 ConfirmActivity의 UI에 다음과 같이 출력하는 코드를 작성하라. 아래 스크린에서 TextView 자리에 입력 받은 값을 출력.

이름: TextView	@+id/textViewName
전화번호: TextView	@+id/textViewPhone
구분: TextView	@+id/textViewClass

# 실습: 전달 받은 데이터로 UI 업데이트하기

onClick을 수정해서 필요한 데이터를 전달하도록 함

MainActivity.java

```
@Override
public void onClick(View v) {
    if(v.getId()==R.id.buttonApply){
        Intent intent = new Intent(this, ConfirmActivity.class);
        intent.putExtra("name", binding.editTextName.getText().toString());
        intent.putExtra("phone", binding.editTextPhone.getText().toString());
        String userClass="Adult";
        if(binding.radioButtonStudent.isChecked()) userClass="Student";
        intent.putExtra("class", userClass);
        startActivity(intent);
    } else {
        updateProgress();
    }
}
```

# 실습: 전달 받은 데이터로 UI 업데이트하기

activity\_confirm.xml 을 수정해서 전달 받은 데이터를 출력할 TextView들을 추가함.

```
<TextView
    android:id="@+id/textViewNameLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:text="이름:"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.2"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<TextView
    android:id="@+id/textViewPhoneLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="전화번호:"
    app:layout_constraintStart_toStartOf="@+id/textViewNameLabel"
    app:layout_constraintTop_toBottomOf="@+id/textViewNameLabel" />
<TextView
    android:id="@+id/textViewClassLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="구분:"
    app:layout_constraintStart_toStartOf="@+id/textViewPhoneLabel"
    app:layout_constraintTop_toBottomOf="@+id/textViewPhoneLabel" />
```

```
<TextView
    android:id="@+id/textViewName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:text="TextView"
    app:layout_constraintBottom_toBottomOf="@+id/textViewNameLabel"
    app:layout_constraintStart_toEndOf="@+id/textViewNameLabel"
    app:layout_constraintTop_toTopOf="@+id/textViewNameLabel" />
<TextView
    android:id="@+id/textViewPhone"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:text="TextView"
    app:layout_constraintBottom_toBottomOf="@+id/textViewPhoneLabel"
    app:layout_constraintStart_toEndOf="@+id/textViewPhoneLabel"
    app:layout_constraintTop_toTopOf="@+id/textViewPhoneLabel" />
<TextView
    android:id="@+id/textViewClass"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:text="TextView"
    app:layout_constraintBottom_toBottomOf="@+id/textViewClassLabel"
    app:layout_constraintStart_toEndOf="@+id/textViewClassLabel"
    app:layout_constraintTop_toTopOf="@+id/textViewClassLabel" />
```

# 실습: 전달 받은 데이터로 UI 업데이트하기

ConfirmActivity에서 출력

ConfirmActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding=ActivityConfirmBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());

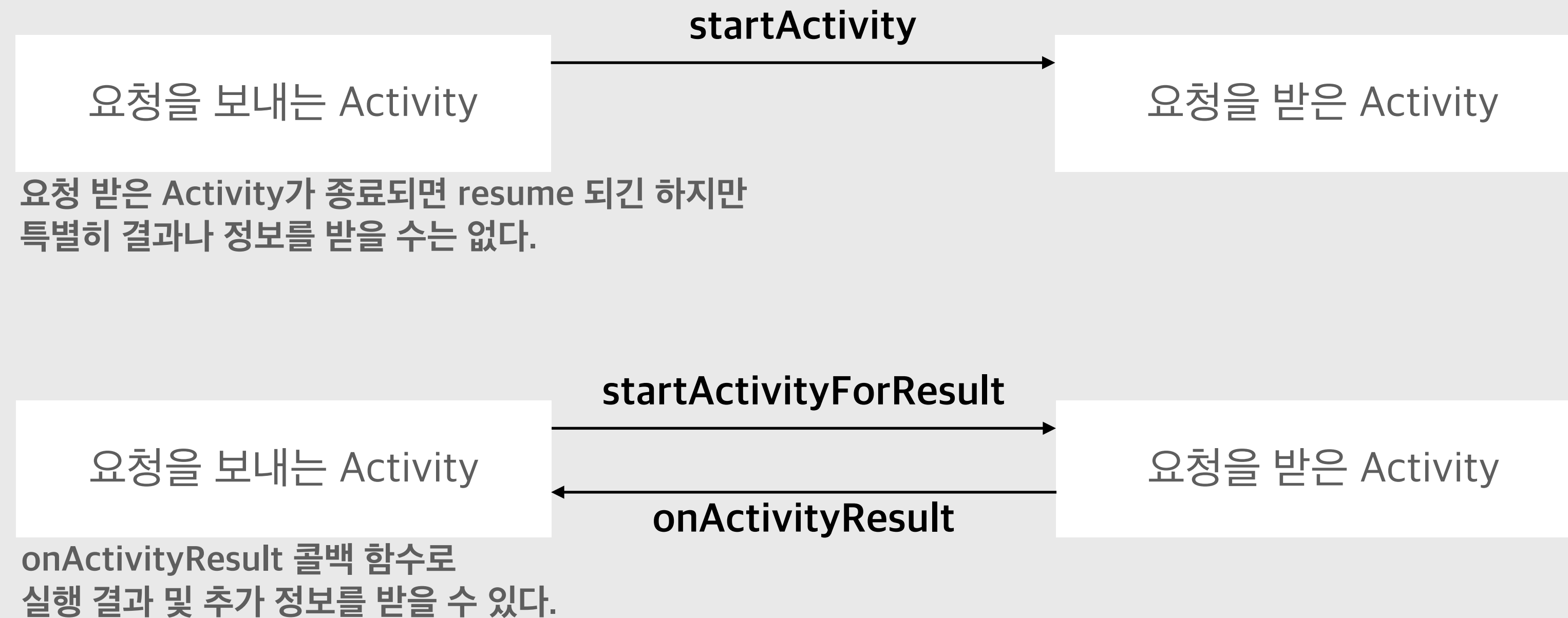
    Intent intent=getIntent();
    String name=intent.getStringExtra("name");
    String phone=intent.getStringExtra("phone");
    String userClass=intent.getStringExtra("class");

    if(name!=null) binding.textViewName.setText(name);
    if(phone!=null) binding.textViewPhone.setText(phone);
    if(userClass!=null) binding.textViewClass.setText(userClass);
}
```



# Activity의 결과 받기

요청한 Activity의 실행 결과를 받을 수 있다.





# 실습: ConfirmActivity의 결과 받기

ConfirmActivity에 “확인”, “취소” 버튼을 넣고 확인 일때는 OK 결과와 메시지를 데이터로 전달.

취소 버튼일때는 CANCEL 결과만 전달.

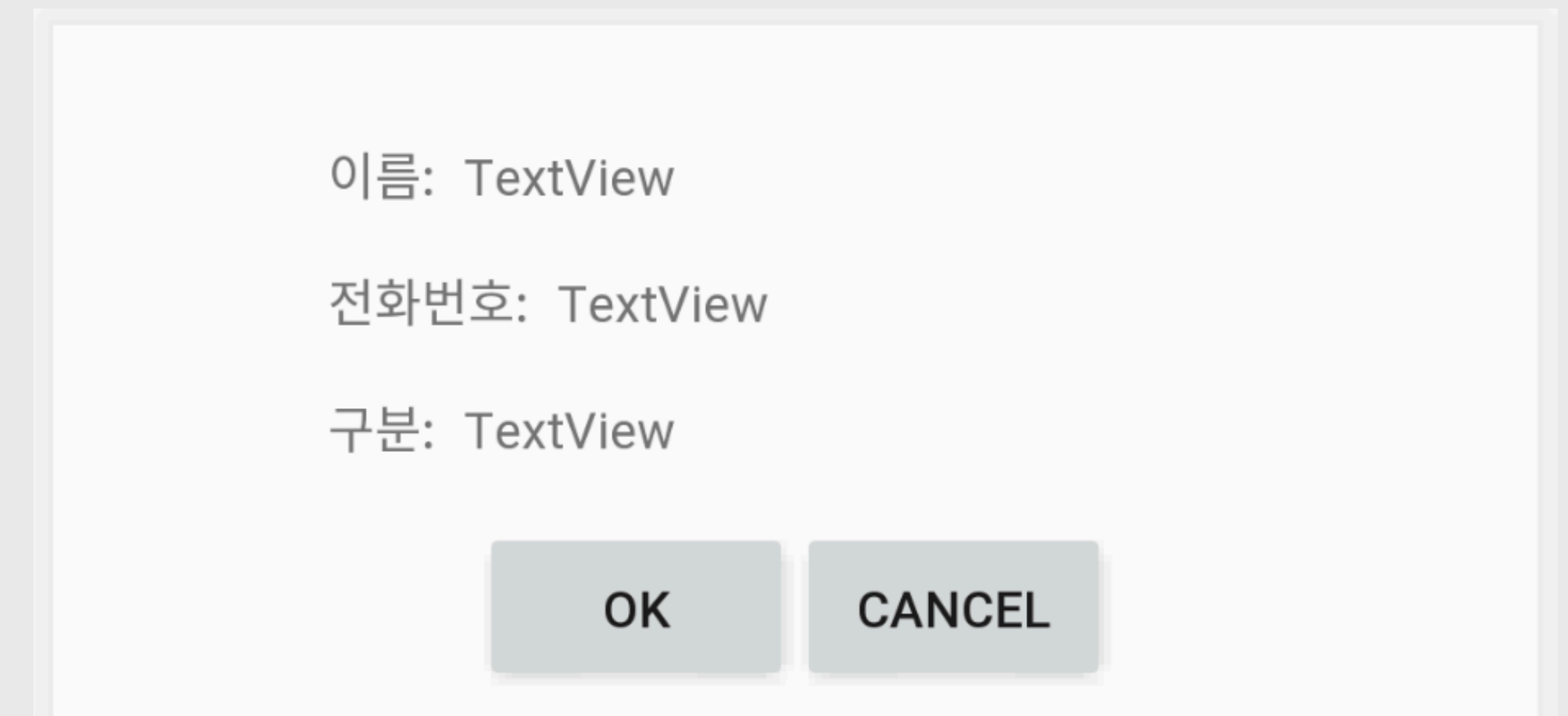
MainActivity에서는 결과를 받아 Toast를 출력.

Toast: 안드로이드의 화면 하단에 잠시 출력되는 메시지로 모든 앱이 사용할 수 있음.

일단 activity\_confirm에 버튼 2개 추가.

@+id/buttonOK

@+id/buttonCancel



이름: TextView

전화번호: TextView

구분: TextView

OK CANCEL

# 실습: ConfirmActivity의 결과 받기

위젯 2개 이상이 서로 서로 연결되었을 때를 chain이라 하며 이 때는 chain head(제일 왼쪽)가 스타일을 지정함.

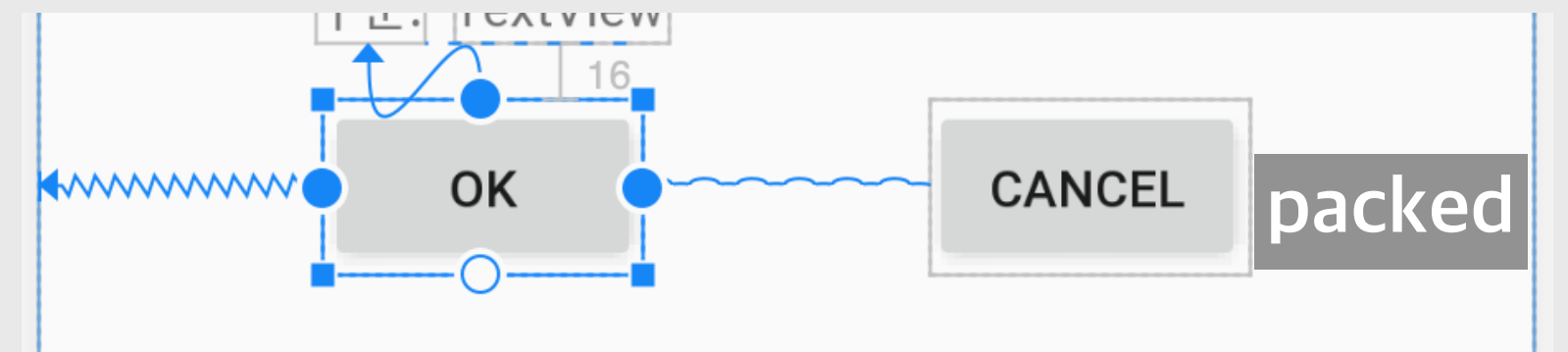
chain이 만들어진 경우 우측 화면 같이 버튼과 버튼 사이

선의 모양이 물결무늬로 출력 됨.

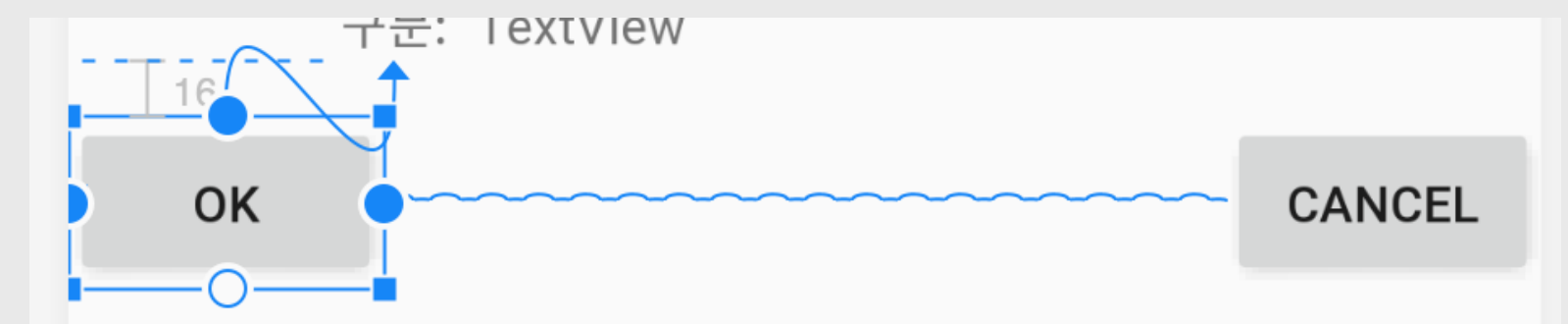
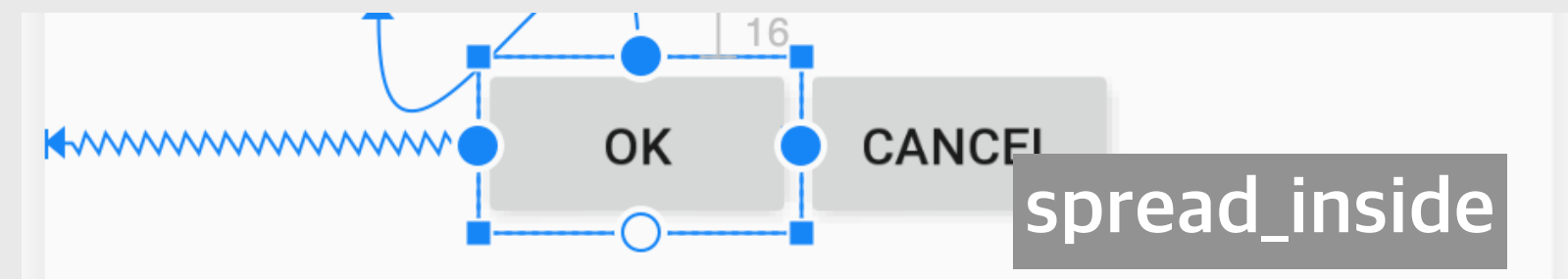
```
<Button
    android:id="@+id/buttonOK"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="OK"
    app:layout_constraintRight_toLeftOf="@+id/buttonCancel"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textViewClassLabel"
    app:layout_constraintHorizontal_chainStyle="packed"/>
```

```
<Button
    android:id="@+id/buttonCancel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Cancel"
    app:layout_constraintBottom_toBottomOf="@+id/buttonOK"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/buttonOK"
    app:layout_constraintTop_toTopOf="@+id/buttonOK" />
```

spread



spread\_inside



# 실습: ConfirmActivity의 결과 받기

button용 클릭 함수를 만들고

OK 버튼으로 종료하는 경우를 제외하고는 모두 실패 결과를 주기 위해 onCreate를 다음과 같이 수정

```
public class ConfirmActivity extends AppCompatActivity {  
    private ActivityConfirmBinding binding;  
  
    public void onButton(View v){  
        if(v.getId()==R.id.buttonOK){  
            Intent intent = new Intent();  
            intent.putExtra("message", "User confirmed");  
            setResult(Activity.RESULT_OK, intent);  
        }  
        finish();  
    }  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        binding=ActivityConfirmBinding.inflate(getLayoutInflater());  
        setContentView(binding.getRoot());  
    }  
}
```

함수 추가

OK 버튼일 때만 결과 수정하고 데이터 추가

finish() 액티비티가 스스로 종료할 때 부르는 함수

**setResult(Activity.RESULT\_CANCELED);**

OK가 아닐 경우 결과 코드만 반환하고 데이터는 주지 않음

# 실습: ConfirmActivity의 결과 받기

activity\_confirm.xml 의 두 버튼에 모두 android:onClick 속성 추가

```
<Button
    android:id="@+id/buttonOK"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="OK"
    android:onClick="onButton"
    app:layout_constraintEnd_toStartOf="@+id/buttonCancel"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textViewClassLabel"
    app:layout_constraintHorizontal_chainStyle="packed"/>
```

# 실습: ConfirmActivity의 결과 받기

결과를 받을 수 있도록 MainActivity 수정

1. startActivity 함수를 startActivityForResult로 수정
2. 결과를 받을 onActivityResult 함수를 override

MainActivity.java

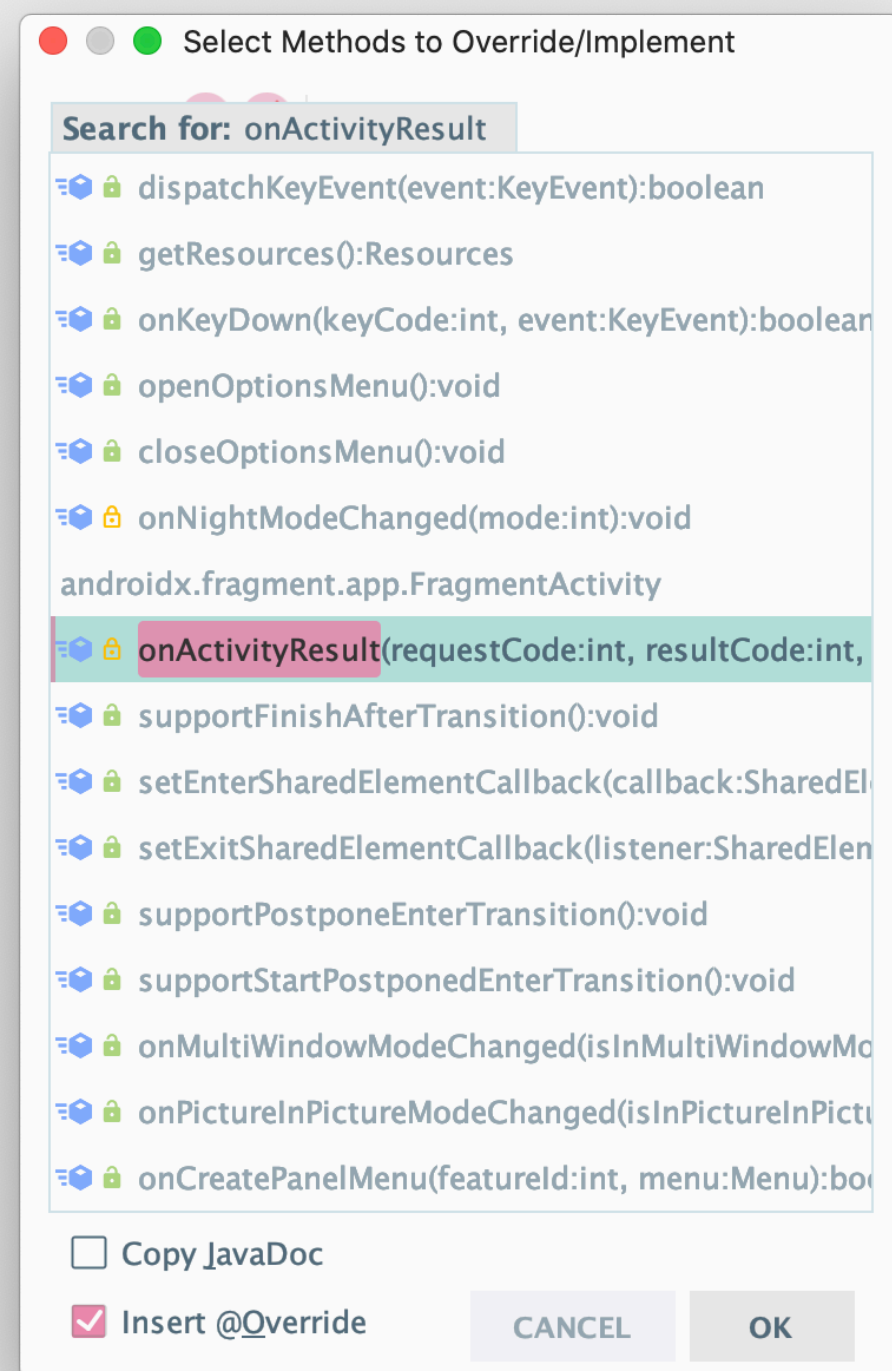
```
@Override
public void onClick(View v) {
    if(v.getId()==R.id.buttonApply){
        Intent intent = new Intent(this, ConfirmActivity.class);
        intent.putExtra("name", name.getText().toString());
        intent.putExtra("phone", phone.getText().toString());
        String userClass="Adult";
        if(student.isChecked()) userClass="Student";
        intent.putExtra("class", userClass);
        //startActivity(intent);
        startActivityForResult(intent, 1);
    } else {
        updateProgress();
    }
}
```

여러 Activity로 요청을 보낼 수 있기 때문에  
각 요청에는 번호를 붙여야 하고 이를  
requestCode 라고 한다.



# 실습: ConfirmActivity의 결과 받기

상단 메뉴 Code > Override methods 선택. 팝업에서 onActivityResult 를 타이핑하여 검색.



MainActivity.java

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode==1){
        if(resultCode== Activity.RESULT_OK ){
            if(data!=null) {
                String message = data.getStringExtra("message");
                if (message != null) Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(this, "Canceled.", Toast.LENGTH_SHORT).show();
        }
    }
}
```

앱을 실행하여 결과 확인.

# 묵시적 Intent

startActivity나 startActivityForResult는 내 앱 내의 Activity 외에 다른 앱의 Activity도 사용할 수 있음.

예를 들어 사진 앱에서 공유하기 버튼을 눌렀을 때 다른 앱이 뜨는 경우.

그러나 다른 앱이 어떤 Activity를 가졌는지는 알기 어려우므로

이 경우에는 Activity의 클래스가 아니라 내가 하고자 하는 동작을 적게 됨.

해당 동작을 수행할 수 있는 앱이 여러개이면 앱의 리스트가 뜨고

그 중 사용자가 선택한 앱으로 내 앱이 정의해 둔 Intent가 전달 됨.

# 묵시적 Intent 예: ACTION\_SEND\*

## Action

ACTION\_SENDTO: 첨부파일이 없는 이메일 또는 SMS

ACTION\_SEND: 첨부파일이 1개인 이메일 또는 SMS

ACTION\_MULTIPLE: 첨부파일이 여러 개인 이메일 또는 MMS

## MIME 유형

text/plain: 글자만 보내는 경우

\*/\*: 다양한 형태의 파일

Extra: 이메일인지 문자메시지인지에 따라 다르다.



# 묵시적 Intent 예: ACTION\_SEND\*

## 이메일 경우

Intent.EXTRA\_EMAIL: 수신자 이메일 주소 문자열의 배열

Intent.EXTRA\_CC: 참조자 이메일 주소 문자열의 배열

Intent.EXTRA\_BCC: 숨긴 참조자 이메일 주소 문자열이 배열

Intent.EXTRA\_SUBJECT: 메일 제목

Intent.EXTRA\_TEXT: 본문

Intent.EXTRA\_STREAM: 첨부파일

위와 같이 설정할 경우 SNS 앱 등이 뜰 수도 있는데 이를 막으려면 DataUri에 mailto:를 추가

# 묵시적 Intent 예: ACTION\_SEND\*

문자 경우

“subject” : MMS의 제목

“sms\_body”: 문자 본문

DataUri

“sms:010-1234-1234”

“smsto:010-1234-1234”

위와 같이 설정할 경우 SNS 앱 등이 뜰 수도 있는데 이를 막으려면 DataUri에 smsto:를 추가

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setData(Uri.parse("smsto:"));
intent.putExtra("sms_body", message);
intent.putExtra(Intent.EXTRA_STREAM, attachment);
if (intent.resolveActivity(getPackageManager()) != null) {
    startActivity(intent);
}
```

예제

# 실습: SMS

activity\_confirm.xml 에 SMS 버튼 추가

```
<Button
    android:id="@+id/buttonSMS"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="SMS"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/buttonOK" />
```

ConfirmActivity.java 에 코드 작성 (onCreate 함수 제일 아래)

```
binding.buttonSMS.setOnClickListener(v->{
    Intent smsIntent = new Intent(Intent.ACTION_SENDTO);
    smsIntent.setData(Uri.parse("smsto:010-4524-5468"));
    if (smsIntent.resolveActivity(getPackageManager()) != null) {
        startActivity(smsIntent);
    }
});
```