

APPENDI X C

{ App Inventor 小秘訣 }

C-1 使用 ActivityStarter 元件	548
C-2 系統記錄	558
C-3 顯示清單列表	561
C-4 設定元件尺寸	565
C-5 存取圖像和聲音	568



C-1 使用 ActivityStarter 元件

ActivityStarter 元件讓您可透過現有的應用程式去呼叫其他的應用程式。ActivityStarter 適合進階的玩家使用，但它也是您的 AppInventor 應用程式更上一層樓的關鍵，因為它能讓您有效利用別人寫好的程式或是開發架構。

您所能呼叫的應用程式，可以是您或是別人所編寫的 App Inventor 程式，或是像 Camera 以及 Google Map 等預先安裝在手機上的程式。甚至只要您提供足夠的資訊，您就可以自由地呼叫任何一個應用程式。此外，您也能在呼叫一個應用程式之後再從接收它的執行結果，這對於我們所要進行的後續動作相當方便。請注意截至目前為止，回傳結果皆為字串格式。

為了能呼叫另一個應用程式，您必須提供一些控制資料給 Android 作業系統。我們需在使用 StartActivity 方法之前，必須先設定好 ActivityStarter 元件中相關的屬性才行。以下提供了一些範例：

C-1-1 呼叫其他的 App Inventor

若知道某個 App Inventor 程式的 package 名稱和 class 名稱時，您就能利用這些資訊來呼叫它。如果您有這個程式的原始檔，您可依照下列方式找到這些資料：

1. 下載原始檔 (.zip) 到您的電腦。
2. 把這個檔案解壓縮，找到 youngandroidproject/project.properties 這個檔案。
3. 第一行的開頭是「main=」，其後就是 package 和 class 的名稱。

例如，<EX1-1>HelloPurr 這個程式，解壓縮之後在 `youngandroidproject/project.properties` 檔案中找到的第一行會是：

```
main=com.gmail.Simpson.Homer.HelloPurr.Screen1
```

`gmail.Simpson.Homer` 這是來自這個程式設計者的 gmail 帳號，例如 `Homer.Simpson@gmail.com`。

要呼叫這個應用程式，您需要先設定好 `ActivityStarter` 元件的這兩個屬性：

```
ActivityPackage: com.gmail.Simpson.Homer.HelloPurr
```

```
ActivityClass: com.gmail.Simpson.Homer.HelloPurr.Screen1
```

接著使用 `ActivityStarter` 元件的 `StartActivity` 方法就能順利呼叫 HelloPurr 小貓程式。當 HelloPurr 結束時（點選手機上的 Menu 鍵 → Stop Application），這時就會呼叫 `AfterActivity` 方法。

請注意：

若您要呼叫另一個 App Inventor 應用程式，請先確認您使用了正確的 `package` 名稱。例如：假設您重新包裝了一個別人寫的應用程式，這時 `package` 名稱就改變了，不再是原來的 `package` 名稱。為了避免混淆，程式啟動之後您可使用 Android 所提供的 `adb logcat` 來檢視 Android 的系統記錄，這樣就知道到底發生些什麼事情了，詳細請參考 [C-1-3 如何設定 `ActivityStarter` 相關屬性]。



C-1-2 呼叫手機內建程式

我們能透過 `package` 名稱 或 `class` 名稱來呼叫特定的手機應用程式。這些應用程式也能對 `intent` 做出回應，這是 Android 系統用來在不呼叫特定應用程式就能執行某些動作的方法，您可以在 Android 開發者網站找到更多有關於 `intent` 的資訊。

有些應用程式需要額外的資訊才能開啟，舉例來說，Google Map 需要地理資訊才能決定要顯示那張地圖；搜尋網頁也同樣需要使用者提供想要搜尋的文字。您必須參閱相關文件才能知道這些資訊的格式以及如何設定它們。通常，您需要在啟動其他應用程式之前，先在 `ActivityStarter` 元件中將相關屬性設定完成。下面有一些例子：

◆ 呼叫相機

要呼叫 Android 裝置上的照相機，需設定以下欄位：

- `Action`：`android.intent.action.MAIN`
- `ActivityPackage`：`com.google.android.camera`
- `ActivityClass`：`com.android.camera.Camera`

我們也可以直接使用 App Invevntor 的 Camera 元件，這裡只是說明如何利用 `ActivityStarter` 來呼叫它。

◆ 網路搜尋

為了能在網頁上搜尋特定資料，例如「App Inventor」，需設定以下欄位：

- `Action`：`android.intent.action.WEB_SEARCH`
- `ExtraKey`：`query`
- `ExtraValue`：App Inventor
- `ActivityPackage`：`com.google.android.providers.enhancedgooglesearch`

- ActivityClass : com.google.android.providers.enhancedgooglesearch.Launcher

◆開啟網頁

要透過 ActivityStarter 元件來開啟指定網頁，需設定以下欄位：

- Action : android.intent.action.VIEW
- DataUri : http://www.cavedu.com (可自由更改)

◆呼叫信件軟體並指定內容

要呼叫 Android 信件程式需使用 android.intent.action.VIEW。您可以使用 ActivityStarter 的 DataUri 屬性來指定收件人、訊息和內文。設定完成之後 ActivityStarter 就會啟動 Android 裝置上的信件軟體，寫好信就可以寄出囉！

例如指定：

- Action : android.intent.action.VIEW
- DataUri : mailto:appinvevntor@cavedu.com

呼叫信件軟體之後將會看到收件者欄位已指定為 appinvevntor@cavedu.com。

或者指定：

- Action : android.intent.action.VIEW
- DataUri : mailto:santa@northpole.com?subject=Please Santa&Bring me a pony

呼叫信件軟體之後將會看到收件者 (santa@northpole.com)、信件標題 (Please Santa) 及內文 (Bring me a pony) 皆已設定完成。當然您可以繼續在信件軟體中來修改這些內容。



◆在地圖上顯示指定位置

若您知道某個位置的經度與緯度座標，就能利用 `ActivityStarter` 將這些資訊顯示至地圖上：

- Action：`android.intent.action.VIEW`
- `DataUri`：`geo:37.8,-122.23?z=23`

請特別注意這裡的 `DataURI` 格式。這裡指定了縮放值 `z` 為 23，這是最大的縮放比。縮放值 `z` 的範圍是從 1（最小，整顆地球）到 23（最大，一棟房子），請注意 `z` 值可以省略，代表使用預設的縮放比例。

若您知道某處的郵遞區號，您可以這樣設定：

- Action：`android.intent.action.VIEW`
- `DataUri`：`geo:0,0&q=94043`

若您知道某處的地址，您可以根據 `URL encoding` 格式（`http://www.w3schools.com/TAGS/ref_urlencode.asp`）來設定：

- Action：`android.intent.action.VIEW`
- `DataUri`：`geo:0,0&q=5000%20MacArthurBlvd%20Oakland%2CCA`

一般而言，您只需要將空格以「%20」、逗號以「%2C」、句點「%2E」來取代即可，以本範例來說，地址是「5000 MacArthurBlvd Oakland,CA」。

◆播放 YouTube 影片

找到您想要播放的 YouTube 影片連結之後，再設定以下欄位：

- Action：android.intent.action.VIEW
- ActivityPackage：com.google.android.youtube
- ActivityClass：com.google.android.youtube.PlayerActivity

以及想要播放的影片連結，例如：

- DataUri：http://www.youtube.com/watch?v=MnueRKonmBE

C-1-2 如何設定 ActivityStarter 相關屬性

如果找不到想要呼叫的程式的相關資料，您還是藉由 Android 的系統記錄來手動啟動這個程式。例如，若您使用 YouTube 看影片，將會在系統記錄中看到類似以下的資訊：

```
I/ActivityManager( 86): Starting activity: Intent { act=android.intent.action.VIEW cat=[android.intent.category.BROWSABLE] dat=http://www.youtube.com/watch?v=8ADwPLSFeY8flg=0x3800000 cmp=com.google.android.youtube/.PlayerActivity }
```

請找到「cmp=」後的字串，斜線前面的部分是 ActivityPackage，以本範例來說就是「com.google.android.youtube」而 ActivityClass 為整個「cmp=」，但需要去掉斜線，就是「com.google.android.youtube.PlayerActivity」。另外可透過「dat=」來指定 DataUri，以本範例來說就是「http://www.youtube.com/watch?v=8ADwPLSFeY8flg=0x3800000」



C-1-4 呼叫任意程式

若您知道 package 名稱、class 名稱或者對應的 intent，您就能透過 ActivityStarter 去呼叫任何應用程式。至於如何使用 intent 來呼叫其他應用程式，請參閱 Android API documentation (<http://developer.android.com/reference/android/content/Intent.html>) 和 OpenIntent's list of Intents (<http://www.openintents.org/en/node/35>)

假若您安裝了某個程式但沒有原始檔，您還是有機會透過 package 名稱和 class 名稱 (有時是 intent) 來呼叫它，請一樣如上述從 Android 系統記錄中來檢視相關資訊。

C-1-5 從所呼叫的應用程式取得資料

有些應用程式可以回傳數值結果，取決於它們的原始設計目的。目前 App Inventor 應用程式只能回傳文字資料。

◆從 App Inventor 應用程式取回結果

您可以建立一個可回傳文字的 App Inventor 程式，因此它們可以當作副程式來使用。為了回傳結果，請使用 Control 類別下的 close screen with result 指令，這時該副程式將會中止，並且將 close screen with result 這個欄位傳送給呼叫這個副程式的 AfterActivity 元件，並由該 AfterActivity 元件的 AfterActivity event 事件來接收參數



具體而言，假設有一個當作副程式來使用的 Subroutine App，它可以在被 ActivityStarter 元件呼叫後回傳一個值；以及一個呼叫 SubroutineApp 的 CallerApp。為了回傳結果，SubroutineApp 會執行 close screen with result 指令，將該指令的 result 參數回傳給 CallerApp。

另一方面，CallerApp 必須設定好正確的 package 名稱和 class 名稱才能透過 ActivityStarter 元件來呼叫 SubroutineApp。它也必須設定 ActivityStarter.ResultName 屬性指定為「APP_INVENTOR_RESULT」。然後 CallerApp 呼叫 SubroutineApp。當 SubroutineApp 結束時會觸發 CallerApp 的 ActivityStarter 的 AfterActivity 事件，且 AfterActivity 的 result 參數就會接到 SubroutineApp 回傳的文字資料。同樣的訊息也可從 ActivityStarter 元件的 Result 屬性中取得。

◆傳遞數值給 App Inventor 應用程式

您可以傳送文字資料給利用 ActivityStarter 元件所呼叫的 App Inventor 應用程式。要達成此目的，請將 ActivityStarter 元件的 ExtraKey 屬性設定為「APP_INVENTOR_START」，接著將 ExtraValue 屬性設定為所要傳遞的文字資料。被呼叫的 App Inventor 程式就能從 Control → get start text 指令來擷取這筆資料。



◆從其他程式回傳結果

從其他程式取回數值的過程和從 App Inventor 應用程式中取回數值很類似。一般情況下，程式都會回傳一個指定名稱的結果，這個名稱是從 ActivityStarter 元件的 ResultName 屬性中來指定。該名稱用於 App Inventor 應用程式是 APP_INVENTOR_RESULT。其他不是 App Inventor 建立的程式將會使用其他名稱，您需要知道這些名稱才能順利從該應用程式取回數值。一般來說，有些開發者商會提供這些資訊的相關說明，不然您就得從原始碼中來尋寶。

不是所有的 Android 程式都使用 Result 和 ResultName 的機制。舉例來說，有些應用程式是透過 ResultType and ResultUri 來回傳資料。同樣地，您會需要從應用程式開發者取得相關訊息才能順利使用。

給進階玩家：

更具體地說 (參照 Android developer 網站)，一個應用程式可以被設計成可回傳一個 intent。ActivityStarter 使用指定的 ResultName 來存取 intent.getStringExtra(resultName) 並產生結果。ResultType 和 ResultUri 的值來自 intent.getType() 和 intent.getType()。

範例：從 SD 記憶卡中選擇資料夾

以下是一個如何使用第三方應用程式的範例：

AndExplorer from Lysesoft 這個程式讓您從 SD 記憶卡中選取資料夾。您可以用 ActivityStarter 呼叫 AndExplorer，以達到選取檔案的功能。您需要先安裝 AndExplorer 在您的手機上，請直接到 Google Play 下載。

要呼叫 AndExplorer 去選擇一個檔案，您需要這樣設定 ActivityStarter：

- Action: android.intent.action.PICK
- dataType: vnd.android.cursor.dir/lysesoft.andexplorer.file
- dataURI: file:///sdcard

當您啟動程式並選擇一個檔案時，該檔案名稱就是 ResultUri。另外，ResultType 會指出該檔案的類型，舉例來說，image/jpeg 或 audio/mpeg。藉此您可以寫一個從 SD 記憶卡中選擇指定檔案的應用程式，也可顯示圖片或者撥放音樂。

如何知道有那些應用程式可以用？

若您的應用程式需要呼叫其他程式，您得在使用前檢查那些程式是否已經安裝好了。您可利用 ActivityStarter 的 ResolveActivity 方法來檢查，它會根據指定的 package 或 intent 資訊來回傳所希望啟動程式的名稱。若此名稱是空的，那麼就可藉此警告使用者，尚未安裝所需要的程式。



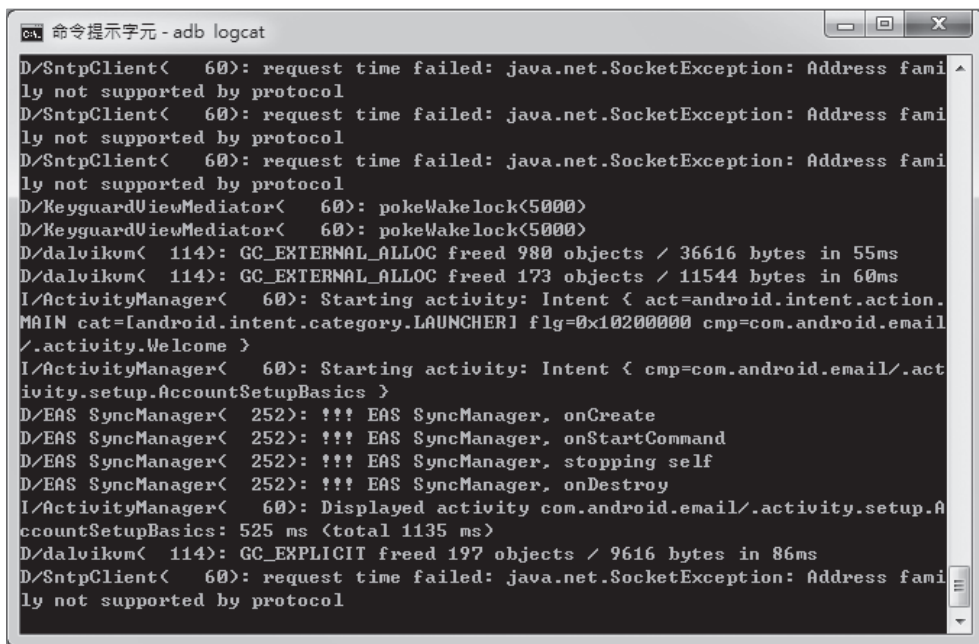
C-2 系統紀錄

除了 Blocks Editor 中的除錯工具，您有時候可以從不同種類的系統紀錄（system log）中得到許多有用的資訊。不過這項技巧適用於資深開發者。

◆ Android 紀錄和 ADB（進階）

若您的 Android 裝置出了點毛病，您可以從查 Android 紀錄找到資訊。例如當應用程式的所需空間不足時，這會儲存在紀錄裡。在紀錄裡的訊息不是太好懂，但您可透過 Notifier 元件的 LogError 指令，讓應用程式可以將訊息寫入紀錄中，而這也是一個不錯的除錯方法之一。

您可使用 Android Debug Bridge（ADB）來讀取紀錄。ADB 在您當初安裝 App Inventor Installer 時就已經安裝好了，然後您可使用 logcat 選項來查看紀錄。請先啟動模擬器或連接實體裝置，接著在電腦上的終端機（terminal）來啟動 ADB，在 App Inventor 安裝資料夾下輸入「adb logcat」就能看到完整的系統紀錄（下圖）。若您正在除錯，您應該把紀錄打開，再次讓錯誤發生，然後看看紀錄中說了些什麼。



```
cat 命令提示字元 - adb logcat
D/SntpClient( 60): request time failed: java.net.SocketException: Address fami
ly not supported by protocol
D/SntpClient( 60): request time failed: java.net.SocketException: Address fami
ly not supported by protocol
D/SntpClient( 60): request time failed: java.net.SocketException: Address fami
ly not supported by protocol
D/KeyguardViewMediator( 60): pokeWakelock<5000>
D/KeyguardViewMediator( 60): pokeWakelock<5000>
D/dalvikom( 114): GC_EXTERNAL_ALLOC freed 980 objects / 36616 bytes in 55ms
D/dalvikom( 114): GC_EXTERNAL_ALLOC freed 173 objects / 11544 bytes in 60ms
I/ActivityManager( 60): Starting activity: Intent < act=android.intent.action.
MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.android.email
/.activity.Welcome >
I/ActivityManager( 60): Starting activity: Intent < cmp=com.android.email/.act
ivity.setup.AccountSetupBasics >
D/EAS SyncManager( 252): !!! EAS SyncManager, onCreate
D/EAS SyncManager( 252): !!! EAS SyncManager, onStartCommand
D/EAS SyncManager( 252): !!! EAS SyncManager, stopping self
D/EAS SyncManager( 252): !!! EAS SyncManager, onDestroy
I/ActivityManager( 60): Displayed activity com.android.email/.activity.setup.A
ccountSetupBasics: 525 ms (total 1135 ms)
D/dalvikom( 114): GC_EXPLICIT freed 197 objects / 9616 bytes in 86ms
D/SntpClient( 60): request time failed: java.net.SocketException: Address fami
ly not supported by protocol
```

ADB 的實際位置根據不同的作業系統而有所差異：

- MacOS：/Applications/AppInventor/commands-for-Appinventor
- GNU/Linux：/usr/google/appinventor-extras/commands-for-Appinventor
- Windows：它有可能出現在好幾個地方，請在您的裝置上搜尋 \Android\appinventor-extras 這個資料夾，裡面應該有 adb.exe 這個檔案。

更多的 ADB 資訊，到 Android Debug Bridge 網頁（<http://developer.android.com/guide/developing/tools/adb.html>）查詢。



◆ Java 系統紀錄（進階）

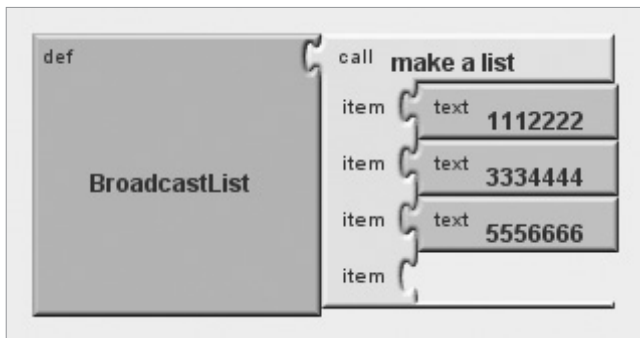
不是所有的錯誤都是您造成的。App Inventor 也是會有錯誤的，且您說不定已經碰過它們了。除了回報錯誤外，您可做的事並不多，但您可能會知道到底發生了什麼事—或者至少看看錯誤訊息也好—透過 Java 控制臺（Java Console）就能顯示系統紀錄。

Blocks Editor 事實上就是一個 Java 程式，許多動作都會直接紀錄在 Java 系統紀錄中。在操作 Blocks Editor 的同時也將 Java 控制臺打開，就可看到這些訊息。您需要在 Java 中進行偏好設定，這樣才能在執行 Blocks Editor 的同時也將 Java 控制臺打開。設定方法也根據您的作業系統而有所差異：

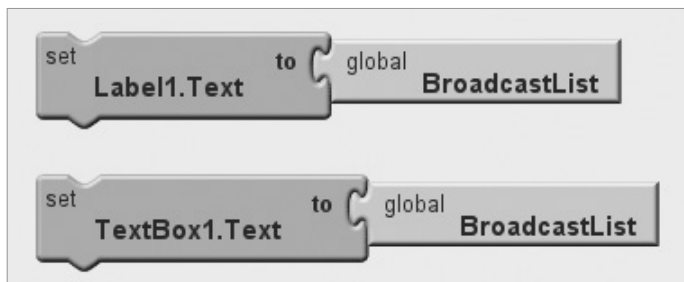
- MacOS：在應用程式（Applications）→ 公用程式（Utilities）→ Java 選項（Java Preferences）中點選 Advanced 選項。您會在此找到顯示控制臺的選項。
- Windows：在控制臺中找到 Java，點選 Advanced 選項。您會在此找到顯示控制臺的選項。
- GNU/Linux：您需要先找到在 Java 安裝目錄中的 javaws 程式。輸入「javaws-viwer」指令就可以打開控制面板。選擇 Advanced 選項，您會在此找到顯示控制臺的選項。

C-3 顯示清單列表

您會常常使用清單來儲存一大筆資料。例如，下面的 BroadcastList 清單儲存了三筆電話號碼：



當您想要在程式畫面上顯示這個清單內容時，您可以把它設為某個元件的 Text 欄位，例如 TextBox 或 Label 都可以，如下圖：



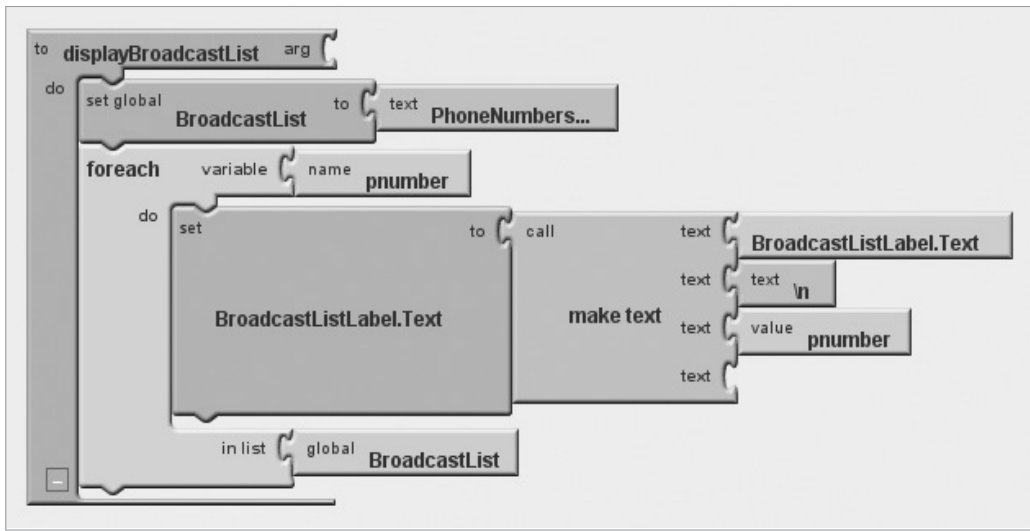
在手機畫面上，清單內容會以空格隔開，並以小括號包起來顯示：

(1112222 3334444 5556666)



您可以看到資料，但這樣還不夠好。一般來說，我們會將清單內容用逗號隔開或斷行顯示。要把清單顯示得更漂亮，您得將它行列化，定義一個副程式來根據您希望的格式來顯示清單內容。您可以在不同的橫列來顯示清單內容，加入說明文字，只要您想的到的幾乎都可以。要行列化一個清單，您可以利用 `foreach` 迴圈來依序將清單內容顯示在 `Label` 上。

以下的副程式可以將 `BroadcastList` 清單內容顯示在不同列。您之後隨時修改內容並妥善運用。



◆ displayBroadcastList 副程式如何運作

BroadcasatListLabel 的 Text 欄位會先被指定為「Phone Numbers...」文字。接著進到 foreach 迴圈，它會根據 BroadcastList 清單內容數目來決定它執行的次數。如果 BroadcastList 清單中有三個項目，foreach 迴圈就會剛好執行 3 次。每次迴圈重複時，pnumber 變數就會依序代表清單內容面。第一次是 1112222，第二次是 3334444，第三次就是 5556666。每次 make a text 指令都會在 BroadcastListLabel.Text 指令與 pnumber 變數之間加入一個換行符號 \n。

◆ 換行符號 \n

文字物件一般來說包含了字母、數字與標點符號，但它也可儲存一些特殊的控制符號，這些符號不會被當作字元來處理。 \n 是一個控制符號，當它出現在一串文字當中時，它的意思是「在顯示下一個項目之前，先移到下一行的開頭」。因此「1112222\n3334444\n5556666」會這樣顯示：

```
1112222
3334444
5556666
```

讓我們來好好看看這段副程式，了解 BroadcastListLabel 為什麼會這樣顯示，因此需要好好檢查所有變數與元件屬性。由於此處使用的是 foreach 迴圈，因此只需要檢查每次迴圈結束時各項目的值，如表 C-1。



表 C-1 不同時間點的 pnumber 變數內容

循環	pnumber 變數內容	PhoneListLabel.Text 內容
進入 foreach 迴圈前	無	PhoneNumbers..
第一次迴圈結束	1112222	PhoneNumbers...\n1112222
第二次迴圈結束	3334444	PhoneNumbers...\n1112222\n3334444
第三次迴圈結束	3334444	PhoneNumbers...\n1112222\n3334444\n5556666

從表 C-1 中可以得知，每次 foreach 迴圈執行時都會將 BroadcastList 清單的下一個內容放入 pnumber 變數中。因此在第一次迴圈結束時，pnumber 變數的內容就是 1112222、第二是 3334444，最後一次則是 5556666。foreach 迴圈幫您省了不少工夫，否則我們必須逐一明確地將各個清單內容指定為 pnumber 變數內容，這樣就輕鬆多了。

C-4 設定元件尺寸

當您在 Designer 頁面中新增一個可視元件時，您可以進一步調整它的寬高等尺寸設定。您有以下三種選擇：

1. 自動 (Automatic)：由系統自動設定寬高。
2. 填滿父元件 (Fill parent)：自動填滿可用空間。
3. 自行設定寬高，單位為像素。

由於各家 Android 裝置的螢幕解析度不一，因此我們建議您盡量指定各元件的絕對像素尺寸比較好。請注意 Automatic and Fill parent 這兩個設定值也適用於可包含其它元件的畫面元件，例如垂直排列元件與水平排列元件，請看下說明：

◆ Screen 元件

Screen 元件有一個 Scrollable 屬性，如果勾選它，您的 Screen 元件就會像一個高度設定為 Automatic 的垂直排列元件。反之如果未勾選 Scrollable 屬性，Screen 元件就是一個固定高度的垂直排列元件。

◆ 垂直排列元件 VerticalArrangement

在垂直排列元件中的所有元件都是貼齊畫面左側，由上而下排列。我們幫您整理出各種可能遇到的情況，請看以下說明。

1. 如果一個垂直排列元件的寬度屬性 Width 為 Automatic，則這個垂直排列元件的實際寬度會由它所包含最寬的元件來決定，且該元件的寬度設定值不可以是 Fill Parent。



2. 如果一個垂直排列元件的 寬度屬性 Width 為 Automatic，且其所包含的元件的寬度設定皆為 Fill Parent，則這個垂直排列元件的實際寬度會由它所包含元件的寬度來決定。
3. 如果一個垂直排列元件的 寬度屬性 Width 為 Automatic 且未包含任何元件，則它的寬度為 100 像素。
4. 如果一個垂直排列元件的高度屬性 Height 為 Automatic，則這個垂直排列元件的實際高度會是由它所包含所有元件的高度總和。
5. 如果一個垂直排列元件的高度屬性 Height 為 Automatic，其所包含所有高度設定為 Fill Parent 的元件所呈現出的效果與設定為 Automatic 相同。
6. 如果一個垂直排列元件的高度屬性 Height 為 Fill Parent 或指定像素，其所包含所有高度設定為 Fill Parent 的元件將會平均分配這個垂直排列元件的高度。

◆ 水平排列元件 HorizontalArrangement

在水平排列元件中的所有元件都是水平排列，高度則是置中對齊。我們幫您整理出各種可能遇到的情況，請看以下說明：

1. 如果一個水平排列元件的高度屬性 Height 為 Automatic，則這個水平排列元件的實際高度會由它所包含最高的元件來決定，且該元件的高度設定值不可以是 Fill Parent。
2. 如果一個水平排列元件的高度屬性 Height 為 Automatic，且其所包含的元件的高度設定皆為 Fill Parent，則這個水平排列元件的實際高度會由它所包含元件的高度來決定。

3. 如果一個水平排列元件的高度屬性 **Height** 為 **Automatic** 且未包含任何元件，則它的高度為 100 像素。
4. 如果一個水平排列元件的寬度屬性 **Width** 為 **Automatic**，則這個水平排列元件的實際寬度會是它所包含所有元件的寬度總和。
5. 如果一個水平排列元件的寬度屬性 **Width** 為 **Automatic**，其所包含所有寬度設定為 **Fill Parent** 的元件所呈現出的效果與設定為 **Automatic** 相同。
6. 如果一個水平排列元件的寬度屬性 **Width** 為 **Fill Parent** 或指定像素，其所包含所有寬度設定為 **Fill Parent** 的元件將會平均分配這個水平排列元件的寬度。

◆表格排列元件 **TableArrangement**

在表格排列元件中的元件都是依照行列排得整整齊齊，如果在同一格中放入多個元件的話，您也只能看到最後被放入的那一個元件而已。

每一列中的元件都是根據高度來置中對齊。我們幫您整理出各種可能遇到的情況，請看以下說明：

1. 每一行的寬度是由其所包含最寬的元件所決定的。當計算每行寬度時，寬度屬性設定為 **Fill Parent** 的元件其寬度就是 **automatic**。然而，各元件會自動填滿它所在那一行的寬度。
2. 每一列的高度是由其所包含最高的元件所決定的，且這些元件的高度設定值不可為 **Fill Parent**。如果某一系列所包含的元件中，它們的高度設定皆為 **Fill Parent**，那麼這一系列的高度就是由所包含元件的高度來自動決定。



◆ 畫布 Canvas

畫布只能放置 **Ball** 與 **ImageSprite** 這兩個動畫元件，我們藉由指定它們的 **X** 與 **Y** 這兩個屬性來設定它們的位置。**Ball** 的寬度與高度當然就是它的半徑，另外如果我們將一個 **ImageSprite** 元件的寬高設定為 **Automatic** 或 **Fill Parent**，那麼它的實際寬高就會是它所包含圖像的尺寸。

C-5 存取圖像和聲音

如同一般的 **Android** 程式，您的 **App Inventor** 程式也能從各種檔案位置來存取音效、圖片以及影像，請看以下說明：

◆ 應用程式資源

在 **Designer** 頁面中的媒體（**Media**）區中所包含的檔案是這個應用程式可以使用的資源（**assets**），這些檔案就是應用程式的一部分。安裝這個應用程式的人會將這些媒體檔案連同應用程式本身一併安裝。您可以從 **Designer** 頁面中來處理這些檔案，相當方便。您也可以透過這些檔案的名稱（檔名不可包含特殊符號）來在應用程式中存取它們。例如，我們可以這樣來使用一張名為 **kitty.png** 的圖檔：將某個 **image** 元件的 **Picture** 屬性設定為「**kitty.png**」這個字串就可以了。對於音效 / 音樂檔（**Sound** 或 **Player** 元件）或是影像檔（**VideoPlayer**）也是一樣的使用方法。

應用程式程式可說是最方便的檔案存取方式，但由於它們會跟著應用程式一起打包，所以檔案大小不建議太大，最好控制在幾 **MB** 左右。我們建議您使用小張圖檔以及較短的聲音片段，最好不要在應用程式中塞入一整首歌或影片檔。

◆ Android 裝置上的 SD 記憶卡

您可藉由以「/sdcard」開頭的檔案名稱來 You can access files on your phone's SD (secure digital) card。以下是一首歌在 SD 記憶卡的路徑：「/sdcard/Music/Blondie/The Best of Blondie/Heart of Glass.mp3」，您可以將這串文字設定為 Player 元件的 source，接著就可以請 Player 開始放音樂了（當然記憶卡中必須真的有這個檔案才行）。請注意檔案名稱要完整，包含副檔名，例如「.mp3」。

Android 系統還有其他方法以 URL 方式來指定 SD 記憶卡中的檔案位置。請使用「file:///sdcard」做為檔案路徑開頭，並使用 URL 編碼就可以了。例如，空格是"%20"。因此上述的 Player 元件 source 屬性就要設定為：「file:///sdcard/Music/Blondie/The%20Best%20of%20Blondie/Heart%20of%20Glass.mp3」

請注意，這時候您應該使用 Player 元件而非 Sound 元件，因為 Sound 元件是用來播放較短的音效檔，而非一整首歌。圖片與影片檔的設定方式也是一樣的。

App Inventor 目前為止還無法將檔案寫入 Android 裝置上的 SD 記憶卡，也無法列出 SD 記憶卡中的檔案名稱。您必須使用其他的應用程式或是 Android 的檔案管理員來做這件事。

SD 記憶卡可以提供寬裕的媒體檔案儲存空間，這比通通和應用程式綁在一起來得好多了。不過這個方法的缺點就在於安裝了您的應用程式的使用者，他們的 SD 記憶卡中就沒有這些檔案，您得另外想辦法讓他們順利取得檔案才能使用應用程式正常運作。



◆ URL 與網頁

您可以使用「http://」開頭的 URL 來存取網路上的檔案，例如您可以將 Image 元件的圖片位址指定為「http://www.google.com/images/srpr/nav_logo14.png」

當然音樂檔與影片檔也可以如法炮製。請確認 URL 確實指定到檔案本身而不是檔案的播放器，這種狀況在網路音樂或影片尤其常見，請務必注意。

◆ 其它 URL

Android 系統使用 URL 路徑來存取裝置上不同位置的媒體檔案。例如，圖片庫的圖片可由「<content://media/external/images/media>」開頭的路徑來存取，您可以透過 ImagePicker 元件來檢視圖片路徑。