AVD 생성 및 실행

기초 원리 학습

OneKick

- 안드로이드 플랫폼
  - 운영체제, 미들웨어, 키 응용프로그램을 가지고 있는 모바일 장치용 소프트웨어 계층구조
  - 기존 자바와 위피 개발자가 쉽게 적응할 수 있도록 스마트폰 기능을 내장
    - 오픈소스기반 Linux를 채택하여 라이센스 문제를 해결
  - 제공하는 애플리케이션 프레임워크는 재사용성이 뛰어난 API를 기본 탑재
    - 생산성이 뛰어난 장점
- 안드로이드 플랫폼은 리눅스를 기반
  - POSIX 리눅스와 동일한 것은 아님

기초 원리 학습

OneKick

- WebKit 엔진을 활용한 통합 브라우저 환경을 제공
  - 크롬이나 사파리 웹브라우저도 플레이스토어를 통해 다운로드 받아 설치할 수 있는 사용자 편의를 제공

- SQLite라는 저용량 고성능 DBMS를 기존 장착
  - 하드웨어에서 기본으로 탑재된 전화 통신망 기능을 활용할 수 있는 서비스 제공
  - 카메라 접근 기능은 안드로이드 플랫폼에서 서비스로 기본적으로 제공

- 사용자 위치 정보를 GPS 기반으로 표시
  - 지도 정보 서비스나, 길 찾기 서비스 등 다양한 응용 컨텐츠 개발에 활용
  - 사용자의 행동을 파악하기 위한 가속도 센서나 근접센서의 API도 제공

기초 원리 학습

OneKick

- 안드로이드 플랫폼
  - 개발자를 위해 AVD(Android Virtual Device)를 제공
  - 디버거와 로그캣을 제공
    - 예외 처리 및 디버깅을 하는데 매우 편리
  - 리눅스 커널 기반
    - 리눅스의 장점 제공
  - 메모리 관리 기능이나 프로세스 스케줄링은 매우 안전하고 효율적인 장점
  - 퍼미션 기반으로 보안을 유지
    - 신뢰도가 높으며 리눅스의 드라이버 모델과 공유 라이브러리도 그대로 지원
  - 오픈소스의 장점
    - 다른 하드웨어로 포팅을 한다든지 할 때 확장성 제공

기초 원리 학습

OneKick

- 안드로이드 플랫폼이 리눅스를 기반으로 하고 있지만 차이가 있음
  - glibc 지원 하지 않음.
  - Native Windowing system(X-Window)이 없음
  - EABI 사용, OpenBinder (No sysVIPC)
  - 커널패치를 통한 추가 컴포넌트 제공
  - Alarm, Ashamem, Lowmemory-killer, debugger, logger...

기초 원리 학습

OneKick

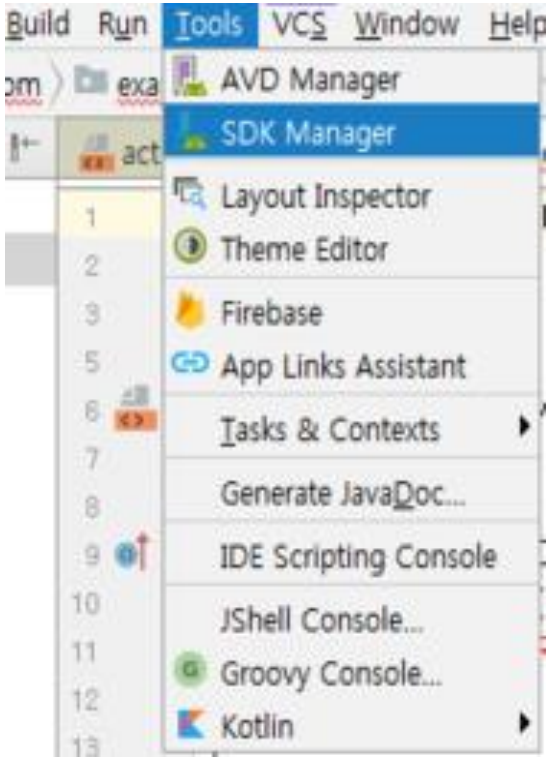
- POSIX를 완벽하게 준수하지 않기 때문에 아래와 같은 제약사항 있음
  - C++ Exception , wide character 지원하지 않음
  - Glibc와 호환되지 않음
  - 모든 Native코드는 bionic과 함께 컴파일 되어야 함.
  - Bionic libc와 함께 빌드되면, 경량, 고성능 보장
  - 3rd party Tools 빌드 사례(ping, iwconfig, 등)

기초 원리 학습

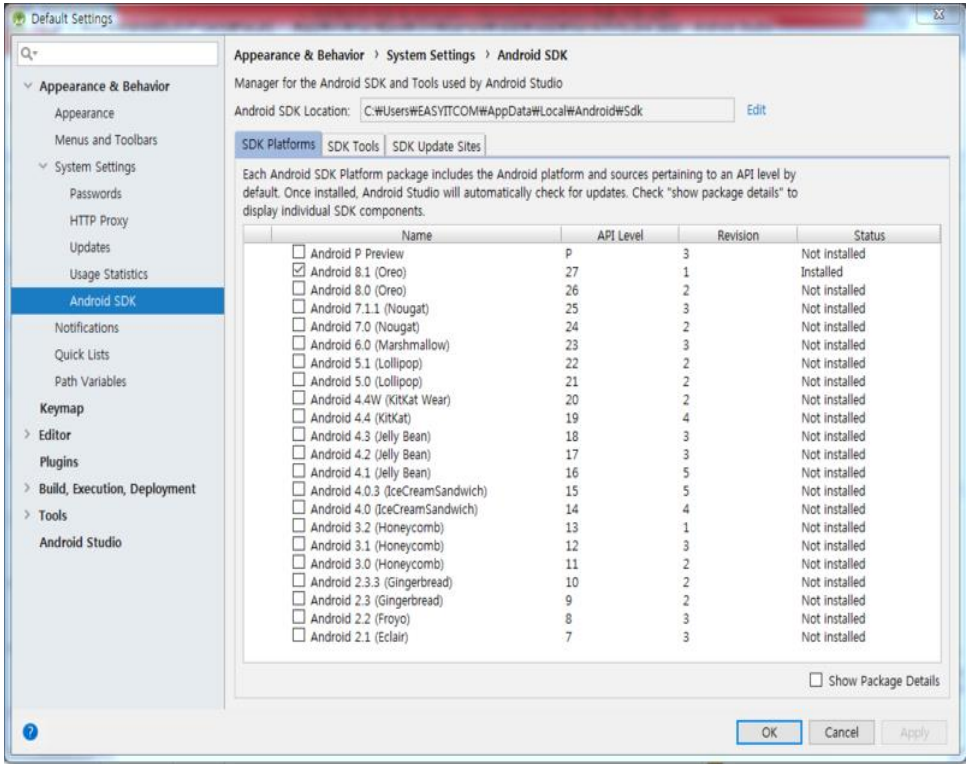
OneKick

- 플레이스토어의 특징
  - 구글 플레이스토어 기반의 중앙 집중형 마켓 형성
  - 정기적인 안드로이드 개발자 대회 개최
  - Java 언어를 채택하여 기존 Java 개발자 유치
  - 많은 컨텐츠 개발자 유입을 위한 큰 온라인 마켓 형성
  - 개발자 등록비 25$
  - 수식 배분 7:2:1 = 개발자: Google : 통신사= 70% : 20 % : 10%
  - 마켓 주소: http://www.android.com/market
  - 안드로이드는 광고수익을 주 전략으로 세움

- **안드로이드 SDK 업데이트하기**
  - Tools 메뉴 아래에 있는 SDK 매니저 실행
  - 왼쪽 메뉴에서 Android SDK 선택
  - SDK Platforms 확인

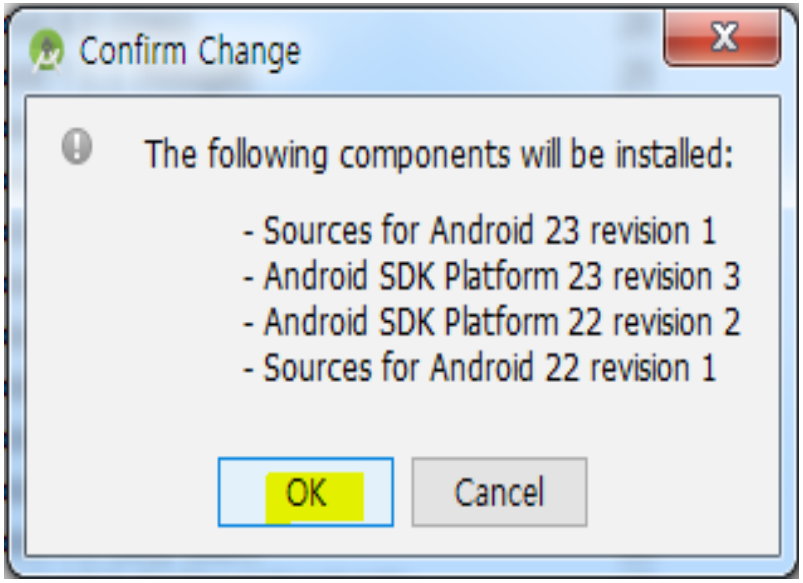


SDK Manager 실행



Android SDK Platforms 선택

- **안드로이드 SDK 업데이트하기**
  - 우측 하단의 Show Package Detail 항목을 선택
  - 설치해야하는 버전을 모두 선택한 뒤 다음으로 진행
  - 버전이 높을수록 다운로드 하는 데이터가 많아서 시간이 많이 소요되기 때문에 당장 필요한 것만 다운로드 하고 추후 필요할 때마다 하나씩 추가

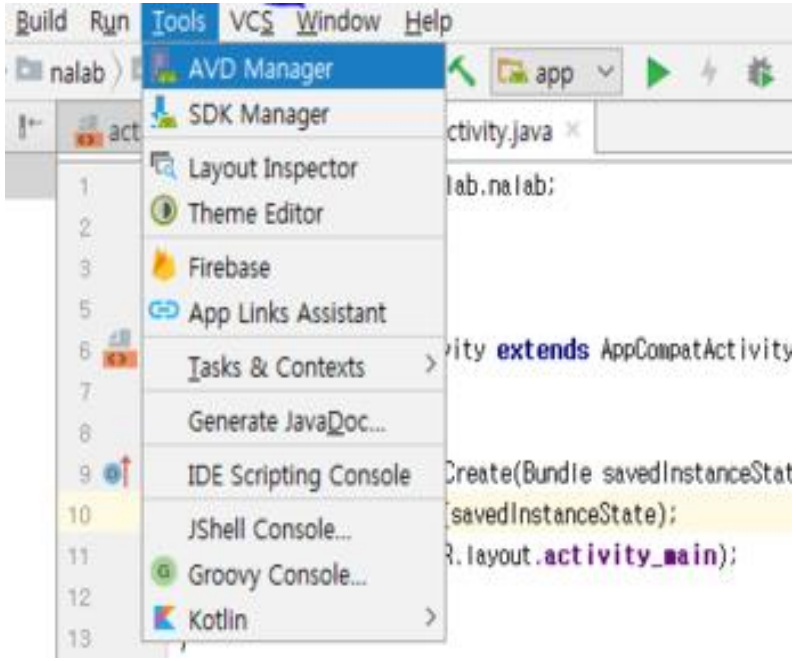


다운로드 확인 화면

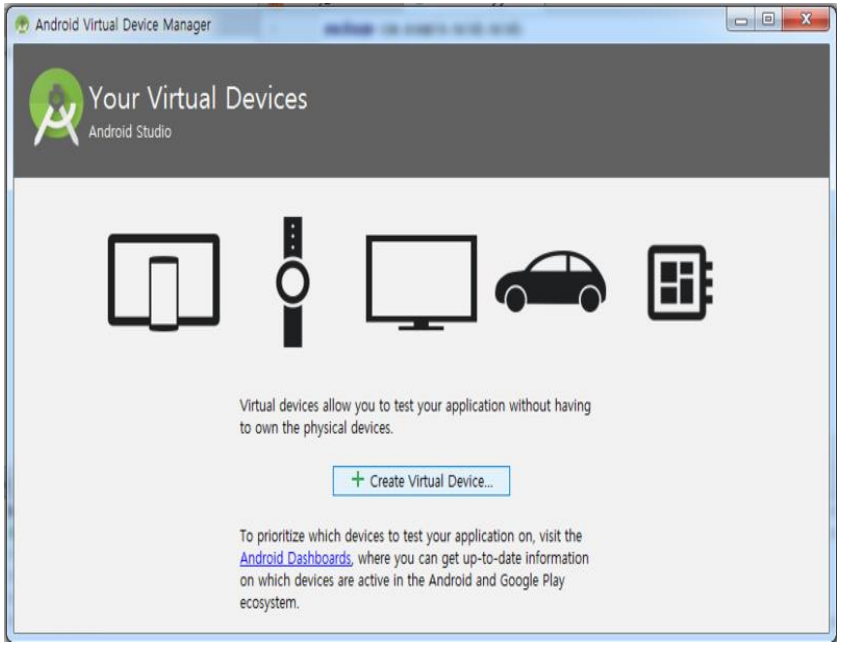
ADK 다운로드 진행 화면

응용 실습

- **AVD 생성 및 실행**
  - Tools 메뉴에서 AVD Manager를 눌러 AVD 생성 창을 염
  - Create Virtual Device를 클릭해서 새로운 에뮬레이터를 생성
  - 하드웨어 선택 화면이 표시되면, 생성할 디바이스의 화면 크기를 선택
  - Nexus 4를 선택하고 Next 버튼



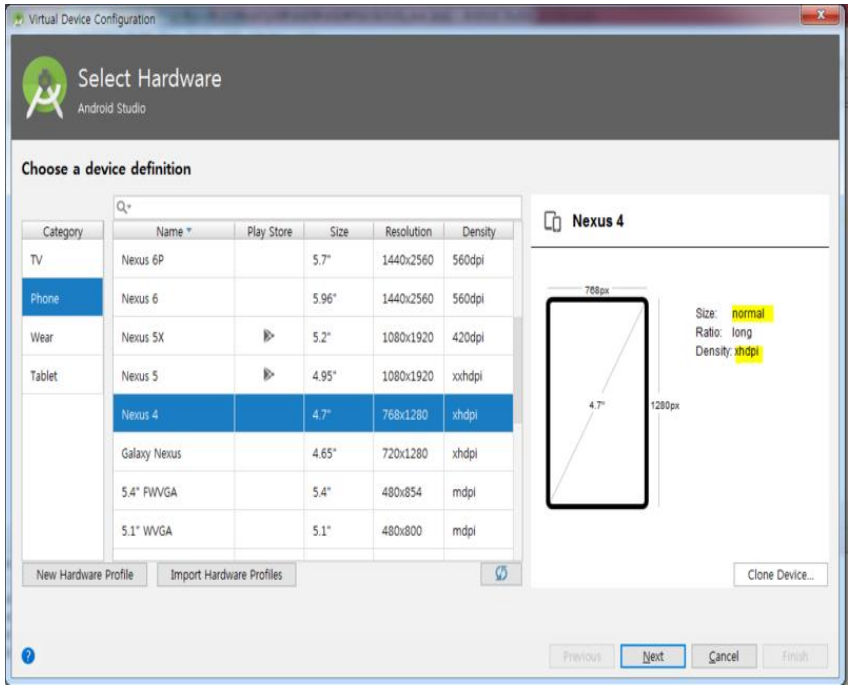
AVD Manager 실행하기



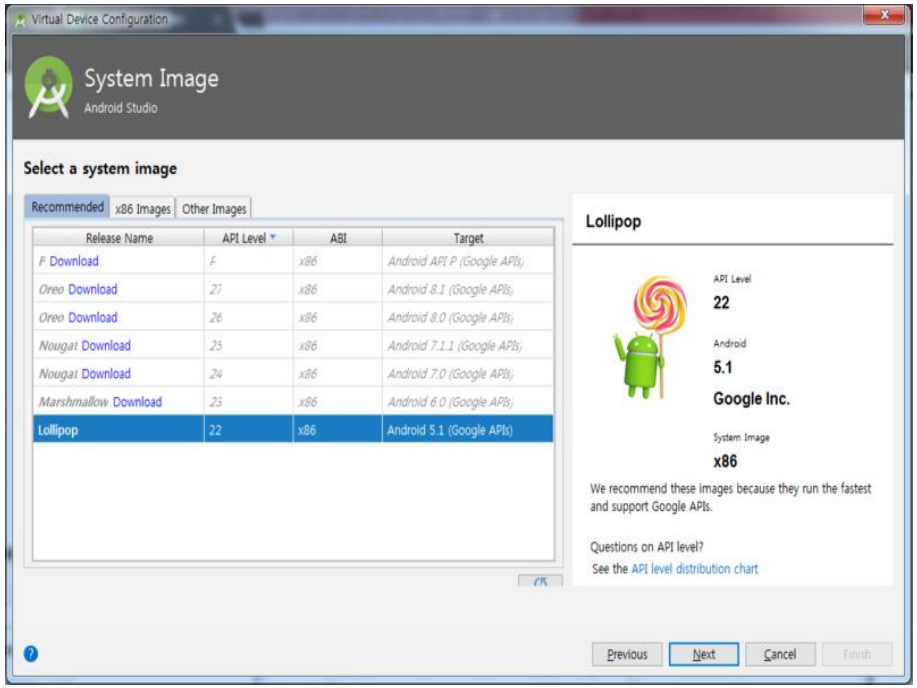
Create Virtual Device 생성 화면

응용 실습

- **AVD 생성 및 실행**
  - 하드웨어 선택 화면이 표시되면, 생성할 디바이스의 화면 크기를 선택
  - Nexus 4를 선택하고 Next 버튼
  - 스템의 이미지를 선택해야 하는데 여기서는 롤리팝을 선택 후 Next 버튼



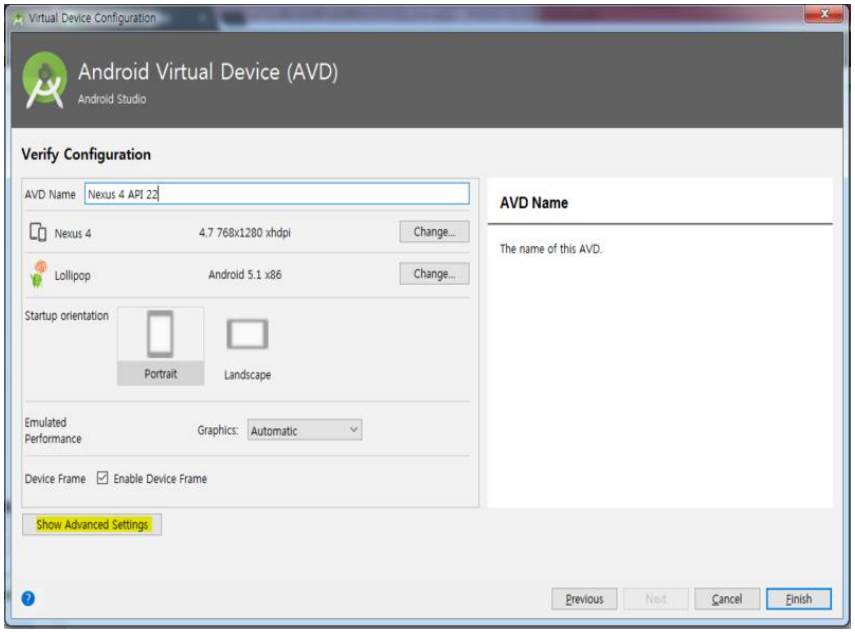
하드웨어 선택 화면



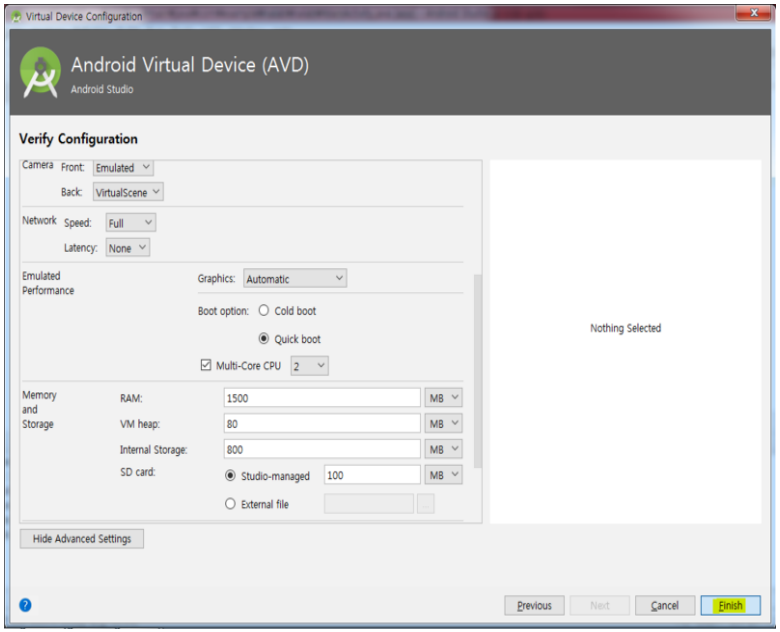
시스템 이미지 선택화면

응용 실습

- **AVD 생성 및 실행**
  - 아래와 같이 AVD 설정 확인 화면이 표시되고, 단말기 기종과 함께 이미지 버전을 확인 한 후 AVD 이름을 입력
    - 영문으로 이름을 기입한 후 Finish 버튼
  - AVD의 이름을 입력한 후 상세 설정을 확인하기 위해 왼쪽 아래에 있는 Show Advanced Setting
    - 고급 설정에서 Camera 에서 Front 와 Back을 모두 None으로 설정하고 Memory and Storage에서 값 조정



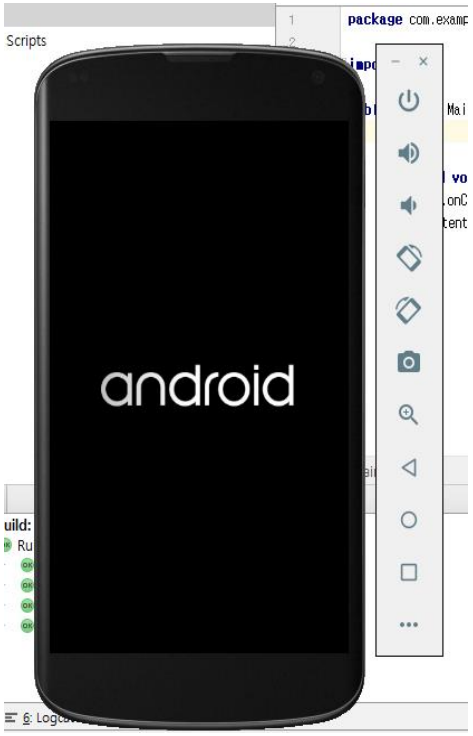
AVD 이름 지정



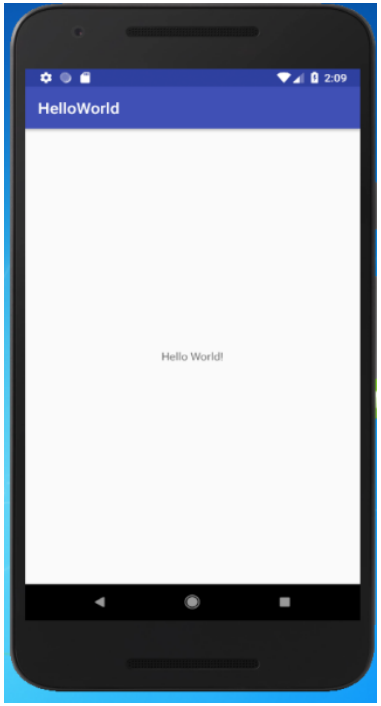
디바이스 고급 설정

응용 실습

- **AVD 생성 및 실행**
  - AVD가 목록에 추가됐고, 이제 오른쪽 화살표 버튼을 클릭하면 AVD가 실행
  - 실제 스마트폰이 아니기 때문에 센서 등이 작동하지 않는데 상세 설정을 통해 가상의 입력 신호를 만들어 줄 수 있는 기능이 있음
  - 기본으로 생성된 프로젝트의 소스코드를 빌드하고 실행
  - AVD를 한번 실행 시키고 나면 Android Studio를 종료하더라도 AVD를 함께 종료시킬 필요가 없음



AVD 실행 화면



디바이스 고급 설정화면

- Manifest.xml

```
<!-- XML 에서 주석은 이렇게 넣는다. -->
<!-- 앞쪽에 첫 태그는 XML 파일이며 한글을 지원한다는 뜻이다. -->
<?xml version="1.0" encoding="utf-8"?>
<!-- 매니페스트 파일은 수하물 목록이란 사전적 의미를 가진다. -->
<manifest xmlns:android="안드로이드 APK 파일의 스키마 경로는 생략합니다."
    package="com.iot.helloworld">

        <!-- XML 에서 주석은 이렇게 넣는다. -->
    <application
        <!-- 데이터 자동 백업을 허용하도록 설정한다. -->
        android:allowBackup="true"
        <!-- ic_launcher는 이미지 아이콘인데 기본 아이콘을 사용한다. -->
        android:icon="@mipmap/ic_launcher"
        <!-- 앱의 이름은 string.xml에 app_name 변수로 생성되어 있다. -->
        android:label="@string/app_name"
        <!-- 안드로이드 아이콘 중에서 모서리가 둥근 형 아이콘을 지정한다. -->
```

- Manifest.xml

```
            android:roundIcon="@mipmap/ic_launcher_round"
            <!-- RTL을 지원하도록 설정한다. (기본값) -->
            android:supportsRtl="true"

<!-- 앱의 테마를 이용하면 기본 글꼴이나 배경색을 지정할 수 있어 편리하다. -->
            android:theme="@style/AppTheme">
            <!-- Activity는 반드시 Manifest 파일에 등록해 주어야 한다. -->
            <activity android:name=".MainActivity">
                <intent-filter>
                <!-- 앱 실행 시 이 액티비티가 가장 먼저 실행됨을 의미한다. -->
                    <action android:name="android.intent.action.MAIN" />
                    <category android:name="android.intent.category.LAUNCHER" />

                </intent-filter>
            </activity>
        </application>
</manifest>
```

- MainActivity.java

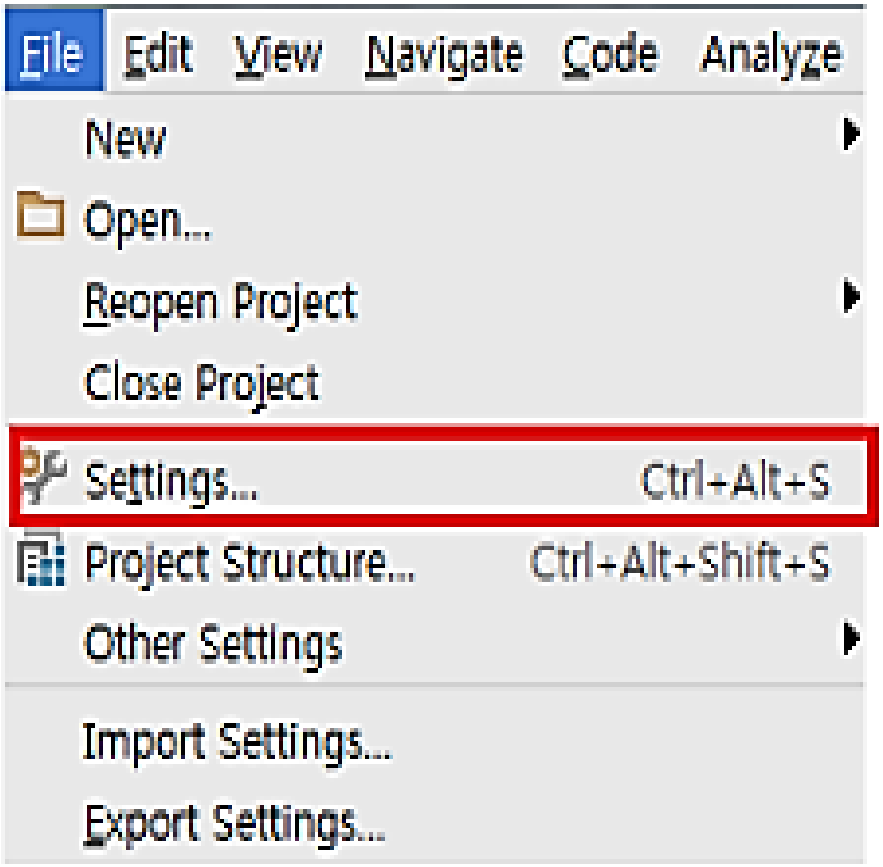
```
package com.iot.helloworld;                                    // 현재 패키지

import android.support.v7.app.AppCompatActivity; // 액티비티 부모 클래스
                                                               // 앱의 호환성을 위해 추가
import android.os.Bundle;                                      // 액티비티 생성 번들
                                                   // 액티비티 데이터 번들 사용 목적

public class MainActivity extends AppCompatActivity {      // 메인 화면
                                                            // 기본 화면 상속 받기
    // Override는 부모 클래스의 기능을 재정의 한다는 의미
    @Override // 부모 메소드 재정의
    // onCreate 메소드는 화면이 만들어 질때 최초로 호출 됨
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);      // 부모 생성자 호출
                                                 // 부모의 onCreate() 호출
        setContentView(R.layout.activity_main); // 메인 화면 표시
                                                 // activity_main XML을 화면 출력
    }
}
```

테마 바꿔보기

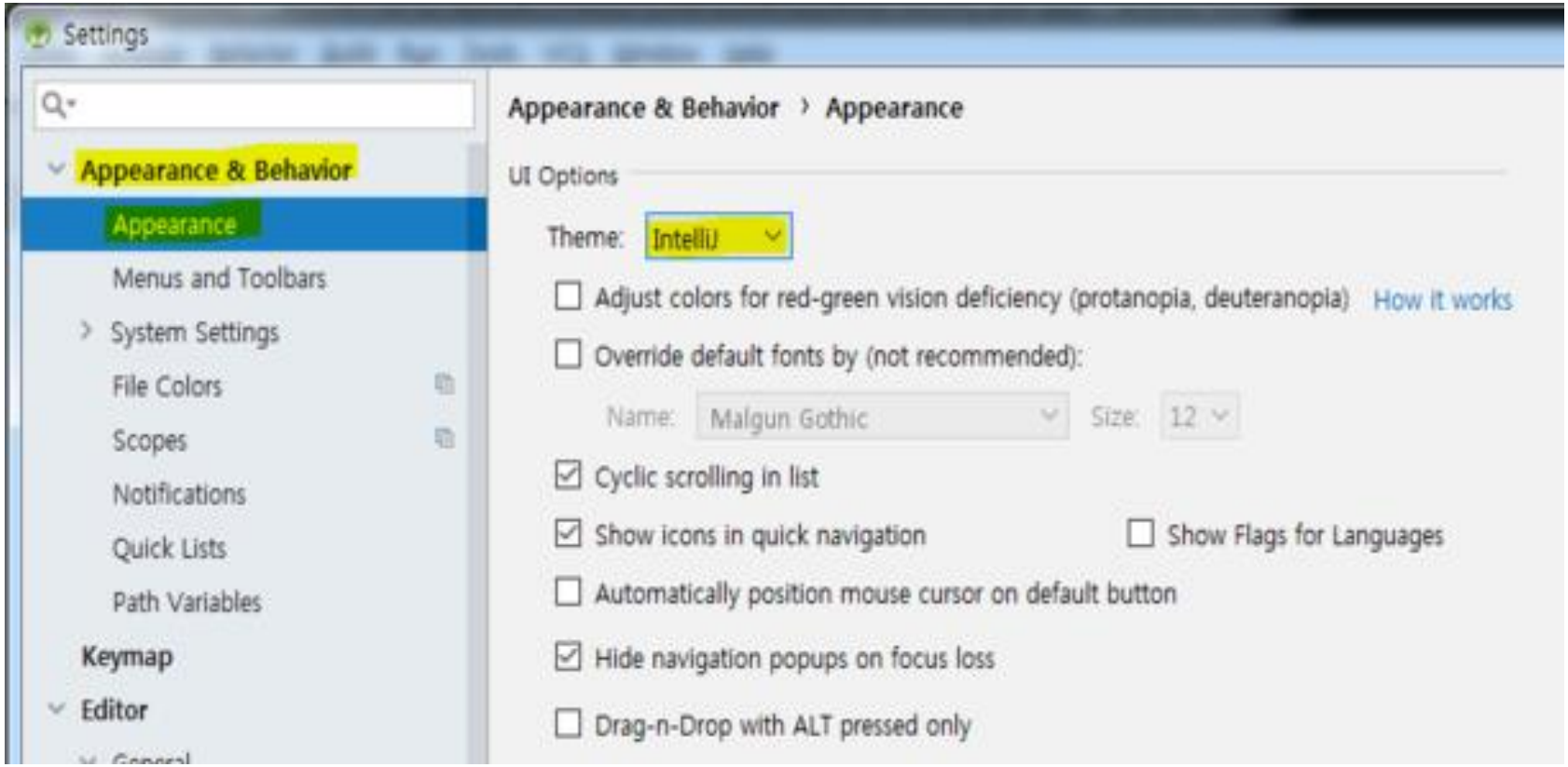
- File > Settings...를 선택



Setting 창 열기

테마 바꿔보기

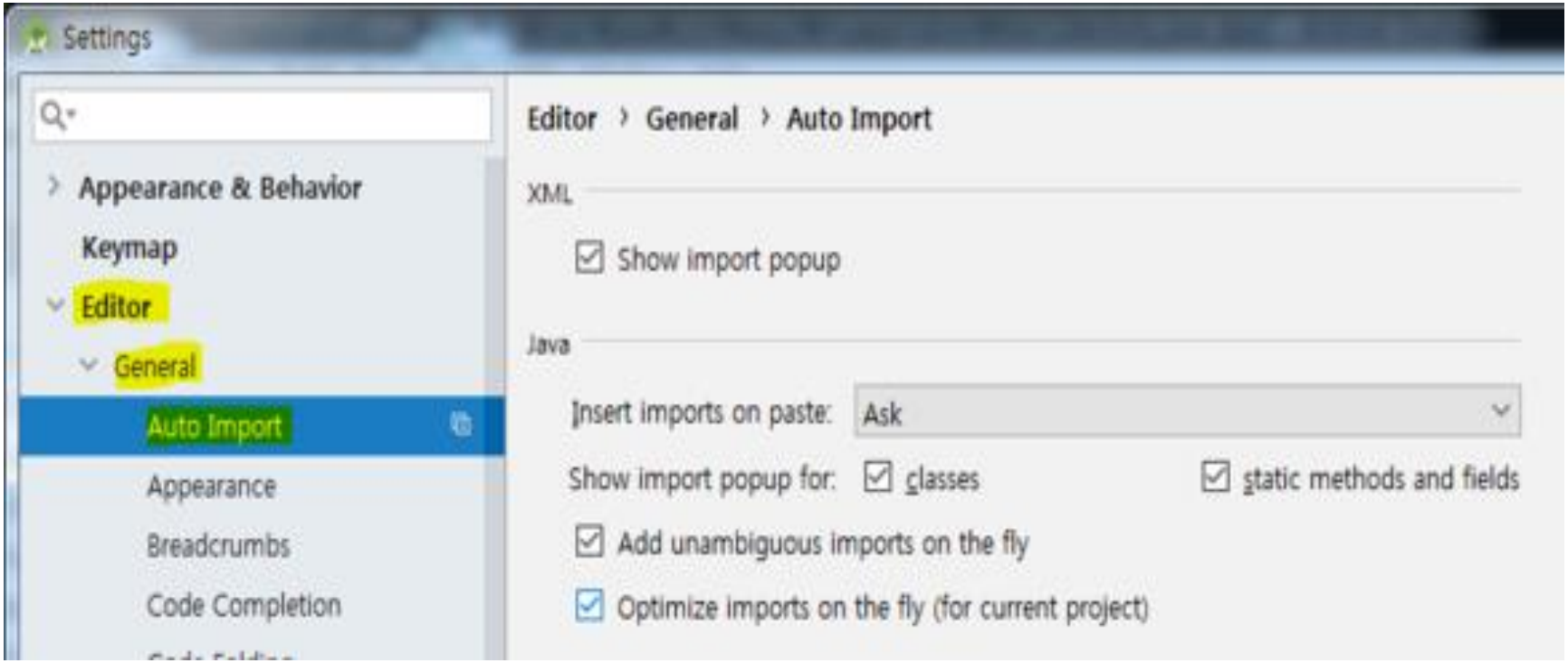
- Appearance & Behavior의 아래에 있는 Appearance로 들어 가면 기본 테마 변경



테마 바꾸기 기능 설정

줄 번호 표시하기

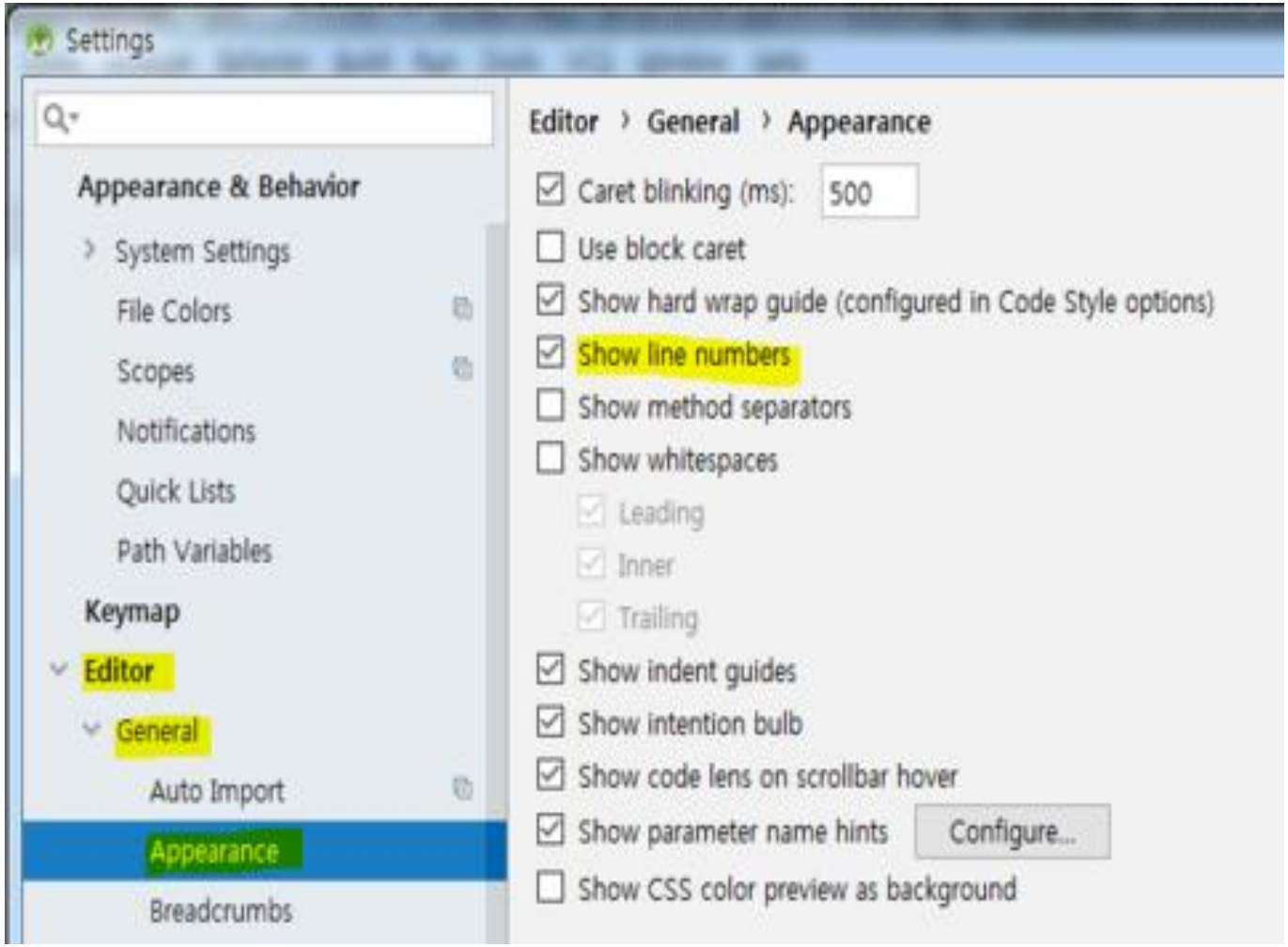
- 자동으로 import
  - Editor > General > Auto Import 에서 모든 항목에 체크



Auto Import 기능 켜기

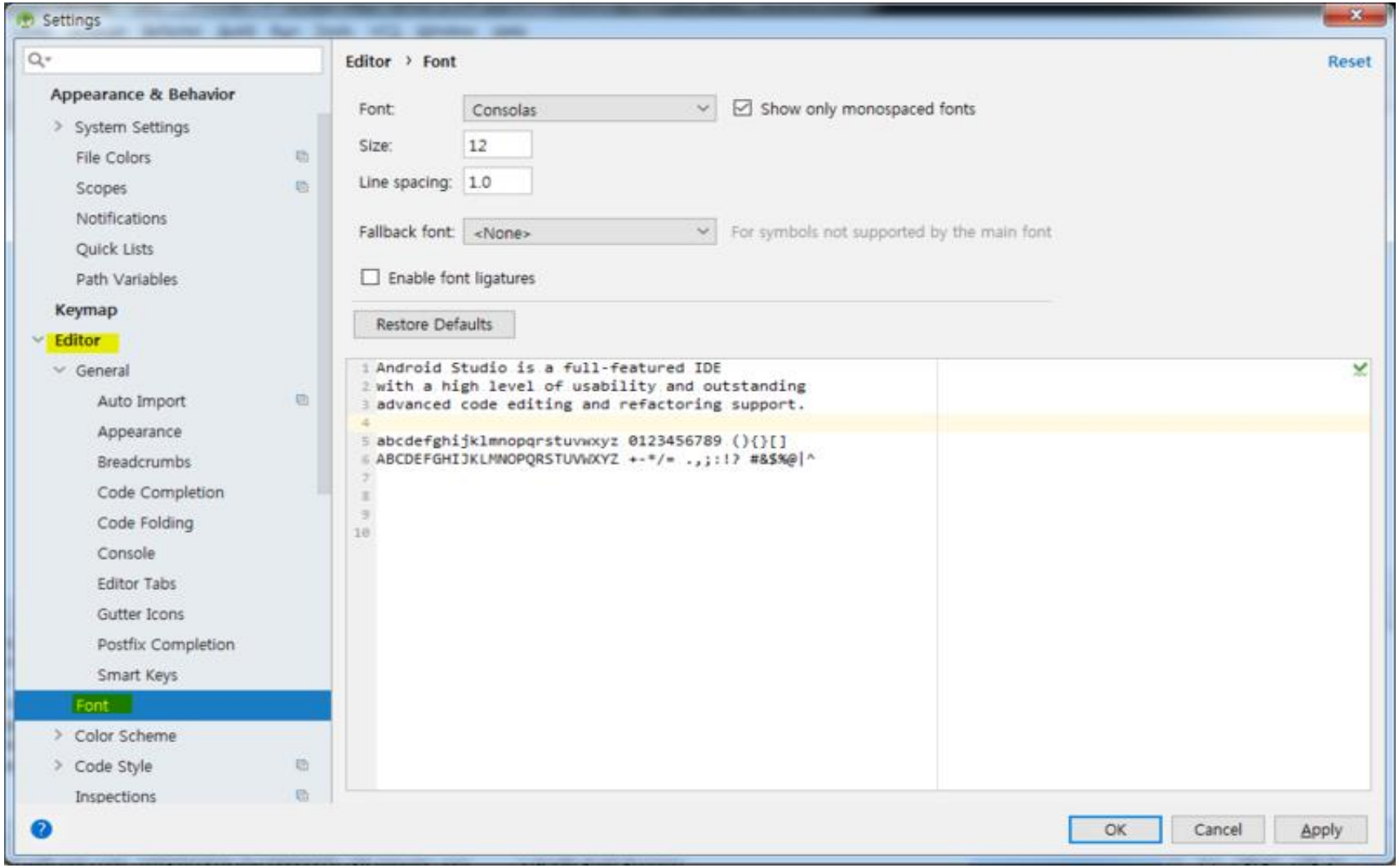
줄 번호 표시하기

- 라인 넘버 생성
  - Editor > General > Appearance 에서는 코드에 라인넘버가 나오도록 체크



폰트 변경하기

- Editor 아래의 Font 항목 클릭
  - Font에서는 폰트나 글씨 크기 변경



Gradle 설정하기

OneKick

- 컴파일을 수행할 때 어떤 프로젝트를 대상으로 할지 지정해야 함
  - 이러한 설정이 귀찮다면 좌측의 프로젝트 아이콘에 마우스 오른쪽 버튼을 눌러 run as > Android Application으로 실행 시키면 자동으로 실행