Lab8
進階清單元件

本節目的：

- 了解什麼是 ViewHolder。
- 了解 Adapter 與 ViewHolder 的關係。
- 學習使用清單元件 RecyclerView 與 RecyclerView.ViewHolder。

8.1 View 的複用

現在我們學會了如何在 Android 中使用清單元件，透過 Adapter 為每一筆資料創建一個 View。在 ListView 與 GridView 中有時我們需要載入大量的資料，如果每次都要創建一個 View 實體，便會佔據大量內存影響效能表現。

這時候我們便需要使用 ViewHolder，ViewHolder 並不是 Android SDK 所提供的 API，而是一種設計方法。藉由設計一個客製化的靜態類別來緩存 View，省去 Adapter 在更新畫面時創建新的實體。

清單元件為畫面的資料創建各自的 View 實體，資料越多 View 的實體也越多，佔據可用的內存空間，如圖 8-1 所示。

圖 8-1 清單元件中的 View 實體

8.1.1ViewHolder 介紹

ViewHolder 通常運用在 Adapter 中，目的是提升清單元件的效率，而不需要重複創建相同結構的 View 實體，讓資源得以重複利用。

在 Android 開發中清單是一個很重要的元件，它以列表的形式根據資料的型態展示具體內容，使用者可以自由的定義 Adapter 每一列的布局，但當清單有大量的資料需要載入的時候，會佔據大量內存，影響性能，這時候就需要重新使用 View 來減少對象的創建。

圖 8-2 為 ViewHolder 運作示意圖，當 View1 離開畫面後，只要更新顯示的內容就可以作為其他資料的 View 使用，如此一來便不需要重複創建的步驟。

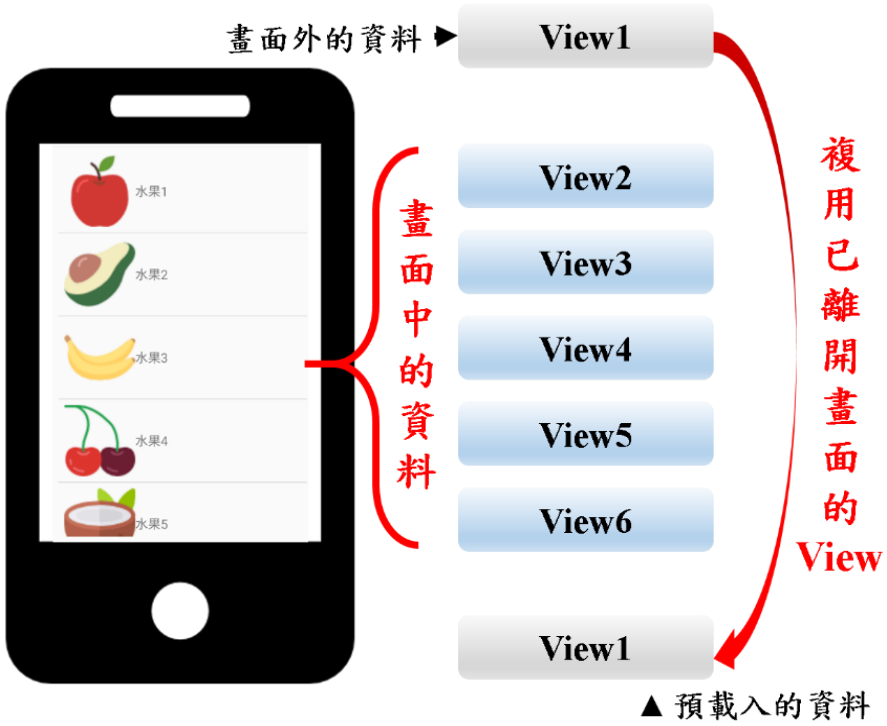

圖 8-2 ViewHolder 運作示意圖

8.1.2在 Adapter 中使用 ViewHolder

以下我們將以 Lab7 的練習為例，改寫 Lab7 的 MyAdapter 類別，加入 ViewHolder 概念。

Step1 首先我們要創建一個客製化的 ViewHolder 類別，用來緩存我們的 View。

```
private class ViewHolder {
    lateinit var img_photo: ImageView
    lateinit var tv_name: TextView
}
```

Step2 接著改寫 getView()函式，實現 View 的回收機制。

```
override fun getView(position: Int, convertView: View?,
parent: ViewGroup?): View {
    //由於 convertView 無法賦值，必須額外建立一個 View 物件
    val view: View
    val holder: ViewHolder

    if(convertView==null){
        //建立新的 View
        view = View.inflate(parent?.context, layout, null)
        holder = ViewHolder()
        //為 View 加上標籤以便重複使用
        view.tag = holder
        //連結畫面中的元件
        holder.img_photo = view.findViewById(R.id.img_photo)
        holder.tv_name = view.findViewById(R.id.tv_name)
    }else{
        //重複使用已存在的 View
        holder = convertView.tag as ViewHolder
        view = convertView
    }

    ... 省略

    return view
}
```

8.1.3RecyclerView

RecyclerView 被視為下一代的清單元件，用來取代 ListView 與 GridView，不只是因為 RecyclerView 擁有多元的呈現樣貌，更因為它強制開發者實現 View 的回收機制，也就是前面我們提到的 ViewHolder 類別。在 RecyclerView 時，當你定義一個新的 Adapter 時，必須同時定義好對應的 ViewHolder，並且實作 onCreateViewHolder()與 onBindViewHolder()以運行回收機制。

此處將略過客製化 Data 與 Layout 的實作步驟，詳情請見第七章。

- 客製化 Adapter 與 ViewHolder：RecyclerView 必須搭配 RecyclerView 類別中的 Adapter 與 ViewHolder 使用，無法使用第七章所介紹的 BaseAdapter 類別以及 8.1.2 的 ViewHolder。

```
class MyAdapter() :
RecyclerView.Adapter<MyAdapter.ViewHolder>(){
    //實作 RecyclerView.ViewHolder 來緩存 View
    class ViewHolder(v: View):RecyclerView.ViewHolder(v){
    }
    //創建新的 ViewHolder 並連結畫面
    override fun onCreateViewHolder(viewGroup: ViewGroup,
position: Int): ViewHolder {
        val v = LayoutInflater.from(viewGroup.context)
                    .inflate(R.layout.layout, viewGroup,false)
        return ViewHolder(v)
    }
    //回傳資料來源筆數
    override fun getItemCount() = 0
    //將資料與元件綁定
    override fun onBindViewHolder(holder: ViewHolder, position:
Int) {
    }
}
```

- 呈現方式：RecyclerView 的呈現方式是由 LayoutManager 決定，最常見的就是 LinearLayoutManager 與 GridLayoutManager 兩種，可以做出 ListView 與 GridView 的效果外，還能修改 orientation 屬性改變清單的呈現方向。

```
//創建 LinearLayoutManager 物件
val linearLayoutManager = LinearLayoutManager(this)
//設定清單的呈現方向
linearLayoutManager.orientation = LinearLayoutManager.VERTICAL
//連結 LinearLayoutManager
recyclerView.layoutManager = linearLayoutManager
//連結 Adapter
recyclerView.adapter = MyAdapter(list)
```

```
//創建 GridLayout 物件，並設定每列/行的資料數
val gridLayoutManager = GridLayoutManager(this, 3)
//設定清單的呈現方向
gridLayoutManager.orientation = GridLayoutManager.VERTICAL
//連結 GridLayoutManager
recyclerView.layoutManager = gridLayoutManager
//連結 Adapter
recyclerView.adapter = MyAdapter(list)
```

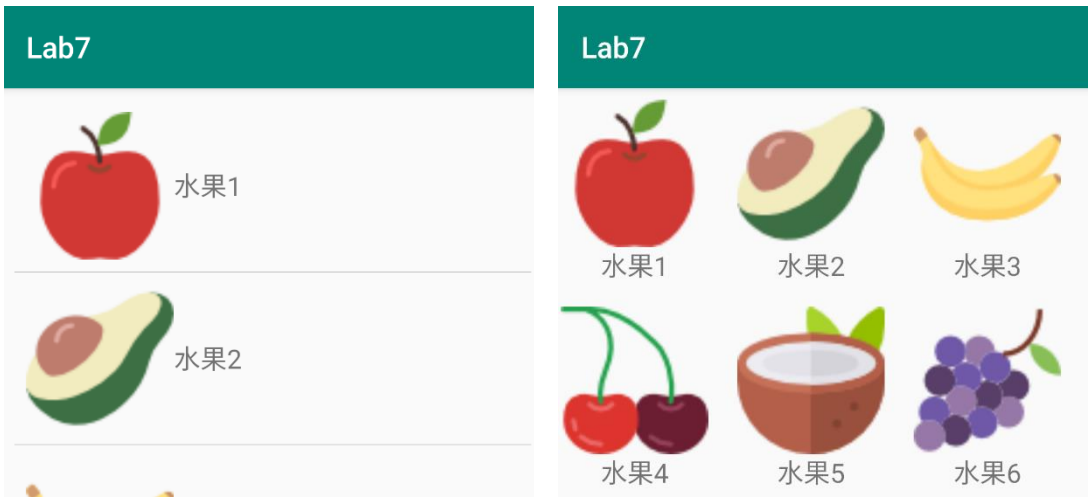

圖 8-3 LinearLayoutManager（左）、GridLayoutManager（右）

8.2 電話簿

- 設計一個有新增刪除功能的電話簿。
- 點擊「新增聯絡人」新增資料，點擊「X」刪除列表中的資料。
- 藉由第四章所學的 Bundle 與 setResult()等功能傳遞聯絡人資料。

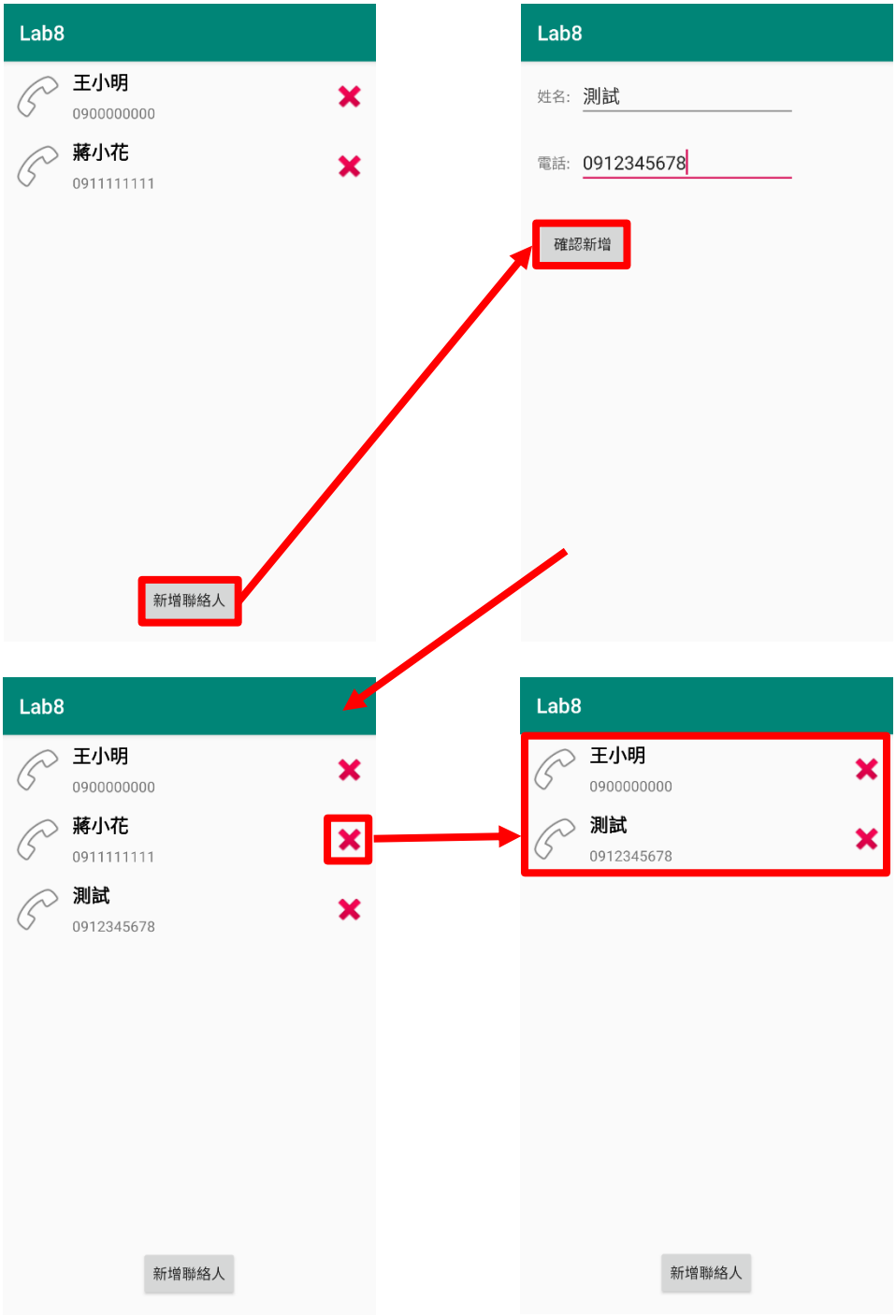

圖 8-4 聯絡人清單（左一）、新增聯絡人（右一）、聯絡人清單（左二）、刪除聯絡人（右二）

8.2.1電話簿與聯絡人畫面設計

Step1 建立新專案，以及圖 8-5 對應的 class 與 xml 檔。

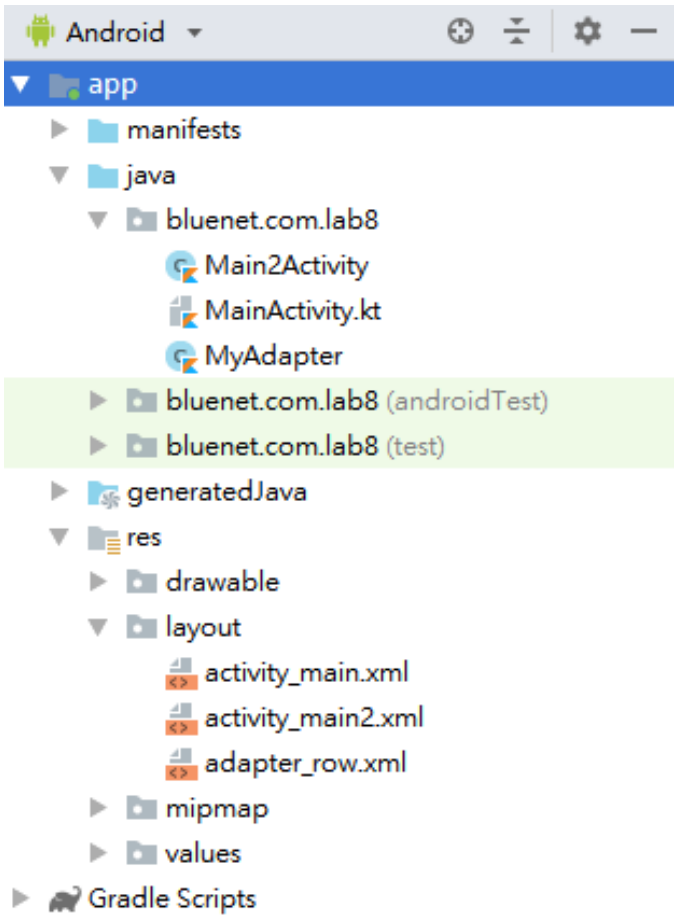

圖 8-5 電話簿專案架構

Step2 繪製 activity_main.xml，如圖 8-6 所示。

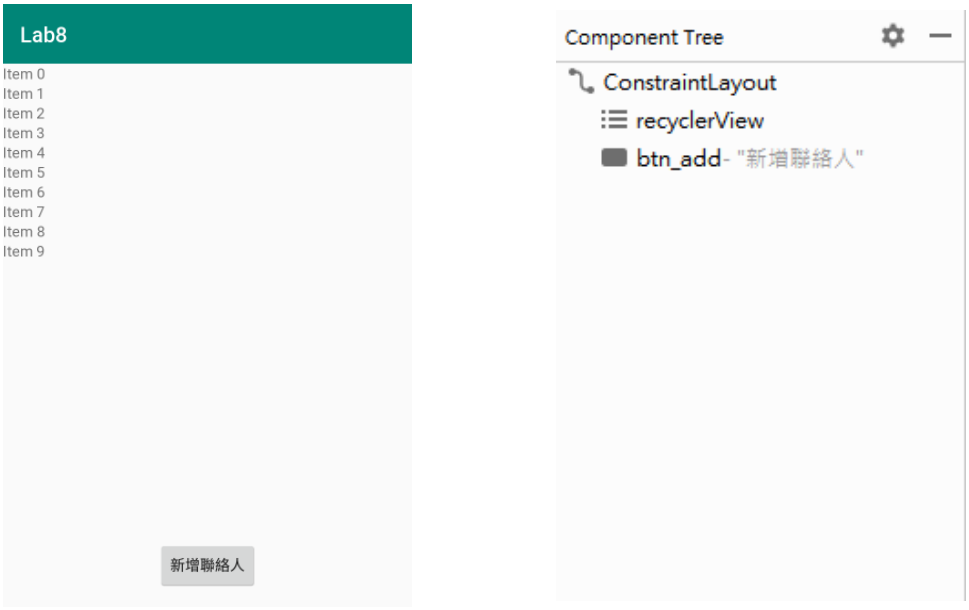

圖 8-6 聯絡人列表預覽畫面（左）與布局元件樹（右）

對應的 xml 如下：

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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" >

    <android.support.v7.widget.RecyclerView
            android:id="@+id/recyclerView"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            app:layout_constraintBottom_toTopOf="@+id/btn_add"
            app:layout_constraintTop_toTopOf="parent"/>

    <Button
            android:id="@+id/btn_add"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
```

```
            android:layout_marginBottom="16dp"
            android:layout_marginStart="8dp"
            android:layout_marginEnd="8dp"
            android:text="新增聯絡人"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"/>
</android.support.constraint.ConstraintLayout>
```

Step3 繪製新增聯絡人頁面 activity_main2.xml，如圖 8-7 所示。

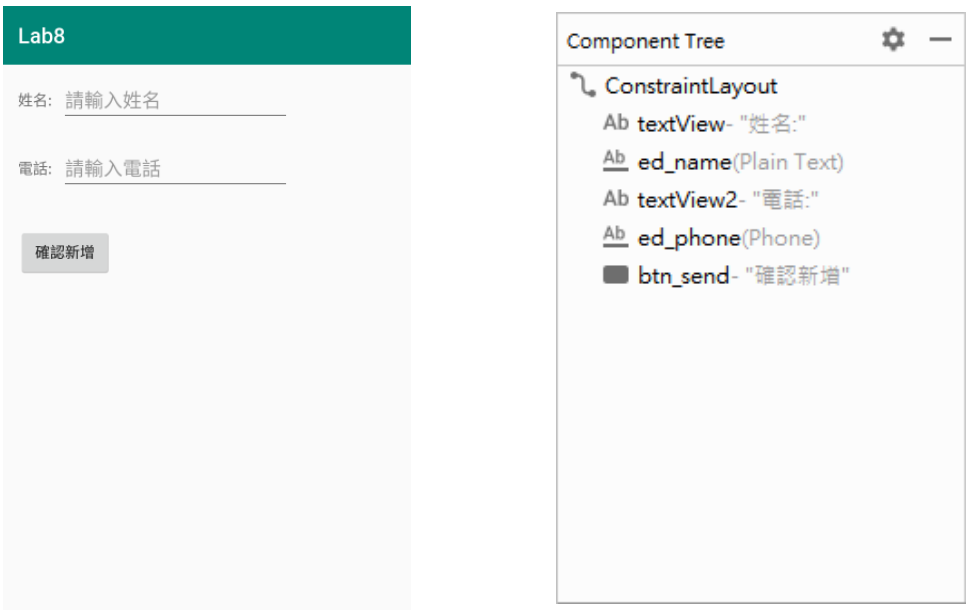

圖 8-7 新增聯絡人預覽畫面（左）與布局元件樹（右）

對應的 xml 如下：

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".Main2Activity">
```

```
<TextView
        android:text="姓名:"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:layout_marginStart="16dp"
        android:layout_marginTop="24dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

<EditText
        android:id="@+id/ed_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:hint="請輸入姓名"
        android:ems="10"
        app:layout_constraintTop_toTopOf="@+id/textView"
        app:layout_constraintBottom_toBottomOf="@+id/textView"
        app:layout_constraintStart_toEndOf="@+id/textView"
        android:layout_marginStart="8dp" />

<TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="電話:"
        android:layout_marginTop="32dp"
        app:layout_constraintStart_toStartOf="@+id/textView"
        app:layout_constraintTop_toBottomOf="@+id/ed_name" />

<EditText
        android:id="@+id/ed_phone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="phone"
        android:hint="請輸入電話"
        android:ems="10"
```

```
            android:layout_marginStart="8dp"
            app:layout_constraintTop_toTopOf="@+id/textView2"
            app:layout_constraintBottom_toBottomOf="@+id/textView2"
            app:layout_constraintStart_toEndOf="@+id/textView2" />

    <Button
            android:id="@+id/btn_send"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="確認新增"
            android:layout_marginTop="32dp"
            app:layout_constraintStart_toStartOf="@+id/textView2"
            app:layout_constraintTop_toBottomOf="@+id/ed_phone" />
</android.support.constraint.ConstraintLayout>
```

Step4 繪製聯絡人清單的布局 adapter_row.xml，如圖 8-8 所示。

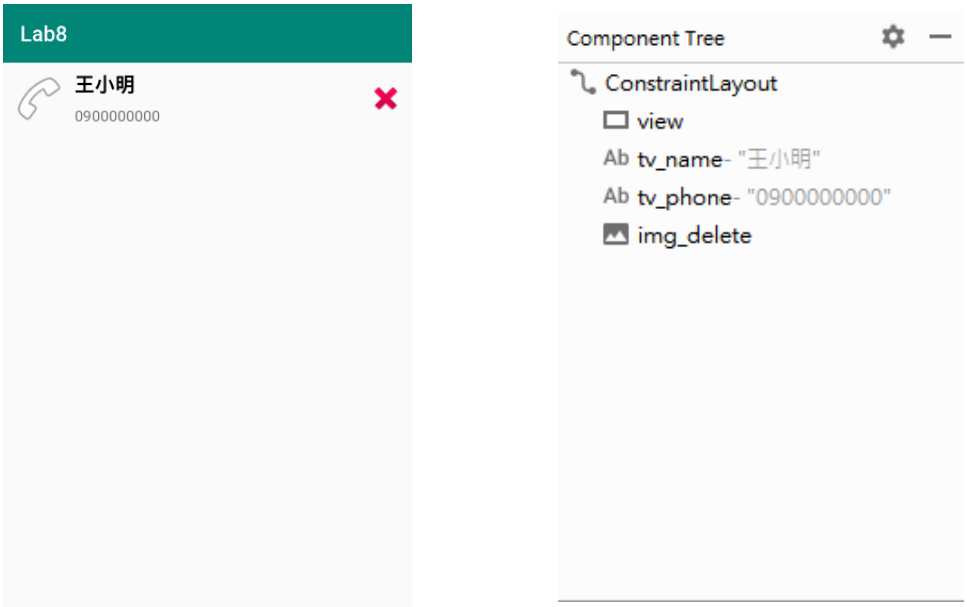

圖 8-8 聯絡人預覽畫面（左）與布局元件樹（右）

對應的 xml 如下：

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

<View
        android:id="@+id/view"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@android:drawable/ic_menu_call"
        android:layout_marginStart="8dp"
        app:layout_constraintTop_toTopOf="@+id/tv_name"
        app:layout_constraintBottom_toBottomOf="@+id/tv_phone"
        app:layout_constraintDimensionRatio="1:1"
        app:layout_constraintStart_toStartOf="parent" />

<TextView
        android:id="@+id/tv_name"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="王小明"
        android:textColor="@android:color/black"
        android:textStyle="bold"
        android:textSize="18sp"
        android:layout_marginTop="8dp"
        android:layout_marginStart="8dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toEndOf="@+id/view" />

<TextView
        android:id="@+id/tv_phone"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginBottom="8dp"
        android:text="0900000000"
        app:layout_constraintTop_toBottomOf="@+id/tv_name"
        app:layout_constraintStart_toEndOf="@+id/view"
        app:layout_constraintBottom_toBottomOf="parent" />
```

```
    <ImageView
            android:layout_width="35dp"
            android:layout_height="35dp"
            app:srcCompat="@android:drawable/ic_delete"
            android:id="@+id/img_delete"
            android:layout_marginEnd="8dp"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp"
            app:layout_constraintTop_toTopOf="@+id/tv_name"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintBottom_toBottomOf="@+id/tv_phone" />
</android.support.constraint.ConstraintLayout>
```

8.2.2RecyclerView 程式設計

Step1 編寫 MainActivity，建立一個客製化的聯絡人資料類別，包含姓名與電話，用於保存之後要顯示於客製化 Adapter 的資料。

```
data class Contact (
    val name: String, //姓名
    val phone: String //電話
)
```

Step2 建立 MyAdapter 來顯示 RecyclerView 的客製化畫面。由於 RecyclerView 必須搭配 ViewHolder 來使用，因此我們需要在 MyAdapter 中創建一個繼承 RecyclerView.ViewHolder 類別的 Holder 使用。

```
import android.support.v7.widget.RecyclerView
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView

class MyAdapter(private val contacts:ArrayList<Contact>) :
RecyclerView.Adapter<MyAdapter.ViewHolder>(){
    //實作 RecyclerView.ViewHolder 來緩存 View
    class ViewHolder(v: View):RecyclerView.ViewHolder(v){
        //元件宣告
        val tv_name = v.findViewById<TextView>(R.id.tv_name)
```

```
        val tv_phone = v.findViewById<TextView>(R.id.tv_phone)
        val img_delete =
                    v.findViewById<ImageView>(R.id.img_delete)
    }
    //創建新的 ViewHolder 並連結畫面
    override fun onCreateViewHolder(viewGroup: ViewGroup,
position: Int): ViewHolder {
        val v = LayoutInflater.from(viewGroup.context)
                .inflate(R.layout.adapter_row, viewGroup, false)
        return ViewHolder(v)
    }
    //回傳資料來源筆數
    override fun getItemCount() = contacts.size
    //將資料與元件綁定
    override fun onBindViewHolder(holder: ViewHolder, position:
Int) {
        holder.tv_name.text = contacts[position].name
        holder.tv_phone.text = contacts[position].phone
        //設定按鈕監聽事件，使用 removeAt()刪除指定位置的資料
        holder.img_delete.setOnClickListener {
            contacts.removeAt(position)
            notifyDataSetChanged()
        }
    }
}
```

Step3 撰寫 MainActivity 程式，初始化聯絡人清單，並透過 onActivityResult()接收新的聯絡人資料。

```
import android.app.Activity
import android.content.Intent
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.support.v7.widget.LinearLayoutManager
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    private lateinit var adapter : MyAdapter
    private val contacts =  ArrayList<Contact>()
    //接收回傳資料
    override fun onActivityResult(requestCode: Int, resultCode:
Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
```

```
        data?.extras?.let {
            if(requestCode==1 && //判斷發送資料對象
                            resultCode== Activity.RESULT_OK){
                //新增聯絡人資料
                contacts.add(Contact(it.getString("name"),
                                        it.getString("phone")))
                //更新列表
                adapter.notifyDataSetChanged()
            }
        }
    }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        //創建 LinearLayoutManager 物件，設定垂直顯示
        val linearLayoutManager = LinearLayoutManager(this)
        linearLayoutManager.orientation =
                                    LinearLayoutManager.VERTICAL
        recyclerView.layoutManager = linearLayoutManager
        //創建 MyAdapter 並連結 recyclerView
        adapter = MyAdapter(contacts)
        recyclerView.adapter = adapter
        //設定按鈕監聽事件，使用 startActivityForResult()啟動 Main2Activity
        btn_add.setOnClickListener {
            startActivityForResult(Intent(this,
                                Main2Activity::class.java),1)
        }
    }
}
```

Step4 撰寫 Main2Activity 程式，加入按鈕監聽事件，並判斷使用者是否輸入資料。

```
import android.app.Activity
import android.content.Intent
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
```

```kotlin
import kotlinx.android.synthetic.main.activity_main2.*

class Main2Activity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main2)
        //設定按鈕監聽事件
        btn_send.setOnClickListener {
            when{
                //判斷是否輸入資料
                ed_name.length()<1 ->Toast.makeText(this,
                        "請輸入姓名",Toast.LENGTH_SHORT).show()
                ed_phone.length()<1 ->Toast.makeText(this,
                        "請輸入電話",Toast.LENGTH_SHORT).show()
                else->{
                    val b = Bundle()
                    b.putString("name", ed_name.text.toString())
                    b.putString("phone",
                                    ed_phone.text.toString())
                    //使用 setResult()回傳聯絡人資料
                    setResult(Activity.RESULT_OK,
                                        Intent().putExtras(b))
                    finish()
                }
            }
        }
    }
}
```