

안드로이드 SDK 개발 플랫폼 활용하기

# 액티비티 간 데이터 전달하기

- 인텐트를 이용한 액티비티간 데이터 전달 구현 예제 시나리오
  1. 메인액티비티의 XML 레이아웃에 에디트텍스트 추가
  2. 서브액티비티의 XML 레이아웃에 에디트텍스트 추가
  3. 메인액티비티의 버튼 클릭 시 입력 문자열을 서브액티비티에 전달
  4. 서브액티비티의 onCreate()에서 전달받은 문자열을 화면에 표시
  5. 서브액티비티의 버튼 클릭 시 답변 문자열을 메인에 돌려주도록 구현
  6. 메인액티비티에서 돌아온 문자열을 표시

- 인텐트
  - 액티비티 간 메시지 전달을 위한 중요한 클래스
- 먼저 두 개의 액티비티를 생성
  - 새로 만든 액티비티는 반드시 매니페스트 파일에 등록
  - Androidmanifest.xml을 열고 application에 다음의 코드 추가
    - new → Activity 메뉴로 생성시 매니페스트 파일에 자동 추가됨
  - SubActivity 클래스가 새로 만들어진 클래스
    - 새로 열릴 화면으로 가정.
  - `<activity android:name=".SubActivity" android:label="SubActivity" />`

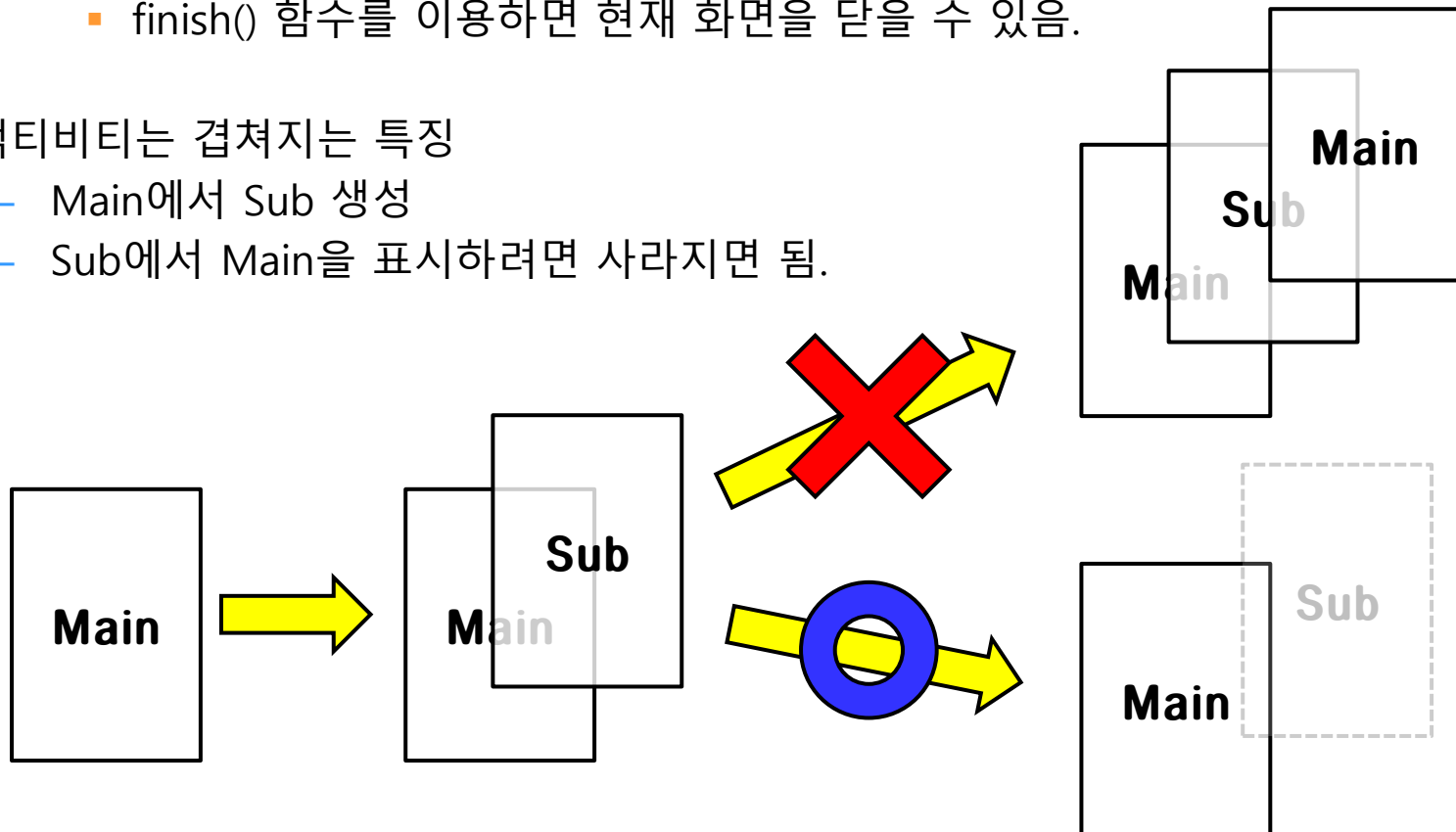
- 각 화면에서 사용할 XML 레이아웃을 생성
  - 버튼1개씩 추가
  - 버튼은 화면 전환을 위한 목적으로 사용
- 메인 액티비티에서 버튼 리스너 추가 후 아래와 같은 기능을 구현.
  1. onClick 리스너를 추가
    - 클릭 시 화면전환 인텐트를 생성하도록 구현
    - startActivity()를 이용해 SubActivity 실행

- 인텐트를 이용한 화면 전환 방법
  - MainActivity에 this를 붙인 이유 ? (인스턴스)
  - SubActivity에 class를 붙인 이유 ? (아직 인스턴스가 생성되지 않은 클래스)

```
Intent intent= new Intent(MainActivity.this, SubActivity.class);
```

```
startActivity(intent);
```

- 두 번째 화면인 서브액티비티
  - 버튼 리스너 추가 후 기능 구현
  - 액티비티를 종료하도록 구현
    - finish() 함수를 이용하면 현재 화면을 닫을 수 있음.
- 액티비티는 겹쳐지는 특징
  - Main에서 Sub 생성
  - Sub에서 Main을 표시하려면 사라지면 됨.



- 기본 프로젝트 만들고 화면 전환하기 기능을 구현
  - 새로 생성된 액티비티는 매니페스트 파일에 등록을 하고 시작

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cafe.naver.com.android21"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".ScreenChange"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".SubActivity" android:label="SubActivity" />
    </application>
    <uses-sdk android:minSdkVersion="8" />
</manifest>
```

- 메인화면에 표시될 화면을 XML로 구현-1/2

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="안드로이드 APK 파일의 스키마 경로는 생략합니다."
    xmlns:app="안드로이드 APK 파일의 스키마 경로는 생략"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:text="Main Activity"
        android:textSize="28dp"
        android:textColor="#000000"
        app:layout_constraintBottom_toBottomOf="parent" // 부모를
        기준으로 맞춤
```



- 메인화면에 표시될 화면을 XML로 구현-2/2

```
app:layout_constraintLeft_toLeftOf="parent" // 부모를 기준으로 맞춤  
app:layout_constraintRight_toRightOf="parent" // 부모를 기준으로  
맞춤  
app:layout_constraintTop_toTopOf="parent" // 부모를 기준으로 맞춤  
</>
```

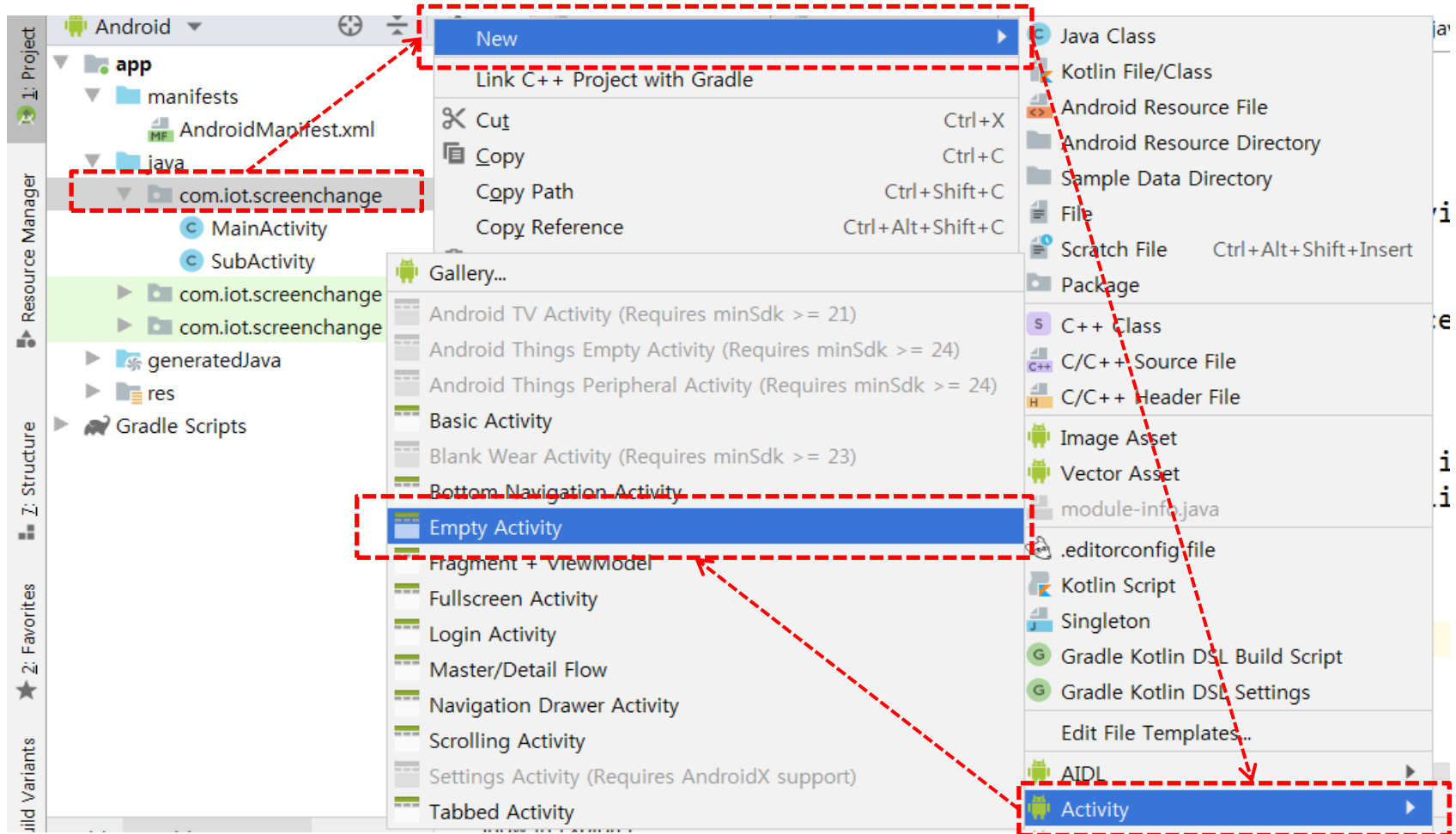
```
<Button
```

```
    android:id="@+id/button"  
    android:layout_width="wrap_content" // 콘텐츠에 맞게 최소화  
    android:layout_height="wrap_content" // 콘텐츠에 맞게 최소화  
    android:text="Push"  
    android:textSize="28dp"  
    android:textColor="#000000"  
    app:layout_constraintTop_toBottomOf="@id/textView"  
    app:layout_constraintLeft_toLeftOf="@id/textView"  
    app:layout_constraintRight_toRightOf="@id/textView"  
</>
```

```
</android.support.constraint.ConstraintLayout>
```


# SubActivity 만들기


- 왼쪽 탐색 창에서 현재 패키지에 마우스 오른쪽 클릭
  - New > Activity > Empty Activity
  - 이렇게 만들면 매니페스트 파일에 **자동으로 액티비티가 등록** 됨




# SubActivity 만들기

New Android Activity

 **Configure Activity**  
Android Studio



**Creates a new empty activity**



Activity Name:

☒ Generate Layout File

Layout Name:

☐ Launcher Activity

Package name:

Source Language:

The name of the activity class to create

- SubActivity용 화면 구현 -1/2

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    android:orientation="vertical"
    android:gravity="center"
    >

    <TextView
        android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:text="Sub Activity"
        android:textColor="#000000"
        android:textSize="28dp"
    />
```

- SubActivity용 화면 구현 -2/2

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
    android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
    android:text="Back"
    android:textSize="28dp"
    android:textColor="#000000"
/>
</LinearLayout>
```

```
package com.iot.screenchange;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button button =(Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, SubActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

```
package com.iot.screenchange;

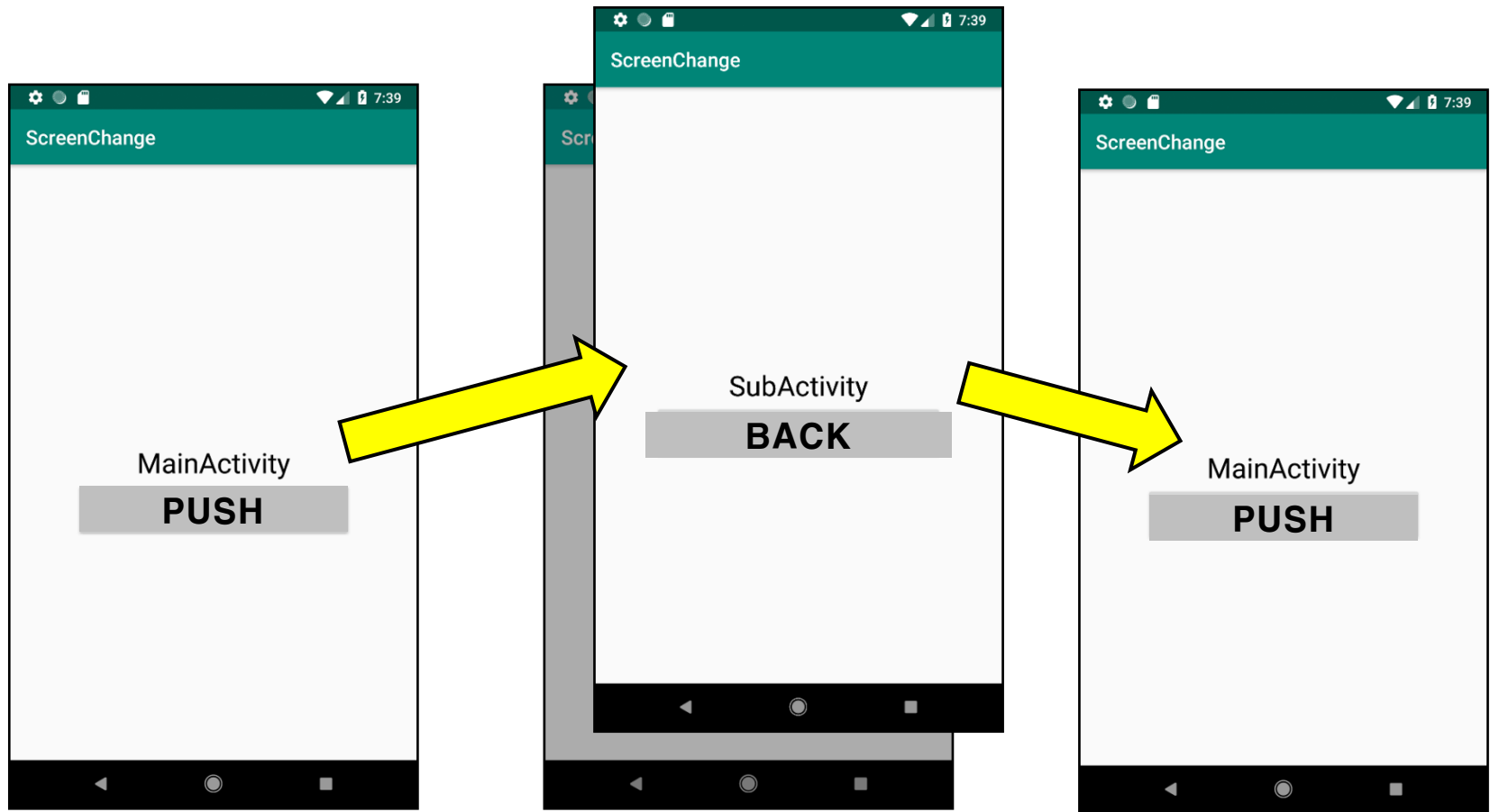
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class SubActivity extends AppCompatActivity {

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

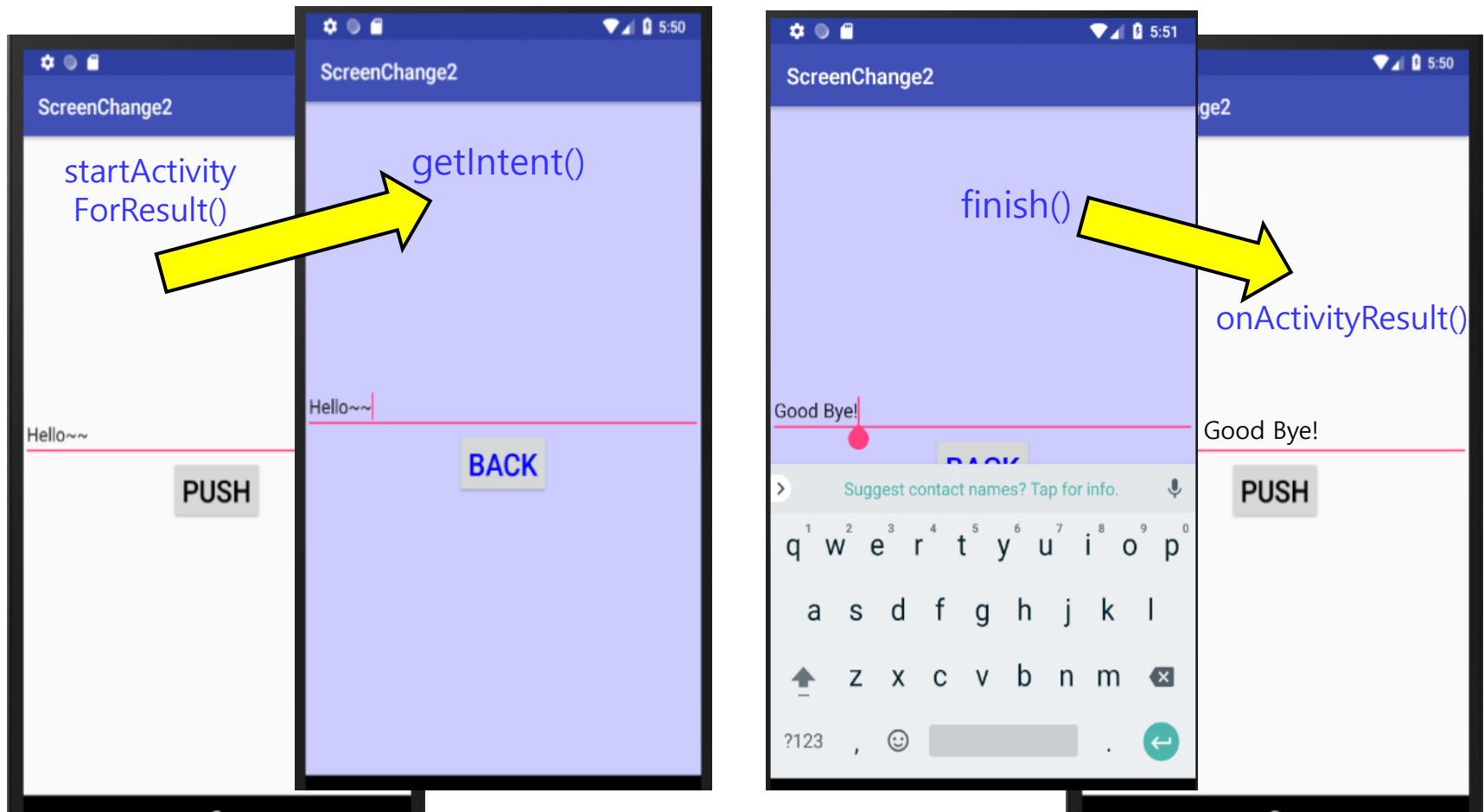
        Button button = (Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}
```

- 액티비티는 화면이 z 축으로 겹치는 형태로 동작





- 다음 화면으로 메시지를 전달하기
  - 다시 이전 화면으로 메시지를 돌려주는 예제를 구현해 보자.



- MainActivity의 onActivityResult() 메소드 오버라이드
  - SubActivity에서 setIntent()로 남겨 놓은 intent를 받을 수 있음.
  - 단, SubActivity 실행 시 startActivityForResult()를 이용해야 함.
- SubActivity에서 setIntent()로 남겨놓은 메시지
  - MainActivity가 다시 실행될 때 돌려 받게 됨
  - onActivityResult()로 전달된 intent를 TAG를 이용해 꺼내서 얻을 수 있음.

- 메인 액티비티 화면 구현하기: 한 개의 글자입력창과 버튼

- activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android "
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    android:orientation="vertical"
    android:gravity="center"
    >

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent" // 부모를 안벗어나게 최대화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:hint="Input Message"
        />
```

- 메인 액티비티 화면 구현하기: 한 개의 글자입력창과 버튼

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content" // 컨텐츠에 맞게 최소화
    android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
    android:text="Push"
    android:textColor="#000000"
    android:textSize="28dp"
/>
</LinearLayout>
```

- 두 번째 화면을 위한 activity\_sub.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    android:orientation="vertical"
    android:gravity="center"
    android:background="#ccccFF"
    >
    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent" // 부모를 안벗어나게 최대화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:hint="Input Message"
    />
```

- 두 번째 화면을 위한 activity\_sub.xml

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content" // 콘텐츠에 맞게 최소화
    android:layout_height="wrap_content" // 콘텐츠에 맞게 최소화
    android:text="Back"
    android:textSize="28dp"
    android:textColor="#0000FF"
/>
</LinearLayout>
```

- 자바코드에서는 아래와 같이 인텐트에 문자열을 넣어서 전달하도록 구현
  - MainActivity.java

```
package com.iot.screenchange2;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {

    public static final int REQUEST_CODE = 1; // 0 이면 onActivityResult() 안 불림
    public static final String TAG_MSG = "message";
    private EditText editText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main); /
```

- 자바코드에서는 아래와 같이 인텐트에 문자열을 넣어서 전달하도록 구현

```
editText = (EditText)findViewById(R.id.editText);
Button button = (Button)findViewById(R.id.button);

button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MainActivity.this, SubActivity.class);
        String msg = editText.getText().toString();
        editText.setText("");
        intent.putExtra(TAG_MSG, msg);

        // startActivityForResult()를 호출 후 서브가 닫히면 onActivityResult가 불림
        startActivityForResult(intent, REQUEST_CODE);
        // Request code는 onActivityResult가 불렸을때 내가 뭘 시켰었는지 구분
    }
});

@Override // 이 메소드는 Sub화면이 종료될 때 자동 호출 됨
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    String msg = data.getStringExtra(TAG_MSG);
    editText.setText(msg);
}
}
```



- SubActivity.java

```
package com.iot.screenchange2;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class SubActivity extends AppCompatActivity {

    private EditText editText;

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

        Intent intent = getIntent();
        String msg = intent.getStringExtra(MainActivity.TAG_MSG);
```

- SubActivity.java

```
editText = (EditText)findViewById(R.id.editText);
editText.setText(msg);

Button button = (Button)findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        Intent intent = new Intent();

        String msg = editText.getText().toString();
        intent.putExtra(MainActivity.TAG_MSG, msg);

        setResult (RESULT_OK, intent); // intent를 시스템에 남김

        finish(); // 화면 닫기 →메인의 onActivityResult() 자동호출
    }
});
}
```

```
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    String msg = data.getStringExtra(TAG_MSG);
    editText.setText(msg);
}
```

**requestCode** : 메인액티비티가 시켰던 일의 종류(1 이상 이어야 동작)

**resultCode** : 서브액티비티가 반환했던 결과 값

**Intent data** : 서브액티비티가 남겼던 intent

- 서브 액티비티가 종료하기 전 남겼던 intent 이용
  - TAG\_MSG를 주고 문자열 msg를 가져온다.

# 실행 확인

- Main 화면에서 Hello~ 입력 후 Push
  - Sub 화면에 Main에서 보낸 메시지 표시
- Sub 화면에서 Good Bye! 입력 후 back
  - Sub 종료 후 Main 화면에 Sub에서 보내준 메시지 표시

