

## 什麼是 Android?

Android 是一個包含作業系統、中介程式與關鍵應用程式的行動裝置軟體積木，初期的 [Android SDK](#) 提供必要的 API(應用程式開發介面)與工具以使用 Java 語言開發在 Android 平台上開發應用軟體。

### 功能

- 應用程式框架 可以重用或置換元件
- Dalvik 虛擬機器 行動裝置最佳化
- 整合瀏覽器 基於開放原始碼 [Webkit](#) 引擎
- 最佳化圖形 加強自訂 2D 圖形;3D 圖形則築基於 OpenGL ES 1.0 規格(硬體加速選項)
- SQLite 結構化資料儲存
- 媒體支援 一般聲音、影片與靜態影像格式(MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM 通訊 (硬體相關)
- 藍芽、EDGE、3G 與 WiFi (硬體相關)
- 相機、GPS、電子羅盤(compass)與加速度計(accelerometer) (硬體相關)
- 多元的開發環境 包含模擬器、除錯工具、記憶體與效率剖析與 Eclipse IDE 的外掛

### Android 架構

以下圖形顯示 Android 作業系統主要的元件，各個部分在後面解說。



## 應用程式

Android 附有一系列以 Java 語言 8 撰寫的核心應用程式，包含郵件程式、簡訊程式、日歷、地圖、瀏覽器、聯絡人與其它應用程式

## 應用程式框架

開發者可以完整使用與核心應用程式相同的 API,應用程式框架為簡化元件重用而設計;應用程式可以發佈功能並為其它應用程式所使用(受限於應用程式框架的安全限制)，使用者用同樣的機制用來置換元件。

應用程式底層是一組系統與服務，包含：

- 豐富且延伸自 [View](#) 用以建立應用程式，包含 lists, grids, text boxes, buttons, 甚至是嵌入式的瀏覽器。
- [Content Providers](#) 使應用程式可以存取或分享資料給其它應用程式(如聯絡人)。
- [Resource Manager](#) 提供存取非程式碼資源如本地化字串、圖形或布局檔案
- [Notification Manager](#) 讓應用程式得以在狀態列顯示自訂的警示。
- [Activity Manager](#) 管理應用程式生命週期並提供一般性的回溯導航(navigation backstack )

更多關於應用程式細節，請參考導覽

## 函式庫

Android 包含一組系統元件使用的 C/C++ 函式庫，使用者透過應用程式框架使用這些功能，部分核心函式庫列示如下：

- **系統 C 函式庫** 引用 BSD 標準系統 C 函式庫(libc)，調整為嵌入式 Linux 裝置。
- **媒體函式庫** 建立在 PacketVideo's OpenCORE 之上;該函式庫支援聲音的播放與錄製、影片格式、與靜態影像格式，包含 MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF
- **Surface Manager** 管理顯示次系統存取、與來自多重程式 2D 與 3D 組合圖層的串流
- **LibWebCore** 現代化的 Web 瀏覽器引擎
- **SGL 2D 圖形底層引擎**
- **3D libraries** OpenGL ES 1.0 API 實做;該函式庫使用硬體加速(視硬體提供功能)與 3D 軟體 rasterizer 最佳化
- **FreeType** 圖形與向量字的繪製
- **SQLite** 提供給所有應用程式使用的強大且輕量的關聯式資料庫存取引擎

## Android Runtime

Android 包含一組核心函式庫提供 Java 程式使用核心函式庫大部分的功能。

每一個 Android 應用程式執行於獨立的行程與獨立的 Dalvik 虛擬機器，Dalvik 被設計成單一裝置可有效率地使用多個虛擬機器，Dalvik 虛擬機器執行 Dalvik 執行檔(附檔名為.dex，最佳化使用記憶體格式)，虛擬機器為登記制，並執行"dx"工具編譯並轉換格式的 class 檔案

## Linux 核心

Android 依靠 Linux2.6 版的核心系統服務如安全性、記憶體管理、行程管理、網路堆疊與趨勢程式模型。該核心同樣是作為硬體與軟體積木間的抽象層

開始

## 安裝 SDK

本頁描述如何安裝 Android SDK 與設定開發環境，如果您還未下載 SDK，請用以下連結下載：

[下載 SDK](#)

### 內容

[系統與軟體需求](#)

[安裝 SDK](#)

[安裝 Eclipse 外掛\(ADT\)](#)

[在 eclipse 上開發 Android 應用程式](#)

[以其它 IDE 或工具開發 Android 應用程式](#)

[除錯](#)

[除錯與測試裝置設定](#)

[重要除錯提示](#)

[建置與安裝 Android 程式](#)

[移除 Android 程式](#)

[Eclipse 提示](#)

### 系統與軟體需求

開發 Android 程式使用 Android SDK 的工具與程式碼，您需要合適的開發電腦與開發環境描述如下：

支援的操作系統

- 視窗 XP 或 Vista
- 蘋果 X 10.4.8 或之後版本(只可是 x86)
- Linux(測試於 Ubuntu Dapper Drake)

支援的開發環境

- Eclipse
  - [Eclipse 3.2, 3.3 \(Europa\)](#)
  - [Android Development Tools plugin](#) (選項)
- 其它開發環境或 IDE
  - [JDK 5 或 JDK 6](#) (單獨的 JRE 並不足夠)
  - 與 Gnu Compiler for Java 不相容(gcj)

- [Apache Ant](#) Linux 與蘋果 1.6.5 或視窗 1.7 之後版本

## 安裝 SDK

下載 SDK 後，將 zip 檔案解開到您電腦路徑上，文件的其它部分，我們將會使用 \$SDK\_ROOT 表示您安裝 SDK 的目錄路徑


另外，您可以把 \$SDK\_ROOT/tools 加入您的參考路徑在 Linux 上，

- 編輯您的 ~/.bash\_profile 或 ~/.bashrc 檔案，編輯並設定 PATH 環境變數並加入 \$SDK\_ROOT/tools 的完整路徑。如果沒有該變數，加入

```
export PATH=${PATH}:<$SDK_ROOT/tools 的完整路徑>
```

- 在蘋果上，如 Linux 般編輯使用者目錄下的 bash\_profile 檔案，如果您的機器上沒有這個檔案，請建立一個



- 在視窗上，在  我的電腦上按右鍵，選擇內容，選擇進階，點擊環境變數按鈕，然後顯示對話視窗，雙擊系統變數內的 Path，並加入 \$SDK\_ROOT/tools 的完整路徑

加入 \$SDK\_ROOT/tools 到您的路徑使您執行 Android Debug Bridge (adb) 和其它命令列工具時，毋須使用 tools 的完整路徑，請記住，如果更新 SDK 於不同路徑時，請勿忘記更新 PATH 到新路徑

## 安裝 Eclipse 外掛(ADT)

如果您要使用 Eclipse 作為開發 Android 應用程式的環境，您應該裝 Android Development Tools (ADT) 的外掛，ADT 提供了 Android 專案與工具的整體支援。ADT 外掛包含了多個有用的強項讓建立、除錯與執行 Android 應用程式更加簡單快速。

如果您不打算使用 Eclipse，那您也毋須下載安裝 ADT。

下載安裝 ADT, 建立 Eclipse remote update site 的步驟如下：

1. 啟動 Eclipse，選擇 **Help > Software Updates > Find and Install...**
2. 對話視窗出現後，選擇 **Search for new features to install** 後按 **Next**.
3. 按 **New Remote Site**.
4. 在最後的對話視窗的 remote site 鍵入名稱(如 Android 外掛)與以下 URL

<https://dl-ssl.google.com/android/eclipse/>

後按 **OK**

5. 然後您會看到新站台出現在搜尋清單上(且被選取)，按 **Finish**.
6. 在接下來的搜尋結果對話視窗核取 **Android Plugin > Eclipse Integration > Android Development Tools** 後按 **Next**.

7. 閱讀許可條款, 如果同意的話選取 **Accept terms of the license agreement** , 按 **Next**
8. 按 **finish**.
9. ADT 外掛未經簽署, 您可以按 **Install All**. 同意安裝
10. 重新啟動 Eclipse
11. 重新啟動後, **更新您的 Eclipse 的 preferences** 指定 SDK 的目錄路徑
  - A. 選擇 **Window > Preferences...** 開啟 Preferences panel. (Mac OS X: **Eclipse > Preferences**)
  - B. 選擇左邊版面的 **Android**
  - C. 在主版面上的 SDK Location , 按 **Browse...** , 指定 SDK 的目錄路徑
  - D. 按 **apply** 後再按 **OK**

## 更新 ADT 外掛

使用以下步驟將 ADT 外掛更新到最新版

1. 選擇 **Help > Software Updates > Find and Install...**
2. 選擇 **Search for updates of the currently installed features** 然後按 **Finish**.
3. 如果有 ADT 的最新版, 選擇並安裝

或是改用以下操作,

1. 選擇 **Help > Software Updates > Manage Configuration**.
2. 找一下樹狀圖選擇 **Android Development Tools <版本號>**
3. 選擇 **Available Tasks**. 下的 **Scan for Updates**

## 在 Eclipse 上開發 Android 應用程式

開始在 Eclipse 上開發 Android 應用程式, 您首先要建立 Android 專案並設定載入配置, 再來您就可以寫作、執行與除錯您的程式

以下的操作是假設您已安裝好 Eclipse 的環境, 如果您還未安裝, 請回到[安裝 Eclipse 外掛 \(ADT\)](#)檢視安裝資訊

## 建立 Android 專案

ADT 外掛提供精靈讓您快速為已經存在的程式碼或是建立全新的專案, 請照以下步驟建立專案:

1. 選擇 **File > New > Project**
2. 選擇 **Android > Android Project** 然後按 **Next**
3. 為專案選擇以下內容:
  - 選擇 **Create new project in workspace** 建立新專案輸入 Project name, Package name、您的應用程式名稱 Application name 與單一類

別 Activity name 以建立樣板的.java 檔案。

- 選擇 **Create project from existing source** 為已存在的程式碼建立專案，您可以使用這個選項來執行 SDK 所附的範例程式，這些程式放在 SDK 下 samples/directory 的路徑內。

#### 4. 按 **Finish**

ADT 外掛為您的專案建立以下目錄與檔案

- src/ 包含所有空殼 .java Activity 檔案的目錄
- res/ 資源檔案目錄
- AndroidManifest.xml 專案艦單

### 建立載入配置

您必須建立載入配置以便能讓 Eclipse 執行與除錯應用程式，載入配置指定專案載入、啟動 Activity，使用模擬器選項等作業

照以下步驟，建立載入配置：

1. 選擇 **Run > Open Run Dialog...** 或是 **Run > Open Debug Dialog...**
2. 在左邊的專案型態 **Android Application** 上按右鍵選擇 **New**.
3. Name 欄位上輸入您的配置名稱
4. 在 android 頁次，按 Browse...選擇專案與 Activity 以執行
5. 在 emulator 頁次設定想要的螢幕(Screen Size)與網路屬性(Network options)，和其它的模擬器啟動選項(:TODO [emulator startup options](#))
6. 您可以 Common 頁次設定其它想要的選項
7. 按 **apply** 儲存載入配置或是酌情按 **Run** 或 **Debug**

### 應用程式執行與除錯

一旦您為您的應用程式設定好專案與載入配置，您可以照以下所說的來執行或除錯。

從 Eclipse 的目錄選擇 **Run > Run** 或 **Run > Debug**，以執行或除錯有效(最近執行過)的載入配置。

若要修改或設定有效的載入配置，選擇 **Run > Open Run Dialog...** 或 **Run > Open Debug Dialog...**

執行或除錯應用程式會觸發以下動作：

- 模擬器如尚未執行則啟動模擬器，
- 如果程式建置好後有過修改，則重新編譯專案，並將程式安裝到模擬器
- **執行**則啟動程式



- **除錯**則將程式啟動為"Wait for debugger"模式，並開啟 Eclipse 的除錯檢視

## 以其它 IDE 或工具開發 Android 應用程式

雖然推薦以 Eclipse 加上 Android 外掛開發應用程式(整合了編輯、建置與除錯功能)，SDK 還是提供工具讓您能以其它 IDE(有包 IntelliJ)開發(或者您不想用 ADT 外掛，單純以 Eclipse 開發)。

## 建立 Android 專案

Android SDK 包含一個 activityCreator 程式，該程式可以為您的專案建立一些樣板程式與建置檔，您可以用該程式建立新專案或為已有的程式碼建立專案(如 SDK 內附的範例程式)；該程式在 Linux 與 Mac 檔案叫做 activityCreator，Python 則是 activityCreator.py，Windows 則是用 activityCreator.bat 批次，不管是那個平台，都可直接使用 activityCreator 命令。

用以下步驟執行 activityCreator 建立專案：

1. 在命令列，變更到 SDK 下的 tools 目錄，並為您的專案建立一個新目錄；如果您要為已經存在的程式碼建立專案，請變更目錄到您應用程式的目錄。
2. 在命令列執行 activityCreator，您必須指定完整的類別名稱做為參數，如果是新建專案，則會以完整的類別名稱建立樣板程式碼；如果是從已存在的程式碼建立專案，則必須指定套件內的 Activity 類別名稱。命令列參數還包括：
  - --out <目錄> 會設定輸出目錄，預設是現在所在的目錄，如果您要將專案檔案放在其它目錄，請用這個參數來指定
  - --ide intellij, 在新建專案產生 intellij 檔案

範例如下

```
~/android_linux_sdk/tools $ ./activityCreator.py --out myproject your.package.name.ActivityName
package: your.package.name
out_dir: myproject
activity_name: ActivityName
~/android_linux_sdk/tools $
```

activityCreator 建立以下的檔案與目錄(但是不會覆蓋已經存在的程式)

- AndroidManifest.xml 應用程式艙單，與專案內的特定的 Activity 類別同步
- build.xml ant 建置檔，可以建置/打包程式
- **src/your/package/name/Activity 檔名.java** 輸入參數指定的 Activity 類別
- 您的\_activity.iml，您的\_activity.ipr，您的\_activity.iws [只有使用--ide intellij 參數才會產生]
- res/ 資源檔目錄
- src/ 源代碼目錄
- bin/ ant 建置檔的輸出路徑

您現在可以帶著目錄到處開發了，但是不要忘了，使用 tools/ 檔案夾內的:todo adb 程式把檔

案送給模擬器。

但是，您應當避免移動 SDK 所在的目錄，因為這會破壞建立檔(除非作業前，更新 SDK 目錄)

## 建置 Android 應用程式

使用 activityCreator 建立的 ant 建置檔來建置您的 Android 應用程式

1. 如果您沒有 Ant，您可以從 [Apache Ant 首頁](#) 獲得，安裝並確定它在您的執行路徑上
2. 叫用 Ant 前，您需要宣告 JAVA\_HOME 環境變數指定到 JDK 的安裝路徑

註：在 Windows 安裝 JDK 預設路徑會裝到 "Program Files" 目錄下，因為中間有空白關係，會導致 Ant 失敗，修正這個問題的方式就是指定 JAVA\_HOME 環境變數，如 `set JAVA_HOME=c:\Progra~1\Java`。更簡單的方法則是將 JDK 安裝到沒有空白路徑如：`c:\java\jdk1.6.02`

3. 如果尚未完成，請照上述建立新專案的方式設定專案
4. 您現在可以在 build.xml 同在的目錄下鍵入 ant 指令執行建置，每次改變程式碼或是資源檔，您應該重新執行 ant 以打包最新的發佈檔案

## 執行 Android 應用程式

執行編譯好 Android 應用程式，您可以照以下的描述用(:todo)abd 工具將.apk 檔案上載到模擬器的 /data/app 目錄

1. 啟動模擬器(從命令列執行 <您的 SDK 目錄>/tools/emulator )
2. 在模擬器內，回到 home 螢幕(最好不要在應用程式執行時重新安裝應用程式，按 Home 按鍵離開應用程式)
3. 執行 `adb install 我的專案/bin/<應用程式名稱>.apk` 上載可執行檔，例如，安裝 Lunar Lander 範例，在命令列上切換到 <您的 SDK 目錄>/sample/LunarLander 後鍵入 `../tools/adb install bin/LunarLander.apk`
4. 在模擬器內，開啟可執行程式，捲動並選擇程式啟動

註：安裝好 Activity 後，最好重新啟動模擬器，因為套件管理員只會在模擬器啟動時完整檢核清單

## 為您的 Android 應用程式加入除錯器

這一段說明如何在螢幕上顯示除錯資訊(如中央處理器使用率)，和如何讓您的 IDE 除錯模擬器的應用程式

使用 Eclipse 外掛會自動加入除錯器，但您可以設定其它的 IDE 聆聽除錯埠的除錯資訊

1. 啟動(:todo)[Dalvik Debug Monitor Server \(DDMS\)工具](#)，提供您的 IDE 與模擬器間的埠傳送服務

2. 在你的模擬器設定除錯選項，如阻斷程式啟動直到除錯器加入，請注意，很多除錯選項並非需要 DDMS 才能使用，如顯示中央處理器使用率或是模擬器的螢幕的更新頻率
3. 設定您的 IDE 聆聽除錯埠到 8700，相關資訊請見(:todo)

## 設定您的 IDE 聆聽除錯埠

DDMS 會為模擬器上的每一個虛擬機器指定特定的除錯埠，您必須設定您的 IDE 聆聽該埠或是連上預設的 8700 埠。

執行中的程式，若是變更除錯選項或是"阻斷程式啟動"選項，會導致系統關畢目前執行的程式，您可以用這招來簡化關畢有問題的程式

## 除錯

Android 有大量的工具來幫您進行程式除錯

- (:TODO)DDMS - 圖形介面程式支援埠傳送(如此您得以在 IDE 設定中斷點)，模擬器的螢幕擷取，執行緒與堆疊資訊，以及其它功能，您也可以執行 logcat 取得您的 Log 資訊，更多資訊請見連結主題。
- (:TODO)logcat – 轉儲系統訊息 log，訊息包含模擬器拋錯時的堆疊追蹤，和 Log 訊息，執行 logcat 請見連結主題。

```
...
I/MemoryDealer( 763): MemoryDealer (this=0x54bda0): Creating 2621440
bytes heap at 0x438db000
I/Logger( 1858): getView() requesting item number 0
I/Logger( 1858): getView() requesting item number 1
I/Logger( 1858): getView() requesting item number 2
D/ActivityManager( 763): Stopping: HistoryRecord{409dbb20
com.google.android.home.AllApps}
...
```

- (:TODO)Android Log – 一個 log 類別可以在模擬器上將 Log 訊息輸出到檔案，如果您執行 logcat，那您可以及時讀取訊息(後面會提到)。在您的程式碼加入少量的 log 方法叫用：

使用 Log 類別，視訊息需求狀況，叫用 Log.v() (verbose 等級), Log.d() (debug 等級), Log.i() (information 等級), Log.w() (warning 等級)或是 Log.e() (error 等級)等方法  
Log.i("MyActivity", "MyClass.getView() - Requesting item number " + position)

您可以用 logcat 來讀取這個訊息

- (:TODO)TraceView – 您可以使用 TraceView 這個圖形化介面讀取器來讀取 Android 所存檔的 log 檔案，更多資訊請見連結主題。
- [Eclipse 外掛](#) – Eclipse Android 外掛包含一些工具(ADB,DDMS,logcat 輸出與其它功

能)，更多資訊請見連結主題。

- **除錯與測試裝置設定** - Android 揭示數種揭示重要訊息的設定諸如中央處理器使用率與顯示頻率，詳見下面的[除錯與測試裝置設定](#)。

## 除錯與測試裝置設定

Android 有數個簡單設定讓您更易測試與除錯應用程式，到模擬器的開發設定頁次，到 **Dev Tools > Development Settings** 將會開啟開發設定頁設定下列等選項

- **Debug app** 選擇要除錯的應用程式，這個選項不是除錯器的選項，但是設定這個值有兩個效應
  - 讓 android 除錯時在中斷點暫停太久時不致於拋出錯誤
  - 讓您可以選擇**阻斷程式啟動**直到您的除錯器加入(見下段)
- **阻斷程式啟動** 阻斷所選程式載入直到您的除錯器加入，這樣您方可在 `OnCreate()` 設定中斷，這對於除錯 Activity 啟動程式時至關重要，執行中的程式，若是變更除錯選項，則會導致系統關畢目前所有該執行的程式，要啟用這個選項，必先上一段所提的 Debug app 選項；或是在您的程式碼加入(:todo)[waitForDebugger\(\)](#) 也可阻斷程式啟動
- **立即 destroy(移除?)activity** 當 activity 停止後，立即加以 destroy (譬如 android 必須回收記憶體時)，在測試(:todo)[onFreeze\(Bundle\)](#) / [onCreate\(android.os.Bundle\)](#) 程式路徑時非常有效，否則非常困難加以強制(This is very useful for testing the [onFreeze\(Bundle\)](#) / [onCreate\(android.os.Bundle\)](#) code path, which would otherwise be difficult to force)。如果您的程式沒有儲存狀態的話，選用這個選項可能會引起一些問題。
- **顯示螢幕更新** 螢幕任何區塊被重繪時，閃現一個短暫的粉紅矩形，要找出不必要的螢幕繪製時很有用
- **顯示中央處理器使用率** 在螢幕上層顯示中央處理器使用率，上方的紅色棒顯示全部的中央處理器使用率，下方的綠色棒則顯示組成這個畫面花了多少中央處理器時間。註：一旦開啟此顯示，只有重新啟動模擬器才能關閉顯示
- **顯示螢幕 FPS(幀/秒)** 顯示目前幀率，大部分用在遊戲，顯示整體達到幀率。註：一旦開啟此顯示，只有重新啟動模擬器才能關閉顯示
- **顯示背景** 沒有 Activity 螢幕時顯示背景模式，實際上這不會發生，除非正在除錯設定值會被記住，即使重新啟動模擬器

## 重要除錯提示

## 快速堆疊轉儲(quick stack dump)

從模擬器取得堆疊轉儲，你可以 adb shell 登錄，用"ps"找到您要的行程，然後下"kill -3"，則堆疊追蹤會出現在 log 檔案

## 在模擬器螢幕顯示有用資訊

裝置可以顯示有用的資訊如顯示中央處理器使用率或是顯示螢幕更新，開啟或關閉這些開發設定，請參考[除錯與測試裝置設定](#)。

## 取得模擬器系統狀態資訊(dumstate 轉儲系統資訊)

您可從 Dalvik Debug Monitor Service 工具存取系統資訊，請參考 adb 主題頁的(:todo)[dumpsys and dumpstate](#)

## 取得模擬器程式狀態資訊(dumsys 轉儲程式資訊)

您可從 Dalvik Debug Monitor Service 工具存取程式資訊，請參考 adb 主題頁的(:todo)[dumpsys and dumpstate](#)

## 取得無線連接資訊

從裝置目錄選擇"Dump radio state"，您可以從 Dalvik Debug Monitor Service 工具取得無線連接資訊

## 記錄追蹤資料

在 Activity 內叫用 `ndroid.os.Debug.startMethodTracing()` 記錄追蹤資料，請參考(:todo)[Running the Traceview Debugging Program](#)

## 記錄無線資料

原則上系統不記錄無線資料，但是您可以使用以下命令記錄無線資料

```
adb shell
logcat -b radio
```

## 執行 adb

Android 提供一款工具叫做 adb，提供不同功能，包含了模擬器上的檔案同步與移動、傳送埠與在模擬器上執行一個 UNIX Shell，請參考(:todo)[Using adb](#)

## 捕捉模擬器螢幕

Dalvik Debug Monitor Server (DDMS) 可以捕捉模擬器螢幕

## 使用除錯輔助類別

Android 使用除錯輔助類別如(:todo)[util.Log](#) 和 [Debug](#) 供您使用

## 建置與安裝 Android 程式

Android 需要自訂的建置工具正確地建置資源檔與 Android 程式的其它部分，所以您必須為您的程式指定一個建立環境。

特定的 Android 編譯步驟包含編譯 XML 與其它資源檔，並產出正確的格式。編譯完成的

Android 程式為一個.apk 檔案，是一個壓縮了(:todo).dex 檔案、資源檔、資料檔與其它相關檔案。您可以從無到有、或是從現有的檔案建立正確的 Android 專案結構。

Android 目前尚未支援第三方使用原生語言(C/C++)來開發程式

**最佳的開發方式**還是建議[在 Eclipse 上開發 Android 應用程式](#)，這種方式提供建立、執行與除錯 Android 程式

如果您使用其它 IDE，則 [Android 提供其它 IDE 工具](#) 建立與除錯 Android 程式，但是它們尚未加以整合。

## 移除 Android 程式

要移除安裝在模擬器上的程式，您必須(:todo)[執行 adb](#) 刪除您送往模擬器並安裝的.apk 檔案，照連結主題所描述，使用 adb shell 進入裝置的 Shell，移到 data/app/，用"rm 您的\_程式.apk"移除檔案。

## Eclipse 提示

### 在 eclipse 內執行任意 Java 表達語法

您可以在 Eclipse 中斷點中斷時執行任意的程式碼，比如說，函式內一個字串參數叫"zip"，您可以得到套件資訊並叫用類別方法，也可以叫用任意的 static method，例如鍵入 `android.os.Debug.startMethodTracing()` 將會啟動 dmTrace。

從主目錄 **Window>Show View>Display** 開啟程式執行視窗，會出現一個本文編輯器，鍵入語法、HighLight 文字、點擊 J 圖示(或按 CTRL + SHIFT + D)執行語法。語法執行於中斷點(或 Single-stop)相同的執行緒(如果您手動中斷執行緒，您必須單步前進，但是在

`Object.wait()`不會有用)。原文:If you suspend the thread manually, you have to single-step once; this doesn't work if the thread is in `Object.wait()`.

如果您現在停在中斷點，您可以選取程式碼，按 CTRL + SHIFT + D 執行。

您可以使用 ALT +SHIFT + ↑選取更大範圍的大括號區塊或是使用 ALT +SHIFT + ↓選取更小範圍的大括號區塊

下面是一些使用 Eclipse Display 視窗輸入與回應的範例

輸入	回應
zip	(java.lang.String) /work/device/out/linux-x86-debug/android/app/android_sdk.zip
zip.endsWith(".zip")	(boolean) true
zip.endsWith(".jar")	(boolean) false

除錯時也可以使用 scrapbook 執行任意程式碼，請找 Eclipse 文件"scrapbook"內容。

## 手動執行 DDMS

雖然建議使用 ADT 外掛，您也可以手動啟動 DDMS，然後設定 Eclipse 於 8700 埠除錯(註：  
(:todo)要先執行 [DDMS](#))

## Android!您好

第一印象，您做為開發者對一個框架的第一印象就是看看寫好個"Android!您好"簡不簡單，在 Android，更加的簡單，看起來就是這樣：

- [建立專案](#)
- [建立界面](#)
- [執程式碼：Android!您好](#)

下面說明所有細節

- [將界面升級到 XML 排版](#)
- [專案除錯](#)
- [以 Eclipse 建立專案](#)

### 建立專案

建立專案再簡單不過了，一個 Eclipse 外掛讓 Android 開發在一瞬間，您必須先安裝 [Eclipse 3.2](#) 或 [33](#)，然後需要再安裝 [Android Eclipse 外掛](#)，完成後再回到這裡

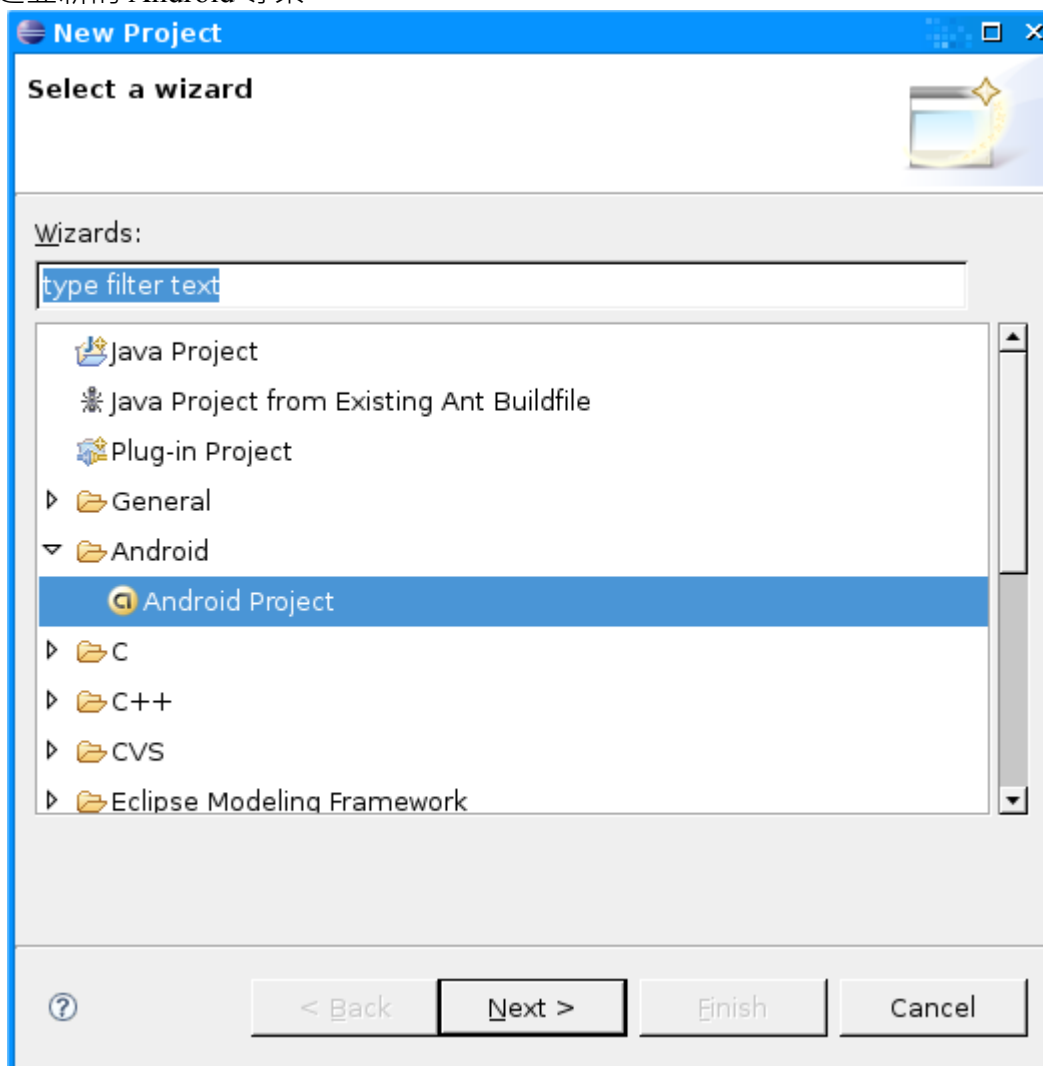
首先，這裡是建立"Android!您好"的大綱

1. 經由 File > New > Project 目錄建立一個"Android 專案"
2. 在新專案視窗填入相關明細
3. 編輯自動產生的空殼程式，顯示輸出



就是這麼簡單，下面說明每一個步驟

## 1. 建立新的 Android 專案



選好 Android Project 後按 Next 按鈕

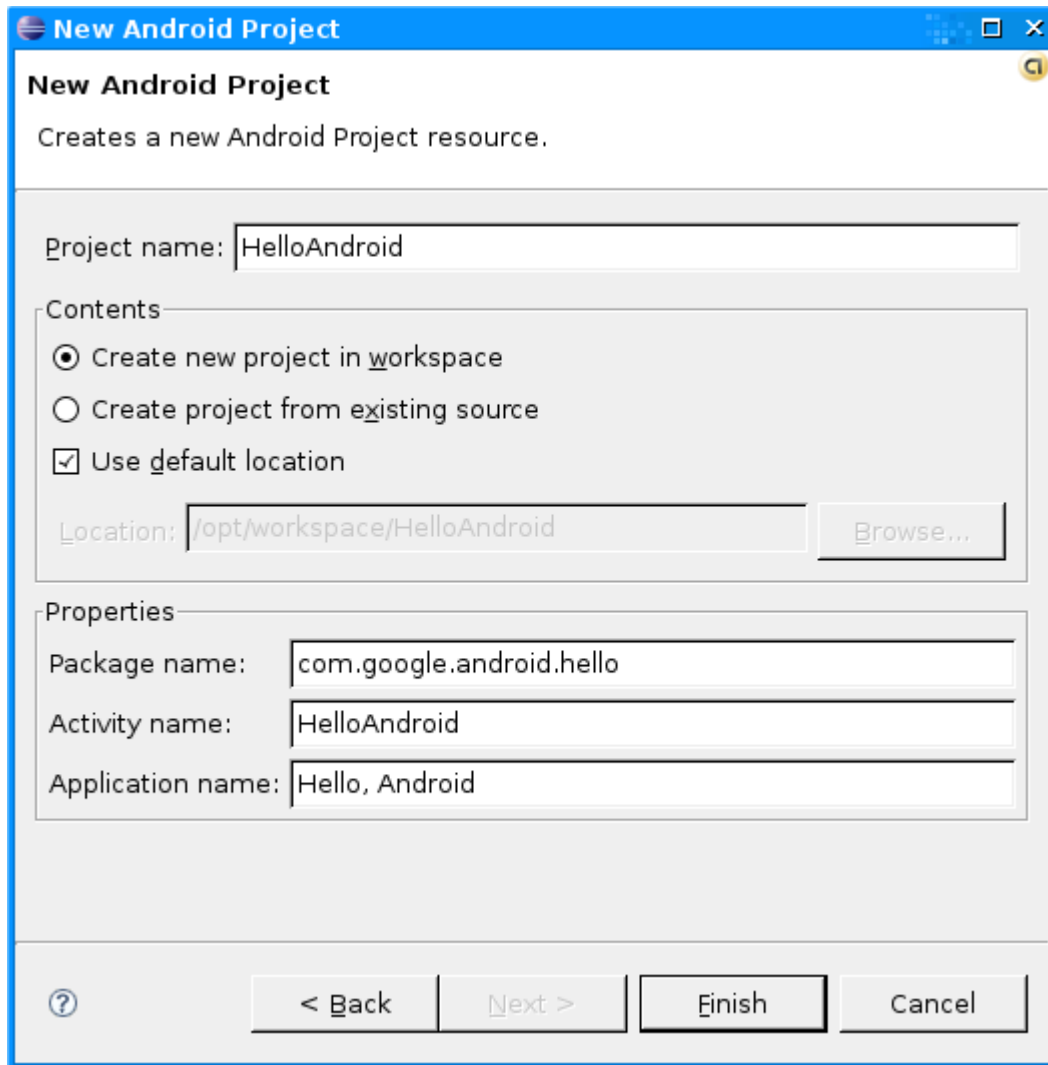
## 2. 填入專案明細

下個視窗填入這個專案的相關細節，例如：

欄位意義如下：

Project Name	包含專案的電腦檔案夾或是目錄
Package Name	套件名稱空間 – (不用我再說了吧) - that you want all your source code to reside under. This also sets the package name under which the stub Activity will be generated. The package name you use in your application must be unique across all packages installed on the system; for this reason, it's very important to use a standard domain-style package for your applications. In the example above, we used the package domain "com.google.android"; you should use a different one appropriate to your organization.

Activity Name	這是外掛產生的空殼程式的類別名，繼承自 Andrid 的 Activity 類別，這是一個可以執行、運做的類別，它也可以建立界面，也可以不用
Application Name	這是程式給人讀的標題



### 3. 編輯自動產生的空殼程式

外掛執行後，有支產生的程式叫 HelloAndroid 看起來像這樣

```
public class HelloAndroid extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle icicle)
    {
        super.onCreate(icicle);
        setContentView(R.layout.main);
    }
}
```

```
}
```

下一步是開始改程式

## 建立界面

一旦完成設定，明顯地下一步就是在螢幕上秀些字，這就是最後要完成的-下面將會逐行說明

```
public class HelloAndroid extends Activity {  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        TextView tv = new TextView(this);  
        tv.setText("Android!您好");  
        setContentView(tv);  
    }  
}
```

請注意，您應該 `import android.widget.TextView` 才能編譯這個範例。

在 Android，使用者界面由稱為 View 的子類別(hierarchies of classes)所組成，View 簡單的說就是一個繪圖物件，比如 radio button、動畫或是(這個例子)文字標籤，處理文字的 View 次類別則稱為 TextView

這裡是 TextView 的建構式

```
TextView tv = new TextView(this);
```

TextView 的建構參數是 Android 的 Context Instance，Context 表示一個系統操作的憑藉，它提供諸如解悉資源、存取資料庫與設定喜好等服務，Activity 就是繼承自 Context，因為我們的 HelloAndroid 繼承自 Activity，所以也是一個 Context，所以用來當作 TextView 的建構參數。

建構好 TextView 後就是要它顯示文字

```
tv.setText("Android!您好");
```

這裡沒有什麼特別的

這時，我們既然已經建構了 TextView 並要它顯示那些文字，最後一個步驟就是將 TextView 對螢幕顯示

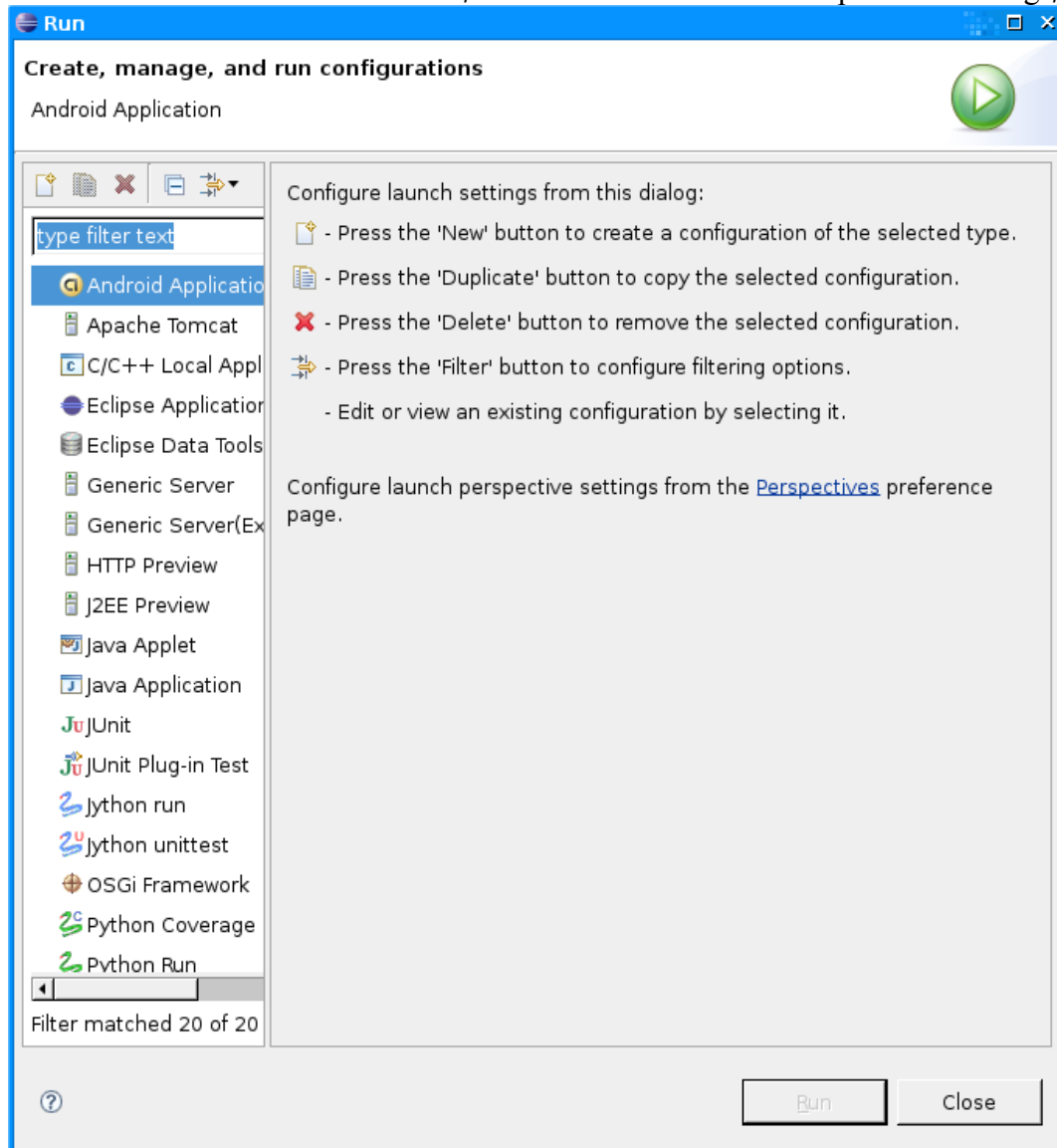
```
setContentView(tv);
```

Activity 的 setContentView() 指示系統應當將 View 關聯到 Android 的界面，如果 Activity 不叫用這個方法，除了空白螢幕什麼也不會顯示，我們的目的就是顯示文字，所以將剛建立的 TextView 傳送

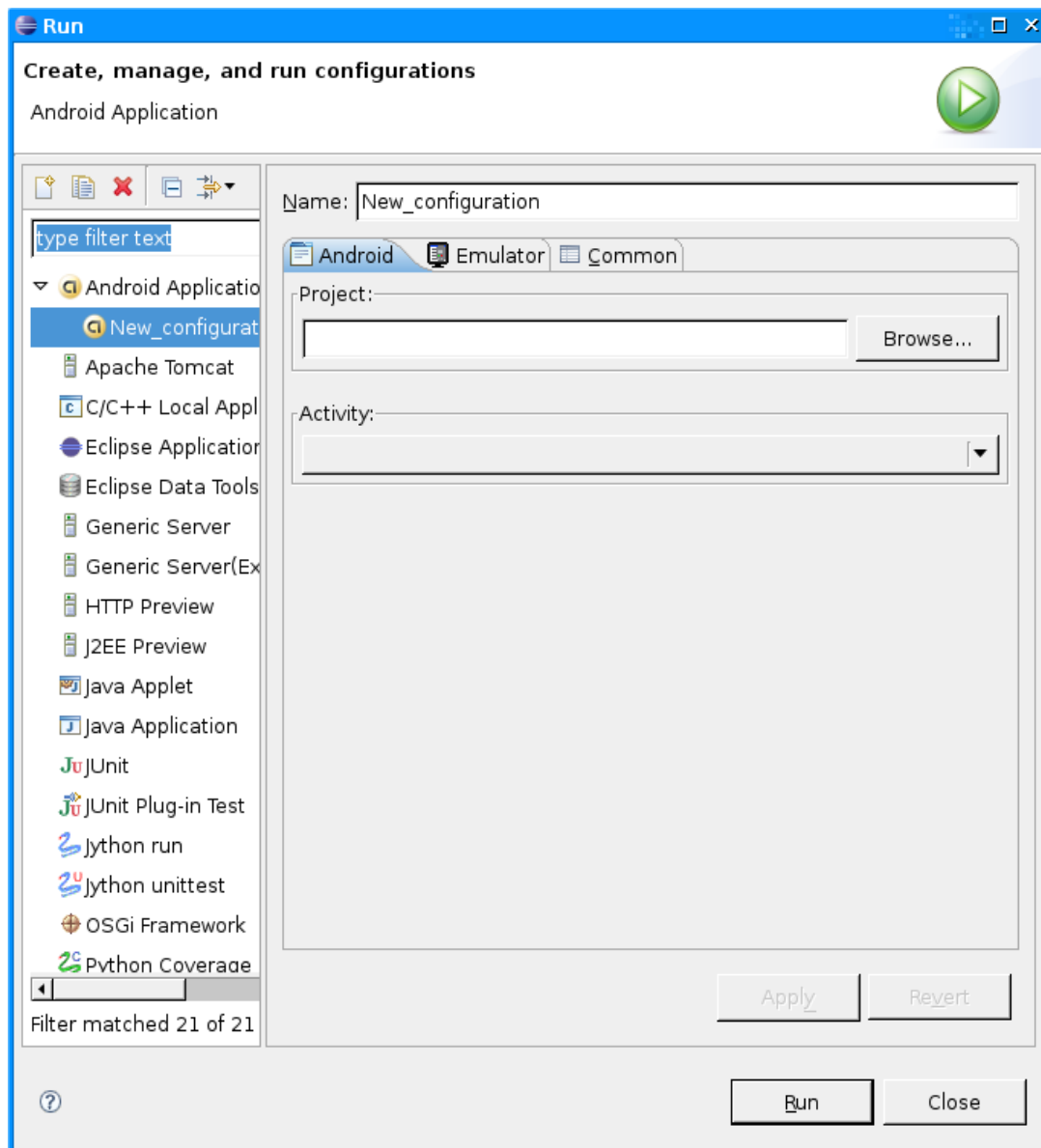
這就是"Android!您好"，再來就是執行了

## 執行程式碼：Android!您好

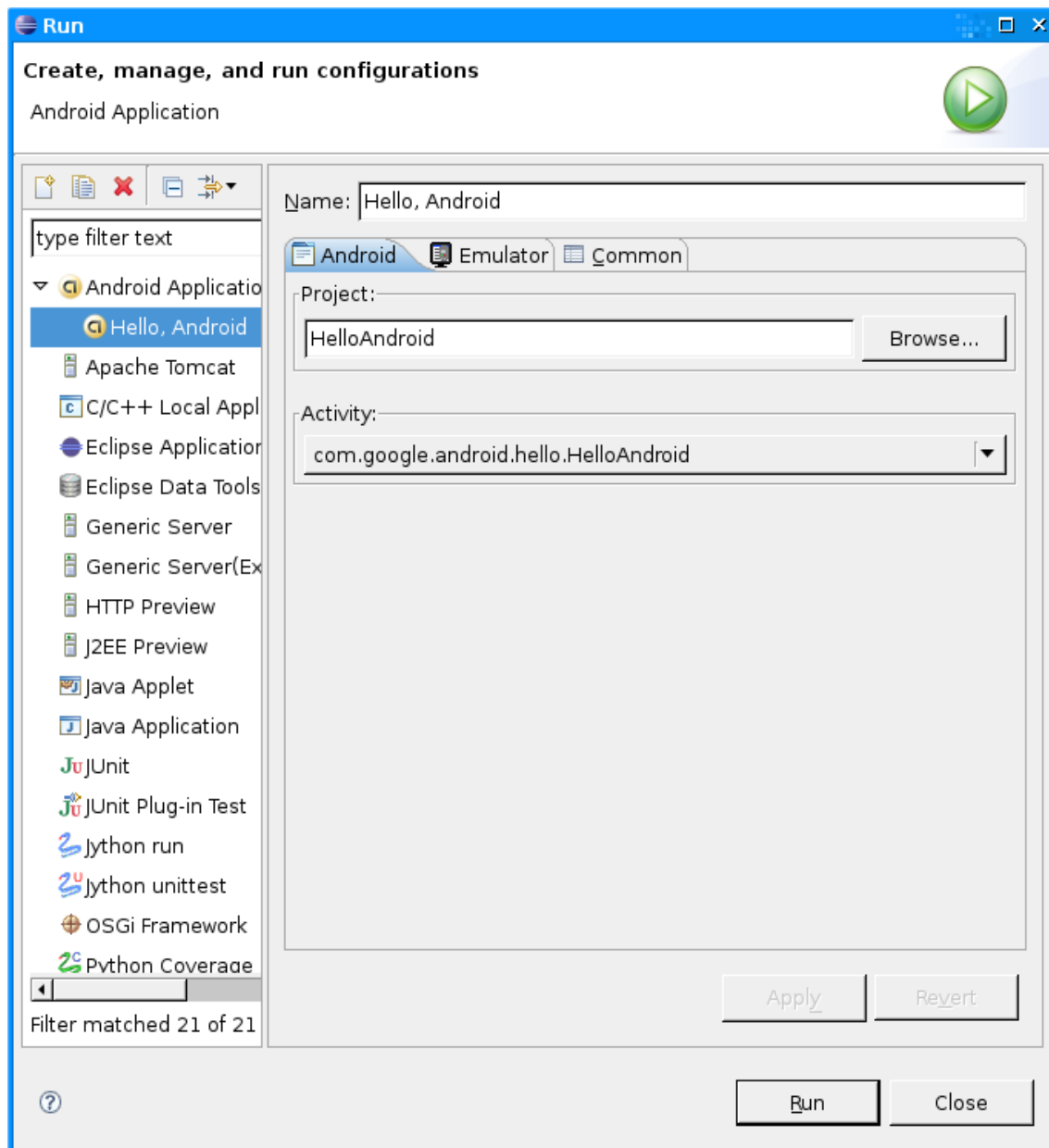
Eclipse 外掛很容易就可以執行你的程式，開始是從目錄選取 Run>Open Run Dialog，會看到



下一步選取"Android Application"，按左上角的圖示(一張紙上面有個+號的那個)，或是直接在 double click "Android Application"，會載入一個名為"New\_configuration" 的新設定



改變一下名稱，如"Android!您好"，然後按 Browse 按鈕選擇您的專案，外掛會掃描專案內的每一個 Activity 的子類別加到 Activity:標籤下的下拉方塊，因為"Android!您好"只有一個 Activity，所以預設就是它了，再來按"Apply"套用，範例如下：



就是這樣 - 您完成了，按 Run 按鈕，Android 模擬器啟動後，您的程式就會出現，看起來就像這樣



這就是"Android!您好"，非常直接，不是嗎?下一段的導覽會提供您學習 Android 更多有價值的細節

## 將界面升級到 XML 排版

剛剛您看到的"Android!您好"是我們稱之為程式化開發的界面，就是說你直接用程式碼來建構使用者界面，您可能熟悉這種方式有多麼脆弱:小小的排版變動可能導致大部分的程式變更，最容易忘記的是把正確的 View 的螢幕顯示，這可能導致錯誤並浪費時間除錯。

這也是為什麼 Android 提供另一種 UI 的建立模式:XML 排版檔，瞭解的最簡單方法就是看個範例，下面的 XML 排版檔就是跟您剛編譯的程式同樣的排版建構

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="Android!您好"/>
```

XML 的 Tag 可以自由使用，結構也很簡單，Tag 的名稱就是 View 的類別名稱，在這個範例，Tag 就只有一個，TextView，你可以使用任何繼承自 View 的類別做為 Tag 名稱，包含那些你自己所做的類別，這樣的結構容易快速建立界面，比起您的程式碼要簡單的多了，這樣的模式受到 Web 開發的啟示，將 presentation 從程式邏輯抽離只用來填入與取得資料。在這個 XML 使用了一些屬性，大致說明如下：

屬性	說明
xmlns:android	名稱空間宣告，讓 Android 工具參考 Android 名稱空間的一般屬性，大部分的 XML 排版檔必須使用這個名稱空間
android:layout_width	這個屬性宣告 View 要佔螢幕多少空間，在這個例子，唯一的 View 要佔用全部的螢幕，所以使用"fill_parent"這個值
android:layout_height	像 android:layout_width 一般，除了它是指螢幕可用的高度(This is just like android:layout_width, except that it refers to available screen height. )
android:text	設定 TextView 包含的文字，這個例子，我們用了"Android!您好"

那麼這個 XML 要放到那裡去?放到您專案目錄下的 res/路徑裡，"res"是資源的縮寫，這個目錄存放了您的專案除了程式碼以外所需要的資產，諸如：圖形、本地化字串與 XML 排版檔。

Eclipse 外掛為您建立這些 XML 檔案，在上面的例子，我們並沒有用到，在 Package Explorer 展開 res/layout 目錄，並且編輯 main.xml 檔案，變更成上面的內容後存檔。

現在打開 Package Explorer 下原始碼目錄內的 R.java 檔案，看起來像這樣

```
public final class R {
    public static final class attr {
    };
    public static final class drawable {
        public static final int icon=0x7f020000;
    };
    public static final class layout {
        public static final int main=0x7f030000;
    };
    public static final class string {
        public static final int app_name=0x7f040000;
    };
};
```

專案的 R.java 檔用來索引定義在這個檔案的內的資源，您在程式碼內使用這個類別做為方便參考包含在您專案內的資源，這在有 Code-Completion 功能的 IDE 如 Eclipse 特別有用，讓您快速互動找到特定的參考。





## 專案除錯

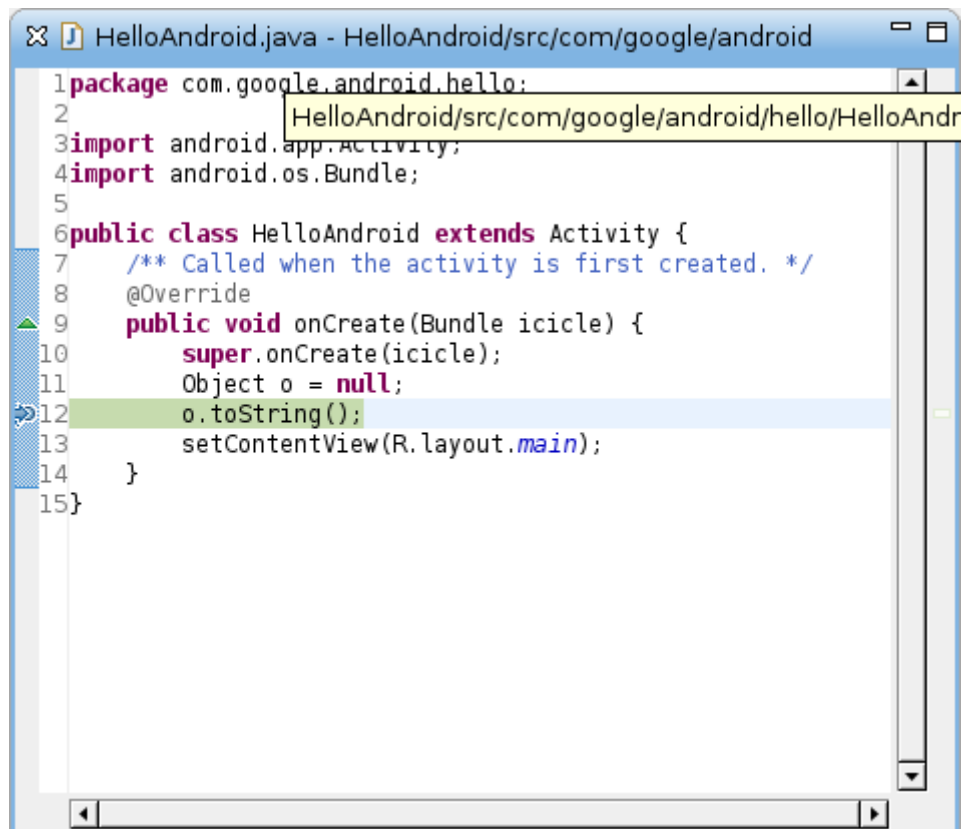
Eclipse 的 Android 外掛與 Eclipse 的除錯整合的非常好，為了展示一下，讓我們在程式碼裡插入一個 bug，修改一下您的程式碼如下：

```
public class HelloAndroid extends Activity {  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        Object o = null;  
        o.toString();  
        setContentView(R.layout.main);  
    }  
}
```

這樣的變動插入了一個 `NullPointerException`，如果您再次執行程式，最後會看到這個



要找出那裡錯了，在"Object o = null;" 那行程式碼建立中斷點(您可以在 Eclipse 行號左邊區域 double-click 建立中斷點)，然後選擇 Run > Debug 載入最後程式進入資錯模式。您的程式重啟模擬器，但這時到中斷點時會中斷，然後您可以在 Eclipse 的 Debug Perspective 一步步執行，如同除錯其它程式一樣



```
1 package com.google.android.hello;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5
6 public class HelloAndroid extends Activity {
7     /** Called when the activity is first created. */
8     @Override
9     public void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         Object o = null;
12         o.toString();
13         setContentView(R.layout.main);
14     }
15 }
```

## 以 Eclipse 建立專案

如果您不使用 Eclipse(比如您喜歡其它 IDE , 或者使用文字編輯器與命令列工具) , 那麼 Eclipse 外掛幫不了您 , 不用擔心 – 你不會因為不使用 Eclipse 而失去這些功能。

Android Eclipse 外掛只是一系列 Android SDK 工具的包裝(工具如模擬器、aapt、adb、ddms、還有其它(:todo)[記錄在文件內](#)) , 因此 , 是有可能用其它工具包裝這些工具 , 如"Ant"。

Android SDK 包含了一個 Ant 相容的 build.xml 檔案與名為"activityCreator.py" 的 python 檔案可以為您的專案建立所有的基本目錄與檔案 , 讓您可以從命令列建立您的專案 , 或者選擇整合到其它的 IDE 工具。

例如 , 用以下命令建立與 Eclipse 建立類似的專案

```
activityCreator.py --out HelloAndroid com.google.android.hello.HelloAndroid
```

為建立專案 , 您得接著執行"ant" , 然後在 bin 目錄下會產生一個 HelloAndroid.apk 檔案 , .apk 檔是一個 Android 封裝 , 使用"adb"工具可以安裝到您的模擬器。

更多使用工具的方式 , 請參考上述提到的文件。

## 剖析 Android 程式

Android 程式包含四個區塊：

- Activity
- Intent Receiver
- Service
- Content Provider

不是每一個程式都包含這四個部分，但您的程式也許會用到其中的一些組合。

一旦決定要用那些部分，您應當將之宣告在一個叫做 `AndroidManifest.xml` 的 XML 檔案，並描述其需求與功能，完整資料請詳見(:todo)[Android manifest file documentation](#)

### Activity

Activity 是 Android 四個部分最常用到的建立部位，Activity 通常是程式的一個單一螢幕(譯註:Screen 指畫面)，每一個 Activity 都是繼承自 [Activity](#) 的單一類別，您的程式顯示由 [Views](#) 組合而成的界面並回應事件，大部份的程式包含多個螢幕，例如，一個文字訊息程式會有一個螢幕顯示一系列的連絡人，第二個螢幕則用來寫訊息，其它的螢幕則是用來看舊訊息與設定，每一個螢幕都會實做為 Activity，移動到另一個螢幕，則是啟始一個新的 Activity，某些時，前一個螢幕也許會回傳一個值 – 例如：一個 Activity 讓使用者選了一個照片後，回傳給呼叫者被選用的照片。

當新螢幕被開啟，前一個螢幕被中止並被放到歷史堆疊內，使用者可以回到歷史堆疊內的前一個螢幕;螢幕不適合存在時，也會從歷史堆疊中移除，Android 為每一個程式維持一個歷史堆疊(第一頁就是主頁)。

### Intent 與 Intent Filters

Android 有一個特別用來在螢幕間移動的類別叫做 [Intent](#)

