Android 개요 및 Android View & Layout &Activity

2020년 1학기

목차

- Android Summary
- View
- Widget
- View Group
- Activity
- Android Studio 개발 환경 설치

안드로이드 개요

- 안드로이드
  - 구글에서 공개한 휴대전화와 모바일 디바이스를 위한 운영체제, 미들웨어 그리고 웹, SNS, MMS 같은 어플리케이션을 포함한 소프트웨어 스택

- 안드로이드의 역사
  - 2005년 안드로이드 사를 구글에서 인수
  - 2007년 휴대전화용 안드로이드 플랫폼 발표 (무료)
    - 안드로이드 SDK, 샘플, 안드로이드 시스템 구조, 안드로이드 내장 SW 개발 구조 공개
  - 2008년 : Android SDK 1.0 발표 및 오픈소스(아파치 라이센스)로 공개
  - 2013년 Android 4.4.x (킷캣) 발표
  - 2014년 Android 5.0.x (롤리팝) 발표
  - 2015년 Android 6.0 (마시멜로) 발표
  - 2016년 Android 7.0 (누가) 발표

안드로이드 개요 (cont.)

- 안드로이드 버전

버전	코드명	특징 및 추가된 기능
1.0	-	최초 버전
1.5	Cupcake	ADV지원, 음성인식, 비디오 녹화
1.6	Donut	다양한 해상도 지원, 제스처, TTS
2.0~2.1	Eclair	HTML5, 블루투스 2.1, MS Exchange 지원
2.2	Froyo	플래시, 테더링(핫스팟), 외장 메모리 앱 설치 지원
2.3	Ginger Bread	NFC, VoIP 지원, NDK 적용 범위 확장
3.0~3.2	Honeycomb	태블릿 지원 (태블릿 전용 OS)
4.0	Icecream Sandwich	태블릿, 핸드셋의 단일 API WiFiDirect,
4.1~4.3	Jelly Bean	플래시 지원 중단, OpenGL ES 3.0 지원, 알림 및 앱위젯 기능 개선
4.4	KitKat	ART(Android Run Time) 탑재, 기본 브라우저 변경
5.0~5.1	Lollipop	ART 기본 적용, 인터페이스 대폭 개선, 64bit 지원, 보안 강화
6.0	Marshmallow	VoLTE, 지문인식 기능 및 외부 저장소 공식 지원
7.0	Nougat	VR 기능 향상, 배터리 대기시간 증가, Vulkan API로 인한 더 높은 퍼포먼스의 3D그래픽 지원

안드로이드 개요 (cont.)

- 안드로이드 버전

버전	코드명	특징 및 추가된 기능
7.1-7.1.2	Nougat	VR 기능 향상, 배터리 대기시간 증가, Vulkan API로 인한 더 높은 퍼포먼스의 3D그래픽 지원
8.0-8.1	Oreo	PIP 다중 디스플레이 지원, 부팅 속도 개선, 외부 APK설치 보안 강화
9.0	Pie	디스플레이 컷 아웃 기능(노치 기능), ART를 통한 앱 성능 및 효율성 향상,
10.0	Android 10	데스크톱 모드, 개발자 옵션 추가, Scoped Storage

안드로이드 구성 및 특징

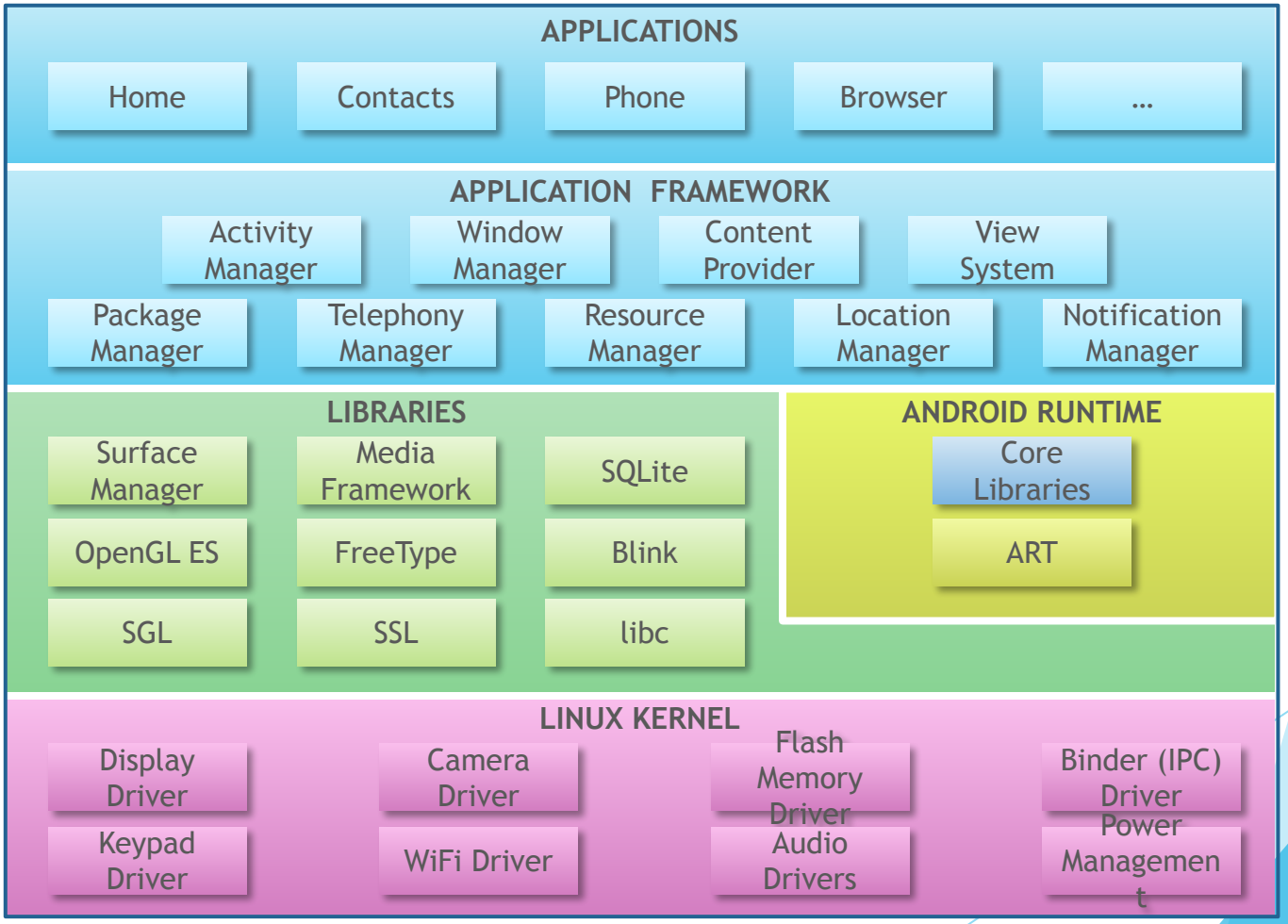
구성 및 특징	내용
개발환경	이클립스 IDE, Android Studio, IntelliJ IDEA
커널	Linux
자바 지원	대부분의 코드를 자바를 이용하여 작성 Dalvik 가상머신(4.4 이전) 혹은 ART를 이용하여 자바 코드를 구동
웹 렌더링	WebKit 엔진 기반 브라우저 이용 (v4.3 이전) v4.4 이후로 Blink엔진 기반(오픈소스) 브라우저로 전환 ※ 현재 Blink 기반 엔진이 웹 표준 지원에 있어 나음
그래픽 엔진	Open GL ES (휴대기기용 Open GL)
보안 모듈	SSL (Secure Socket Layer)

안드로이드 구성 및 특징

구성 및 특징	내용
메시징	SMS, MMS 지원
멀티 태스킹	멀티태스킹 지원
다중 언어 지원	74개 이상 언어 지원 (Android 6.0 기준)
통신	GSM/EDGE, Bluetooth, LTE, CDMA, EV-DO, UMTS, NFC, IDEN, WiMAX의 통신방식을 지원
블루투스	음성 다이얼링, 파일 전송, 키보드/마우스/조이스틱 지원
저장소	데이터 저장을 위해 SQLite를 이용
미디어 지원	WebM, H.263, H,264, AAC, HE-AAC, MPEG-4 SP, AMR, AMR-WB, MP3, MIDI, Ogg Vorbis, FLAC, JPEG, PNG, GIF, BMP, WebP 지원

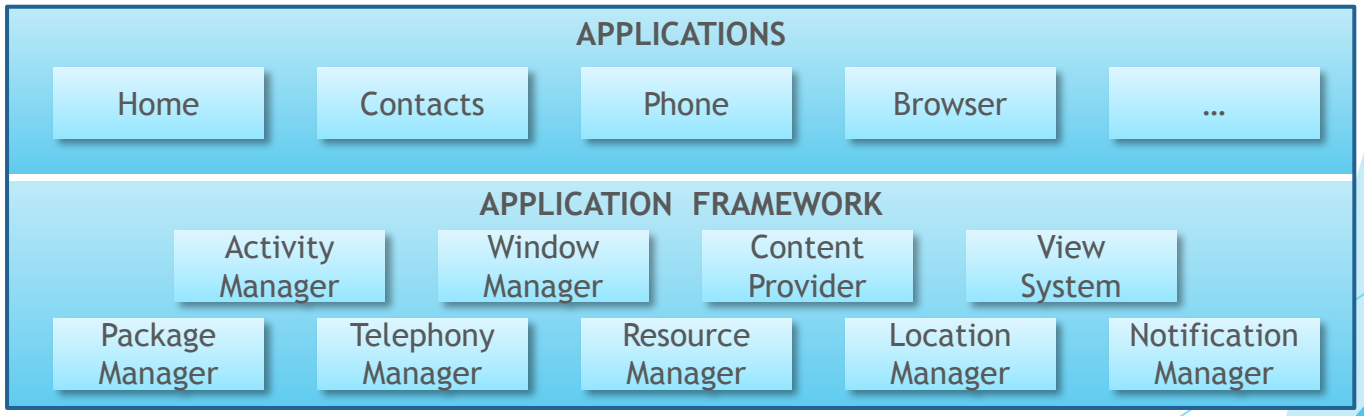
안드로이드 계층 구조

- 구조도



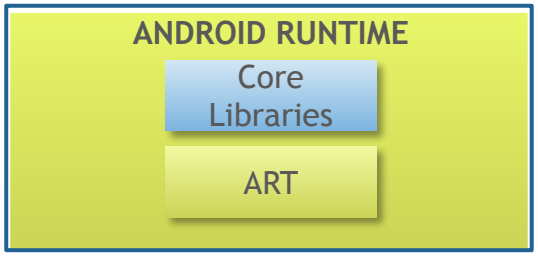
안드로이드 구조 (cont.)

- **애플리케이션 프레임워크(Framework)**
  - Java 기반의 Framework
  - JNI(Java Native Interface)를 통한 native C/C++코드로 맵핑
  - 핵심 시스템 서비스를 담당하는 Core 시스템 서비스들과 하드웨어와의 인터페이스를 담당하는 하드웨어 서비스들로 구성



안드로이드 구조 (cont.)

- **안드로이드 런타임(Runtime)**
  - Core Libraries
    - Java 언어를 위한 Core API를 포함
      (Data Structure, File Access, Utility, Graphic,
      Network Access, etc.)
  - ART
    - Java Bytecode 수행 (4.4(KitKat) 이전까지 Dalvik)
    - 가상머신을 거치지 않고
      곧바로 애플리케이션을 실행하도록 하여 속도 향상

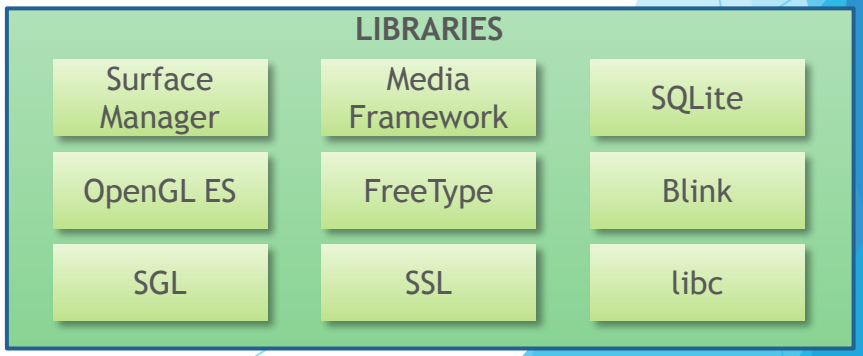


안드로이드 구조 (cont.)

- **Dalvik VM vs ART**
  - Dalvik
    - 안드로이드용 가상머신
    - JIT (Just-In-Time) 컴파일
    - 프로그램 실행시 한번에 일정부분의 바이트코드를 읽어 램에 저장
  - ART
    - 네이티브 코드 동작을 위한 런타임 라이브러리
    - AOT(Ahead-Of-Time) 컴파일
      - 애플리케이션 설치 단계에서 바이트코드가 아닌 네이티브 코드로 변환
      - ART를 이용해 네이티브 앱을 실행하여 실행 속도를 향상 (Java 가상머신을 거치지 않음)
      - JIT방식에 비해 1.5~2.0 배 정도의 설치 용량

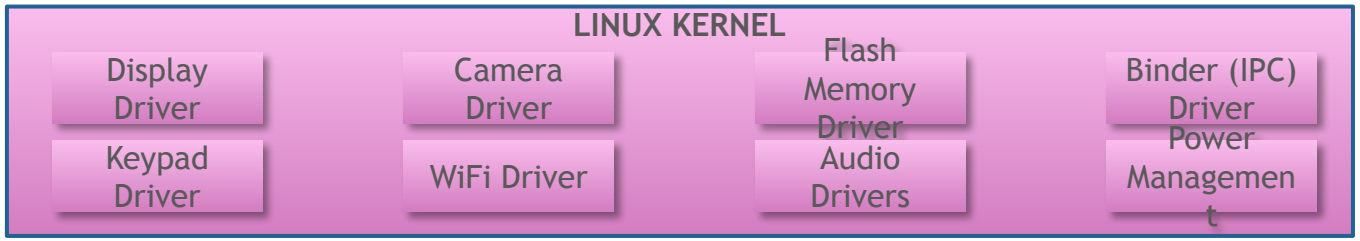
안드로이드 구조 (cont.)

- 라이브러리 (Libraries)
  - Surface Manager
    off-screen 버퍼를 관리하고, 버퍼의 내용을 화면상에 띄울 수 있도록 Window manage에 전달
  - Media Framework
    다양한 미디어 코덱들의 재생, 녹음 등을 지원하는프레임워크
  - OpenGL ES
    화면상의 3D, 2D 그래픽을 렌더링하는 라이브러리
  - SQLite
    데이터 저장을 위한 경량화된 오픈소스 기반 DB엔진
  - Blink
    웹 컨텐츠를 보여주기 위한 브라우저 엔진



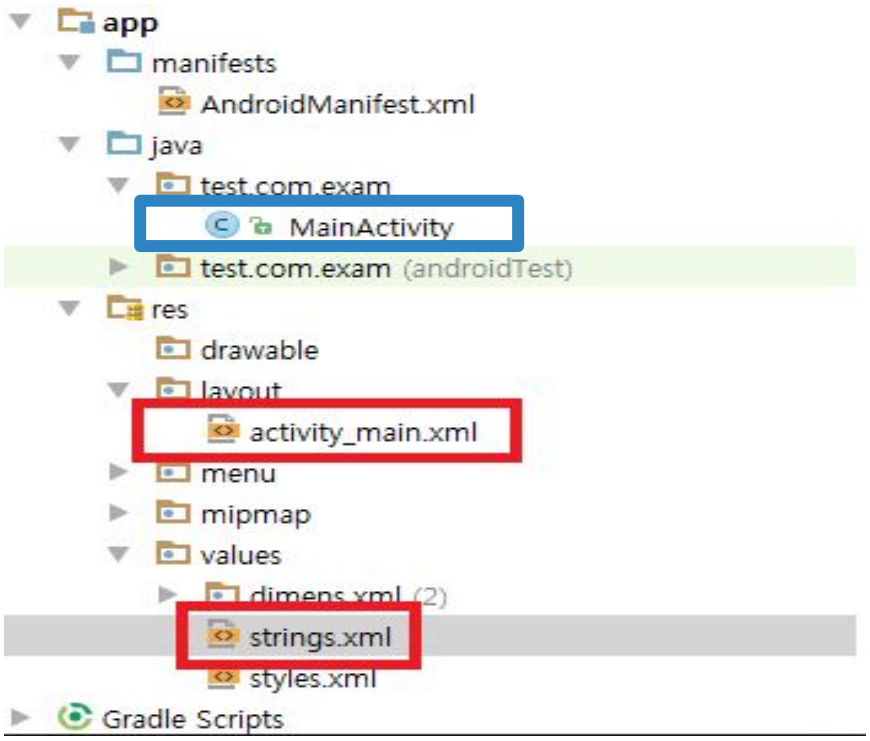
안드로이드 구조 (cont.)

- **리눅스 커널 (Kernel)**
  - 리눅스 커널을 기반으로 하고 있으나, 안드로이드는 리눅스가 아님 (표준 리눅스 유틸리티 등을 제외한, 리눅스의 커널부를 이용)
  - 메모리 및 프로세스 관리, 'Permission' 기반의 보안 모델, 검증된 드라이버 모델, 공유 라이브러리 지원, 오픈 소스 기반 등의 장점
  - 안드로이드 지원을 위해, 리눅스 커널 확장 패치를 포함 (Alarm, Ashmem, Binder, Power Management, Low Memory Killer, Kernel Debugger, Logger 등 확장)



안드로이드 구조 (cont.)

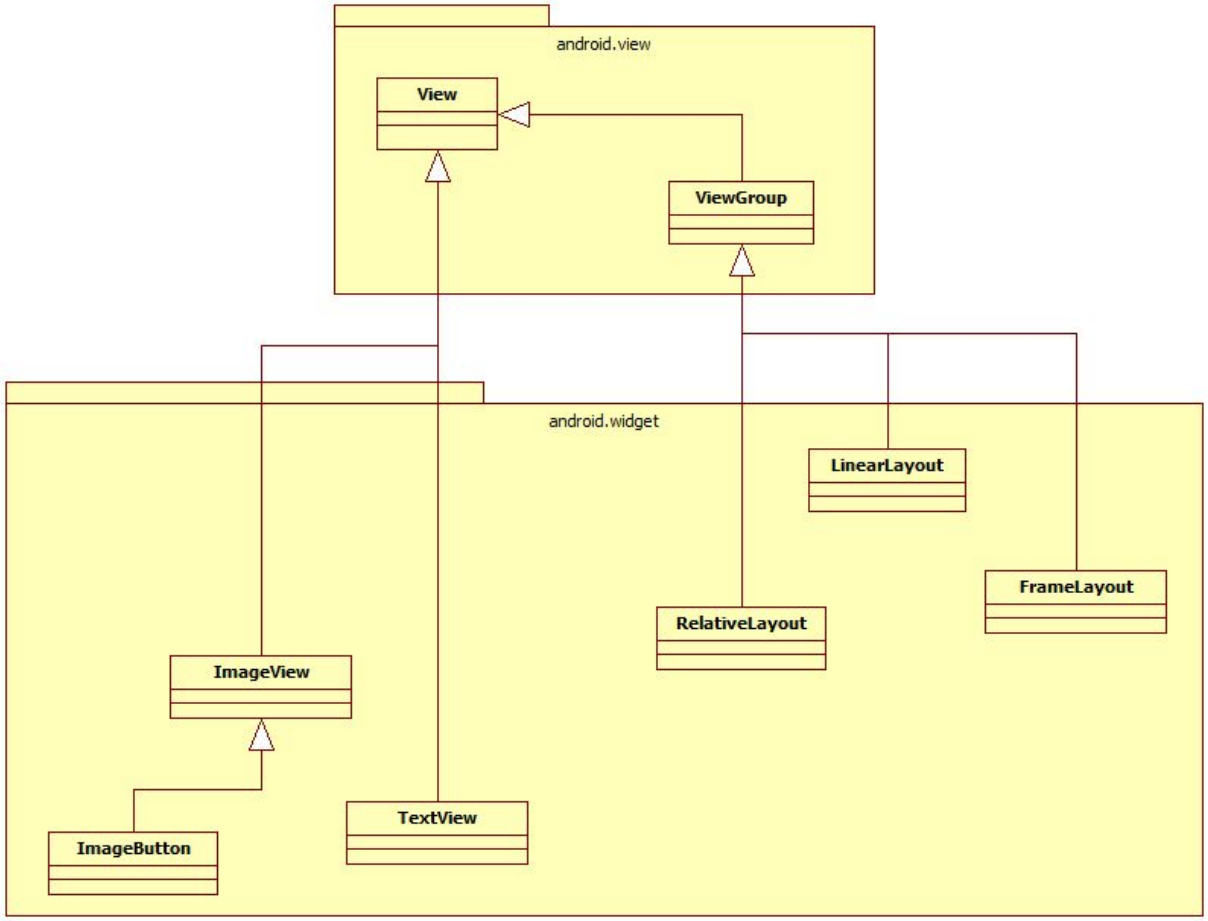
- 안드로이드는 크게 2종류의 코드로 나뉨
- 기능(이벤트)을 담당하는 Java 코드
- 화면 UI를 담당하는 Xml 코드



View, Widget, View Group

- View : UI를 구성하는 기본적인 블럭
- Widget : 애플리케이션 화면에 사용되는 UI 요소로 View를 상속 받는다.
  (TextView, ImageView, Button etc.)
- View Group : 다른 뷰를 내부에 포함 할 수 있는 특수한 뷰
  (layout, etc.)
- App Widget: 애플리케이션 화면의 UI가 아닌,
  안드로이드 기본 화면(런처 화면)에 위치하여 동작하는 형태의 애플리케이션.
  일반적으로 위젯이라 부르지만 실제 명칭은 앱 위젯으로, 위의 위젯과 혼동하기 쉬우니 주의.

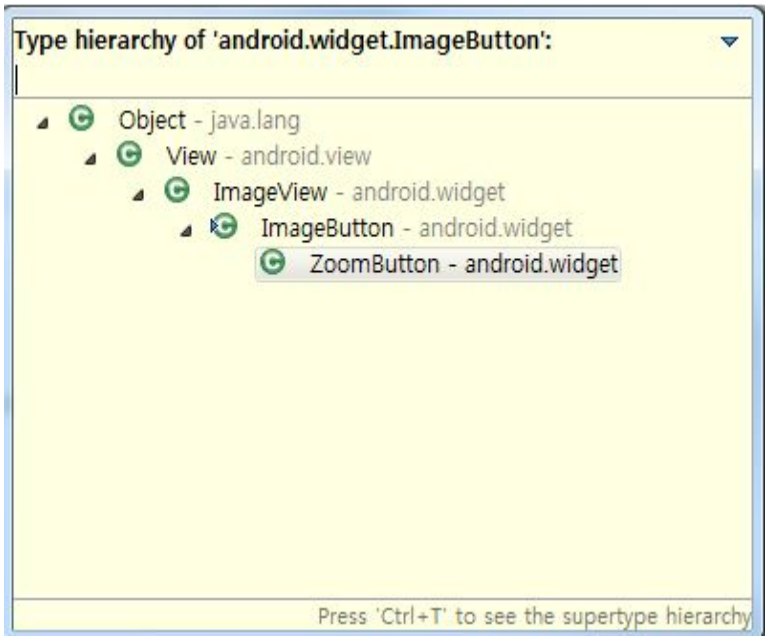
View, Widget, View Group

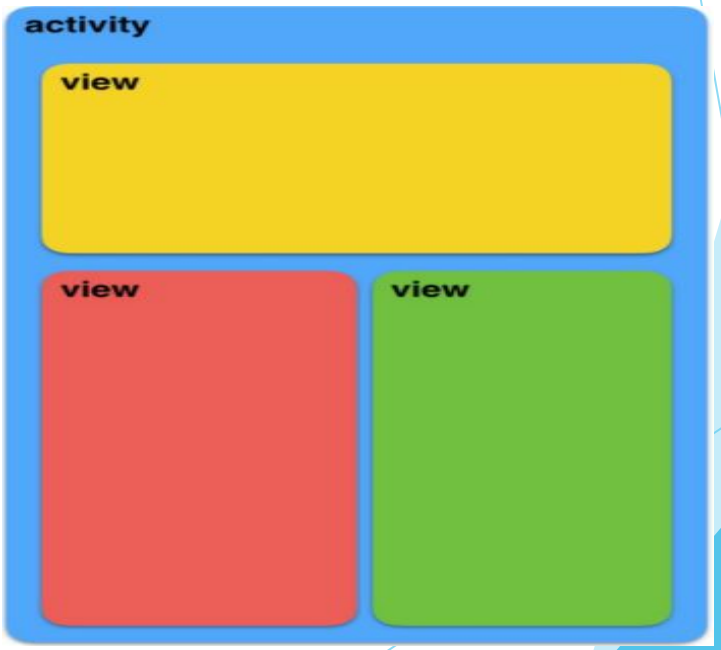


View

- UI를 구성하는 기본적인 블럭

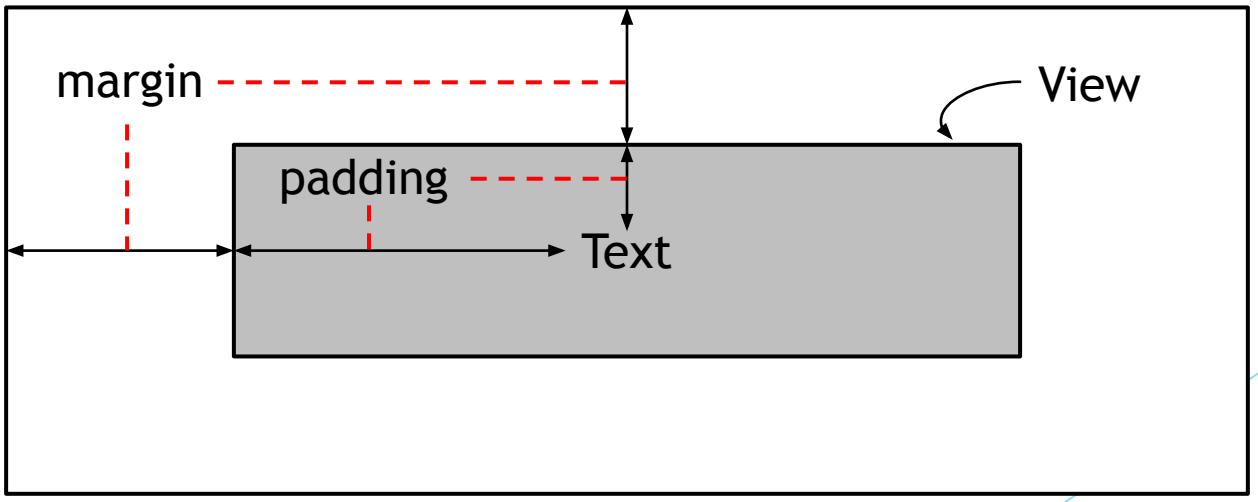
- 애플리케이션 화면에서 사각형의 영역을 차지하며 자신의 모양을 그리고, 일부 위젯은 입력을 받아들임





View – Margin / Padding

- UI 구성 과정에서 View간의 여백을 지정하기 위해 margin과 padding을 사용
- margin : View 외부 여백
  (공간이 모자랄 경우 View의 크기를 줄이고 여백을 생성)
- padding: View 내부 여백
  (padding 값이 View의 크기보다 클 경우 컨텐츠가 사라짐)



px / sp / dp …

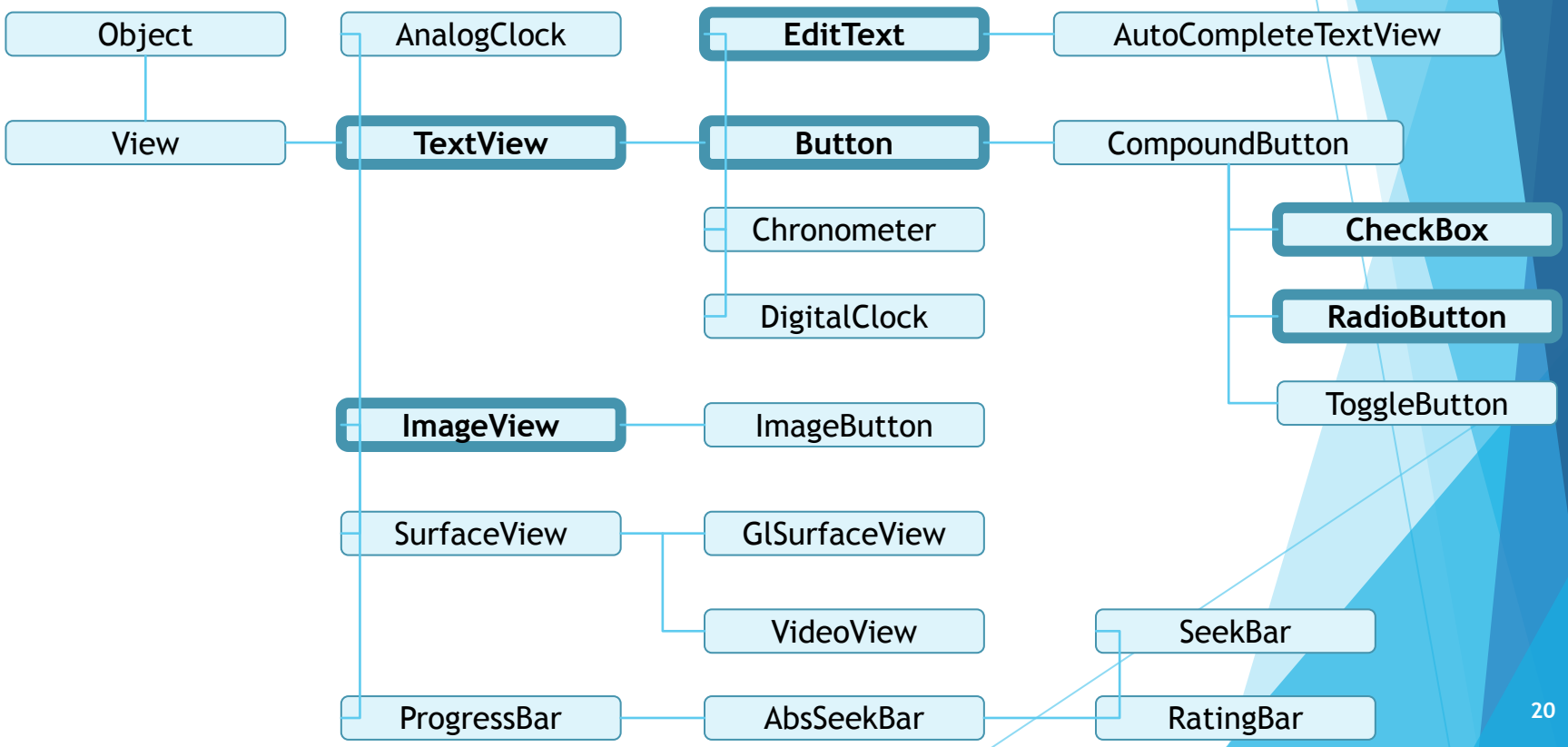
- Android에서 View의 크기 등을 설정하는 크기의 단위로 px, sp, dp, in, mm, pt가 사용 가능

단위	의미	특징
px	Pixels	실제 화면의 픽셀 단위
dp	Density independent pixels	화면 밀도에 기반한 단위 1dp = 160dpi에서의 1px
sp	Scale independent pixels	dp와 동일한 크기지만 시스템 폰트 크기에 따라 사이즈 변경
in	Inches	실제 길이(인치) 단위
mm	Millimeters	실제 길이(밀리미터) 단위
Pt	Points	실제 길이(1/72인치) 단위

※ 해상도의 영향을 최소화하기 위해
텍스트의 크기를 지정할때는 sp, 그 외에는 dp 사용을 권장

Widget

- Widget
  - 애플리케이션 화면에 사용되는 UI 요소로 View를 상속받음 (TextView, ImageView, Button, etc.)



Widget – TextView

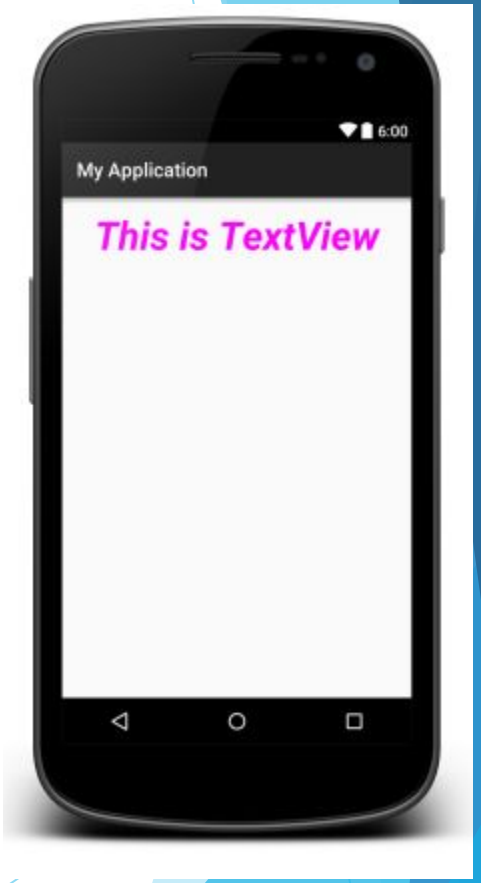
- Text View ?
  - 화면에 텍스트를 출력하는 위젯.
  - 사용자의 입력을 받아들이지는 않는다.
- 주 사용 속성

속성	특징	값
text	출력 문자열을 지정	“text” 형식
textColor	문자열 색상 지정	#RRGGBB형식
textSize	문자열 크기 지정	sp, dp, px 등의 단위
textStyle	문자열 형태 지정	Normal, Bold, italic

Widget – TextView

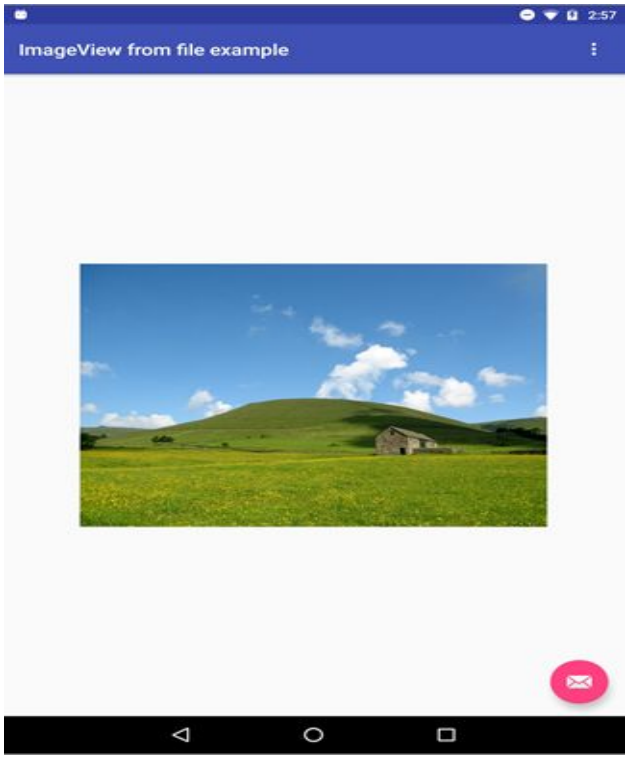
- App > src > main > res > layout > activity_main.xml 에서 다음을 작성

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:tools="http://schemas.android.com/tools"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:paddingLeft="16dp"
                android:paddingRight="16dp"
                android:paddingTop="16dp"
                android:paddingBottom="16dp"
                tools:context=".MainActivity">

    <TextView
            android:text="This is TextView"
            android:textSize="40dp"
            android:gravity="center"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="#FF00FF"
            android:textStyle="bold|italic"/>

</RelativeLayout>
```



Widget – ImageView

- ImageView ?
  - 화면에 이미지를 보여주기 위한 위젯
  - 이미지의 크기, 색조 등을 컨트롤 가능

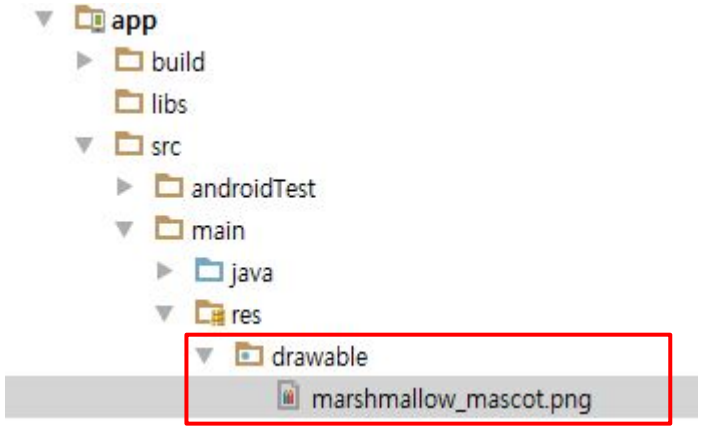


Widget – ImageView 속성

속성	용도	용법
Src	표기할 이미지 지정	@drawable/이미지명
maxWidth maxHeight	이미지 최대크기 지정	수치 (dp, sp, px, etc.)
adjustViewBounds	이미지가 레이아웃보다 클 때, 비율 유지 여부	true / false
cropToPadding	이미지가 레이아웃보다 클 때, 잘라낼지 여부	true / false
tint	이미지 색조 처리 (이미지에 색이 덮혀 출력)	#aarrggbb
ScaleType	이미지 뷰의 크기에 맞게 조작하거나 이동	MATRIX, CENTER, FIT_XY, etc.

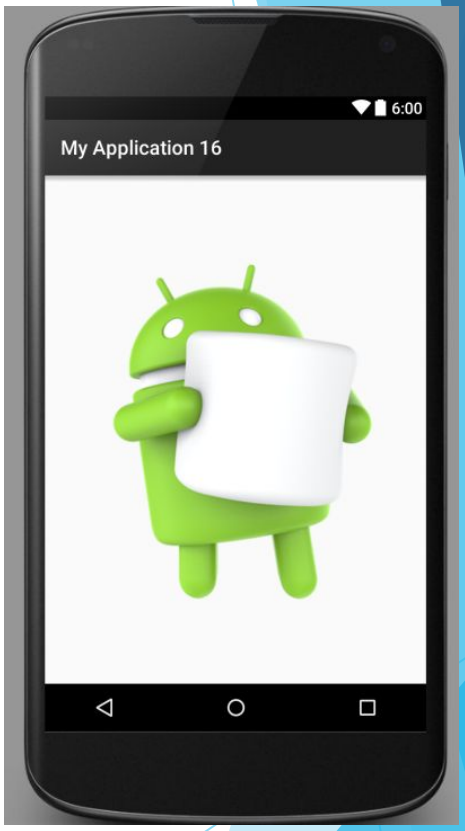
Widget – ImageView

- 이미지 등록 방법
  - app\src\main**res**\drawable\ 경로에 이미지 복사



Widget – ImageView 예제

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:tools="http://schemas.android.com/tools"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingLeft="16dp"
            android:paddingRight="16dp"
            android:paddingTop="16dp"
            android:paddingBottom="16dp"
            tools:context=".MainActivity">

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:src="@drawable/marshmallow_mascot"/>
</RelativeLayout>
```



Widget – Button

- Button ?
  - 사용자의 push입력을 전달받는 위젯
  - 함수를 연결함으로써 다양한 기능을 수행
  - OnClickListener지정된 라이브러리에 입력에 따른 동작 정의

Widget – Button(OnClickListener)

- 버튼의 기본 함수 형태

```
<Button
    android:id="@+id/button"
```

.xml파일에서 id 지정

```
Button btn = (Button)findViewById(R.id.button);
```

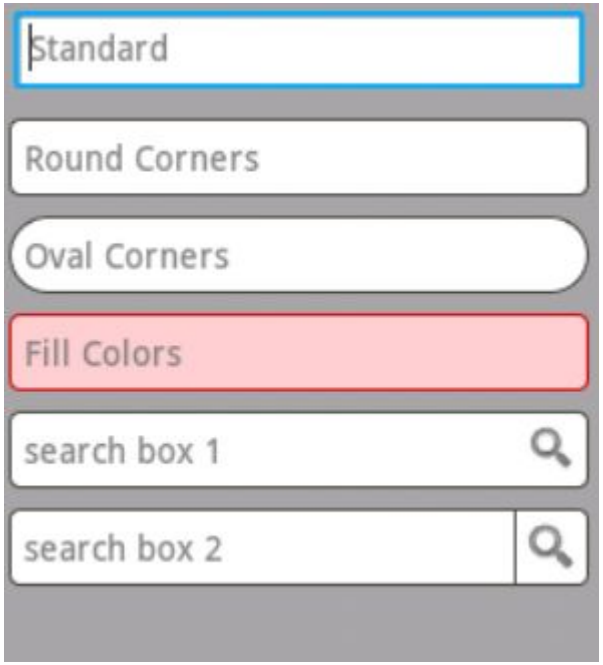
버튼 클래스 객체 선언 및 id를 이용해 UI와 연결

```
btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //버튼 클릭시 수행 할 코드
    }
});//end setOnClickListener
```

버튼에 클릭 입력을 받아들이는 OnClickListener 객체 지정
OnClickListener내부의 Abstract Method인 onClick을 Overriding하여 클릭시 수행 할 코드를 작성

Widget – EditText

- EditText ?
  - 문자열을 입력하기 위한 위젯
  - setText(), getText() 함수를 이용해 문자열을 이용



Widget – Button & EditText 예제

activity_main.xml

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:hint="여기에 입력..."
        android:id = "@+id/editText"/>
    <Button
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:text="입력"
        android:id = "@+id/button"/>
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:textSize="30dp"
        android:text="Text Here"
        android:id="@+id/textView"/>
</LinearLayout>
```

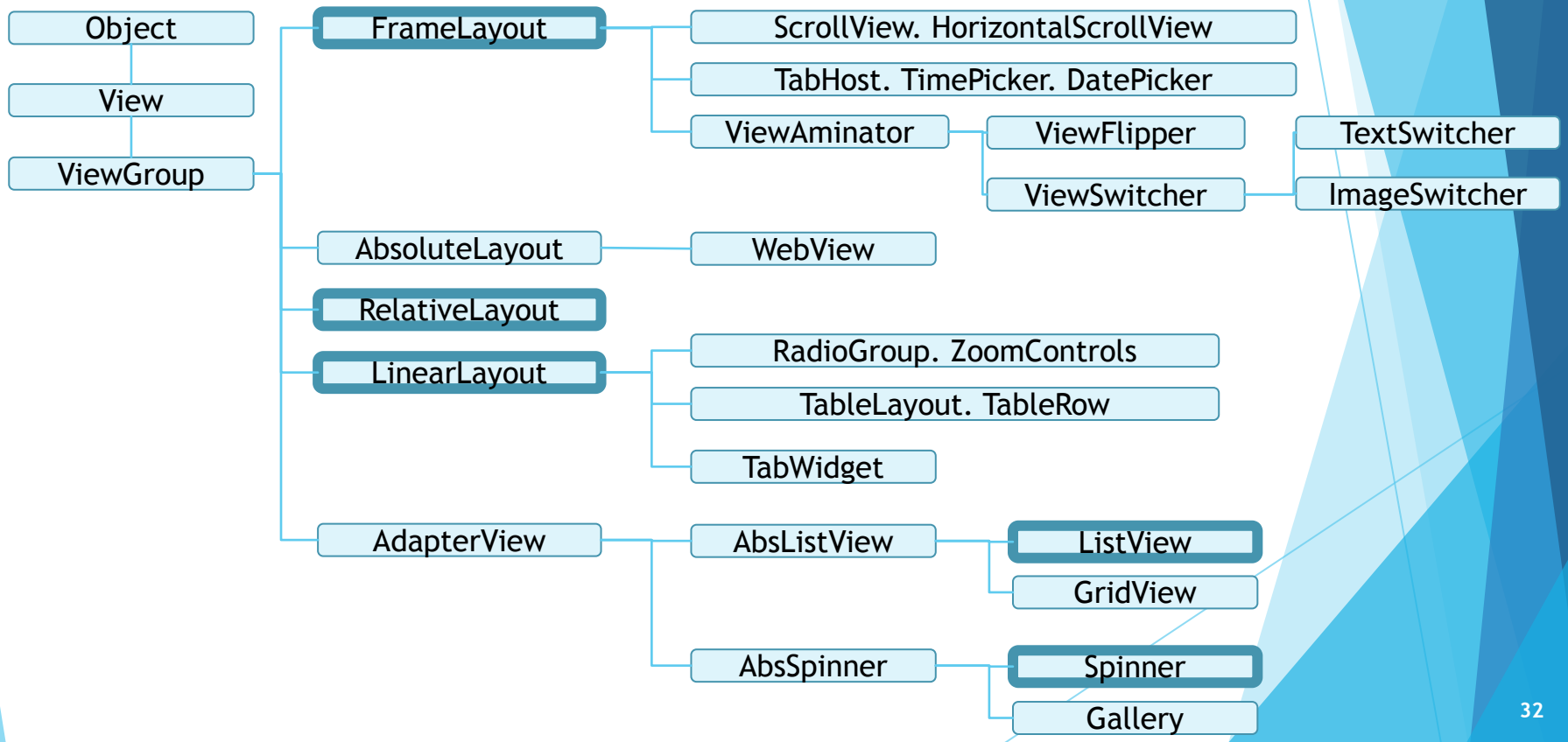
MainActivity.java

```java
public class MainActivity extends AppCompatActivity {

    public EditText edit;
    public Button btn;
    public TextView text;

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

        // UI의 각각의 위젯과 클래스 객체를 연결
        edit = (EditText)findViewById(R.id.editText);
        btn = (Button)findViewById(R.id.button);
        text = (TextView)findViewById(R.id.textView);

        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //버튼 클릭시 수행 할 코드
                String str = edit.getText().toString();
                text.setText(str);
            }
        });//end setOnClickListener
    }   //end OnCreate

}
```

Widget – Button & EditText 예제

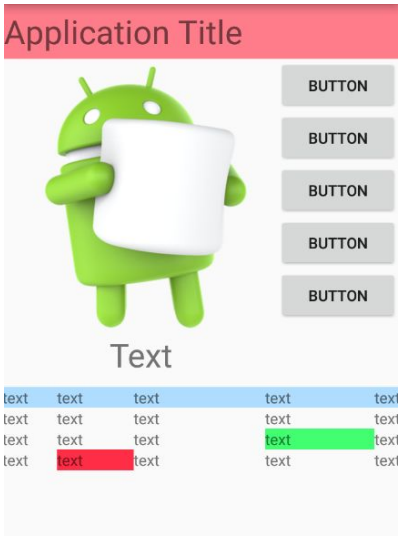
ViewGroup

- View Group
  - 다른 뷰를 내부에 포함 할 수 있는 특수한 뷰로 View를 상속받음



ViewGroup - Layout

- Layout ?
  - 내부에 다른 widget 혹은 Layout을 담는 ViewGroup
  - Layout은 눈에 보이지 않지만, widget의 배치방식을 결정



Layout과 Button, ImageView, TextView만을 이용하여 만들어 본 화면 구성

Layout – LinearLayout

- LinearLayout ?
  - 화면 구성에 가장 많이 사용되는 Layout
  - 자식 view를 일렬로 배치하는 Layout
  - android:orientation 속성을 통해 가로배치(horizon)/세로배치(vertical) 설정

Layout – LinearLayout 예제

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              xmlns:tools="http://schemas.android.com/tools"
              android:layout_width="fill_parent"
              android:layout_height="fill_parent"
              android:orientation="vertical"
              tools:context=".MainActivity">

    <Button
            android:layout_width="fill_parent"
            android:layout_height="60dp"
            android:text="Linear"
            android:textSize="30dp"/>
    <Button
            android:layout_width="fill_parent"
            android:layout_height="60dp"
            android:text="Layout"
            android:textSize="30dp"/>
    <Button
            android:layout_width="fill_parent"
            android:layout_height="60dp"
            android:text="Sample"
            android:textSize="30dp"/>

</LinearLayout>
```

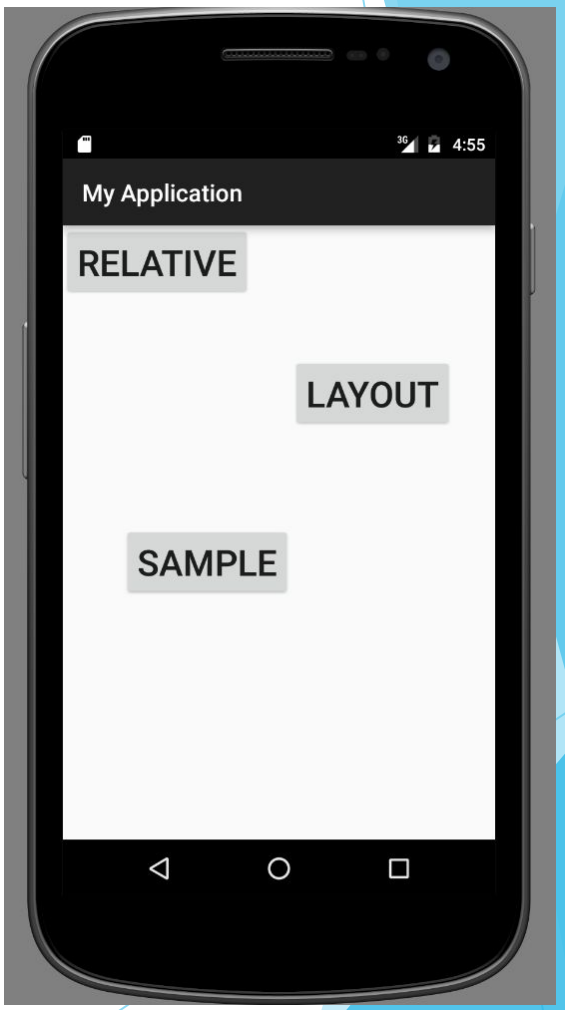
Layout – RelativeLayout

- RelativeLayout ?
  - 자식 view간의 상대적인 위치를 지정하여 배치하는 Layout
  - Layout이 아닌 자식 view의 속성을 통해 각각의 위치를 지정

Layout – RelativeLayout 예제

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:tools="http://schemas.android.com/tools"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            tools:context=".MainActivity">
    <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Relative"
            android:textSize="30dp"
            android:id="@+id/button"/>
    <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Layout"
            android:textSize="30dp"
            android:id="@+id/button2"
            android:layout_below="@+id/button"
            android:layout_toRightOf="@+id/button3"
            android:layout_toEndOf="@+id/button3"
            android:layout_marginTop="49dp"/>
    <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Sample"
            android:textSize="30dp"
            android:id="@+id/button3"
            android:layout_below="@+id/button2"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginLeft="50dp"
            android:layout_marginStart="30dp"
            android:layout_marginTop="80dp"/>
</RelativeLayout>
```

id

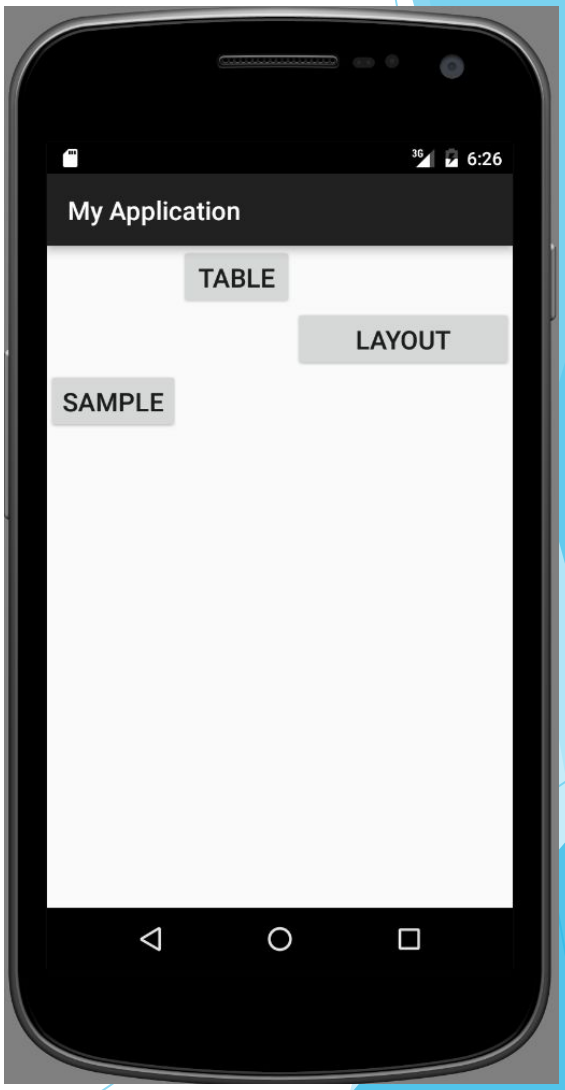


Layout – TableLayout

- TableLayout ?
  - 화면을 표처럼 열과 행으로 구분하여 구성하는 Layout
  - android:stretchColumns속성으로 열 갯수 지정
  - <TableRow> 태그를 이용하여 행 구분
  - 자식 view에 android:layout_column 속성을 사용하여 열 지정

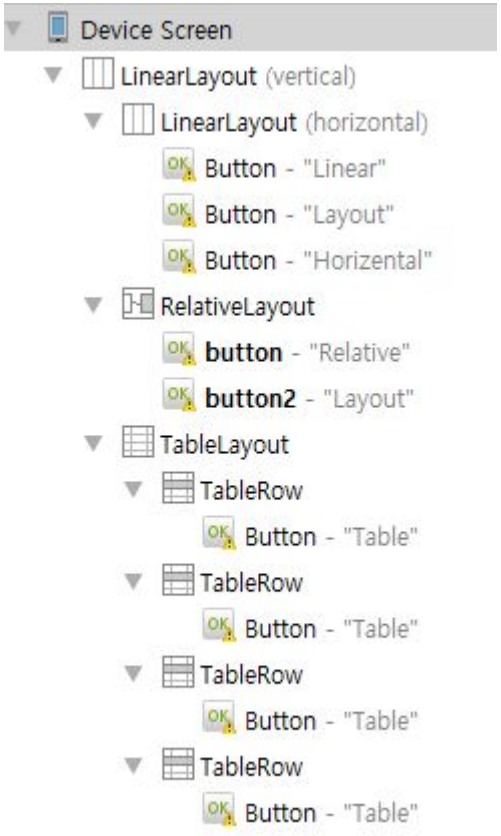
Layout - TableLayout 예제

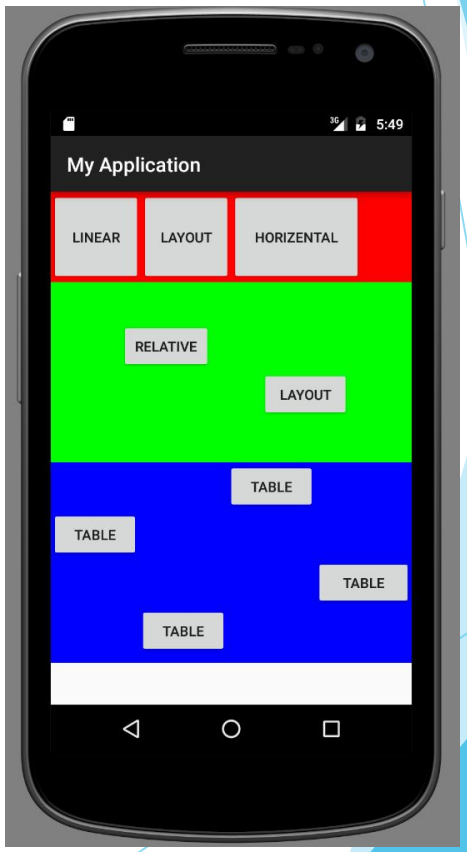
```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="2"
    tools:context=".MainActivity">
    <TableRow>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Table"
            android:textSize="20dp"
            android:id="@+id/button"
            android:layout_column="1"/>
    </TableRow>
    <TableRow>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="layout"
            android:textSize="20dp"
            android:id="@+id/button2"
            android:layout_column="2"/>
    </TableRow>
    <TableRow>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="sample"
            android:textSize="20dp"
            android:id="@+id/butto3"
            android:layout_column="0"/>
    </TableRow>
</TableLayout>
```



Layout in Layout

- 레이아웃 내부에 다른 레이아웃을 넣는 것으로 효율적인 화면 구성이 가능





Layout – gravity와 layout_gravity

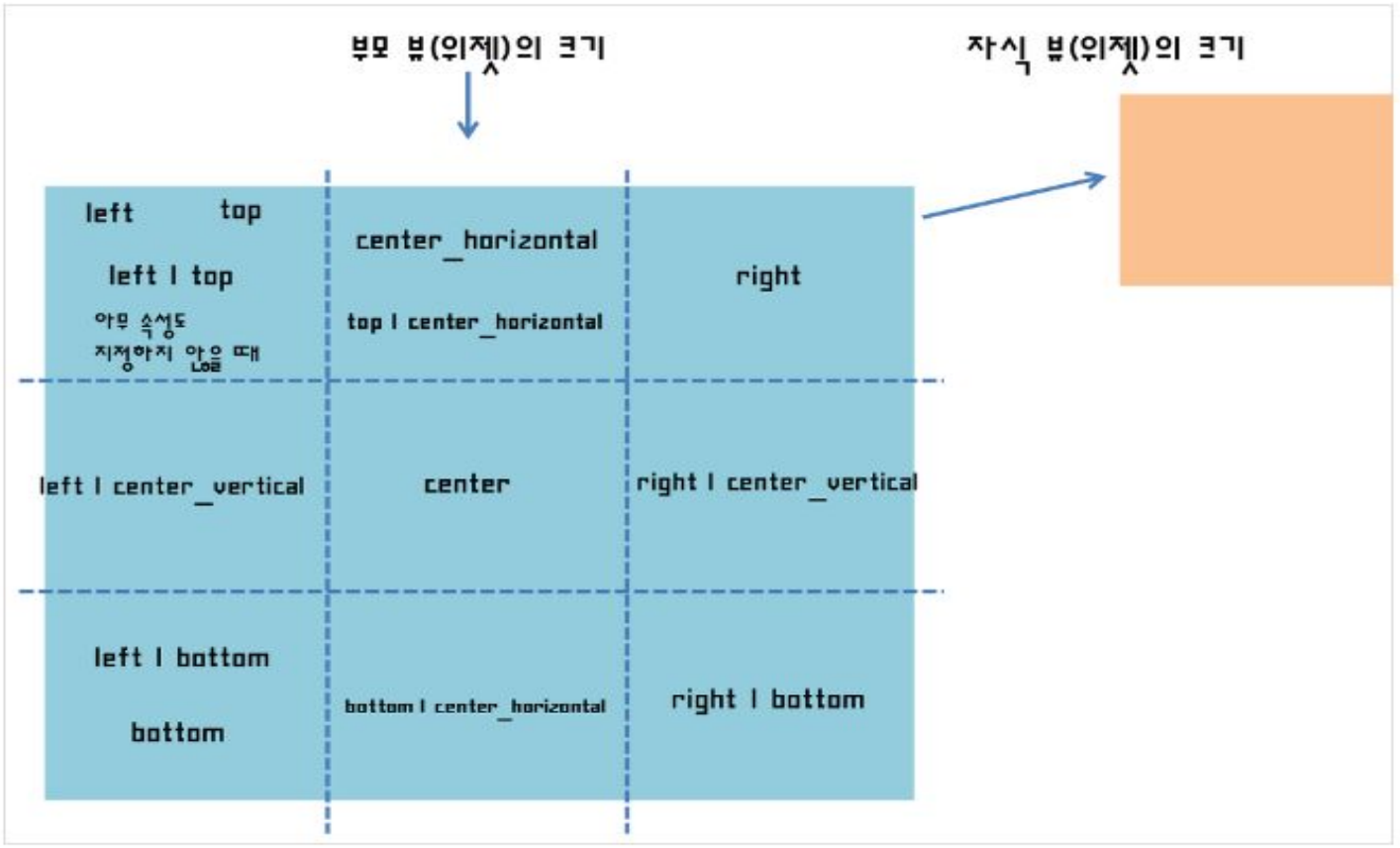
- **Android:gravity 속성**
  - 해당 속성을 지닌 view 내부 컨텐츠(자식 view포함)의 위치를 지정하는 속성
    (ex : <Button **android:gravity="center"** 속성 추가시
    버튼 내부 컨텐츠인 Text의 위치가 버튼의 가운데로 지정됨)

- **Android:layout_gravitiy 속성**
  - 해당 속성을 지닌 view의 부모 ViewGroup 내부 위치를 지정
    (ex: <LinearLayout>
    <Button **android:layout_gravity = "center"**/> 속성 추가시
    부모 ViewGroup인 LinearLayout의 중앙에 버튼의 위치가 지정됨)

※ 속성은 충돌하지 않는 범위 내에서 "center | right" 와 같이 여러개 사용 가능

Layout - gravity와 layout_gravity

- **android:gravity & android:layout_gravity** 에 설정할 수 있는 값
  - Left, right, top, bottom
  - center, center_vertical, center_horizontal



Layout – layout_weight

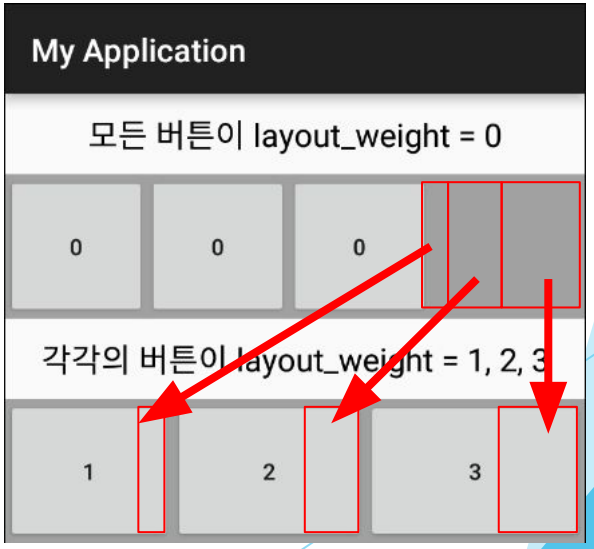
- android:layout_weight
  - 남는 공간을 어떻게 배분 할 지를 지정하는 속성

    layout_weight="0" -> 해당 view는 남는 공간을 사용하지 않음

    layout_weight="1" -> 해당 view는 남는 공간을 1의 비율로 사용

    남는 공간을 사용할 수 있는 모든 view에 대하여
    layout_weight / 모든 view의 layout_weight의 합 의 비율로
    남는 공간을 분배

  - 일반적으로 크기를 0dp를 주고,
    공간을 비율에 맞춰
    나눠 가지도록 하는 방법을 많이 사용



Android - Activity

- 어플리케이션 내의 **하나의 스크린**
- **UI컴포넌트를 화면에 표시하고**
  시스템이나 사용자의 반응을 처리
- 어플리케이션이 **UI**를 가진다면
  하나 혹은 그 이상의 **Activity**와 매칭되며
  기존 **Activity**는 같은 기능을 하는 새로운 것으로 대체 가능

Activity – Activity Stack

- 어플리케이션에서 호출되는 Activity는 Activity Stack에서 관리
  - Stack 구조를 채용하여 오래된 Activity일 수록 하단에 배치

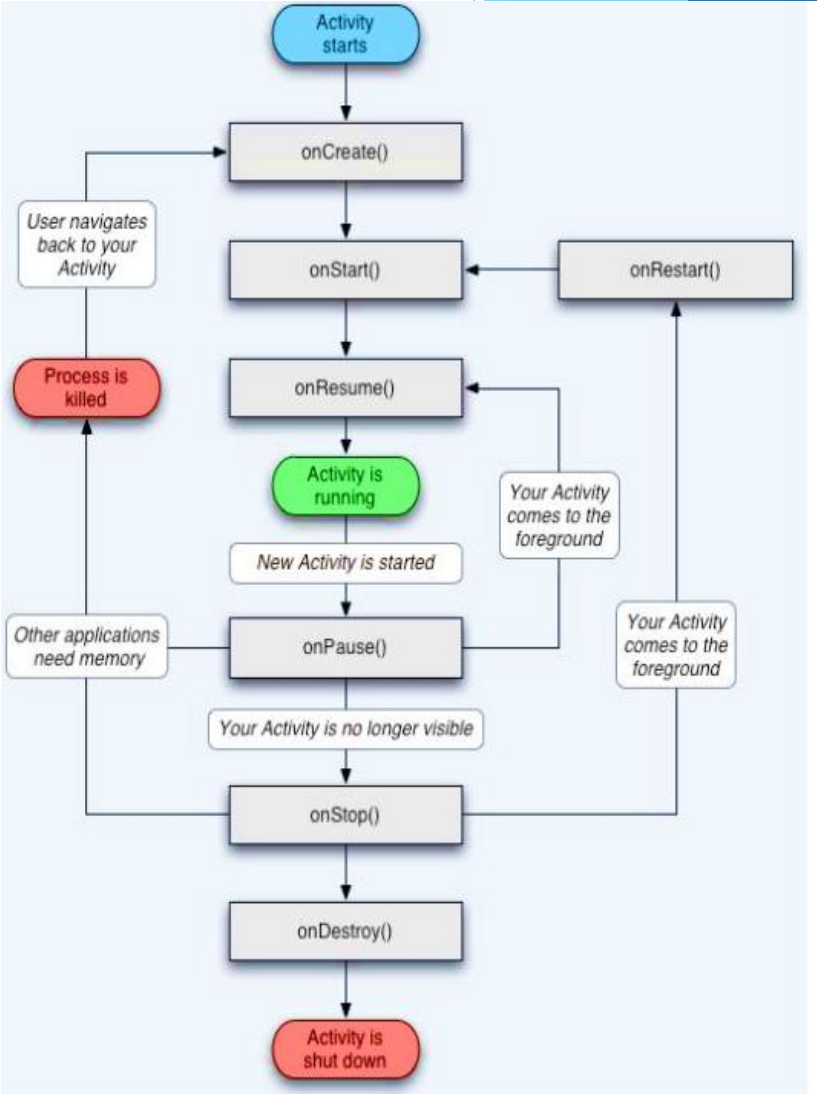
- 모바일 기기 특성에 맞춘 구조
  - PC에 비해 사양이 낮은 모바일 기기 특성상 메모리 부족 현상이 발생하기 쉬움
  - 메모리 부족 현상 방지를 위해 Stack 하단부의 Activity를 제거하여 효율적으로 메모리를 관리
  - Flag 속성을 이용하여 보다 효율적인 스택관리 intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

Activity - Lifecycle

OnCreate()
Activity가 생성될 때 처음으로 호출되는 함수
리소스 초기화 작업 수행

OnStart()
Activity가 사용자에게 보여지기 직전에 호출

OnResume()
Activity가 화면에 보여지고
사용자와 상호작용을 하기 직전에 호출
이때, **Activity Stack**의 가장 상위에 위치

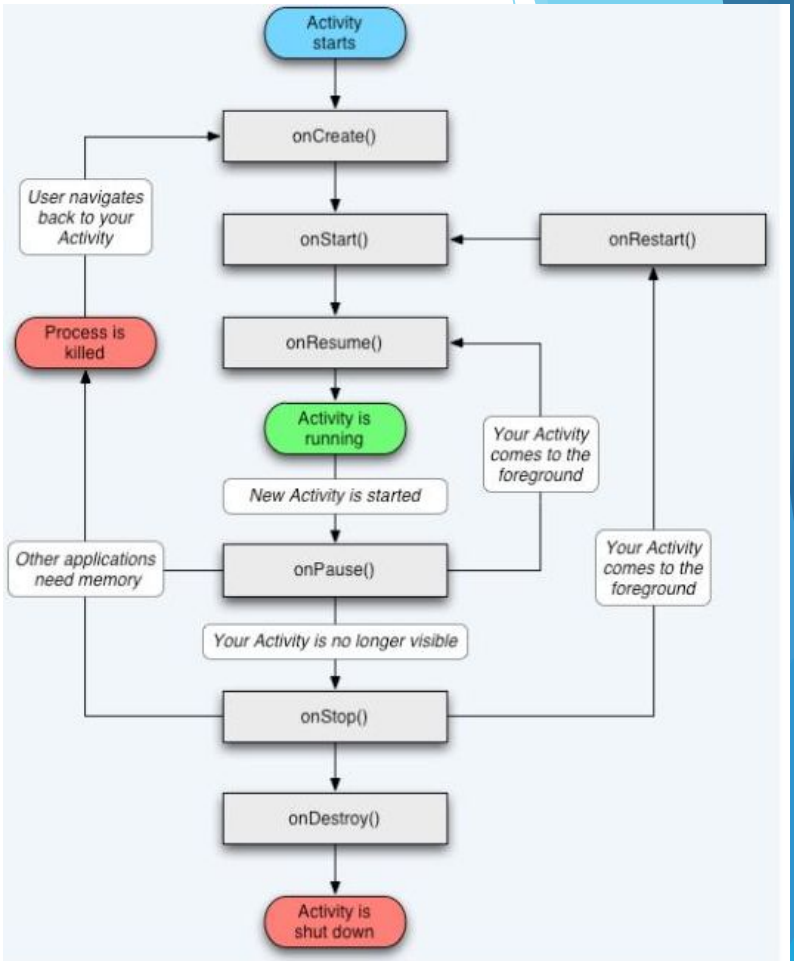


Activity – Lifecycle

OnPause()
다른 **Activity**가 실행되기 전에
데이터 저장, 애니메이션 중지 및
CPU를 이용하는 작업을 중단

OnStop()
Activity가 사용자에게 보여지지 않음
더이상 **Activity Stack**의 최상위가 아님
'Home'키 입력시 여기까지 호출

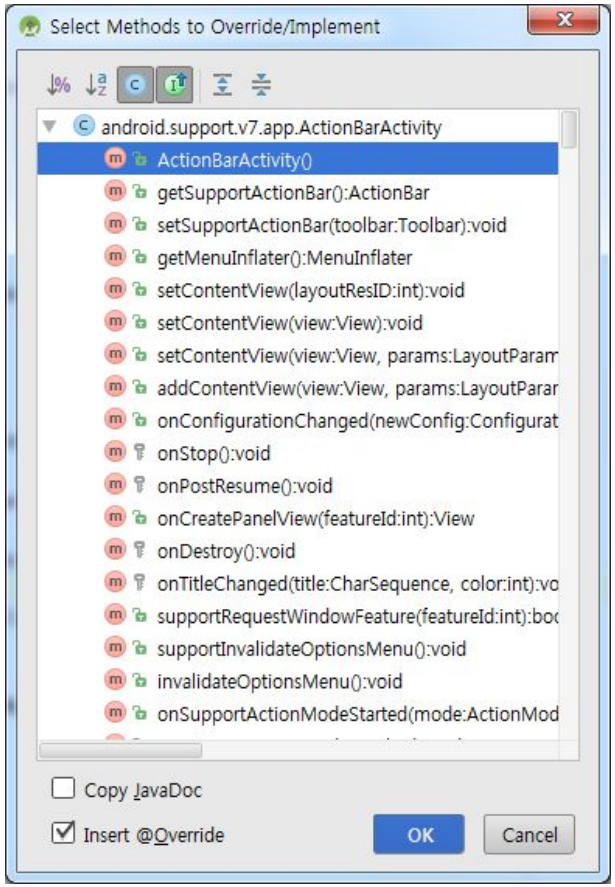
OnDestroy()
Activity를 완전종료하기 위해 호출
시스템 내에 **Activity**가 존재하지 않음



Android - Lifecycle

※ Lifecycle관련 함수를 사용하는 방법

*.java 프로젝트 코드에서 마우스 우클릭 -> Generate -> Override Methods... 선택 -> 원하는 항목 선택




```
public class MainActivity extends ActionBarActivity {

    public Button btn;

    @Override
    protected void onStop() {
        super.onStop();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
    }

    @Override
    protected void onTitleChanged(CharSequence title, int color) {
        super.onTitleChanged(title, color);
    }

    @Override
    public void supportInvalidateOptionsMenu() {
        super.supportInvalidateOptionsMenu();
    }

    @Override
    public void onSupportActionModeStarted(ActionMode mode) {
        super.onSupportActionModeStarted(mode);
    }
```

Android - Activity 예제

activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:textSize="25dp"
        android:text="This is Main Activity"
        android:gravity="center"/>
    <Button
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:text="Call Sub_Activity"
        android:id="@+id/callSubActivityBtn"/>
</LinearLayout>
```

MainActivity.java

```
package com.example.myapplication.app;

import ...

public class MainActivity extends AppCompatActivity {

    public Button btn;

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

        btn = (Button)findViewById(R.id.callSubActivityBtn);

        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, SubActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

SubActivity라는 이름의 액티비티로 전환
(Intent에 대한 내용은 다음주)

Android - Activity 예제

activity_sub.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              xmlns:tools="http://schemas.android.com/tools"
              android:layout_width="match_parent"
              android:layout_height="match_parent"
              android:orientation="vertical"
              tools:context="com.example.myapplication.app.SubActivity">
    <TextView
            android:layout_width="fill_parent"
            android:layout_height="50dp"
            android:textSize="25dp"
            android:text="This is Sub Activity"
            android:gravity="center"/>
    <Button
            android:layout_width="fill_parent"
            android:layout_height="50dp"
            android:text="Finish Sub_Activity"
            android:id="@+id/finSubActivityBtn"/>
</LinearLayout>
```

SubActivity.java

```
package com.example.myapplication.app;

import ...

public class SubActivity extends AppCompatActivity {
    public Button btn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sub);

        btn = (Button)findViewById(R.id.finSubActivityBtn);

        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}
```

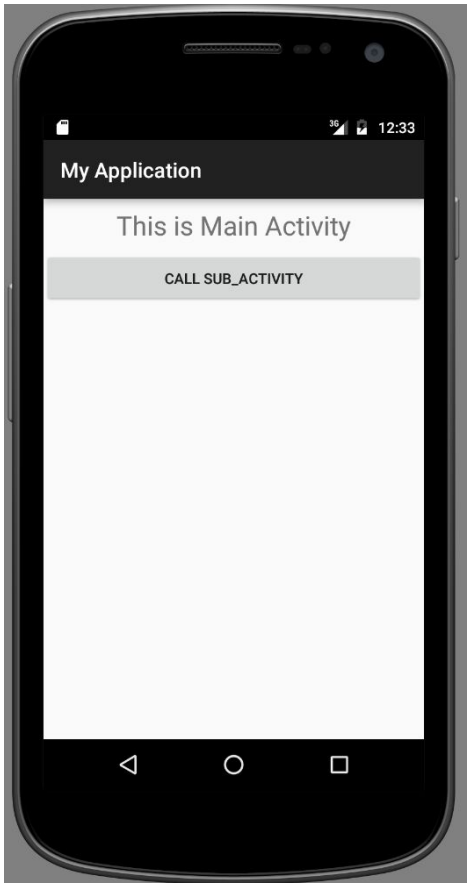
Android - Activity 예제

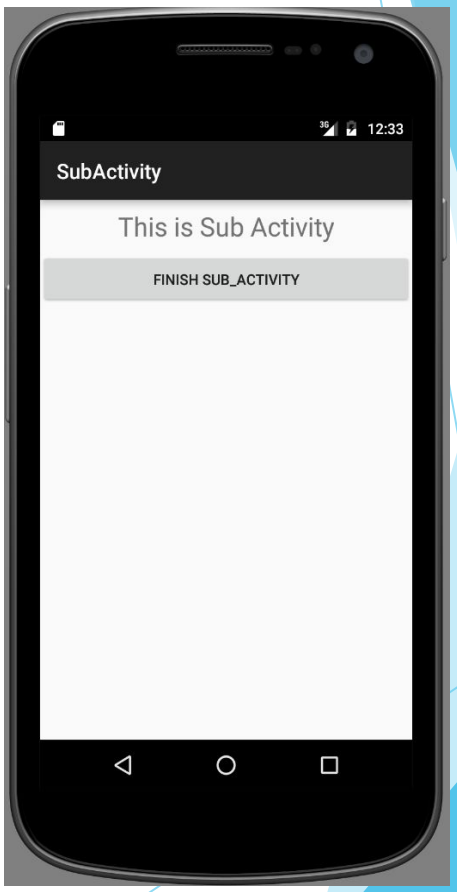
- 매니페스트 등록!
  - 보안상의 이유로 응용 프로그램에 포함된 모든 액티비티는 반드시 매니페스트에 등록되어야 함

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="My Application" >
            <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" >
        </activity>
    </application>

</manifest>
```

Android - Activity 예제

- 결과 화면





Android 개발환경 설치

- **Android 개발 도구**
  - JDK(Java Development Kit)
  - Android SDK
  - Android Studio
- **Android 개발환경 설치 순서**
  - JDK, Android Studio다운로드
    - Java 개발환경 설치하기
    - Android Studio 다운로드 및 설치하기

Java 설치하기

- http://www.oracle.com/technetwork/java/javase/downloads/index.html

Java 설치하기

Java SE Development Kit 8u71

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

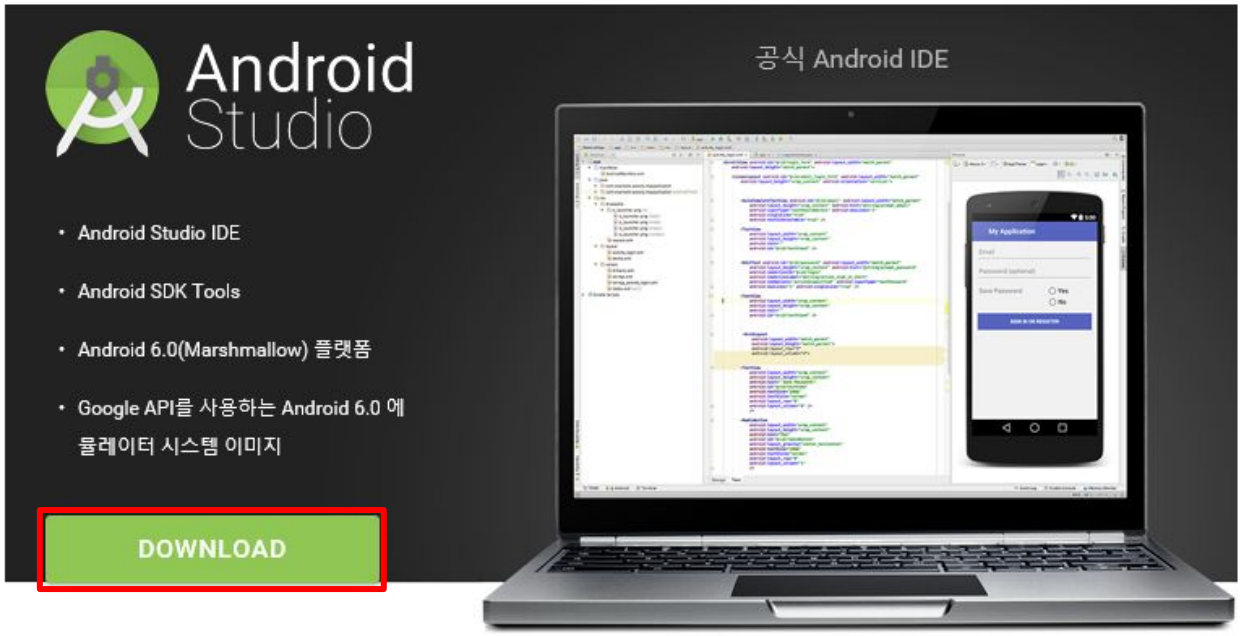
○ Accept License Agreement ◉ Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.71 MB	jdk-8u71-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.65 MB	jdk-8u71-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.75 MB	jdk-8u71-linux-i586.rpm
Linux x86	174.91 MB	jdk-8u71-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u71-linux-x64.rpm
Linux x64	172.9 MB	jdk-8u71-linux-x64.tar.gz
Mac OS X	227.24 MB	jdk-8u71-macosx-x64.dmg
Solaris SPARC 64-bit	139.78 MB	jdk-8u71-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.05 MB	jdk-8u71-solaris-sparcv9.tar.gz
Solaris x64	139.98 MB	jdk-8u71-solaris-x64.tar.Z
Solaris x64	96.19 MB	jdk-8u71-solaris-x64.tar.gz
Windows x86	181.21 MB	jdk-8u71-windows-i586.exe
Windows x64	186.55 MB	jdk-8u71-windows-x64.exe

- 이용약관 동의 후 본인의 OS에 맞는 버전을 다운로드
- Java는 이미 설치되어 있으므로, 이 과정은 생략하셔도 됩니다.

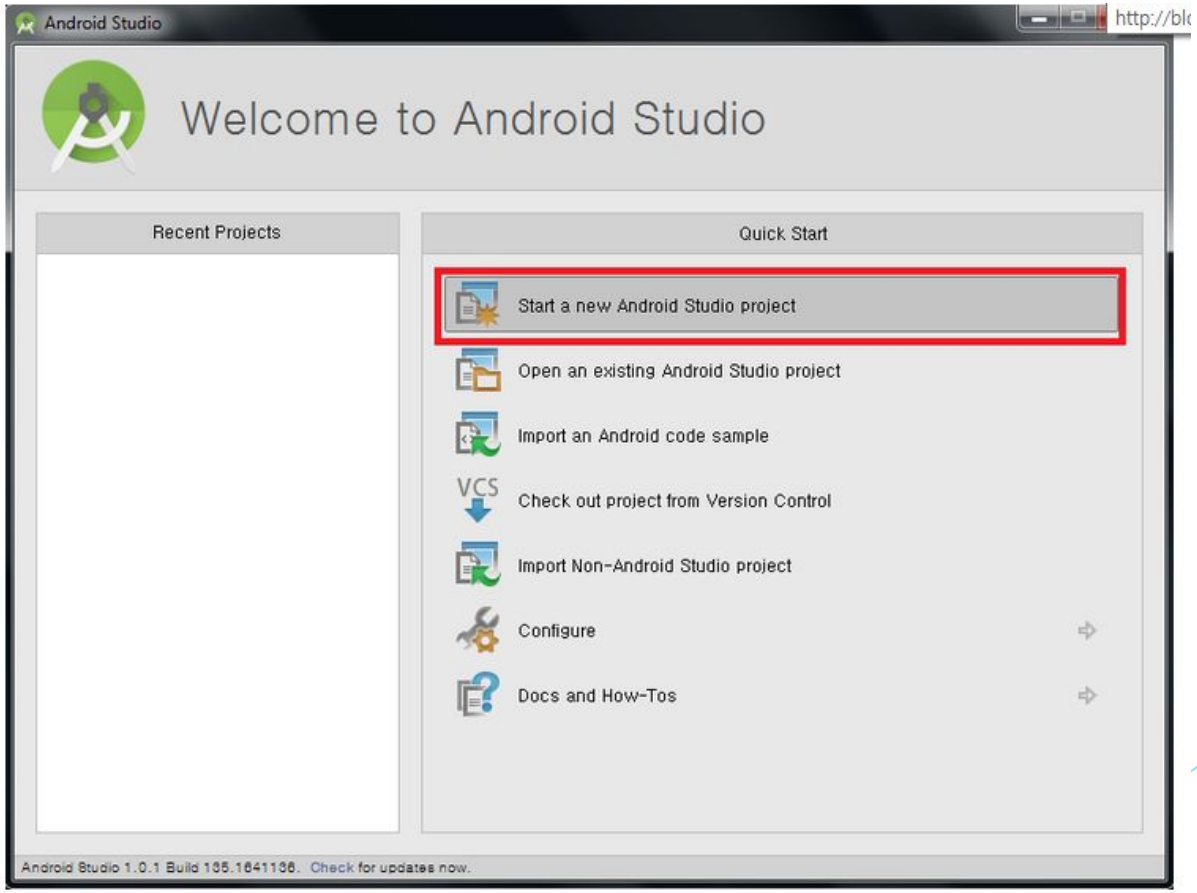
Android Studio 설치

- 해당 링크 에서 **Android Studio** 다운로드 및 설치
  - http://developer.android.com/intl/ko/sdk/index.html



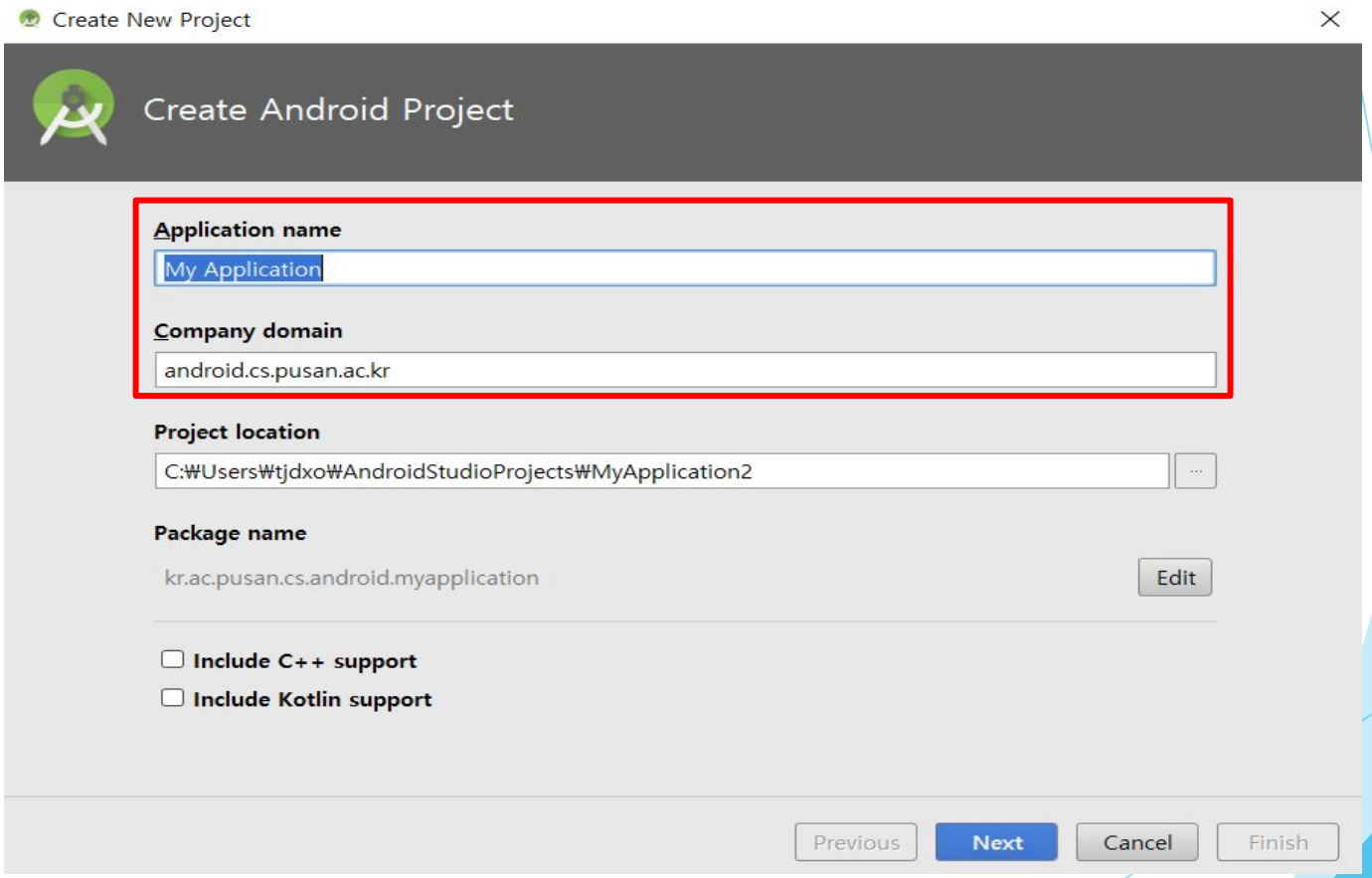
Android 프로젝트 생성

- 초기 시작 창에서
  Start a new Android Studio project 선택



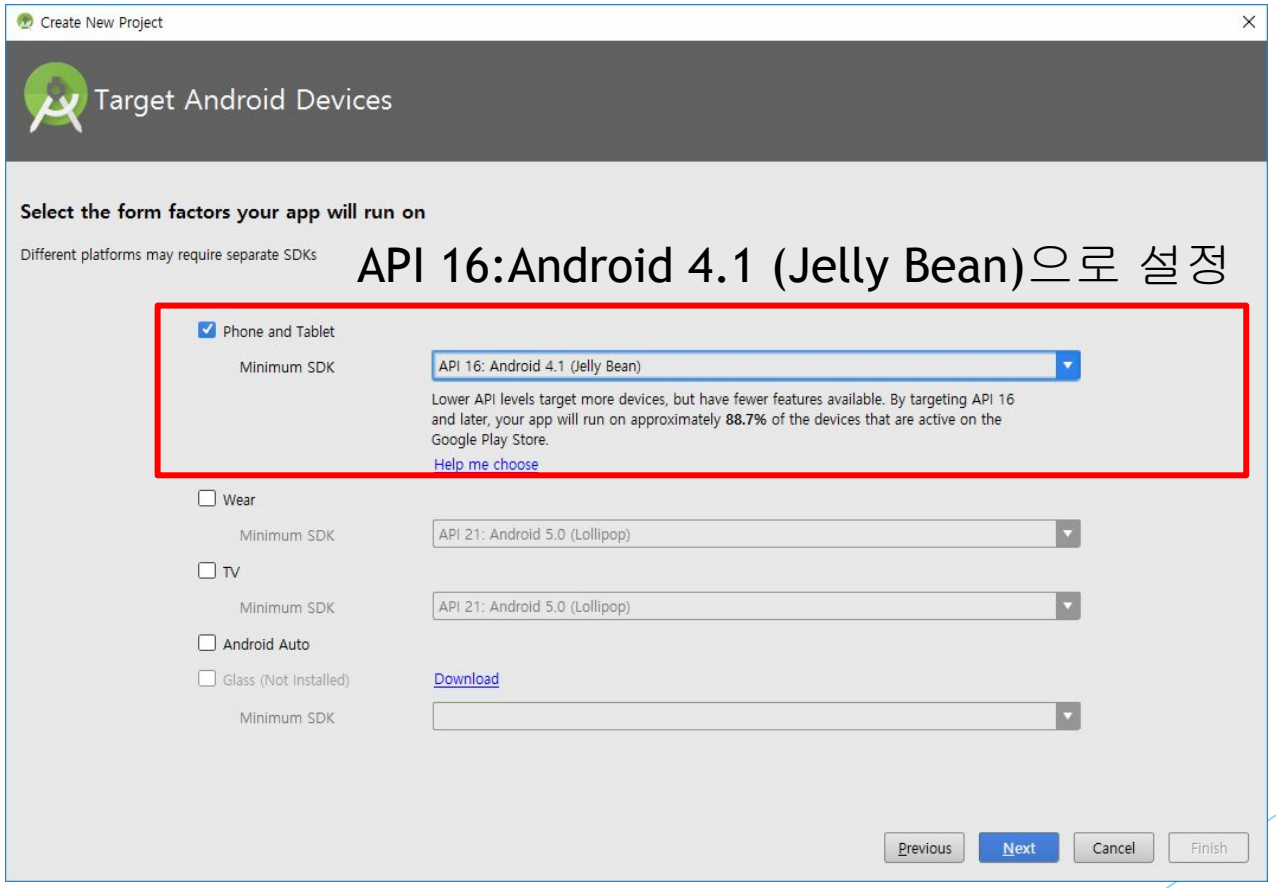
Android 프로젝트 생성

- 원하는 프로젝트 명을 입력



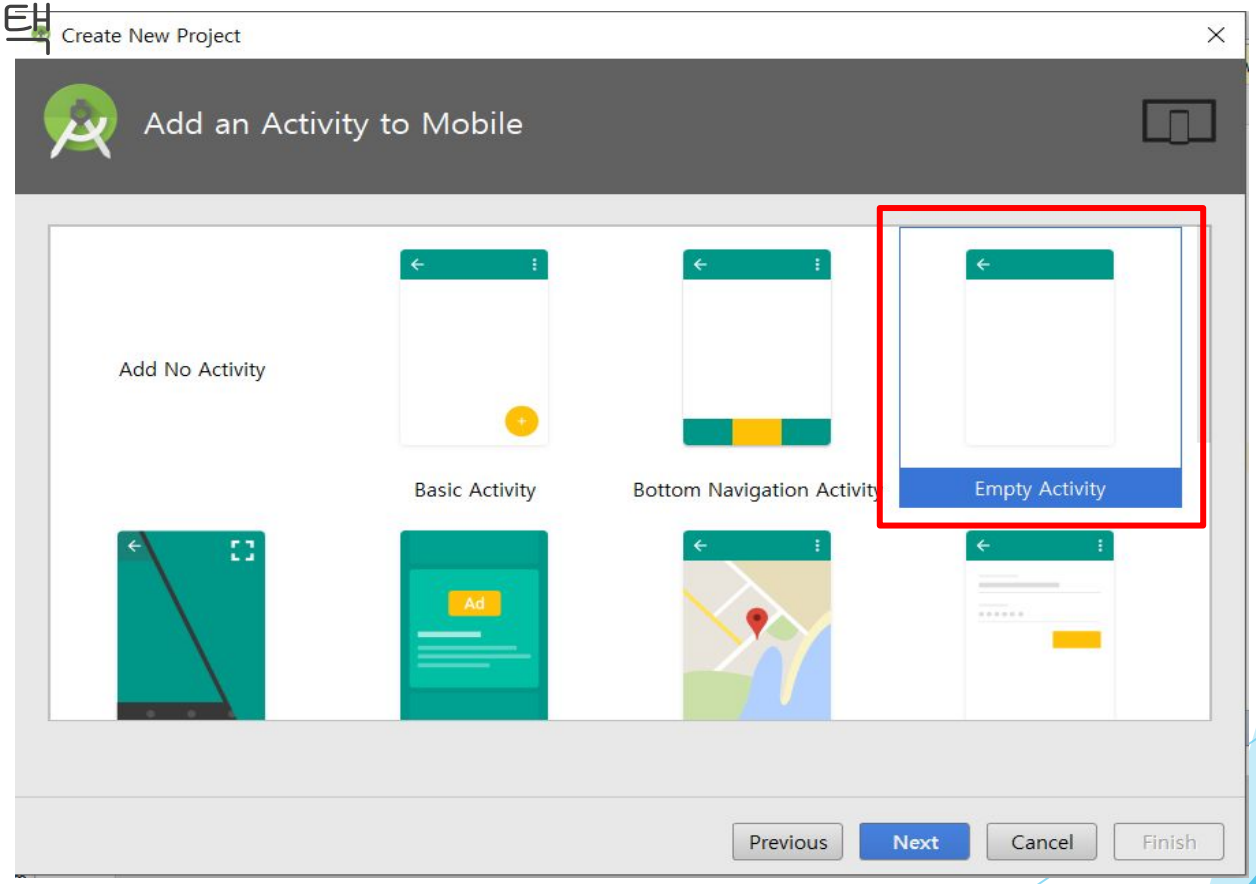
Android 프로젝트 생성

- 생성하는 프로젝트의 최소 지원 버전을 설정



Android 프로젝트 생성

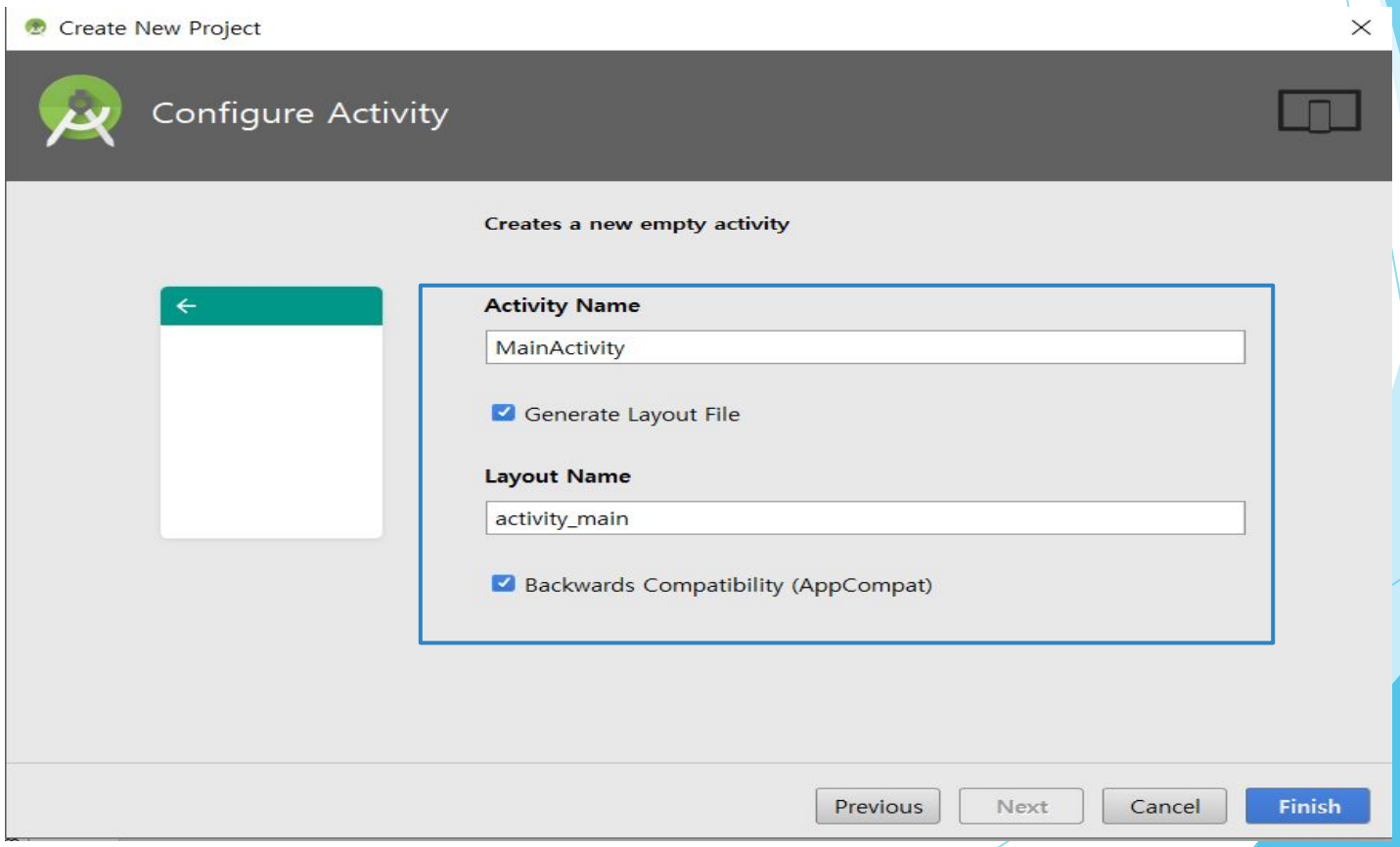
- 프로젝트의 초기화면으로 Blank Activity 선택



이 외에도 자신이 원하는 초기화면에 따라서 **Activity** 를 다르게 선택 가능

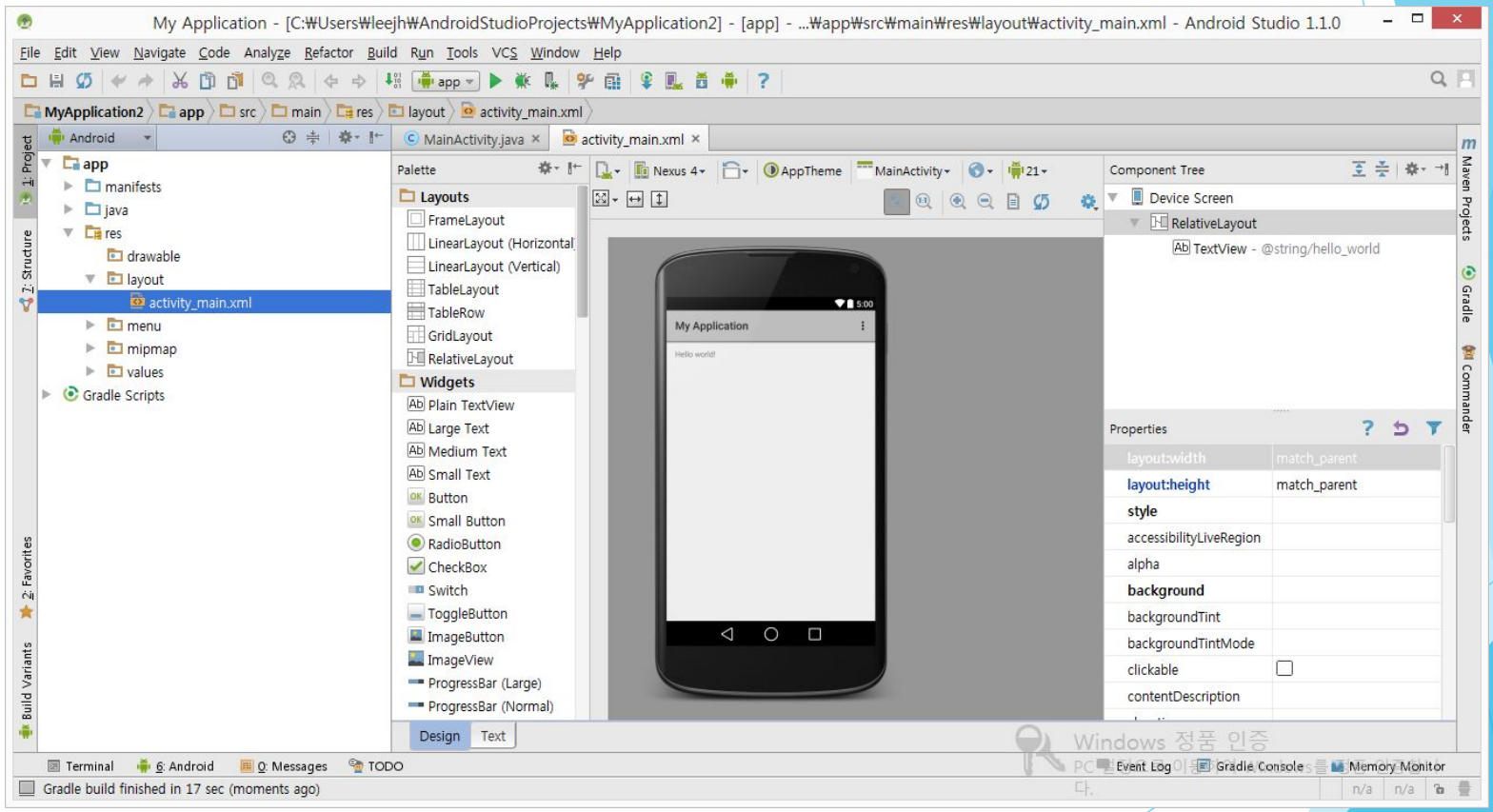
Android 프로젝트 생성

- 자신이 원하는 명칭을 입력하거나 입력 없이 **Next** 버튼 클릭



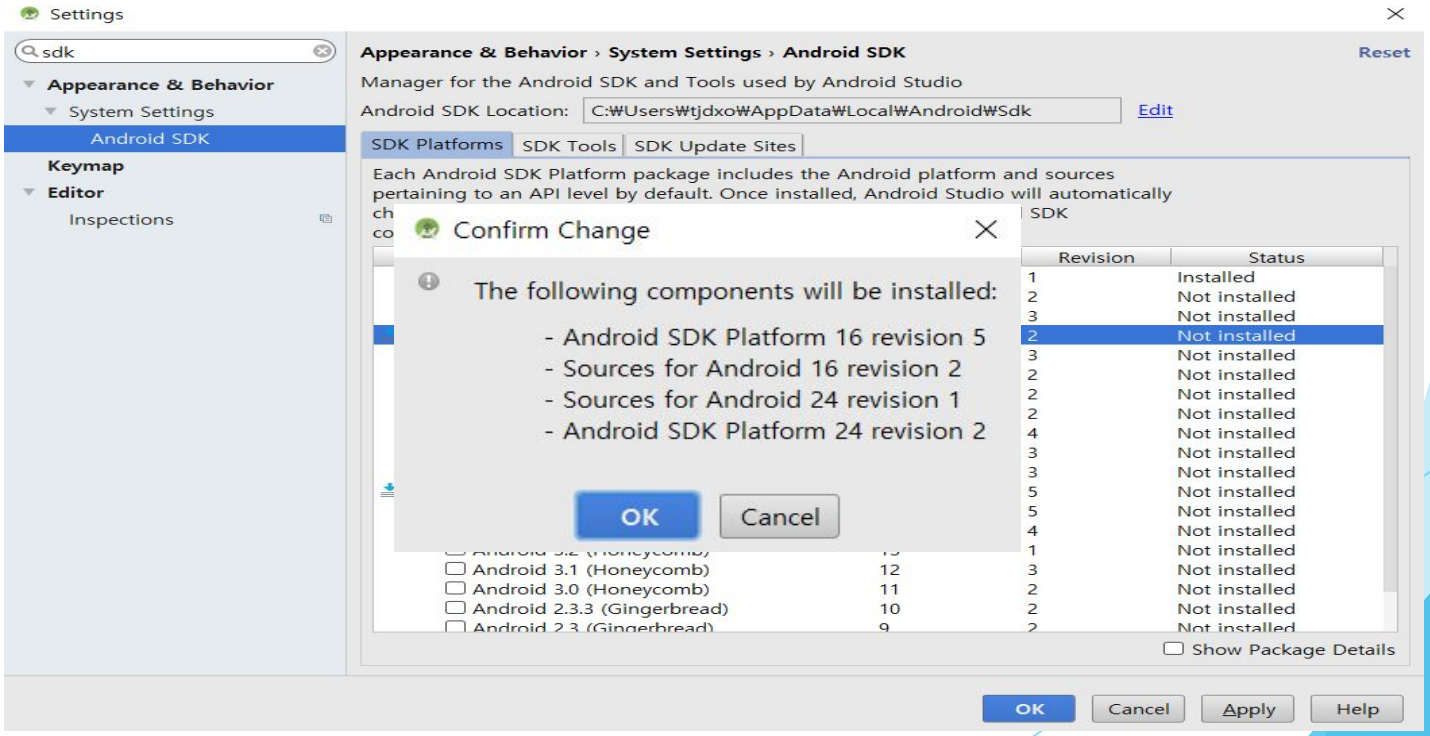
Android 프로젝트 생성

- 프로젝트 생성 완료



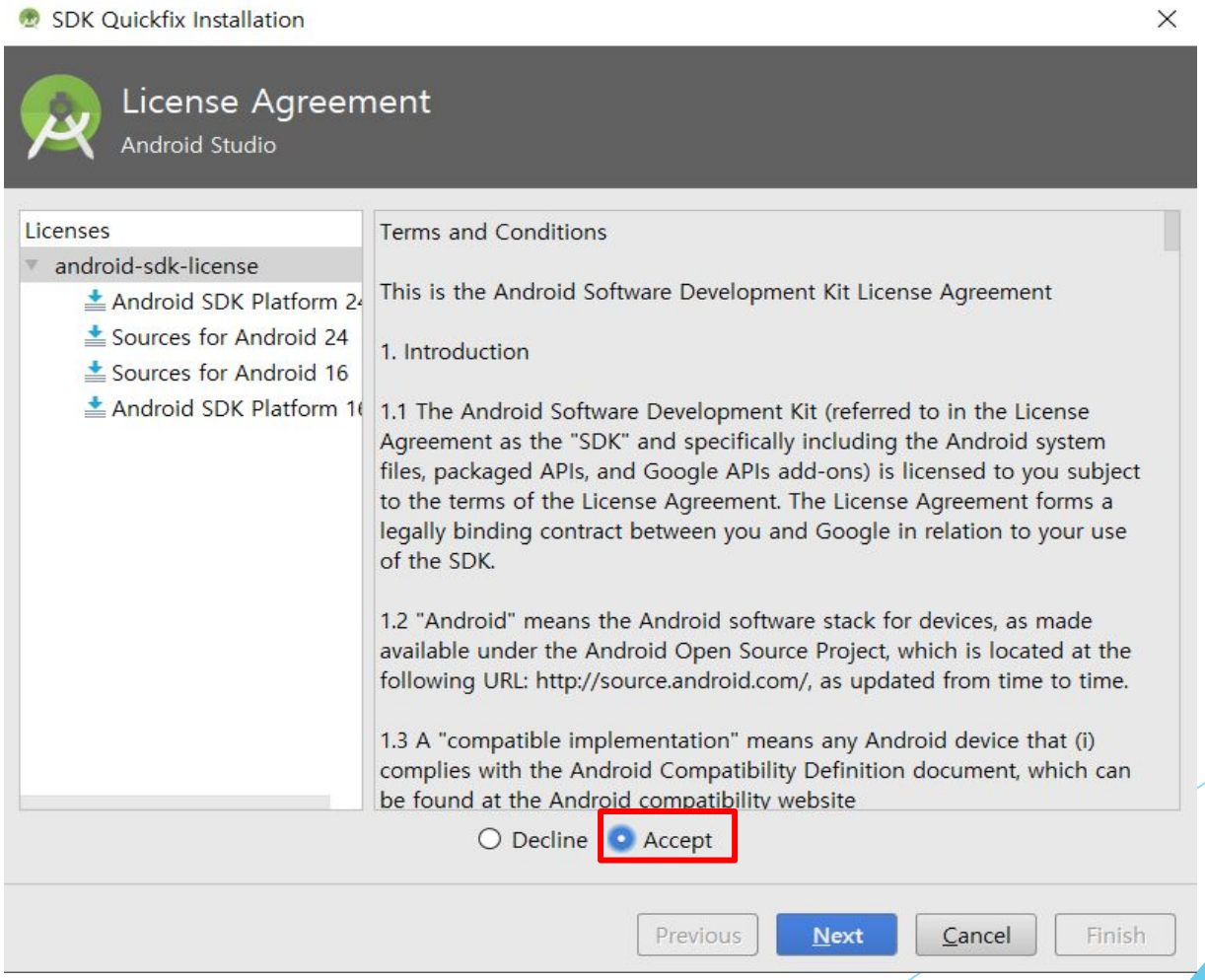
Android SDK 설치

- 상단의 File -> Settings… 클릭
- Settings 창에서 sdk 검색 후
- 해당 SDK Platform(Jelly bean, Nougat etc) 선택 후 OK 클릭 (작은 창도 OK 클릭)



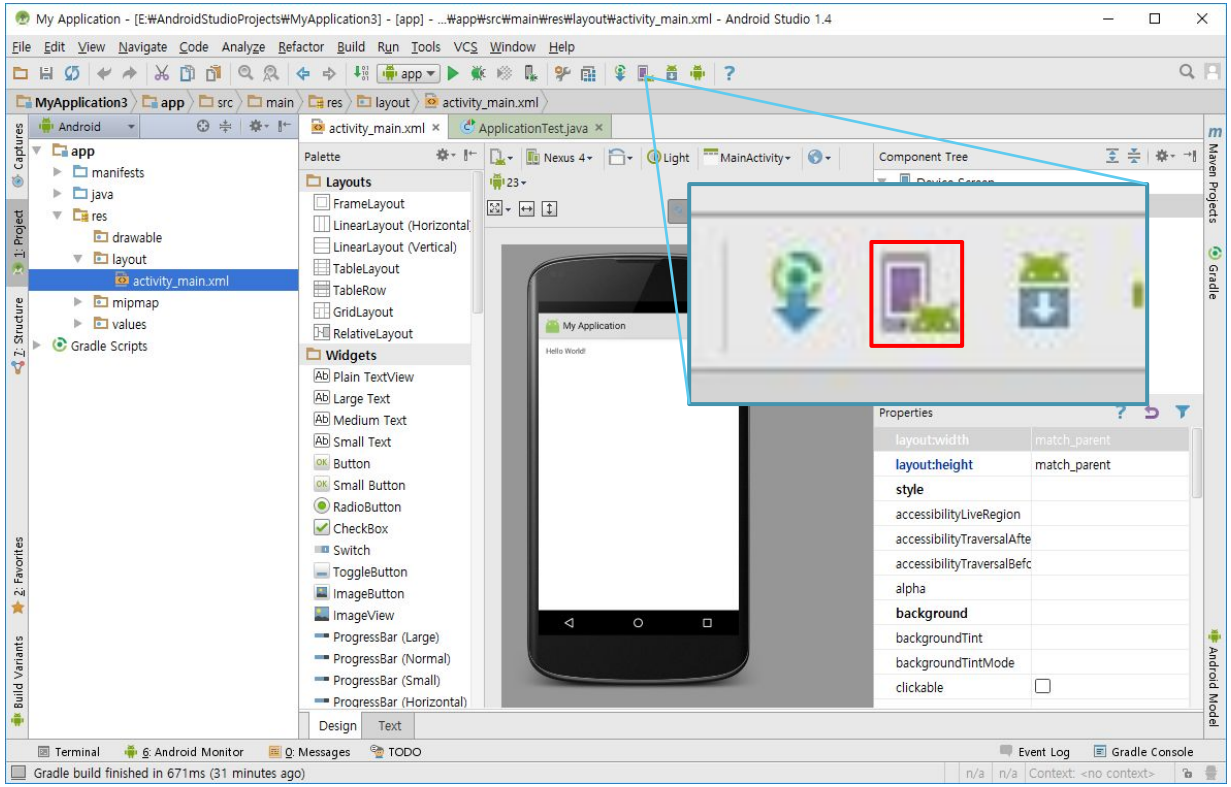
Android SDK 설치(Cont.)

- 하단에 Accept 선택 후 Next



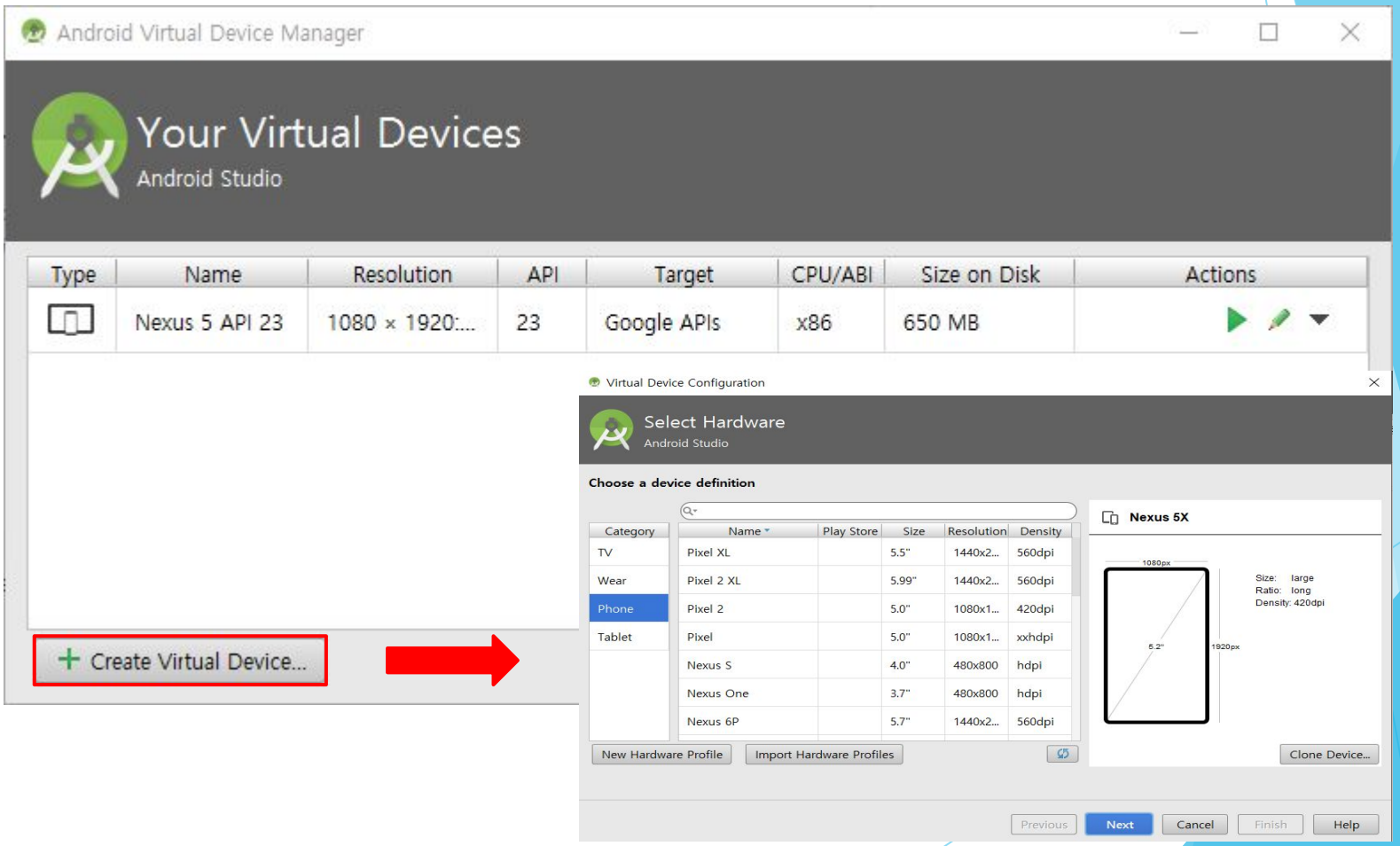
Android 에뮬레이터(AVD) 생성

- AVD Manager 선택



Android 에뮬레이터(AVD) 생성

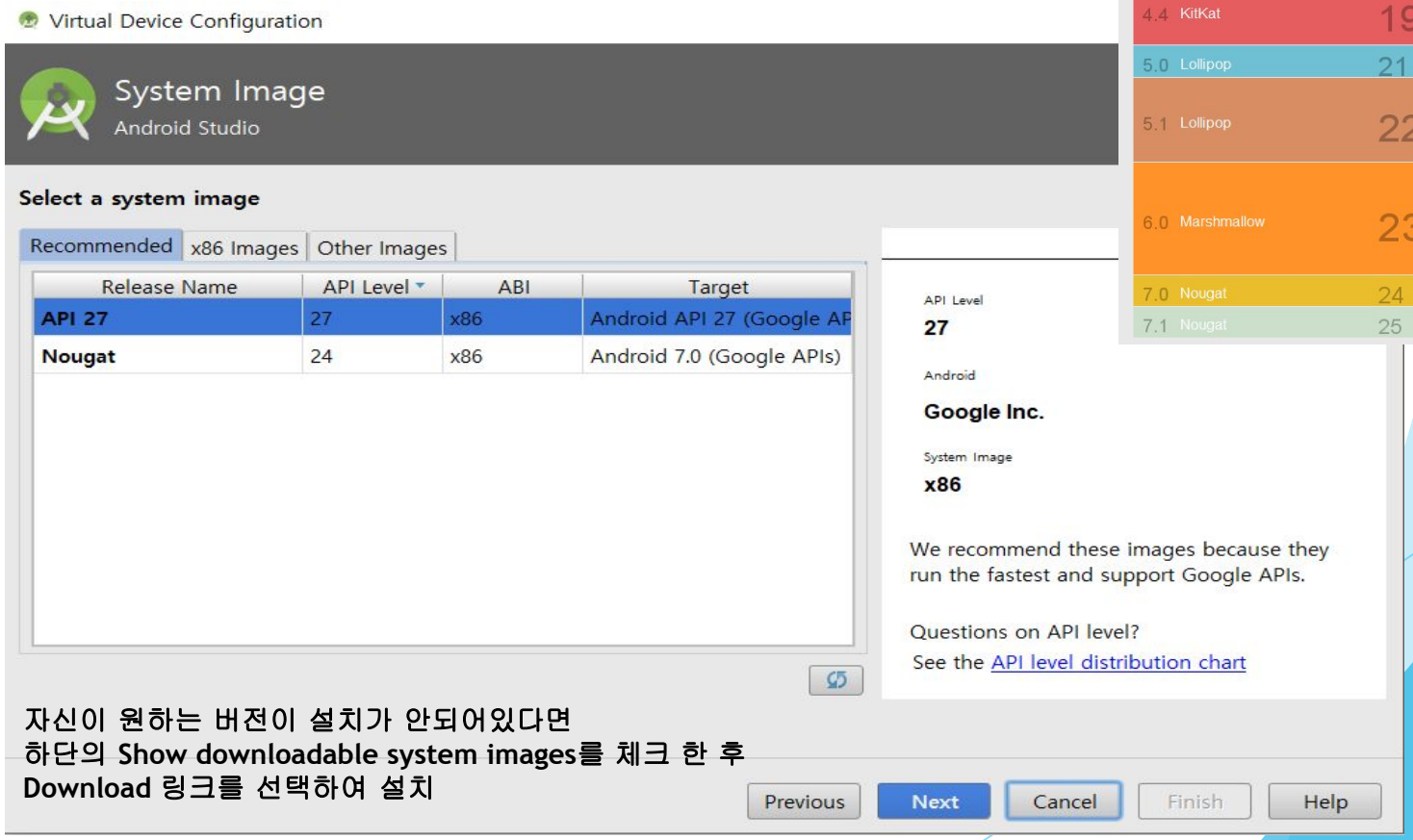
- Create Virtual Device… 버튼 클릭



Android 에뮬레이터(AVD) 생성

- Select Hardware > Galaxy Nexus 선택
- 자신이 원하는 API Level 선택 후 Next

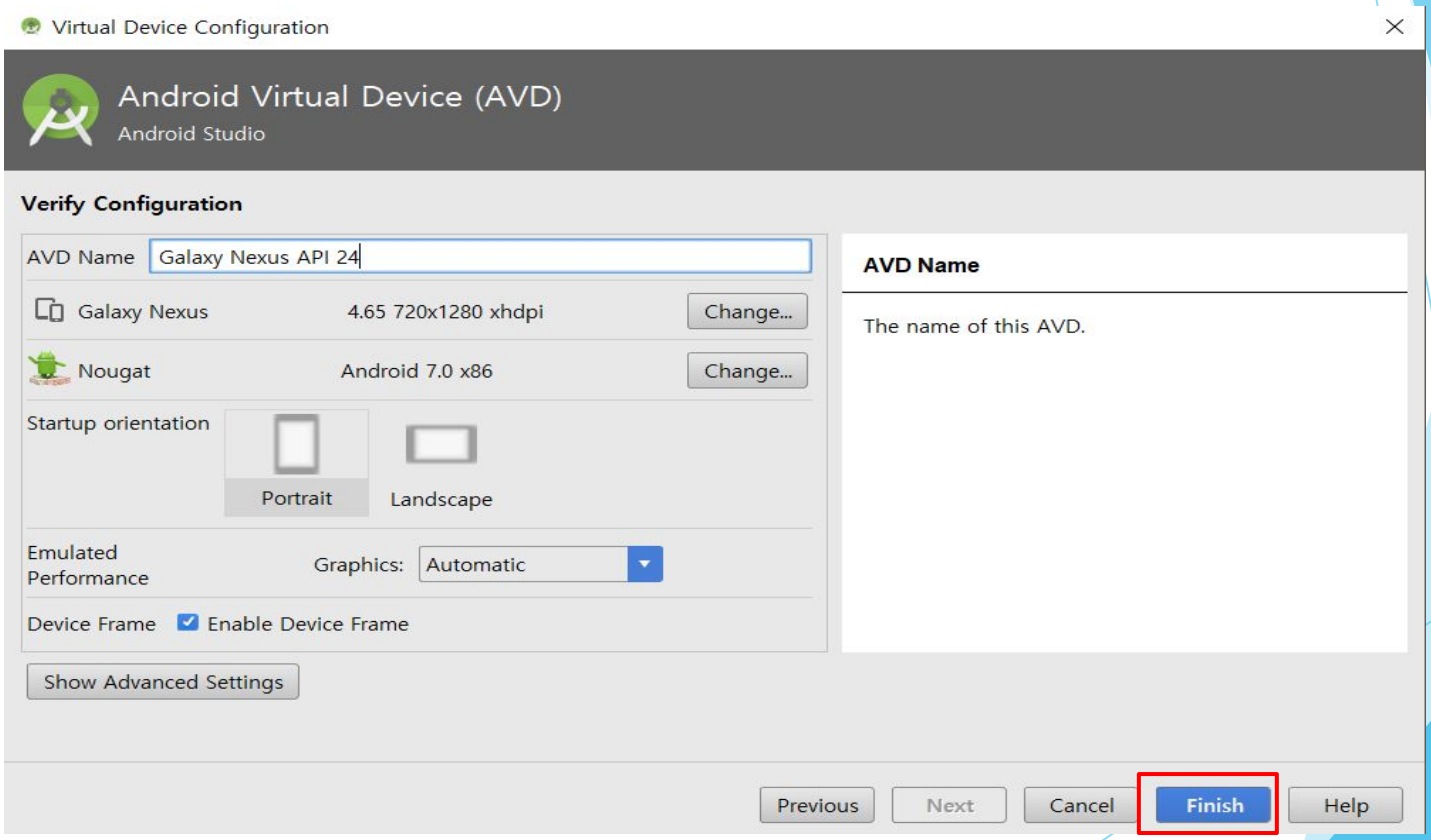
ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99.2%
4.2 Jelly Bean	17	96.0%
4.3 Jelly Bean	18	91.4%
4.4 KitKat	19	90.1%
5.0 Lollipop	21	71.3%
5.1 Lollipop	22	62.6%
6.0 Marshmallow	23	39.3%
7.0 Nougat	24	8.1%
7.1 Nougat	25	1.5%



자신이 원하는 버전이 설치가 안되어있다면
하단의 Show downloadable system images를 체크 한 후
Download 링크를 선택하여 설치

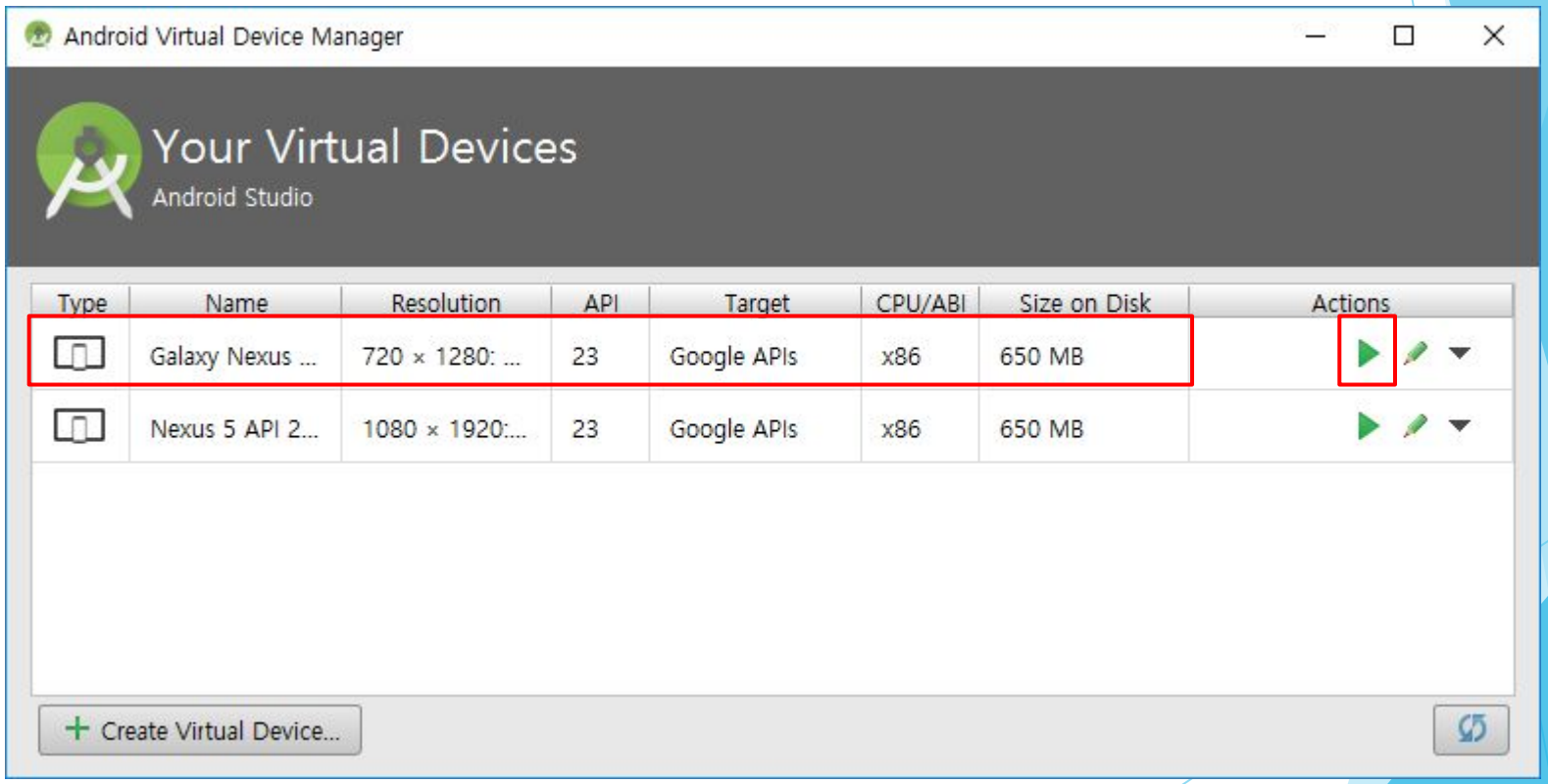
Android 에뮬레이터(AVD) 생성

- Finish 버튼 클릭



Android 에뮬레이터(AVD) 생성

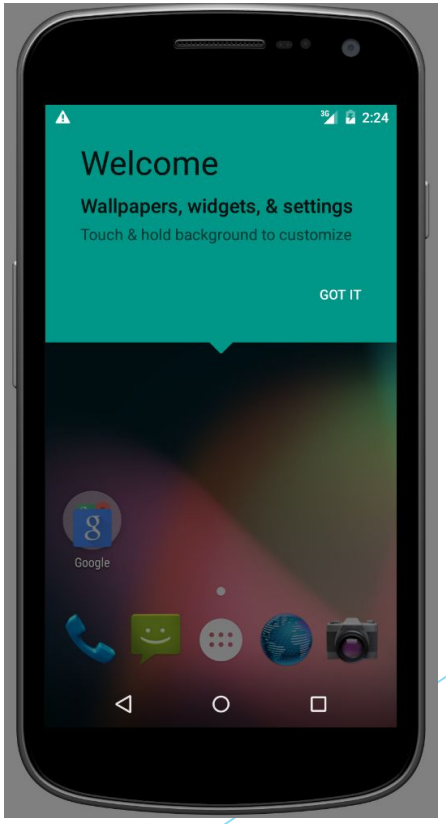
- AVD가 생성 완료 후 오른쪽 실행 버튼 클릭



Android 에뮬레이터(AVD) 생성

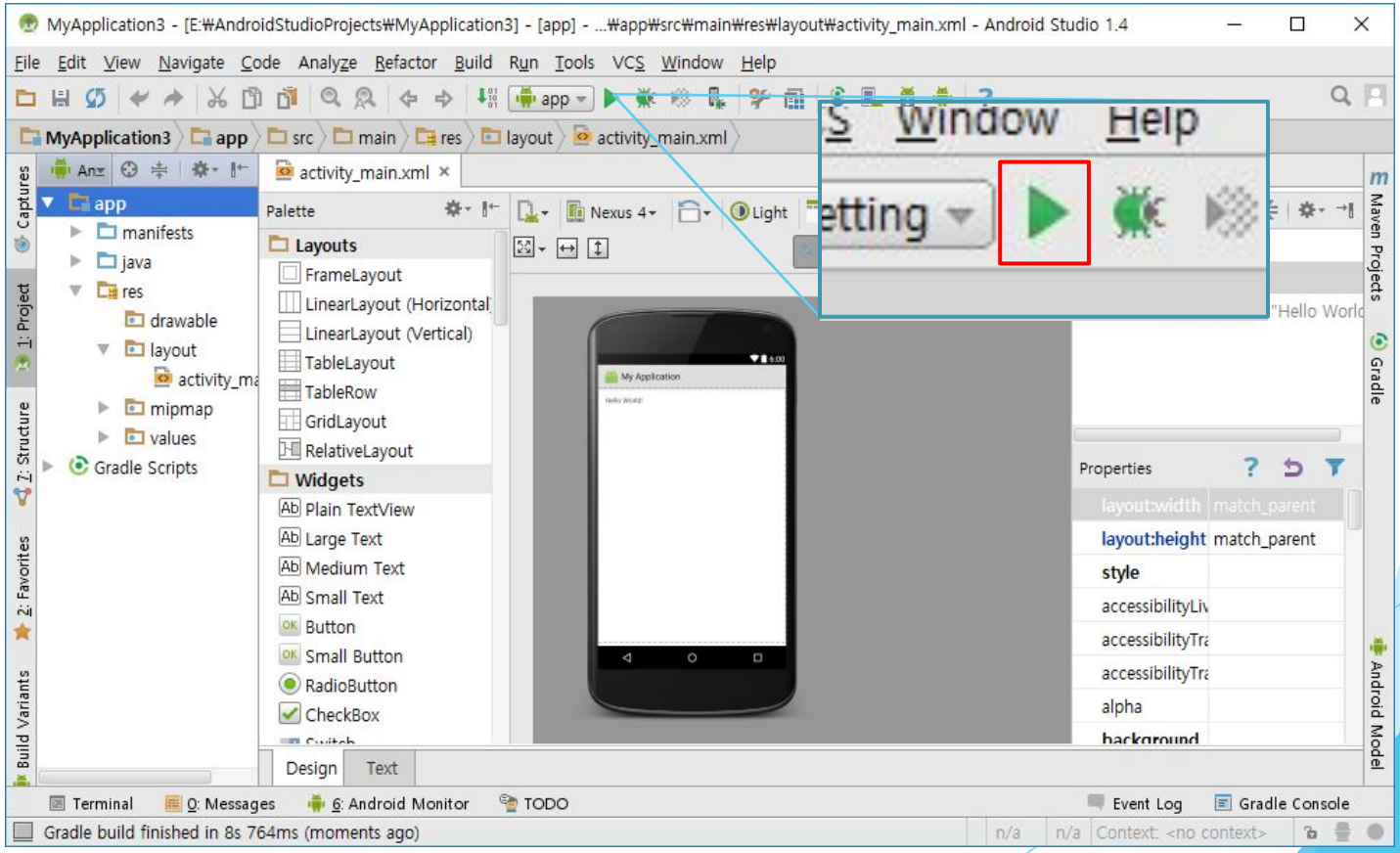
- 실행을 하면 AVD가 실행

(AVD가 실행 되기까지 컴퓨터 사양에 따라서 첫 번째 그림과 같은 화면에서 길게는 30분 정도 소요)



Hello World!를 띄워보자!

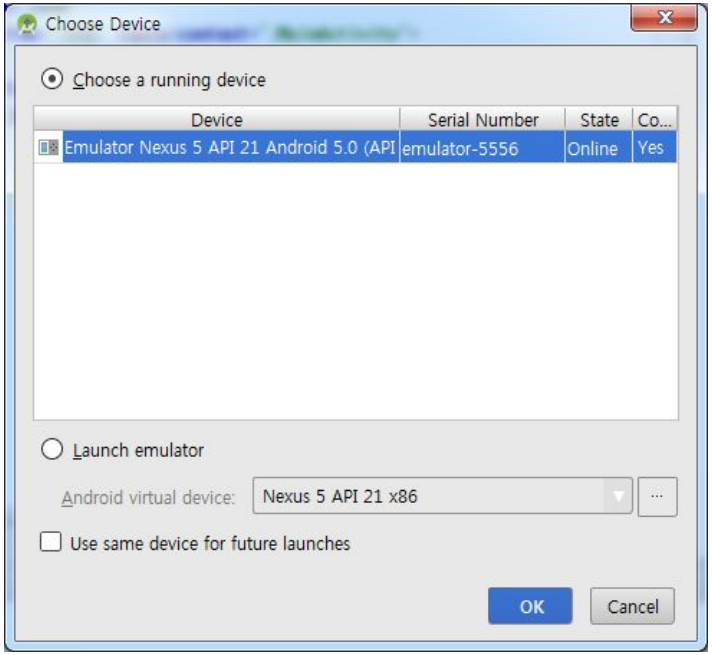
- 현재 실행 된 프로젝트에는 기본적으로 **Hello World!**가 나오도록 되어 있으므로 아래 그림과 같이 실행 버튼을 클릭
  (실행 시에 구동 될 스마트 폰 또는 **AVD**를 연결 및 구동시켜 놔야 함)



Hello World!를 띄워보자!

- 현재 실행할 플랫폼을 선택

(AVD 이외에 일반 스마트 폰이 연결 되어 있다면 목록에 같이 정렬되어 선택이 가능)



Hello World!를 띄워보자!

- Hello World!가 출력