

HBE-SM5-S4210

시스템 개요 및 개발환경 구축

2012 내장형 시스템 설계

HBE-SM5-S4210 시스템 개요

HBE-SM5-S4210 시스템 개요

- ▶ 안드로이드/리눅스 및 펌웨어 실습용 임베디드 장비
- ▶ S5PC210 어플리케이션 프로세서 채용
- ▶ 7인치 LCD와 정전 터치 적용
- ▶ 이동성을 위한 배터리 모듈 제공
- ▶ HSDPA 모뎀 추가 장착 가능(option)

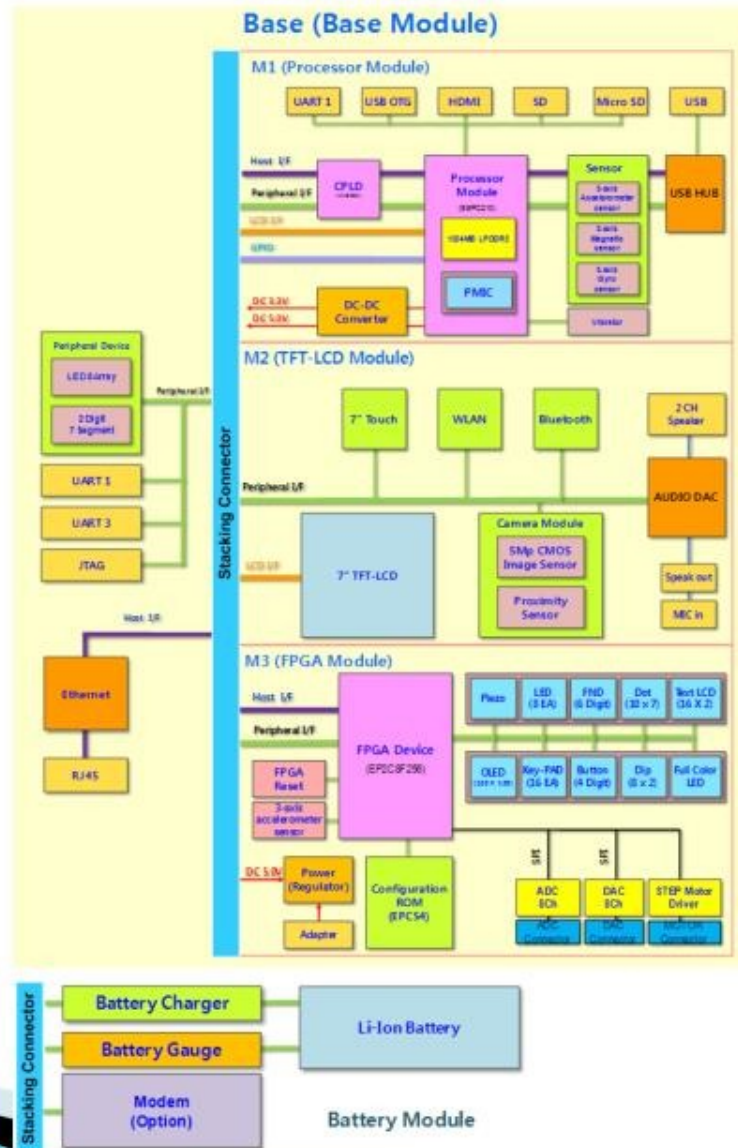


HBE-SM5-S4210 시스템 구성

▶ 장비 사양

Processor	Samsung S5PC210 (ARM Cortex A9 Dual Core)	Audio	Built-in stereo speaker, 3.5mm Speaker jack, 3.5mm Mic jack
Memory	1GB LPDDR2 (integrated on Processor)	I/O	Debug port, USB 2.0 port, Serial Port, Ethernet Port
Storage	1 Slot micro SD (T-flash, 8GB) 1 Slot SD/MMC	Camera	5Mega Pixel CMOS Camera
LCD	7" 800x480 Resolution	Sensor	30axis Accelerometer, 3-axis Magnetic field, 3-axis Gyroscope, Vibration moter
Touch	7" 정전식 터치, 최대 10포인트 멀티터치, 4포인트 정전버튼	Peripherals	Text LCD, 6-digit 7Segment, 8-bit LED, Dotmatrix, 16-bit Dip Switch, Piezo, Full color LED etc.
Connectivity	10/100 Base-T Ethernet, Wifi(802.11 b/g/n), Bluetooth (2.1+EDR class 1)	Power	5V 3A Adapter, Lithium Polymer Battery (3.7V / 3150mAh)
Video	1080p HDMI out	System Software	Android 4.0.4 (Ice Cream Sandwich) Linux Kernel 3.0.1.5

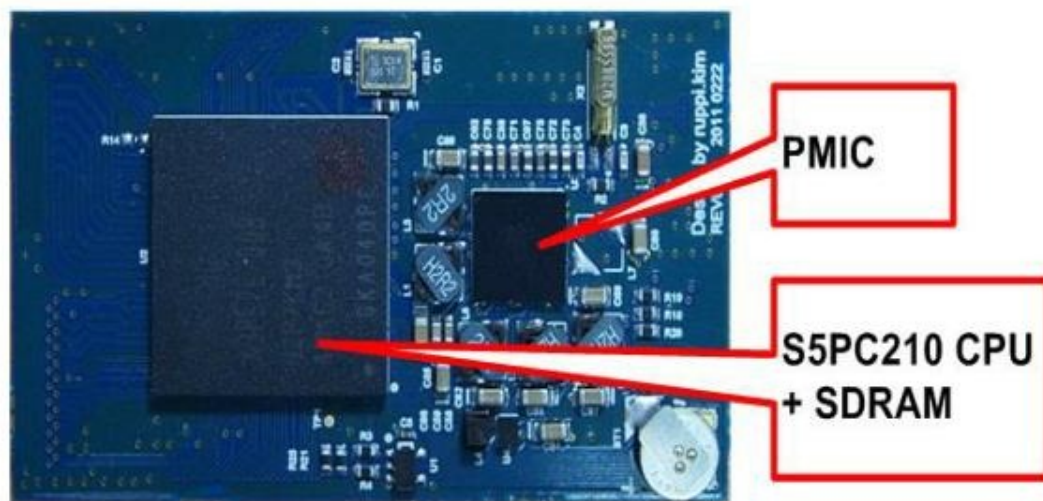
HBE-SM5-S4210 시스템 구성



HBE-SM5-S4210 시스템 구성

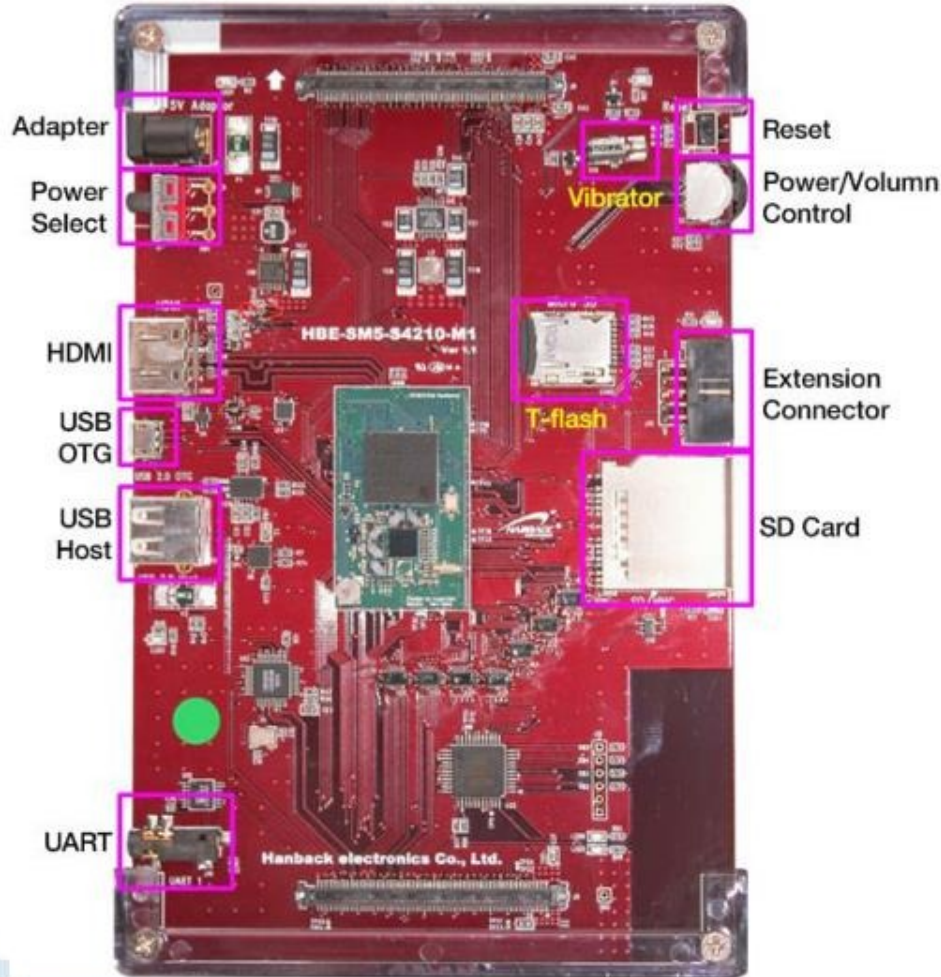
▶ CPU Module

- 삼성 S5PC210(ARM Cortex-A9 Dual Core + 1GB LPDDR2)
 - 프로세서와 메모리가 결합된 단일칩 패키지
- PMIC : 7 DC/CD Converters 21 LDOs



HBE-SM5-S4210 시스템 구성

▶ HBE-SM5-S4210 M1 Module



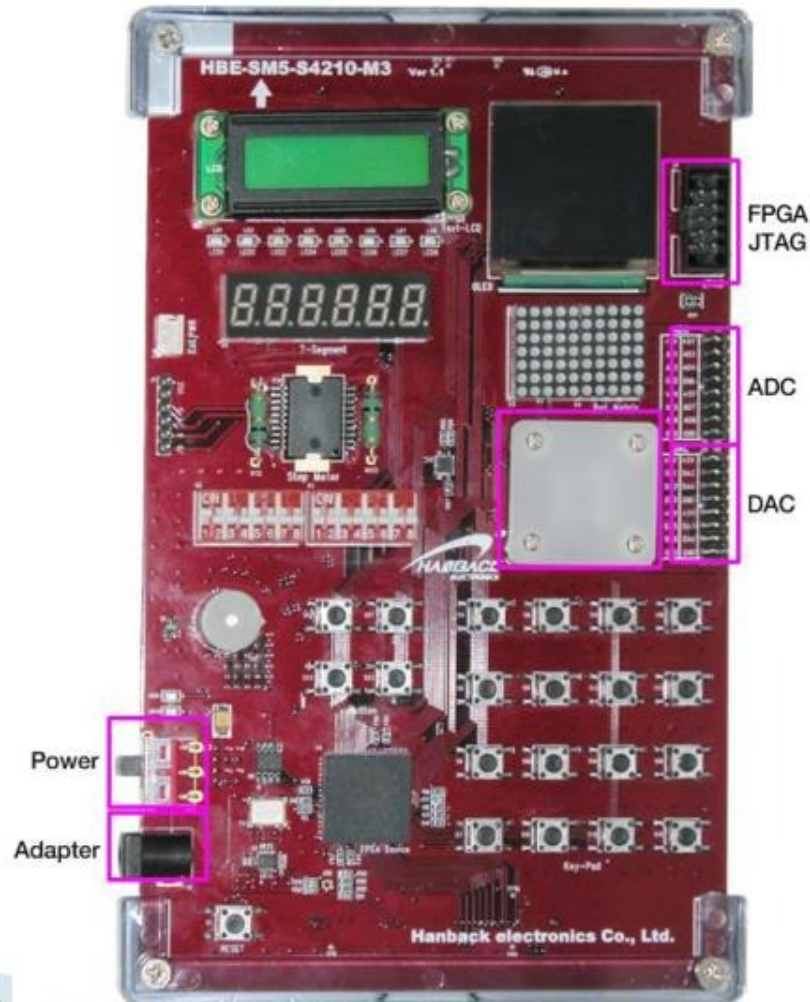
HBE-SM5-S4210 시스템 구성

- ▶ HBE-SM5-S4210 M2(LCD) Module + Camera Mod.



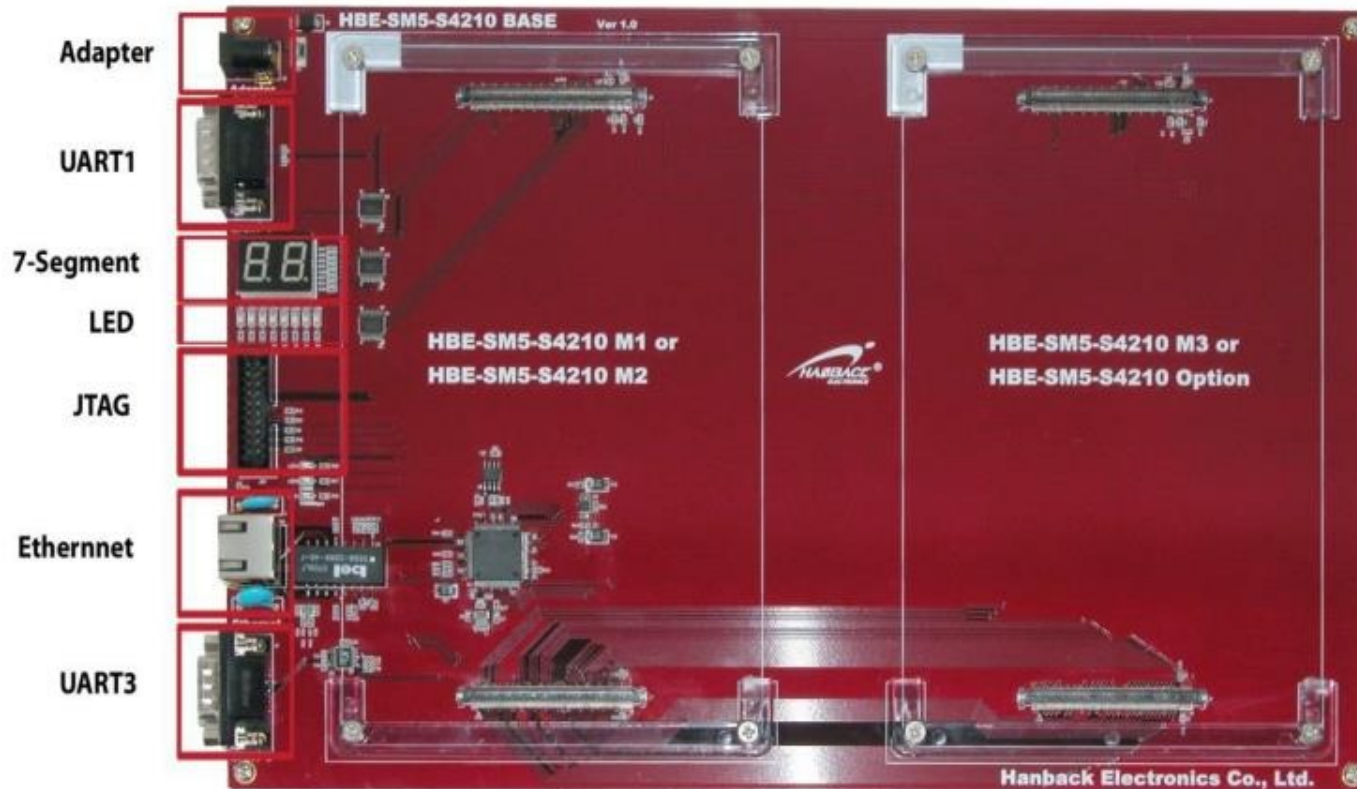
HBE-SM5-S4210 시스템 구성

▶ HBE-SM5-S4210 M3(FPGA) Module



HBE-SM5-S4210 시스템 구성

▶ HBE-SM5-S4210 Base Board



HBE-SM5-S4210 Battery Module



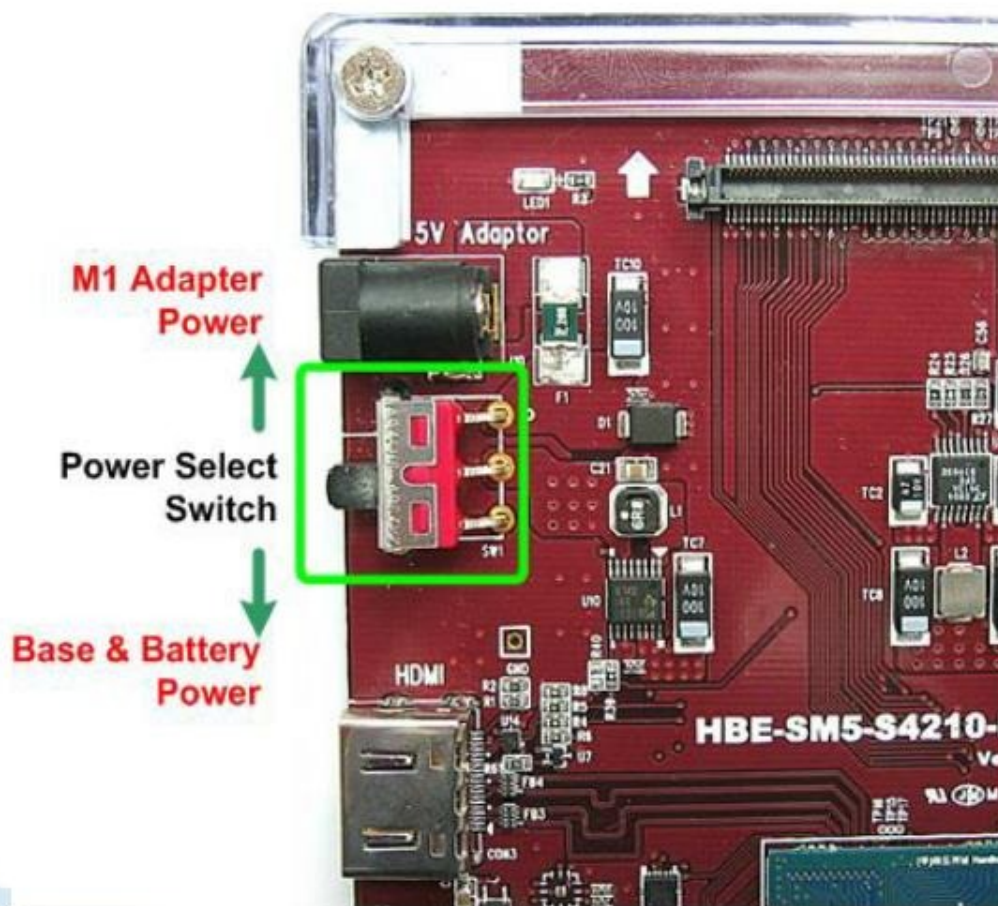
HBE-SM5-S4210 장비 사용법

▶ 케이블 연결



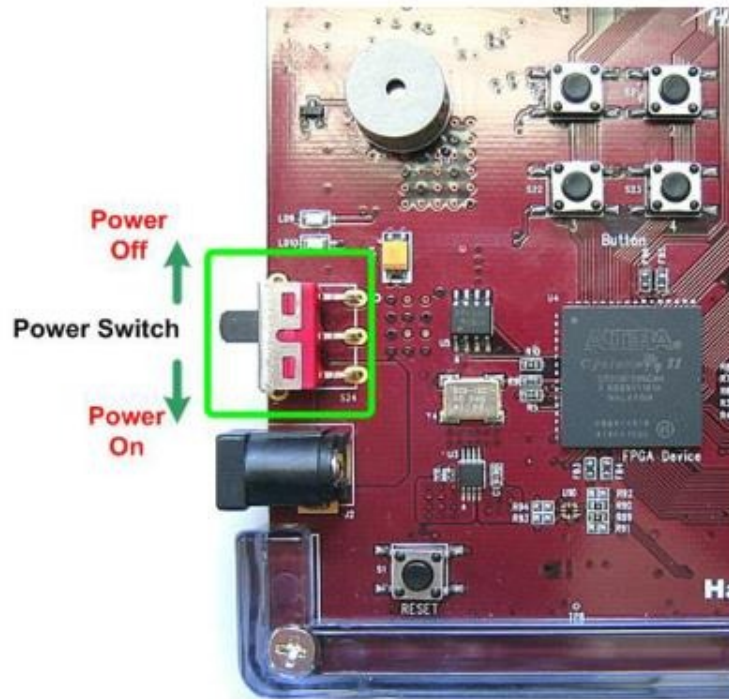
HBE-SM5-S4210 장비 사용법

▶ M1 모듈 전원 선택 스위치



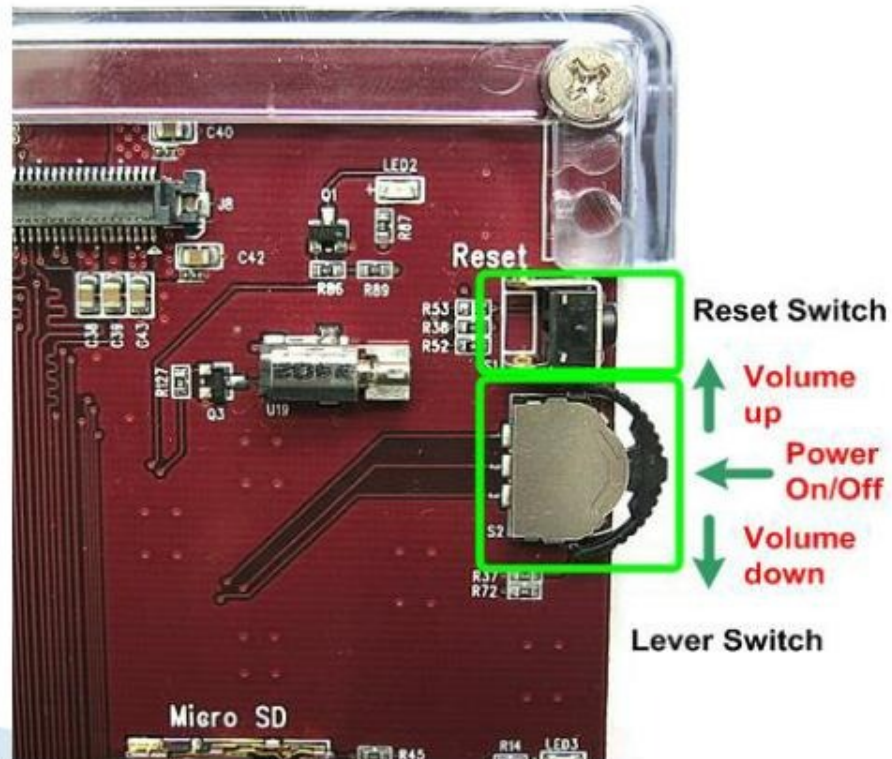
HBE-SM5-S4210 장비 사용법

- ▶ M3 모듈 전원 선택 스위치
 - Base 모듈 연결시 자동으로 전원이 켜짐
 - FPGA 실습을 위한 M3 모듈 단독으로 사용시 Power On/Off 사용



HBE-SM5-S4210 장비 사용법

- ▶ M1 모듈의 스위치 ; Lever 타입
 - 전원 On/Off ; 길게 3~5초 정도 눌러줘야 함
 - Suspend/Resume ; 짧게 눌러줌
 - 볼륨 조절 ; 위 아래로 당김



Handheld 구성에서의 장비 사용법

▶ M1 + M2 + 배터리 모듈



USB OTG
Cable

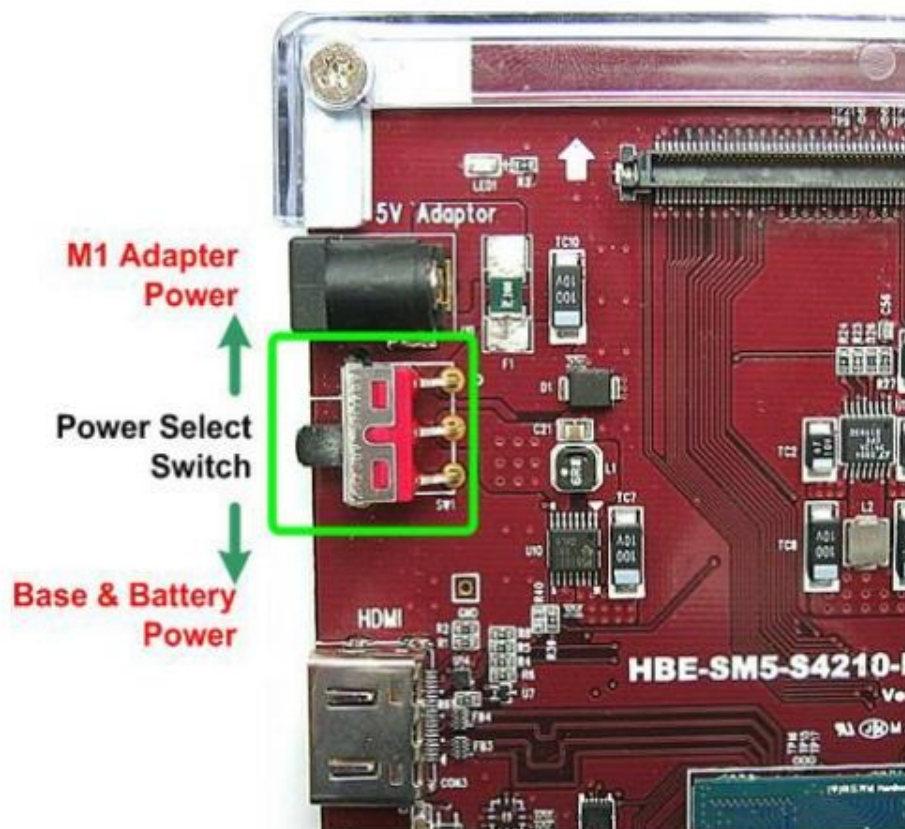


RS232
Cable



Handheld 구성에서의 장비 사용법

- ▶ 배터리는 완충시 대략 2시간 정도 사용 가능



정전 Touch 및 버튼 사용



안드로이드 부팅

▶ 부팅 및 리셋 스위치

- Power On/Off Switch
 - 레버 스위치를 길게 3~5초 정도 눌러서 동작
 - 짧은 누름은 Suspend/Wakeup으로 동작
- Reset Button
 - PowerOn 상태에서 시스템 재시작

▶ 부팅 장치 ; microSD 카드 사용

- 카드 내에는 (부트로더, 리눅스커널, 안드로이드)가 포함

안드로이드 응용 프로그램 개발환경 (윈도우)

안드로이드 응용 프로그램 개발환경

- ▶ 1. Java Development Kit 설치
- ▶ 2. Android SDK & eclipse 설치
- ▶ 3. Android App 작성 및 실행하기
- ▶ 4. NDK를 이용한 JNI 개발 환경 구축

다운로드 링크

JDK: <http://cslab.cau.ac.kr/tempfile/jdk-7u6-windows-i586.exe>
통합 개발환경: <http://cslab.cau.ac.kr/tempfile/Android-devenv.exe>

JDK 다운로드 및 설치

- ▶ 최신 Java Development Kit 다운로드 사이트
 - <http://www.oracle.com/technetwork/java/javase/downloads/index.html>



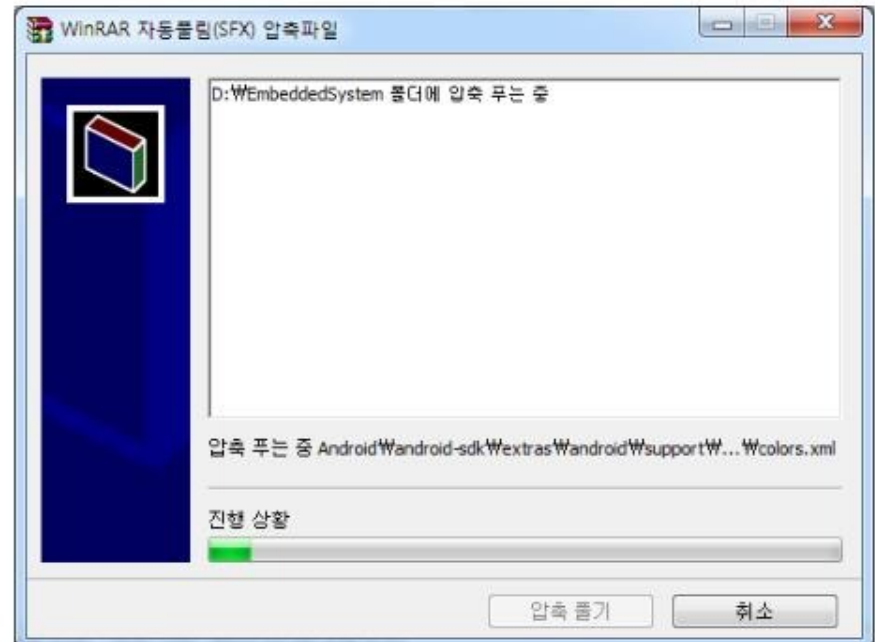
JDK 다운로드 및 설치



Android SDK & Eclipse 설치

▶ 개발환경설치

- 압축된 개발 환경 파일 Android-devenv.exe 파일을 실행하여 다음처럼 설치하도록 한다.
 - ① "대상 폴더"를 변경한다. 여기서는 "D:/EmbeddedSystem"로 변경
 - ② "압축 풀기" 버튼을 클릭하여 압축을 해제한다.
 - ③ 압축 해제가 곧 설치이므로 압축 해제와 동시에 설치가 완료된다.



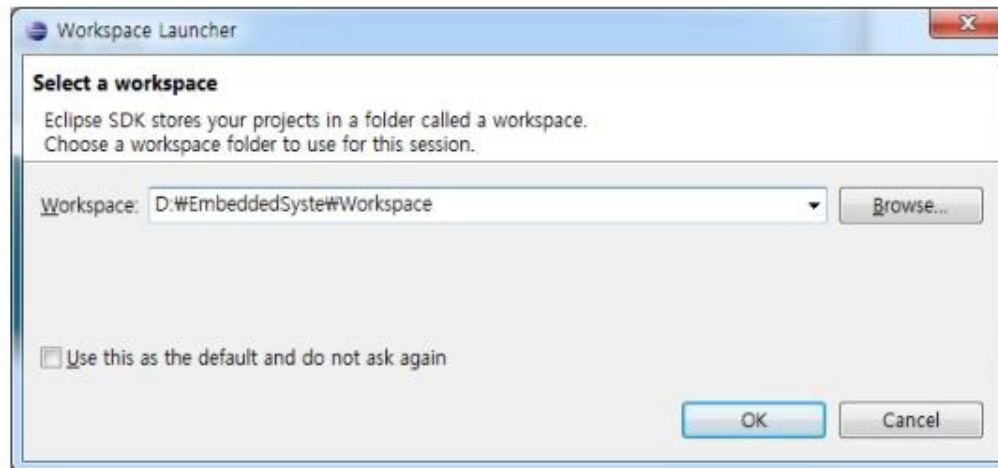
Android SDK & Eclipse 설치

▶ Eclipse 실행

- D:\EmbeddedSystem\Android\eclipse\eclipse.exe

▶ Workspace 설정

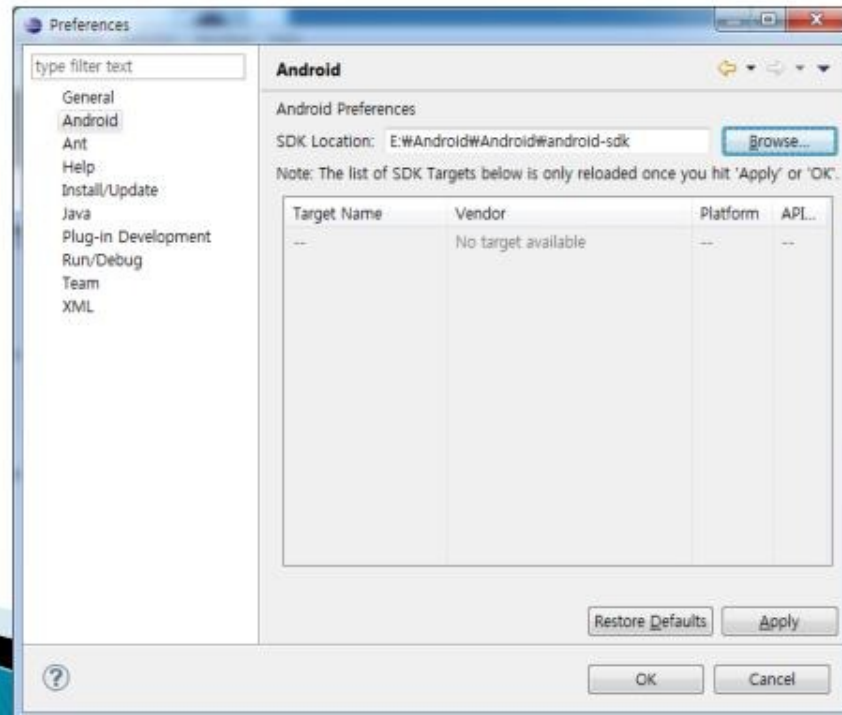
- D:\EmbeddedSystem\Workspace



Android SDK & eclipse 설치

▶ SDK 위치 설정

- ① Window -> Preferences를 실행한다.
- ② 왼쪽 메뉴에서 "Android" 항목을 선택한다.
- ③ "SDK Location"을 설치한 위치로 변경해 준다.
 - D:\EmbeddedSystem/Android/android-sdk
- ④ 변경 후에는 오른쪽 아래의 "Apply" 버튼을 클릭하여 변경 내용을 반영한다.



Android App 작성 및 실행하기

▶ 안드로이드 App

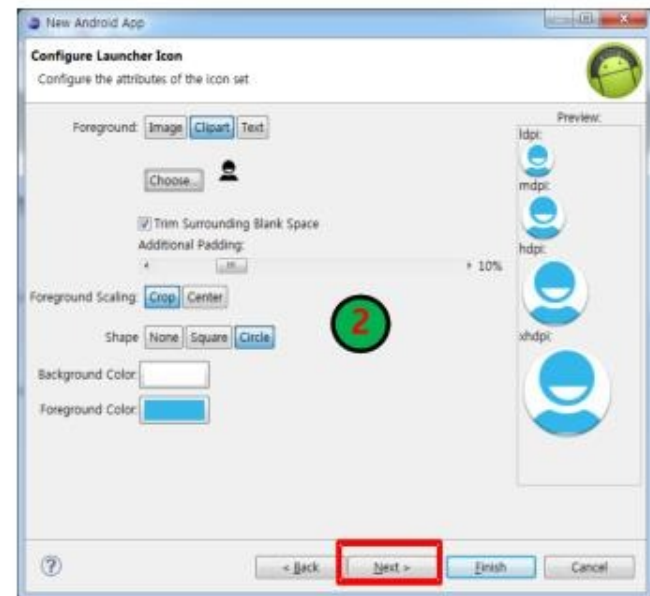
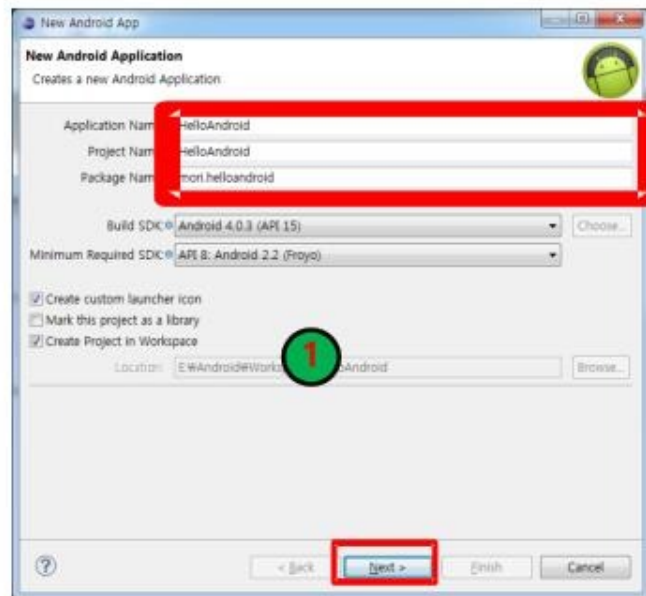
- 안드로이드 App은 JAVA 프로그래밍 언어로 작성된다.
- App에 필요한 데이터와 리소스 파일들과 함께 컴파일된 JAVA 코드는 하나의 압축파일로 묶여서 Android Package가 된다.
- Android Package는 ".apk"라는 확장자를 갖는다. 이 파일은 모바일 장치를 위해 App을 배포하여 설치하는 수단이다.
- 하나의 ".apk" 파일안의 모든 코드는 하나의 App로 간주된다.

Android App 작성 및 실행하기

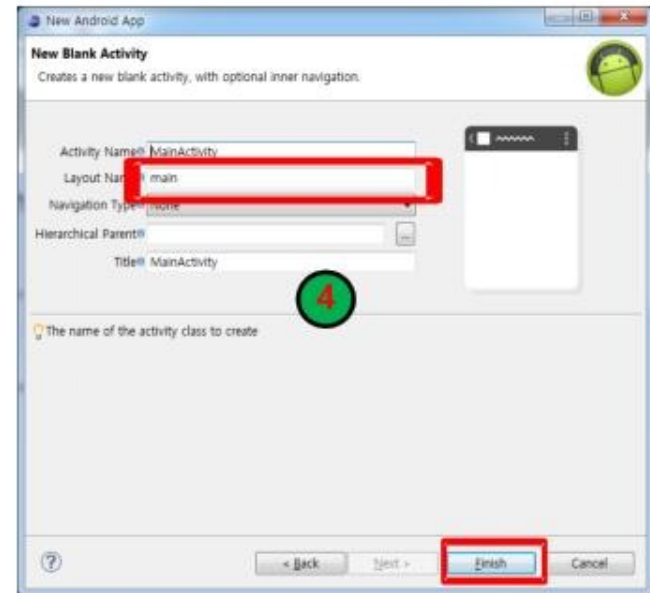
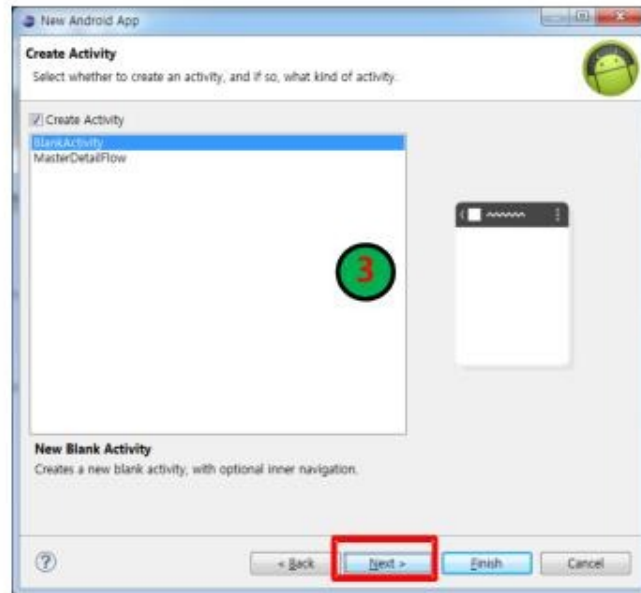
▶ App 구조

- 안드로이드 App은 Activity, Broadcast Receiver, Service, Content Provider 이 4가지로 구성되어 있다.
 - 모든 App이 이 4가지 모두 필요로 하지 않으며, 이들의 조합으로 이루어진다.
- AndroidManifest.xml
 - 애플리케이션에 대한 정보 기록하는 파일이다.
 - 사용할 구성요소 매니페스트 파일에 추가해야 하며, 이를 통해 애플리케이션을 구성하고 있는 정보를 시스템에 알려준다.

Hello World 예제 생성



Hello World 예제 생성



AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="mori.helloandroid"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

AndroidManifest.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
```

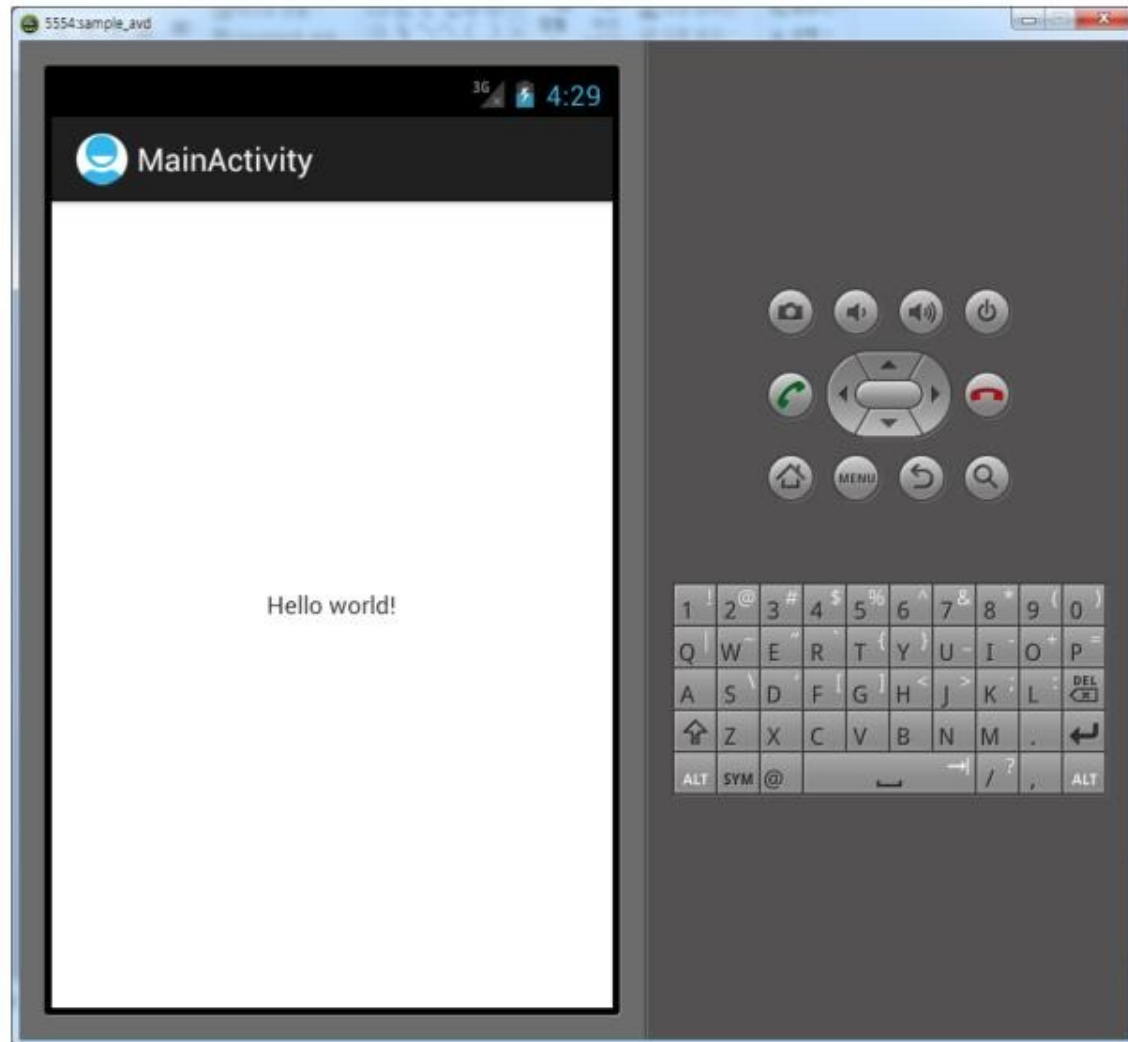
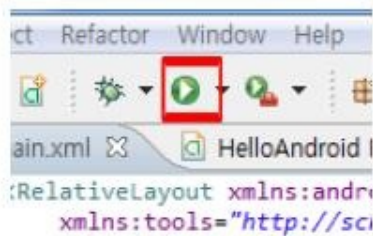
<TextView

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:text="@string/hello_world"
    tools:context=".MainActivity" />
```

```
</RelativeLayout>
```



Hello World 결과화면



NDK를 이용한 JNI개발 환경 구축

▶ NDK(Native Development Kit)

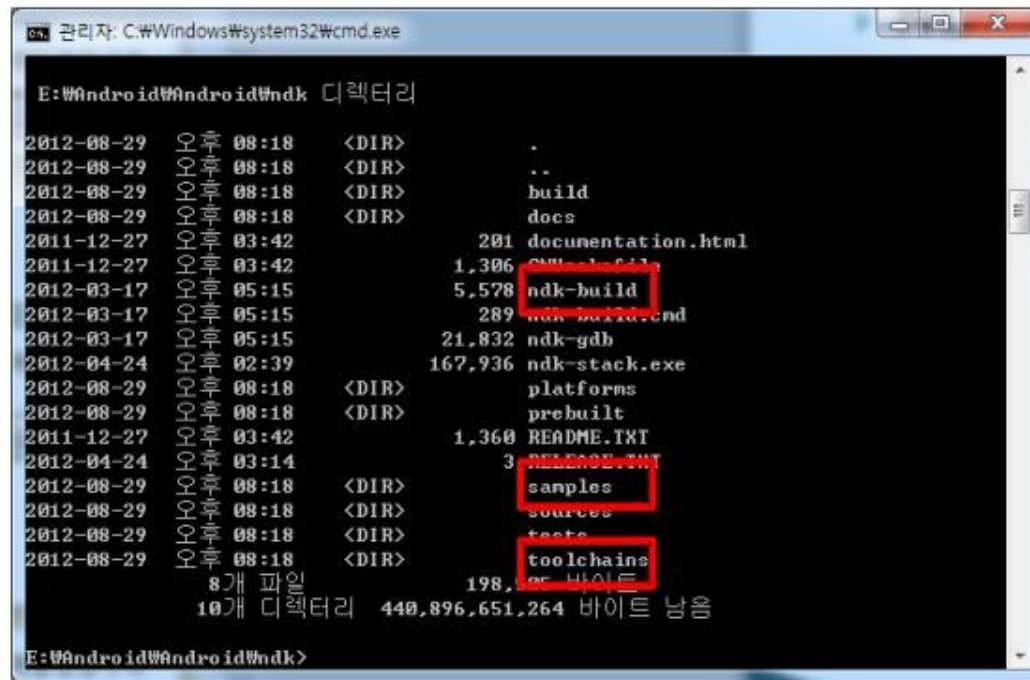
- Java 언어에는 C/C++로 작성된 라이브러리를 사용할 수 있는 방법이 존재한다. 이것을 JNI(Java Native Interface)라고 부른다.
 - NDK는 안드로이드용 JNI를 개발할 수 있도록 제공되는 개발 도구라고 생각하면 된다.
- 안드로이드 자체도 하드웨어를 제어하는 부분들은 모두 C/C++로 구현되어 있다. 이러한 라이브러리들을 안드로이드 내부의 프레임워크에서 사용할 수 있도록 하는 JNI가 존재한다.
- NDK는 사용자 프로그램에서도 이와 같은 구조를 사용할 수 있도록 제공하는 개발도구이다.

NDK를 이용한 JNI개발 환경 구축

▶ NDK 개발환경

- NDK는 현재 Windows/Linux/MacOS 등을 지원한다.
 - Windows용을 사용하기 위해서는 Windows에서 리눅스 환경을 사용하게 해주는 cygwin 환경을 구축해야 한다.
 - D:\EmbeddedSystem\Android\ndk\

NDK를 이용한 JNI개발 환경 구축



```
관리자: C:\Windows\system32\cmd.exe

E:\Android\Android\ndk 디렉터리
2012-08-29 오후 08:18 <DIR> .
2012-08-29 오후 08:18 <DIR> ..
2012-08-29 오후 08:18 <DIR> build
2012-08-29 오후 08:18 <DIR> docs
2011-12-27 오후 03:42 201 documentation.html
2011-12-27 오후 03:42 1,306 ndk-build
2012-03-17 오후 05:15 5,578 ndk-build.cmd
2012-03-17 오후 05:15 289 ndk-gdb
2012-03-17 오후 05:15 21,832 ndk-stack.exe
2012-04-24 오후 02:39 167,936 platforms
2012-08-29 오후 08:18 <DIR> prebuilt
2012-08-29 오후 08:18 <DIR> README.TXT
2011-12-27 오후 03:42 1,360 RELEASE.TXT
2012-04-24 오후 03:14 3 samples
2012-08-29 오후 08:18 <DIR> sources
2012-08-29 오후 08:18 <DIR> tests
2012-08-29 오후 08:18 <DIR> toolchains
8개 파일 198,400 바이트
10개 디렉터리 440,896,651,264 바이트 남음

E:\Android\Android\ndk>
```

- ▶ ndk-build: 빌드 스크립트
- ▶ Samples: NDK로 빌드할 안드로이드용 App 샘플
- ▶ toolchains: 빌드하는데 사용되는 컴파일러 및 라이브러리가 포함

NDK를 이용한 JNI개발 환경 구축

▶ JNI 라이브러리 빌드 방법

- 빌드할 App이 있는 폴더에서 NDK 루트 폴더의 ndk-build 스크립트를 실행
 - **'samples/hello-jni'** 예제 프로그램을 빌드
 - > cd samples/hello-jni
 - > /EmbeddedSystem/Android/ndk/ndk-build
 - **'libs/armeabi/libhello-jni.so'** 생성 확인
 - > cd libs/armeabi
 - > dir

NDK를 이용한 JNI 개발 환경 구축

▶ 안드로이드앱에서 JNI 라이브러리 사용하기

- NDK에서 빌드한 라이브러리는 안드로이드 앱에서 사용할 수 있다.
- JNI 라이브러리 형식
 - JNI 라이브러리의 함수 이름에는 규칙이 있다.
 - 이러한 규칙을 지키지 않으면 안드로이드 앱에서 함수를 사용할 수가 없다. 다음 그림을 참고한다.

```
jstring Java_com_example_hellojni_HelloJni_stringFromJNI( JNIEnv* env, jobject thiz )
```

1

2

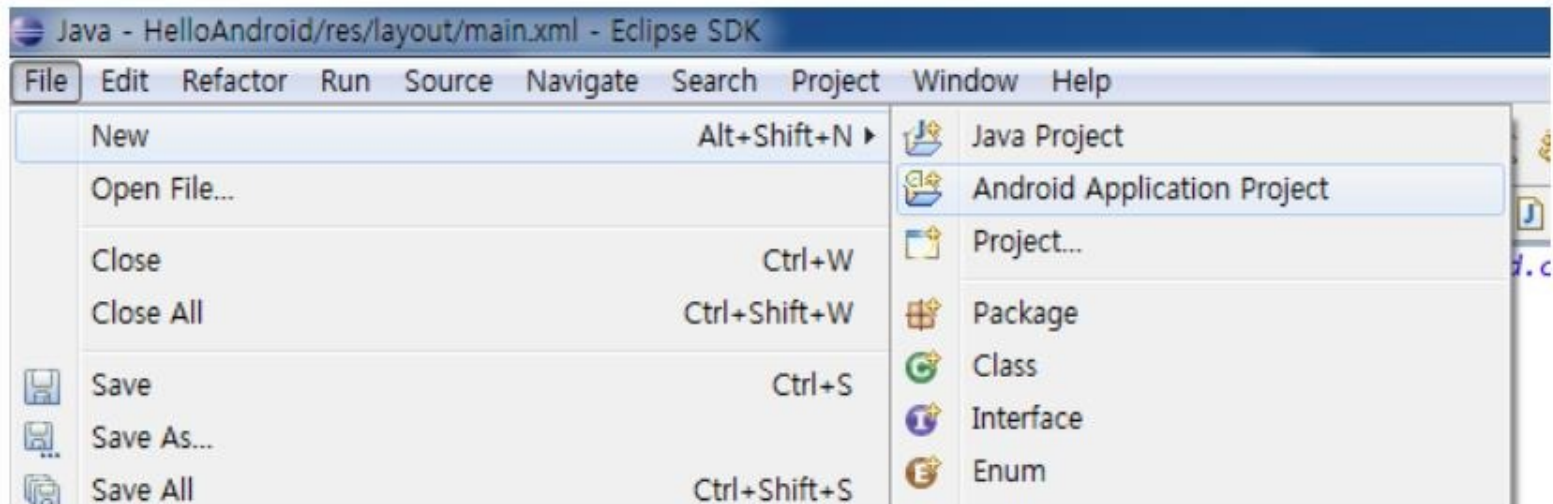
3

4

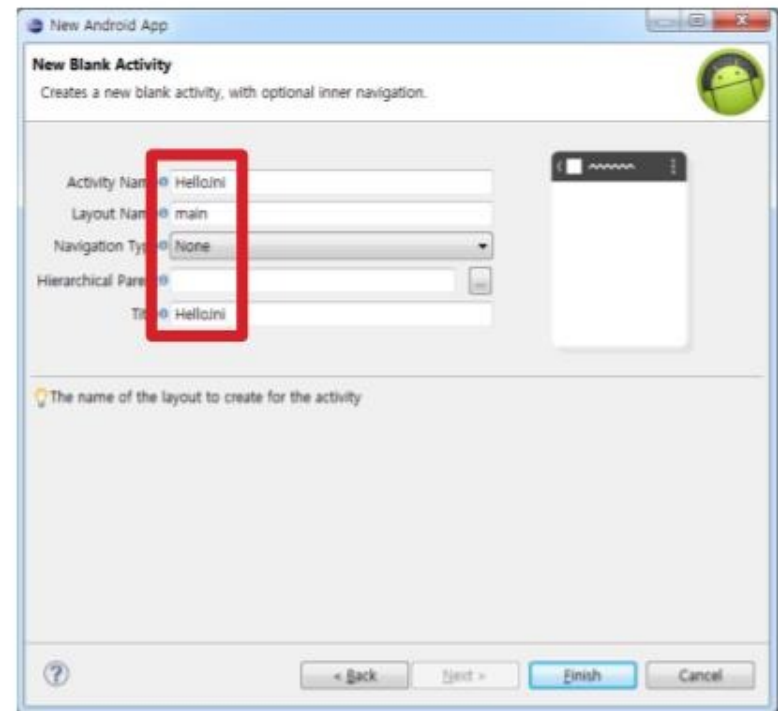
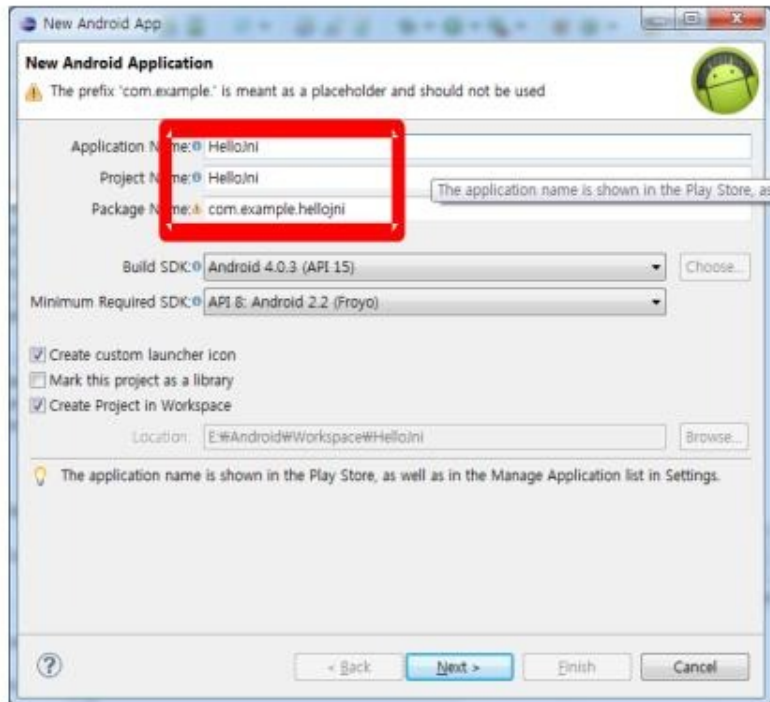
- 함수의 리턴값, 현재 이 함수는 java string 형태로 값을 리턴한다.
- Package 이름이다. 현재 이 함수를 사용하는 안드로이드 프로그램은 com.example.hellojni 이라는 package 이름을 가져야 한다.
- Activity 이름이다. 이 라이브러리를 사용하는 안드로이드 프로그램의 Activity 이름은 반드시 HelloJni 이어야 한다.
- 함수 이름이다. 안드로이드에서 사용하는 함수이름이다.

NDK를 이용한 JNI 개발 환경 구축

▶ JNI 예제 작성



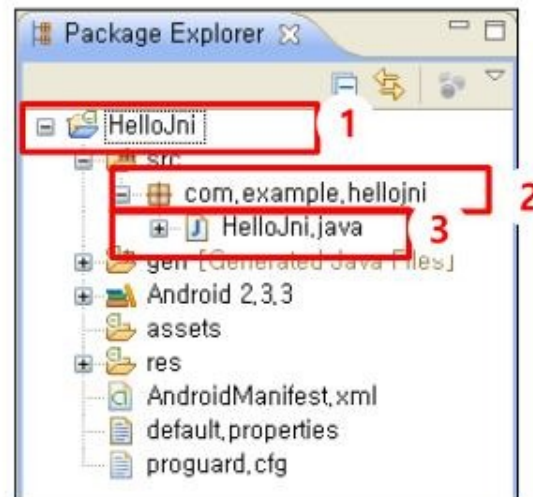
NDK를 이용한 JNI개발 환경 구축



NDK를 이용한 JNI개발 환경 구축

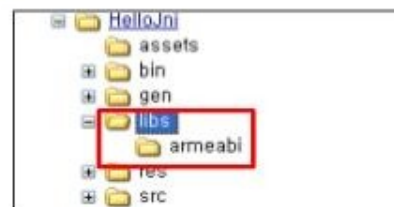
▶ eclipse에 생성된 프로젝트를 보여준다

- 1 은 앞서 입력한 project name 이다.
- 2 는 package 이름이다.
- 3 은 입력한 Activity 이름으로 Java 파일이 생성된 것을 볼 수 있다.



▶ 프로젝트에 빌드한 라이브러리 추가

- Eclipse에서 드래그 & 드롭으로 복사 가능




HelloJni.java

```
package com.example.hellojni;

import android.os.Bundle;
import android.app.Activity;
import android.widget.TextView;

public class HelloJni extends Activity
{
    static { System.loadLibrary("hello-jni");}
    public native String stringFromJNI();
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);

        TextView tv = new TextView(this);
        tv.setText( stringFromJNI() );
        setContentView(tv);
    }
}
```



Hello-jni.c

```
#include <string.h>
#include <jni.h>

jstring
Java_com_example_hellojni_HelloJni_stringFromJNI( JNIEnv* env,
                                                    jobject thiz )
{
    return (*env)->NewStringUTF(env, "Hello from JNI !");
}
```

HelloJni 결과화면

