

2. 첫번째 안드로이드 애플리케이션

1. 처음 만드는 [Hello Android] 프로그램
2. AVD 명칭과 사용법
3. 완전한 기능의 안드로이드 애플리케이션 작성
4. 안드로이드 프로젝트의 구성



1. [Hello Android] 프로그램

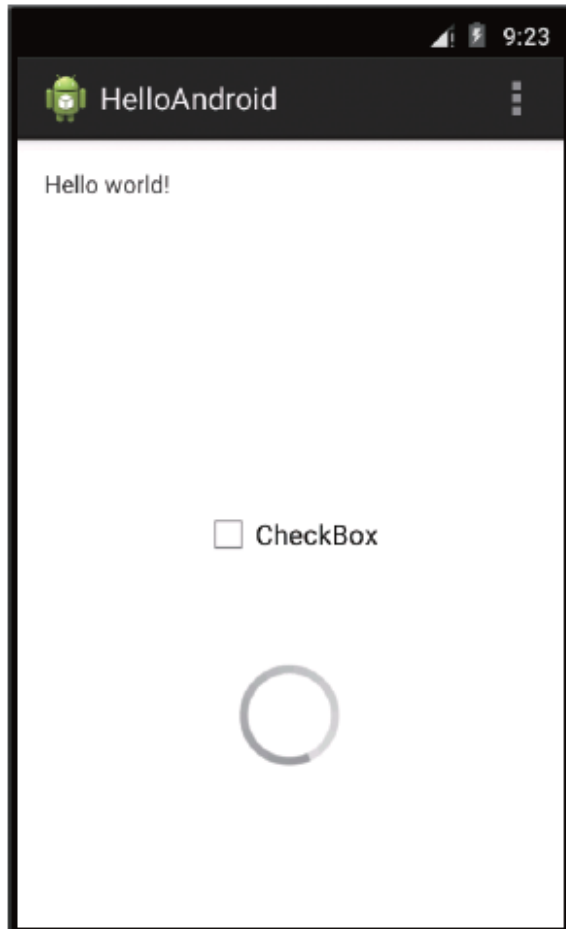


그림 2-1 처음 만든 안드로이드 애플리케이션

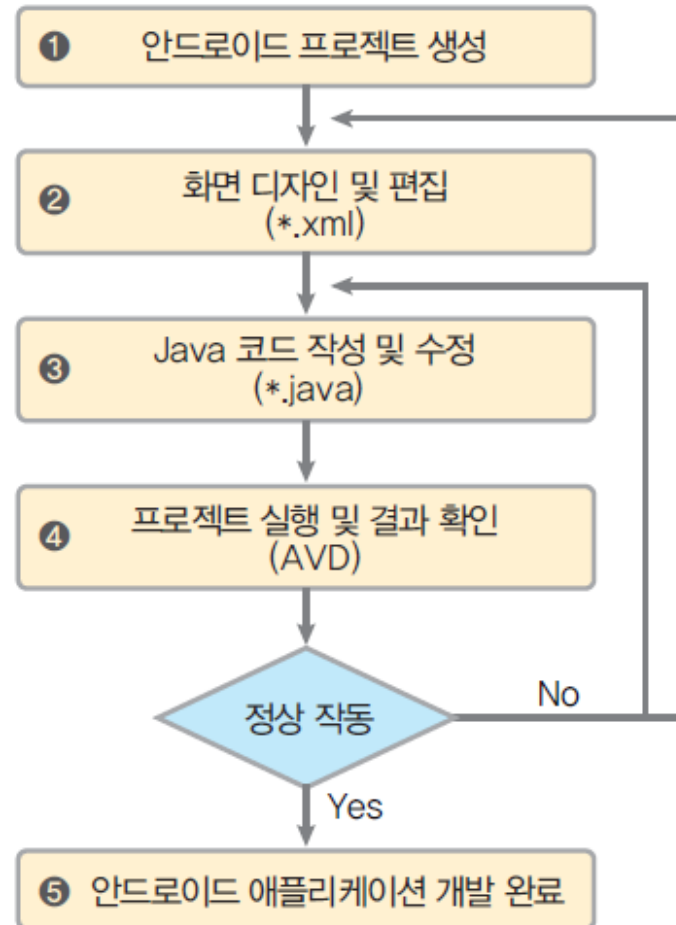


그림 2-2 안드로이드 프로젝트 개발 단계

1. [Hello Android] 프로그램

□ 프로젝트 생성[1/6]

- ▣ Android Studio를 실행한 후 [Start a new Android Studio project]를 클릭

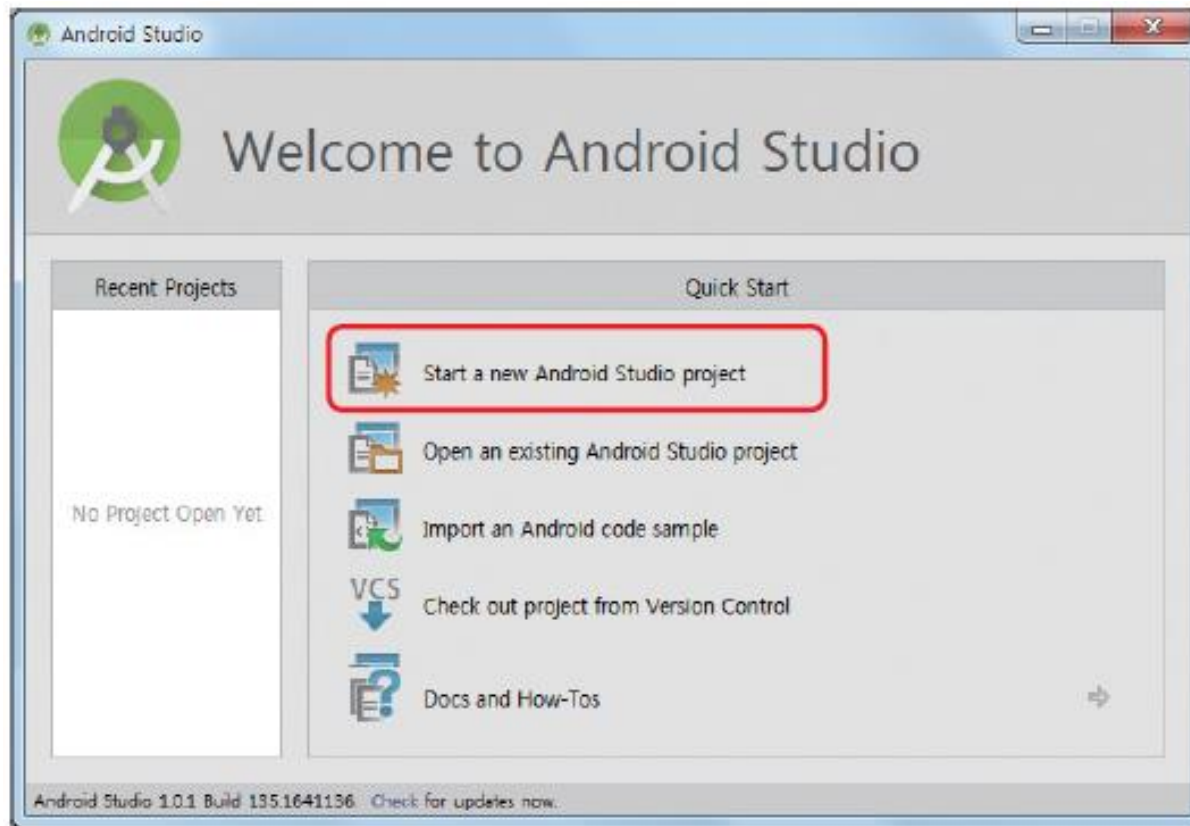


그림 2-3 새로운 안드로이드 프로젝트

1. [Hello Android] 프로그램

□ 프로젝트 생성[2/6]

- [New Project] 창의 [Configure your new project]에서 프로젝트 정보 입력

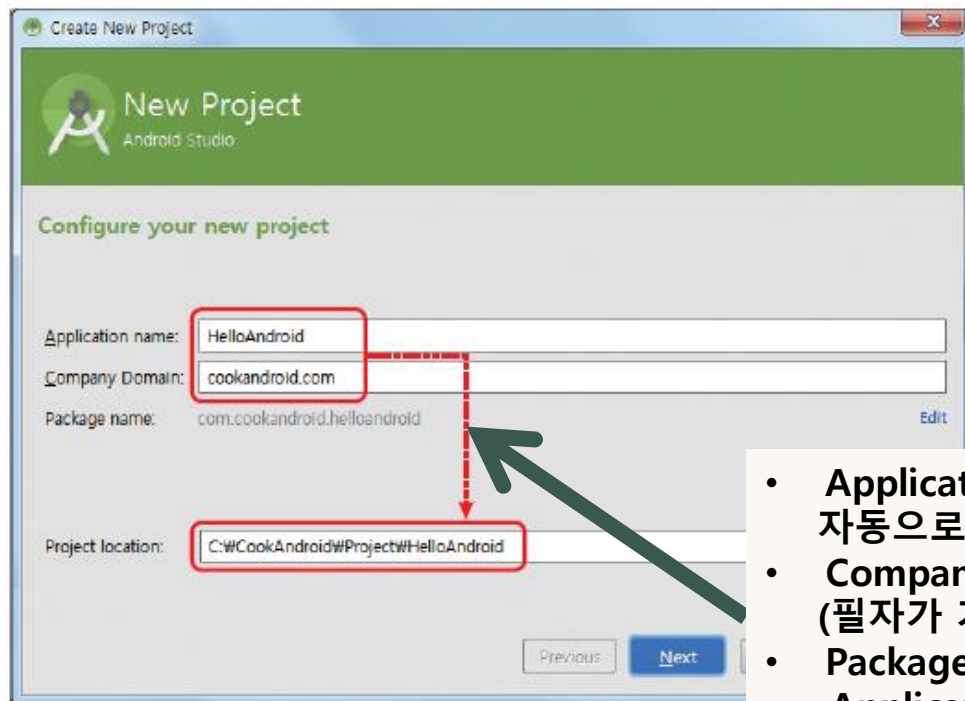


그림 2-4 프로젝트 정보 입력 1

- Application Name : Project Name은 자동으로 Application Name과 동일하게 설정
- Company Domain : 회사의 도메인 (필자가 가정한 가상 주소)
- Package Name : 자동 생성되며 도메인 이름과 Application 이름이 반대로 이어짐
- Project location : 적당한 폴더 지정(한글 불가능)

1. [Hello Android] 프로그램

□ 프로젝트 생성[3/6]

- [Select the form factors your app will run on]에서 앱을 실행한 환경을 선택
 - ‘Phone and Tablet’ 체크

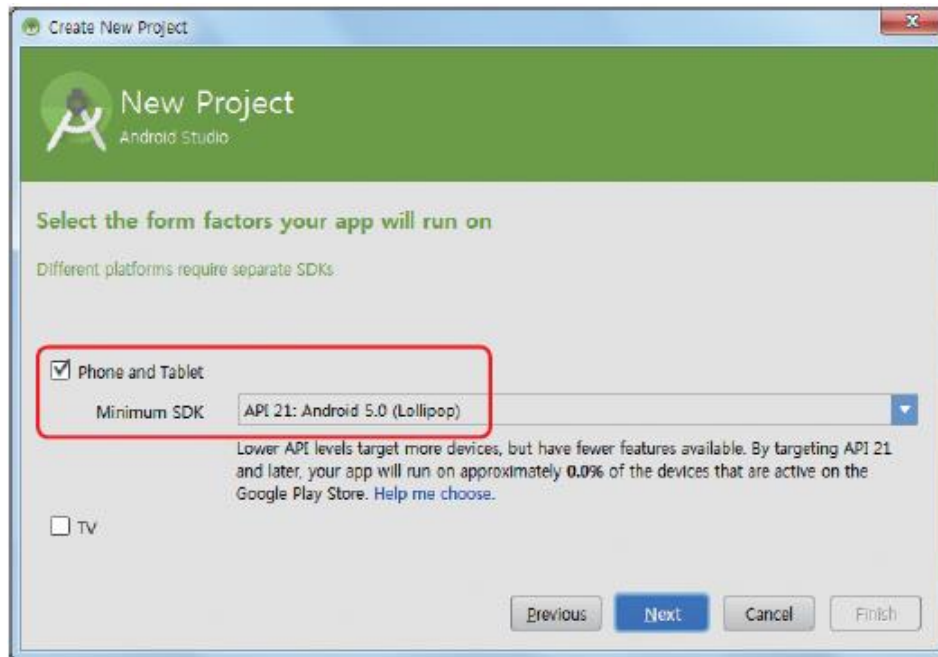


그림 2-5 프로젝트 정보 입력 2

1. [Hello Android] 프로그램

□ 프로젝트 생성[4/6]

- [Add an activity to Mobile] 창에서 디폴트인 Blank Activity 선택

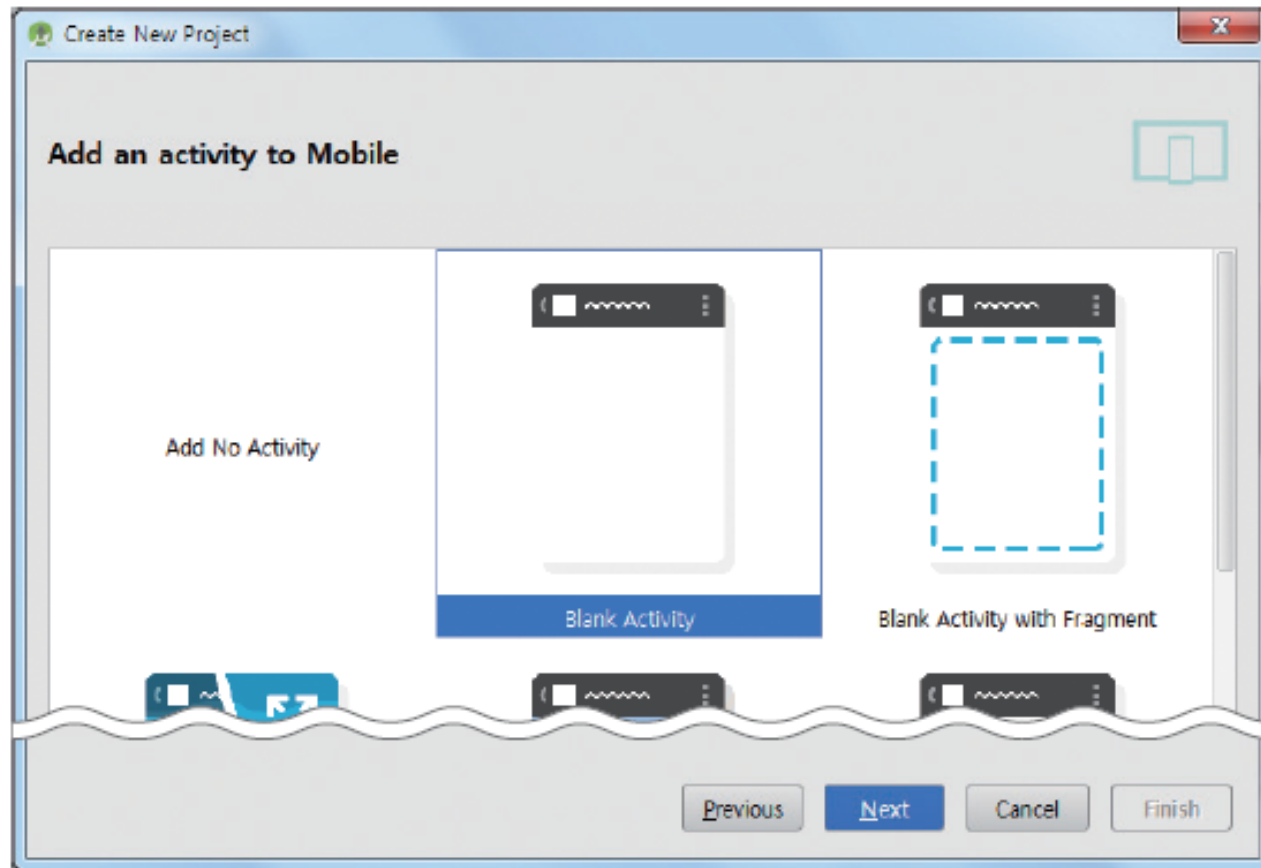
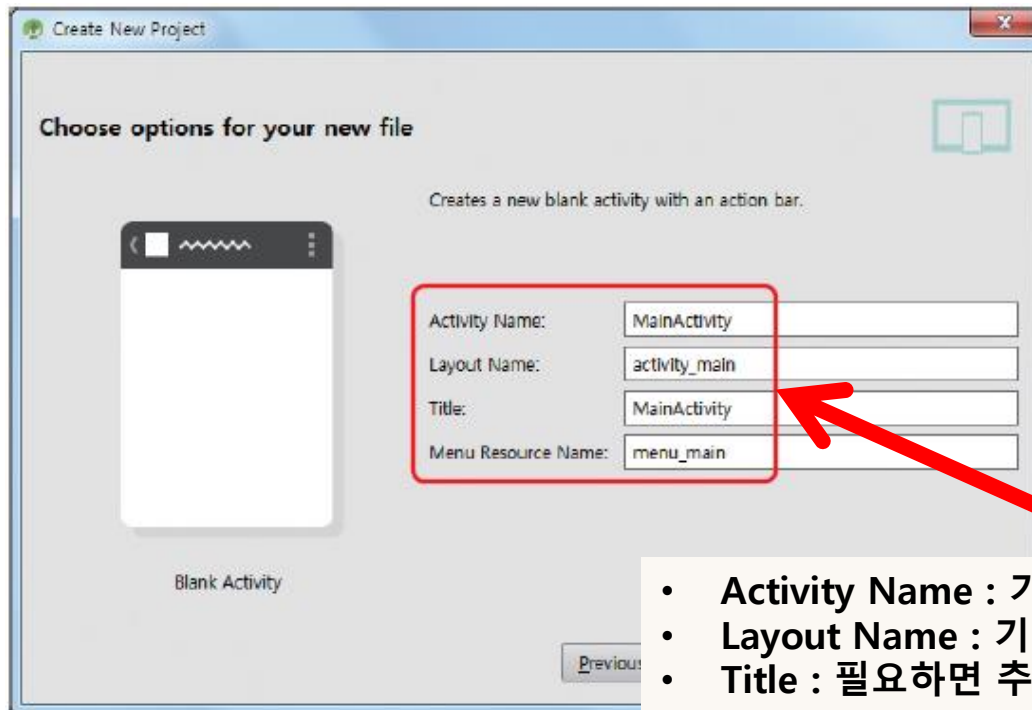


그림 2-6 액티비티 선택 화면

1. [Hello Android] 프로그램

□ 프로젝트 생성[5/6]

- [Choose options for your new file] 창도 디폴트로 두고 <Finish>를 클릭



- **Activity Name** : 기본 소스인 Java 파일 이름으로 지정
- **Layout Name** : 기본 화면인 XML 파일 이름으로 지정
- **Title** : 필요하면 추후 바꿀 수 있음
- **Menu Resource Name** : 나중에 필요할 때 사용

그림 2-7 관련 이름 지정

1. [Hello Android] 프로그램

□ 프로젝트 생성[6/6]

- Android Studio의 왼쪽 [Project Tree]에 'HelloAndroid' 프로젝트가 추가됨

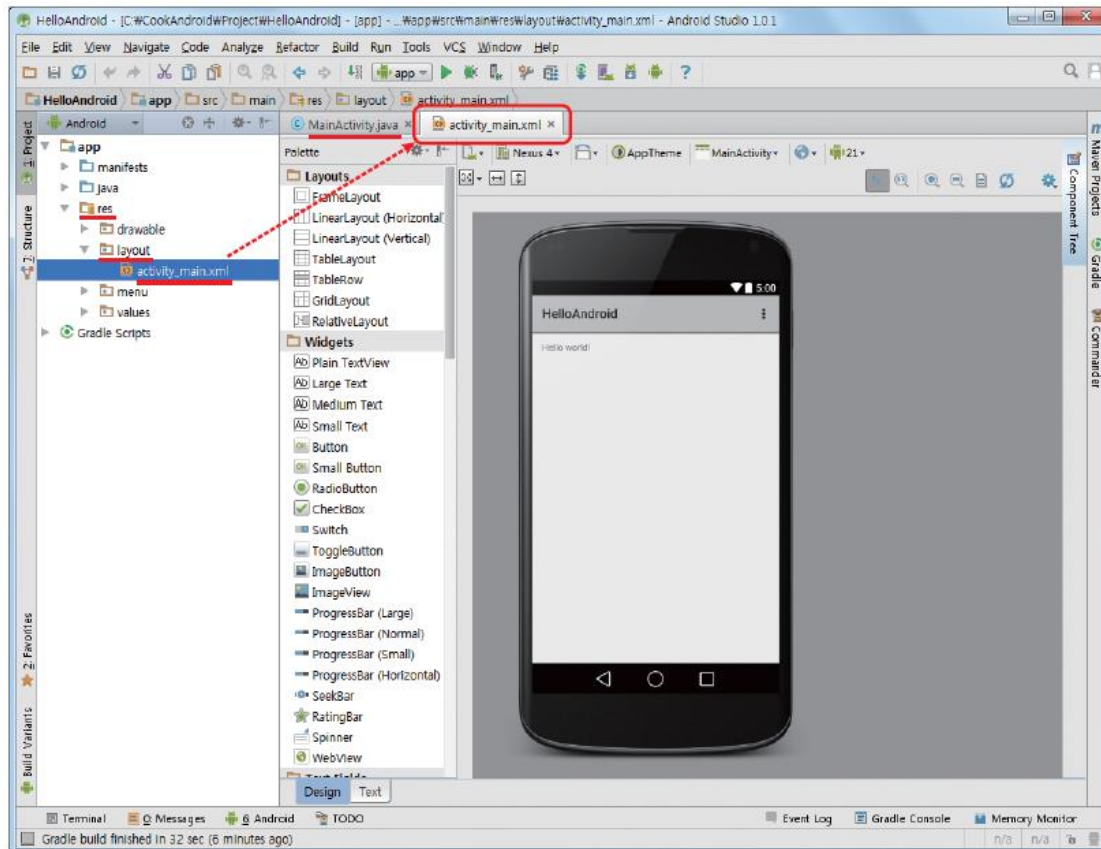
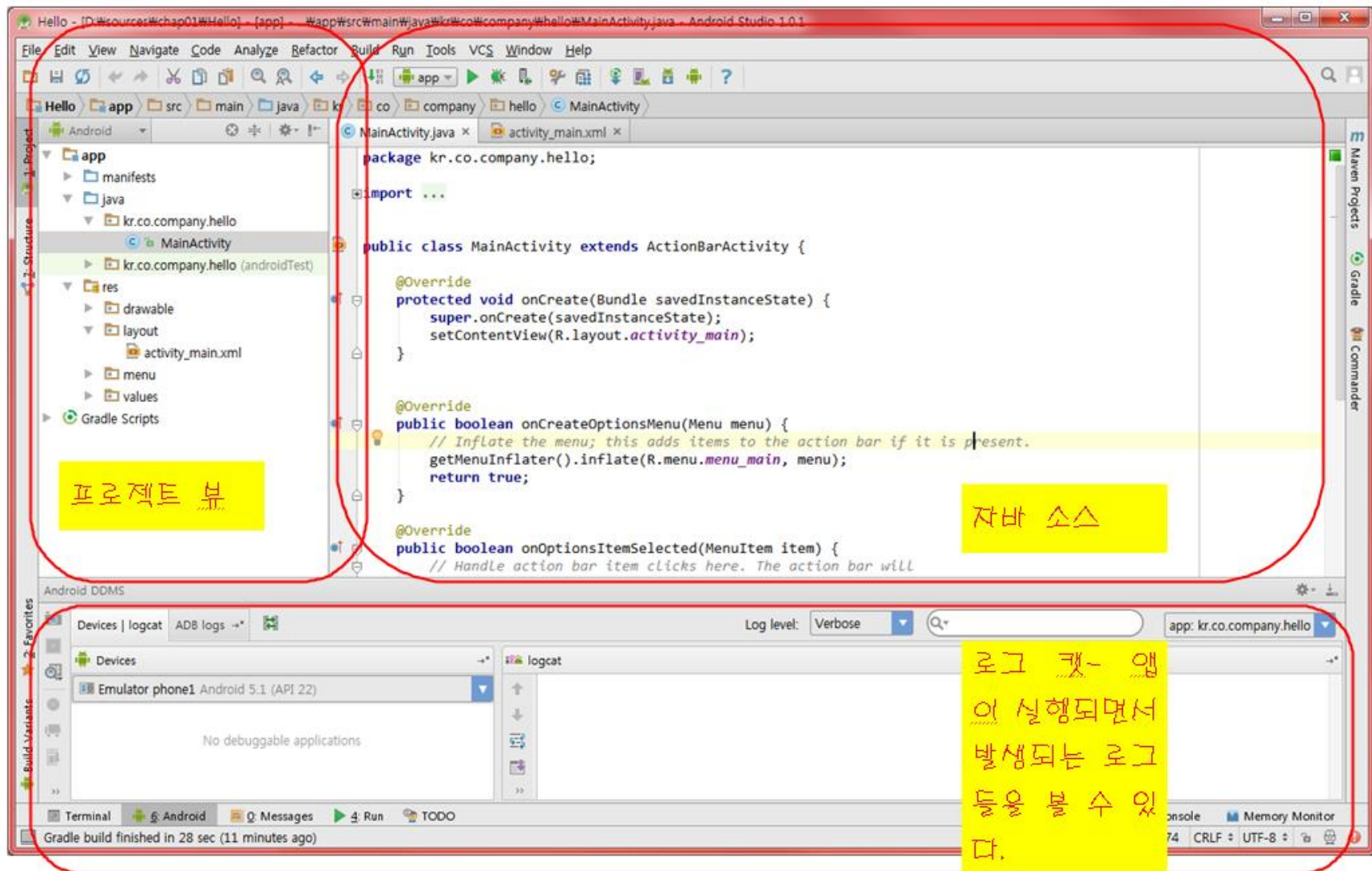


그림 2-8 안드로이드 프로젝트 생성 결과

애플리케이션의 구성



패키지 폴더의 설명

표 1, 프로젝트 뷰의 폴더

폴더 또는 파일	설명
java	소스 파일들이 들어있는 폴더이다. 폴더 안의 <u>kr.co.company.hello</u> 는 패키지의 이름이다.
Gradle Scripts	그레이들(Gradle)은 빌드 시에 필요한 스크립트이다.
res	각종 리소스(자원)들이 저장되는 폴더이다. drawable 에는 해상도 별로 아이콘 파일들이 저장된다. layout 에는 화면의 구성을 정의한다. values 에는 문자열과 같은 리소스가 저장된다. menu 에는 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;  
import android.view.Menu;  
import android.view.MenuItem;
```

필요한 클래스를 포함시키는 import 문장들

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

자동 생성된 소스 관찰

```
@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);
}
```

MainActivity 클래스 정의

자동 생성된 소스 관찰

□ **package com.android.hello**

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

□ **import android.support.v7.app.ActionBar...**

- ▣ import 문장은 외부에서 패키지나 클래스를 포함
- ▣ 앞에 android가 붙은 패키지는 안드로이드가 제공하는 패키지를 의미

□ **public class MainActivity extends ActionBarActivity { ... }**

- ▣ 클래스의 정의
- ▣ Activity로부터 상속받았으므로 액티비티가 된다.
- ▣ 액티비티는 안드로이드에서 애플리케이션을 구성하는 4가지의 컴포넌트 중의 하나로 화면을 나타낸다.

자동 생성된 소스 관찰

□ @Override

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

□ **public void onCreate() { ... }**

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

자동 생성된 소스 관찰

- **super.onCreate(savedInstanceState);**
 - ▣ 위의 문장은 부모 클래스인 ActionBarActivity 클래스의 onCreate()를 호출하는 문장
- **setContentView(R.layout.activity_main);**
 - ▣ setContentView()라는 함수는 액티비티의 화면을 설정하는 함수
 - ▣ R.layout.activity_main은 activity_main.xml 파일을 나타낸다.

1. [Hello Android] 프로그램

- 화면 디자인 및 편집[1/4]
 - ▣ 화면을 가상 AVD와 동일하게 설정

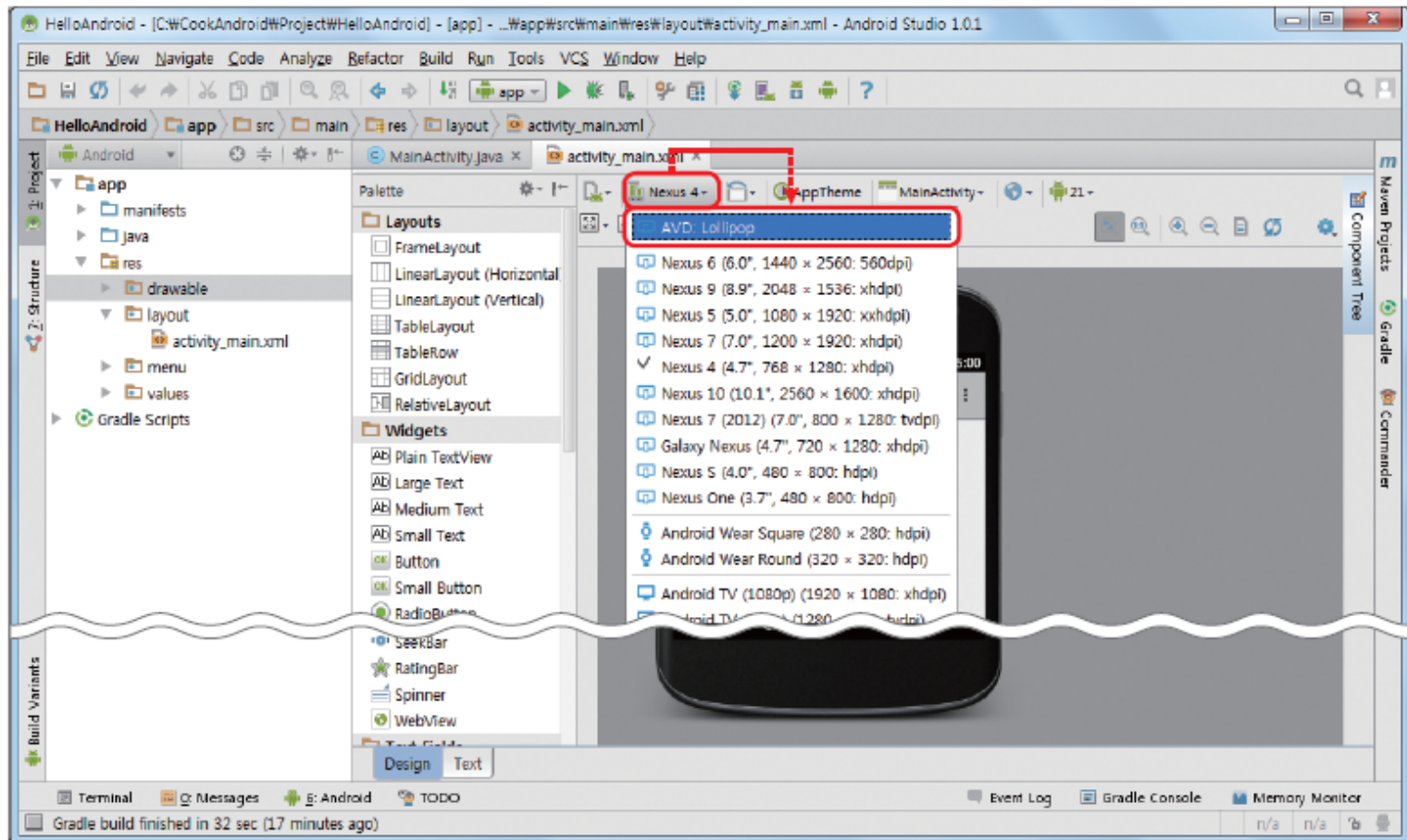


그림 2-9 디자인 환경 설정 1

1. [Hello Android] 프로그램

□ 화면 디자인 및 편집[2/4]

- ❖ 왼쪽 [Widgets]에서 몇 개를 오른쪽에 옮김
- ❖ 오른쪽 위의 확대/축소 아이콘으로 화면 크기 조절 가능

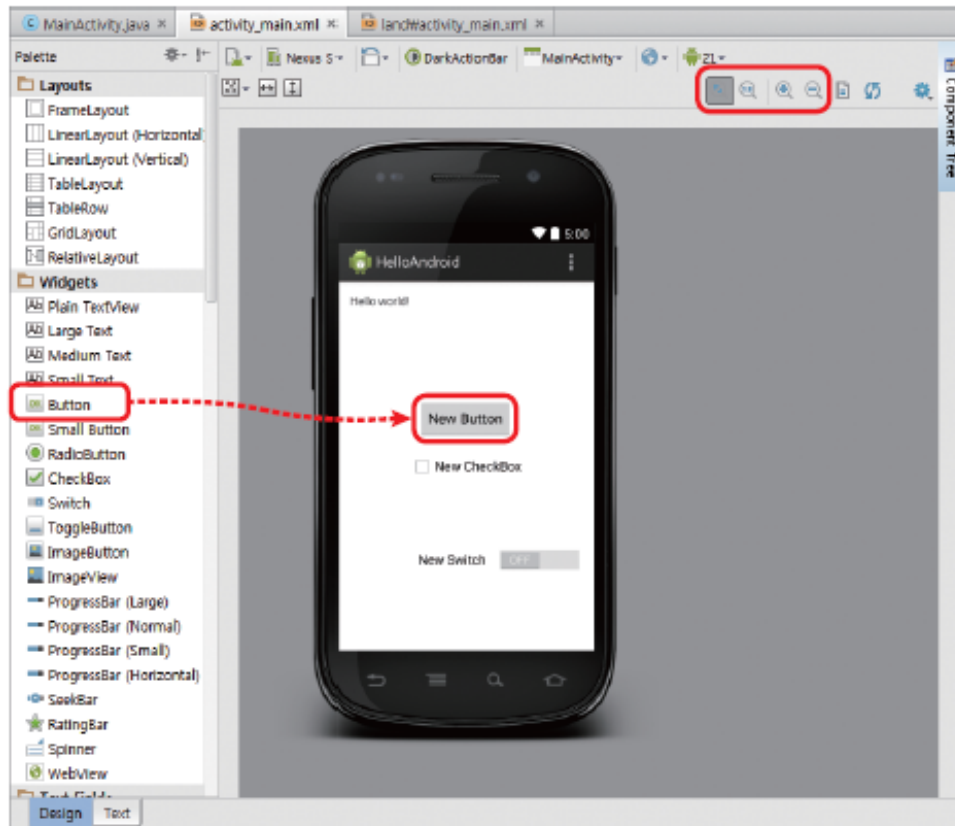


그림 2-11 그래픽 화면에서 위젯을 끌어다놓기

1. [Hello Android] 프로그램

□ 화면 디자인 및 편집[3/4]

- 왼쪽 아래 [Text] 탭을 클릭하면 화면에 xml 코드가 표시됨

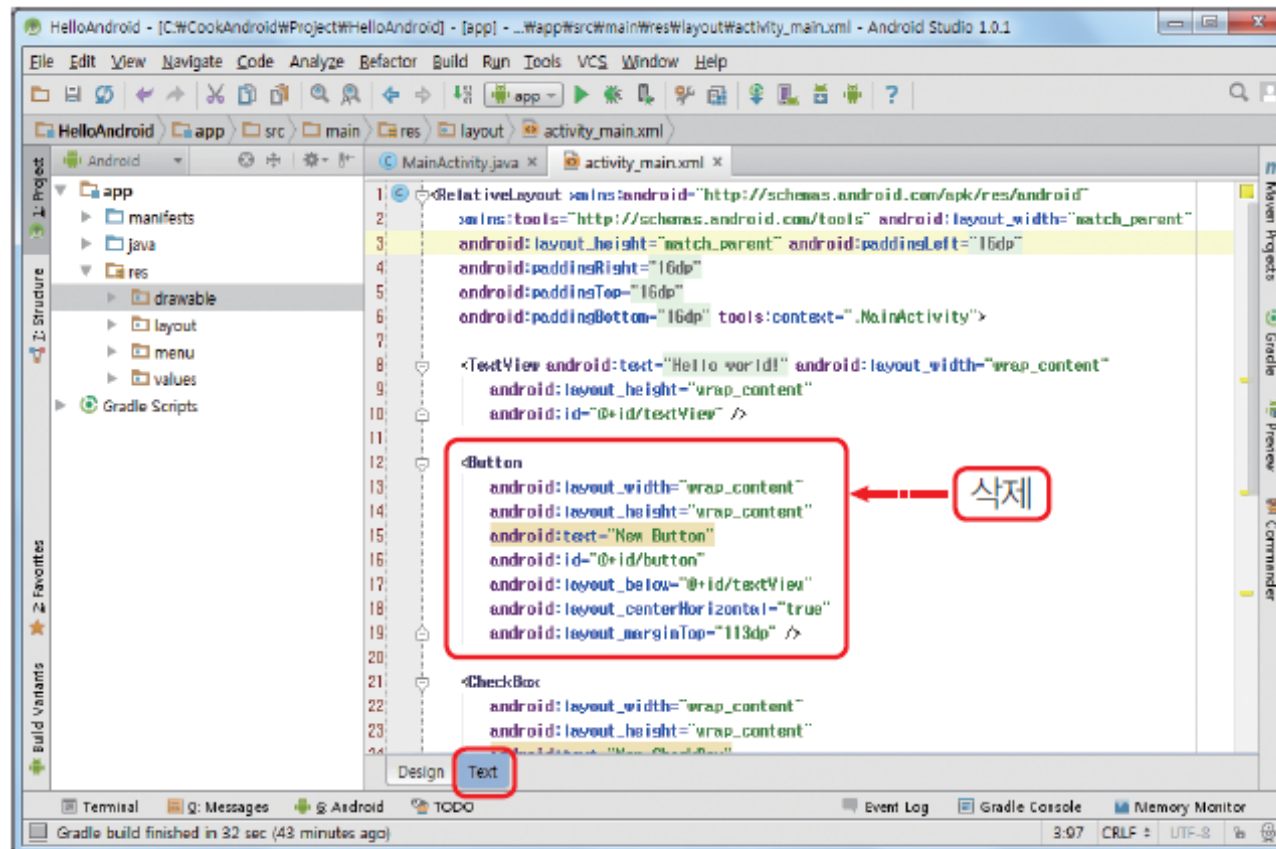


그림 2-12 XML 코드 변경

1. [Hello Android] 프로그램

□ 화면 디자인 및 편집[4/4]

- ❖ [Design]을 클릭하면 삭제한 코드가 화면에도 삭제되어 있음
- ❖ 왼쪽 상단의 저장 아이콘을 클릭하거나 메뉴 [File]–[Save All]을 클릭하여 저장

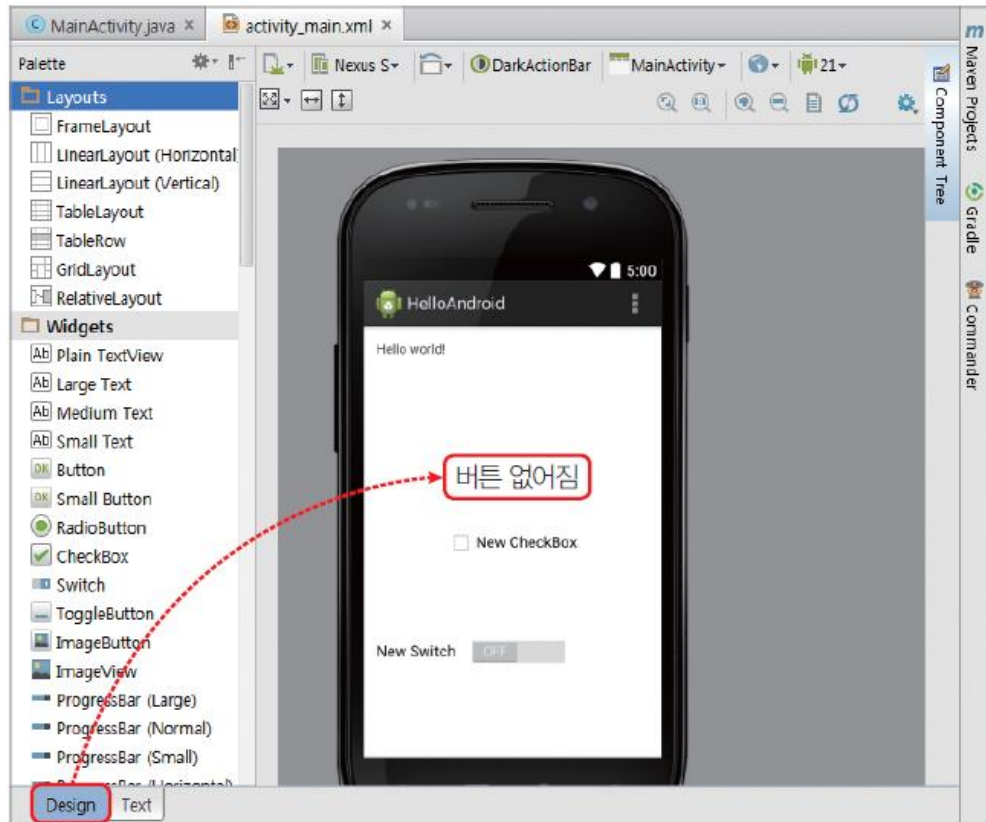


그림 2-13 그래픽 화면 확인

1. [Hello Android] 프로그램

□ JAVA 코드 작성 및 수정

■ Project Tree의 [java]-[com.android.helloandroid]-[MainActivity] 선택

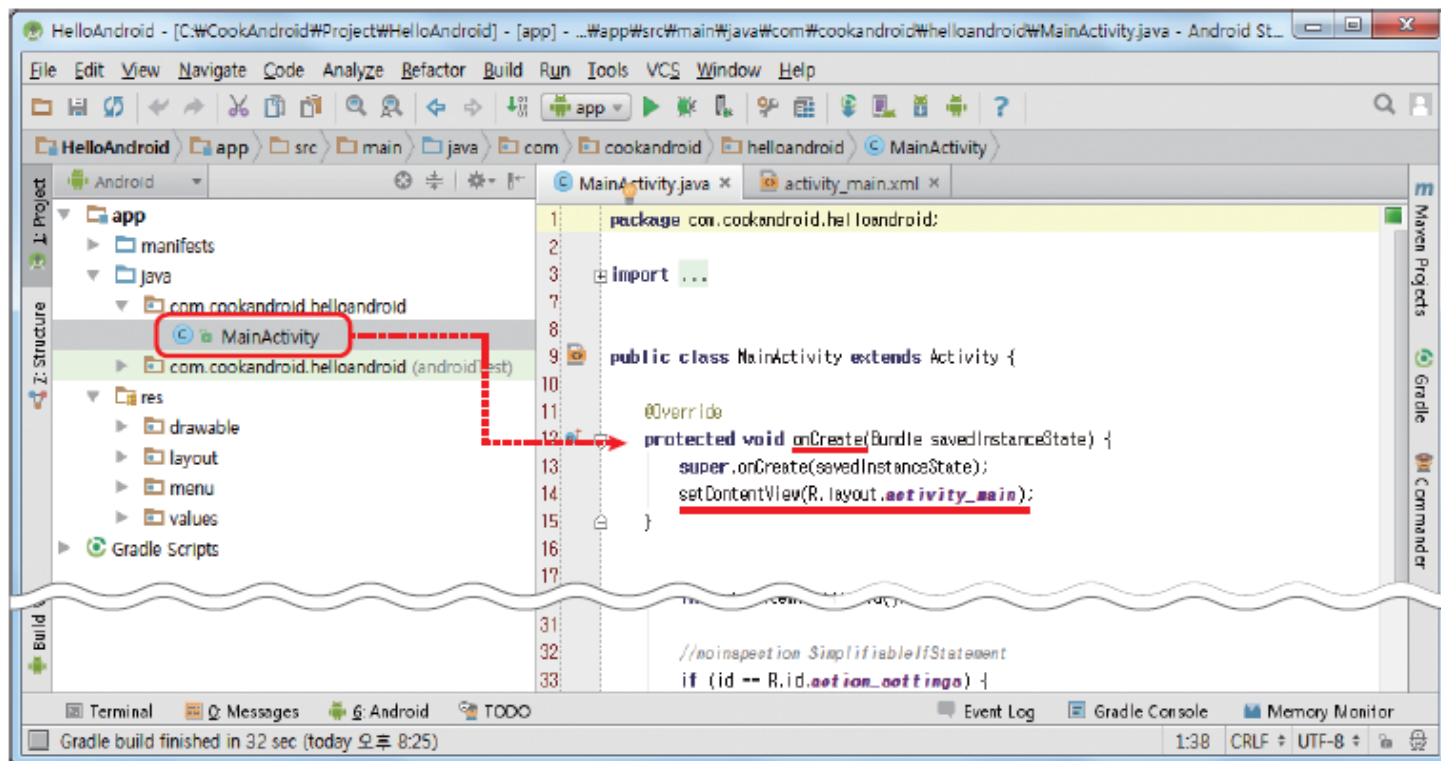


그림 2-14 메인 Java 코드 확인

1. [Hello Android] 프로그램

□ 프로젝트 실행, 확인[1/4]

- 메뉴의 [Run As]-[Run 'app']을 선택하거나 [Run 'app'] 아이콘을 클릭

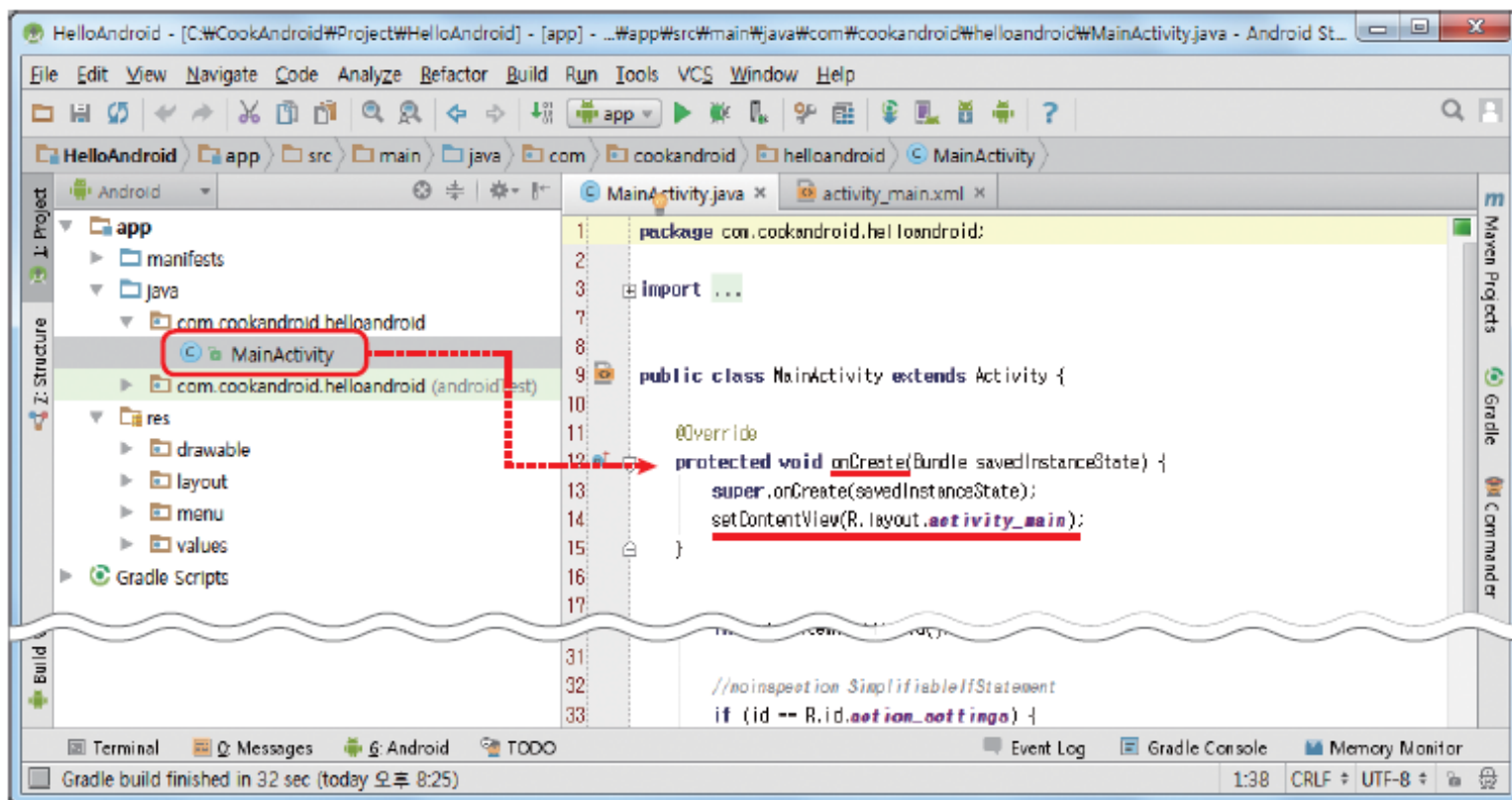


그림 2-14 메인 Java 코드 확인

1. [Hello Android] 프로그램

□ 프로젝트 실행, 확인[2/4]

- ▣ [Choose Device] 장치에서 실행할 안드로이드 기기나 AVD를 선택

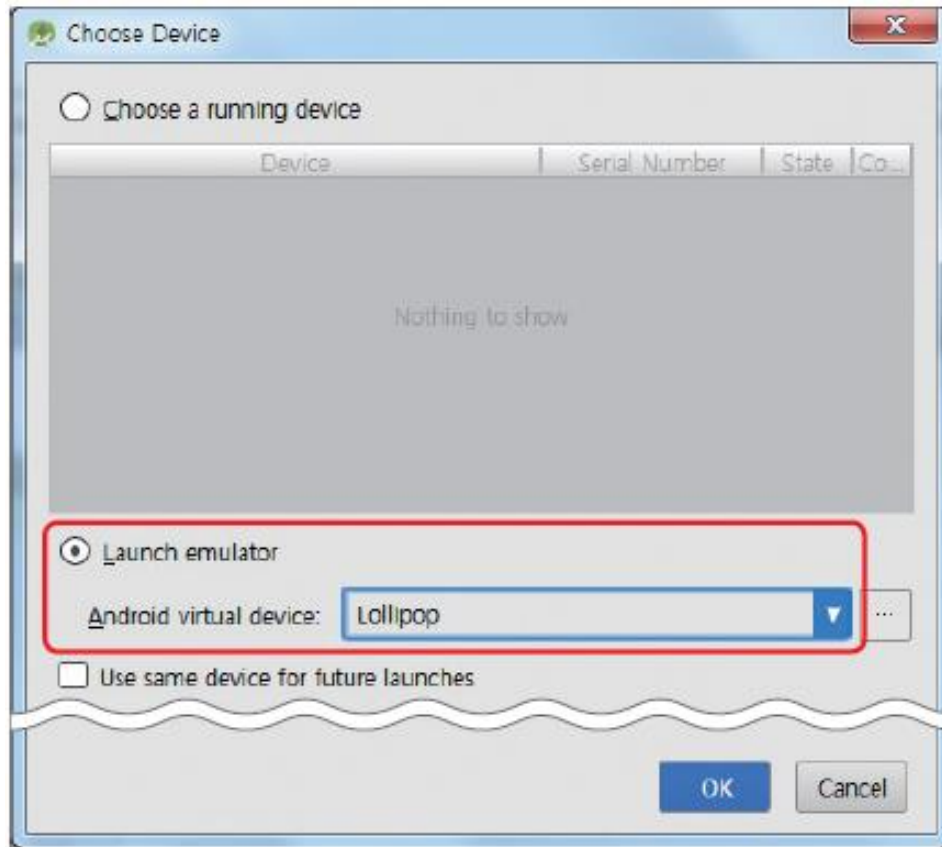


그림 2-16 실행할 안드로이드 장치 선택

1. [Hello Android] 프로그램

- 프로젝트 실행, 확인[3/4]
 - ▣ AVD가 부팅된 후 실행결과 화면이 나타남

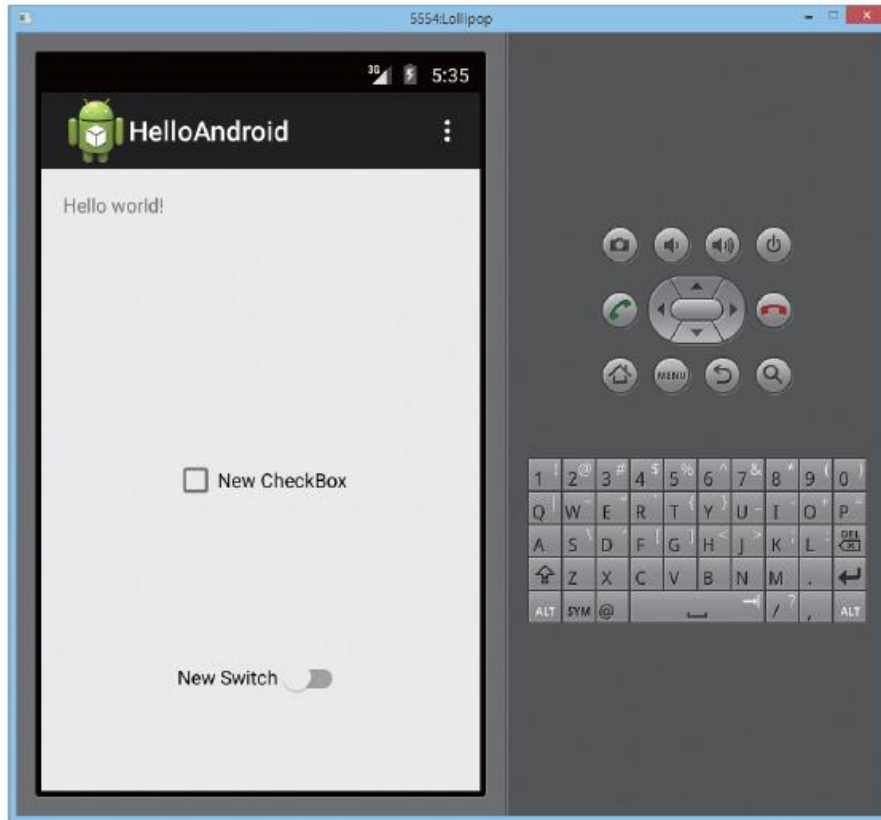


그림 2-17 프로젝트 실행 결과

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[1/6]

▣ 프로젝트 닫기와 열기

- [File]-[Close Project]를 선택해서 프로젝트 닫음
- 프로젝트 이름을 클릭하면 다시 프로젝트가 열림, 폴더 검색해서 열어도 됨



그림 2-32 최근에 사용한 프로젝트 열기 1

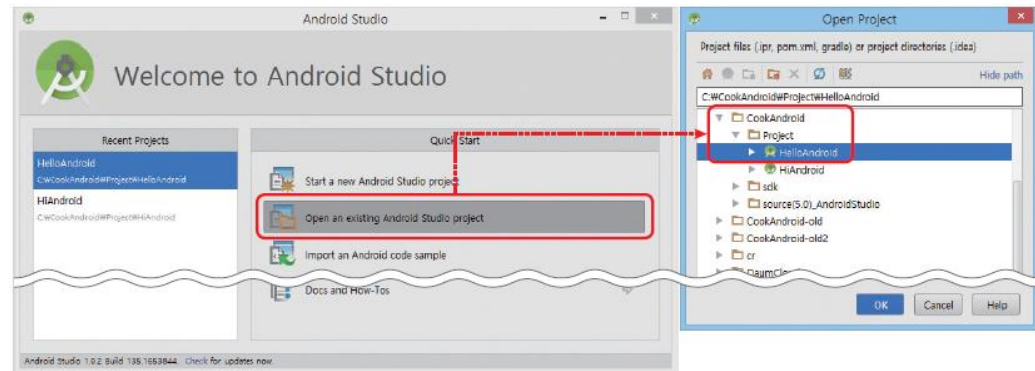


그림 2-34 폴더에서 직접 프로젝트 열기

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[2/6]

▣ 프로젝트 닫기와 열기

- 여러 개의 프로젝트를 열려면 [File]-[Reopen Project]-[프로젝트 이름] 선택
- <New Window>를 클릭하면 새로운 창이 열려서 여러 개의 프로젝트 동시 작업 가능

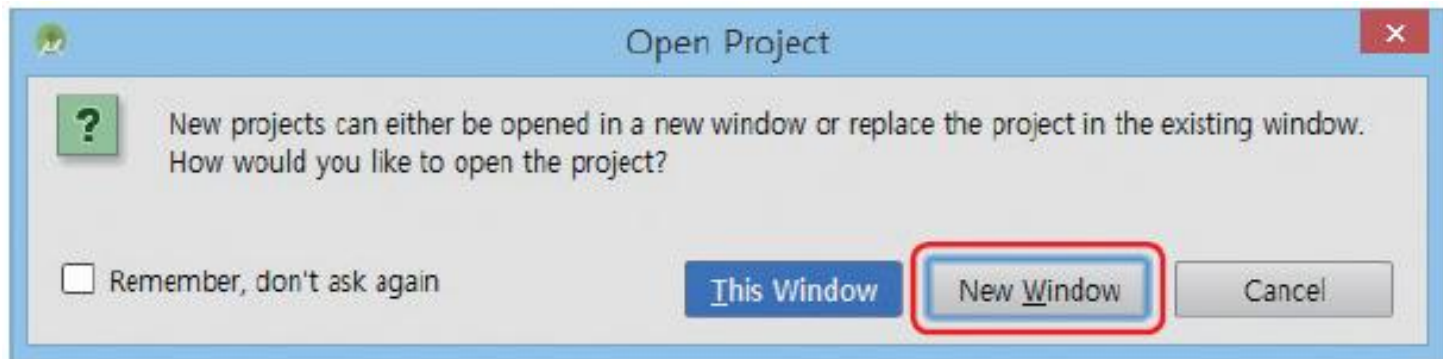


그림 2-33 최근에 사용한 프로젝트 열기 2

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[3/6]

▣ Android Studio 프로젝트 내보내기/가져오기

- 프로젝트가 생성된 폴더를 통째로 복사하거나 압축해서 보내거나 가져오면 됨

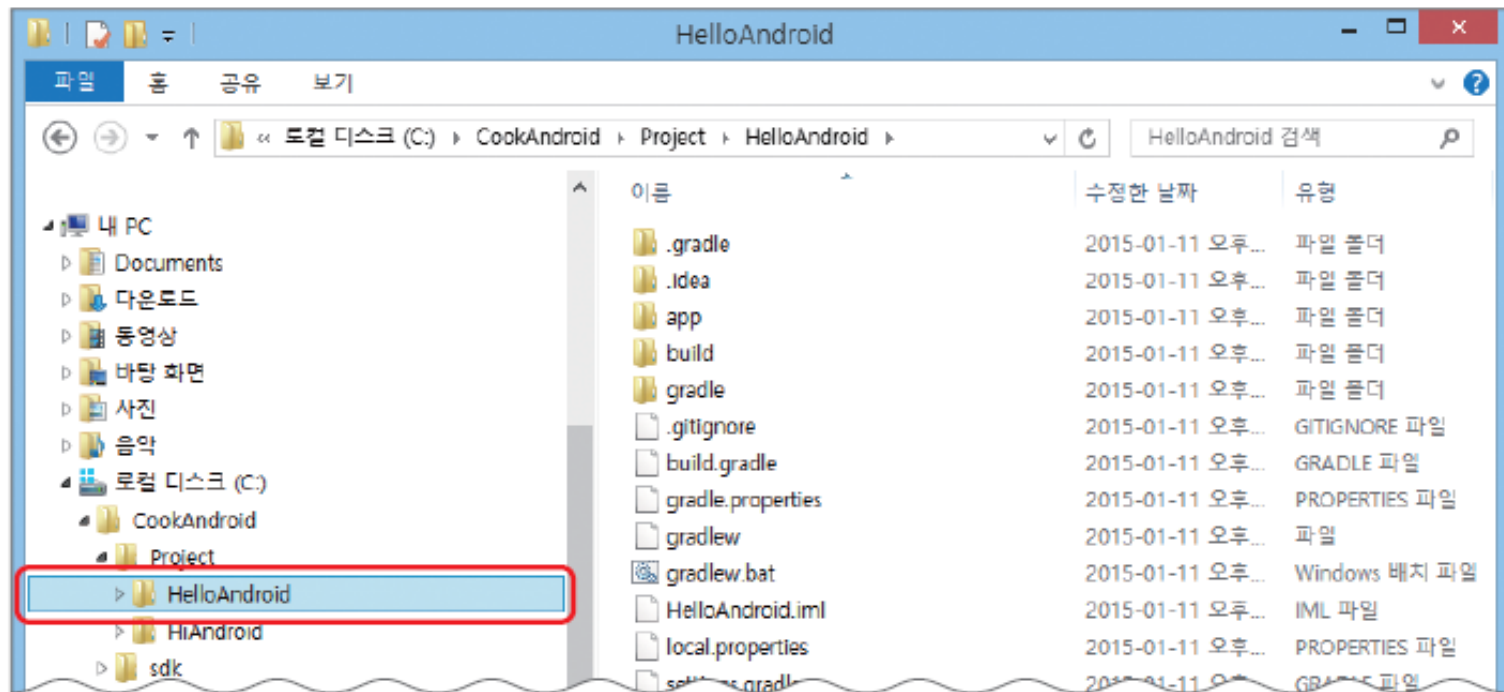


그림 2-35 프로젝트 폴더를 통째로 복사

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[4/6]

■ 이클립스용 프로젝트 가져오기

- [Import Non-Android Studio project] 선택 후 작성된 프로젝트 폴더 선택

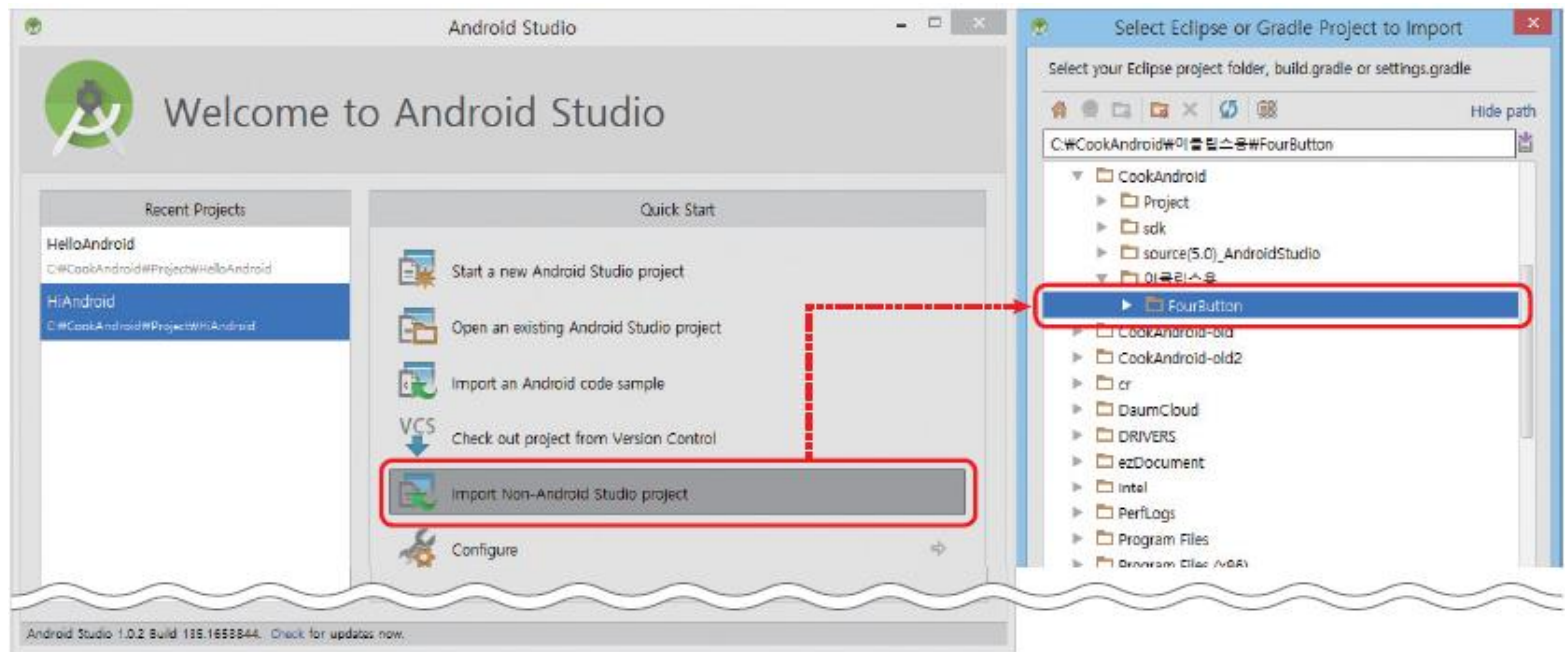


그림 2-36 이클립스용 프로젝트 가져오기 1

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[5/6]

■ 이클립스용 프로젝트 가져오기

- 가져올 폴더 지정 후 프로젝트 변환 옵션 체크한 상태에서 <Finish> 클릭

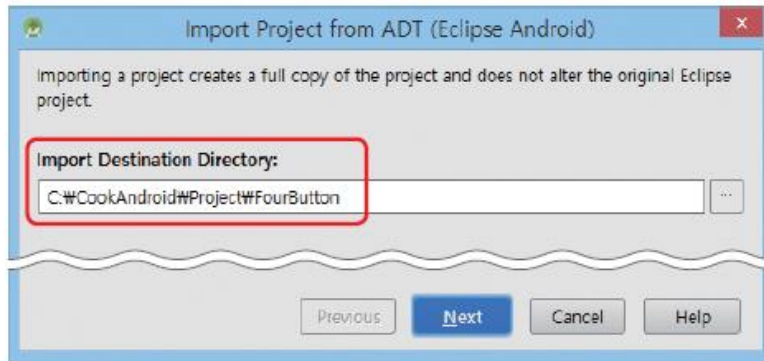


그림 2-37 이클립스용 프로젝트 가져오기 2

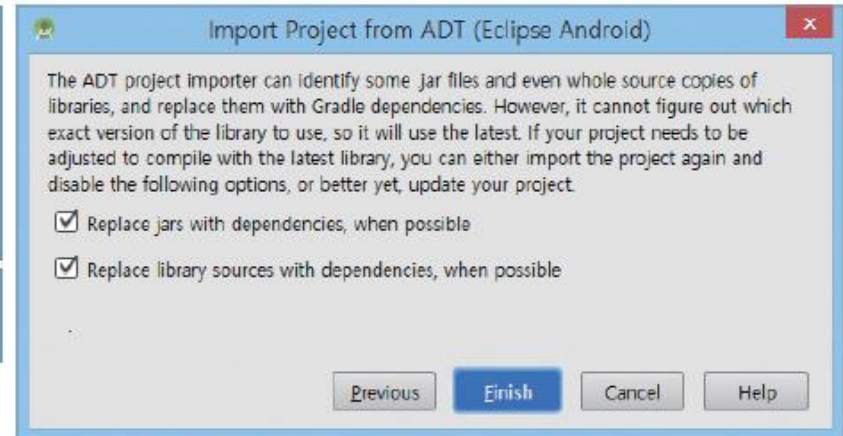


그림 2-38 이클립스용 프로젝트 가져오기 3

3. 안드로이드 애플리케이션 작성

□ 프로젝트 관리[6/6]

■ 이클립스용 프로젝트 가져오기

- 변환 완료된 이클립스 프로젝트는 Android Studio 프로젝트와 같은 방법으로 사용

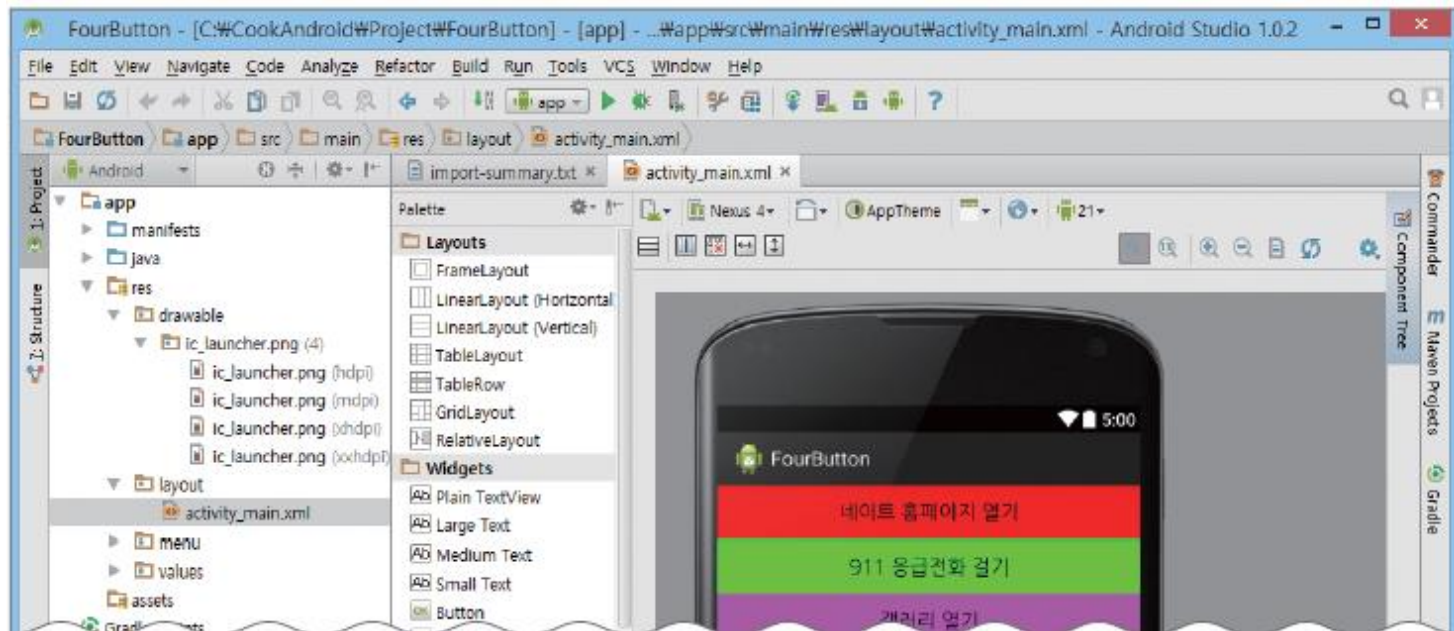


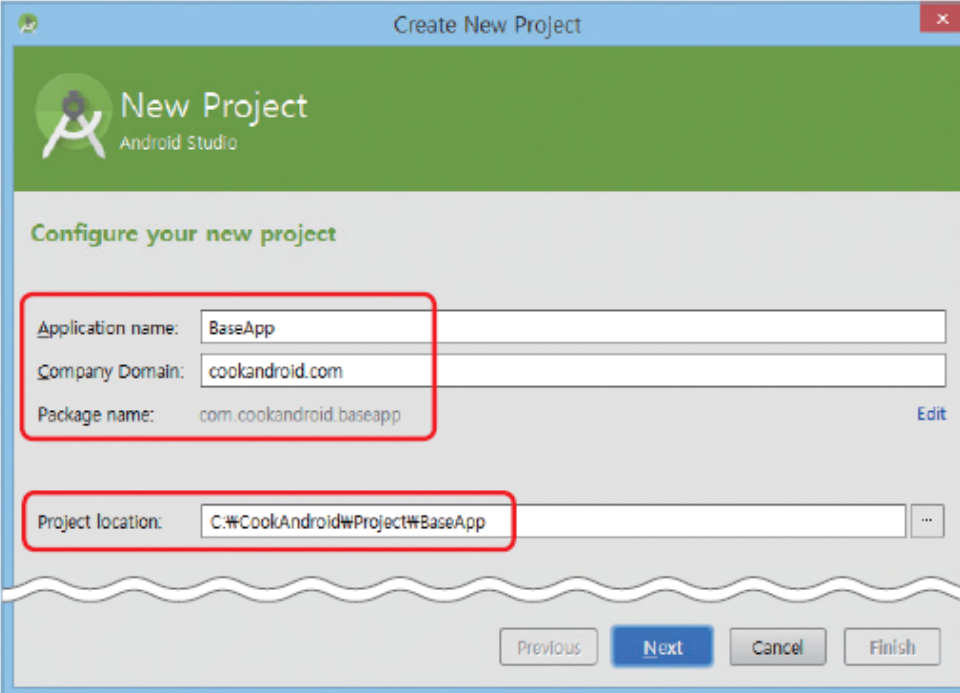
그림 2-39 변환 완료된 이클립스용 프로젝트

3. 안드로이드 애플리케이션 작성(실습)

□ 표준 틀[1/20]

▣ 안드로이드 프로젝트 생성

- 프로젝트 이름은 BaseApp이라고 지정



The screenshot shows the 'Create New Project' window in Android Studio. The window has a title bar 'Create New Project' and a close button. The main area is titled 'New Project' with the Android Studio logo. Below this, it says 'Configure your new project'. There are four input fields: 'Application name' with the value 'BaseApp', 'Company Domain' with 'cookandroid.com', 'Package name' with 'com.cookandroid.baseapp', and 'Project location' with 'C:\CookAndroid\Project\BaseApp'. The 'Application name', 'Company Domain', and 'Package name' fields are grouped together and highlighted with a red rectangle. The 'Project location' field is also highlighted with a red rectangle. At the bottom, there are four buttons: 'Previous', 'Next' (which is highlighted in blue), 'Cancel', and 'Finish'.

그림 2-41 애플리케이션 정보 입력 1

3. 안드로이드 애플리케이션 작성

□ 표준 틀[2/20]

▣ 안드로이드 프로젝트 생성

- [Select the form factors your app will run on]에서 앱을 실행한 환경을 선택

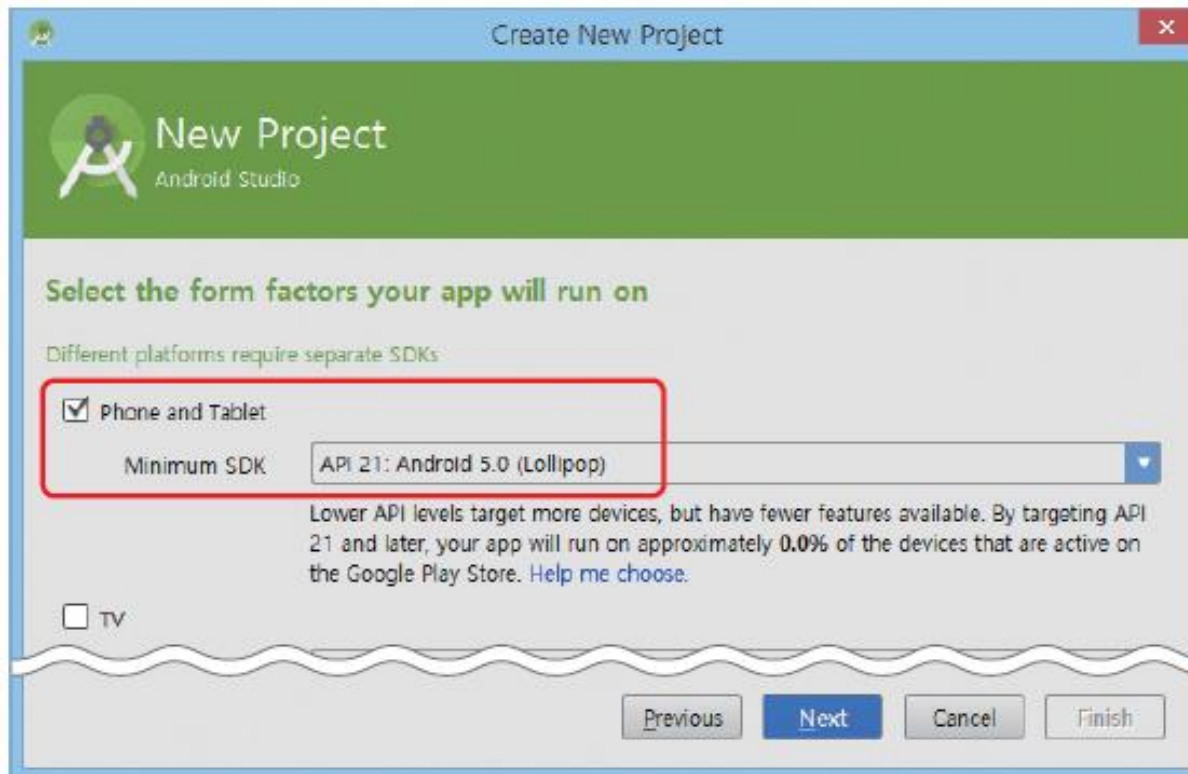


그림 2-42 최소 버전 선택

3. 안드로이드 애플리케이션 작성

□ 표준 틀[3/20]

▣ 안드로이드 프로젝트 생성

- [Add an activity to Mobile]에서 디폴트인 Blank Activity를 선택
- [Choose options for your new file] 창도 디폴트로 두고 <Finish>를 클릭

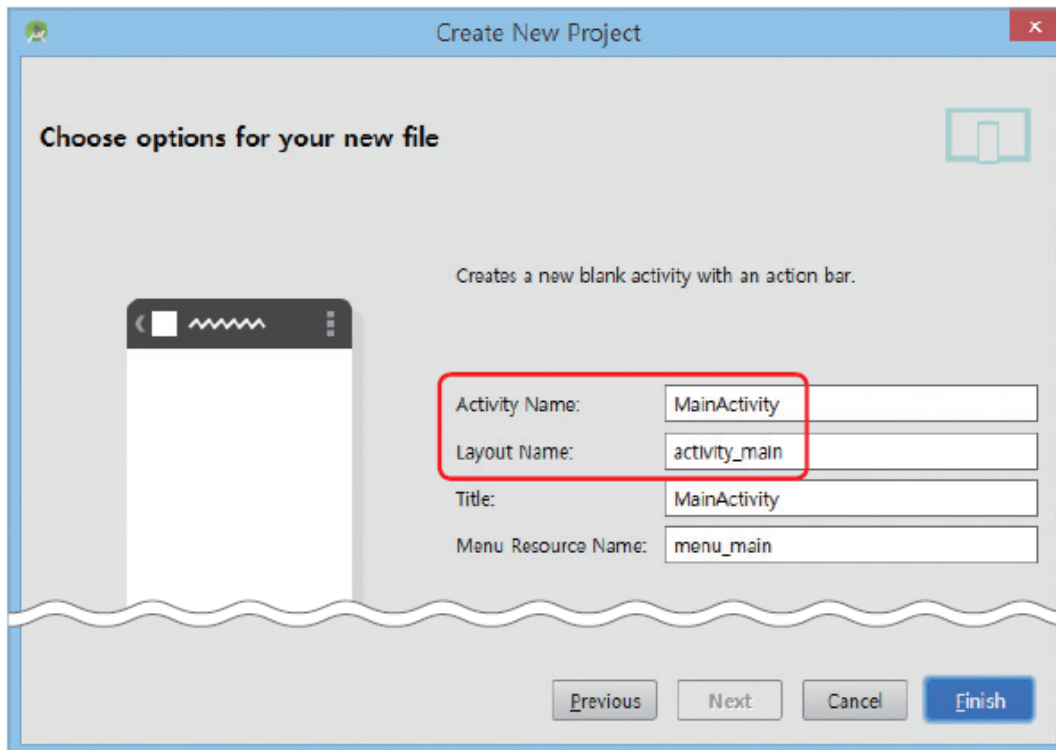


그림 2-43 액티비티 및 레이아웃 이름 입력

3. 안드로이드 애플리케이션 작성

□ 표준 틀[4/20]

▣ 화면 디자인 및 편집

- Project Tree에서 [res]-[layout]-[activity_main.xml]가 기본적으로 열려있을 것
- 열려있지 않으면 Project Tree에서 더블클릭

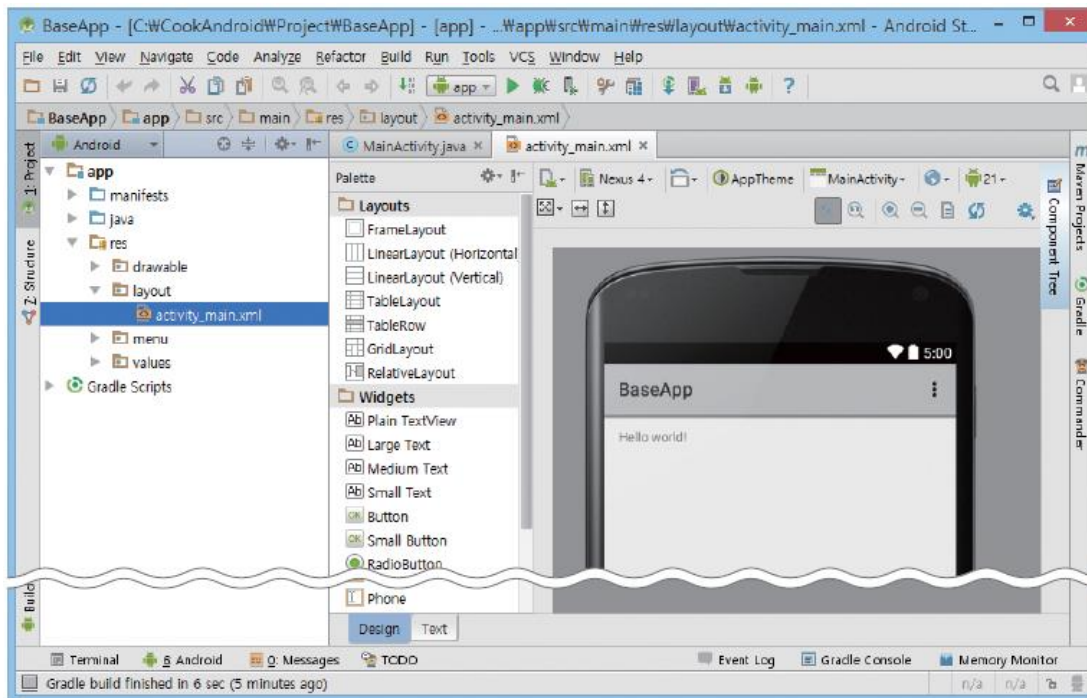


그림 2-44 activity_main.xml 확인

3. 안드로이드 애플리케이션 작성

□ 표준 틀[5/20]

▣ 화면 디자인 및 편집

- 화면 아래쪽의 [Text] 탭을 클릭해서 XML 코드를 확인
- <LinearLayout>과 </LinearLayout> 사이에 버튼을 추가

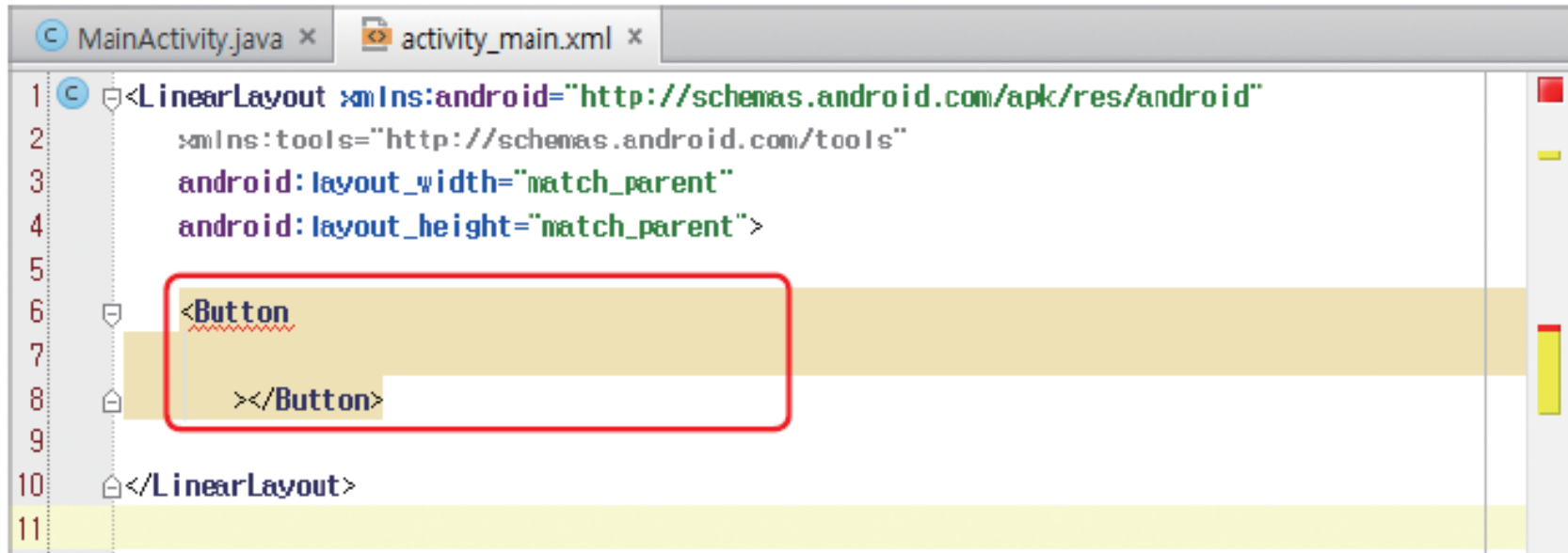


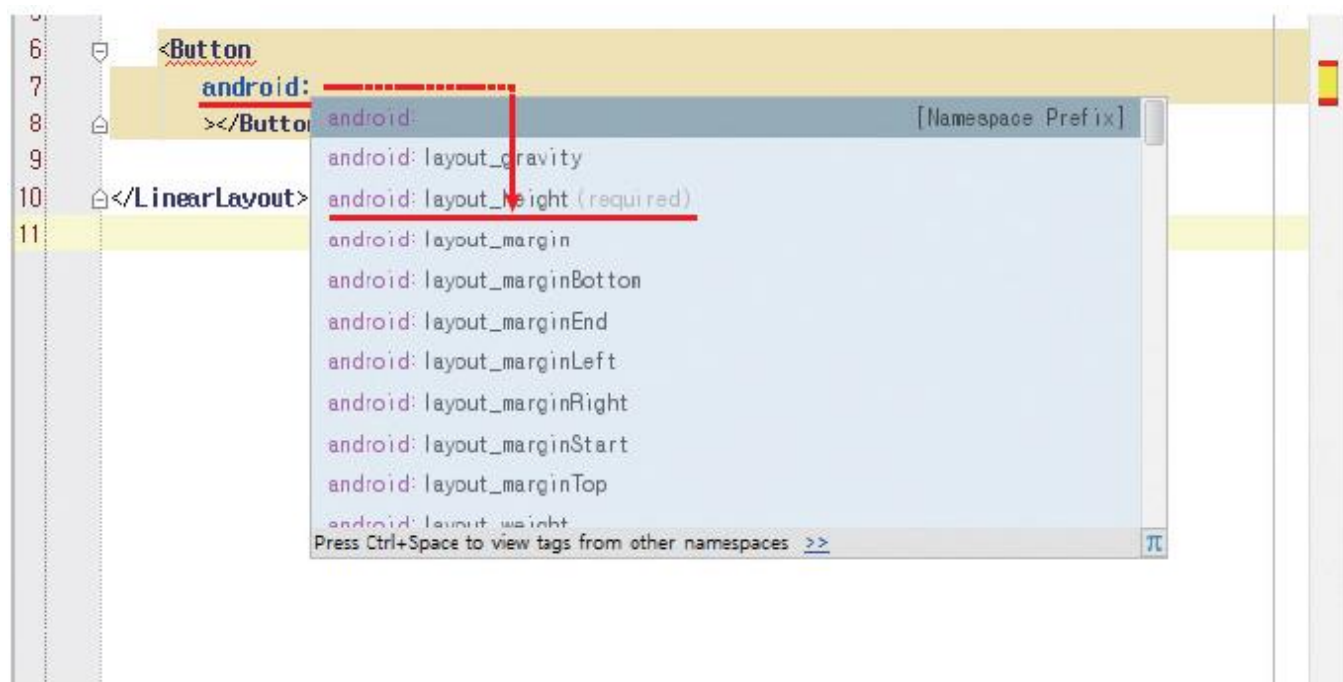
그림 2-46 Button 추가

3. 안드로이드 애플리케이션 작성

□ 표준 틀[6/20]

▣ 화면 디자인 및 편집

- “android:”을 입력하면 자동으로 여러 개를 선택할 수 있는 목록이 나옴
- 그 중에서 layout_height를 더블클릭해서 선택하면 자동완성됨



3. 안드로이드 애플리케이션 작성

□ 표준 틀[7/20]

▣ 화면 디자인 및 편집

- 쌍따옴표(“ ”) 안에 커서를 가져다놓고 [Ctrl]+[Space Bar]를 누르면 내용을 쓸 수 있음

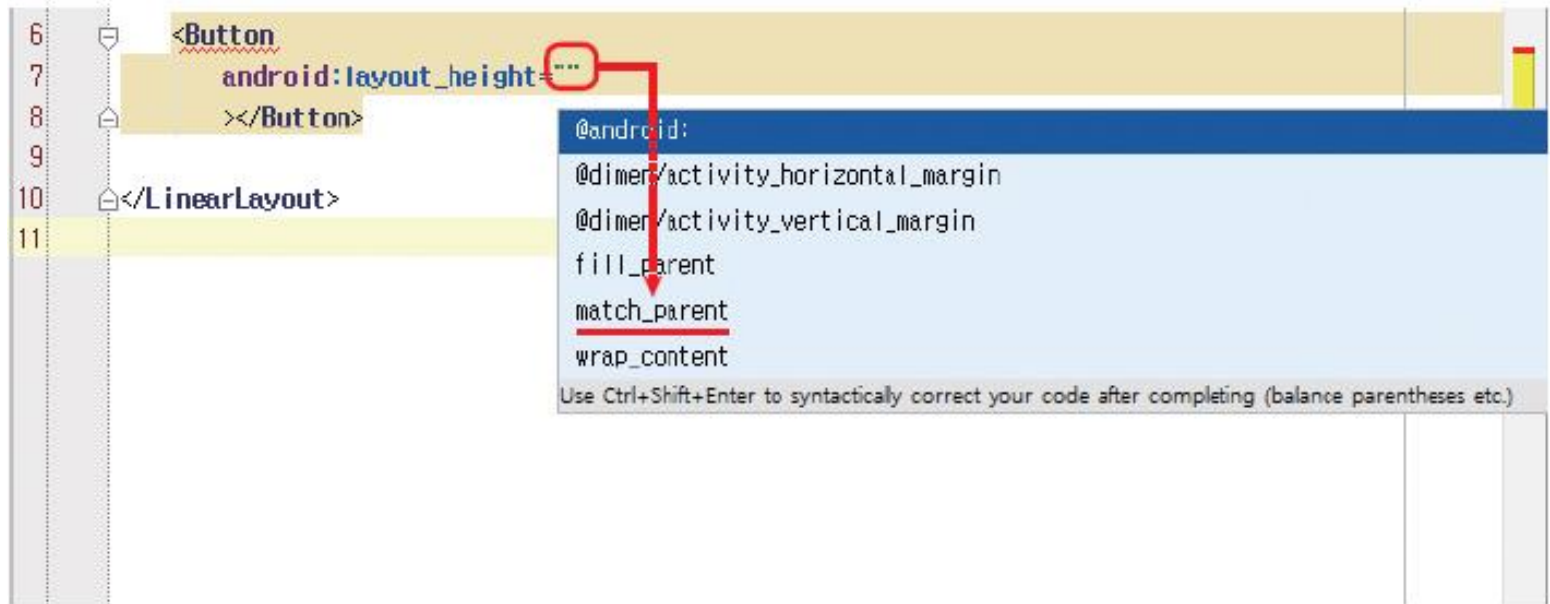


그림 2-48 버튼 속성 중 layout_width 편집

3. 안드로이드 애플리케이션 작성

□ 표준 틀[8/20]

▣ 1차 코드 완성

예제 2-1 activity_main.xml

```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     xmlns:tools="http://schemas.android.com/tools"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent" >
5
6
7     <Button
8         android:layout_width="match_parent"
9         android:layout_height="wrap_content"
10        android:id="@+id/button1"
11        android:text="@string/strBtn1"
12    </Button>
13
14 </LinearLayout>
```

3. 안드로이드 애플리케이션 작성

□ 표준 틀[9/20]

▣ 오류 수정

- Project Tree의 [res]-[values]-[strings.xml]을 더블클릭해서 코드 편집

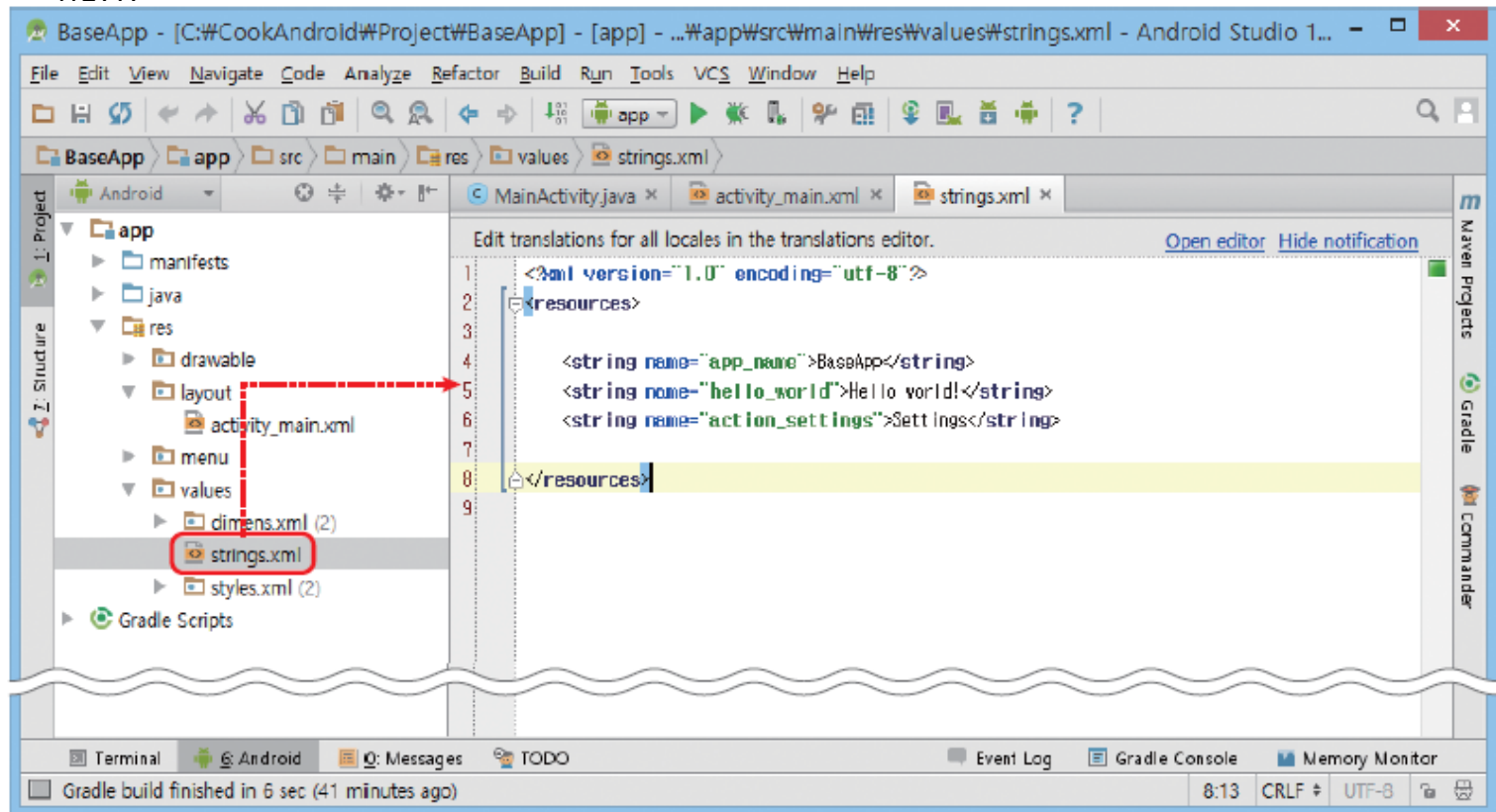


그림 2-49 strings.xml 파일

3. 안드로이드 애플리케이션 작성

□ 표준 틀[10/20]

▣ 오류 수정

- strBtn1 문자열을 추가한 후 저장

예제 2-2 strings.xml

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <resources>
03     <string name="app_name">BaseApp</string>
04     <string name="hello_world">Hello world!</string>
05     <string name="action_settings">Settings</string>
06     <string name="strBtn1">버튼입니다</string>
07 </resources>
```

3. 안드로이드 애플리케이션 작성

□ 표준 틀[11/20]

□ 확인

- Activity_main.xml에서 하단의 [Design] 탭을 클릭해서 그래픽 화면으로 확인

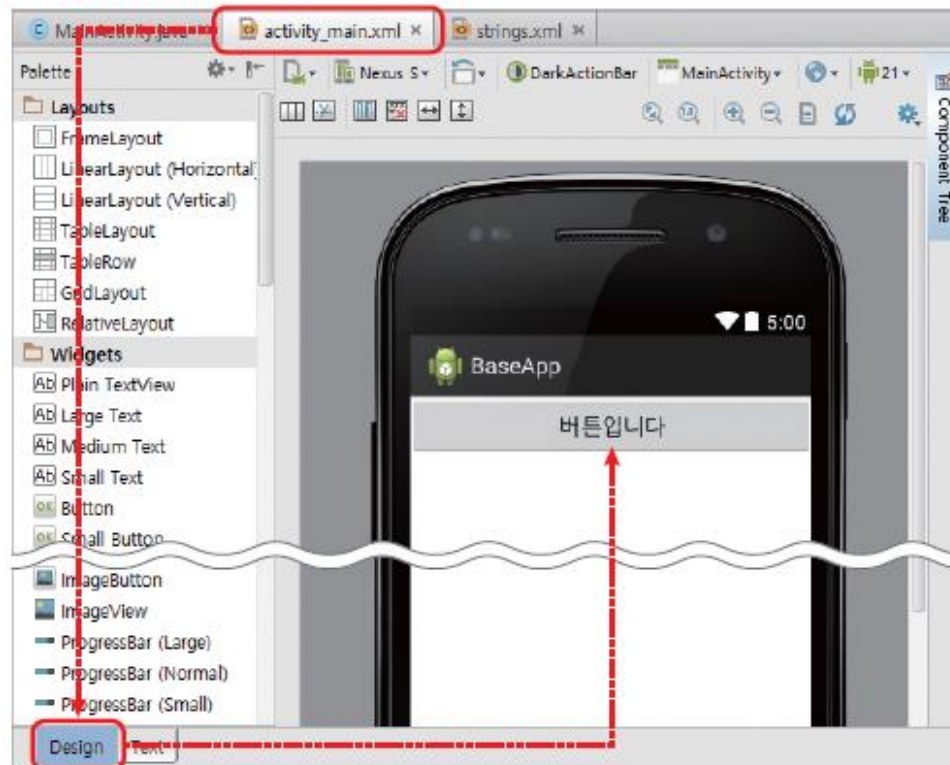


그림 2-50 activity_main.xml 코드의 그래픽 화면

3. 안드로이드 애플리케이션 작성

□ 표준 틀[12/20]

▣ JAVA 코드 작성 및 수정

- Project Tree의 [java]-[패키지 이름]-[MainActivity]를 더블클릭하면 Java 코드가 열림

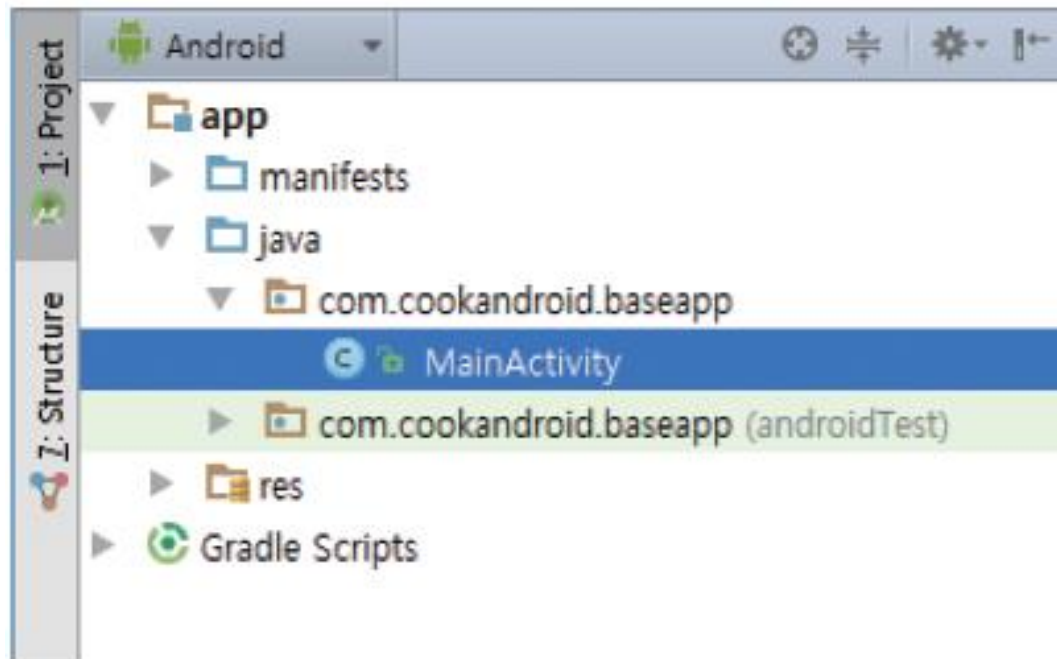


그림 2-53 Java 파일 선택

3. 안드로이드 애플리케이션 작성

□ 표준 틀[13/20]

▣ JAVA 코드 작성 및 수정

- 자동완성된 코드에 사용하지 않는 부분 삭제

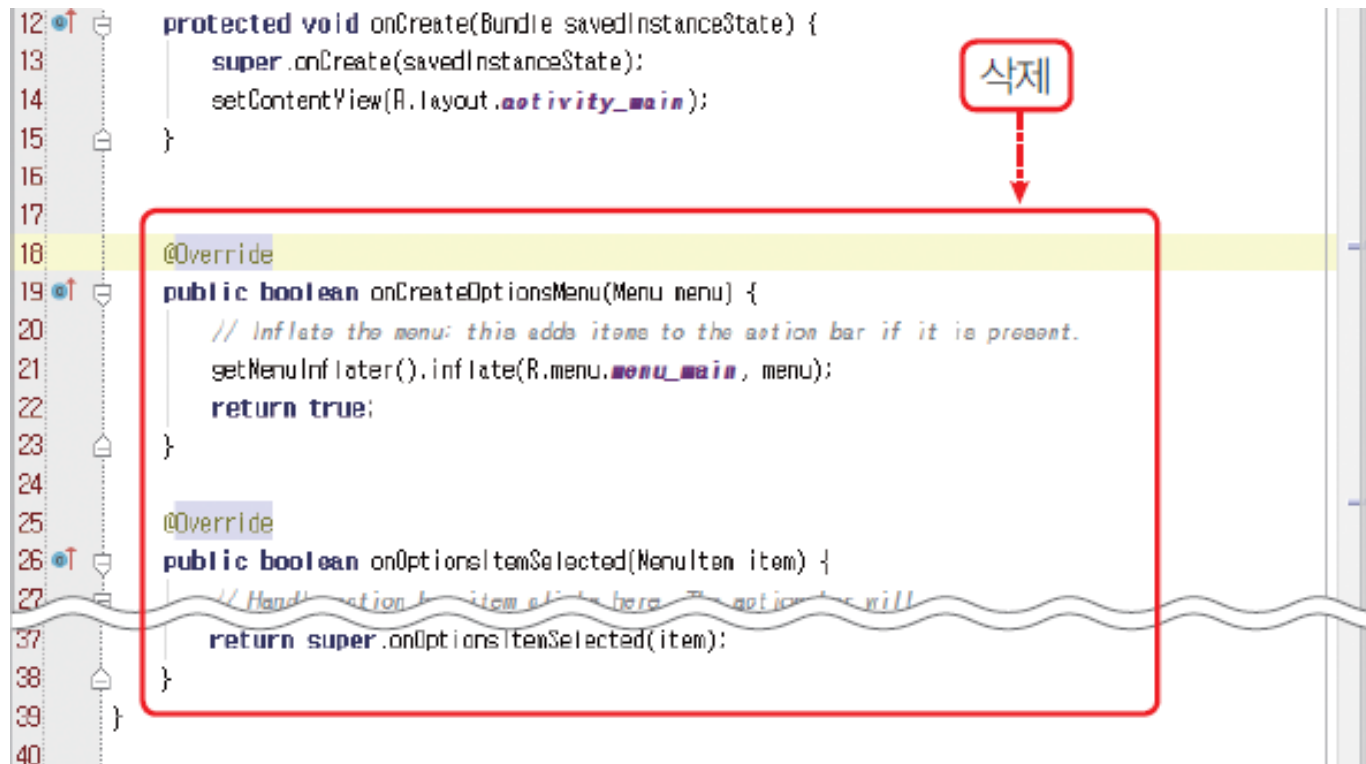


그림 2-54 필요 없는 메소드 삭제

3. 안드로이드 애플리케이션 작성

□ 표준 틀[14/20]

▣ JAVA 코드 작성 및 수정

- import 앞의 작은 (+) 아이콘을 클릭하면 행이 확장됨
- [ct기]+[shift]+[O]를 누르고 <Run> 버튼을 클릭 → 불필요하게 import된 문장 제거

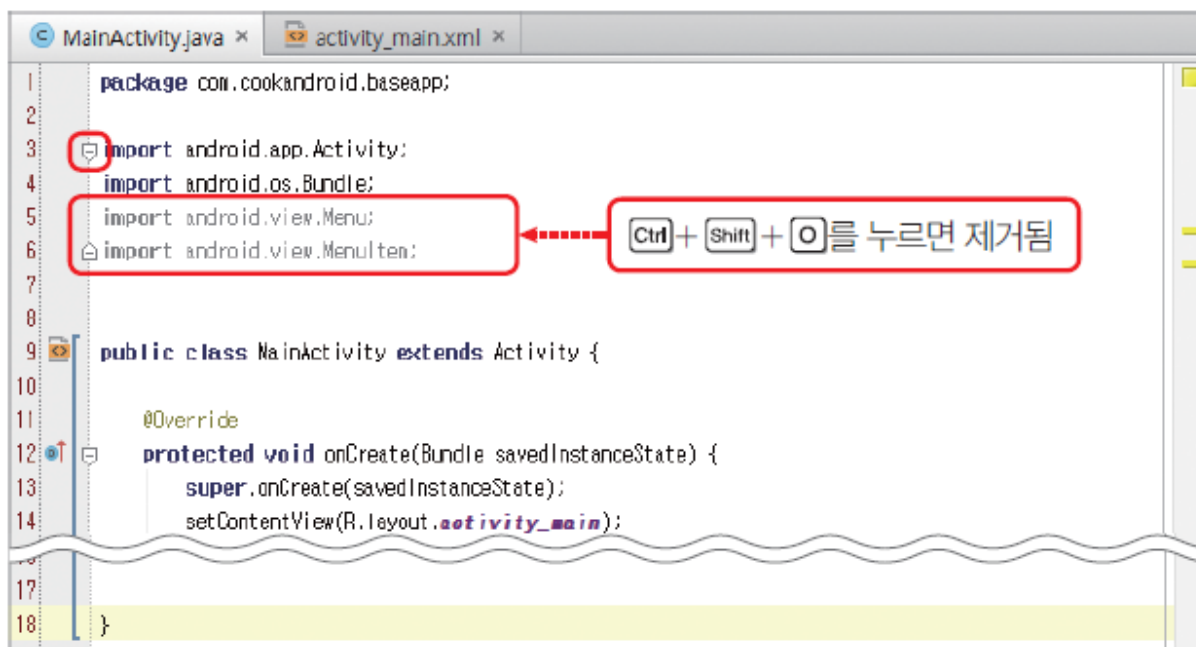


그림 2-55 필요 없는 import문 삭제

3. 안드로이드 애플리케이션 작성

□ 표준 틀[15/20]

▣ JAVA 코드 작성 및 수정

■ Button 변수 추가

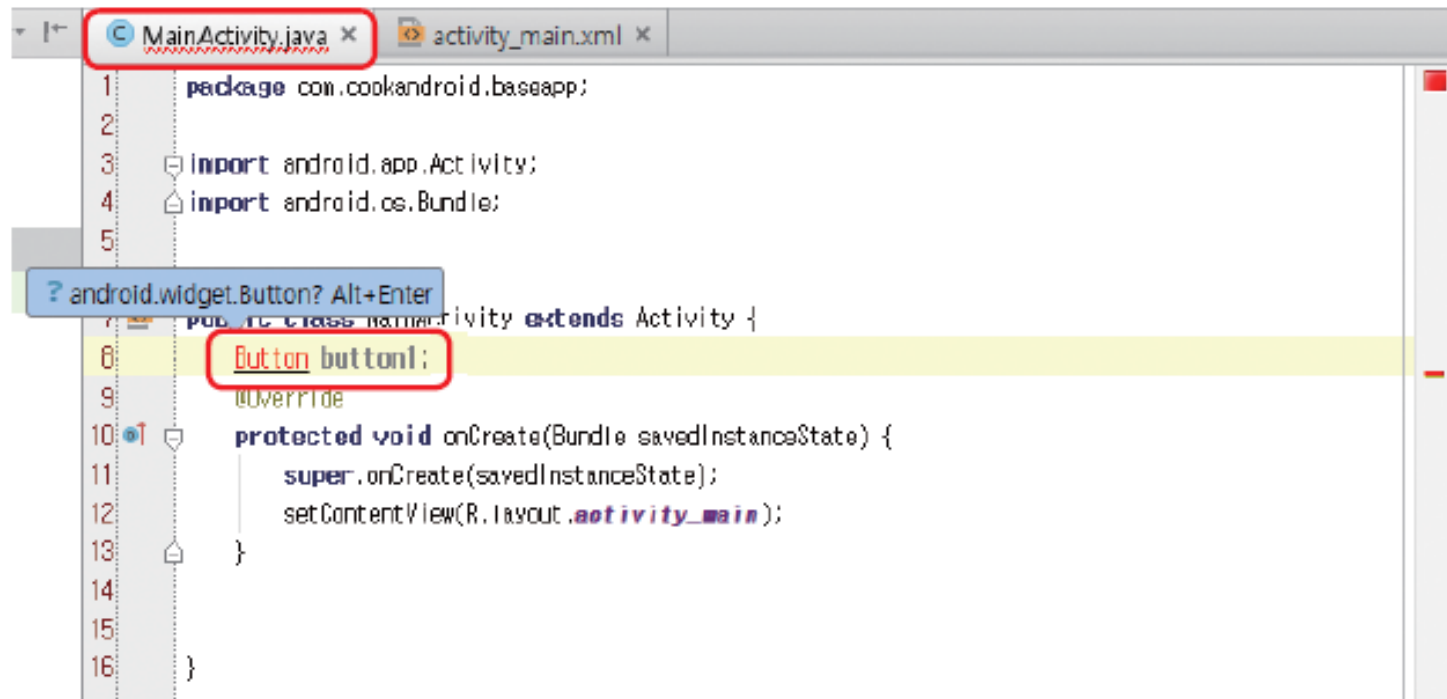


그림 2-56 Button 변수 추가

3. 안드로이드 애플리케이션 작성

□ 표준 틀[16/20]

▣ JAVA 코드 작성 및 수정

- [Alt]+[Enter] 누르면 Button과 관련된 클래스가 자동으로 import문에 추가됨



그림 2-57 자동 import 확인

3. 안드로이드 애플리케이션 작성

□ 표준 틀[17/20]

▣ JAVA 코드 작성 및 수정

- findViewById() 메소드를 사용하여 activity_main.xml 파일에서 만든 객체에 접근
- setContentView() 메소드 바로 아래에 다음 코드를 추가*

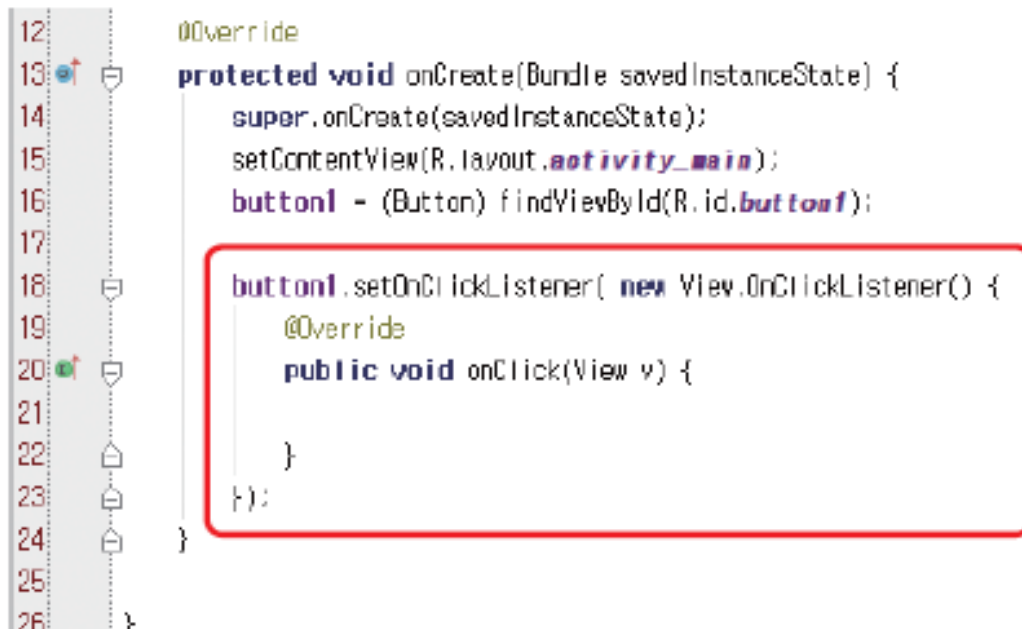
```
button1 = (Button) findViewById(R.id.button1);
```

3. 안드로이드 애플리케이션 작성

□ 표준 틀[18/20]

▣ JAVA 코드 작성 및 수정

- “button1.setOncl”까지 입력 후 `setOnClickListener()`를 선택
- `setOnClickListener()`에 “new View.oncl”을 입력 후 `View.OnClickListener()`선택



```
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    button1 = (Button) findViewById(R.id.button1);  
    button1.setOnClickListener( new View.OnClickListener() {  
        @Override  
        public void onClick(View v) {  
        }  
    });  
}
```

그림 2-60 자동완성된 코드

3. 안드로이드 애플리케이션 작성

□ 표준 틀[19/20]

▣ JAVA 코드 작성 및 수정

- 버튼을 클릭했을 때 작동하기 원하는 모든 코드를 onClick() 메소드 안에 입력

예제 2-3 MainActivity.java

```
1 package com.cookandroid.baseapp;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5 import android.view.View;
6 import android.widget.Button;
7 import android.widget.Toast;
8
9 public class MainActivity extends Activity {
10     Button button1;
11
12     @Override
13     public void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_main);
16
```

```
16
17         button1 = (Button) findViewById(R.id.button1);
18
19         button1.setOnClickListener( new View.OnClickListener() {
20             @Override
21             public void onClick(View v) {
22
23                 Toast.makeText(getApplicationContext(), "버튼을 눌렀어요",
24                     Toast.LENGTH_SHORT).show();
25             }
26         });
27     }
```


3. 안드로이드 애플리케이션 작성

□ 표준 틀[20/20]

▣ 프로젝트 실행 및 결과 확인

- [Run As]-[Run 'app']을 선택하거나 [Run 'app']을 클릭해서 프로젝트를 실행



그림 2-61 실행 결과

3. 안드로이드 애플리케이션 작성

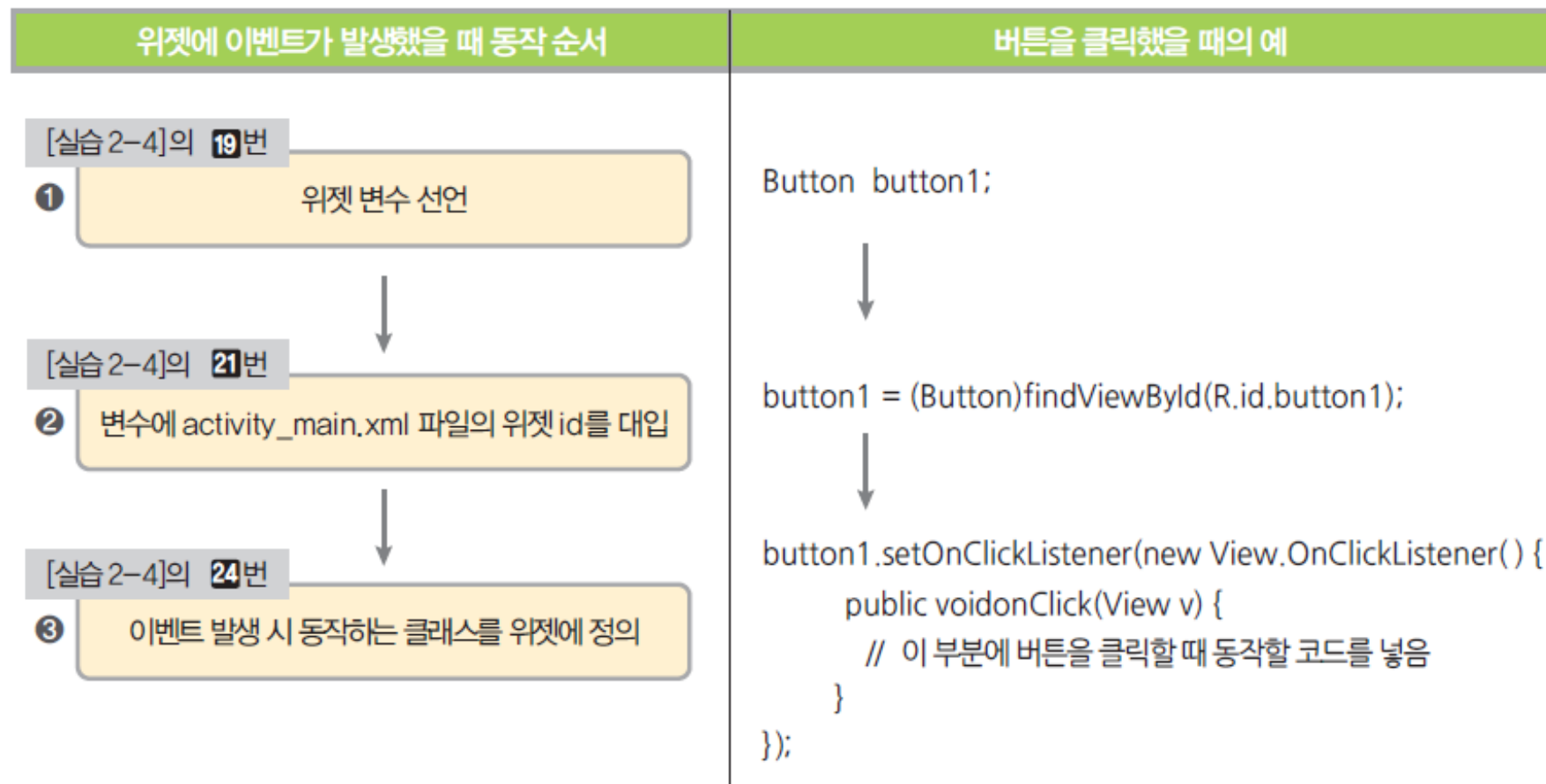


그림 2-63 위젯에 이벤트를 작동하기 위한 코딩 요약

▶ 직접 풀어보기 2-3

다음 그림과 같이 버튼 4개를 만든 후에 각 버튼을 클릭하면 필요한 내용이 작동되는 프로젝트 FourButton을 작성하라. 각 버튼의 색상은 다른 색상으로 변경한다.

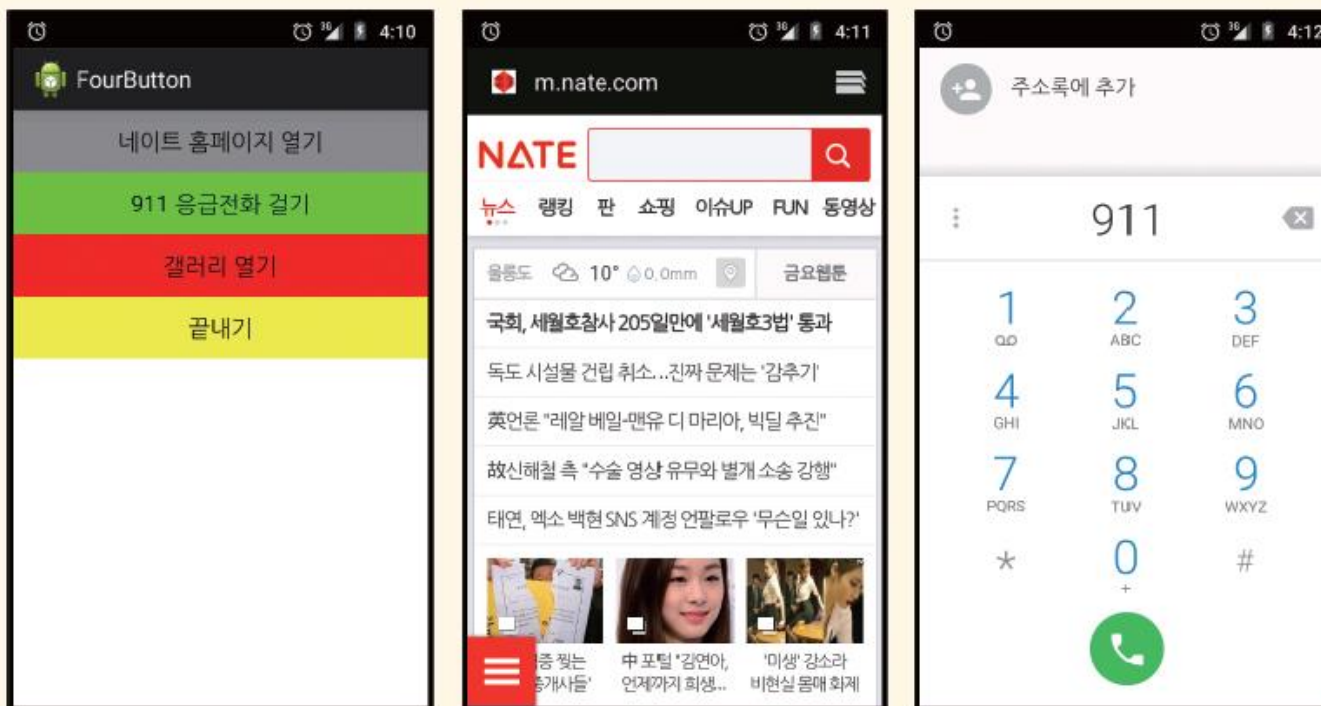


그림 2-64 실행 결과

3. 프로젝트의 구성

□ BaseApp 프로젝트 구성[1/3]

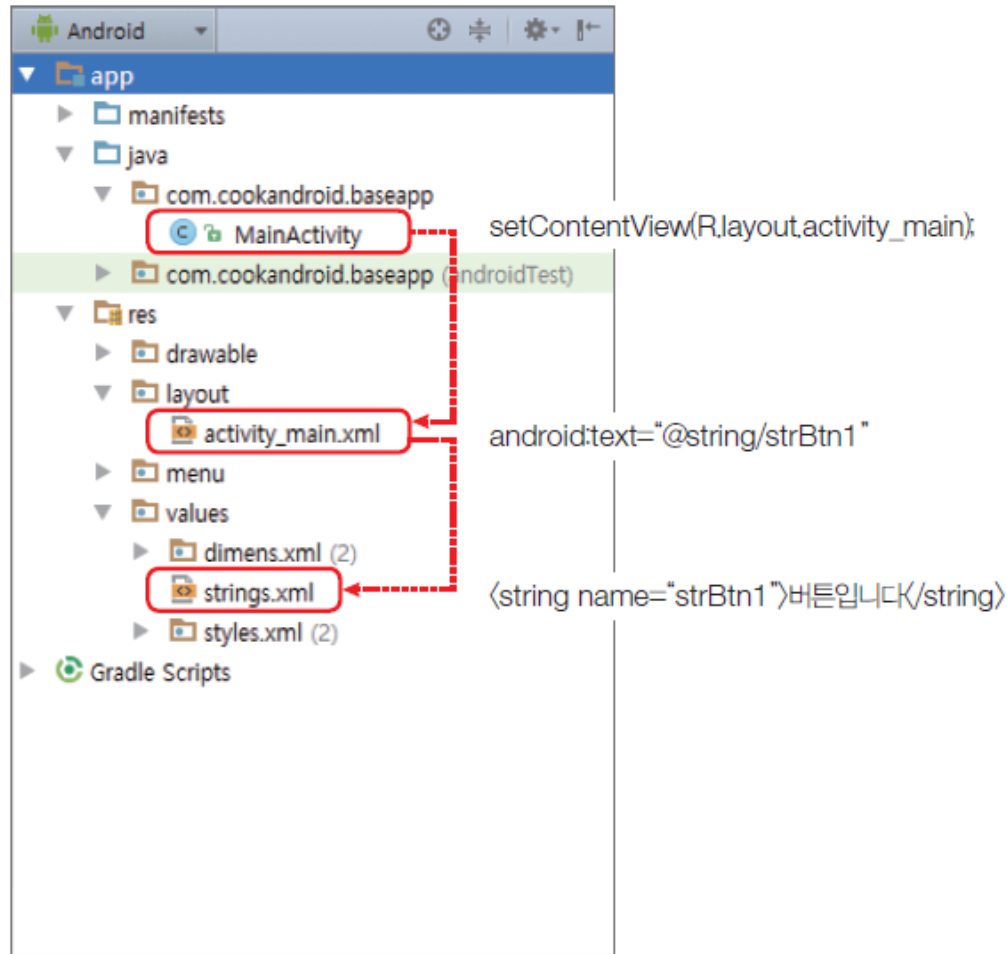


그림 2-65 안드로이드 프로젝트 구성

3. 프로젝트의 구성

□ BaseApp 프로젝트 구성[2/3]

▣ java 폴더

- 하위에 패키지명의 하위 폴더가 있는데, 이는 안드로이드 프로젝트를 생성할 때 입력한 패키지 이름과 동일
- 패키지 이름 아래에 MainActivity.java로 메인 Java 소스가 들어 있음

▣ res 폴더

- 앱 개발에 사용되는 이미지, 레이아웃, 문자열 등이 들어가는 폴더
- 이미지 파일은 drawable 폴더에 넣음
- layout 폴더는 액티비티(화면)을 구성하는 xml 파일을 넣으면 됨
- values 폴더는 문자열을 저장하는 string.xml이 들어 있음
- menu 폴더는 메뉴 XML 파일이 저장되어 있음



3. 프로젝트의 구성

□ BaseApp 프로젝트 구성[3/3]

▣ manifests 폴더

- AndroidManifest.xml 파일이 들어 있는데, 앱의 여러 가지 정보를 담고 있음

▣ Gradle Scripts 폴더

- Gradle 빌드 시스템과 관련된 파일이 들어 있음
 - *build.gradle (Module: app)* : 빌드 스크립트 핵심 파일
 - *local.properties* : 컴파일 되는 SDK의 경로가 들어 있음