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。

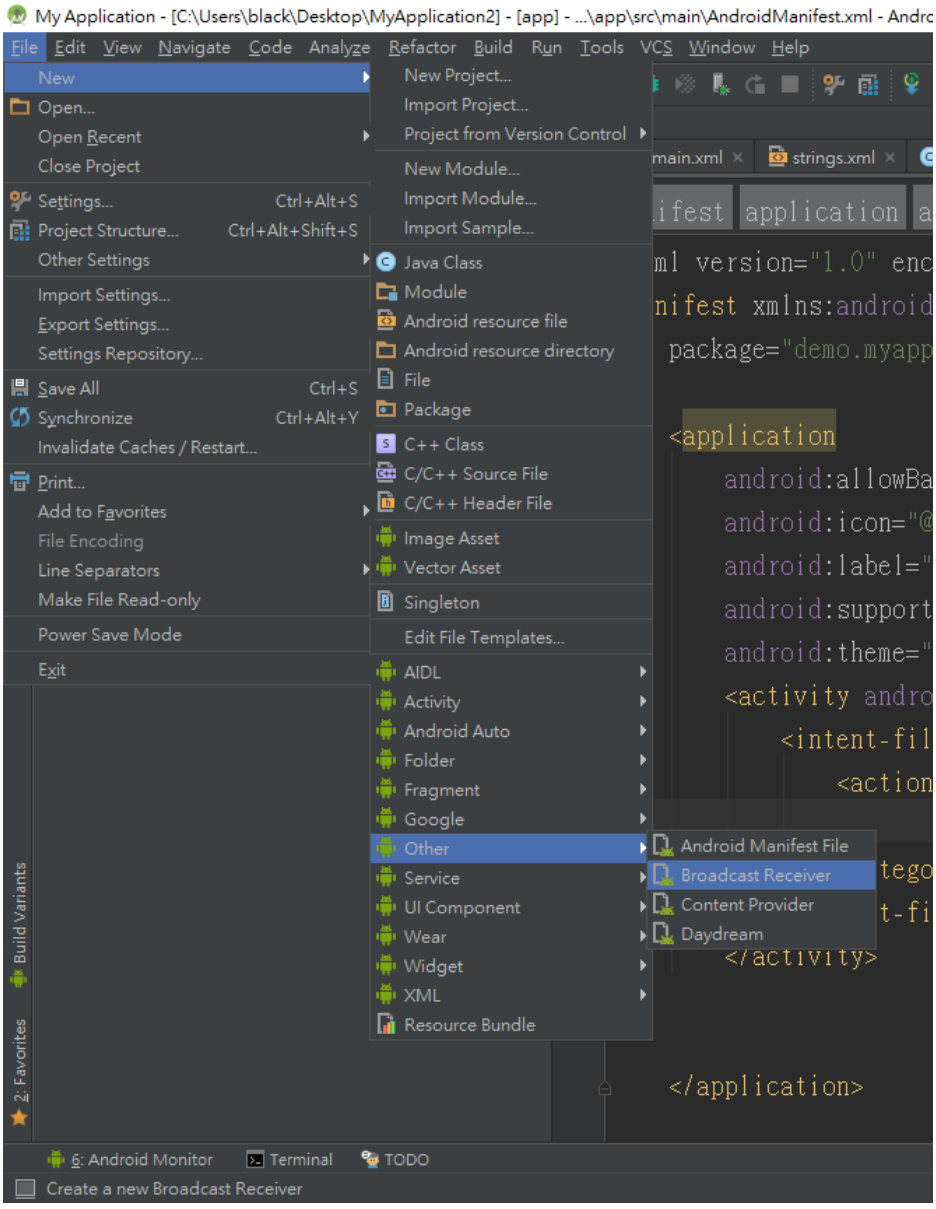
Broadcast：

- 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。

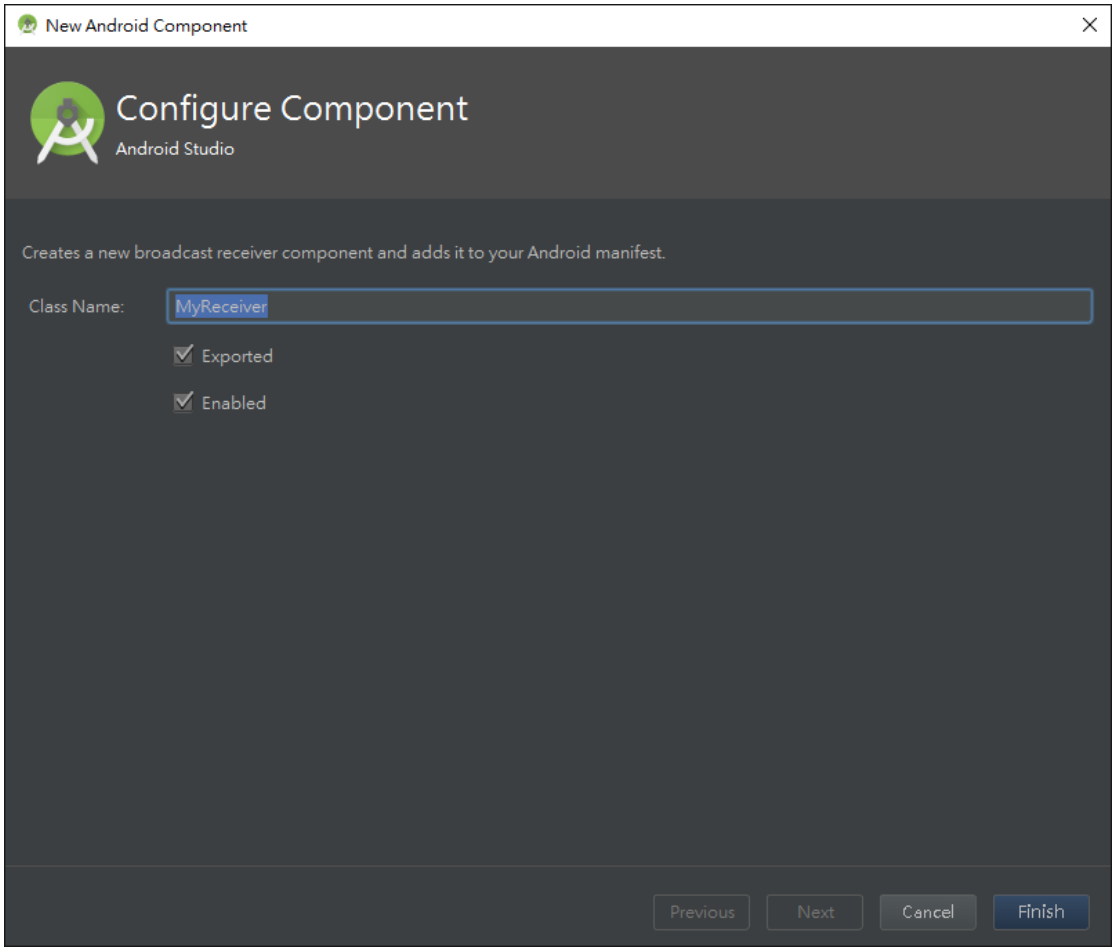
	listener	Broadcast
接收訊息	特定事件(點擊、長按…等)	Intent
發送對象	明確	不明確
彈性	限於特定事件	IntentFilter 決定要接收對象
範圍	限於特定元件對象	可接收系統訊息與自訂訊息

2　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。

```java
package demo.myapplication;

import ...

public class MyReceiver extends BroadcastReceiver {
    public MyReceiver() {
    }

    @Override
    public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the BroadcastReceiver is receiving
        // an Intent broadcast.
        throw new UnsupportedOperationException("Not yet implemented");
    }
}
```

AndroidManifest.xml 也會自動增加 Receiver 的資訊

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <receiver
        android:name=".MyReceiver"
        android:enabled="true"
        android:exported="true"></receiver>
</application>
```

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。

```
41  public class MainActivity extends AppCompatActivity {
42
43      @Override
44      protected void onCreate(Bundle savedInstanceState) {
45          super.onCreate(savedInstanceState);
46          setContentView(R.layout.activity_main);
47          直接建立 BroadcastReceiver 物件
48          BroadcastReceiver receiver = new BroadcastReceiver() {
49
50              @Override
51              public void onReceive(Context context, Intent intent) {
52              }
53          };
```

3 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

```
BroadcastReceiver receiver = new BroadcastReceiver() {

    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(MainActivity.this,"螢幕亮起",Toast.LENGTH_LONG).show();
    }
};
```

Step2:在 onReceive()中加入接收廣播後要執行的動作

用 Toast 顯示訊息通知

```
IntentFilter filter = new IntentFilter();
filter.addAction(Intent.ACTION_SCREEN_ON);
```

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

```
registerReceiver(receiver, filter);
```

Step4:註冊 Receiver

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。

- **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。

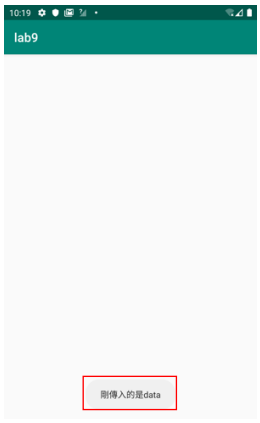
如下例子，我們使用「MyMessage」來當作「識別字串」，註冊 Receiver 時，IntentFilter 須設定接收「MyMessage」字串，當收到識別字串為「MyMessage」的 Intent 時，會取出 Intent 中夾帶的字串訊息，並用 Toast 做顯示。

```
Step1:建立 BroadcastReceiver 物件
BroadcastReceiver receiver = new BroadcastReceiver() {

    @Override      Step2:在 onReceive()中加入接收廣播後要執行的動作
    public void onReceive(Context context, Intent intent) {
        String msg = intent.getExtras().getString( key: "msg");   解析 Intent 取得字串訊息
        Toast.makeText( context: MainActivity.this,  text: "剛剛傳入的是" + msg, Toast.LENGTH_LONG).show();
    }                                                              用 Toast 顯示訊息通知
};

IntentFilter filter = new IntentFilter( action: "MyMessage"); Step3:建立 IntentFilter 物件來指定
                                                               要接收的識別字串 MyMessage
registerReceiver(receiver, filter); Step4:註冊 Receiver
```

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：

```
62          Intent intent = new Intent("MyMessage");識別字串
63          intent.putExtra("msg", "data");
64          sendBroadcast(intent);   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。

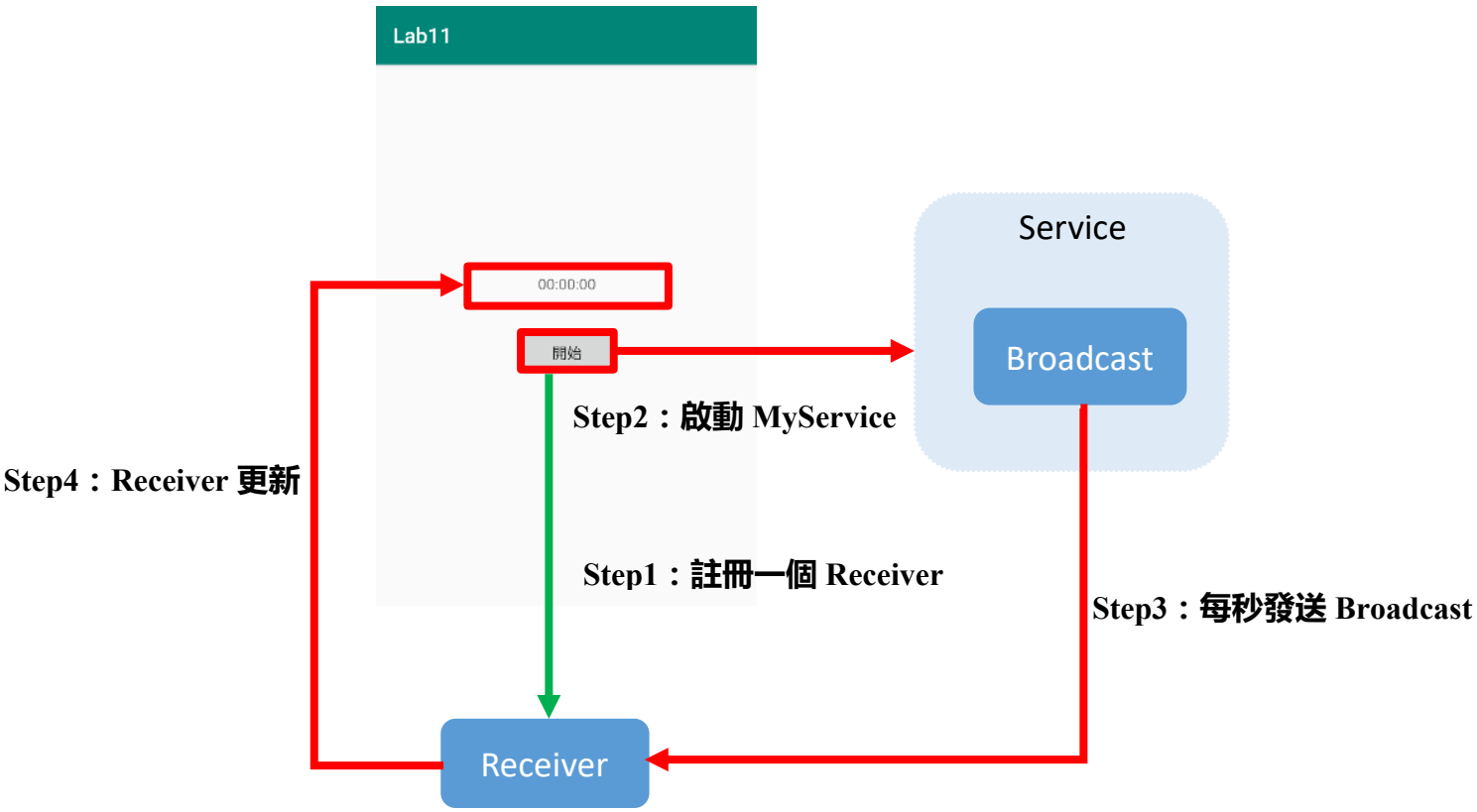


而當不想繼續接收廣播時，要使用 unregisterReceiver(receiver)註銷 Receiver。

```
unregisterReceiver(receiver); 註銷 Receiver
```

三、設計重點：

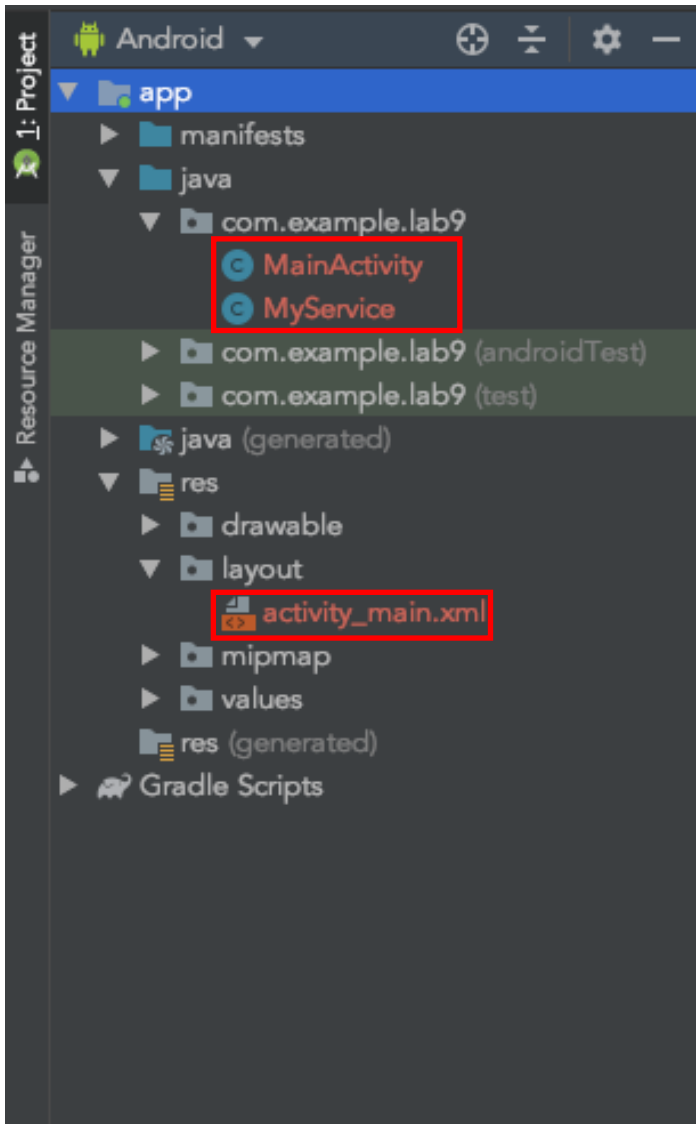
- 延續 Service 與 Thread 的應用實作一個計數器。
- MainActivity 會註冊一個 Receiver，接收廣播後會得到一個秒數 (整數值)，並將秒數值呈現於 TextView。
- MainActivity 按下開始「開始」按鈕後啟動 MyService。
- 後台的 MyService 會建立一個 Thread 開始讀秒。
- 每秒 MyService 都會透過 Broadcast receiver 送一個訊息給前台的 MainActivity，來讓 MainActivity 更新秒數。



四、設計步驟：

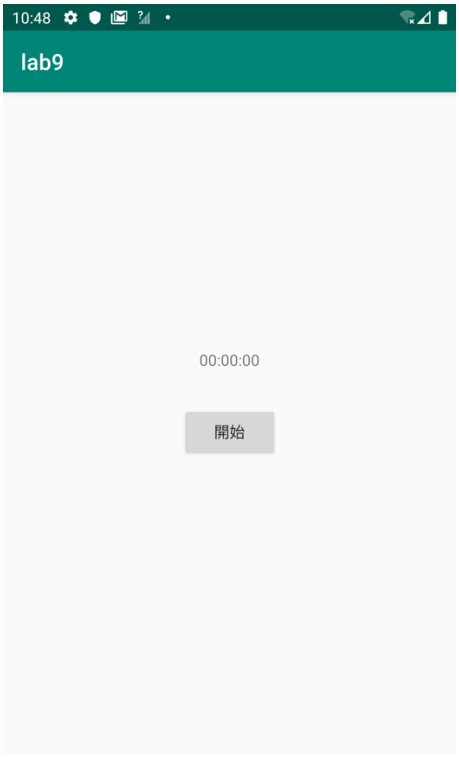
Step1

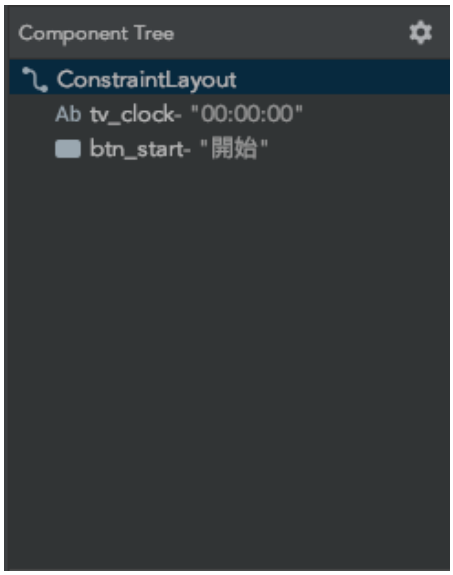
建立專案，以及對應的 java 檔和 xml 檔，如下圖:



Step2

繪製 activity_main.xml，如下圖:





對應的 xml 如下：

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/tv_clock"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="00:00:00"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
```

```
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.4" />

    <Button
        android:id="@+id/btn_start"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="開始"
        app:layout_constraintEnd_toEndOf="@+id/tv_clock"
        app:layout_constraintStart_toStartOf="@+id/tv_clock"
        app:layout_constraintTop_toBottomOf="@+id/tv_clock" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

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。

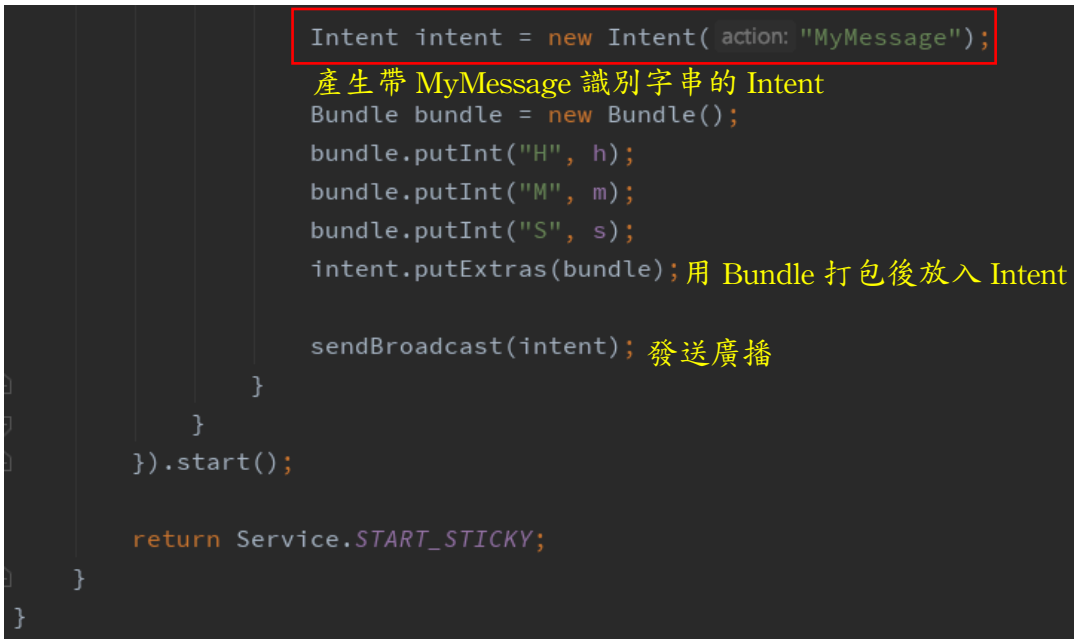
```
public class MainActivity extends AppCompatActivity {
    private TextView tv_clock;
    private Button btn_start;

    private Boolean flag = false;
    建立 BroadcastReceiver 物件
    private BroadcastReceiver receiver = new BroadcastReceiver() {
        @Override
        在 onReceive()中加入接收廣播後要執行的動作
        public void onReceive(Context context, Intent intent) {

            Bundle b = intent.getExtras();解析 Intent 取得秒數資訊
            tv_clock.setText(String.format("%02d:%02d:%02d",
                    b.getInt( key: "H"), b.getInt( key: "M"), b.getInt( key: "S")));
        }
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        tv_clock = findViewById(R.id.tv_clock);
        btn_start = findViewById(R.id.btn_start);

        registerReceiver(receiver, new
                IntentFilter( action: "MyMessage"));   註冊 Receiver

        flag = MyService.flag;取得 Service 狀態
        if(flag)
            btn_start.setText("暫停");
        else
            btn_start.setText("開始");
        btn_start.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                flag = !flag;
                if(flag) {
                    btn_start.setText("暫停");
                    Toast.makeText( context: MainActivity.this,  text: "計時開始",
                            Toast.LENGTH_SHORT).show();
                }else {
                    btn_start.setText("開始");
                    Toast.makeText( context: MainActivity.this,  text: "計時暫停",
                            Toast.LENGTH_SHORT).show();
                }
                啟動 Service
                startService(new Intent( packageContext: MainActivity.this,
                        MyService.class).putExtra( name: "flag", flag));
            }
        });
    }
    public void onDestroy(){
        super.onDestroy();

        unregisterReceiver(receiver);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

```
public class MyService extends Service {

    static Boolean flag = false;計數器狀態

    private int h=0, m=0, s=0; 計數器數值

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        throw new UnsupportedOperationException("Not yet implemented");
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startID){
        flag = intent.getBooleanExtra( name: "flag",  defaultValue: false);
使用 Thread 來計算秒數
        new Thread(new Runnable() {
            @Override
            public void run() {
                while(flag){
                    try {
                        使用 Thread 來計算秒數，延遲 1 秒
                        Thread.sleep( millis: 1000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }

                    s++;  計數器+1
                    if (s >= 60) {
                        s = 0;
                        m++;
                        if (m >= 60) {
                            m = 0;
                            h++;
                        }
                    }
```


Intent intent = new Intent(action: "MyMessage");
產生帶 MyMessage 識別字串的 Intent
Bundle bundle = new Bundle();
bundle.putInt("H", h);
bundle.putInt("M", m);
bundle.putInt("S", s);
intent.putExtras(bundle);用 Bundle 打包後放入 Intent
sendBroadcast(intent); 發送廣播
}
}
}).start();
return Service.START_STICKY;
}
}