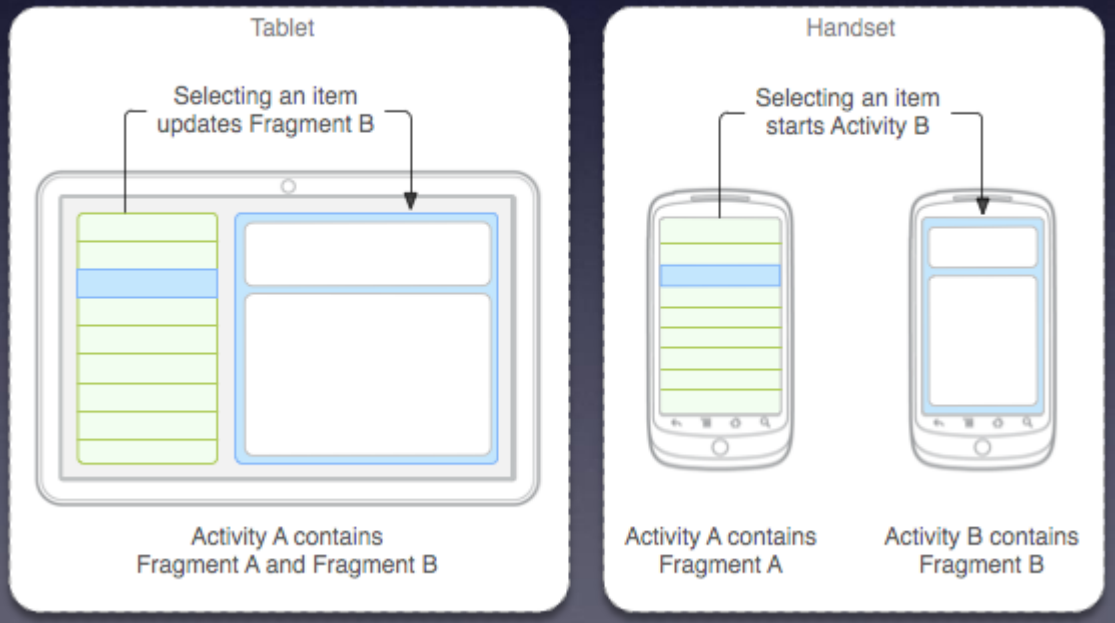
第十二章

FRAGMENT

什麼是FRAGMENT?

什麼是FRAGMENT?

- 早期Android只有Activity，並沒有針對不同大小螢幕設計出彈性架構
- Fragment便是針對彈性缺乏的Activity而設計的補強機制



什麼是FRAGMENT?

- Fragment可以讓開發人員維持如同在寫Activity一般的感覺
- Fragment的重點在於提供了一套View Stack的管理機制
- Fragment相較於Activity使用的資源更少
- 可以將Fragment當成是「輕量版」的Activity
- 未來Android推出的新功能，都將架構在Fragment上
  - ViewPager, GoogleMap, ActionBar, Navigation Drawer

學習內容

- Fragment的建立
- Fragment啟動另一個Fragment的方式
- Fragment之間的數值傳遞
- Fragment啟動其他Activity的方式

前置作業

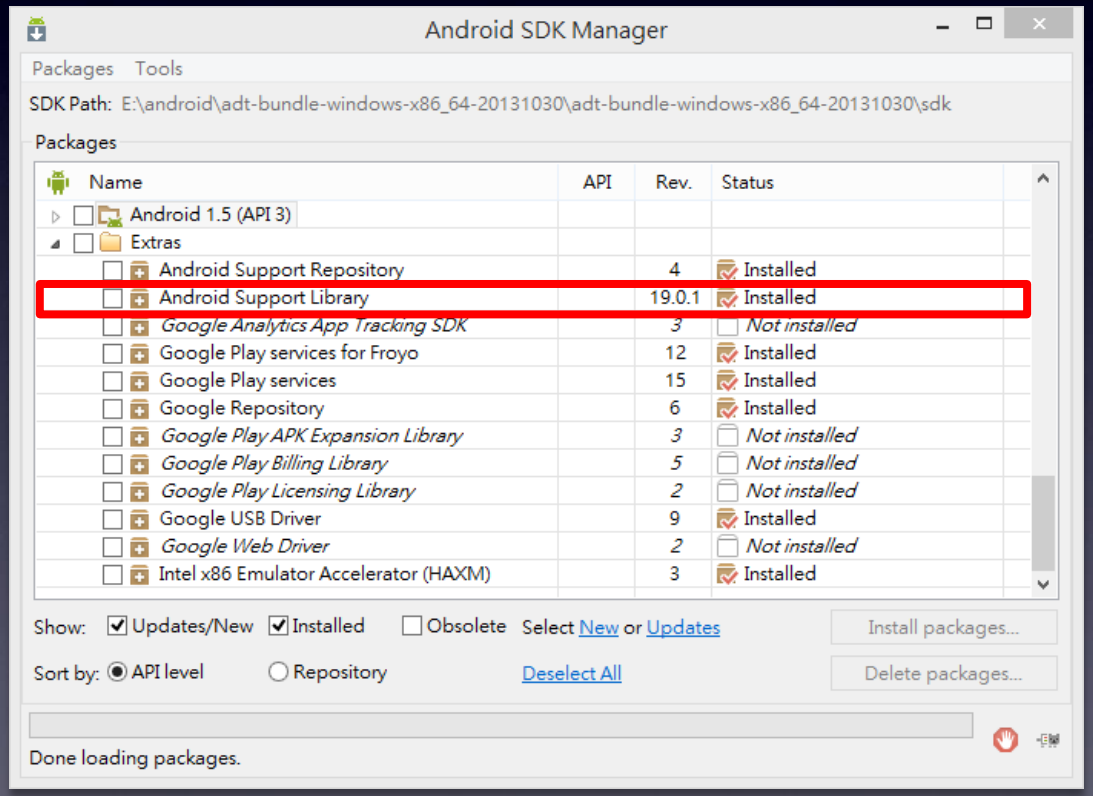
前置作業

- 因為Fragment屬於新版Android (3.0以上)的機制，所以原本舊版不支援

- 但Android為了讓Fragment被廣泛應用，所以增加了Support Library，讓舊版的Android也使用Fragment的機制

前置作業

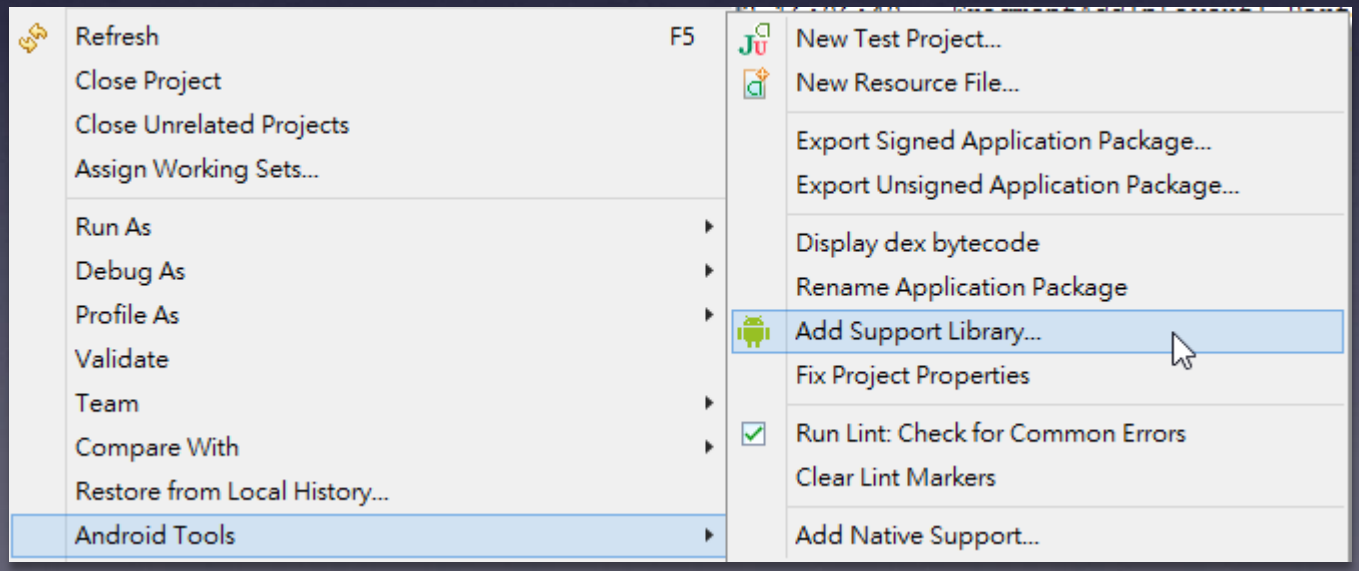
- 打開Android SDK Manager，確認Android Support Library有下載



前置作業

- 建立Android專案後，確認libs資料夾下有android-support-v4.jar

- 若沒有可對專案按下滑鼠右鍵→Android Tools→Add Support Library…



Fragment/FragmentAddInLayout
Fragment/FragmentAddInCode

FRAGMENT使用方法

FRAGMENT使用方法

- 使用步驟
  1. 建立Fragment class
  2. 設定Fragment使用的layout
  3. 建立FragmentActivity
  4. 設定FragmentActivity的layout
  5. 將Fragment設定在Activity上

FRAGMENT使用方法

- 首先Fragment class要繼承Fragment

```
public class SimpleFragment extends Fragment {
```

- 要確定使用的是Fragment是 **android.support.v4.app.Fragment**

FRAGMENT使用方法

- 寫Fragment的方式與Activity類似
- 建立UI，Override方法onCreateView()

```
public View onCreateView(LayoutInflater
inflater, ViewGroup container, Bundle
savedInstanceState) {
```

- onCreateView()的回傳值，設定layout

```
return inflater.inflate(R.layout.fragment_layout,
null);
```

- 上述方法就建立好了一個基本的Fragment

FRAGMENT使用方法

- Fragment是依附在Activity上，所以必須得將Fragment設定給Activity
- 如同setContentView()將View設定成Activity
- 有兩種方式可以讓Fragment設定給Activity
  - 在Activity用的layout的xml中宣告
  - 在Activity程式碼中撰寫

在LAYOUT的XML中宣告

- main.xml

```
<FrameLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <fragment
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       class="com.example.fragmentaddinlayout.SimpleFragment"/>
</FrameLayout>
```

在LAYOUT的XML中宣告

- 在layout的xml中以<fragment>開頭
- class="<fragment class>"
  屬性class要指定fragment的package + class name

```
<fragment
android:layout_width="match_parent“
android:layout_height="match_parent"
class="com.example.fragmentaddinlayout.SimpleFragment"
/>
```

在LAYOUT的XML中宣告

使用Fragment的Activity都必須得是FragmentActivity

```
public class MainActivity extends FragmentActivity {
```

- 使用setContentView(layout)將layout設定給Activity

```
setContentView(R.layout.main);
```

在程式碼中撰寫

- 也可以使用程式碼將Fragment設定到Activity中

- 首先，layout xml中不用寫<fragment>，寫個FrameLayout或RelativeLayout並且設定id

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    android:id="@+id/root"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
</FrameLayout>
```

在程式碼中撰寫

- Activity中使用程式將Fragment設定到id為root的Layout上

```
getSupportFragmentManager().
beginTransaction().
replace(R.id.root, new SimpleFragment())
.commit();
```

- 細節說明見下頁

在程式碼中撰寫

```
getSupportFragmentManager().
beginTransaction().
replace(R.id.root, new SimpleFragment())
.commit();
```

- 在FragmentActivity中可以藉由getSupportFragmentManager()取得管理器
- 管理器可以管理Fragment之間的切換

在程式碼中撰寫

```
getSupportFragmentManager().
beginTransaction().
replace(R.id.root, new SimpleFragment())
.commit();
```

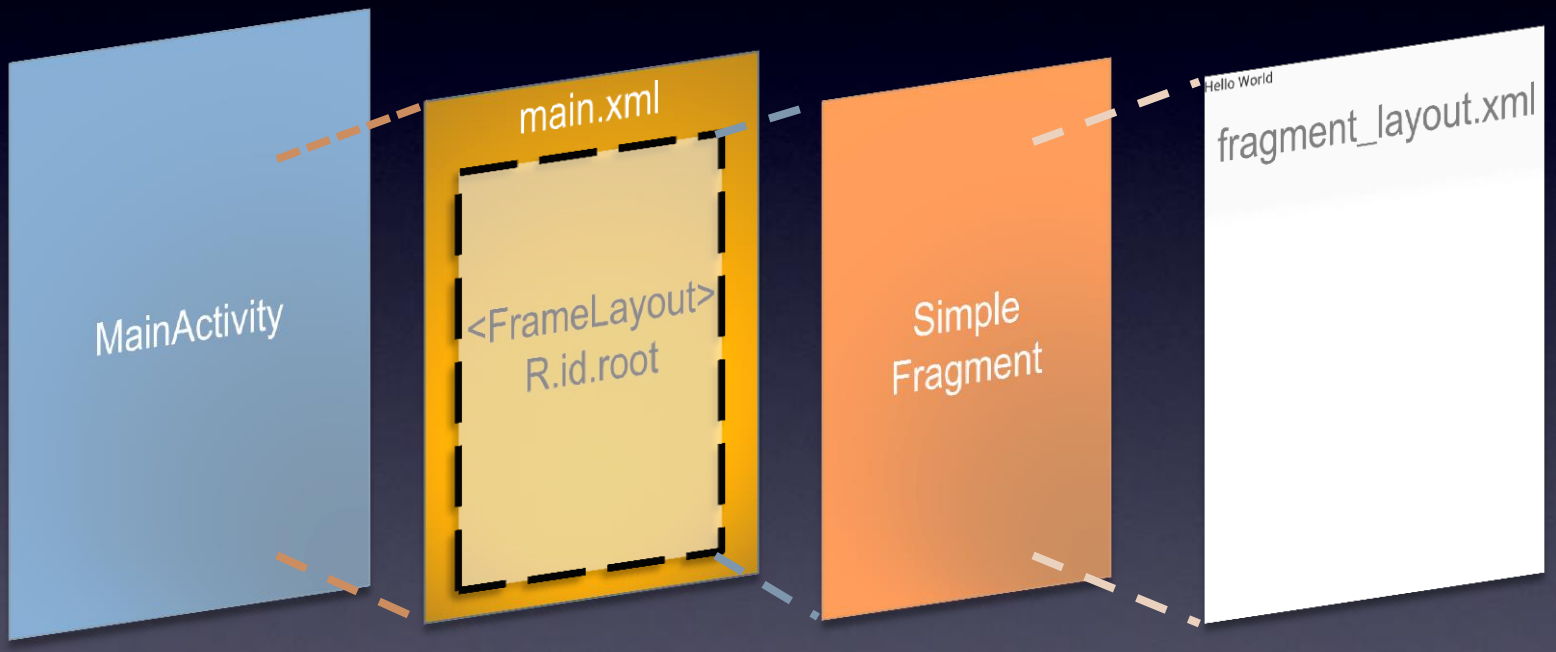
- 表示要進行Fragment的轉換工作
- 基本上轉換工作都是寫在beginTransaction()和commit()之間

在程式碼中撰寫

```
getSupportFragmentManager().
beginTransaction().
replace(R.id.root, new SimpleFragment())
.commit();
```

- replace()是Fragment的轉換方式
- 表示SimpleFragment建立後貼到R.id.root這view上
- 若R.id.root已經有Fragment，原本的Fragment將會消除，以新的替代

剖面圖

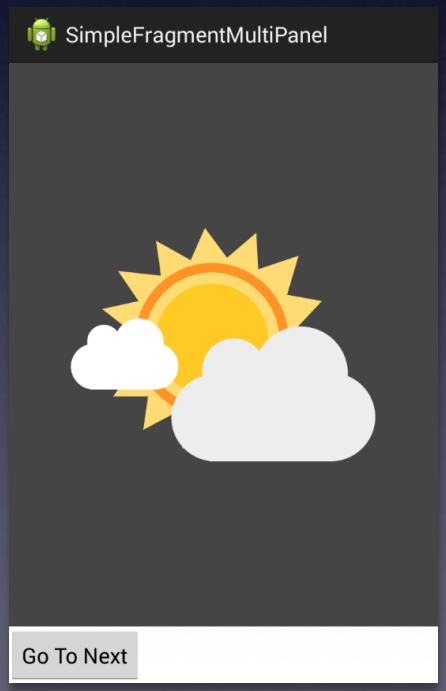

MainActivity
main.xml
<FrameLayout>
R.id.root
Simple
Fragment
Hello World
fragment_layout.xml

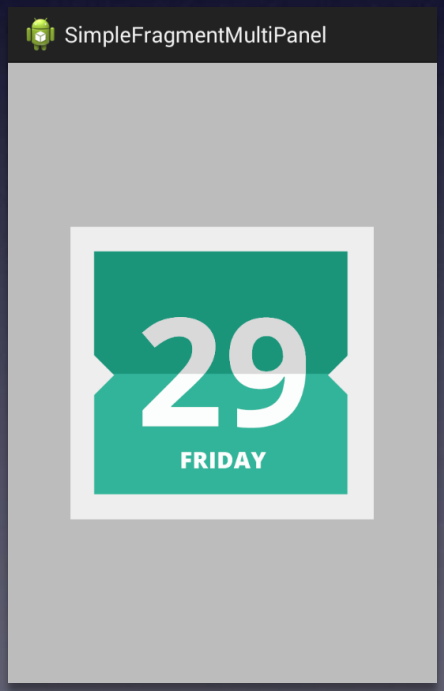
Fragment/SimpleFragmentMultiPanel

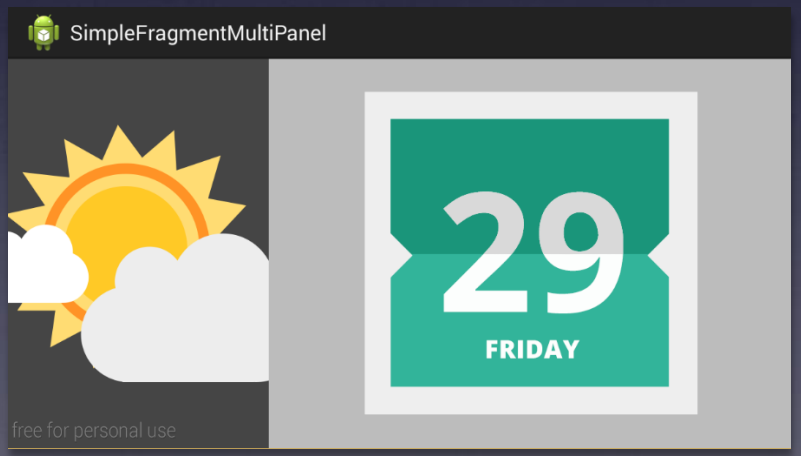
如何製作多版面?

如何製作多版面?

- 要做到裝置直向時顯示左圖，且可以按鈕切換畫面
- 橫向時顯示右圖，該怎麼做呢？

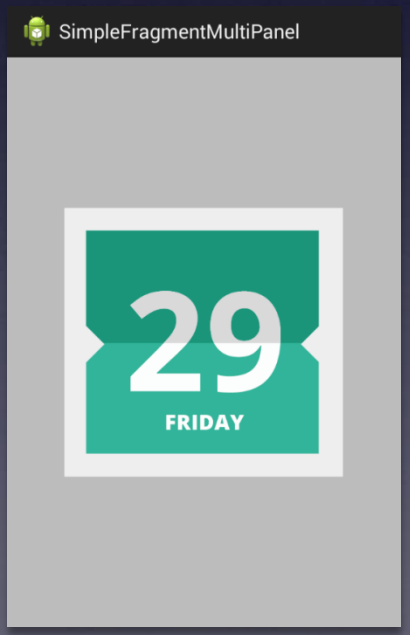






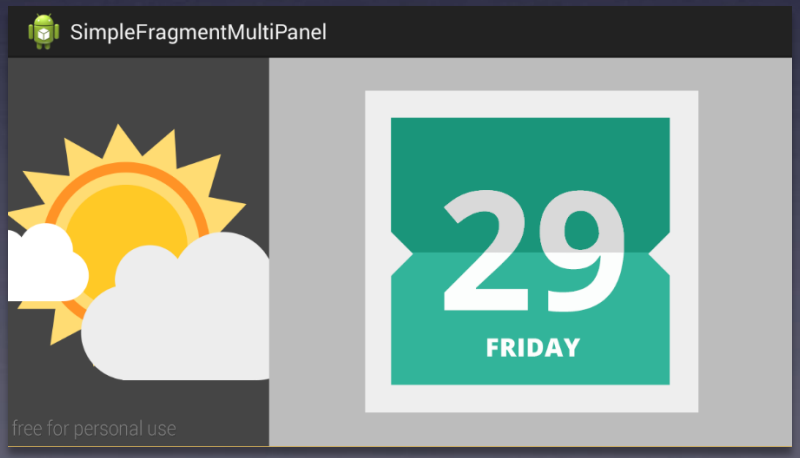
原理

- 裝置直向(portrait)時，是兩個activity，分別為MainActivity與SecondActivity
- 點下Go To Next時，以intent切換到SecondActivity
- 兩個Activity中分別包含Fragment1以及Fragment2



原理

- 裝置橫向(landscape)時，藉由自動選取資源的機制，讓Activity選擇水平的layout
- 水平的layout分為左右兩塊區域
- 左右分別貼上Fragment1與Fragment2



原理

- 所以不論裝置方向怎麼變，Fragment都保持獨立性以便重複使用

- 依照十一章的教學內容，可以知道裝置水平時activity所要用的layout必須得擺在res/layout-land資料夾下

建立步驟

1. 建立Fragment class
2. 設定Fragment使用的layout
3. 建立FragmentActivity
4. 設定FragmentActivity的layout
5. 將Fragment設定在Activity上

1.建立FRAGMENT CLASS

- 建立兩個Fragment，分別是Fragmen1.java與Fragment2.java

2.建立FRAGMENT的LAYOUT

- 在res/layout資料夾下建立fragment1.xml與fragment2.xml
- 不管裝置怎麼旋轉，Fragment本身負責呈現的樣式都不會改變，排列的方式交給Activity負責
- 記得在Fragment的onCreateView()設定layout

```
public class Fragment1 extends Fragment {
        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup
                container, Bundle savedInstanceState) {
                return inflater.inflate(R.layout.fragment1, null);
        }
}
```

3. 建立FRAGMENTACTIVITY

- 建立MainActivity與SecondActivity，Activity要繼承FramgmentActivity

```
public class MainActivity extends FragmentActivity {
        ……

public class SecondActivity extends FragmentActivity {
        ……
```

4. 設定FRAGMENTACTIVITY的LAYOUT

- 在res/layout下，設定activity_main.xml與activity_second.xml
- 在res/layout-land下，建立裝置水平時的layout，activity_main.xml。給MainActivity使用

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >
    <FrameLayout
        android:id="@+id/fragment1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="2"></FrameLayout>
    <FrameLayout
        android:id="@+id/fragment2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"></FrameLayout>
</LinearLayout>
```

4.設定FRAGMENTACTIVITY的LAYOUT

- 分別設定MainActivity和SecondActivity的layout
- 差別在於當裝置轉向時，MainActivity會更換layout

```
public class MainActivity extends FragmentActivity {
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                ……

public class SecondActivity extends FragmentActivity {
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_second);
                ……
```

5.將FRAGMENT設定在ACTIVITY上

```
if(getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE) {
```

- getConfiguration()可以得到目前App的裝置資訊 orientation就是其中一項

- 上述表示，如果裝置是橫向的(landscape)
  - ORIENTATION_LANDSCAPE

5.將FRAGMENT設定在ACTIVITY上

- 在橫向狀態下

```
getSupportFragmentManager().
beginTransaction().replace(R.id.fragment1,
new Fragment1()).commit();

getSupportFragmentManager().
beginTransaction().replace(R.id.fragment2,new
Fragment2()).commit();
```

- 將Fragment1及Fragment2設定到R.id.fragment1以及R.id.fragment2兩個View上

5.將FRAGMENT設定在ACTIVITY上

- 在垂直狀態下

```
getSupportFragmentManager().beginTransaction().replace
(R.id.fragment1, new Fragment1()).commit();
```

- 將Fragment1設定到R.id.fragment1的View上，因為Fragment2是由另一個Activity負責

5.將FRAGMENT設定在ACTIVITY上

```
Button button = (Button) findViewById(R.id.btn_next);
button.setOnClickListener(new OnClickListener() {
	@Override
	public void onClick(View v) {
		Intent intent = new Intent(MainActivity.this,
				SecondActivity.class);
		startActivity(intent);
	}
});
```

- 找出裝置垂直時的button並且設定onClickListenr
- 當按下Button後，切換至SecondActivity

5.將FRAGMENT設定在ACTIVITY上

- SecondActivity.java

```
if (getResources().getConfiguration().orientation ==
Configuration.ORIENTATION_LANDSCAPE) {
      finish();
      return;
}
getSupportFragmentManager().beginTransaction().replace(R.
id.fragment2, new Fragment2()).commit();
```

5.將FRAGMENT設定在ACTIVITY上

```
if(getResources().getConfiguration().orientation ==
Configuration.ORIENTATION_LANDSCAPE) {
      finish();
      return;
}
```

- 若裝置橫向時，是不會出現SecondActivity
- 上述程式碼檢查目前裝置方向，如果是橫向就把SecondActivity停止

5.將FRAGMENT設定在ACTIVITY上

```
getSupportFragmentManager().beginTransaction(
).replace(R.id.fragment2, new
Fragment2()).commit();
```

- 若執行上述程式碼，表示SecondActivity是垂直狀態，就把Fragment2設定給R.id.fragment2的View上

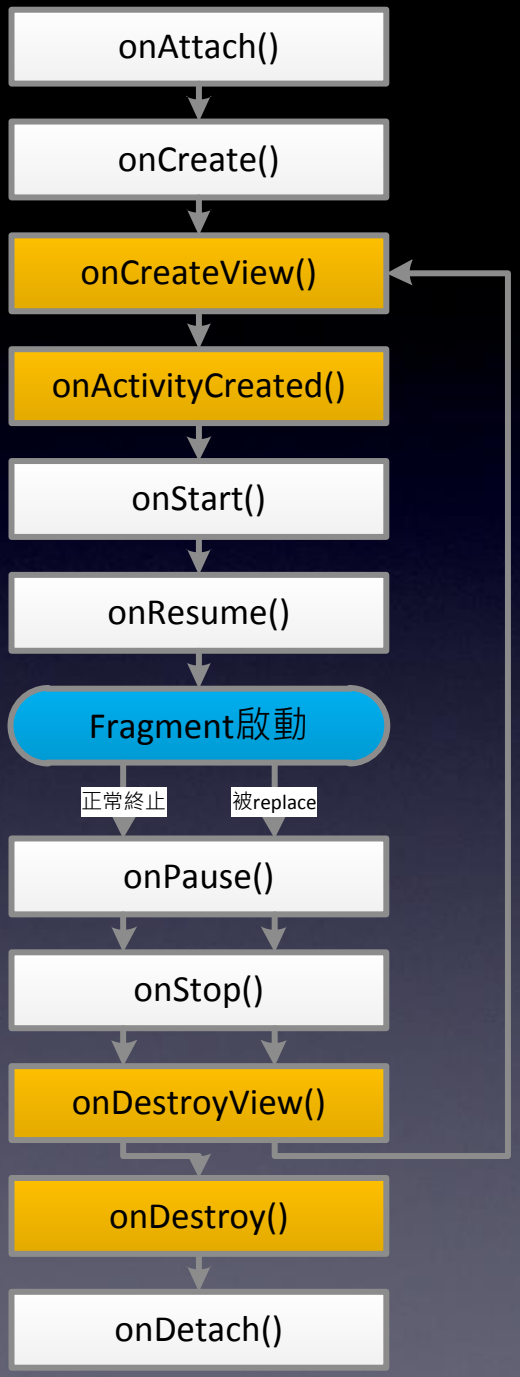
Fragment/FragmentLifeCycle

FRAGMENT生命週期

FRAGMENT生命週期

- Fragment類似於Activity有自己的生命週期

- 開發者可以藉由生命週期來建立物件、findViewById、設定物件的OnClickListener以及釋放資源


onAttach()
onCreate()
onCreateView()
onActivityCreated()
onStart()
onResume()
Fragment啟動
正常終止
被replace
onPause()
onStop()
onDestroyView()
onDestroy()
onDetach()

FRAGMENT生命週期

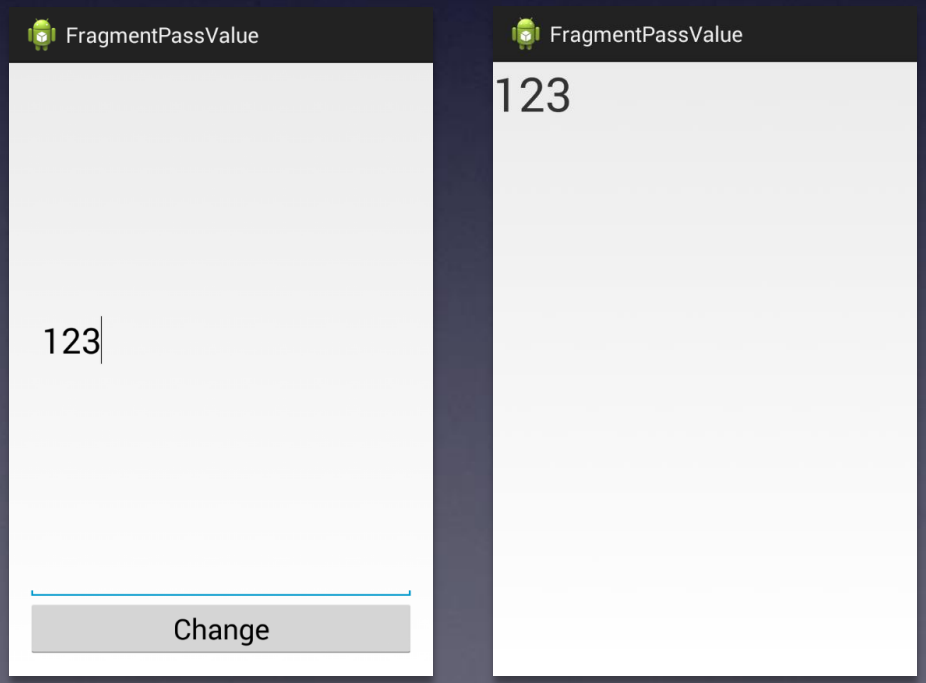
- onCreateView()
  - 建立Fragment使用的View
- onActivityCreated()
  - 使用findViewById()找出Fragment中需要操作的View
- onDestroyView()
  - 釋放屬於View的資源在釋放所有資源
- onDestroy()
  - 釋放所有資源

Fragment/FragmentPassValue

FRAGMENT切換與傳值

FRAGMENT切換與傳值

- Activity之間切換時可以藉由Intent來傳值
- Fragment要透過Bundle來傳遞
- 左圖是FragmentOne，按下Change的按鈕，會將輸入的值帶到FragmentTwo並呈現在畫面上



ACTIVITY部分

```
getSupportFragmentManager().
beginTransaction().
replace(android.R.id.content, new
FragmentOne()).commit();
```

- 首先啟動FragmentOne
- Fragment可以直接加入Android系統給予Activity的View，id是android.R.id.content

FRAGMENT ONE部分

```
Button changeToNextButton = (Button)
getActivity().findViewById(R.id.btn_change
);
```

- 在FragmenOne中，按下按鈕要傳遞EditText內的值到FragmenTwo

- 在onActivtyCreated()可以使用findViewById()來找Button

FRAGMENT ONE部分

```
FragmentTwo two = new FragmentTwo();
Bundle data = new Bundle();
data.putString("value", value);
two.setArguments(data);
getFragmentManager().beginTransaction().replace(android.R.id.content, two).commit();
```

- 首先先new一個FragmentTwo
- 建立Bundle負責儲存要帶到FragmentTwo的數值

FRAGMENT ONE部分

```
FragmentTwo two = new FragmentTwo();
Bundle data = new Bundle();
data.putString("value", value);
two.setArguments(data);
getFragmentManager().beginTransaction().repla
ce(android.R.id.content, two).commit();
```

- 每個Fragment都有名為setArguments()的方法可以儲存Bundle

FRAGMENT ONE部分

```
FragmentTwo two = new FragmentTwo();
Bundle data = new Bundle();
data.putString("value", value);
two.setArguments(data);
getFragmentManager().beginTransaction().replace(android.R.id.content, two).commit();
```

- 在Fragment中可以呼叫getFragmentManager()就可以取得Fragment管理器
- 接下來進行Transaction讓FragmentTwo replace FragmentOne目前的位置

FRAGMENT TWO部分

```
TextView text = (TextView)
getActivity().findViewById(R.id.value);
Bundle data = getArguments();
String strValue = data.getString("value");
text.setText(strValue);
```

- 在FragmentTwo可以藉由getArguments()取得FragmentOne帶過來的Bundle
- 藉由相同的key可以從Bundle中取得數值

Fragment/FragmentPassValueWithStack

FRAGMENT STACK

問題

- 上節範例在FragmentTwo按下Back鍵沒有回到FragmentOne，App直接結束
- 按Back鍵應該把現在的Fragment收起來，然後回到上一個Fragment
- 必須得使用到Fragment Stack

FRAGMENT STACK

```
getFragmentManager().beginTransaction().
replace(android.R.id.content,
two).addToBackStack(null).commit();
```

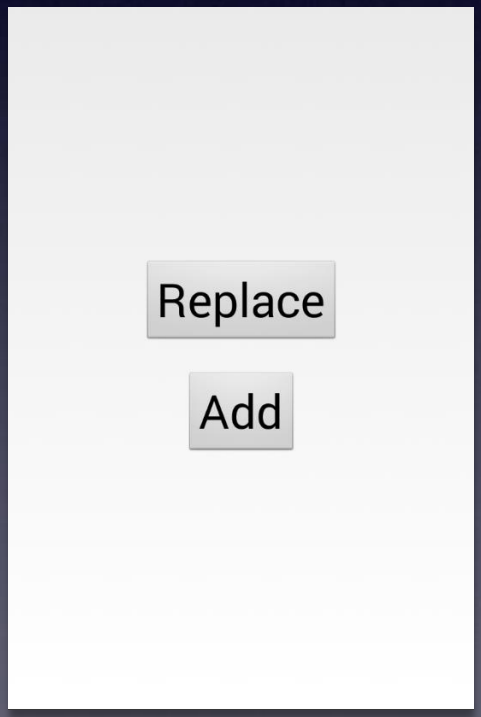
- 上列程式碼是FragmentOne切換到FragmentTwo
- 在transaction增加一個addToBackStack(null)

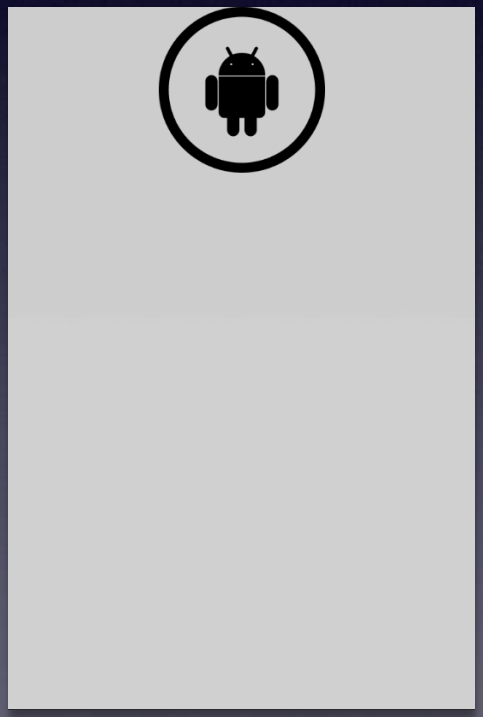
Fragment/FragmentOperation

FRAGMENT ADD與REPLACE的差別

ADD REPLACE

- Fragment間的切換可以使用replace()或是add()，兩個之間的差別在哪？
- 圖左為FragmenOne，圖右為FragmentTwo

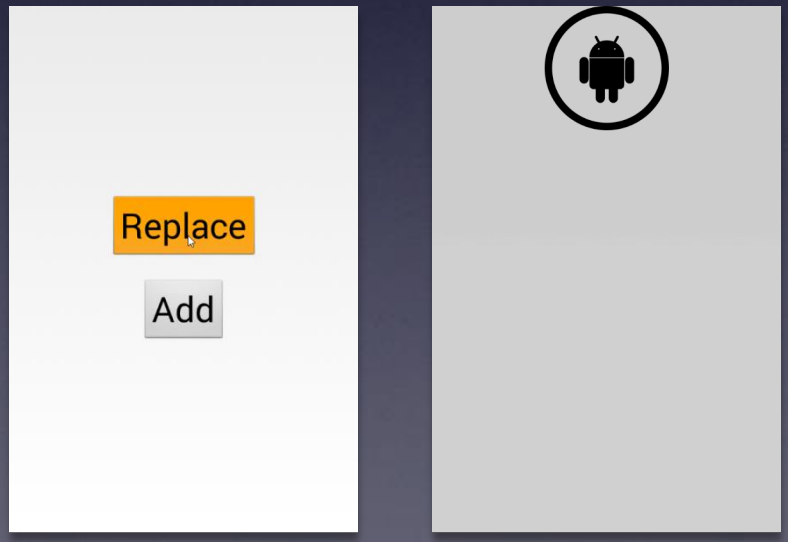




使用REPLACE

```
getFragmentManager().beginTransaction().replace(android.R.id.content, new FragmentTwo()).addToBackStack(null).commit();
```

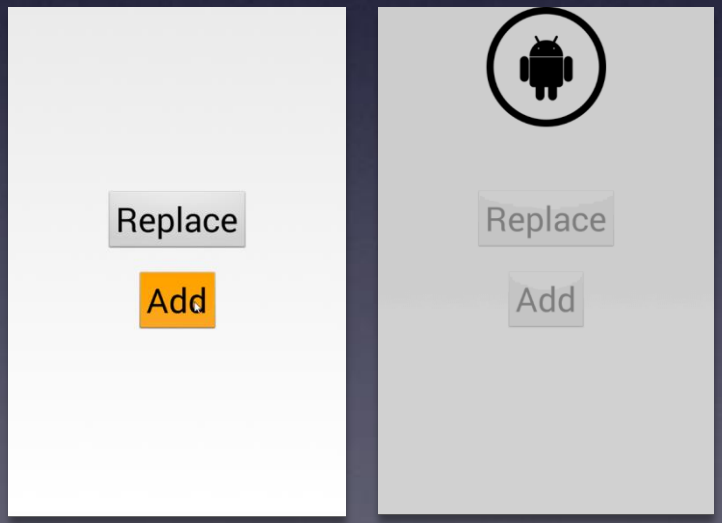
- 使用replace，一個view上一次只有一個Fragment



使用ADD

```
getFragmentManager().beginTransaction().add(android.R.id.content, new FragmentTwo()).addToBackStack(null).commit();
```

- 使用add則新加入的Fragment不會取代原本的
- 而且下層的Fragment依然可以操作

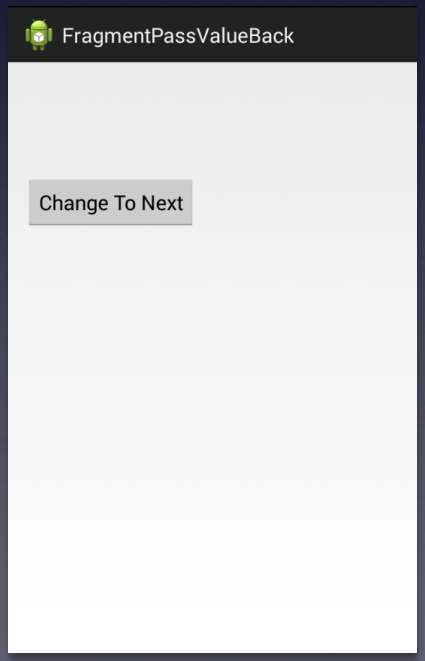


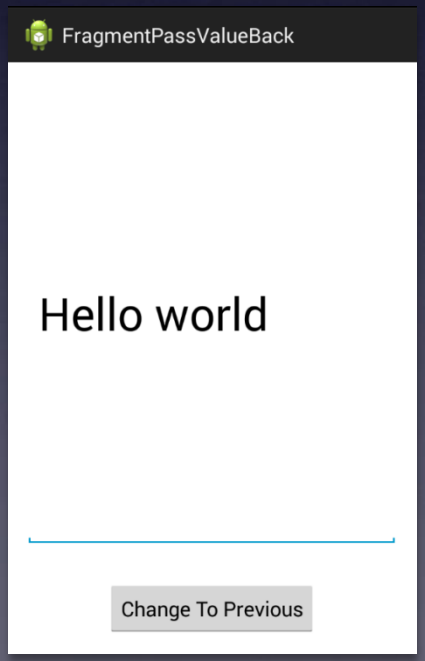
Fragment/FragmentPassValueBack

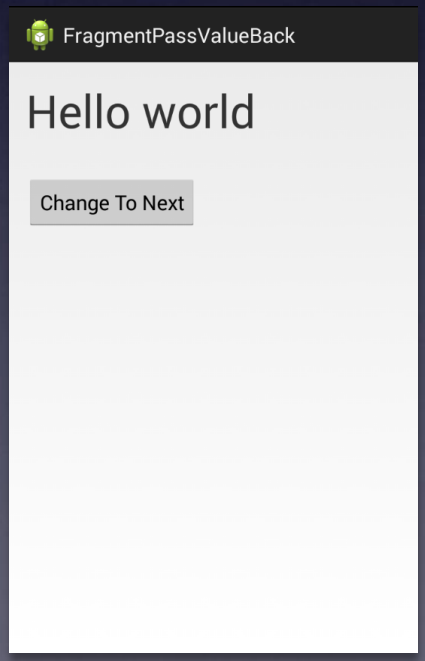
FRAGMENT傳值回來

FRAGMENT傳值回來

- 若現在是FragmentOne啟動FragmentTwo
- FragmentTwo輸入的數值要顯示在FragmentOne該怎麼做？







ACTIVITY部分

```
getSupportFragmentManager().beginTransaction().replace(android.R.id.content, new FragmentOne(), "One").commit();
```

- Activity要啟動FragmentOne時的程式碼
- replace()這次帶了三個參數
  - 第三個參數: 索引值，以便我們後續可以藉由索引值找到這個Fragment

FRAGMENT ONE的部分

```
mResult = (TextView) getActivity().findViewById(R.id.result);
Button changeToNext = (Button)
getActivity().findViewById(R.id.btn_next);
changeToNext.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                getFragmentManager().beginTransaction().
                        add(android.R.id.content, new FragmentTwo()).
                        addToBackStack(null).commit();
        }
});
```

- mResult負責顯示FragmentTwo帶回來的值

FRAGMENT ONE的部分

```
mResult = (TextView) getActivity().findViewById(R.id.result);
Button changeToNext = (Button)
getActivity().findViewById(R.id.btn_next);
changeToNext.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                getFragmentManager().beginTransaction().
                        add(android.R.id.content, new FragmentTwo()).
                        addToBackStack(null).commit();
        }
});
```

- 這邊不是使用replace()而是使用add()啟動FragmentTwo
- 這樣的狀況就是FragmentTwo將會蓋在FragmentOne上方，但FragmentOne不會消失

FRAGMENT ONE的部分

```
mResult = (TextView) getActivity().findViewById(R.id.result);
Button changeToNext = (Button)
getActivity().findViewById(R.id.btn_next);
changeToNext.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                getFragmentManager().beginTransaction().
                        add(android.R.id.content, new FragmentTwo()).
                        addToBackStack(null).commit();
        }
});
```

- 記得要addToBackStack(null)，才能夠順利地從FragmentTwo回到FragmentOne

FRAGMENT ONE的部分

```
public void setResult(String value) {
    mResult.setText(value);
}
```

- FragmentOne新建立一個方法，這是讓FragmentTwo要切回FragmentOne時呼叫的
- 讓FragmentTwo的數值由參數設定到FragmentOne的TextView – mResult

FRAGMENT TWO部分

```
FragmentOne oneFragment = (FragmentOne)
getFragmentManager().findFragmentByTag("One");
oneFragment.setResult(strContent);
getFragmentManager().popBackStack();
```

- 上述程式碼是在FragmentTwo按下Change To Previous時做的
- 使用FragmentManager的findFragmentByTag()找到FragmentOne
  - 參數就是在Activity啟動FragmentOne時的索引值

FRAGMENT TWO部分

```
FragmentOne oneFragment = (FragmentOne)
getFragmentManager().findFragmentByTag("One");
oneFragment.setResult(strContent);
getFragmentManager().popBackStack();
```

- 還記得在FragmentOne新增的方法嗎？

- 在FragmentTwo找到FragmentOne後就可以呼叫FragmentOne的方法，將數值帶回FragmentOne

FRAGMENT TWO部分

```
FragmentOne oneFragment = (FragmentOne)
getFragmentManager().findFragmentByTag("One");
oneFragment.setResult(strContent);
getFragmentManager().popBackStack();
```

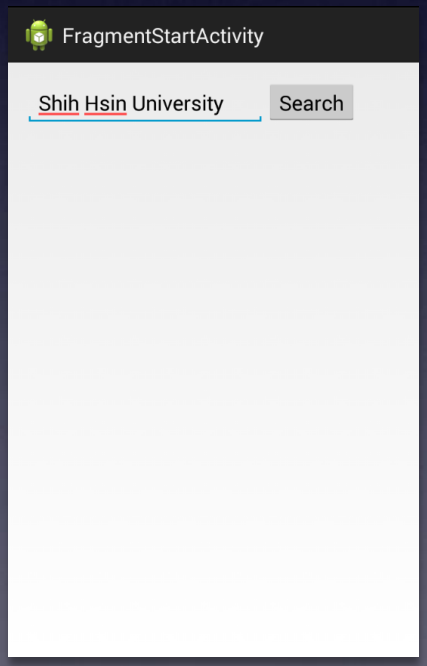
- popupBackStack() 表示清除目前畫面上的Fragment，也就是FragmentTwo
- 因為之前啟動FragmentTwo是用add()的方式，讓FragmentTwo壓在FragmentOne上
- 清除FragmentTwo自然就可以回到FragmentOne了

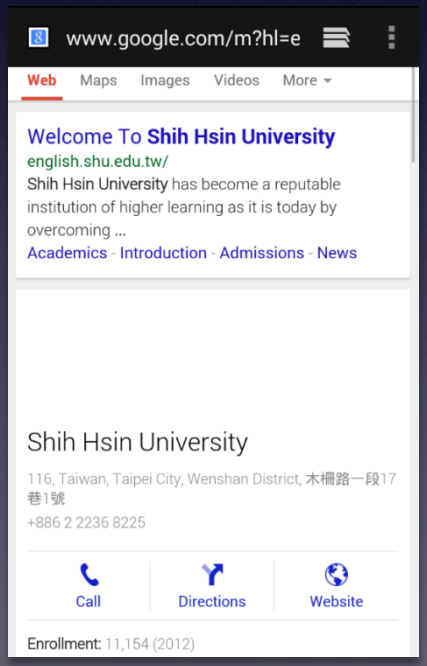
Fragment/FragmentStartActivity

FRAGMENT啟動ACTIVITY

FRAGMENT啟動ACTIVITY

- Fragment也可以與其他Activity溝通，與Activity啟動其他Activity相同
- 本例子是在Fragment中啟動瀏覽器搜尋特定關鍵字





FRAGMENT啟動ACTIVITY

```
Intent intent = new
Intent(Intent.ACTION_WEB_SEARCH);
intent.putExtra(SearchManager.QUERY, keyword);
startActivity(intent);
```

- 在Fragment之中，要啟動其他Activity直接呼叫 startActivity()

重點整理與更多資料

重點整理

- 使用Fragment的Activity一定要是FragmentActivity

- 在FragmentActivity下取得FragmentManager的方式為getSupportFragmentManager()
- 在Fragment下取得FragmentManager的方式為getFragmentManager()

- 進行Fragment切換只要用getFragmentManager().beginTransaction().replace(view id, fragment).commit()

重點整理

- 要讓Fragment切換後可以回到上一頁，要使用addToBackStack(null)
- Fragment之間傳遞數值使用setArguments(Bundle)
- 要找到FragmentManager中的Fragment，可以使用findFragmentByTag()
- Fragment中啟動Activity的方法startActivity()

更多資料

- http://developer.android.com/guide/components/fragments.html

- http://developer.android.com/guide/practices/tablets-and-handsets.html

練習

- 製作以下畫面，裝置垂直時
  - 第一頁是ListView
  - 點下選項後，切入第二頁
- 畫面水平時，一次顯示兩個畫面，且點選左方選單，右方畫面會跟著變更