Android App 程式設計教本之無痛起步

引領入門

最簡單、最易懂的初學教材

第 8 章

用 Intent 啟動程式中的其他 Activity

前言

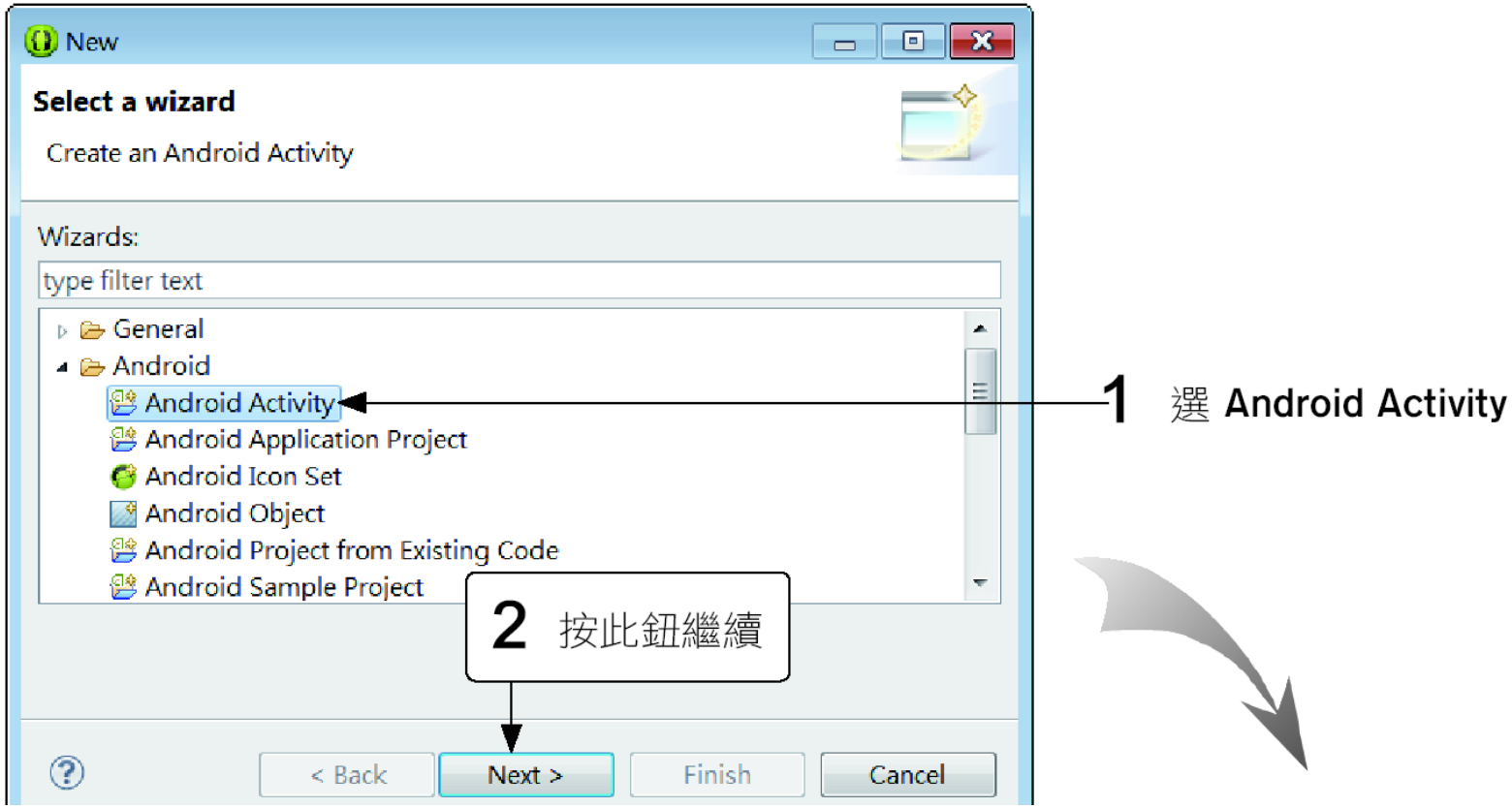
- 8-1 在程式中新增 Activity
- 8-2 用 Intent 啟動程式中的 Activity
- 8-3 在 Intent 中夾帶資料傳給新 Activity
- 8-4 要求新 Activity 傳回資料

8-1 在程式中新增 Activity

- 範例8-1 ：在專案中新增 Activity

step 1

step 2



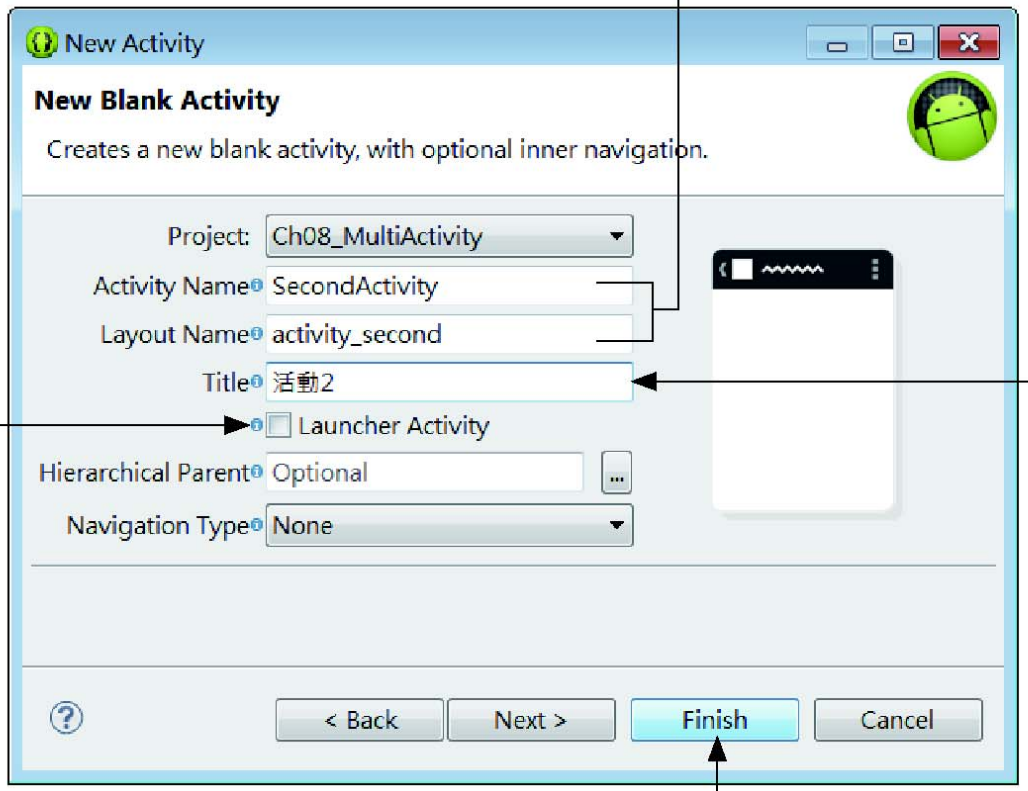
在專案中新增 Activity

在專案中新增 Activity

不要勾選此項，否則這個 Activity 會成為程式的另一個啟動入口

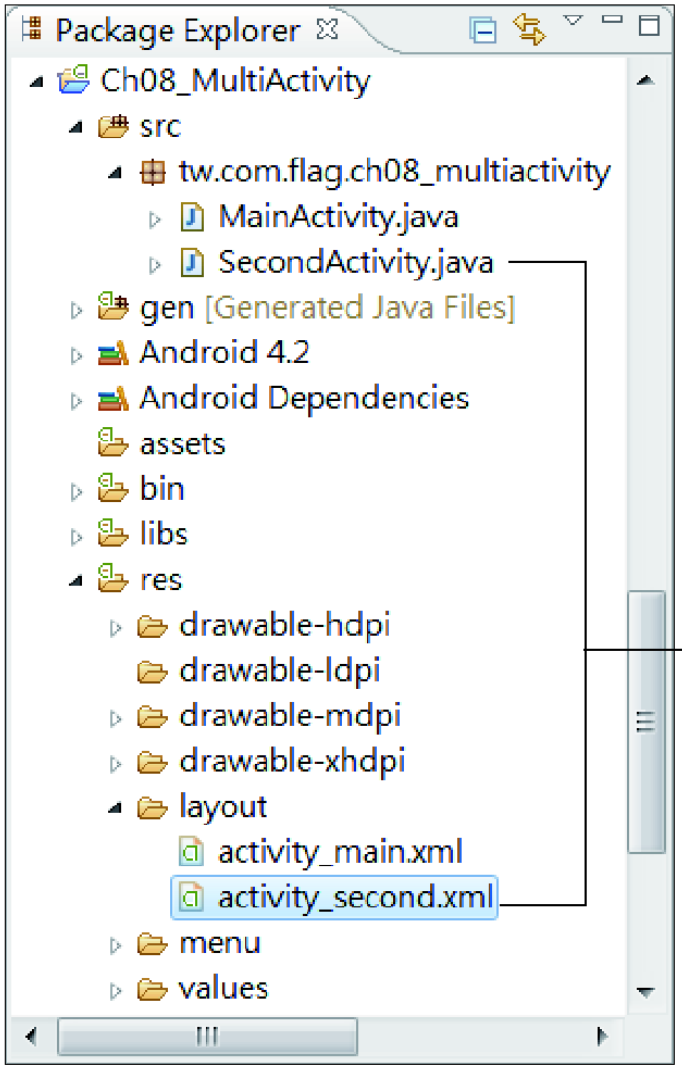
5 輸入新 Activity 的類別名稱，以及佈局檔名稱

6 將標題設為 "活動 2"



7 按此鈕完成設定

在專案中新增 Activity



已加入新 Activity 的類別檔及佈局檔了

8-2 用 Intent 啟動程式中的 Activity

- startActivity()：用明示 Intent 啟動 Activity
- finish()：結束 Activity

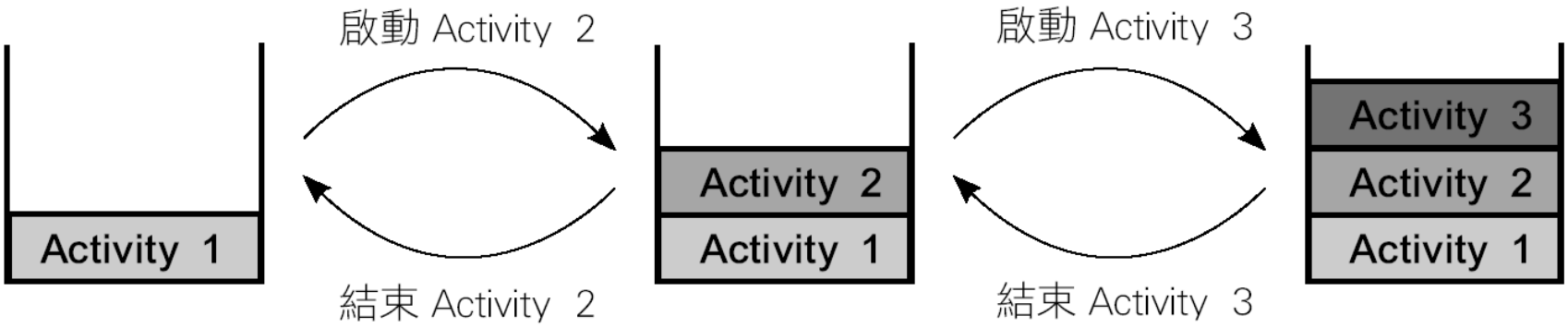
startActivity()：用明示 Intent 啟動 Activity

```
Intent it = new Intent();                ← 建立 Intent 物件
it.setClass(this, Act2.class);           ← 設定要啟動的 Activity 類別
startActivity(it);                       ← 啟動目標 Activity
```

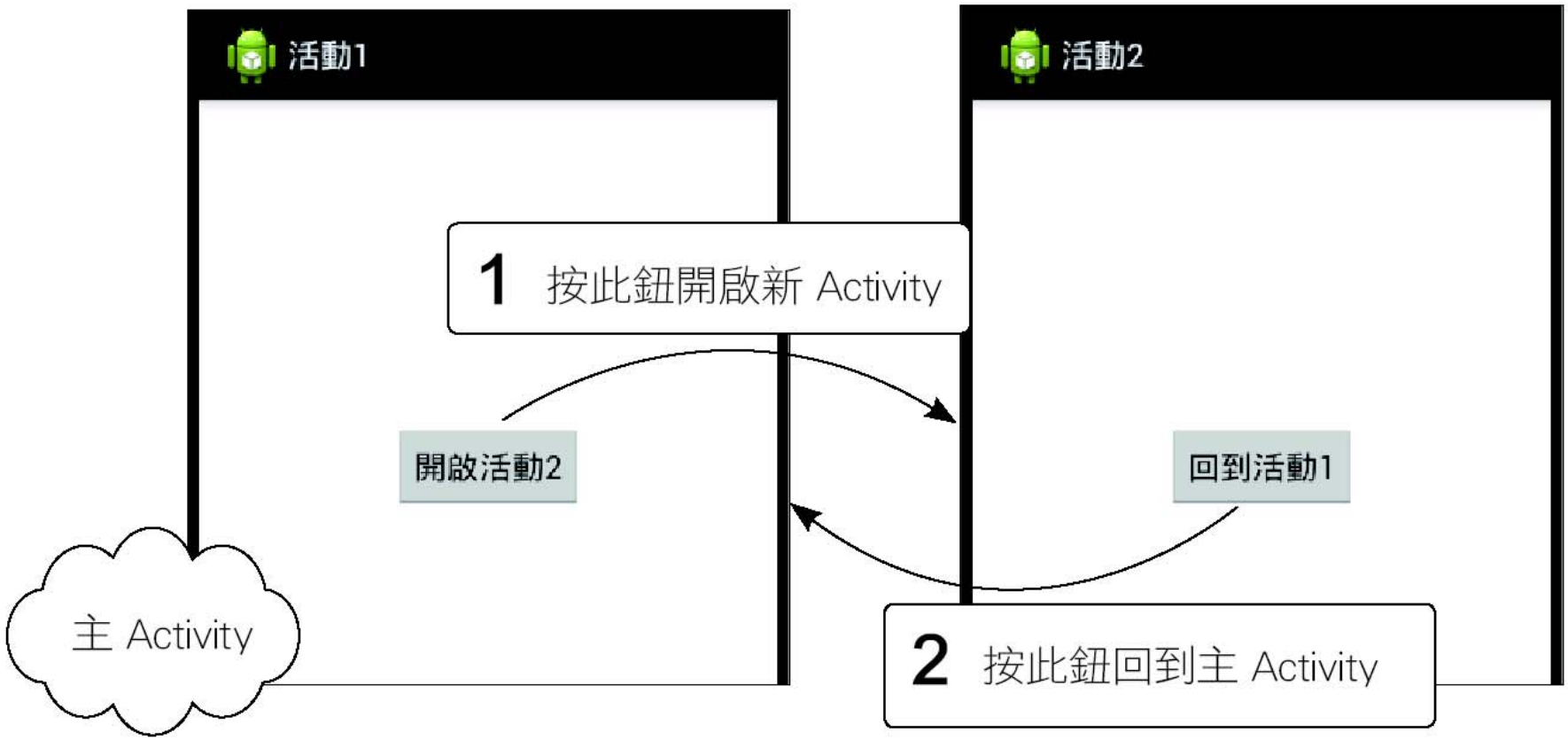
```
Intent it = new Intent(this,  Act2.class);
                                  建立 Intent 物件並設定要啟動的 Activity 類別
startActivity(it);        ← 啟動目標 Activity
```

```
startActivity(new Intent(this, Act2.class));  ← 就地建立 Intent 物件來啟動
```

finish()：結束 Activity



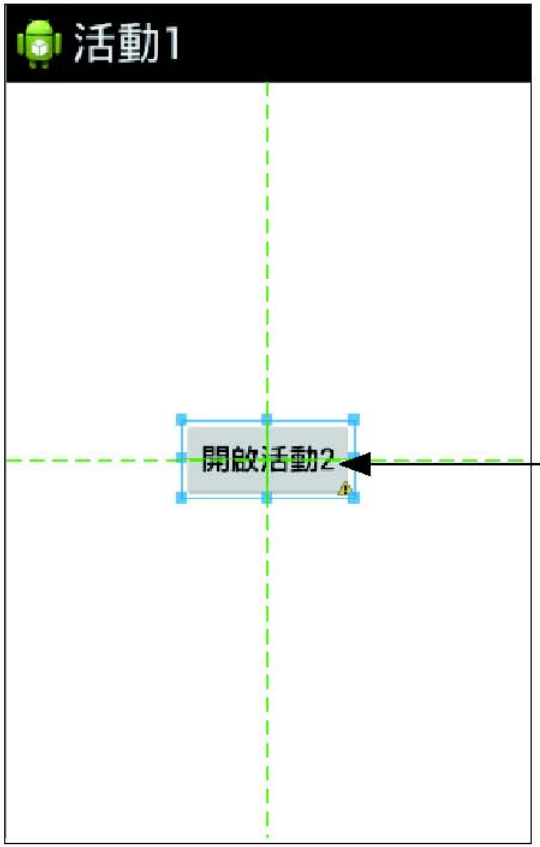
範例8-2 ：用 Intent 來啟動 Activity



用 Intent 來啟動 Activity

step 1

step 2



加入一個 **Button** 元件

Id	@+id/button1
Text	開啟 Activity 2
On Click	gotoSecondActivity

用 Intent 來啟動 Activity

step 3

```
01 public class MainActivity extends Activity {
02
03     @Override
04     protected void onCreate(Bundle savedInstanceState) {
05         super.onCreate(savedInstanceState);
06         setContentView(R.layout.activity_main);
07     }
08
09     public void gotoSecondActivity(View v) {
```

用 Intent 來啟動 Activity

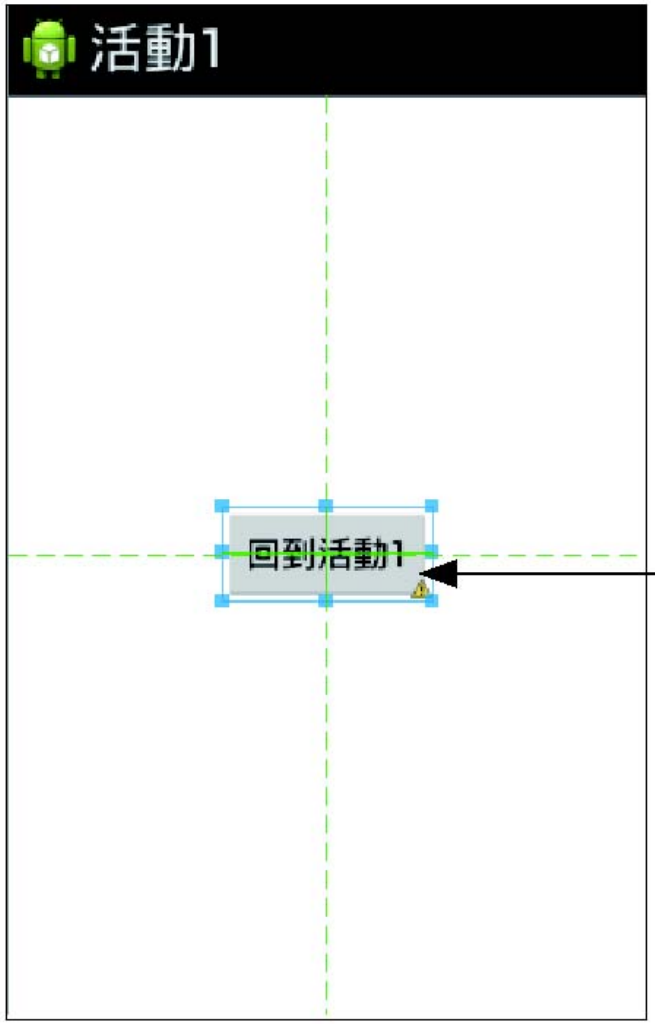
```
10          Intent it = new Intent(this, SecondActivity.class);
11          startActivity(it);
12      }
13
14      @Override
15      public boolean onCreateOptionsMenu(Menu menu) {
16          getMenuInflater().inflate(R.menu.activity_main, menu);
17          return true;
18      }
19  }
```

建立 Intent 並設定目標 Activity

啟動 Intent 中的目標 Activity

用 Intent 來啟動 Activity

step 4



加入一個 **Button** 元件

Id	@+id/button1
Text	回到 Activity 1
On Click	goBack

用 Intent 來啟動 Activity

step 5

```
01      public void goBack(View v) {
02          finish();    ◄── 結束 Activity , 即可回到前一個 Activity
03      }
```

8-3 在 Intent 中夾帶資料傳給新 Activity

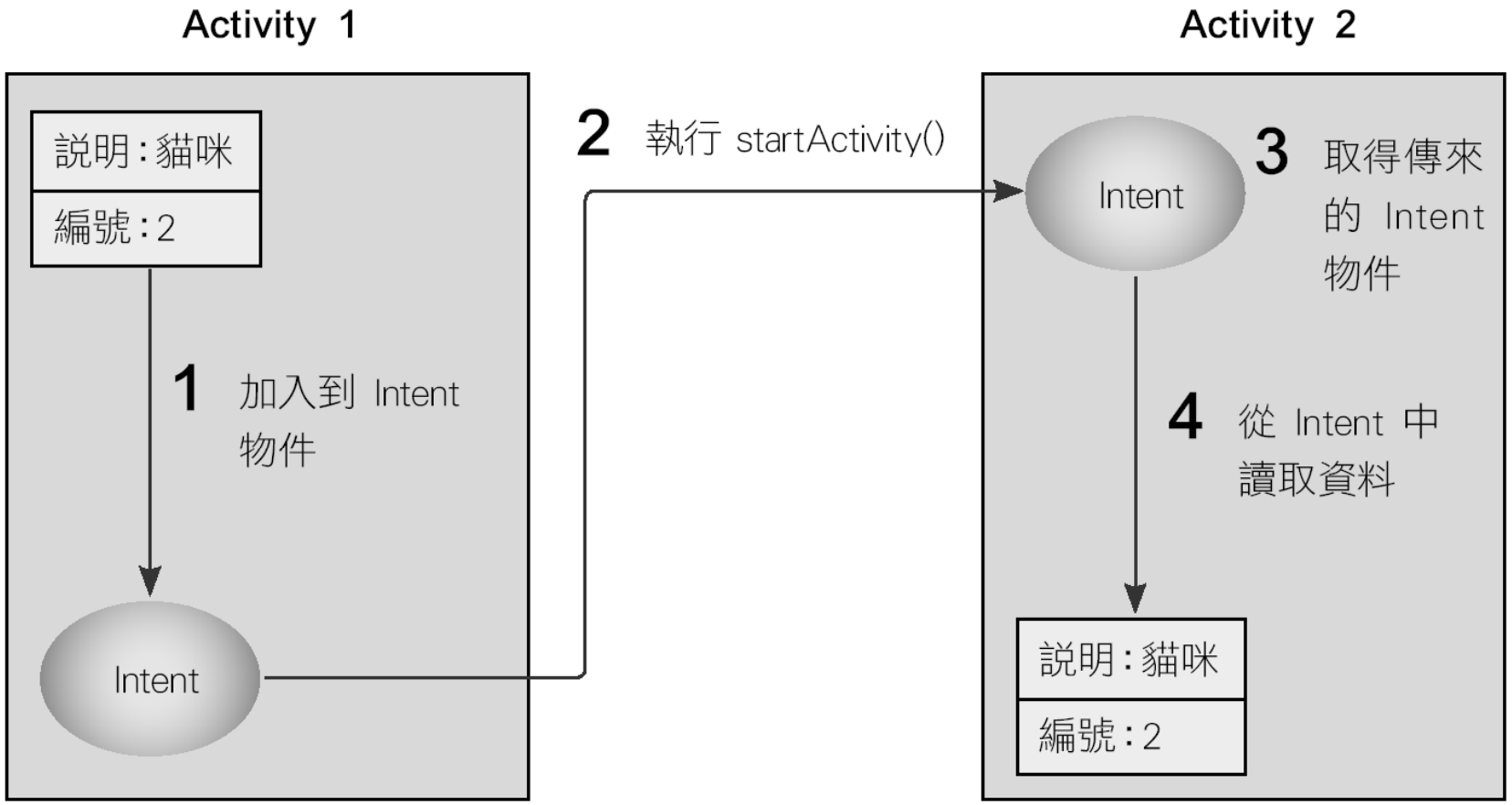


圖 8-1　夾帶資料傳給新 Activity

在 Intent 中夾帶資料傳給新 Activity

- putExtra()：附加資料到 Intent 中
- getIntent() 與 getXxxExtra()：從 Intent 中取出資料

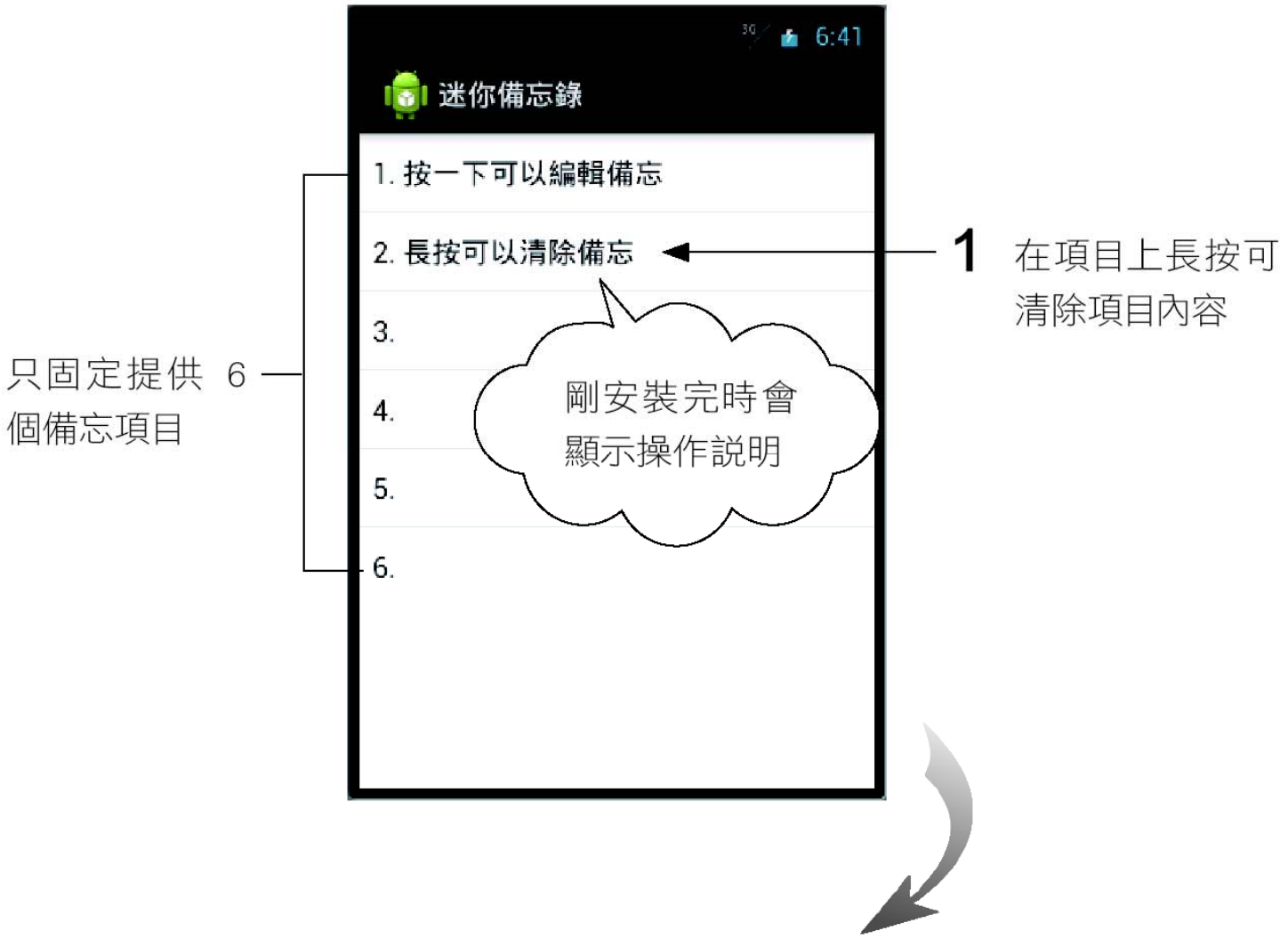
putExtra()：附加資料到 Intent 中

```
String favor[] = { "魚肉","小魚干","貓餅干" };
Intent it = new Intent(this, Act2.class);
it.putExtra("編號", 2);          ←— 入名稱為 "編號" 的 2 (Int 型別)
it.putExtra("說明", "貓咪");     ←— 加入名稱為 "說明" 的 "貓咪" (String 型別)
it.putExtra("愛吃", favor);      ←— 加入名稱為 "愛吃" 的 String 陣列型別
startActivity(it);               ←— 啟動新 Activity Act2
```

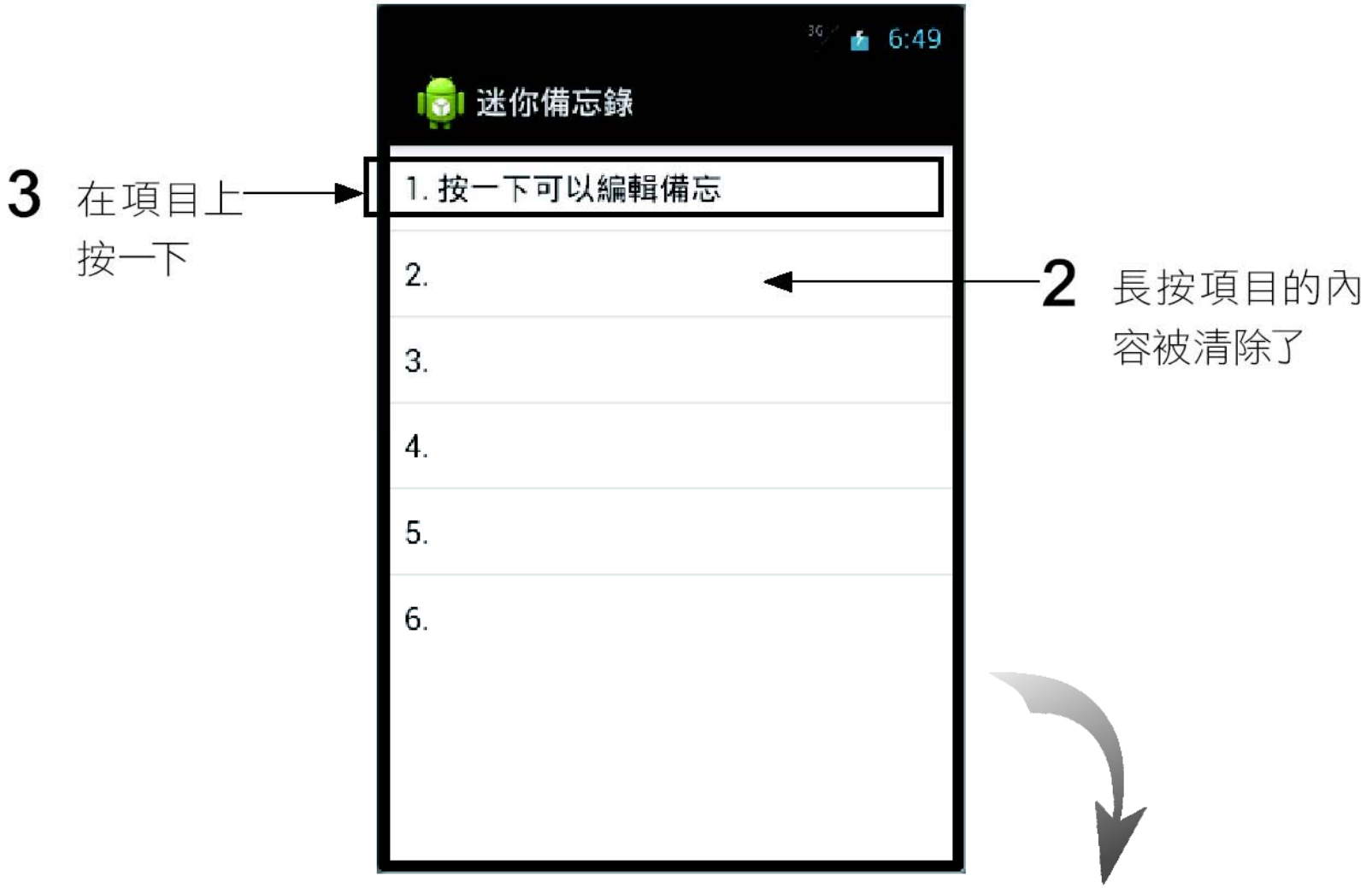
getIntent() 與 getXxxExtra()：從 Intent 中取出資料

```
Intent it = getIntent();                    ← 取得傳入的 Intent 物件
int no = it.getIntExtra("編號", 0);         ← 讀出名為 "編號" 的 Int 資料,
                                              若沒有則傳回 0
String da = it.getStringExtra("說明");      ←┐
                    讀出名為 "說明" 的 String 資料, 若沒有則傳回 null
String a[] = it.getStringArrayExtra("愛吃");←┐
                    讀出名為 "愛吃" 的 String[] 資料, 若沒有則傳回 null
```

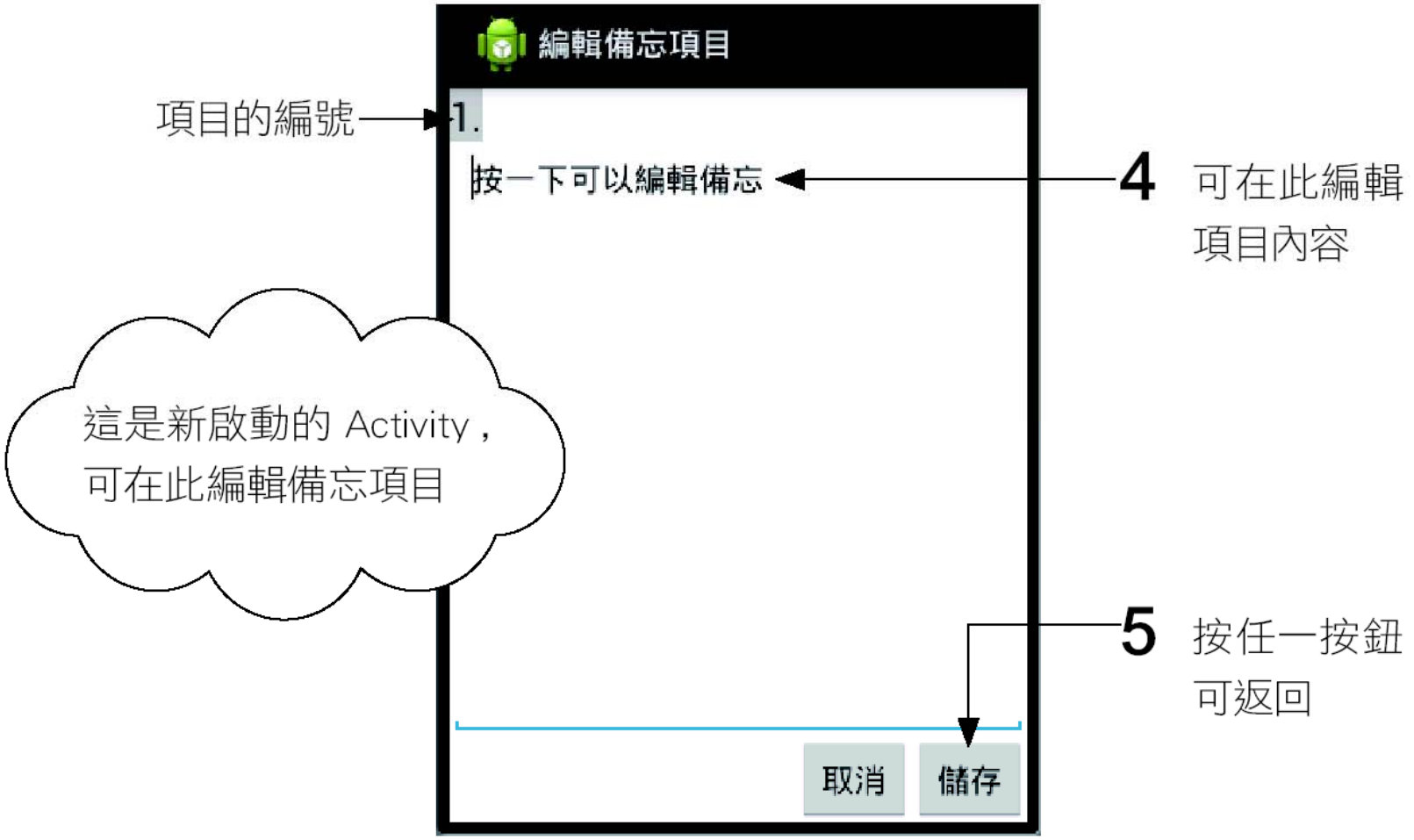
範例8-3 ：在啟動新 Activity 時傳送資料



在啟動新 Activity 時傳送資料



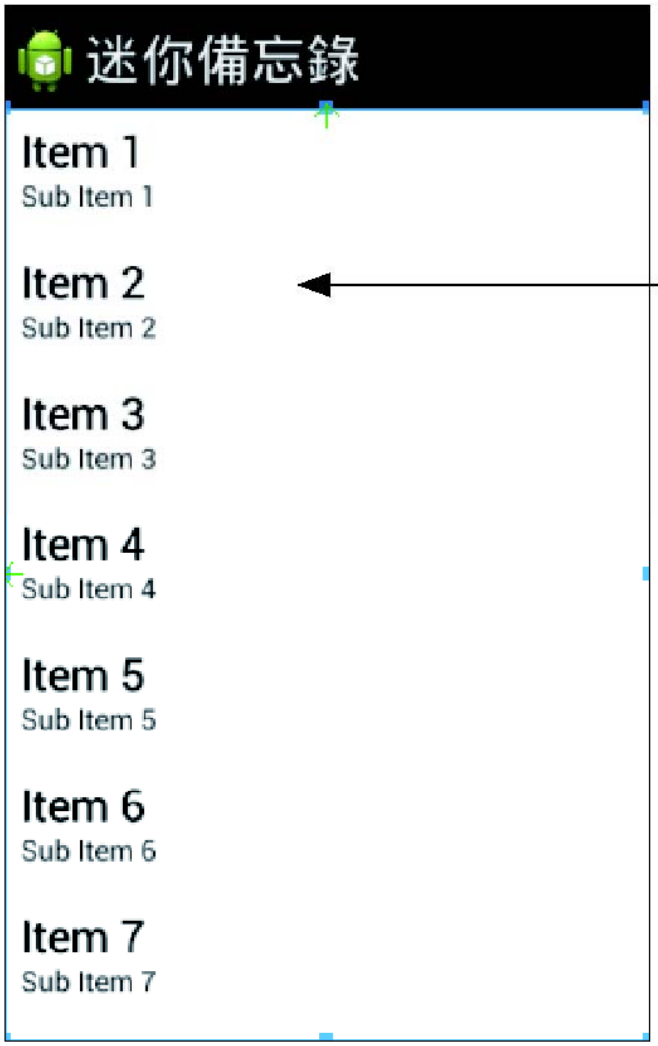
在啟動新 Activity 時傳送資料



在啟動新 Activity 時傳送資料

step 1

step 2

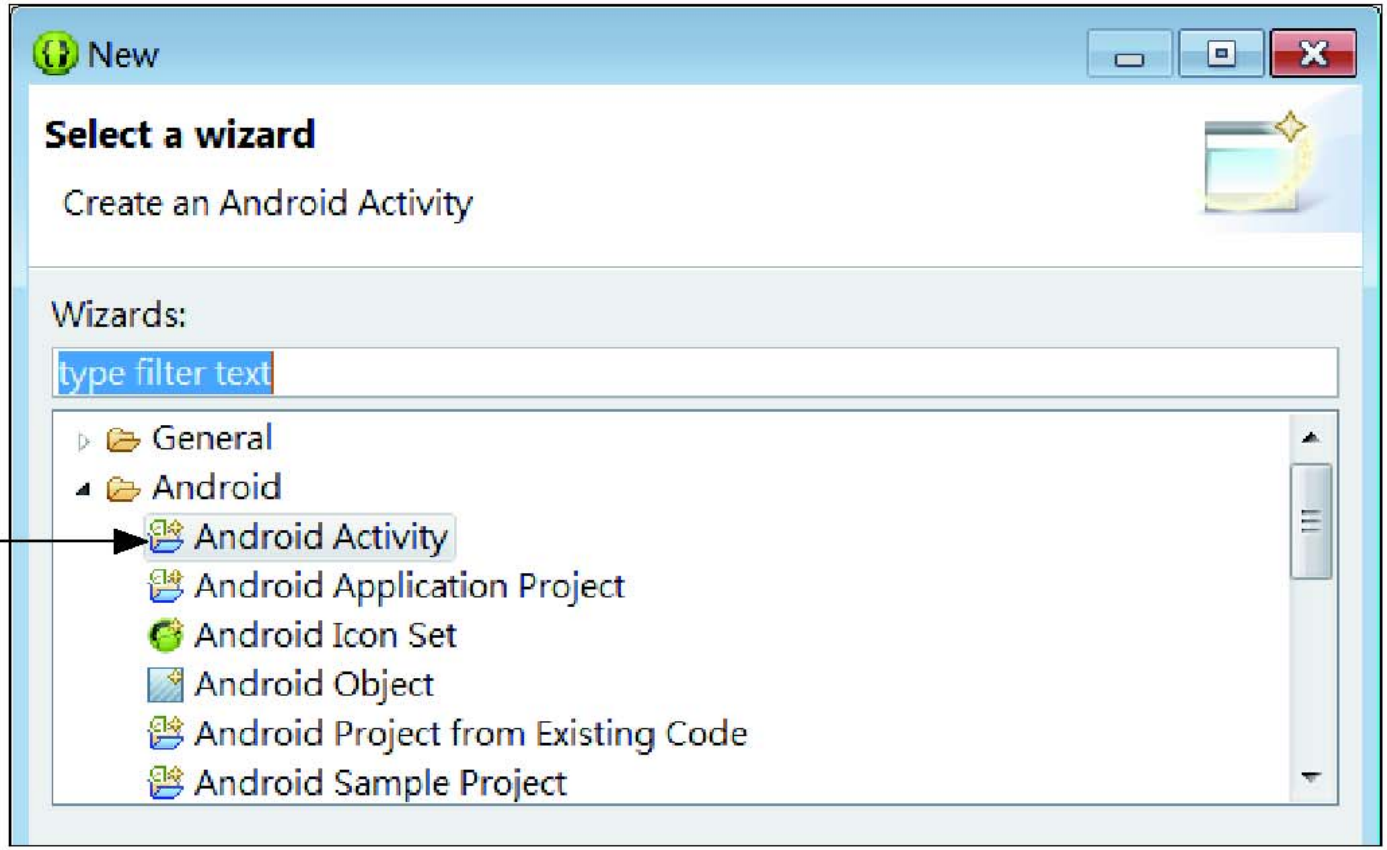


ListView

Id	@+id/listView1
Width	match_parent
Height	wrap_content

在啟動新 Activity 時傳送資料

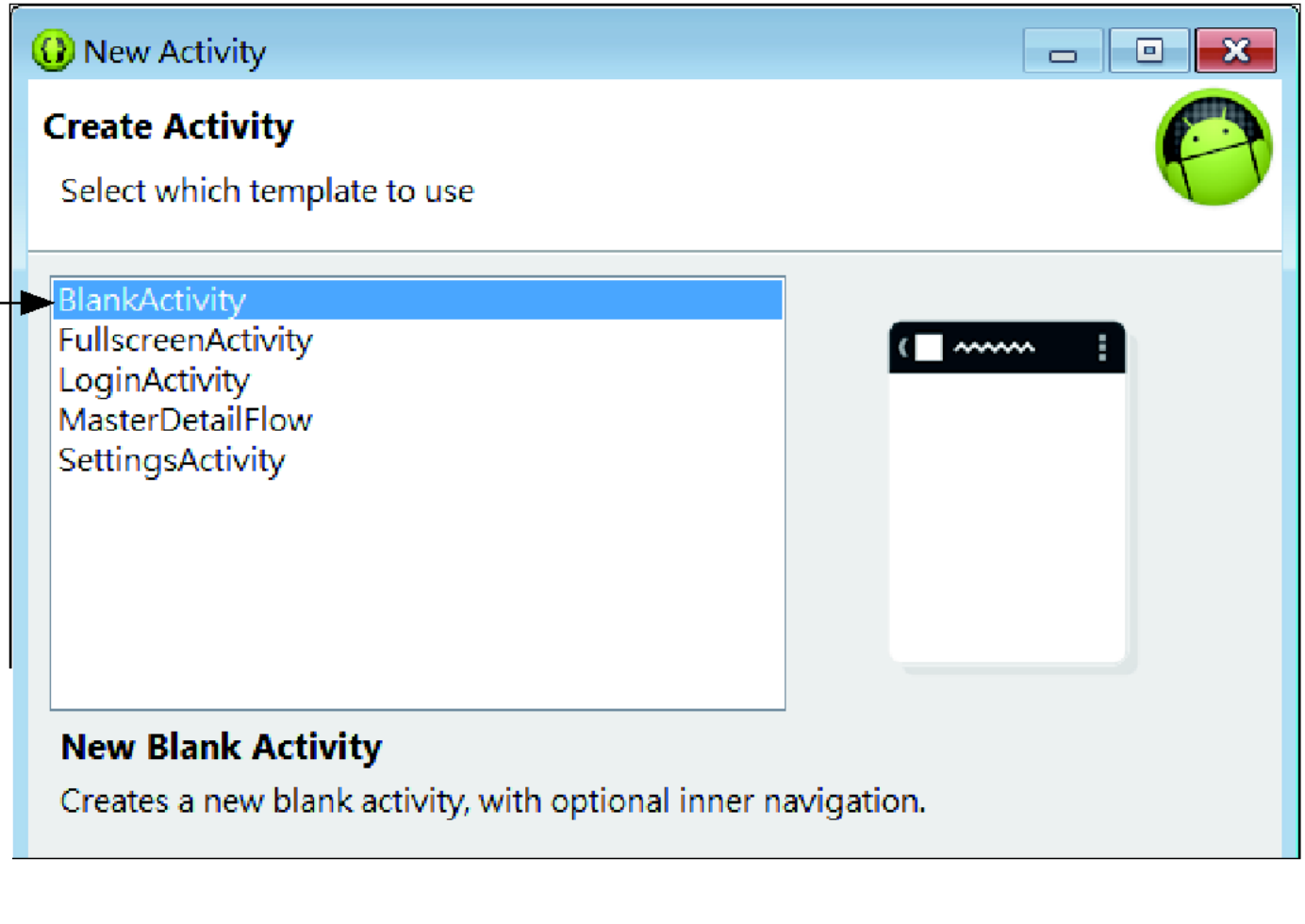
step 3



1 選 Android Activity

在啟動新 Activity 時傳送資料

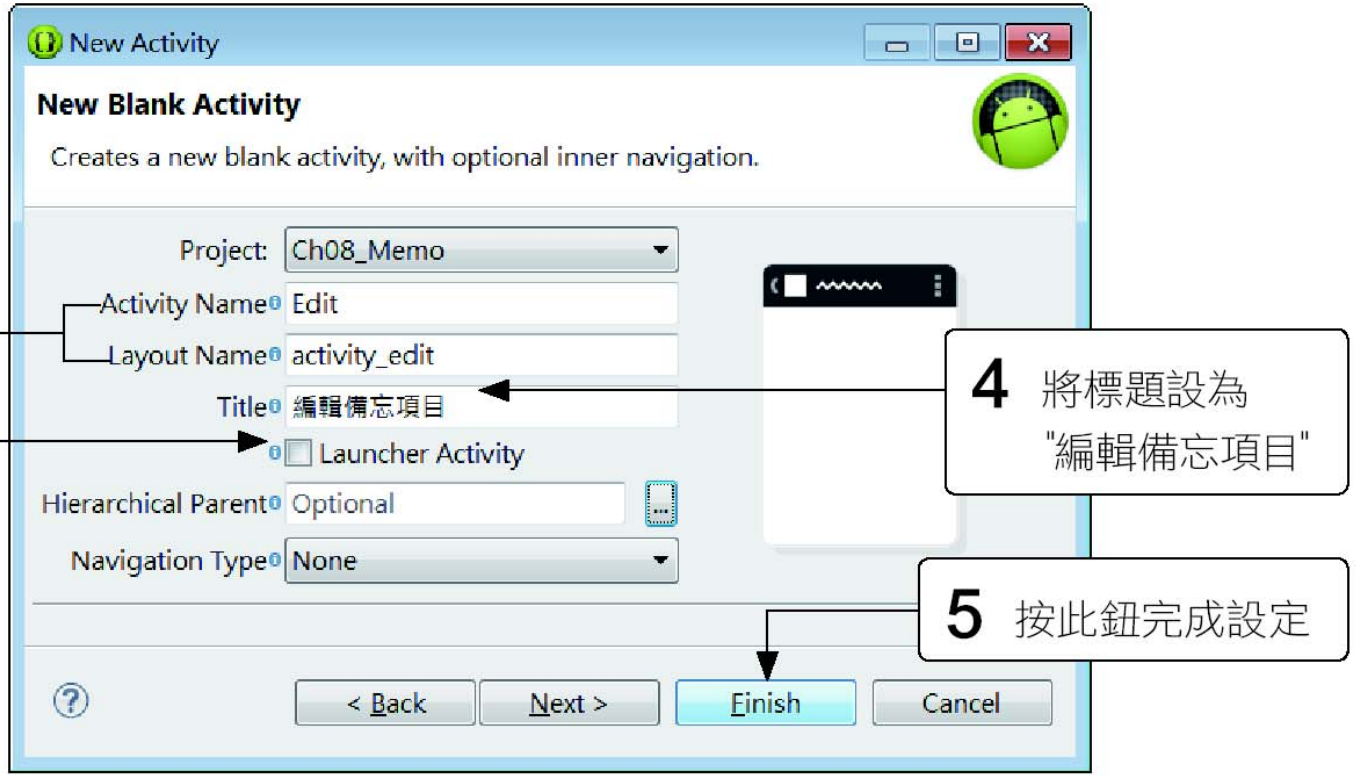
2 選此項



在啟動新 Activity 時傳送資料

3 輸入新 Activity 的類別名稱：Edit, 以及佈局檔名稱：activity_edit

這項不要選取

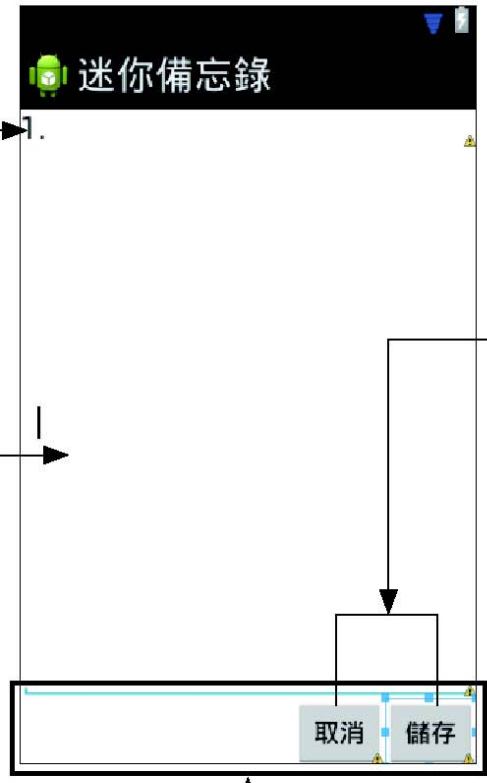


在啟動新 Activity 時傳送資料

step 4

1 加入 Large TextView 顯示編號, 背景設為淺灰色

Id	@+id/textView1
Width	match_parent
Height	wrap_content
Text	1.



2 加入 EditText 元件, 設 Weight=1 使高度填滿剩餘空間

Id	@+id/editText1
Width	match_parent
Height	0dp
Weight	1

3 加入 LinearLayout (Horizontal) 並設定 Gravity 使內容向右對齊

Width	match_parent
Height	wrap_content
Gravity	right

4 在 LinearLayout 中放 2 個按鈕

Id	@+id/btnCancel
Text	取消
On Click	onCancel

Id	@+id/btnSave
Text	儲存
On Click	onSave

在啟動新 Activity 時傳送資料

step 5

```
01  public class MainActivity extends Activity
02      implements OnItemClickListener, OnItemLongClickListener{
03
04      String[] aMemo = { ◄── 預設的備忘內容
05              "1. 按一下可以編輯備忘",
06              "2. 長按可以清除備忘", "3.", "4.", "5.", "6." };
07      ListView lv; ◄── 顯示備忘錄的 ListView
08      ArrayAdapter<String> aa; ◄── ListView 與備忘資料 aMemo 的橋樑
09
```

在啟動新 Activity 時傳送資料

step 6

```
01      @Override
02      protected void onCreate(Bundle savedInstanceState) {
03          super.onCreate(savedInstanceState);
04          setContentView(R.layout.activity_main);
05
06          lv = (ListView)findViewById(R.id.listView1);
07          aa = new ArrayAdapter<String>(this,
```

在啟動新 Activity 時傳送資料

```
08                  android.R.layout.simple_list_item_1, aMemo);
09
10          lv.setAdapter(aa);      ◄── 設定 listView1 的內容
11
12          //設定 listView1 被按一下的監聽器
13          lv.setOnItemClickListener(this);
14          //設定 listView1 被長按的監聽器
15          lv.setOnItemLongClickListener(this);
16      }
```

在啟動新 Activity 時傳送資料

step 7

```
01      public void onItemClick(AdapterView<?> a,
02                              View v, int pos, long id) {
03          Intent it = new Intent(this, Edit.class);
04          it.putExtra("編號", pos+1);          ← 附加編號
05          it.putExtra("備忘", aMemo[pos]);     ← 附加備忘項目的內容
06          startActivity(it);                   ← 啟動 Edit 活動
07      }
08
09      public boolean onItemLongClick(AdapterView<?> a,
10              View v, int pos, long id) {
11          aMemo[pos] = (pos+1) + ".";  ← 將內容清除 (只剩編號)
12          aa.notifyDataSetChanged();   ← 通知 ListView 要更新顯示的內容
13          return true;                 ← 傳回 true 表示此事件已處理
14      }
```

在啟動新 Activity 時傳送資料

step 8

```
01  protected void onCreate(Bundle savedInstanceState) {
02      super.onCreate(savedInstanceState);
03      setContentView(R.layout.activity_edit);
04
05      Intent it = getIntent();                    ← 取得傳入的 Intent 物件
06      int no = it.getIntExtra("編號", 0);          ←
                              讀出名為 "編號" 的 Int 資料, 若沒有則傳回 0
07      String s = it.getStringExtra("備忘");        ←
                                          讀出名為 "備忘" 的 String 資料
08
09      TextView txv = (TextView)findViewById(R.id.textView1);
```

Next

在啟動新 Activity 時傳送資料

```
10        txv.setText(no + ".");      ← 在畫面左上角顯示編號
11        EditText edt = (EditText)findViewById(R.id.editText1);
12        if(s.length() > 3)
13            edt.setText(s.substring(3)); ←┐
                                  將傳來的備忘資料去除前3個字, 然後填入 EditText 元件中
14    }
15
16    public void onCancel(View v) {   ← 按取消鈕時
17        finish();    ← 結束 Activity
18    }
19    public void onSave(View v) {    ← 按儲存鈕時
20        finish();    ← 結束 Activity
21    }
```

8-4 要求新 Activity 傳回資料

```
startActivityForResult(Intent it, int 識別碼)
```

```
setResult(int 結果碼, Intent it)
```

```
onActivityResult(int 識別碼, int 結果碼, Intent it)
```

要求新 Activity 傳回資料

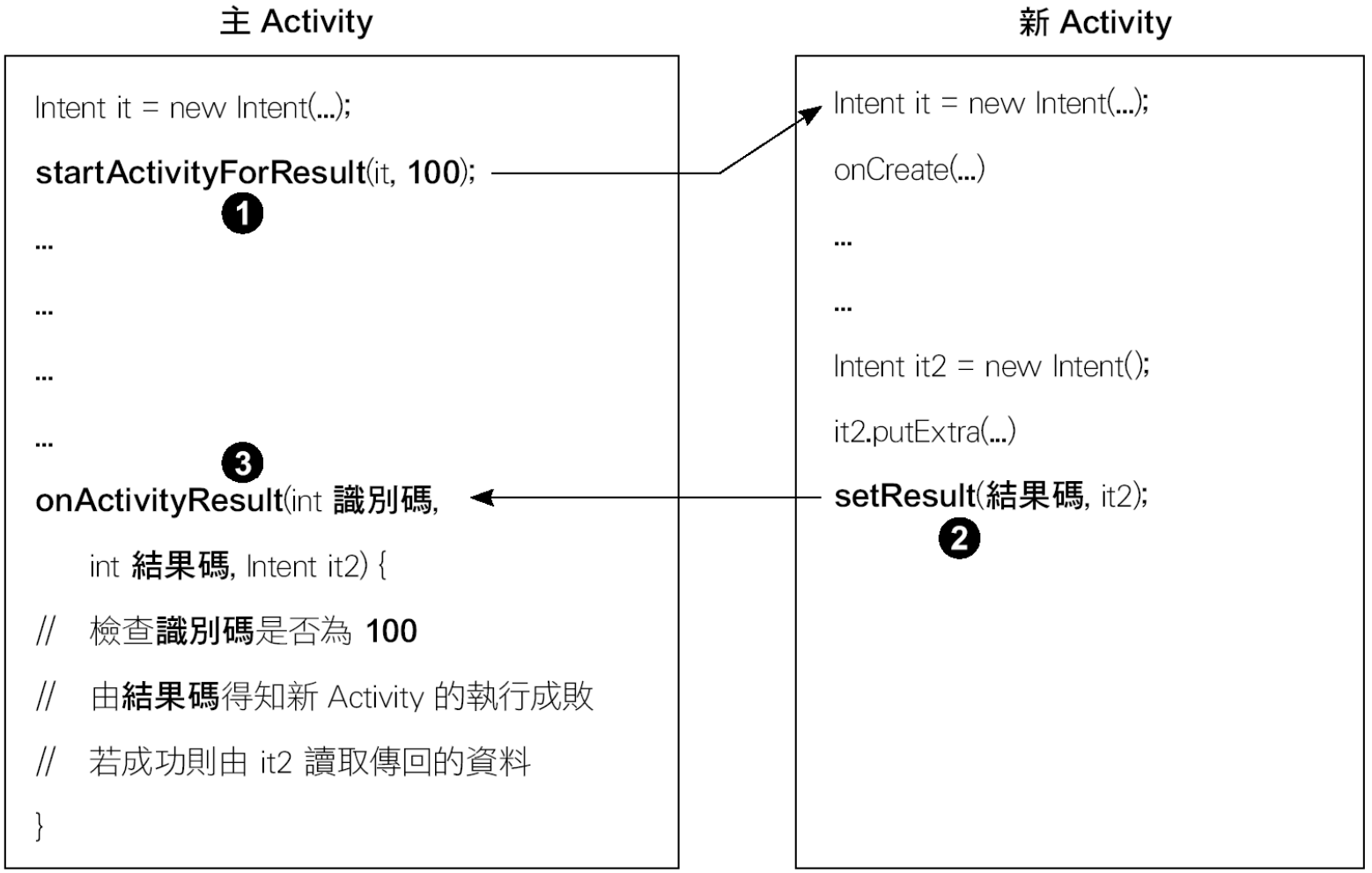


圖 8-2 從 Activity 傳回資料

範例8-4 ：在新 Activity 結束時將資料傳回

step 1

step 2

```
01      public void onItemClick(AdapterView<?> a, View v,
02              int pos, long id) {
03          Intent it = new Intent(this, Edit.class);
04          it.putExtra("備忘", aMemo[pos]);
05          startActivityForResult(it, pos);
06      }
```

只附加備忘項目的內容(而不用再附加編號)

啟動 Edit 並以項目位置為識別碼

在新 Activity 結束時將資料傳回

step 3

```
01      protected void onActivityResult(int requestCode,
02              int resultCode, Intent it) {
03          if(resultCode == RESULT_OK) {
04              aMemo[requestCode] = it.getStringExtra("備忘");
05              aa.notifyDataSetChanged();
06          }
07      }
```

使用傳回的資料更新陣列內容

通知 Adapter 陣列內容有更新

在新 Activity 結束時將資料傳回

step 4

```
01 public class Edit extends Activity {
02     TextView txv;
03     EditText edt;
04
05     @Override
06     protected  void onCreate(Bundle savedInstanceState) {
07         super.onCreate(savedInstanceState);
08         setContentView(R.layout.activity_edit);
09
10         Intent it = getIntent();
11         String s = it.getStringExtra("備忘");
```

← 將 12、14 行的變數宣告移到此處

← 取得傳入的 Intent 物件

讀出名為 "備忘" 的 String 資料(而不用再讀取編號資料)

在新 Activity 結束時將資料傳回

```
12          txv = (TextView)findViewById(R.id.textView1);
                                                  將 txv 改在第 2 行宣告
13          txv.setText(s.substring(0, 2));     ← 將編號顯示在畫面左上角
14          edt = (EditText)findViewById(R.id.editText1);
15          if(s.length() > 3)                    將 edt 改在第 3 行宣告
16              edt.setText(s.substring(3)); ← 將備忘資料去除前3個字,
17      }                                        再填入編輯元件中
18
19      public void onCancel(View v) {    ← 按取消鈕時
20          setResult(RESULT_CANCELED);   ← 傳回取消訊息
21          finish();                      ← 結束 Activity
22      }
```

在新 Activity 結束時將資料傳回

```
23      public void onSave(View v) {          ← 按儲存鈕時
24          Intent it2 = new Intent();
25          it2.putExtra("備忘", txv.getText() + " " + edt.getText());
                                                  附加項目編號與修改後的內容
26          setResult(RESULT_OK, it2);        ← 傳回成功訊息，及修改後的資料
27          finish();                         ← 結束 Activity
28      }
...
```

在新 Activity 結束時將資料傳回

step 5