**手機即時訊息顯示程式製作說明書**

**校訂歷史(Revision History)**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **說明** | **作者** |
| 10/28 2013 | 1.0 | 初版 | 張家榮 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. 何謂警報(Alarm)和通知(Notification) 4](#_Toc106471292)

[2. AlarmManager 的使用 4](#_Toc106471294)

[3. 接收警報 6](#_Toc106471297)

[4. NotificationManager 的使用 7](#_Toc106471298)

[5. 通知內容 8](#_Toc106471298)

[6. 成品範例 9](#_Toc106471298)

[7. 成品Demo](#_Toc106471298) 15

[8. 完整程式碼 16](#_Toc106471298)

**何謂警報(Alarm)和通知(Notification)？**

Alarm原意有警告、警鈴之意思，在此系統裡Alarm具有提醒的功效。Alarm能在指定的時間點或是設定好的時間間隔下產生動作以提醒使用者。這些動作就算在App為關閉的狀態下，依然允許啟動。Alarm就像鬧鐘一般，可以自由設定時間點來做提醒。就以手機鬧鐘來說，鬧鈴並不會因為使用者沒開啟鬧鐘程式而停止觸發動作，即便是手機螢幕並未開啟，鬧鐘也依然執行動作。

Notification則是提醒的方式，簡單來說就像發出震動、鈴聲、閃爍等等來達到通知使用者的效果。

Alarm並不會包含在應用程式裡而是存在於外界的程式，而Notification 通常會被用在 Broadcast Receiver 或 Service 等不可見的應用程式中，將兩者相互結合便可以達到提醒、通知等作用

**AlarmManager 的使用**

AlarmManager可以指定於某個時間點動作，只要經由設定變可以達到預期效果。當到達設定時間，程式接到廣播通知便會執行通知動作。App應該要有一段對應 Android 設備重開機後，也應重新設定所對應的程式碼。

1. **宣告AlarmManager**

**private** AlarmManager alarm;

1. **設定AlarmManager服務**

alarm = (AlarmManager) getSystemService(*ALARM\_SERVICE*);

1. **宣告PendingIntent**

**private** PendingIntent pending;

1. **設定Intent服務**

Intent intent = **new** Intent();

intent.setAction("com.example.raito.MY\_ACTION");

1. **設定 PendingIntent服務將Intent服務加入其中**

pending = PendingIntent.*getBroadcast*

(getApplicationContext(), 0, intent, 0);

**PendingIntent的類型如下:**

**0是不做類型設定**

1、PendingIntent.FLAG\_UPDATE\_CURRENT

如果該PendingIntent已經存在，則用新傳入的Intent的Extra更新當前的數據。

2、PendingIntent.FLAG\_ONE\_SHOT

send()只能被執行一次，即是說，假如該通知點擊後不消失，那麼再次點擊不會發生任何事。

3、PendingIntent.FLAG\_NO\_CREATE

如果該PendingIntent不存在，直接返回null而不是創建一個PendingIntent.

4、PendingIntent.FLAG\_CANCEL\_CURRENT

如果該PendingIntent已經存在，則在生成新的之前取消當前的。

1. **將PendingIntent加入AlarmManager服務中**

alarm.setRepeating

(AlarmManager.*RTC\_WAKEUP*, time, 8000,pending);

Time為開始時間  
 8000為8秒後重複執行

Pending為要執行的服務

**設定類型有兩種:**

* 單次: 要設定一個警報只要使用 set 方法，同時指定一個**警報類型**、**觸發時間**及一個**待處理意圖**。如果這個觸發時間是在過去，那這個警報會被立即觸發。

## 多次: 重複性警報就像鬧鐘設定多次提醒一樣。要設定重覆性警報有 2 種方法，setRepeating() 或 setInexactRepeating() ，兩者的差別在於 setRepeating() 可以以精確到毫秒的時間間隔來執行，當然就會比較耗電，使用 setInexactRepeating() 則可以避免耗電。(比較兩者差異程式碼,或者有別的提醒警爆)

setInexactRepeating() 可以使用較不精確的時間間隔來設定，而這些在 AlarmManager 類別中已有內定的常數可使用，如下：

* + - 1. INTERVAL\_FIFTEEN\_MINUTES(每隔 15 分鐘)
      2. INTERVAL\_HALF\_HOUR(每隔半小時)
      3. INTERVAL\_HOUR(每隔 1 小時)
      4. INTERVAL\_HALF\_DAY(每隔12小時)
      5. INTERVAL\_DAY(每隔 24小時)

使用範例如下：

－精確

alarm.setRepeating(AlarmManager.ELAPSED\_REALTIME\_WAKEUP, 10 \* 1000, 10 \* 1000, pending);  
－非精確

alarm.setInexactRepeating(AlarmManager.ELAPSED\_REALTIME\_WAKEUP, 60 \* 1000, AlarmManager.INTERVAL\_FIFTEEN\_MINUTES, pending);

**警報類型有四種：**

1. RTC\_WAKEUP：在指定時間觸發意圖並喚醒裝置。
2. RTC：同上但不喚醒裝置。
3. ELAPSED\_REALTIME：在裝置啟動(開機)後開始計算經過的時間，在到達指定的經過時間觸發意圖，但不喚醒裝置。
4. ELAPSED\_REALTIME\_WAKEUP：同上，但會喚醒裝置。
5. **關閉alarm**

alarm.cancel(pending);

**接收警報**

當 alarm 被觸發時，需要一個 receiver 來接收它，然後指示要做什麼事，所以我們要繼承一個 BroadcastReceiver 並實作它的 onReceive 方法。如下：

public class AlarmReceiver extends BroadcastReceiver {

@Override

public void onReceive(Context context, Intent intent) {

//....do something

}

}

你可以在此發出通知告知使用者，或執行資料更新的動作等等。

**警報補充:**

記得當你實作 BroadcastReceiver 時，在 AndroidManifest.xml 要加入你的 Receiver ，如下：

<receiver android:name="AlarmReceiver"></receiver>

//AlarmReceiver 是你的 Receiver 類別的名稱

此外，在一開始介紹 Alarm 時有說過，當裝置(手機)重新開機後，所有的 Alarm 都將被取消，那該怎麼解決呢？一樣利用 Receiver ，我們可以如同建立 AlarmReceiver 一樣，建立一個 AlarmInitReceiver (繼承 BroadcastReceiver 並實作 onReceive)，接著在 AndroidManifest.xml 中加入這個 Receiver 來接收開機訊息，如下：

<receiver android:name=*"alarm"* > // alarm 為class 名稱

<intent-filter>

<action android:name=*"com.example.raito.MY\_ACTION"* >

// *com.example.raito*為專案名稱

</action>

</intent-filter>

</receiver>

BOOT\_COMPLETED 是 Android 內建的 action，會在開機後送出廣播，所以我們建立一個接收者去接收這個廣播即可。使用這個 action 必須增加使用權限，如下：

<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED"/>

## NotificationManager 的使用

## NotificationManager和AlarmManager一樣只要經過設定就可以達到目的。

## 1. 建立NotificationCompat

NotificationCompat.Builder builder = **new** NotificationCompat.Builder(context);//建立NotificationCompat

**2.再來設定當按下這個通知之後要執行的activity**

Intent notificationIntent = **new** Intent(context, oil.**class**);

//設定當按下這個通知之後要執行的activity

**3.建立所需的通知設定**

builder.setSmallIcon(R.drawable.*fire*)//設定通知小圖示

.setOngoing(**true**)//設定通知是持續的

.setAutoCancel(**true**)//更新Notification而不觸發任何關聯的閃燈、音頻或震動

.setTicker("錢不夠用");//設定狀態列的標題文字

.setWhen(System.currentTimeMillis())//設定何時觸發

.setLargeIcon(icon)//設定通知大圖示

.setContentTitle("Progress") //設定通知的標題

.setProgress(100, 50, false) //設定完成度圖條

.setContent(myRemoteView) //設定自訂的通知view

.setContentIntent(contentIntent)//設定點選呼叫的PendingIntent

.setContentText(intent.getStringExtra("message"))//設定通知內容文字

**4.取得Notification服務**

Notification nf = builder.build();

**5.設定通知類型**

nf.flags = Notification.*FLAG\_INSISTENT*;//設定通知類型

## 通知類型有下列幾種:

## 1. notification.flags |= FLAG\_AUTO\_CANCEL; //在通知欄上點擊此通知後自動清除此通知

## 2. notification.flags |= FLAG\_INSISTENT; //重複發出聲音，直到用戶響應此通知

## 3. notification.flags |= FLAG\_ONGOING\_EVENT; //將此通知放到通知欄的"Ongoing"即"正在運行"組中

## 4. notification.flags |= FLAG\_NO\_CLEAR; //表明在點擊了通知欄中的"清除通知"後，此通知不清除，

## 6.設定PendingIntent服務

PendingIntent contentIntent =

PendingIntent.*getActivity*(context, 0,notificationIntent, 0);

**PendingIntent類型依前面介紹**

## 7.將PendingIntent加入到Notification裡

## nf.contentIntent = contentIntent;

## 8.呼叫NotificationManager服務

NotificationManager nm = (NotificationManager) context.getSystemService(context.*NOTIFICATION\_SERVICE*);

## 9.將Notification加入NotificationManager服務中

nm.notify(0, nf);

## 通知內容

## 

1. 內容標題。

2. 大型圖示。

3. 内容文字。

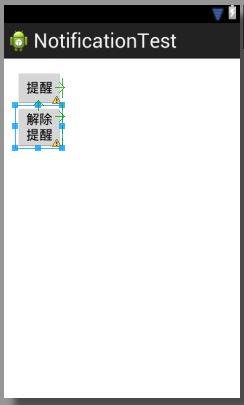
4. 通知個數。

5. 小型圖示。

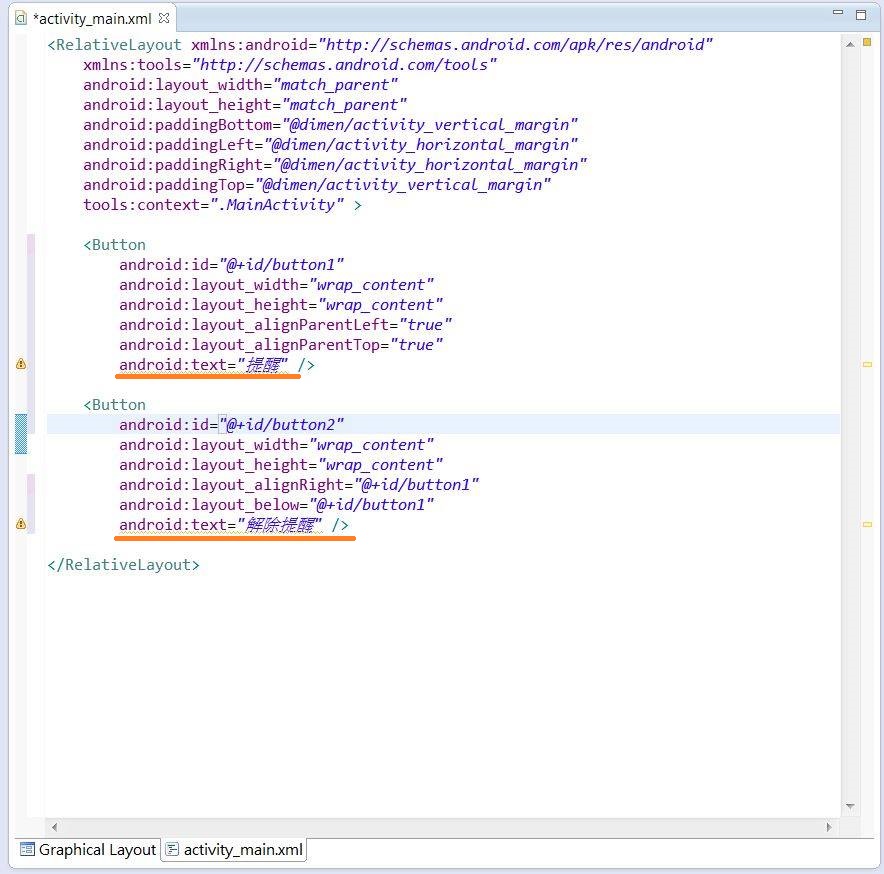
6. 通知時間。

**成品範例**

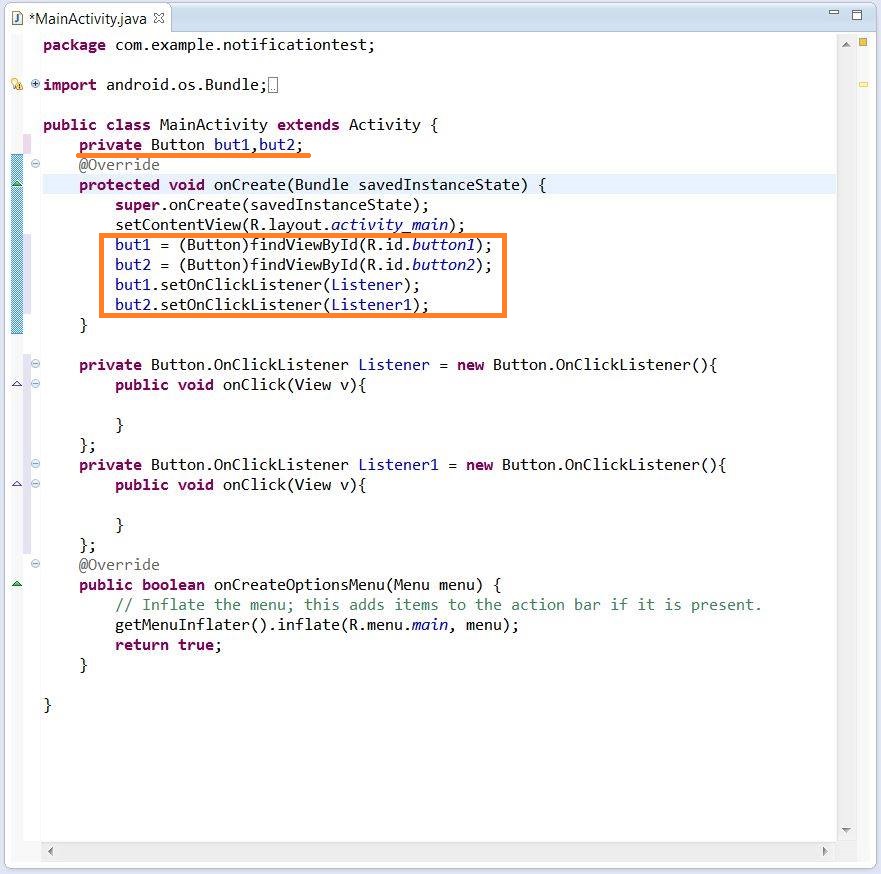
首先建立兩個按鈕。



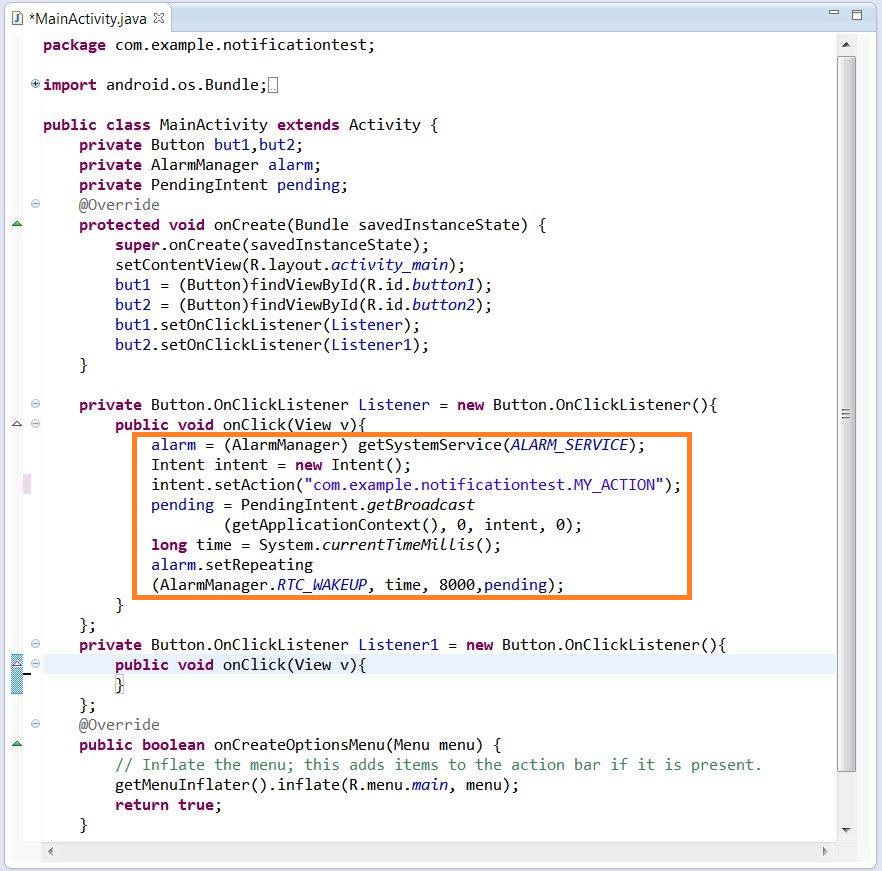
接著在activity\_main.xml裡會看到方才所設置的兩個按鈕，在android:text=”輸入所要的按鈕名稱”便可完成按鈕設置。



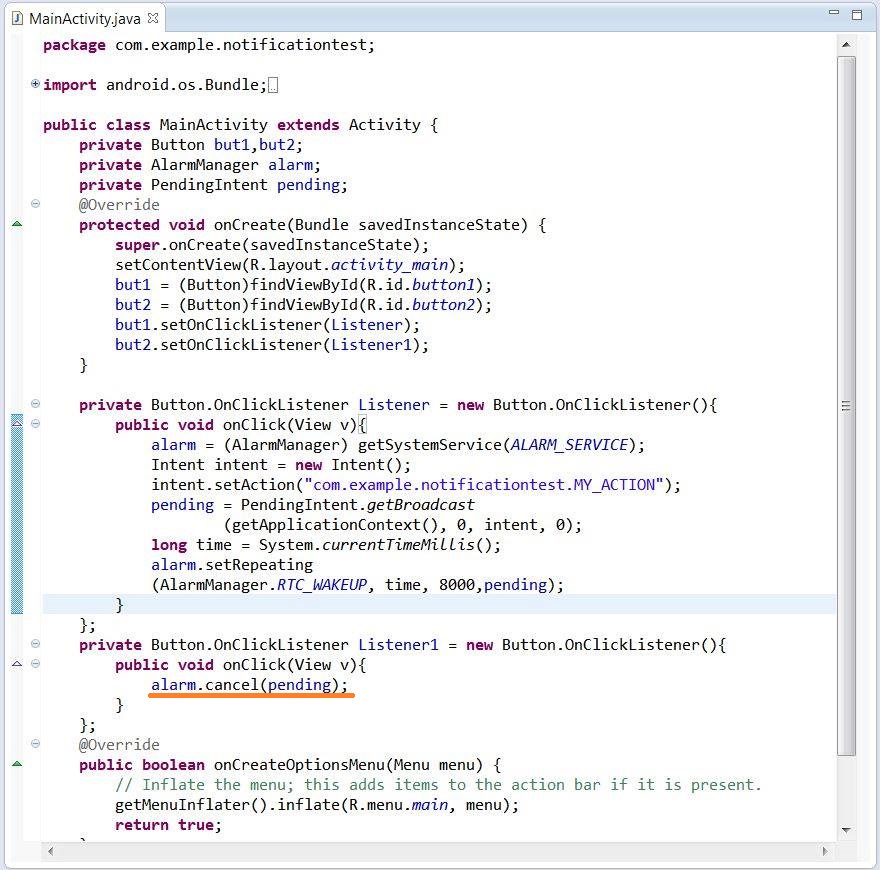
在MainActivity.java內宣告兩個Button，如下圖橘色線條部分；在橘色方框部分則是設置按鈕。



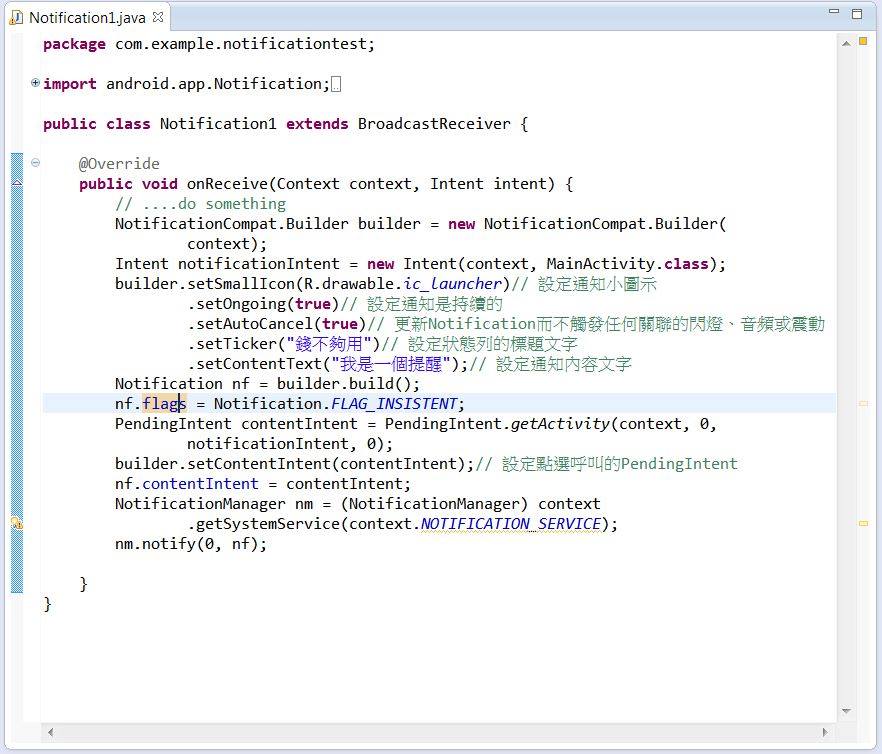
下圖橘色方框部分為button1之動作。



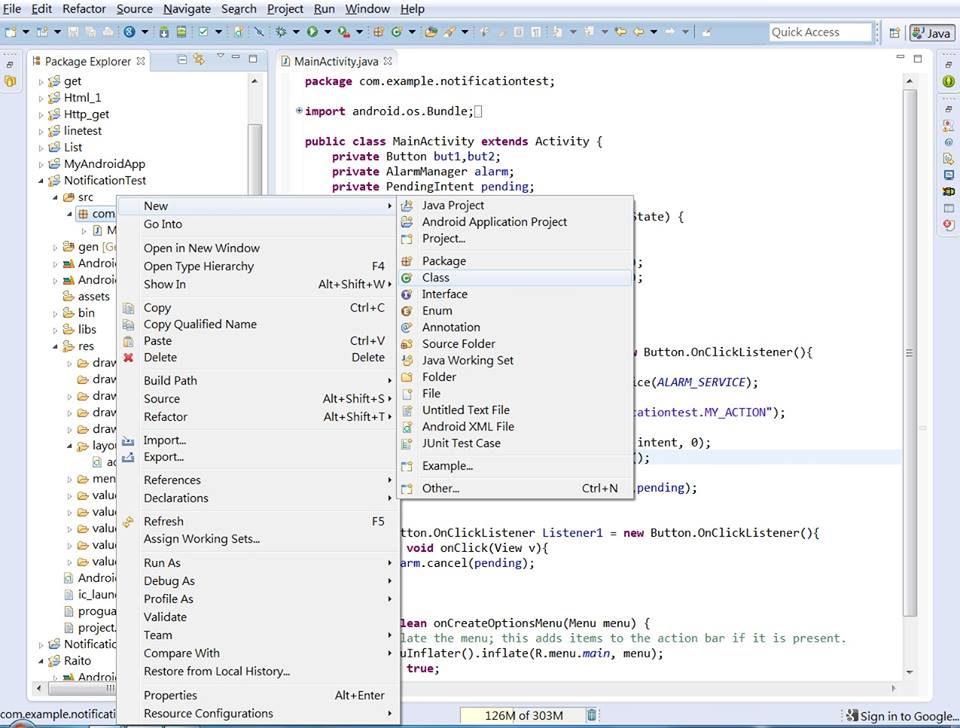
下圖橘線部分為關閉alarm之語法。



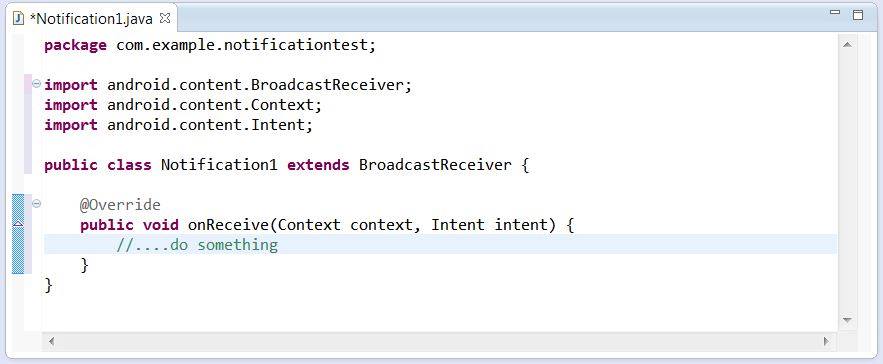
輸入以下程式碼以建置提醒功能。



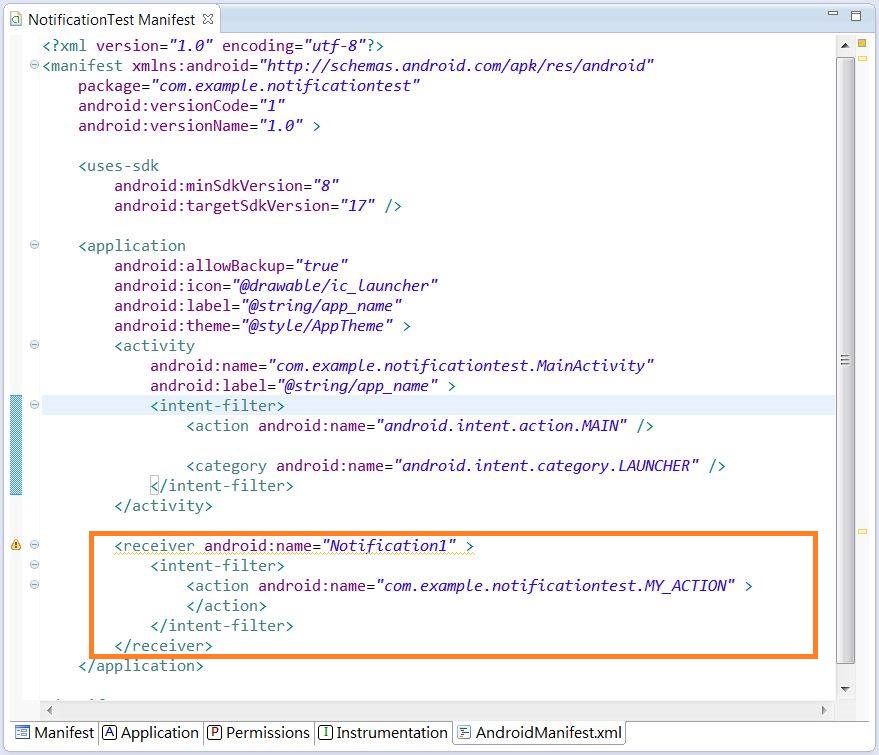
接著按照下圖步驟建置新的class，用來產生提醒功能。

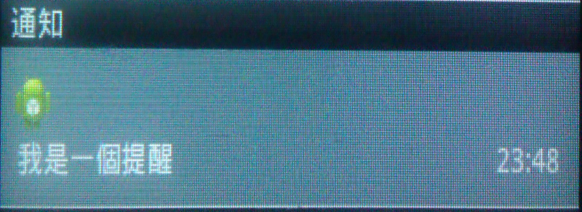


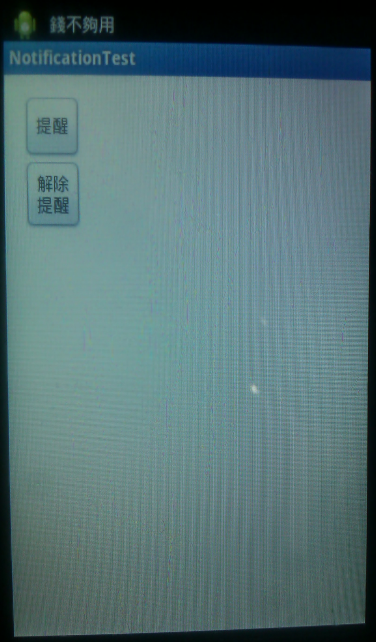
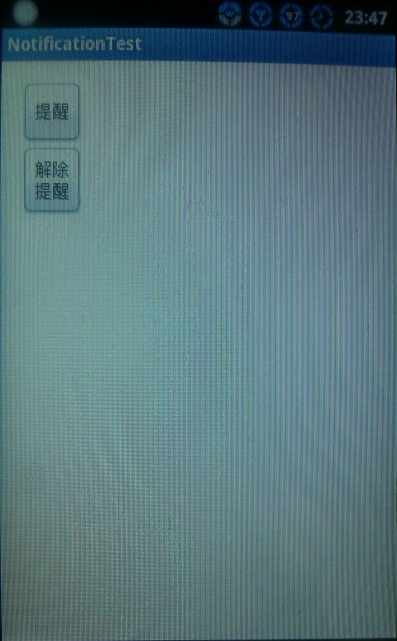
建置好新的class之後，便開始設定接收警報(Alarm)



設置接受警報指令之前，必須建立如下表橘色方框之xml碼。



**成品Demo**



**完整程式碼**

**MainActivity.java**

package com.example.notificationtest;

import android.os.Bundle;

import android.app.Activity;

import android.app.AlarmManager;

import android.app.PendingIntent;

import android.content.Intent;

import android.view.Menu;

import android.view.View;

import android.widget.Button;

public class MainActivity extends Activity {

private Button but1,but2;

private AlarmManager alarm;

private PendingIntent pending;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

but1 = (Button)findViewById(R.id.button1);

but2 = (Button)findViewById(R.id.button2);

but1.setOnClickListener(Listener);

but2.setOnClickListener(Listener1);

}

private Button.OnClickListener Listener = new Button.OnClickListener(){

public void onClick(View v){

alarm = (AlarmManager) getSystemService(ALARM\_SERVICE);

Intent intent = new Intent();

intent.setAction("com.example.notificationtest.MY\_ACTION");

pending = PendingIntent.getBroadcast

(getApplicationContext(), 0, intent, 0);

long time = System.currentTimeMillis();

alarm.setRepeating

(AlarmManager.RTC\_WAKEUP, time, 8000,pending);

}

};

private Button.OnClickListener Listener1 = new Button.OnClickListener(){

public void onClick(View v){

alarm.cancel(pending);

}

};

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.main, menu);

return true;

}

}

**Notification1.java**

package com.example.notificationtest;

import android.app.Notification;

import android.app.NotificationManager;

import android.app.PendingIntent;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.support.v4.app.NotificationCompat;

public class Notification1 extends BroadcastReceiver {

@Override

public void onReceive(Context context, Intent intent) {

// ....do something

NotificationCompat.Builder builder = new NotificationCompat.Builder(

context);

Intent notificationIntent = new Intent(context, MainActivity.class);

builder.setSmallIcon(R.drawable.ic\_launcher)// 設定通知小圖示

.setOngoing(true)// 設定通知是持續的

.setAutoCancel(true)// 更新Notification而不觸發任何關聯的閃燈、音頻或震動

.setTicker("錢不夠用")// 設定狀態列的標題文字

.setContentText("我是一個提醒");// 設定通知內容文字

Notification nf = builder.build();

nf.flags = Notification.FLAG\_INSISTENT;

PendingIntent contentIntent = PendingIntent.getActivity(context, 0,

notificationIntent, 0);

builder.setContentIntent(contentIntent);// 設定點選呼叫的PendingIntent

nf.contentIntent = contentIntent;

NotificationManager nm = (NotificationManager) context

.getSystemService(context.NOTIFICATION\_SERVICE);

nm.notify(0, nf);

}

}

**activity\_main.xml**

<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" >

<Button

android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:layout\_alignParentTop="true"

android:text="提醒" />

<Button

android:id="@+id/button2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignRight="@+id/button1"

android:layout\_below="@+id/button1"

android:text="解除提醒" />

</RelativeLayout>

**AndroidMainifest.xml**

<?xml version="1.0" encoding="utf-8"?>

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

package="com.example.notificationtest"

android:versionCode="1"

android:versionName="1.0" >

<uses-sdk

android:minSdkVersion="8"

android:targetSdkVersion="17" />

<application

android:allowBackup="true"

android:icon="@drawable/ic\_launcher"

android:label="@string/app\_name"

android:theme="@style/AppTheme" >

<activity

android:name="com.example.notificationtest.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>

<receiver android:name="Notification1" >

<intent-filter>

<action android:name="com.example.notificationtest.MY\_ACTION" >

</action>

</intent-filter>

</receiver>

</application>

</manifest>