

# Lab4 Activity

## 一、 本節目的：

- 理解什麼是 Activity，並產生一個 Activity
- 如何透過 Intent 切換 Activity
- 理解透過 Bundle 攜帶資料
- 理解透過 onActivityResult() 方法返回資料

## 二、 觀念說明

Android 應用程式元件包含 Activity、Service、BroadcastReceiver、Content Providers 這類的 Java 程式，而 Activity 是最重要的應用程式元件，負責提供應用程式顯示畫面上的相關工作，大部分的 APP 所顯示的畫面都是寫在 Activity 之上，不論是列表、圖片或是地圖的畫面，都是基於 Activity 來呈現。

如以下的 APP 畫面，就是基於 Activity 實現出來的。

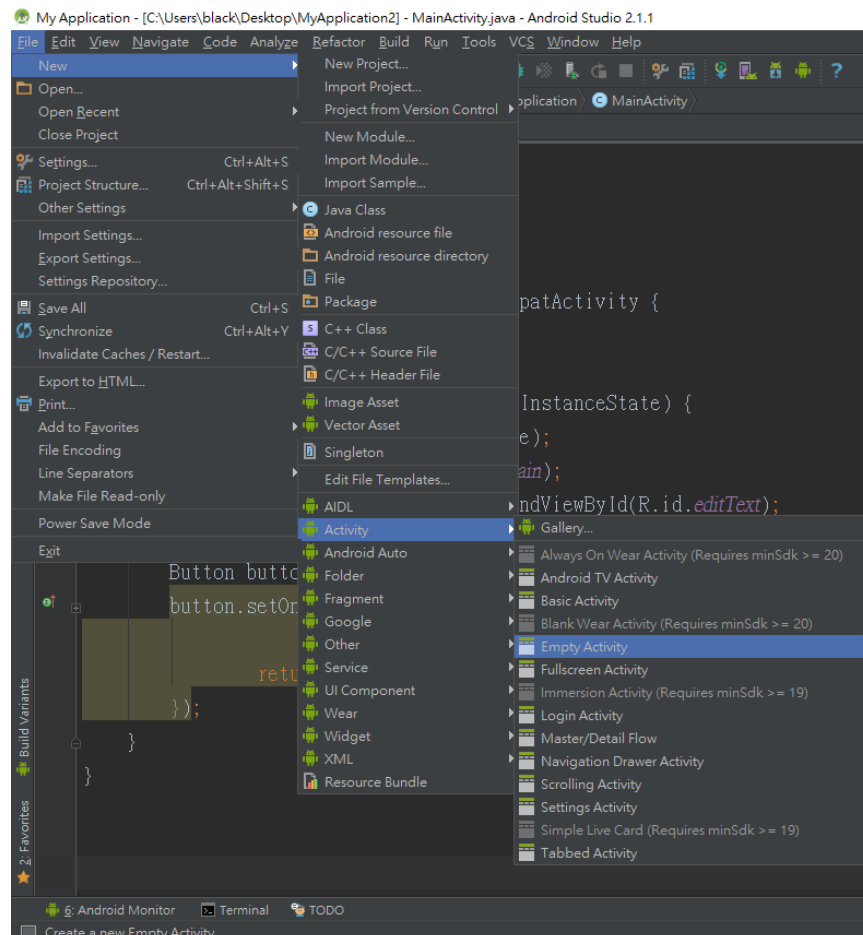


要在螢幕上顯示畫面需要透過畫面配置元件(\*.xml)與產生控制的應用程式元件(\*.java)。前面我們學到畫面配置元件即為 Layout，用於決定了每個元件的擺放位置，而 Activity 賦予這個畫面配置能與使用者互動的功能。

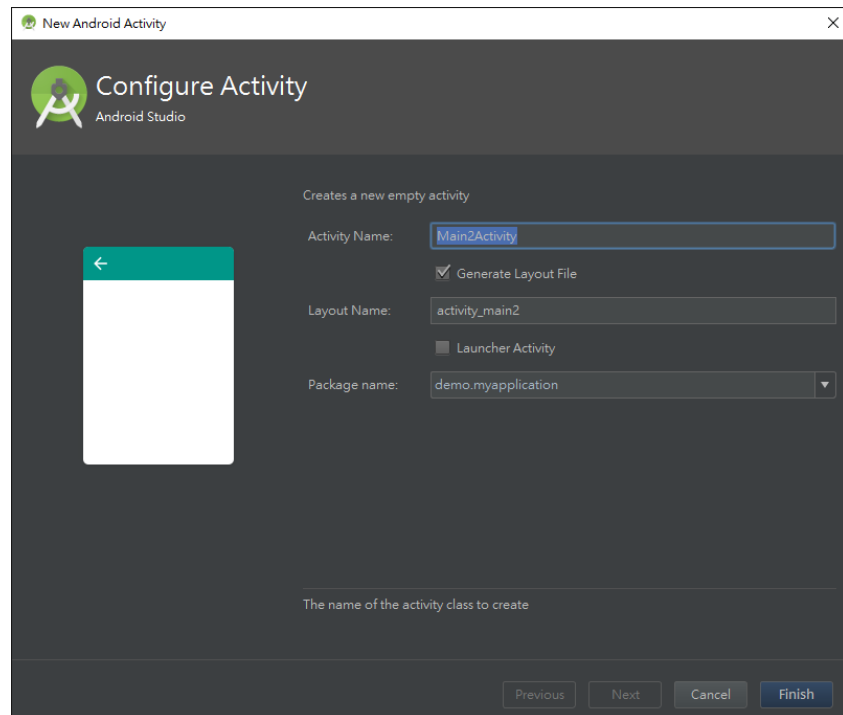
我們能透過 Activity 來顯示出某些資訊(圖片、文字或是地圖)給使用者，或是將使用者的操作傳送給程式來做控制(監聽器)，因此 Activity 扮演著 Android 使用者介面的角色。

# 1 產生 Activity

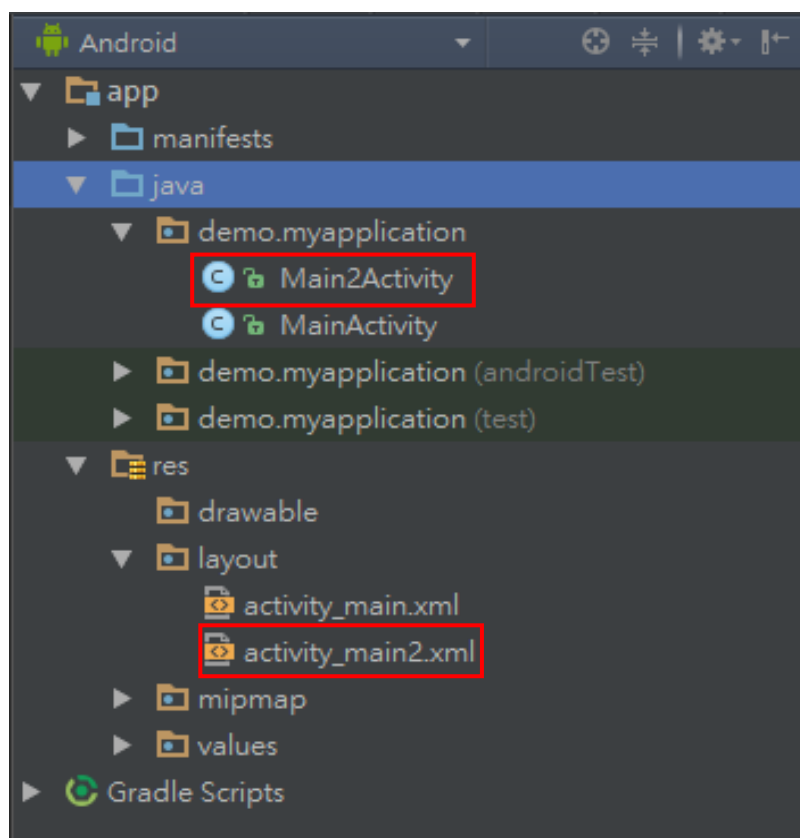
要產生出一個新的 Activity，首先先選擇 File/New/Activity/Empty Activity 來產生出空白的 Activity。



選擇後可於下面的視窗中修改 Activity 的名稱與對應 Layout 的名稱，如果只有更改 Activity 的名稱，Android 會自動幫你修改 Layout 的名稱。



按下 finish 後，可以於左邊目錄中看到系統幫你產生出 Main2Activity.java 以及 activity\_main2.xml。



而 AndroidManifest.xml 也會自動增加 Main2Activity 的資訊。

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="demo.myapplication">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        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>
        <activity android:name=".Main2Activity"></activity>
    </application>

</manifest>
```

## 2 使用 Intent 切換 Activity

Android 應用程式元件(Activity、Service、BroadcastReceiver、Content Providers)之間的切換會需要透過「Intent」。Intent 是可用來向另一個應用程式元件(Activity、Service、BroadcastReceiver、Content Providers)要求動作的傳遞物件。最基本的 Intent 用途是來啟動其他的應用程式元件。如果啟動的對象是 Activity，則可以在畫面上顯示一個新的 Activity，我們也可以說這是 Activity 間的切換動作。

Intent 字義上是指「意圖」，以 Activity 切換的目的上來解釋我們可以口語描述成「A 元件意圖啟動 B 元件」下面我們以 MainActivity 切換至 Main2Activity 為例，MainActivity 就表示 A 元件，Main2Activity 就表示 B 元件，兩者透過意圖傳遞把顯示畫面由 MainActivity 改為 Main2Activity。



實際編寫的程式碼如下：

```
Intent intent = new Intent(this, Main2Activity.class);
startActivity(intent);
```

Intent 有兩個參數，第一個參數我們要描述由哪個元件發起這個意圖，如從 MainActivity 發起則要填入 MainActivity 或是 this(表示 MainActivity 本身)，第二個參數則要描述要接受意圖(被啟動)的是哪個元件，對象如果為 Main2Activity 則要描述成 Main2Activity.class。

而要將這個 Intent 的發出，我們需要用到 startActivity()方法來送出 Intent，Main2Activity 獲得通知後便會被啟動，並覆蓋在 MainActivity 之上。

### 3 傳遞資料

Intent 除了可以做到基本的切換之外，Intent 也可夾帶一些資料到接收意圖方去，例如某個使用者在 MainActivity 填寫了一個表單，並希望在 Main2Activity 看到結果。這時我們就要使用 Intent 傳遞資料的方法，以下則是最簡單的傳遞資料語法，主要是描述(1)將 MainActivity 透過 Intent 切換到 Main2Activity，(2)透過 Intent 傳送 123 的資料，從 MainActivity 到 Main2Activity：

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        int value = 123;    想傳遞的資料  
        Intent intent = new Intent(this, Main2Activity.class);  
        intent.putExtra("key", value);    夾帶資料  
        startActivity(intent);  
    }  
}
```

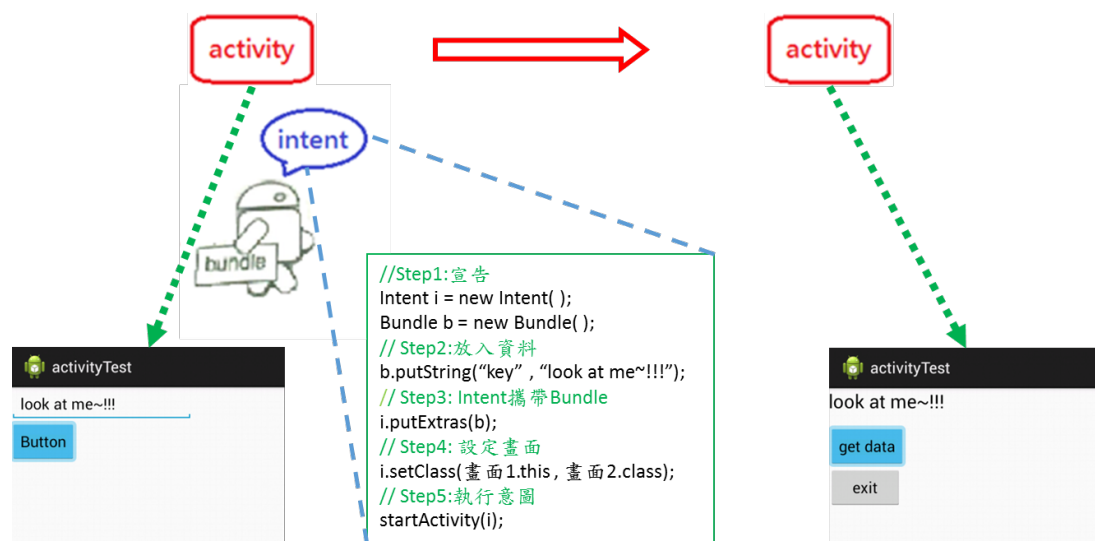
intent.putExtra()可以想像是把想傳遞的資料 (value)貼上一個標籤(key)，接收的對象可以透過標籤去得到他要的資料。

接收到 intent 而被喚起的 Main2Activity 如果要取得傳過來的資料，可以用以下語法：

```
public class Main2Activity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main2);  
        int getData = getIntent().getExtras().getInt("key");  
        得到資料的對象    以 key 找到對應的資料並讀取  
    }  
}
```

getIntent()可以取得從 MainActivity 傳過來的 Intent，而 getExtras()則可以取得底下夾帶的資料，我們可以使用 getIntent(key)方法去找到你傳遞的資料，他的返回值就是 MainActivity 夾帶的資料。此外，由於傳遞的資料是 int 型態，因此使用 getIntent()，如果是 float 型態則要用 getFloat()，以此類推。

然而 intent.putExtra()每一次傳遞資料我們都必須要知道傳遞的資料的型態是甚麼，而且資料只能單獨的傳入、單獨的讀出，這樣資料的完整性並不高，有時候我們希望某些資料能被視為整體一次傳遞，例如一份菜單資料，我們不要每一項餐點資料都單獨傳過去，而是能以訂單為單位傳送。因此我們就會需要用到打包成包裹的概念，而在 android 中這就是 Bundle。



Bundle 可以一次打包不同的資料型別，例如 int 或 String，打包時需要依據型態透過，putInt()或 putString()來儲存資料。

舉個例子，我們希望從 MainActivity 中夾帶一筆整數資料以及一筆字串資料到 Main2Activity 去，我們編寫的程式如下：

```
int value1=123;  
String value2="123";  
  
Bundle bundle = new Bundle();  
bundle.putInt("value1",value1);  
bundle.putString("value2",value2);  
  
Intent intent =new Intent(this,Main2Activity.class);  
intent.putExtra("key",bundle);  
startActivity(intent);
```

想傳遞的多筆資料

把資料打包成 Bundle

存入一筆整數

存入一筆字串

夾帶 Bundle

而新的 Activity 只需要取出 Bundle 就可以還原資料。

```
public class Main2Activity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main2);  
  
        Bundle bundle = getIntent().getExtras().getBundle("key");  
  
        int value1 = bundle.getInt("value1");  
        String value2 = bundle.getString("value2");  
    }  
}
```

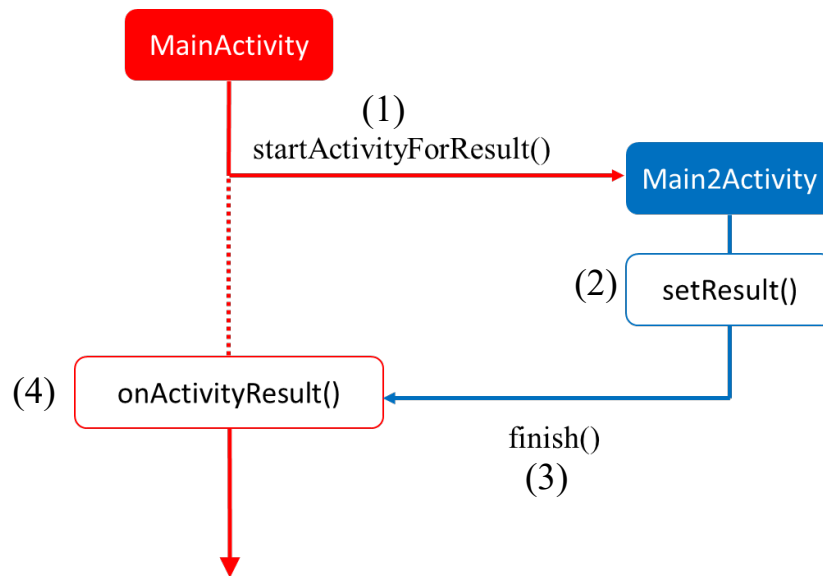
以 key 找到對應的 Bundle 並讀取

打開 Bundle  
取出資料



## 4 返回資料

透過 Intent 方法啟動的 Activity，除了之前介紹的 `startActivity()` 方法之外，某些情況我們希望新的 Activity 在接收到資料後，能再夾帶資料返回到前一個 Activity，實現兩個 Activity 資料往來的目的，這時我們就會使用到 `startActivityForResult()`，實現的步驟流程如下：



- 1) MainActivity 使用 `startActivityForResult()` 方法，啟動 Main2Activity。
- 2) Main2Activity 使用 `setResult()` 方法，儲存要返回的資料。
- 3) Main2Activity 使用 `finish()` 方法結束 Main2Activity，並返回 MainActivity。
- 4) MainActivity 使用 `onActivityResult()` 方法，取得返回資料。

此例中，我們要從 MainActivity 傳送夾帶一筆整數資料以及一筆字串資料到 Main2Activity 去，並且再接收 Main2Activity 回傳的資料。依據上述四個步驟編寫後的程式如下：

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        int value1 = 123;
        String value2 = "123";

        Bundle bundle = new Bundle();
        bundle.putInt("value1", value1);
        bundle.putString("value2", value2);
        1)使用 startActivityForResult()方法，啟動 Main2Activity
        Intent intent = new Intent(this, Main2Activity.class);

        int requestCode = 100; 標記發出意圖的對象是誰
        startActivityForResult(intent, requestCode); 啟動 Main2Activity
    }
    4)使用 onActivityResult()方法，取得返回資料。
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        switch(requestCode){ 驗證發出對象
            case 100:
                if(resultCode==0){ 確認 Main2Activity 執行的狀態
                    Bundle bundle = data.getExtras().getBundle("key"); 取得返回資料
                    int value1 = bundle.getInt("value1");
                    String value2 = bundle.getString("value2");
                }else{
                    //Do something
                }
                break;
            default:
                break;
        }
    }
}
```

- requestCode 的目的就是在於我們向新 Activity 提出要求並得到回覆時，用來判斷發出需求者(某個功能或頁面)為誰，以及要如何應對。requestCode 扮演著一種需求者的編號，讓原 Activity 可以根據這編號來判斷發出需求的對象。
- onActivityResult()會等待新的 Activity 返回結果，我們可以根據傳過去的 requestCode 去驗證發出對象，之後透過 resultCode 確認在 Main2Activity 中執行的情況如何。

```

public class Main2Activity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        Bundle bundle = getIntent().getExtras().getBundle("key");

        int value1 = bundle.getInt("value1");
        String value2 = bundle.getString("value2");

        int resultCode=0; 標記狀態
        Intent intent=new Intent();
        bundle.putString("key", "value");
        intent.putExtras(bundle);
        setResult(resultCode,intent); 2)使用 setResult()方法。儲存要返回的資料
        finish(); 3)使用 finish()方法結束 Main2Activity，並返回 MainActivity 結束 Activity
    }
}

```

- resultCode 可以用於回報執行結果給 MainActivity，例如成功時我們可以考慮用 resultCode=0 來表示，失敗時 resultCode=-1，用這方法來告知原 Activity 該如何處理。

### 三、 設計重點:

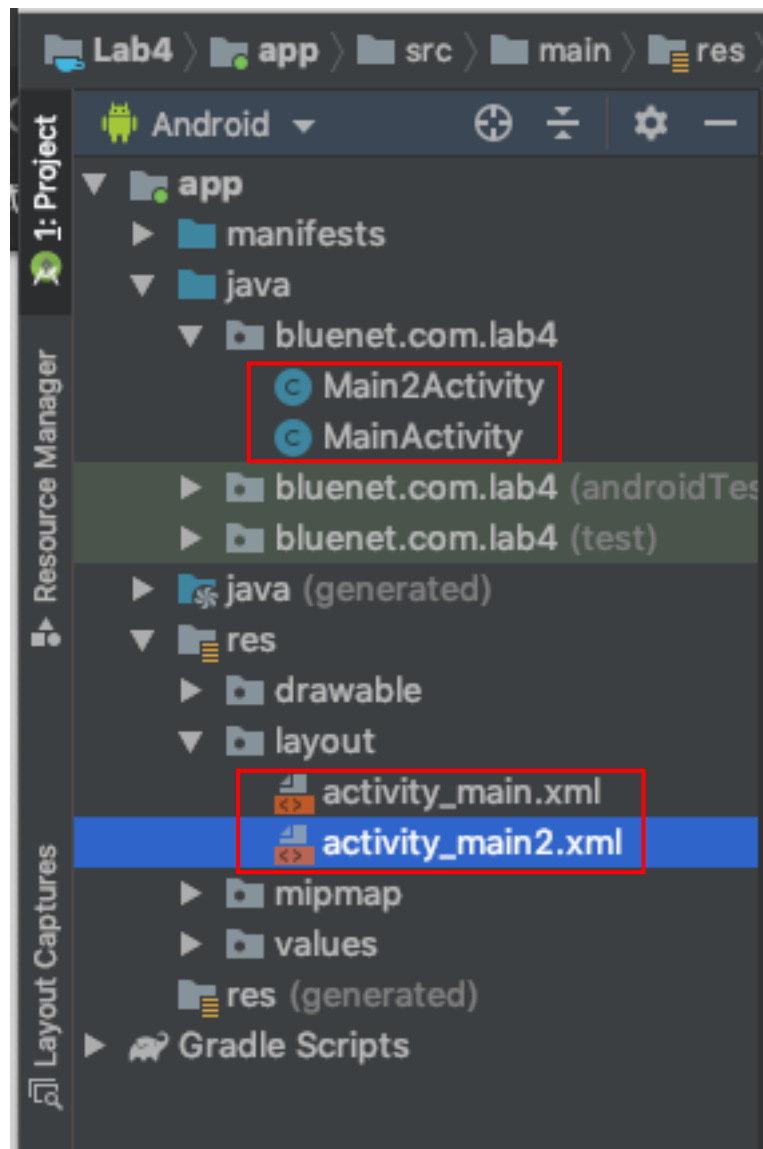
- 本次範例實作一個點餐系統，透過設計兩個不同的 Activity，分別帶有不同的布局來完成點餐作業
- Activity aty1 按下選擇按鈕後會切換到 Activity aty2
- Activity aty2 中可以輸入飲料與選擇甜度、冰塊，設定完點餐資訊後再回傳 aty1 點餐資訊
- Activity aty2 接收點餐資訊資訊後可以顯示訂單資訊



#### 四、設計步驟:

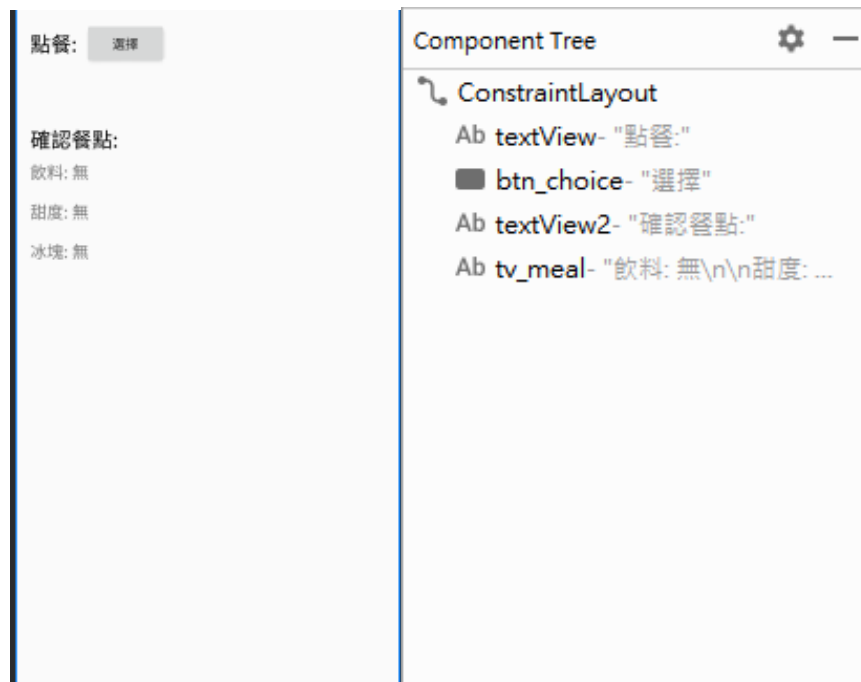
##### Step1

新建專案，建立 MainActivity，以及 Main2Activity



## Step2

繪製 MainActivity layout



對應的 xml 如下：

```
<?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/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="16dp"
        android:text="點餐:"
        android:textSize="22sp"
        android:textColor="@android:color/black"
        app:layout_constraintBottom_toBottomOf="@+id/btn_choice"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="@+id/btn_choice" />

    <Button
        android:id="@+id/btn_choice"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:text="選擇"
        app:layout_constraintStart_toEndOf="@+id/textView"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

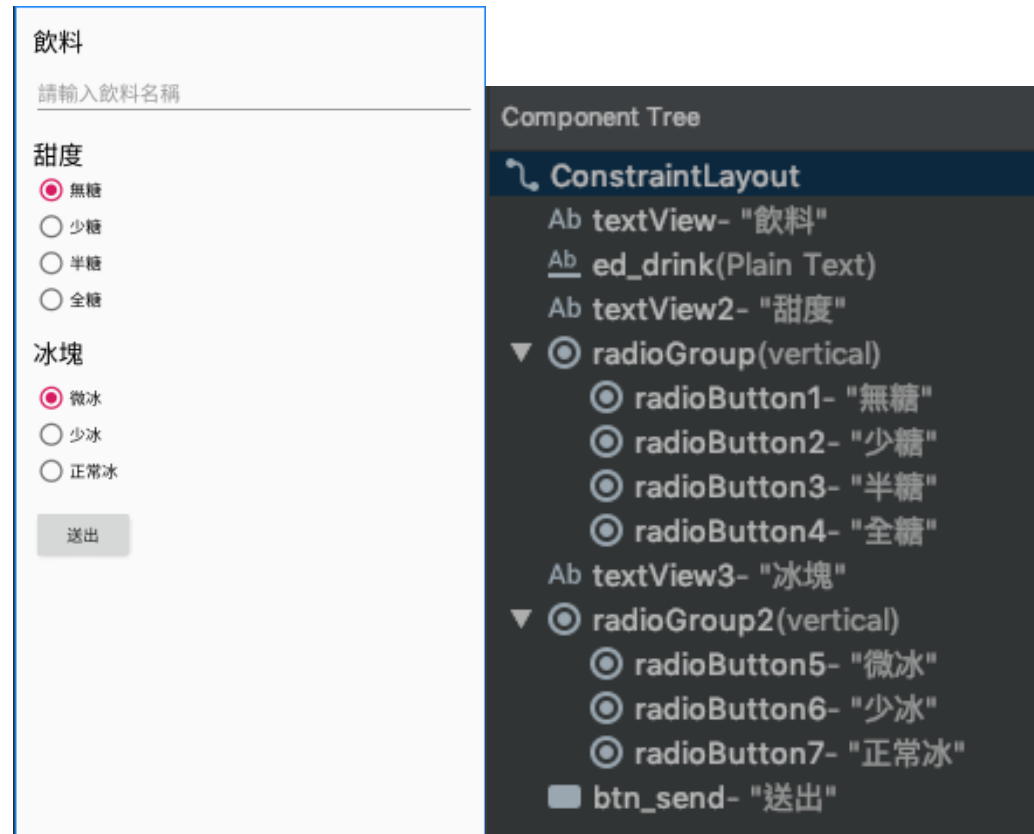
```
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="64dp"
    android:text="確認餐點:"
    android:textSize="22sp"
    android:textColor="@android:color/black"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/btn_choice" />
```

```
<TextView
    android:id="@+id/tv_meal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:text="飲料：無\n\n甜度：無\n\n冰塊：無"
    android:textSize="18sp"
    app:layout_constraintStart_toStartOf="@+id/textView2"
    app:layout_constraintTop_toBottomOf="@+id/textView2" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

### Step3

繪製 Main2Activity layout





對應的 xml 如下：

```
<?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=".Main2Activity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:text="飲料"
        android:textSize="22sp"
        android:textColor="@android:color/black"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/ed_drink"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="請輸入飲料名稱"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="@+id/textView"
        app:layout_constraintTop_toBottomOf="@+id/textView" />
```

#### <TextView

```
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="甜度"
    android:textSize="22sp"
    android:textColor="@android:color/black"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/ed_drink" />
```

#### <RadioGroup

```
    android:id="@+id/radioGroup"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/textView2">
```

#### <RadioButton

```
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:checked="true"
        android:text="無糖" />
```

#### <RadioButton

```
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="少糖" />
```

#### <RadioButton

```
        android:id="@+id/radioButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="半糖" />
```

#### <RadioButton

```
        android:id="@+id/radioButton4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:checked="false"
        android:text="全糖" />
```

</RadioGroup>

```
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="冰块"
    android:textSize="22sp"
    android:textColor="@android:color/black"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup" />
```

```
<RadioGroup
    android:id="@+id/radioGroup2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/textView3">
```

```
    <RadioButton
        android:id="@+id/radioButton5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:checked="true"
        android:text="微冰" />
```

```
    <RadioButton
        android:id="@+id/radioButton6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="少冰" />
```

```
    <RadioButton
        android:id="@+id/radioButton7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="正常冰" />
```

```
</RadioGroup>
```

```
<Button
    android:id="@+id/btn_send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="送出"
    app:layout_constraintStart_toStartOf="@+id/textView"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup2" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

## Step4

- 1) 撰寫 MainActivity 程式，按下按鈕後切換至 Main2Activity

```
public class MainActivity extends AppCompatActivity {
    //宣告元件
    private Button btn;
    private TextView tv_meal;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);    初始化 Activity
        setContentView(R.layout.activity_main);    連接 main1.xml 畫面
        //連結畫面元件
        tv_meal = findViewById(R.id.tv_meal);    連接 TextView 元件
        //『選擇』按鈕點擊監聽
        btn = findViewById(R.id.btn_choice);    連接 Button 元件
        btn.setOnClickListener(new View.OnClickListener() {    Button 點擊事件
            @Override
            public void onClick(View v) {
                //透過startActivityForResult發出Intent，並紀錄請求對象
                startActivityForResult(new Intent( packageContext: MainActivity.this,
                                                    Main2Activity.class),    requestCode: 1);
            }
        });    透過 Intent 切換至 aty2 並傳遞
    }    requestCode=1 來標記發出者
}
```



- 2) 建立 onActivityResult()接收返回資料後，將 data 的內容讀出以 TextView 作顯示

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (data == null) return;
    //驗證發出的對象回傳狀態
    if(requestCode == 1) {    驗證發出對象
        if (resultCode == 101) {    確認 Main2Activity 執行的狀態
            //讀取Bundle中的資料
            Bundle b = data.getExtras();
            String str1 = b.getString( key: "drink");    取得飲料資料
            String str2 = b.getString( key: "sugar");    取得甜度資料
            String str3 = b.getString( key: "ice");    取得冰塊資料
            tv1.setText(String.format("飲料: %s\n\n甜度: %s\n\n冰塊: %s\n\n",
                str1,
                str2,
                str3));    透過 TextView.setText()顯示資料
        }
    }
}
```

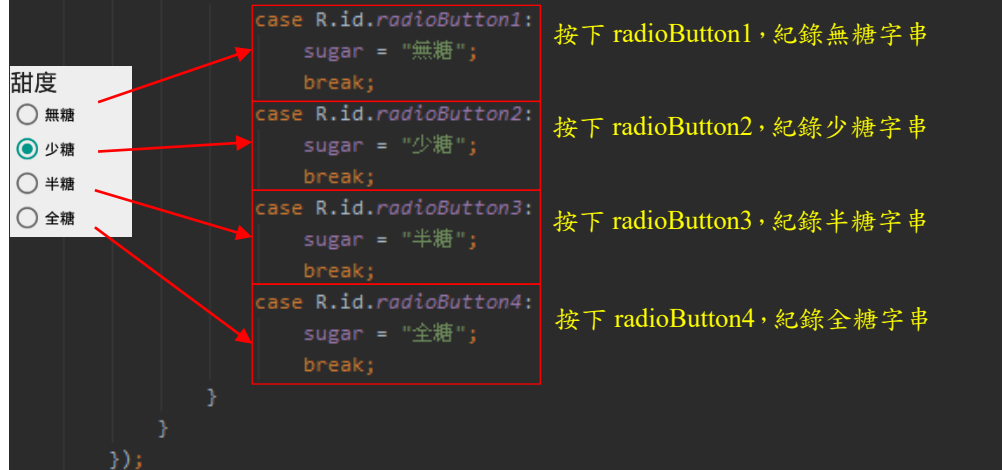


## Step5

撰寫 aty2 程式

- 1) 用 RadioGroup 監聽 RadioButton 的按下事件，並把字串改成甜度資訊

```
public class MainActivity2 extends AppCompatActivity {  
    //宣告元件  
    private Button send_btn;  
    private EditText set_drink;  
    private RadioGroup rg1, rg2;  
  
    private String sugar = "無糖";  
    private String ice_opt = "去冰";  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState); 初始化 Activity  
        setContentView(R.layout.activity_main2); 連接 main2.xml 畫面  
        //連結畫面元件  
  
        rg1 = findViewById(R.id.radioGroup); 連接 RadioGroup 畫面元件  
        rg1.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {  
            @Override  
            public void onCheckedChanged(RadioGroup radioGroup, int i) {  
                switch(i){  
                    case R.id.radioButton1:  
                        sugar = "無糖";  
                        break;  
                    case R.id.radioButton2:  
                        sugar = "少糖";  
                        break;  
                    case R.id.radioButton3:  
                        sugar = "半糖";  
                        break;  
                    case R.id.radioButton4:  
                        sugar = "全糖";  
                        break;  
                }  
            }  
        });  
    }  
}
```



甜度

- ☐ 無糖
- ☒ 少糖
- ☐ 半糖
- ☐ 全糖

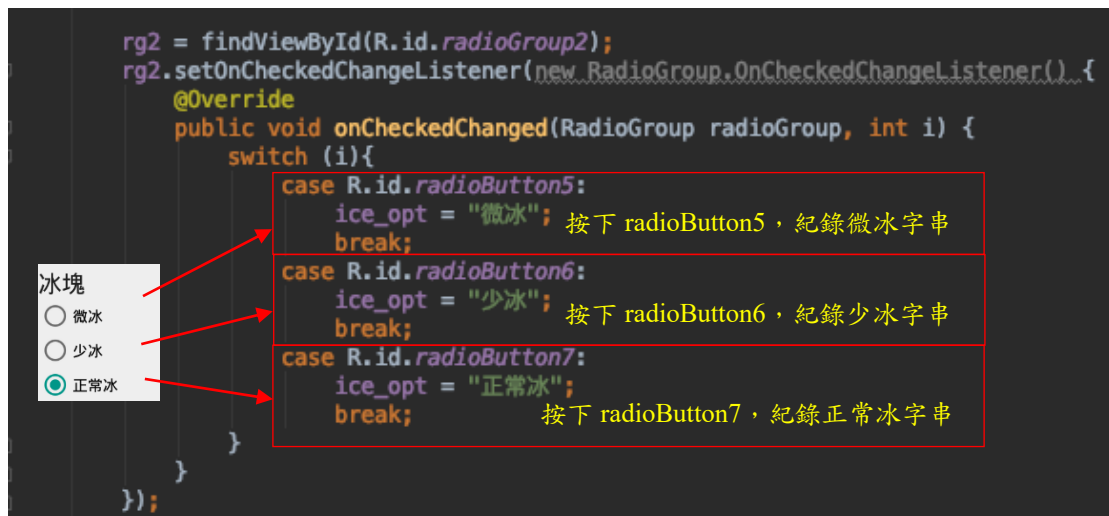
按下 radioButton1, 紀錄無糖字串

按下 radioButton2, 紀錄少糖字串

按下 radioButton3, 紀錄半糖字串

按下 radioButton4, 紀錄全糖字串

- 2) 用 RadioGroup 監聽 RadioButton 的按下事件，並改更改成的字串當作甜度資訊



- 3) 用 Button 監聽按下事件，飲料、甜度、冰塊的資訊回傳到前一頁面

