

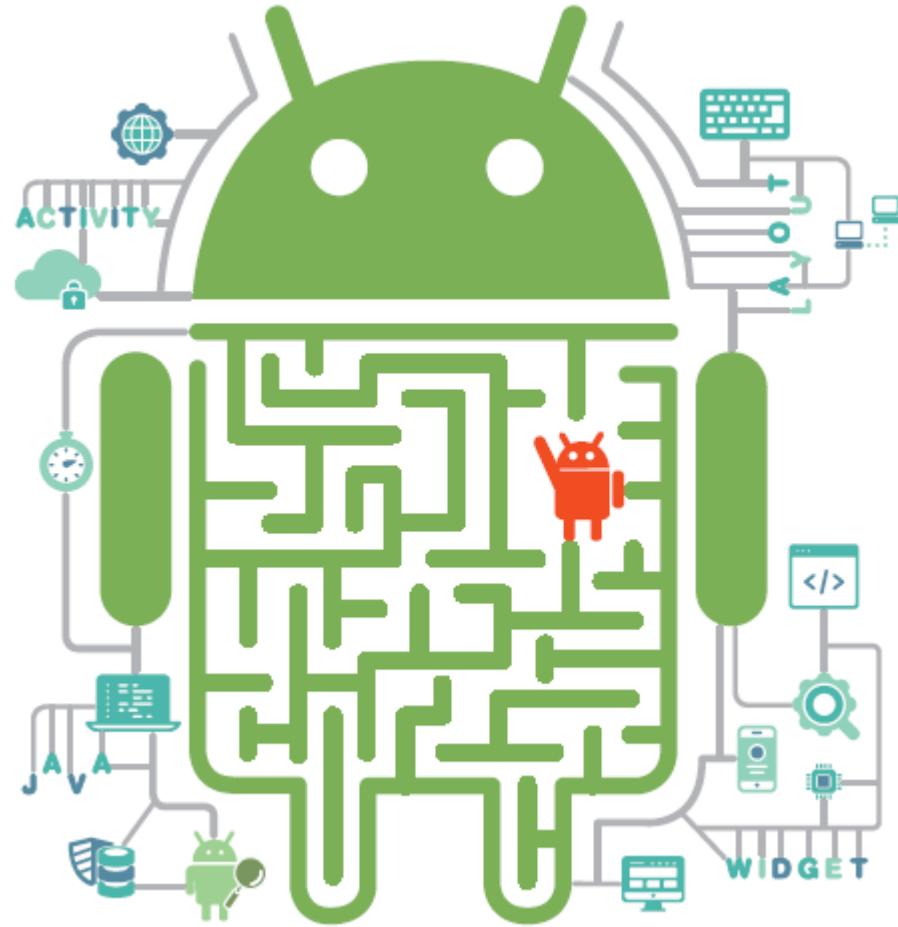
단계별로 배우는 안드로이드 프로그래밍

[강의교안 이용 안내]

- 본 강의교안의 저작권은 한빛아카데미(주)에 있습니다.
- 이 자료를 무단으로 전제하거나 배포할 경우 저작권법 136조에 의거하여 최고 5년 이하의 징역 또는 5천만원 이하의 벌금에 처할 수 있고 이를 병과(併科)할 수도 있습니다.

단계별로 배우는

안드로이드 프로그래밍



Chapter 01. 앱 개발 시작

목차

- 01 안드로이드란?
- 02 개발 환경 설정
- 03 프로젝트 구조 분석
- 04 안드로이드 스튜디오 기초

학습목표

- 안드로이드 시스템 구조를 이해한다.
- 안드로이드 개발 환경을 설정한다.
- 안드로이드 앱 프로젝트를 분석한다.
- 안드로이드 스튜디오 사용법을 파악한다.

01 안드로이드란? ▶ 버전 역사

■ 안드로이드(Android)

- 전 세계에서 가장 많이 사용되는 모바일 운영 체제
- 2003년에 설립된 소프트웨어 업체 안드로이드(Android Incorporation)에서 개발 시작
- 2005년 구글이 인수

■ 안드로이드의 버전

표 1-1 안드로이드 버전 역사

버전(발표 연도)	코드명	API 레벨	특징
1.0 (2008)	공식 코드명 없음	1	<ul style="list-style-type: none">• 마켓 탑재로 다양한 앱 설치와 업데이트 지원• 다양한 기본 앱(전화, 연락처, 일정, 메일 등) 탑재• 카메라, Wi-Fi, 블루투스 지원
1.1 (2009)	공식 코드명 없음	2	<ul style="list-style-type: none">• 버그 수정
1.5 (2009)	Cupcake ["컵케이크"]	3	<ul style="list-style-type: none">• 제3자 가상 키보드 지원• 홈 화면 위젯 Widget과 자동 회전 기능 지원
1.6 (2009)	Donut ["도넛"]	4	<ul style="list-style-type: none">• 다국어 TTS Text-To-Speech(텍스트 음성 변환) 지원• WVGA 해상도 지원

01 안드로이드란? ▶ 버전 역사

2.0 / 2.0.1 / 2.1 (2009~2010)	Eclair [“이클레어”]	5 / 6 / 7	<ul style="list-style-type: none">둘 이상의 동기화 계정(일정, 연락처, 메일) 지원블루투스 2.1 지원다양한 카메라 기능 추가웹 브라우저의 HTML5 일부 지원새로운 화면 크기와 비율 지원
2.2, 2.2.1, 2.2.2 (2010~2011)	Froyo [“프로요”]	8	<ul style="list-style-type: none">최적화로 인한 성능 향상USB 테더링^{Tethering}과 Wi-Fi 핫스팟^{Hotspot} 지원외장 메모리에 앱 설치 지원새로운 화면 크기와 비율 지원
2.3, 2.3.1, 2.3.2 (2010~2011)	Gingerbread [“진저브레드”]	9	<ul style="list-style-type: none">둘 이상의 카메라(예: 전/후면) 지원최적화로 인한 성능과 전원 관리 향상NFC^{Near Field Communication} 지원VoIP^{Voice over IP}(인터넷 음성 통신) 지원새로운 화면 크기와 비율 지원
2.3.3~2.3.7 (2011)	Gingerbread [“진저브레드”]	10	<ul style="list-style-type: none">NFC와 블루투스 기능 향상배터리 효율 향상

01 안드로이드란? ▶ 버전 역사

3.0 (2011)	Honeycomb [“허니콤”]	11	<ul style="list-style-type: none">• 새로운 UI와 더불어 최초의 태블릿 지원• 새로운 멀티태스킹 버튼으로 편리해진 앱 전환• 하드웨어 가속 기능과 멀티코어 CPU 지원• 사용자 데이터 전체 암호화 기능 지원
3.1 (2011)	Honeycomb [“허니콤”]	12	<ul style="list-style-type: none">• USB OTG^{On-The-Go} 지원• 홍 화면 위젯의 크기 변경 가능• 외장 키보드, 마우스, 조이스틱, 게임패드 지원
3.2, 3.2.1~3.2.6 (2011~2012)	Honeycomb [“허니콤”]	13	<ul style="list-style-type: none">• 다양한 화면 크기 지원 강화• 태블릿 미지원 앱을 위한 호환 모드 추가
4.0, 4.0.1, 4.0.2 / 4.0.3, 4.0.4 (2011~2012)	Ice Cream Sandwich [“아이스크림 샌드위치”]	14 / 15	<ul style="list-style-type: none">• 새로운 UI와 <u>소프트</u> 버튼 지원• 얼굴 인식을 이용한 잠금 해제 지원• 안드로이드 빔^{Android Beam} 데이터 교환• WebP 이미지 형식 지원, Wi-Fi Direct 지원• 제3자 VPN^{Virtual Private Network} 기능
4.1, 4.1.1, 4.1.2 (2012)	Jelly Bean [“젤리빈”]	16	<ul style="list-style-type: none">• 부드러운 사용자 인터페이스를 위한 최적화• 확장 가능한 알림^{Expandable notification}• 앱별로 알림 메시지 허용 여부 제어• USB 오디오 지원으로 외장 사운드 DAC 가능
4.2, 4.2.1, 4.2.2 (2012~2013)	Jelly Bean [“젤리빈”]	17	<ul style="list-style-type: none">• 잠금 화면 위젯과 화면 보호기 지원• 다중 사용자 지원(태블릿만)

01 안드로이드란? ▶ 버전 역사

4.3, 4.3.1 (2013)	Jelly Bean [“젤리빈”]	18	<ul style="list-style-type: none">저전력 블루투스 Bluetooth Low Energy 지원OpenGL ES 3.0 지원으로 게임 그래픽스 향상
4.4, 4.4.1~4.4.3 (2013~2014)	KitKat [“킷캣”]	19	<ul style="list-style-type: none">소프트 버튼과 상태 표시줄 제어(숨기기·튜명도) 지원앱의 외장 메모리 접근 제약 강화메모리 RAM 가 적은 기기를 위한 최적화무선 인쇄 기능, 화면 녹화 기능
4.4W, 4.4W.1~W.2 (2014)	KitKat [“킷캣”]	20	<ul style="list-style-type: none">최초의 안드로이드 웨어 Android Wear 플랫폼 : 스마트워치 지원
5.0, 5.0.1, 5.0.2 (2014)	Lollipop [“롤리팝”]	21	<ul style="list-style-type: none">달빛 가상 머신을 아트 ART로 변경64비트 CPU와 OpenGL ES 3.1 지원머티리얼 디자인 Material design 도입다중 사용자 지원(태블릿과 휴대폰 모두)앱의 외장 메모리 접근 제약 완화
5.1, 5.1.1 (2015)	Lollipop [“롤리팝”]	22	<ul style="list-style-type: none">둘 이상의 심 SIM 카드 공식 지원
6.0, 6.0.1 (2015)	Marshmallow [“마시멜로”]	23	<ul style="list-style-type: none">지문 인식 공식 지원, 배터리 성능 향상실행 중 퍼미션 Permission 기능으로 보안 향상외장 메모리를 내장 메모리처럼 사용 가능USB 타입-C 지원, 악기 제어를 위한 MIDI 지원
7.0 / 7.1~7.1.2 (2016~2017)	Nougat [“누가”]	24 / 25	<ul style="list-style-type: none">다중 창 모드 지원으로 두 개의 앱 동시 사용 가능새로운 전원 절약 모드 도입새로운 3D API인 Vulkan 지원앱 설치 시간과 코드 크기 단축
8.0 (2017)	Oreo [“오레오”]	26	<ul style="list-style-type: none">PIP Picture-in-Picture 모드 지원으로 유튜브 등의 앱을 작은 창으로 실행 가능백그라운드 실행 제한으로 배터리 성능 향상다양한 최적화로 인한 OS와 앱 성능 향상

01 안드로이드란? ▶ 시스템 구조

■ 안드로이드

- 리눅스Linux를 기반으로 여러 소프트웨어 계층으로 구성
- 크게 다섯 개의 계층으로 나눌 수 있음
- 최상위 계층에 자바Java로 개발된 응용 프로그램 위치

〈애플리케이션〉

내장 앱 : 전화, 연락처, 메시지, 일정, 카메라, 웹 브라우저 등 제조사가 미리 설치한 앱
사용자 앱 : 구글 플레이 스토어와 같은 앱 마켓에서 사용자가 직접 선택하여 설치한 앱

〈애플리케이션 프레임워크〉

액티비티 관리자, 패키지 관리자, 원도우 관리자, 위치 관리자 등

〈라이브러리〉

Bionic : C 라이브러리
WebKit : 웹 엔진
SQLite : 데이터베이스 엔진
SSL : 보안 통신
FreeType : 폰트 처리 엔진
미디어 프레임워크 : 다양한 멀티미디어 형식 지원
SGL : 2D 그래픽스
OpenGL ES : 3D 그래픽스
...
...

〈안드로이드 런타임〉

핵심 자바 라이브러리
Dalvik 또는 ART

〈리눅스 커널〉

하드웨어 장치 드라이버 : 디스플레이, USB, 카메라, 블루투스, Wi-Fi, 오디오 등
공유 메모리 Shared Memory : 프로세스 간 데이터 공유 기능 제공
바인더 Binder : 프로세스 간 통신 기능 제공
...

그림 1-1 안드로이드 시스템 구조

01 안드로이드란? ▶ 시스템 구조

■ 리눅스 커널 (Linux Kernel)

- 1990년대에 개발된 무료 운영 체제인 리눅스의 핵심 부분

■ 라이브러리 (Library)

- 소프트웨어 개발의 생산성을 높이고자 미리 작성해놓은 코드의 집합

■ 안드로이드 런타임 (Android Runtime)

- 자바 언어로 개발된 코드를 실행하기 위한 가상 머신 Virtual Machine

■ 애플리케이션 프레임워크 (Application Framework)

- 안드로이드 시스템의 핵심적인 서비스로 구성
- 대부분 자바 언어로 작성

■ 애플리케이션 (Application)

- 응용 프로그램이 존재하는 계층
- 안드로이드 시스템에 내장된 앱과 사용자가 직접 설치하는 앱으로 나뉨

01 안드로이드란? ▶ 앱 개발 절차

■ 앱 개발 방식

- 안드로이드 SDK(Software Development Kit)는 필수이되 나머지는 선택 사항

- ① 안드로이드 SDK+Eclipse +ADT^{Android Development Tools} 플러그인
- ② 안드로이드 SDK+IntelliJ IDEA+안드로이드 플러그인
- ③ 안드로이드 SDK+안드로이드 스튜디오^{Android Studio}+안드로이드 플러그인
- ④ 안드로이드 SDK+비주얼 스튜디오^{Visual Studio}+Xamarin 플러그인

■ 앱 개발 절차



그림 1-2 앱 개발 절차

01 안드로이드란? ▶ 앱 개발 절차

■ 안드로이드 스튜디오로 앱 개발 시 구성 요소 간 상호 작용

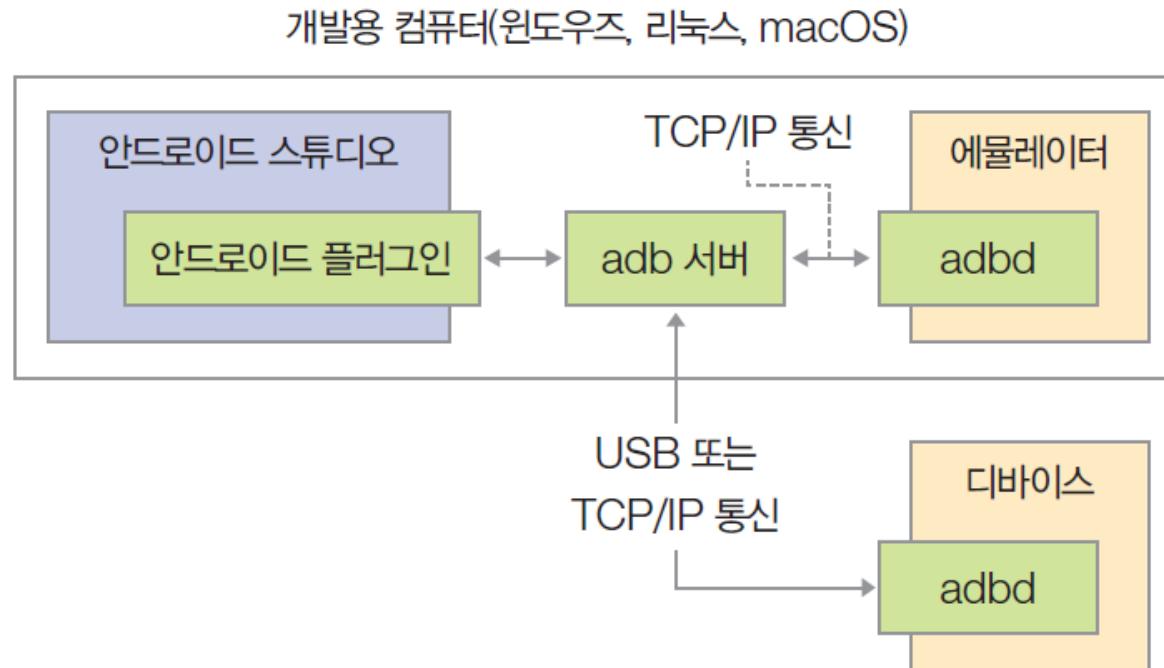


그림 1-3 앱 개발 구성 요소

01 안드로이드란? ▶ 앱 개발 절차

■ **에뮬레이터(Emulator) 혹은 디바이스(Device)**

- 안드로이드 스튜디오에서 개발한 앱 테스트

■ **adb(Android Debug Bridge) 프로그램**

- 안드로이드 스튜디오를 실행하면 자동으로 실행되어 [에뮬레이터/디바이스]와의 연결을 관리

■ **abrd(Android Debug Bridge Daemon) 프로그램**

- 안드로이드 디바이스 내부에서 adb 서버와의 통신을 담당

■ **Run 메뉴**

- [안드로이드 플러그인] → [adb 서버] → [abrd] → [에뮬레이터/디바이스]로 앱이 전송되어 설치 및 실행

■ **디버거(Debugger)**

- 프로그램의 버그를 찾기 위한 소프트웨어
- 앱을 직접 사용하거나 안드로이드 스튜디오에 내장

02 개발 환경 설정

■ 안드로이드 스튜디오를 이용하여 앱 개발하기 위한 환경 설정 절차



그림 1-4 개발 환경 설정 절차

02 개발 환경 설정 ▶ 안드로이드 스튜디오 설치



실습 1-1 ▶ 안드로이드 스튜디오 설치

■ 안드로이드 스튜디오 아카이브 사이트 방문

- [https://developer.android.com/studio/archive.html'](https://developer.android.com/studio/archive.html)

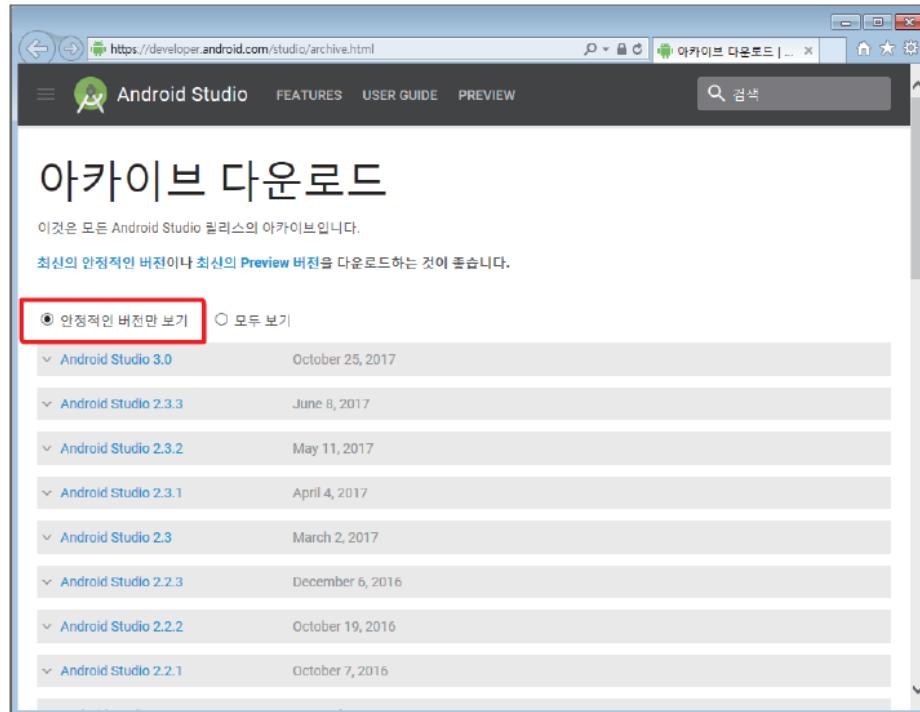


그림 1-5 안드로이드 스튜디오 다운로드 (1)

02 개발 환경 설정 ▶ 안드로이드 스튜디오 설치



실습 1-1 ▶ 안드로이드 스튜디오 설치

■ 안드로이드 스튜디오 다운로드

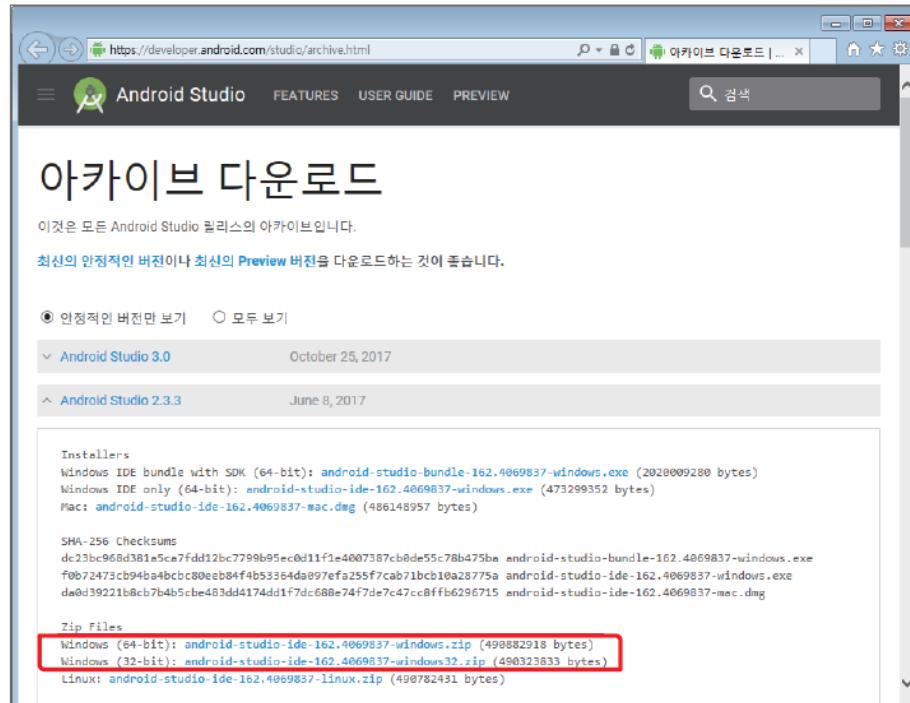


그림 1-6 안드로이드 스튜디오 다운로드 (2)

02 개발 환경 설정 ▶ 안드로이드 스튜디오 설치



실습 1-1 ▶ 안드로이드 스튜디오 설치

■ 안드로이드 스튜디오에 압축 풀기

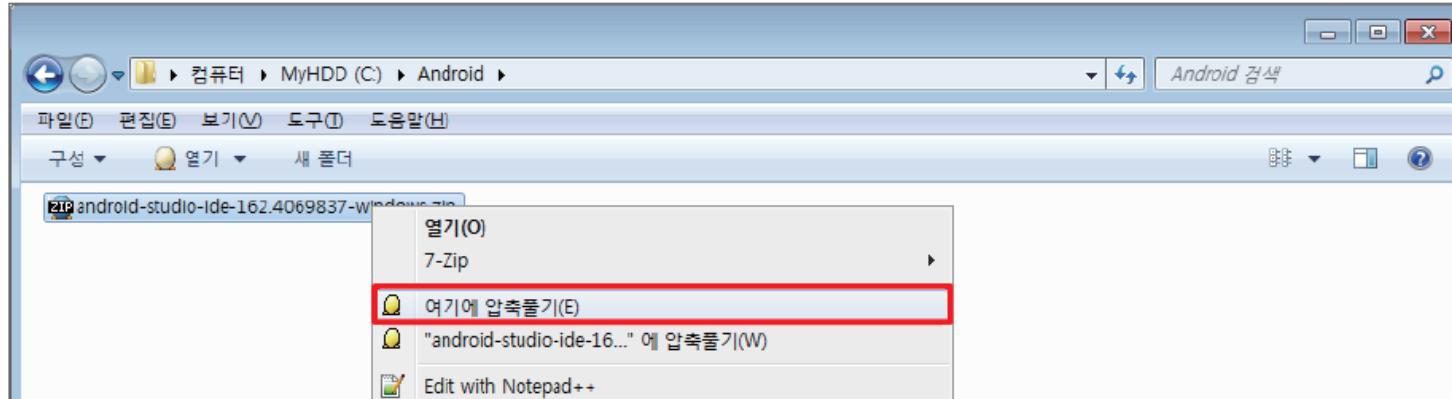


그림 1-7 안드로이드 스튜디오 압축 풀기

02 개발 환경 설정 ▶ 안드로이드 스튜디오 설치



실습 1-1 ▶ 안드로이드 스튜디오 설치

■ 안드로이드 스튜디오 설치 완료



그림 1-8 안드로이드 스튜디오 설치 결과

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 안드로이드 SDK 설치

■ 32비트 윈도우

- C:\Android\Android-studio\bin\studio.exe 실행

■ 64비트 윈도우

- C:\Android\Android-studio\bin\studio64.exe를 실행

■ 설정값 여부 대화상자에 [OK] 클릭

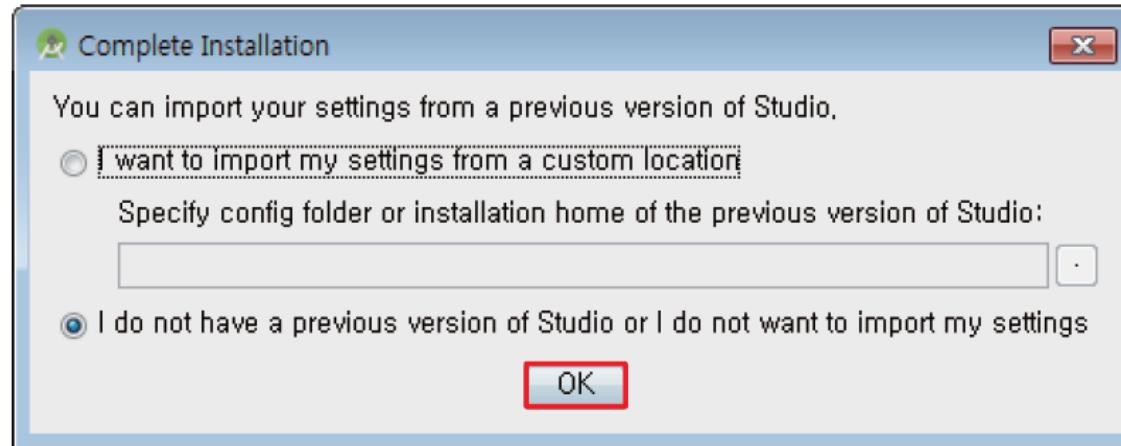


그림 1-9 최초 실행 시 설정값 가져오기 여부 선택

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 ▶ 안드로이드 SDK 설치

- [Next]를 클릭

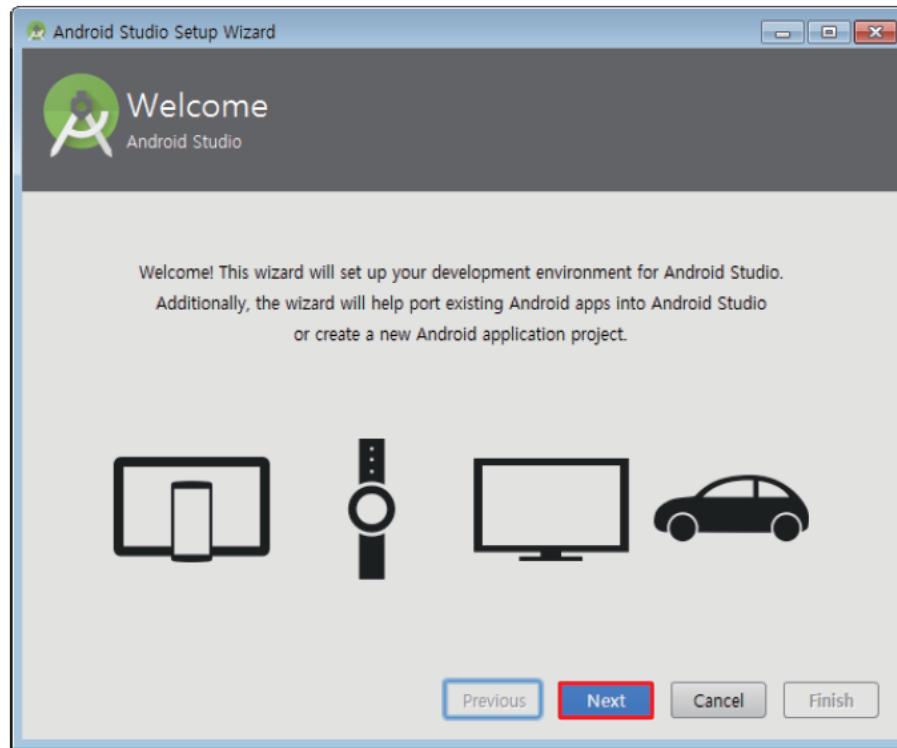


그림 1-10 설정 마법사 시작

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 ▶ 안드로이드 SDK 설치

■ 설치 타입을 'Custom'으로 변경

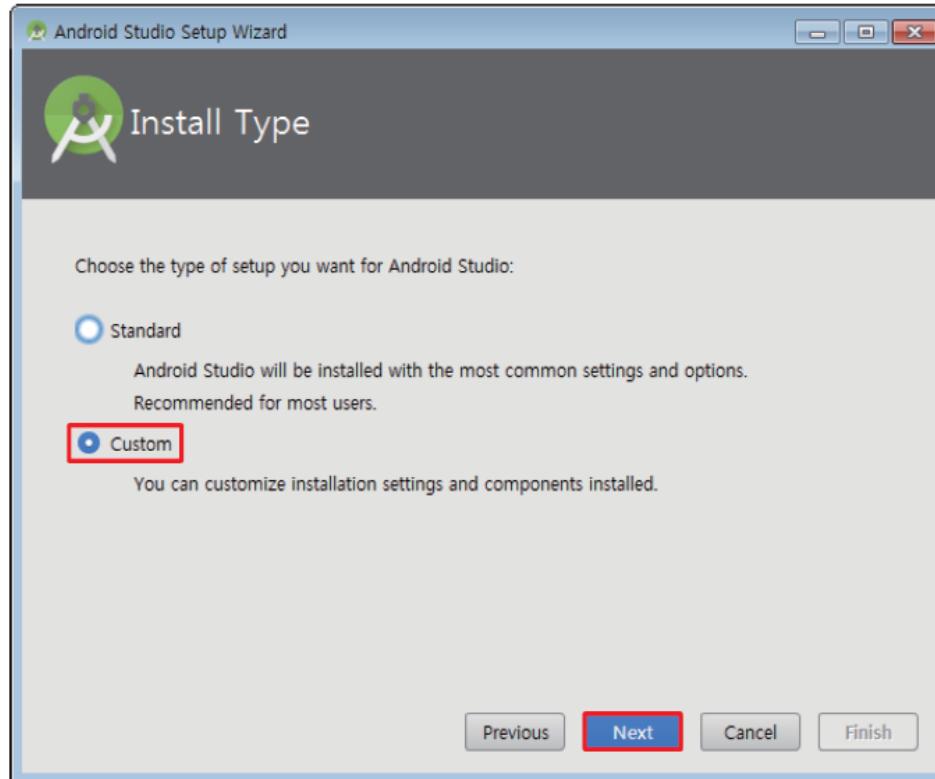


그림 1-11 설치 타입 변경

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 안드로이드 SDK 설치

■ UI(User Interface) 테마 선택

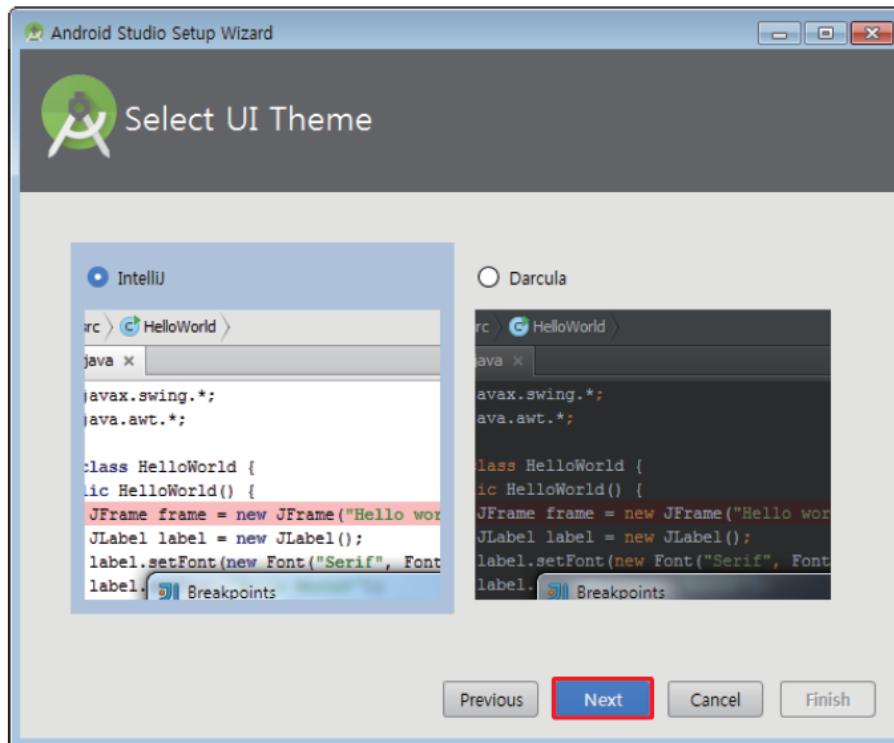


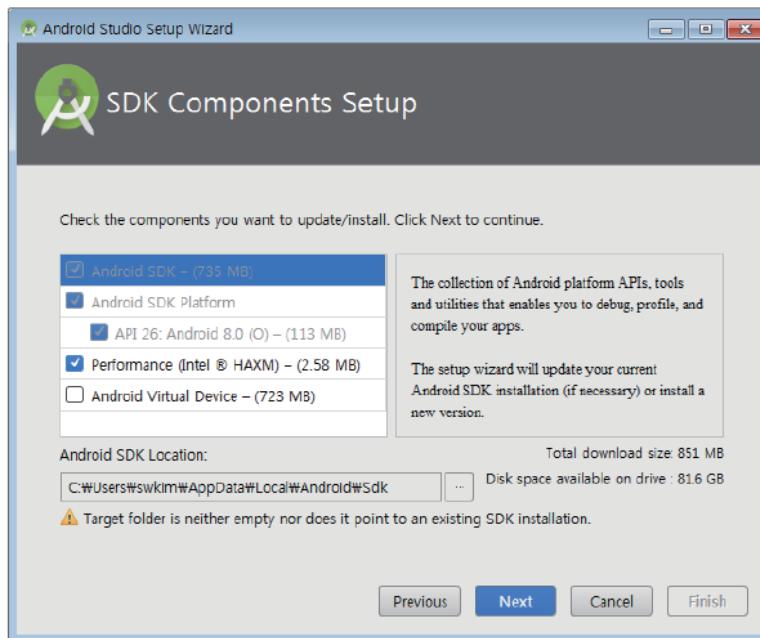
그림 1-12 UI 테마 선택

02 개발 환경 설정 ▶ 안드로이드 SDK 설치

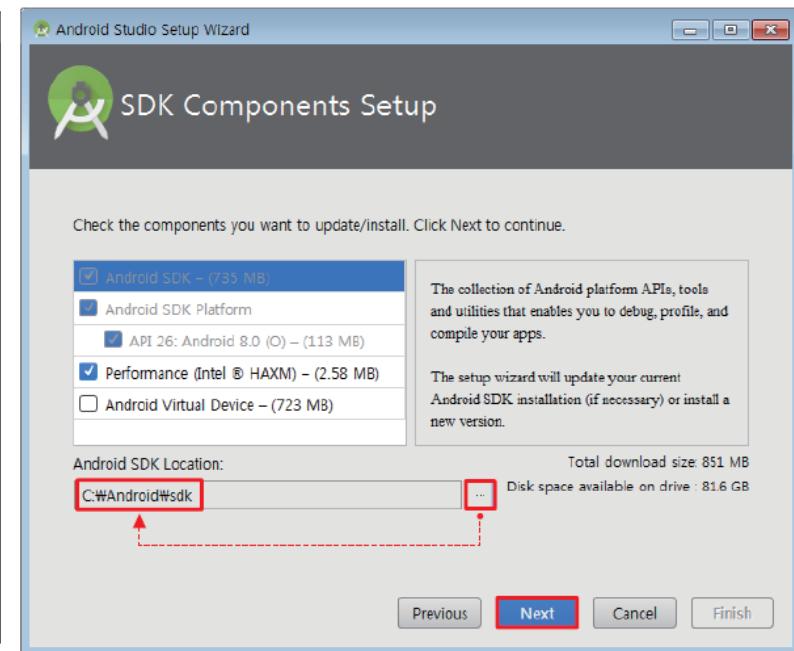


실습 1-2 안드로이드 SDK 설치

C:\Android\ sdk 폴더 만들고 위치 지정



(a) 초기 상태



(b) SDK 위치 지정

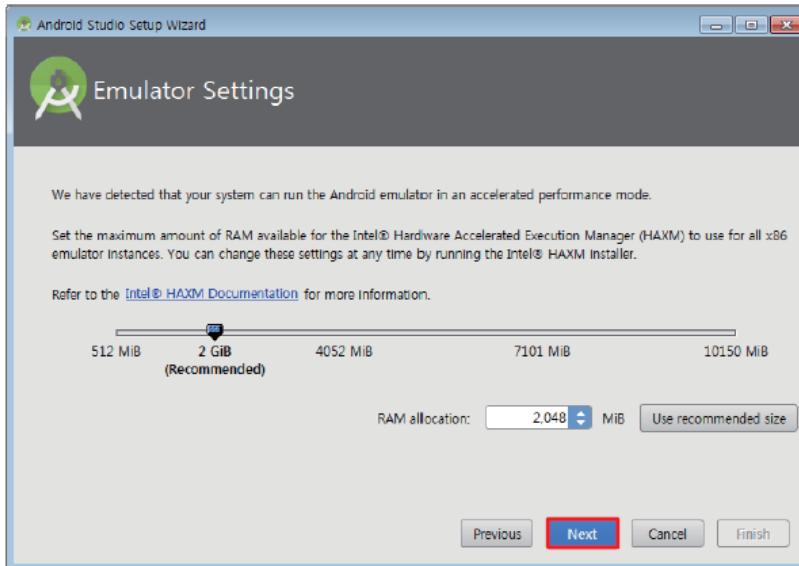
그림 1-13 안드로이드 SDK 설치

02 개발 환경 설정 ▶ 안드로이드 SDK 설치

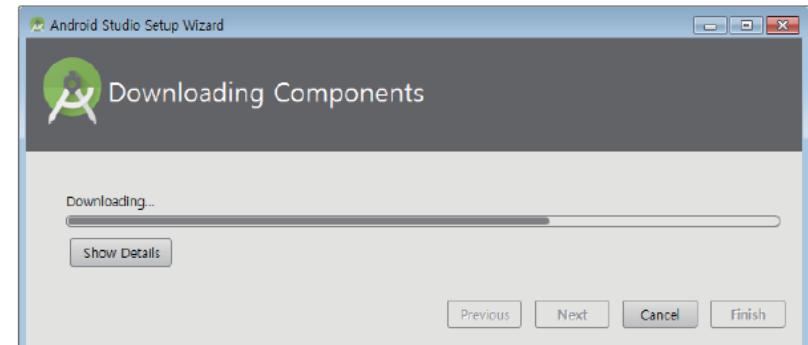


실습 1-2 안드로이드 SDK 설치

■ 기본값 선택하여 진행



(a) SDK 구성 요소 설치 시작



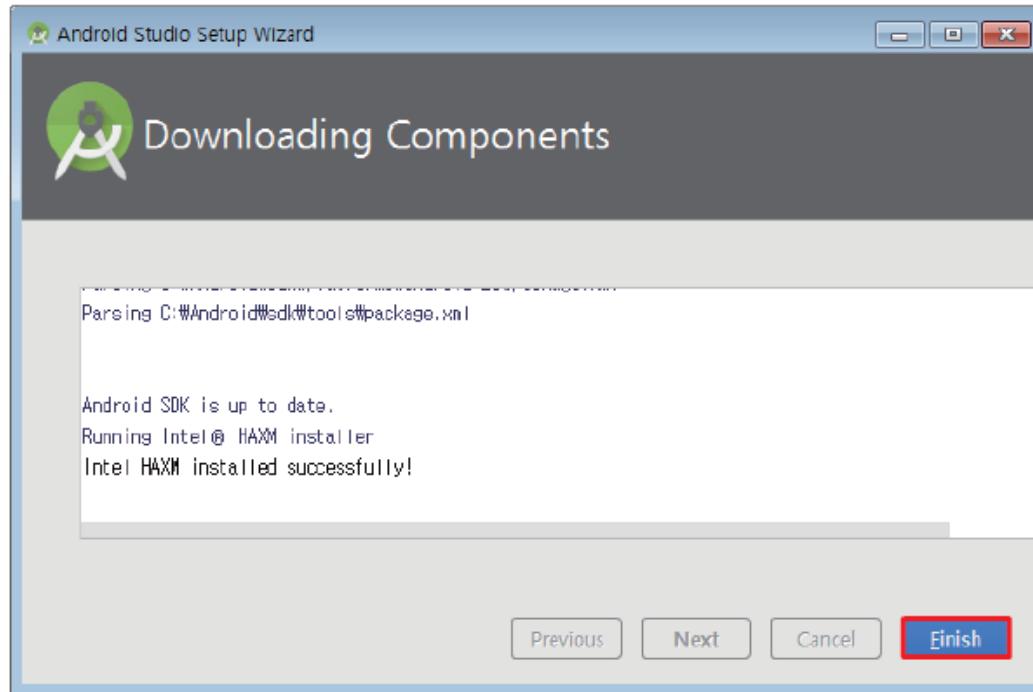
(b) SDK 구성 요소 설치 도중

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 ▶ 안드로이드 SDK 설치

■ [Finish] 클릭



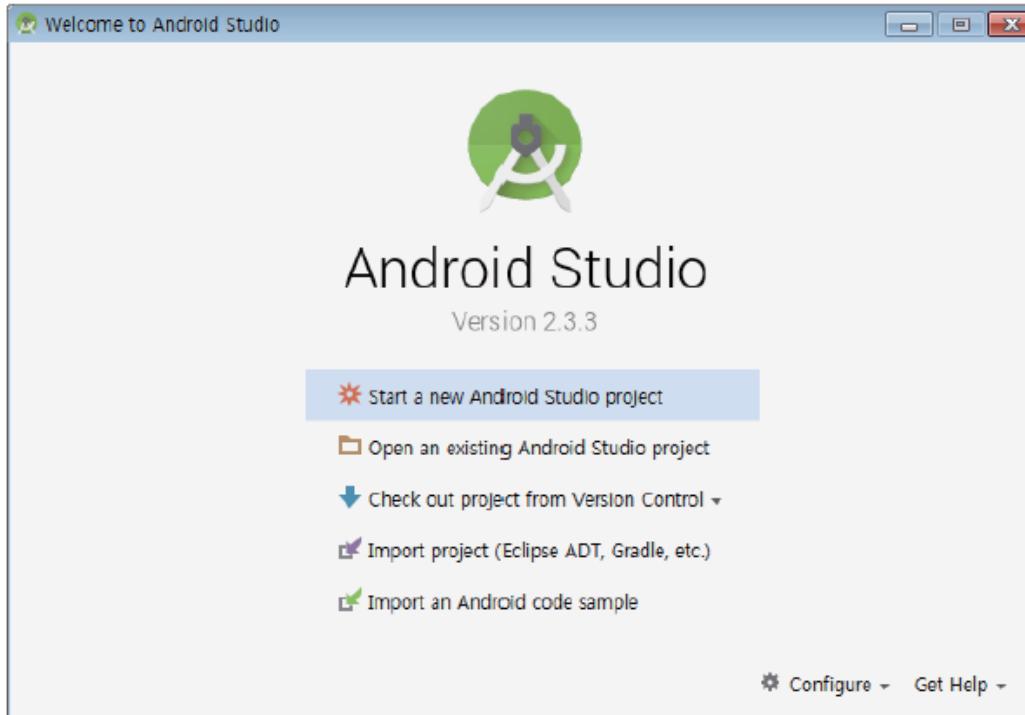
(c) SDK 구성 요소 설치 완료

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-2 ▶ 안드로이드 SDK 설치

■ 안드로이드 스튜디오 실행



(d) 안드로이드 스튜디오 시작 화면

그림 1-14 안드로이드 SDK 설치와 안드로이드 스튜디오 실행

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-3

환경 변수 설정

- 윈도우즈의 시작 메뉴 ▶ [제어판]-[시스템]-[고급 시스템 설정]-[고급]-[환경 변수...]
- 현재 로그인 사용자의 환경 변수를 수정

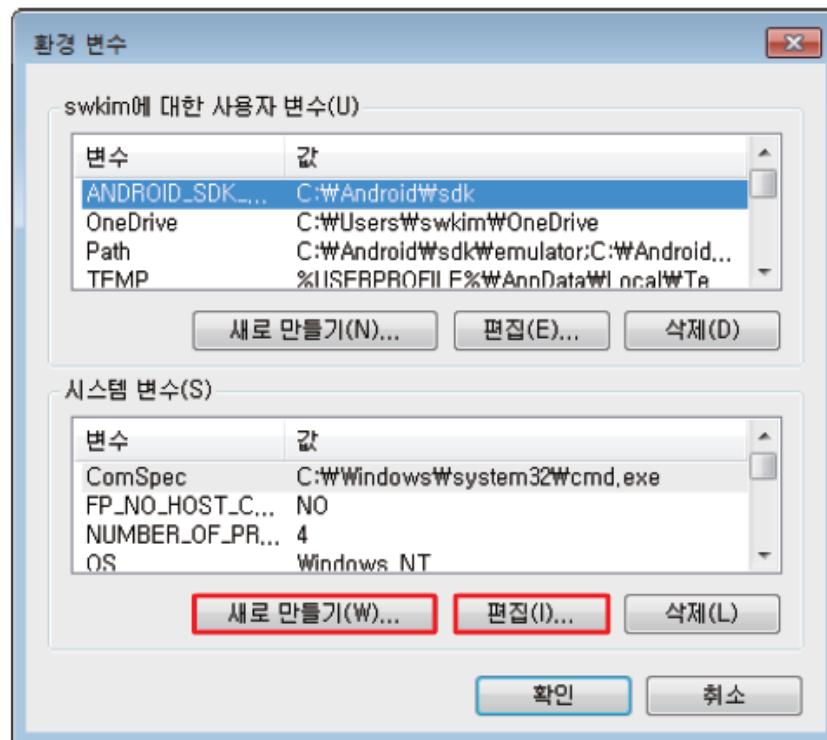


그림 1-15 환경 변수 설정 화면

02 개발 환경 설정 ▶ 안드로이드 SDK 설치



실습 1-3

환경 변수 설정

- [새로 만들기...]를 클릭하여 환경 변수를 추가
- 혹은 [편집...]을 클릭하여 기존값 변경
- 권장 환경 변수 이름과 값

표 1-2 안드로이드 SDK를 위한 환경 변수 설정

환경 변수 이름	값
ANDROID_SDK_ROOT	C:\Android\ sdk
PATH	C:\Android\ sdk\ emulator; C:\Android\ sdk\ platform-tools; C:\Android\ sdk\ tools;기존값

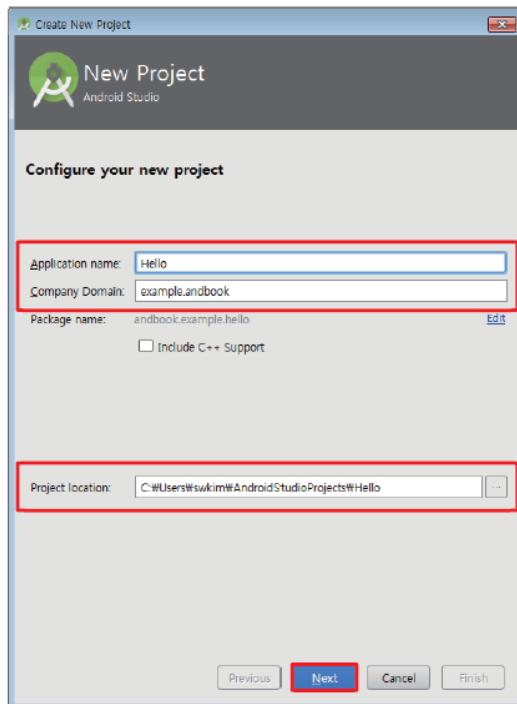
02 개발 환경 설정 ▶ 프로젝트 생성



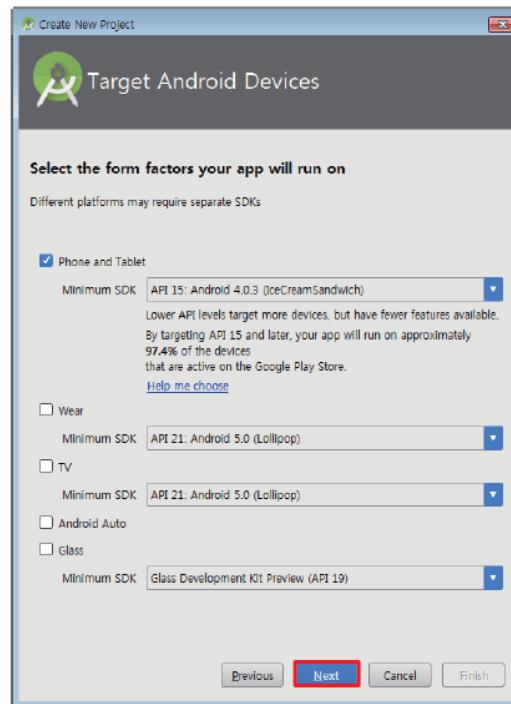
실습 1-4

Hello 프로젝트 생성

■ 프로젝트 단계적으로 만들기



(a) 프로젝트 생성 1단계



(b) 프로젝트 생성 2단계

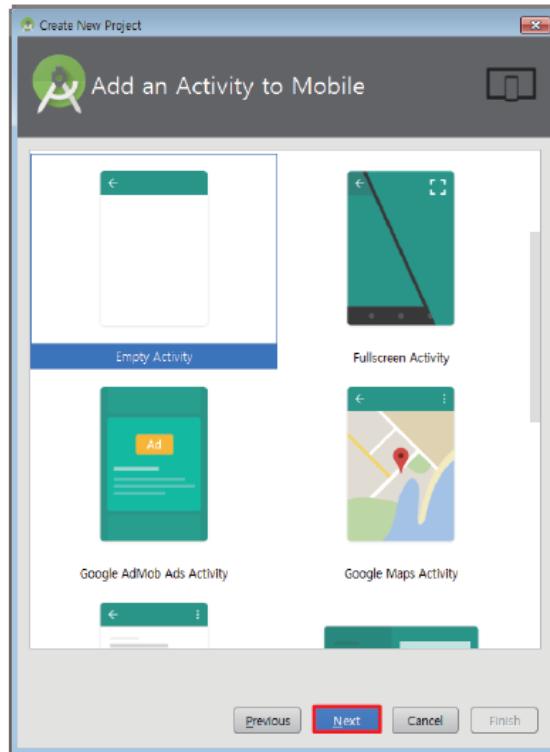
02 개발 환경 설정 ▶ 프로젝트 생성



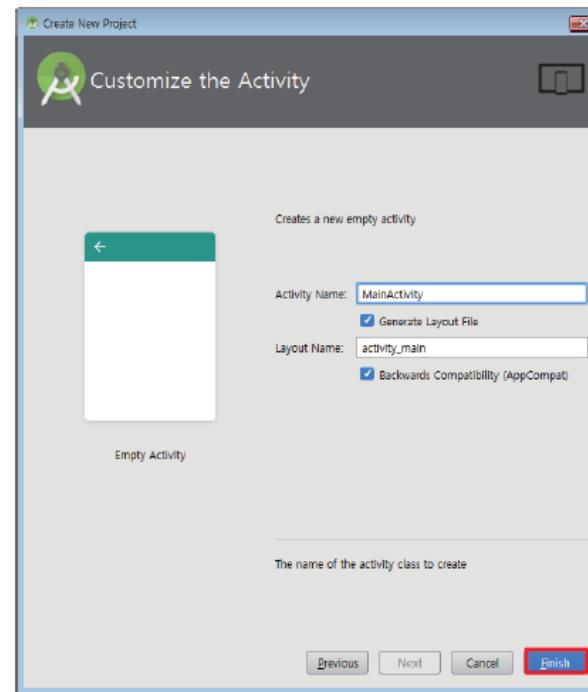
실습 1-4

Hello 프로젝트 생성

■ 프로젝트 단계적으로 만들기



(c) 프로젝트 생성 3단계



(d) 프로젝트 생성 4단계 (끝)

그림 1-16 프로젝트 생성 단계

02 개발 환경 설정 ▶ 프로젝트 생성



실습 1-4

Hello 프로젝트 생성

표 1-3 프로젝트 생성 단계별 세부 항목

단계	세부 항목 설명
1	<ul style="list-style-type: none">Application name: 응용 프로그램 이름이다. 구글 플레이 스토어에서도 이 이름이 사용자에게 보이게 된다.Company domain: 자바 패키지 이름을 만들 때 사용할 인터넷 도메인 이름이다. 존재하는 도메인 이름이 아니어도 상관없으며 여러 단어를 조합한 고유한 이름이면 된다.Package name: 도메인 이름과 응용 프로그램 이름을 조합하여 자동으로 만들어진 패키지 이름이다. 안드로이드에서 패키지 이름은 응용 프로그램의 고유한 이름으로 사용되며, 관례로 모두 소문자를 사용한다. 우측의 'Edit'를 클릭하면 패키지 이름을 변경할 수 있다.Project location: 프로젝트 구성 파일들이 위치할 폴더를 지정한다. 지정한 폴더가 존재하지 않으면 자동 생성된다.
2	앱이 지원하는 안드로이드 디바이스의 종류와 최소 안드로이드 버전을 지정한다. 이 책에서는 폰과 태블릿만을 고려하지만 Wear(스마트워치), TV(스마트TV), Android Auto(자동차), Glass(스마트안경) 지원을 추가할 수 있다. 참고 이 책의 모든 예제는 안드로이드 4.0.3 이상을 지원하도록 프로젝트를 생성한다.
3	자동 생성되는 코드가 지원할 앱의 형태를 지정한다. 참고 이 책의 예제 대부분은 Empty Activity를 선택하여 생성한다.
4	프로젝트 구성 파일들의 이름을 지정한다. 'Activity Name'은 초기 화면의 동작을 정의하는 자바 클래스 이름이고 'Layout Name'은 초기 화면을 정의하는 XML 파일 이름이다. 참고 이 책의 예제 대부분은 기본값을 사용한다.

02 개발 환경 설정 ▶ 프로젝트 생성



실습 1-4

Hello 프로젝트 생성

■ 프로젝트 생성 직후 안드로이드 스튜디오 화면

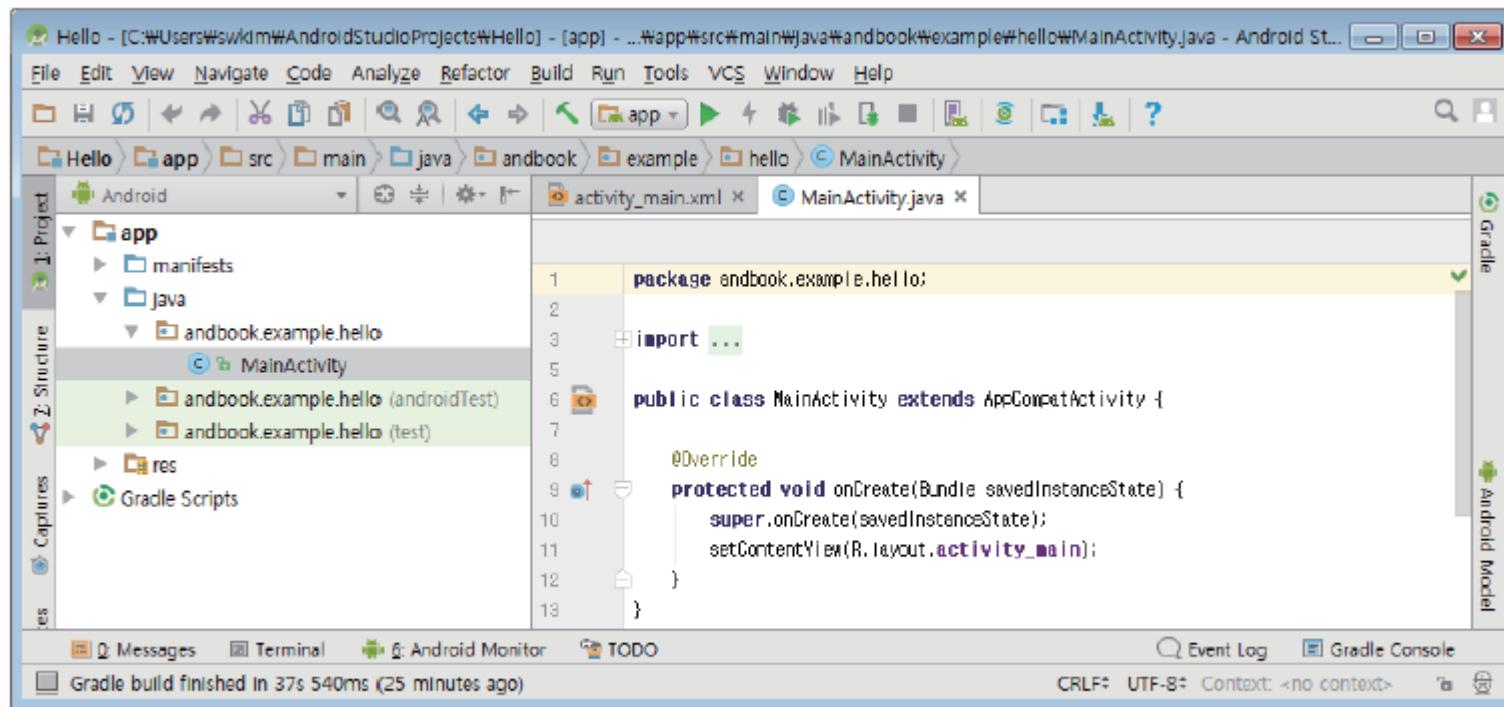


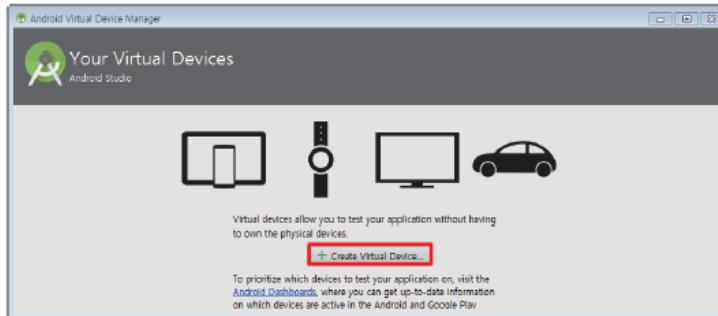
그림 1-17 프로젝트 생성 직후

02 개발 환경 설정 ▶ 앱 설치와 제거

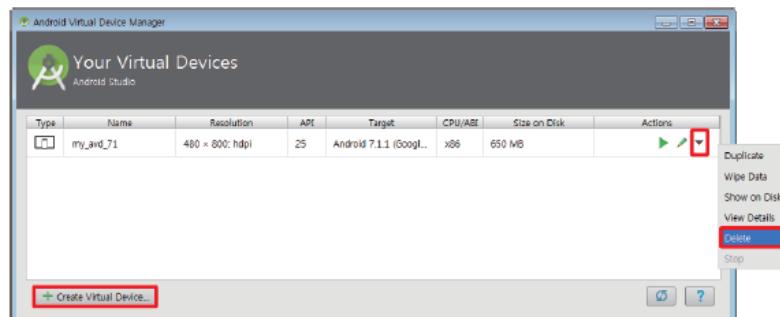


실습 1-5 AVD 생성, 실행, 설정

- [Create Virtual Device...]를 클릭하여 다음 화면으로 진행



(a) 생성된 AVD가 없는 경우



(b) 생성된 AVD가 있는 경우

그림 1-18 AVD 관리자 실행

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

Nexus S 선택

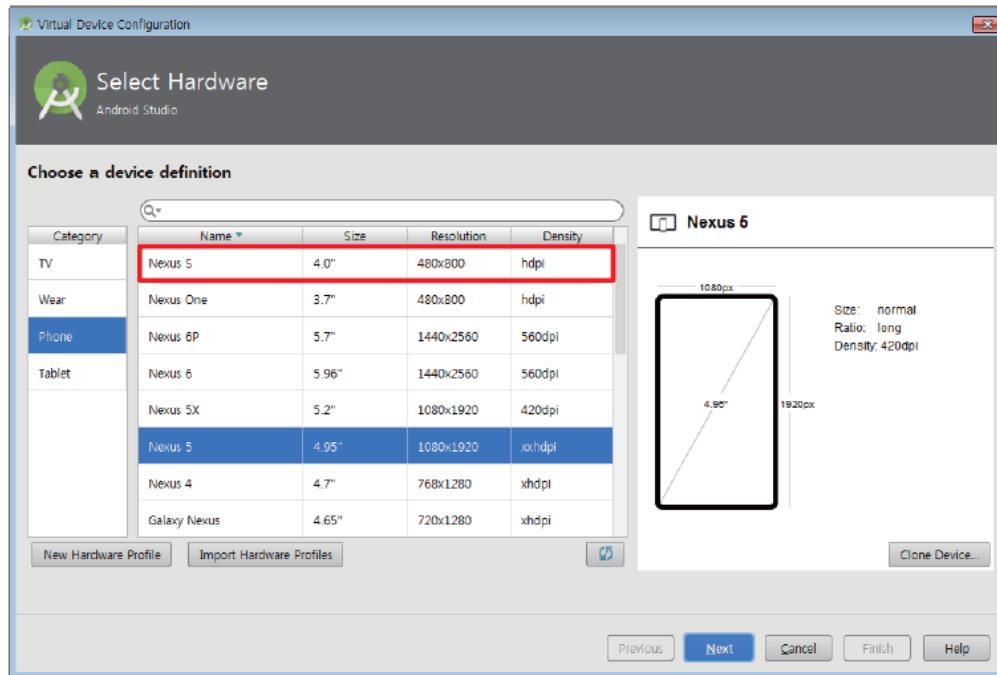


그림 1-19 하드웨어 정의 선택

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

- 설치가 끝나면 [Next]를 클릭하여 다음 화면으로 이동

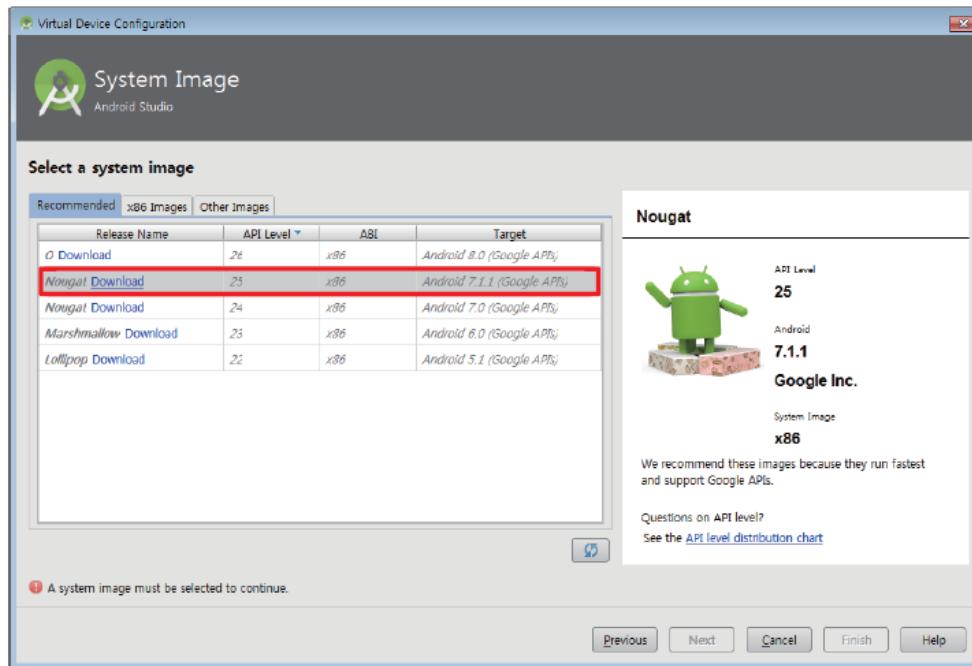


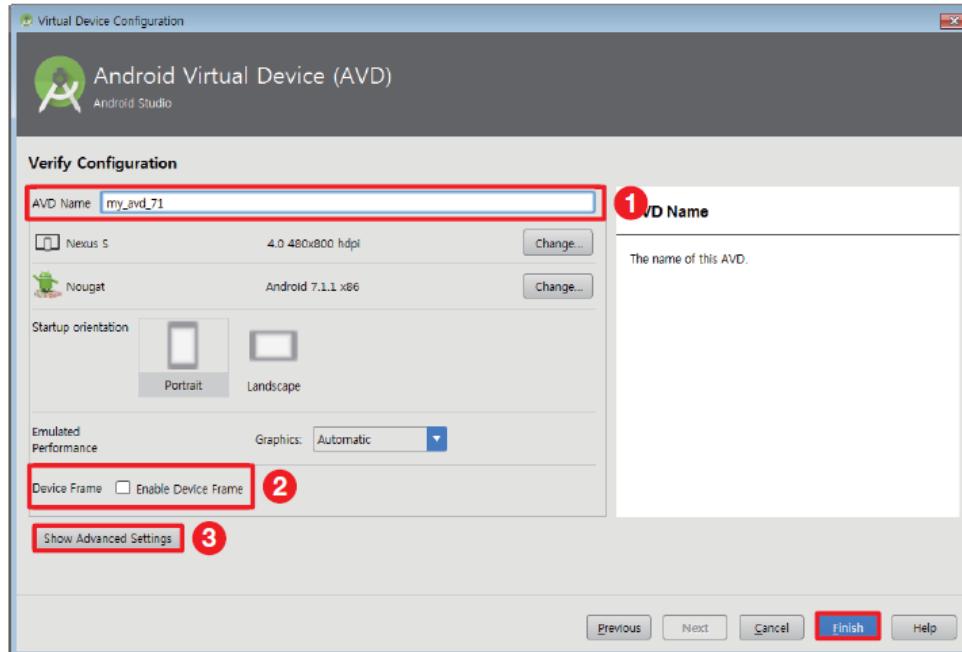
그림 1-20 시스템 이미지 다운로드

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

■ AVD 세부 설정



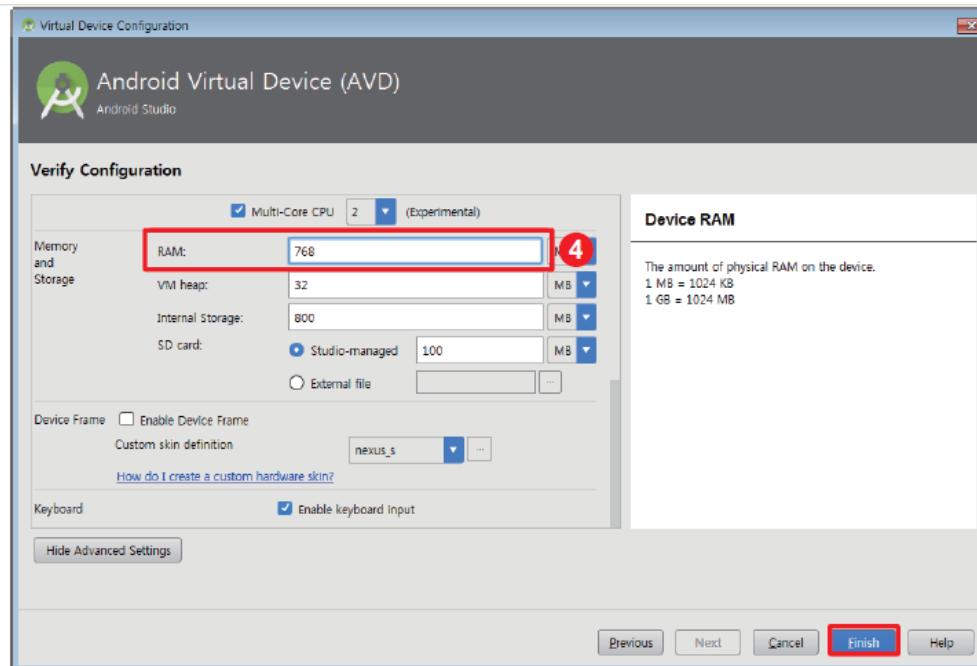
(a) 기본 설정 변경

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

■ AVD 세부 설정 화면



(b) 고급 설정 변경

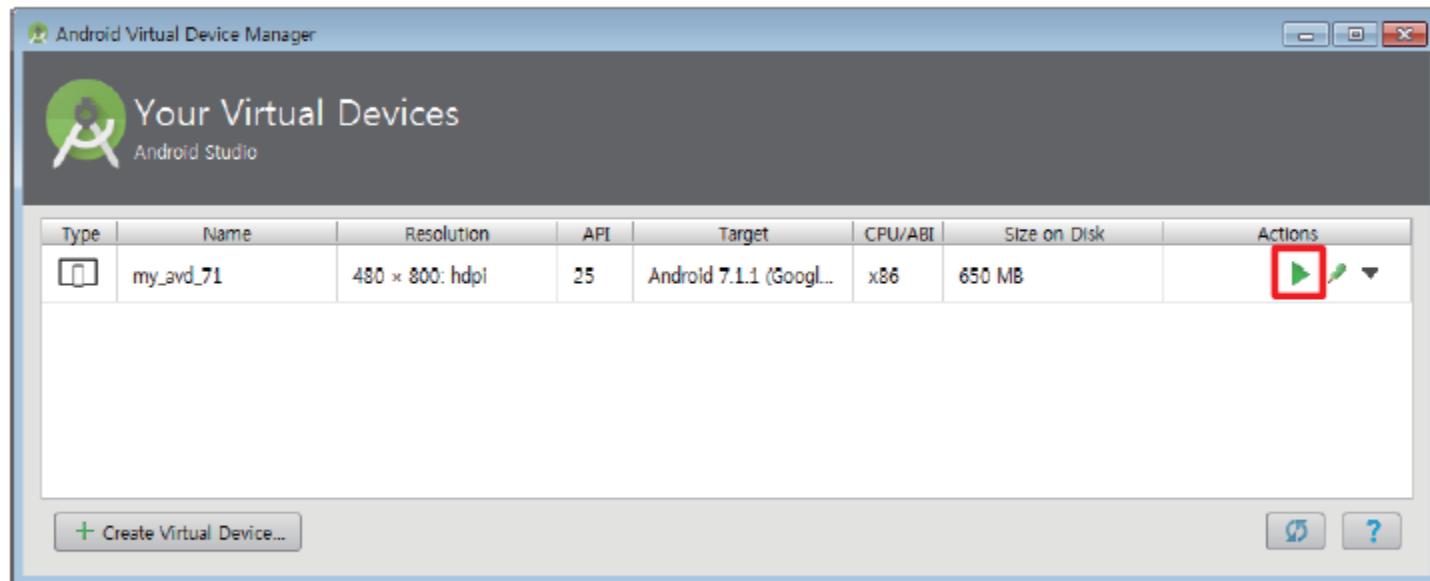
그림 1-21 AVD 세부 설정 화면

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

■ 에뮬레이터 실행



(a) 에뮬레이터 시작

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

■ 실행 에뮬레이터



(b) 에뮬레이터 화면 구성

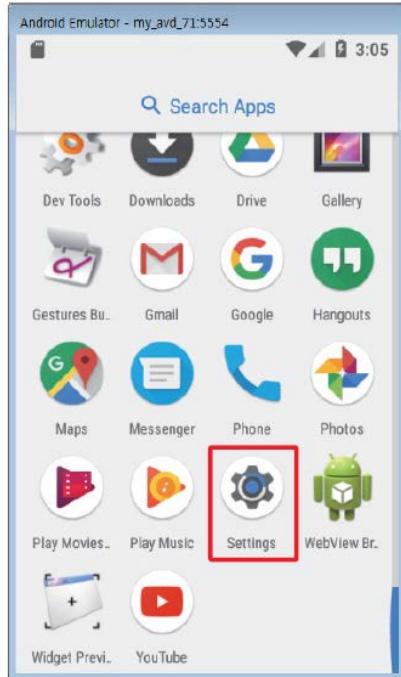
그림 1-22 에뮬레이터 실행

02 개발 환경 설정 ▶ 앱 설치와 제거

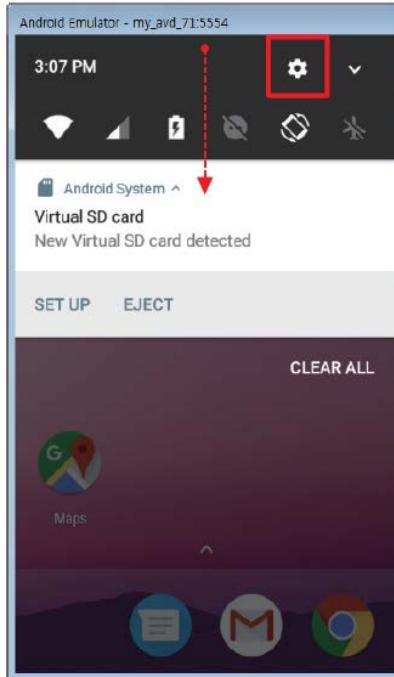


실습 1-5 AVD 생성, 실행, 설정

■ 시스템 기본 언어 한국어로 변경하기



(a) Settings 앱 실행



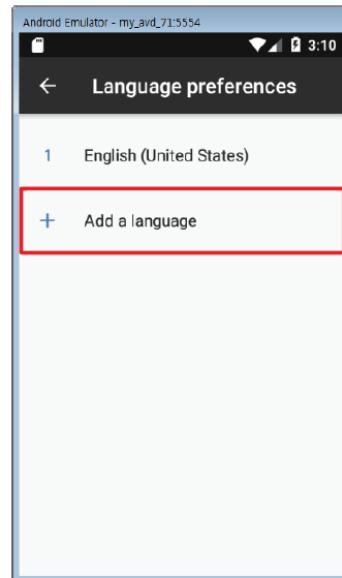
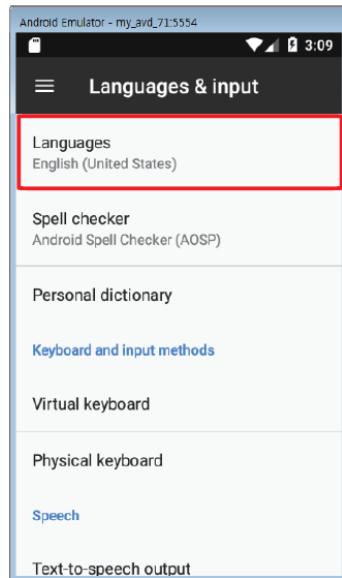
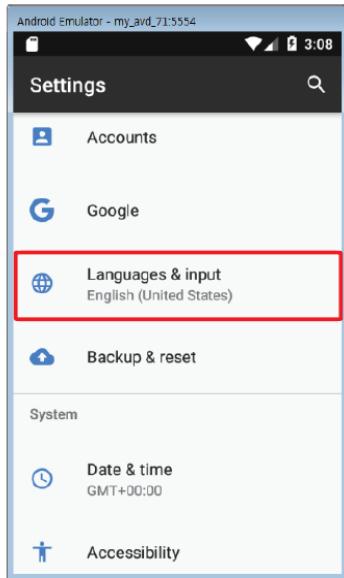
(b) 상단의 설정 아이콘 클릭

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

■ 표시된 항목으로 들어가 '대한민국' 선택



(c) 언어 항목 선택 (1)

(d) 언어 항목 선택 (2)

(e) 언어 항목 선택 (3)

(f) 언어 항목 선택 (4)

(g) 언어 항목 선택 (5)

02 개발 환경 설정 ▶ 앱 설치와 제거

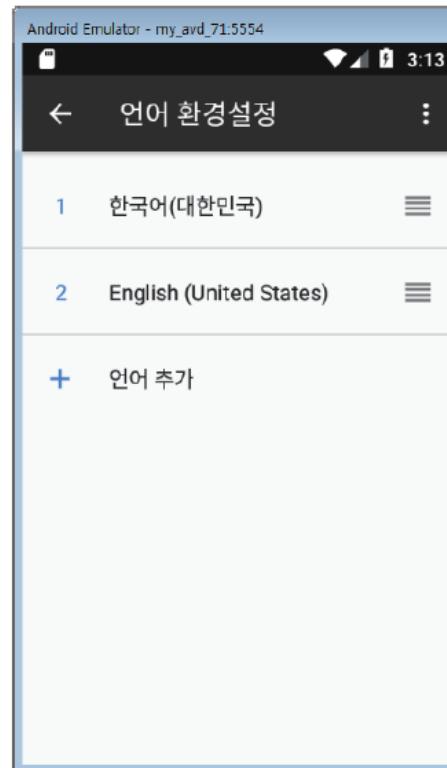


실습 1-5 AVD 생성, 실행, 설정

- '한국어(대한민국)'을 드래그하여 맨 위로 끌어올리기
- 시스템 기본 언어가 한국어로 변경



(h) 언어 항목 선택 (6)



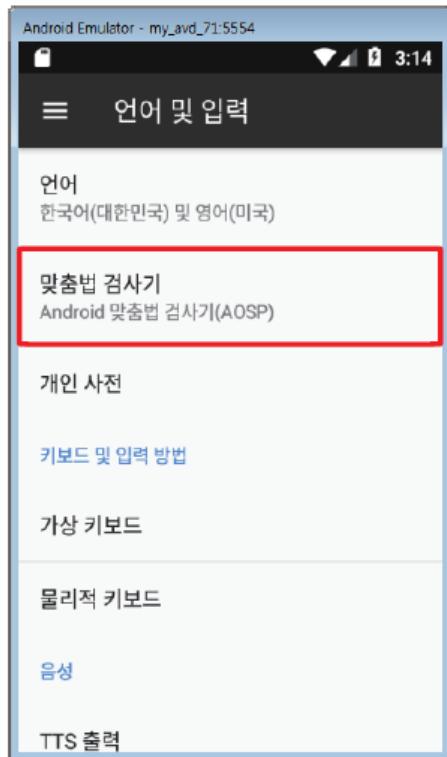
(i) 시스템 언어를 한국어로 변경

02 개발 환경 설정 ▶ 앱 설치와 제거

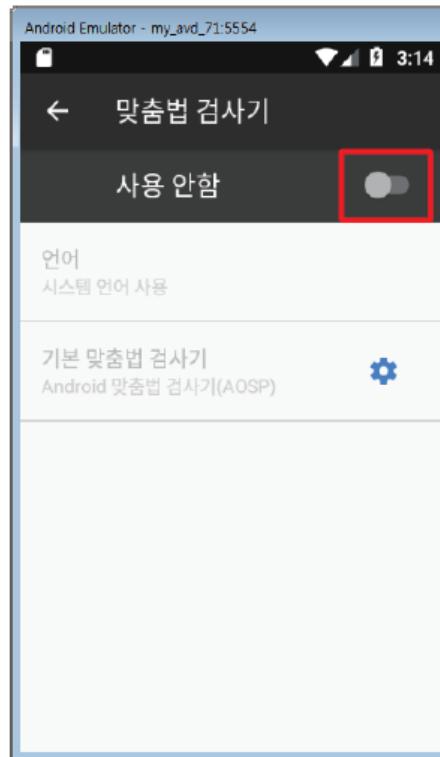


실습 1-5 AVD 생성, 실행, 설정

■ 맞춤법 검사기 해제



(j) 맞춤법 검사기 항목 선택



(k) 맞춤법 검사기 사용 해제

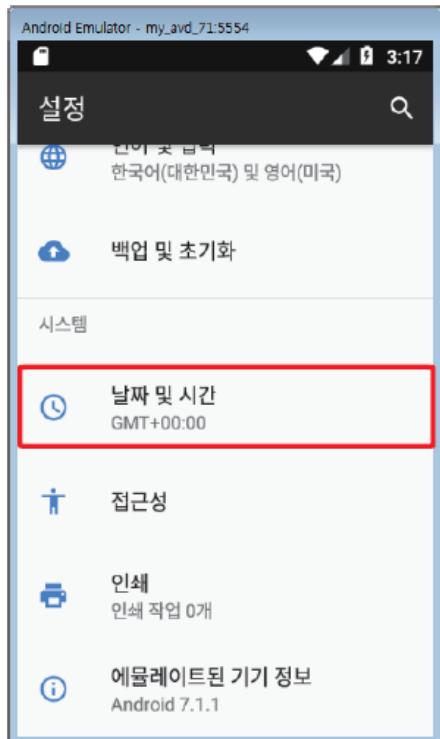
그림 1-23 시스템 언어 변경, 맞춤법 검사기 해제

02 개발 환경 설정 ▶ 앱 설치와 제거

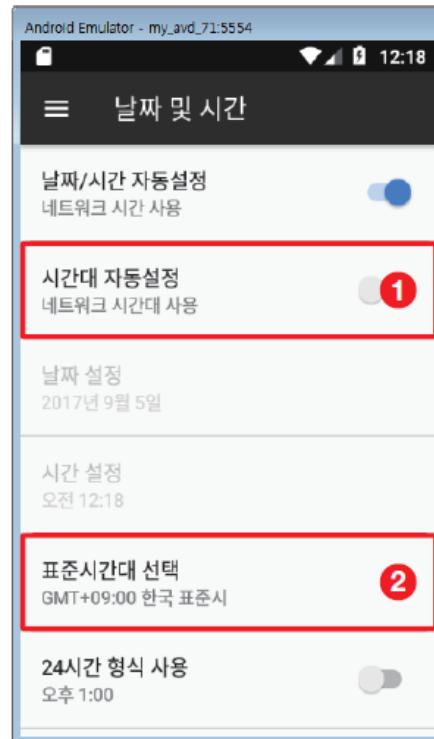


실습 1-5 AVD 생성, 실행, 설정

■ 표준시간대 변경하기



(a) 날짜 및 시간 선택



(b) 한국 표준시 선택

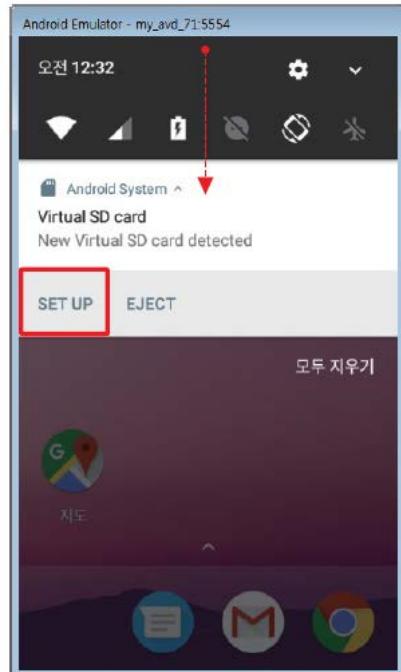
그림 1-24 표준시간대 변경

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-5 AVD 생성, 실행, 설정

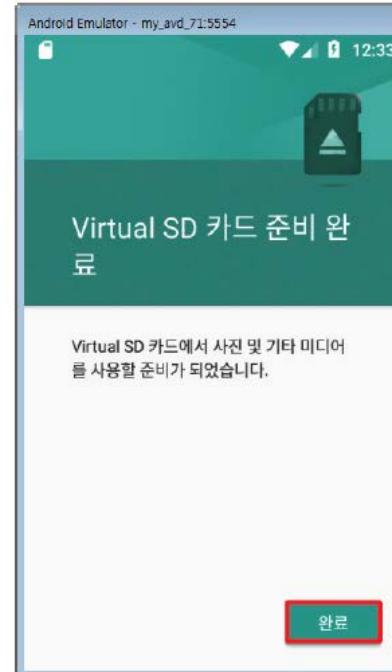
■ 인식된 SD 카드의 사용 방식 결정하기



(a) 가상 SD 카드 설정



(b) 휴대용 저장소 선택



(c) 선택 완료



(d) 메시지 지우기

그림 1-25 SD 카드 사용 방식 선택

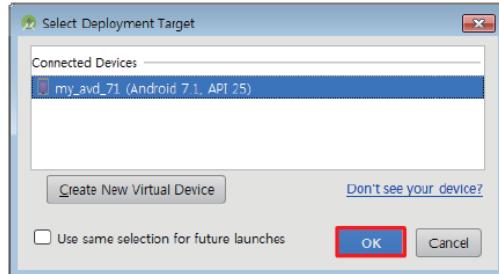
02 개발 환경 설정 ▶ 앱 설치와 제거



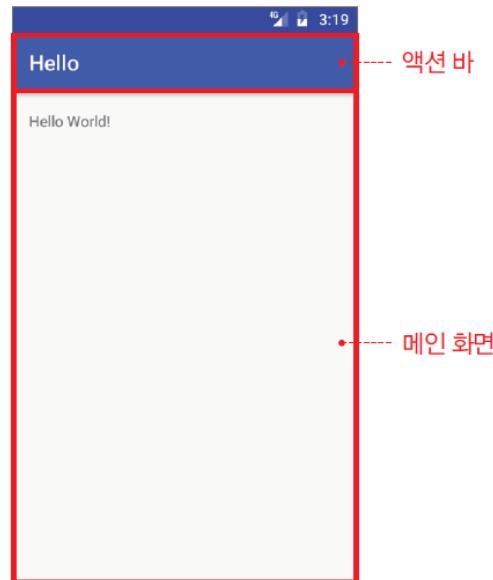
실습 1-6

앱 설치와 제거

■ Hello 앱을 설치하고 실행하기



(a) 디바이스 선택



(b) 실행 화면

그림 1-26 앱 설치 및 실행

02 개발 환경 설정 ▶ 앱 설치와 제거



실습 1-6

앱 설치와 제거

■ Hello 앱을 제거하기



(a) 앱 아이콘 길게 누름



(b) 홈 화면에서 드래그



(c) 앱 제거 대화상자

그림 1-27 앱 제거 절차

03 프로젝트 구조 분석 ▶ 프로젝트 구성

■ 프로젝트 구성

- 매니페스트 + 코드 + 리소스로 구성
- 개발 툴에 따라 달라지는 설정 파일을 포함

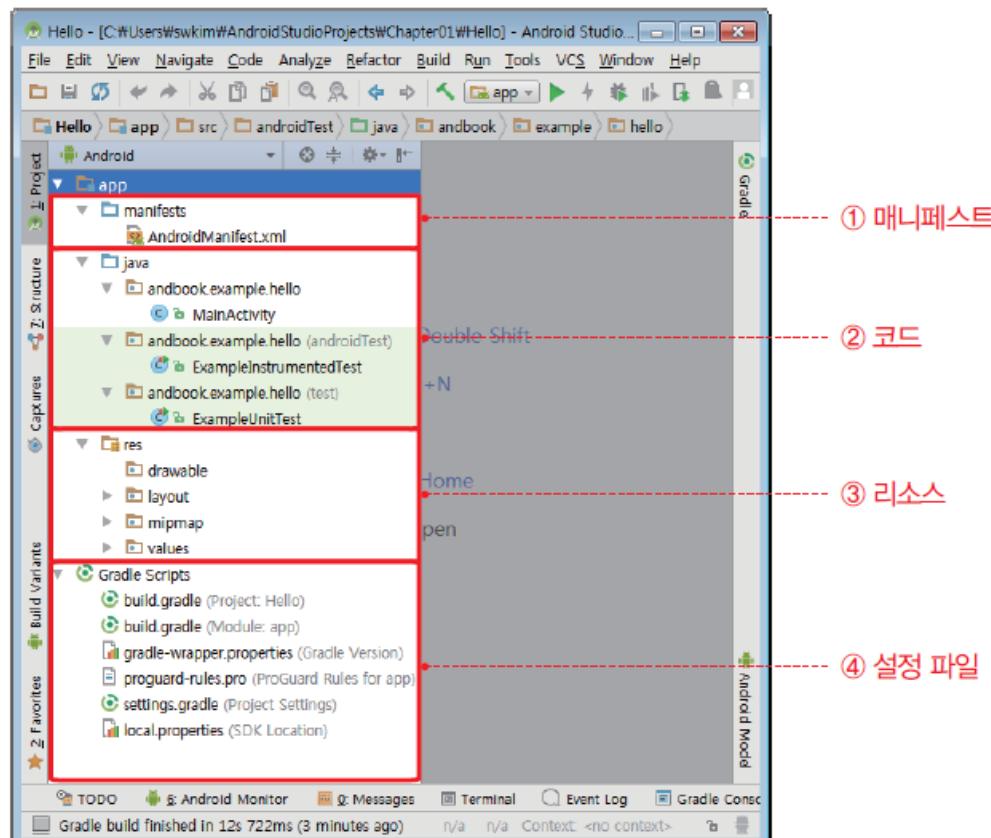


그림 1-28 안드로이드 프로젝트 구성 요소

03 프로젝트 구조 분석 ▶ 프로젝트 구성

표 1-4 프로젝트 구성 요소

구성 요소	역할
① 매니페스트	매니페스트 ^{manifest} 는 사전적 의미로 '승객 명단', '적하 목록'인데 안드로이드에서는 앱의 구성 요소(액티비티, 서비스, 브로드캐스트 수신기, 콘텐트 제공자) 목록을 담고 있다. 그 밖에 시스템에 전달할 여러 요구 사항이 포함되어 있는데 내용이 많으므로 학습을 진행하면서 필요할 때마다 소개하기로 한다. 매니페스트 파일 이름은 항상 <code>AndroidManifest.xml</code> 이고 다른 이름은 사용할 수 없다.
② 코드	코드는 메인 코드와 테스트 코드 두 종류가 있다. Hello 프로젝트의 메인 코드는 <code>MainActivity.java</code> 파일이고 나머지 파일들은 테스트 코드다. 테스트 코드는 자동화된 기능 테스트를 위한 자바 코드인데 실전에서 활용 빈도가 낮으므로 다루지 않기로 한다. 참고 이 책의 모든 예제는 메인 코드만 사용한다.
③ 리소스	리소스는 응용 프로그램이 사용할 자원이다. <code>res</code> 와 <code>assets</code> 폴더 두 종류가 있지만, 대부분의 리소스는 <code>res</code> 폴더에 존재한다. <code>assets</code> 폴더는 활용도가 낮아서 초기 프로젝트에는 존재하지 않는다. 각 리소스 폴더의 특징은 [표 1-5]에 정리되어 있다. 참고 이 책의 모든 예제는 <code>res</code> 폴더만 사용한다.
④ 설정 파일	매니페스트/코드/리소스 외의 구성 요소는 개발 툰에 따라 달라질 수 있다. 이를테면 안드로이드 스튜디오를 사용하느냐 이를립스를 사용하느냐에 따라 전혀 다른 파일들이 존재한다. 안드로이드 스튜디오는 Gradle이라 는 자동화 도구를 내부적으로 사용하므로 Gradle 스크립트라 통칭하는 파일들이 마련되어 있다.

03 프로젝트 구조 분석 ▶ 프로젝트 구성

표 1-5 리소스 폴더별 특징

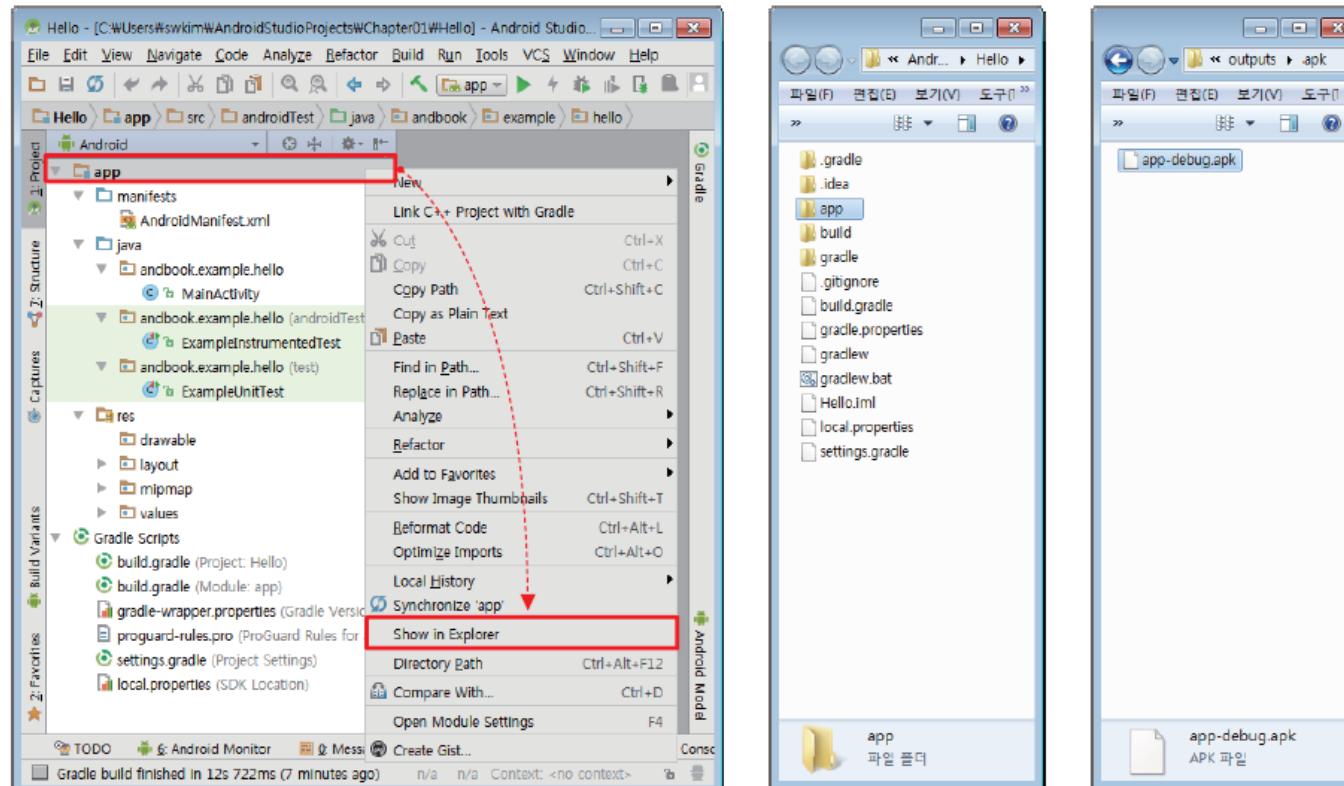
리소스 폴더	특징
res	리소스 파일의 이름은 a~z(영어 소문자), 0~9(숫자), _(밑줄 문자), .(점)으로만 구성해야 하며, 하위 폴더의 이름 (drawable, layout, ...)에 따라 넣을 수 있는 리소스 파일의 종류에 제약이 있다. 자바 코드에서 R 클래스(→ 자동 생성되는 자바 코드)로 접근한다.
assets	리소스 파일의 이름이나 종류에 제약이 없다. 자바 코드에서 AssetManager 클래스로 접근한다.

03 프로젝트 구조 분석 ▶ 설치 형식(.apk)

■ 안드로이드 앱의 최종 설치 형식은 apk 확장자를 가진 파일

■ 내부적으로는 표준 zip 형식이다.

- 'app'에서 우클릭 ▶ [Show in Explorer] 메뉴를 선택 ▶ Hello 프로젝트 폴더 열기



(a) 탐색기 열기

(b) 프로젝트 폴더

(c) .apk 파일

그림 1-29 .apk 파일 찾아가기

03 프로젝트 구조 분석 ▶ 설치 형식(.apk)

- .apk는 매니페스트 + 코드 + 리소스로 구성
- 디지털 서명이 되어 있음

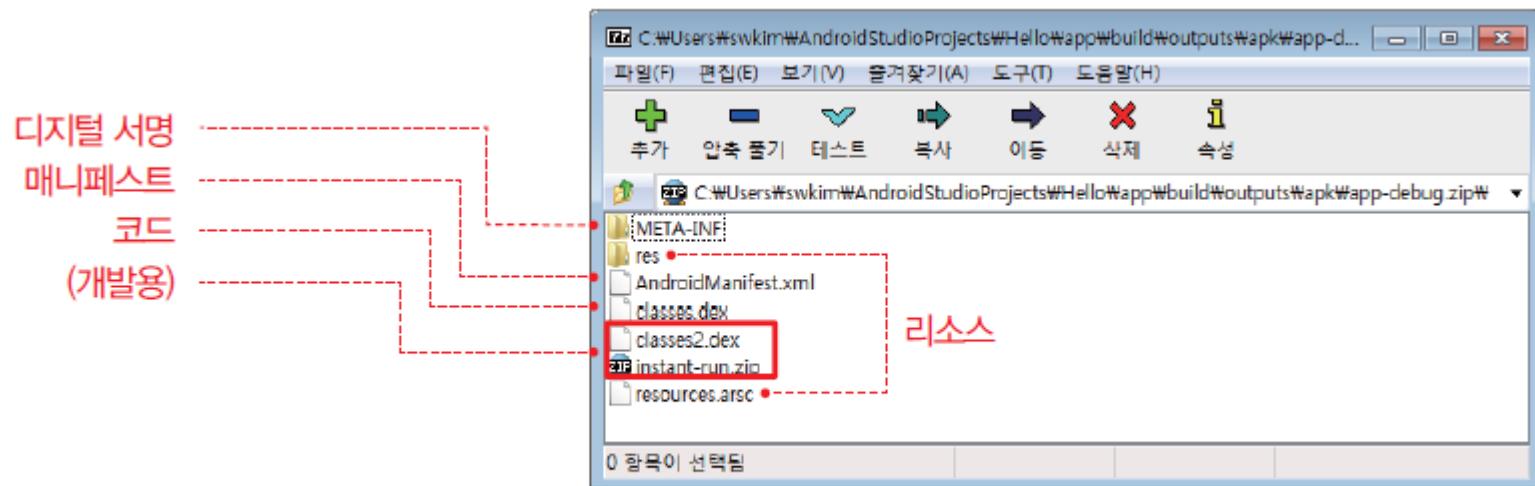


그림 1-30 .apk 파일 내부 구성

03 프로젝트 구조 분석 ▶ 설치 형식(.apk)

■ 안드로이드 앱

- 자바 언어로 작성하므로 자바 가상 머신 JVM이 있어야 실행

■ 안드로이드 시스템

- 표준 JVM 대신 Dalvik (안드로이드 4.4 이하) 또는 ART(안드로이드 5.0 이상) 사용

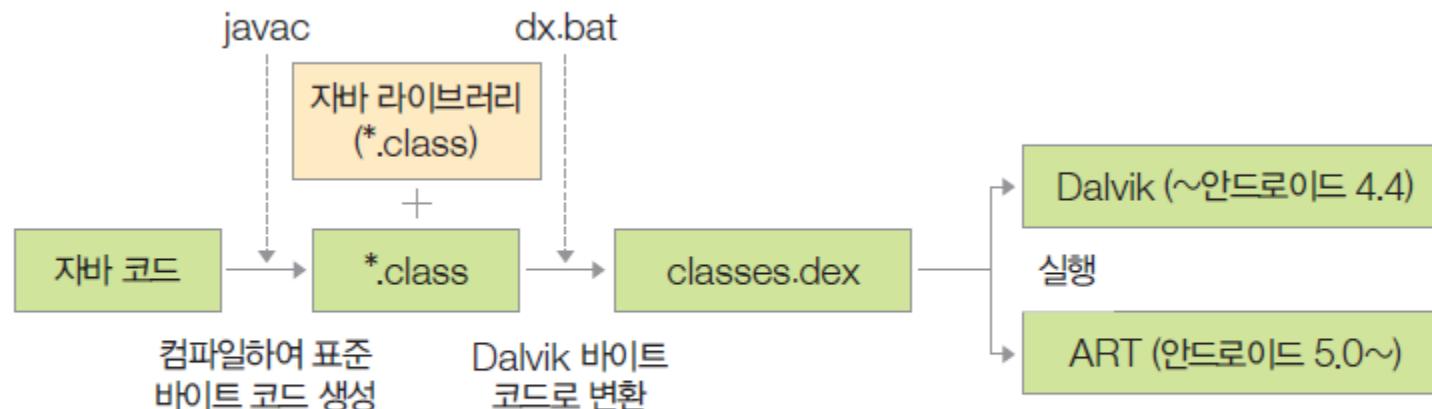


그림 1-31 자바 코드 컴파일 → 변환 → 실행

03 프로젝트 구조 분석 ▶ 앱 기능 추가

■ 매니페스트 분석

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

03 프로젝트 구조 분석 ▶ 앱 기능 추가

■ 액티비티 코드 분석

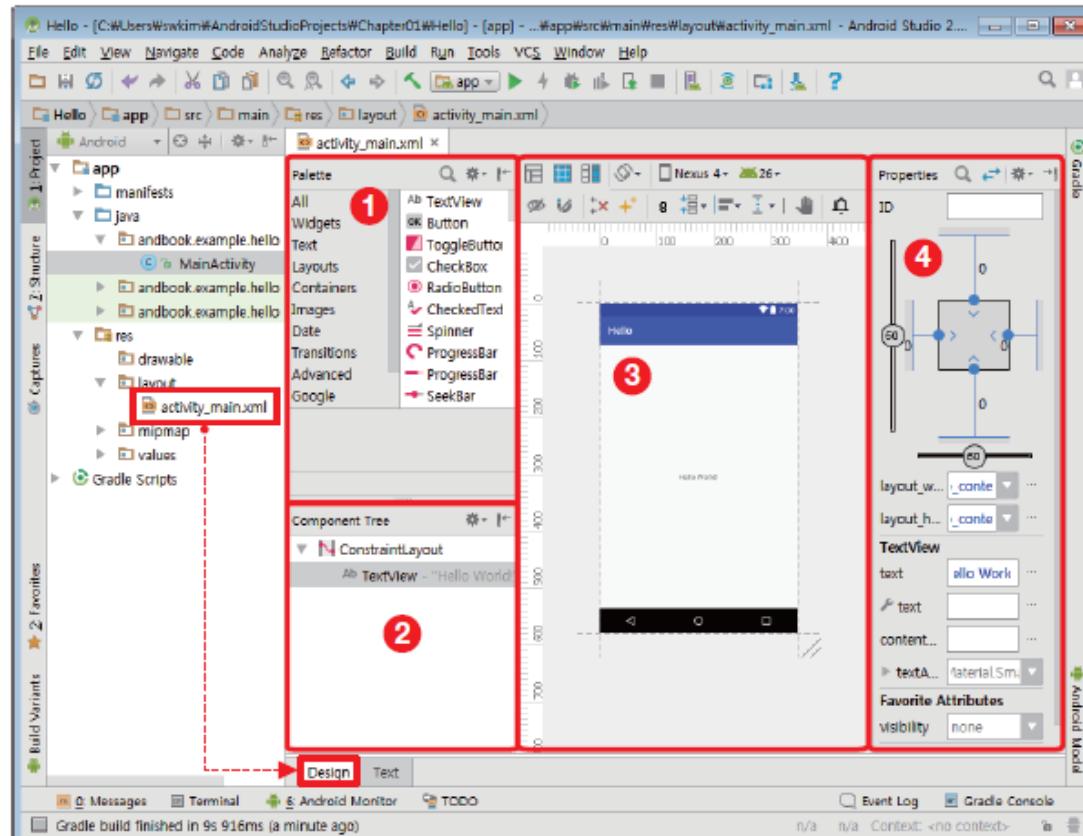
- Hello 앱의 액티비티 코드가 들어 있는 MainActivity 클래스

```
1 package andbook.example.hello;  
2  
3 import android.support.v7.app.AppCompatActivity;  
4 import android.os.Bundle;  
5  
6 public class MainActivity extends AppCompatActivity {  
7  
8     @Override  
9     protected void onCreate(Bundle savedInstanceState) {  
10         super.onCreate(savedInstanceState); // 수퍼클래스의 onCreate() 메서드를 호출한다.  
11         setContentView(R.layout.activity_main); // 액티비티 화면을 초기화한다.  
12     }  
13 }
```

03 프로젝트 구조 분석 ▶ 앱 기능 추가

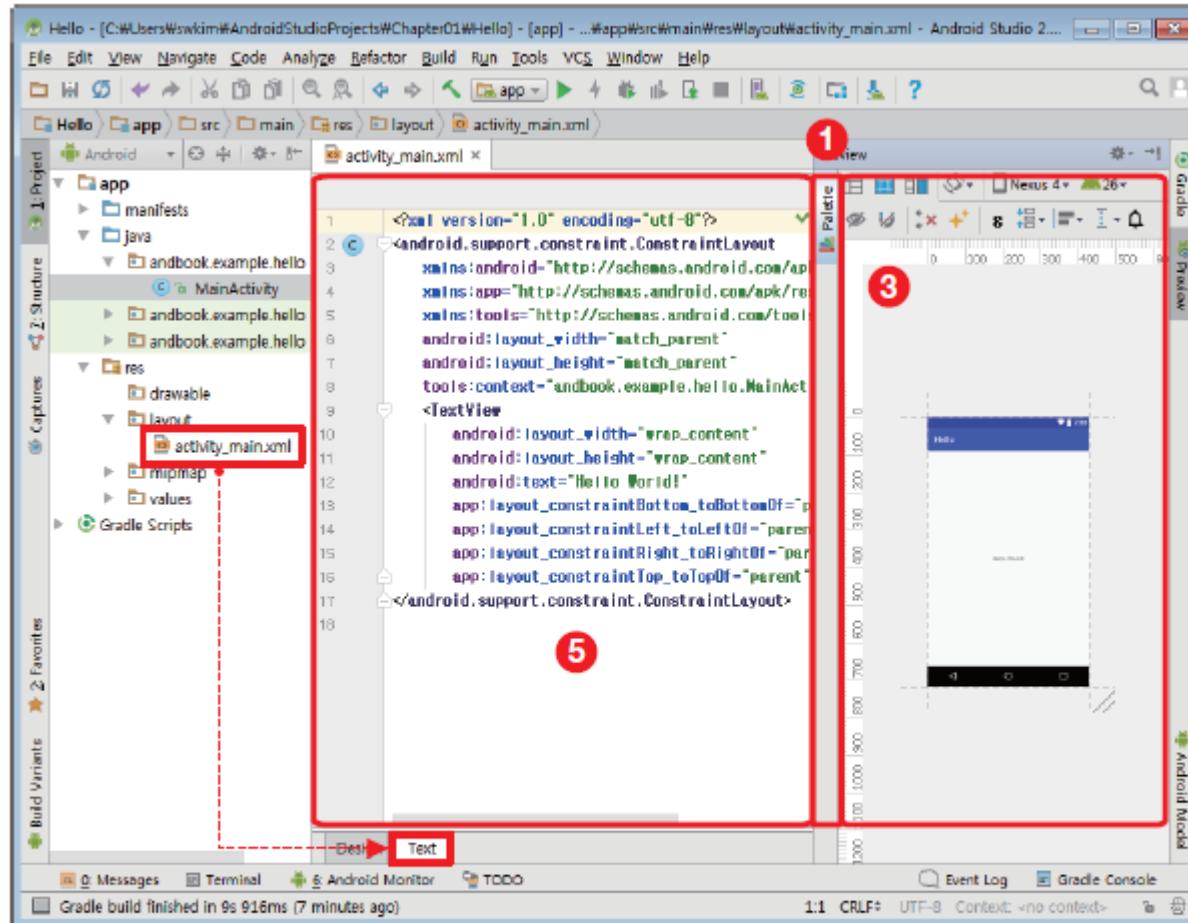
■ 레이아웃 파일 보기

- 레이아웃(layout) : 액티비티 화면을 정의하는 XML 파일



(a) Design 탭

03 프로젝트 구조 분석 ▶ 앱 기능 추가

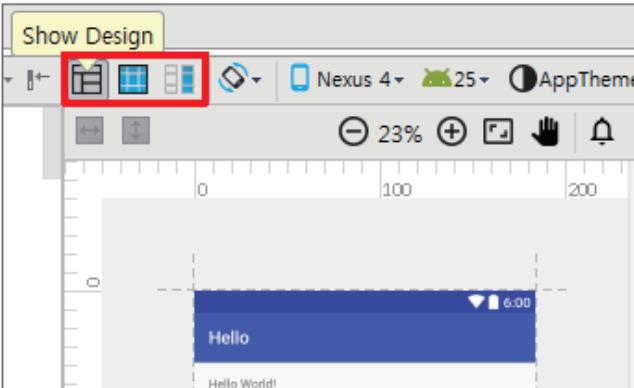
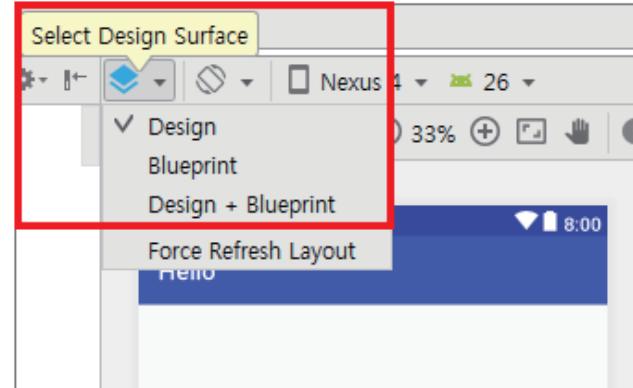


(b) Text 탭

그림 1-32 activity_main.xml – 초기 상태

03 프로젝트 구조 분석 ▶ 앱 기능 추가

표 1-6 Design 탭과 Text 탭의 구성 요소

번호	기능
①	드래그 & 드롭 방식으로 화면을 디자인할 수 있도록 팔레트를 제공한다.
②	레이아웃의 계층 구조를 보여주며, 마우스로 드래그하여 계층 구조를 변경할 수 있다.
③	미리보기 화면을 보여주며, 키보드나 마우스를 이용하여 디자인을 변경할 수 있다. 예를 들어 Delete 키를 누르면 특정 요소를 삭제할 수 있고, 마우스로 드래그하면 특정 요소의 위치를 변경할 수 있다. 좌측 상단의 툴바를 사용하면 미리보기 화면을 Design/Blueprint/Design+Blueprint 형식으로 바꿔가며 볼 수 있다.  
④	각종 속성을 보거나 변경한다.
⑤	레이아웃 XML을 직접 편집한다.

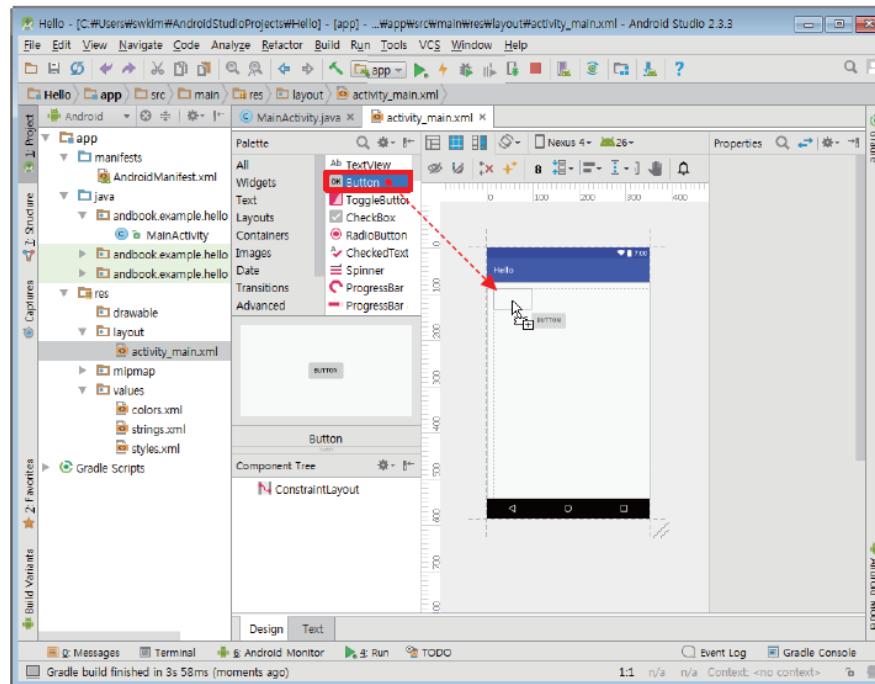
03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-7

버튼 추가

- “Hello World!” 텍스트를 마우스로 클릭한 후 Delete 키를 눌러 삭제
- Design 탭의 좌측 화면에서 ‘Button’을 클릭 ▶ 미리보기 화면의 좌측 상단에 드래그 & 드롭



(a) Design 탭

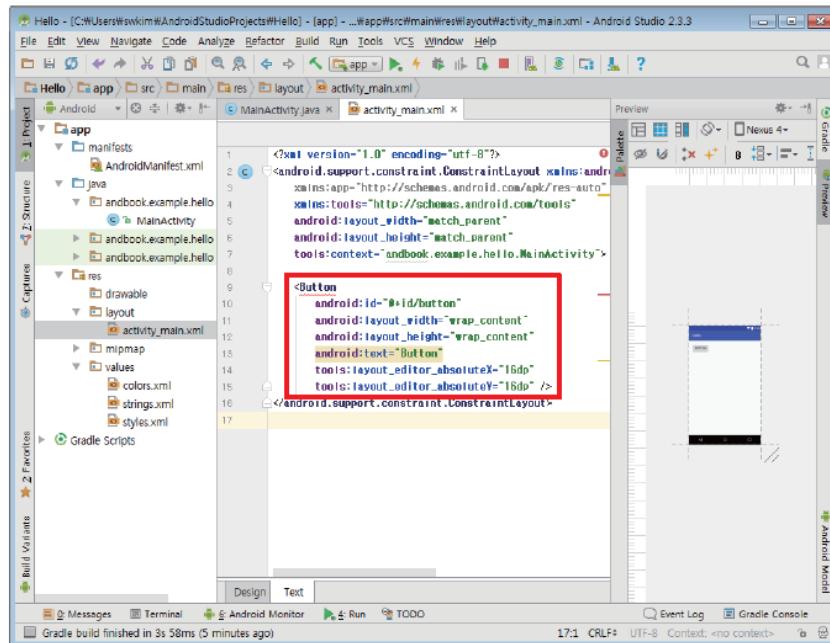
03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-7

버튼 추가

- 버튼을 추가한 결과는 미리보기 화면에 곧바로 반영
- Text 탭에서도 확인 가능



(b) Text 탭

그림 1-33 activity_main.xml – 레이아웃에 버튼 추가



실습 1-7

버튼 추가

■ 자바 코드에서 사용할 준비하기

- MainActivity 클래스에 9행과 16행을 입력
- 5행은 9행을 입력하고 Alt+Enter를 누르면 자동으로 추가

```
1 package andbook.example.hello;  
2  
3 import android.support.v7.app.AppCompatActivity;  
4 import android.os.Bundle;  
5 import android.widget.Button;  
6  
7 public class MainActivity extends AppCompatActivity {  
8  
9     private Button mButton; // 이 부분을 입력하고 Alt+Enter를 누른다.  
10  
11    @Override  
12    protected void onCreate(Bundle savedInstanceState) {  
13        super.onCreate(savedInstanceState); // 수퍼클래스의 onCreate() 메서드를 호출한다.  
14        setContentView(R.layout.activity_main); // 액티비티 화면을 초기화한다.  
15  
16        mButton = (Button) findViewById(R.id.button); // 레이아웃에서 버튼 객체를 찾아낸다.  
17    }  
18}
```

03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-7

버튼 추가

■ Hello 앱을 에뮬레이터나 디바이스에 설치해서 실행

- 안드로이드 스튜디오의 [Run]-[Run 'app'] 메뉴를 선택

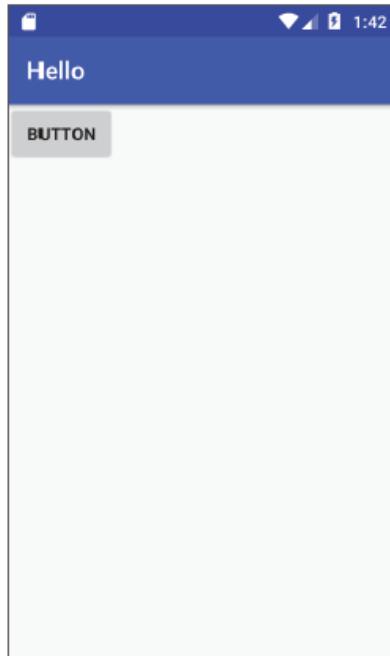


그림 1-34 실행 화면

03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-8 토스트 메시지 출력

■ 버튼을 클릭하면 간단한 메시지를 출력하는 코드 작성

```
1 package andbook.example.hello;  
2  
3 import android.support.v7.app.AppCompatActivity;  
4 import android.os.Bundle;  
5 import android.view.View;  
6 import android.widget.Button;  
7  
8 public class MainActivity extends AppCompatActivity {  
9  
10    private Button mButton; // 이 부분을 입력하고 Alt+Enter를 누른다.  
11  
12    @Override  
13    protected void onCreate(Bundle savedInstanceState) {  
14        super.onCreate(savedInstanceState); // 수퍼클래스의 onCreate() 메서드를 호출한다.  
15        setContentView(R.layout.activity_main); // 액티비티 화면을 초기화한다.  
16  
17        mButton = (Button) findViewById(R.id.button); // 레이아웃에서 버튼 객체를 찾아낸다.  
18        mButton.setOnClickListener(new View.OnClickListener() {  
19            @Override  
20            public void onClick(View view) {  
21                // 버튼 클릭 시 실행할 코드를 여기에 작성한다.  
22            }  
23        });  
24    }  
25}
```

03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-8 토스트 메시지 출력

표 1-7 코드 입력 순서

단계	입력 내용
1	"m" 입력 후 후보 목록 맨 위에 mButton이 보이므로 Enter
2	".setOn" 입력 후 후보 목록에서 setOnClickListener…를 선택하여 Enter
3	자동 완성된 () 안에서 "new On" 입력 후 후보 목록 맨 위에 setOnClickListener…가 보이므로 Enter

03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-8 토스트 메시지 출력

간단한 메시지를 출력할 경우 Toast 클래스 사용

```
1 package andbook.example.hello;  
2  
3 import android.support.v7.app.AppCompatActivity;  
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 AppCompatActivity {  
10  
11     private Button mButton; // 이 부분을 입력하고 Alt+Enter를 누른다.  
12  
13     @Override  
14     protected void onCreate(Bundle savedInstanceState) {  
15         super.onCreate(savedInstanceState); // 수퍼클래스의 onCreate() 메서드를 호출한다.  
16         setContentView(R.layout.activity_main); // 액티비티 화면을 초기화한다.  
17  
18         mButton = (Button) findViewById(R.id.button); // 레이아웃에서 버튼 객체를 찾아낸다.  
19         mButton.setOnClickListener(new View.OnClickListener() {  
20             @Override  
21             public void onClick(View view) {  
22                 Toast.makeText(MainActivity.this, "버튼 클릭!", Toast.LENGTH_SHORT).show();  
23             }  
24         });  
25     }  
26 }
```

03 프로젝트 구조 분석 ▶ 앱 기능 추가



실습 1-8 토스트 메시지 출력

■ 실행 화면

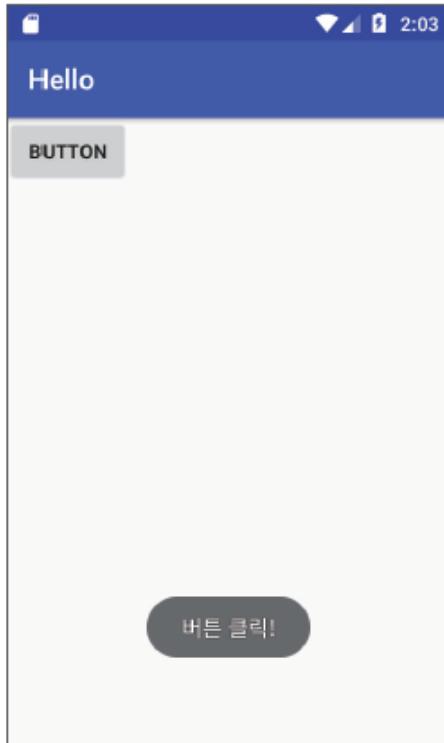


그림 1-35 실행 화면

04 안드로이드 스튜디오 기초 ▶ 화면 구성

■ 안드로이드 스튜디오 화면의 요소

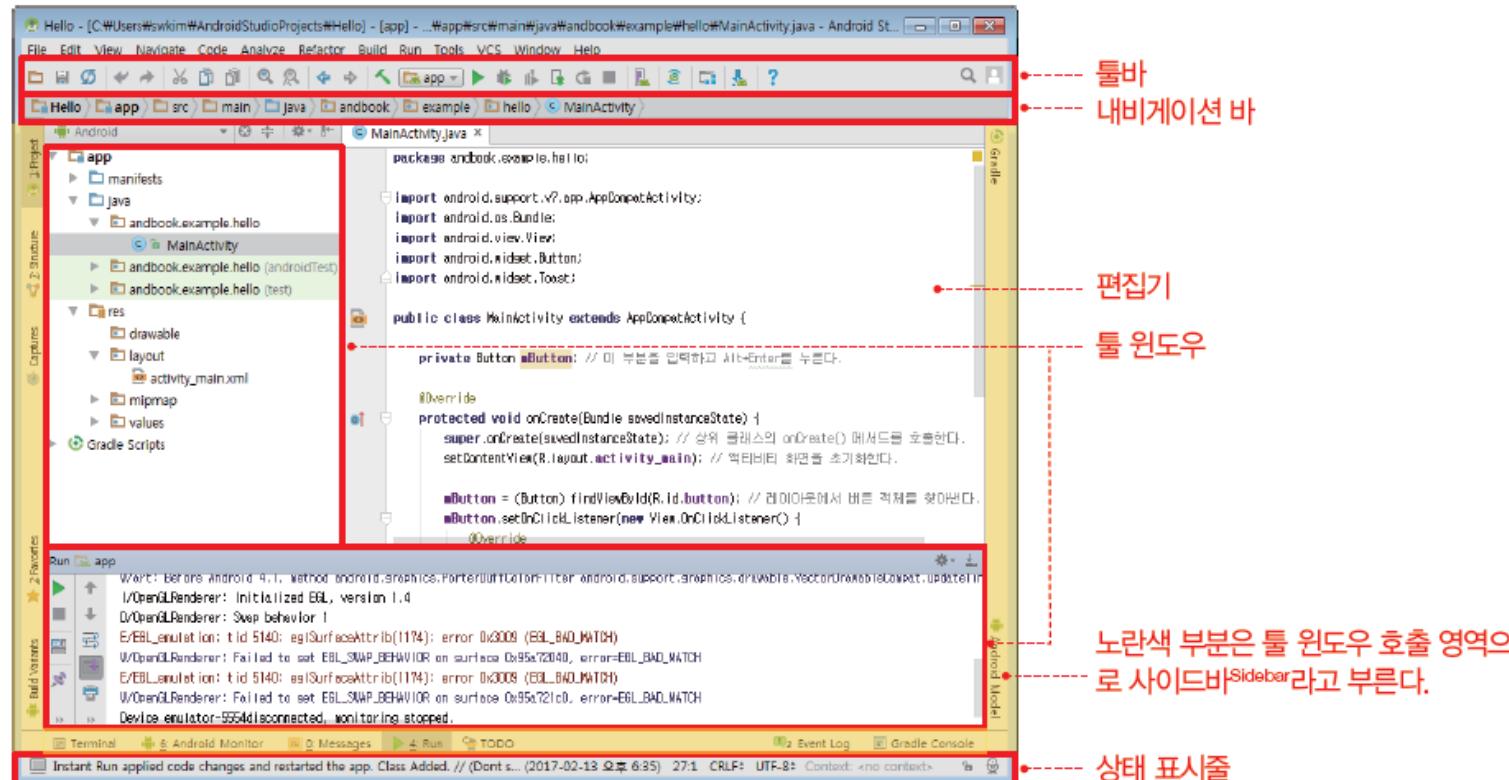


그림 1-36 안드로이드 스튜디오 화면 구성

04 안드로이드 스튜디오 기초 ▶ 툴 윈도우

■ 자주 사용되는 툴 윈도우

표 1-8 주요 툴 윈도우

이름	단축키	기능
Project	Alt+1	프로젝트의 계층 구조와 구성 파일을 보여준다.
Structure	Alt+7	Project 윈도우에서 선택한 파일의 내부 구성을 계층적으로 보여준다.
Android Monitor[AS 2.3]/ Logcat[AS 3.0]	Alt+6	안드로이드에서 실행 중인 프로세스 목록과 더불어 내부에서 발생하는 로그 메시지를 표시하므로 디버깅할 때 자주 사용된다.

04 안드로이드 스튜디오 기초 ▶ 설정 변경

■ 코드 접기(code folding) 기능 해제

- [File]-[Settings...]-[Editor]-[General]-[Code Folding] 설정
- 최상위 옵션을 제외하고 모두 해제

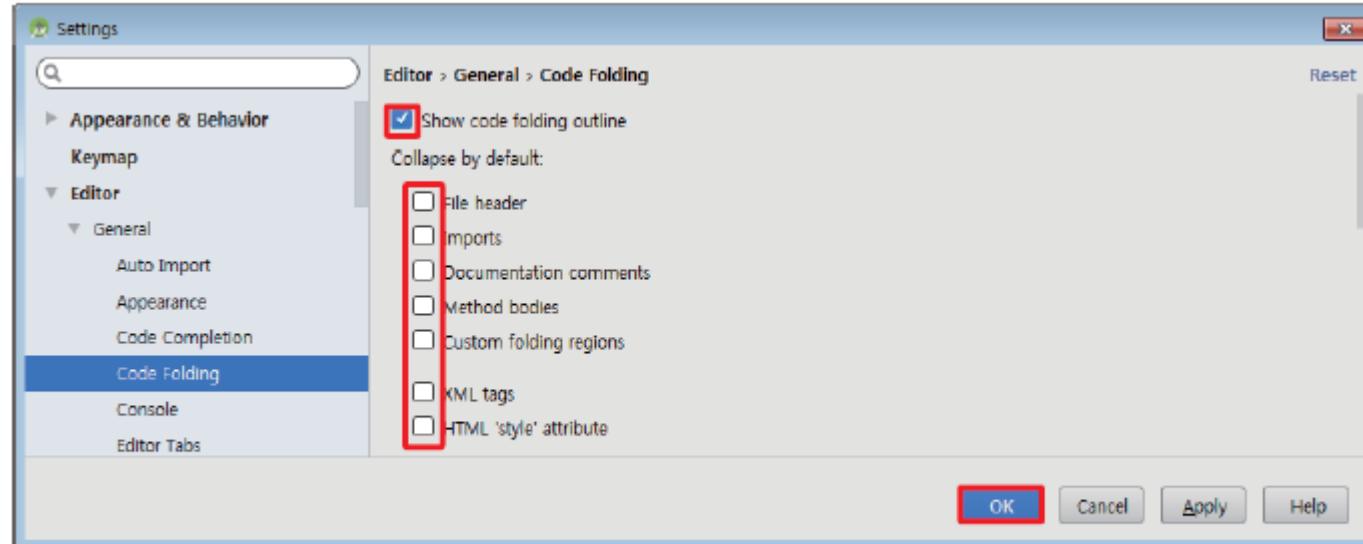


그림 1-37 코드 접기 비활성화

04 안드로이드 스튜디오 기초 ▶ 설정 변경

■ 철자 체크 비활성화

- [File]-[Settings...]-[Editor]-[Inspections] 설정에서 Spelling 옵션을 해제

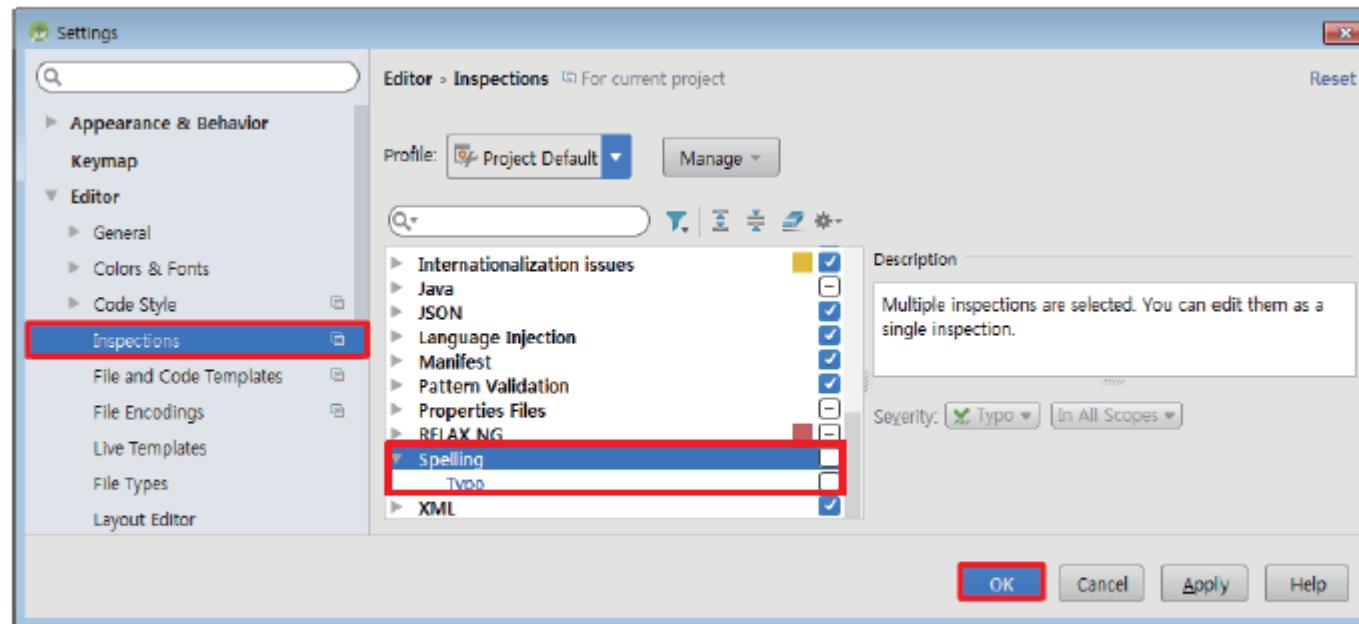


그림 1-38 철자 체크 비활성화

04 안드로이드 스튜디오 기초 ▶ 설정 변경

■ 자동 임포트 기능 활성화

- 자바 코드 수정 시 불필요한 임포트는 자동으로 제거되고 새로운 자동으로 임포트
- [File]-[Settings...]-[Editor]-[General]-[Auto Import] 설정을 모두 체크

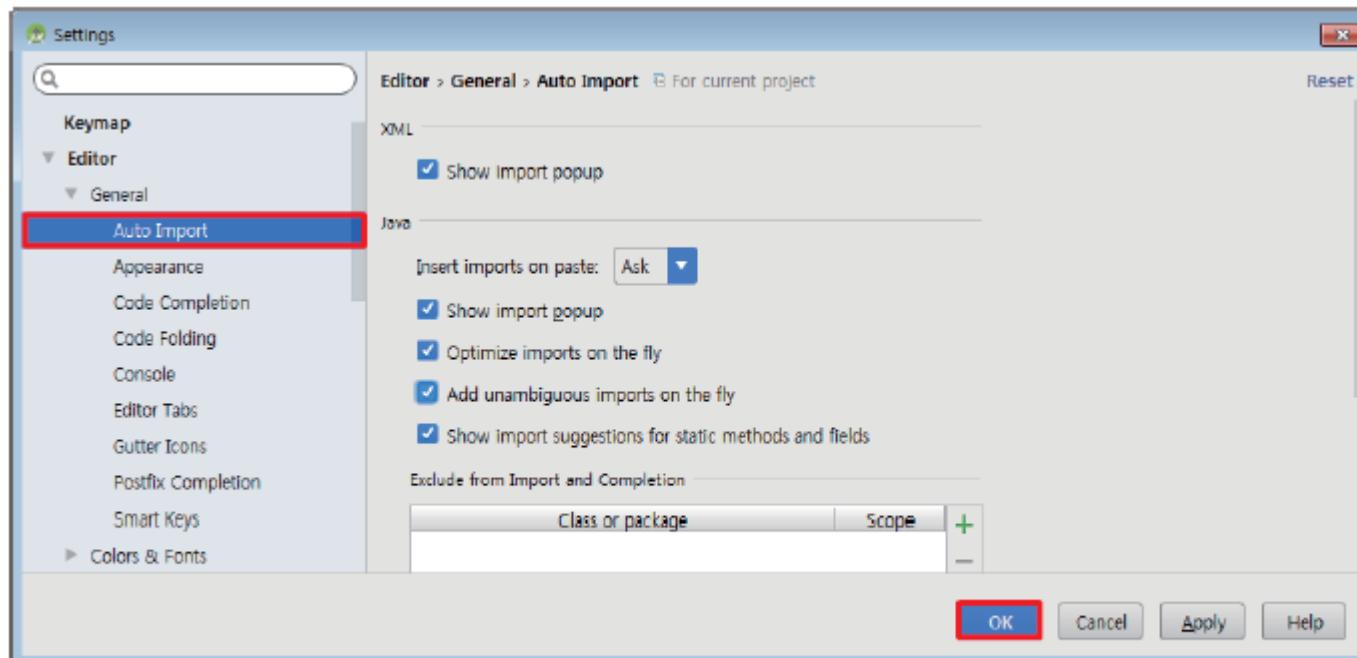


그림 1-39 자동 임포트 활성화

04 안드로이드 스튜디오 기초 ▶ 프로젝트 관리

■ 기존 프로젝트 열기

- 프로젝트가 열려 있지 상태에서 기존 프로젝트를 열 경우
- [Open anexisting Android Studio project] 메뉴를 선택
- 'Open File or Project' 대화상자가 열리면 프로젝트 폴더 경로를 찾아서 [OK] 버튼 클릭

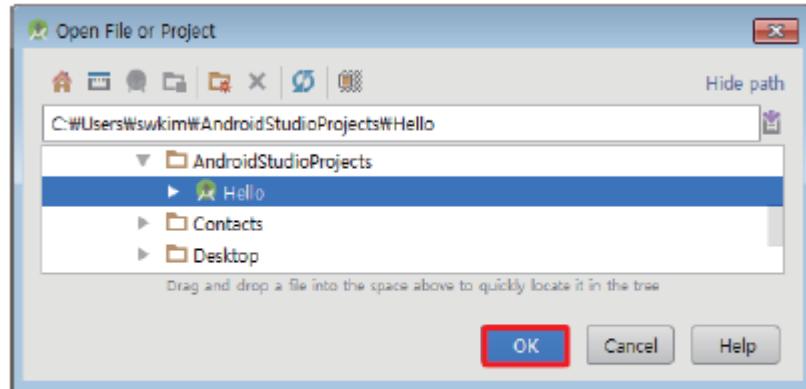


그림 1-40 'Open File or Project' 대화상자

■ 다른 프로젝트 추가 열기

- 프로젝트가 열려 있는 상태에서 다른 프로젝트를 추가로 열 경우
- 프로젝트 폴더를 찾아서 [OK] 버튼을 클릭
- [This Window]: 현재 프로젝트가 닫히면서 새로운 프로젝트로 대체
- [New Window]: 현재 프로젝트와는 별도의 안드로이드 스튜디오 창이 실행되어 새로운 프로젝트 열림

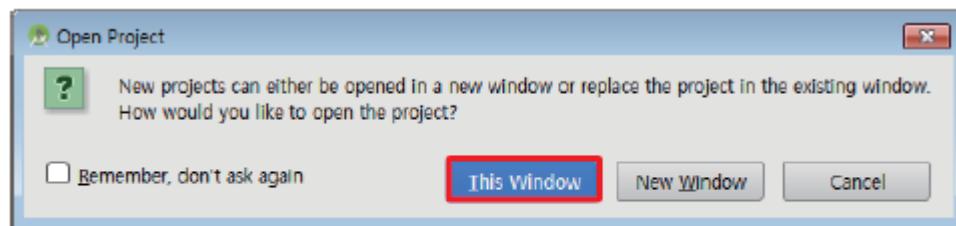


그림 1-41 'Open Project' 대화상자

■ 프로젝트 닫기

- [File]-[Close Project] 메뉴를 선택

단계별로 배우는

안드로이드 프로그래밍

