

先至

<http://www.oracle.com/technetwork/java/javase/archive-139210.html>

下載 **JavaFX Scene Builder 2.0**

目前網路上的教學大多是 1.1 版,如果找到 1.1 版的資料還是能用,只是有些功能的位置不一樣

通常使用 **FXML** 的建立順序為：

1. FXML -> FXML Control class -> Main class

另一種順序是：

2. FXML Control class -> FXML -> Main class

第二種順序建議先在紙上大概寫出你需要的元件和元件名稱,本教學採用第一種順序

FXML 檔由 **JavaFX Scene Builder** 編寫,主要為面板的排版

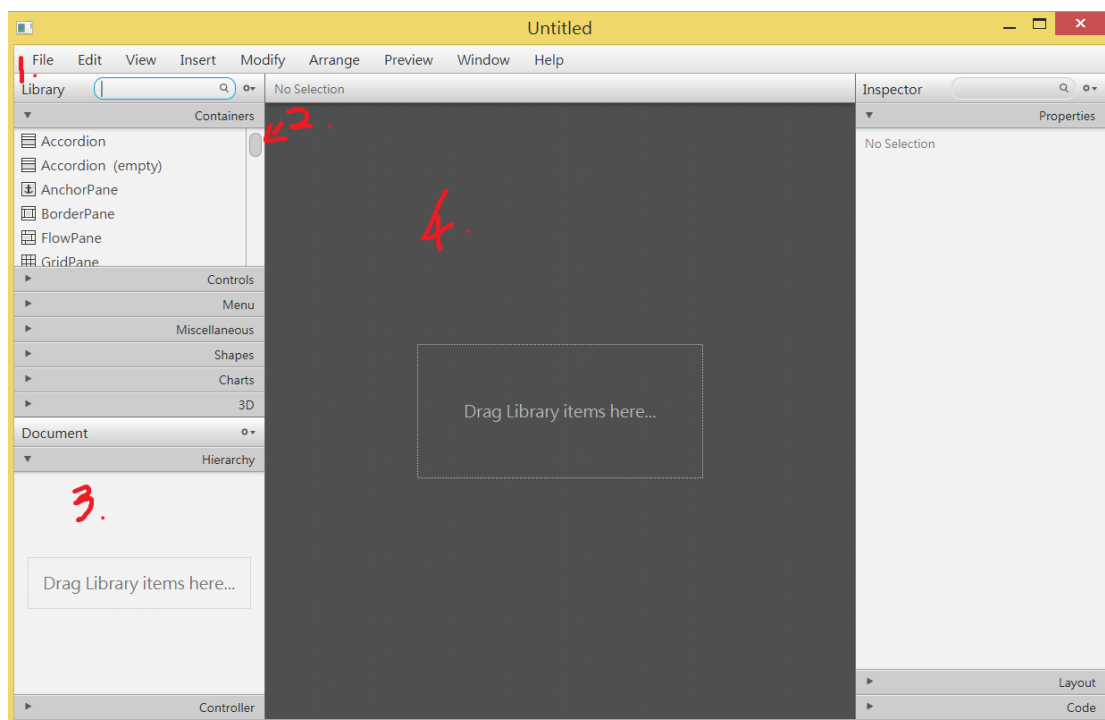
FXML Control class 用來控制 **FXML** 裡的元件,基本上所有的事件(event)寫在這個 **class** 裡

Main class 主要為讀取 **FXML** 檔,除非有特殊需求,否則此 **class** 僅用於讀 **FXML** 檔

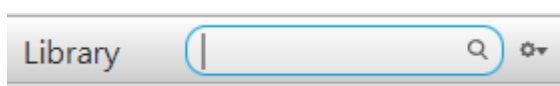
建立 FXML 檔

下載完後,雙擊圖示開啟 JavaFX Scene Builder

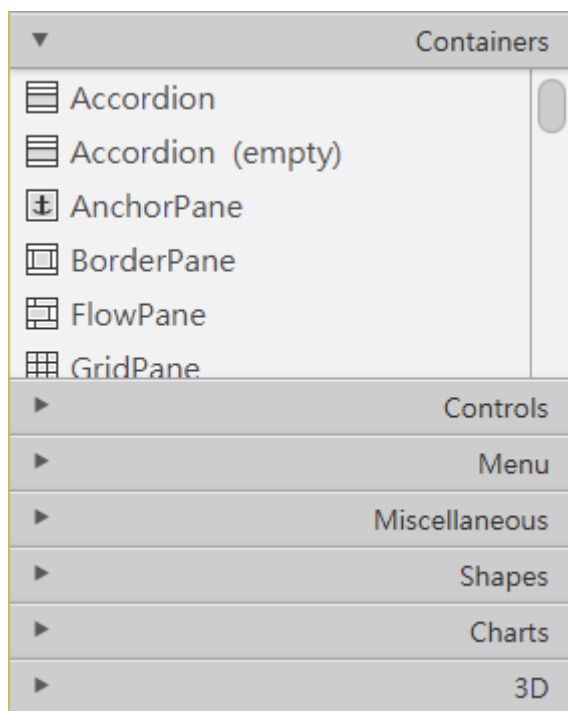
P.S.1.1 版長這樣



1. 從這邊可以直接搜尋你要的元件
(ex.Button,TextField)

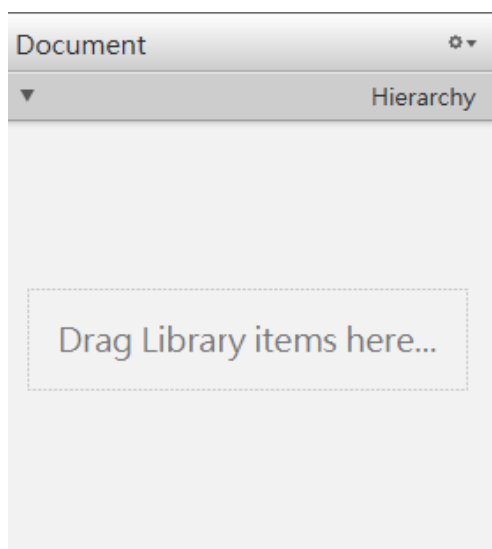


2. 從這邊可以尋找你需要的元件,並將其拖曳至 4

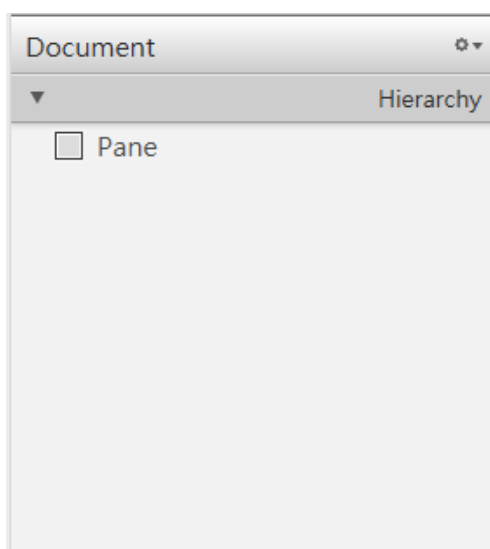


3. 你從 2 拖曳至 4 的元件會顯示(名稱)在這裡

拖曳前

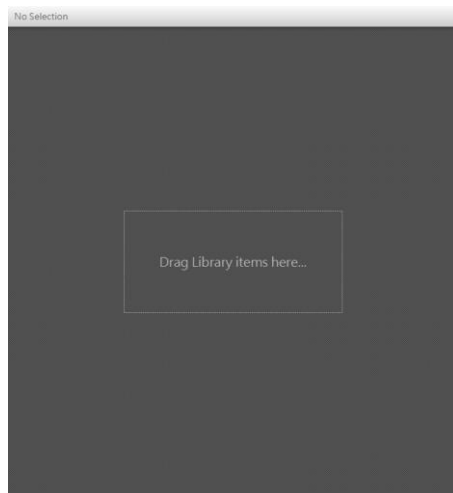


拖曳後(以 **Pane** 為例)

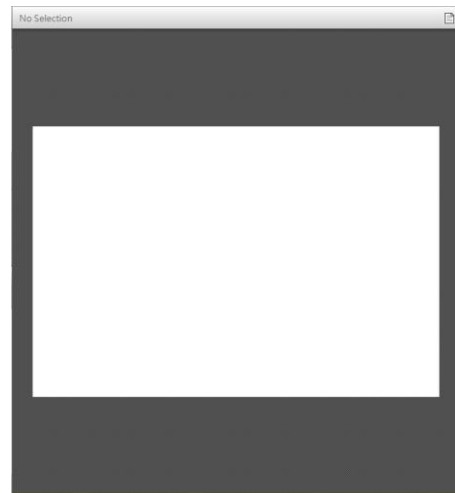


4. 從 2 拖曳至 4 的元件會顯示在這裡

拖曳前



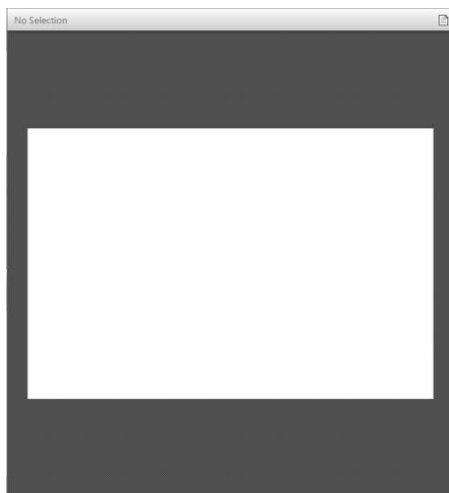
拖曳後(以 **Pane** 為例)



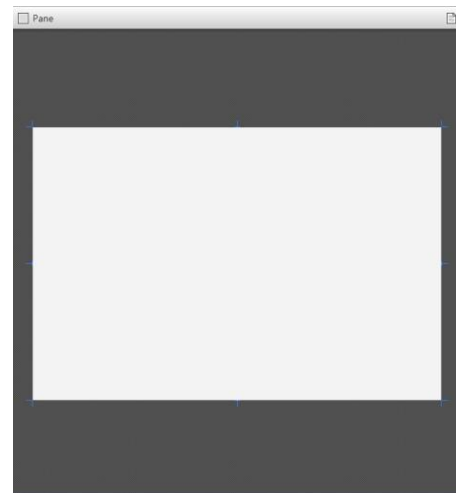
修改元件

選擇你要修改的元件



選取前

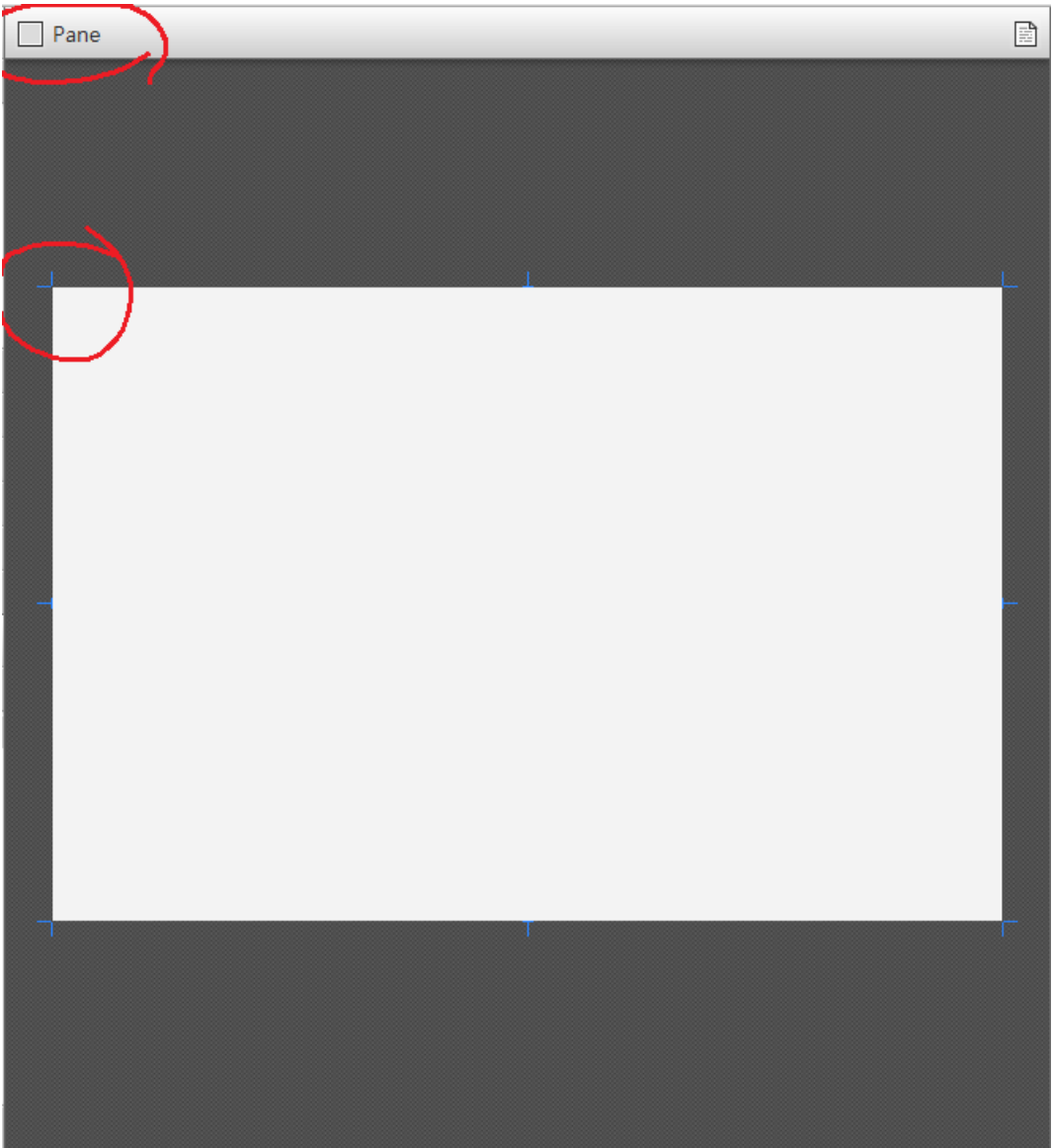


選取後

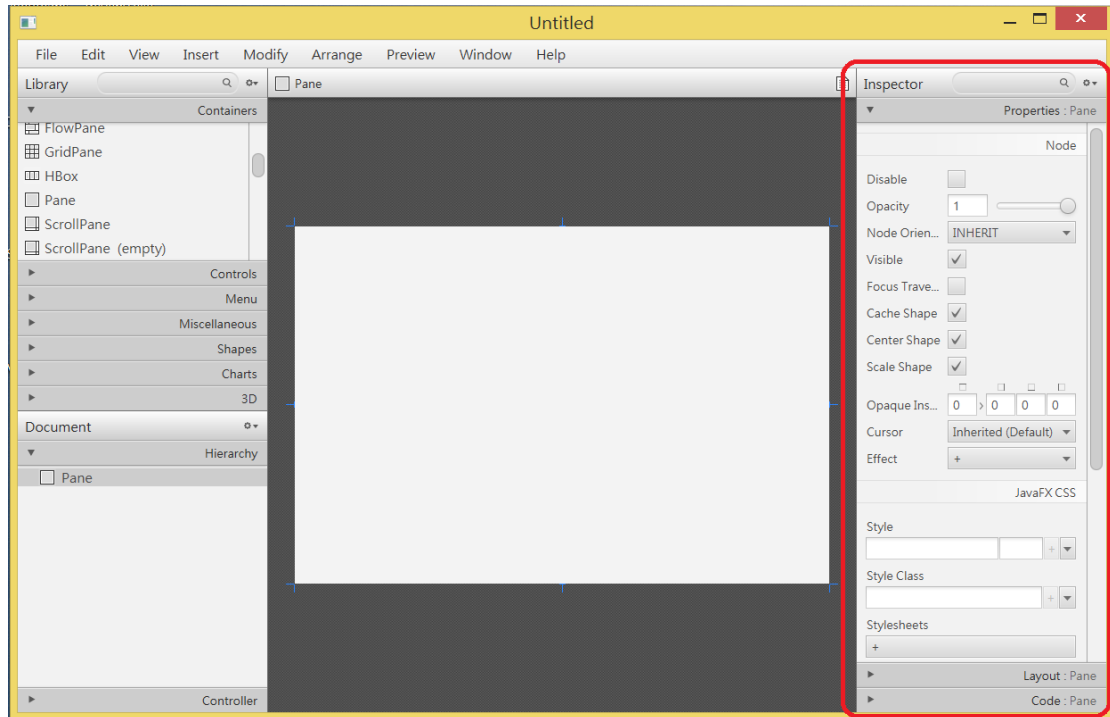


差異比較:

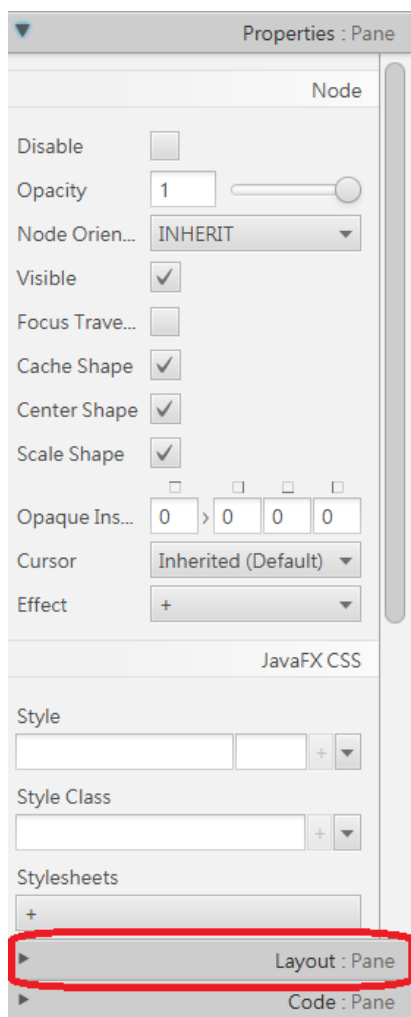
名稱	No Selection	<input type="checkbox"/> Pane
選取框		



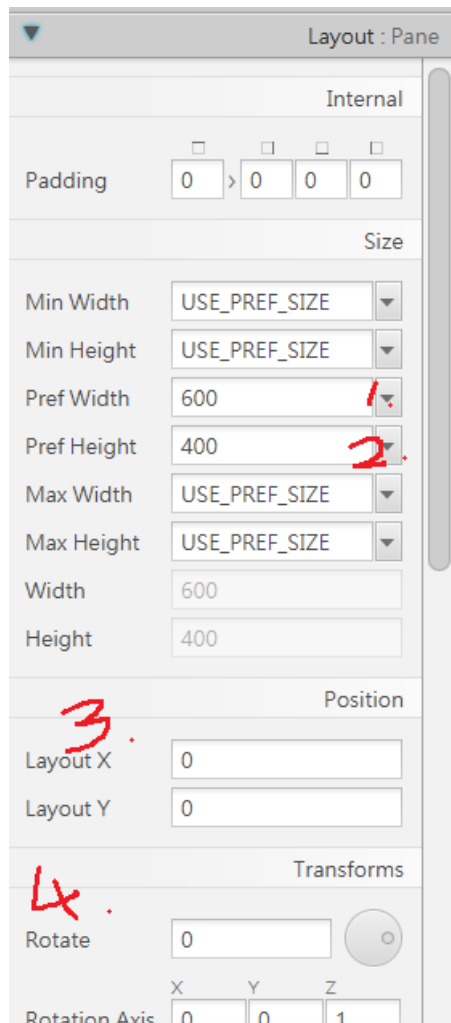
從框選的地方可以改變面板的配置



最常使用的應是 **Layout**



按 ▶ 點開



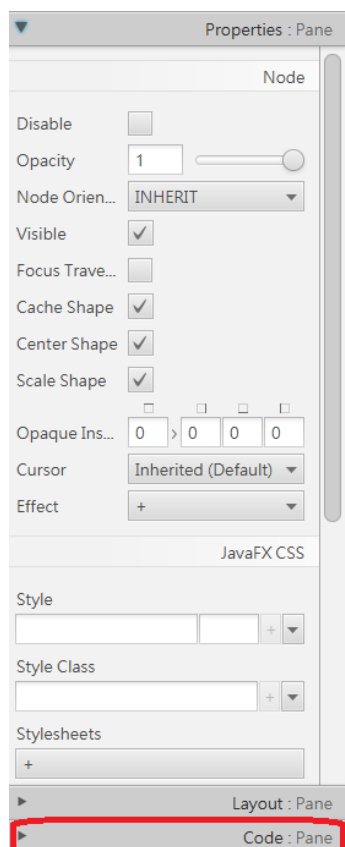
1.決定寬度

2.決定高度

3.設定位置的 x,y 座標

4.旋轉角度

如果要在程式碼控制這項元件要在 **Code** 設定



按 ▶ 點開

在 id 欄位替你的元件取名

Code : Pane

Identity

fx:id

DragDrop

On Drag Detected
#

On Drag Done
#

On Drag Dropped
#

On Drag Entered
#

On Drag Exited
#

On Drag Over
#

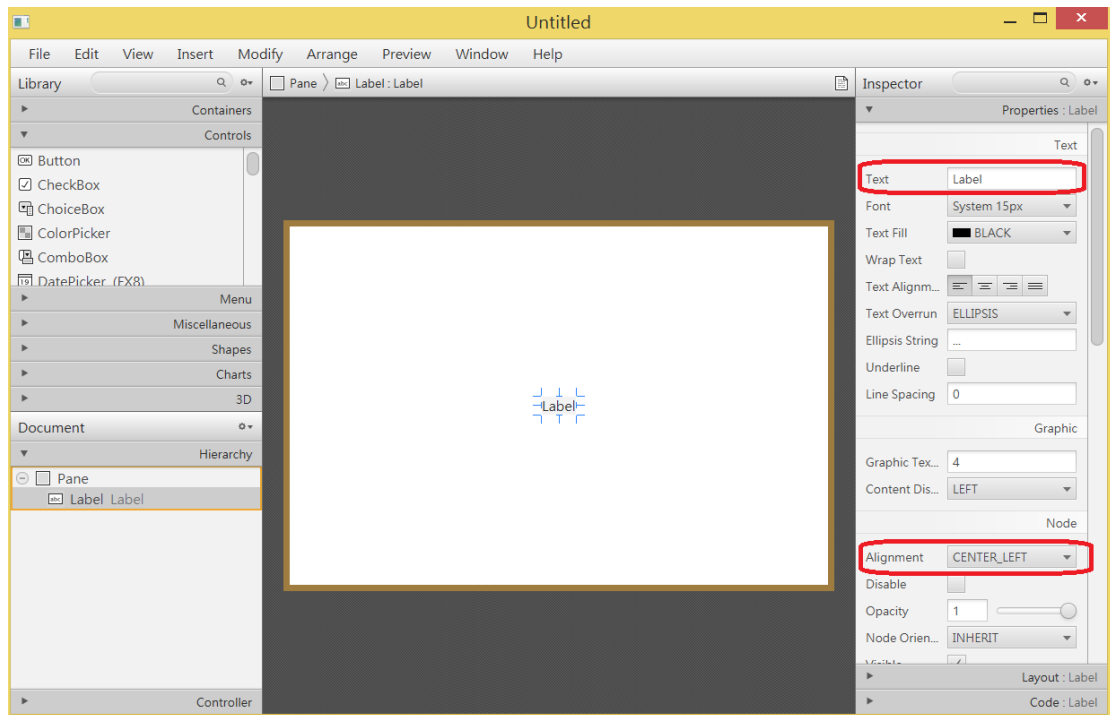
On Mouse Drag Entered
#

On Mouse Drag Exited
#

On Mouse Drag Over
#

範例解說

加入一個 Label 元件



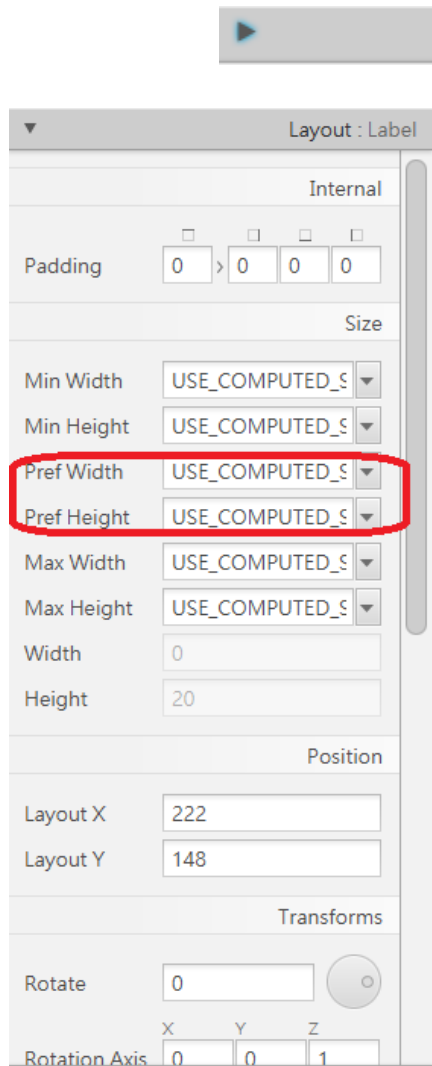
將 Text(文字)設為空



將 Alignment(排版)設為置中



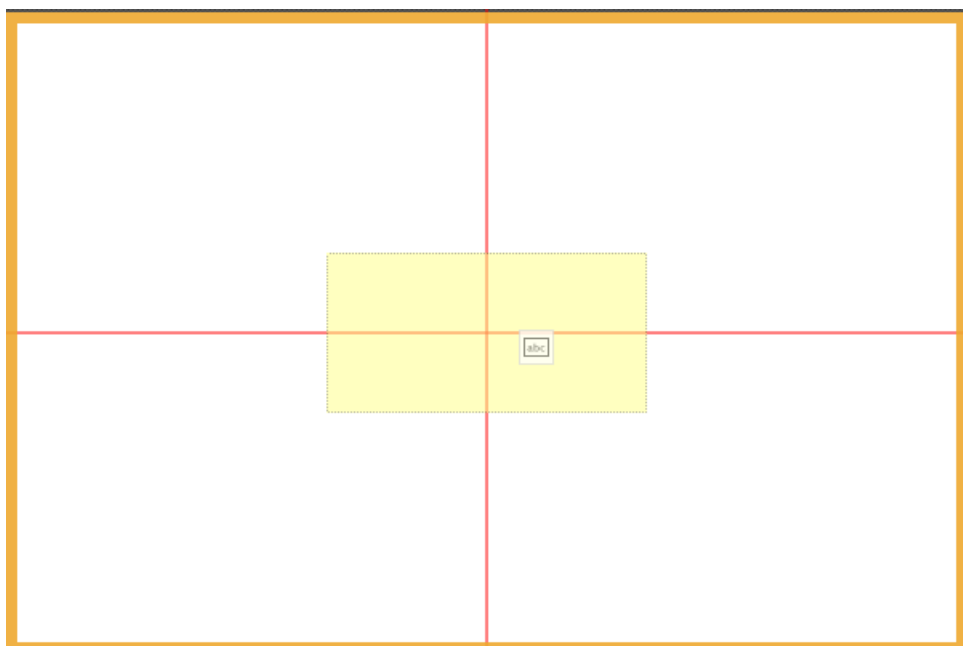
開啟 Layout 面板



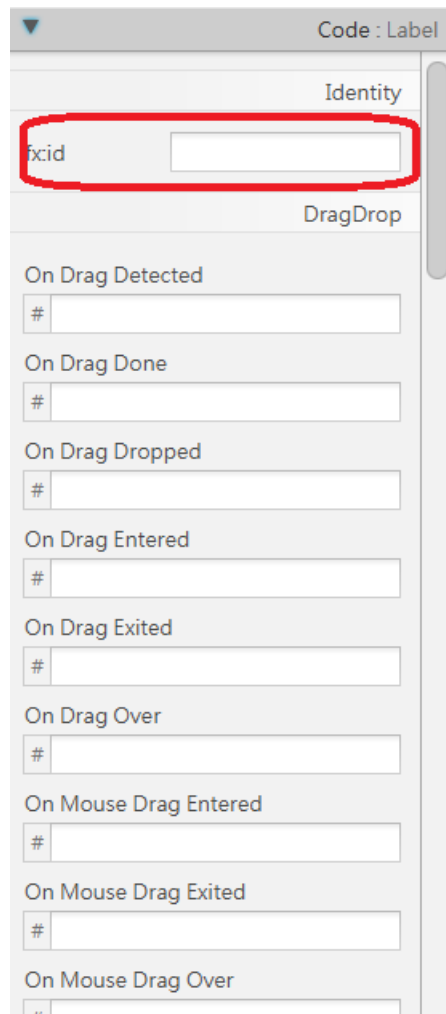
設寬為 200

長為 100

以拖曳方式設定 Label 位置(紅色為輔助線)

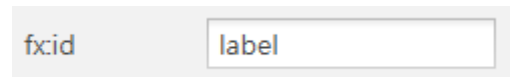
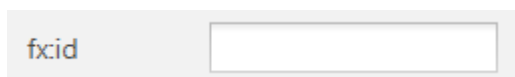


開啟 Code 面板

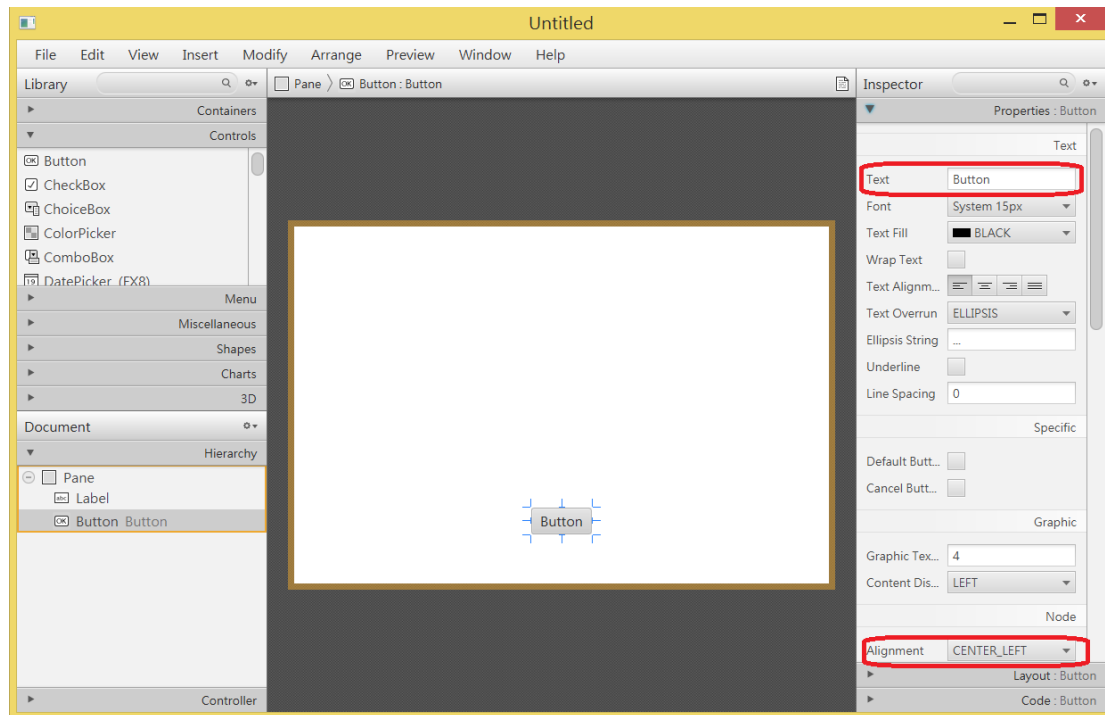


設定 Label 的 id

(即在 **Control class** 的變數名稱)



加入一個 Button 元件



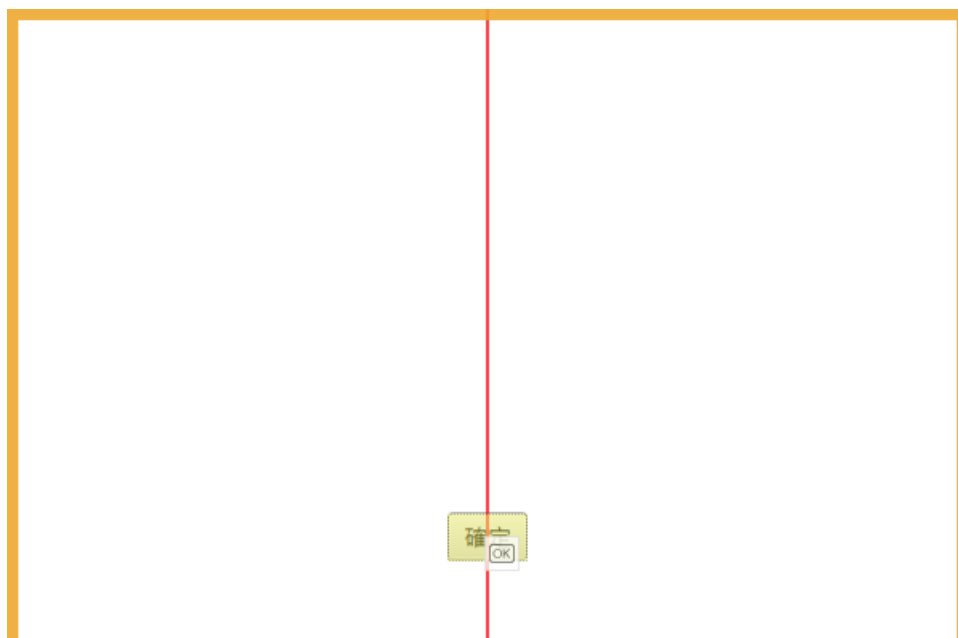
設定 Text(文字)為確定



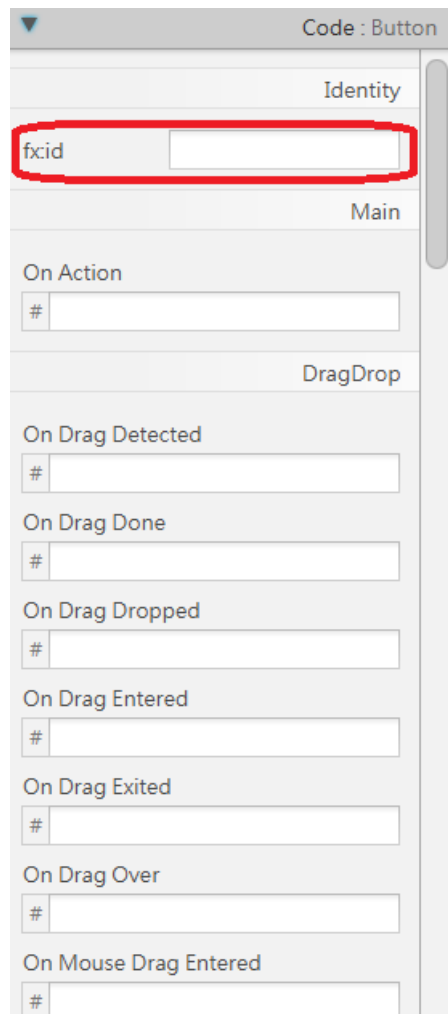
將 Alignment(排版)設為置中



以拖曳方式設定 Button 位置



開啟 Code 面板

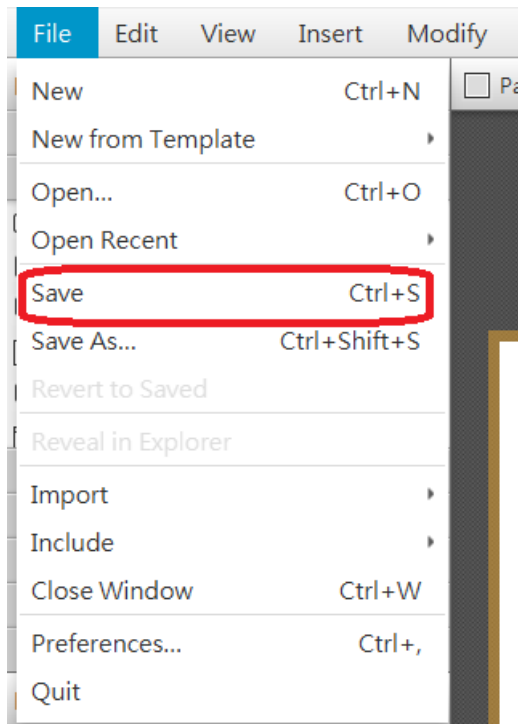


設定 Button 的 id

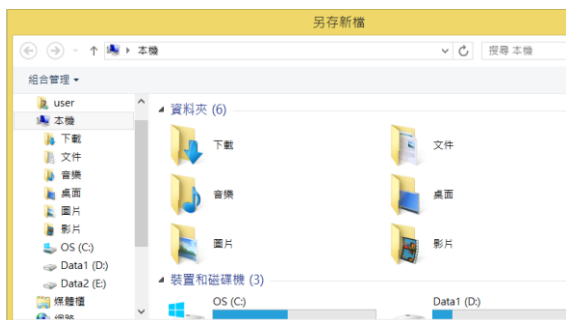
(即在 Control class 的變數名稱)



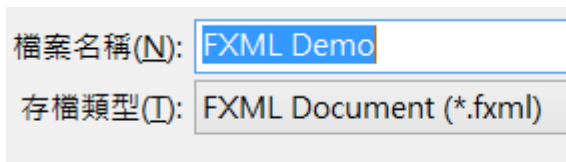
儲存檔案(Ctrl+S)



點選 **Save**



注意存檔路徑!
不要連自己存在哪裡
都不知道



注意存檔類型應
為.fxml

建立 Control class

基本程式碼

```
import java.util.*;
import java.net. *;
import javafx.event.*;
import javafx.fxml.*;

public class Control implements Initializable{

    public Control(){
    }

    @Override
    public void initialize(URL location, ResourceBundle resources){
    }
}
```

- Control class 必須繼承 Initializable,且必須

Override initialize method

import java.util.*;	initialize 的 參 數 (ResourceBundle)需要
import java.net. *;	initialize 的參數(URL) 需要
import javafx.event.*;	設定物件事件需要用到
import javafx.fxml.*;	讀取元件需要用到

控制元件事件的方法有兩種

1. 在 initialize 裡面直接寫
2. 寫一個 method 指定給元件

第一種方法：

```
import javafx.scene.control.Button;
import javafx.scene.control.Label;
@FXML
Button okay;
@FXML
Label label;
```

- okay 跟 label 為前面替元件取的名稱

```
@Override
public void initialize(URL location, ResourceBundle
resources){
    okay.setOnAction(e->{
        label.setText("123");
    });
}
```


完整程式碼

```
import java.util.*;
import java.net.*;
import javafx.event.*;
import javafx.fxml.*;
import javafx.scene.control.Button;
import javafx.scene.control.Label;

public class Control implements Initializable{

    @FXML
    Button okay;
    @FXML
    Label label;

    public Control(){
    }

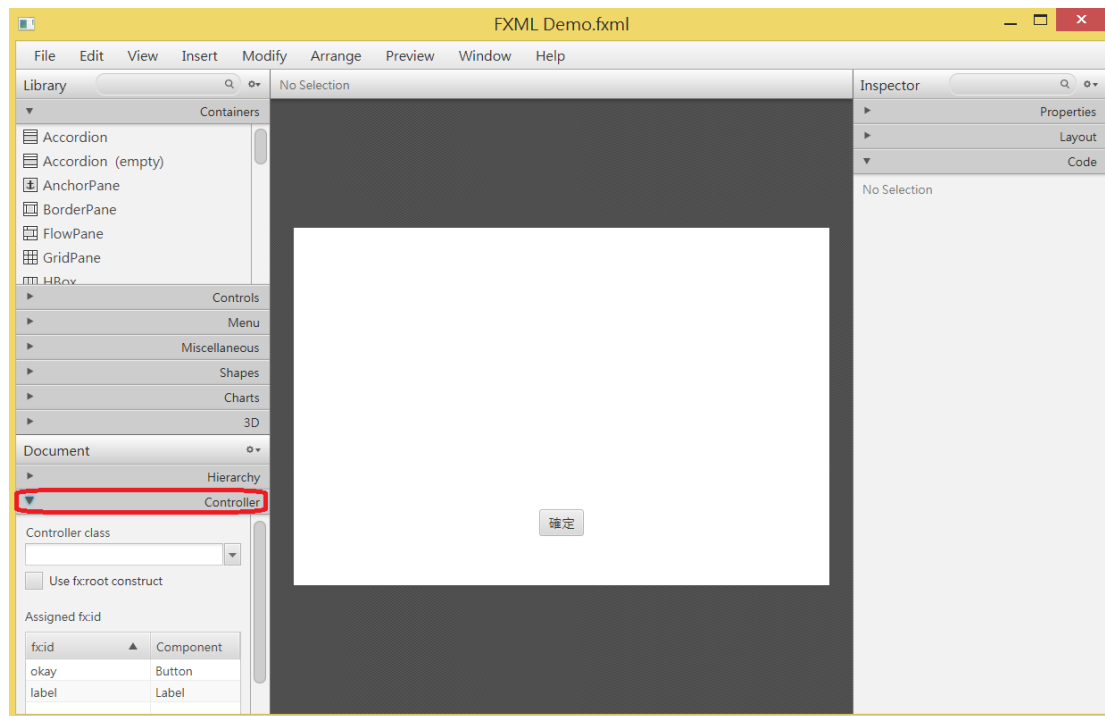
    @Override
    public void initialize(URL location, ResourceBundle resources){
        okay.setOnAction(e->{
            label.setText("123");
        });
    }
}
```

• 注意! Control class 的路徑需與 fxml 檔的路徑相同

編譯成功後會產生一個.class 檔



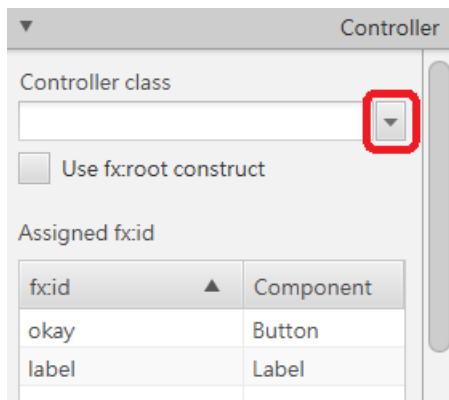
雙擊 開啟先前儲存的 fxml 檔



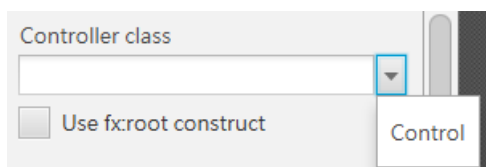
開啟 Controller



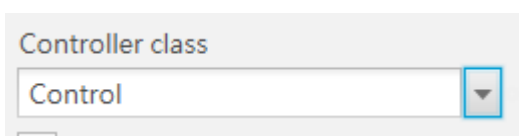
設定 Controller class



點擊



已經編譯好的 class 就會出現在選單中



選擇完成

• 注意! 記得再次**存檔**

第二種方法：

```
import javafx.scene.control.Label;
```

```
@FXML
```

```
Label label;
```

```
public void buttonControl(){  
    label.setText("123");  
}
```

完整程式碼

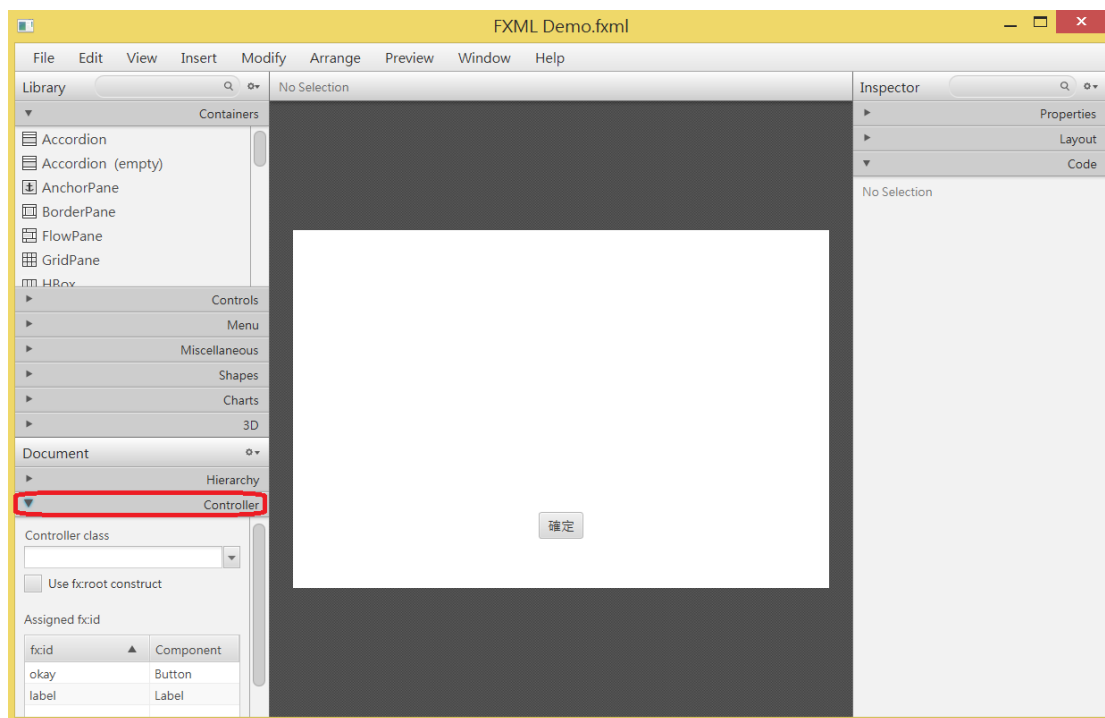
```
import java.util.*;  
import java.net.*;  
import javafx.event.*;  
import javafx.fxml.*;  
import javafx.scene.control.Label;  
  
public class Control implements Initializable{  
  
    @FXML  
    Label label;  
  
    public Control(){  
    }  
  
    public void buttonControl(){  
        label.setText("123");  
    }  
  
    @Override  
    public void initialize(URL location, ResourceBundle resources){  
    }  
}
```

• 注意! Control class 的路徑需與 fxml 檔的路徑相同

編譯成功後會產生一個.class 檔



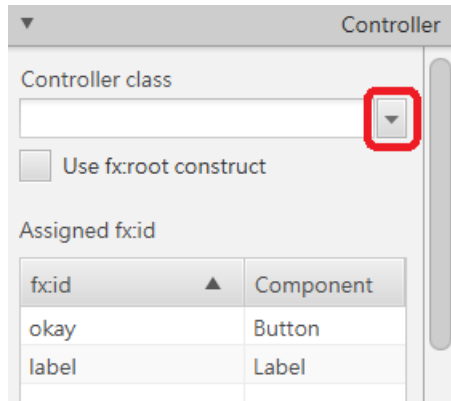
雙擊開啟先前儲存的 fxml 檔



開啟 Controller



設定 Controller class

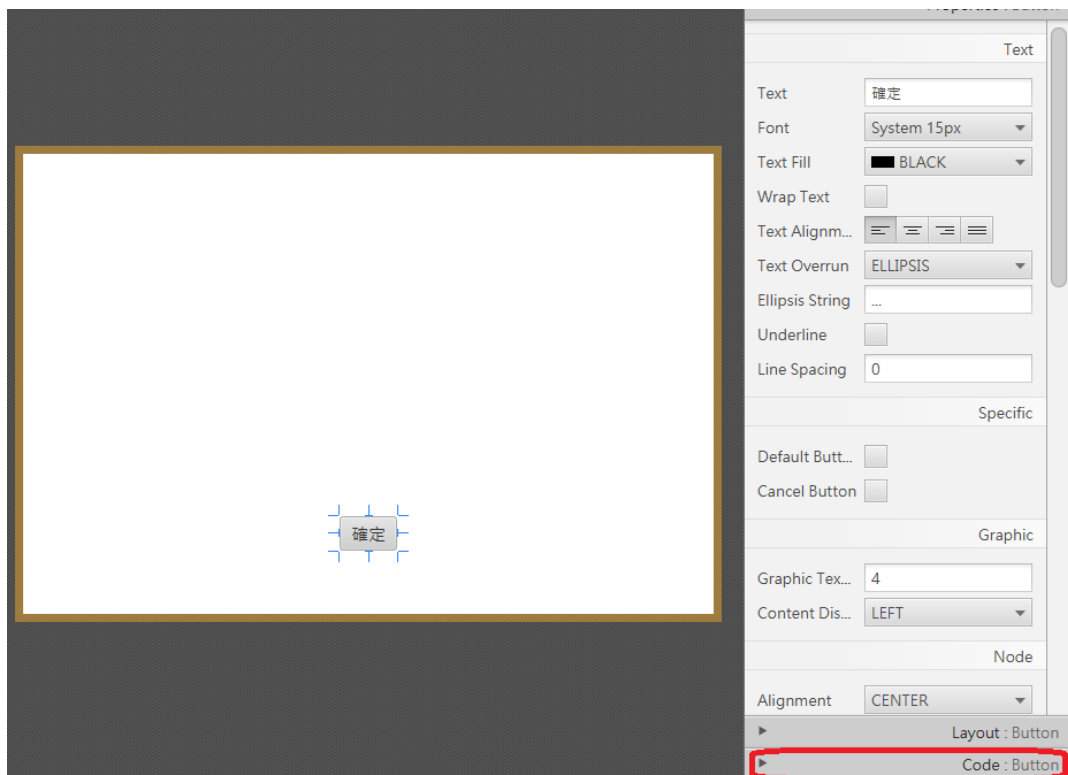


點擊

已經編譯好的 class
就會出現在選單中

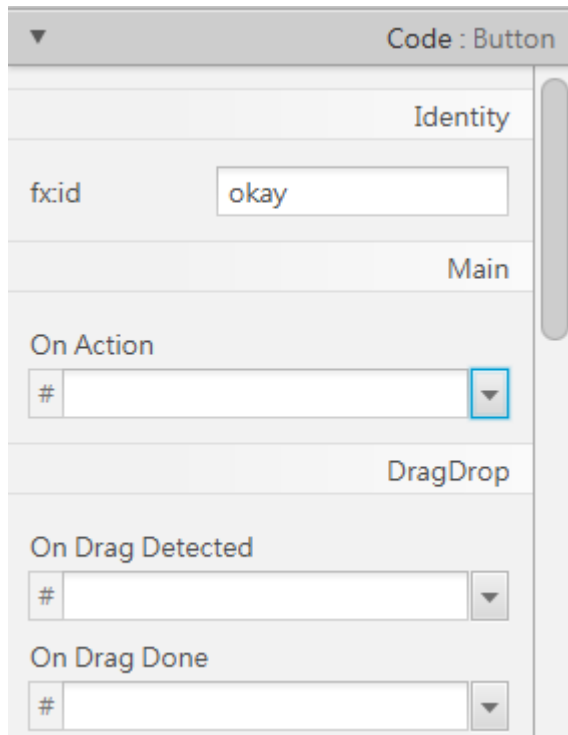
選擇完成

選擇 okay Button

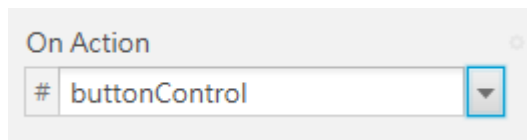
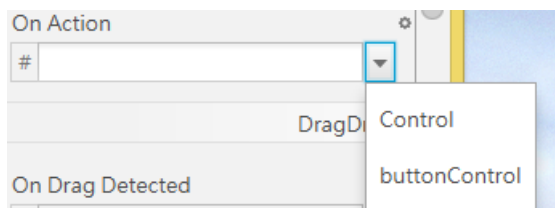


開啟 Code





點擊 



這裡會顯示除了
initialize 以外的所有
method

選擇 buttonControl

- 注意! 記得再次**存檔**

建立 Main class

基本程式碼

```
import javafx.fxml.FXMLLoader;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.Parent;
import javafx.stage.Stage;

public class FXMLDemo extends Application{
    Scene scene;
    Parent parent;

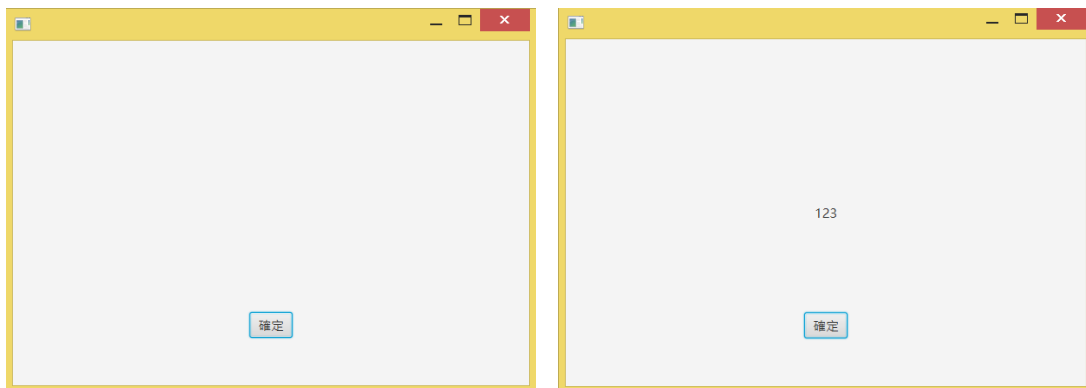
    @Override
    public void start(Stage mystage){
        try{
            parent = FXMLLoader.load(getClass().getResource("FXML
Demo.fxml"));
        }
        catch (Exception ex) {
            System.out.println(ex.toString());
        }
        scene = new Scene(parent);
        mystage.setScene(scene);
        mystage.show();
    }

    public static void main(String[] args){
        launch(args);
    }
}
```

import javafx.fxml.FXMLLoader;	讀取 fxml 檔
import javafx.application.Application;	繼承 Application

import javafx.scene.Parent;	將讀取的 FXML 定義為 Parent 型態
讀取 FXML 檔必須寫在 try-catch 裡	
<pre>try{ parent = FXMLLoader.load(getClass().getResource("FXML Demo.fxml")); } catch (Exception ex) { System.out.println(ex.toString()); }</pre>	
parent = FXMLLoader.load(getClass().getResource("FXML Demo.fxml"));	getResource("")裡 面放 fxml 的完整 檔名(包含副檔名)

執行結果



2016/4/30 陳熏妤