

02 처음 만드는 안드로이드 애플리케이션



IT CookBook, 코틀린을 활용한 안드로이드 프로그래밍

학습목표

- ❖ 안드로이드 애플리케이션의 기본적인 작성법을 익힌다.
- ❖ AVD의 명칭을 파악한다.
- ❖ 안드로이드 프로젝트의 구성을 이해한다.

차례

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

01 Hello Android 프로젝트

- [그림 2-2]의 순서로 안드로이드 프로젝트를 개발을 진행함

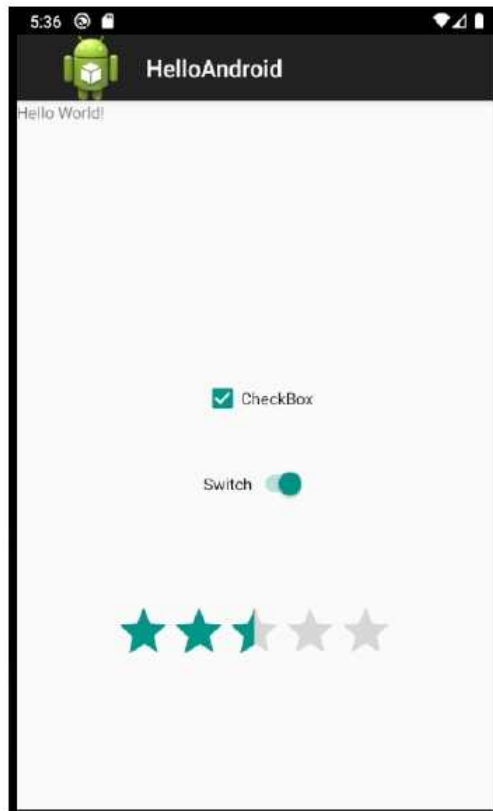


그림 2-1 처음 만드는 안드로이드 애플리케이션

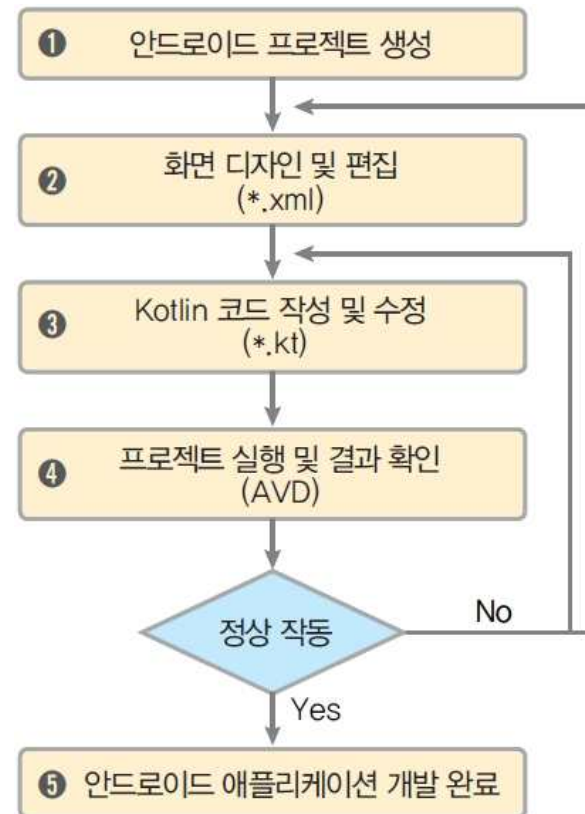


그림 2-2 안드로이드 프로젝트 개발 단계

01 Hello Android 프로젝트

- <실습 2-1> 첫 응용 프로그램 작성하기
 - 1 안드로이드 프로젝트 생성
 - (1) Android Studio를 실행함
 - (2) 초기 창 왼쪽에 기존에 사용했던 프로젝트의 목록이 나타남
 - 필요 없다면 마우스 오른쪽 버튼을 클릭하고 [Remove Selected from Welcome Screen]을 선택하여 제거함

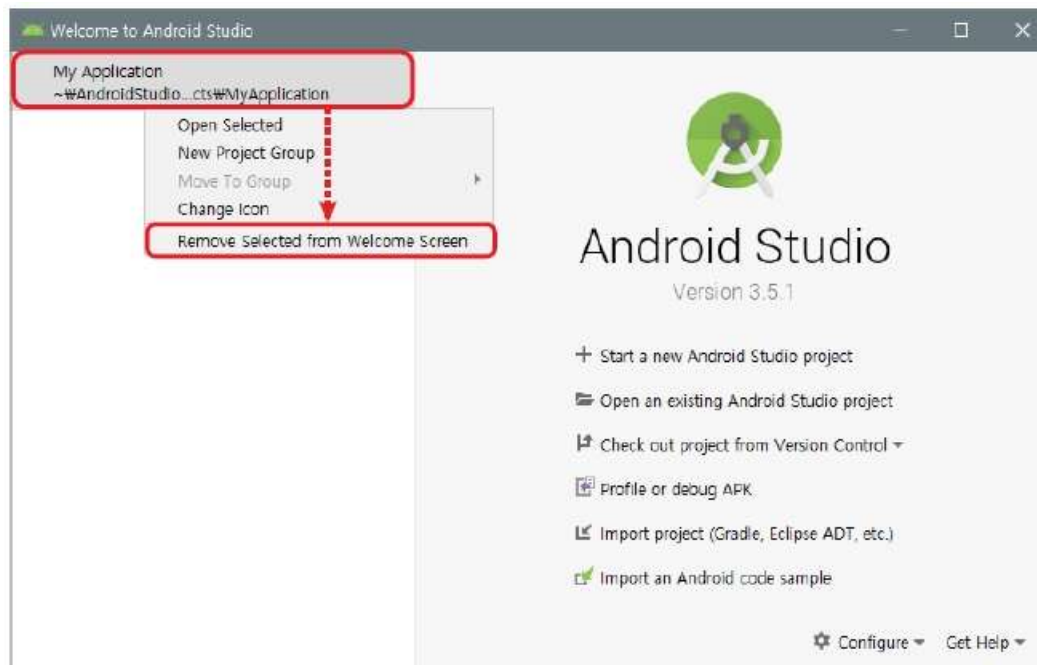


그림 2-3 기존 프로젝트 목록 제거

01 Hello Android 프로젝트

- (3) 초기 창에서 [Start a new Android Studio project]를 클릭함

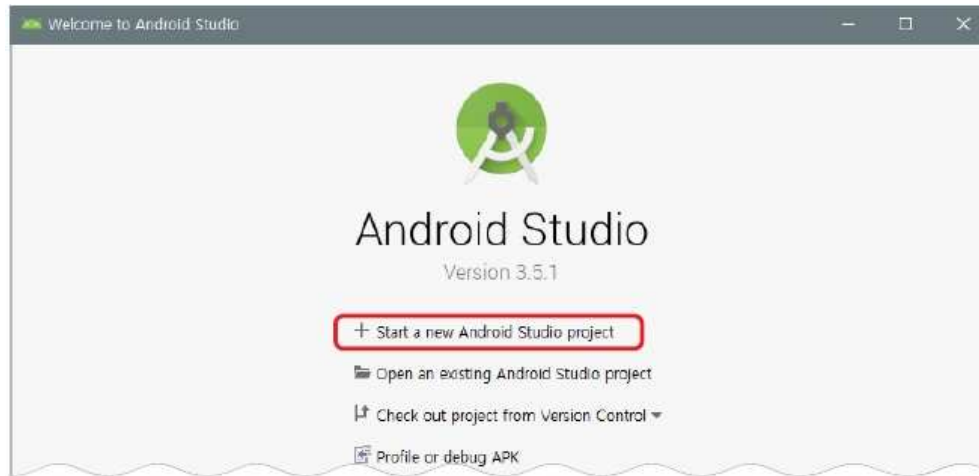


그림 2-4 새로운 안드로이드 프로젝트

- (4) [Choose your project]에서는 생성할 프로젝트의 종류를 선택함
- [Phone and Tablet] 탭의 'Empty Activity'를 선택하고 <Next>를 클릭함.

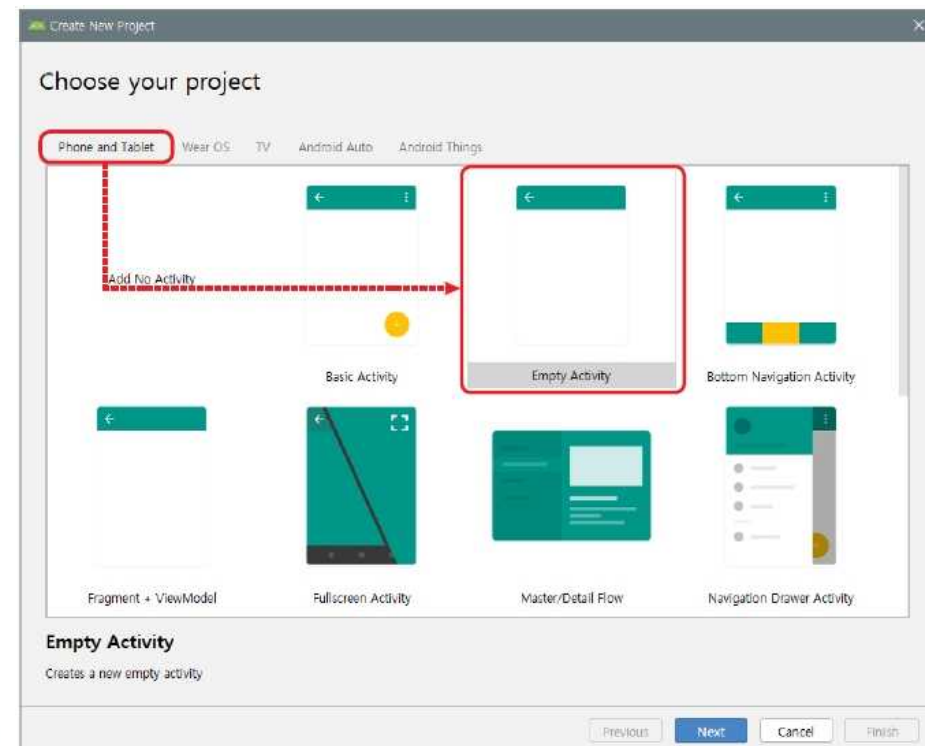
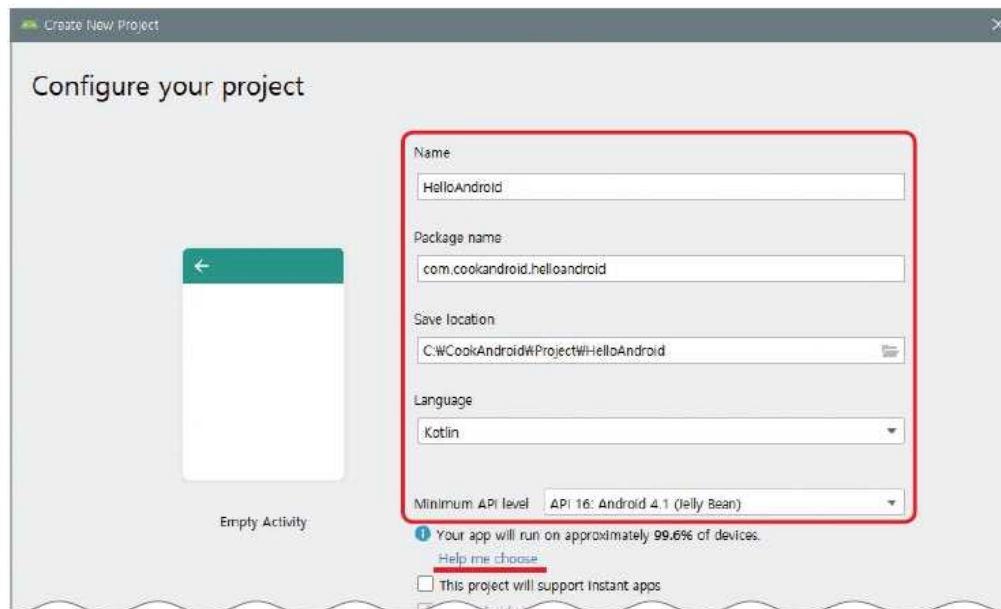


그림 2-5 프로젝트 선택

01 Hello Android 프로젝트

- (5) [Configure your project]에서는 프로젝트에 대한 정보를 입력하거나 선택함
 - **Name:** HelloAndroid → 프로젝트 이름이며 대문자로 시작하는 것이 좋음.
 - **Package name:** com.cookandroid.helloandroid
→ 도메인 이름(cookandroid.com)과 애플리케이션 이름을 이어서 만듦
 - **Save location:** C:\CookAndroid\Project\HelloAndroid
→ 프로젝트가 저장될 폴더를 지정함(한글 폴더명은 불가)
 - **Language:** Kotlin → 사용할 프로그래밍 언어는 Kotlin임.
 - **Minimum API level:** API 16: Android 4.1 (Jelly Bean)
→ 이 앱이 작동하는 최하 버전을 선택함



01 Hello Android 프로젝트

- (6) 설정 중에는 생략되었으나 다음 2개 항목이 내부적으로 추가되어 있음
 - Activity Name: MainActivity
→ 기본 소스인 Kotlin 파일 이름(MainActivity.kt)으로 지정됨
 - Layout Name: activity_main
→ 기본 화면인 XML 파일 이름(activity_main.xml)으로 지정됨

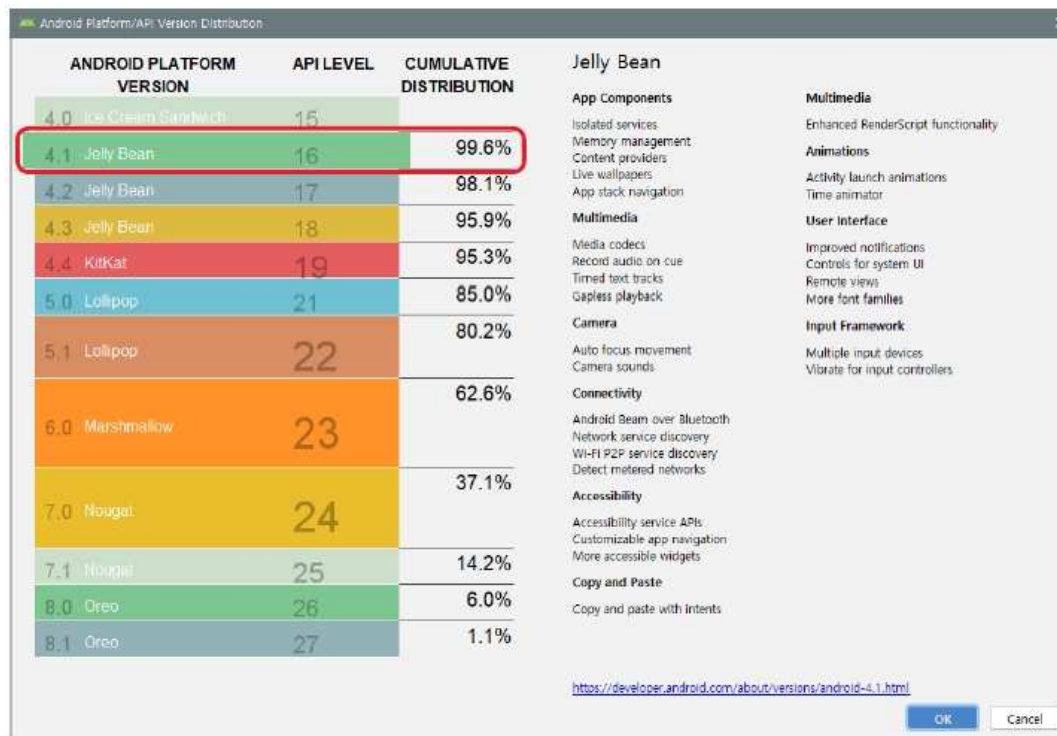


그림 2-7 안드로이드 버전별 누적 점유율

01 Hello Android 프로젝트

- (7) 프로젝트의 [res]-[layout]-[activity_main.xml]을 더블 클릭하면 오른쪽에 프로젝트의 화면이 보임
 - 만약 스마트폰 화면이 안 보이면 아래쪽의 [Design] 탭을 클릭함

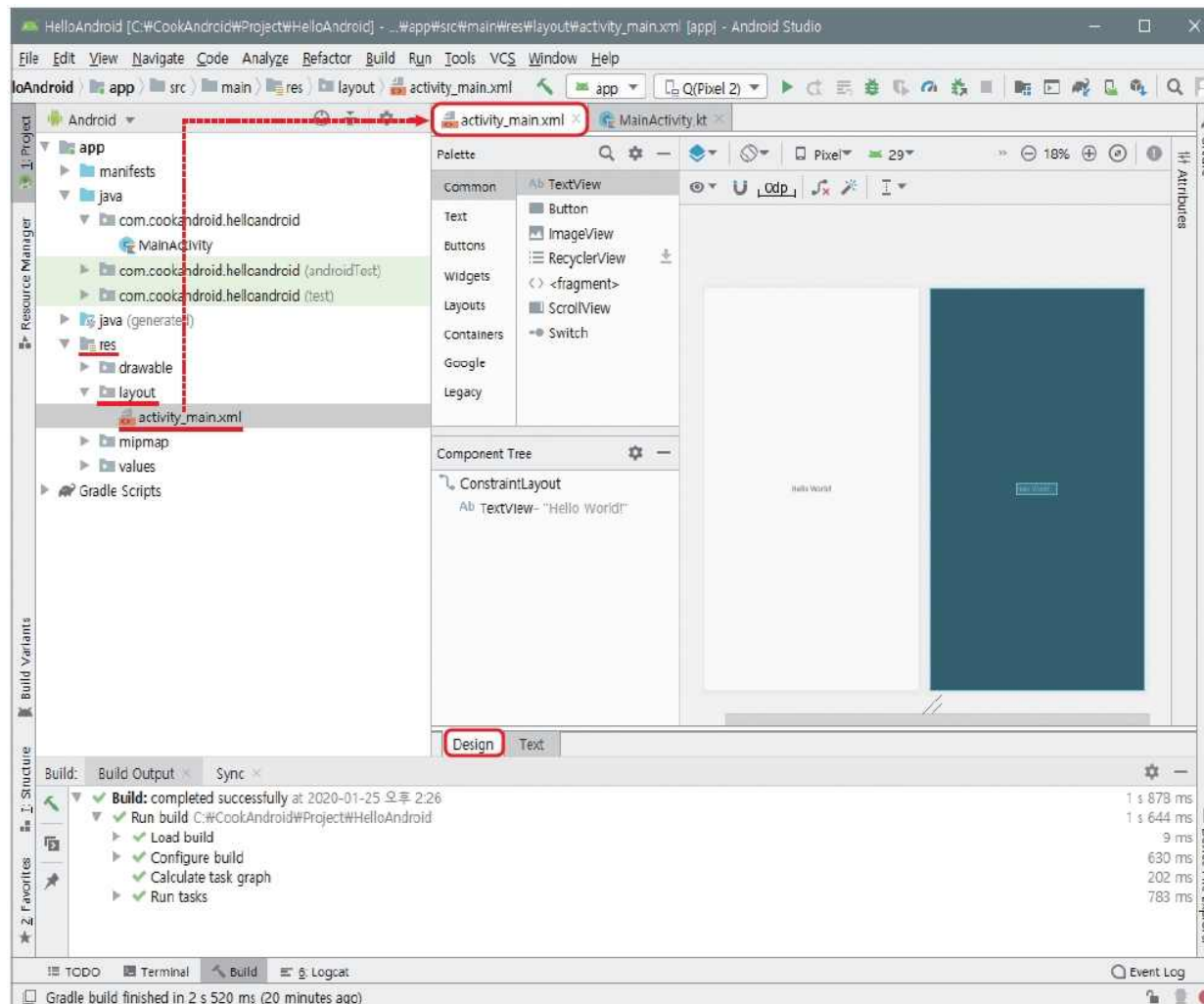


그림 2-8 안드로이드 프로젝트 생성 결과

01 Hello Android 프로젝트

- 2 화면 디자인 및 편집
 - (8) 스마트폰 그림 위쪽의 'Pixel'을 클릭하여 아랫부분의 'AVD: QPixel_2_'를 선택함

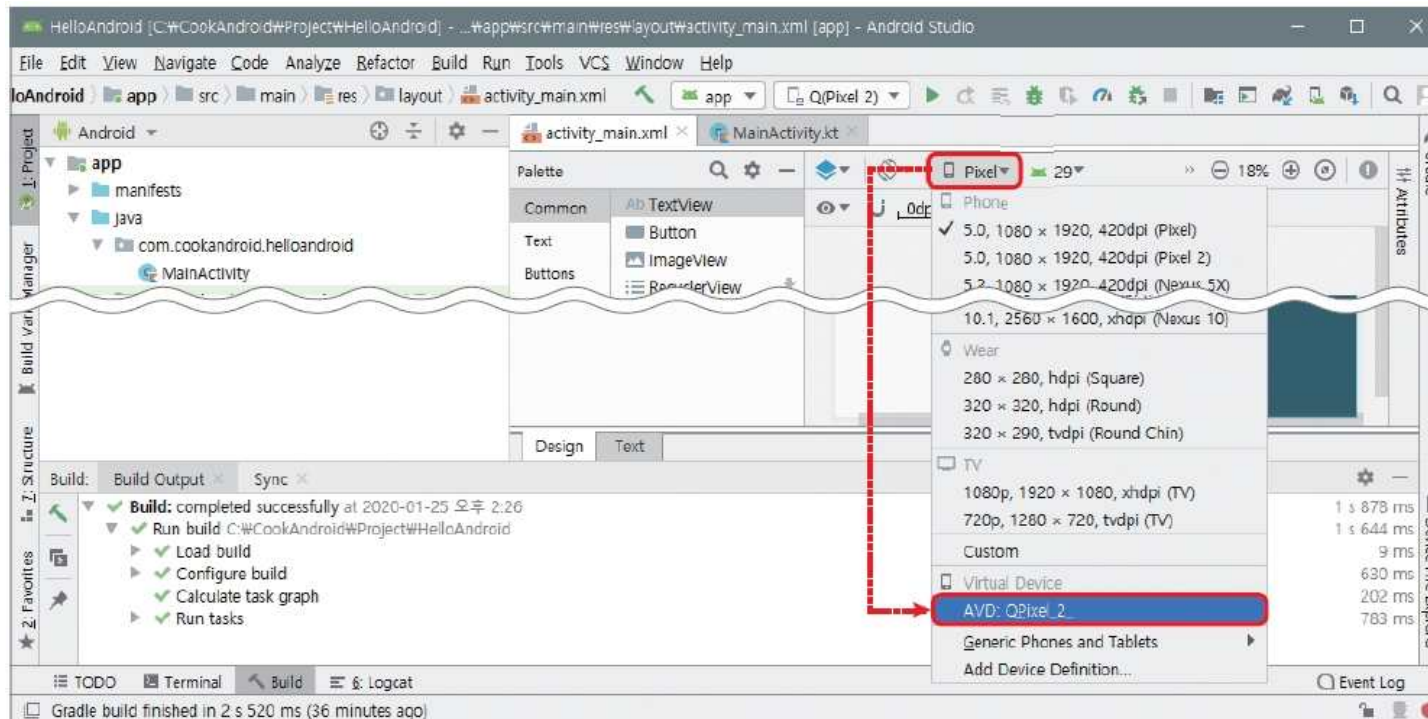


그림 2-9 디자인 환경 설정

01 Hello Android 프로젝트

- (9) 자동 생성되는 레이아웃은 ConstraintLayout임. 우선은 RelativeLayout으로 변경함. 아래쪽의 [Text] 탭을 클릭한 후 2행의 androidx.constraintlayout.widget.Constraint Layout을 RelativeLayout으로 변경함.



그림 2-10 레이아웃 변경

01 Hello Android 프로젝트

- (10) 다시 아래쪽의 [Design] 탭을 클릭한 후 [Enable Autoconnection to Parent] 아이콘을 클릭하고 왼쪽의 위젯 중에서 몇 개를 오른쪽에 끌어다 놓음
 - 오른쪽 위의 확대/축소 아이콘으로 화면 크기를 조절할 수 있음

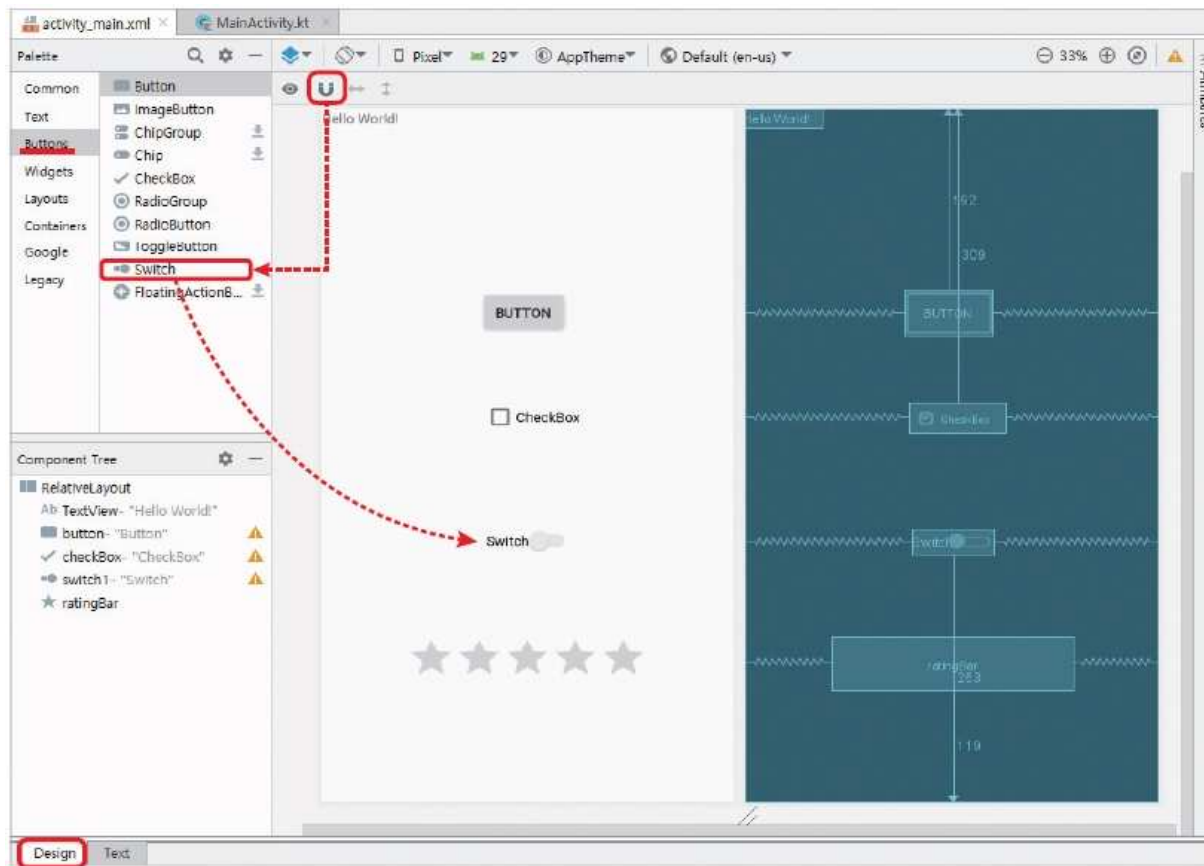


그림 2-11 그래픽 화면에서 위젯 끌어다 놓기

01 Hello Android 프로젝트

- (11) 왼쪽 아래의 [Text] 탭을 클릭하면 화면에 XML 코드가 표시됨
 - 오른쪽의 [Preview] 탭을 클릭하면 미리보기가 나타나거나 사라짐
 - Button 코드를 삭제해 보기

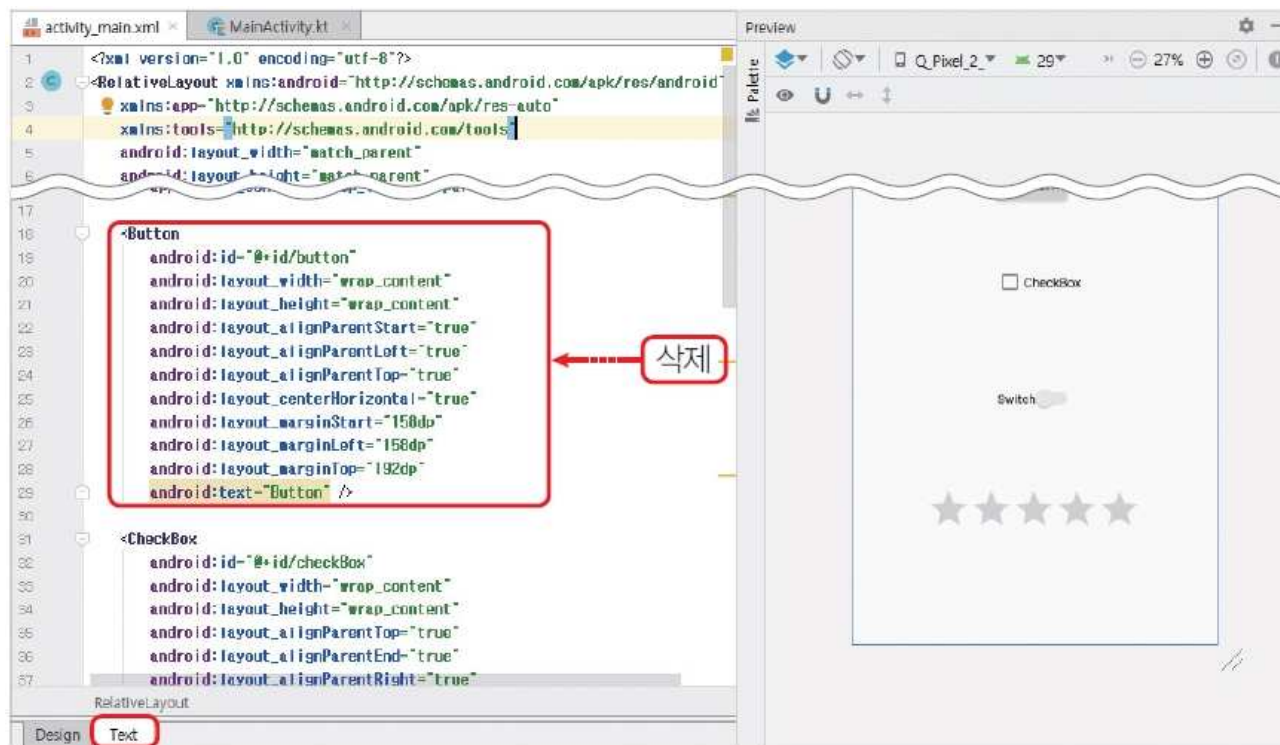


그림 2-12 XML 코드 변경

01 Hello Android 프로젝트

- (12) 다시 [Design] 탭을 클릭하면 방금 삭제한 코드(버튼)가 화면에서도 삭제된 것을 확인할 수 있음
 - Android Studio 왼쪽 상단의 저장 아이콘을 클릭하거나 메뉴의 [File]-[Save All]을 클릭하여 변경 사항을 저장함.

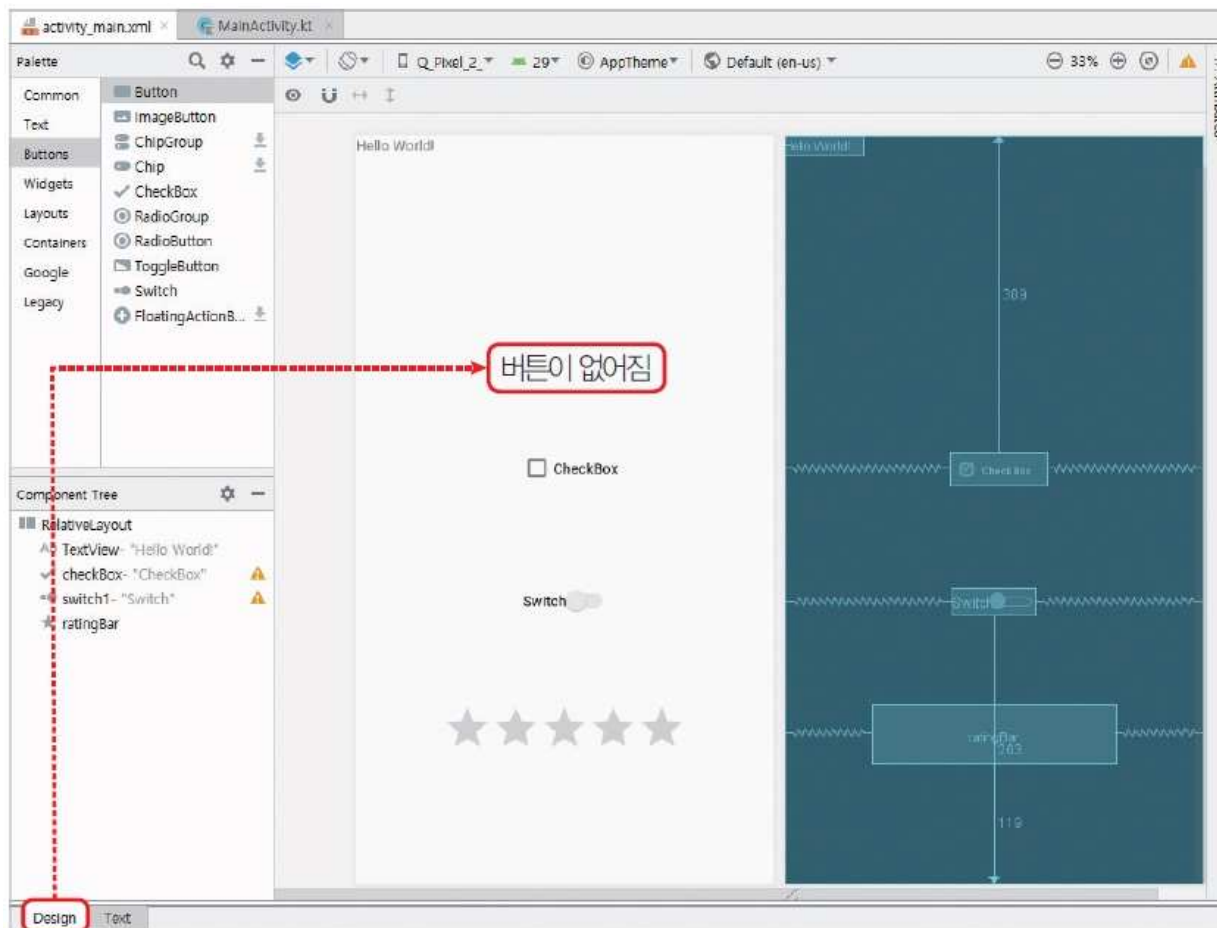


그림 2-13 그래픽 화면 확인

01 Hello Android 프로젝트

- 3 Kotlin 코드 작성 및 수정
 - (13) 왼쪽 Project Tree에서 [java]-[com.cookandroid.helloandroid]-[MainActivity]를 더블클릭하거나 위쪽 [MainActivity.kt] 탭을 클릭함
 - 오른쪽 창의 Kotlin 코드를 수정함

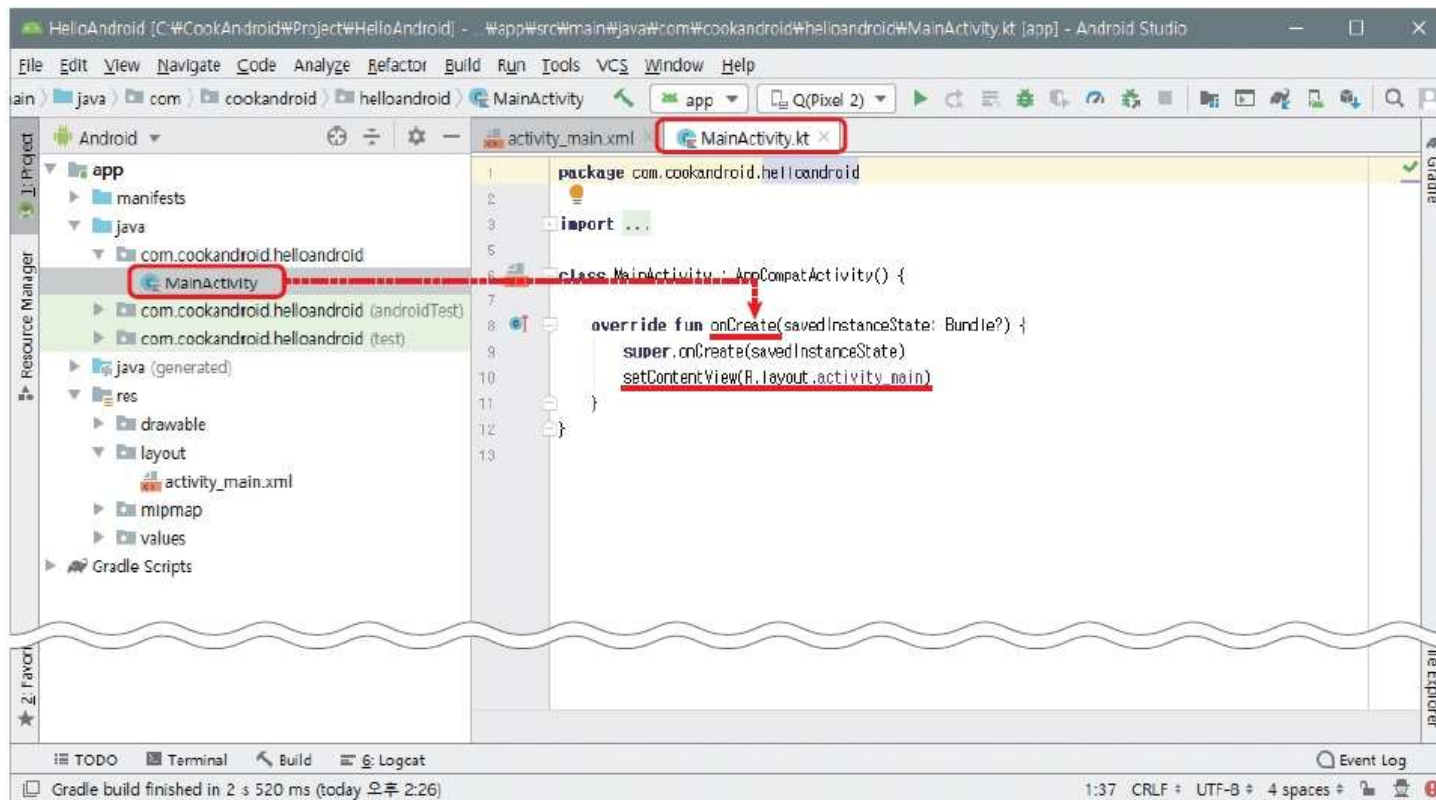


그림 2-14 메인 Kotlin 코드 확인

01 Hello Android 프로젝트

- 4 프로젝트 실행 및 결과 확인
 - (14) 메뉴의 [Run]-[Run 'app']을 선택하거나 [Run 'app'] 아이콘을 클릭함

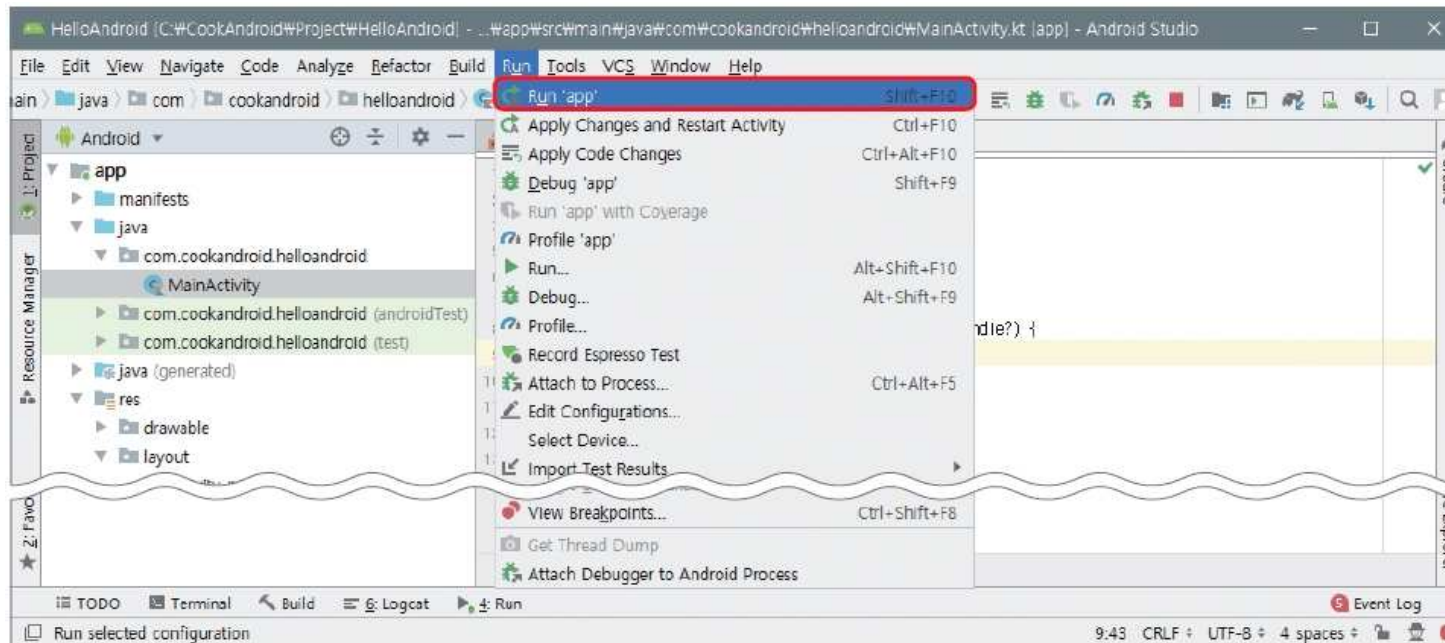


그림 2-15 애플리케이션 실행

01 Hello Android 프로젝트

- (15) AVD가 여러 개이거나 스마트폰이 연결되어 있다면 먼저 작동할 장치를 상단 중앙의 툴바에서 선택할 수 있음

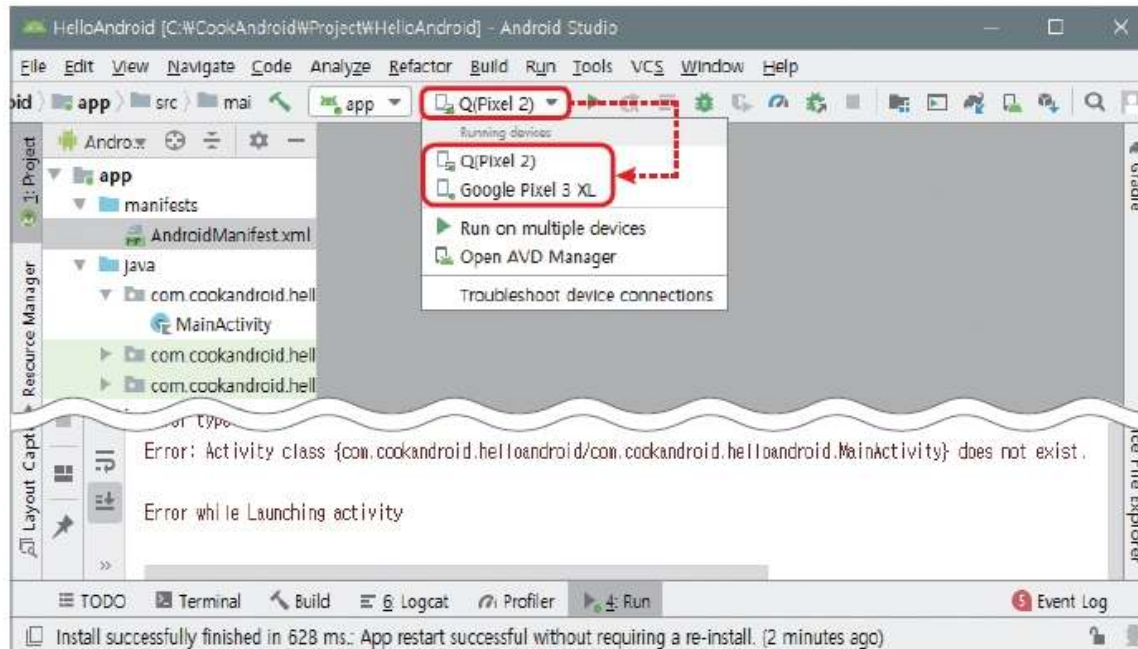


그림 2-16 실행할 안드로이드 장치 선택

01 Hello Android 프로젝트

- (16) AVD가 부팅된 후 잠시 동안 기다리면 실행 결과 화면이 나타남
 - 만약 오류가 발생하거나 원하는 결과가 안 나오면 [그림 2-2]의 ❷번이나 ❸번으로 돌아감

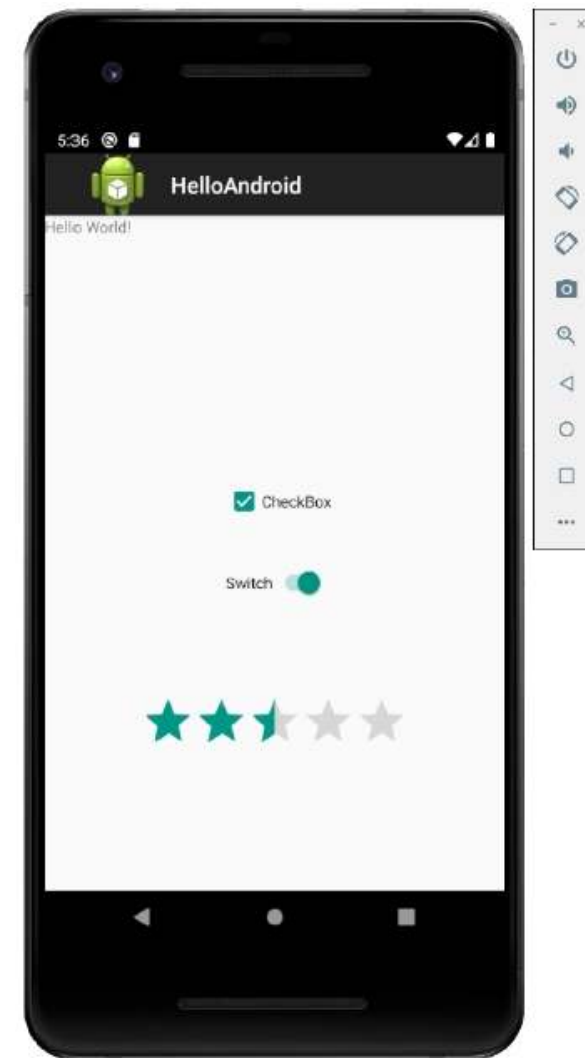


그림 2-17 프로젝트 실행 결과

01 Hello Android 프로젝트

- 5 안드로이드 애플리케이션 개발 완료
 - (17) 복잡한 애플리케이션 개발도 기본적으로는 이와 같은 방식으로 진행함
 - AVD의 초기화면으로 돌아가려면 오른쪽 키패드에서 돌아가기 버튼을 클릭하거나 AVD 아래의 맨 왼쪽 버튼(◀)을 클릭함

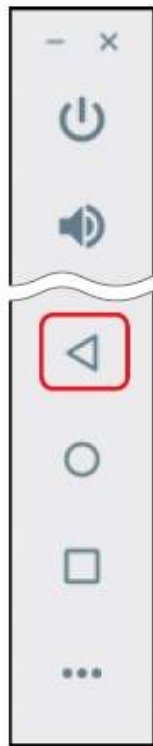


그림 2-18 돌아가기 버튼

01 AVD의 명칭

■ AVD의 기본적인 명칭



그림 2-21 AVD의 명칭

02 AVD 사용법

■ <실습 2-2> AVD 사용법 알아보기

■ 화면 전환

- (1) AVD를 초기화면 상태로 둠
 - 초기화면이 아니라면 홈 버튼을 누름
- (2) AVD의 화면을 가로로 바꾸려면 <ctrl> + <←> 또는 <ctrl> + <→>를 누름
- (3) 다시 <ctrl> + <←> 또는 <ctrl> + <→>를 눌러서 세로로 돌려놓음



그림 2-22 초기화면

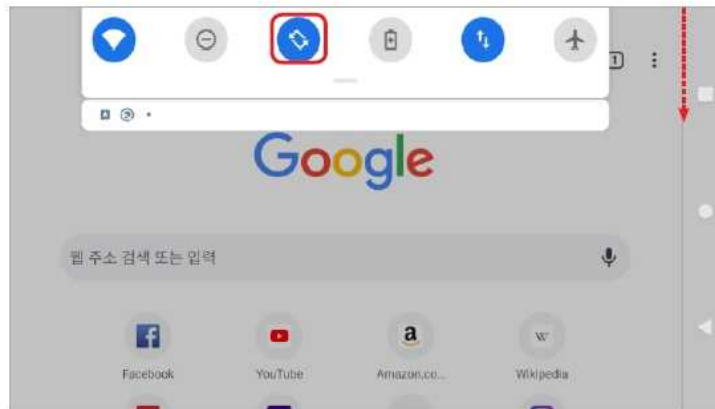


그림 2-23 가로로 전환한 화면(크롬 앱을 실행하고 자동 회전을 켜)

02 AVD 사용법

- 알람 추가
 - (4) 초기화면을 위쪽으로 스와이프하여 [애플리케이션] 화면에서 시계 버튼을 클릭함
 - (5) 알람을 추가할 수 있는 화면이 나옴
 - (6) 아래쪽의 (+) 버튼을 클릭하여 알람 시간을 추가할 수 있음

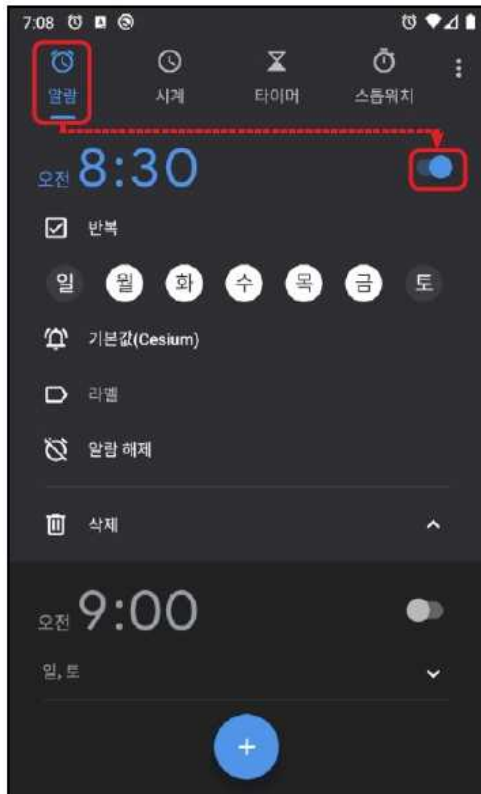


그림 2-24 알람 시간 추가

02 AVD 사용법

- 카메라와 갤러리
 - (8) 초기화면을 위쪽으로 스와이프하고 카메라 버튼을 클릭하여 메시지 창이 나오면 <항상 허용>을 클릭함
 - [Entering Camera Mode] 메시지 창이 나오면 <GOT IT>을 클릭함
 - 실제 카메라는 없지만 가상의 카메라 화면이 나옴
 - 셔터 버튼을 클릭하면 가상 화면이 촬영됨

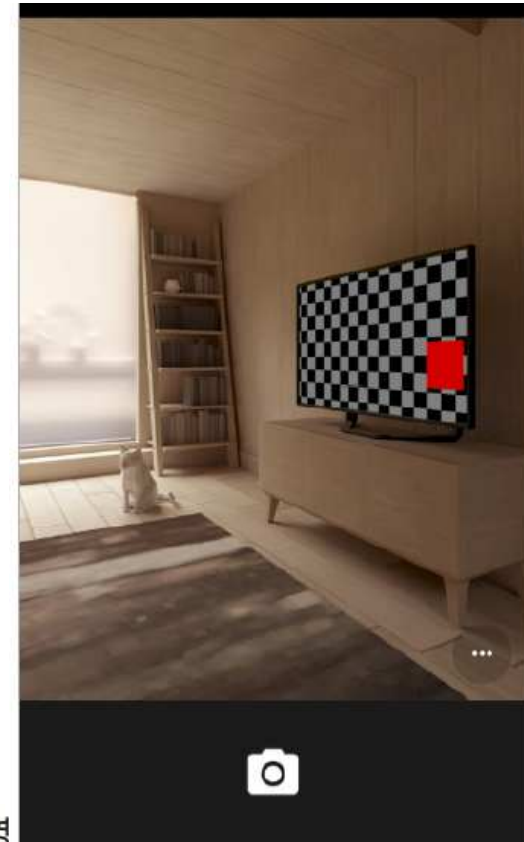


그림 2-25 가상 카메라 촬영

02 AVD 사용법

- (10) 초기화면을 위쪽으로 스와이프하고 [포토]를 클릭하면 백업과 관련된 내용이 나옴
 - 사진 편집/변경 및 삭제, 공유 등의 작업이 가능함.

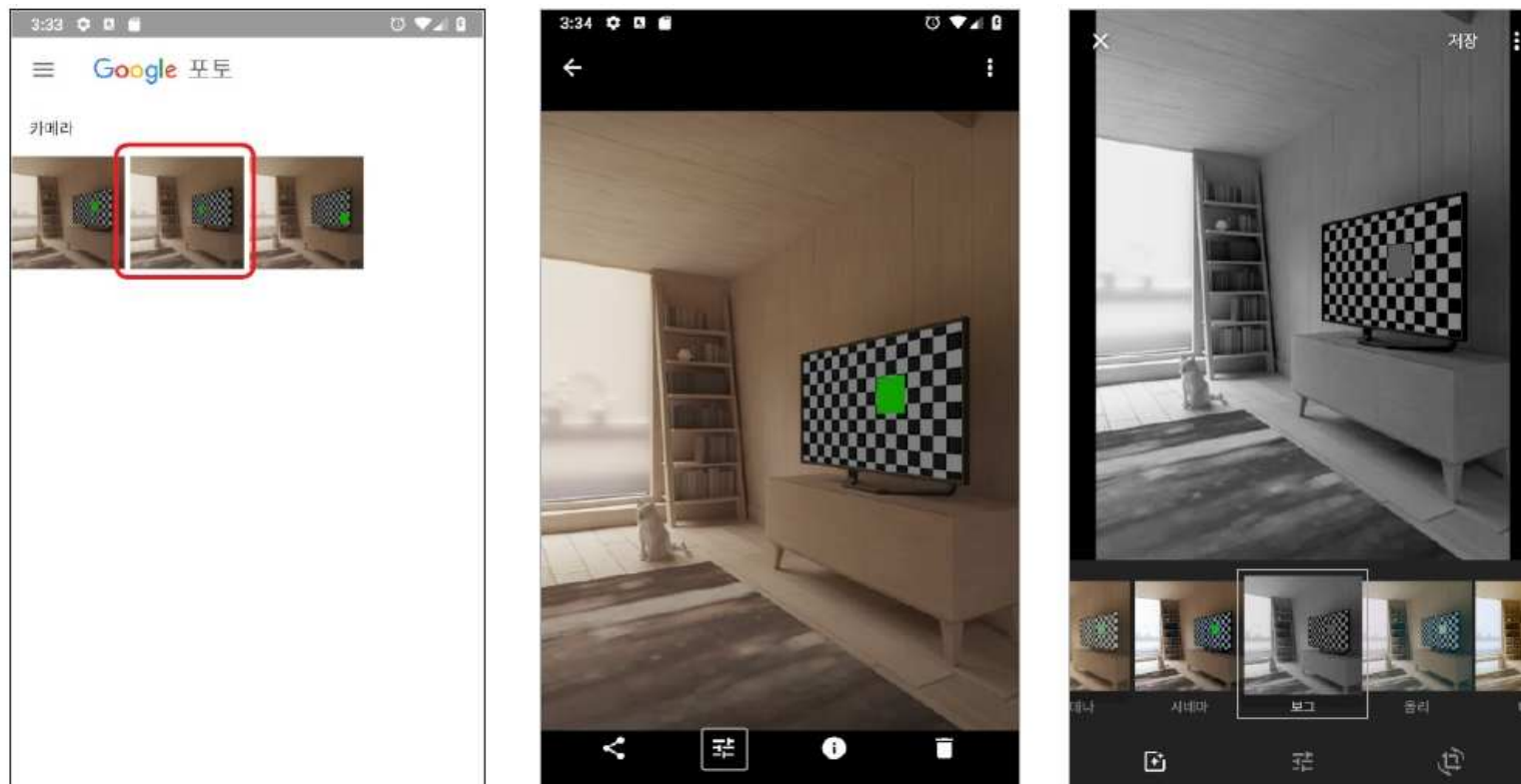


그림 2-26 촬영된 사진 확인

02 AVD 사용법

- 웹브라우저
 - (12) 크롬 버튼을 클릭함
 - (13) 처음에는 구글 사이트에 접속됨. '웹 주소 검색 또는 입력' 부분에 주소(예: m.daum.net)를 입력하고 <Enter>를 누르면 해당 페이지로 이동함
 - 나머지는 일반 웹 브라우저와 동일함

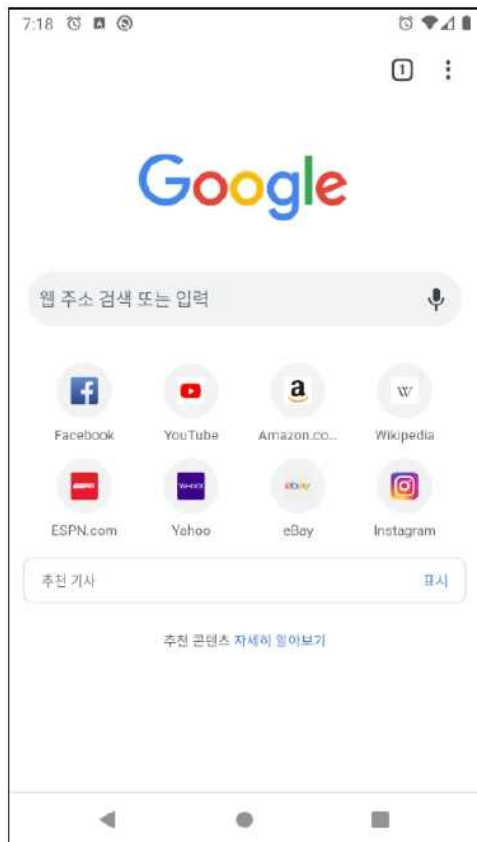


그림 2-27 웹 서핑

02 AVD 사용법

- 배경화면 변경
 - (15) 바탕화면을 몇 초 동안 누른 후 [배경화면]을 선택하면 제공되는 다양한 그림 중에서 고를 수 있음

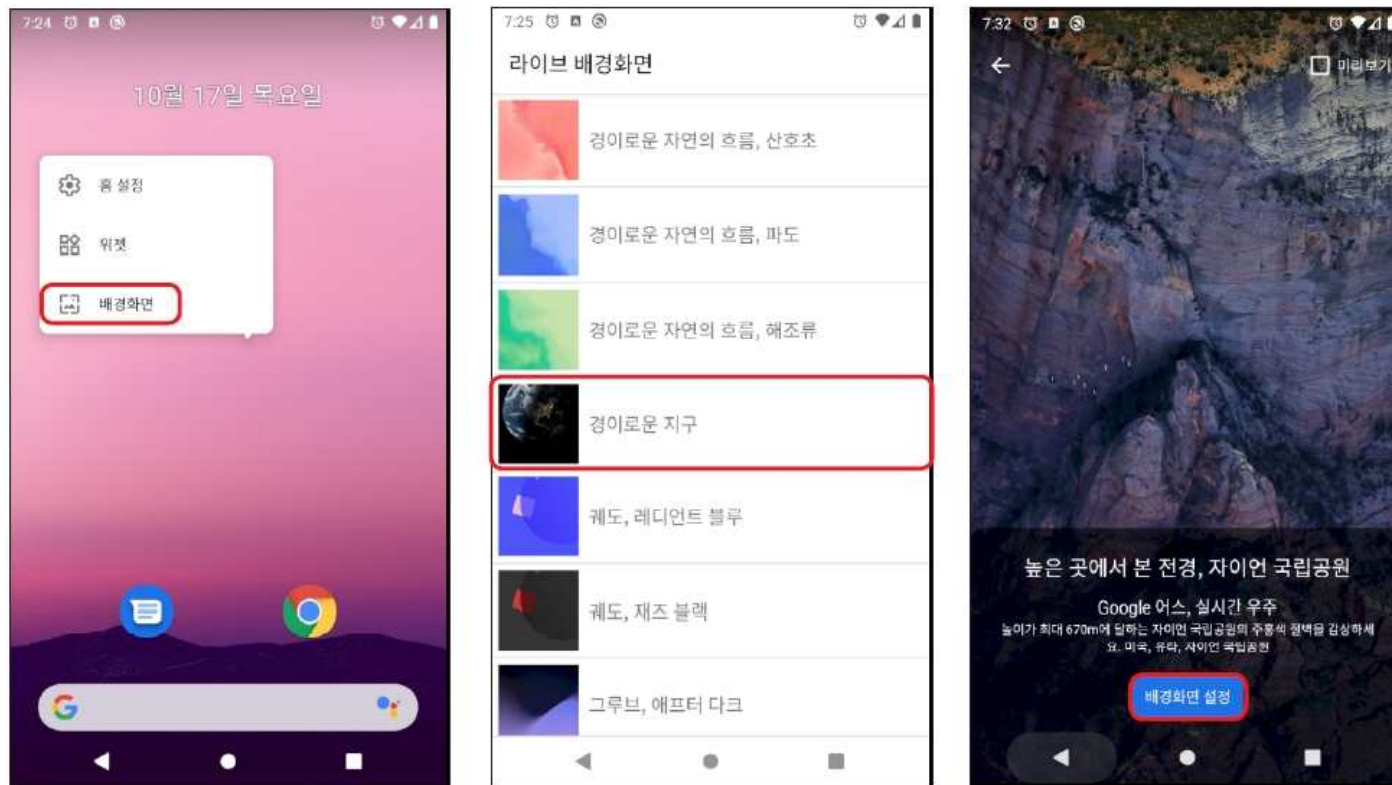


그림 2-28 배경화면 변경

02 AVD 사용법

- 오버뷰 버튼
 - (16) 안드로이드는 앱을 실행하다가 돌아가기 버튼이나 홈 버튼을 클릭하면 화면에서 사라짐
 - 사라진 앱은 종료된 것이 아니라 화면의 뒤쪽에 숨어있음
 - 오버뷰 버튼: 실행 중인 전체 앱을 보여주고 앱을 화면 앞으로 가져오거나 종료하는 기능을 수행함

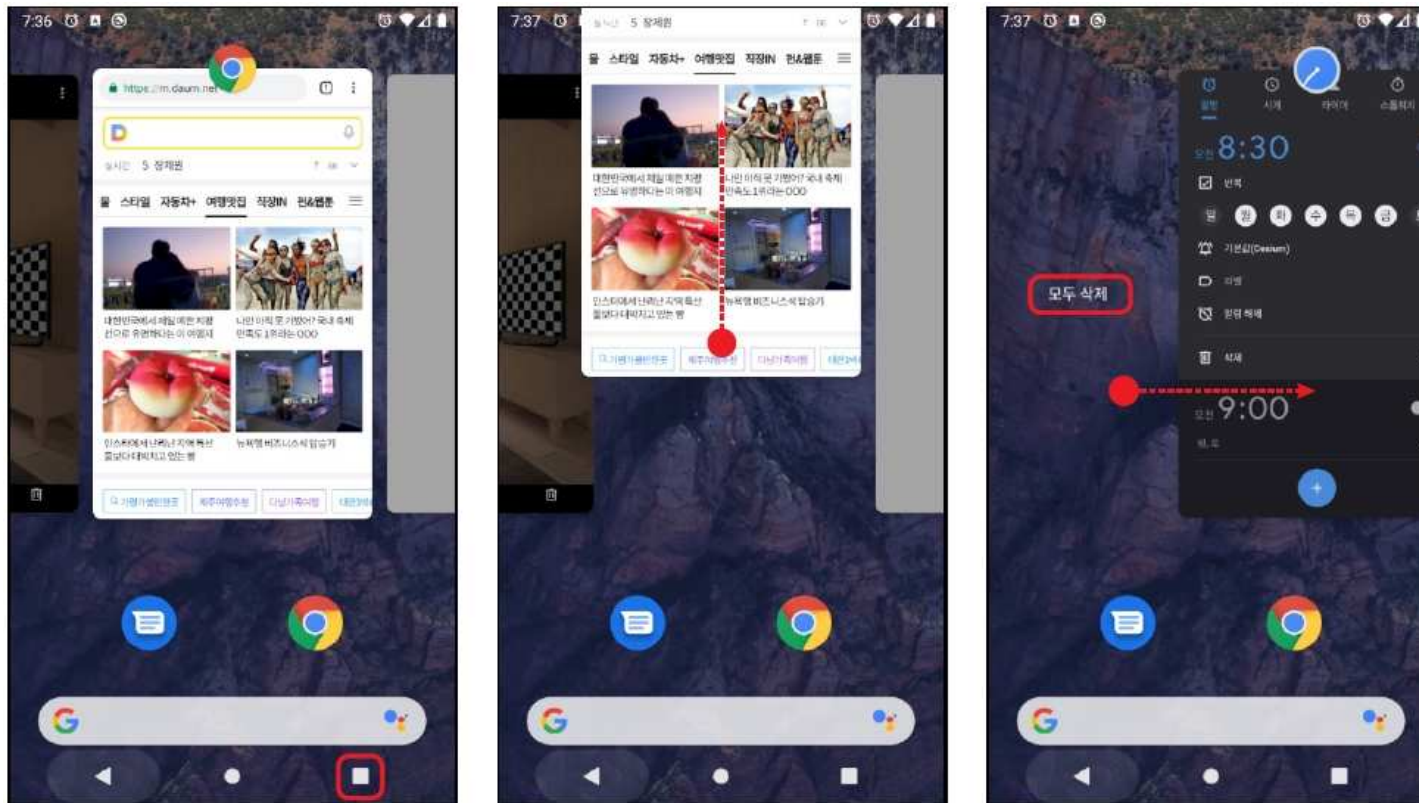


그림 2-29 실행 중인 프로그램 종료

02 AVD 사용법

- (17) 각각의 앱을 위로 스와이프하면 앱이 종료됨
- (18) 앱을 클릭하면 해당 앱이 화면 앞으로 나와서 실행됨
- (19) 맨 왼쪽으로 스와이프하여 <모두 제거>를 클릭하면 모든 앱이 종료됨

02 AVD 사용법

- 바탕화면에 애플리케이션 복사
 - (20) 초기화면을 위쪽으로 스와이프하여 앞서 작성한 HelloAndroid 아이콘을 몇 초 동안 누르면 <앱 정보>가 나옴
 - 다시 몇 초 동안 누르고 위로 움직이면 바탕화면으로 복사되며, 바탕화면의 원하는 위치에 놓을 수 있음

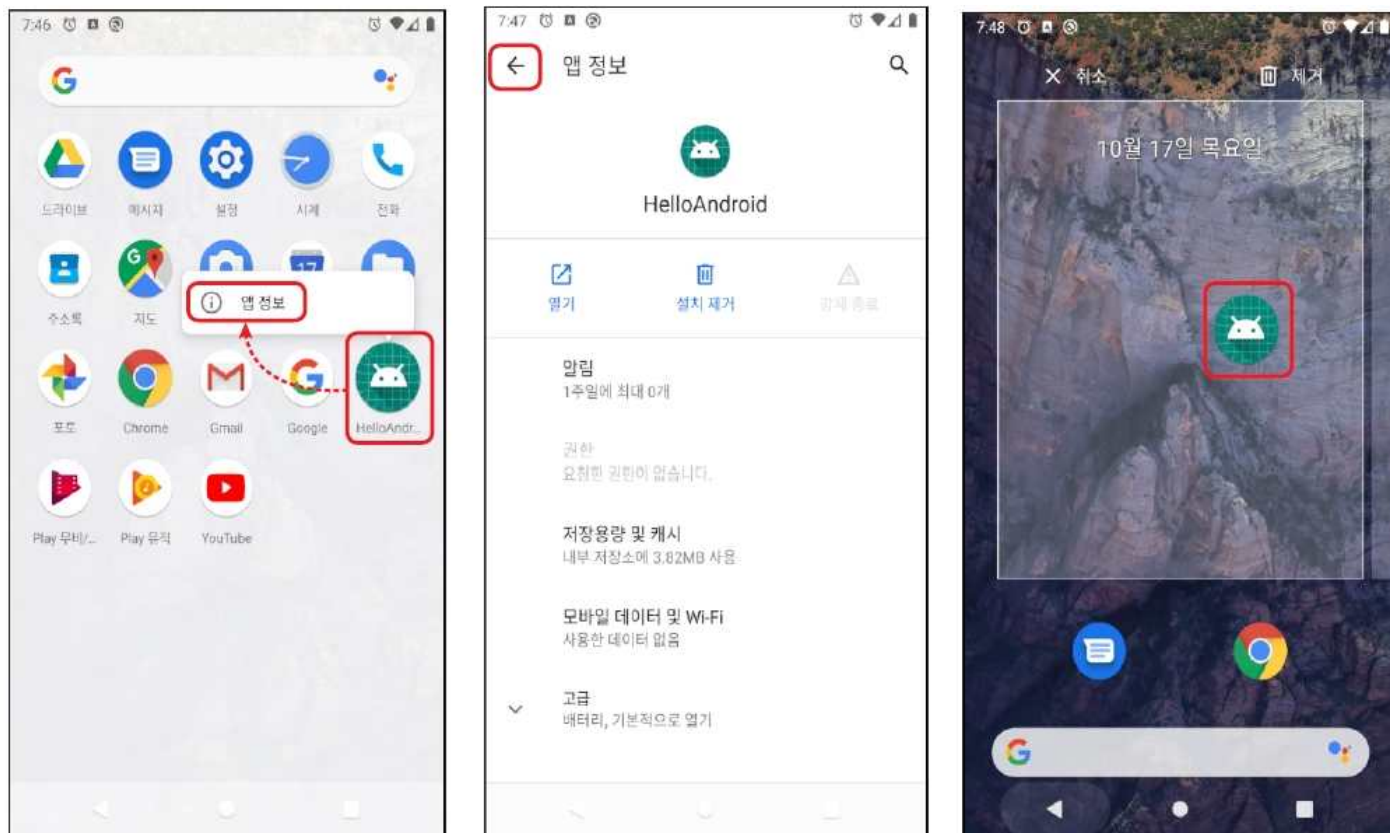


그림 2-30 바탕화면에 애플리케이션 복사

02 AVD 사용법

- (21) 바탕화면에 복사한 아이콘을 삭제하려면 아이콘을 몇 초 동안 누른 후 상단의 <삭제>로 가져다 놓음
 - 이때 애플리케이션 자체가 삭제되는 것이 아니라 아이콘만 삭제됨
 - 아이콘을 <제거>로 가져다 놓으면 애플리케이션 자체가 AVD에서 완전히 제거됨

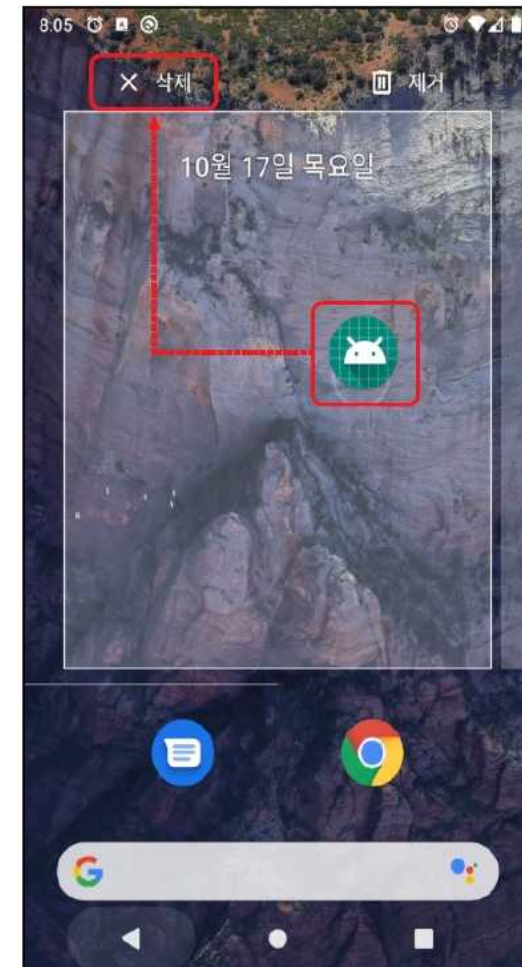


그림 2-31 바탕화면의 아이콘 삭제

01 Android Studio 프로젝트 관리

■ <실습 2-3> 안드로이드 프로젝트 관리하기

■ 프로젝트 닫기와 열기

- (1) 왼쪽의 Recent Project에 지금까지 작업한 프로젝트가 보임
 - 프로젝트 이름 바로 아래에서는 작은 글씨로 프로젝트가 저장된 폴더도 확인할 수 있음
 - 프로젝트 이름을 클릭하면 프로젝트가 열리고 작업을 진행할 수 있음

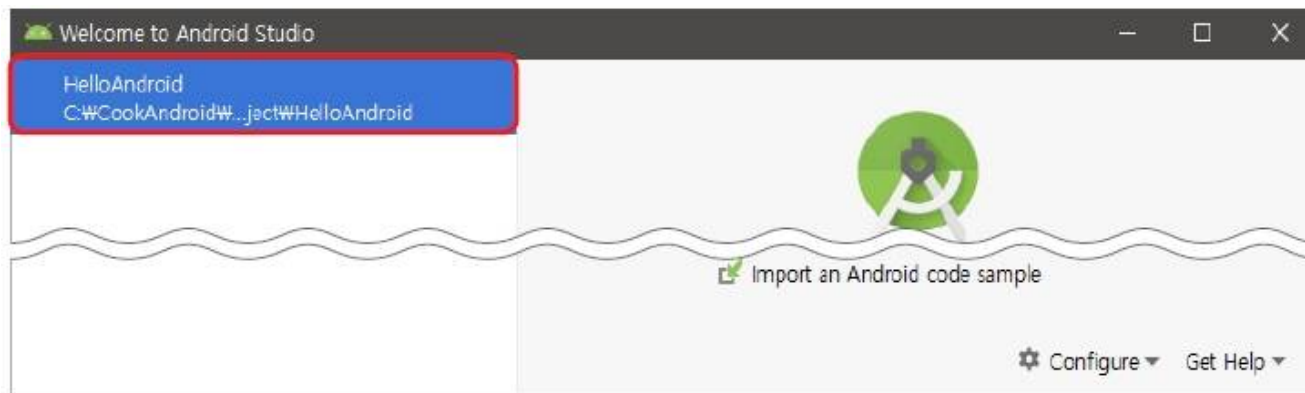


그림 2-32 최근에 사용한 프로젝트 열기 1

01 Android Studio 프로젝트 관리

- (2) 여러 개의 프로젝트를 열고 작업하려면 Android Studio 메뉴의 [Open]을 클릭하고 열고자 하는 프로젝트 폴더를 선택함
 - <New Window>를 클릭하면 새로운 창이 열려서 여러 개의 프로젝트를 동시에 작업할 수 있음

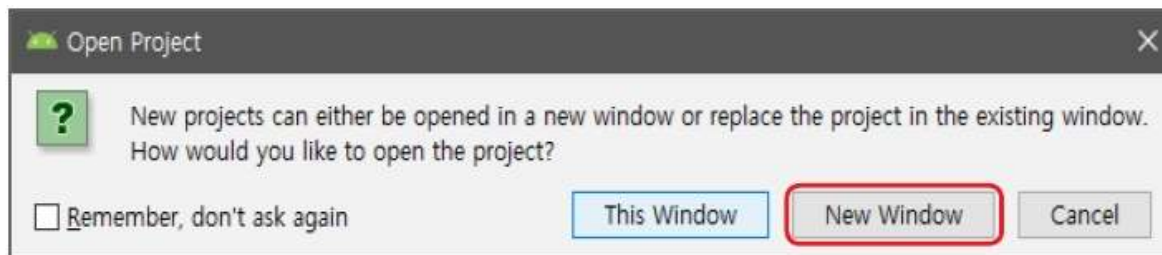


그림 2-33 최근에 사용한 프로젝트 열기 2

- (3) Android Studio 메뉴의 [File]-[Close Project]를 선택하여 열린 창을 모두 닫음

01 Android Studio 프로젝트 관리

- (4) 폴더를 검색하여 프로젝트를 열기
 - Android Studio 초기화면에서 [Open an existing Android Studio project]를 클릭하고 프로젝트가 저장된 폴더를 지정함

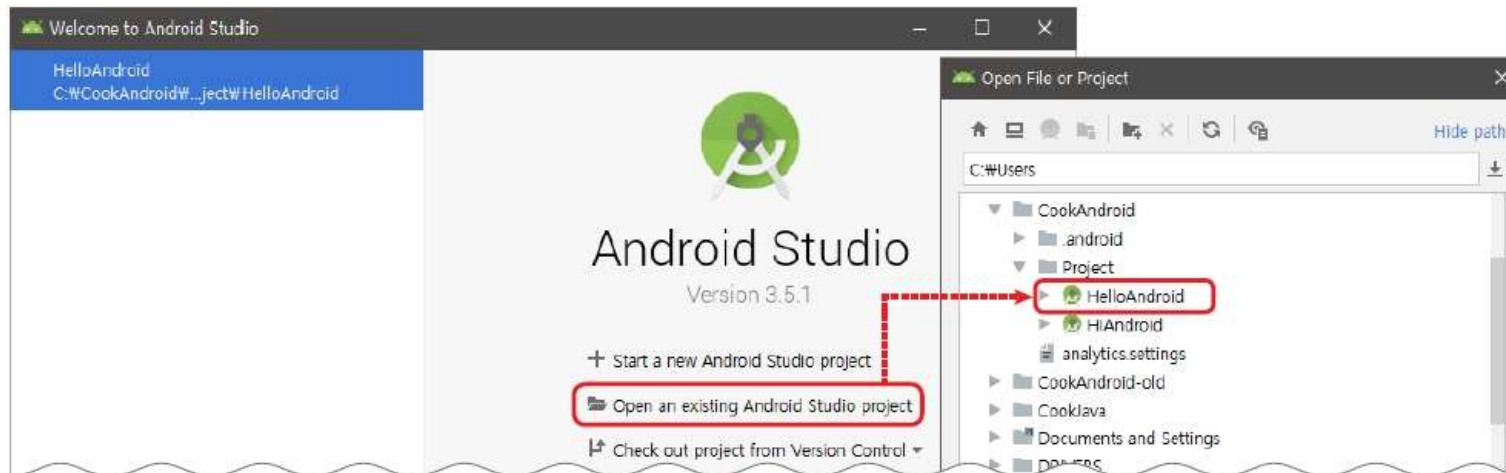


그림 2-34 폴더에서 직접 프로젝트 열기

01 Android Studio 프로젝트 관리

- Android Studio 프로젝트 내보내기/가져오기
 - (5) Android Studio 프로젝트는 내보내기 기능이 따로 없음
 - 프로젝트가 생성된 폴더를 통째로 복사하거나 압축하여 다른 사람에게 또는 컴퓨터로 보내야 함

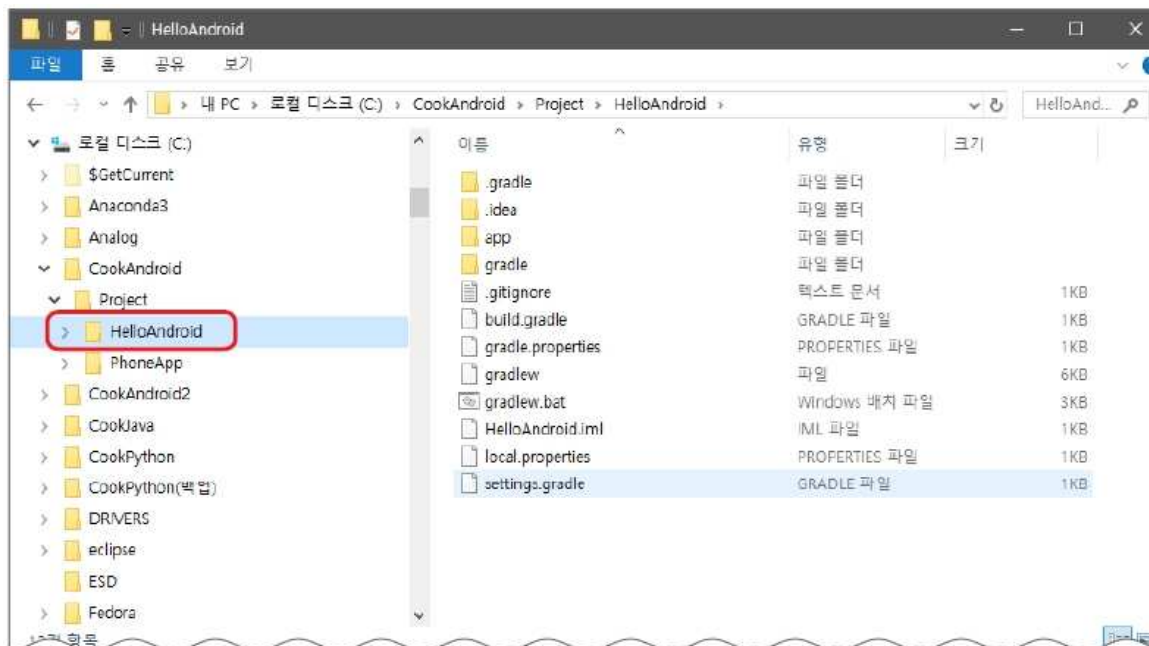


그림 2-35 프로젝트 폴더를 통째로 복사

- (6) Android Studio 프로젝트는 가져오기 기능 또한 따로 없으므로 복사해 온 프로젝트를 (4)번처럼 열어서 사용함

01 Android Studio 프로젝트 관리

- Eclipse용 프로젝트 가져오기
 - (7) Eclipse에서 작성된 프로젝트를 가져올 수 있음
 - Android Studio 초기화면의 [Import project (Gradle, Eclipse ADT, etc.)]를 선택하고 Eclipse에서 작성된 프로젝트 폴더를 선택함

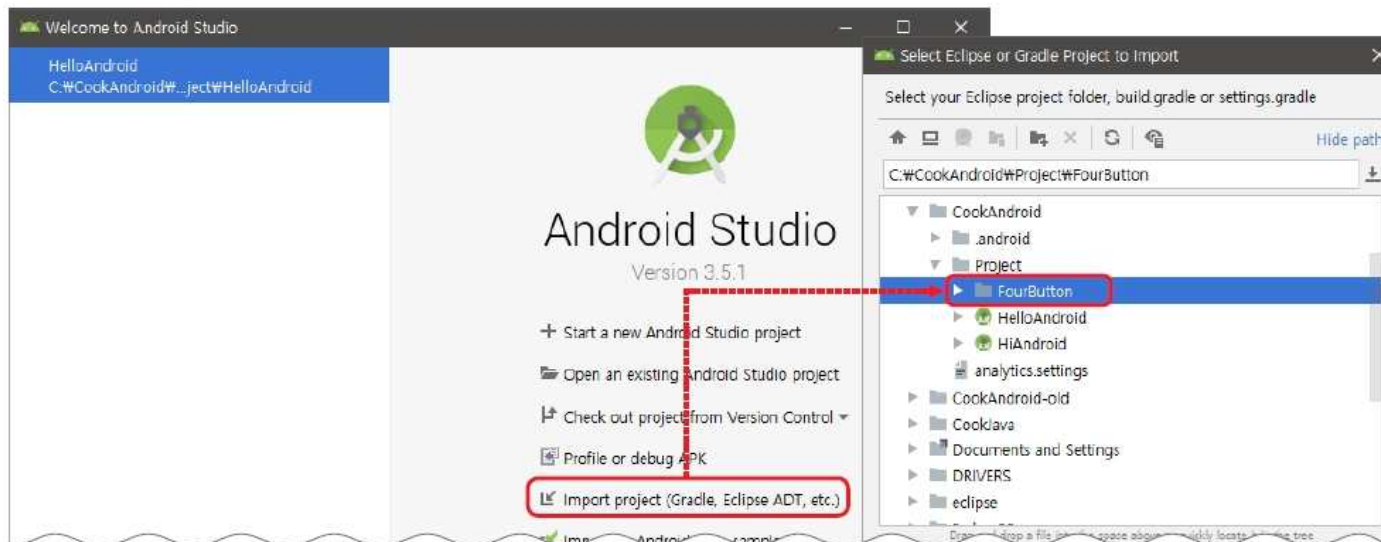


그림 2-36 Eclipse용 프로젝트 가져오기 1

01 Android Studio 프로젝트 관리

- (8) 가져올 폴더를 지정하고 <Next>를 클릭함

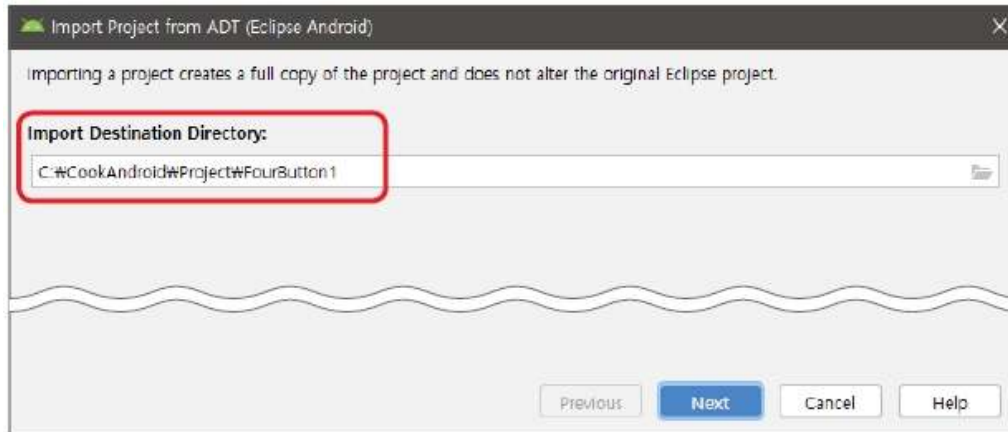


그림 2-37 Eclipse용 프로젝트 가져오기 2

- (9) 프로젝트를 변환하는 옵션
 - 모두 체크된 상태에서 <Finish>를 클릭함

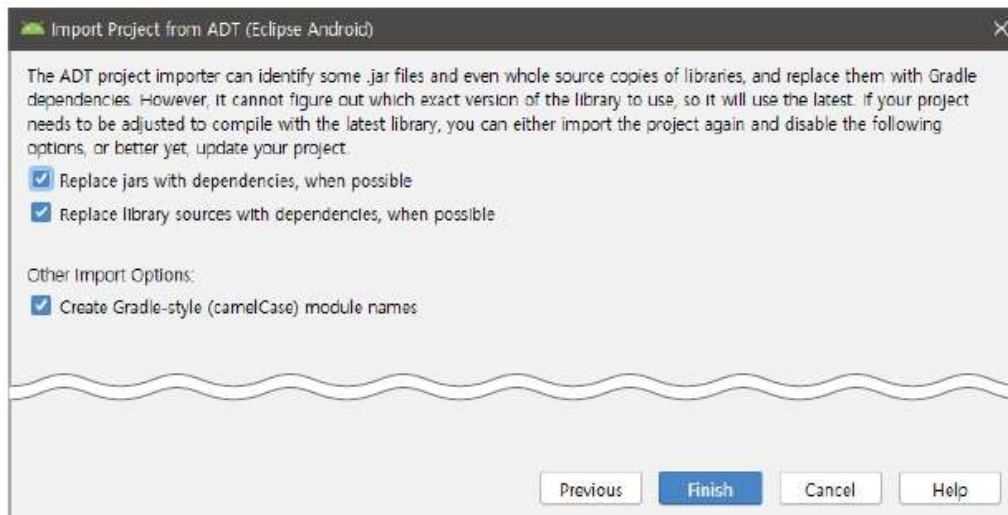


그림 2-38 Eclipse용 프로젝트 가져오기 3

01 Android Studio 프로젝트 관리

- (10) 변환이 완료된 Eclipse 프로젝트는 Android Studio에서 생성한 프로젝트와 같은 방법으로 사용하면 됨

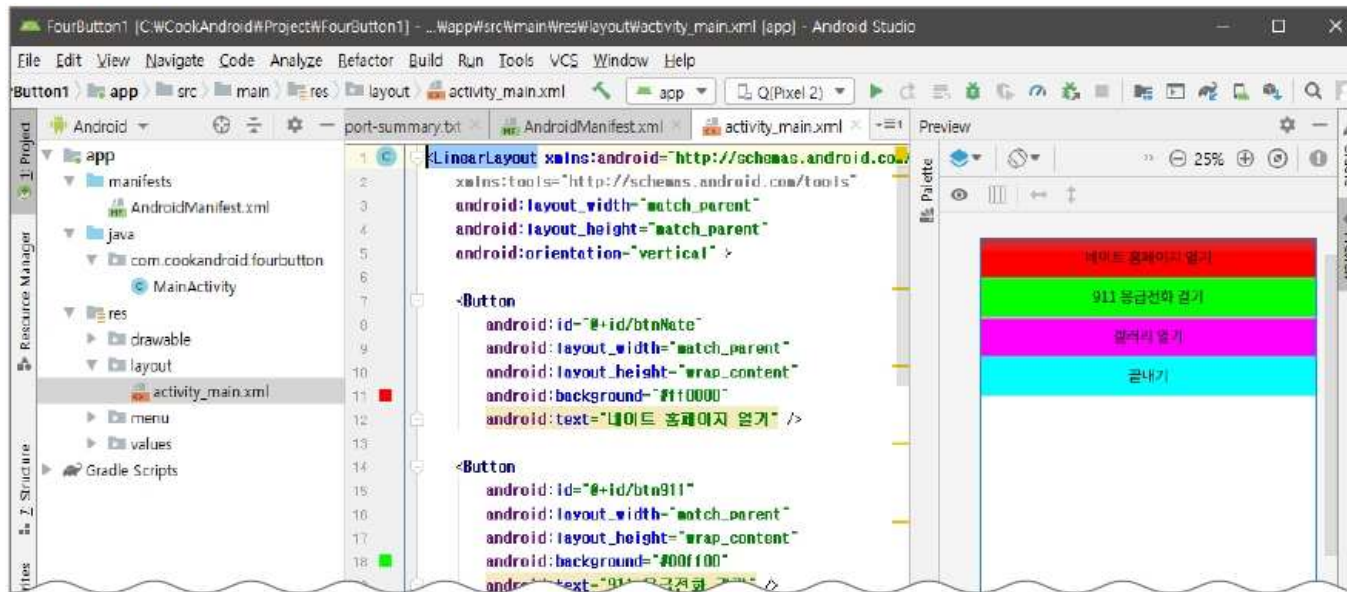


그림 2-39 변환 완료된 Eclipse용 프로젝트

02 안드로이드 프로젝트의 표준 틀

- 이번 프로젝트를 통해 다음 내용을 파악할 수 있음
 - Android Project 사용법과 자동 완성 기능
 - 화면 구성을 위한 XML 파일의 문법
 - Kotlin 코드 코딩 방법
 - activity_main.xml 파일에 포함된 위젯의 접근 방법
 - 위젯의 이벤트 발생 시 작동하는 코드 작성법
 - R.java의 내용

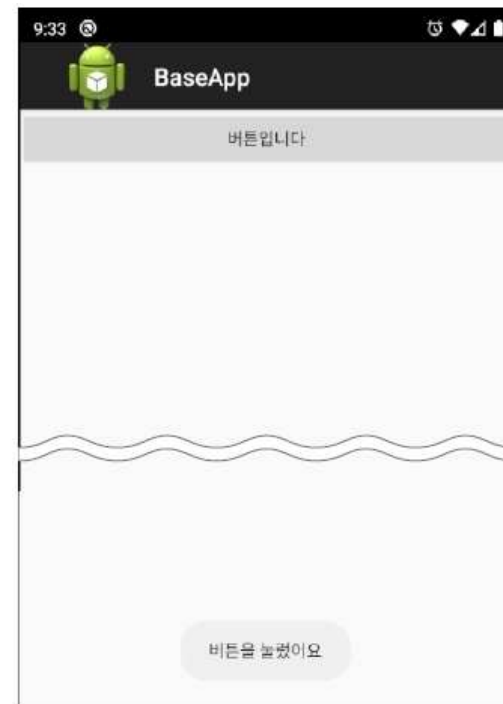


그림 2-40 안드로이드 프로젝트 표준 틀 실행 결과

02 안드로이드 프로젝트의 표준 틀

■ <실습 2-4> 기본적인 애플리케이션 작성하기

■ 1 안드로이드 프로젝트 생성

- (1) Android Studio 메뉴의 [File]-[New Project]를 선택
 - 또는 Android Studio 초기화면의 [Start a new Android Studio project] 클릭

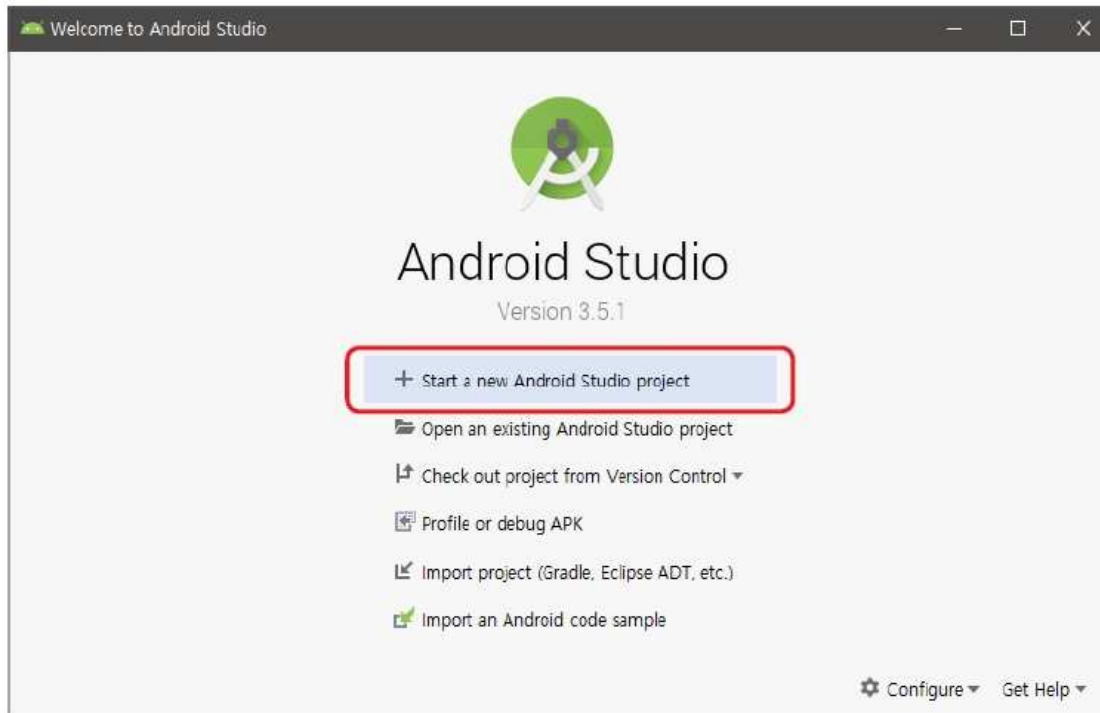


그림 2-41 새 프로젝트 생성

02 안드로이드 프로젝트의 표준 틀

- (2) [Choose your project]에서 [Phone and Tablet]을 클릭하고 'Empty Activity'를 선택함
 - 'Empty Activity' 외에도 다양한 형태의 기본 틀을 지정할 수 있음

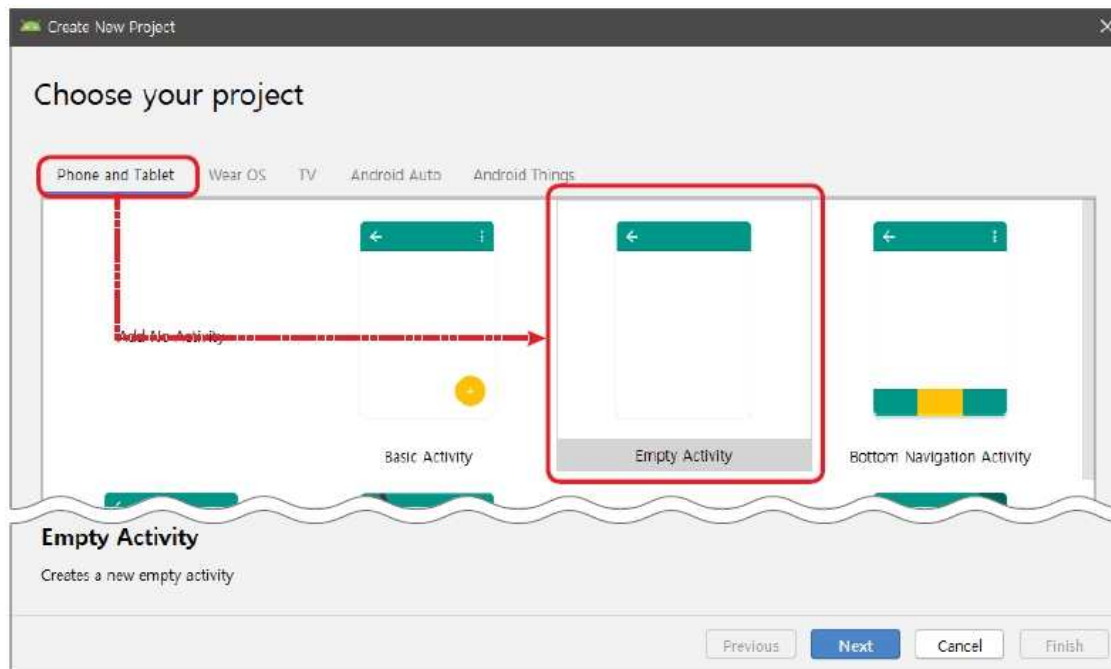


그림 2-42 프로젝트 형태 선택

02 안드로이드 프로젝트의 표준 틀

- (3) [Configure your project]에서 프로젝트에 대한 정보를 입력하거나 선택함
 - **Name:** BaseApp
→ 프로젝트 및 응용 프로그램 이름이며 대문자로 시작하는 것이 좋음
 - **Package name:** com.cookandroid.baseapp
→ 도메인 이름(cookandroid.com)과 애플리케이션 이름을 이어서 만듦
 - **Save location:** C:\CookAndroid\Project\BaseApp
→ 프로젝트가 저장될 폴더를 지정함(한글 폴더명은 불가)
 - **Language:** Kotlin → 사용할 프로그래밍 언어는 Kotlin임
 - **Minimum API level:** API 16: Android 4.1 (Jelly Bean)
→ 이 앱이 작동하는 최하 버전을 선택함
- 다른 프로젝트를 새로 생성할 때 기존의 정보를 변경하지 말고 Name만 입력해도 나머지는 자동으로 지정됨

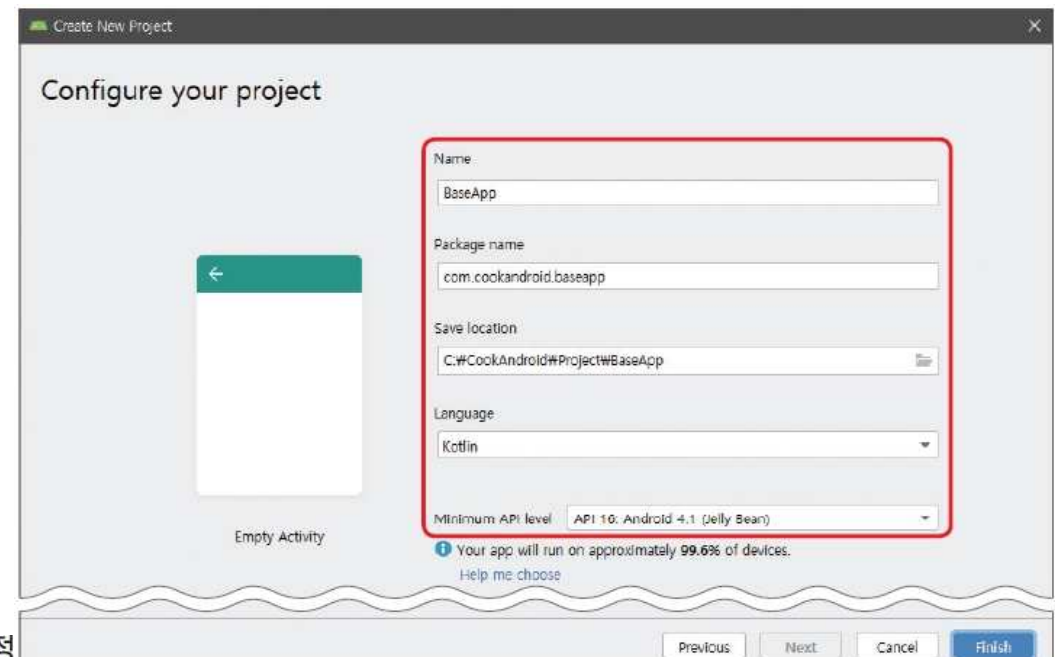


그림 2-43 프로젝트 환경 설정

02 안드로이드 프로젝트의 표준 틀

- 화면 디자인 및 편집
 - (4) Project Tree에서 [java]-[com.cookandroid.baseapp]-[MainActivity]가 기본적으로 열려 있음
 - 위쪽의 [activity_main.xml]을 클릭하고 아래쪽의 [Design] 탭을 클릭하면 초기화면이 보임

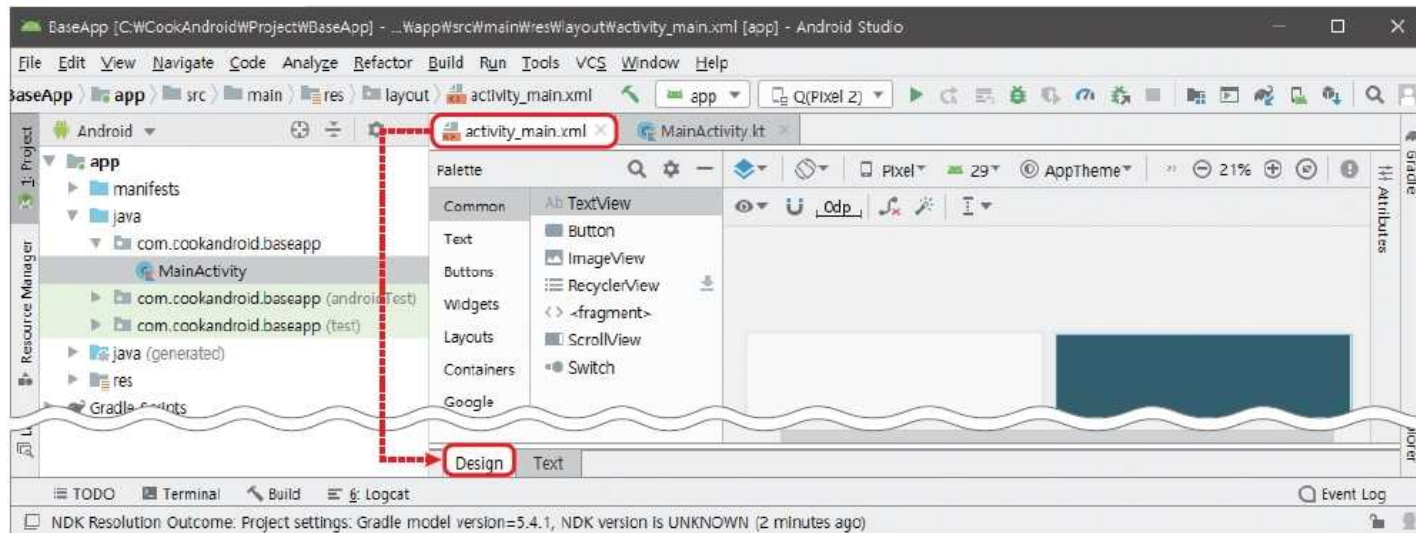


그림 2-44 activity_main.xml 확인

02 안드로이드 프로젝트의 표준 틀

- (5) 화면 아래쪽의 [Text] 탭을 클릭하여 XML 코드를 확인함
 - `<androidx.constraintlayout.widget.ConstraintLayout...>`
`</androidx.constraintlayout.widget.ConstraintLayout>`을
`<LinearLayout...>` `</LinearLayout>`으로 변경
 - LinearLayout에는 필요 없는 `xmlns:app`, `tools:context` 속성을 삭제
 - 앞으로 프로젝트는 대부분 기본 레이아웃을 LinearLayout으로 사용함
 - `<TextView.../>` 부분을 지우기



그림 2-45 XML 코드 변경

02 안드로이드 프로젝트의 표준 틀

- (6) <LinearLayout>과 </LinearLayout> 사이에 버튼을 추가함
 - <Button>을 입력하면 자동으로 </Button>이 붙음
 - 버튼의 시작과 끝을 나타냄



그림 2-46 Button 추가

02 안드로이드 프로젝트의 표준 틀

- (7) 'android:'을 입력하면 자동으로 여러 개를 선택할 수 있는 목록이 나옴
 - 'layout_height'를 더블 클릭해서 선택함

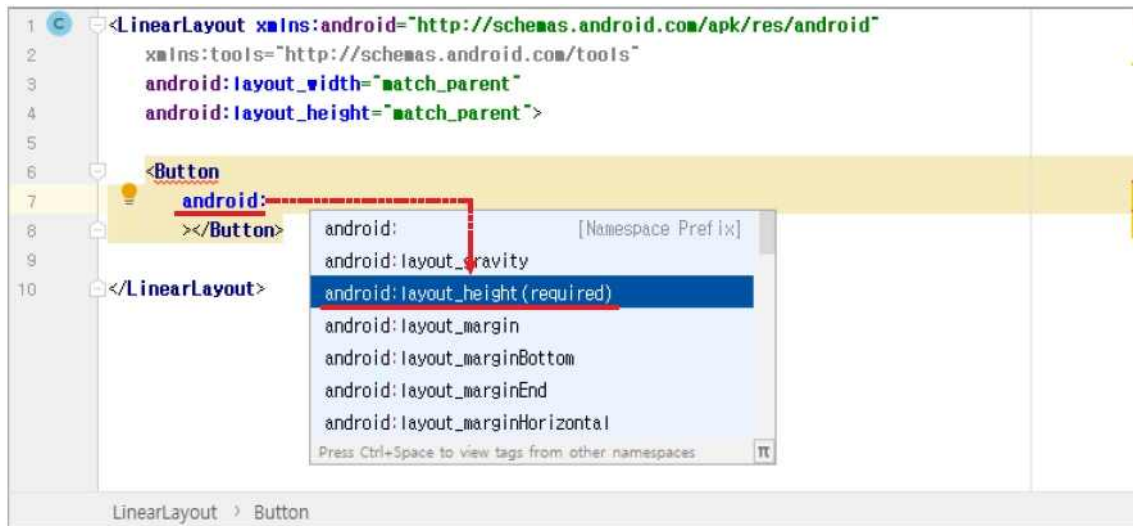


그림 2-47 버튼 속성 편집

02 안드로이드 프로젝트의 표준 틀

- (8) android:layout_height=""가 자동 완성됨
 - 'wrap_content'를 선택하고 <Enter>를 누르면 글자가 자동 입력됨



그림 2-48 버튼 속성 중 layout_height 편집

02 안드로이드 프로젝트의 표준 틀

- (9) 같은 방식으로 나머지 코드를 입력하고 저장함
 - 최종 코드는 [예제 2-1]과 같으며, 저장하면 11행에서 오류가 나타남

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>
```

02 안드로이드 프로젝트의 표준 틀

- (10) 11행의 오류는 strings.xml에 strBtn1이 아직 들어 있지 않기 때문
- 오류를 해결하기 위해 Project Tree의 [app]-[res]-[values][strings.xml]을 더블 클릭하여 xml 코드를 확인함.

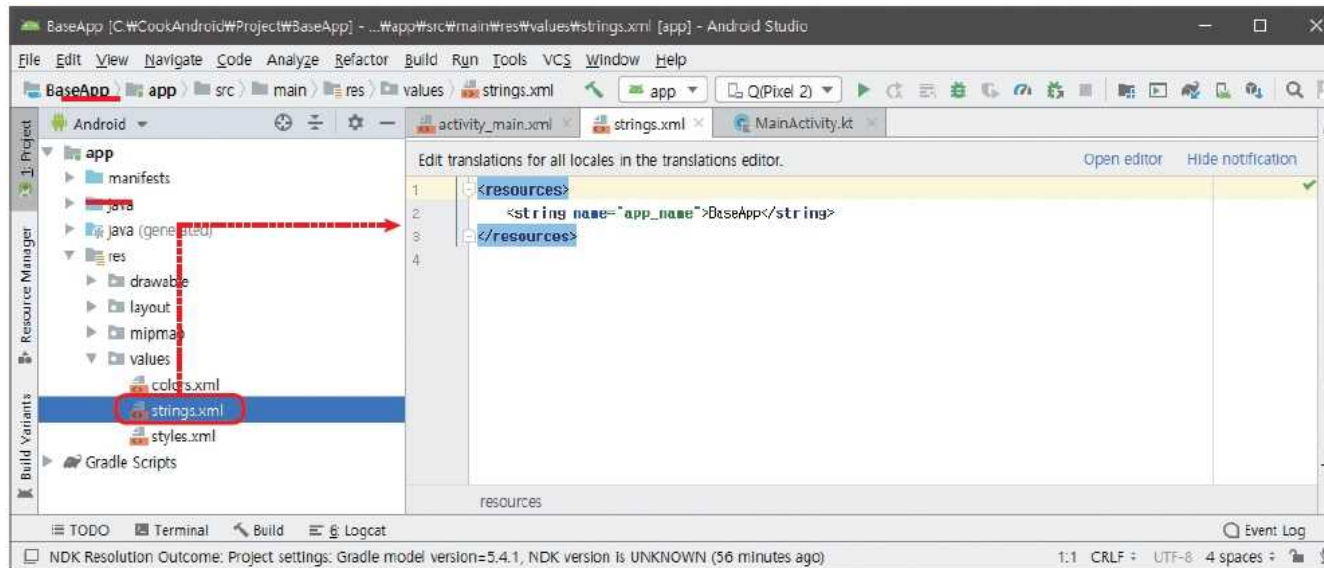


그림 2-49 strings.xml 파일

02 안드로이드 프로젝트의 표준 틀

- (11) activity_main.xml의 11행과 관련된 strBtn1 문자열을 추가한 후 저장하면 빨간색 오류가 사라짐

예제 2-2 strings.xml

```
1 <resources>
2     <string name="app_name">BaseApp</string>
3     <string name="strBtn1">버튼입니다</string>
4 </resources>
```

02 안드로이드 프로젝트의 표준 틀

- (12) 다시 activity_main.xml에서 아래쪽의 [Design] 탭을 클릭하여 그래픽 화면으로 보면 버튼이 1개 추가된 것을 확인할 수 있음

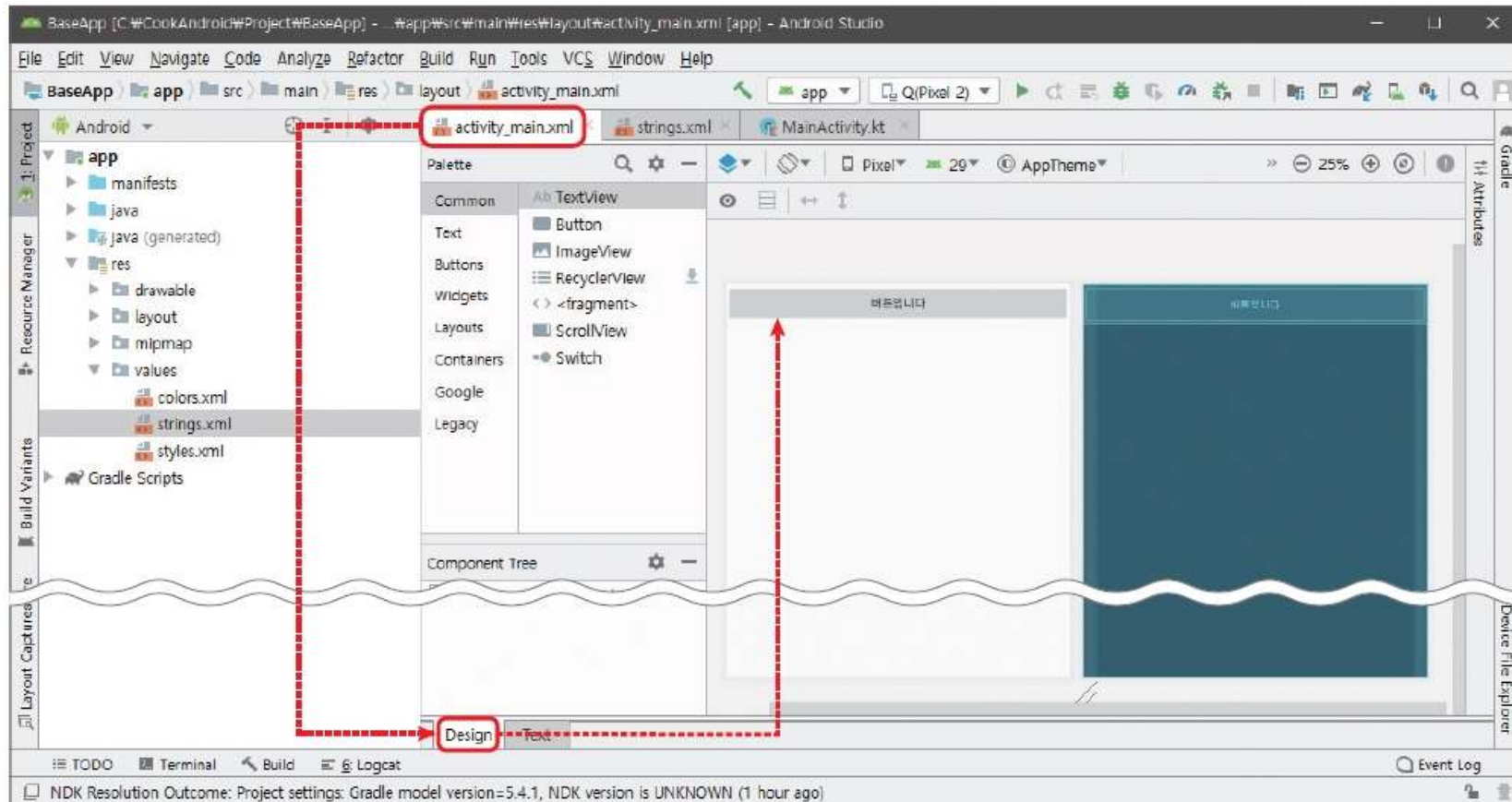


그림 2-50 activity_main.xml 코드의 그래픽 화면

02 안드로이드 프로젝트의 표준 틀

- (13) Android Studio 메뉴의 [Run]-[Run 'app']을 선택하거나 [Run 'app'] 아이콘을 클릭하여 프로젝트를 실행함

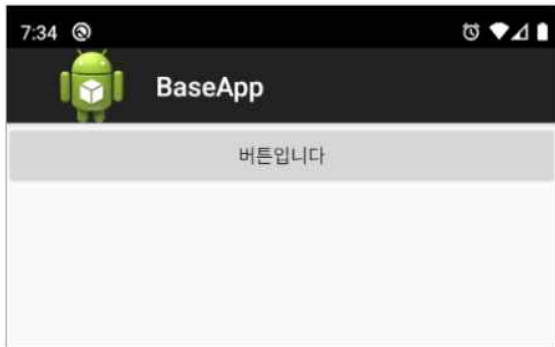


그림 2-52 프로젝트 실행

- (14) 버튼을 터치하면 아무 일도 일어나지 않음
 - 버튼의 모양을 만들었을 뿐, 작동을 위한 코딩은 하지 않았기 때문

02 안드로이드 프로젝트의 표준 틀

- Kotlin 코드 작성 및 수정
 - (15) Project Tree의 [java]-[패키지 이름]-[MainActivity]를 더블 클릭하면 Kotlin 코드가 열림

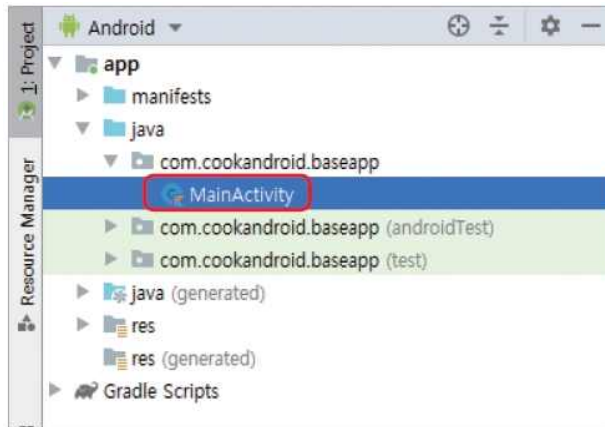


그림 2-53 Kotlin 파일 선택

- (16) 기본적인 코드의 구성은 다음 그림과 같음(행 번호 다를 수 있음)



그림 2-54 MainActivity.kt 기본 코드

02 안드로이드 프로젝트의 표준 틀

- (17) 3행의 import 앞에 있는 작은 '+' 아이콘을 클릭하면 행이 확장됨
 - <Ctrl> + <Alt> + <O> 를 누르면 불필요하게 임포트된 문장이 제거됨



그림 2-55 필요 없는 import 문 삭제

02 안드로이드 프로젝트의 표준 틀

- (18) Kotlin 코드에서 activity_main.xml의 버튼(Button)에 대해 접근해야 하므로 버튼에 대한 멤버변수를 하나 만들기
 - override fun onCreate()메소드 바로 위

```
button1 = findViewById<Button>(R.id.button1)
```

- Button 글자가 빨간색으로 표시되고 [MainActivity.kt] 탭에 빨간 줄이 생김
 - Button 과 관련된 클래스나 인터페이스가 임포트되지 않았기 때문



그림 2-56 Button 변수 추가

02 안드로이드 프로젝트의 표준 틀

- (19) <Alt> + <Enter>를 누르면 Button과 관련된 클래스가 자동으로 import 문에 추가됨
 - 만약 3행의 import 앞 에 '+' 기호가 있으면 클릭하여 확인할 수 있음

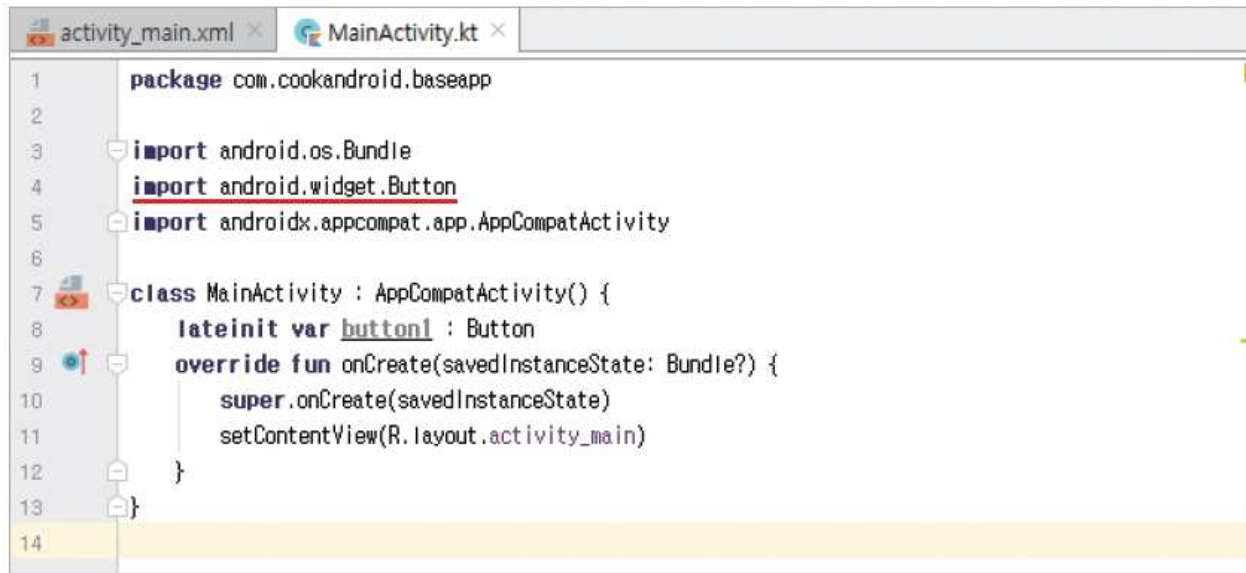


그림 2-57 자동 import 확인

02 안드로이드 프로젝트의 표준 틀

- (20) activity_main.xml 파일에서 만든 버튼 객체에 접근하기
 - findViewById() 메소드를 사용
 - setContentView() 메소드 바로 아래에 다음 코드를 추가

```
button1 = findViewById<Button>(R.id.button1)
```

- (21) button1을 클릭하면 작동하는 이벤트 메소드를 정의
 - 'button1.setOnCl'까지만 입력하면 팝업 창이 나타남
 - 두 번째의 setOnClickListener {...}를 선택함

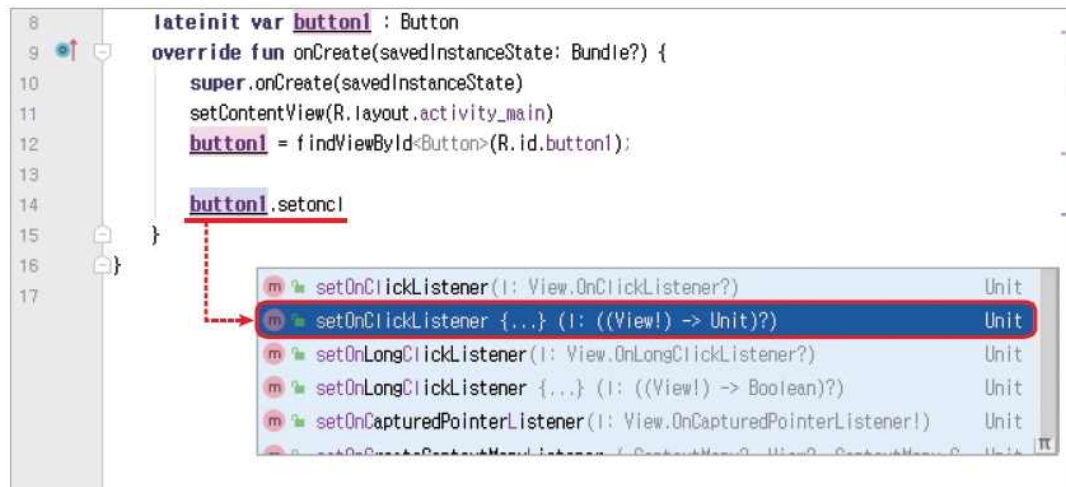


그림 2-58 Kotlin 코드 자동 완성 1

02 안드로이드 프로젝트의 표준 틀

- (22) `setOnClickListener{ }`가 자동 완성됨



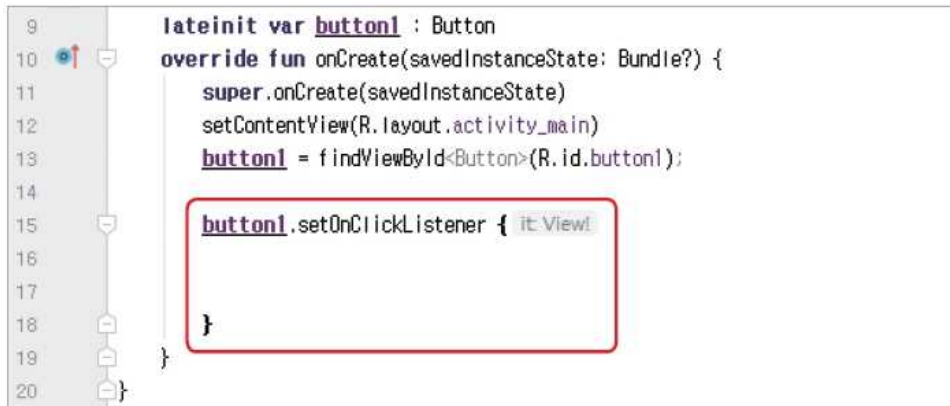
```

8  class MainActivity : AppCompatActivity() {
9      lateinit var button1 : Button
10     override fun onCreate(savedInstanceState: Bundle?) {
11         super.onCreate(savedInstanceState)
12         setContentView(R.layout.activity_main)
13         button1 = findViewById<Button>(R.id.button1);
14
15         button1.setOnClickListener { }
16     }
17 }
18

```

그림 2-59 Kotlin 코드 자동 완성 2

- (23) 버튼을 클릭했을 때 작동할 내용을 `setOnClickListener` 뒤의 `{ }` 안에 작성



```

9      lateinit var button1 : Button
10     override fun onCreate(savedInstanceState: Bundle?) {
11         super.onCreate(savedInstanceState)
12         setContentView(R.layout.activity_main)
13         button1 = findViewById<Button>(R.id.button1);
14
15         button1.setOnClickListener { it: View?
16
17         }
18     }
19 }
20

```

그림 2-60 자동 완성된 코드

02 안드로이드 프로젝트의 표준 틀

- (23) 버튼을 클릭했을 때 화면에 메시지가 잠깐 나오는 코드를 [예제 2-3]의 16~19행에 추가함

예제 2-3 MainActivity.kt

```
1 package com.cookandroid.baseapp
2
3 import android.os.Bundle
4 import android.view.View
5 import android.widget.Button
6 import android.widget.Toast
7 import androidx.appcompat.app.AppCompatActivity
8
9 class MainActivity : AppCompatActivity() {
10     lateinit var button1 : Button
11     override fun onCreate(savedInstanceState: Bundle?) {
12         super.onCreate(savedInstanceState)
13         setContentView(R.layout.activity_main)
14         button1 = findViewById<Button>(R.id.button1)
15
16         button1.setOnClickListener {
17             Toast.makeText(applicationContext, "버튼을 눌렀어요",
18                 Toast.LENGTH_SHORT).show()
19         }
20     }
21 }
```

02 안드로이드 프로젝트의 표준 틀

- 프로젝트 실행 및 결과 확인
 - (25) Android Studio 메뉴의 [File]-[Save All]을 선택하여 현재까지의 내용을 모두 저장함
 - Android Studio 메뉴의 [Run]-[Run 'app']을 선택하거나 [Run 'app'] 아이콘을 클릭하여 프로젝트를 실행하고 버튼을 클릭해 봄

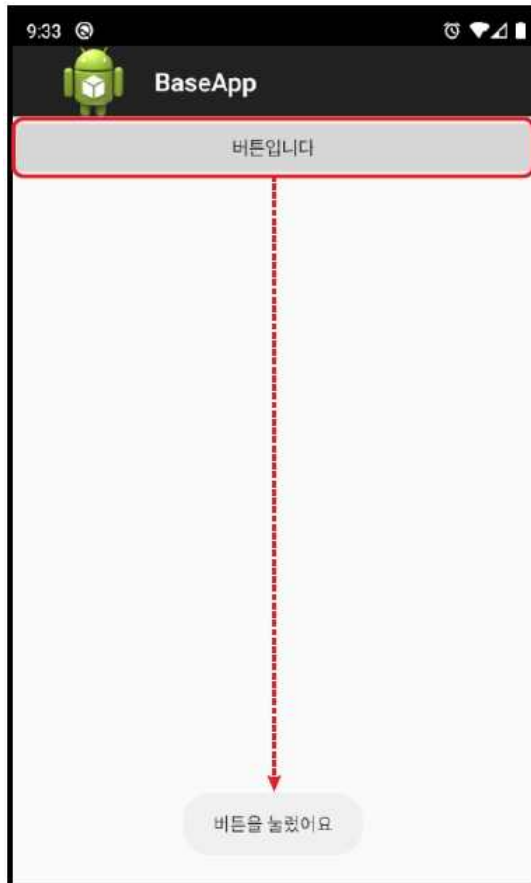


그림 2-61 실행 결과

02 안드로이드 프로젝트의 표준 틀

■ 위젯에 이벤트를 작동하기 위한 동작 순서 요약&응용

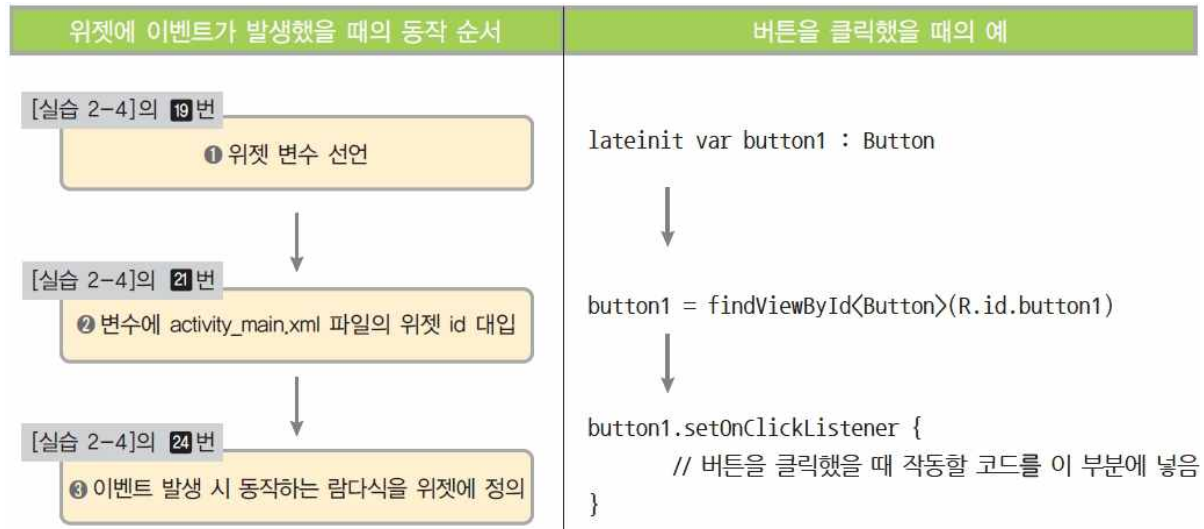


그림 2-63 위젯에 이벤트를 작동하기 위한 코딩 요약

```
// ① 체크박스 변수 선언
lateinit var check1 : CheckBox

// ② 변수에 activity_main.xml 파일의 체크박스를 대입
check1 = findViewById<CheckBox>(R.id.check1)

// ③ 체크박스를 클릭했을 때 작동하는 람다식 정의
check1.setOnClickListener {
    // 체크박스를 클릭했을 때 작동할 코드를 이 부분에 넣음
}
```

02 안드로이드 프로젝트의 표준 틀

▶ 직접 풀어보기 2-3

다음 그림과 같이 버튼 4개를 만들고 각 버튼을 클릭하면 필요한 내용이 작동되는 FourButton 프로젝트를 작성하라.

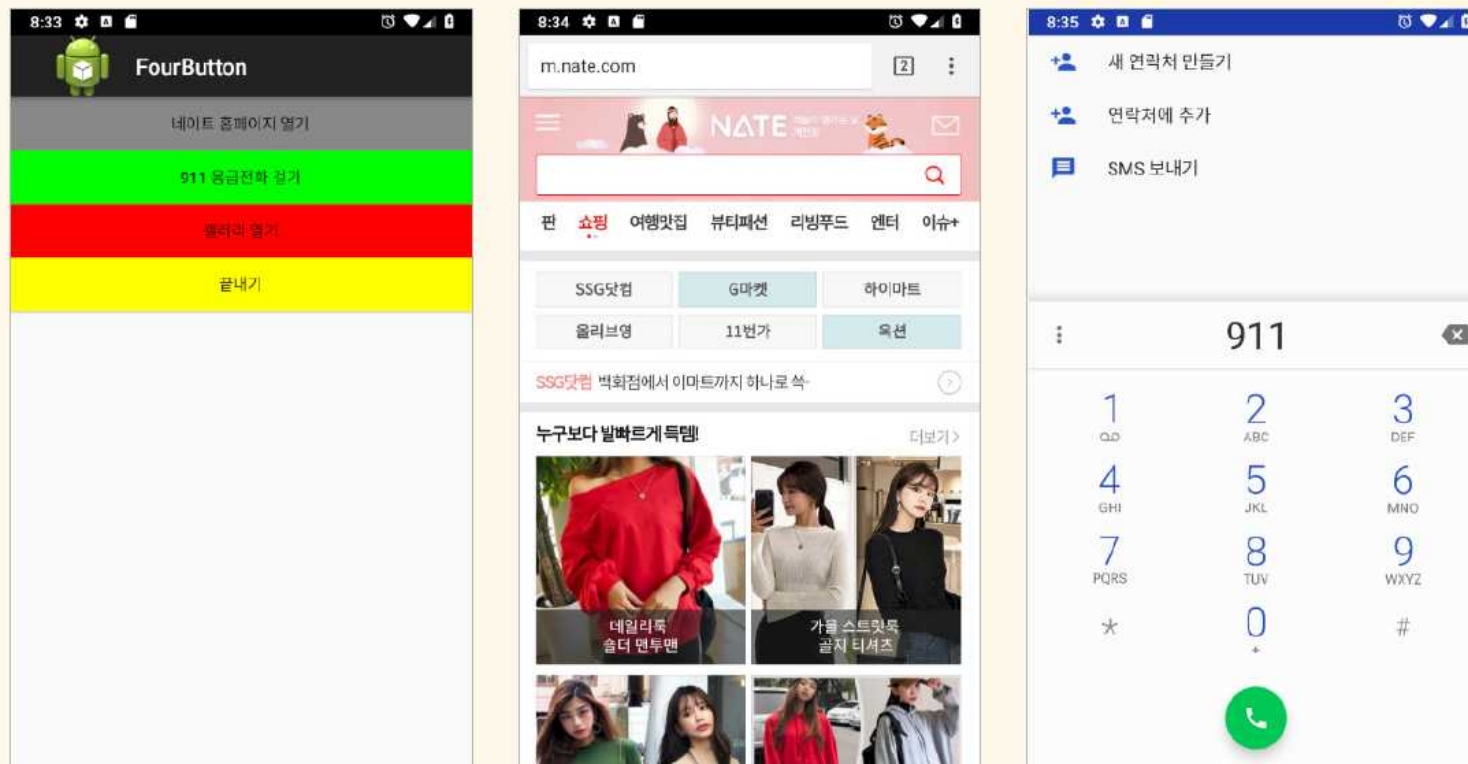


그림 2-64 FourButton 프로젝트 실행 결과

01 BaseApp 프로젝트의 구성

- 앞에서 만든 BaseApp 프로젝트의 구성

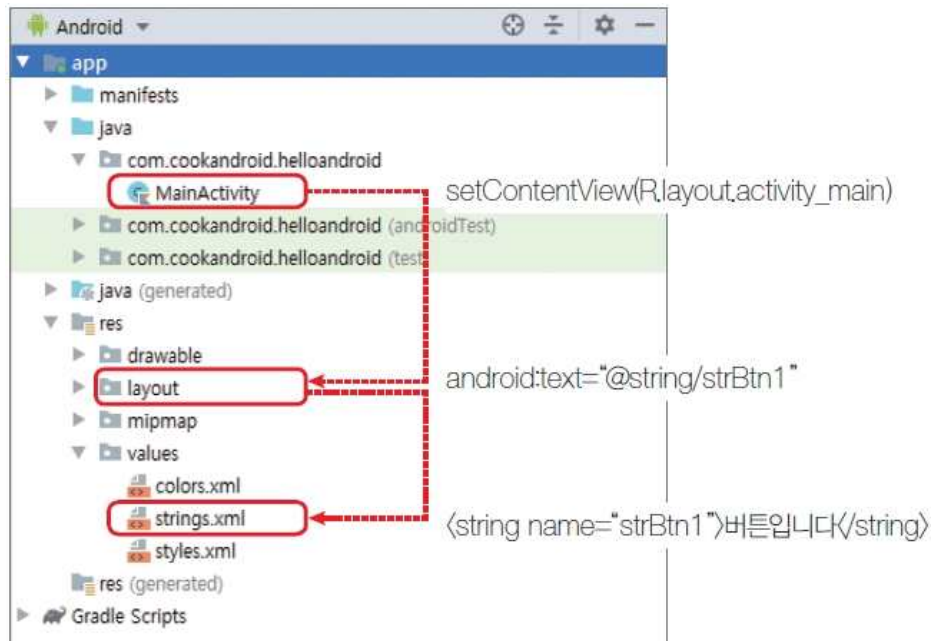


그림 2-65 안드로이드 프로젝트의 구성

02 프로젝트에서 사용되는 폴더와 파일의 용도

- 프로그래머가 직접 변경하는 폴더와 파일
 - **java 폴더**
 - 하위에 패키지명의 폴더가 있는데, 이는 안드로이드 프로젝트를 생성할 때 입력한 패키지 이름과 동일한 것임([그림 2-43] 참조)
 - 패키지 이름 아래에 MainActivity.kt로 메인 Kotlin 소스가 들어 있음
 - 실제로 프로그래머가 가장 많이 수정할 소스 파일이며, 주로 액티비티(화면, activity_main.xml)에서 어떤 일을 할지를 프로그래밍하게 됨
 - **java (generated) 폴더**
 - Android Studio 3.2부터 제공되는 폴더이며 시스템 내부적으로 사용되므로 특별히 신경 쓰지 않아도 됨

02 프로젝트에서 사용되는 폴더와 파일의 용도

- 프로그래머가 직접 변경하는 폴더와 파일
 - **res 폴더**
 - 앱 개발에 사용되는 이미지, 레이아웃, 문자열 등이 들어가는 폴더
 - drawable 폴더
 - 이미지 파일을 넣음
 - mipmap 폴더
 - 디자인 화면이나 앱이 설치된 후에 보이는 런처 아이콘을 넣음
 - xxxhdpi, xxhdpi, xhdpi : 초고해상도 런처 아이콘 파일을 넣음
 - hdpi : 고해상도 런처 아이콘 파일을 넣음
 - mdpi : 중해상도 런처 아이콘 파일을 넣음
 - layout 폴더
 - 액티비티(화면)를 구성하는 XML 파일을 넣음
 - 기본적으로 activity_main.xml이 초기화면으로 지정되어 있음. 추가로 화면이 필요하면 이곳에 XML 형태로 생성함.
 - values 폴더에는 문자열을 저장하는 strings.xml, 색상표를 저장하는 colors.xml, 스타일을 저장하는 styles.xml 등이 들어 있음.

02 프로젝트에서 사용되는 폴더와 파일의 용도

- 프로그래머가 직접 변경하는 폴더와 파일
 - **res 폴더**
 - values 폴더
 - strings.xml : 문자열을 저장
 - colors.xml : 색상표를 저장
 - styles.xml : 스타일을 저장
 - menu 폴더
 - 메뉴 XML 파일을 저장하는데, 필요시 생성해서 사용
 - anim 폴더
 - 애니메이션을 저장
 - XML 폴더
 - XML 파일을 저장

02 프로젝트에서 사용되는 폴더와 파일의 용도

- 프로그래머가 직접 변경하는 폴더와 파일
 - **res (generated) 폴더**
 - Android Studio 3.5부터 제공되는 폴더이며 내부적으로 사용되므로 신경 쓰지 않아도 됨
 - **manifests 폴더**
 - AndroidManifest.xml 파일이 들어 있음
 - 앱의 여러 가지 정보를 담고 있는 중요한 파일
 - '매니페스트 파일'이라고 읽으며, 필요에 따라 수정하거나 변경함

02 프로젝트에서 사용되는 폴더와 파일의 용도

- 프로그래머가 직접 변경하는 폴더와 파일
 - **Gradle Scripts** 폴더
 - Gradle 빌드 시스템과 관련된 파일이 들어 있음
 - 주요한 3개 파일
 - build.gradle (Module: app) : 빌드 스크립트 핵심 파일로 컴파일 버전, 실행되는 최하 버전, 컴파일 라이브러리 등을 등록함
 - local.properties : 컴파일되는 SDK의 경로가 들어 있음
 - gradle.properties : JVM 관련 메모리가 설정되어 있음

Thank You !