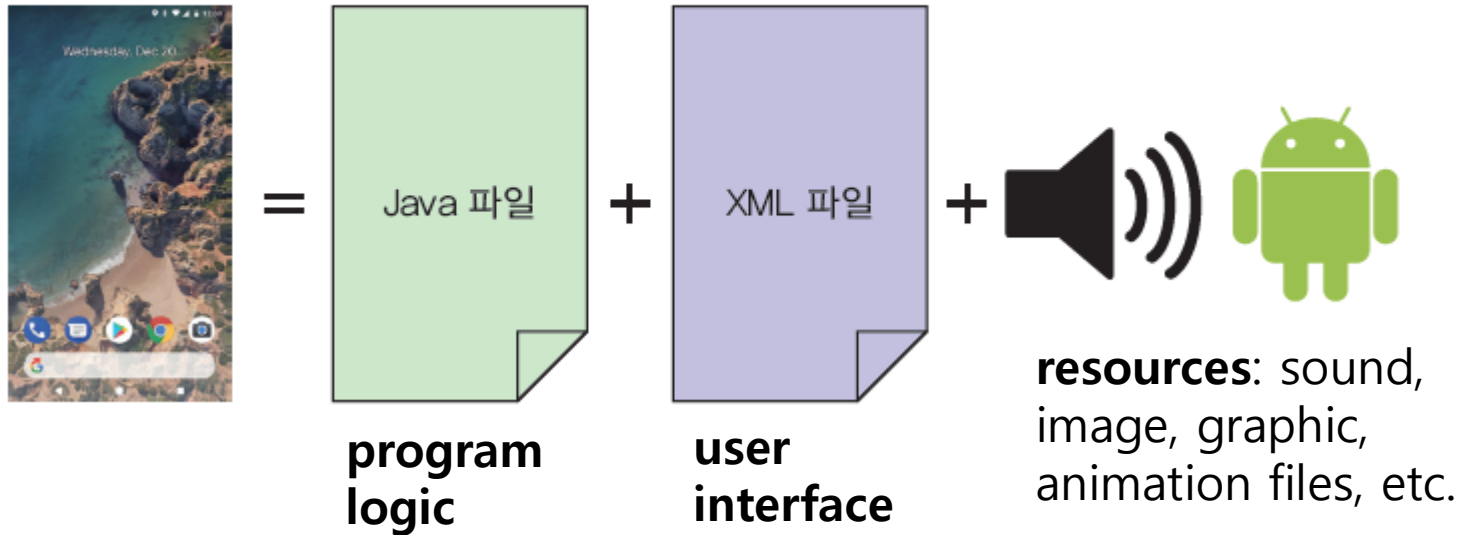


CHAP 2. 첫 번째 애플리케이션

애플리케이션의 구성

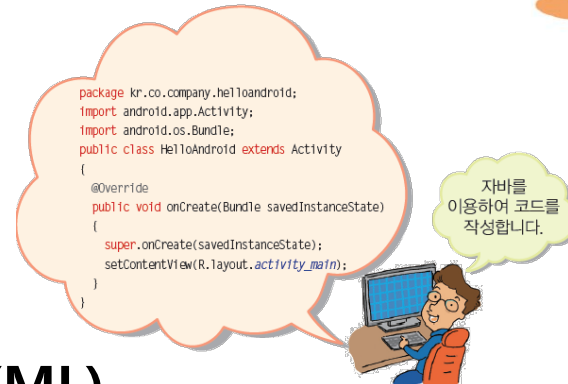


일반적인 애플리케이션 작성 절차

① 사용자 인터페이스 작성(XML)



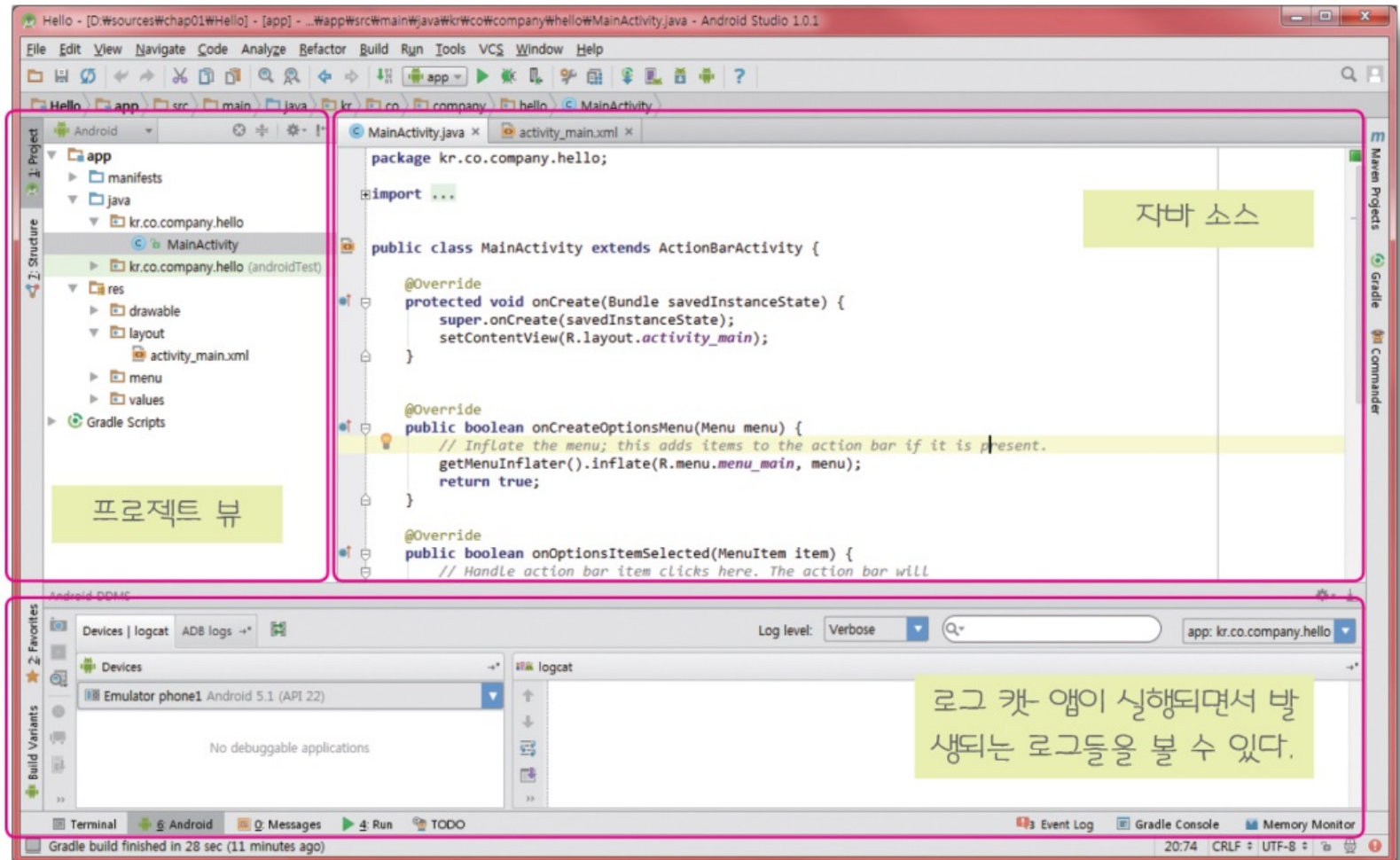
② 자바 코드 작성(JAVA)



③ 매니페스트 파일 작성(XML)

- 애플리케이션을 구성하고 있는 컴포넌트를 기술하고 실행 시에 필요한 권한을 지정한다.

애플리케이션의 구성



패키지 폴더의 설명

- 프로젝트 뷰의 폴더

폴더 또는 파일	설명
java	소스 파일들이 들어있는 폴더이다. 폴더 안의 kr.co.company.hello는 패키지의 이름이다.
Gradle Scripts	그레이들(Gradle)은 빌드 시에 필요한 스크립트이다.
res	각종 리소스(자원)들이 저장되는 폴더이다. drawable에는 해상도 별로 아이콘 파일들이 저장된다. layout에는 화면의 구성을 정의한다. values에는 문자열과 같은 리소스가 저장된다. menu에는 메뉴 리소스들이 저장되어 있다.
manifest	XML 파일로 애플리케이션의 전반적인 정보, 즉 이름이나 내장 컴포넌트 구성과 같은 정보를 가지고 있다.

자동 생성된 소스 관찰



MainActivity.java

```
package kr.co.company.hello;
```

← 패키지 지정 문장

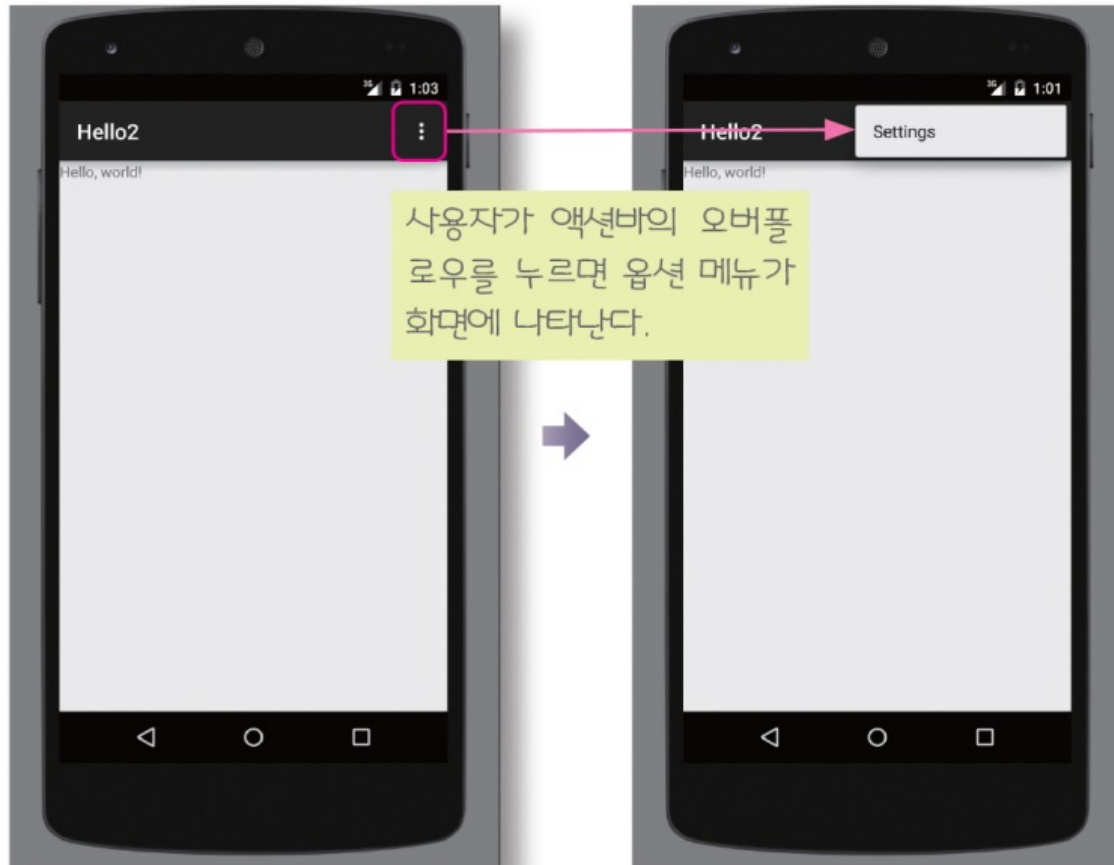
```
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;
```

← 필요한 클래스를 포함
시키는 import 문장들

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

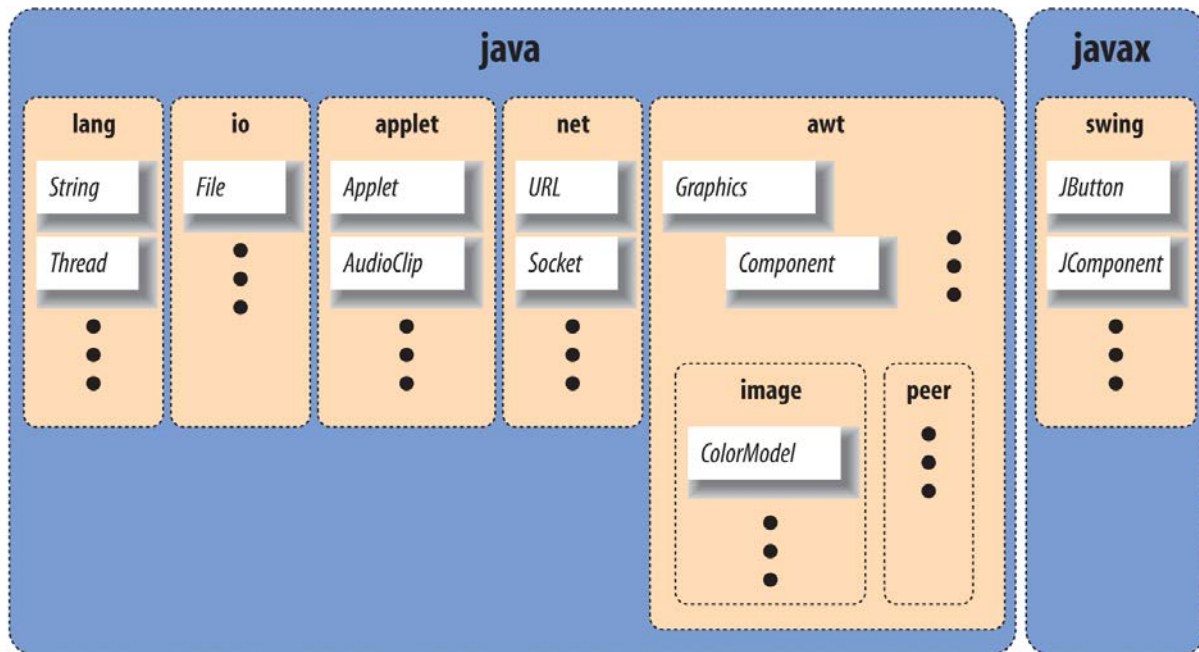
← MainActivity 클래스 정의

자동 생성 앱의 실행 결과



package kr.co.company.hello;

- 패키지(package)는 클래스들을 보관하는 컨테이너
- 일반적으로 인터넷의 도메인 이름을 역순으로 사용



import android.support.v7.app.AppCompatActivity...

- import 문장은 외부에서 패키지나 클래스를 포함
- 앞에 android가 붙은 패키지는 안드로이드가 제공하는 패키지를 의미한다.

MainActivity.java

```
package kr.co.company.hello;
```

← 패키지 지정 문장

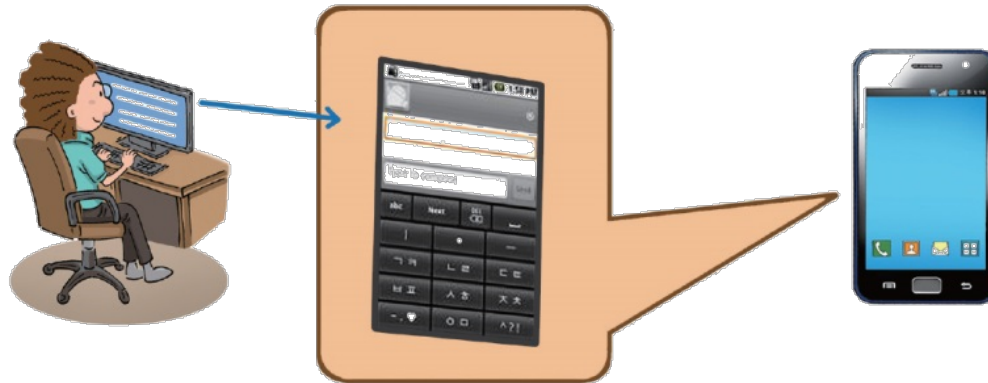
```
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;
```

← 필요한 클래스를 포함
시키는 import 문장들

```
public class MainActivity extends AppCompatActivity {  
    @Override
```

```
public class MainActivity extends  
AppCompatActivity { ... }
```

- 클래스의 정의
- **Activity**로부터 상속받았으므로 액티비티가 된다.
- **액티비티**는 안드로이드에서 애플리케이션을 구성하는 4가지의 컴포넌트 중의 하나이다.



액티비티는 화면을 통하여 사용자와
상호작용하는 활동을 의미한다.

@Override

- 어노테이션의 하나
- 어노테이션은 컴파일러에게 추가적인 정보를 주는 것
- @Override은 메소드가 부모 클래스의 메소드를 재정의(오버라이드)하였다는 것을 나타낸다.

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

public void onCreate() { ... }

- onCreate() 메소드는 액티비티가 생성되는 순간에 딱 한번 호출
- 모든 초기화와 사용자 인터페이스 설정이 여기에 들어간다.

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

super.onCreate(savedInstanceState);

- 위의 문장은 부모 클래스인 AppCompatActivity 클래스의 onCreate()를 호출하는 문장

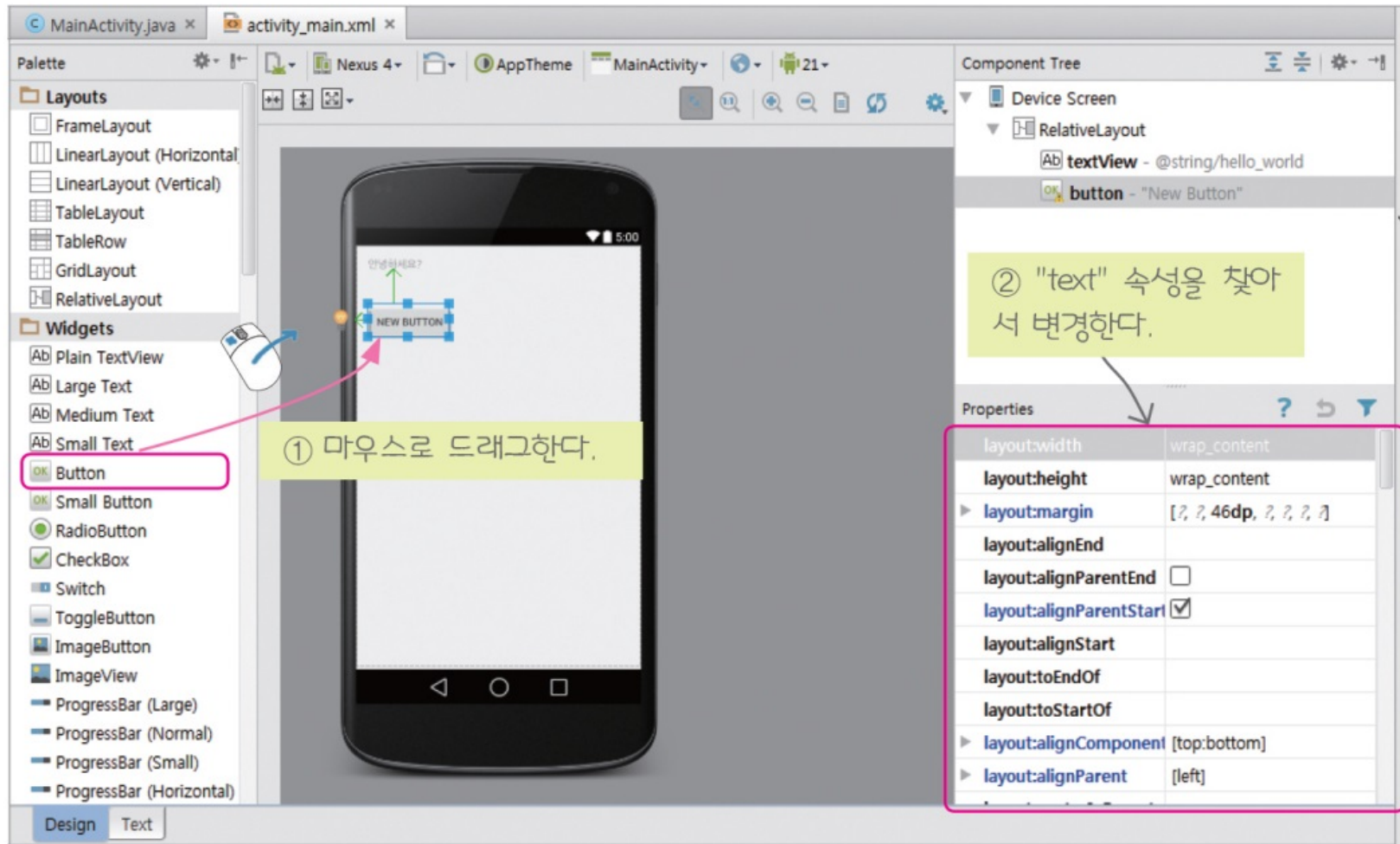
```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

setContentView(R.layout.activity_main);

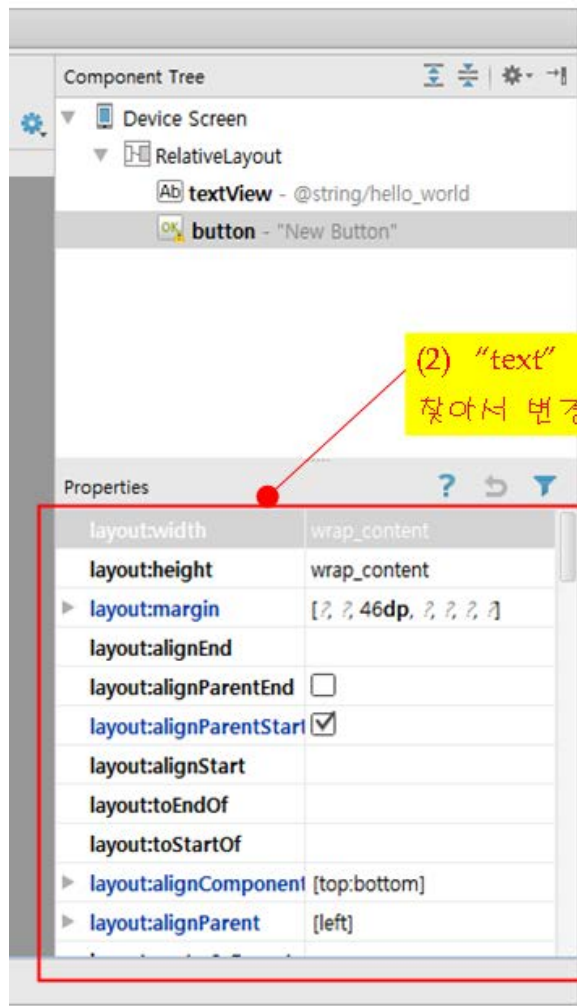
- setContentView()라는 함수는 액티비티의 화면을 설정하는 함수
- R.layout.activity_main은 activity_main.xml 파일을 나타낸다.

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

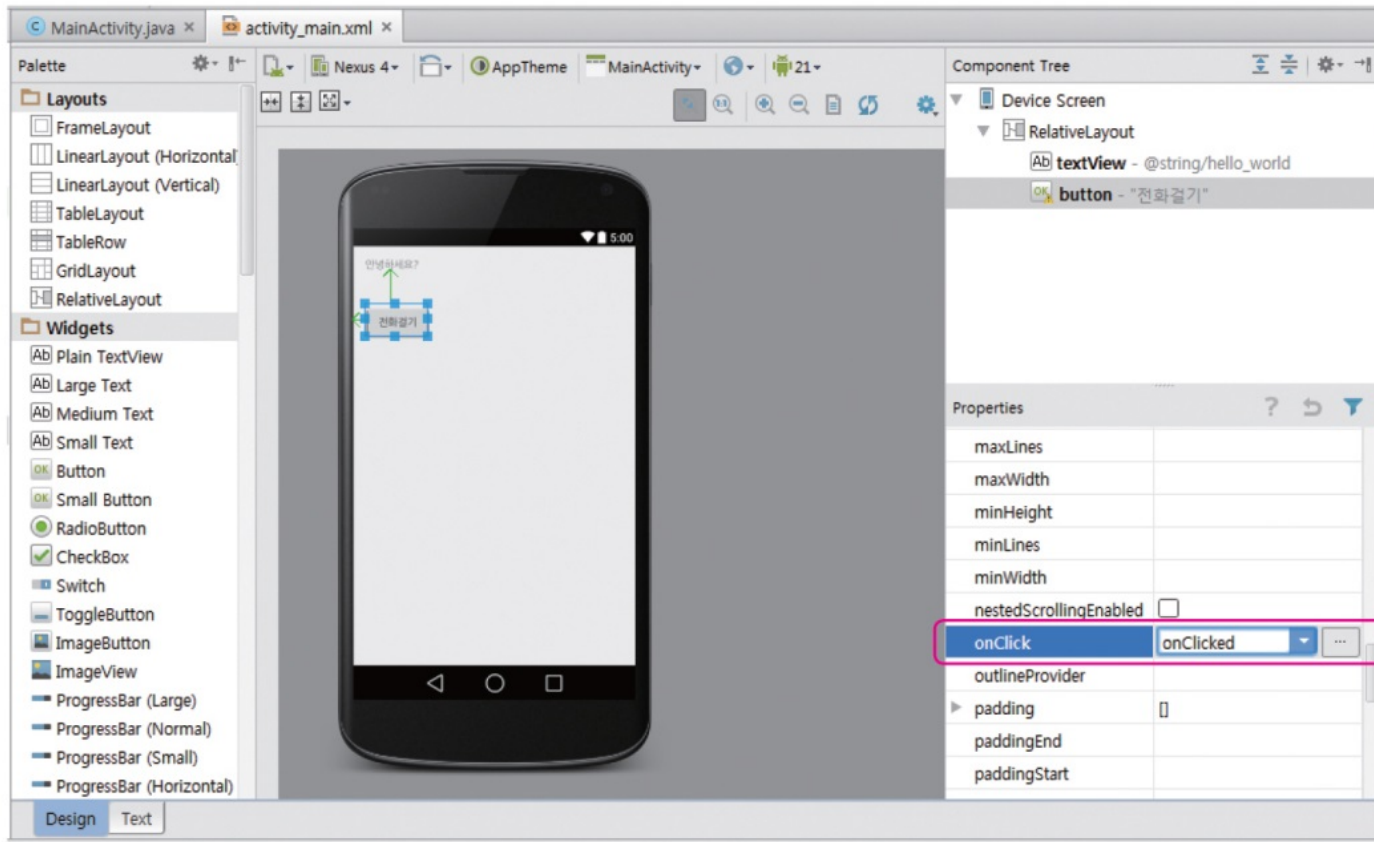
자동 생성된 소스 변경



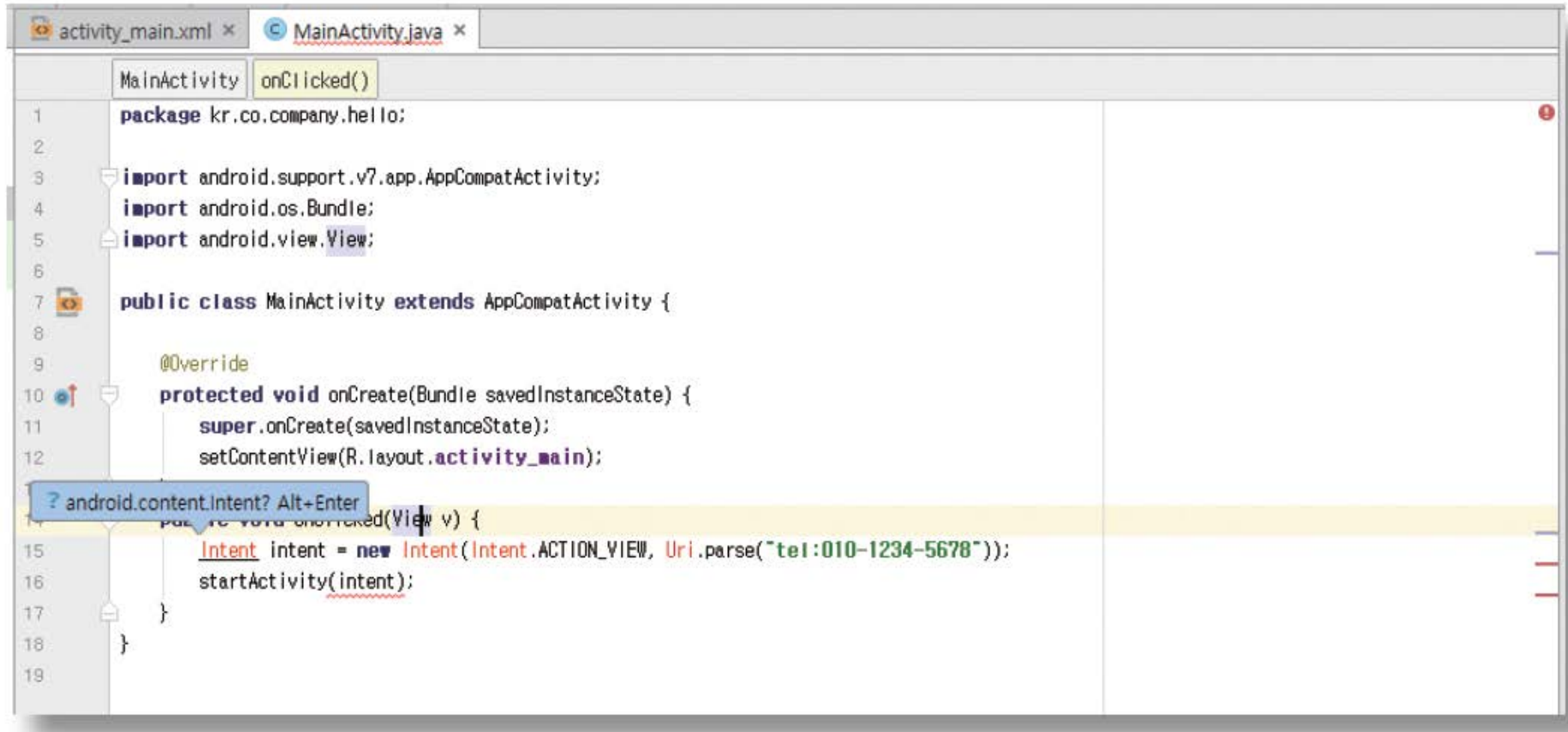
버튼의 텍스트 변경



onClick 속성 변경

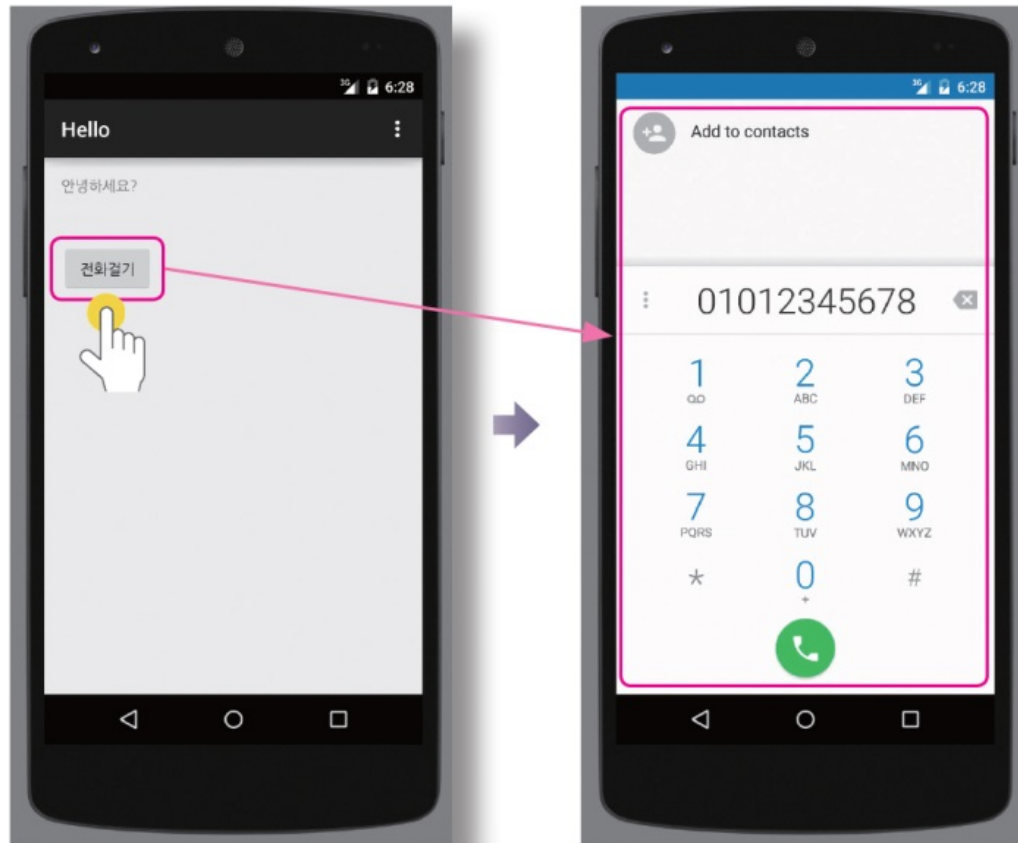


소스 추가



```
1 package kr.co.company.hello;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.view.View;
6
7 public class MainActivity extends AppCompatActivity {
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13
14        findViewById(R.id.button).setOnClickListener(new View.OnClickListener() {
15            @Override
16            public void onClicked(View v) {
17                Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("tel:010-1234-5678"));
18                startActivity(intent);
19            }
20        });
21    }
22}
```

변경된 앱 실행



MainActivity

- 안드로이드 애플리케이션의 실행이 시작되는 곳
- 안드로이드에는 `main()`이 없음.
- 액티비티별로 실행된다.
- 액티비티 중에서는 `onCreate()` 메소드가 가장 먼저 실행

XML을 이용하여서 사용자 인터페이스 기술

- 사용자 인터페이스 작성 방법
 - 코드를 사용하는 방법(기존의 자바)
 - XML을 사용하는 방법(안드로이드 권장)
- 안드로이드에서는 UI 화면의 구성을 XML을 이용하여서 선언적으로 나타내는 방법을 선호
 - 애플리케이션의 외관과 애플리케이션의 로직을 서로 분리
 - 빠르게 UI를 구축

XML을 이용한 사용자 인터페이스 작성

- 코드로 작성하였던 UI를 XML로 표현하면

```
TextView tv = new TextView(this);  
tv.setText("Hello, world!");
```



```
<?xml version="1.0" encoding="utf-8"?>  
<TextView xmlns:android="http://schemas.android.com/apk/res/android"  
    android:id="@+id/textview"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Hello, world!" />
```

텍스트뷰를 나타내는
요소



XML을 이용한 사용자 인터페이스 작성

- UI 컴포넌트 들은 XML의 하나의 요소로 표현된다.
- TextView 컴포넌트는 <TextView ... /> 요소로 표현된다.

XML 파일의 분석

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/textview"
    android:layout_width= "match_parent"
    android:layout_height="wrap_content"
    android:text="Hello, world!" />
```


<TextView>의 속성 설명

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/textview"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Hello, world!" />
```

속성	의미
xmlns:android	XML 이름공간의 선언으로 안드로이드 도구에 안드로이드 이름공간에 정의된 속성들을 참조하려고 한다는 것을 암시한다. XML 파일에서 항상 최외곽 태그는 이 속성을 정의하여야 한다.
android:id	TextView 요소에 유일한 아이디를 할당한다. 이 아이디를 이용하여서 소스 코드에서 이 텍스트 뷰를 참조할 수 있다.
android:layout_width	화면에서 얼마나 폭을 차지할 것인지를 정의한다. "match_parent"는 전체 화면의 폭을 다 차지하는 것을 의미한다.
android:layout_height	화면에서 길이를 얼마나 차지할 것인지를 정의한다.
android:text	화면에 표시하는 텍스트를 설정한다. 이 속성은 예제와 같이 하드코딩될 수도 있고 아니면 문자열 리소스의 개념을 사용할 수도 있다.

XML

- **요소(element):** 시작 태그로 시작되어 종료 태그로 끝나는 논리적인 구성 요소

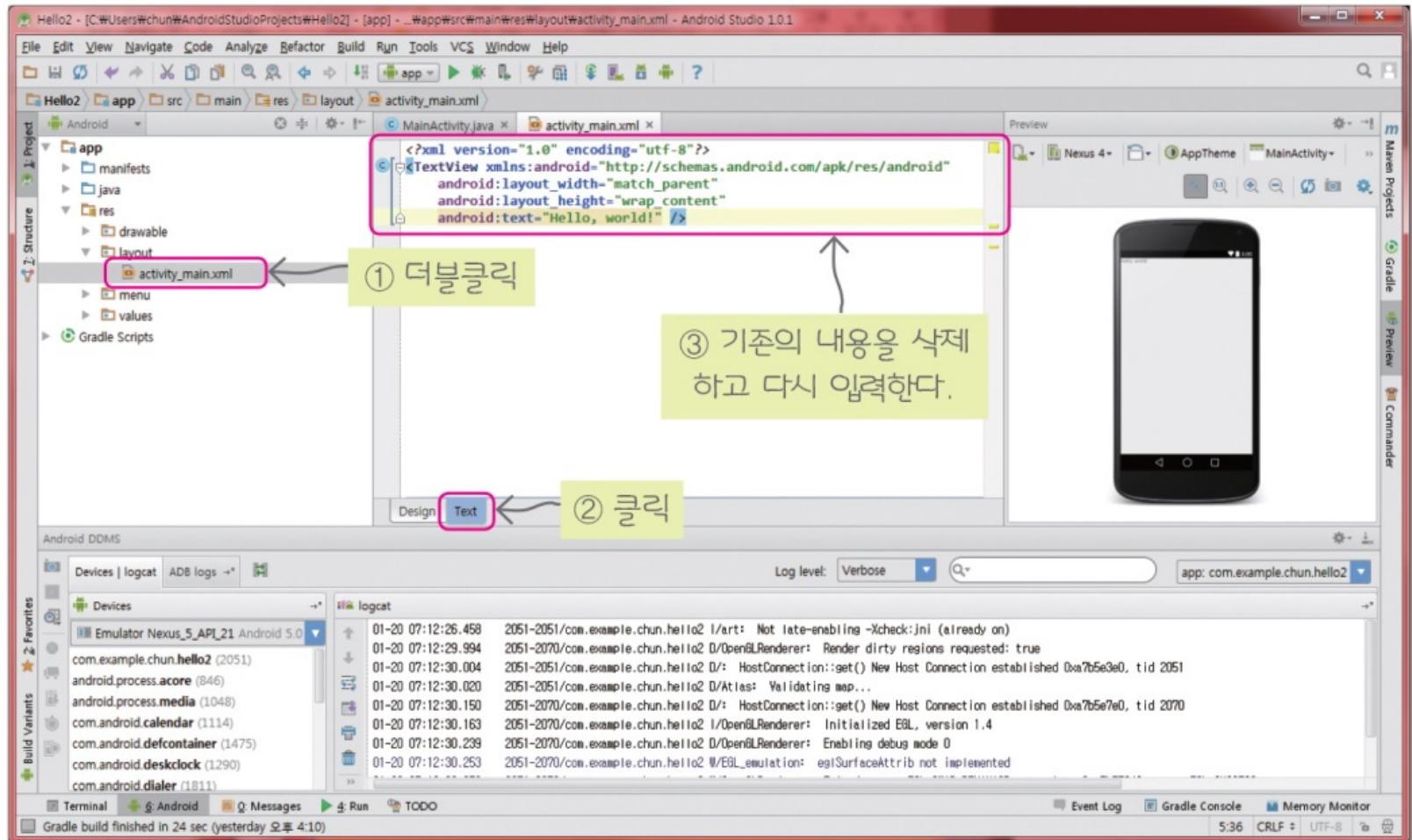
예) <**Greeting**>Hello, world.</**Greeting**>

- **속성(attribute):** 요소의 속성으로서 "이름/값"의 쌍으로 구성

예)

img 요소 안에 **src**와 **alt**라는 2개의 속성 설정.

XML 파일의 위치



XML파일과 코드와의 연결

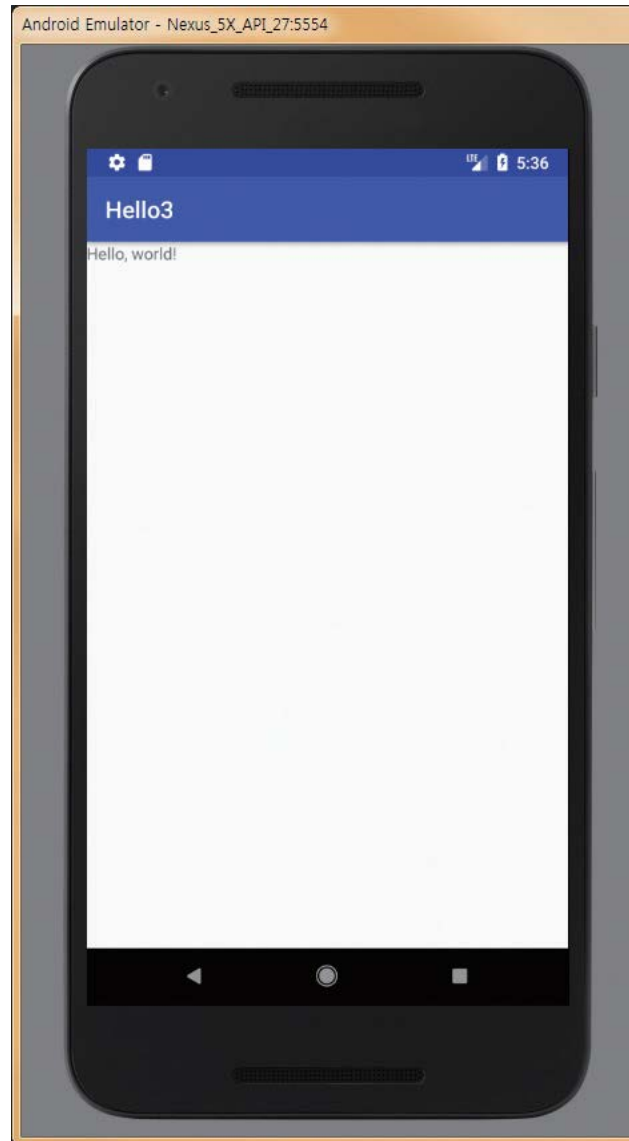
MainActivity.java

```
package kr.co.company.hello3;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

액티비티의 화면을
activity_main.xml로 설정

실행 결과



코드에서 리소스를 참조하는 방법

MyActivity.java

Files under the build folder are generated and should not be edited.

activity_my

```
public static final int abc_action_mode_close_item_material=0x7f040006;
public static final int abc_activity_chooser_view=0x7f040007;
public static final int abc_activity_chooser_view_include=0x7f040008;
public static final int abc_activity_chooser_view_list_item=0x7f040009;
public static final int abc_expanded_menu_layout=0x7f04000a;
public static final int abc_list_menu_item_checkbox=0x7f04000b;
public static final int abc_list_menu_item_icon=0x7f04000c;
public static final int abc_list_menu_item_layout=0x7f04000d;
public static final int abc_list_menu_item_radio=0x7f04000e;
public static final int abc_popup_menu_item_layout=0x7f04000f;
public static final int abc_screen_content_include=0x7f040010;
public static final int abc_screen_simple=0x7f040011;
public static final int abc_screen_simple_overlay_action_mode=0x7f040012;
public static final int abc_screen_toolbar=0x7f040013;
public static final int abc_search_dropdown_item_icons_2line=0x7f040014;
public static final int abc_search_view=0x7f040015;
public static final int abc_simple_dropdown_hint=0x7f040016;
public static final int activity_my=0x7f040017;
public static final int support_simple_spinner_dropdown_item=0x7f040018;
```

public static final class menu {
 public static final int menu_my=0x7f040000;
}

리소스 XML 파일

R.java 파일

```
package com.ajoumedia.myfirstapp;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

public class MyActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu: this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_my, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

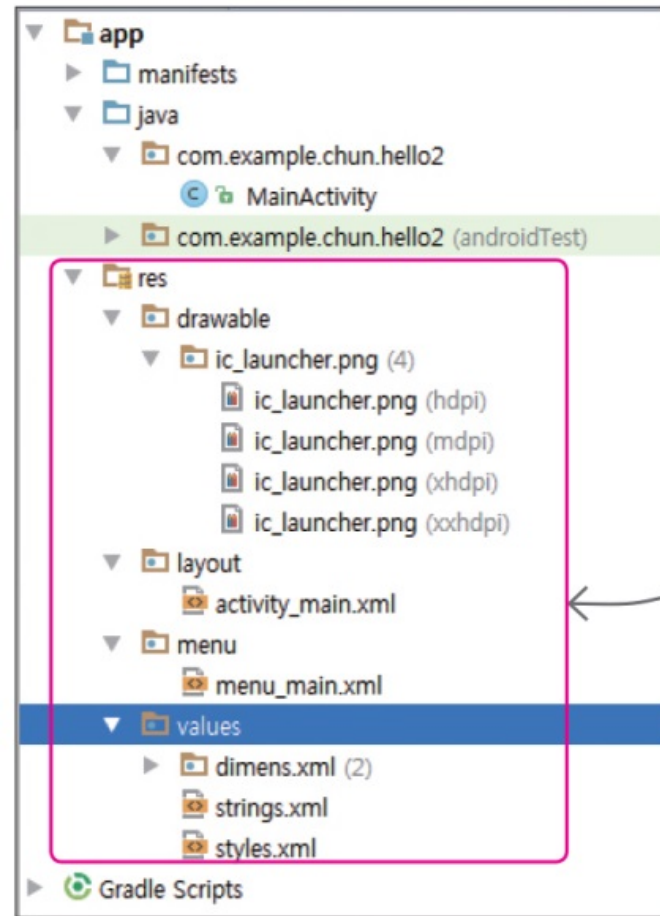
        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

location: *project folder/*
app/build/generated/source/r/debug/c
om.android.*projectname*/R.java

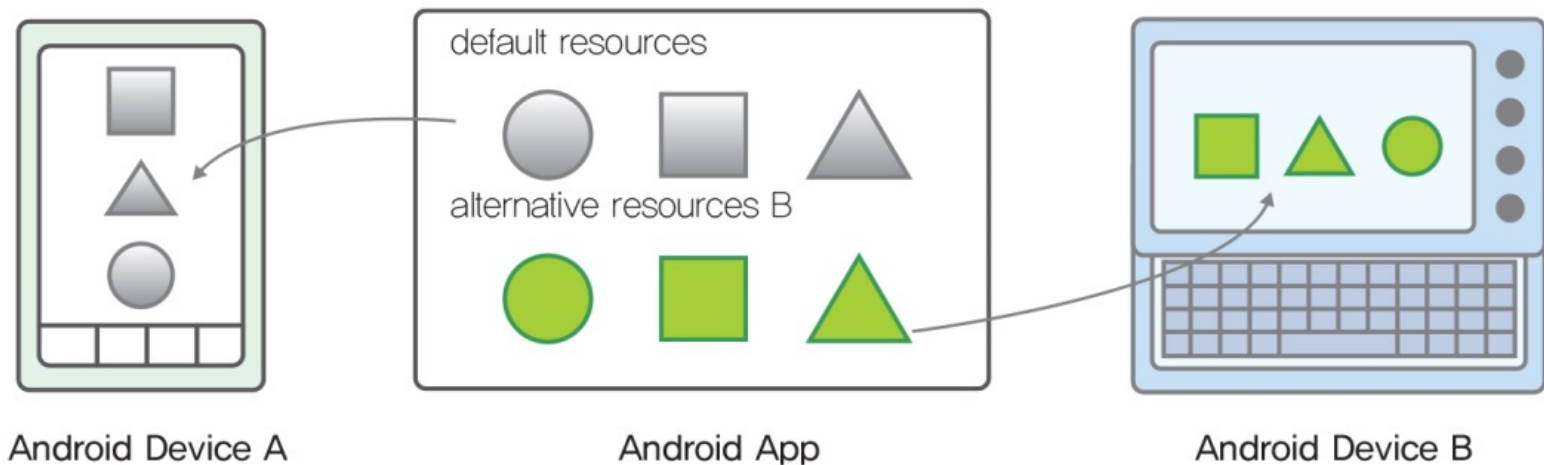
리소스

- 안드로이드에서 레이아웃, 이미지, 문자열 등을 리소스로 취급



코드와 리소스를 분리하는 이유

- 안드로이드가 탑재된 장치들이 다양해지면서 언어나 화면 크기에 따라서, 리소스를 다르게 하는 것이 필요



현재 실행되는 스마트폰에 따라 서로 다른 리소스가 선택되어 사용된다
(출처 : developer.android.com)

문자열 리소스

- 문자열도 XML로 기술하는 것이 바람직하다.
- 영어 버전, 한국어 버전, ...

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/textview"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello_world" />
```

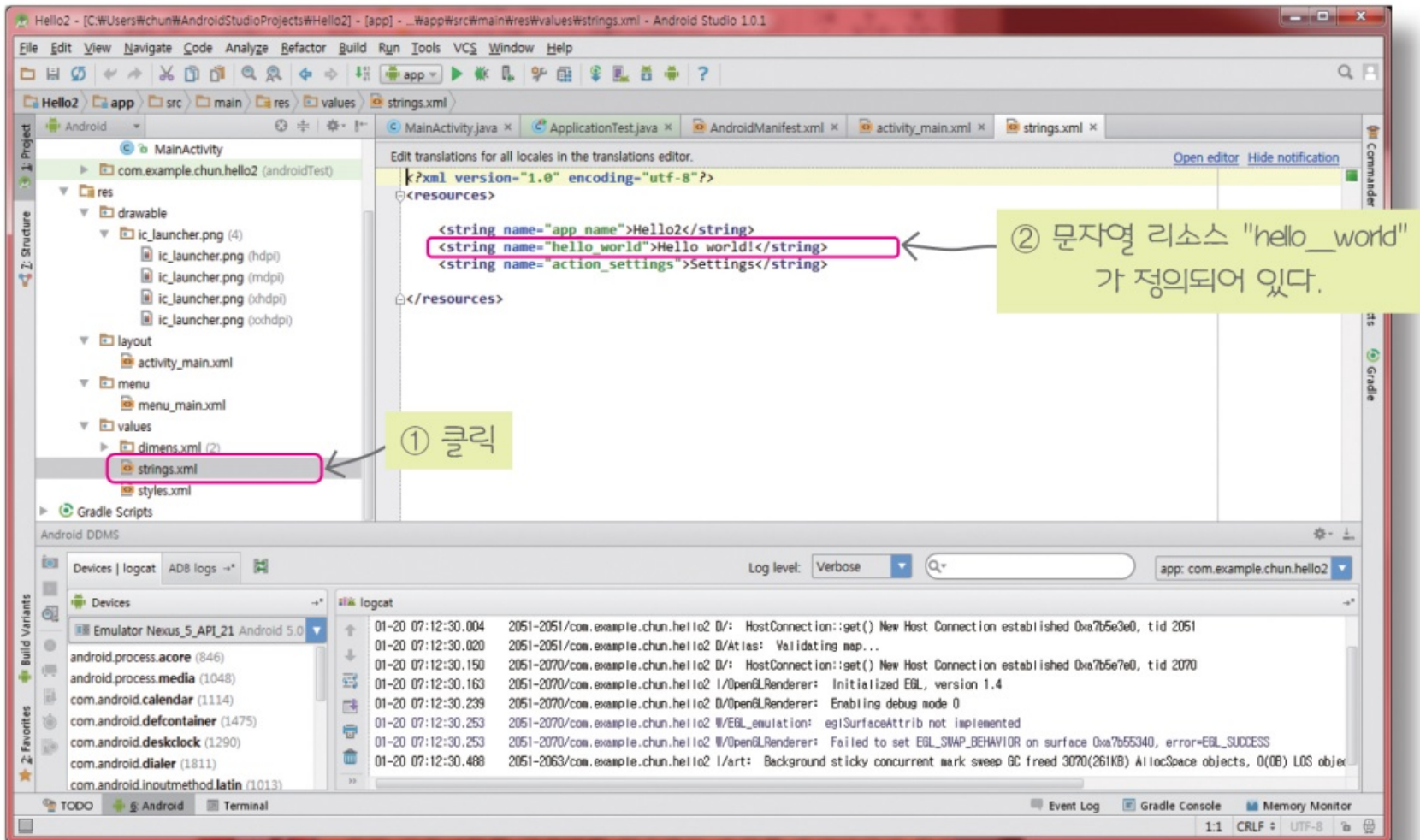
values-ko/strings.xml

```
<resources>
    <string name="app_name">첫번째 프로그램</string>
    <string name="hello_world">안녕하세요!</string>
    <string name="menu_settings">설정</string>
    <string name="title_activity_main">첫번째 액티비티</string>
</resources>
```

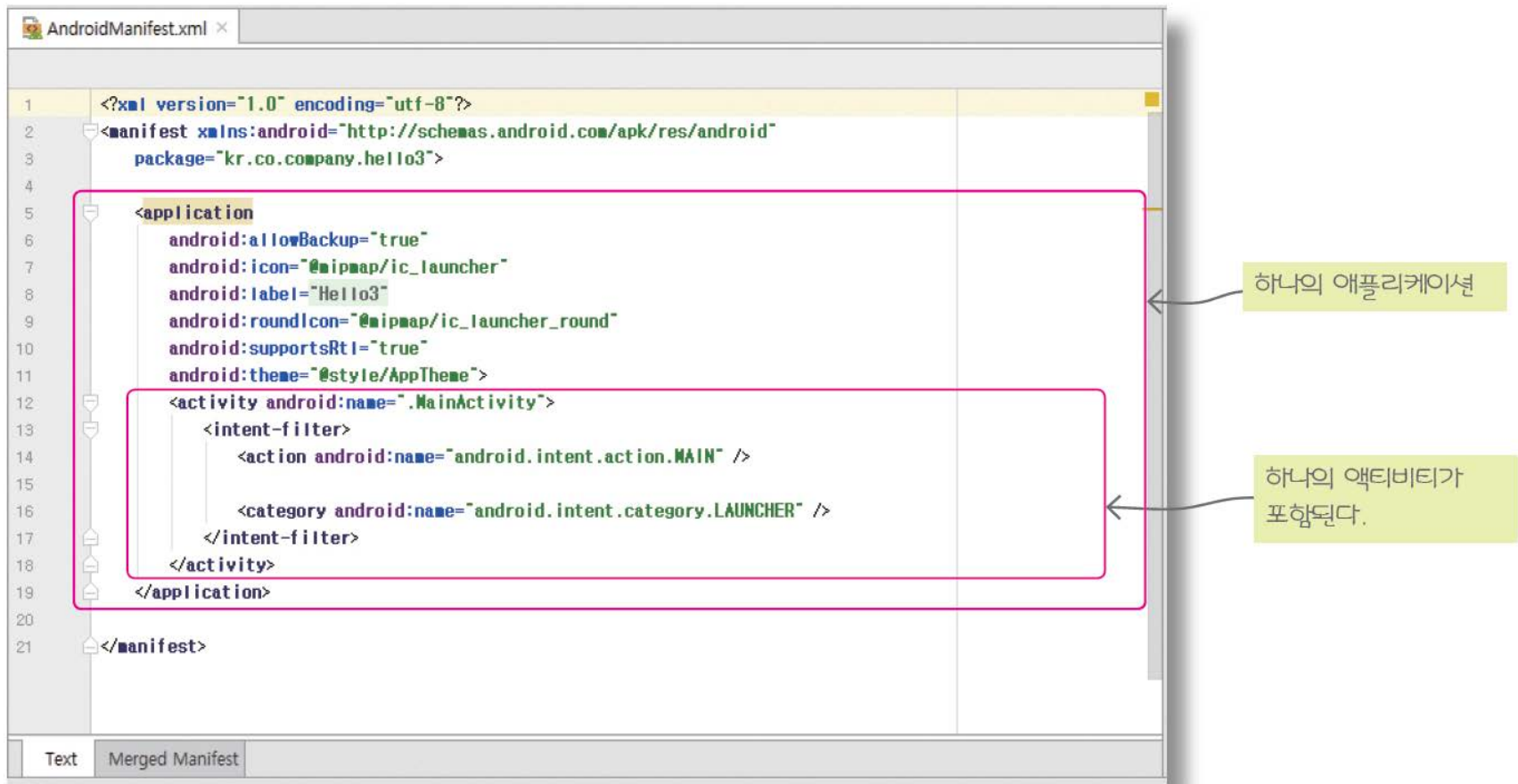
values/strings.xml

```
<resources>
    <string name="app_name">First program</string>
    <string name="hello_world">Hello!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">First activity</string>
</resources>
```

문자열 리소스



매니페스트 파일



The screenshot shows an IDE window titled "AndroidManifest.xml" with a line number gutter on the left. The XML code is as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="kr.co.company.hello3">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="Hello3"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportRtl="true"
11        android:theme="@style/AppTheme">
12
13         <activity android:name=".MainActivity">
14             <intent-filter>
15                 <action android:name="android.intent.action.MAIN" />
16
17                 <category android:name="android.intent.category.LAUNCHER" />
18             </intent-filter>
19         </activity>
20     </application>
21 </manifest>
```

Annotations on the right side of the image:

- A yellow box containing the text "하나의 애플리케이션" (One application) with an arrow pointing to the `<application>` tag and its contents.
- A yellow box containing the text "하나의 액티비티가 포함된다." (One activity is included.) with an arrow pointing to the `<activity>` tag and its contents.

At the bottom of the IDE window, there are two tabs: "Text" and "Merged Manifest".

매니페스트 파일의 분석

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.chun.hello2" >

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="Hello2"
        android:theme="@style/AppTheme" >

        <activity
            android:name=".MainActivity"
            android:label="Hello2" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

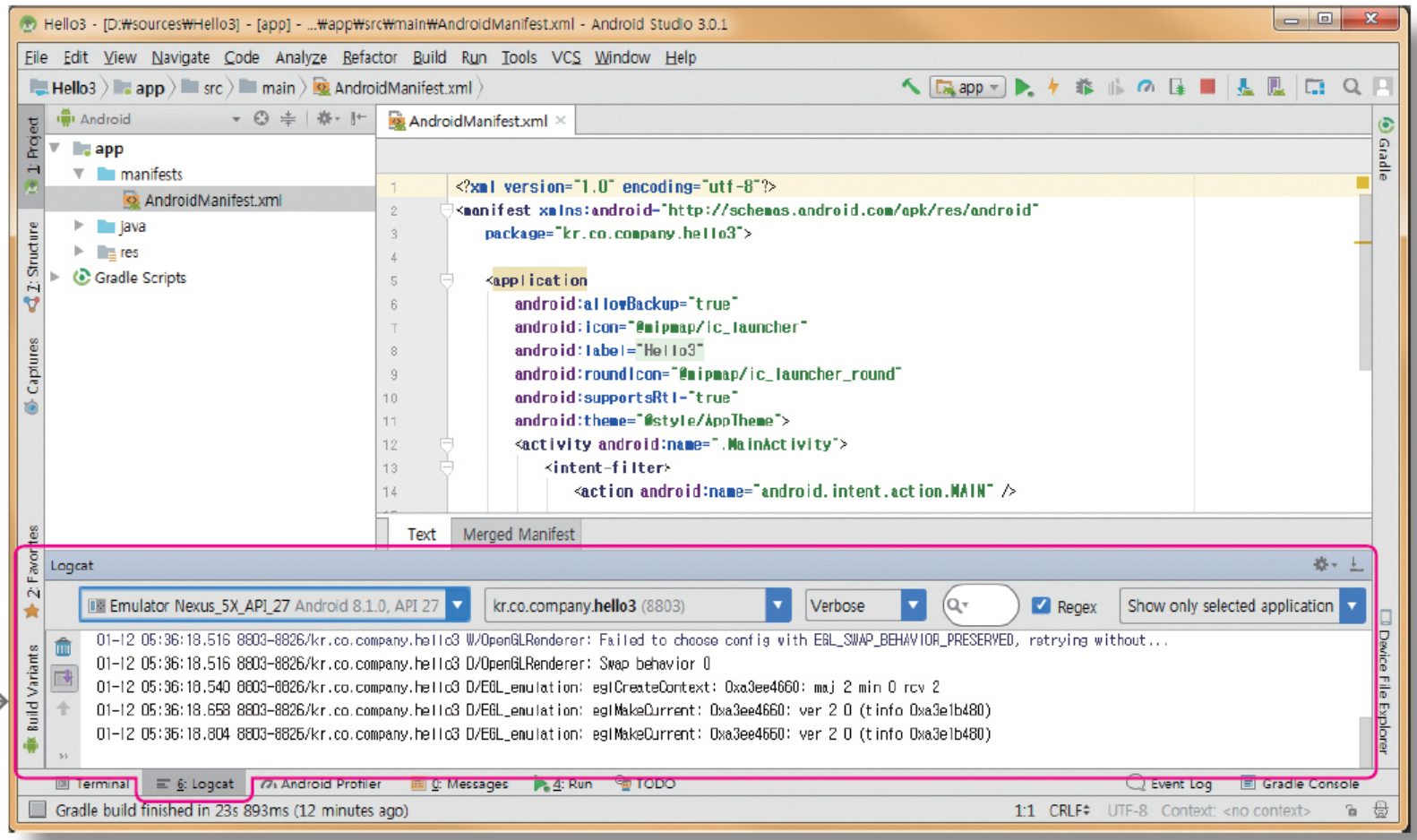
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

← 하나의 애플리케이션

← 하나의 액티비티가 포함된다.

에뮬레이터 로그캣



실제 장치와의 연결

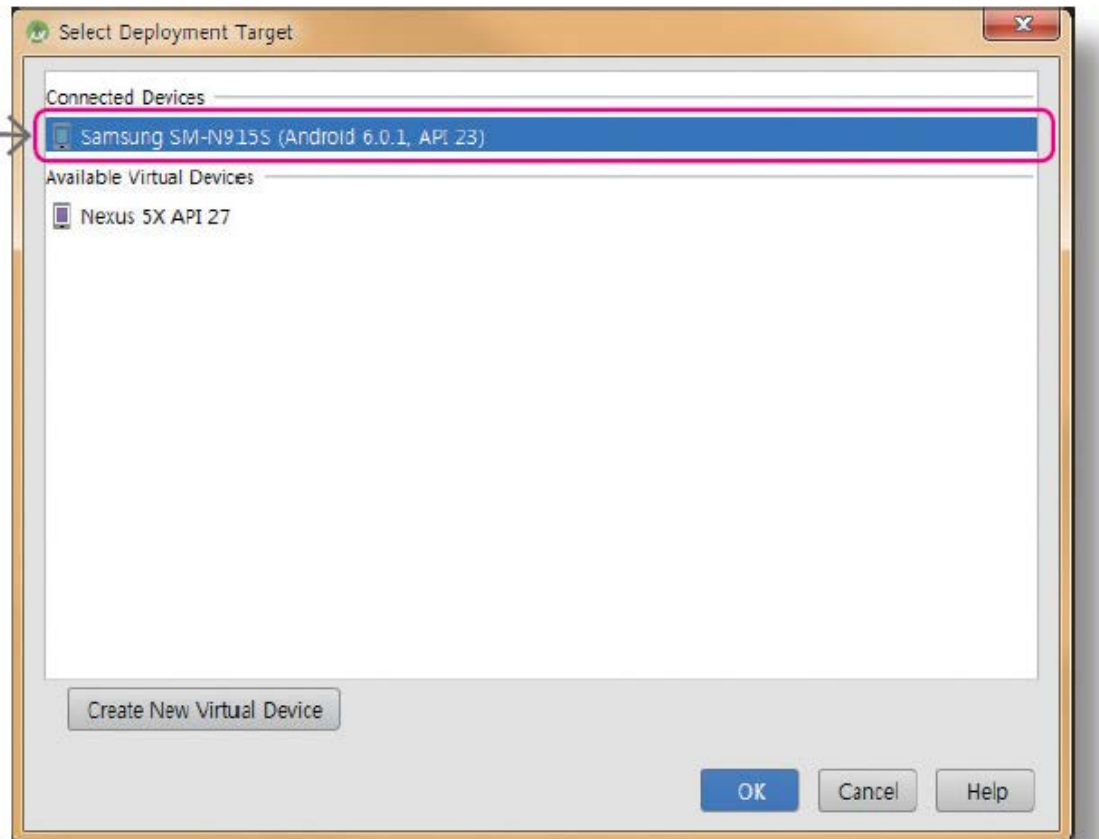
1. USB 드라이버 설치
2. 장치에서 "USB 디버깅"을 항목을 켜다.
3. 안드로이드 스튜디오에서 실행하기를 원하는 장치 선택

장치에서 USB 디버깅 켜기

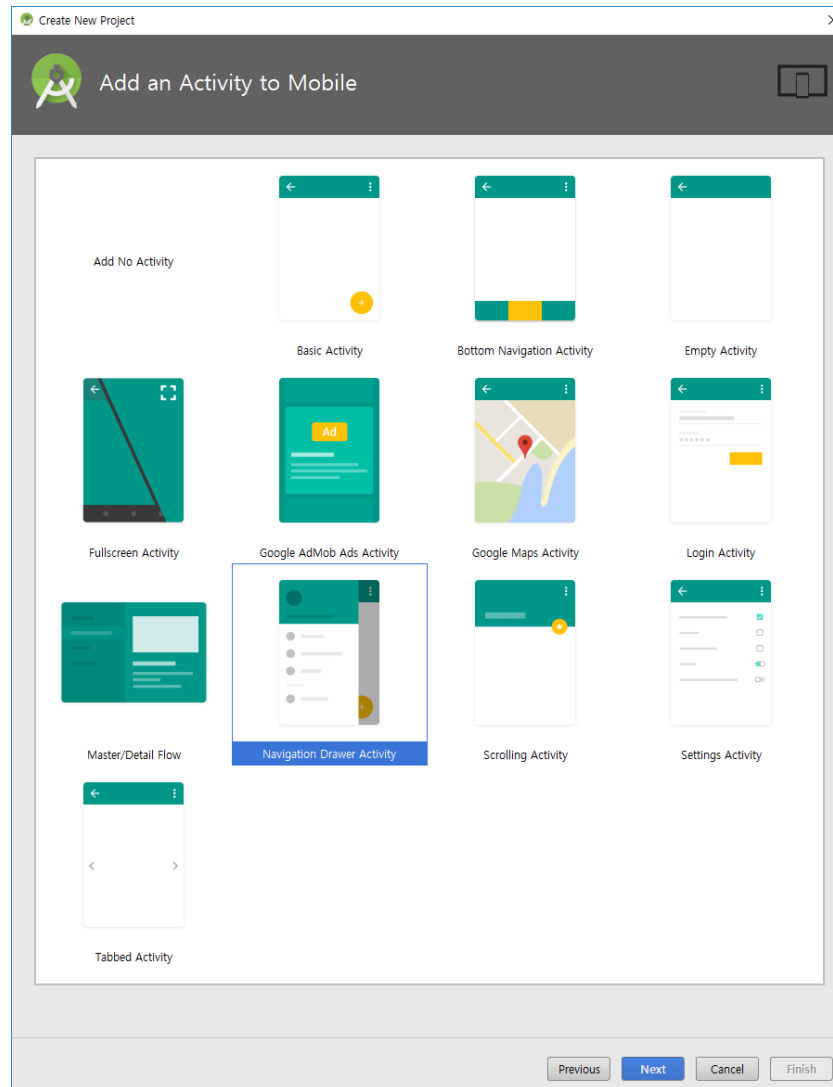


실행할 장치 선택 화면

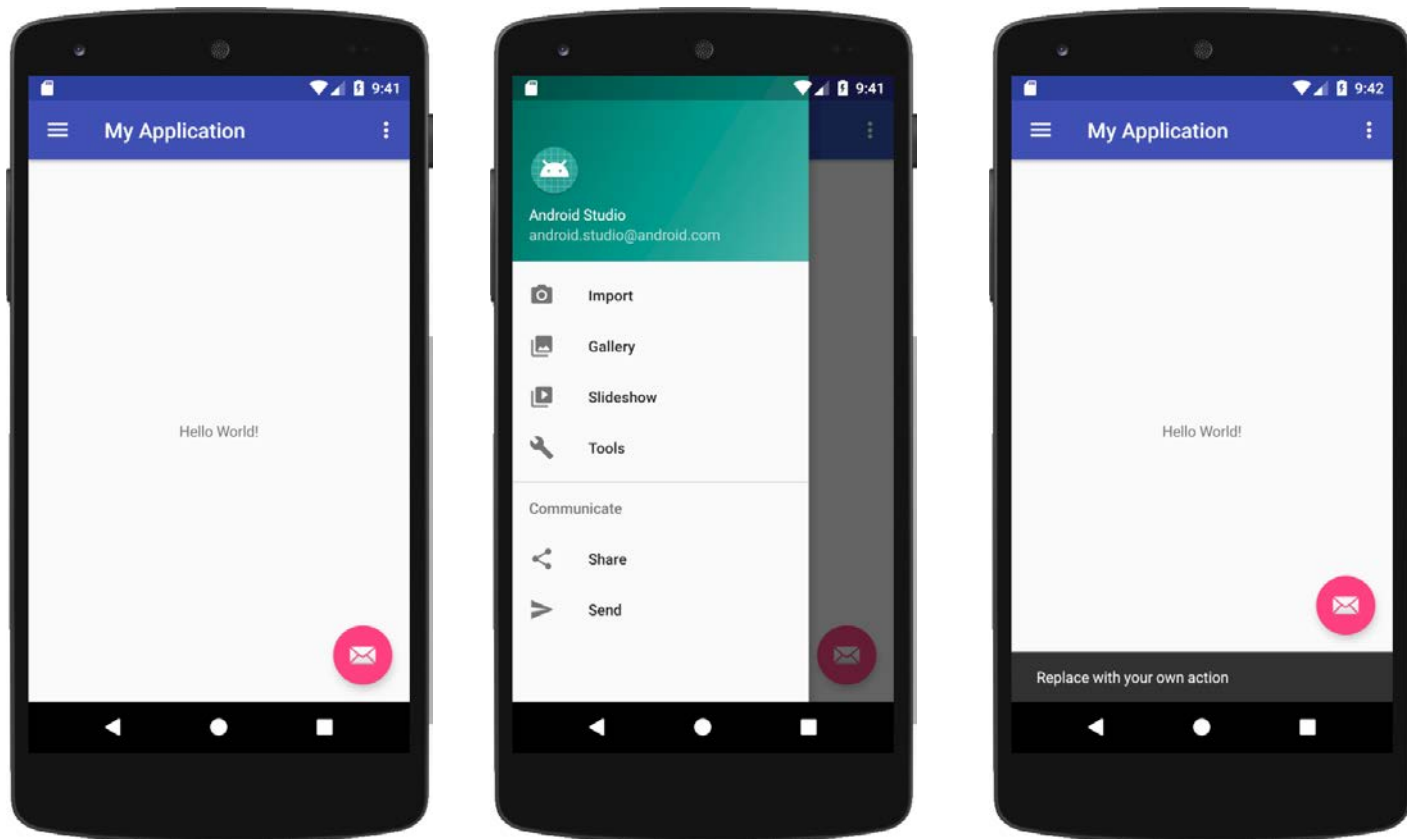
실제 장치를 선택한다.



여러 가지 형식의 앱



실행 결과



Summary

- 애플리케이션은 **컴포넌트**들의 조합으로 만들어진다.
- **코드**와 **리소스**는 철저하게 분리된다.
- 코드와 리소스는 개발 도구에 의하여 자동으로 생성되는 **R.java**에 의하여 서로 연결된다.