Lab8 Service

一、本節目的：

- 了解什麼是 Service。
- 使用 Service 執行背景工作。

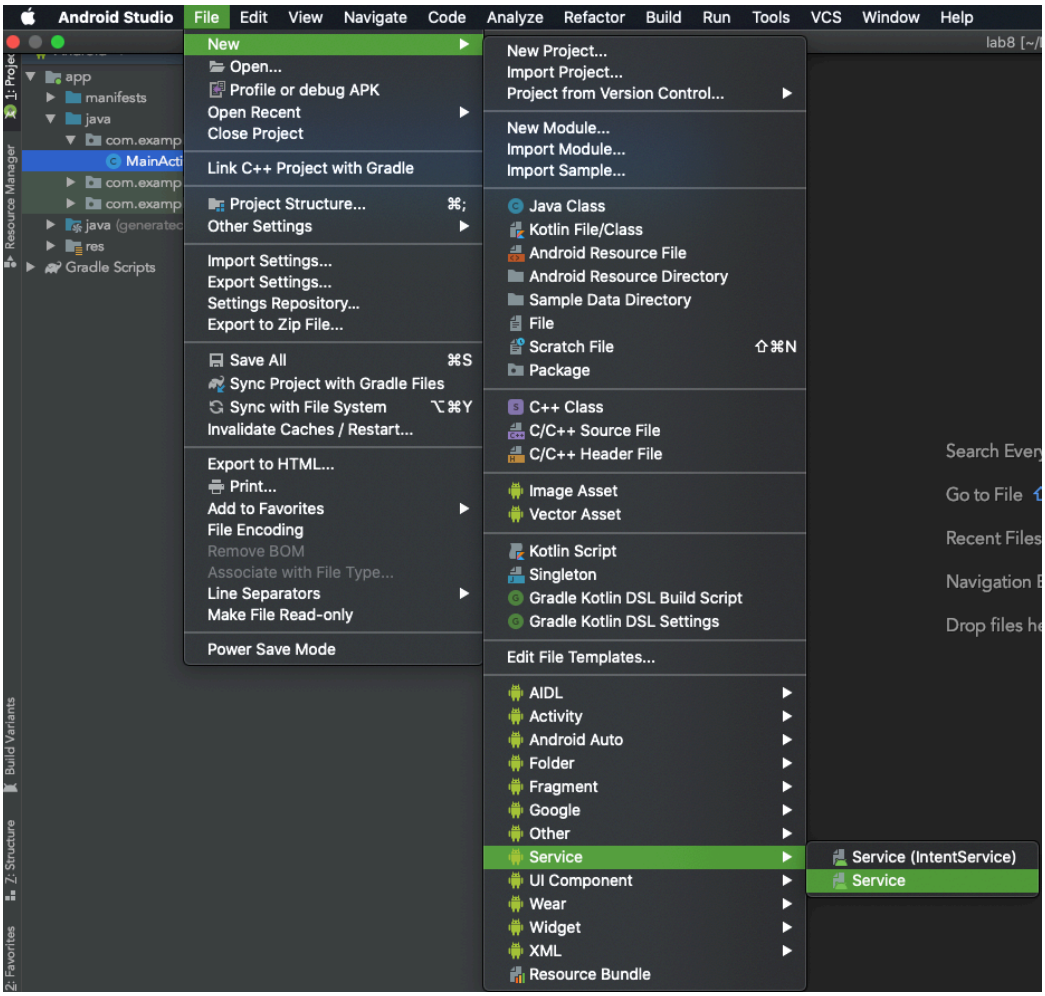
二、觀念說明：

Activity 在離開畫面後會進入停止狀態，在這狀態下我們是無法控制 Activity，除非透過調用 Thread 的方法，但是當 APP 完全結束關閉後，這些工作也會停止，如果希望能在背景中繼續執行工作，我們就會需要使用到 Service 來執行背景作業，像是在背景等待網路連線、背景作業等工作。

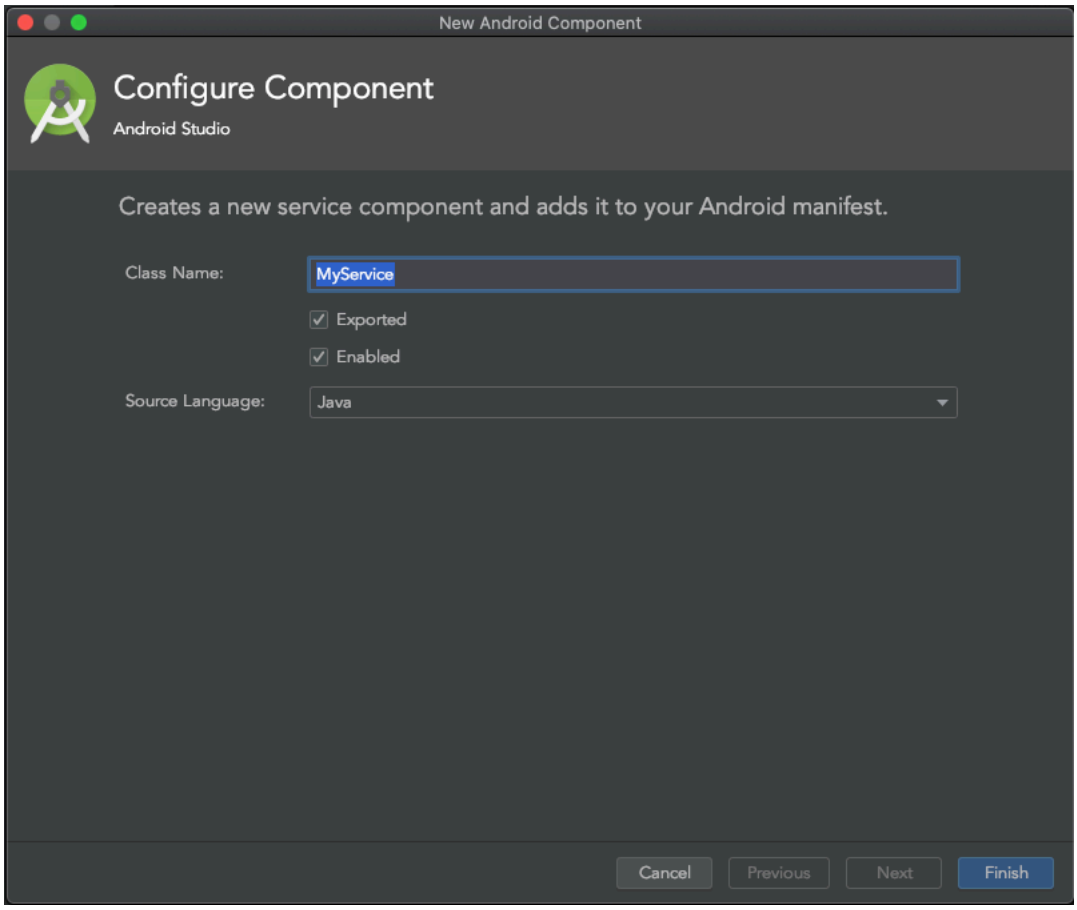
Service 最大的特色是他的執行任務與使用者的操作無關，Service 會獨立運行於背景，因此能夠 APP 完全結束關閉後能保持啟動，甚至能在使用者操作別的 APP 時繼續執行，如等待網路訊息通知、下載資料等作業，一旦任務完成 Service 就可以被結束，也可能就此常駐於裝置上。

1　創建 Service

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



選擇後可於下面的視窗中修改 Service 的名稱，完成後按下 Finish。



完成後，系統會幫你產生出 Service 的 java 檔，之後就可以在此編寫 Service。

```java
package com.example.lab8;

import ...

public class MyService extends Service {
    public MyService() {
    }

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

在 AndroidManifest.xml 中也會自動增加 Service 的資訊。

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.lab8">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="lab8"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning">
        <service
            android:name=".MyService"
            android:enabled="true"
            android:exported="true">
        </service>
```

2 啟動 Service

Service 擁有獨立運行的能力，第一次產生的 Service 會執行 onCreate()方法，之後會自動調用 onStartCommand()方法，我們可以將要執行的工作寫於 onStartCommand()之中，而當需要結束時可以調用 stopSelf()來讓 Service 自行進入結束程序，觸發結束程序後皆會調用 onDestroy()來結束服務。


```
public class MyService extends Service {
    @Override
    public void onCreate() {
        super.onCreate();
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        stopSelf();
        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    public void onDestroy(){
        super.onDestroy();
    }

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

Service 最重要的執行階段便是在調用 onStartCommand()，onStartCommand()會告知系統如何重啟 Service，如異常終止後是否要重新啟動。而運行中的 Service 如果又接收到 Activity 發出 startService()的請求，Service 會執行 onStartCommand()而不會再次執行 onCreate()。換言之，onStartCommand()扮演著接收外來請求並操作 Service 的角色。

onStartCommand()的第一個參數為 Intent，如果從 Activity 啟用，可以藉由 Intent 夾帶資訊傳遞到 Service，而返回值主要有三種定義：

- START_NOT_STICKY：如果 Service 被結束時，便結束服務。
- START_STICKY：如果 Service 被結束時，系統會嘗試重啟並再次調用 onStartCommand()，不過 Intent 會被清空。
- START_REDELIVER_INTENT：如果 Service 被結束時，系統會嘗試重啟並再次調用 onStartCommand()，不過 Intent 會保留前次的並重新傳入。

```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
    super.onStartCommand(intent, flags, startId);

    return START_STICKY;Service 被結束時會重啟並清空 Intent。
}
```

當要從 Activity 中啟用一個 Service，需要在 Activity 中透過 Intent 來做切換，並調用 startService()方法發出訊息。

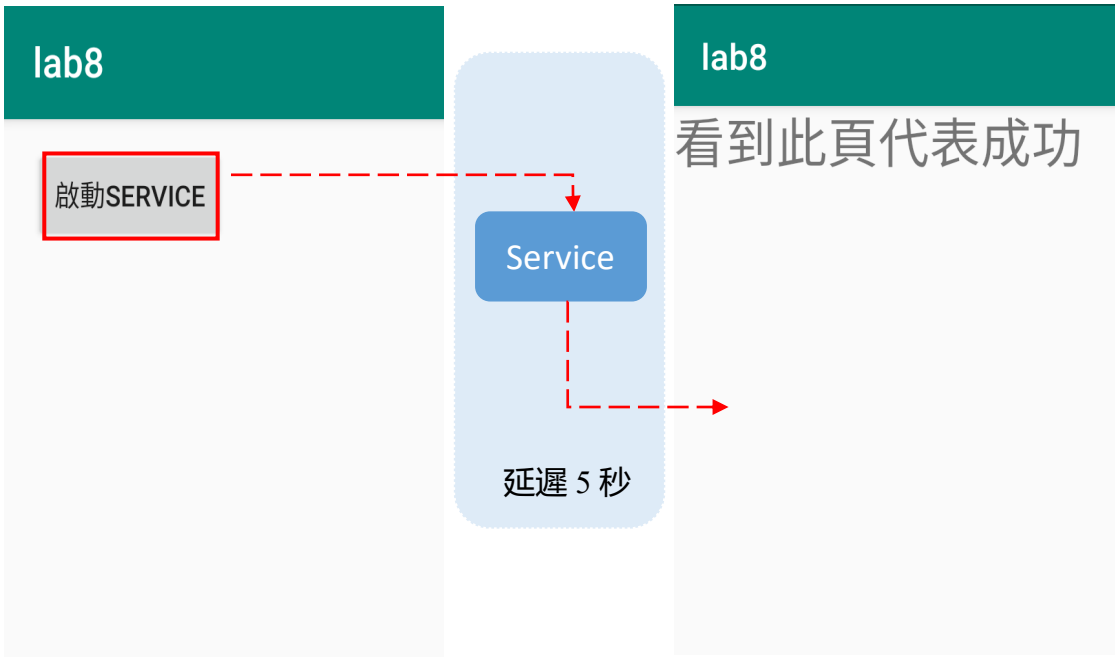
```
Intent intent = new Intent( packageContext: this,MyService.class);
startService(intent);
```

如果要關閉 Service 可以從 Activity 調用，stopService()方法來做停止。

```
Intent intent = new Intent( packageContext: this,MyService.class);
stopService(intent);
```

三、 設計重點:

- 本次範例要透過 Service 來啟動另一個 Activity。
- 在 MainActivity 中，按下”啟動 SERVICE”按鈕後啟動 Service，並結束 MainActivity。
- 在 Service 中，透過 Thread 延遲 5 秒後啟動 Main2Activity。

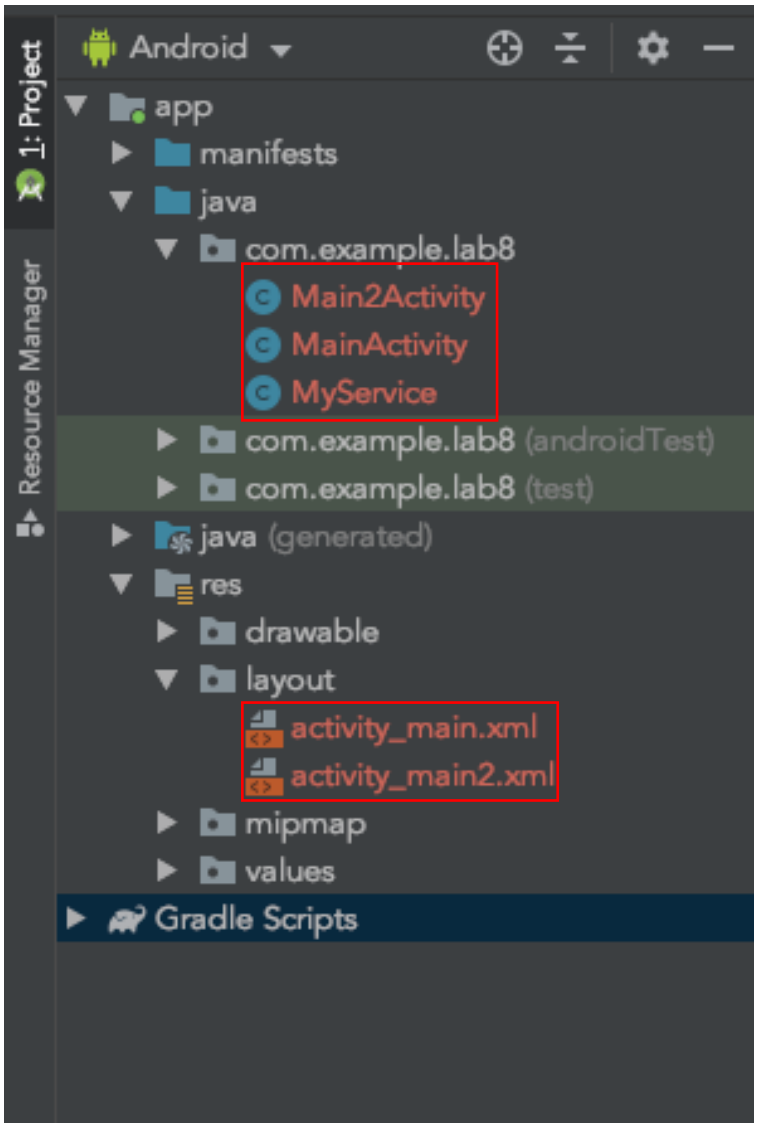


MainActivity　　Main2Activity

四、設計步驟:

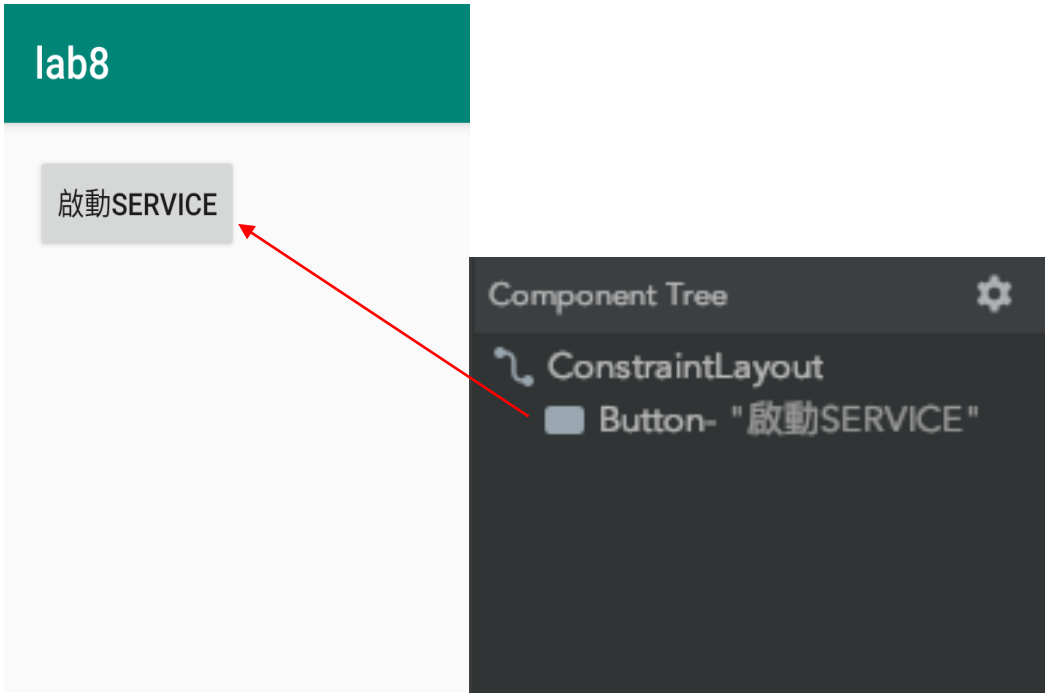
Step1

新建專案，並建立兩個 Activity 與一個 Service



Step2

繪製 activity_main.xml 檔



對應的 xml 如下：

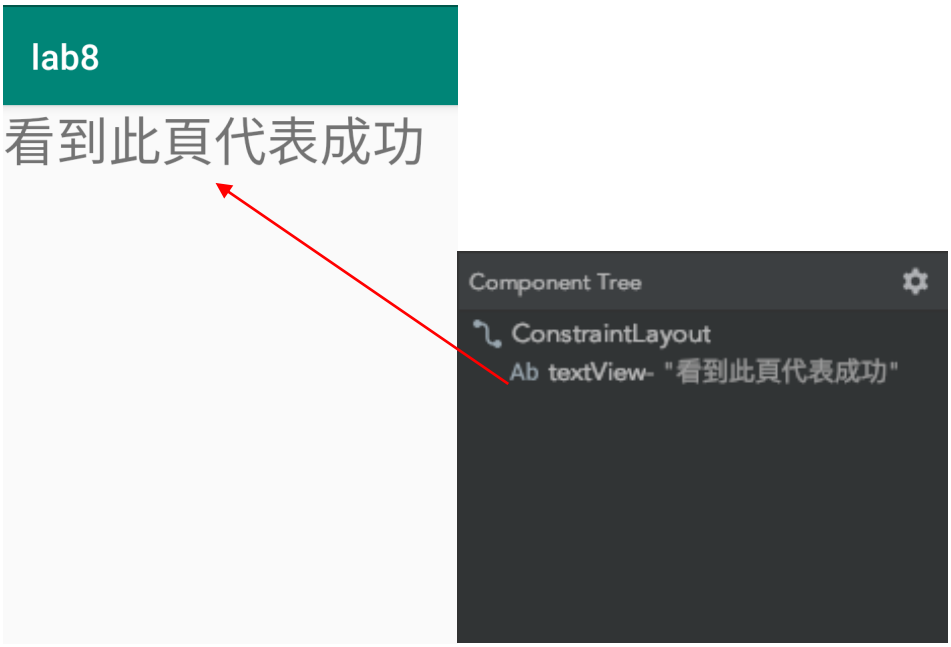
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"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

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="啟動SERVICE"
        android:layout_marginStart="15dp"
        android:layout_marginTop="10dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

Step3

繪製 activity_main2.xml 檔



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=".Main2Activity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="看到此頁代表成功"
        android:textSize="30dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

Step4

編寫 MainActivity 程式，按下按鈕後啟動 Service，Main2Activity 不做設計，僅用於顯示結果

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);   初始化 Activity
        setContentView(R.layout.activity_main);  連接 activity_main.xml 畫面

        Button button = findViewById(R.id.button);   連接 Button 元件
        button.setOnClickListener(new View.OnClickListener() {    Button 點擊事件
            @Override
            public void onClick(View view) {
                Intent intent = new Intent( packageContext: MainActivity.this,MyService.class);
                startService(intent); 啟動 Service
                Toast.makeText( context: MainActivity.this, text: "啟動Service",Toast.LENGTH_LONG).show();
                finish();  關閉 Activity
            }
        });
    }
}
```

Step5

編寫 MyService 程式，在其中加入執行緒延遲 5 秒後，啟動 Main2Activity

```
public class MyService extends Service {
    @Override
    public void onCreate() {
        super.onCreate();                                    使用 Thread 來模擬執行耗時任務
        new Thread(new Runnable(){
            @Override
            public void run() {
                try{
                    Thread.sleep( millis: 5000); 延遲 5 秒

                    Intent intent = new Intent( packageContext: MyService.this,
                                                            Main2Activity.class);
                    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    MyService.this.startActivity(intent);啟動 Main2Activity

                }catch (InterruptedException e){
                    e.printStackTrace();
                }
                    Service 要啟動 Activity 要加入 Flag 定義一個新任務去產生 Activity
            }
        }).start();

        stopSelf();
    }
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        super.onStartCommand(intent, flags, startId);
        return START_STICKY; Service 被結束時會重啟並清空 Intent。
    }

    @Override
    public void onDestroy() { super.onDestroy(); }

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