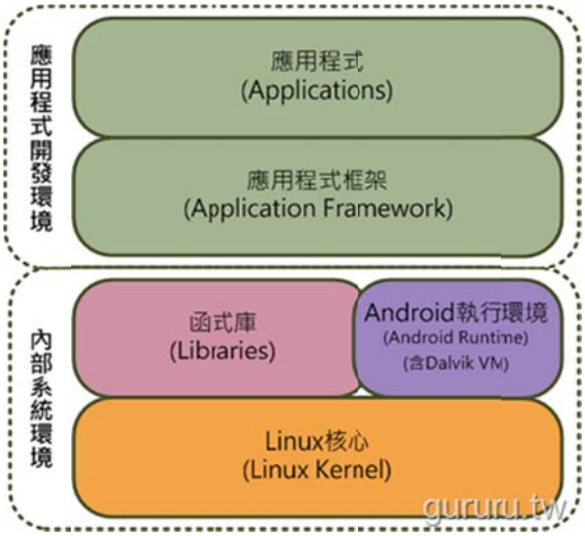
臺北市立第一女子高級中學 102 學年度資研社社課
Android App 手機應用程式設計課程

本講義參考底本：http://code.google.com/p/androidbmi/wiki
Google Android 手機應用程式設計入門 | 第五版 Gasolin 松崗

認識 Android

Android 是由 Google 領頭打造的手持式設備標準作業系統平台。由於 Android 系統完全開放，任何人都可以查看原始碼並加以應用。2010 年底調查報告指出，Android 作業系統為全球占有率最高 (32.9%) 的智慧型手機平台。

■ Android 系統基本架構



引自：http://blog.chinatimes.com/tomsun/archive/2010/08/31/533666.html

- Android 系統的底層：是以 Linux 核心為中心，包含和硬體相關的各種螢幕、鍵盤、無線網路、記憶體、音效驅動程式。
- Android 為使用者介面和系統服務提供各種函式庫 (Libraries)，方便移動設備的開發。
- Android 執行環境：主要以 Java 語言編寫，Google 特別為移動平台訂製 Dalvik 虛擬機器運行環境，可以執行編譯成 .dex 格式的應用程式。
- Android 應用程式框架和使用者介面：提供各種 API 和工具，如活動 (Activity)管理、內容 (Content Provider) 管理、資源管理、套件管理、通知管理等子系統。
- **Android 應用程式層**：提供使用者直接操作系統應用程式。開發者可以開發不同的應用程式並安裝到 Android 手持設備中。Android 應用程式以 java 語言編寫。

基礎篇 – Android Developers

參考官方網站：http://developer.android.com/sdk/index.html
參考教學影音：http://thenewboston.org/list.php?cat=6

Unit 1 安裝 Android 開發工具

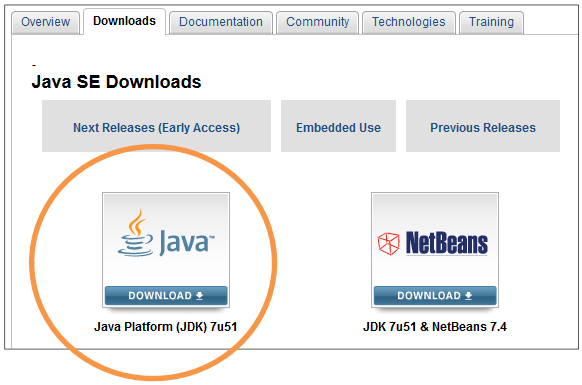
Android 開發環境所需的工具程式可由網路免費下載取得。安裝流程分以下步驟：

- 下載安裝 Java 開發工具 (Java SE Development Kit, JDK)
- 下載並安裝 Android 開發工具 (ADT 套件-Android Development Kit)
- 下載 Android SDK (程式開發組件 Software Development Kit)

Step 1: 下載安裝 Java 開發工具 (Java SE Development Kit, JDK)

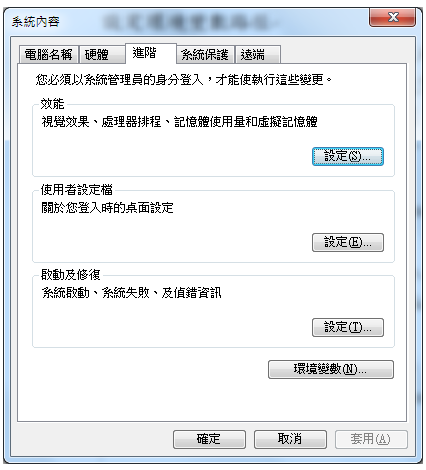
[1] 下載並安裝 Java 開發套件

前往「http://www.oracle.com/technetwork/java/javase/downloads/index.html」網站，點選 Java Platform (JDK) 7u51。下載、安裝。



[2] 設定環境變數路徑

- 到「我的電腦」找到 Java 安裝的路徑 (example: "C:\Program Files\Java\jdk1.7.0_51\bin" 。複製下來即為 javac 路徑(Path)。
- 「開始」/「電腦」按右鍵 /選「內容」
- 視窗左側選「進階系統設定」，點選下方的「環境變數」

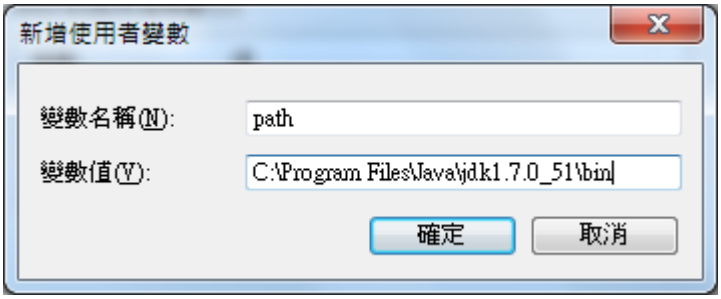


- 新增使用者變數：按下「新增…」

  設定變數名稱：Path

  設定變數值：貼上 javac 的位置

  例如：C:\Program Files\Java\jdk1.7.0_51\bin

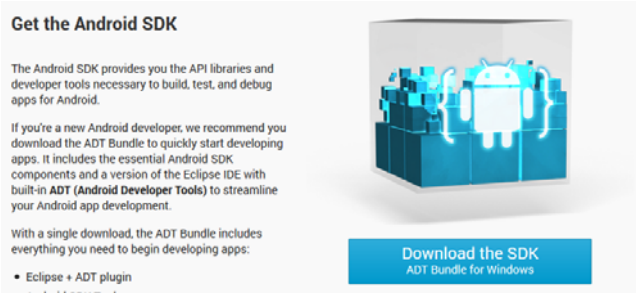


- 按下「確定」
- 測試一下：開啟命令列模式（「開始」/ 輸入「cmd」）。進入命令提示字元，輸入 javac 。若能看見說明文件即表示環境變數設定成功。

Step 2: 下載並安裝 Android 開發工具 (ADT 套件-Android Development Kit)

[1] Get the Android SDK：

前往「http://developer.android.com/sdk/index.html」網站，按下「Download the SDK -ADT Bundle for Windows」取得 SDK 完整開發程式套件



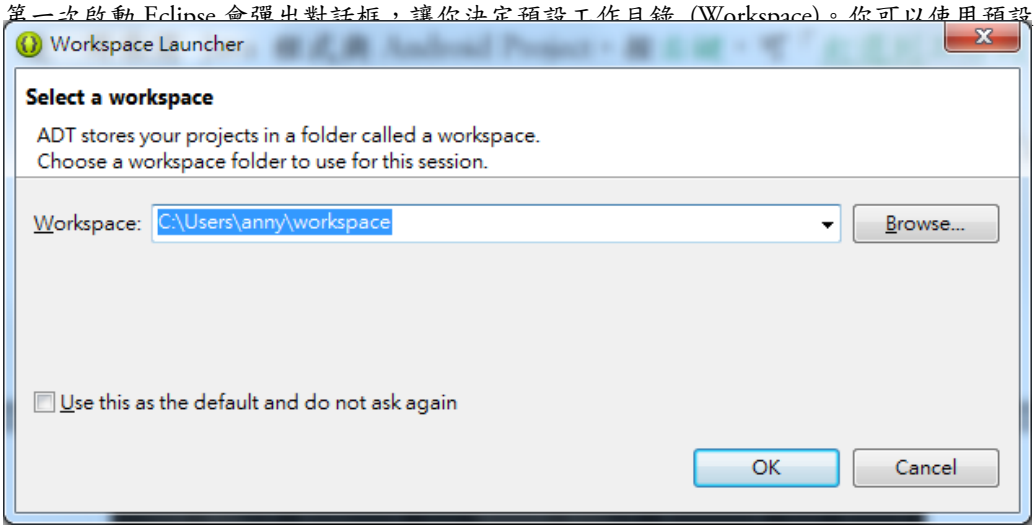
ADT 套件(ADT Bundle) 包含完整的 Eclipse 整合開發環境、Android 開發工具擴充套件 (ADT Plugin)、與 Android SDK 工具

[2] ADT 套件下載後，不需安裝，只要解壓縮到 C:\ 即可執行。可將解壓縮的資料夾取名為「adt」。裡面有「eclipse」和「SDK」兩個資料夾。

[3] 裡面的「eclipse」子目錄中的「Eclipse.exe」執行檔即為 Eclipse 的整合開發環境，用來寫 Java 程式與 Android Project。按右鍵，可「釘選到工作列」上。

[4] 啟動 Eclipse：

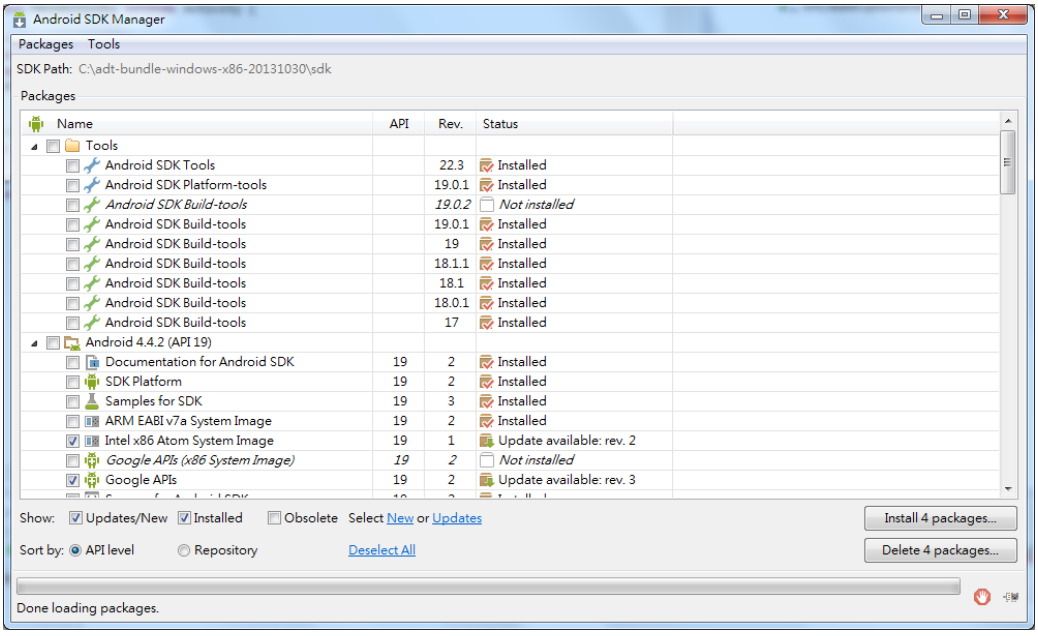
第一次啟動 Eclipse 會彈出對話框，讓你決定預設工作目錄 (Workspace)。你可以使用預設



Step 3:下載 Android SDK (程式開發組件 Software Development Kit)

接著，必須開啟 Android SDK 管理工具，下載最新的可用組件列表。下載完後，在「Packages」列表中，可以看到目前可用的目標 SDK 平台(platform)、文件、範例與 Google APIs 列表。

[1] 選單上選「Window > Android SDK Manager」 選項，開啟 Android SDK 管理工具。



每個組件後面的「Status」欄位會顯示各組件目前的安裝狀態。安裝狀態分為「Installed (已安裝)」、「Not Installed (未安裝)」、和「Upgrade (可更新)」。

[2] 開始安裝：

勾選對應版本， 如 Android 4.4.2 (API 19) 的目標 SDK、文件、範例、Google API 組件，然後按下「Install x packages」(安裝選取項目) 按鈕即可。 (x 代表將安裝的數目)

[3] Android 多個 SDK 版本可以並存，但一般開發時只需選擇最新版本的 SDK，就能開發向前版本相容的 Android 應用程式。

[4] 下一個視窗，可確認預備安裝的 SDK 組件。勾選「Accept ALL」(接受全部授權宣告)選項，按下「Install」（安裝）按鈕，即開始自動下載並安裝。

[5] 安裝完成後，可看到剛剛勾選的 SDK 組件都已經變成「Installed」(已安裝)。

Unit 2 設定 Android 虛擬機器 (AVD, Android Virtual Device)

「Android 開發工具」(ADT)提供了支援不同版本虛擬機器的功能。在使用虛擬機器之前，必須先建立一個虛擬機器，才可在 Eclipse 開發環境中使用。

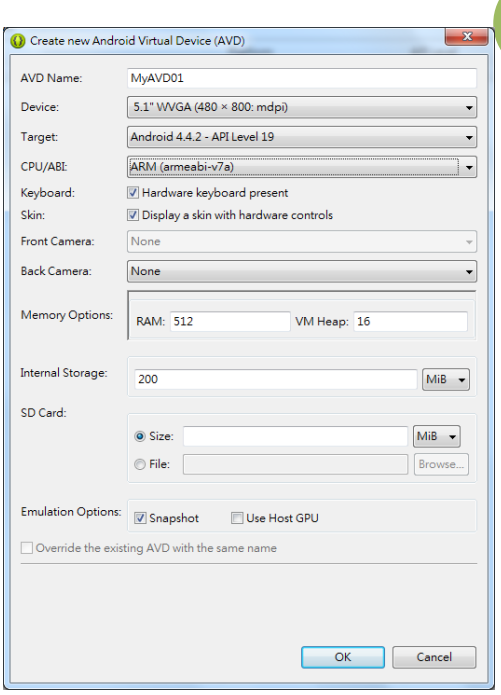
建立新的虛擬機器：

[1] 開啟「Window > Android Virtual Device Manager」 。

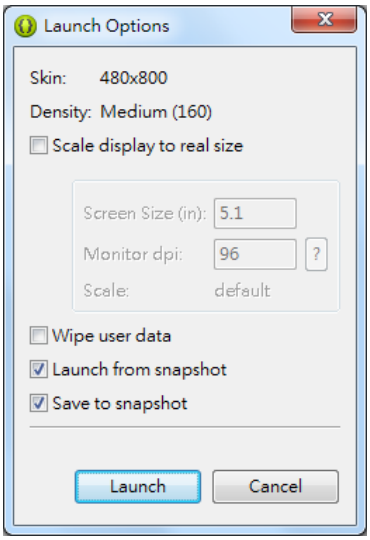
[2] New 一台虛擬機器：

點選右側的「New…」按鈕，會彈出「Create New Android Virtual Device (AVD)」 視窗。

- 輸入虛擬機器名稱(AVD Name): 例 MyAVD01
- 選擇手持裝置(Device): 例 5.1"WVGA
- 選擇適當的目標版本(Target): 例 Android 4.4.2 – API Level 19
- 選擇 CPU: 例 ARM
- 勾選「Snapshot」(加速)
- 按下「OK」按鈕。即可建立新的虛擬機器



[3] 啟動虛擬機器：建立好後，點選妳的虛擬機器，按下「Start…」，在彈出的視窗點選「Launch」 按鈕，即可啟動虛擬機器。



Unit 3 建立一個 Android 應用程式

3-1：建立新程式專案 BMI

[1] 回到 Eclipse 整合環境

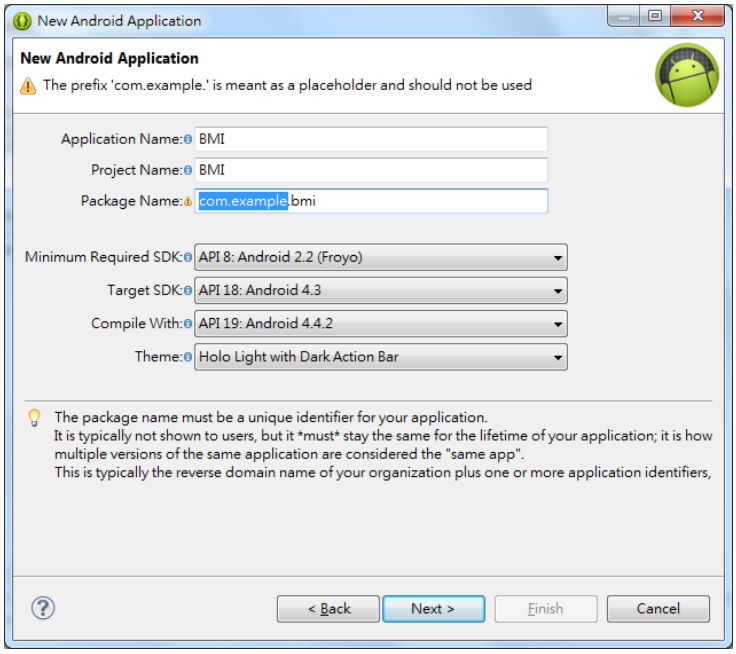
[2] 螢幕上方選單列「File > New > Project… > Android Application Project」

[3] 開啟「New Android Application」對話視窗：

Application Name: BMI

Project Name: BMI

Package Name: com.example.bmi



設定畫面說明：

名稱	描述
Application Name	應用程式顯示在螢幕上的標題
Project Name	這個專案所在的資料夾名稱
Package Name	套件(package)名稱
Compile With	選擇用來編譯專案的目標 SDK 版本
Minimum Required SDK	本應用程式所支援的最低 SDK 版本代號

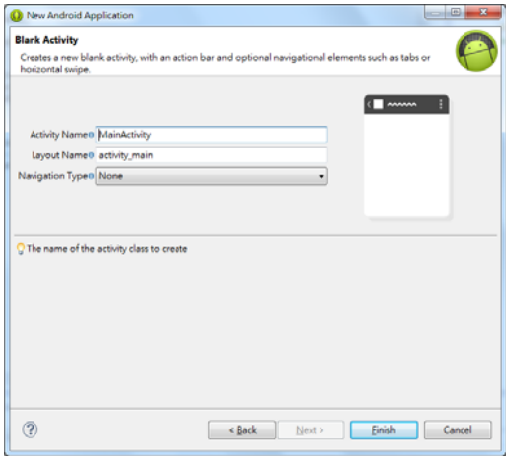
[4] 按「Next」，前往專案設定畫面。

- 勾選「Create custom launcher icon」（建立 App 圖示）
- 勾選「Create activity」(產生專案程式碼)
- 勾選「Create project in Workspace」（自訂儲存專案的位置。預設值即可）

[5] 選擇 Activity 樣式：Blank Activity

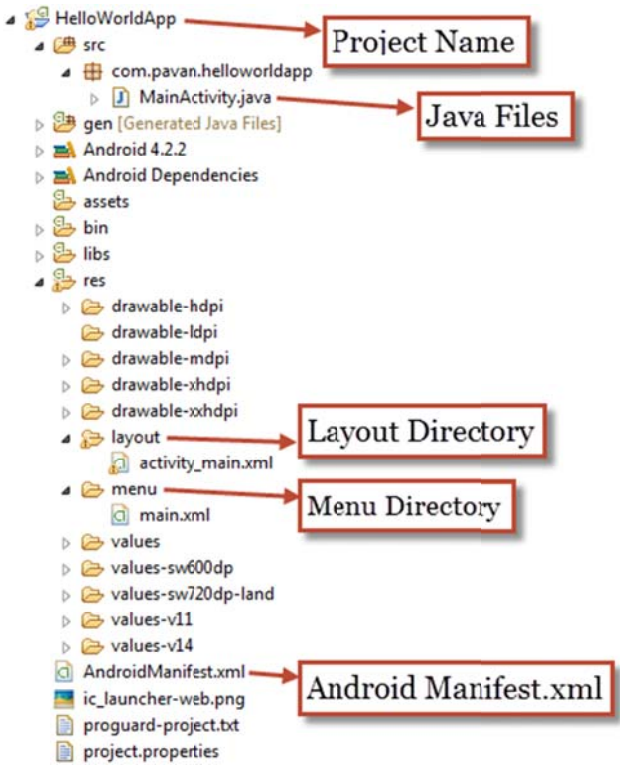
[6] 建立新的 Activity：

名稱	預設值	說明
Activity Name	MainActivity	建立的 Activity 類別名稱
Layout Name	activity_main	預設的 layout XML 檔案名稱
Navigation Type	None	畫面上提供的導覽方式



[7] 按下「finish」按鈕即完成建立應用程式專案的工作

3-2：程式專案架構



src/: 原始檔目錄

gen/: 自動生成(Generate)目錄

R.java 中產生的 R 類別，就像是資源字典大全。包含了使用者介面、圖像、字串等對應各式資源的識別符號(id).

你完全不需要，也避免手工修改。

res/: 資源(Resource)目錄

存放「layout」,「values」與「drawable」

Layout: 版面配置目錄

Values: 參數值目錄

Drawable: 圖片目錄

HW01- 下載安裝 Android APP 開發環境、新增 2 個 AVD (手機、平板)、新增 2 個程式專案 (CtoF, FtoC, TriangleArea, ...)

Unit 4 描述使用者介面

1. 設定並執行模擬器，看看模擬器運作後的結果

我們看到一個文字欄位，上面有一串文字 「Hello World!」。這就是 Android 預設程式架構的範例囉。

由於才剛開始實際接觸到 Android 應用程式，我們先從簡單的開始：現在，我們的目標是將「Hello World!」 換成別的文字。

那麼，「Hello World!」，這串字串藏在哪裡呢？

2. 點選視窗左下角，切換到「activity_main.xml」標籤

它是使用者介面的描述檔。我們會看到以下程式碼：

```
1    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2         xmlns:tools="http://schemas.android.com/tools"
3    android:layout_width="match_parent"
4    android:layout_height="match_parent"
5    android:paddingBottom="@dimen/activity_vertical_margin"
6    android:paddingLeft="@dimen/activity_horizontal_margin"
7    android:paddingRight="@dimen/activity_horizontal_margin"
8    android:paddingTop="@dimen/activity_vertical_margin"
9    tools:context=".MainActivity" >
10
11   <TextView
12        android:layout_width="wrap_content"
13        android:layout_height="wrap_content"
14        android:text="@string/hello_world" />
15
16   </RelativeLayout>
```

原來「Hello World, Bmi!」字串就藏在「res/layout/main.xml」這個檔案的第 10 行中。

3. **修改看看：**
   我們只要簡單地將第 10 行修改成 android:text=**"Hello World, BMI!"**

4. **看看結果：**
   切換回「Graphical Layout」，或再執行一次模擬器，就可以得到一個相似的應用程式，只是內文變成了我們剛剛修改的內容。

5. 再改改看：
   既然找到了「Hello World, BMI!」字串，我們就試著將「android:text」屬性值從「Hello World, BMI!」改成「哈囉，BMI」，然後執行看看吧。

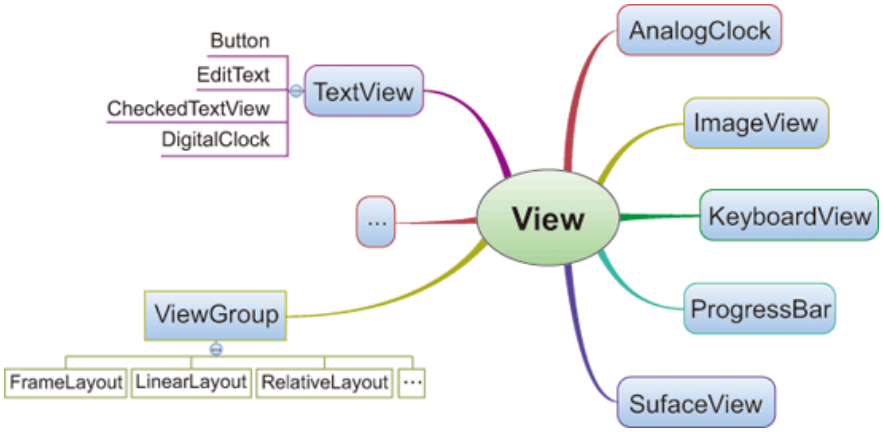
   android:text="哈囉，BMI"

   結果發現 Android 模擬器中文嘛也通，字型也相當漂亮。

4-1 Android App 的使用者介面說明：

Android App 的使用者介面由 XML 描述檔案設定，也可以由圖形化版型介面設計工具(Graphical Layout Editor)，經過元件拖、拉、放的方式，做簡單版面設計工具。

在 Android 的應用程式架構中，提供了一套使用者介面（User Interface；UI）架構給予開發者建立 Android 平台上各種 UI，對每一個 Android 應用程式內的使用者介面來看，皆是由不同的 View 與 ViewGroup 物件所構成，在 Android 使用者介面架構內提供了多種不同的 View 與 ViewGroup 類別，如圖一所示，這些 Android UI 架構內所定義的基礎類別，分別提供給開發者針對使用者介面設計、開發及管理使用。



取自：http://www.hope.com.tw/art/print.asp?O=HJTBJ8U01E0AR-STDV

要開始學習 Android 應用程式確實很簡單吧？不過為了顯示「Hello World, BMI」，也用到了許多程式碼。到底這些程式碼有什麼含意呢？

4-2 講解 active_main.xml 檔案內容

XML (Extensible Markup Language) 是一種標記描述語言，不管是語法還是看起來的樣子，都相當類似網頁所使用的 HTML 標記語言。XML 被廣泛地運用在 Java 程式的設定中，副檔名為「.xml」結尾的檔案通常就是包含 XML 語言描述的檔案。

「active_main.xml」文件裡，第一行是每個 XML 描述檔固定的開頭內容，用來指示這個文字檔案是以 XML 格式描述的。

```
1   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2        xmlns:tools="http://schemas.android.com/tools"
3   android:layout_width="match_parent"
4   android:layout_height="match_parent"
5   android:paddingBottom="@dimen/activity_vertical_margin"
6   android:paddingLeft="@dimen/activity_horizontal_margin"
7   android:paddingRight="@dimen/activity_horizontal_margin"
8   android:paddingTop="@dimen/activity_vertical_margin"
9   tools:context=".MainActivity" >
10
11  <TextView
12       android:layout_width="wrap_content"
13       android:layout_height="wrap_content"
14       android:text="@string/hello_world" />
15
16  </RelativeLayout>
```

1. 第 1 行與第 16 行

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

</RelativeLayout>
```

接著我們看到第一個標籤，與 HTML 網頁標籤相當類似。

```
<RelativeLayout></RelativeLayou>
```

"相對位置版面配置"(RelativeLayout)標籤，使用了兩個「RelativeLayout」標籤，來表示一個介面元件的區塊。後頭的標籤前加上一個「/」符號來表示結束標籤。"相對位置版面配置" 所指的是包含在 「RelativeLayout」 標籤中，所有元件的配置方式，是根據介面元件之間關係排列的意思。


```
xmlns:android="http://schemas.android.com/apk/res/android"
```

以「xmlns」 開頭的這串敘述，是用來宣告這個 XML 描述檔案的的名稱空間(NameSpace)，後面接的 URL(網址)，表示這個描述檔案會參照到 Android 名稱空間提供的定義。 所有 Android 版面配置檔案的最外層標籤中，都必須包含這個屬性。

注意標籤需要兩兩對稱。一個標籤「<RelativeLayout>」在一串敘述的前頭，另一個標籤「</RelativeLayout>」在敘述的末尾。 如果你修改過的標籤沒有閉合(忘了加 <、/、> 等符號)，Eclipse 畫面上也會出現小小的警示符號來提醒你。

2. 第 3-4 行

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

這些包含在「< RelativeLayout >」標籤中的敘述被稱為「RelativeLayout」標籤的「屬性」。Android 應用程式在 layout 目錄中的標籤，大多數的屬性前都有一個「android:」前綴。同一個介面元件的屬性之間，是以空白做區隔，因此事實上你也能將多個屬性寫在同一行。當然，將屬性排成多行更易於閱讀。我們應該保持這個好習慣。

介面元件都有許多共同的屬性，例如介面元件的長，寬度設定屬性。
Android 介面元件的寬度、長度設定屬性分別叫做「android:layout_width」、「android:layout_height」。兩個都設定為 「match_parent」參數值。

「match_parent」 如其名，所表達的的意思就是"配合上層元件大小"。
預設 「RelaveLayout」 介面元件就會佔滿整個螢幕空間。

介面元件彼此間也會有一些不同的屬性，例如 「LinearLayout」(線性版面配置) 標籤的「android:orientation」(版面走向) 屬性。填入 「vertical」 (垂直)屬性值，表示這個介面的版面配置方式是從上而下垂直地排列其內含的介面元件。

「android.view.ViewGroup」 是各種佈局配置(layout)元件的基礎類別。常見的有

- LinearLayout(線性版面配置)
- FrameLayout(框架版面配置)
- TableLayout(表格版面配置)
- AbsoluteLayout(絕對位置版面配置)
- RelativeLayout(相對位置版面配置)

雖然有這麼多種版面配置方式可以選用，但大多數的應用程式並不需特地去改變預設的「RelaveLayout」的配置，只要專注在其中填入需要的介面元件即可。

決定配置方式後，標籤內的內容才是一般應用程式開發時較常修改之處。

3. 第 11 和 14 行

```
    <TextView
       />
```

「TextView」（文字檢視）是我們看到的第一個熟悉的介面元件。其作用是顯示文字到螢幕上。你可能注意到這個標籤結尾使用了 「/>」 符號。 「/>」符號表示這個 XML 敘述中沒有內文，亦即此介面元件描述中不再包含其他介面元件，也表示這個介面元件就是這個螢幕中最小的組成單元了。

4. 第 6-9 行

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/hello_world""
```


我們來看看 「TextView」 介面元件中包含了哪些屬性。

[1] 「android: layout_width」和「android:layout_height」我們剛剛已經學過了，分別代表寬度跟長度。

[2] 兩個屬性都用到一個新的參數值「wrap_content」(包住內容)，亦即隨著文字欄位行數的不同而改變這個介面元件的寬度和高度。

[3] 最後的 「android:text」 屬性則是 「TextView」 介面元件的主要屬性，亦即文字欄位中顯示的文字內容。

[4] 至於「@string/hello_world 」這段字串所代表的意義，馬上會接著在後面章節說明。我們現在已知道是：只要將「android:text」屬性內容替換成我們想要文字，在預覽畫面或在模擬器中就會顯示對應的文字。

將以上的 XML 描述綜合起來，我們就可以得知「active_main.xml」 想表達的介面。

Unit 5 設計使用者介面

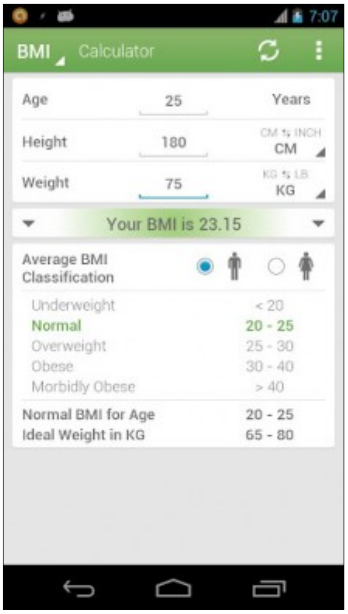
5-1 視圖(View)

軟體設計的過程中，常常會遇到需要頻繁修改使用者介面的情境。改著改著程式設計師們就累積起了一些經驗，也歸納出了許多應對之道。如著名的 MVC（Model-View-Controller） 模式。

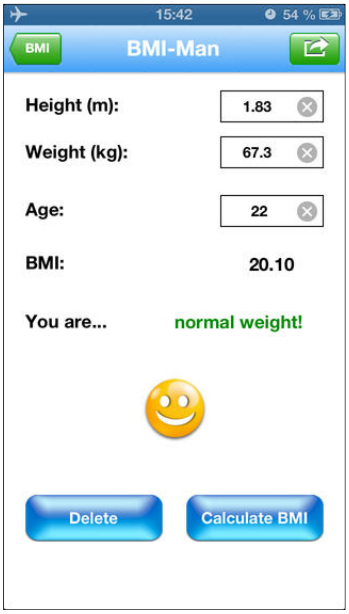
Google Android 為我們考慮了介面修改問題。Android 為了單純化介面修改方式，採用了目前比較流行的解決方案--即將介面描述部份的程式碼，抽取到程式外部的 XML 描述文件中。

我們在前面的過程裡已經學到，如何在 Android 應用程式中替換 「TextView」 介面元件所顯示的純文字內容。那麼...這個經驗能不能直接用到 BMI 應用程式的設計上呢？

我們先回過頭來思考， 要實現基本的 BMI 計算功能，在螢幕上至少需要哪些介面元件，以供使用者輸入數值和查看結果？



http://cdn1.tnwcdn.com/wp-content/blogs.dir/1/files/2012/09/unnamed-3-220x391.jpg



Iphone BMI APP
https://itunes.apple.com/de/app/bmi/id393742293?mt=8

為了輸入 BMI 程式所需的身高體重值，大致上我們需要兩個 「TextView」 元件用來提示填入身高 Height(cm)、體重 Weight(Kg) 數字。

另外也需要兩個文字輸入欄位元件「EditText」用來填入身高體重數字。

我們還需要一個**按鈕(Button)**來開始計算，而計算完也需要一個 「TextView」 元件來顯示計算結果。

於是初版的「BMI 應用程式」介面的樣子就浮現出來了。

5-2 開始設計

使用視覺化介面設計工具輔助編輯「activity_main.xml」檔案。

1、 輸入介面設計

定義一個基本 BMI 程式所需的身高(Height)輸入欄位，就會用到 EditText，與 TextView 介面元件，其描述如下：

「TextView」介面元件：

```
<TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="身高 (cm)"
/>
```


「EditText」介面元件：

```
<EditText android:id="@+id/height"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:numeric="numberDecimal"
        android:text=""
/>
```

說明：

[1] EditText 介面元件描述的基本的組成與 TextView 介面元件相似，都用到了「android:layout_width」與「android:layout_height」屬性。

[2] 指定的另外兩個屬性「android: numberDecimal」、「android:text」則是 EditText 介面元件的特別屬性。

[3] 「android:text」屬性是繼承自 TextView 介面元件的屬性。

android:numeric=" numberDecimal "

android:text=""

[4] 將 「android:numeric」 指定為 「numberDecimal」，可以限制使用者只能在 EditText 文字欄位中輸入十進位數字。

[5] 「android:text」屬性則是指定 EditText 介面元件預設顯示的文字(數字)。

2、 按鈕介面設計

我們再來看看 Button (按鈕)介面元件

```
<Button android:id="@+id/submit"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="計算 BMI 值"
    />
```

說明：Button 介面元件同樣有 「android:layout_width」與「android:layout_height」屬性，另外一個「android:text」屬性則用來顯示按鈕上的文字。

3、 整合

我們這就從文件中挑出我們需要的 TextView(文字檢視)、EditText(編輯文字)、Button(按鈕) 三種介面元件，照前面的設計擺進 RelativeLayout (版面配置)元件中。

4、 完整的「activity_main.xml」介面描述檔

```
<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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:text="@string/bmi_height" />

    <EditText
        android:id="@+id/height"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView1"
        android:layout_below="@+id/textView1"
        android:inputType="numberDecimal" >
        <requestFocus />
    </EditText>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/height"
        android:layout_below="@+id/height"
        android:text="@string/bmi_weight" />

    <EditText
        android:id="@+id/weight"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView3"
        android:layout_below="@+id/textView3"
        android:inputType="numberDecimal" />

    <Button
        android:id="@+id/button_bmi"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/weight"
        android:layout_below="@+id/weight"
        android:text="@string/bmi_btn" />

    <TextView
        android:id="@+id/text_result"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/button_bmi"
        android:layout_below="@+id/button_bmi"
        android:text="@string/bmi_result"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <TextView
        android:id="@+id/text_suggest"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/text_result"
        android:layout_below="@+id/text_result"
        android:layout_marginTop="36dp"
        android:text="@string/bmi_suggest"
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```

5、 檢視結果

我們可以啟動模擬器檢視執行結果。或是在頁面標籤下選擇「Layout」標籤，來預覽頁面配置。啟動模擬器之後，模擬器畫面上出現了兩個輸入欄位。欄位上方分別標示著「身高 (cm)」、「體重 (kg)」。在兩個輸入欄位下方，是一個標示著「計算 BMI 值」的按鈕。當你在欄位中試著輸入文字或數字(你可以直接用電腦鍵盤輸入，或按著模擬器上的虛擬鍵盤輸入)時，你也會發現，正 XML 描述檔的描述中對兩個 EditText 欄位所規定的，欄位中只能輸入數字。

我們在上面 XML 描述檔中定義的最後兩個 TextView 介面元件，由於並未替這兩個介面元件指定「android:text」屬性，所以在螢幕上並未顯示。這兩個介面元件在後面會派上用場。

6、 革命的路還長

高興了沒多久，你發現按下"計算 BMI 值" 按鈕後，應用程式完全沒反應。
這是正常的，因為我們還沒處理從介面輸入取得身高體重、將數值導入 BMI 計算方式、將結果輸出到螢幕上...等等 BMI 應用程式的關鍵內容。 不過在進入了解程式流程之前，我們還有一個「android:id」屬性尚未解釋哩。 接著我們將透過講解「android:id」屬性，來進一步了解 Android UI。

7、 視覺化的介面開發工具

目前的 ADT 版本提供了預覽介面的功能，但尚未提供方便地視覺化拖拉介面元件的開發工具。以後也許 ADT 會加入完整的 GUI 拖拉設計工具。
但在 ADT 加入完整的 GUI 拖拉設計工具之前，已經有人寫出來了對應 Android 的 GUI 拖拉設計工具，可供使用。
DroidDraw - Android GUI 拖拉設計工具 http://code.google.com/p/droiddraw/

Homework#2: 開啟 2 個專案，設計類似 BMI 的應用程式介面。例如：面積計算器、我的幸運數字、利息計算器…

Unit 6 程式設計

接著要觀察主要程式邏輯的內容囉。

6-1 預設程式內容

打開「src/com/demo/android/bmi」目錄下的「MainActivity.java」檔案，Eclipse+Android 開發工具已經幫我們預先建立好了基本的程式邏輯。其預設的內容如下：

```
1  package com.demo.android.bmi;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5
6  public class MainActivity extends Activity {
7      /** Called when the activity is first created. */
8      @Override
9      public void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12     }
13 }
```


講解

比起什麼標籤都對稱的 XML 介面描述檔來說，這個以 Java 程式語言寫成的檔案雖然篇幅短，但反而要難讀得多。

我們將程式直接拆開，分成幾個部份來講解這個「MainActivity.java」檔案的內容：

第 1 行:

```
package com.demo.android.bmi;
```

這一行的作用是指出這個檔案所在的名稱空間。「package」(套件)是其關鍵字。使用名稱空間的原因是程式一旦擴展到某個大小，程式中的變數名稱、方法名稱、類別名稱難免重複， 這時就可以將定義的名稱區隔管理在 package 下，以避免相互衝突的情形發生。Java 的 package 設計成與檔案系統結構相對應，如我們的 package 設定是「package com.demo.android.bmi」，則這個類別就該在指定目錄的「com/demo/android/bmi」路徑下可以找到。

同時也別忘了 Java 程式語言每段敘述語句的結尾處，與大部分的程式語言一樣需加上一個分號「;」，以表示一行程式敘述的結束。

第 3,4 行:

```
import android.app.Activity;
import android.os.Bundle;
```

程式中預設導入了「android.app.Activity」跟「android.os.Bundle」兩個 Package，在所有的 Android 應用程式中都會用到這兩個 Package。「import」(導入)是用作導入 Package 的關鍵字。在 Java 語言中，使用到任何 API 前都要事先導入相對應的 Package。我們馬上將學到這兩個 Package 的用途。

完整的 API 可查閱官方的 package 列表： http://code.google.com/android/reference/packages.html

第 6,13 行:

```
public class MainActivity extends Activity {
}
```

第 6 行開始了程式的主體。其組成是這樣的：

```
public class MainActivity
```

「MainActivity」是這個類別的名稱。「class」則是用作宣告類別關鍵字。「public」關鍵字是用來修飾「MainActivity」這個類別。表示「MainActivity」是個「公開」的類別，可以從 package 外部取用。

「public class MainActivity」後面再加上「extends Activity」敘述，則表示 「MainActivity」 這個類別的功能、型別等全**繼承**自「Activity」類別。「extends」是繼承(Inherit)類別的關鍵字。「Activity」是來自於我們在第 3 行剛導入的 Package。

因此整句話的含意即：「**宣告一個公開的 MainActivity 類別。這個 MainActivity 類別繼承了程式開頭導入的 Activity 類別**」。

「{}」大括號規範了一個程式區塊。大括號中的程式表達的這個程式區塊的主要內容。

第 7 行:

```
    /** Called when the activity is first created. */
```

第 7 行提供了位於其下的函式的註釋。「/**/」是 Java 語言的多行註解符號，位於其中的文字內容不會被編譯。「/*」敘述後多出來的一個「*」號被視為內文。順便提醒一下，Java 程式語言中兩個斜線「//」表示的是單行註解符號。單行註解符號「//」與多行註解符號「/**/」不同的地方是，只有與「//」符號同行的文字才不會被編譯。

第 8-9, 12 行:

```
@Override
public void onCreate(Bundle savedInstanceState) {
}
```

第 9 行開始了這個方法(Method)的主體。其組成是這樣的：

```
public void onCreate(Bundle savedInstanceState) {
}
```

「onCreate」是這個方法的名稱。「void」則是宣告了這個方法的回傳值的型別(type)。「public」關鍵字是用來修飾「onCreate」這個方法。表示「onCreate」是個「公開」的方法，可以由 MainActivity 類別外部取用。

方法的回傳值的型別，即是這個方法的型別。「onCreate」這個方法使用「void」型別，表示「onCreate」這個方法不需回傳值。

同時，這個方法傳入了一個名為「savedInstanceState」的「Bundle」型別參數，「Bundle」型別正是來自我們前面所導入的 Package 之一。我們並不需要知道太多「Bundle」型別或「savedInstanceState」實體的細節，只要知道「Bundle」的內容與手機平台的記憶體管理有關。

當 Android 應用程式啟動、換到背景等待、關閉時，都會用到 「savedInstanceState」 這個實體來處理記憶體相關的事宜。當然，你也可以用其他名稱來代替它。還好「onCreate」這個方法永遠都是傳入「Bundle savedInstanceState」這個參數，寫應用程式時只要正確照規定傳入即可，你可以不用太去在意它。

「Bundle」可以保存程式上一次關閉(凍結)時的狀態。你可以透過覆寫 onFreeze 方法(與 onCreate 方法的作用類似) 來保存凍結前的狀態。 當程式啟動（Activity 重新初始化）時，會再次呼叫 onCreate 方法，你就能從 savedInstanceState 中得到前一次凍結的狀態。我們也可以透過「Bundle」來將這個 Activity 的內容傳到下一個 Activity 中。

「{}」大括號規範了一個程式區塊。大括號中的程式表達 onCreate 這個程式區塊的主要內容。

```
@Override
public void onCreate(Bundle savedInstanceState)
```

從前面的講解中，我們學到了在任何一個 Android 專案目錄裡，只要打開「Referenced Libraries」目錄的「android.app」 分類，都可以找到 「Activity.class」這個類別。現在我們再深入一些查看「Activity.class」 類別。你要做的，只是依照圖示，找到 Android 工具中的「Referenced Libraries」目錄，從「android.app」 分類裡找到「Activity.class」類別，並按下「Activity.class」 類別左側的三角形圖示，如此即可展開這個類別的屬性/方法列表。

我們在 Activity 類別的屬性/方法列表中，發現了現在正要講解的 onCreate 方法(Method)。

因為「MainActivity」 類別繼承自 Activity 類別，所以預設「MainActivity」 類別中其實已經有「onCreate」方法了。

事實上，「onCreate」 方法正是每個 Activity 類別初始化時都會去呼叫的方法。「@」開頭的語句表示裝飾子(decorator)語句，「@Override」語句的作用是**告訴程式我們要覆寫這個「onCreate」方法**。當我們打開程式時，程式不再使用從「MainActivity」 類別中繼承來的「onCreate」方法，而是使用我們在程式中自訂的行為。

```
@Override
public void onCreate(Bundle savedInstanceState) {
}
```

我們講解了整段程式，其含意是「**覆寫 MainActivity 類別中公開的 onCreate 方法。這個「onCreate」方法無回傳值型別，而且這個方法傳入了一個名為「savedInstanceState」的 Bundle 型別參數。**

現在來看看「onCreate」方法中包含的程式內容。

第 10, 11 行:

```
super.onCreate(savedInstanceState);
```

「super」是關鍵字。代表著這個 「MainActivity」 類別的**上層類別(Activity)**。
「super.onCreate(savedInstanceState);」的意思就是：**「執行 Activity 類別中 onCreate 方法的內容」**。

這麼做的目的是什麼呢？

Google Android 將其應用程式的介面稱為視圖(View)，而負責控制各種動作行為的程式主體(Controller)，則稱為活動(Activity)。因此一個 Android 應用程式，必定會對應到一個以上的Activity。「onCreate」 方法則是每個 Activity 類別初始化時都會去呼叫的方法。我們想做的事，是保持原本「onCreate」 方法預設的動作，然後在其中加入我們想要的內容。

而 Android 產生的程式預設卻覆載(@Override)了「MainActivity」 類別的「onCreate」 方法。原本繼承自「Activity」類別的「onCreate」方法，其原本內容都被覆載掉了。因此想將原本的「onCreate」方法內容保留，並在其中加入我們的內容的話，就要使用「super」語句。當程式運行到我們覆寫的「onCreate」 方法時，透過「super.onCreate(savedInstanceState);」語句，會先將原本「Activity」類別中的「onCreate」方法執行一次，然後再執行我們覆寫的「onCreate」方法裡面其他的程式內容。

我們要執行原本的「onCreate」 方法時，仍然需要提供原本「onCreate」方法所需的傳入參數。因此「super.onCreate(savedInstanceState);」語句中，我們將「savedInstanceState」這個參數傳入原本的「onCreate」函式中。「savedInstanceState」是我們在「public void onCreate(Bundle savedInstanceState)」語句中所宣告的傳入參數。

```
setContentView(R.layout. activity_main);
```

透過螢幕顯示的各種元素是按照介面層次結構來描述的。要將一個顯示元素的層次結構轉換顯示到一個螢幕上，Activity 會呼叫它用來設定 View 的「setContentView」方法，並傳入想引用的 XML 描述文件。當 Activity 被啟動並需要顯示到螢幕上時，系統會通知 Activity，並根據引用的 XML 文件敘述來描繪出使用者介面。上一章中我們定義好的 res/layout/ *activity_main*.xml 描述檔，就是透過這個機制繪出到螢幕上。

6-2 完整 BMI 程式

完整的程式如下：

```
1  package com.demo.android.bmi;
2
3  import java.text.DecimalFormat;
4
5  import android.app.Activity;
6  import android.os.Bundle;
7  import android.view.View;
8  import android.view.View.OnClickListener;
9  import android.widget.Button;
10 import android.widget.EditText;
11 import android.widget.TextView;
12
13 public class MainActivity extends Activity {
14     /** Called when the activity is first created. */
15     @Override
16     public void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         //Listen for button clicks
21         Button button = (Button)findViewById(R.id.submit);
22         button.setOnClickListener(calcBMI);
23     }
24
25     private OnClickListener calcBMI = new OnClickListener()
26     {
27         public void onClick(View v)
28         {
29             DecimalFormat nf = new DecimalFormat("0.00");
30             EditText fieldheight = (EditText)findViewById(R.id.height);
31             EditText fieldweight = (EditText)findViewById(R.id.weight);
32             double height = Double.parseDouble(fieldheight.getText().toString())/100;
33             double weight = Double.parseDouble(fieldweight.getText().toString());
34             double BMI = weight / (height * height);
35
36             TextView result = (TextView)findViewById(R.id.result);
37             result.setText("Your BMI is "+nf.format(BMI));
38
39             //Give health advice
40             TextView fieldsuggest = (TextView)findViewById(R.id.suggest);
41             if(BMI>25){
42                 fieldsuggest.setText(R.string.advice_heavy);
```

```
43                }else if(BMI<20){
44                    fieldsuggest.setText(R.string.advice_light);
45                }else{
46                    fieldsuggest.setText(R.string.advice_average);
47                }
48            }
49        };
50 }
```

我們會學到：導入其他用到的模組、如何取得介面元件、如何對按鈕設定動作。

從上面的完整程式中，我們看到上面介紹到的程式主體都還在，不過也增加了一些內容。這些內容即我們的主要程式邏輯。

講解

```
//Listen for button clicks
```

兩個反斜線是 java 語言所支援的另一種註解方式。

按鈕

```
Button button = (Button)findViewById(R.id.submit);
```

宣告一個 button 實體，透過 findViewById 方法，從資源檔中取得對應的介面元件(按鈕)。這邊取出的是「R.id.submit」按鈕元件。

「R.id.submit」對應到 XML 描述檔的資源是

```
<Button id="@+id/submit"
/>
```

為了確保宣告的型別跟 XML 描述檔中描述的介面元件型別一致，好使程式運作正常，我們在「findViewById」 方法前加上 「(Button)」修飾，強制將取得的資源型別設成 「button」型別。

```
button.setOnClickListener(calcBMI);
```

這句包含了「Button」類別中的「setOnClickListener」方法。方法中傳入了一個「calcBMI」函式。Android 系統的 UI 互動方式採用常見的事件驅動方式。也就是當使用者按下「button」按鈕的時候，Android 系統會去觸發按鈕的「setOnClickListener」方法中所指定的「calcBMI」函式。

觸發事件函式

```
    private OnClickListener calcBMI = new OnClickListener() {
        public void onClick(View v) {
            ...
        }
    };
```

範例中所有的邏輯與運算內容，都放置在這個事件函式中。

當使用者按下按鈕時，即觸發「OnClickListener」類型的事件函式。也有人將這種函式稱為回呼（Call Back）函式，因為是當有動作做完（使用者按下了按鈕），才會呼叫（算出 BMI 值）。

onClick(View v) 是 OnClickListener 中實作的方法，代表按下按鈕時會執行的事件方法。

編輯欄位

```
EditText fieldheight = (EditText)findViewById(R.id.height);
EditText fieldweight = (EditText)findViewById(R.id.weight);
```

與上面 button 的宣告類似，只是改成宣告 EditText 實體，透過 findViewById 方法，從資源檔中取得對應的文字欄位元件。這邊取出的是「R.id.*height*」和「R.id.*weight*」文字欄位元件。

運算

```
double height = Double.parseDouble(fieldheight.getText().toString())/100;
double weight = Double.parseDouble(fieldweight.getText().toString());
double BMI = weight / (height * height);
```


BMI 值的算法是「體重除以身高的平方」。 用計算式來表示，就是

```
體重(weight) / 身高(height)*身高(height)
```

這麼看來，上面的程式碼就很清晰了。我們先從身高欄位(fieldheight)、體重欄位(fieldweight)中取得使用者輸入的身高體重數字，再定義一個雙倍精度幅點數(double) 型態的變數 BMI 來儲存運算的結果，因此，BMI 變數中儲存了運算出來的實際 BMI 值。

顯示結果

我們把 BMI 值運算出來了，接著要將結果顯示回螢幕上。

```
TextView result = (TextView)findViewById(R.id.result);
```

為了將結果顯示到螢幕上，在之前 XML 定義檔中我們已經預留了一個名為「result」的 TextView 欄位。在程式碼中，我們再宣告一個 TextView 實體，透過 findViewById 方法，從資源檔中取得對應的介面元件(文字顯示)。這邊取出的就是「R.id.*result*」介面元件。

```
DecimalFormat nf = new DecimalFormat("0.00");
result.setText("Your BMI is "+nf.format(BMI));
```

透過 java 內建的 DecimalFormat 函式，我們可以將運算結果，以適當的格式顯示。

透過「setText」方法，我們可以將指定的字串，顯示在螢幕上文字類型的介面元件中。

顯示建議

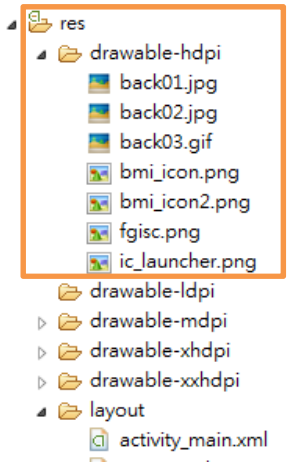
「顯示建議」的方式與「顯示結果」完全相同，只有多加了幾個「if」判斷式：當 BMI 值大於 25 顯示過重，小於 20 顯示過輕。程式碼留給讀者自習。

```
//Give health advice
TextView fieldsuggest = (TextView)findViewById(R.id.suggest);
if(BMI>25){
    fieldsuggest.setText(R.string.advice_heavy);
}else if(BMI<20){
    fieldsuggest.setText(R.string.advice_light);
}else{
    fieldsuggest.setText(R.string.advice_average);
}
```

完整的 BMI 程式動作流程，就是使用者在身高體重欄位中輸入好身高體重後，按下「計算 BMI 值」按鈕，程式根據識別符號，從對應的身高體重欄位讀取輸入的值，並做計算，最後將計算的結果與建議顯示到螢幕上。

Unit 7 更改 App 圖示、更改背景圖

1. 上網下載或利用繪圖工具自行繪製圖示(icon)、背景圖(background)。大小依虛擬裝置(AVD)的螢幕尺寸而定。儲存於 <res/drawable/hdpi> 資料夾中。



2. 修改圖示(icon)設定

   [1] 開啟 Android 應用程式清單檔 <AndroidManifest.xml>

   [2] 在<application 中加入 android:icon 屬性設定
   android:icon="@drawable/bmi_icon"

   其中 bmi_icon 為 icon 圖檔名稱

```
<application
        android:allowBackup="true"
        android:icon="@drawable/bmi_icon"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.bmi.MainActivity"
            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>
```

3. 修改背景圖設定

   [1] 開啟欲修改的 layout 檔，例如： <layout / activity_main.xml>

   [2] 在以下程式碼加入一個背景屬性
   android:background="@drawable/back01"
   其中，*back01* 是背景圖檔的名稱，不需加.png 或.jpg

```
<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"
    android:background="@drawable/back01"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
```

練習：Anny's 教學資源網中下載 image.rar, 解壓縮，並 copy 至 workspace/bmi/res/drawable/hdpi 。設定 icon，並修改 Activity 的 background images

Unit 8 發佈應用程式到實體裝置

- **Android Market 的運作方式**

「Android Market」的運作方式如下：

1. 開發者可以將自己寫好的軟體上傳到 Android Market 中。
2. 開發者透過 Android Market 販賣軟體的 30% 收入，得分給電信商跟電子收費商(如手機月費帳單或 Google Checkout 等)，所以開發者可以拿到應用程式定價的 70%。
3. 註冊為「Android Market Developer」要收美金 25 元的「入場費」。推測可能是種為了保證「Android Market」上應用程式的質量，也為了促使開發者寫一點收費軟體，好讓電信商有得分成的策略。

- **註冊 Android Market**

前往 http://www.android.com/market/ ，畫面右上角有一段「Interested in having your application in Android Market?」敘述，按下其下方的「learn more」按鈕，即可開始註冊成為「Android」開發者。

開發者用的網址是 http://market.android.com/publish

開發者可以透過「Android Market」發佈「Android」應用程式。首先，開發者得註冊一個 Google 帳號。然後使用(Google Checkout)以信用卡付出 $25 美元的註冊費用。 最後得同意「Android Market」的使用授權協議。

註冊一個 Google 帳號不難，相信大部分讀者都已經擁有一個 Google 帳號。

在申請「Android Market」時要填入加上國碼的手機號碼。台灣加上國碼的手機號碼為「+8869xxxxxxxx」。「886」是國碼，加上一個「0」之後，「09xxxxxxxx」是你的手機號碼。「+」則是「加上國碼的手機號碼」表示方式。 接著按下「Google Checkout」圖示，如果沒有「Google Checkout」的話，也需作先設定。一切完成後在「Google Checkout」中勾選「I agree and I am willing to associate my credit card and account registration above with the Android Market Developer Distribution Agreement.」。畫面會出現「等待信用卡認證的訊息」，並有 「Google Checkout」的確認函寄到我們設定的電子信箱中。接著想要繼續登錄開發者網頁時，會發現這個網頁似乎壞掉了。其實是等待信用卡認證完成，需要一點時間（一兩個鐘頭），等認證好，完成付款程序後，網頁就能再次開啟。

開啟後會出現「Your Registration to the Android Market is approved! You can now upload and publish software to the Android Market.」(已經註冊完成)訊息。以後點擊「Android Market」網頁右上角的按鈕時，就會進入開發者面板（Developer Console）頁面。

在開發者面板畫面的左上角是開發者的暱稱。暱稱旁邊可以選擇「Edit profile » 」（編輯個人資料）來編輯之前填入的「Android Market Developer」資訊。

- **上傳應用程式到 Android Market**

選擇右下角的「Upload Application」（上傳應用程式）按鈕，出現應用程式上傳畫面。各個欄位的作用都寫的很明白，也可以為應用程式自行定價。

「Android Market」上所有的程式可分為「應用程式」與「遊戲」兩大類。選擇好大分類後，其下會出現各自可選的子分類。 在「Upload assets」區塊中，點選「Application .apk file」旁的「瀏覽...」按鈕，就可以**上傳已經簽署好金鑰的「.apk」**程式。

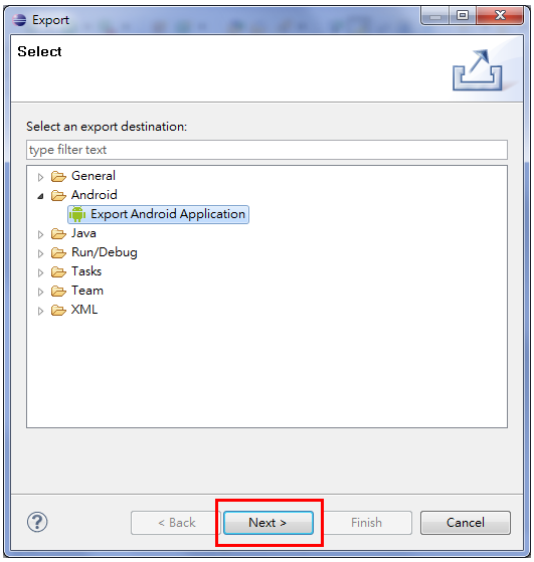
- **APK 檔案 (Android Application Package)**

所謂的 APK 檔（Android application package）是一個包含程式本身以及相關的資源檔所形成的檔案。

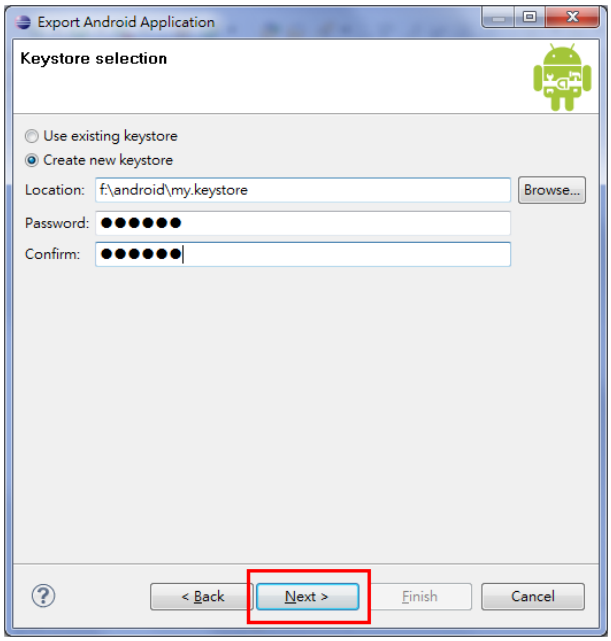
程式開發完了，也在手機模擬器跑過沒問題了，可是我能不能把開發好的程式安裝到真實的手機上呢？想把開發好的程式安裝到手機上分成兩個主要的步驟， 一個是將成是打包成 APK(Android application package)，然後利用 USB 或者上傳到 Android Market 來安裝。

- **產生 APK 檔的步驟：**

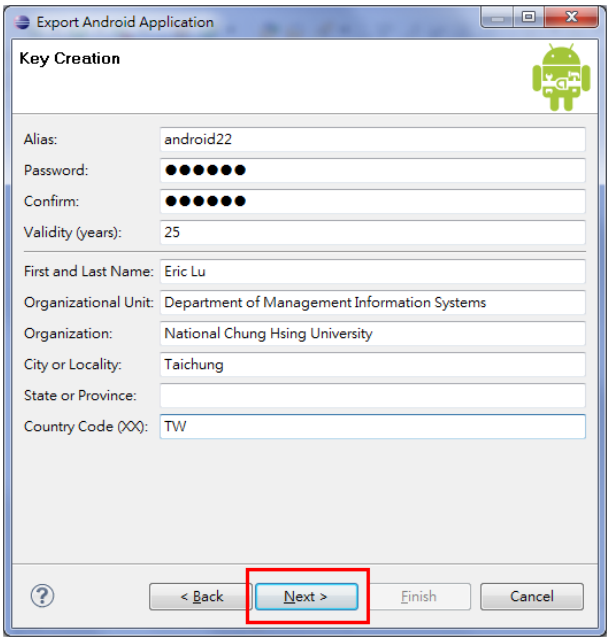
1. 開啟程式的專案，點選 File --> Export，然後如下畫面選擇 Export Android Application 之後，按 "Next>" 按鈕。



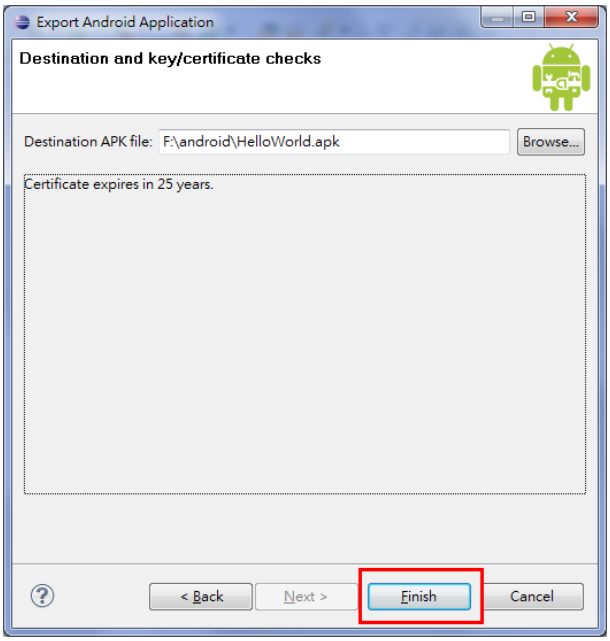
2. 為了確保降低資訊安全上的顧慮(例如，內含病毒碼)，Google 要求所有的 apk 檔都必須經過電子簽章簽署之後，才能被使用。 雖然有了電子簽章的機制可以比較容易保證不會有惡意的第三者(有興趣的讀者,請自行參考資訊安全相關的文件)，但是並不能保證 apk 檔案內不包含病毒碼；實際上，過去已經發生過惡意的程式在 Android Market 上出現，但是這些程式也被 Google 移除下架。 由於要產生必要的電子簽章，所以如果你在 Eclipse 下是第一次產生 APK 檔，那麼你必須如以下畫面般產生新的 "key store"； 第二次以後，只需要使用現有的 key store 即可。



3. 產生 key store 的步驟如下：請在如下的畫面中，輸入任意但是好記的 alias 名稱；一個 key store 中可以有多個 alias，而一個 alias 大概可以想成是一個電子簽章，因此除了密碼的部分是在作必要的保護之外，其它就填入代表你或者你的單位的相關資料。



4. 如下圖所示，請選擇產生的 APK 檔所放置的地點後，按下 "Finish" 按鈕之後產生 apk 檔。

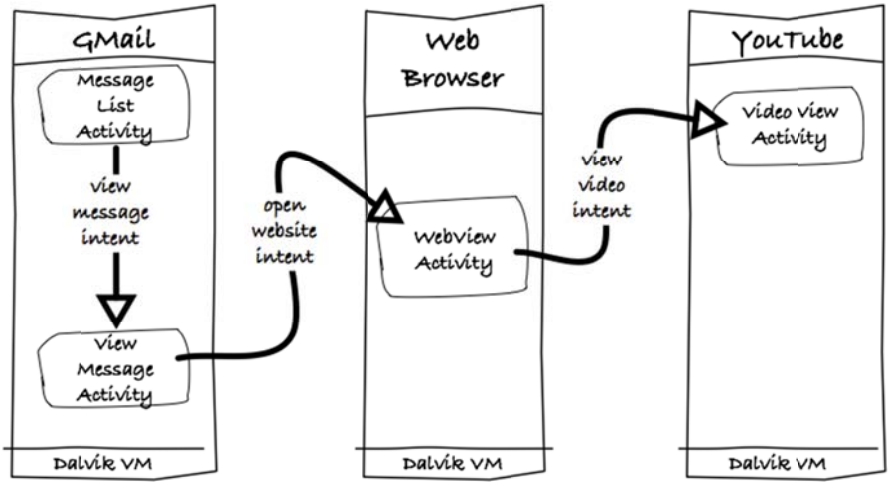


5. 有了 apk 檔，就可以寄到信箱，或放在雲端硬碟分享連結網址，即可安裝在 Android 行動裝置中。

- **解除安裝程式**

一般來說，官方提供的應用程式是無法解除安裝的。可以解除安裝的程式，都是屬於使用者自行安裝的；解除安裝的方式有下列兩種：

1. 在手機上，依序執行"設定" --> 應用程式--> 管理應用程式，然後選取希望解除安裝的程式進行解除安裝。
2. 如果是從 Market 下載的，也可以從 Market 中的"我的應用程式"中解除安裝。



圖片來源：https://thenewcircle.com/s/post/1178/architecting_android_apps

Android Activity

直觀來看，每個 Activity 通常會負責處理一個螢幕的內容 (包含介面、選單、彈出對話框、程式動作等)。當我們需要從一個螢幕畫面切換到另一個螢幕畫面的時候，就涉及到了 Activity 切換的動作。 我們可以將 Activity 看成 MVC 模式中的 Control。Activity 負責管理 UI，並接受事件觸發。

以是否需要與其他 Activity 交換資料來區分，Activity 可以粗分為兩種類型：「獨立的 Activity」與「相依的 Activity」。不同類型的 Activity，其動作也不盡相同：

- **獨立的 Activity**

獨立的 Activity 是不需要從其他地方取得資料的 Activity。只是單純的從一個螢幕跳到下個螢幕，不涉及資料的交換。 從一個獨立的 Activity 呼叫另一個獨立的 Activity 時，我們只要填好 Intent 的內容和動作，使用 startActivity 函式呼叫，即可喚起獨立的 Activity。

- **相依的 Activity**

相依的 Activity 是需要與其他 Activity 交換資料的一種 Activity。相依的 Activity 又可再分為單向與雙向。從一個螢幕跳到下個螢幕時，攜帶資料供下一個螢幕（Activity）使用，就是單向相依的 Activity; 要在兩個螢幕之間切換，螢幕上的資料會因另一個螢幕的操作而改變的，就是雙向相依的 Activity。 與獨立的 Activity 比起來，相依的 Activity 變化更加精采。

獨立的 Activity

要完成獨立的 Activity 的動作，我們要做幾件事：

1. 在程式碼 (src)中建立新 Activity 類別檔案 (.java)
2. Layout 中建立新的頁面配置檔(.xml)
3. AndroidManifest 清單中新增 Activity 描述
4. 在原 Activity 類別中加入 startActivity 函式

Step1: 建立新的 Activity 類別檔案

1. 首先，使用 Navigator 檔案瀏覽視窗，切換到「src/com/demo/android/bmi」資料夾。在「bmi」資料夾中，現存有 「MainActivity.java」與「R.java」兩個檔案。我們準備在此建立一個新的 Activity 類別檔案。
2. 在「bmi」資料夾圖示上按右鍵，選擇「New->Class」選項。 Eclipse 會跳出一個「New Java Class」對話框。
3. 在對話框中的「Name」一欄上填入「ReportActivity」。「Report」的**字頭需大寫**，這是 Java 程式語言的默認規則。
4. 在「Superclass」一欄右方，按下「Browse...」，Eclipse 會跳出「Superclass Selection」對話框。在對話框中的「Choose a type」欄位中輸入「activity」，輸入框下方的「Matching items」欄位中，會顯示出所有可能的類別。我們選擇「Activity - android.app - ...」這個選項，點擊右下方的「ok」按鈕，回到上一個對話框。
5. 此時，「Superclass」欄位中將填入「android.app.Activity」訊息。按下對話框右下角的「Finish」鍵，Eclipse 會在「bmi」資料夾中，產生一個對應的「ReportActivity.java」檔案。

剛產生（尚未修改過的）的「ReportActivity.java」檔案如下：

```
1  package com.demo.android.bmi;
2
3  import android.app.Activity;
4
5  public class ReportActivity extends Activity {
6
7  }
```

Android 程式碼的基本架構，即 XML 描述檔與程式碼兩個主要組成部分。稍後我們要處理建立新程式碼的相關工作，包含定義對應的 XML 描述檔，與在程式碼中，填入這個 class 的內涵。

Step2: Layout 中建立新的頁面配置檔(.xml)

1. 在「res/layout」中新增一個「activity_report.xml」檔案：按右鍵選「new/Android XML File」選項，開啟「Android XML 檔案產生精靈」新增一個 XML 版面配置檔。
2. **Resource Type:**「Layout」、**File:** 填入「activity_report」、**Root Element:**「RelativeLayout」，按「Finish」。如此，即產生「activity_report.xml」檔。
3. 把 xml 檔案複製一份過來：

```
1   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2           xmlns:tools="http://schemas.android.com/tools"
3   android:layout_width="match_parent"
4   android:layout_height="match_parent"
5   android:paddingBottom="@dimen/activity_vertical_margin"
6   android:paddingLeft="@dimen/activity_horizontal_margin"
7   android:paddingRight="@dimen/activity_horizontal_margin"
8   android:paddingTop="@dimen/activity_vertical_margin"
9   tools:context=".MainActivity" >
10
11  <TextView
12          android:layout_width="wrap_content"
13          android:layout_height="wrap_content"
14          android:text="@string/hello_world" />
15
16  </RelativeLayout>
```

4. 打開「src/com/demo/android/bmi/ReportActivity.java」，把預設的程式碼複製進來：

```
1  package com.demo.android.bmi;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5
6  public class ReportActivity extends Activity {
7      /** Called when the activity is first created. */
8      @Override
9      public void onCreate(Bundle savedInstanceState) {
10             super.onCreate(savedInstanceState);
11             setContentView(R.layout. activity_main);
12     }
13 }
```


上面的程式碼中，我們將

```
setContentView(R.layout. activity_main);
```

換成了：

```
setContentView(R.layout. activity_report);
```

以對應我們新定義的 XML 描述檔產生的資源識別符號。

Step 3: AndroidManifest 清單中新增 Activity 描述

5. 打開「AndroidManifest.xml」檔案，定義「AndroidManifest.xml」清單檔案可以在原本的「AndroidManifest.xml」檔案中直接修改原始碼，加入如下敘述：

```
<activity android:name="ReportActivity"></activity>
```

修改 Report 頁面標題文字

如果希望在打開 Report Activity 頁面時，標題欄上的文字將會是「BMI 報告」，而不是預設跟 Activity 名稱相同的「Report」，我們可以在清單裡的「report activity」標籤中，加入標籤（label）屬性的描述。步驟如下。

6. 在「res/values/」目錄下，新建一個 report.xml 描述檔，存放 Report 活動頁面用到的字串。

檔案內容如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="report_title">BMI 報告</string>
</resources>
```

7. 有了描述檔後，我們可以再修改「AndroidManifest.xml」清單，為 ReportActivity 添加標籤(label)屬性：

```
<activity android:name="ReportActivity" android:label="@string/report_title"></activity>
```

Step4: 原 Activity 類別中加入 startActivity 函式

在準備好相關資源、並在「AndroidManifest.xml」清單添加新 activity 描述後，來到實際控制整個程式流程的程式碼部份。

在程式碼中，要把原本按下按鈕即開始計算，並顯示 Bmi 值到同一螢幕的按鈕動作，改成按下按鈕後跳轉到 「Report」 這個 Activity。

首先修改「MainActivity」這個 Activity。打開「src/com/demo/android/bmi/ MainActivity.java」檔案，修改「OnClickListener」函式(定義按下按鈕時做的動作)的內容。按下按鈕後從 MainActivity 切換（跳轉）到 ReportActivity 的程式片段如下：


```
1   private Button.OnClickListener calcBMI = new Button.OnClickListener()
2   {
3       public void onClick(View v)
4       {
5           //Switch to report page
6           Intent intent = new Intent();
7           intent.setClass(MainActivity.this, ReportActivity.class);
8           startActivity(intent);
9        }
10 };
```

修改好後，開啟 Android 模擬器。當我們按下「計算 BMI 值」按鈕後，螢幕會切換到「Hello World, Bmi」頁面(已經進入了 ReportActivity)。在這個頁面上，我們並沒有辦法可以直接回到前一個頁面，因為這兩個頁面都是獨立的 Activity。但我們還是可以透過按下硬體的「Undo」鍵，使螢幕切換回原本輸入身高體重的頁面 (即回到 Main Activity)。

講解

```
Intent intent = new Intent();
```

我們建立一個新的「意圖」(Intent) 實體。

```
intent.setClass(MainActivity.this, ReportActivity.class);
```

為這個意圖指定來源的 Activity 所在 class(MainActivity.class)，與要前往的 Activity 所在的 class(ReportActivity.class)。

```
startActivity(intent);
```

將定義好的 intent 傳入「startActivity」函式中。「startActivity」函式會將 intent 傳入 Android 框架，Android 框架會根據各應用程式在系統中註冊的資料(有沒有聯想到我們剛剛為 ReportActivity 增加的 activity 清單描述?)，找出 ReportActivity，並呼叫它。

Unit 10 傳送資料到新的活動 -- Intent

Intent 是一個動作與內容的集合。Intent 像是一串網址，傳送到系統並意圖靠其他 Activity 來處理網址中所指定的動作跟內容。

Android 使用 Intent 來完成在螢幕間切換的動作（獨立的 Activity）。Intent 包含 Activity 間切換所需的**動作、分類、傳送資料**等訊息，就像是 Activity 之間的宅急便一樣。

要完成在 Activity 之間透過 Intent 傳送資訊的動作，可以分成「傳遞資訊」與「接收資訊」兩部分。

使用 Intent 傳遞資訊

上一單元新增了一個 Report Activity 頁面，但是還沒有為新頁面填入實值內容。本單元會完成將 BMI 應用程式從一個頁面改寫成為兩個頁面：「輸入頁面」(原本的 MainActivity)，與「結果頁面」(ReportActivity)的應用程式。「輸入頁面」從介面上取得身高、體重值，透過傳送 Intent，將值攜帶到「結果頁面」。「結果頁面」從 Intent 中取出其攜帶的身高、體重值，用這兩個參數來產生 BMI 報告結果。

打開「src/com/demo/android/bmi/MainActivity.java」，修改「Button.OnClickListener」函式：

```
1   private Button.OnClickListener calcBMI = new Button.OnClickListener()
2   {
3         public void onClick(View v)
4         {
5           //Switch to report page
6            Intent intent = new Intent();
7            intent.setClass(MainActivity.this, ReportActivity.class);
8            Bundle bundle = new Bundle();
9           bundle.putString("KEY_HEIGHT", field_height.getText().toString());
10          bundle.putString("KEY_WEIGHT", field_weight.getText().toString());
11          intent.putExtras(bundle);
12          startActivity(intent);
13    }
14  };
```

講解

```
Intent intent = new Intent();
intent.setClass(MainActivity.this, ReportActivity.class);
...
startActivity(intent);
```

這段程式碼的作用是透過 Intent 通知系統(Android 框架)：我們將要從 MainActivity 這個頁面（輸入頁面）前往 Report Activity 頁面（結果頁面）。如果把我們多加的用來附加資料的程式碼拿掉，這段程式碼即原來獨立的 Activity 的程式碼。

```
Bundle bundle = new Bundle();
...
intent.putExtras(bundle);
```

相依的 Activity 與獨立的 Activity 不同之處，就在於相依的 Activity 會附帶傳送額外資訊到新的 Activity。這些額外資訊都是靠著 Intent 物件來攜帶的。

傳送 intent 時，我們可以在其上附加一些訊息，比如說本例中我們從輸入介面中取出了的身高、體重值，要將身高、體重值傳送給 ReportActivity 後作計算。這些附加在 Intent 上的訊息都儲存在 Bundle 物件中。透過「intent.putExtras(bundle)」敘述，我們將「bundle」物件附加在 Intent 上，隨著 Intent 送出而送出。

```
bundle.putString("KEY_HEIGHT", field_height.getText().toString());
bundle.putString("KEY_WEIGHT", field_weight.getText().toString());
```

這段程式是實際用來附加資料的程式碼。將使用者輸入的身高、體重值，儲存到 bundle 物件中。Bundle 其實是一種特別定義的映射（map）型別。「KEY_HEIGHT」、「KEY_WEIGHT」是為儲存在 bundle 物件中的身高、體重值所指定的「**識別符號**」。在這邊，直接把身高、體重值都儲存成字串。因為整個程式都是我們控制，到時候在接收的 Activity 一端，再透過「KEY_HEIGHT」、「KEY_WEIGHT」這兩個「識別符號」來取得實際的身高、體重值。讀出的值也是字串，等值讀出來以後，再去做型別轉換就好了。當然你也可以直接把身高、體重值存成數字。

Bundle 型別額外提供了很多 API。在傳送 Intent 時，使用 Bundle 型別的物件來攜帶資料，相當方便。

使用 Intent 接收資訊

在使用 Intent 接收資訊前，我們先來加上「Report」這個 Activity 的介面。

相關工作

打開「res/values/report.xml」檔案，修改如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="report_title">BMI 報告</string>
    <string name="report_back">前一頁</string>
</resources>
```

打開「res/layout/activity_report.xml」檔案，修改如下：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        >
         <TextView android:id="@+id/result"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text=""
            />
         <TextView android:id="@+id/suggest"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text=""
            />
        <Button android:id="@+id/report_back"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/report_back"
            />
</LinearLayout>
```

這時打開 Eclipse 開發環境的 layout 檢視，或是執行模擬器，我們可以看到一個「前一頁」按鈕，按鈕前其實還有兩個沒有內容的 TextView 介面元件，下一節中我們將在「ReportActivity」中取得從「MainActivity」 傳過來的身高體重資料，根據這些資料產生報告資訊。

在 Activity 中解開資訊

用作接收 Intent，透過 Intent 攜帶的資訊來計算出 BMI 值的「src/com/demo/android/bmi/ReportActivity.java」完整程式碼如下：

```
1  package com.demo.android.bmi;
2
3  import java.text.DecimalFormat;
4  import android.app.Activity;
5  import android.os.Bundle;
6  import android.view.View;
7  import android.widget.Button;
8  import android.widget.TextView;
9
10 public class ReportActivity extends Activity {
11         /** Called when the activity is first created. */
12 @Override
13         public void onCreate(Bundle savedInstanceState) {
14                 super.onCreate(savedInstanceState);
15                 setContentView(R.layout.report);
16                 findViews();
17                 showResults();
18                 setListensers();
19 }
20
21 private Button button_back;
22 private TextView view_result;
23 private TextView view_suggest;
24
25 private void findViews()
26 {
27         button_back = (Button) findViewById(R.id.report_back);
28         view_result = (TextView) findViewById(R.id.result);
29         view_suggest = (TextView) findViewById(R.id.suggest);
30 }
31
32 //Listen for button clicks
33 private void setListensers() {
34         button_back.setOnClickListener(backMain);
35 }
36
37 private Button.OnClickListener backMain = new Button.OnClickListener()
38 {
39       public void onClick(View v)
40       {
41          // Close this Activity
42          Report.this.finish();
43       }
44 };
45
46 private void showResults() {
47       DecimalFormat nf = new DecimalFormat("0.00");
48
49       Bundle bundle = this.getIntent().getExtras();
50        double height = Double.parseDouble(bundle.getString("KEY_HEIGHT"))/100;
51        double weight = Double.parseDouble(bundle.getString("KEY_WEIGHT"));
52       double BMI = weight / (height * height);
53       view_result.setText(getString(R.string.bmi_result) +nf.format(BMI));
54
55       //Give health advice
56       if(BMI>25){
57           view_suggest.setText(R.string.advice_heavy);
58       }else if(BMI<20){
59           view_suggest.setText(R.string.advice_light);
```

```
60      }else{
61          view_suggest.setText(R.string.advice_average);
62      }
63
64    }
65 }
```

講解

整個程式的架構我們在「完成 BMI 程式」與「重構程式」兩章中已經詳細說明過了。

```
Bundle bundle = this.getIntent().getExtras();
```


當我們透過 Intent 傳到新的 Activity 後，只要使用 Activity.getIntent() 函數，就可以得到傳來的 Intent 物件。然後使用「getExtras」函式，就能取得附加在 Intent 上的 bundle 物件。

```
                double height = Double.parseDouble(bundle.getString("KEY_HEIGHT"))/100;
                double weight = Double.parseDouble(bundle.getString("KEY_WEIGHT"));
```

在當前的 Activity 取得了 bundle 物件後，我們可以透過指定儲存在 bundle 物件中的身高、體重值的識別符號「KEY_HEIGHT」、「KEY_WEIGHT」來取出身高、體重值的資料。由於我們傳參數值來時，所使用的是是字串格式，所以我們在此得做個型別轉換，將參數從字串轉換成雙倍精度浮點數(Double)型別。

```
    private Button.OnClickListener backMain = new Button.OnClickListener()
    {
        public void onClick(View v)
        {
           // Close this Activity
           Report.this.finish();
        }
    };
```

當按下 backMain 按鈕元件後，結束 ReportActivity，顯示出原本的 MainActivity。