行動裝置互動系統設計與應用

授課老師：黃朝曦

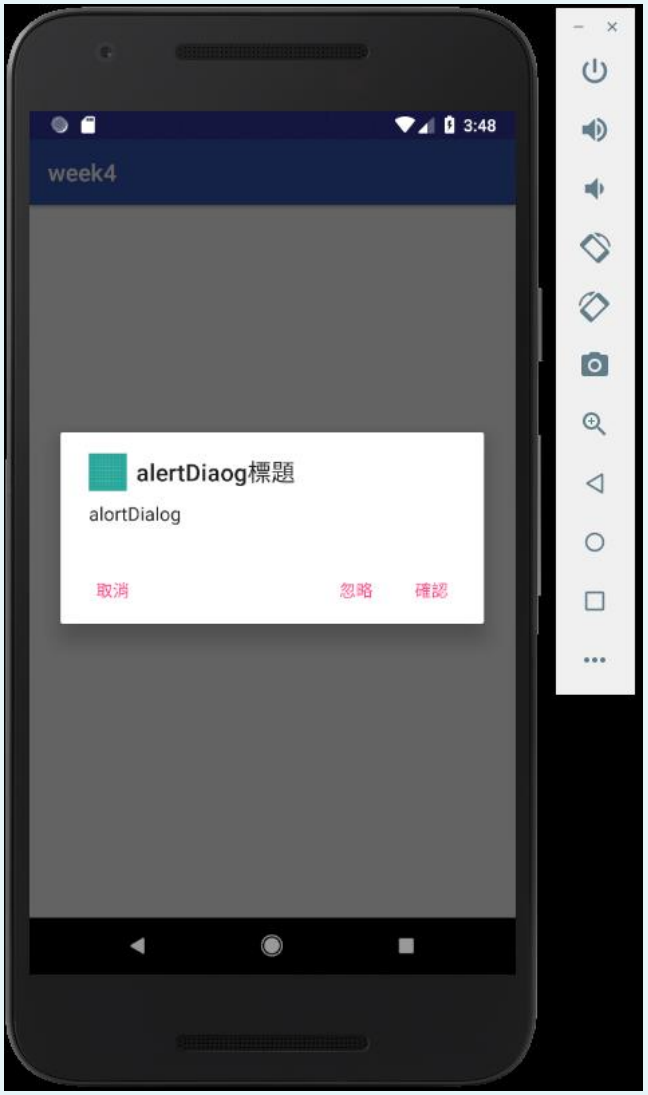
國立宜蘭大學 National Ilan University NIU

單元目標

- AlertDialog(對話方塊)
- 時間日期介面元件
- 時間日期介面元件對話盒
- 小專題-點餐系統

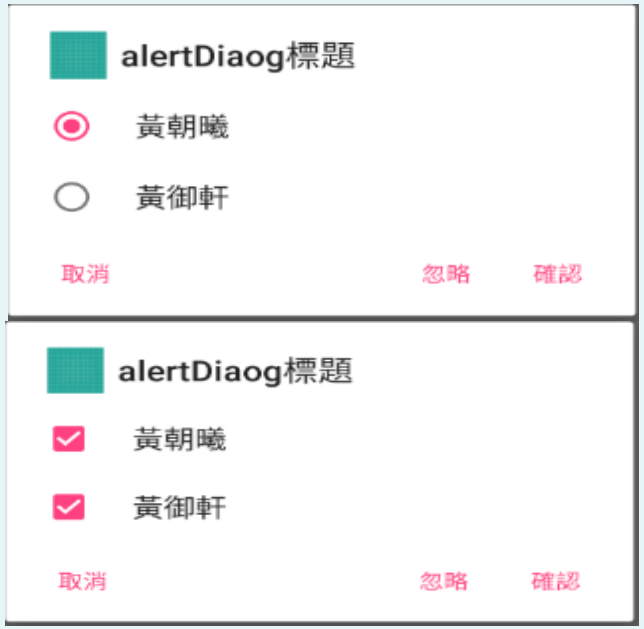
AlertDialog
(對話方塊)

AlertDialog範例



AlertDialog

- AlertDialog元件顯示訊息後不會自動消失，與Toast元件不同可以與使用者進行互動。
- 最多可以顯示**3個**按鈕，最少是**0個**按鈕，也可以顯示**訊息**、**列表**，或者是**一個xml** 。



AlertDialog基本型態

設定值	說明
create()	創建對話框
setTitle()	標題
setIcon()	圖示
setMessage()	內容
setView()	自訂樣式
setItems()	列表內容
setSingleChoiceItems()	單選列表
setMultiChoiceItems()	多選列表
setNeutralButton()	Ignore按鈕(中間)
setPositiveButton()	Yes按鈕(右間)
setNegativeButton()	No按鈕(左間)
show()	顯示對話框

開始打程式 MainActivity.java

- 步驟1. 建立AlertDialog的對話方塊

```
new AlertDialog.Builder(this)
```

- 步驟2. 設定標題名稱

```
.setTitle("alertdialog標題")
```

- 步驟3. 設定圖示(內建圖片)

```
.setIcon(R.drawable.ic_launcher)
```

- 步驟4. 設定內容文字

```
.setMessage("alertdialog內容")
```

開始打程式 MainActivity.java

- 步驟5. 設定確認按鈕

```
.setPositiveButton("確定", new DialogInterface.OnClickListener() {

    @Override
    public void onClick(DialogInterface dialog, int which) {
        // TODO Auto-generated method stub

    }
})
```

- 步驟6. 顯示對話框

```
.show();
```

AlertDialog對話盒的功能

AlertDialog對話盒的功能是通知使用者某些訊息，這個訊息可以單純只是一些資訊、或是警告訊息或錯誤訊息，甚至是詢問使用者一個問題，然後使用者再以對話盒下方的按鈕進行回應。AlertDialog對話盒中的按鈕個數和按鈕上所顯示的文字可以由程式進行控制，但是最多只能有3個按鈕。

建立AlertDialog對話盒的方法

建立AlertDialog對話盒有二種方法：

1. 利用AlertDialog.Builder類別
2. 利用AlertDialog類別

使用AlertDialog.Builder類別建立AlertDialog對話盒

Step 1. 建立一個AlertDialog.Builder類別的物件。建立物件的同時必須指定它的擁有者，請參考下列程式碼範例：

```
AlertDialog.Builder altDlgBuilder =
    new AlertDialog.Builder(Activity類別名稱.this);
```

以上的程式碼是建立一個屬於主程式類別的AlertDialog.Builder物件，物件名稱叫做altDlgBuilder。

使用AlertDialog.Builder類別建立AlertDialog對話盒

Step 2. 設定對話盒的標題、訊息、圖示，另外我們也可以將Cancelable屬性設定為false，它讓使用者無法利用手機按鍵中的回復鍵離開對話盒：

```
altDlgBuilder.setTitle("AlertDialog");
altDlgBuilder.setMessage("AlertDialog範例");
altDlgBuilder.setIcon(android.R.drawable.ic_dialog_info
);
altDlgBuilder.setCancelable(false);
```

使用AlertDialog.Builder類別建立AlertDialog對話盒

Step 3. 加上按鈕。AlertDialog對話盒可以視需要加上按鈕，當然也可以不加，只是沒有按鈕的話我們必須把上一個步驟中的Cancelable屬性設定為true，這樣使用者才可以利用手機按鍵中的回復鍵離開對話盒。可以加到AlertDialog對話盒中的按鈕有3個，分別為PositiveButton、NegativeButton、和NeutralButton，它們分別用不同的方法加入，詳細程式碼請參考書上說明。

Step 4. 呼叫show()方法顯示對話盒。

使用AlertDialog類別建立AlertDialog對話盒

如果嘗試建立一個AlertDialog的物件將會出現錯誤訊息，原因在於AlertDialog類別把建構式定義成protected，所以我們無法直接產生它的物件，解決方法是要用繼承的方式，也就是我們要自己新增一個繼承自AlertDialog的類別，然後在該類別中建立一個建構式呼叫AlertDialog類別的建構式，然後我們就可以利用自己的類別產生一個AlertDialog物件。

使用AlertDialog類別建立AlertDialog對話盒

以下的程式碼就是我們自己新增的類別，名稱叫做MyAlertDialog。它的程式碼很簡單，第一它是繼承AlertDialog類別，第二是它只有一個public的建構式，而且該建構式就只有呼叫AlertDialog類別的建構式，沒有其它的程式碼。

```
package …
import …

public class MyAlertDialog extends AlertDialog {
        public MyAlertDialog(Context context) {
                super(context);
        }
}
```

使用AlertDialog類別建立AlertDialog對話盒

建立好MyAlertDialog類別之後，就可以利用以下的步驟來產生AlertDialog對話盒：

Step 1. 建立一個MyAlertDialog類別的物件。建立物件的同時必須指定它的擁有者，請參考下列程式碼範例：

```
MyAlertDialog myAltDlg =
        new MyAlertDialog(主程式類別.this);
```

以上的程式碼是建立一個屬於主程式類別的MyAlertDialog物件，物件名稱叫做myAltDlg。

使用AlertDialog類別建立AlertDialog對話盒

Step 2. 設定對話盒的標題、訊息、圖示，另外我們還要將Cancelable屬性設定為false，它讓使用者無法利用手機按鍵中的回復鍵離開對話盒。

```
myAltDlg.setTitle("AlertDialog");
myAltDlg.setMessage("使用MyAlertDialog類別產生");
myAltDlg.setIcon(android.R.drawable.ic_dialog_info);
myAltDlg.setCancelable(false);
```

在MyAlertDialog類別的程式碼中除了建構式之外，我們並沒有定義任何其它方法，因此以上程式碼用到的方法都是繼承AlertDialog類別而來。

使用AlertDialog類別建立AlertDialog對話盒

Step 3. 加上按鈕。加上按鈕的方法和AlertDialog.Builder類別的方法不一樣。AlertDialog類別是利用引數來決定要加上Positive、Negative、或是Neutral按鈕，另外這一次我們也把按鈕的OnClickListener物件先建立好，再傳給建立按鈕的方法。

```
myAltDlg.setButton(DialogInterface.BUTTON_POSITIVE,
        "是", altDlgOnClkPosiBtnLis);
myAltDlg.setButton(DialogInterface.BUTTON_NEGATIVE,
        "否", altDlgOnClkNegaBtnLis);
myAltDlg.setButton(DialogInterface.BUTTON_NEUTRAL,
        "取消", altDlgOnClkNeutBtnLis);
```

使用AlertDialog類別建立AlertDialog對話盒

以下程式碼是建立對話盒中的按鈕的OnClickListener物件

```
private  DialogInterface.OnClickListener altDlgPositiveBtnOnClk = new
                                                DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
                // 按下PositiveButton後要執行的程式碼
        }
};
private  DialogInterface.OnClickListener altDlgNegativeBtnOnClk = new
                                                DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
                // 按下NegativeButton後要執行的程式碼
        }
};
private  DialogInterface.OnClickListener altDlgNeutralBtnOnClk= new
                                                DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
                // 按下NeutralButton後要執行的程式碼
        }
};
```

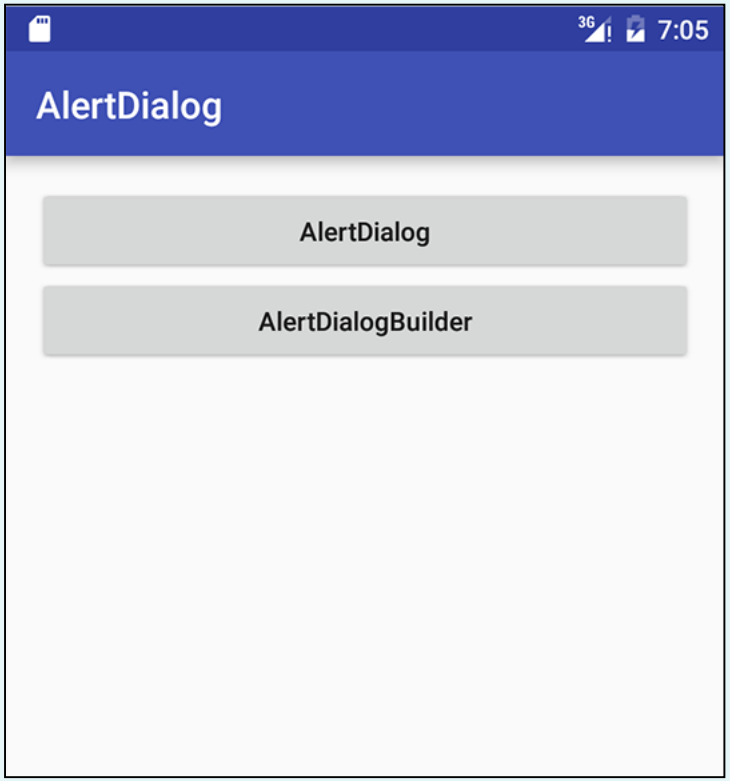
使用AlertDialog類別建立AlertDialog對話盒

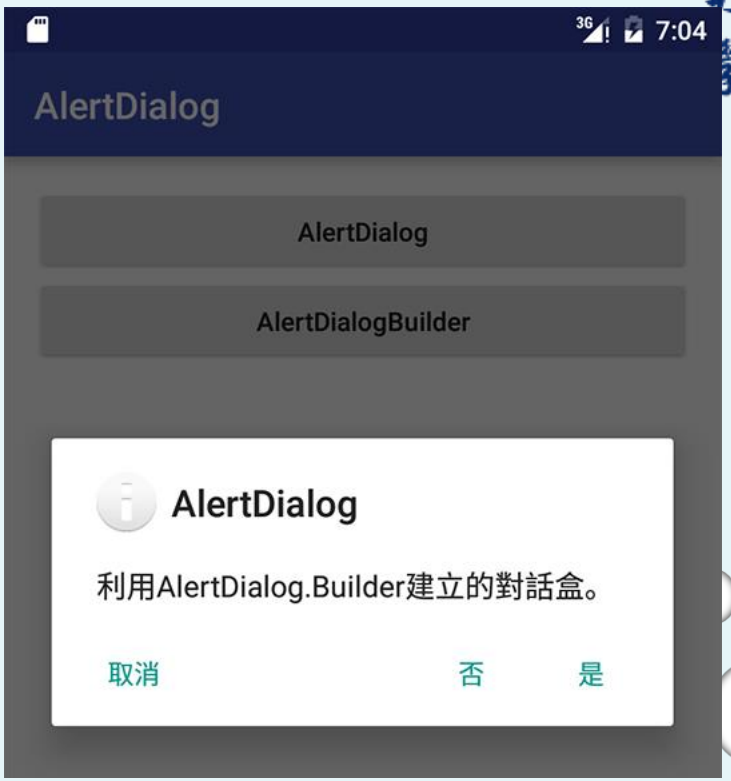
Step 4.　呼叫show()方法顯示對話盒

```
myAltDlg.show();
```

範例程式

有關介面佈局檔和程式檔請參考書上說明。





自訂Dialog對話盒

使用Dialog類別自訂對話盒

自訂對話盒必須使用Dialog類別，首先必須寫好一個對話盒專用的介面佈局檔，其中定義了在對話盒中用到的所有介面元件和它們的編排方式，我們學過的所有介面元件和編排模式都可以用在對話盒的介面佈局檔中。

接著在程式碼中建立一個Dialog類別的物件，然後把上述的對話盒介面佈局檔載入到該對話盒物件中，並設定好它的標題和其它相關屬性以及事件處理程序，最後把該對話盒顯示出來。

使用Dialog類別自訂對話盒

Step 1. 設計好對話盒使用的介面佈局檔，該檔案必須放在專案的res/layout資料夾中，我們可以在Android Studio左邊的專案檢視視窗中，用滑鼠右鍵點選程式專案的layout資料夾，然後從快顯功能表中選擇 New > Layout resource file。在出現的對話盒的File name欄位輸入檔案名稱，例如my_dlg（請注意，檔名只能夠使用小寫英文字母或是底線字元）。在Root element欄位輸入想要套用的編排模式，最後按下OK按鈕。新增的介面佈局檔會自動開啟在編輯視窗中，我們可以使用之前學過的所有介面元件和編排模式，來設計對話盒的介面佈局檔。

使用Dialog類別自訂對話盒

Step 2. 在程式碼中建立一個Dialog類別的物件，建立物件的同時必須指定它的擁有者，請參考下列程式碼：

```
Dialog myDlg = new Dialog(Activity類別名稱.this);
```

以上的程式碼是建立一個屬於主程式類別的Dialog物件，物件名稱叫做myDlg。

使用Dialog類別自訂對話盒

Step 3. 設定對話盒的屬性，以及指定使用哪一個介面佈局檔：

```
myDlg.setCancelable(false);  // 無法利用「回上一頁」按鈕離開對話盒。
myDlg.setContentView(R.layout.對話盒的介面佈局檔名稱);
```

使用Dialog類別自訂對話盒

Step 4. 如果對話盒中的介面元件需要設定事件處理程序，例如Button，就需要建立相關的物件。以下的程式碼是以Button的OnClickListener為例

```
private View.OnClickListener myDlgBtnOKOnClick =
new
                View.OnClickListener() {
        public void onClick(View v) {
                // Button按下後要執行的程式碼
                …
                myDlg.cancel();  // 也可以呼叫dismiss()
        }
};
```

當執行完按鈕中的工作之後，記得最後要呼叫對話盒的cancel()或是dismiss()方法結束對話盒。

使用Dialog類別自訂對話盒

如果需要取得對話盒中的介面元件的資料，例如要知道使用者在對話盒的EditText元件中輸入的字串，必須呼叫對話盒的findViewById()方法取得其中的介面元件，例如

EditText editText = (EditText) **myDlg**.findViewById(R.id.介面元件的id);

使用Dialog類別自訂對話盒

Step 5. 取得對話盒中需要設定事件處理程序的介面元件，然後把建立好的事件處理程序物件設定給它，例如：

```
Button btn = (Button)myDlg.findViewById(R.id.介面元件id);
btn.setOnClickListener(myDlgBtnOKOnClick);
```

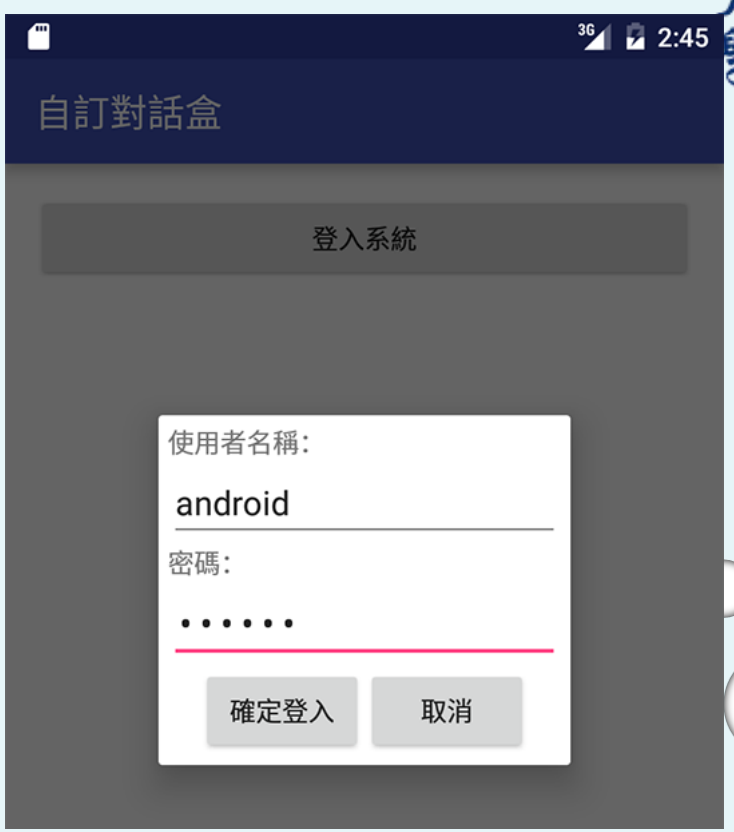
其中的myDlgBtnOKOnClick是在步驟4建立的物件。

Step 6. 呼叫show()方法顯示對話盒

```
myDlg.show();
```

範例程式

對話盒介面佈局檔、程式介面佈局檔和程式碼請參考書上說明。



小專題-
點餐系統

點餐系統完成畫面

activity_main.xml

- 步驟4. 加入元件。

元件名	text
Button	歡迎光臨
TextView	(空白)

MainActivity.java

- 步驟1. 宣告元件與菜單內容。

```
Button btn;
TextView tv;
String[] menu=new String[] {"大麥克","麥香雞","牛排","烏龍麵"};
boolean[] menubool=new boolean[] {false,false,false,false};
```

  - menubool是為了讓一開始呈現時，可以是**非點選狀態**。

MainActivity.java

- 步驟2. 建立程式與元件的關係。

```
btn=(Button) findViewById(R.id.button1);
tv=(TextView) findViewById(R.id.textView1);
```

- 步驟3. 將按鈕建立一個監聽動作

```
btn.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {
        // TODO Auto-generated method stub

    }
});
```

MainActivity.java

- 步驟4. 在按鈕的監聽裡面建立AlertDialog。

```
new AlertDialog.Builder(MainActivity.this)
```

- 步驟5.設定多選表單。

```
.setMultiChoiceItems(menu,menubool, new DialogInterface.OnMultiChoiceClickListener() {

    @Override
    public void onClick(DialogInterface arg0, int arg1, boolean arg2) {
        // TODO Auto-generated method stub
        menubool[arg1]=arg2;
    }
})
```

  - menubool[arg1]=arg2;指的是將空白多選變成打勾點選樣子。

MainActivity.java

- 步驟6.建立AlertDialog按鈕，並將它呈現出來。

```
.setPositiveButton("確認點餐", new DialogInterface.OnClickListener() {

    @Override
    public void onClick(DialogInterface dialog, int which) {
        // TODO Auto-generated method stub
        String menustring="";
        for(int i=0;i<menu.length;i++)
        {
            if(menubool[i]==true)
                menustring+=menu[i]+"、";
        }
        Toast.makeText(MainActivity.this, "你所點的菜有："+menustring, Toast.LENGTH_LONG).show();
        Log.v("menu", "你所點的菜有："+menustring);
        tv.setText("你所點的菜有："+menustring);
    }
}).show();
```

將menubool裡面判定為打勾的加入到menustring中，並用、分開。

完整按鈕程式碼

```
btn.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub
        new AlertDialog.Builder(MainActivity.this)
        .setTitle("請選擇菜單")
        .setMultiChoiceItems(menu,menubool, new DialogInterface.OnMultiChoiceClickListener() {

            @Override
            public void onClick(DialogInterface arg0, int arg1, boolean arg2) {
                // TODO Auto-generated method stub
                menubool[arg1]=arg2;
            }
        })
        .setPositiveButton("確認點餐", new DialogInterface.OnClickListener() {

            @Override
            public void onClick(DialogInterface dialog, int which) {
                // TODO Auto-generated method stub
                String menustring="";
                for(int i=0;i<menu.length;i++)
                {
                    if(menubool[i]==true)
                        menustring+=menu[i]+"、";
                }
                Toast.makeText(MainActivity.this, "你所點的菜有："+menustring, Toast.LENGTH_LONG).show();
                Log.v("menu", "你所點的菜有："+menustring);
                tv.setText("你所點的菜有："+menustring);
            }
        }).show();

    }
});
```