第十六章

BROADCAST

BROADCAST RECEIVER簡介

BROADCAST RECEIVER

- Broadcast Receiver讓使用者可以接收到系統或是app發出的特定狀態
  - e.g. 系統開啟、網路改變、螢幕暗亮
  - e.g. App自訂的action

- 與啟動Activity類似，啟動Broadcast Receiver和接收Broadcast都是經由Intent在處理

BROADCAST

- Broadcast繼承BroadcastReceiver

- Broadcast Receiver使用分為靜態和動態兩種
  - 靜態：直接在AndroidManifest.xml中定義
    沒有特別設定，當App安裝完畢時就會被註冊在系統中
  - 動態：在Activity或Service建立時定義
    當App或Service結束時，該Broadcast Receiver就結束

Broadcast/BroadcastByDeclaration

靜態註冊BROADCAST RECEIVER

靜態註冊BROADCAST RECEIVER

- 靜態註冊是在AndroidManifest.xml中註冊

- 寫法與Activity的註冊非常類似

靜態註冊BROADCAST RECEIVER

```
<application …>
  ……
  <receiver
    android:name=".Receiver"
    android:exported="true">
    <intent-filter>
       <action android:name="com.example.broadcast.MAIN"/>
       <action android:name="android.intent.action.USER_PRESENT"/>
       <action android:name="android.intent.action.BOOT_COMPLETED"/>
       <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
     </intent-filter>
  </receiver>
</application>
```

- <receiver …> Broadcast Receiver要註冊的宣告
- android:name 對應到Broadcast Receiver的class名稱
- android:exported 傳入true表示此Receiver可以收到外部的Intent，一般都是設定為true

靜態註冊BROADCAST RECEIVER

```
<application …>
  ……
  <receiver
    android:name=".Receiver"
    android:exported="true">
    <intent-filter>
       <action android:name="com.example.broadcast.MAIN"/>
       <action android:name="android.intent.action.USER_PRESENT"/>
       <action android:name="android.intent.action.BOOT_COMPLETED"/>
       <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
     </intent-filter>
  </receiver>
</application>
```

- <intent-filter> 內包含的就是Broadcast Receiver要接收action符合定義的Intent

靜態註冊BROADCAST RECEIVER

```
<application …>
  ……
  <receiver
    android:name=".Receiver"
    android:exported="true">
    <intent-filter>
       <action android:name="com.example.broadcast.MAIN"/>
       <action android:name="android.intent.action.USER_PRESENT"/>
       <action android:name="android.intent.action.BOOT_COMPLETED"/>
       <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
     </intent-filter>
  </receiver>
</application>
```

- 第一個註冊要接收的action為自訂的Action
- 第二個註冊要接收的action為Lock Screen解開時，由系統發出的通知

靜態註冊BROADCAST RECEIVER

```
<application …>
  ……
  <receiver
    android:name=".Receiver"
    android:exported="true">
    <intent-filter>
       <action android:name="com.example.broadcast.MAIN"/>
       <action android:name="android.intent.action.USER_PRESENT"/>
       <action android:name="android.intent.action.BOOT_COMPLETED"/>
       <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
     </intent-filter>
  </receiver>
</application>
```

- 第三個註冊要接收的action為開機成功由系統發出的通知
- 第四個註冊要接收的action為網路狀態改變時系統發出的通知

靜態註冊BROADCAST RECEIVER

```
<uses-permission
android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

- 因為BOOT_COMPLETED和CONNECTIVITY_CHANGE兩種系統發出的通知比較特別
  依照Androrid規定需要加上上述兩個permission才可以使用

靜態註冊BROADCAST RECEIVER

- AndroidManifest.xml中靜態註冊完畢後，接下來要建立Receiver的Class

Broadcast/BroadcastByDeclaration

建立BROADCAST RECEIVER

建立BROADCAST RECEIVER

- 建立的BroadcastReceiver class名稱要與AndroidManifest.xml中定義的相同

- 依照下列程式碼，可以得知class名稱為Receiver

```
<receiver
    android:name=".Receiver"
    android:exported="true">
```

建立BROADCAST RECEIVER

```
public class Receiver extends BroadcastReceiver {

      @Override
      public void onReceive(Context c, Intent intent){
            ……
      }
}
```

- 建立Broadcast Receiver就是class必須得繼承BroadcastReceiver

建立BROADCAST

```
public class Receiver extends BroadcastReceiver {

	@Override
	public void onReceive(Context c, Intent intent){
		……
	}
}
```

- 當有符合Broadcast Receiver條件的Intent被系統發出時，onReceive()就會接到通知

建立BROADCAST

```
public void onReceive(Context c, Intent intent) {
        String action = intent.getAction();
        Log.i("Receiver", action);
        if ("com.example.broadcast.MAIN".equals(action)) {
                String text = intent.getStringExtra("text");
                Toast.makeText(c, text, Toast.LENGTH_LONG).show();
        } else {
                showNotification(c, action);
        }
}
```

- 在onReceive()收到的Intent，可以藉由intent.getAction()得到傳入Intent的Action

Broadcast/BroadcastByDeclaration

發送RECEIVER的INTENT

發送RECEIVER的INTENT

- 開發者可以自訂Broadcast Receiver要接收的Intent action
  - 程式發出Broadcast通知自己建立的Receiver做事

- 本例子是在某個按鈕按下後，發出自訂的Broadcast Intent

發送RECEIVER的INTENT

```
Button button = (Button) findViewById(R.id.button1);
button.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent("com.example.broadcast.MAIN");
            intent.putExtra("text", "Received!!");
            sendBroadcast(intent);
        }
});
```

- 首先建立Intent，建構子帶入自訂的Intent Action
- 如果有多餘的資訊要帶入到Broadcast Receiver，可以使用intent.putExtra()

發送RECEIVER的INTENT

```
Button button = (Button) findViewById(R.id.button1);
button.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent("com.example.broadcast.MAIN");
            intent.putExtra("text", "Received!!");
            sendBroadcast(intent);
        }
});
```

- 使用sendBroadcast()就可以送出建立的Intent，若有符合條件的Receiver就會收到發出的Intent
  - 參數：設定要送出的Intent

Broadcast/BroadcastByProgramming

動態註冊BROADCAST RECEIVER

動態註冊BROADCAST RECEIVER

- 動態註冊就不用在AndroidManifest.xml中定義
- 動態註冊是當需要使用Receiver時再註冊
  - 註冊的方法為registerReceiver()

- 要注意，**不使用時要取消註冊**
  - 取消註冊的方法為unregisterReceiver()

- 多半會在Activity或Service的onCreate()註冊 onDestroy()取消註冊

動態註冊BROADCAST RECEIVER

```
Receiver mReceiver;
mReceiver = new Receiver();
IntentFilter filter = new IntentFilter();
filter.addAction("com.example.broadcast.MAIN");
filter.addAction(Intent.ACTION_USER_PRESENT);
filter.addAction(Intent.ACTION_SCREEN_ON);
filter.addAction(Intent.ACTION_SCREEN_OFF);
filter.addAction(Intent.ACTION_BOOT_COMPLETED);
filter.addAction(ConnectivityManager.CONNECTIVITY_ACTION);
registerReceiver(mReceiver, filter);
```

- Receiver宣告為成員變數，因為不使用時要取消註冊
- 使用new的方式來建立Receiver

動態註冊BROADCAST RECEIVER

```
Receiver mReceiver;
mReceiver = new Receiver();
IntentFilter filter = new IntentFilter();
filter.addAction("com.example.broadcast.MAIN");
filter.addAction(Intent.ACTION_USER_PRESENT);
filter.addAction(Intent.ACTION_SCREEN_ON);
filter.addAction(Intent.ACTION_SCREEN_OFF);
filter.addAction(Intent.ACTION_BOOT_COMPLETED);
filter.addAction(ConnectivityManager.CONNECTIVITY_ACTION);
registerReceiver(mReceiver, filter);
```

- 建立IntentFilter，要註冊的Action必須得使用Intent Filter來註冊

動態註冊BROADCAST RECEIVER

```
Receiver mReceiver;
mReceiver = new Receiver();
IntentFilter filter = new IntentFilter();
filter.addAction("com.example.broadcast.MAIN");
filter.addAction(Intent.ACTION_USER_PRESENT);
filter.addAction(Intent.ACTION_SCREEN_ON);
filter.addAction(Intent.ACTION_SCREEN_OFF);
filter.addAction(Intent.ACTION_BOOT_COMPLETED);
filter.addAction(ConnectivityManager.CONNECTIVITY_ACTION);
registerReceiver(mReceiver, filter);
```

- filter.addAction()用來註冊要用的action
- 一般系統的Action都會放在Intent.ACTION_XXX

動態註冊BROADCAST RECEIVER

```
Receiver mReceiver;
mReceiver = new Receiver();
IntentFilter filter = new IntentFilter();
filter.addAction("com.example.broadcast.MAIN");
filter.addAction(Intent.ACTION_USER_PRESENT);
filter.addAction(Intent.ACTION_SCREEN_ON);
filter.addAction(Intent.ACTION_SCREEN_OFF);
filter.addAction(Intent.ACTION_BOOT_COMPLETED);
filter.addAction(ConnectivityManager.CONNECTIVITY_ACTION);
registerReceiver(mReceiver, filter);
```

- 呼叫Activity的方法registerReceiver()
  - 參數1：擺放Receiver
  - 參數2：Receiver註冊用的IntentFilter

取消註冊BROADCAST RECEIVER

```
unregisterReceiver(mReceiver);
```

- 呼叫unregisterReceiver()已取消Receiver的註冊
  - 參數：要取消註冊的Receiver

Broadcast/BroadcastByProgramming

RECEIVER中啟動ACTIVITY

RECEIVER中啟動ACTIVITY

- Broadcast Receiver時常搭配系統的服務或是自己開發的Service運作

- 因為當系統背景服務做完某件特定的事情發出通知時，app不見得是開啟的，Activity無法收到通知
- 藉由Receiver收到系統的通知後，開啟某個Activity或Service

- 本例子就是收到BOOT_COMPLETED後，開啟app的Activity

RECEIVER中啟動ACTIVITY

```
public void onReceive(Context c, Intent intent) {
      String action = intent.getAction();
      showNotification(c, action);
      Intent myIntent = new Intent(c, MainActivity.class);
      myIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
      c.startActivity(myIntent );
}
```

- 首先建立起動Activity的Intent
- 接下來使用addFlag()加入Intent啟動Activity的條件，這邊非常重要，請用FLAG_ACTIVITY_NEW_TASK
- 最後呼叫startActivity()

更多資料

更多資料

- http://developer.android.com/reference/android/content/BroadcastReceiver.html
- http://developer.android.com/guide/components/fundamentals.html
- http://android-developers.blogspot.tw/2011/01/processing-ordered-broadcasts.html