11주차 정리

1. 안드로이드 4대 컴포넌트

안드로이드 4대 컴포넌트 - **액티비티, 서비스, 브로드캐스트 리시버, 콘텐트 프로바이더**

- **액티비티(Activity)**
  - 화면을 구성하는 가장 기본적인 컴포넌트

- **서비스(Service)**
  - 액티비티와 상관없이 백그라운드에서 동작하는 컴포넌트

```
서비스 생성 → 서비스 시작 → 서비스 종료
```

- **브로드캐스트 리시버(Broadcast Receiver)**
  - 문자 메시지 도착, 배터리 방전, SD 카드 탈부착, 네트워크 환경 변화 등이 발생하면 전체 응용프로그램이 들을 수 있도록 방송 신호 보냄 (14장에서 더 자세히 다룰 예정)

- **콘텐트 프로바이더(Content Provider)**
  - 응용프로그램 사이에 데이터를 상호 공유하기 위한 컴포넌트
  - 안드로이드 애플리케이션의 데이터는 자신만 접근할 수 있고, 자신의 데이터를 외부에 공개하기 위해서는 콘텐트 프로바이더를 만들어야 함
  - 콘텐트 프로바이더의 정보를 제공하는 방법으로는 **URI(Uniform Resource Identifier)**가 있음
  - 콘텐트 프로바이더에서 처리된 데이터는 일반적으로 데이터베이스 또는 파일로 저장됨

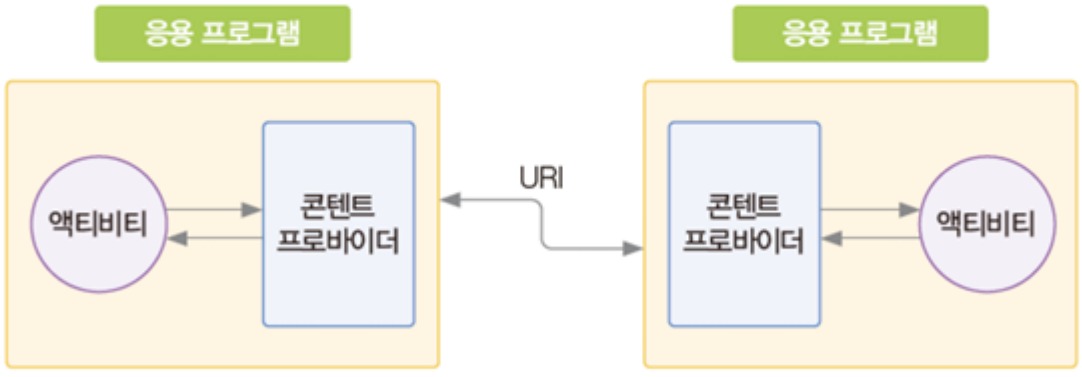

그림 10-1 **콘텐트 프로바이더와 URI의 개념**

액티비티 - 화면을 구성하는 가장 기본적인 컴포넌트

서비스 - 액티비티와 상관없이 백그라운드에서 동작하는 컴포넌트

브로드캐스트 리시버 - 전체 응용 프로그램이 변화를 들을 수 있도록 방송 신호를 받아서 보내는 컴포넌트

콘텐트 프로바이더 - 응용 프로그램 사이에 데이터를 상호 공유하기 위한 컴포넌트

---

2. 액티비티

일반적으로 액티비티 하나당 XML 파일 하나를 만들어서 사용한다.

MainActivity.java코드는 **Activity 클래스를 상속**받으므로 MainActivity.java를 액티비티라 부름

```
public class MainActivity extends Activity {
        @Override
        public void onCreate(Bundle savedInstanceState) {
        ~~~~~ 생략 ~~~~~
```

새로운 액티비티 추가하기 - ex) [프로젝트] 우클릭 - java - classname -SecondActivty
+ second.xml 추가 →onCreate안에 setContentView(R.layout.second);추 가

액티비티 여러개 사용시 AndroidManifest에 등록을 해줘야한다.

예제 10-3 SecondActivity.java 코드 1

```
1  public class SecondActivity extends Activity {
2
3      @Override
4      protected void onCreate(Bundle savedInstanceState) {
5
6          super.onCreate(savedInstanceState);
7          setContentView(R.layout.second);
8
9      }
10 }
```

액티비티의 필수 메소드인 onCreate()를 추가하고 자동 완성해야 한다. → setContentView를 추가해줘야한다.
finish()를 사용하면 현재 액티비티를 끝낸다. - 세컨트 액비티비 java 코드에 추가

메인 액티비티에서 세컨트 액티비티 호출하기

예제 10-5 MainActivity.java 코드

```
1  Button btnNewActivity = (Button) findViewById(R.id.btnNewActivity);
2  btnNewActivity.setOnClickListener(new View.OnClickListener() {
3      public void onClick(View v) {
4          Intent intent = new Intent(getApplicationContext(),
5                                     SecondActivity.class);
6          startActivity(intent);
7      }
8  });
```

getApplicationContext()는 MainActivity.this와 같다.

즉, 메인 액티비티에서 세컨드 액티비티로 넘어가라는 뜻이다.

참고 서브 액티비티는 /res/layout 폴더에 생성하는 듯 하다

- 완성된 프로젝트를 실행하면 다음과 같은 오류가 발생함

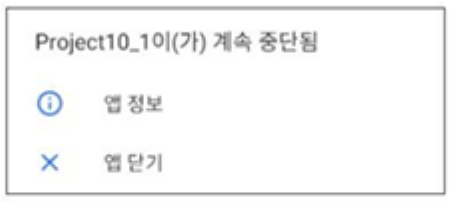

그림 10-4 오류 발생 화면

- 안드로이드에서는 사용될 액티비티를 AndroidManifest.xml에 꼭 등록해야 함
- 메인 액티비티(MainActivity)는 자동으로 등록되지만, 추가한 세컨드 액티비티는 별도로 등록해줘야 함

SecondActivity extends Activity의 SecondActicity를 AndroidManifest.xml에 추가

- AndroidManifest.xml 파일을 열고 SecondActivity를 등록
  - 아래 코드를 </application> 바로 윗행에 넣고 다시 실행하기

```
<activity android:name=".SecondActivity" android:label="Second 액티비티"/>
```

3. 명시적 인텐트

인텐트 - 안드로이드 4대 컴포넌트가 상호 간에 데이터를 주고 받기 위한 메시지 객체

명시적 인텐트와 **암시적 인텐트**로 구분한다.

명시적 인텐트와 데이터의 전달

- **명시적 인텐트와 데이터의 전달**
  - 명시적 인텐트 : 다른 액티비티의 이름을 명확히 지정할 때 사용하는 방법

```
Intent intent = new Intent(getApplicationContext(), SecondActivity.class);
startActivity(intent);
```

  - 메인 액티비티에서 인텐트에 데이터를 실어서 넘긴 후, 세컨드 액티비티에서 받은 데이터 처리하는 방식

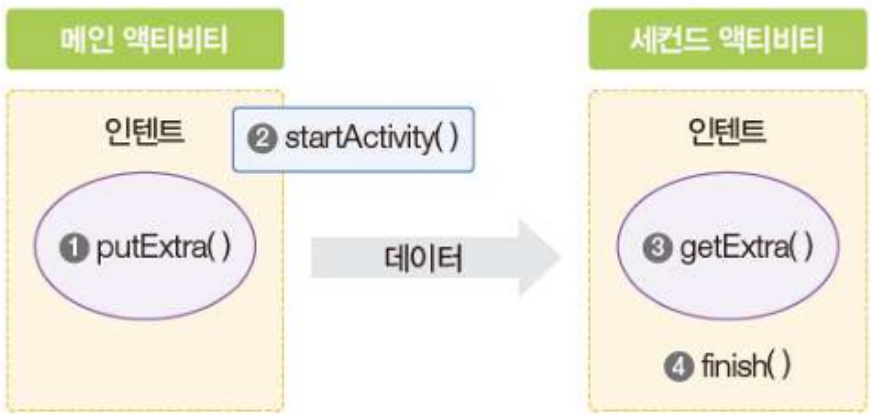

그림 10-6 한쪽 방향으로 데이터를 전달하는 방법

약간 Spring에서 Model 객체랑 비슷함

1. putExtra()로 데이터 포장
2. startActivity()로 데이터 이동
3. getExtra()로 데이터 꺼내기
4. finish() 명시적 종료 선언 - 끝내기

액티비티와 인텐트의 응용

양방향 액티비티

메인 액티비티에서 세컨트 액티비티로 데이터를 넘긴 후 다시 데이터를 돌려받는 경우

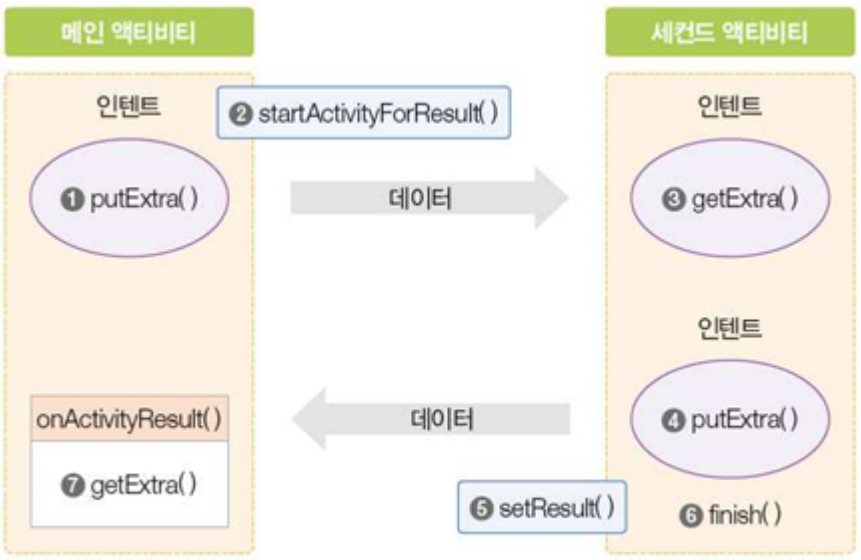

그림 10-9 양방향으로 데이터를 전달하는 방법

1. putExtra()
2. startActivityForResult()
3. getExtra()
4. putExtra()
5. setResult()
6. finish()
7. getExtra()
   → onActivityResult() 메소드 안에 구현

예제 10-17 메인 액티비티의 Java 코드

```
1  public void onCreate(Bundle savedInstanceState) {
2    super.onCreate(savedInstanceState);
3    setContentView(R.layout.activity_main);
4    setTitle("메인 액티비티");
5
6    Button btnNewActivity = (Button) findViewById(R.id.btnNewActivity);
7    btnNewActivity.setOnClickListener(new View.OnClickListener() {
8    public void onClick(View v) {
9      EditText edtNum1 = (EditText) findViewById(R.id.edtNum1);
10     EditText edtNum2 = (EditText) findViewById(R.id.edtNum2);
11     Intent intent = new Intent(getApplicationContext(), SecondActivity.class);
12     intent.putExtra("Num1", Integer.parseInt(edtNum1.getText().toString()));
13     intent.putExtra("Num2", Integer.parseInt(edtNum2.getText().toString()));
14     startActivityForResult(intent, 0);
15   }
16   });
17 }
18
19 @Override
20 protected void onActivityResult(int requestCode,
21                 int resultCode, Intent data) {
22   if(resultCode == RESULT_OK) {
23   int hap = data.getIntExtra("Hap", 0);
24   Toast.makeText(getApplicationContext(),
25                 "합계 :" + hap, Toast.LENGTH_SHORT).show();
26   }
27 }
```

예제 10-18 세컨드 액티비티의 Java 코드

```
1  protected void onCreate(Bundle savedInstanceState) {
2     super.onCreate(savedInstanceState);
3     setContentView(R.layout.second);
4     setTitle("Second 액티비티");
5
6     Intent inIntent = getIntent();
7     final int hapValue = inIntent.getIntExtra("Num1", 0)
8                                  +inIntent.getIntExtra("Num2", 0 );
9
10    Button btnReturn = (Button) findViewById(R.id.btnReturn);
11    btnReturn.setOnClickListener(new View.OnClickListener() {
12       public void onClick(View v) {
13          Intent outIntent = new Intent(getApplicationContext(),
14                                        MainActivity.class);
15          outIntent.putExtra("Hap", hapValue);
16          setResult(RESULT_OK,outIntent);
17          finish();
18       }
19    });
20 }
```

위 코드는 **버튼을 클릭했을 때 '0'이라는 requestCode를 가지고 WriteActivity로 이동하는 코드**이다. WriteActivity에서 작업을 마치고 돌아올 때 그대로 requestCode를 들고 원래 액티비티로 돌아오게 된다.

암시적 인텐트

약속된 액션(Action)을 지정하여 안드로이드에서 제공하는 기존 응용 프로그램 실행하는 것이다.

- 암시적 인텐트의 예시 : 전화 걸기
  - 전화번호를 인텐트로 넘긴 후에 전화 걸기 응용 프로그램이 실행되는 것과 같음

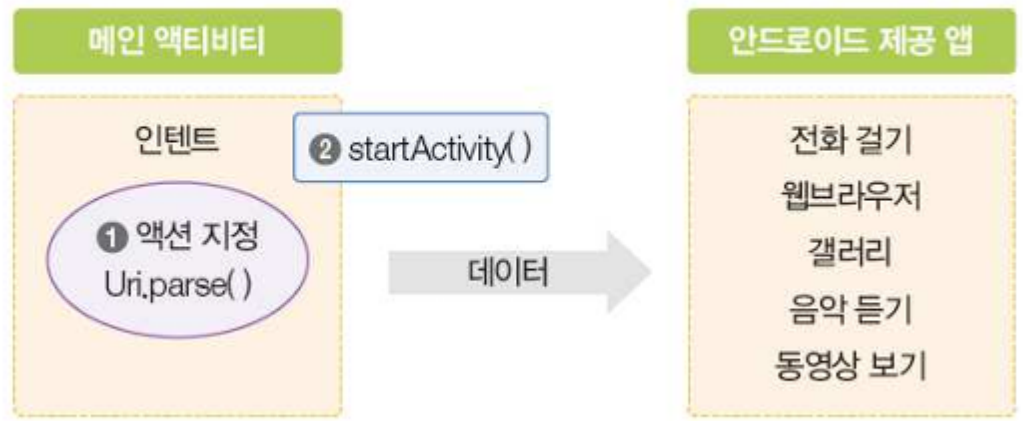

그림 10-12 암시적 인텐트의 개념

- 119에 응급 전화를 거는 형식

```
Intent intent = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:/119"));
startActivity(intent);
```

- 전화 걸기 및 구글 맵 사용을 위해 AndroidManifest.xml의 **<application 위에** 다음과 같이 권한 추가

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.CALL_PHONE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

Intent.ACTION_VIEW 이런 값들은 이미 정해진 값

```
Uri uri = Uri.parse("tel:01012345678");
Intent intent = new Intent(Intent.ACTION_DIAL, uri);
```

uri가 다음 액티비티로 넘어간다.

Intent.ACTION_DIAL에서 DIAL 대신, VIEW, QUERY, IMAGE_CAPTURE 등이 있음 → 교재 393

액티비티 생명주기

- 액티비티의 생성부터 소멸T까지의 주기를 뜻한다.
- 안드로이드 응용프로그램은 화면이 작아 동시에 여러 개의 액비티비가 나올 수 없다.
- 앞에 나오는 화면 하나만 활성화된 상태이고, 나머지는 모두 비활성화 상태로 남게된다.

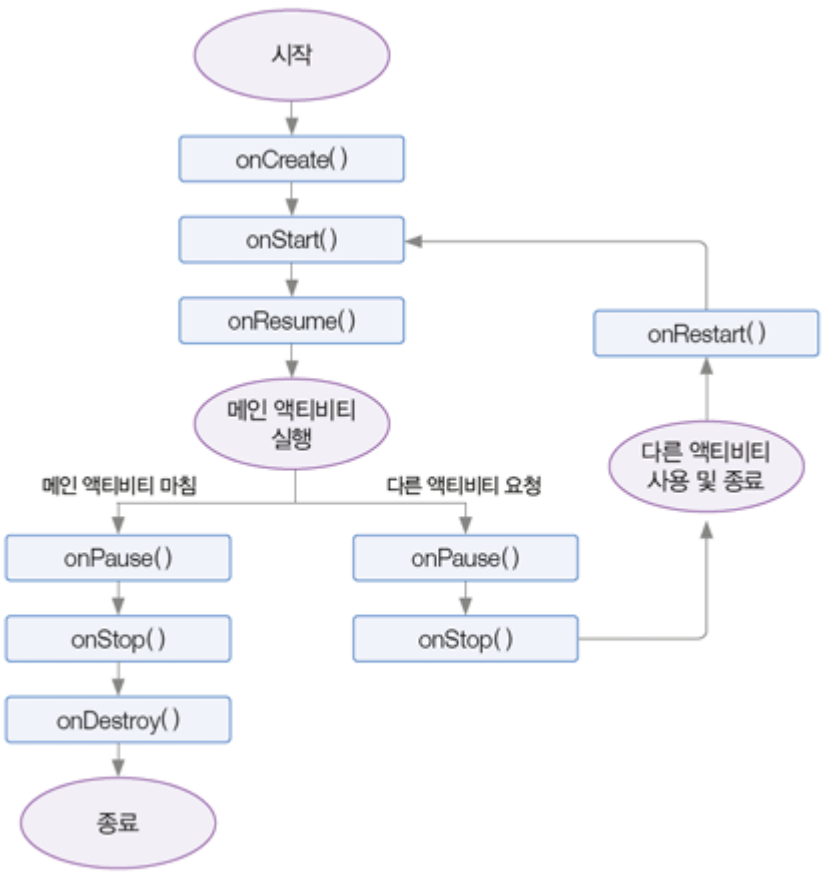

그림 10-13 액티비티 생명주기

onCreate() - onDestory()

onStart() - onStop()

onResume() - onPause()

onRestart()

첫번쨰 액티비티가 인터럽트를 받으면 일시정지(onStop)이 되는것이고 다른 액티비티가 종료되면 onRestart해서 다시 실행할 수 있다. 완전히 종료하려면 onDestory()

완전 종료된 액티비티를 다시 실행하려면 onCreate()해서 생성해야함.

다른 액티비티 사용 및 종료 - finish() 사용으로 알 수 있음

즉, 첫번째 화면에서 다른 화면으로 넘어갈 때 첫번쨰 화면은 죽은게 아니라 일시정지 상태임 → **onRestart()**하면됨

로그캣

d - Debugging - 디버깅 용도로 남기는 로그

e - Error - 심각한 오류 남기기

i - Information - 정보 남기기

v - Verbose - 상세한 기록 남기기

w - Warning - 경고 수준을 남기기

여기서 잠깐 로그캣(LogCat)

작성 중인 프로그램에 예기치 못한 오류가 발생했을 때 원인을 파악하는 방법 중 하나가 로그(log)를 남기는 것이다. 안드로이드는 android.util.Log 클래스를 제공하여 로그를 남기고, 로그캣(logCat)이라는 화면을 제공하여 로그를 확인한다. 프로그래머가 로그를 남기기 위해 사용하는 메소드는 다음 표와 같다. 하지만 절대적인 기준은 아니며 프로그래머가 적절한 메소드를 골라 사용해야 한다.

메소드	설명
android.util.Log.d("태그", "메시지")	Debugging: 디버깅 용도로 남기는 로그
android.util.Log.e("태그", "메시지")	Error: 가장 심각한 오류 발생 시 남기는 로그
android.util.Log.i("태그", "메시지")	Information: 정보를 남기기 위한 로그
android.util.Log.v("태그", "메시지")	Verbose: 상세한 기록을 남기기 위한 로그
android.util.Log.w("태그", "메시지")	Warning: 경고 수준을 남기기 위한 로그

page 47

위 코드는 **버튼을 클릭했을 때 '0'이라는 requestCode를 가지고 WriteActivity로 이동하는 코드**이다. WriteActivity에서 작업을 마치고 돌아올 때 그대로 requestCode를 들고 원래 액티비티로 돌아오게 된다.

page 53
intent.ACTION_VIEW 이런 값들은 이미 정해진 값

```
Uri uri = Uri.parse("tel:01012345678");
Intent intent = new Intent(Intent.ACTION_DIAL, uri);
```

uri가 다음 액티비티로 넘어간다.

page 57

액티비티 생명주기 봐두기 - 암기(대칭성)

onCreate() - onDestory()

onStart() - onStop()

onResume() - onPause()

onRestart()

첫번쨰 액티비티가 인터럽트를 받으면 일시정지(onStop)이 되는것이고 다른 액티비티가 종료되면 onRestart해서 다시 실행할 수 있다. 완전히 종료하려면 onDestory()

완전 종료된 액티비티를 다시 실행하려면 onCreate()해서 생성해야함.

다른 액티비티 사용 및 종료 - finish() 사용으로 알 수 있음

즉, 첫번째 화면에서 다른 화면으로 넘어갈 때 첫번쨰 화면은 죽은게 아니라 일시정지 상태임 → onRestart()하면됨