ANDROID基本教學

前置作業

講義、工具、範例下載

- 講義和範例
  - https://github.com/silencecork/AndroidBluetoothWorkshop2014
- 開發工具
  - https://developer.android.com/sdk/index.html
  - http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

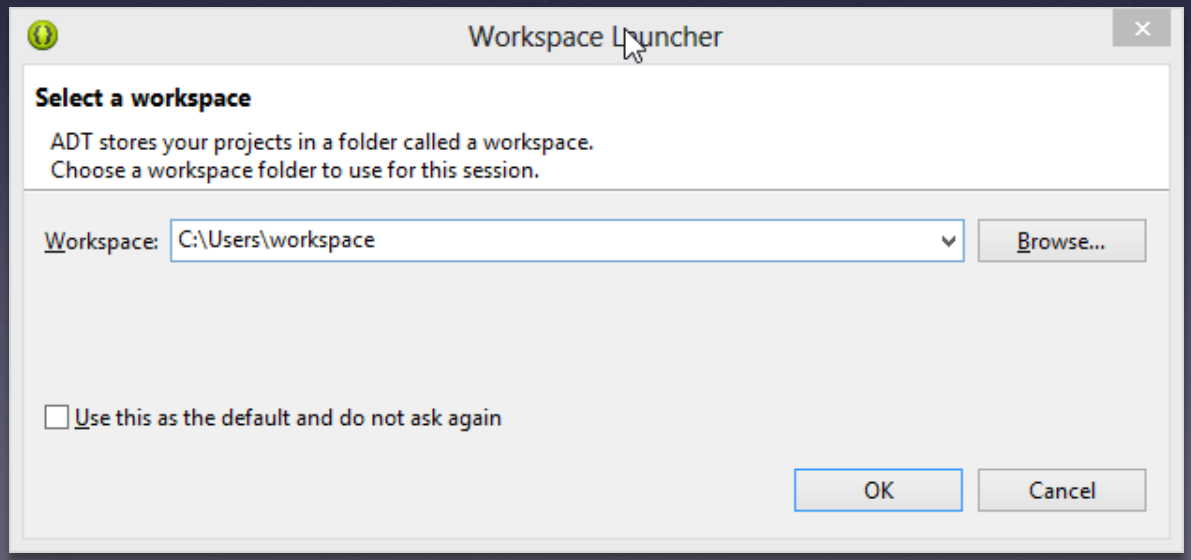
第一章

認識IDE與SDK

建立WORKSPACE

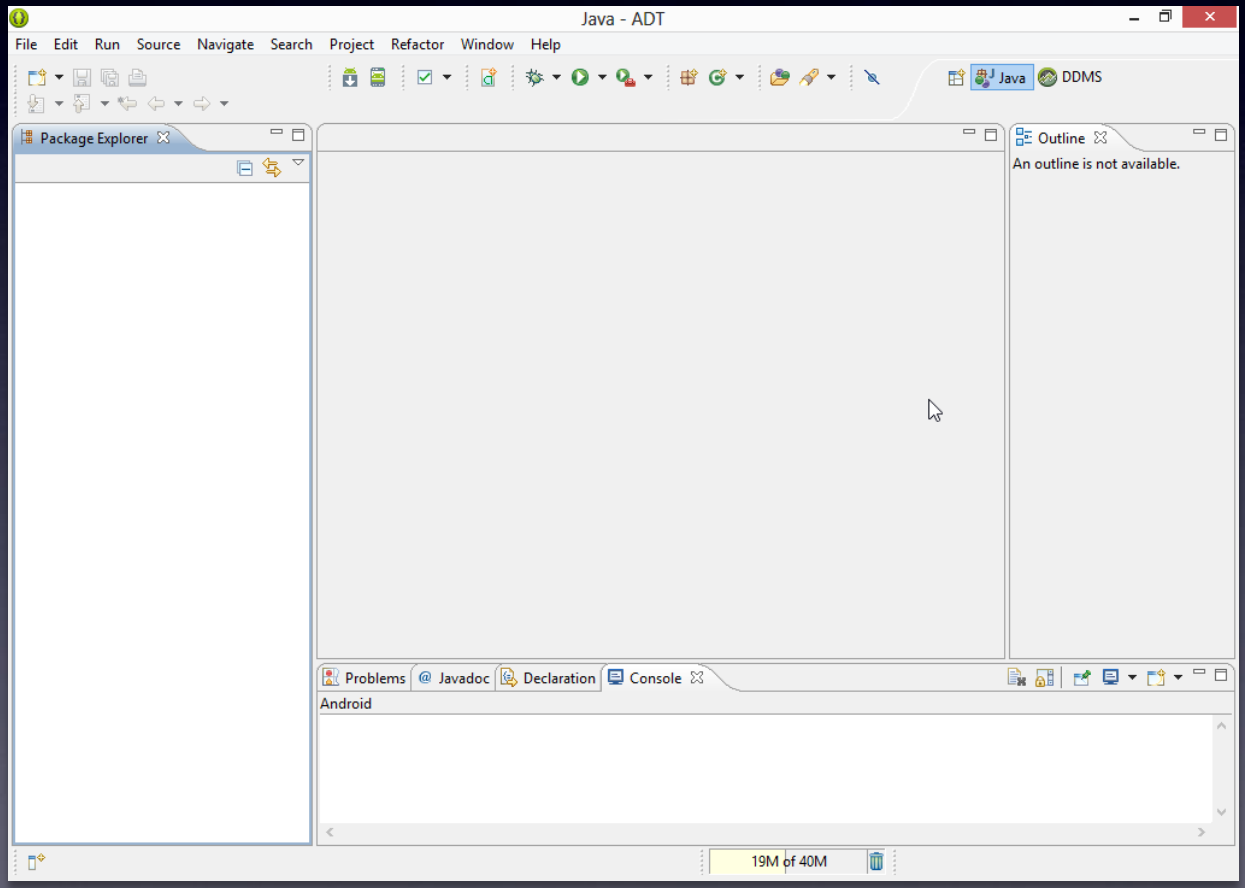
開啟第一個WORKSPACE

- Eclipse是以專案型式來管理程式碼
- workspace表示眾多專案集合的地方
- 可以針對不同的專案，或不同的時間，建立不同的workspace



開啟第一個WORKSPACE

- 恭喜你，開啟成功了



ECLIPSE與ADT操作說明

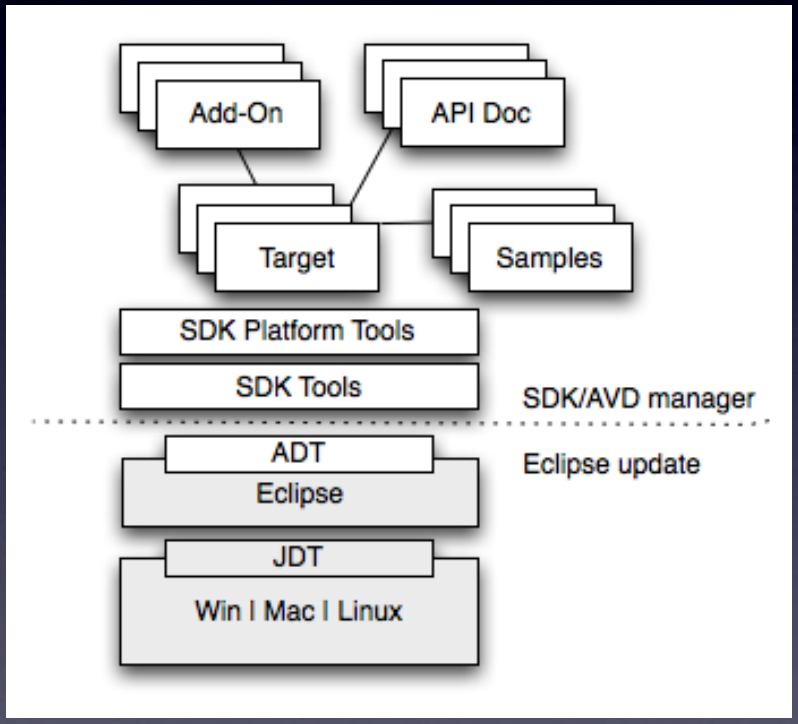

Android ADT
工具區
專案編譯
除錯工具
切換模式
Java - ADT
File Edit Run Source Navigate Search Project Refactor Window Help
Java
DDMS
Package Explorer
Outline
An outline is not available.
專案
列表區
程式
編輯區
Problems
Javadoc
Declaration
Console
Android
[2013-03-11 22:12:14 - SDK Manager] Deleting file C:\Program Files (x86)\Android\sdk\.android\avd\android4.
[2013-03-11 22:12:14 - SDK Manager] Deleting folder C:\Program Files (x86)\Android\sdk\.android\avd\andro:
[2013-03-11 22:12:14 - SDK Manager] AVD 'androi4.2.2' deleted.
of 40M
資訊輸出區

ECLIPSE基本介紹

- Android ADT工具區
  - 由Google提供給eclipse的plugin
  - 可在eclipse內連結模擬器、Android SDK管理器
- 切換模式
  - Android開發常用模式
    - Java模式
    - Debug模式
    - DDMS模式

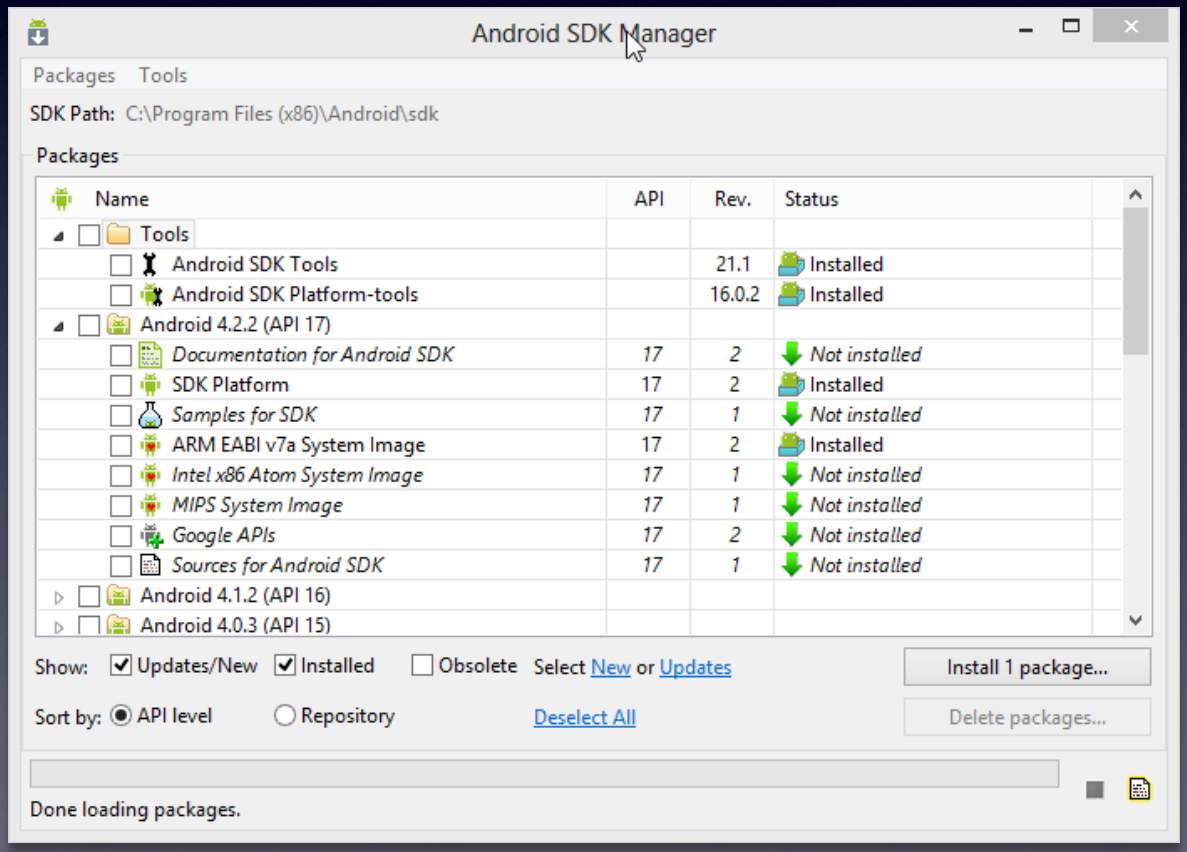
什麼是ADT

- Android Development Toolkit
- Android開發工具原本分為
  - Eclipse 負責開發程式碼
  - Android SDK Manager負責管理Android SDK和模擬器
- ADT整合兩者，在eclipse就可以連結SDK Manager



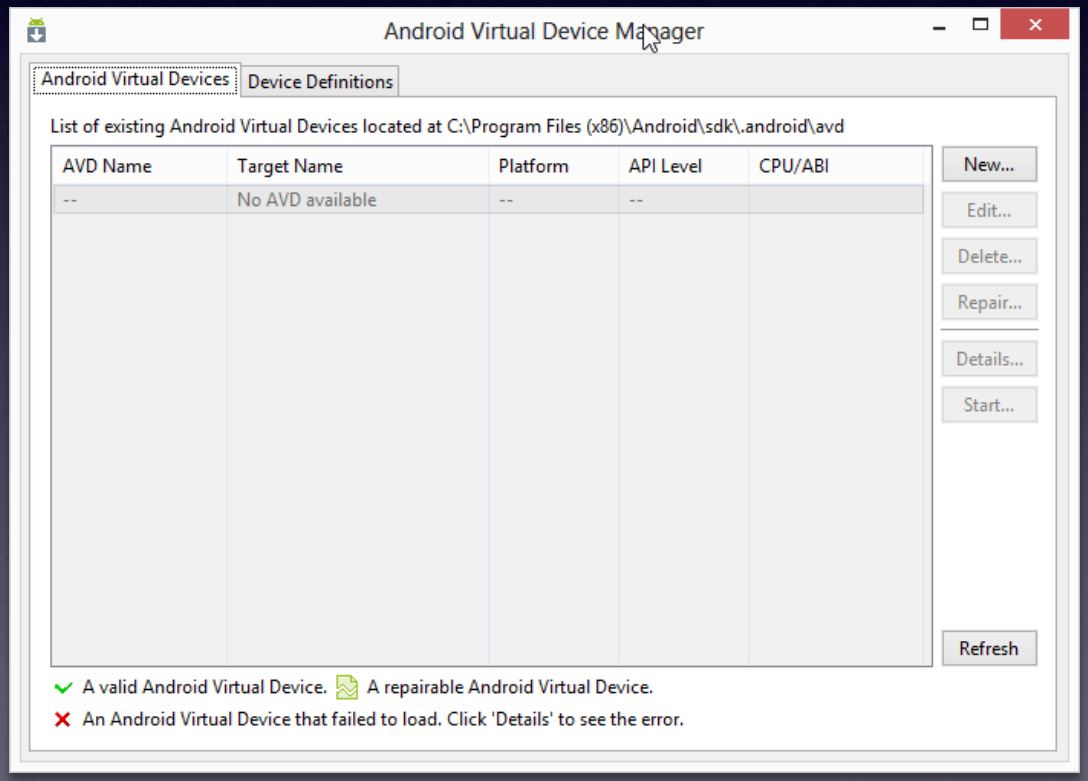
打開SDK MANAGER

- 點選Eclipse上方的ADT工具區圖示
- 可以勾選要下載的SDK版本來進行開發



打開AVD MANAGER

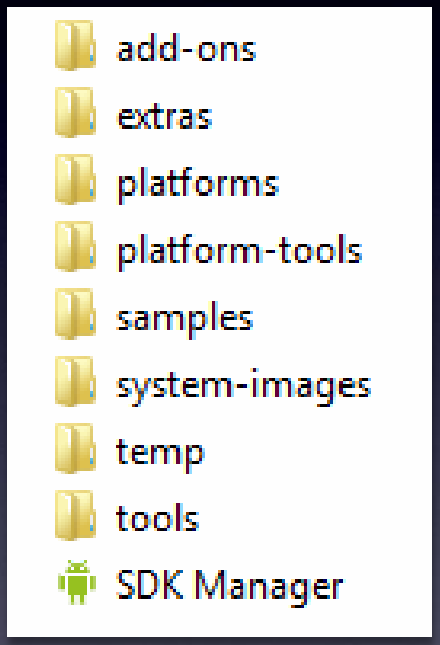
- AVD: Android Virtual Device
- Eclipse上方的ADT工具區圖示



ANDROID SDK架構

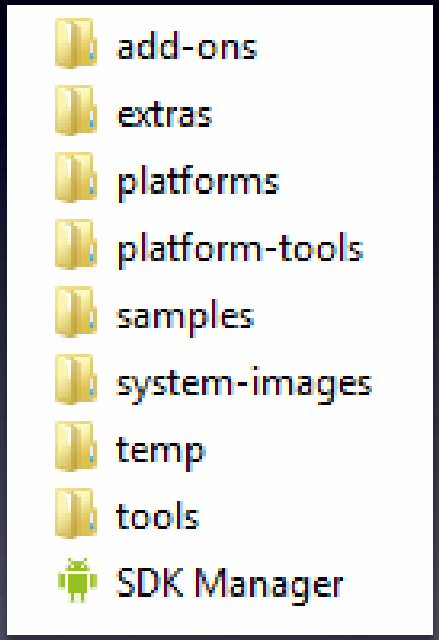
ANDROID SDK架構

- SDK Manager.exe
  - Android SDK管理器
- tools資料夾
  - Android執行時所需執行檔和函式庫
- samples資料夾
  - 依照不同target而提供的範例程式碼



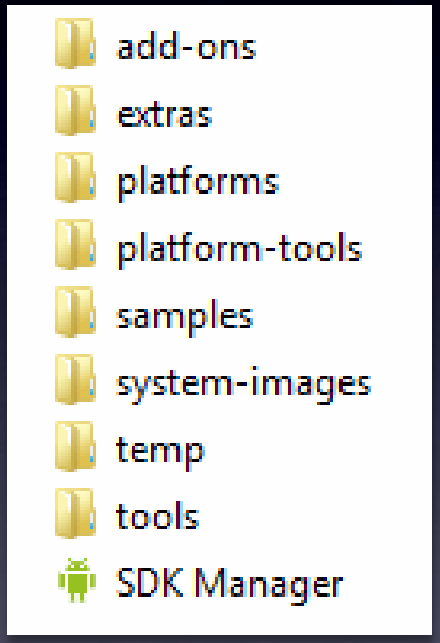
ANDROID SDK架構

- platform-tools
  - 編譯android和除錯時所需的執行檔和函式庫
- platforms
  - 不同版本的android模擬器的資料及映像檔



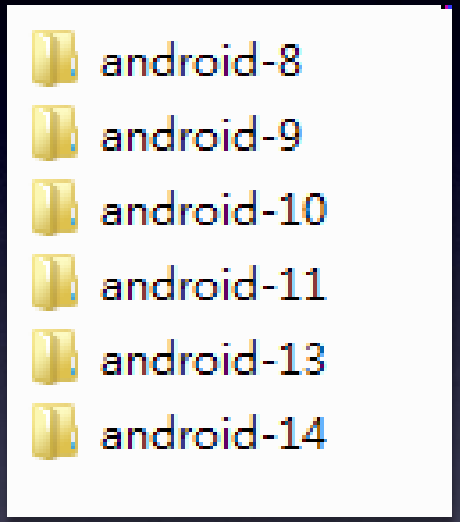
ANDROID SDK架構

- extras
  - 附加的函式庫，例如android手機的usb驅動程式、Marketing billing的防護函式庫
- add-ons
  - 其他的插件，像是Google map的library等等



ANDROID SDK架構

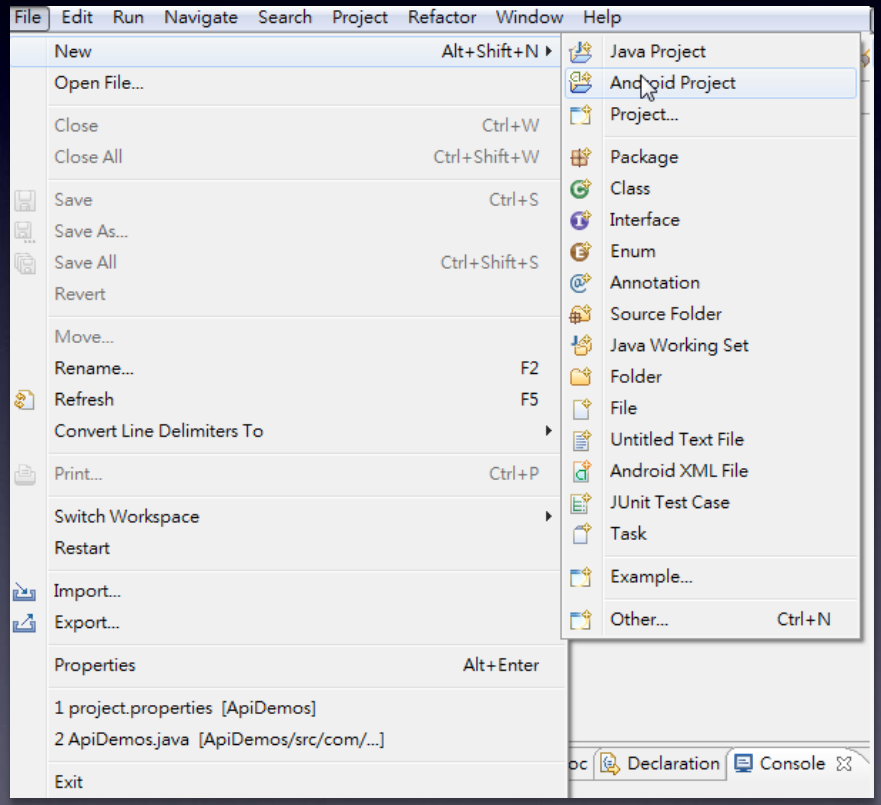
- 在platforms、samples、add-ons等資料夾下，都會依照android不同的target來分類
- android-<版本號>



建立新專案

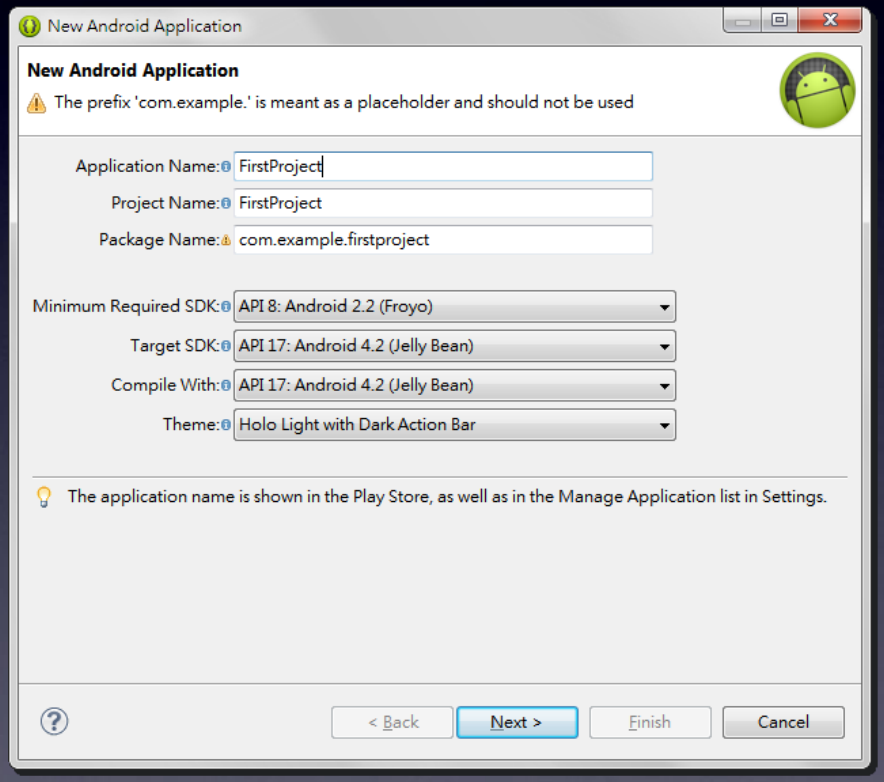
建立新專案

- 選擇上方選單
  File→New→Android Project



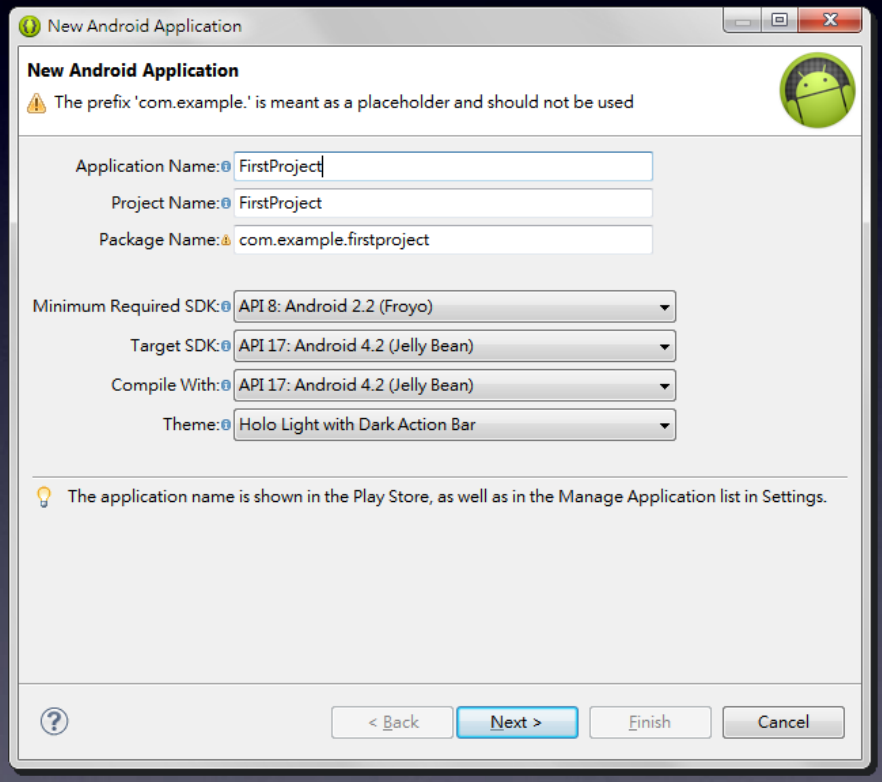
建立新專案

- Application Name
  - 安裝後呈現的名稱
- Project Name
  - 在workspace的名稱
- Package Name
  - 要獨立的名稱
- Minimum Require SDK
  - 可執行App的最低Android版本



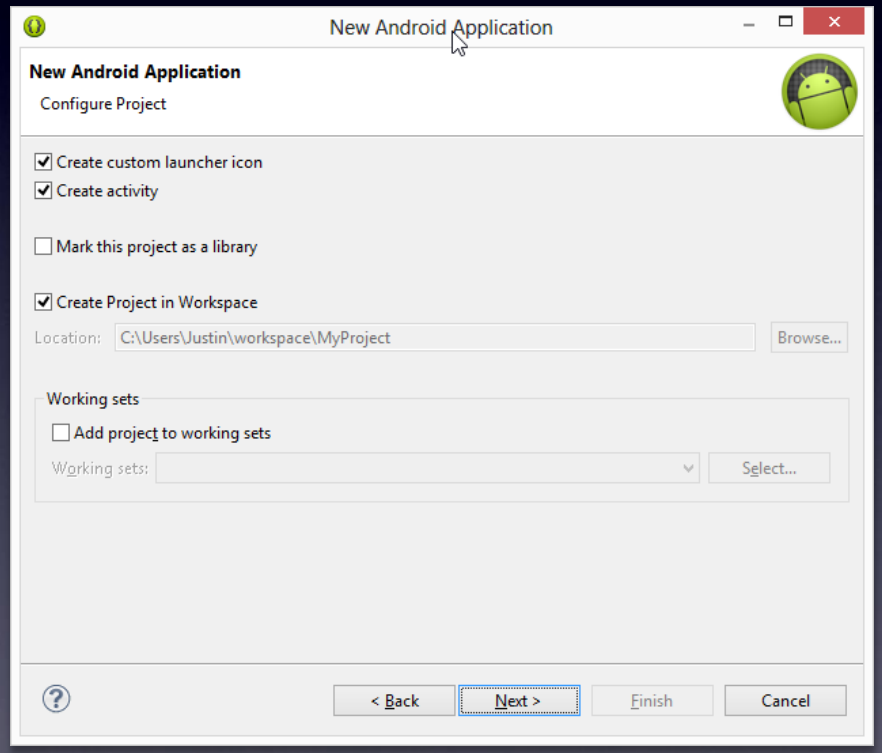
建立新專案

- Target SDK
  - App最重點執行的Android版本
- Compile With
  - 使用哪個版本的SDK來編譯
- Theme
  - 應用程式的佈景主題



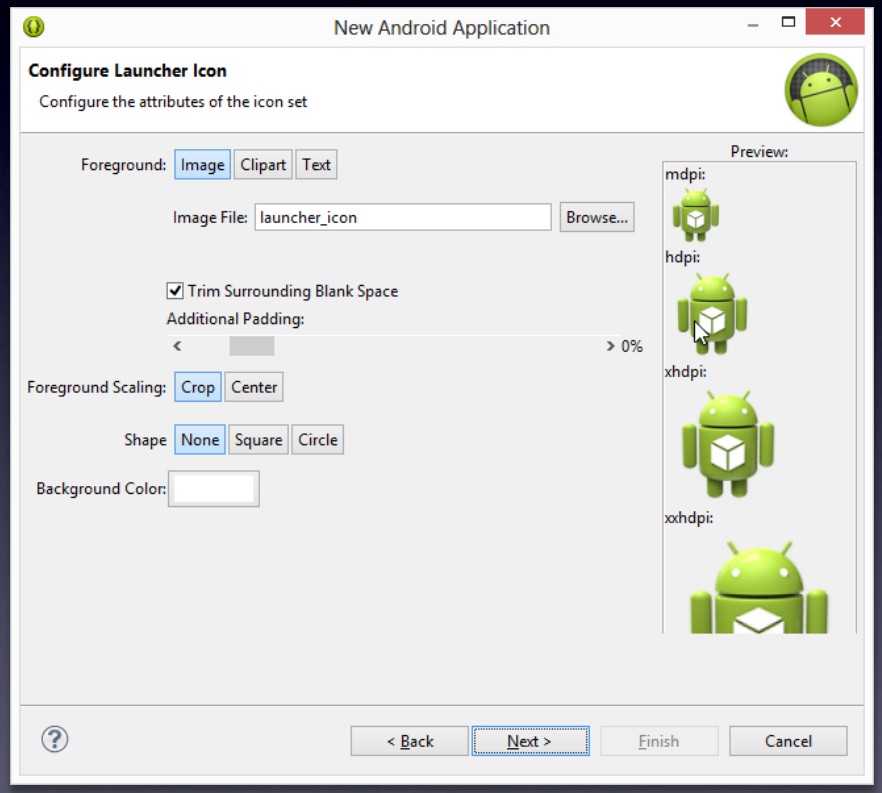
建立新專案

- Create custom launcher icon
  - 可以藉由工具建立app的圖示
- Create activity
  - 建立最基本的畫面
- Create Project in Workspace
  - 建立的專案是否擺放在workspace中



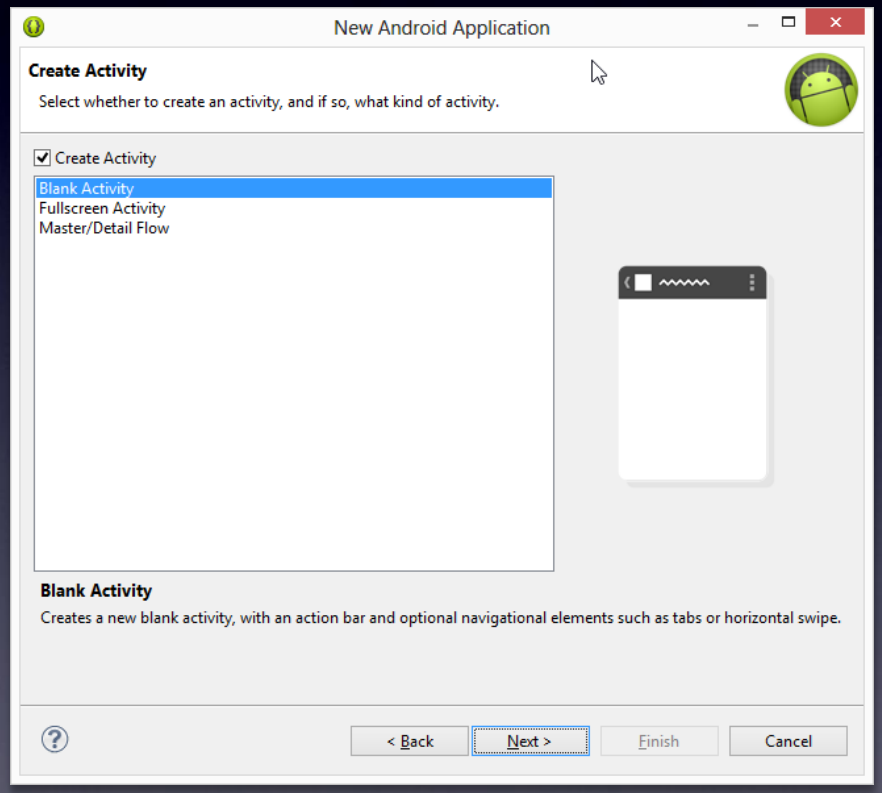
建立新專案

- Create Custom launcher icon
  - 建立客製化的app圖示



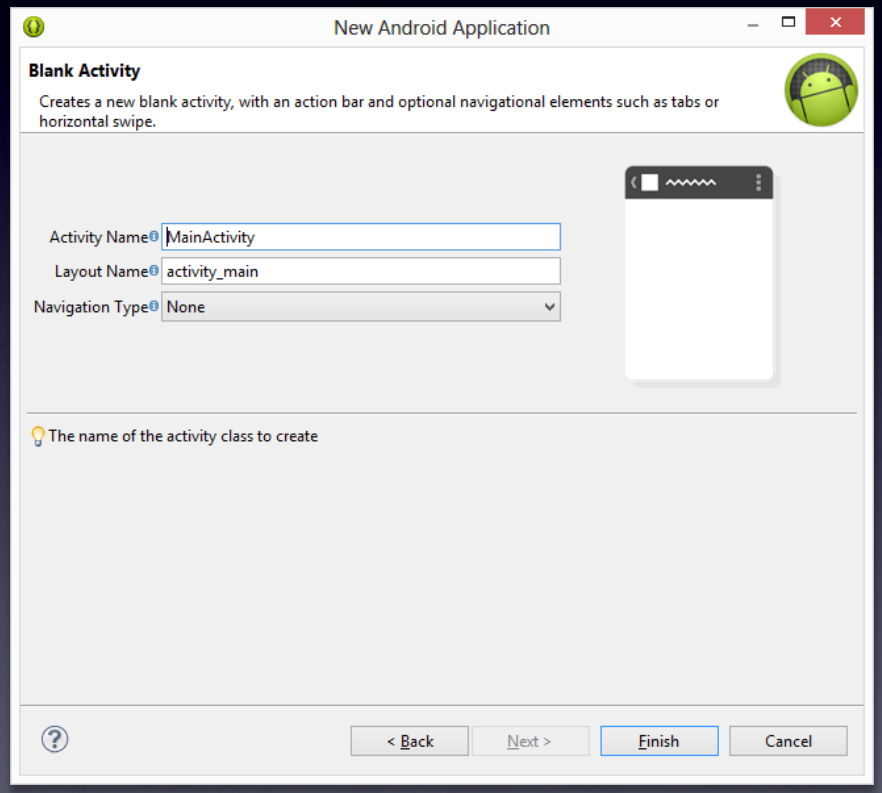
建立新專案

- Create Activity
  - 建立基本畫面的類型

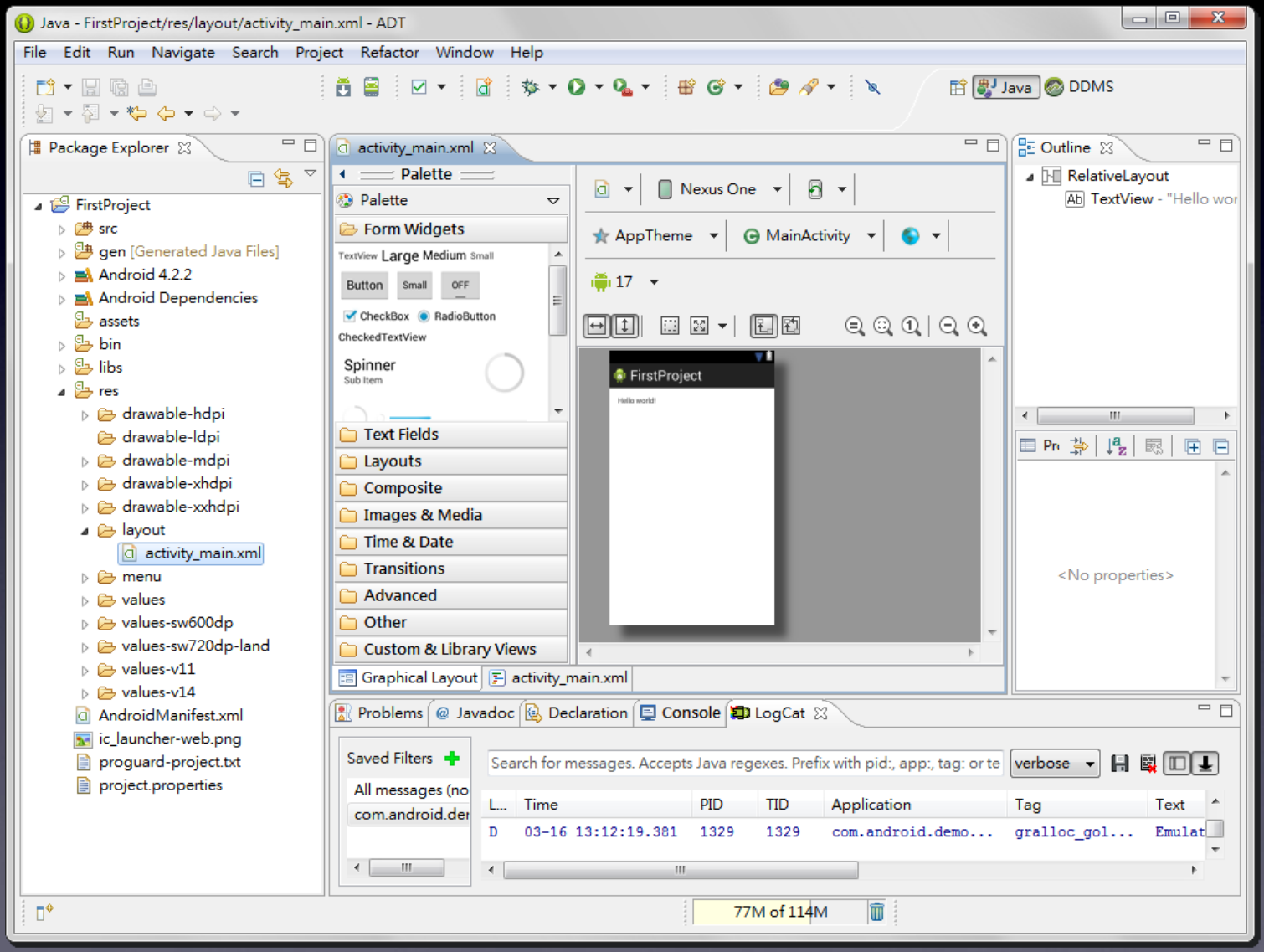


建立新專案

- Activity Name
  - 畫面會產生一個Java檔，檔案的名稱
- Layout Name
  - 介面定義檔的名稱
  - Java檔要呈現的畫面會以XML定義



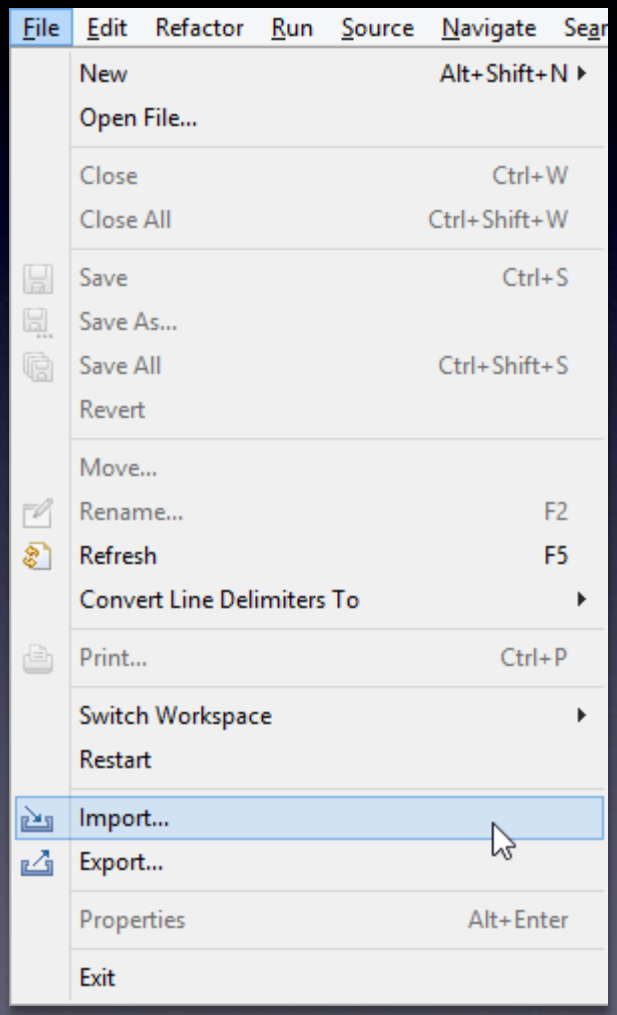
建立完成


Java - FirstProject/res/layout/activity_main.xml - ADT
File Edit Run Navigate Search Project Refactor Window Help
Java DDMS
Package Explorer
FirstProject
src
gen [Generated Java Files]
Android 4.2.2
Android Dependencies
assets
bin
libs
res
drawable-hdpi
drawable-ldpi
drawable-mdpi
drawable-xhdpi
drawable-xxhdpi
layout
activity_main.xml
menu
values
values-sw600dp
values-sw720dp-land
values-v11
values-v14
AndroidManifest.xml
ic_launcher-web.png
proguard-project.txt
project.properties
activity_main.xml
Palette
Form Widgets
Text Fields
Layouts
Composite
Images & Media
Time & Date
Transitions
Advanced
Other
Custom & Library Views
Nexus One
AppTheme
MainActivity
17
FirstProject
Graphical Layout
activity_main.xml
Outline
RelativeLayout
TextView - "Hello wor
<No properties>
Problems
Javadoc
Declaration
Console
LogCat
Saved Filters
All messages (no
com.android.der
Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or te
verbose
L... Time PID TID Application Tag Text
D 03-16 13:12:19.381 1329 1329 com.android.demo... gralloc_gol... Emulat
77M of 114M

匯入專案

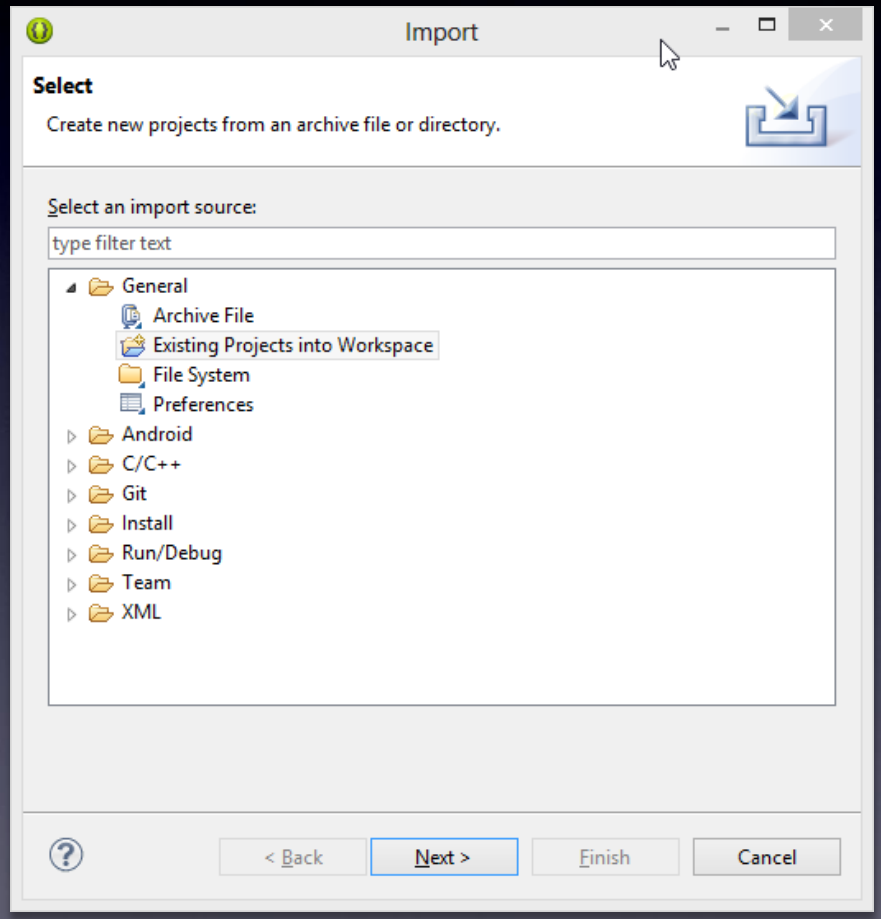
匯入專案

- File -> Import…
- 這功能非常常用，網路上抓的Android例子幾乎都可以用這方式開啟



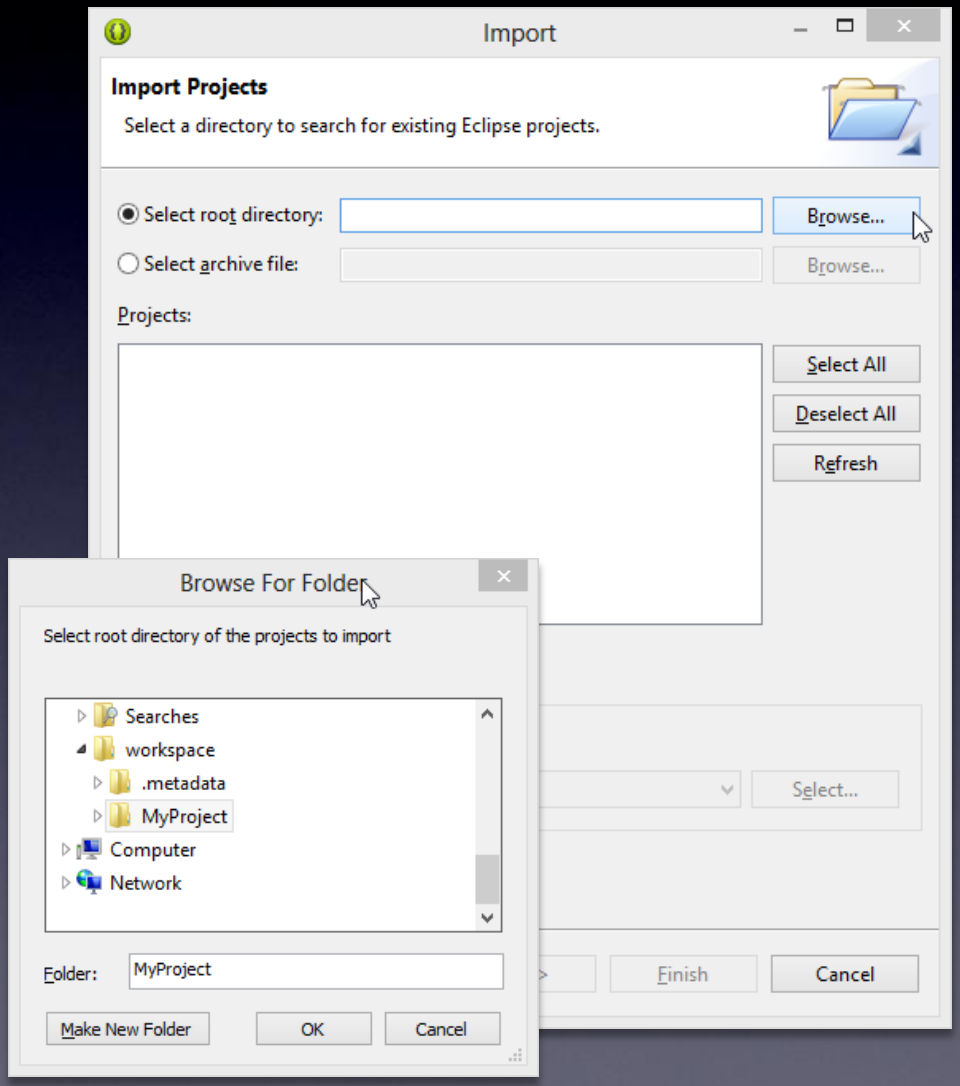
匯入專案

- General -> Existing Projects into Workspace
- Next



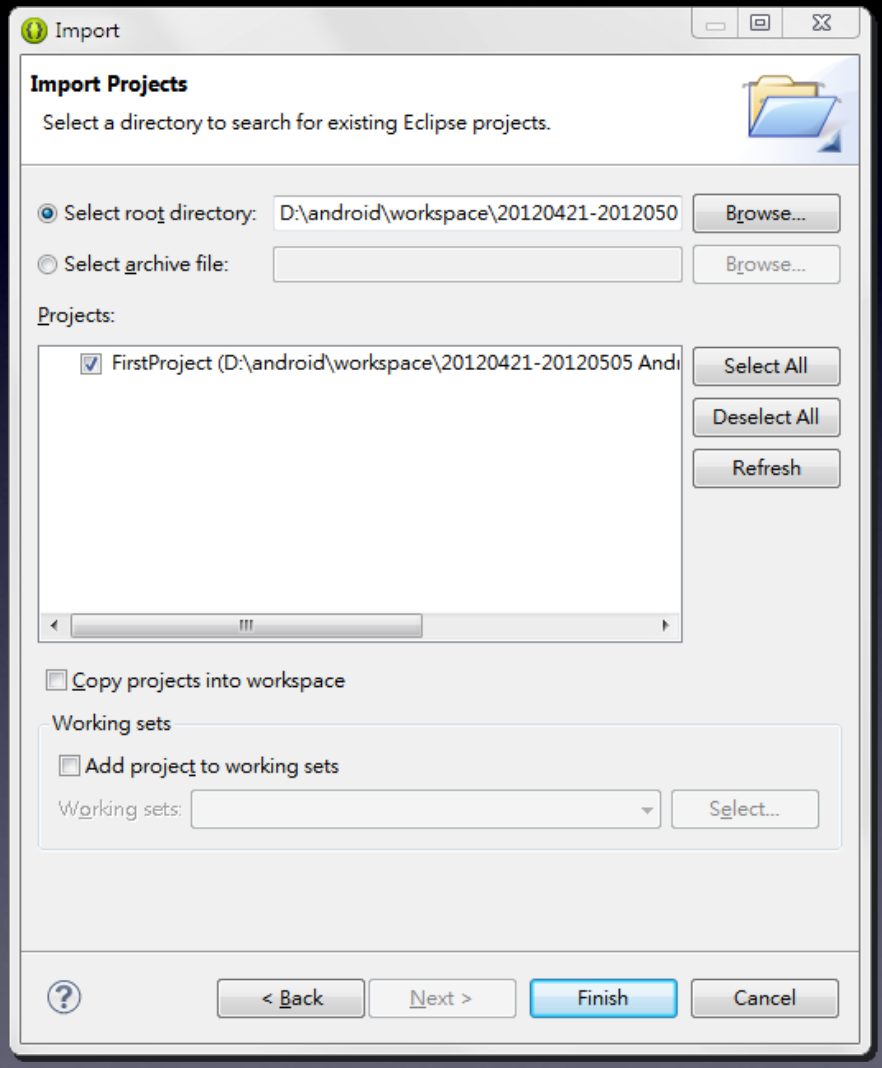
匯入專案

- Browse
- 在Browser For Folder中找到剛剛建立的Project位置
- OK



匯入專案

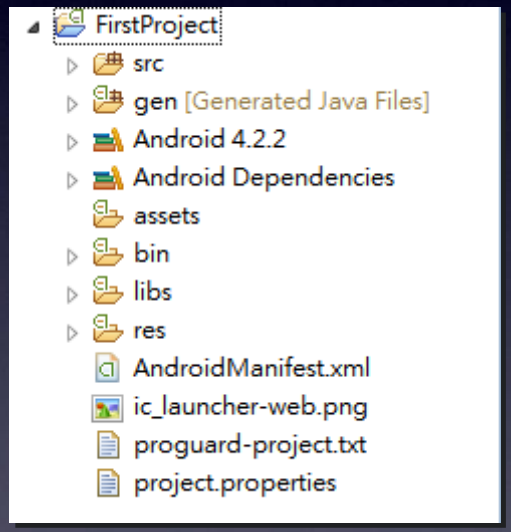
- 若專案可以匯入就會出現在列表
- 若選擇的資料夾下有超過一個Project，就會有多個專案在列表
- Finish



專案架構

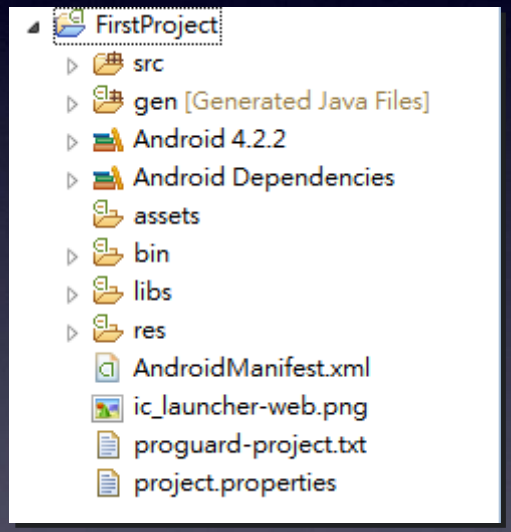
專案架構

- src
  - 專案原始碼的擺放位置
- gen
  - 編譯後自動產生Android需要用的程式碼的擺放處
- Android 4.2.2
  - Android SDK函式庫



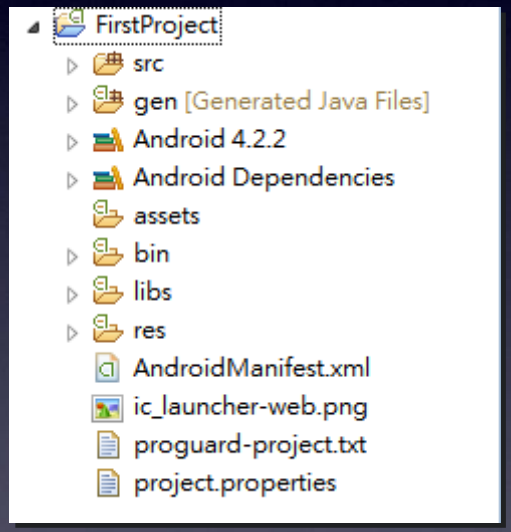
專案架構

- Android Dependencies
  - 編譯Android需要用的其他的官方library
- assets
  - 專案額外用的資源檔
- bin
  - 編譯後產生目的檔和APK的地方



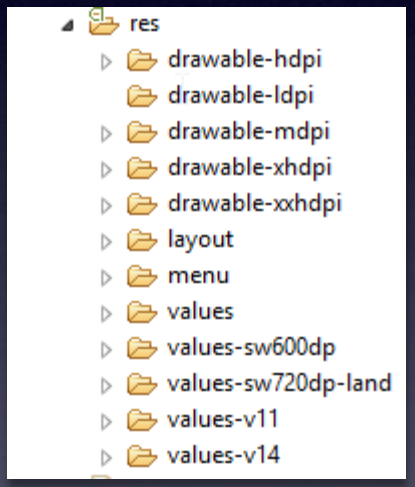
專案架構

- libs
  - 編譯專案自己客製的 library
- res
  - 資源檔擺放處
- AndroidManifest.xml
  - Android app屬性定義檔



專案架構

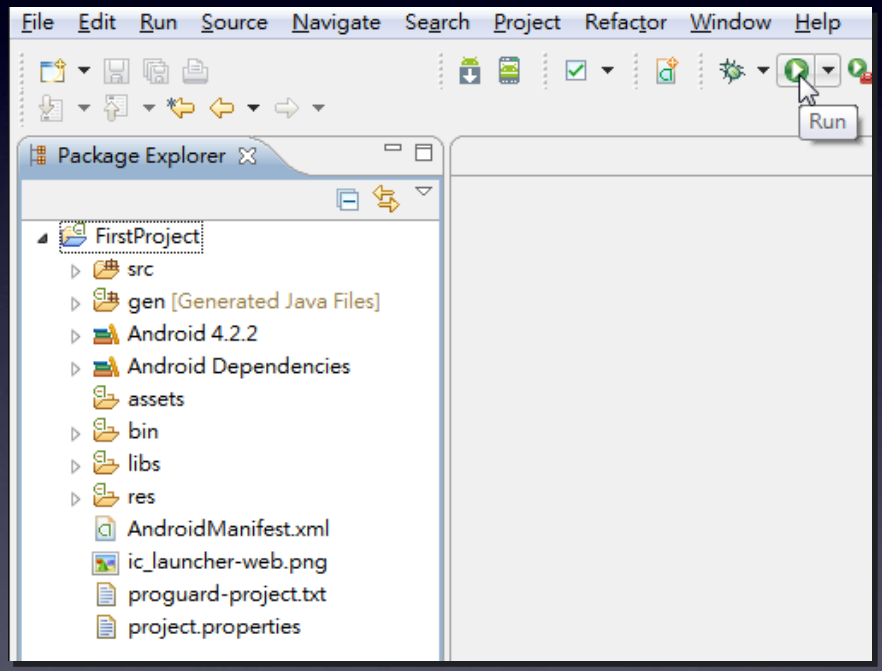
- Res資料夾
  - drawable
    - 圖檔
  - layout
    - 介面定義檔
  - menu
    - 選單定義檔
  - values
    - 字串
    - 程式參數定義檔
    - 佈景主題



執行專案

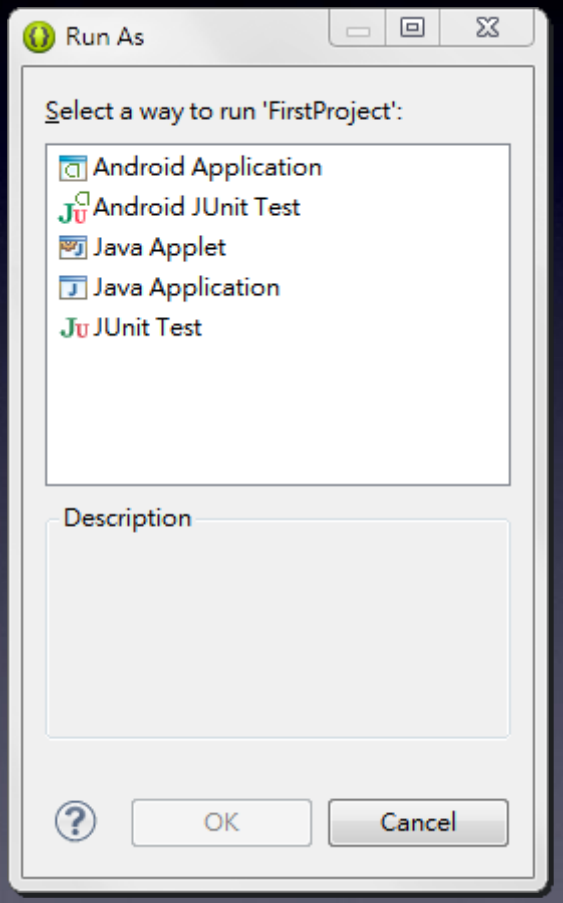
執行專案

- 選擇要執行的專案
- 按下右圖中右上方的按鈕 



執行專案

- 選擇Android Application
- OK



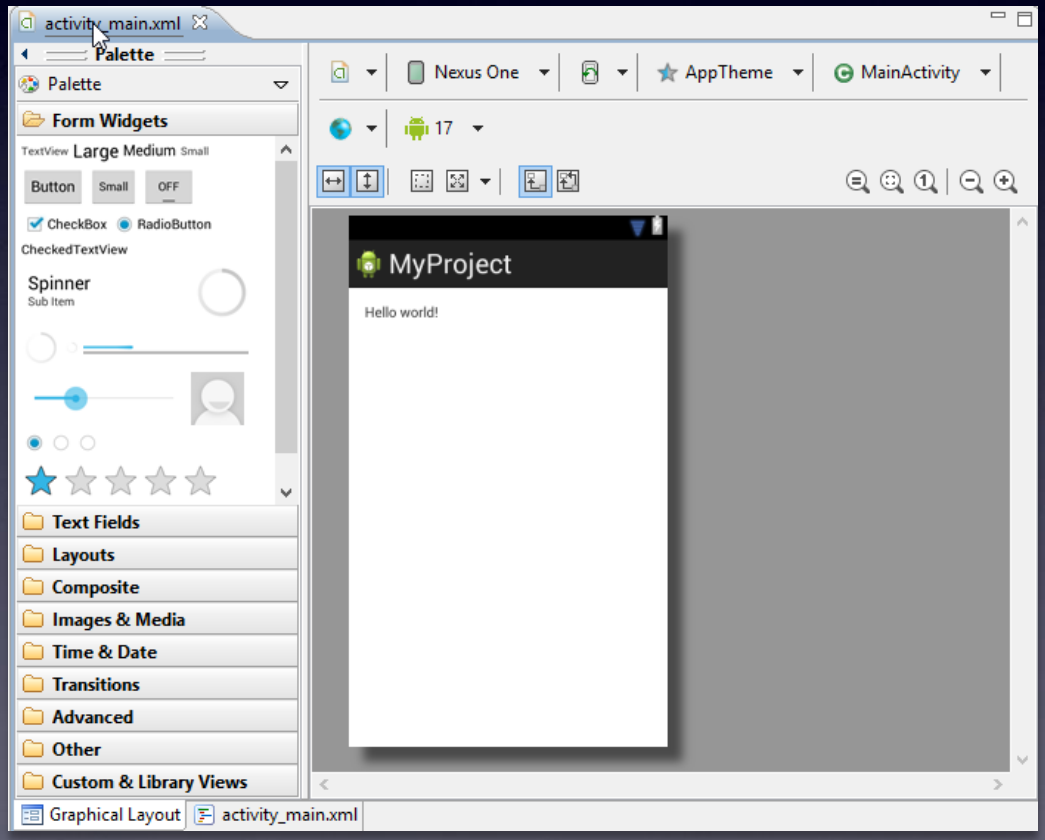
第二章

基本介面設定與互動

介面(LAYOUT)定義檔

介面定義檔

- 打開專案資料夾res/layout中的activity_main.xml
- 下圖為Android Layout編輯器



介面定義檔

- Android的介面是由XML撰寫
- 可以在下方切換模式
  - Graphical Layout (所見即所得編輯器)
  - activity_main.xml (xml原始檔)
- 現在所見即所得編輯器已經非常完備
- 但還是要學習基礎的介面XML語法

常用基本元件

對應專案course/UsefulWidgetLayoutLogin

常用基本元件

- 常用基本元件屬性
  - `id`
    - 給每個建立的元件一個索引值
    - @+id/<索引值名稱>
  - `layout_width, layout_height`
    - 元件長寬
  - `gravity`
    - 元件內容物的排列方式
    - 如TextView內的文字、LinearLayout內的元件

常用基本元件

- 常用基本元件屬性
  - enable
    - 元件是否可以使用
  - clickable
    - 元件是否可以點選
  - padding<位置>
    - 位置包含Top, Left, Right, Bottom
    - 元件周邊的留白與元件內容之間的間細

常用基本元件

- 常用基本元件屬性
  - `layout_margin<位置>`
    - 位置包含Top, Left, Right, Bottom
    - 元件與父元件之間的
  - `visibility`
    - 元件是否顯示
    - visible, invisible, gone

常用基本元件

- Button
  - `text`
    - 要呈現的文字
  - `onClick`
    - 與程式有關，按下按鈕後要連結的程式方法

常用元件

- ImageView
  - 負責顯示圖片
  - src
    - 值為圖片，@drawable/<圖檔名稱>
    - 圖片要擺在res/drawable資料夾下或res/drawable-<不同條件>
      - 這是資源選擇性，晚點討論

course/LoginExample

呈現畫面

程式讓畫面顯示

- 打開專案的MainActivity.java
- extends Activity
  - Android應用程式頁面的基本單位
- 關鍵程式碼1
  - `onCreate()`
  - 當Activity要出現之前，會被Android系統呼叫的方法
  - 這是你一定要實作的方法
  - 關於Activity的概念，後續會再說明

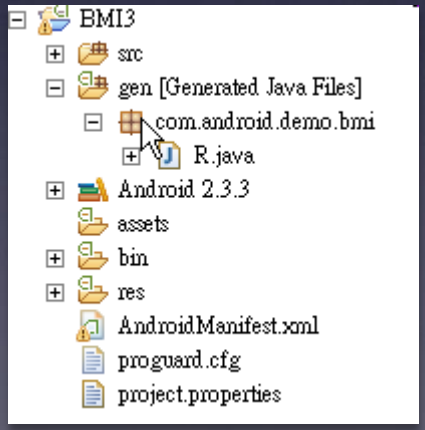
程式讓畫面顯示

- 關鍵程式碼2
  - `setContentView(R.layout.activity_main);`
  - 繼承Activity的class都可以使用
  - 專門設定呈現畫面的方法

- 甚麼是R.layout.activity_main?

程式讓畫面顯示

- 在Android中，會自動將資源檔以它的類型轉為class，再以索引值轉變為特殊的變數，每個變數都可以讓我們存取到該資源
- 這個Android自動產生的檔案稱為R.java，它存在於/gen/<應用程式的package>/R.java



程式讓畫面顯示

- 我們在程式中要存取資源檔，就是使用 R.<資源類型>.<索引值>
- 例如
  - R.layout.main
  - R.string.hello
  - R.drawable.launcher
  - R.id.btn_ok

程式讓畫面顯示

- setContentView()會依照傳入的參數來將整個XML的介面元件轉為Java的物件

- 那我要切換應用程式畫面時，就使用setContentView()設定其他頁面就可以了嗎
  - 千萬不要!!
  - 一個Activity最好只呼叫一次setContentView()
  - 要換頁面就使用Activity切換 (後續會提到)

透過ID取得VIEW

- 關鍵程式碼3
  - `findViewById()`
  - 取得由XML轉為Java物件的方法
- 需藉由R.id的資源來取得
- 什麼是R.id資源?

透過ID取得VIEW

- 例如
  - activity_main.xml

```
<TextView
        android:id="@+id/title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

  - MainActivity.java
    - `TextView title = (TextView) findViewById(R.id.title);`

透過ID取得VIEW

- findViewById()
  - 必須在setContentView()之後才能使用
  - 藉由索引值找到物件的實體
  - 不過回傳的是通用型態View，必須得強制轉型

接收事件

- 如何知道Button按到?
  - 使用OnClickListener來接收事件
- 使用方法

```
OnClickListener loginListener = new OnClickListener() {
    public void onClick(View v) {
    }
}
```

- onClick()中的程式碼，就是按鈕按到後要做的事情
- 參數傳入的就是被按到的View

接收事件

- 其他事件
  - 長按事件

```
new View.OnLongClickListener() {
        public void onLongClick(View v) {
        }
}
```

  - 指觸事件

```
new View.OnTouchListener() {
        public void onTouch(View v, MotionEvent e) {
        }
}
```

  - 按鍵事件

```
new View.OnKeyListener() {
    public void onKey(View v, int keyCode,  KeyEvent e) {
    }
}
```

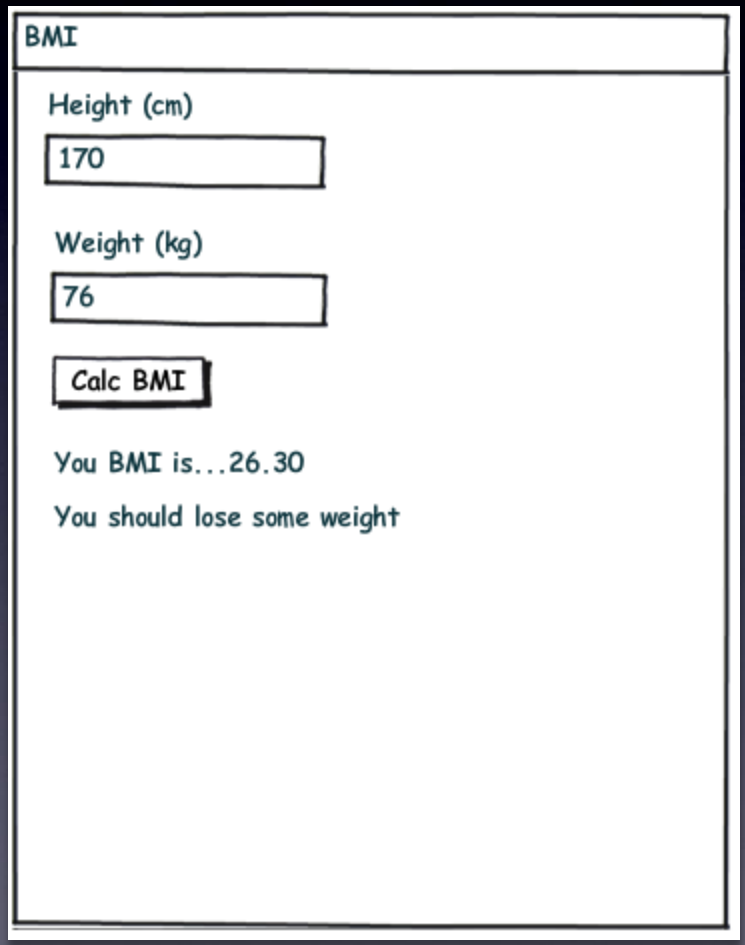
結束ACTIVITY

- 想要結束目前的頁面(Activity)該怎麼做？

- `finish()`
  - 只要在Activity內呼叫，Activity就會關閉

練習

- 製作一個BMI的計算程式
- BMI的算法
  - 體重(公斤) / 身高(公尺)的平方
  - BMI > 24過胖
  - BMI < 18.5 太瘦



course/ToastExample

畫面提示

畫面提示

- Toast是短暫性跳出通知使用者的一種方式
  - 只能顯示資訊，使用者無法互動

- 常用在通知APP狀態
  - 登入失敗、資料未填

畫面提示

- 使用方法
  - `Toast.makeText(Activity實體, 要顯示的字, 顯示時間長短).show();`

  - Activity實體：就是Activity的名稱加上.this
  - 要顯示的字：字串或使用R.string
  - 顯示時間長短：Toast.LENGTH_SHORT或Toast.LENGTH_LONG
  - show() 呼叫後顯示

第三章

應用程式配置

應用程式配置設定檔

應用程式配置設定檔

- 前面有說到每個App都可包含多個載體
- 而載體在App安裝到裝置時，必須得向Android Framework註冊
- 註冊的內容就是寫在應用程式配置設定檔AndroidManifest.xml中
- 打開MyProject的AndroidMainfest.xml

應用程式配置設定檔

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.demo.project"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="4" android:targetSdkVersion="17"/>

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".Main"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

應用程式配置設定檔

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.demo.project"
    android:versionCode="1"
    android:versionName="1.0" >

</manifest>
```

- package="com.android.demo.project"
  - 應用程式的Package
- android:versionCode
  android:versionName
  - 應用程式開發的版號
  - versionCode是給開發者看的版本號
  - versionName是給使用者看的版號

應用程式配置設定檔

```
<uses-sdk android:minSdkVersion="4" android:targetSdkVersion="17"/>
```

- 最低相容的SDK版本與目標版本的定義
  - 還記得API Level嗎？
- 若minSdkVersion要是10，那麼API Level 10以下的手機全部都不能安裝

應用程式配置設定檔

```
<application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name" >

</application>
```

- android:icon
  - 定義了應用程式的圖示
- android:label
  - 定義了應用程式呈現在選單中的名稱
- 載體要定義在<application></application>之間

應用程式配置設定檔

```
<activity
      android:name=".Main"
      android:label="@string/app_name" >
</activity>
```

- 應用程式中要被使用的Activity都要使用<activity>定義在<application>中

- 一個應用程式有多個activity就必須定義多個<activity>

應用程式配置設定檔

```
<activity
      android:name=".Main"
      android:label="@string/app_name" >
</activity>
```

- android:label
  - 定義activity的名稱
  - 會顯示在應用程式上方的標題欄上

應用程式配置設定檔

```
<activity
      android:name=".Main"
      android:label="@string/app_name" >
</activity>
```

- android:name
  - 定義activity的class名稱
    - 這要搭配<manifest>中的屬性package
    - 本例子中package是com.android.demo.project
    - Main剛好是擺在com.android.demo.project中
    - 所以寫.Main的意思就表示這個activity的完整名稱是com.android.demo.project.Main

應用程式配置設定檔

```
<activity
      android:name=".Main"
      android:label="@string/app_name" >
</activity>
```

- android:name
  - 假設Main現在改到com.android.demo.project.activity的package之下
  - 那這邊就要改成android:name=".activity.Main"

應用程式配置設定檔

```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- <intent-filter>
  - 載體向Android Framework註冊的條件
- <action>
  - android:name
    可以把這個參數的值，當成是載體啟動的索引值

應用程式配置設定檔

```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- <category>
  - 定義載體的分類
  - android:name
    - 這邊使用內建的android.intent.category.LAUNCHER
    - 另一個比較常用的是android.intent.category.DEFAULT

應用程式配置設定檔

```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- action是android.intent.action.MAIN category是android.intent.category.LAUNCHER
  - 表示應用程式安裝後會有一個程圖示在應用程式列表中
  - 在開發時使用eclipse將應用程式放到模擬器(或手機)時，Android會自動啟動這個Activity

應用程式配置設定檔

```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- 一個Activity可以有多個<intent-filter>
- 一個<intent-filter>內也可以定義多個<action>和<category>

第四章

ACTIVITY

ACTIVITY概觀

ACTIVITY概觀

- 盡量保持每個Activity只負責一個畫面
- 要轉至其他畫面就使用Activity切換

- 依照是否需要與其它Activity交換資料來區分，Activity可分為兩種類型
  - 獨立的Activity
  - 相依的Activity

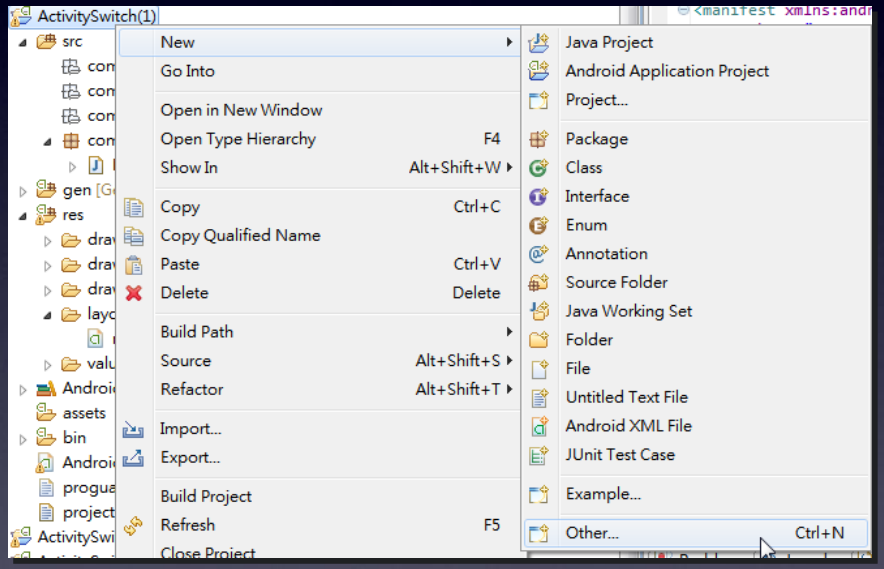
ACTIVITY概觀

- 獨立的Activity
  - 單純從一個螢幕跳到另一個螢幕，不涉及資料交換
- 相依的Activity
  - 需要與其它的Activity交換資料
  - 分為單向交換與雙向交換
    - 單向：資料由一個螢幕攜帶至另一個螢幕
    - 雙向：螢幕上的資料除了攜帶至另一個螢幕外，還會因為另一個螢幕的操作而改變，進而影響到原本螢幕的資料呈現

新增一個ACTIVITY

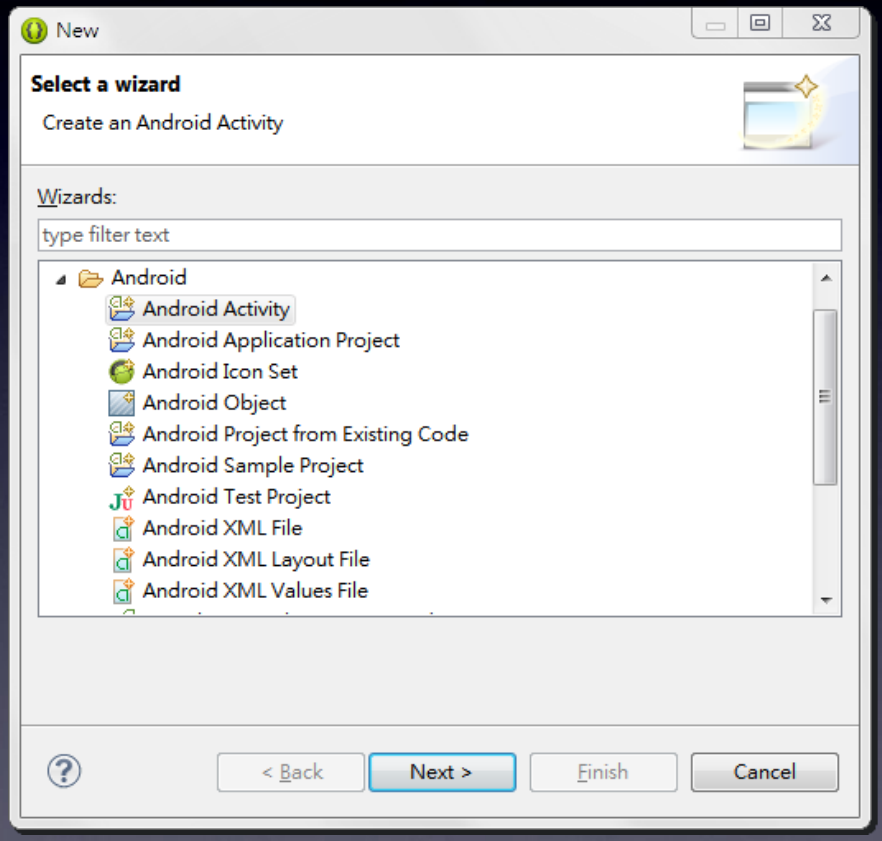
新增一個ACTIVITY

- 對專案按下滑鼠右鍵
- New
- Others



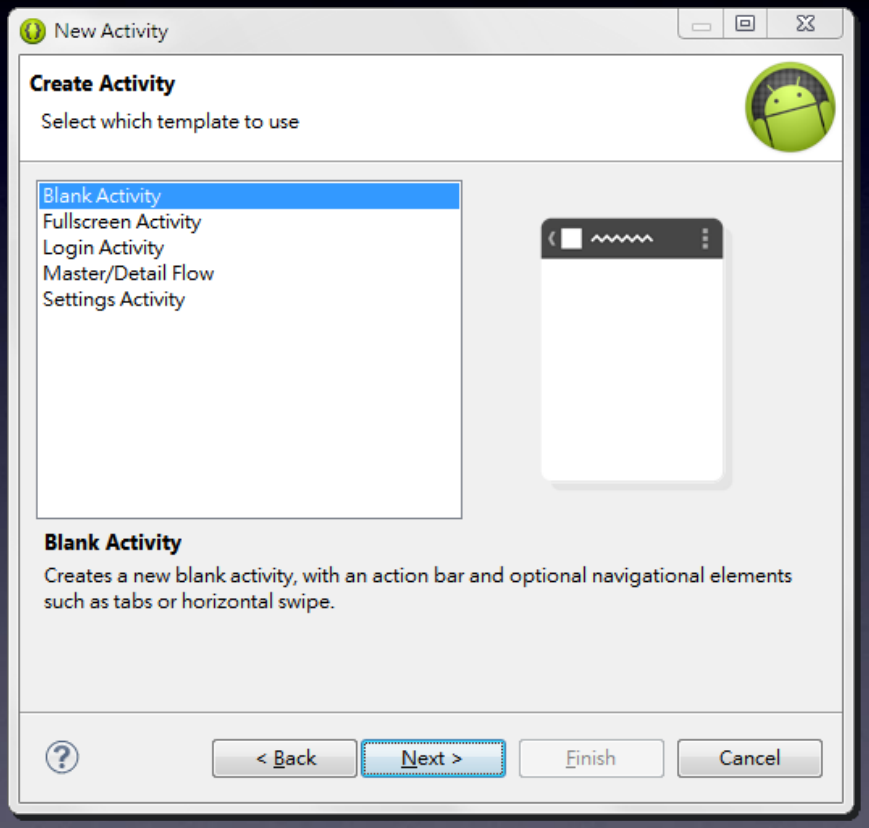
新增一個ACTIVITY

- 選擇Android的 Android Activity
- Next



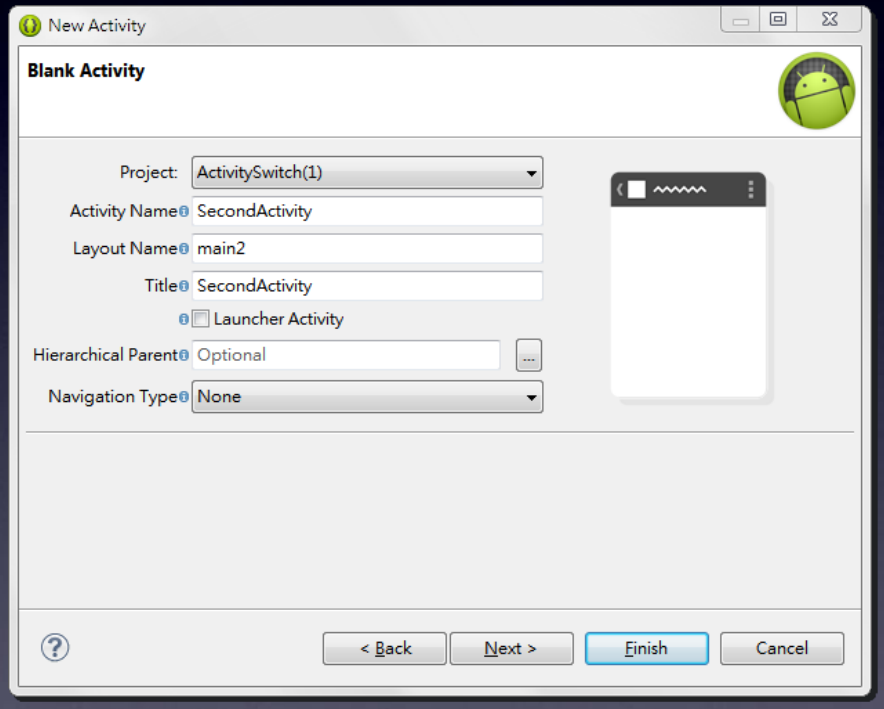
新增一個ACTIVITY

- 選擇Blank Activity
- Next



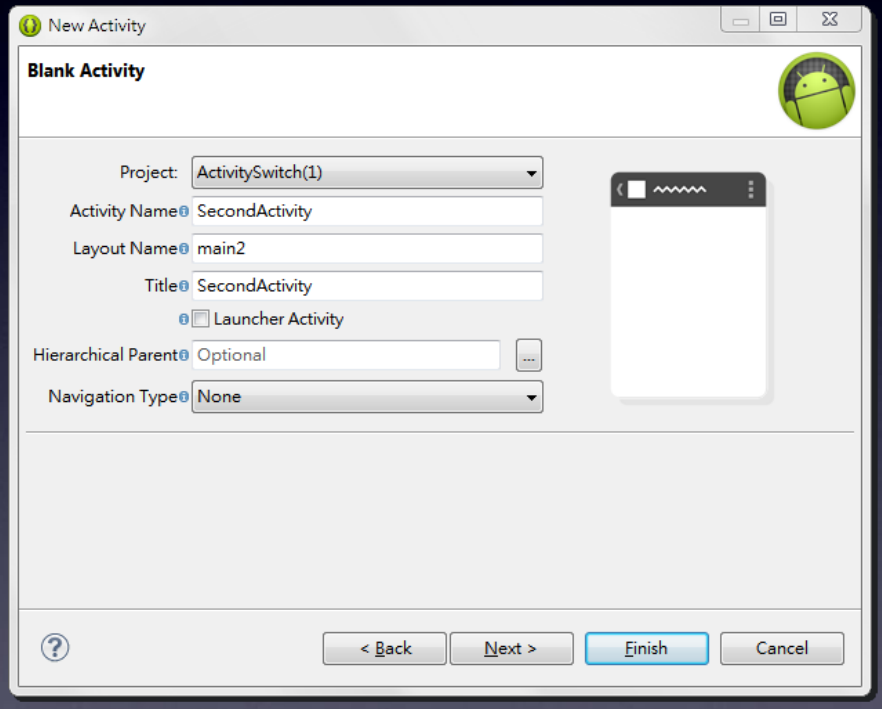
新增一個ACTIVITY

- Activity Name
  - 建立Java檔
- Layout Name
  - 介面設定檔的名稱
- Title
  - Activity的名稱
- Launcher Activity
  - 勾選表示這是程式進入的Activity



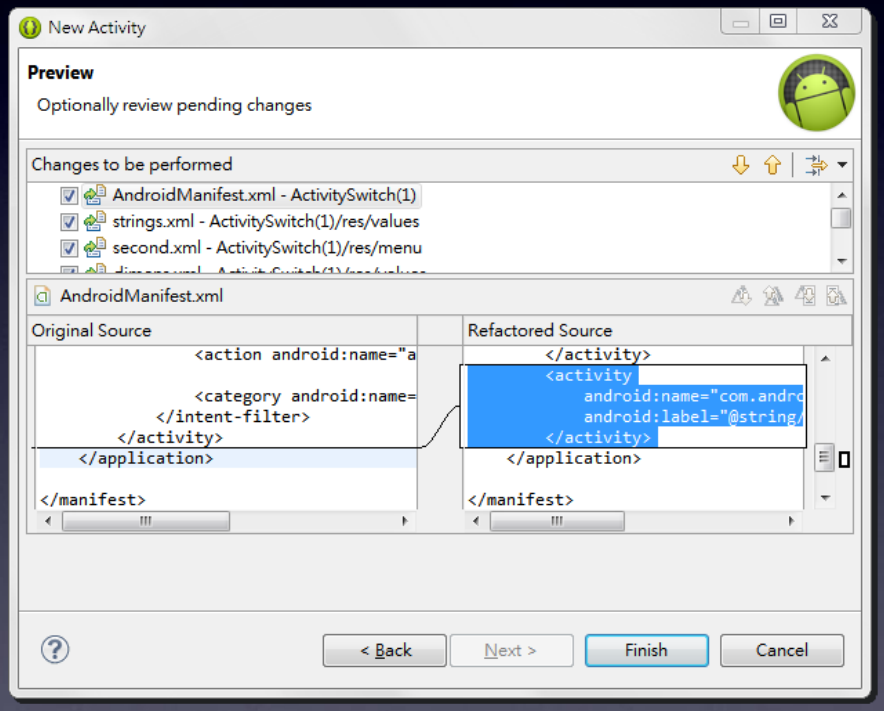
新增一個ACTIVITY

- Launcher Activity
  - 勾選表示這是程式進入的Activity
- Next

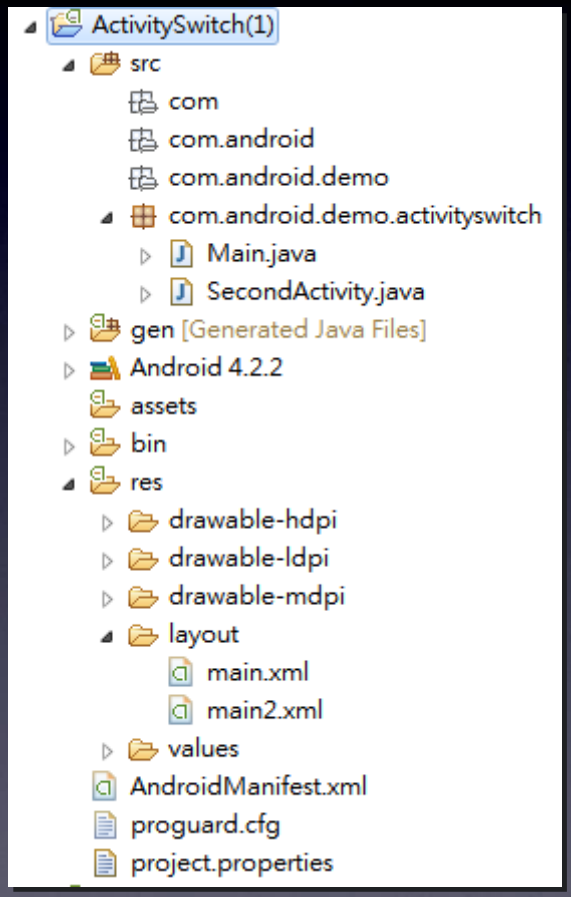


新增一個ACTIVITY

- 接下來ADT會自動將相關程式碼加入AndroidManifest.xml中
- Finish



新增一個ACTIVITY



- 建立完成
- 確認AndroidManifest.xml有新的Activity定義

course/ActivitySwitch(1)

切換至自訂的ACTIVITY

切換至自訂的ACTIVITY

- 與一般Intent啟動方式相同
- Intent.setClass(
  目前的Activity.this, 目標Activity.class)

- startActivity(Intent)
  - 一樣使用startActivity啟動

```
Intent intent = new Intent();
intent.setClass(Main.this, SecondActivity.class);
startActivity(intent);
```

使用ACTION切換

- 除了直接指定的方法外，也可以使用動作(Action)來切換
- 首先要在AndroidManifest.xml中指定Activity的<intent-filter>

```
<activity android:name=".SecondActivity"
        android:label="@string/second_activity">
         <intent-filter>
                <action android:name="com.android.demo.action.CHANGE"/>
                <category android:name="android.intent.category.DEFAULT"/>
         </intent-filter>
</activity>
```

course/ActivitySwitch(3)

ACTIVITY間單向資料傳遞

ACTIVITY間單向資料傳遞

- 經過之前Intent啟動email的例子，你應該可以知道Intent可以攜帶一些資料到另一個Activity

傳送端

- `Intent.putExtra(String key, Value val)`
  - 傳遞資料到另一個Activity的方法
  - 支援的傳遞類型
    - Int, boolean, byte, char, double, float, long, String, short
    - 上述型態的陣列
    - Serializable, Parcelable

- 基本上不支援物件的傳遞 (除非將物件建立成Serializable或Parcelable)

接收端

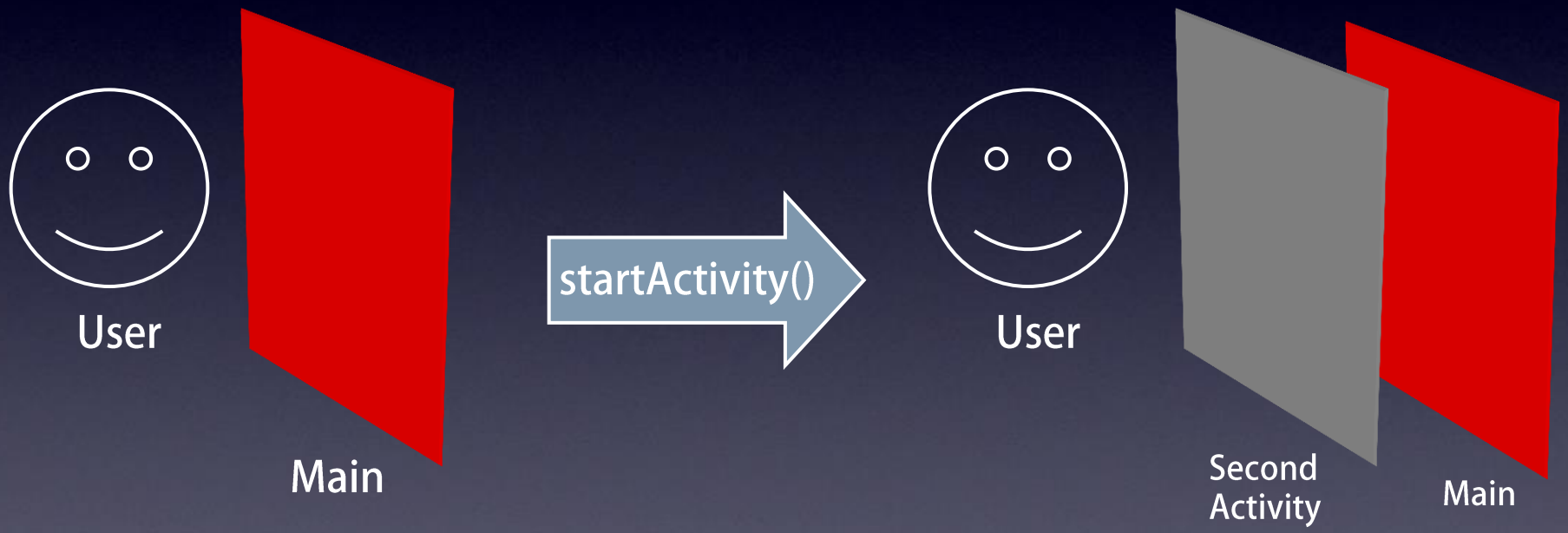
- `getIntent()`
  - 在接收的Activity內可以取得傳送來的Intent

- `Intent.getStringExtra(String key)`
  `Intent.getIntExtra(String key, int defVal)`
  …
  - 自己要知道傳遞過來的型態為何、索引值為何，並呼叫對應的方法取出傳遞的數值

ACTIVITY切換的原理

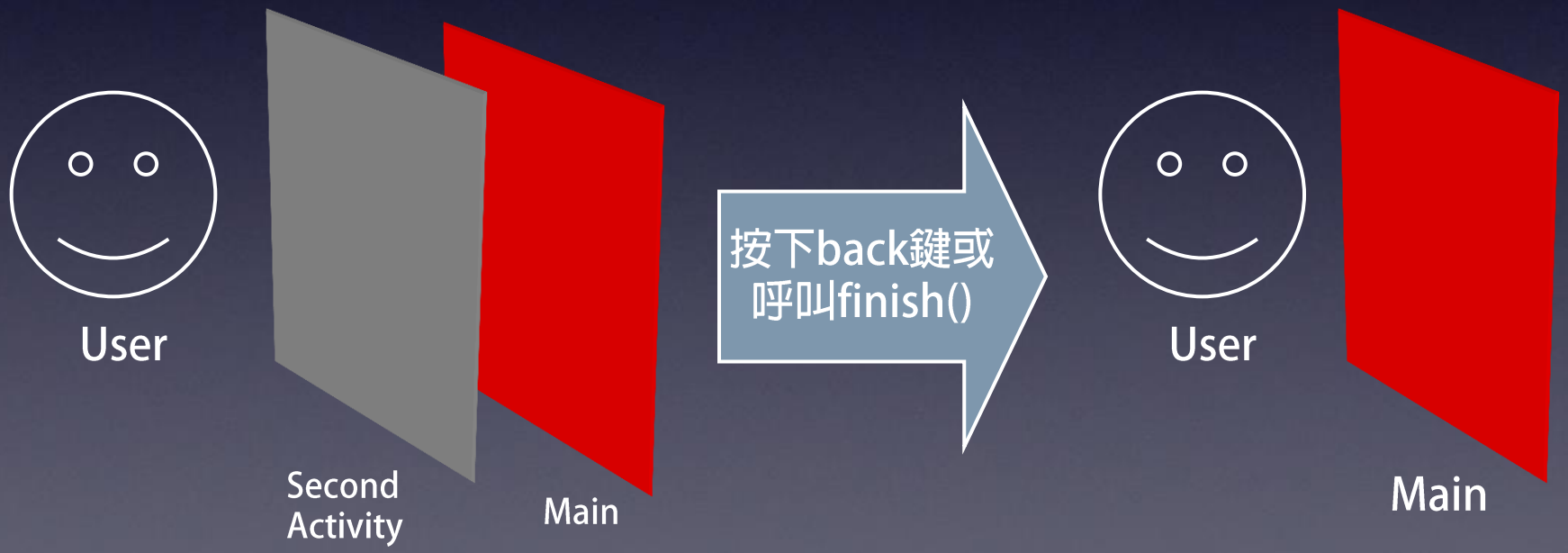
ACTIVITY切換的原理

- 轉換Activity時，其實就是把新的畫面推到使用者眼前



ACTIVITY切換的原理

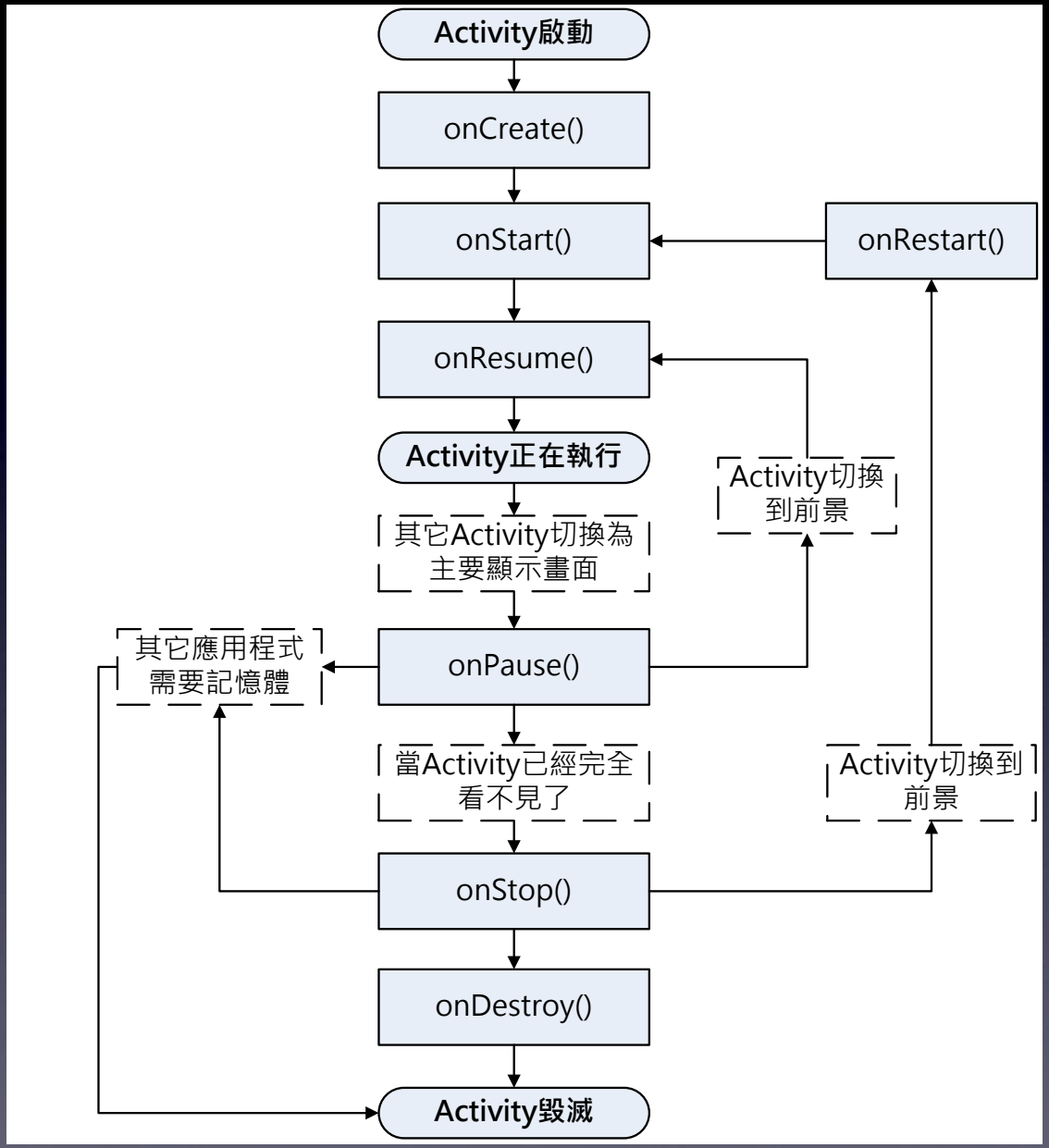
- 所以要回到上一個畫面時，千萬不要用startActivity()來啟動，而是要移除現在最前方的Activity
- 當你按下back鍵，或是在第二個Activity中呼叫finish()都可以使Activity結束



ACTIVITY生命週期

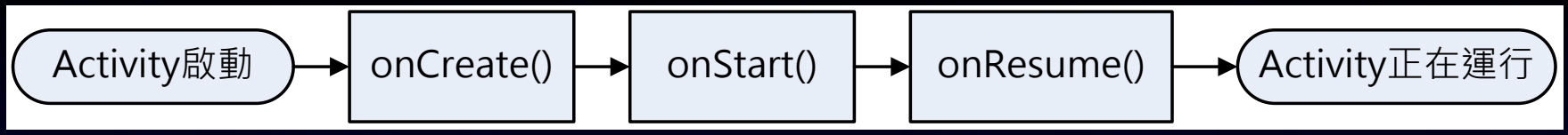
ACTIVITY生命週期

- Android的Activity有各種不同的狀態需要處理
- onCreate()就是狀態之一，表示Activity要建立了
- 除了onCreate()之外，還有很多應用程式的狀態需要我們了解
- onStart(), onResume(), onPause(), onStop(), onDestroy(), onRestart()


Activity啟動
onCreate()
onStart()
onRestart()
onResume()
Activity正在執行
Activity切換到前景
其它Activity切換為主要顯示畫面
其它應用程式需要記憶體
onPause()
當Activity已經完全看不見了
Activity切換到前景
onStop()
onDestroy()
Activity毀滅

生命週期

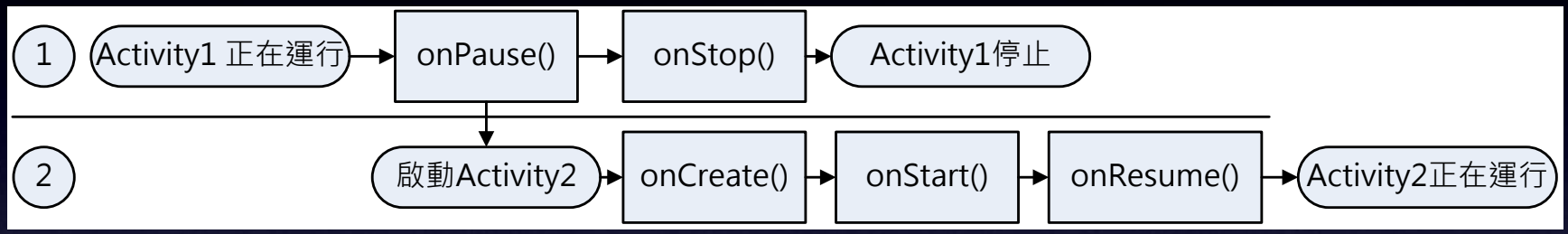
- 正常啟動Activity



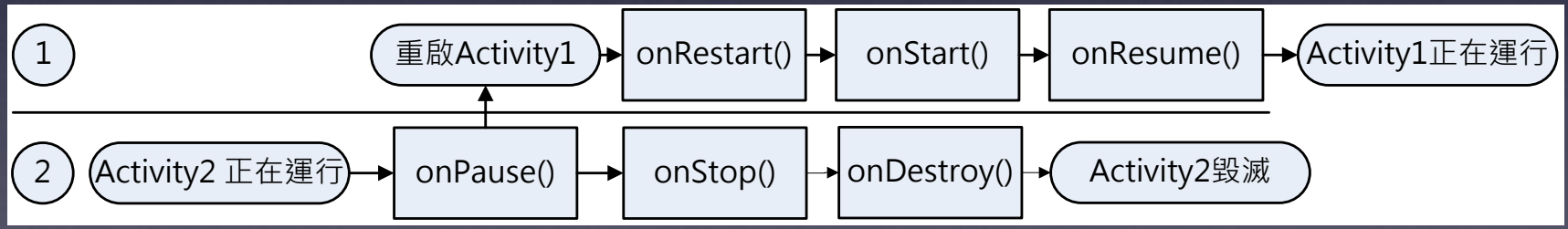
- 正常中止Activity

生命週期

- 呼叫另一個Activity (由1到2)



- 按下Back鍵返回原Activity (由2到1)



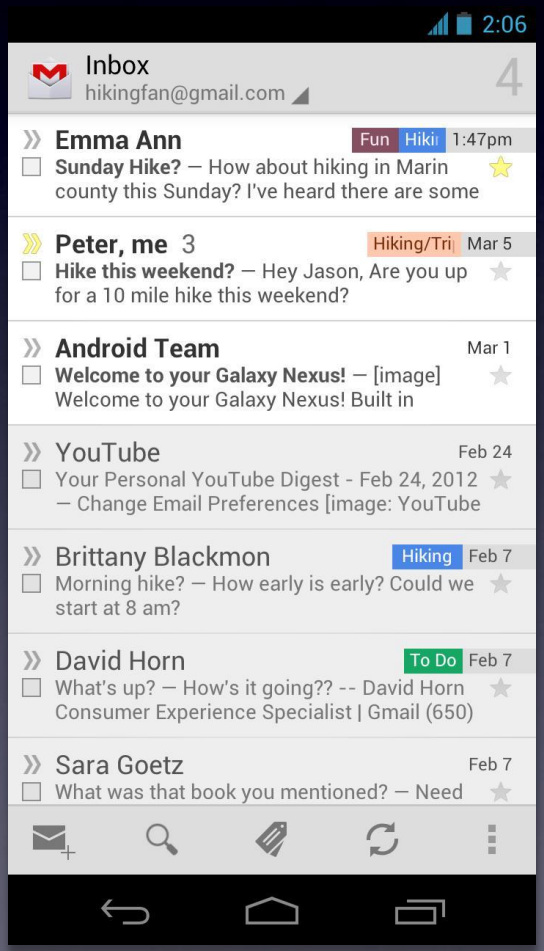
第五章

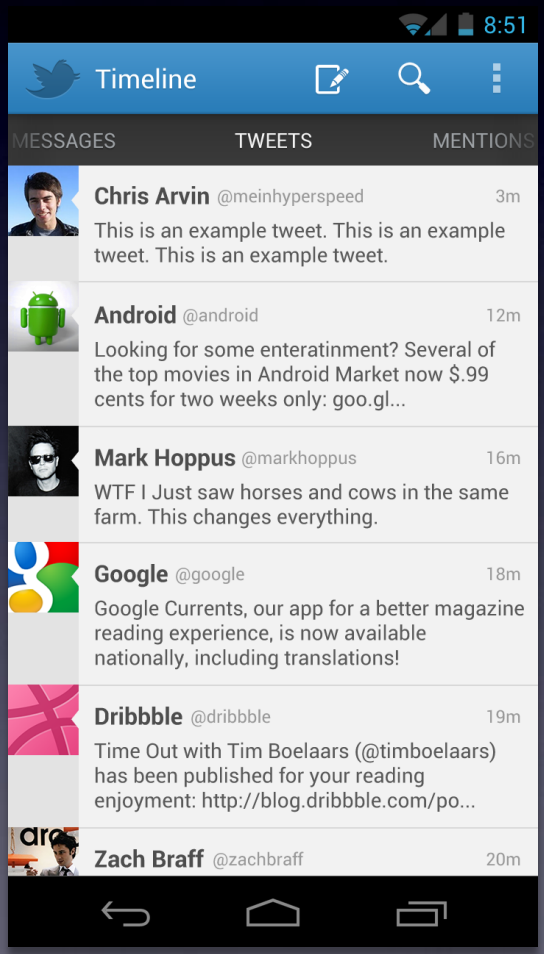
LISTVIEW

LISTVIEW與ADAPTER

LISTVIEW

- ListView可以說是資料在手機上呈現最好的方法





LISTVIEW

- 建立ListView的流程
  - 準備一連串資料
  - 準備呈現資料的Layout
  - 準備整合資料和Layout的Adapter (適配器)
  - 將Adapter設定給ListView

什麼是ADAPTER

- Adapter就是你的資料、要呈現的Layout與ListView之間的溝通橋樑
- Adapter提供給ListView呈現的畫面與資料的資訊
- Adapter整合資料到Layout上
- Adapter是資料與畫面的結合處
- Adapter是一種Design Pattern
  - http://en.wikipedia.org/wiki/Adapter_pattern

什麼是ADAPTER

- 為了方便大家製作，Android提供了簡便的Adapter
  - ArrayAdapter
  - SimpleAdapter

course/ArrayAdapterListViewExample

ARRAY ADAPTER

建立ARRAY ADAPTER

- 準備資料，建立String array

```
private static final String[] NAMES = {"John",
"Luke", "Matthew", "Peter", "James"};
```

- 建立Array Adapter

```
ArrayAdapter<String> adapter = new
ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, NAMES);
```

  - 參數1：Activity
  - 參數2：Layout XML
  - 參數3：資料

建立ARRAY ADAPTER

- 建立Array Adapter

```
ArrayAdapter<String> adapter = new
ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, NAMES);
```

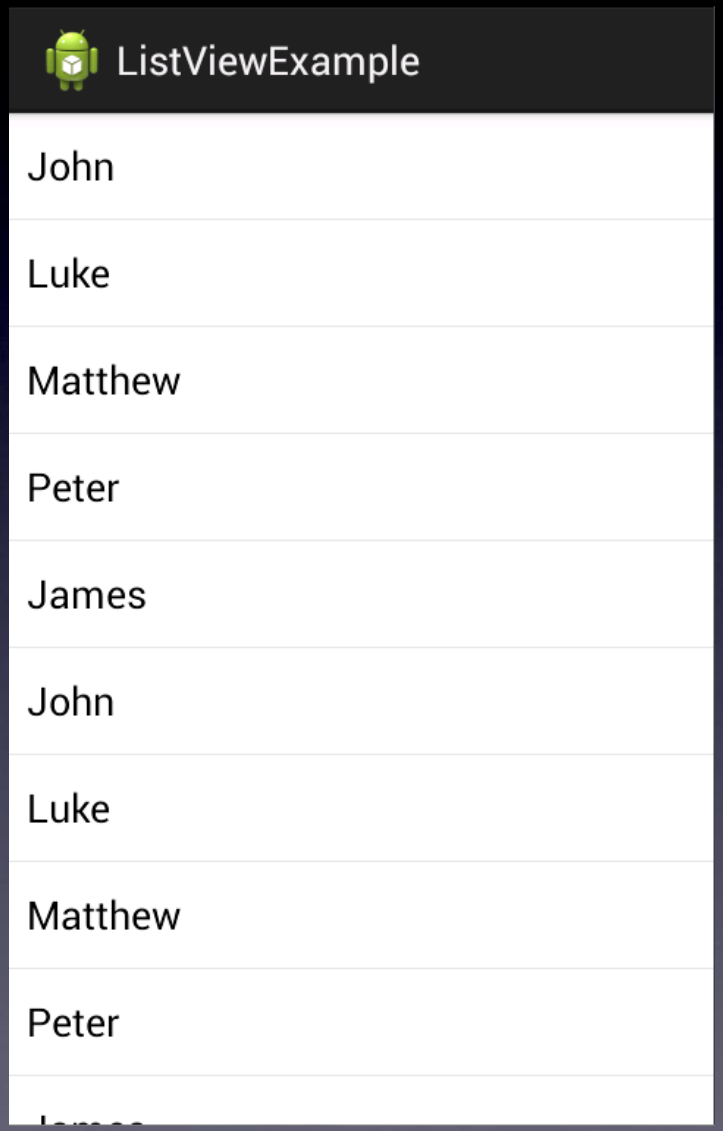
- 第二個參數的Layout必須要是TextView

- android.R.layout.simple_list_item_1 是使用系統的layout，為TextView

使用ARRAY ADAPTER

- `ListView.setAdapter(Adapter)`
  - 表示設定Adapter給ListView
  - 設定完後ListView就會開始向Adapter要呈現的畫面並排列在ListView中

執行結果


ListViewExample
John
Luke
Matthew
Peter
James
John
Luke
Matthew
Peter

course/ListViewEventExample

LISTVIEW事件

點擊事件

- 在ListView中可以使用OnItemClickListener接收到點擊事件

點擊事件

```
private OnItemClickListener mListener = new OnItemClickListener() {
     public void onItemClick(AdapterView<?> parent, View v, int position, long id) {}
};
```

- new OnItemClickListener
  - onItemClick()被呼叫時表示ListView的項目被按到了
    - 參數1：就是ListView
    - 參數2：ListView中被點中的View
    - 參數3：點到的位置，位置是從0開始

點擊事件

- `ListView.setOnItemClickListener()`
  - 將OnItemClickListener設定給ListView