



好比说UWP看作是一个浏览器，里面的按键可以跳转到不同的窗口。Xamarin发展 学习uwp可以为以后深入xamarin和。。。打下基础。

**按下 Ctrl K 和 X，以開啟 [插入程式碼片段] 快顯視窗。**

**知識檢定**

1. 您應該用來撰寫 Windows 10 應用程式的工具名稱為何？visual studio
2. 什麼時候要在 Windows 應用程式開發中使用 XAML 標記語言？定義UWP和WPF應用程式中的使用者介面。
3. 如果您要開發下列類型的應用程式，則需要安裝 .NET 工作負載：WPF,winForms
4. Windows 開發人員帳戶可讓您執行什麼作業？將您的應用程式提交到WIN STORE
5. 在此課程模組中，您將了解如何使用三種不同的 Windows 技術，以三種不同的方式撰寫相同的應用程式。 您將會使用 C#，但結合三種不同的開發架構：通用 Windows 平台 (UWP)、Windows Presentation Foundation (WPF) 和 Windows Forms。
6. 我們會接著介紹 Windows Template Studio，而您將了解如何以最省力的方式快速產生新式應用程式的核心。

Windows 10 引進了[通用 Windows 平台](https://docs.microsoft.com/windows/uwp/get-started/universal-application-platform-guide) (UWP)，提供可在所有執行 Windows 10 的裝置上使用的通用應用程式平台。 UWP 核心 API 在所有 Windows 裝置上是相同的。 如果您的應用程式只使用核心 API，它會在任何 Windows 10 裝置 (無論是桌上型電腦、Xbox 或混合實境耳機等) 上執行。

如果您的目標是 Windows 10，而且您想要不受限地存取最新的 Windows 10 API 和功能，最好使用 UWP。 UWP 也可提供新式應用程式平台的其他優勢。 其中包括以 XAML 為基礎的 UI 模型 (其內建適應 DPI 和螢幕大小變更的支援)、使用者可明確授權應用程式存取裝置功能的安全性模型，以及支援全新安裝並且能夠直接發佈至 Microsoft Store 的封裝模型。

[Windows Presentation Foundation](https://docs.microsoft.com/visualstudio/designers/getting-started-with-wpf) (WPF) 是可供建立桌面用戶端應用程式的 UI 架構。 WPF 開發平台支援一組廣泛的應用程式開發功能，包含應用程式模型、資源、控制項、圖形、版面配置、資料繫結、文件和安全性。 它是 .NET Framework 的子集，所以若您先前已使用 ASP.NET 或 Windows Forms 透過 .NET Framework 建置應用程式，應該會對此程式設計體驗感到熟悉。 WPF 使用可延伸應用程式標記語言 (XAML) 來提供應用程式設計的宣告式模型。如果您需要以 Windows 10 之前的作業系統版本為目標，而且想擁有比 C++ 和 Win32 更高層級的編碼生產力，最好使用 WPF 和 Windows Forms。 您也可以使用可攜式類別庫，與其他 Microsoft 平台共用一些 .NET Framework 程式碼和程式庫。

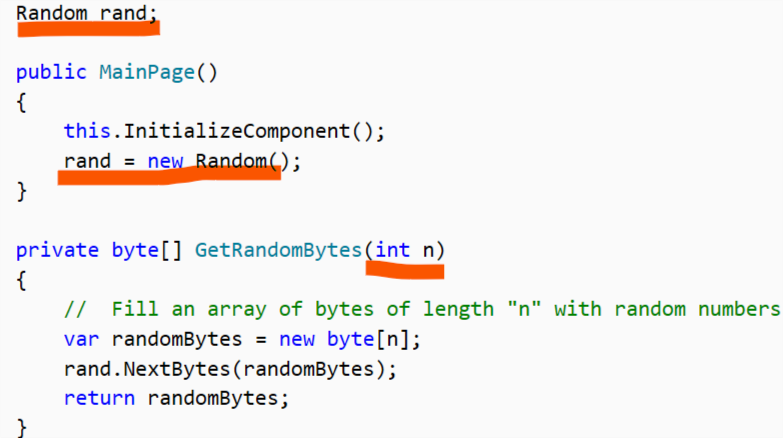
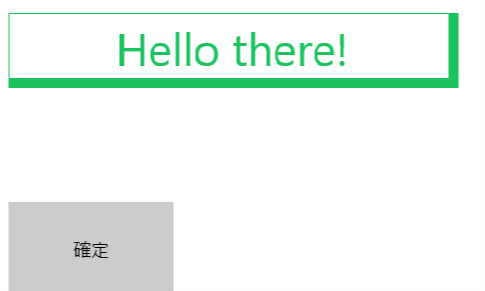
[Windows Forms](https://docs.microsoft.com/dotnet/framework/winforms/)是一種適用於 .NET Framework 的智慧型用戶端技術，它是一組 Managed 程式庫，可簡化常見的應用程式工作，例如讀取和寫入檔案系統。 當您使用類似 Visual Studio 的開發環境時，可以建立 Windows Forms 智慧型用戶端應用程式來顯示資訊、要求使用者輸入，以及透過網路與遠端電腦通訊。 Windows Forms 是較舊的開發平台，缺少 WPF 和 UWP 所提供的一些功能。

Visual Studio 中的通用 Windows 平台 (UWP) 是一項令人興奮的技術，可讓單一應用程式在任何裝置上執行：桌上型電腦、膝上型電腦、平板電腦、Xbox One、Surface Hub 或 Hololens 裝置。 針對 UWP 開發與針對 Windows Presentation Foundation (WPF) 開發有許多相似之處，但也有一些細微的差異。 本教學課程會說明這些差異，而且是進入 UWP 程式設計世界的第一個步驟。

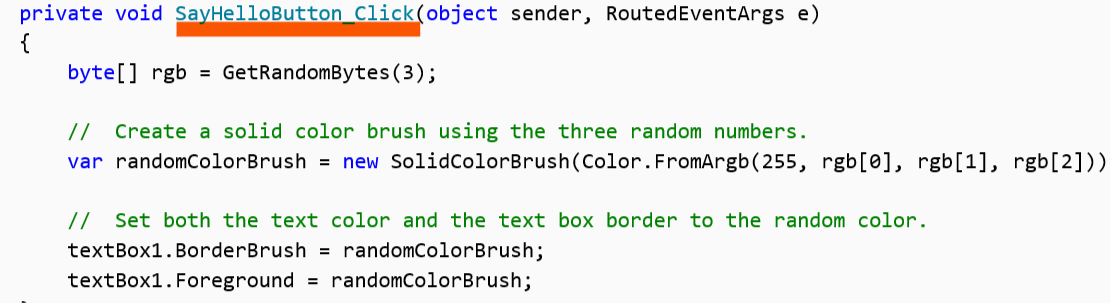
在本單元中，我們將：

* 使用 Visual Studio 來建立新的 UWP 專案。
* 將 UI 和程式碼元素新增至專案，以建立有趣的 "hello world" 程式變化。 每次按一下 "Say hello" 按鈕時，"Hello there!" 文字和框線就會隨機改變色彩。
* 了解如何設定屬性及建立事件。

每次按下按鈕時，應用程式應會以隨機色彩呈現 hello 文字。 所以我們需要新增全域 [隨機] 物件，而 **GetRandomBytes** 方法會隨機填入我們將作為 RGB 設定的值



最後，我們需要從**按鈕按下事件**呼叫 **GetRandomBytes**，以從傳回的位元組陣列建立隨機色彩，然後將文字方塊屬性變更為該色彩。 您完成的 **SayHelloButton\_Click** 方法



撰寫 UWP 和 WPF 應用程式的差別為何？

UWP應用程式可以存取自己的執行階段API，不一定會使用.NET。

WTS用途爲何？

建立具有你所需基本功能嘅新UWP應用程式。

**建置使用 Azure 認知服務的應用程式=應用程式將文字傳送至bing影像搜尋並擷取了包含影像URL的回應。**

Bing 圖像式搜尋 API 可對 Bing.com/images 上顯示的影像提供類似的影像詳細資料。 上傳影像或提供影像 URL，此 API 即可識別各種不同的詳細資料，包括外觀上類似的影像、購物來源、包含影像的網頁等等。https://docs.microsoft.com/en-us/azure/cognitive-services/bing-visual-search/

**TextBox** (用以輸入搜尋字詞) 和一個**影像**控制項 (用以顯示影像)。

現在，我們可以開始編寫會從 **TextBox** 中取得文字的 C# 程式碼、將其傳送至 Azure 服務，並取得傳回的影像連結。

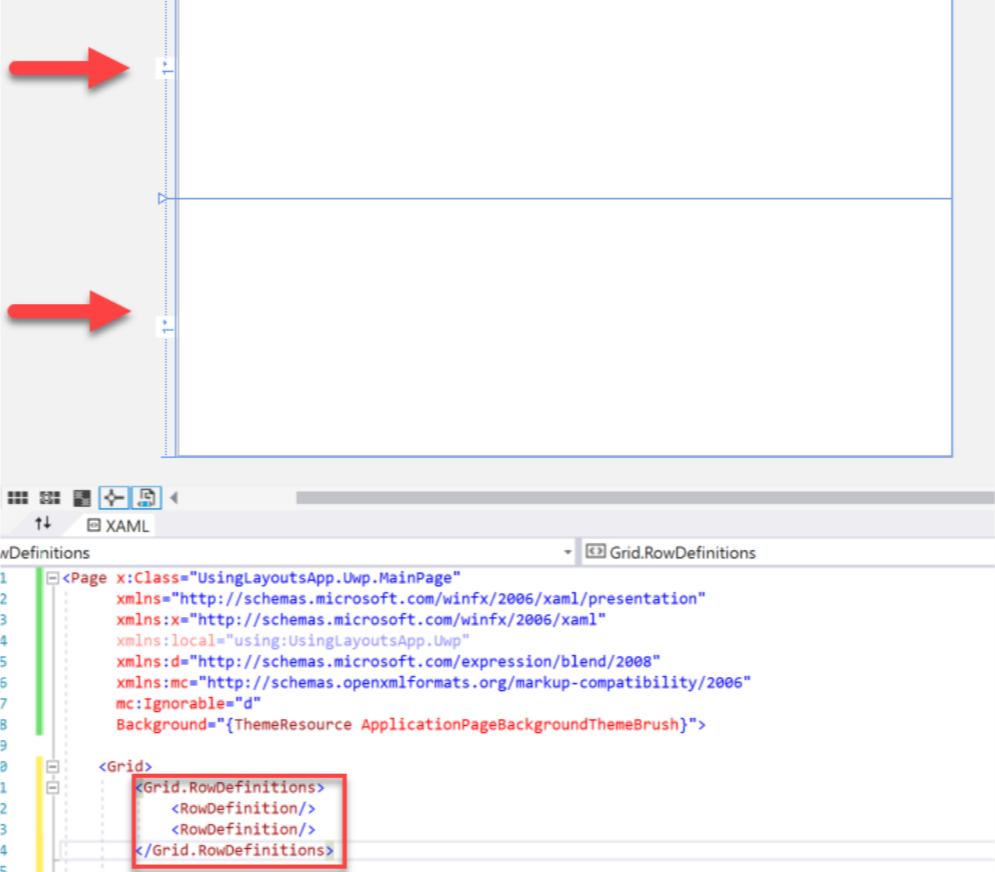
[JSON](https://www.wikipedia.org/wiki/JSON) 是在應用程式與服務之間共用資料時採用的標準格式。 JSON 以文字為基礎，因此您可以在「記事本」中加以檢視，且其中包含不同的欄位和值。 您可以利用一些程式碼程式庫來讀取和寫入 JSON 格式的資料。

**建置可存取網際網路入口網站的應用程式**

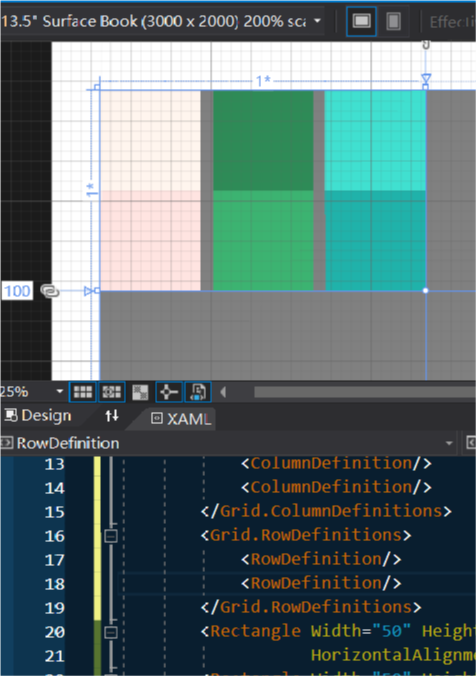
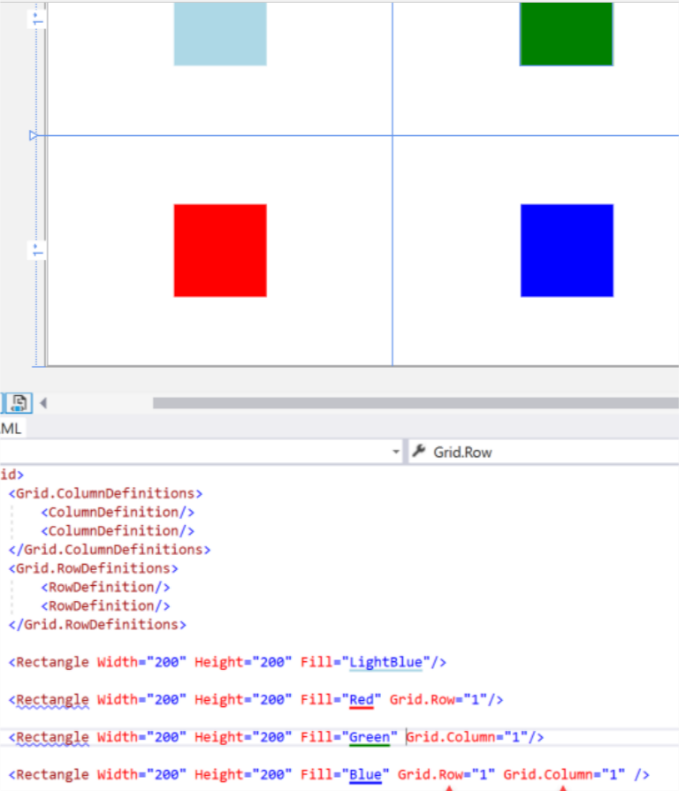
**正確！ UWP 中可用的 API 可讓您直接載入和儲存設定。**

****

**Content**

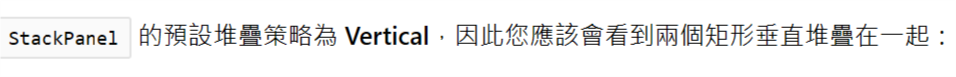
**兩行列。。。Grid.Row="0" 和 Grid.Column="0" 是格線的任何子系的預設值。 您可以選擇設定此項或將其保留**

您會看到它被放置在 Grid 的左上角。 這是因為預設情況下，任何子項目的資料列和資料行都設定為 0 (資料列定義和資料行定義使用以零為起始的編號系統)

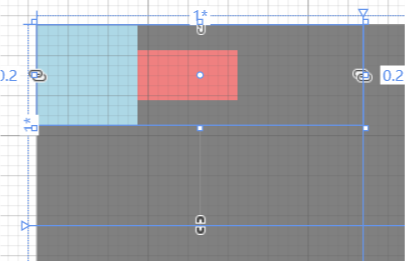


您會看到第一個資料列被壓縮到最小的空間量，**即該資料列中內容的高度**。

第二個資料列會填滿其餘的可用空間，因為它仍然是星狀大小 (請記住，您不必輸入 Height=""\*，因為這是預設值\*\*)。

****

** **

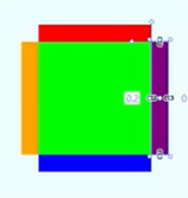
****

<StackPanel Orientation="Horizontal" VerticalAlignment="Top">

Textblock text=”” button content=””

**1.滚动**<ScrollViewer HorizontalScrollBarVisibility="Auto" VerticalScrollBarVisibility="Auto">

**2. RelativePanel**

****

3.visualstatemanager

4.DATA BINDING 数据源与数据展示者之间的协同关系 x:bind

5.汉堡菜单，展开和关闭！推出来DisplayMode="CompactInline" <!--DisplayMode="Overlay"-不显示的-> compactoverlay 展出来盖住！

6. <StackPanel Orientation="Horizontal">

<SplitView Name="mySplitView" OpenPaneLength="70"

CompactPaneLength="20" DisplayMode="CompactOverlay">

<SplitView.Pane>

<StackPanel>

<TextBlock Text="first"/>

<TextBlock Text="second"/>

<TextBlock Text="third"/>

</StackPanel>

</SplitView.Pane>

<SplitView.Content>

<StackPanel>

<TextBlock Text="fourth"/>

<TextBlock Text="fifth"/>

<TextBlock Text="sixth"/>

</StackPanel>

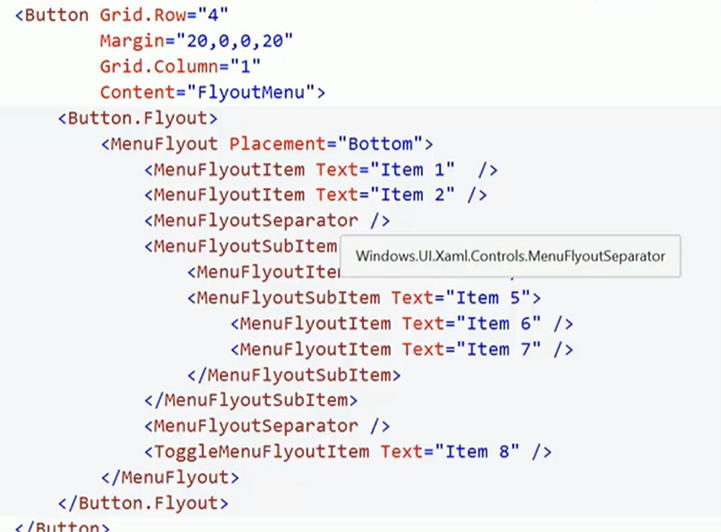
</SplitView.Content>

</SplitView>

<Button Content="ClickME" Click="Button\_Click"/>

</StackPanel>

7. mySplitView.IsPaneOpen = !mySplitView.IsPaneOpen;

8. 跳出个菜单

9. 建议列表菜单 autosuggetbox

10. slider

11.progressbar

12.progressring

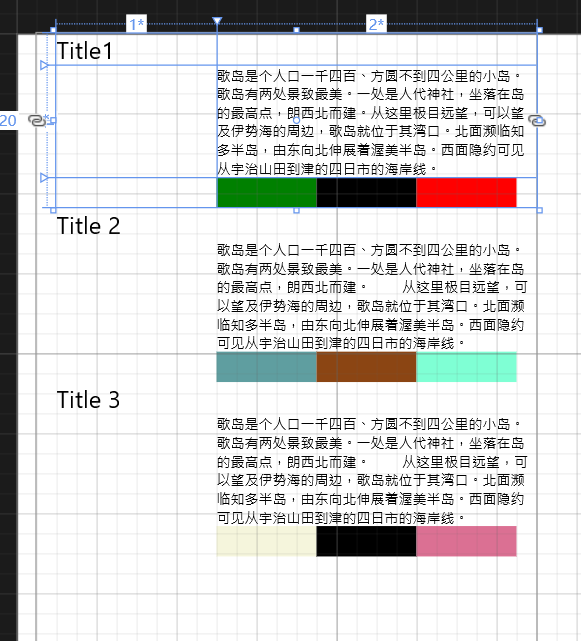
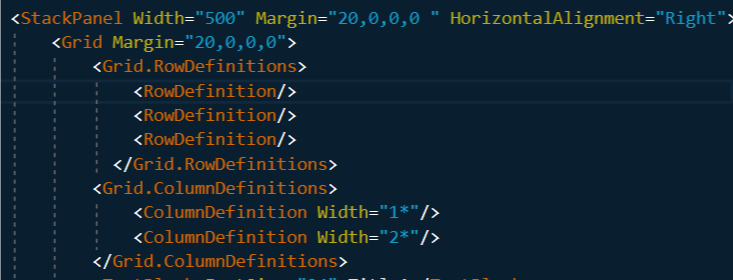
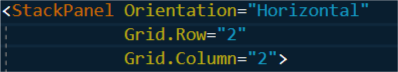
13. you can change resize, reposition, set the visibility.

14. <StackPanel >之后写的东西会自动规矩排好，<StackPanel Orientation="Horizontal" >是水平横着排布。 vertical

15. VerticalAlignment="Top" center 垂直排布是在什么位置剧中，置顶，底部

15. HorizontalAlignment="Right" 排布的位置是整体在右边。

16 做个stackpanel 整体（vertical），



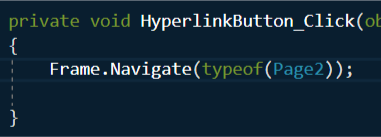
17 跳转 navigation：

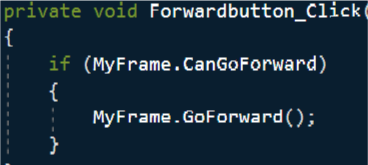
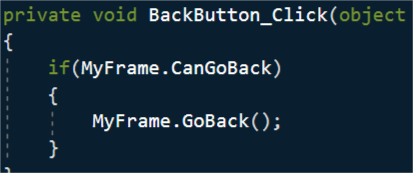
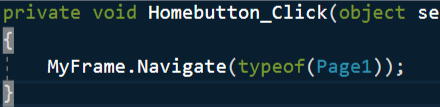
<StackPanel Orientation="Horizontal"> 排在一起，每个按钮写上x：name



<HyperlinkButton Content="go to page 2" Click="HyperlinkButton\_Click"/>



之后按下按钮跳转到

<Frame Name="MyFrame"></Frame>。Mainpage.xaml.cs文件添加MyFrame.Navigate(typeof(Page1)); 

18．<TimePickerGrid.Row="0"Grid.Column="1"ClockIdentifier="12HourClock"Margin="20,0,0,200"/>

19 <CalendarDatePicker

Grid.Row="1" Grid.Column="1"

Margin="20,0,0,20"

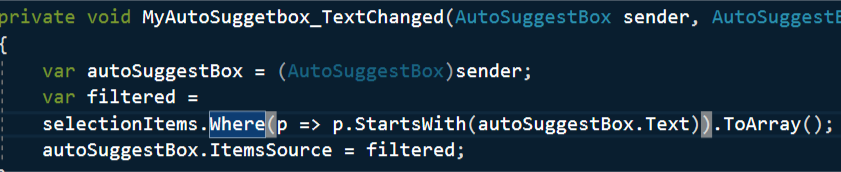
PlaceholderText="choose a date"

DisplayMode="Year"/>

20 **找东西搜索箱** <AutoSuggestBox x:Name="myAutoSuggetbox"QueryIcon="Emoji"PlaceholderText="Find Something" **TextChanged** =" MyAutoSuggetbox\_TextChanged" />之后要写数组

private string[] selectionItems = new string[]

{"Ferdinad","frank","falary","jack","Tom","todd"}; **写事件**



21. 进度圈圈 <ProgressRing x:Name="myprogressRing" HorizontalAlignment="Left" Width="50" Height="50" IsActive="True"/>

22.进度<ProgressBarx:Name="myprogreeBar"Value="{x:Bind **MySlider.Value**,Mode=OneWay}"/> 与myslider值绑在一起。

23.**添加字典**.xzaml，在字典里面写属性

<SolidColorBrush x:Key="brush" Color="Red"/>

<x:String x:Key="greeting"> Hello world </x:String>

之后回main里面添加引用开头：

<Page.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Dictionary1.xaml"/>

<ResourceDictionary Source="Dictionary2.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Page.Resources>

到main主页：只需要绑定属性

<TextBlock Text="{StaticResource greeting}" Foreground="{StaticResource brush}"/>

在**APP.XAML**添加元素属性：

<Application.Resources>

<SolidColorBrush x:Key="Mybrush" Color="Brown"/>

<Style TargetType="Button" x:Key="MyButtonStyle">

<Setter Property="Background" Value="Blue"/>

<Setter Property="FontFamily" Value="Arial"/>

<Setter Property="FontSize" Value="36"/>

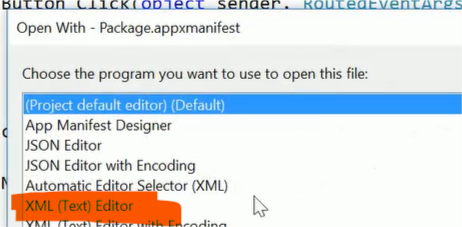
</Style>

</Application.Resources>

回到main绑定属性： <Button Content="MY BUTTON STYLE EXAMPLE" Height="100" Style="{StaticResource MyButtonStyle}"/>

24.　 添加拖拽

25. Mainfest 部署講解。。。





26. 上傳win10 store

-bit.do/tiles-bagdges

27．增強應用程式的UI 瀏覽應用程式對於包含多個頁面 (也稱為「檢視」) 的應用程式來說，是一個重要的功能。 除了直接瀏覽之外，您的應用程式可能也需要在其檢視之間傳遞資訊。 這裡，我們將示範如何接受來自使用者的輸入，並在使用者瀏覽時，將其傳遞到另一個檢視。**不同檢視之間瀏覽 在檢視之間傳遞資料**

Frame - 顯示網頁執行個體，並提供瀏覽功能。

Button事件裏面添加Frame.Navigate(typeof(Page1));。

之後返回事件添加if(Frame.CanGoBack) {Frame.GoBack();}

有時候您需要在第一頁和第二個頁之間傳遞資訊。 這類情況的其中一個好範例是，第一頁顯示產品清單，而選取一個產品可瀏覽至另一個頁面，以顯示該產品的詳細資料。 您必須設法依序傳遞所選產品的資訊，第二頁才能顯示詳細資料。若要將此盡量簡化，**Frame.Navigate** 方法有一個選項，可傳遞另一個參數，也就是「瀏覽參數」

**在 Page1 和 Page2 之間傳遞資料** 讓我們可以在瀏覽時，將數值或字串資料傳遞至 Page2**。**

**功能強大的 MediaElement 控制項** 在 Windows 傳統型應用程式中使用媒體非常簡單。 您不需要擔心媒體通訊協定或轉碼器。 因此，您可以將注意力放在應用程式的使用者體驗。MediaElement 可以從本機檔案來源或線上來源處理不同的媒體類型，如視訊和音訊。 根據預設，它將處理播放和串流處理媒體更複雜的層面，例如緩衝處理內容以獲得順暢的體驗。

讓我們建立簡單的媒體播放程式，在應用程式啟動時開始串流處理影片。

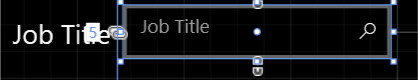
WebView 或 WebBrowser 控制項的 Navigate 方法如何運作？

**正確Navigate 用來顯示網頁和離線 HTML 內容。**在 UWP 中，MediaElement 控制項的 CurrentStateChanged 事件不存在 WPF 中。 此事件可讓您做什麼？ **CurrentStateChanged 事件可讓 UWP 應用程式更方便管理媒體播放，因為它們可以在單一位置處理所有邏輯。**

Windows 應用程式的頁面之間傳遞資訊最簡單的方式是什麼？

**正確！ Frame.Navigate 可以採用繼承自物件的任何一段資訊，作為傳遞至目標頁面的參數。**

28. TextChanged="JobTitle\_TextChanged" QuerySubmitted="JobTitle\_QuerySubmitted"

SuggestionChosen="JobTitle\_SuggestionChosen"/> 



29. <AutoSuggestBox Grid.Row="0" Grid.Column="2" HorizontalAlignment="Right"

x:Name="MyAutoSuggestBox" QueryIcon="Find"

PlaceholderText="Search搜寻"

Width="200" Margin="7,7,7,7" />

30. <i:Interaction.Behaviors>

<behaviors:NavigationViewHeaderBehavior

DefaultHeader="{x:Bind Selected.Content, Mode=OneWay}">

<behaviors:NavigationViewHeaderBehavior.DefaultHeaderTemplate>

<DataTemplate>

<Grid>

<TextBlock

Text="{Binding}"

Style="{ThemeResource TitleTextBlockStyle}"

Margin="{StaticResource SmallLeftRightMargin}" />

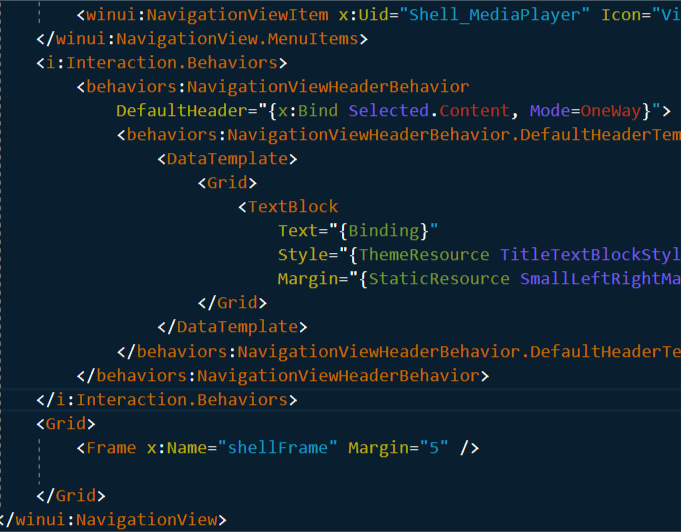
</Grid>

</DataTemplate>

</behaviors:NavigationViewHeaderBehavior.DefaultHeaderTemplate>

</behaviors:NavigationViewHeaderBehavior>

</i:Interaction.Behaviors>



31.

32.