Tutorial – Drawing for Form and Store

Tai-Hu Huang
Ver. 1.0, 18th November, 2013
Jyun-Wei Chen
Ver. 2.0, 22th November, 2022

Contents

About This Document

在本次的練習中，將會練習到如何使用同一個 Model 來建置兩種不同 View (Windows Form 與 Windows Store APP)的繪圖程式。

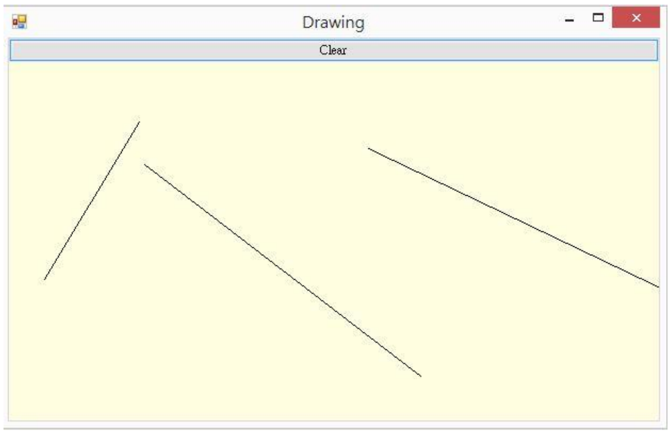


Figure 1 程式執行結果 - Windows Form

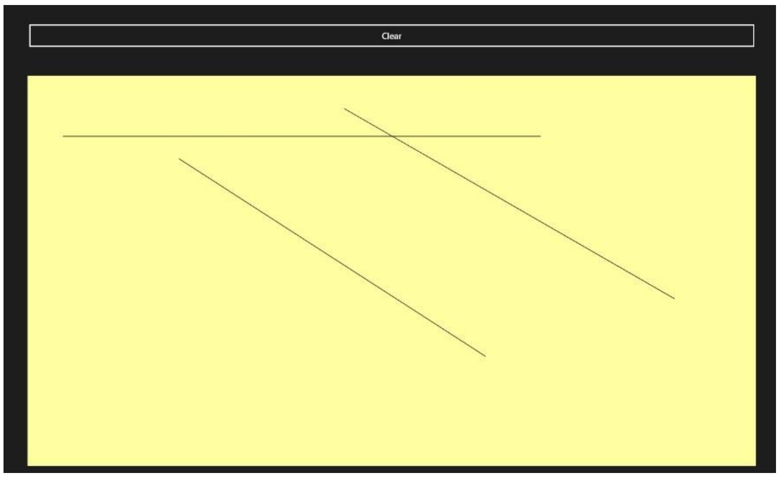
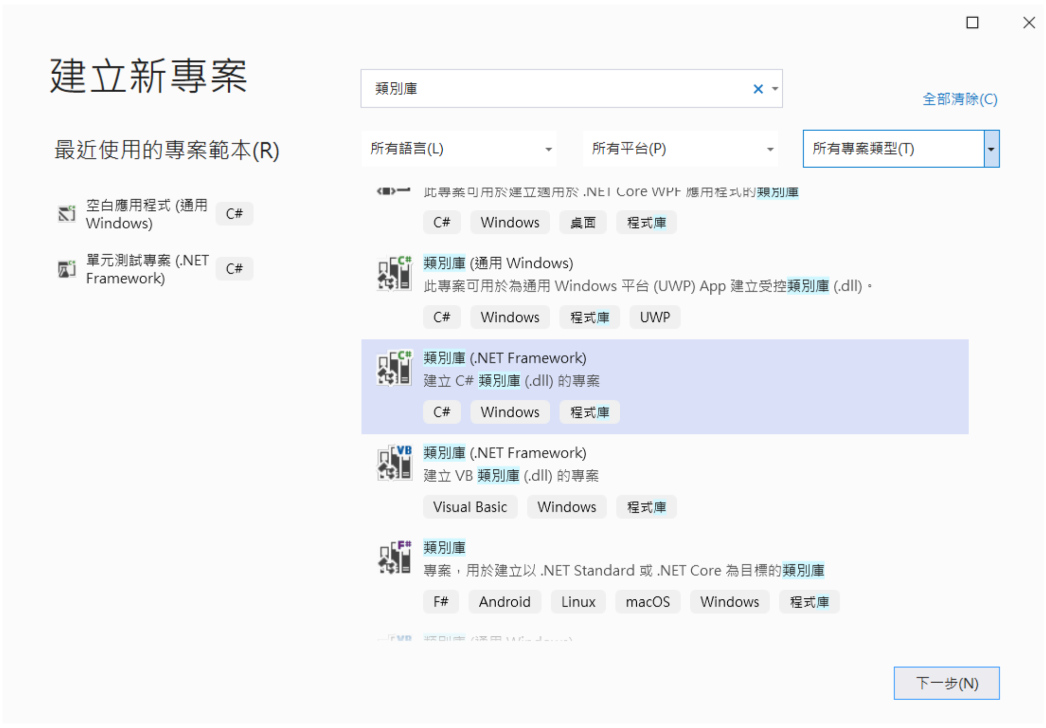


Figure 2 程式執行結果 - Windows Store APP

Tutorial – Drawing for Form and Store

Step 1 Create Model Project

打開 Visual Studio 2019 Community 後，點選 檔案 > 新增 > 專案 > 搜尋 ”類別庫” > 選擇 “類別庫(.NET Framework) ”，然後輸入專案名稱 “DrawingModel” ，即可成功建立一個空白的 C#專案，此為用來放置 Model 的專案。



建立新專案

類別庫

全部清除(C)

最近使用的專案範本(R)

所有語言(L) 所有平台(P) 所有專案類型(T)

空白應用程式 (通用 Windows) C#

單元測試專案 (.NET Framework) C#

此專案可用於建立適用於 .NET Core WPF 應用程式的類別庫

C# Windows 桌面 程式庫

類別庫 (通用 Windows)

此專案可用於為通用 Windows 平台 (UWP) App 建立受控類別庫 (.dll)。

C# Windows 程式庫 UWP

類別庫 (.NET Framework)

建立 C# 類別庫 (.dll) 的專案

C# Windows 程式庫

類別庫 (.NET Framework)

建立 VB 類別庫 (.dll) 的專案

Visual Basic Windows 程式庫

類別庫

專案，用於建立以 .NET Standard 或 .NET Core 為目標的類別庫

F# Android Linux macOS Windows 程式庫

下一步(N)

Figure 3 建立類別庫專案

Step 2 Create Canvas Model and Adaptee

刪除掉原本系統自己建立的 Class1.cs，於 “DrawingModel” 中新增三個 Class，如下：

(a) IGraphics.cs

```
namespace DrawingModel
{
    interface IGraphics
    {
        void ClearAll();
        void DrawLine(double x1, double y1, double x2, double y2);
    }
}
```

(b) Line.cs

```
namespace DrawingModel
{
    class Line
    {
        public double x1;
        public double y1;
        public double x2;
        public double y2;

        public void Draw(IGraphics graphics)
        {
            graphics.DrawLine(x1, y1, x2, y2);
        }
    }
}
```

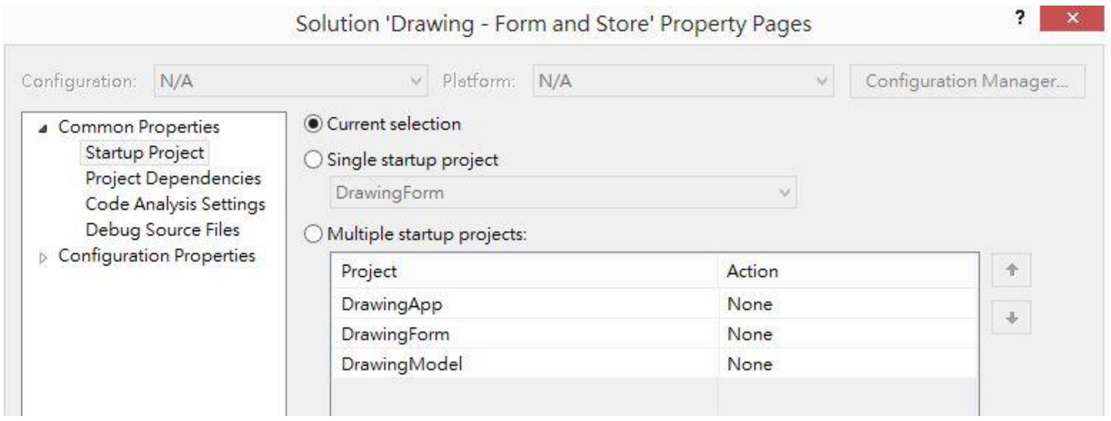
(c) Model.cs

```
using System.Collections.Generic;

namespace DrawingModel
{
    class Model
    {
        public event ModelChangedEventHandler _modelChanged;
        public delegate void ModelChangedEventHandler();
        double _firstPointX;
        double _firstPointY;
        bool _isPressed = false;
        List<Line> _lines = new List<Line>();
        Line _hint = new Line();

        public void PointerPressed(double x, double y)
        {
            if (x > 0 && y > 0)
            {
                _firstPointX = x;
                _firstPointY = y;
                _hint.x1 = _firstPointX;
                _hint.y1 = _firstPointY;
                _isPressed = true;
            }
        }

        public void PointerMoved(double x, double y)
        {
            if (_isPressed)
            {
                _hint.x2 = x;
                _hint.y2 = y;
                NotifyModelChanged();
            }
        }

        public void PointerReleased(double x, double y)
        {
            if (_isPressed)
            {
                _isPressed = false;
                Line hint = new Line();
                hint.x1 = _firstPointX;
                hint.y1 = _firstPointY;
                hint.x2 = x;
                hint.y2 = y;
                _lines.Add(hint);
                NotifyModelChanged();
            }
        }

        public void Clear()
        {
            _isPressed = false;
            _lines.Clear();
            NotifyModelChanged();
        }

        public void Draw(IGraphics graphics)
        {
            graphics.ClearAll();
            foreach (Line aLine in _lines)
                aLine.Draw(graphics);
            if (_isPressed)
```

```
                _hint.Draw(graphics);
        }

        void NotifyModelChanged()
        {
            if (_modelChanged != null)
                _modelChanged();
        }
    }
}
```

Step 3 Modify the Setting of Startup Project

當你需要執行多個專案(在這個 Lab 中，你將會建立三個專案)，又不想要一直切換 Setup Project 時，可以於 Solution 內設定，對著 Solution 點擊右鍵，點擊 Properties，並選擇 Current selection，如此，當你點擊某一特定專案時，即會執行該專案。


Solution 'Drawing - Form and Store' Property Pages

Configuration: N/A   Platform: N/A   Configuration Manager...

- Common Properties
  - Startup Project
  - Project Dependencies
  - Code Analysis Settings
  - Debug Source Files
- Configuration Properties

(●) Current selection
( ) Single startup project: DrawingForm
( ) Multiple startup projects:

Project	Action
DrawingApp	None
DrawingForm	None
DrawingModel	None

Figure 4 Right Click > Properties > Current selection

Step 4 Compile Model

1. 於 “DrawingModel” 專案點擊右鍵，選擇建置(Build)。

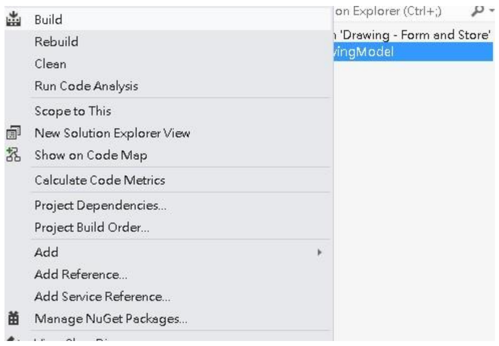


Figure 5 Build

2. 於 “DrawingModel” 專案點擊右鍵，選擇在檔案總管中開啟資料夾(Open Folder in File Explorer)。

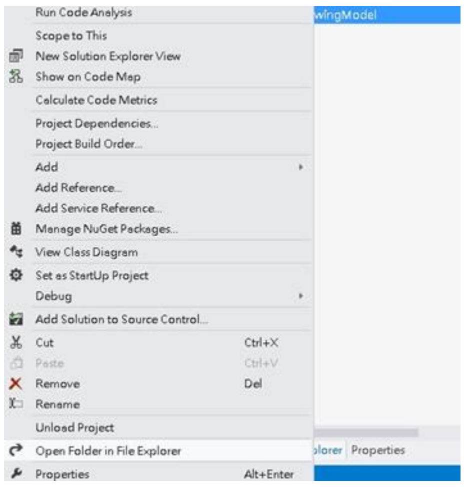
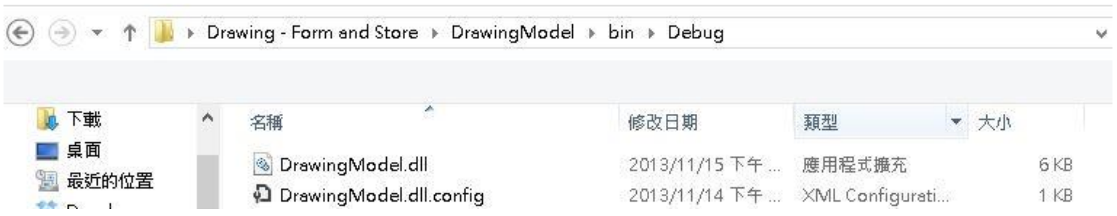


Figure 6 Open Folder in File Explorer

3. 開啟資料夾後，在 bin > Debug 可以找到一個.dll 檔。這個檔案即為上述三個 Class 編譯後的檔案。

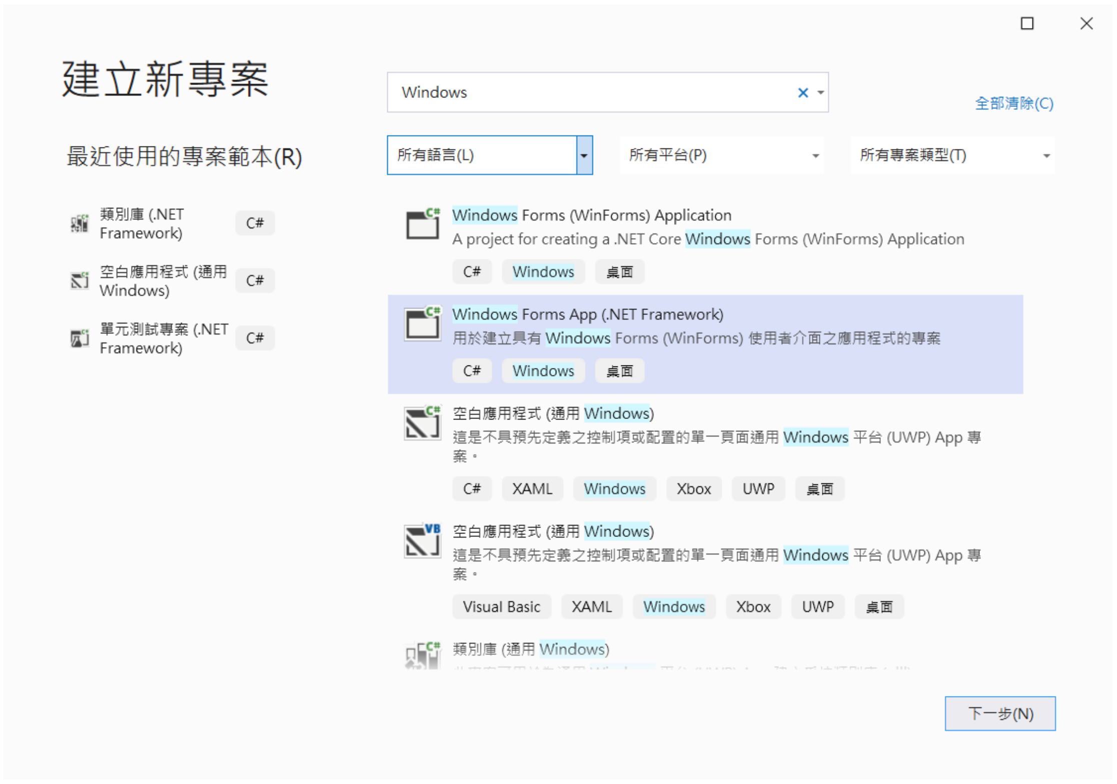


名稱	修改日期	類型	大小
DrawingModel.dll	2013/11/15 下午 ...	應用程式擴充	6 KB
DrawingModel.dll.config	2013/11/14 下午 ...	XML Configurati...	1 KB

Figure 7 bin > Debug > DrawingModel.dll

Step 5 Create View in Windows Form

新增一個 Windows Form 專案命名為 “DrawingForm” 。



建立新專案

Windows

全部清除(C)

最近使用的專案範本(R)

所有語言(L) | 所有平台(P) | 所有專案類型(T)

類別庫 (.NET Framework) C#

空白應用程式 (通用 Windows) C#

單元測試專案 (.NET Framework) C#

Windows Forms (WinForms) Application
A project for creating a .NET Core Windows Forms (WinForms) Application
C# Windows 桌面

Windows Forms App (.NET Framework)
用於建立具有 Windows Forms (WinForms) 使用者介面之應用程式的專案
C# Windows 桌面

空白應用程式 (通用 Windows)
這是不具預先定義之控制項或配置的單一頁面通用 Windows 平台 (UWP) App 專案。
C# XAML Windows Xbox UWP 桌面

空白應用程式 (通用 Windows)
這是不具預先定義之控制項或配置的單一頁面通用 Windows 平台 (UWP) App 專案。
Visual Basic XAML Windows Xbox UWP 桌面

類別庫 (通用 Windows)

下一步(N)

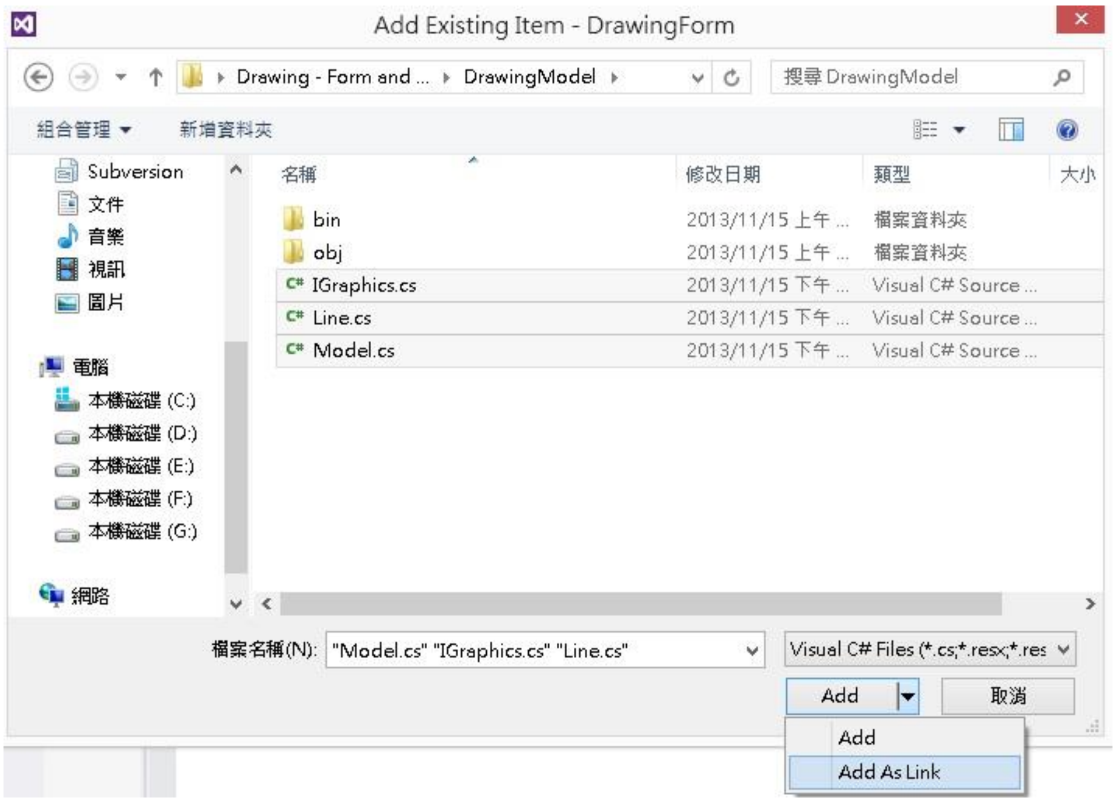
Figure 8 檔案 > 新增 > 專案 > Windows Forms App (.NET Framework)

Step 6 Add Model in Views

於 “DrawingForm” 專案執行下述之步驟。下列步驟，目的是讓目前的 View 可以擁有不同 Project 的檔案之 Link，而非擁有一個實體檔案。

1. 對著專案點擊右鍵加入一個新資料夾，命名為 “Model”

2. 對著資料夾 “Model” 點擊右鍵，選擇加入現有項目**(**注意：要選擇 “加入做為連結**(Add As Link)**” **)**，選擇於 “DrawingModel” 內的三個.cs 檔


Add Existing Item - DrawingForm
Drawing - Form and ... › DrawingModel ›
搜尋 DrawingModel
組合管理 ▾
新增資料夾
Subversion
文件
音樂
視訊
圖片
電腦
本機磁碟 (C:)
本機磁碟 (D:)
本機磁碟 (E:)
本機磁碟 (F:)
本機磁碟 (G:)
網路
名稱
修改日期
類型
大小
bin 2013/11/15 上午 ... 檔案資料夾
obj 2013/11/15 上午 ... 檔案資料夾
IGraphics.cs 2013/11/15 下午 ... Visual C# Source ...
Line.cs 2013/11/15 下午 ... Visual C# Source ...
Model.cs 2013/11/15 下午 ... Visual C# Source ...
檔案名稱(N): "Model.cs" "IGraphics.cs" "Line.cs"
Visual C# Files (*.cs;*.resx;*.res
Add
取消
Add
Add As Link

Step 7　Create Canvas Model and Adaptor in DrawingForm

於 “DrawingForm” 專案加入下列檔案。

1. 對著專案點擊右鍵加入一個新資料夾，命名為 “PresentationModel”

2. 對著資料夾 “PresentationModel” 點擊右鍵，選擇加入新類別

(a) WindowsFormsGraphicsAdaptor.cs

```
using System.Windows.Forms;
using System.Drawing;
using DrawingModel;

namespace DrawingForm.PresentationModel
{
    class WindowsFormsGraphicsAdaptor : IGraphics
    {
        Graphics _graphics;
        public WindowsFormsGraphicsAdaptor(Graphics graphics)
        {
            this._graphics = graphics;
        }

        public void ClearAll()
        {
            // OnPaint時會自動清除畫面，因此不需實作
        }

        public void DrawLine(double x1, double y1, double x2, double y2)
        {
            _graphics.DrawLine(Pens.Black, (float) x1, (float) y1, (float) x2,
(float) y2);
        }
    }
}
```

(b) PresentationModel.cs

```
using DrawingModel;
using System.Windows.Forms;

namespace DrawingForm.PresentationModel
{
    class PresentationModel
    {
        Model _model;
        public PresentationModel(Model model, Control canvas)
        {
            this._model = model;
        }

        public void Draw(System.Drawing.Graphics graphics)
        {
            // graphics物件是Paint事件帶進來的，只能在當次Paint使用
            // 而Adaptor又直接使用graphics，這樣DoubleBuffer才能正確運作
            // 因此，Adaptor不能重複使用，每次都要重新new
            _model.Draw(new WindowsFormsGraphicsAdaptor(graphics));
        }
    }
}
```

Step 8　　Create View in DrawingForm

並於 "DrawingForm" 專案新增一個 Component Class，命名為 "DoubleBufferedPanel"

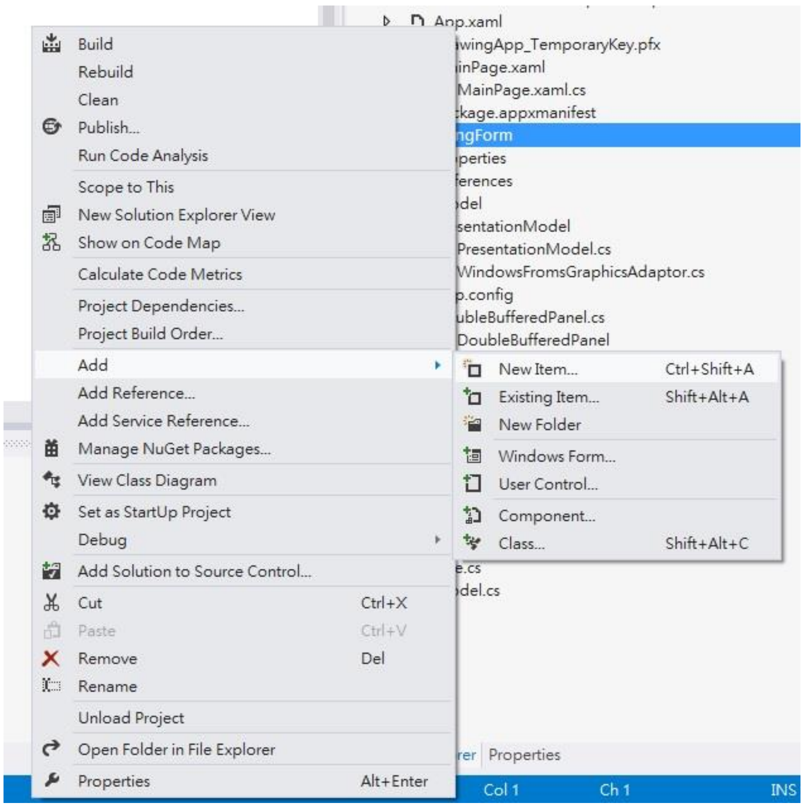
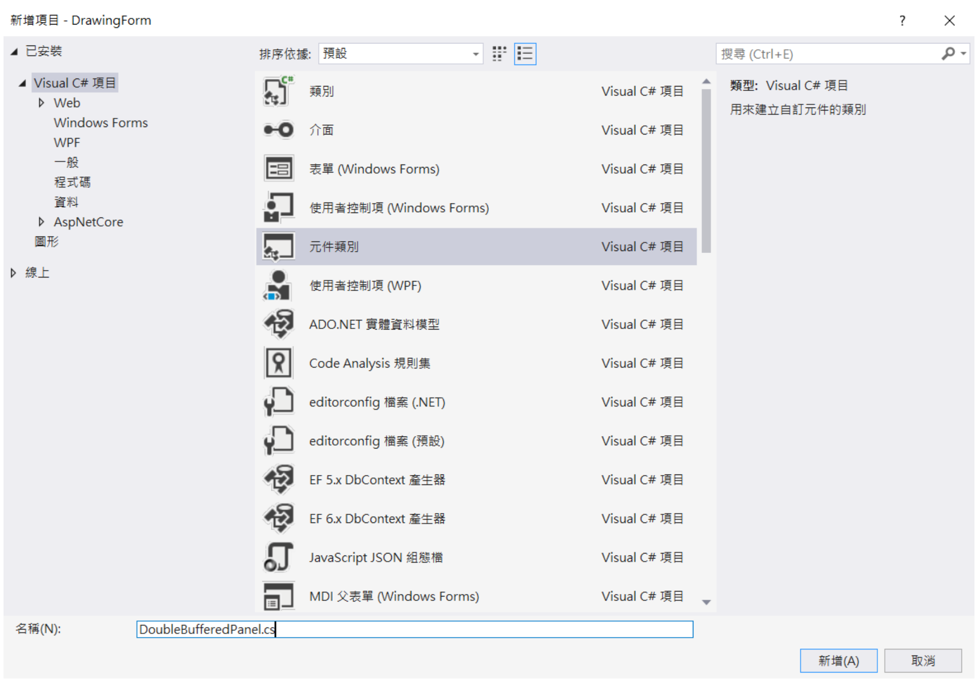


Figure 9 Add > New Item


新增項目 - DrawingForm
已安裝
Visual C# 項目
Web
Windows Forms
WPF
一般
程式碼
資料
AspNetCore
圖形
線上
排序依據: 預設
類別 Visual C# 項目
介面 Visual C# 項目
表單 (Windows Forms) Visual C# 項目
使用者控制項 (Windows Forms) Visual C# 項目
元件類別 Visual C# 項目
使用者控制項 (WPF) Visual C# 項目
ADO.NET 實體資料模型 Visual C# 項目
Code Analysis 規則集 Visual C# 項目
editorconfig 檔案 (.NET) Visual C# 項目
editorconfig 檔案 (預設) Visual C# 項目
EF 5.x DbContext 產生器 Visual C# 項目
EF 6.x DbContext 產生器 Visual C# 項目
JavaScript JSON 組態檔 Visual C# 項目
MDI 父表單 (Windows Forms) Visual C# 項目
搜尋 (Ctrl+E)
新增(A)
取消

類型: Visual C# 項目
用來建立自訂元件的類別

名稱(N): DoubleBufferedPanel.cs

Figure 10 Component Class

將 DoubleBufferedPanel.cs 展開後，刪除掉 Component1.Designer.cs，並點擊 DoubleBufferedPanel 加入以下程式碼。

```
using System.Windows.Forms;


namespace DrawingForm
{
    class DoubleBufferedPanel : Panel
    {
        public DoubleBufferedPanel()
        {
            DoubleBuffered = true;
        }
    }
}
```

於 “DrawingForm” 專案點選 Form1.cs 展開後的 Form1，並加入以下程式碼，黃底部分，為我們所撰寫的程式碼。

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DrawingForm
{
    public partial class Form1 : Form
    {
        DrawingModel.Model _model;
        PresentationModel.PresentationModel _presentationModel;
        Panel _canvas = new DoubleBufferedPanel();

        public Form1()
        {
            InitializeComponent();
            //
            // prepare canvas
            //
            _canvas.Dock = DockStyle.Fill;
            _canvas.BackColor = System.Drawing.Color.LightYellow;
            _canvas.MouseDown += HandleCanvasPressed;
            _canvas.MouseUp += HandleCanvasReleased;
            _canvas.MouseMove += HandleCanvasMoved;
            _canvas.Paint += HandleCanvasPaint;
            Controls.Add(_canvas);
            //
            // prepare clear button
            //
            Button clear = new Button();
            clear.Text = "Clear";
            clear.Dock = DockStyle.Top;
            clear.AutoSize = true;
            clear.AutoSizeMode = 
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            clear.Click += HandleClearButtonClick;
            Controls.Add(clear);
            //
            // prepare presentation model and model
            //
            _model = new DrawingModel.Model();
            _presentationModel = new PresentationModel.PresentationModel(_model, 
_canvas);
            _model._modelChanged += HandleModelChanged;
        }

        public void HandleClearButtonClick(object sender, System.EventArgs e)
        {
            _model.Clear();
        }

        public void HandleCanvasPressed(object sender, 
System.Windows.Forms.MouseEventArgs e)
        {
            _model.PointerPressed(e.X, e.Y);
        }

        public void HandleCanvasReleased(object sender, 
System.Windows.Forms.MouseEventArgs e)
        {
            _model.PointerReleased(e.X, e.Y);
        }

        public void HandleCanvasMoved(object sender, 
System.Windows.Forms.MouseEventArgs e)
        {
            _model.PointerMoved(e.X, e.Y);
        }

        public void HandleCanvasPaint(object sender, 
System.Windows.Forms.PaintEventArgs e)
```

```
        {
            _presentationModel.Draw(e.Graphics);
        }

        public void HandleModelChanged()
        {
            Invalidate(true);
        }
    }
}
```

Step 9　Execute DrawingForm

執行 DrawingForm 專案，你將可以看到 Form 的執行結果。於此階段，你已經完成了第一個 View，接下來你會需要第三個專案來製作 Windows Store App 的 View。

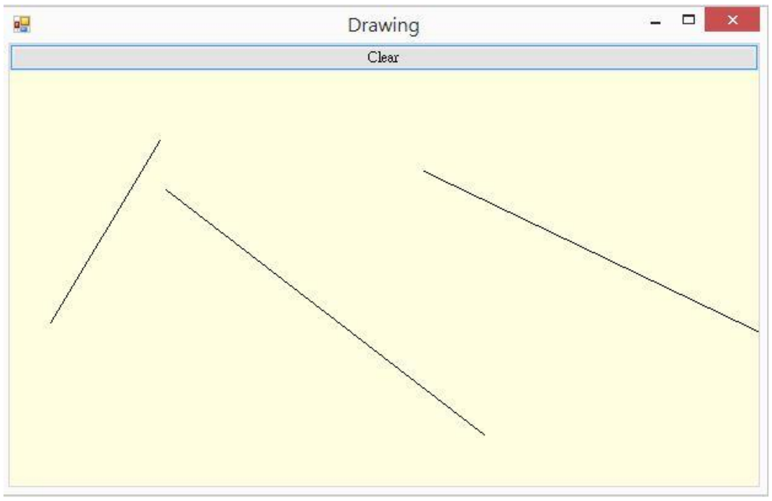
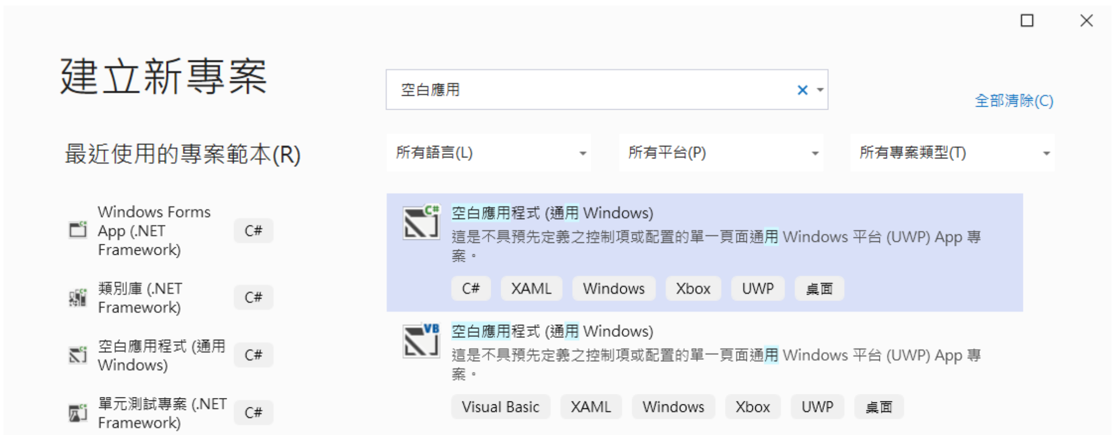


Figure 11 Execute DrawingForm

Step 10 Create View in Windows App

新增一個 Windows Store App 專案命名為 “DrawingApp” 。並且參照 **Step 6**，將 Model 的連結加入至該專案


建立新專案

空白應用

全部清除(C)

最近使用的專案範本(R)

所有語言(L)　所有平台(P)　所有專案類型(T)

Windows Forms App (.NET Framework)　C#

類別庫 (.NET Framework)　C#

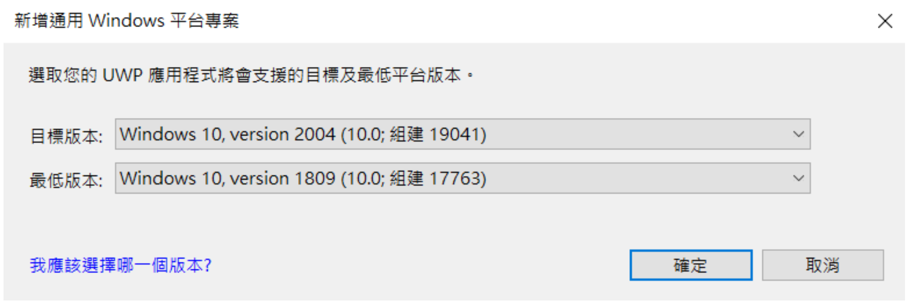
空白應用程式 (通用 Windows)　C#

單元測試專案 (.NET Framework)　C#

空白應用程式 (通用 Windows)
這是不具預先定義之控制項或配置的單一頁面通用 Windows 平台 (UWP) App 專案。
C#　XAML　Windows　Xbox　UWP　桌面

空白應用程式 (通用 Windows)
這是不具預先定義之控制項或配置的單一頁面通用 Windows 平台 (UWP) App 專案。
Visual Basic　XAML　Windows　Xbox　UWP　桌面

Figure 12-1 檔案 > 新增 > 專案 > 空白應用程式 (通用 Windoes)


新增通用 Windows 平台專案

選取您的 UWP 應用程式將會支援的目標及最低平台版本。

目標版本: Windows 10, version 2004 (10.0; 組建 19041)

最低版本: Windows 10, version 1809 (10.0; 組建 17763)

我應該選擇哪一個版本?

確定　取消

Figure 13-2 使用預設的 UWP 版本

Step 11　Create Canvas Model and Adaptor in DrawingApp

於 “DrawingApp” 專案加入下列檔案。

1. 對著專案點擊右鍵加入一個新資料夾，命名為 “PresentationModel”

2. 對著資料夾 “PresentationModel” 點擊右鍵，選擇加入新類別

(a) WindowsStoreGraphicsAdaptor.cs

由於 **Windows Store** 和 **Windows Form** 繪圖所使用的介面不相同，故需要使用 **Adaptor** 來做轉接。繼承的是 **IGraphics**，故一定會有 **ClearAll** 與 **DrawLine** 兩個介面，但介面實作的 **Code** 則依照不同的需求來決定。

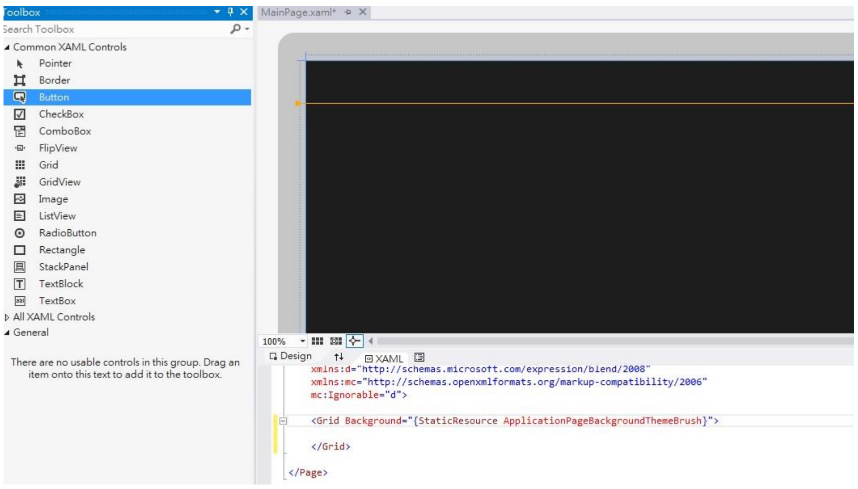
```
using Windows.UI;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Shapes;
using Windows.UI.Xaml.Media;
using DrawingModel;

namespace DrawingApp.PresentationModel
{
    class WindowsStoreGraphicsAdaptor : IGraphics
    {
        Canvas _canvas;

        public WindowsStoreGraphicsAdaptor(Canvas canvas)
        {
            this._canvas = canvas;
        }

        public void ClearAll()
        {
            _canvas.Children.Clear();
        }

        public void DrawLine(double x1, double y1, double x2, double y2)
        {
            Windows.UI.Xaml.Shapes.Line line = new Windows.UI.Xaml.Shapes.Line();
            line.X1 = x1;
            line.Y1 = y1;
            line.X2 = x2;
            line.Y2 = y2;
            line.Stroke = new SolidColorBrush(Colors.Black);
            _canvas.Children.Add(line);
        }
    }
}
```

(b) PresentationModel.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Windows.UI.Xaml.Controls;
using DrawingModel;

namespace DrawingApp.PresentationModel
{
    class PresentationModel
    {
        Model _model;
        IGraphics _igraphics;

        public PresentationModel(Model model, Canvas canvas)
        {
            this._model = model;
            _igraphics = new WindowsStoreGraphicsAdaptor(canvas);
        }

        public void Draw()
        {
            // 重複使用igraphics物件
            _model.Draw(_igraphics);
        }
    }
}
```

Step 12　Create View in DrawingApp

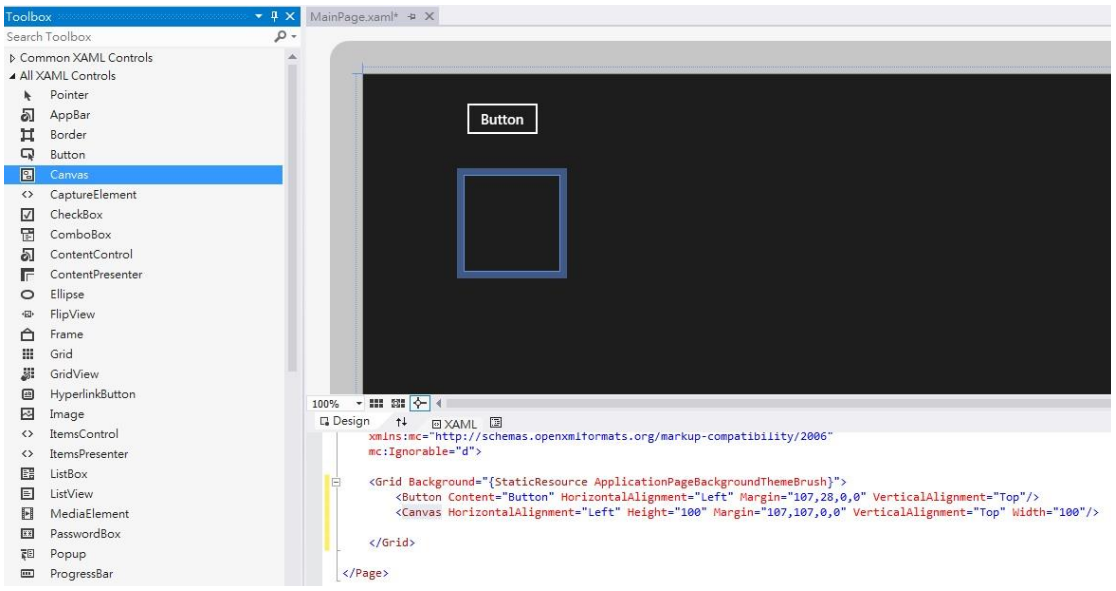
於 “DrawingApp” 專案點選 MainPage.xaml，初始畫面為一個空白的視窗，你可以將需要的元件使用拖拉的方式放置上來。



```
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">

    </Grid>

</Page>
```

Figure 14 Initial Designer

於此 App，你會需要一個 Button 與一個 Canvas



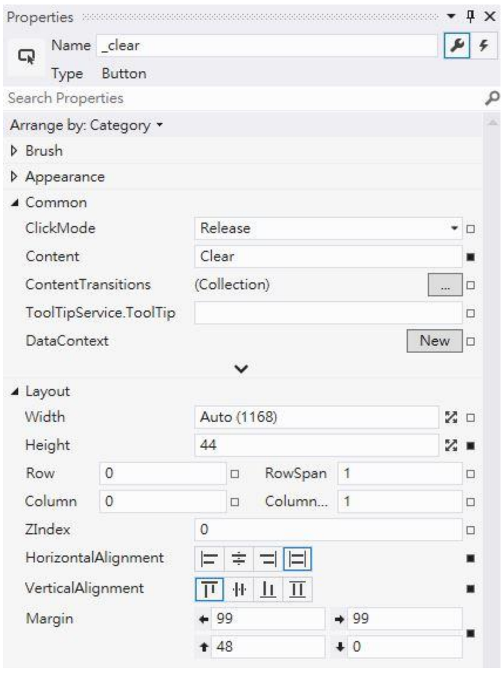
```
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <Button Content="Button" HorizontalAlignment="Left" Margin="107,28,0,0" VerticalAlignment="Top"/>
        <Canvas HorizontalAlignment="Left" Height="100" Margin="107,107,0,0" VerticalAlignment="Top" Width="100"/>

    </Grid>

</Page>
```

Figure 15 App Contains a Button and a Canvas

當你需要修改元件屬性時，你可以使用 Properties 視窗進行設定，當然，你也可以直接修改 XML。於此 App 中，你需要將 Button 顯示的文字修改成 “Clear” ，為了讓 Button 可以保持在最上面，故需要將 Vertical Alignment 設定成 “Top” ，由於只有一個 Button 所以我們希望可以將其占滿整個空間，故需要設定 HorizontalAlignment 為 “Stretch” 。為了美觀考量，可以修改 Margin，讓元件與

元件或元件與整個程式的視窗保持一定的距離。最後，請務必記得，一定要修改

該元件的命名。


Properties

Name _clear

Type Button

Search Properties

Arrange by: Category

Brush

Appearance

Common

ClickMode	Release
Content	Clear
ContentTransitions	(Collection)
ToolTipService.ToolTip	
DataContext	New

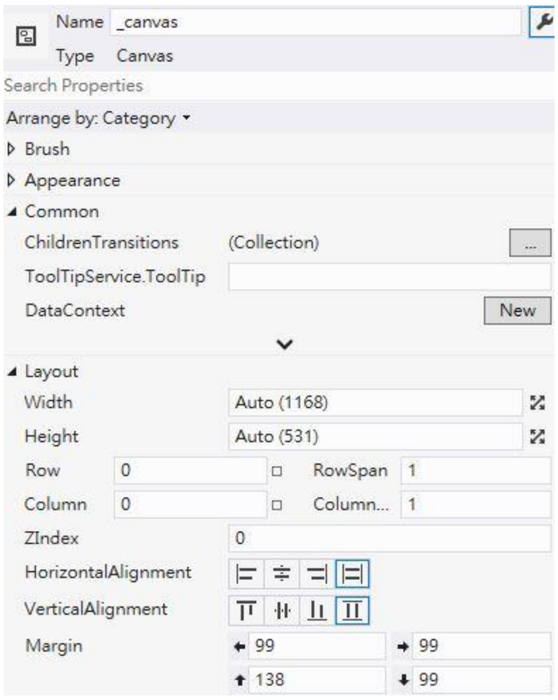
Layout

Width	Auto (1168)		
Height	44		
Row	0	RowSpan	1
Column	0	Column...	1
ZIndex	0		
HorizontalAlignment			
VerticalAlignment			
Margin	99		99
	48		0

Figure 16 Properties of the Button

由於預設 Canvas 背景顏色與 App 顏色相同，故我們希望可以將 Canvas 換成另一

種顏色，並且透過修改 Margin，讓 Canvas 可以擴展到最大。



Name _canvas
Type Canvas
Search Properties
Arrange by: Category
Brush
Appearance
Common
ChildrenTransitions (Collection)
ToolTipService.ToolTip
DataContext New
Layout
Width Auto (1168)
Height Auto (531)
Row 0 RowSpan 1
Column 0 Column... 1
ZIndex 0
HorizontalAlignment
VerticalAlignment
Margin 99 99 138 99

Figure 17 Properties of Canvas

最後產生出來的 XML 如下(建議同學可以透過 Designer 自行設計，惟 x:Name 須相同)：

```
<Page
    x:Class="DrawingApp.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:DrawingApp"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">
    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <Button x:Name="_clear" Content="Clear" HorizontalAlignment="Stretch"
Height="44" Margin="99,48,99,0" VerticalAlignment="Top"/>
        <Canvas x:Name="_canvas" Margin="99,138,99,99" Background="#FFFFFFA0"/>
    </Grid>
</Page>
```

然後開啟 MainPage.xaml.cs 檔案(將在 Solution Explorer 的 MainPage.xaml 展開即

可看到)，並且加入以下程式碼，黃底部分，為我們所撰寫的程式碼。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using Windows.Foundation;
using Windows.Foundation.Collections;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Controls.Primitives;
using Windows.UI.Xaml.Data;
using Windows.UI.Xaml.Input;
using Windows.UI.Xaml.Media;
using Windows.UI.Xaml.Navigation;

// The Blank Page item template is documented at
http://go.microsoft.com/fwlink/?LinkId=234238

namespace DrawingApp
{
    /// <summary>
    /// An empty page that can be used on its own or navigated to within a Frame.
    /// </summary>
    public sealed partial class MainPage : Page
    {
        DrawingModel.Model _model;
        PresentationModel.PresentationModel _presentationModel;

        public MainPage()
        {
            this.InitializeComponent();
            _model = new DrawingModel.Model();
            _presentationModel = new PresentationModel.PresentationModel(_model,
_canvas);
            _canvas.PointerPressed += HandleCanvasPressed;
            _canvas.PointerReleased += HandleCanvasReleased;
            _canvas.PointerMoved += HandleCanvasMoved;
            _clear.Click += HandleClearButtonClick;
            _model._modelChanged += HandleModelChanged;
        }

        /// <summary>
        /// Invoked when this page is about to be displayed in a Frame.
        /// </summary>
        /// <param name="e">Event data that describes how this page was reached.
The Parameter
        /// property is typically used to configure the page.</param>
        protected override void OnNavigatedTo(NavigationEventArgs e)
        {
        }

        private void HandleClearButtonClick(object sender, RoutedEventArgs e)
        {
            _model.Clear();
        }

        public void HandleCanvasPressed(object sender, PointerRoutedEventArgs e)
        {
            _model.PointerPressed(e.GetCurrentPoint(_canvas).Position.X,
e.GetCurrentPoint(_canvas).Position.Y);
        }

        public void HandleCanvasReleased(object sender, PointerRoutedEventArgs e)
        {
            _model.PointerReleased(e.GetCurrentPoint(_canvas).Position.X,
e.GetCurrentPoint(_canvas).Position.Y);
        }

        public void HandleCanvasMoved(object sender, PointerRoutedEventArgs e)
        {
            _model.PointerMoved(e.GetCurrentPoint(_canvas).Position.X,
e.GetCurrentPoint(_canvas).Position.Y);
        }
        public void HandleModelChanged()
        {
            _presentationModel.Draw();
        }
    }
}
```

Step 13　Execute DrawingApp

執行 DrawingApp 專案，你將可以看到 App 的執行結果。

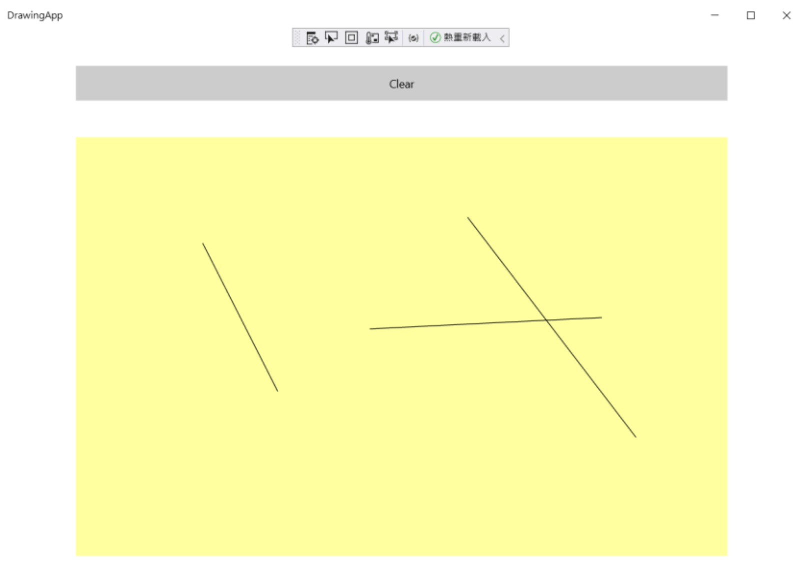


Figure 17　執行結果

- The End -