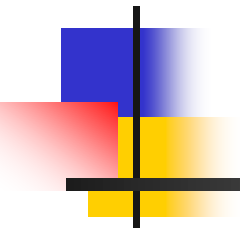
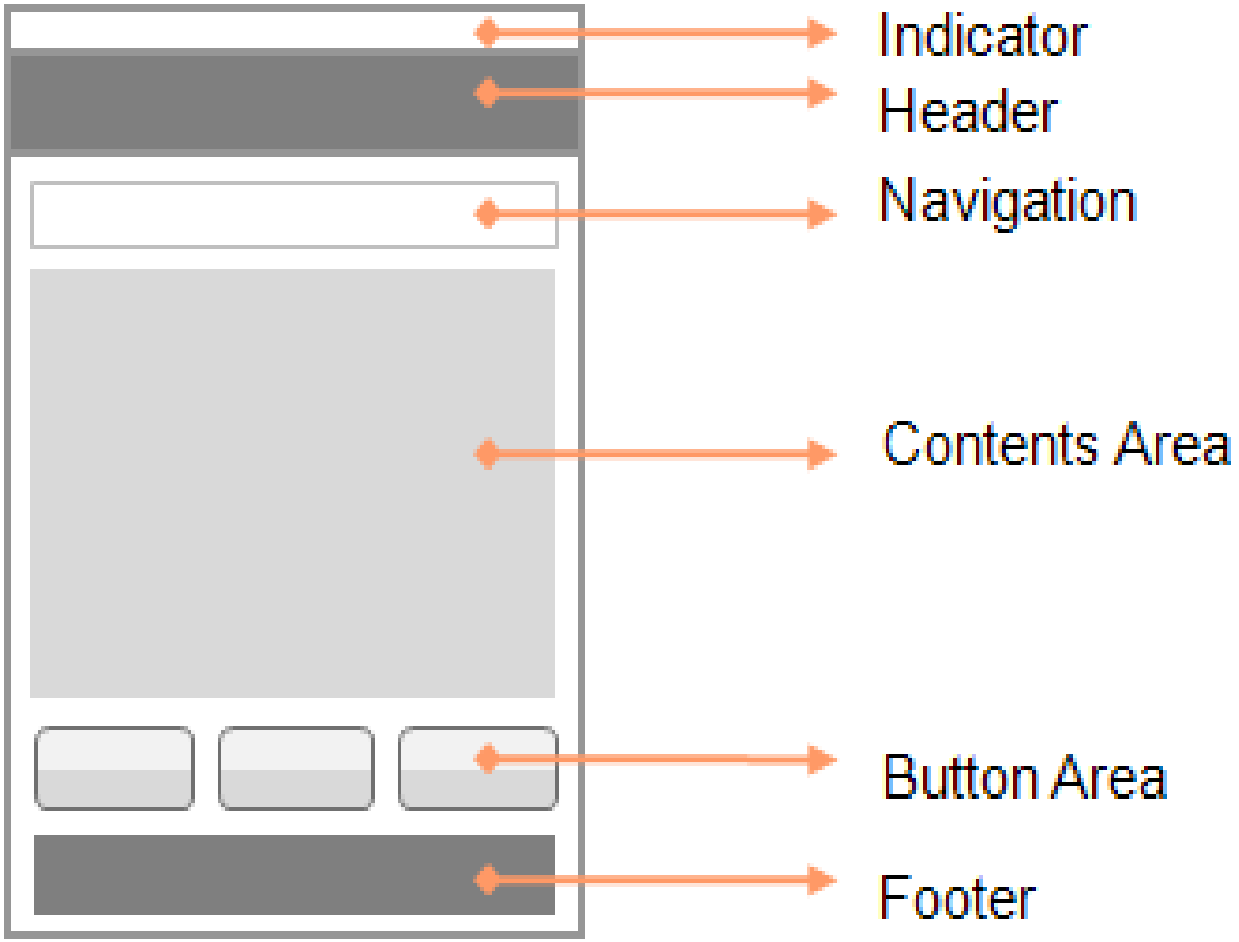


UI 구성요소

배 희호 교수
경복대학교
스마트IT과

모바일 UI 패턴

- 구성요소



모바일 UI 패턴

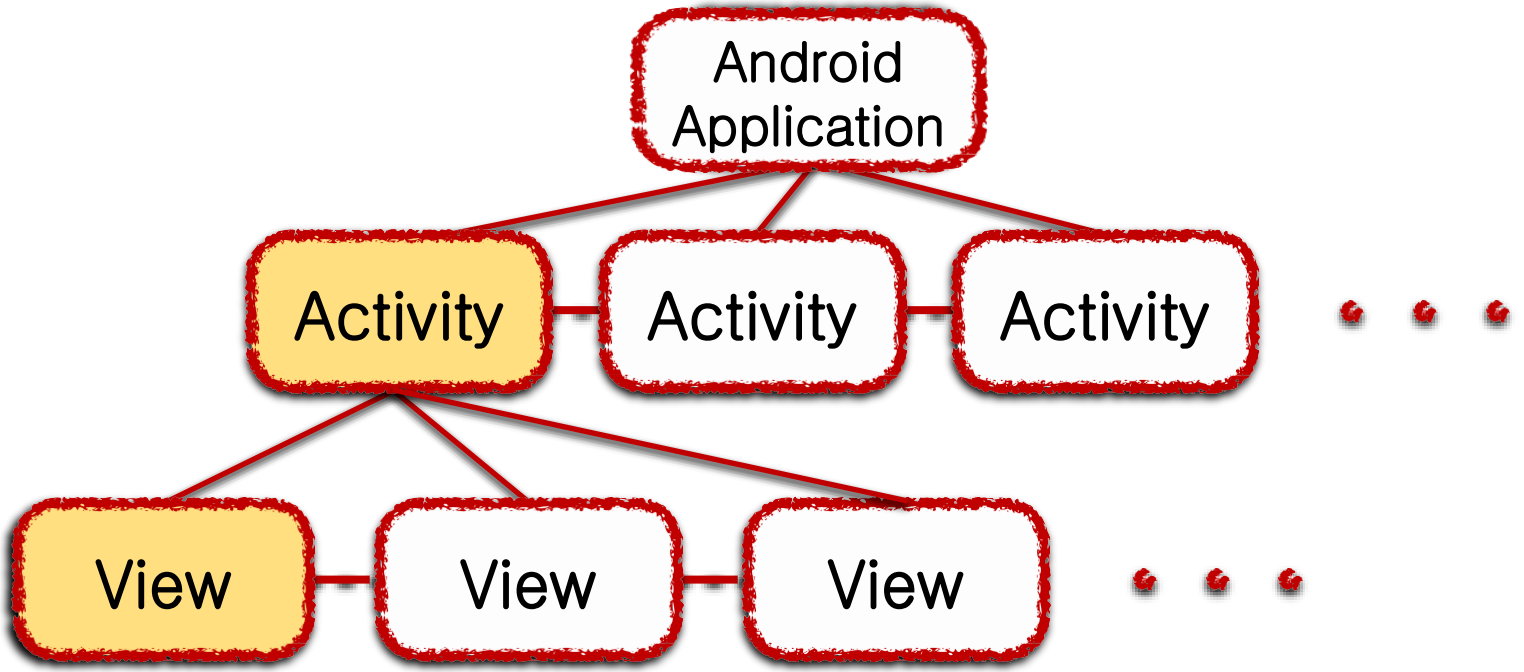
- 구성요소
  - Indicator 영역
    - 각종 Service의 알림 및 수신, 네트워크 ,배터리 상태 등을 Icon화 하여 제공
  - Header 영역
    - 기관의 Log 또는 Site 명을 제공하여 Site의 정체성(Identity)을 전달
  - Navigation 영역
    - 주로 홈이나 이전 또는 최상위 메뉴 같은 주요 이동만 제공되며 모바일 웹의 경우 브라우저 네비게이션 버튼을 활용하므로 생략하는 경우도 있음
  - Content 영역
    - 사용자에게 전달하고자 하는 주요 정보를 담음

모바일 UI 패턴

- 구성요소
  - Button 영역
    - 필요한 경우에만 제공
    - Button은 해당 Content 아래에 배치하는 것이 일반적이나 상단에 위치시키는 경우도 있음
  - Footer 영역
    - 저작권, 연락처, PC 웹으로의 Link 등의 내용이 제공

Android Application

- Android Application의 대략적인 구조

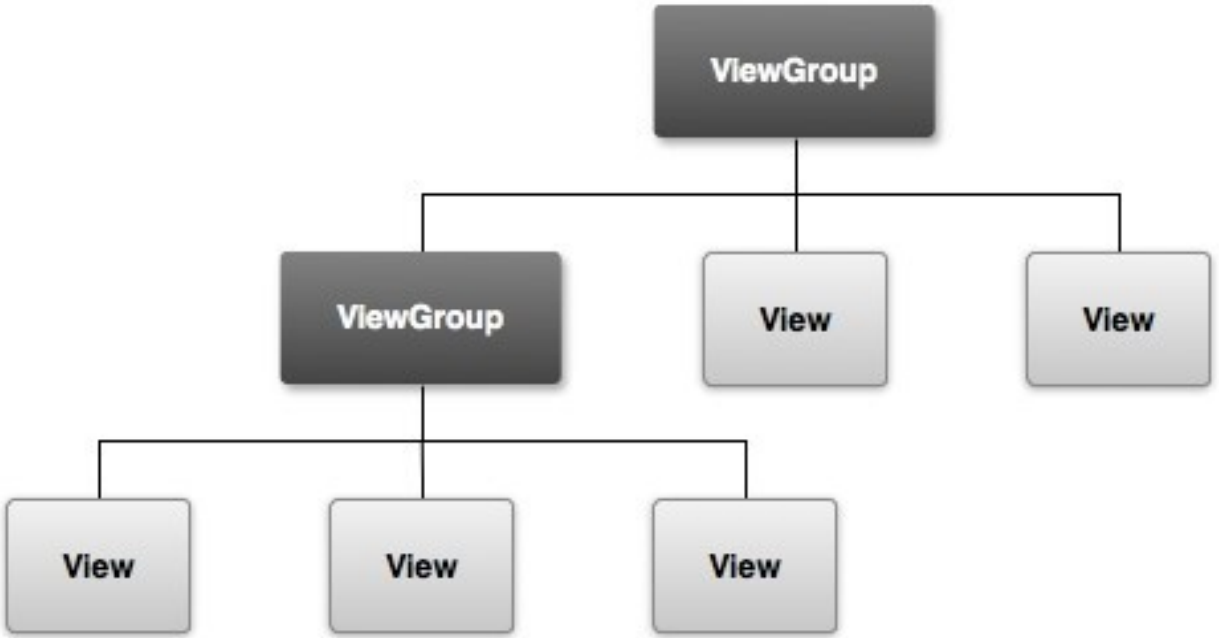


Android Application

- View 여러 개가 모여 Activity를 구성하고, Activity 여러 개가 모여 Application이 됨
  - Android Application?
    - 우리가 만들어야 하는 Application
    - Android App을 만든다는 것은 바로 Activity를 완성하는 것
  - Activity(액티비티)?
    - Activity는 Android App이 사용자들에게 보여주는 화면 단위의 UI Component
    - Activity 자체는 화면 상에 직접적으로 보이지 않음
  - View(뷰)?
    - Android 사용자 인터페이스(UI)를 구성하는 핵심 Component
    - 화면 상의 사각 영역에 직접적으로 드러나 사용자 입력을 받아들임

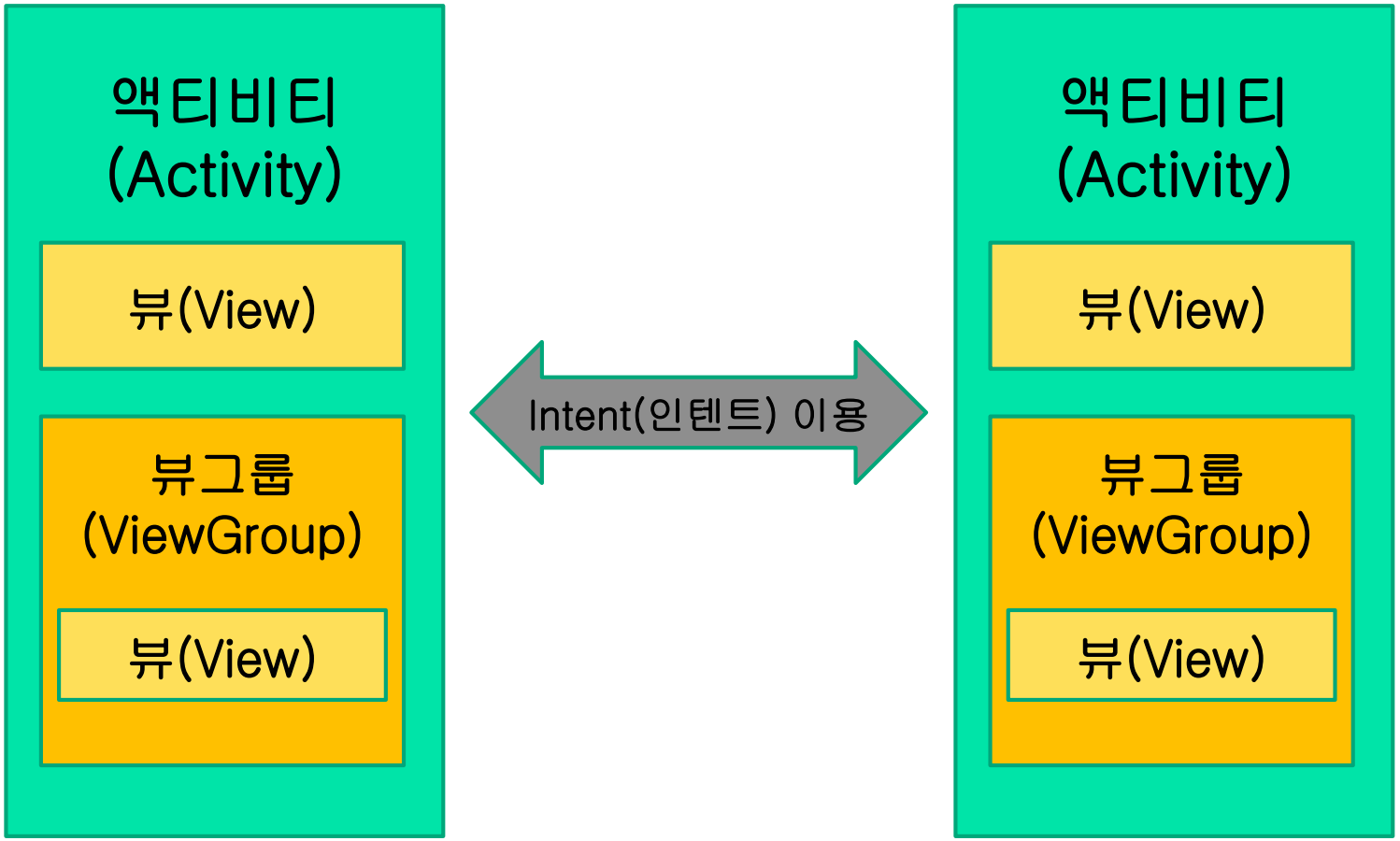
Android Application

- Android App의 모든 유저 인터페이스(UI) 요소들은 View와 ViewGroup 객체들을 기반으로 함
  - View는 화면의 사각형 영역에 있는 Contents나 화면 Layout을 담당
  - ViewGroup은 각 View들을 합쳐 화면 전체 Layout을 그려냄



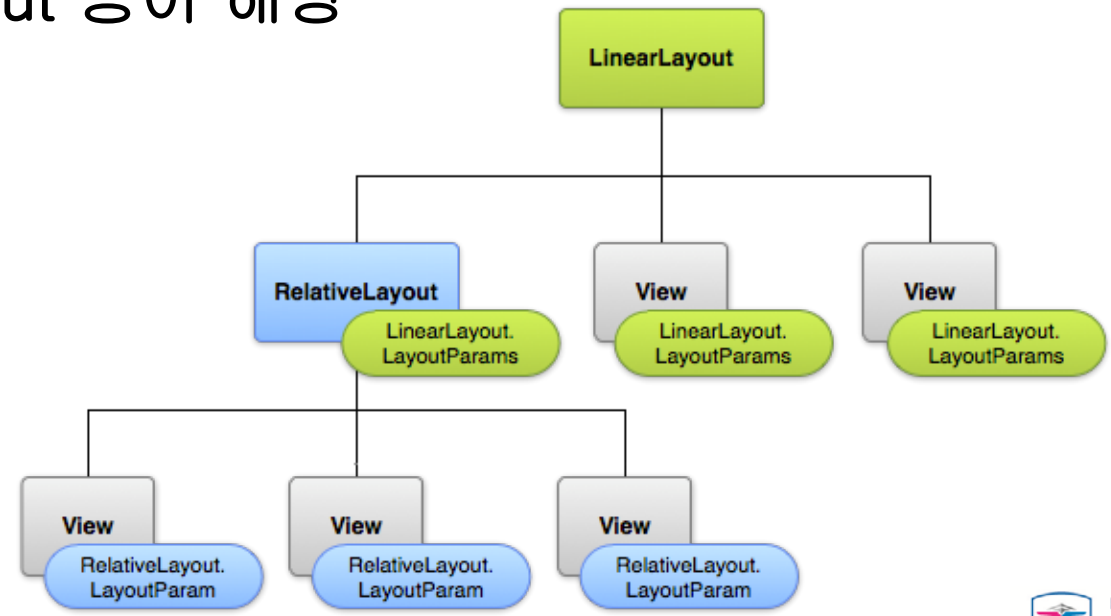
Android Application

- Activity 전환 시 정보를 전달하는 목적으로 만들어진 장치를 Intent(인텐트)라고 함



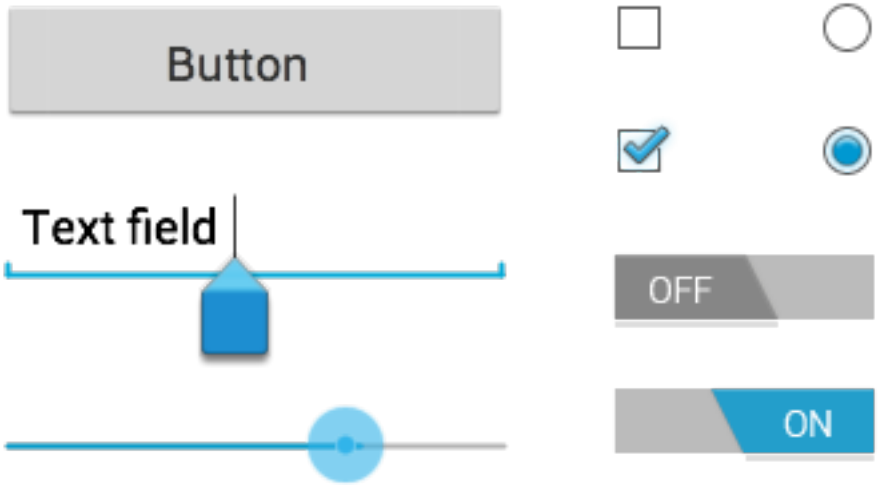
UI Component

- Android UI Component에는 Layout Component와 Control 요소들이 있음
- Layout은 사용자 Interface에 대한 시각적 구조를 말함
- Activity 또는 Widget에 대한 큰 틀이 정의되며, 여러 속성과 매개변수를 정의할 수 있음
- Layout Component는 LinearLayout, RelativeLayout, GridLayout 등이 해당



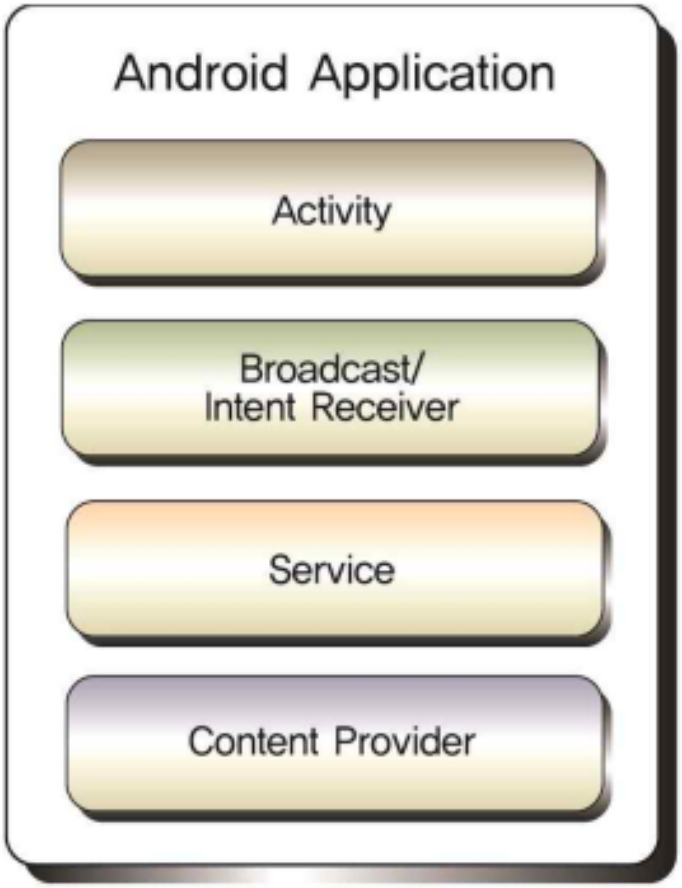
UI Control

- UI Control
  - 사용자 Interface에 있는 대화형 구성 요소
  - Button, 확인란과 같이 App을 사용하고 제어하는데 필요한 요소들을 말함
  - Control 요소에는 Button, TextView, EditText, Radio Button, CheckBox 등이 있음
  - UI에 Control 요소를 추가하려면 XML에 요소를 하나 추가하기만 하면 됨



Android Application

- 안드로이드 애플리케이션 구조



Application Component

구분	기능
Activity	사용자와 Application간의 Interface 제공 객체 하나의 화면에 대응
Service	Background로 수행되는 객체
Content Provider	다른 프로그램에게 자신이 보유한 data를 제공하는 객체
BroadCast Receiver	다른 프로그램이나 OS로부터 intent를 수신하는 객체

Application Component

- Android Application은 최소 1개 이상의 Activity가 포함됨
- Activity
  - 사용자에게 UI를 제공
  - 사용자가 전달한 Event를 처리하는 객체
  - 하나의 화면은 하나의 Activity에 대응
  - Application은 필요에 따라 여러 개의 Activity를 가질 수 있으나 화면에는 하나의 Activity만 표시
- Activity간에 인자, 결과값 전달 가능
- Activity 관리자가 Activity 상태를 관리
  - Activity 동작에 따른 Life Cycle을 가짐
  - Back, Home Button에 의해서 화면이 바뀌어도 기동상태
  - 명시적 종료, 프로세스 관리자에 의한 종료 필요
  - 시스템 메모리가 부족하면 Activity 강제 종료 될 수 있음

Application Component

- Activity의 예
  - Activity들이 모여서 Application이 됨

이메일 애플리케이션

이메일 애플리케이션은 3개의 액티비티로 이루어진다.

Application Component

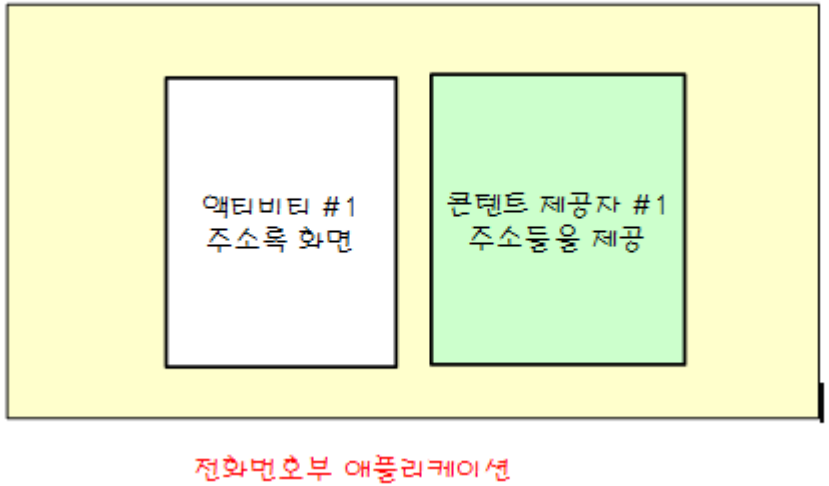
- Service
  - Background에서 실행되는 Component로 오랫동안 실행되는 작업이나 원격 Process를 위한 작업을 할 때 사용
    - UI가 없음
    - 한번 시작된 Service는 Application이 종료되고 다른 Application으로 이동해도 계속 Background에서 실행
    - 모든 Service는 Service 클래스를 상속받아서 작성
    - Network를 통하여 데이터를 꺼내 올 수도 있음
    - 예) 배경 음악을 연주하는 작업

Application Component

- Broadcast Receiver
  - Android 단말기에서 발생하는 다양한 Event/정보(방송)를 받고 반응하는 Component
  - 단말기에서 발생하는 일 중에서 Application이 알아야 하는 상황이 발생하면 방송을 함
  - System Booting, Battery 부족, 전화/문자 수신, 네트워크 끊김을 알려주는 것이 방송임
  - 방송 수신기(BroadcastReceiver)를 통해 상황을 감지하고 적절한 작업을 수행
  - 일반적으로 UI가 없음

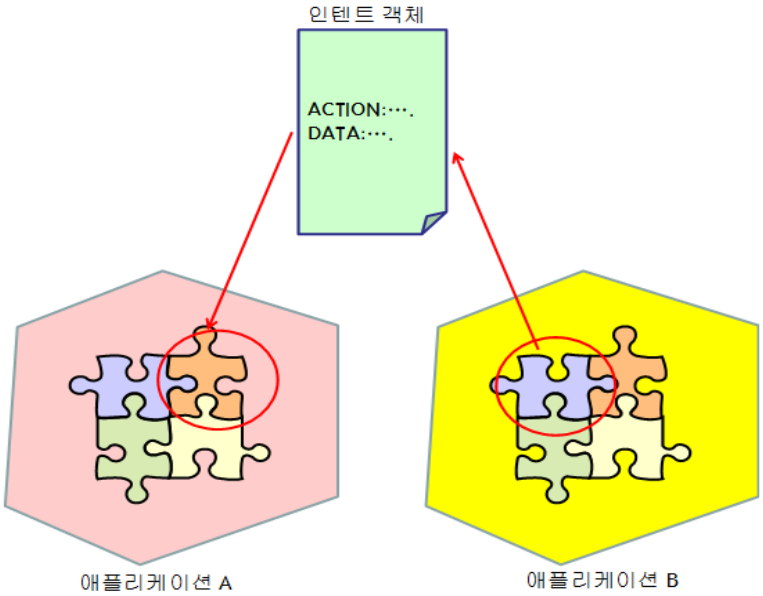
Application Component

- Content Provider
  - Data를 관리하고 다른 Application Data를 제공하는 Component
    - Data는 File System이나 SQLite 데이터베이스, Web상에 저장될 수 있음
    - Content Provider를 통해서 다른 Application의 Data를 Query하거나 변경 가능



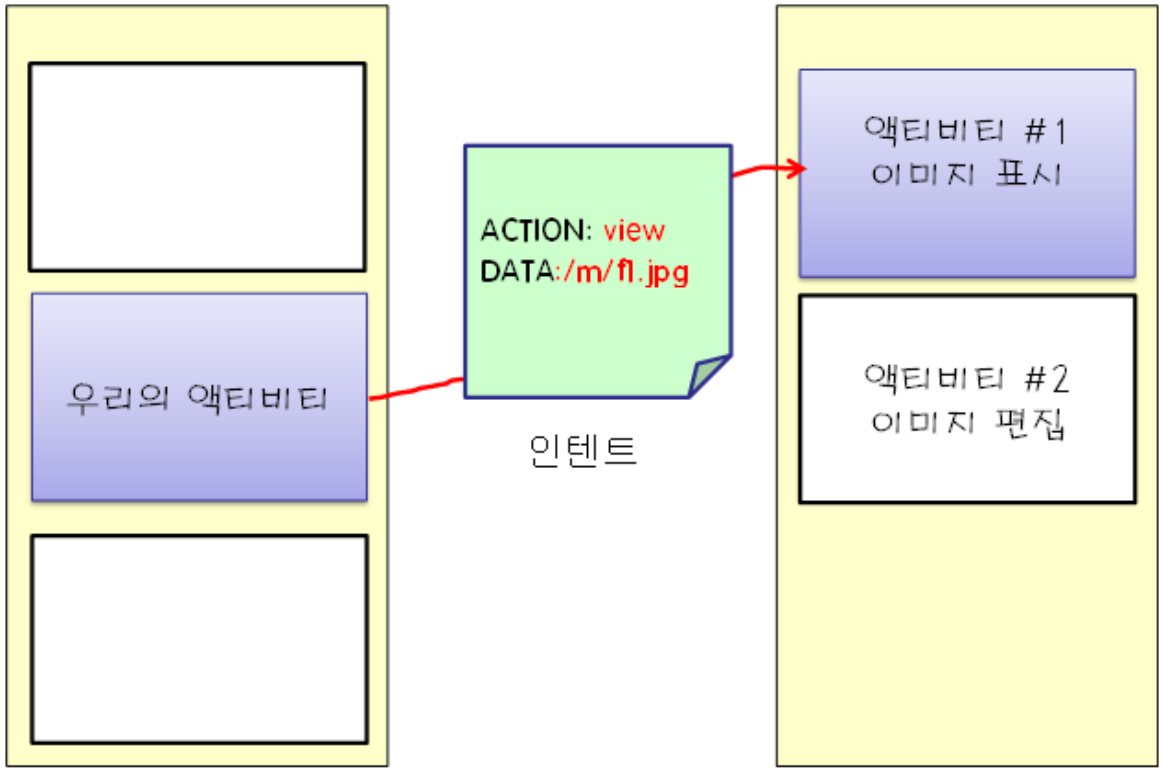
Intent

- 서로 독립적으로 동작하는 4가지 Component들 간의 상호 통신을 위한 장치 (Component 간의 통신수단)
- Intent를 통하여 다른 Application의 Component를 활성화시킬 수 있음
- Application의 의도를 적어서 Android에 전달하면 Android가 가장 적절한 Component를 찾아서 활성화하고 실행



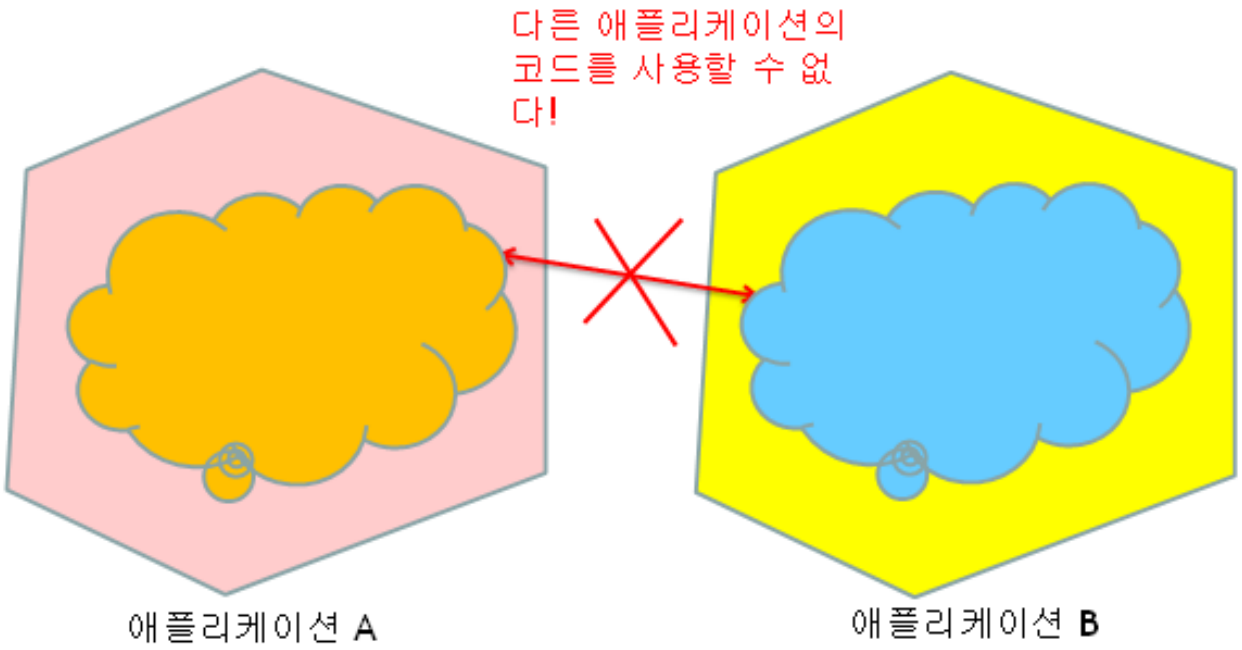
Intent

- Intent 사용의 예



Intent

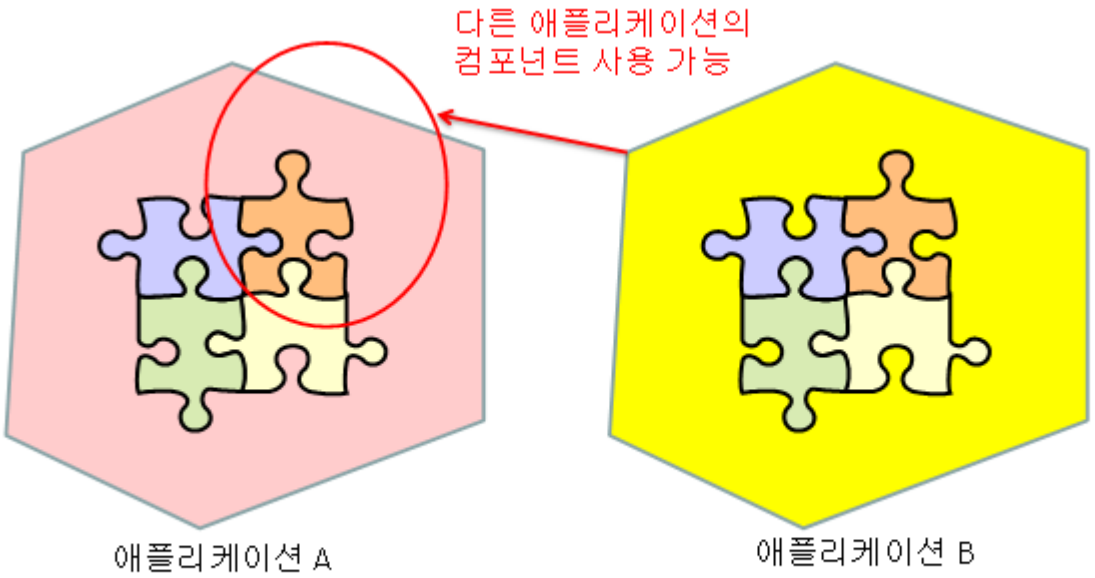
- PC의 Application



PC에서는 다른 애플리케이션이 가지고 있는 코드를 사용할 수 없다.

Intent

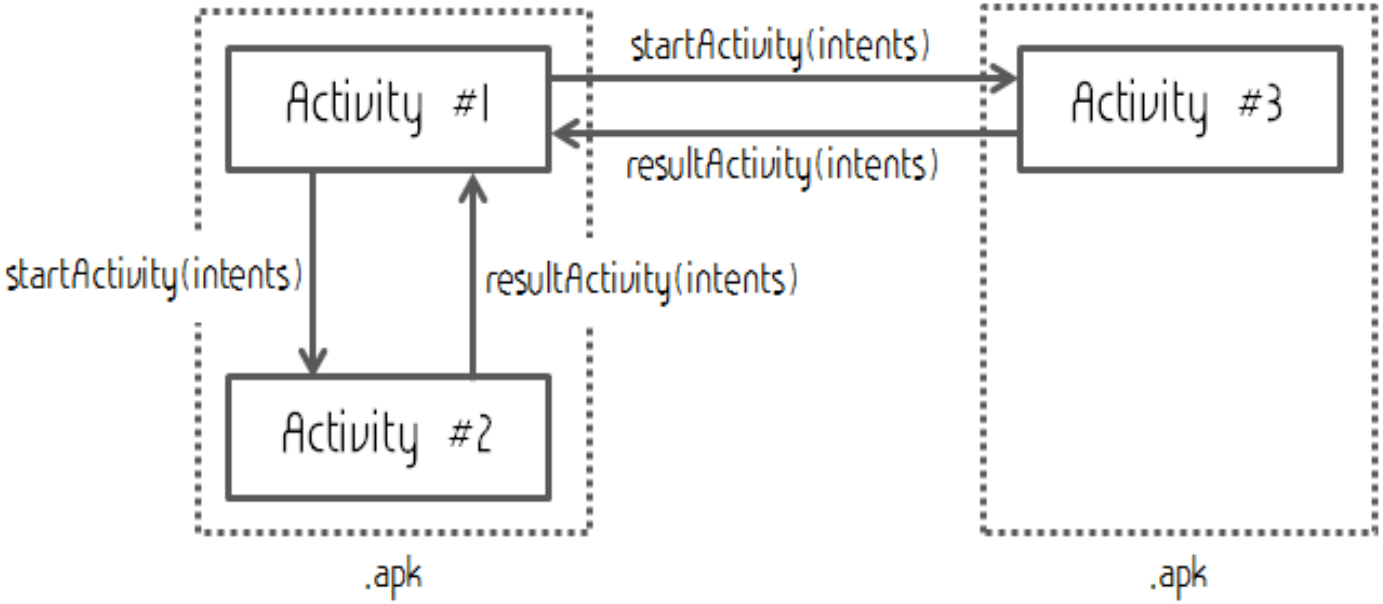
- Android에서는 다른 Component를 사용할 수 있음



안드로이드에서는 다른 애플리케이션이 가지고 있는 컴포넌트를 사용할 수 있다.

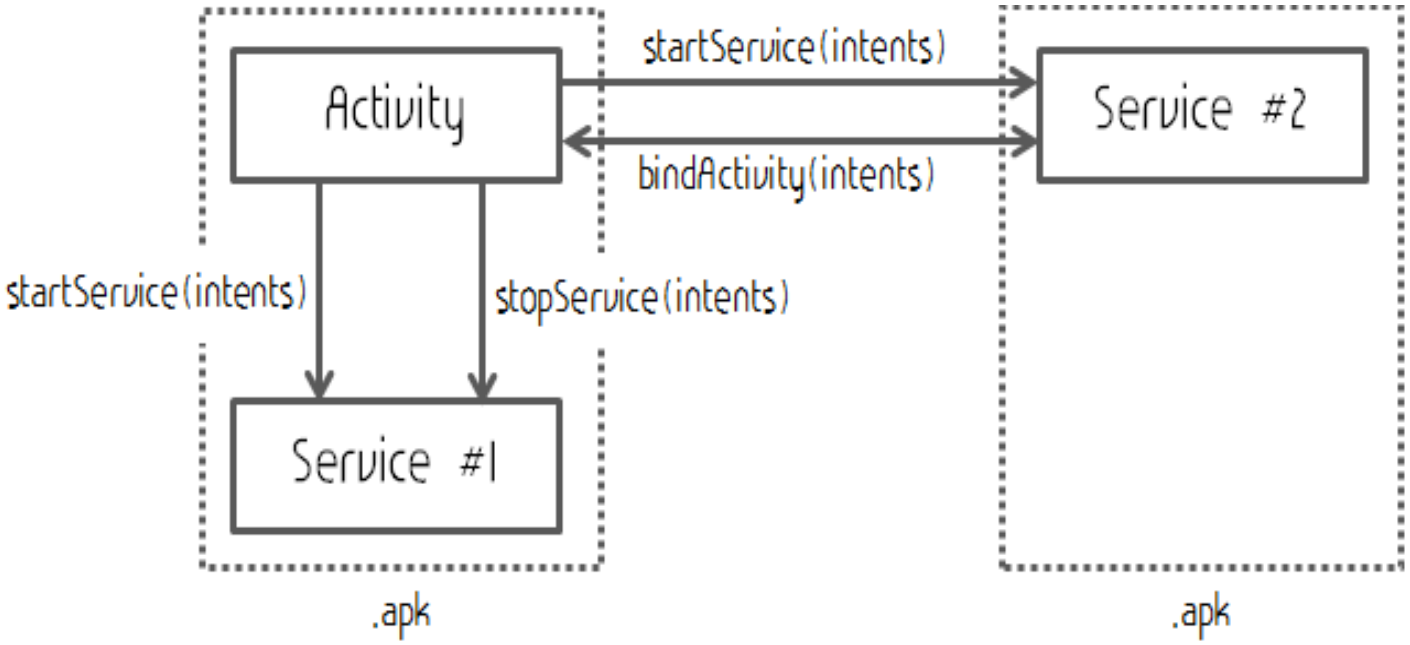
Intent

- Activity와 상호작용
  - 다른 프로그램에 소속된 Activity까지 확장 가능
  - 실행
    - Intent를 인자로 Context 클래스의 startActivity() 호출
  - 반환
    - Intent 인자와 함께 Context 클래스의 setResult() 호출



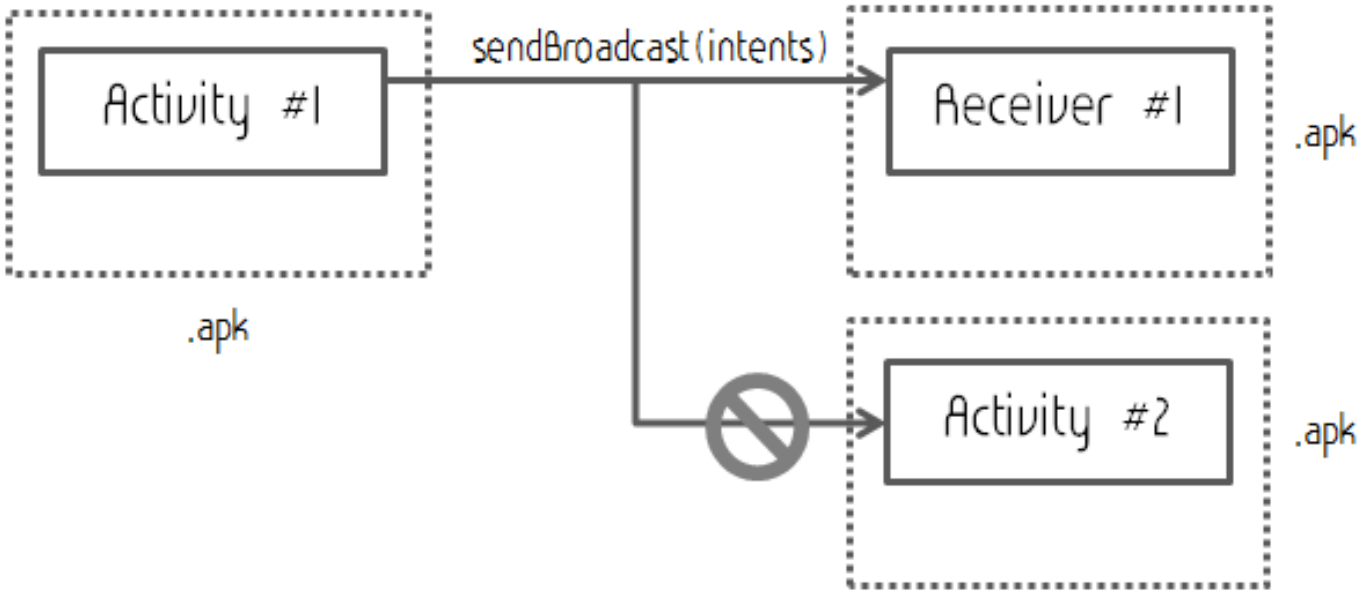
Intent

- Service와 상호작용
  - Activity와 유사
  - startService(), stopService() 호출
  - Service와의 상호작용은 AIDL 구현과 bindService() 원격 호출



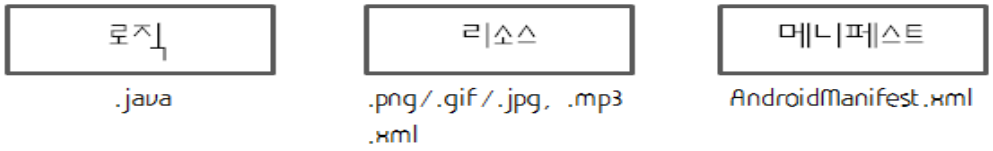
Intent

- BroadCast Receiver 통지 전달
  - 수신자 없이 Intent 인자와 Context의 sendBroadcast() 사용
  - BroadCast Receiver를 갖는 Application만 수신 가능

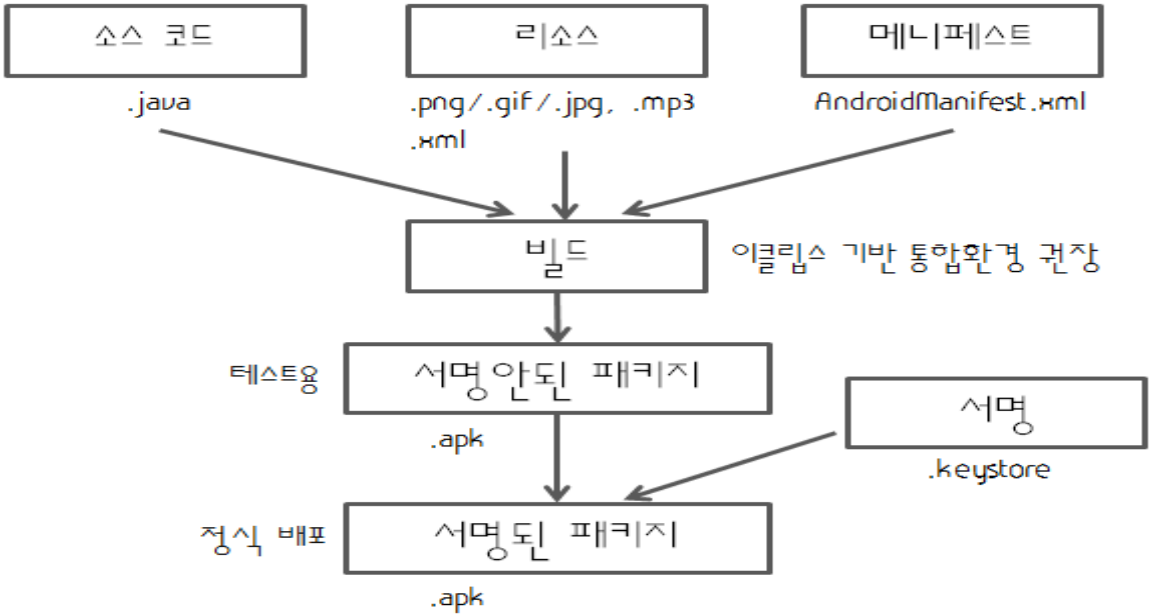


Application 작성 흐름

- Application 작성 구분

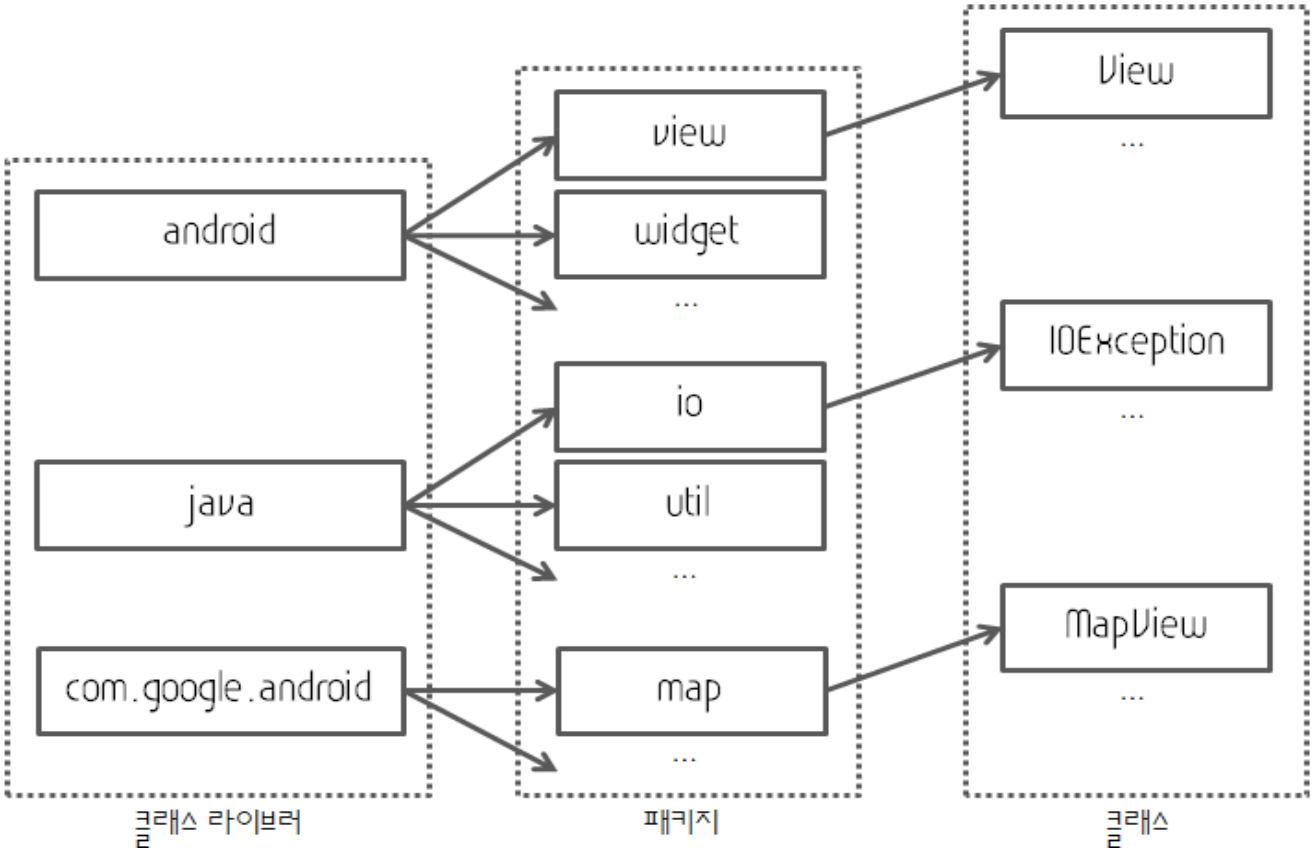


- Application 작성



Application 작성 흐름

- 클래스, 패키지, 패키지 라이브러리



Activity를 만드는 과정

- Android App은 하나 또는 그 이상의 Activity들로 구성

- Activity를 만드는 과정
  1. View들을 배치하여 Activity 화면 구성하기
  2. View들의 Event 처리를 구현하기

Activity 화면 구성하기

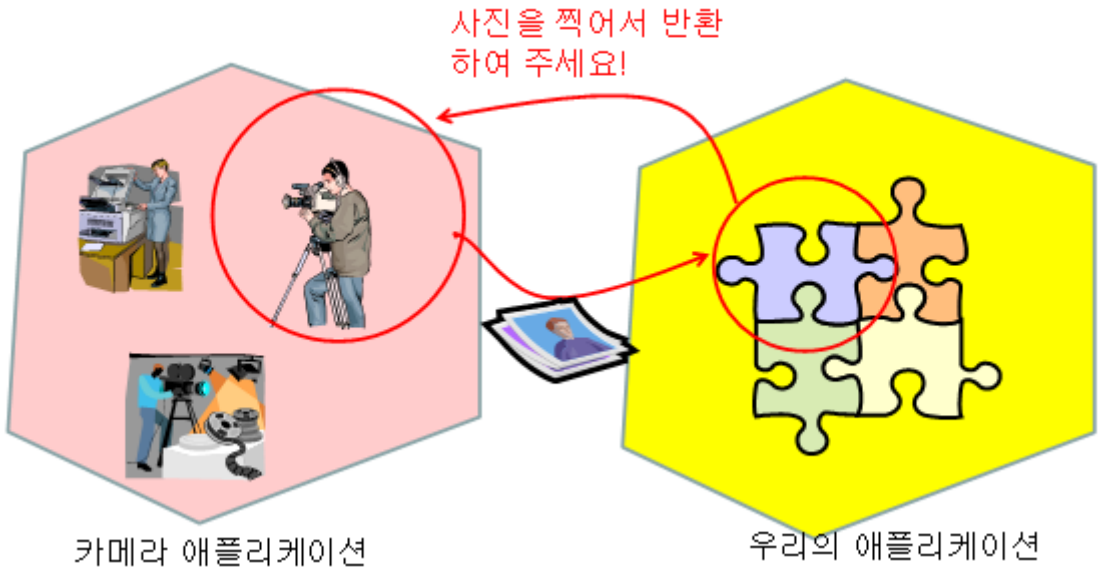
- Activity는 사용자들에게 화면으로 보여주는 UI Component
- 개발자나 UI Designer는 Activity를 통해 무엇을 어떻게 보여줘야 하는지를 고안하고 구현해야 함
- 이 때 만들어 사용되는 여러 종류의 UI 객체들이 있음
- 이들을 통틀어 우리는 View라고 부름
- 우리는 Android가 제공하는 여러 종류의 View들을 생성하고 화면에 배치
- 이를 통해 우리는 Activity의 화면 구성을 할 수 있음

View들의 Event 처리를 구현하기

- 사용자들은 App의 화면을 통해 어떤 작업을 할 것임
- 그 과정에서 App이 처리해줘야 할 작업들이 있다. 이들을 개발자들은 Code로 구현해줘야 함
- 이것이 이벤트 처리(Event Handling)임
- Android에는 다양한 종류의 Event들이 있음
- 사용자들이 Activity의 View를 사용하는 과정에서 Event가 발생하고, 이 때 무엇을 해야 하는지를 개발자는 Code로 메소드 등의 단위로 구현을 해야 함
- 이와 같이 Event 발생시 필요한 작업을 처리하는 객체를 이벤트 핸들러(Event Handler)라고 함

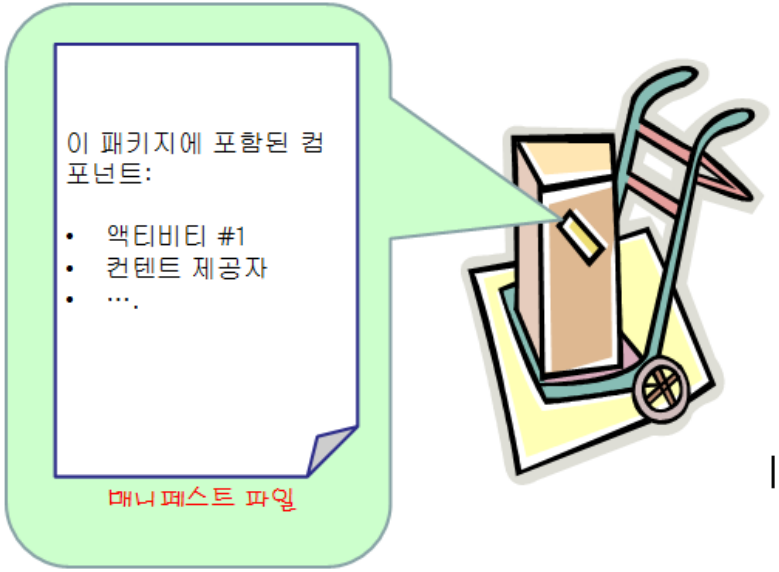
View들의 Event 처리를 구현하기

- Application에서 사용자가 사진을 촬영하도록 하고 싶은 경우



manifest 파일

- Android가 Application의 특정 component를 시작하려면 Android는 Application에 어떤 Component들이 있는지 알고 있어야 함
- 이를 위해서 Application은 자신의 Component들을 manifest 파일에 선언(적재 목록)을 해야 하며, 이 manifest 파일도 Android Package인 *.apk 파일안에 번들 되어야 함



manifest 파일

- 모든 Application은 항상 AndroidManifest.xml로 명명된 매니패스트 파일을 가지고 있어야 함
- 매니패스트 파일은 구조화된 XML 파일
- 매니패스트 파일 내용
  - Component(Activity, Service, BroadCast Receiver, Content Provider)들이 어떤 것들이 사용되고 있는 지
  - Link 시에 필요한 Library가 어떤 것들이 있는 지
  - Application의 권한(permission) 등

manifest 파일

```
<?xml version="1.0" encoding="utf-8"?>
<manifest . . . >
    <application . . . >
        <activity android:name="com.example.project.MaincActivity"
                  android:icon="@drawable/small_pic.png"
                  android:label="@string/freneticLabel"
                      . . .  >
        </activity>
        . . .
    </application>
</manifest>
```

manifest 파일

- XML을 사용
  - <activity> 요소 : 액티비티 선언
  - <service> 요소 : 서비스 선언
  - <receiver> 요소 : 방송 수신자
  - <provider> 요소 : 컨텐트 제공자

XML

- XML은 Android에서 아주 많이 사용
- SGML의 부분 집합으로 Web 상에서 구조화된 Text 형식의 문서를 전송하고 수신하며 처리가 가능하도록 만든 Markup Language