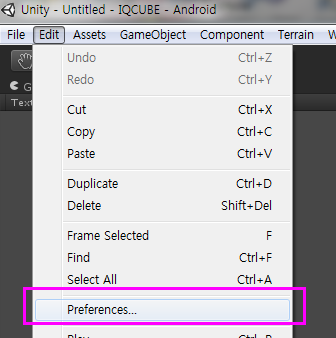
|  |  |
| --- | --- |
| |  | | --- | | **1-1. 유니티 안드로이드 플러그인 만들기 - Android 개발 환경 구축 하기 - JDK 설치 및 윈도우 설정** http://blogimgs.naver.net/imgs/nblog/spc.gif [Unity](http://blog.naver.com/PostList.nhn?blogId=saram95&categoryNo=30&from=postList) / [프로그래밍관련](http://blog.naver.com/PostList.nhn?blogId=saram95&categoryNo=11&parentCategoryNo=11&from=postList) http://blogimgs.naver.net/imgs/nblog/spc.gif  2013/07/09 14:21  복사<http://blog.naver.com/saram95/90176731230>  [전용뷰어 보기](http://blog.naver.com/PostView.nhn?blogId=saram95&logNo=90176731230&parentCategoryNo=&categoryNo=&viewDate=&isShowPopularPosts=false&from=postView) |   Android 개발하기 위해 필요한 요소 JDK, eclips, Android SDK 세가지 설치 방법에 대해 알아본다.  JDK 설치하기  <http://www.oracle.com/technetwork/java/index.html>  사이트에서 Java SE 최신 버젼을 받아 설치한다.    http://postfiles2.naver.net/20130709_17/saram95_1373347930187idEhJ_PNG/Image.png?type=w2    http://postfiles11.naver.net/20130709_26/saram95_1373347120911SytYO_PNG/Image_1.png?type=w2  http://postfiles14.naver.net/20130709_125/saram95_1373347121069aJ8tW_PNG/Image_2.png?type=w2        위에 표시된 Accept License Agreement 를 선택해주면 아래와 같이 화면이 바뀐다 http://postfiles6.naver.net/20130709_69/saram95_1373347121187wT5BH_PNG/Image_3.png?type=w2    설치하려는 컴퓨터의 OS 에 맞는 것을 다운 받는다.  아래부터 설명은 OS 는 Windows x64 로 (64비트 윈도우) 설명된다.  jdk-7u21windows-x64.exe 를 다운 받았으면 설치를 하자  참고> JDK 는 개발용 java 이며 jre 는 자바 실행용 이다. 우리는 jdk 를 받아 설치한다. 왜 개발해야하니까~!  http://postfiles3.naver.net/20130709_82/saram95_1373347121280UfL70_PNG/Image_4.png?type=w2  http://postfiles11.naver.net/20130709_202/saram95_1373347121528WTH15_PNG/Image_5.png?type=w2    설치 경로는 원하는 곳으로 바꿔줘도 된다. 단 바꿔준 폴더는 기억해둬야 한다.  뒤에 윈도우 환경 설정에서 폴더 경로를 정보를 설정 해 줘야한다.  일단 Default 로 놔두고 Next 를 눌러 설치를 시작한다.  http://postfiles9.naver.net/20130709_184/saram95_13733471216280JsAy_PNG/Image_6.png?type=w2  http://postfiles9.naver.net/20130709_216/saram95_1373347121754MCL5v_PNG/Image_7.png?type=w2      Close 를 눌러 설치를 완료하자.  JDK 설치가 완료 되었다 이제 윈도우 환경 설정을 해주자.  아래는 Windows 7 64 비트 환경변수 설정하는 윈도우 여는 방법이다.  제어판>시스템 및 보안 > 시스템 의 고급 시스템 설정 창을 선택 또는  탐색기 에서 컴퓨터 를 마우스 우 클릭하여 속성창을 열면 위 제어판>시스템 및 보안 > 시스템  을 열은것과 같은 창이 아래 그림과 같이 열린다 여기서  고급 시스템 설정 을 선택 한다.  http://postfiles15.naver.net/20130709_222/saram95_13733471219555kOD2_PNG/Image_8.png?type=w2    고급 시스템 설정을 열고 고급 탭의 환경 변수 버튼을 클릭 해준다.  http://postfiles9.naver.net/20130709_8/saram95_13733471220506Cvcp_PNG/Image_9.png?type=w2      아래 표시된 시스템 변수의 새로 만들기 버튼을 클릭한다.  http://postfiles2.naver.net/20130709_17/saram95_13733471221464U3SN_PNG/Image_10.png?type=w2      열린 창에 설치한 Path 정보를 가지는 시스템 변수 를 추가한다  변수 이름은 JAVA\_HOME 으로 해준다.  (JAVA\_HOME 으로 해주는 이유는 JAVA\_HOME 환경변수를 찾는 LIB 나 SDK 가 있다.)  변수 값 부분은 위에서 JAVA 설치한 폴더 이름을 넣어주고 확인을 누른다.  http://postfiles1.naver.net/20130709_144/saram95_1373347122290auyQy_PNG/Image_11.png?type=w2      시스템 변수 목록 중 Path 항목이 있으면 편집을 누르고 없으면 새로 만들기를 눌러  변수 이름을 Path 로 생성한다.  Path 가 있다면 변수 값 뒤에 ;%JAVA\_HOME%\bin 를 넣어준다. 값들 분리자는 ; 로  끝에 ; 가 있다면 그냥 %JAVA\_HOME%\bin 만 넣어주면 된다.  변수 값이 아무것도 없을 경우는 없다면 %JAVA\_HOME%\bin 만 넣어주면 된다.  http://postfiles13.naver.net/20130709_172/saram95_1373347122385poHQI_PNG/Image_12.png?type=w2  윈도우 콘솔창을 열고 (widowkey + R 로 실행창이 열리면 cmd 치고 엔터 하면 열린다)  javac -version 을 실행하면 현재 Java version 정보가 나오면 위 path 설정이 제대로 이루어진것이다.  그렇지 않다면 Path 설정 경로 이름이 잘 모소디었거나 변수 값을 추가할때 ; 빼먹었거나 다시 확인해보자.  위 작없이 끝났으면 path 말고도 새로운 변수 를 하나더 추가해야한다  변수 이름: CLASSPATH  값 :   .;%JAVA\_HOME%\lib\  http://postfiles7.naver.net/20130709_6/saram95_1373347122477VPpQL_PNG/Image_13.png?type=w2      추가가 다 끝났으면 확인을 눌러 설정을 끝낸다.  위와 같은 과정이 끝나면 JDK 설치 및 설정은 끝이다.  다음은 이클립스 설치에 대해 알아본다.  **[출처]** [1-1. 유니티 안드로이드 플러그인 만들기 - Android 개발 환경 구축 하기 - JDK 설치 및 윈도우 설정](http://blog.naver.com/saram95/90176731230)|**작성자** [강이](http://blog.naver.com/saram95) |

|  |  |
| --- | --- |
| |  | | --- | | **1-2. 유니티 안드로이드 플러그인 만들기 - Android 개발 환경 구축 하기 - Android SDK 설치 및 eclipse 설치** http://blogimgs.naver.net/imgs/nblog/spc.gif [Unity](http://blog.naver.com/PostList.nhn?blogId=saram95&categoryNo=30&from=postList) / [프로그래밍관련](http://blog.naver.com/PostList.nhn?blogId=saram95&categoryNo=11&parentCategoryNo=11&from=postList) http://blogimgs.naver.net/imgs/nblog/spc.gif  2013/07/09 14:31  복사<http://blog.naver.com/saram95/90176731853>  [전용뷰어 보기](http://blog.naver.com/PostView.nhn?blogId=saram95&logNo=90176731853&parentCategoryNo=&categoryNo=&viewDate=&isShowPopularPosts=false&from=postView) |     jdk 이 설치가 끝났다면 Android SDK 를 설치하도록 한다.  Android SDK 설치  <http://developer.android.com/index.html>  에서 Get SDK 항목을 선택 한다 또는  <http://developer.android.com/sdk/index.html>  로 들어가면 바로 가진다.  http://postfiles2.naver.net/20130709_49/saram95_1373347522010WFfsB_PNG/Image.png?type=w2  http://postfiles14.naver.net/20130709_29/saram95_1373347518503mHsRC_PNG/Image_1.png?type=w2    Check Box 를 Check 해준 다음  운영체제에 맛는 32-bit 또는 64-bit 를 선택 후  Download 버튼을 눌러 파일을 다운 받는다.  아래부터 설명은 64-bit 운영체제를 선택 후 설치에 대해 설명된다.  다운 받은 zip 파일을 적당한 폴더를 설정하여 압축을 풀어준다.  압축을 풀어준 경로를 [D:\AndroidSDK](file:///D:\AndroidSDK) 로 해줬다.  (파일이름으로 폴더를 만들도록 압축을 풀지 않았다. 만약 폴더 이름이 다르다면 압축푼 파일을 원하는 폴더로  이름을 바꾸거나 이동 시킨다.)  D:\AndroidSDK 폴더 안을 보면  http://postfiles14.naver.net/20130709_13/saram95_1373347518613M2pFC_PNG/Image_2.png?type=w2    와 같이 되어있다.  만약 AndroidSDK 에 압축을 풀었는데 위와 같은 구성이 아니라 압축 파일 이름과같은 폴더가 있다면 그 폴더 안에 이와 같이 되어있을 것이다.  해당 폴더에 있는 내용을 [D:\AndoridSDK](file:///D:\AndoridSDK) 로 옮겨도 상관 없다. 일단 위 처럼 SDK Manager.exe 및 SDK 폴더가 확인한다.  본인의 경우는 [D:\AndroidSDK](file:///D:\AndroidSDK) 폴더안이 위와 같이 구성되어 있으므로 이를 기준으로 설명 하도록 하겠다.  안드로이드 tools 들이 어디서든 사용 할 수 있도록 환경변수 path 에  [D:\AndroidSDK\sdk\tools](file:///D:\AndroidSDK\sdk\tools) 를 추가 하도록 한다.  아래는 Windows 7 64 비트 환경변수 설정하는 윈도우 여는 방법이다.  제어판>시스템 및 보안 > 시스템 의 고급 시스템 설정 창을 선택 또는  탐색기 에서 컴퓨터 를 마우스 우 클릭하여 속성창을 열면 위 제어판>시스템 및 보안 > 시스템  을 열은것과 같은 창이 아래 그림과 같이 열린다 여기서  고급 시스템 설정 을 선택 한다.  http://postfiles13.naver.net/20130709_76/saram95_1373347518873nolKF_PNG/Image_3.png?type=w2    고급 시스템 설정을 열고 고급 탭의 환경 변수 버튼을 클릭 해준다.  http://postfiles2.naver.net/20130709_193/saram95_13733475190248FmIE_PNG/Image_4.png?type=w2  http://postfiles11.naver.net/20130709_218/saram95_1373347519185x6ssU_PNG/Image_5.png?type=w2  http://postfiles7.naver.net/20130709_150/saram95_13733475192846ojcV_PNG/Image_6.png?type=w2  Path 항목이 없다면 새로 만들기 한다음  [D:\AndroidSDK\sdk\tools](file:///D:\AndroidSDK\sdk\tools) 을 추가한다.  만약 기존 항목이 있다면 ; 를 앞에 써준다음 [D:\AndroidSDK\sdk\tools](file:///D:\AndroidSDK\sdk\tools) 를 추가해주면 된다  ; 는 항목 구분자 이다.  위 설정이 끝났다면  Android SDK 없데이트를 해주자  위 [D:\AndroidSDK](file:///D:\AndroidSDK) 폴더의 SDK Manager.exe 를 실행하면 다음과 같이 실행된다.  http://postfiles9.naver.net/20130709_168/saram95_1373347519396NUBlL_PNG/Image_7.png?type=w2  위처럼 보면 Android 4.2.2 등 버젼별로 Check 박스가 있다.  개발에 필요한 버젼에 대해 Check 를 설정하고 오른쪽 아래에 있는 Install 버튼을 눌러 설치를 한다.  전체 설치는 엄청난 시간을 필요로 한다. 각자 필요한 버젼에 대해 설치를 하도록 한다.  이렇게 하면 일단 기본적인 SDK 설치 및 설정은 끝났다.  =============================================================  다음은 Android SDK 를 설치했는데 AndroidSDK 폴더 안에 eclipse 가 없다면 (Eclipse 가 들어있지 않다면)  Eclipse 를 설치 및 설정 하는 방법에 대해 알아본다.  Eclipse 는 Unity 에서 사용할 Android 용 Plugin 을 만들 수 있도록 돕는 프로그램이다.  Plugin을 만들 일이 없다면 설치 할 필요가 없다.  Eclipse 는 Android SDK 를 받아 설치하면 안에 포함되어있기도 하지만 없는 경우도 있다.  여기서는 Android SDK 를 설치 후 Eclipse 가 포함되어 있지 않다면 다운 받아 설치 하는 방법에 대해 나와있다.  먼저 Eclipse 를 다운 받자.  <http://www.eclipse.org/>  에서 다운 로드 가능하다.  http://postfiles3.naver.net/20130709_226/saram95_1373347519743A8HnE_PNG/Image_8.png?type=w2  http://postfiles8.naver.net/20130709_135/saram95_1373347520297sEFz5_PNG/Image_9.png?type=w2  http://postfiles4.naver.net/20130709_131/saram95_13733475204107YoFd_PNG/Image_10.png?type=w2      위와 같이 Eclipse Classic 버젼을 다운 받는다.  다운로드 파일을 원하는 위치에 압축을 풀어준다 (설치없이 바로 실행가능하도록 되어있다)  압축을 풀어 eclipse.exe 파일이 있는 폴더를 [D:\eclipse](file:///D:\eclipse) 로 하였다  eclipse.exe 파일을 실행해준다.  http://postfiles1.naver.net/20130709_272/saram95_1373347520521xEYKW_PNG/Image_11.png?type=w2      과 같이 창이 뜬다.  Eclipse 로 작업할 폴더를 Workspace 에 설정해주는 것이다.  Use this as the default and do not ask again 을 check 해주면 이번에 설정된것을  default 로 항상 eclipse 가 열린다.  check 해주지 않으면 workspace 를 몇군대 설정하여 각 workspace 별 구성을 따로 해줄 수 있다.  [D:\AndroidWorkRoot](file:///D:\AndroidWorkRoot) 라는 폴더를 만들어 위 같이 설정 하고 OK 를 눌렀다.  http://postfiles5.naver.net/20130709_84/saram95_1373347520847HANCT_PNG/Image_12.png?type=w2  이제 Eclipse 가 실행 되었다.  Android 개발을 위해 ADT 를 설정하도록 한다.  메뉴의 Help > Install New Software 를 선택 한다.  http://postfiles10.naver.net/20130709_137/saram95_13733475209550Poyo_PNG/Image_13.png?type=w2  Add 버튼을 누른다.  http://postfiles3.naver.net/20130709_226/saram95_1373347521060k0kAk_PNG/Image_14.png?type=w2  Name 은 알아서 설정해 준다 ex: ADT Plugin  Location 은 <http://dl-ssl.google.com/android/eclipse> 로 설정해준다음 ok 를 클릭한다.  http://postfiles8.naver.net/20130709_39/saram95_1373347521209DLVfe_PNG/Image_15.png?type=w2    아래와 같이 두가지 항목이 생긴다  http://postfiles16.naver.net/20130709_127/saram95_137334752212624hP4_PNG/ScreenClip.png?type=w2      둘다 선택 한 다음 오른쪽 밑에 next 버튼을 눌러준다.  http://postfiles5.naver.net/20130709_52/saram95_1373347521336SGJC7_PNG/Image_16.png?type=w2  그 후 나오는 설치 관련 메세지에 대해 Next 및 라이센스 동의를 한 다음 설치를 하도록 한다.  (설치 후 eclipse 가 재시작을 한다고 나올 수 있다 그럼 동의 하면 자동으로 재시작 된다.)  이제 Android SDK 경로를 설정해주도록 한다.  메뉴에 Window>Preferences 를 선택한다.  http://postfiles5.naver.net/20130709_100/saram95_1373347521466Llipa_PNG/Image_17.png?type=w2  아래 와 같이 창이 뜨는데 Android 항목을 선택하면 오른쪽에 SDK Location 이 있다.  이게 설정 되어있지 않다면 오른쪽 항목에 Adnroid SDK 관련 정보가 없다.  위 Android SDK 를 설치한 폴더의 하위 폴더 sdk 폴더(D:\AndroidSDK\sdk)를 SDK Location 에 Browse 버튼을 눌러  설정해주도록 한다.  설정이 끝나면 오른쪽 아래 리스트 박스에 안드로이드 SDK 버젼들이 나타날 것이다.  오른쪽 아래 Apply 버튼을 눌러 적용시킨다음 OK 를 눌러준다.  http://postfiles16.naver.net/20130709_127/saram95_1373347521583QpTrP_PNG/Image_18.png?type=w2  기본 적인 Android 개발 준비가 끝났다.  Unity 에서 Plugin 을 만들어 보도록 하자. |

Unity 에서 Android SDK 경로 설정해주기

Unity 로 개발된 Project를 Android 용 설치 파일 apk 로 빌드 하려면 Android SDK 가 있는 경로를 설정 해줘야 한다.

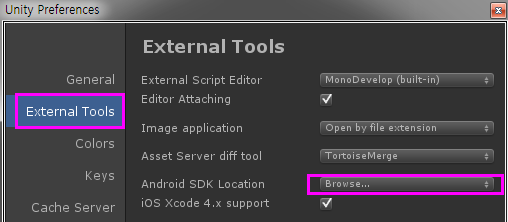
Unity menu 에서 Edit > Preferences 를 선택 한다.



External Tools 탭을 선택한다음 Android SDK Location 항목의 Browse.. 버튼을 누르면 탐색기가 뜬다.

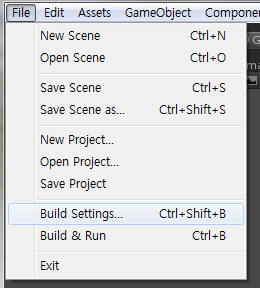
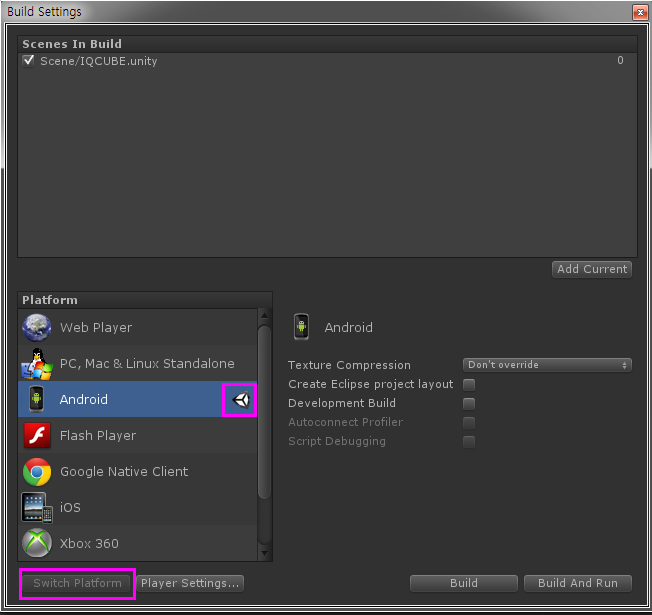
해당 탐색기에서 Android SDK 가 설치 되어있는 폴더 (이전글 1-2 에서는 [D:\AndroidSDK\sdk\](file:///D:\AndroidSDK\sdk\) 폴더)를

선택한다.



아드로이드 빌드는 Build setting 에서 Platform 이 android 로 되어있지 않다면 Android 로

변경해준다.

위 에서 보듯이 Android 부분에 Unity icon 표시가 있는것이 현재 설정되어있는 Platform 을 나타낸다.

Android 옆에 아니라 다른곳에 icon 이 있다면 Android 를 선택한다음 왼쪽 아래 Switch Platform 을 눌러주면 변경이 된다.

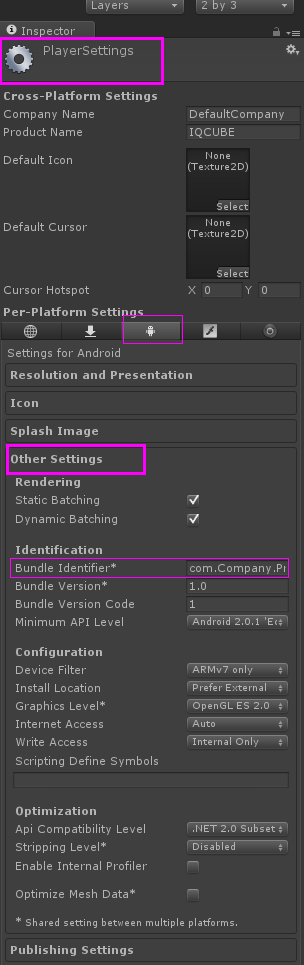
Bundle Identifier 설정

BundleIdentifier 는 만든 앱을 구글 Play 에 올릴때 필요한 ID로 Plugin 개발 시

Activity 관련 설정이 필요하다면 이 정보가 필요하다.

지금 임시로 정했다면 나중에 수정하는 방법도 있다.

이를 설정하려면 위 Build Settings 에서 Player Settings 버튼을 클릭한다.



Palyer Setting 은 Unty 창 중 Inspector 에 나타난다.

위 그림에서 안드로이드 탭이 선택되어 있지 않다면 선택해주고 Other Settings 를 선택해줘 메뉴를 펼쳐준다.

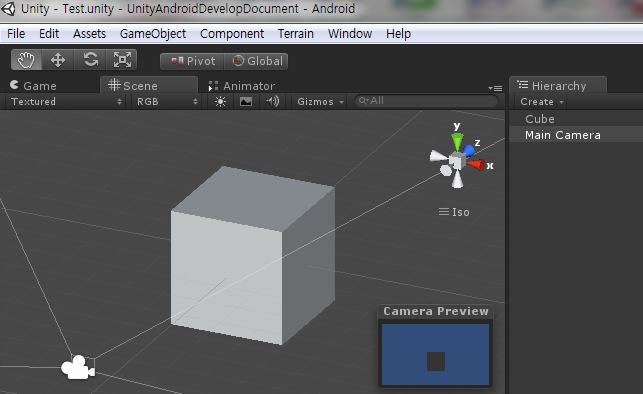
Bundle Identifier 부분을 수정해주도록 한다.

예제로 com.Kangs.TestGame 으로 설정했다고 하고 앞으로 이 BundleID 를 기준으로 설명 하도록 하겠다.

이제 안드로이드 핸드폰에서 우리가 만든 Untiy 프로젝트를 실행할 수 있는 준비가 되었다.

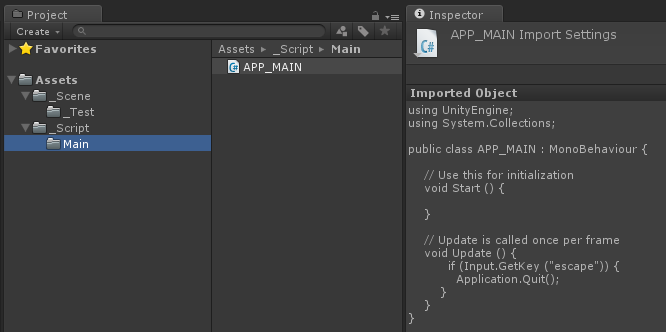
우리가 작성한 프로젝트가 안드로이드용으로 만드는 방법일 먼저 알아보도록 하겠다.

일단 화면에 CUBE 를 만들어 카메라에 보이도록 한다음 핸드폰에서 실행하면 보이도록 만들어보자

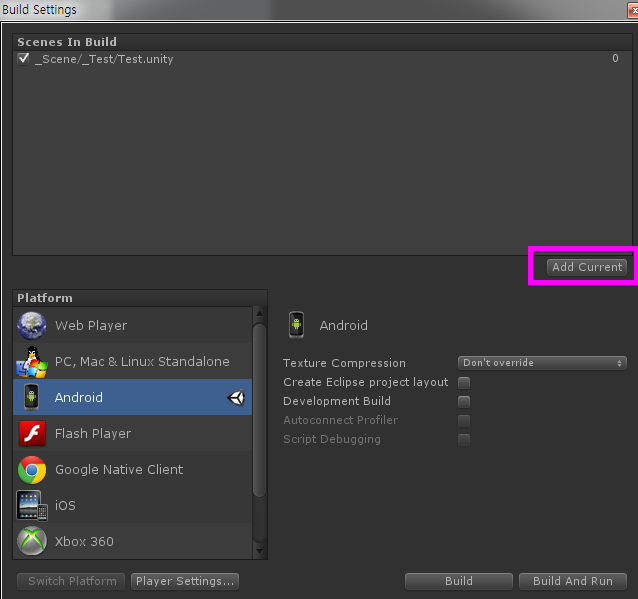


Build Scene 을 저장해주고

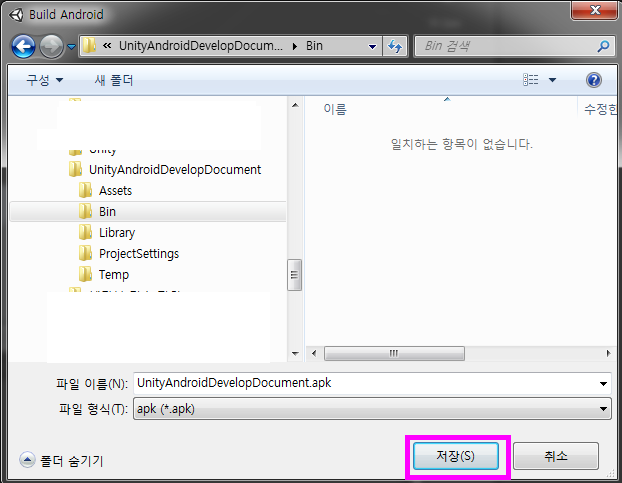
종료 가능하도록 간단하게 스크립트를 만들어 카메라에 붙여줬다.



Build Setting 에 현재 Scene 을 Add Current 버튼을 눌러 추가해준다.



Build 버튼을 눌러 Android 용 APK 를 만들자



핸드폰을 USB에 연결하고 기타 USB 드라이브를 설치등을 한다음 Build And Run 을 사용하여 바로 핸드폰에서 확인 할 수도 있으나

이 방법은 몇가지 설정이 필요 할 수도 있으므로 설정 방법은 여기서 다루지 않도록 하겠다.

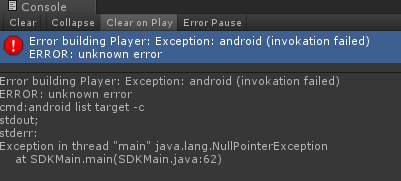
만들어진 apk 파일을 핸드폰에 설치 해서 확인해본다.

\* 에러가 발생하는 경우

여기서 Build 를 눌렀을때 Android 를 폴더 경로를 위에서 설정 했음에도 불구 하고 경로를 묻는  탐색기 창이 뜨는 경우가 있다.

이때 Android SDK 를 설치한 폴더 하위의 sdk 폴더를 선택해준다. (ex: [D:\AndroidSDK\sdk)](file:///D:\AndroidSDK\sdk))

설정해줬는데도 빌드 후에 콘솔 창에 다음과 같은 에러가 표시 되는 경우가 있다.



Error building Player: Exception: android (invokation failed) 에러는 여러가지 경우가 있는데

내가 겪은 것에대해서 설명하겠다.

구글링하다가 찾았던 부분인데 Android SDK 내부의 파일들을 복사하면 해결된다.

sdk 폴더 밑에 build-tools 폴더의 android-4.2.2 폴더안에 있는 파일들을

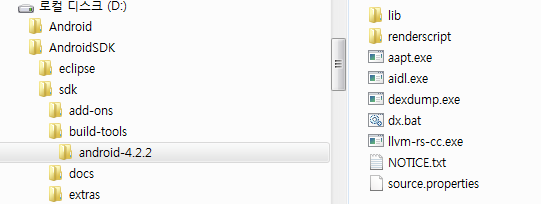
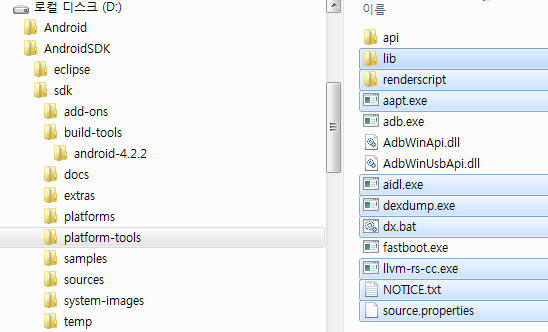
sdk 폴더 밑의 platform-tools 폴더 안에 덮어 씌여 주면 된다.

(ex [D:\AndroidSDK\sdk\build-tools\android-4.2.2](file:///D:\AndroidSDK\sdk\build-tools\android-4.2.2) -> [D:\AndroidSDK\sdk\platform-tools](file:///D:\AndroidSDK\sdk\platform-tools) )

이번에 설치한 SDK 가 아니라 이전부터 있던 SDK 를 사용 하던 사람들은

build-tools 밑에 android-4.2.2 폴더가 아니라. 17.0.0 폴더가 있을 수도 있다.

그런 경우는 17.0.0 폴더 안에 있는 파일들을 platform-tools 폴더에 덮어씌워주면 된다.

위 처럼 설정 해주고 다시 빌드 하면 SDK 폴더를 다시 물어본다 (아까 못찾았으니 다시 물어보는 것 같다)

sdk 폴더(D:\AndroidSDK\sdk\)를 설정해주고 계속 진행하면 드디어 apk 파일이 생성된다.

이를 설치해서 확인하자



여기까지 안드로이드용 프로젝트 빌드 까지 알아봤다.

**[출처]** [1-3. 유니티 안드로이드 플러그인 만들기 - Unity 안드로이드 빌드](http://blog.naver.com/saram95/90176732345)|**작성자** [강이](http://blog.naver.com/saram95)

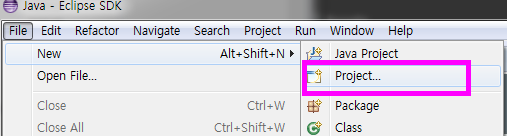
안드로이드용으로 제공되는 SDK 나 Lib 를 사용하기 전에 java 라이브러리를 Untiy 에 native 하게 연동 하는 방법을 알아보도록 한다.

다른 라이브러리를 사용하기 앞서 우리가 만든 테스트용 Java 함수를 Unity 내부에서 호출하는 Plugin 을 만들어 보도록 한다.

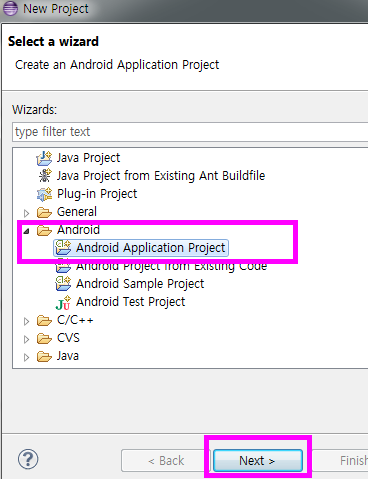
Java로 간단한 기능을 하는 Class 를 만들어 보도록 한다.

Eclipse 를 실행한다

이제 Android 용 Project 를 만들어 보자



File > New > Project 를 선택한다.



Android 의 Android Application Project 를 선택한다음 Next > 를 눌러준다.

아래 그림에서

Application Name 은 구글 Play 스토어에서 표시될 application 이름 이며

Project Name은 Eclipse 에서 관리된 프로젝트 이름이다.

PackageName 은 위 Unity 에서 설정한 BundleIdentifier 에 설저한 값으로 적어준다.

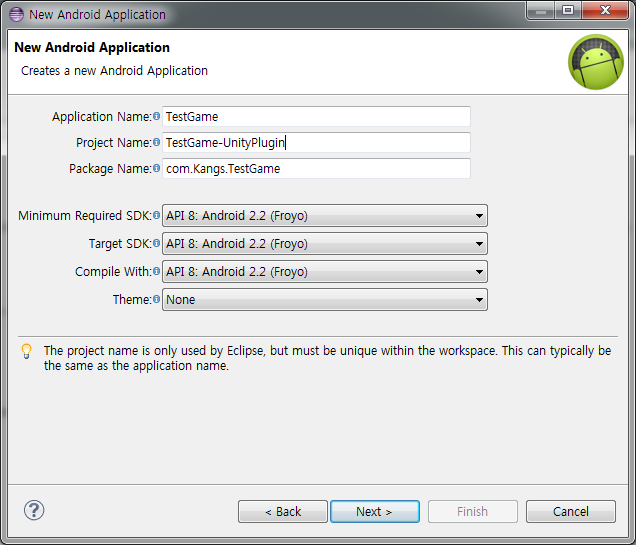
밑에 Minimum Required SDK 는 최소 요구 SDK 버젼이고 Target SDK 는 이 프로젝트에서 대상으로 하는 SDK 버젼

이다. 이는 알아서 하도록 한다 Android SDK 버젼에 따라 사용 할 수 있는 기능 제약과 사용 가능한 기계가 정해지므로

2.2 를 추천하는 바이다.

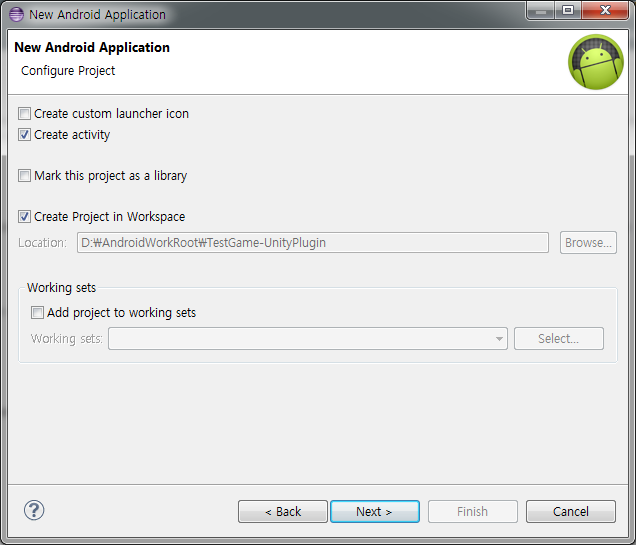
Complie Width 도 2.2 로 설정하였다.

Theme 는 None 으로 선택하도록 한후 Next > 버튼을 눌러준다.

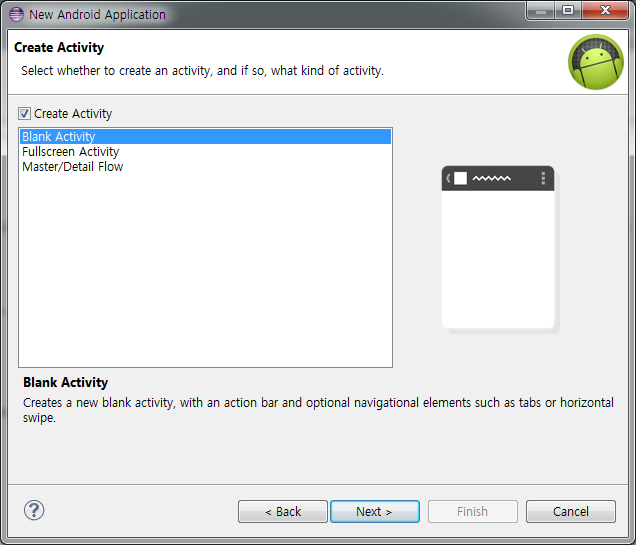


다음 단계에서는 기본적으로 Create custom launcher icon 항목과 Create activity 항목, Create Project in Worksapce 항목이 선택되어있다.

이중 Create custom launcher icon 항목은 아래 그림처럼 선택 해제를 시켜주고 Next 버튼을 누른다.



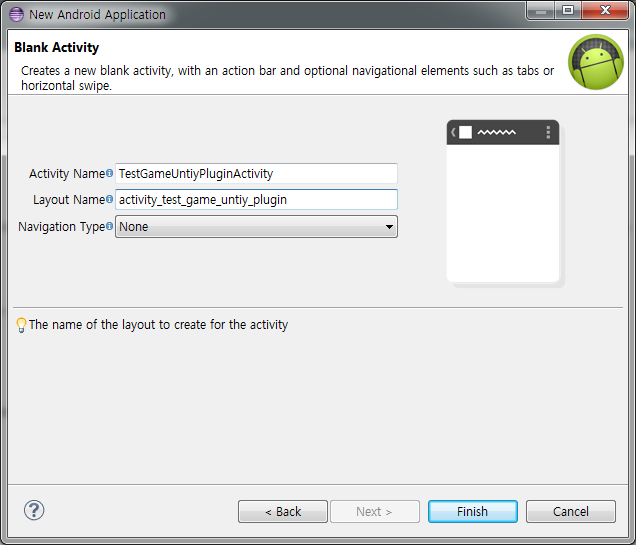
Blank Activity 항목으로 놔둔상태로 Next 버튼을 누른다.



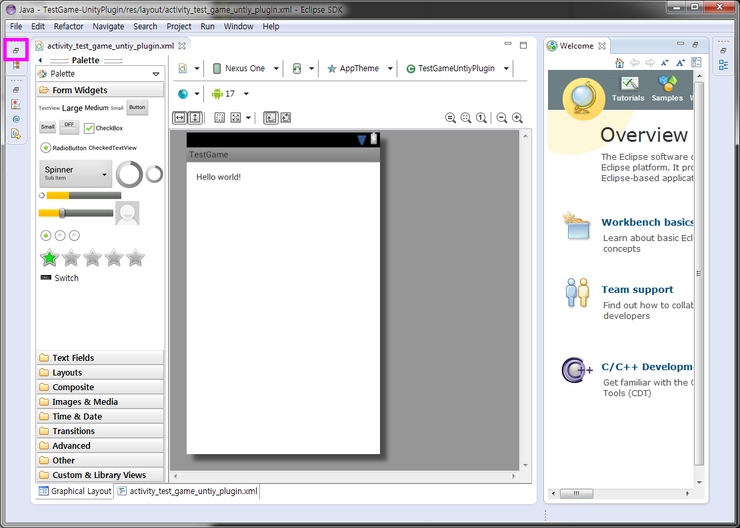
다음단계에서 Activity Name 은 적당한 이름을 설정해준다. (Unity 의 안드로이드 메인 윈도우 클래스 이름이라고 생각해도 된다)

이름을 변경하면 자동으로 LayoutName 도 변경되는데 이는 그냥 놔둬도 된다.

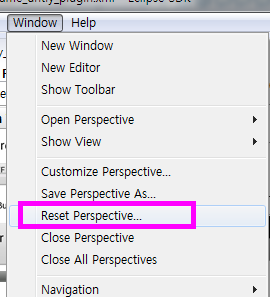
변경했다면 Finsh 버튼을 누른다.



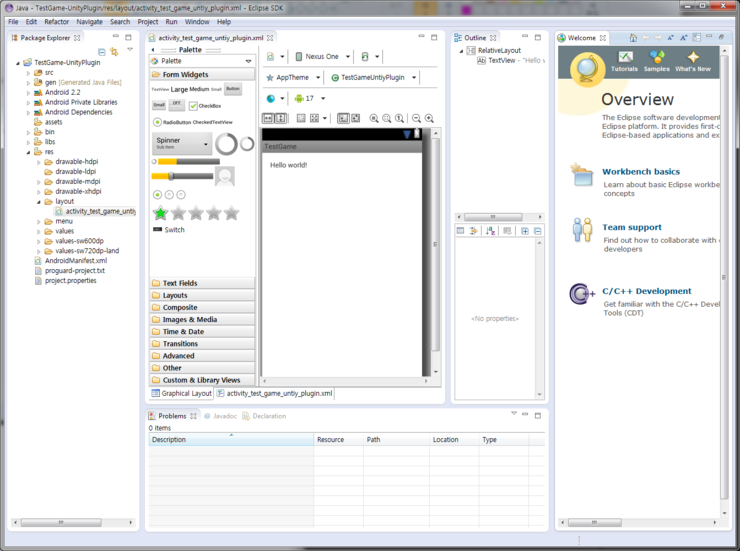
Eclipse 에서 처음으로 프로젝트를 만들었다면 아래와 같은 창이 뜬다.



메뉴에 Window > Reset Persepective... 항목을 선택해 화면 구성을 reset 시켜준다.



아래와 같이 화면 구성이 변경된다.

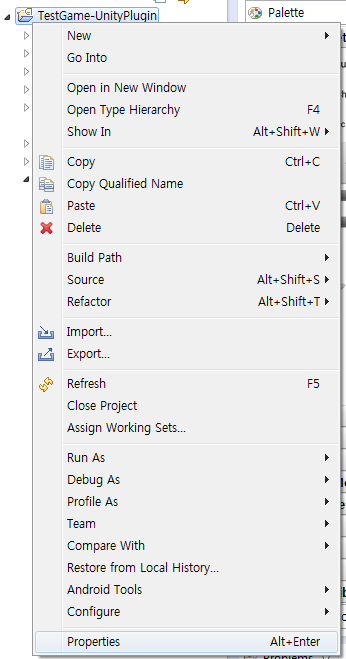


유니티와 연동을 위해 Unity 에서 제공하는 안드로이드용 라이브러리를 연결해 준다.

연결하지 않고 작업을 해도 상관은 없으나 Eclipse 에서 빌드시 에러가 날 수 있다 Eclipse 에서 에러가 나더라도 코드에 문제 없다면

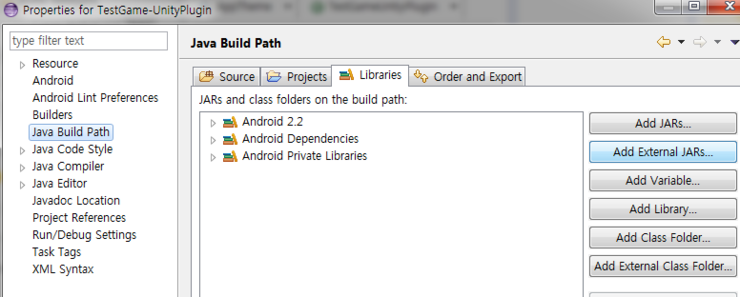
Unity 에 사용 할 수 도 있으나 문제의 여지가 있으므로 일단 에러가 나지 않은 방법으로 작업한다.

라이브러리르 연결하기 프로젝트를 마우스 우클릭해서 Properties 를 선택해준다.



Properties 창에서

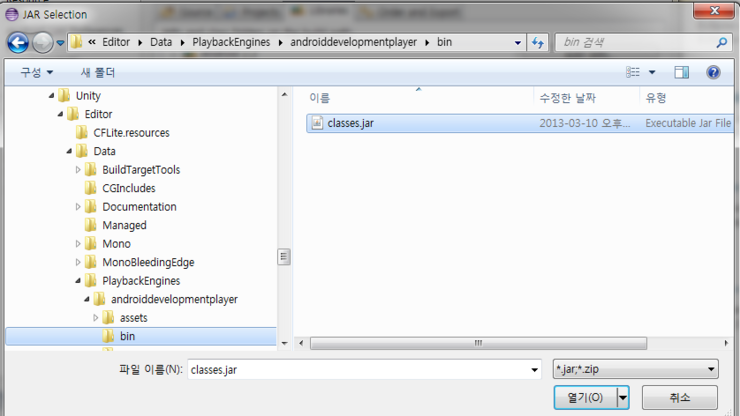
Java Build Path 를 선택하고 Libraries 탭을 선택한다음 Add External JARS... 버튼을 클릭한다.



선택할 JAR 파일 경로는 Unity 설치된 폴더를

C:\Program Files (x86)\Unity 라고 할때

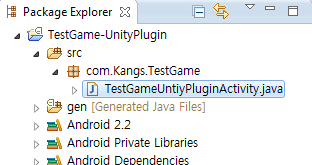
C:\Program Files (x86)\Unity\Editor\Data\PlaybackEngines\androiddevelopmentplayer\bin 폴더 의 classes.jar 파일을 선택해준다.



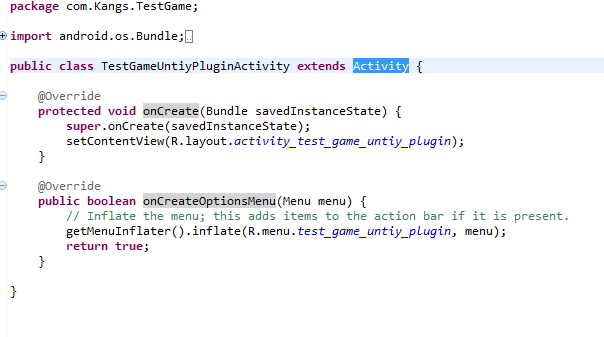
아래와 같이 classes.jar 가 추가 된것을 볼 수 있다. OK 버튼을 눌러 준다.



프로젝트에서 src 아래 com.Kangs.TestGame 아래 TestGameUnityPluginActivity.java 파일을  더블클릭으로 열어준다.



기본적으로 아래와 같이 코드가 구성되어있다.



위 Activity 부분을 UnityPlayerActivity 로 변경 해준다.

그러면

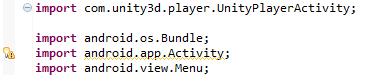
위에서 세번째줄

http://postfiles16.naver.net/20130709_95/saram95_1373348420653GwyLy_PNG/Image_15.png?type=w2  
부분이

http://postfiles3.naver.net/20130709_194/saram95_1373348420752Ah0q1_PNG/Image_16.png?type=w2  
로 바뀌는걸  볼 수 있다.

http://postfiles13.naver.net/20130709_220/saram95_1373348420847LPF7T_PNG/Image_17.png?type=w2  표시는 사용하지 않는 import 된 사항이 있다는 것이다.

http://postfiles6.naver.net/20130709_37/saram95_1373348420977KwryQ_PNG/Image_18.png?type=w2 를 클릭하면 import  된 항목들이 다음과 같이 확장되어진다.



전에 있던 Activity 를 UnityPlayerActivity 로 변경하였기 때문에 import android.app.Activity; 부분을 사용 하지 않게되어 그부분에 표시가 생긴 것이다.

control+shift+O 키를 누르면 현재 문서에서 사용되는 import 를 정리하여 없는 import 항목은 추가해주고 사용 안하는 import 항목은 삭제해준다.

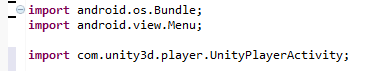
위에서 Unity 에서 class.jar lib 를 속서엥 추가해줬기 때문에 UnityPlayerActivity 항목도 추가되었던 것이다.

만약 문서에서 사용한 항목 중 import 될 수 있는 종류 항목이 (클래스 이름과 같은 같은 항목) 몇군데 있다면 어느것을 import 시킬건지 선택창이 뜬다.

알아서 선택해서 해줘도 된다.

위 단축키를 사용하지 않고 직접 해줘도 상관 없다. 단축키는 편리성 제공일 뿐이다.

control+shift+O 를 눌르면 아래와 같이 변경 된 것을 볼 수 있다.



현재로서 불필요한 부분을 몇군대 지워 아래처럼 만들자

      setContentView(R.layout.*activity\_test\_game\_untiy\_plugin*);

부분과

     @Override

**public** **boolean** onCreateOptionsMenu(Menu menu) {

               // Inflate the menu; this adds items to the action bar if it is present.

              getMenuInflater().inflate(R.menu. *test\_game\_untiy\_plugin*, menu);

**return** **true** ;

       }

부분을 아래처럼 지워 코드를 깔끔하게 만든다.

**package** com.Kangs.TestGame;

**import** android.os.Bundle;

**import** com.unity3d.player.UnityPlayerActivity;

**public** **class** TestGameUntiyPluginActivity **extends** UnityPlayerActivity {

        @Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

       }

}

위에서 UnityPlayerActivity 항목을 상속 받도록 Activity 를 변경 했지만 그냥 Activity 항목을 상속 받도록 하고

사용 할수 도 있다. 이 방법은 UnityPlayerActivity 클래스가 어떻게 되어있는지 소스를 보고 참고해서 현재 Activity 를 수정 해주면 된다.

소스는 유니티 설치 폴더를

C:\Program Files (x86)\Unity\ 라고 하면

C:\Program Files (x86)\Unity\Editor\Data\PlaybackEngines\androiddevelopmentplayer\src\com\unity3d\player

안에 UnityPlayerActivity.java 에 있다. 한번 쯤 이 파일을 보는 것도 좋을 듯 하다.

참고) C:\Program Files (x86)\Unity\Editor\Data\PlaybackEngines\androiddevelopmentplayer\src\com\unity3d\player

폴더에 가면 UnityPlayerActivity.java 와 UnityPlayerNativiActivity.java, UnityPlayerProxyActivity.java 3개의 파일이 있다.

UnityPlayerProxyActivity 에서  Android 버젼이 2.3 이상 UnityPlayerNativiActivity 를 사용하고 아래면 UnityPlayerActivity 로 사용하도록 되어있지만

AndroidMenifest.xml 에서 Main Activity 를 정해주고 그 MainActivity 를 UnityPlaerActivity 를 상속 받은 것으로 만들어 주었다면 해당 Activity 를 사용하게 된다.

UnityPlayerActivity 는 그냥Activity 를 상속 받은 것이며 UnityPlayerNativiActivity  는 좀더 향상된 NativeActivity 를 상속 받아 구현 되어있다.

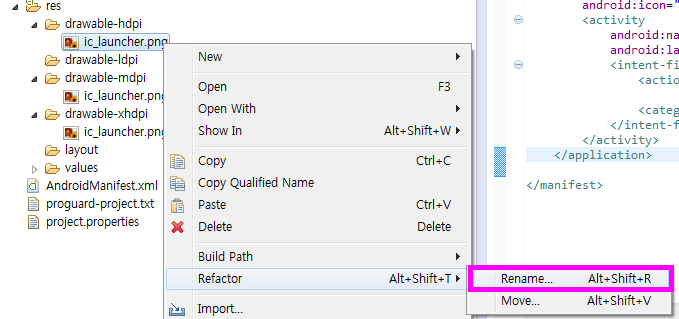
icon 이름을 바꿔주자 작업후 AndroidManifest.xml 파일을 Unity 프로젝트에 넣어줄때 icon 이름을 변경해줘야하는데

플러그인 수정할때마다 이름 변경해주는 귀찮음을 없애기 위해 ic\_launcher.png 로 되어있는 이름을 app\_icon.png 로 바꿔주도록 한다.

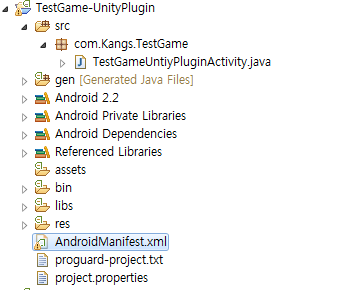
res 하위 폴더의 drawable 로 시작하는 것들을 확장해보면 ic\_launcher.png 들이 들어있다. 이 파일들을 마우스 우클릭해서

refactor 를 메뉴의 rename 을 이용하여 app\_icon.png 로 바꿔준다 (확장자까지 써야한다)

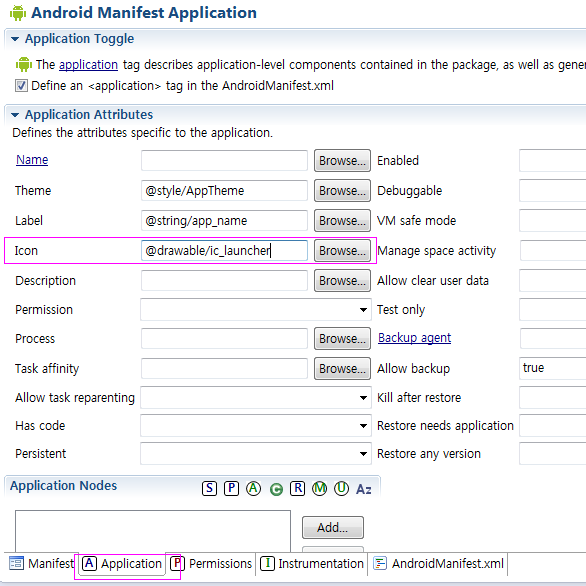
하나만 바꾸면 자동으로 바뀌는 경우가 있지만 그렇지 않은 경우도 있다. 안바뀐게 있다면 전부 app\_icon.png 로 바꿔주자



변경 후 프로젝트 하위의 AndroidManifest.xml 파일을 더블 클릭 하여 열어준다.



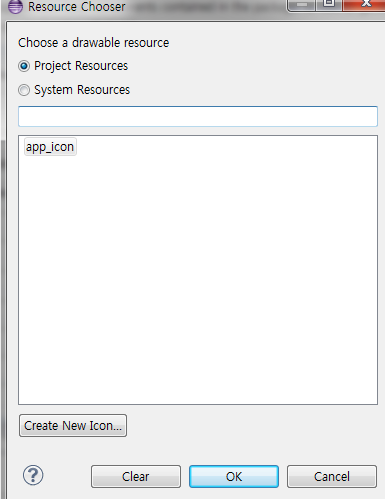
파일이 열린 후 문서 아래쪽 탭 중 Application 탭을 선택해 보면 Icon 이라는 부분이 있다



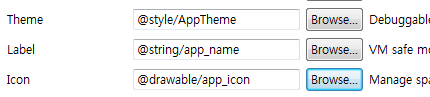
위처럼 @drawable/ic\_launcher 로 되어있거나 자동으로 @drawable/app\_icon 으로 되어있을 것이다.

만약 @drawable/ic\_launcher 로 되어있다면 옆의 Browse... 버튼을 클릭하여 app\_icon 을 선택해

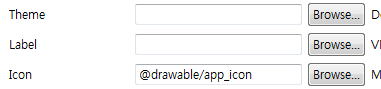
@drawable/app\_icon 이 되도록 바꿔준다.



변경이 되었다면 Theme 와 Label 안에 있는 부분은 그냥 지워주자



이처럼 되어있는 부분을 지워 아래처럼 만들어준다.



이제 간단한 테스트를 위해 TestGameUnityPluginActivity.java 에 코드를 추가해보자.

**package** com.Kangs.TestGame;

**import** android.os.Bundle;

**import** com.unity3d.player.UnityPlayer;

**import** com.unity3d.player.UnityPlayerActivity;

**public** **class** TestGameUntiyPluginActivity **extends** UnityPlayerActivity {

        @Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

       }

**int** GetPluginNum(**int** nA, **int** nB)

       {

**return** nA+nB;

       }

**void** PluginToUnitySendMessage()

       {

              UnityPlayer. *UnitySendMessage*( "GameObjectName"

                           , "CallFunctionName"

                           , "Message is Txt....Called PluginToUnitySendMessage");

       }

}

위와 같이 코드를 추가해 주었다.

**int** GetPluginNum()

함수는 Unity 에서 호출 하면 인자로 두개를 받아 더한 후 즉시 int 형 값를 받아 올 수 있는 함수이다

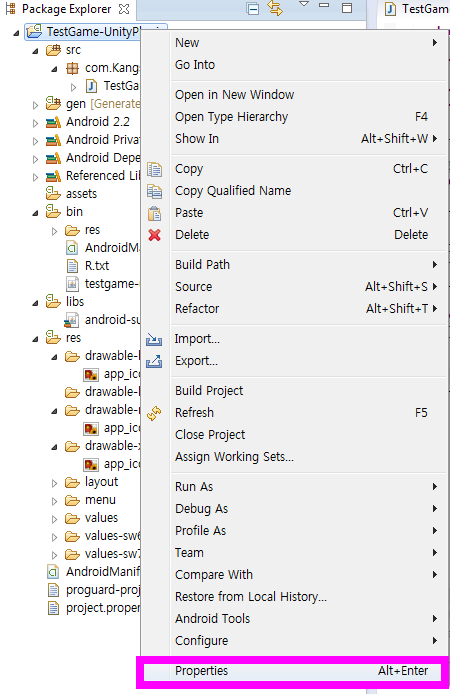
**void** PluginToUnitySendMessage()

함수는 현재 씬에 있는 GameObject 중 이름이 "GameObjectName" 인 Object에게 SendMessage 로

"CallFunctionName" 을 가진 함수에 string 을 인자로 Message 를 보내는 함수이다.

자 이렇게 작성된 내용을 Untiy 에 서 호출해보도록 하겠다.

프로젝트의 Properties 를 연다

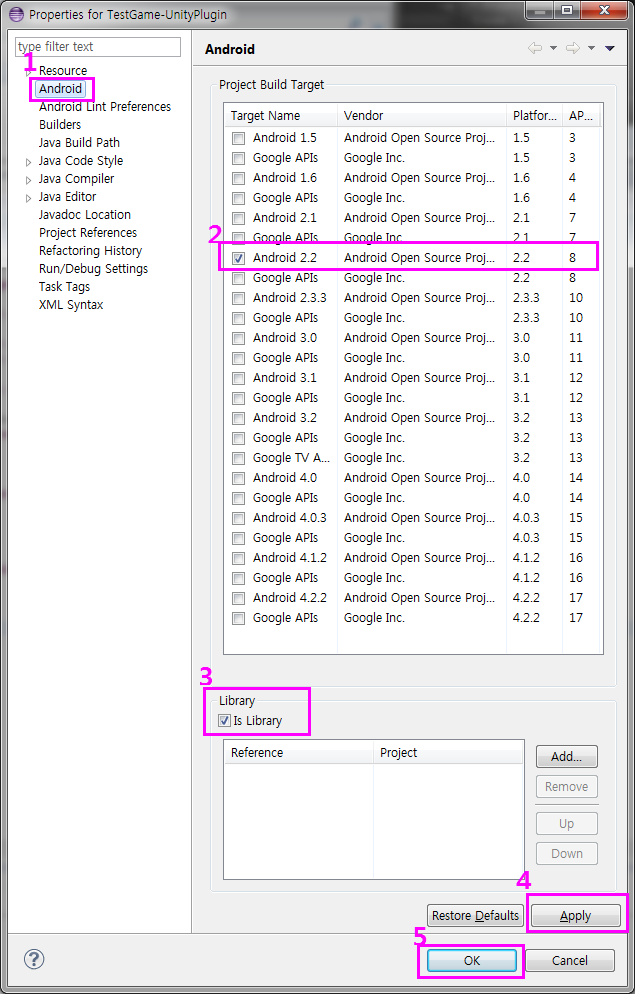


Android 를 선택하고 Project Build Target 이 Android 버젼 2.2 로 되어있지 않다면 2.2 로 해주자

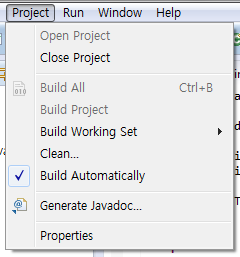
(2.2 정 도 되어야 후에 추가 해주는 Plugin 지원이 원활하다. 아니여도 되지만 2.2가 가장 적당한 듯하다.

후에 추가하려는 플러그인 SDK 등이 2.2 보다 더 높은 버젼에서만 지원한다면 이 버젼도 높여주도록 하자.)

Library 항목의 Is Library Check 박스가 Cehck 되도록 해준다음 Apply 버튼을 누르고 OK 버튼을 누른다.



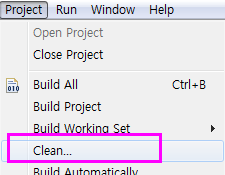
메뉴의 Project 항목의 Build Automatically 항목이 Check 되어있다면



한번더 클릭하여 Check 표시를 없애준다.

자동 빌드 옵션인데 프로젝트가 하나 이상되면 종종 오류가 없음에도 오류가 있다고 표시 되기도 한다.

메뉴의 Project 항목의 Clean... 을 선택한다.



다음과 같이 Clean all projects 로 되어있고 Start a build immediately 항목이 채크 되어있다.

이도로 OK 를 진행해도 되지만 후에 라이브러리 프로젝트들을 추가 하면 빌드 순서를 고려해야 하는 경우가

생길 수도 있다.

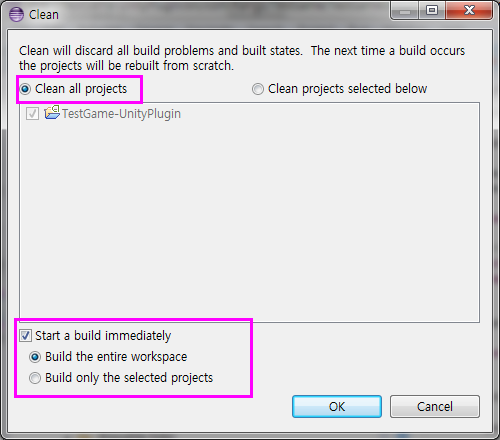
Clean all projects 는 모든 프로젝트를 Clean 하는 것이다 (현재는 하나밖에 없으니 하나만 clean 된다.)

Clean projects selected below 를 선택하면 현재 workspace 에 등록된 프로젝트중 선택하여 clean 을 동작 시킬 수 있다.

Start a build immediately 항목은 clean 수행호 자동으로 바로 build 되도록 할 것인지 여부등을 선택한다.

현재는 그대로 ok 를 해도 되지만 선택해서 clean 하고 Start a build immediately 를 uncheck 하여 임으로 빌드 되도록 하는것을

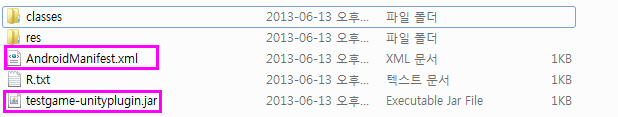
추천한다. 빌드 순서가 바뀌여 오류가 나오는 경우도 있다 그럴 경우 다시 빌드 해주면 문제가 없기는 하다.



Clean 은 build 상태를 초기화 시키는 것으로 가끔 코드를 수정하고 작성하다보면 오류가 없음에도 build 시 오류가 나타나는 경우가 있다.

이럴경우 해당 프로젝트를 Clean 한다음 다시 빌드 하면 된다.

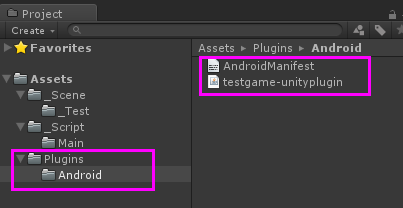
빌드가 끝나면 Project 가 있는 폴더 하위의 Bin 폴더(ex: D:\AndroidWorkRoot\TestGame-UnityPlugin\bin )를 탐색기로 열어보자.



위와 같이 나오는데 AndroidManifest.xml 파일과 testgame-unityplugin.jar 파일을 Unity Project 에 넣어줘야 한다.

넣어주는 방법은 Proejct에 Plugins 폴더를 만들고 그 하위에 Android 폴더를 만든다음 Android 폴더에

이 두 파일을 드래그 드롭하여 넣어주면 된다.



추가한 Plugin 을 사용하는 Script 를 작성해준다.

위에서 Plugin 에서 Untiy 를 보내는 메세지를 받을 Object 이름을  "GameObjectName" 로 하였으므로 Scene 에

GameObjectName 이라는 Object를 만들어준다.

안드로이드 Plugin 과 통신 하기 위한 스크립트를 만들어 준다.



PluginMgr\_Android 스크립트를 만들어줬으며 안에 아래와 같이 Sample 예제를 만들었다.

아래 샘플 예제는 버튼이 2개가 나오고

Called IntValue 버튼을 누르면 Eclipse 에서 만든 함수 GetPluginNum 함수를 호출하고 인자로 10 과 20을 넣어주어 그 리턴값을 아래쪽에 찍어주고

Called RequestMessage 버튼은 Eclipse 에서 만든 함수 PluginToUnitySendMessage 함수를 호출하여 스크립트의 CallFunctionName 함수로

스트링 Message is Txt....Called PluginToUnitySendMessage 을 보내주어 화면에 찍어주는 기능을 하도록 하였다.

스크립트 내용은 다음과 같다

using UnityEngine;  
using System.Collections;  
  
public class PluginMgr\_Android : MonoBehaviour {  
#if UNITY\_ANDROID  
      
     AndroidJavaObject          \_AndroidpluginObj = null;  
      
      
     // Use this for initialization  
     void Start () {  
          AndroidJNI.AttachCurrentThread();  
          \_Init();  
     }  
      
     void Destroy()  
     {  
          if( \_AndroidpluginObj != null )  
               \_AndroidpluginObj.Dispose();  
     }  
      
     public void CallFunctionName( string strParam )  
     {  
          strLabelReceiveMessage = strParam;  
     }  
     // Plugins\_Android\_Init  
      
     int \_Init()  
     {  
          AndroidJavaClass           \_androdiJC = new AndroidJavaClass("com.unity3d.player.UnityPlayer");  
      
          if( \_androdiJC != null )  
          {  
               \_AndroidpluginObj = \_androdiJC.GetStatic<AndroidJavaObject>("currentActivity");  
               \_androdiJC.Dispose();  
               if( \_AndroidpluginObj == null )  
               {  
                    return -1;  
               }  
                
               return 0;  
          }  
           
          return -1;  
     }  
      
      
     int CallIntValue()  
     {  
          if( \_AndroidpluginObj != null )  
          {  
               int nIntValue = \_AndroidpluginObj.Call<int>("GetPluginNum", 10, 20);  
               return nIntValue;  
          }  
           
          return 0;  
     }  
      
     void Req\_CheckPluginMessage()  
     {  
          if( \_AndroidpluginObj != null )  
          {  
               \_AndroidpluginObj.Call("PluginToUnitySendMessage");  
          }  
     }  
      
     string strLabelReturnInt = "";  
     string strLabelReceiveMessage = "";  
     void OnGUI()  
     {  
          if( GUILayout.Button("Called IntValue", GUILayout.Width(200)) )  
          {  
               int nReturnValue = CallIntValue();  
               strLabelReturnInt = nReturnValue.ToString();  
          }  
           
          if( GUILayout.Button("Called RequestMessage", GUILayout.Width(200)) )  
          {  
               Req\_CheckPluginMessage();  
          }  
           
           
          GUILayout.Label("ReturnInt : " );  
          GUILayout.Label( strLabelReturnInt );  
           
          GUILayout.Label("ReceiveMessage: " );  
          GUILayout.Label( strLabelReceiveMessage );  
     }  
#endif // UNITY\_ANDROID  
}

위에서 Init 함수에 보면

 AndroidJavaClass           \_androdiJC = new AndroidJavaClass("com.unity3d.player.UnityPlayer");

Class 접근을 위한 AndroidJavaClass 객채를 생성했다.

이는 Unity 가 안드로이드용으로 빌드 되면 내부에 UnityPlayer Class 를 사용하여 Unity가 구동 되는데.

UnityPlayer 객채 생성은 위에 이클립스에서 UnityPlayerActivity  클래스를 상속 받은 plugin activity class를 만들었는데

UnityPlayerActivity 의 소스를 확인해 보면 OnCreate 에서 UnityPlayer 객채를 new 를 해주면서 생서자에서 Activity 자신 this 를 넣어주는것을 볼 수있다.

this 를 넣어주는 이유는 UnityPlayer 내부에 있는 static 변수 currentActivity 를 생성자에서 설정한 this로 설정하기 위한 것인듯 하다.

currentActivity 변수로 현재 Activity를 어디서든 접근할 수 있다 static 변수이므로 UnityPlayer.currentActivity 로 접근 가능

그래서 Init 함수에서

AndroidJavaClass           \_androdiJC = new AndroidJavaClass("com.unity3d.player.UnityPlayer");

를 호출하므로서 자바 클래스 접근 객채를 만들고

클래스 내부에 있는 static 변수 currentActivity 를 얻어오기 위해

 \_AndroidpluginObj = \_androdiJC.GetStatic<AndroidJavaObject>("currentActivity");

로해서 Static 객채를 가져온다.

이로써 위에서 작성한 Activity 내부의 함수나 변수들이 접근 가능하다.

즉 위 이클립스에서 작성한 Activity 는 실행될때 자동으로 호출되어 OnCreate 가 불린다.

OnCreate 가 불리면서 UnityPlayer 객채가 생성 되고 만약 개인이 만든 class 나 SDK 클래스를 OnCreate 에서 생성해 Activity 내부에 static 변수 (ex 변수이름: testvalue) 로 할당한다면

Activity 클래스를 아래처럼

AndroidJavaClass            \_ActivityJavaClass         = new AndroidJavaClass("com.Kangs.TestGame.TestGameUntiyPluginActivity");

로 Class 접근 object 를 얻어온다음

AndroidJavaObject  \_memObj =  \_ActivityJavaClass.GetStatic<AndroidJavaObject>("testvalue");

로 해당 객체를 얻어올 수 있다.

해당 객체에 있는 함수는

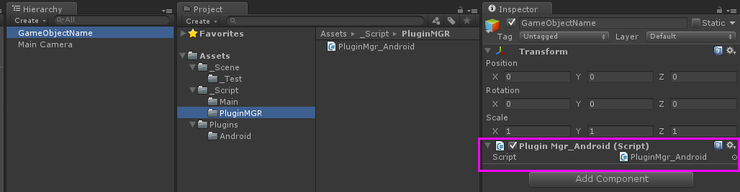
 \_memObj.Call("함수이름");  <- 반환값이 없는 함수.

또는

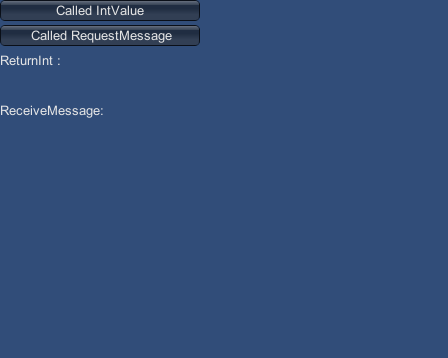
\_memObj.Call<int>("함수이름"); <-반환값이 있는 함수

와 같은 형식으로 호출 할 수 있다.

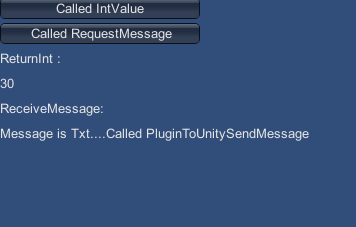
작성된 스크립트를 GameObjectName 이라는 이름으로 만든 Object에 연결해 준다.



이제 Build 하고 핸드폰에 실행해 보면 다음과 같이 화면이 뜬다.



각 각 버튼을 누르면 아래와 같은 결과가 나온다.



이제 기본적인 Plugin 추가 방법에 대해 알아봤다.