**Lab9 Broadcast receiver**

一、本節目的：

* 了解廣播接收程式(Broadcast Receiver)

二、觀念說明：

在啟動Activity與Service時，我們透過startActivity()或是startService()使用Intent來指定要啟動的對象為何。不過Intent也可以用於通知訊息。例如手機電源不夠時，系統就會發出Intent通知，如果有人去接收這個訊息，就能夠顯示電源不足的資訊給使用者。Android中能簡單發出Intent與接收Intent是Broadcast receiver (廣播接收器)元件，他能將實現接收特定的Intent，來回應某個應用程式或系統傳來的訊息，並讓該應用程式做出對應的工作，以下會就會就Broadcast receiver來做介紹。

1. **Broadcast receiver的運作機制**

Broadcast 的運作機制包含兩個部份，送出Intent物件的廣播器(Broadcast)與監聽廣播訊息的接收器(Receiver)。需要這兩個元件彼此搭配才可以完成廣播的功能。

我們前面有教過，要讓某個元件回應使用者事件時，我們會使用Listener (監聽器)來監聽使用者動作，並回應給使用者。這點與Broadcast的目的有些類似，不過兩者皆存在著差異性：

Listener：

* 每個Listener都只能處理一種事件，根據需求有不同的監聽動作，如點擊、長按等，無法接收未定義的事件。
* Listener必須被特定對象綁定後才可以使用。
* Listener的影響範圍受制於特定對象，如對按鈕做監聽，那當按鈕不在螢幕時，監聽事件就沒有效果。

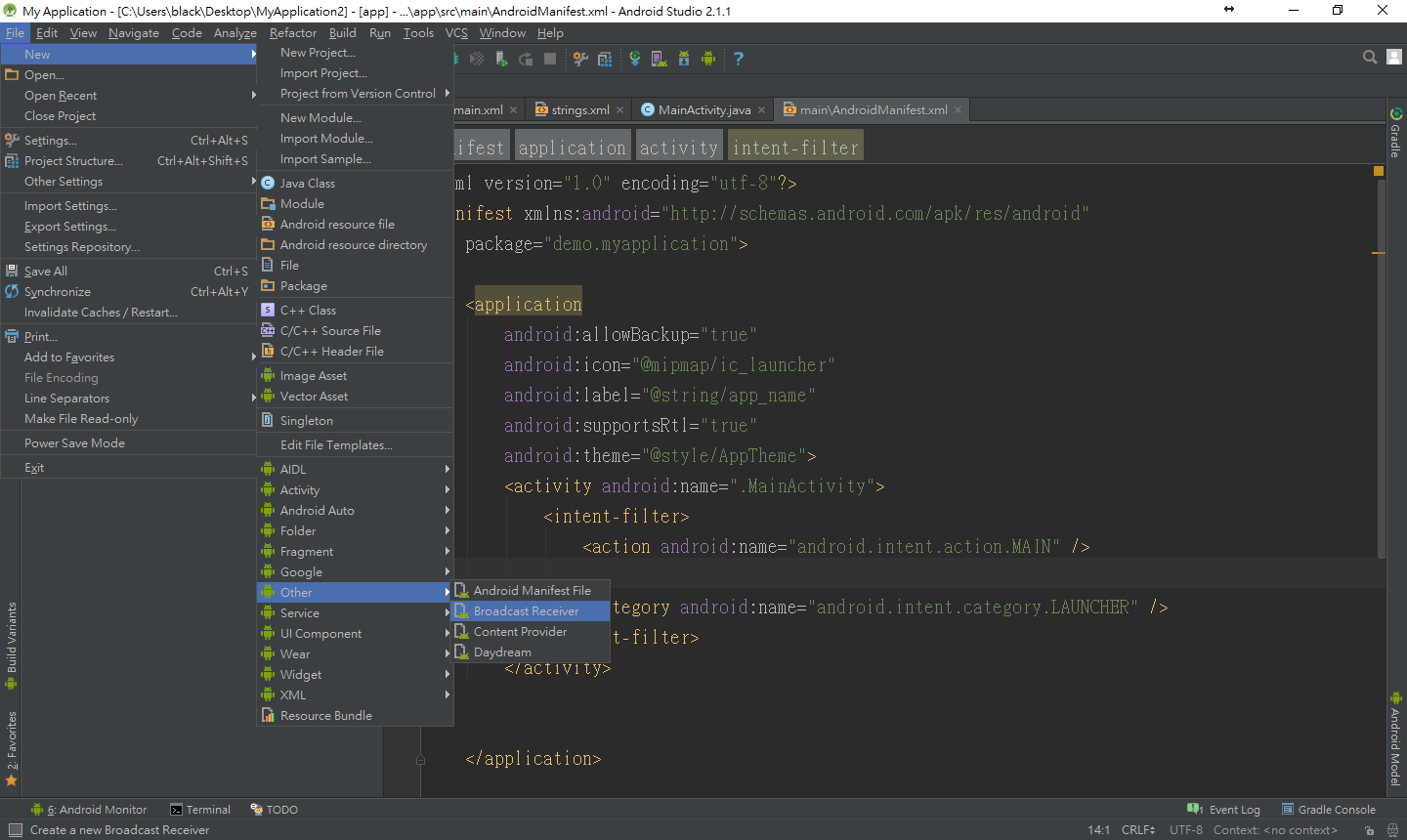
Broadcast：

* Broadcast透過IntentFilter決定要接收對象，只要定義對應的IntentFilter就可以接收複數的廣播。
* Broadcast不需要綁定，是透過註冊與註銷來決定是否接收訊息，但是只能被動的接收訊息，無法知道訊息發送者是誰。
* Broadcast只要有定義註冊，可以接收系統訊息或是自訂訊息。

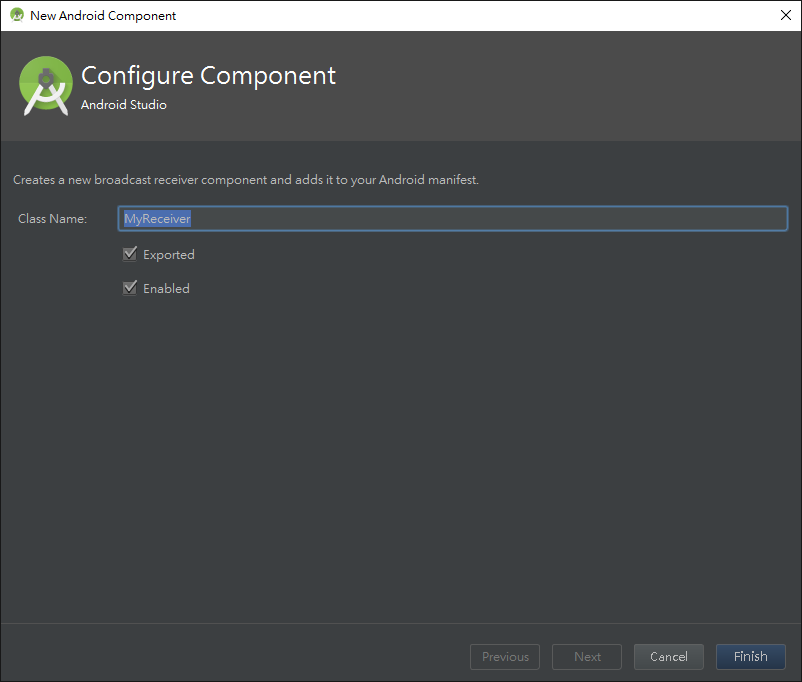
|  |  |  |
| --- | --- | --- |
|  | listener | Broadcast |
| 接收訊息 | 特定事件(點擊、長按…等) | Intent |
| 發送對象 | 明確 | 不明確 |
| 彈性 | 限於特定事件 | IntentFilter決定要接收對象 |
| 範圍 | 限於特定元件對象 | 可接收系統訊息與自訂訊息 |

1. **建立Broadcast receiver**

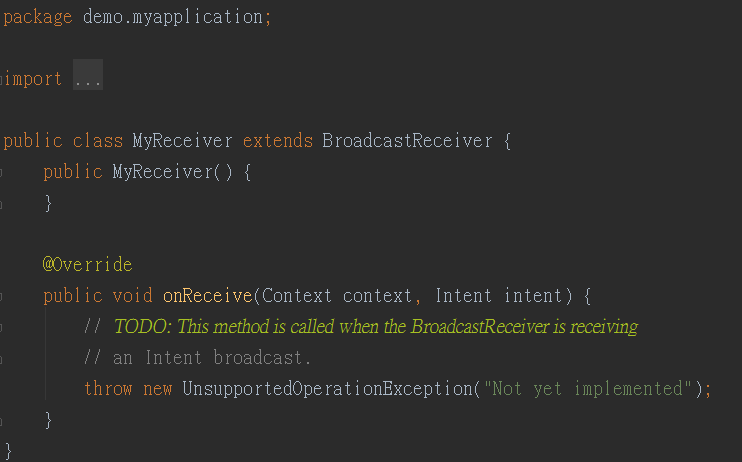
要使用Broadcast，首先我們需要有回應廣播事件的接收器－Receiver，要產生出一個Receiver，首先先選擇File/New/Other/Broadcast receiver來產生出空白的Receiver。



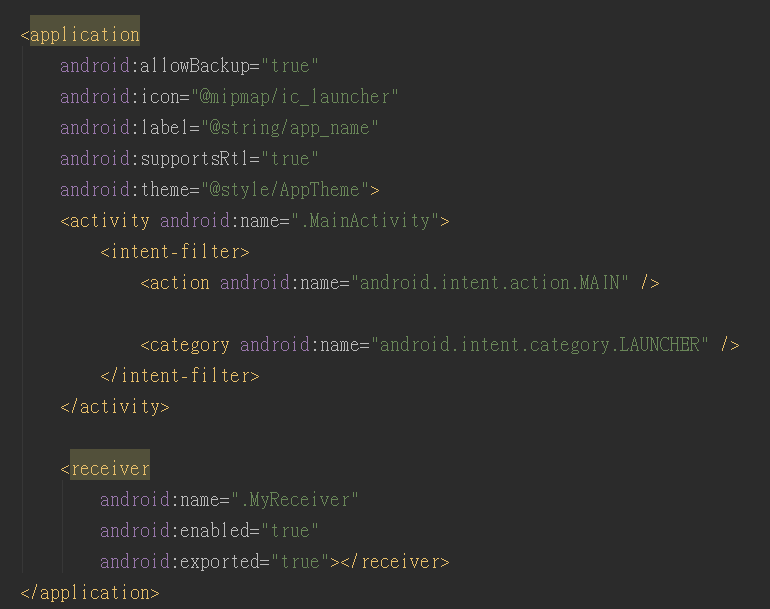
選擇後可於下面的視窗中修改Receiver的名稱，完成後按下Finish。



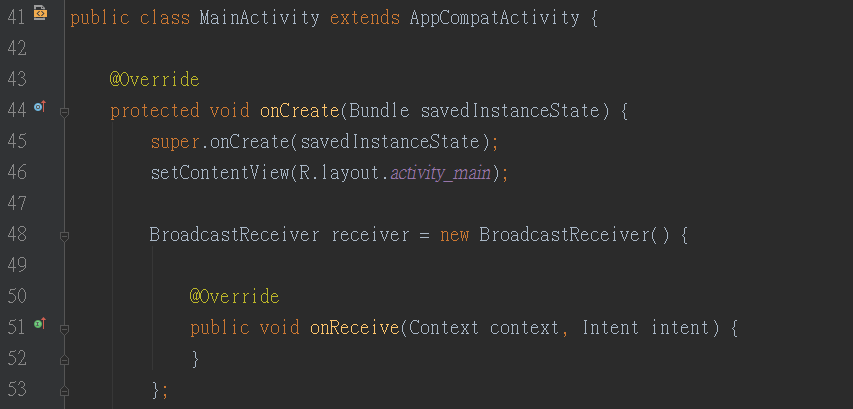
完成後，系統會幫你產生出Receiver的java檔。



AndroidManifest.xml也會自動增加Receiver的資訊



此外也可以直接在Activity中加入BroadcastReceiver類別，這樣就可以免去在AndroidManifest.xml中新增Receiver類別標籤。



直接建立BroadcastReceiver物件

1. **使用Broadcast receiver**

BroadcastReceiver使用上需要透過registerReceiver()「註冊接收器」與unregisterReceiver()「註銷接收器」來建立Receiver。Receiver的用途就是等待廣播傳來，並執行對應的工作。

要讓Receiver接收到廣播，我們需要先定義Receiver想要接收那些廣播事件，這會需要使用到IntentFilter類別。IntentFilter用於定義與過濾想要接收的廣播事件。廣播器必須發出帶有對應「識別字串」的訊息，IntentFilter會藉由「識別字串」決定是否要接收該廣播。而「識別字串」可以是**系統定義**或者是**自行定義**。

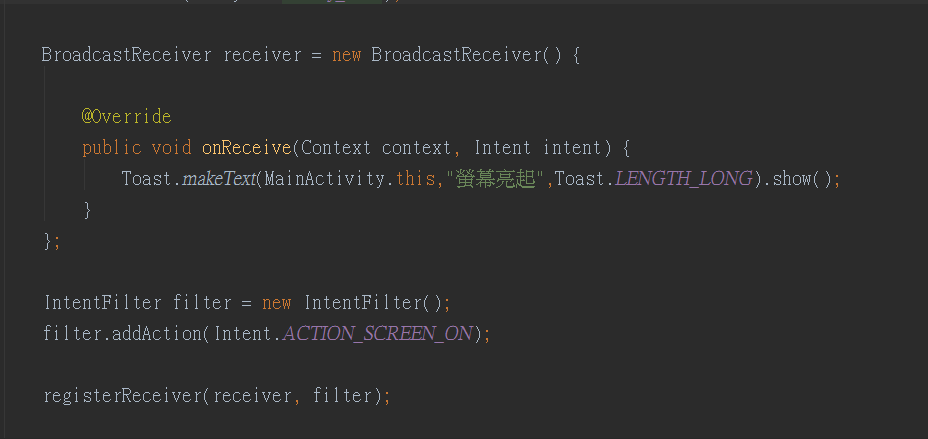
* **系統定義**

系統定義的事件包括:低電量、螢幕開關、耳機插入……等等。以下列出幾個常見廣播接收的識別字串:

|  |  |
| --- | --- |
| **ACTION\_BATTERY\_LOW** | 低電量通知 |
| **ACTION\_HEADSET\_PLUG** | 耳機插入或拔除 |
| **ACTION\_SCREEN\_ON** | 螢幕亮起 |
| **ACTION\_TIMEZONE\_CHANGED** | 時區改變 |

下面例子中透過IntentFilter來監聽螢幕亮起的事件，當螢幕亮起時，會顯示”螢幕亮起”的Toast訊息：

Step1:建立BroadcastReceiver物件



Step4:註冊Receiver

用Toast顯示訊息通知

Step3:建立IntentFilter物件來指定要接收的廣播(螢幕亮起事件)

)

Step2:在onReceive()中加入接收廣播後要執行的動作

IntentFilter.addAction()用於加入一組「識別字串」，之後放入到registerReceiver()中來註冊Receiver。接收到系統發生的事件後便會執行onReceive()來顯示訊息。



當螢幕亮起…

* **自行定義**

自行定義的事件中我們可以在IntentFilter中傳入自行設計的「識別字串」來辨識事件觸發。

如下例子，我們使用「MyMessage」來當作「識別字串」，註冊Receiver時，IntentFilter須設定接收「MyMessage」字串，當收到識別字串為「MyMessage」的Intent時，會取出Intent中夾帶的字串訊息，並用Toast做顯示。

Step1:建立BroadcastReceiver物件



用Toast顯示訊息通知

解析Intent取得字串訊息

要接收的識別字串MyMessage

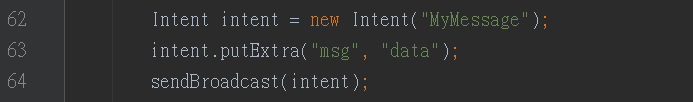
Step3:建立IntentFilter物件來指定

Step4:註冊Receiver

Step2:在onReceive()中加入接收廣播後要執行的動作

有別於上面是系統定義，我們使用sendBroadcast()來自行觸發Receiver，而實作中，我們通常會把sendBroadcast()寫在其他地方，例如其他方法、Activity甚至是Service，並從該處發出sendBroadcast()，實作的程式碼如下：

識別字串



對Receiver發送Intent

如果事前有註冊好Receiver，就可以將其觸發，並傳遞一個Intent至Receiver，該Intent中必須要夾帶「MyMessage」的識別字串才能觸發前面定義的Receiver，也可以在Intent中使用putExtra()加入要傳遞的資料來讓Receiver接收。



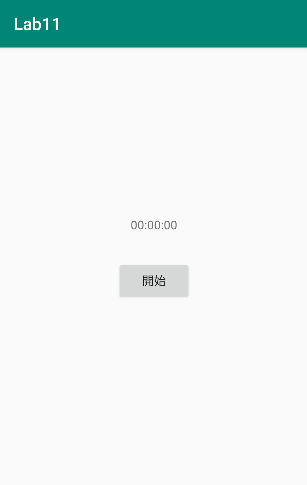
而當不想繼續接收廣播時，要使用unregisterReceiver(receiver)註銷Receiver。

註銷Receiver



1. 設計重點：

* 延續Service與Thread的應用實作一個計數器。
* MainActivity會註冊一個Receiver，接收廣播後會得到一個秒數 (整數值)，並將秒數值呈現於TextView。
* MainActivity按下開始「開始」按鈕後啟動MyService。
* 後台的MyService會建立一個Thread開始讀秒。
* 每秒MyService都會透過Broadcast receiver送一個訊息給前台的MainActivity，來讓MainActivity更新秒數。



**Step2：啟動MyService**

**Step1：註冊一個Receiver**

**Step3：每秒發送Broadcast至Receiver**

**Step4：Receiver更新TextView秒數**

Service

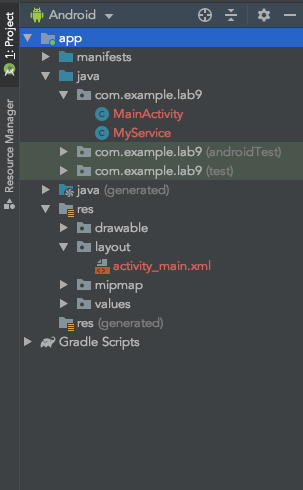
Broadcast

Receiver

四、設計步驟 :

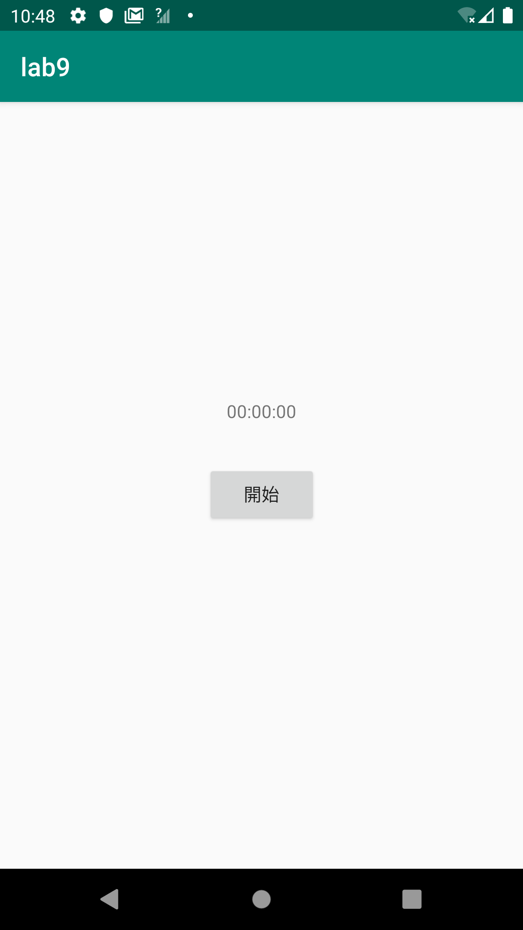
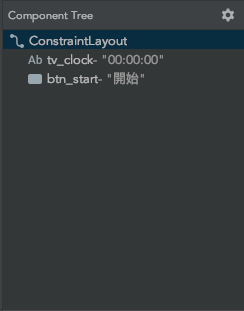
Step1

建立專案，以及對應的java檔和xml檔，如下圖:



Step2

繪製activity\_main.xml，如下圖:

對應的xml如下：





Step3

編寫MainActivity.java，一個啟動Service的按鈕，與註冊一個BroadcastReceiver。

BroadcastReceiver中當收到識別字串為「MyMessage」的Intent時，會取出Intent中夾帶的秒數資訊，並使用TextView做顯示。



建立BroadcastReceiver物件

解析Intent取得秒數資訊

在onReceive()中加入接收廣播後要執行的動作



註冊Receiver

取得Service狀態

啟動Service

註銷廣播

Step4

編寫MyService.java，建立一個Thread每秒發送一次廣播，並把累加的秒數發送到Receiver



使用Thread來計算秒數

計數器+1

計數器數值

計數器狀態

使用Thread來計算秒數，延遲1秒



發送廣播

用Bundle打包後放入Intent

產生帶MyMessage識別字串的Intent