



FPT POLYTECHNIC



LẬP TRÌNH C# 3

BÀI 8: WINDOWS PRESENTATION FOUNDATION

www.poly.edu.vn

- ◎ Windows Presentation Foundation
- ◎ Các control cơ bản WPF



- ❑ Windows Presentation Foundation hay gọi tắt là WPF – là một nền tảng cho phép developer có thể tạo ra các ứng dụng trên nền .NET framework
- ❑ Công nghệ WPF (Windows Presentation Foundation) là thế hệ kế tiếp của WinForm dùng lập trình các ứng dụng Windows phát triển trên nền tảng .NET 3.5 trở về sau
- ❑ WPF tăng cường khả năng lập trình giao diện của lập trình viên bằng cách cung cấp các API cho phép tận dụng những lợi thế về đa phương tiện hiện đại.

- ❑ WPF được xây dựng nhằm vào ba mục tiêu cơ bản
 - ❖ Cung cấp một nền tảng thống nhất để xây dựng giao diện người dùng
 - ❖ Cho phép người lập trình và người thiết kế giao diện làm việc cùng nhau một cách dễ dàng
 - ❖ Cung cấp một công nghệ chung để xây dựng giao diện người dùng trên cả Windows và trình duyệt Web
- ❑ Có thể gọi WPF là một GUI framework

□ Tạo ứng dụng WPF Application

Create a new project

Search for project templates 🔍 Language ▾

Recent project templates

- WPF App (.NET Framework) C#
- Console App (.NET Framework) C#
- Windows Forms App (.NET Framework) C#
- Console App C++
- ASP.NET Core Web Application C#
- Empty Project C++

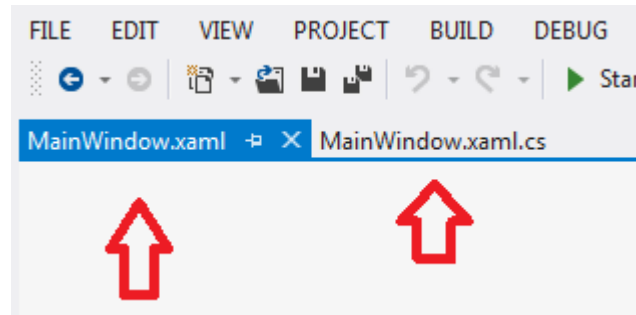
Filtering by: Desktop, Windows, C#

WPF App (.NET Framework)
Windows Presentation Foundation client application
C# Windows Desktop

WPF App (.NET Core)
Windows Presentation Foundation client application
C# Windows Desktop

Blank App (Universal Windows)
A project for a single-page Universal Windows Platform

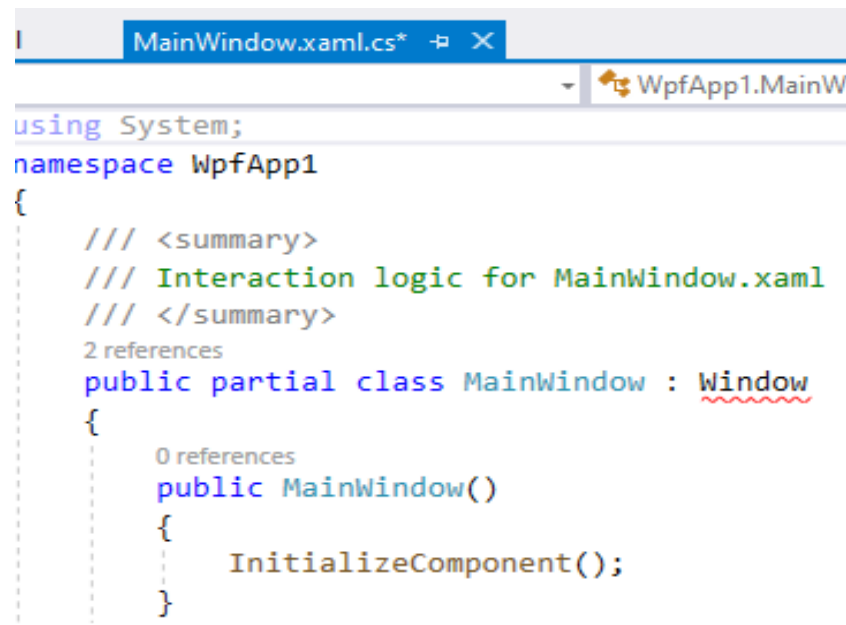
- ❑ Khi nhấn OK, VS sẽ phát sinh hai tập tin MainWindow.xaml và MainWindow.xaml.cs



- ❑ Khi chọn tab MainWindow.xaml chúng ta sẽ có giao diện trong như sau,



- ❑ tab MainWindow.xaml: là cửa sổ dùng cho việc thiết kế giao diện của ứng dụng. Chúng ta có thể thiết kế kéo-thả bằng cách dùng thanh Toolbox hay có thể viết mã trong khung XAML
- ❑ tab MainWindow.xaml.cs: là cửa sổ được dùng để viết các chức năng cho ứng dụng bằng C#



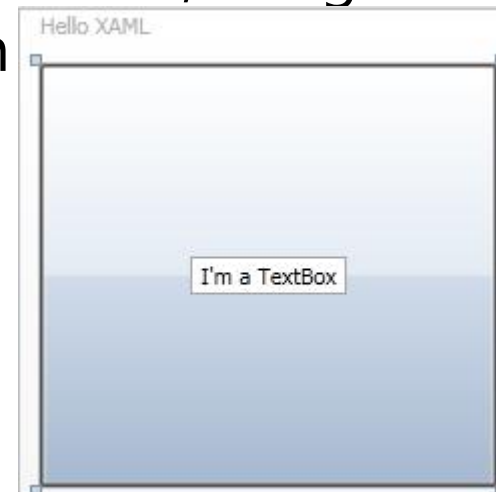
```
MainWindow.xaml.cs* X
WpfApp1.MainW
using System;
namespace WpfApp1
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    2 references
    public partial class MainWindow : Window
    {
        0 references
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}
```

❑ XAML là gì?

- ❖ Là viết tắt của từ eXtensible Application Markup Language, là biến thể của Microsoft dựa trên XML nhằm mô tả một GUI.

```
<Window x:Class="WpfApplication1.MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2019/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2019/xaml"
Title="Hello XAML" Height="250" Width="250">
    <Button>
        <TextBox>I'm a TextBox</TextBox>
    </Button>
</Window>
```

- ❖ Một thẻ XAML khi khai báo phải được kết thúc, bằng cách viết thẻ kết thúc hoặc đặt dấu gạch chéo "/" ở cuối thẻ: <Button> </Button>
Hoặc <Button />



❑ Thêm thuộc tính cho thẻ

❖ có thể thêm trực tiếp vào thẻ:

```
<Button FontWeight="Bold" Content="A button"/>
```

❖ hoặc có thể được định nghĩa bằng cách viết thẻ bắt đầu và thẻ kết thúc

```
<Button>
    <Button.FontWeight>Bold</Button.FontWeight>
    <Button.Content>A button</Button.Content>
</Button>
```

❑ Events (sự kiện) trong XAML: Trên hầu hết các control, bạn sẽ tìm thấy các sự kiện như KeyDown, KeyUp, MouseDown, MouseEnter, MouseLeave, MouseUp và nhiều sự kiện khác.

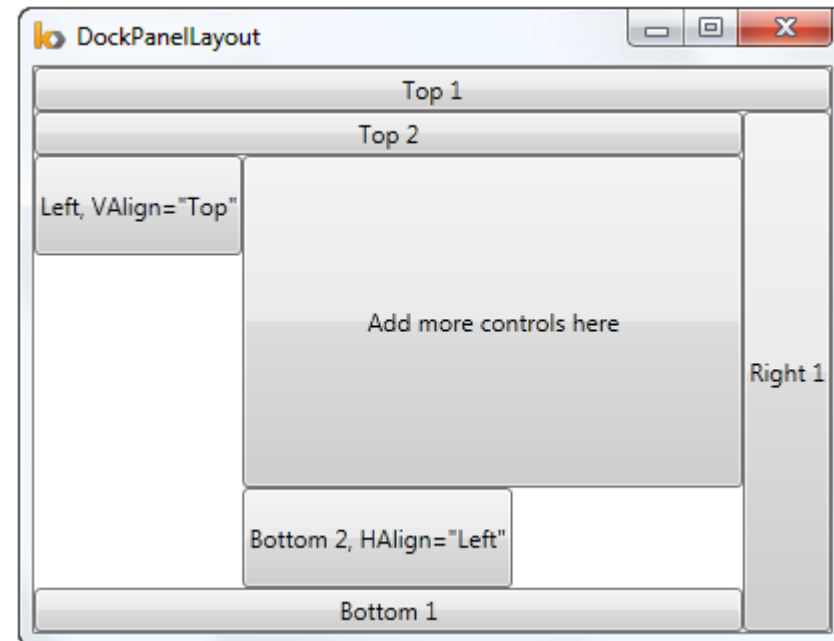
❑ Cách liên kết một sự kiện của control trong XAML đến một phần code trong file Code-behind

```
<Window x:Class="WpfTutorialSamples.XAML.EventsSample"
  xmlns="http://schemas.microsoft.com/winfx/2019/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2019/xaml"
  Title="EventsSample" Height="300" Width="300">
  <Grid Name="pnlMainGrid" MouseUp="pnlMainGrid_MouseUp" Background="LightBlue">

  </Grid>
</Window>
```

```
private void pnlMainGrid_MouseUp(object sender, MouseButtonEventArgs e)
{
    MessageBox.Show("You clicked me at " + e.GetPosition(this).ToString());
}
```

- ❑ Cách bố trí (Layout) là một vấn đề quan trọng và không thể thiếu trong việc thiết kế giao diện. WPF hỗ trợ nhiều kiểu layout tương ứng với mỗi control được gọi là layout container/layout control
- ❑ Các layout control thông dụng bao gồm: Canvas, StackPanel, WrapPanel, DockPanel, Grid

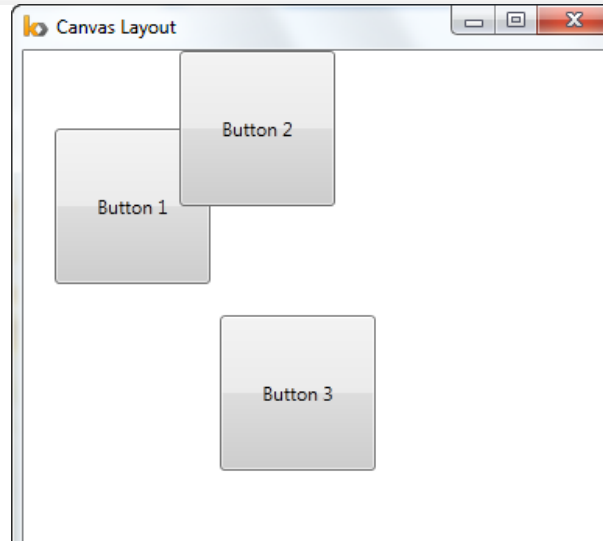


□ Canvas

- ❖ Cho phép bố trí các control bằng cách xác định vị trí cố định của chúng
- ❖ Gán tọa độ cho các control con thông qua các thuộc tính (attached property) Canvas.Left, Canvas.Right, Canvas.Top, Canvas.Bottom
- ❖ Cần lưu ý là Left và Top có độ ưu tiên cao hơn Right và Bottom. Nghĩa là nếu bạn gán giá trị cho cả Canvas.Left và Canvas.Right, khi đó tọa độ theo chiều ngang của control sẽ lấy từ Canvas.Left, tương tự với Canvas.Top và Canvas.Bottom.

Canvas

```
<Window
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Title="Canvas Layout">
  <Canvas Background="White">
    <Button Width="100" Height="100"
      Canvas.Left="20" Canvas.Top="50">Button 1</Button>
    <Button Width="100" Height="100"
      Canvas.Left="100" Canvas.Right="100">Button 2</Button>
    <Button Width="100" Height="100"
      Canvas.Right="150" Canvas.Bottom="50">Button 3</Button>
  </Canvas>
</Window>
```

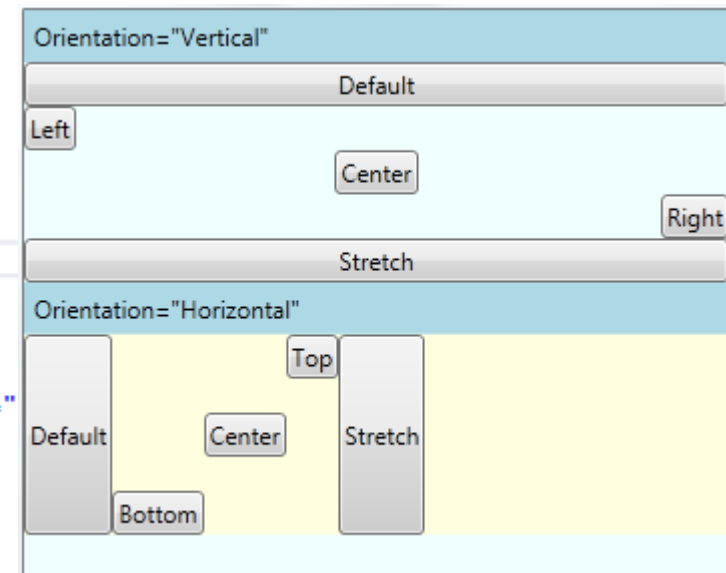


- ❑ **StackPanel**: layout này sẽ sắp xếp các control con của nó theo dòng hoặc cột tùy theo giá trị của thuộc tính **Orientation** là Vertical hay Horizontal. Các control con sẽ được sắp xếp với vị trí liên tiếp và không chồng lên nhau.

```
<StackPanel Orientation="Vertical" Background="Azure">
  <Label Background="LightBlue">Orientation="Vertical"</Label>
  <Button>Default</Button>
  <Button HorizontalAlignment="Left">Left</Button>
  <Button HorizontalAlignment="Center">Center</Button>
  <Button HorizontalAlignment="Right">Right</Button>
  <Button HorizontalAlignment="Stretch">Stretch</Button>

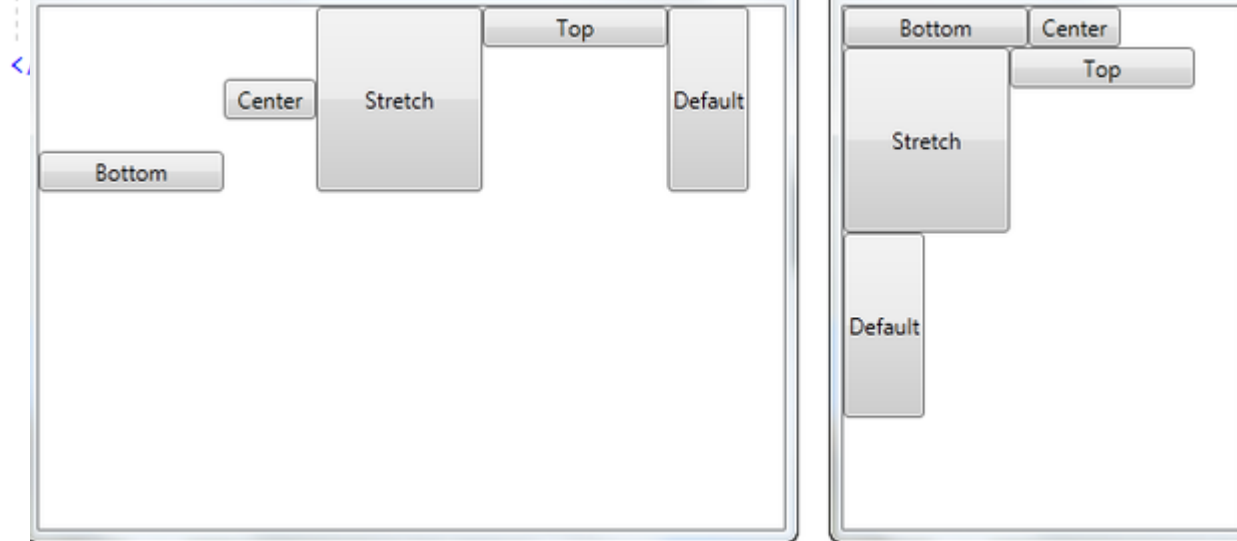
  <Label Background="LightBlue">Orientation="Horizontal"</Label>

  <StackPanel Orientation="Horizontal" Background="LightYellow" Height="100">
    <Button>Default</Button>
    <Button VerticalAlignment="Bottom">Bottom</Button>
    <Button VerticalAlignment="Center">Center</Button>
    <Button VerticalAlignment="Top">Top</Button>
    <Button VerticalAlignment="Stretch">Stretch</Button>
  </StackPanel>
</StackPanel>
```



- ❑ WrapPanel: sắp xếp các control lần lượt theo hàng hoặc cột, nhưng WrapPanel sẽ tự động cho các control sang hàng/cột mới nếu như kích thước của hàng/cột còn lại không đủ chứa control.

```
<WrapPanel Orientation="Horizontal">  
    <Button VerticalAlignment="Bottom" Width="100">Bottom</Button>  
    <Button VerticalAlignment="Center" Width="50">Center</Button>  
    <Button VerticalAlignment="Stretch" Width="90" Height="100">Stretch</Button>  
    <Button VerticalAlignment="Top" Width="100" >Top</Button>  
</WrapPanel>
```



- ❑ DockPanel: các control con sẽ được gắn thêm một attached property là DockPanel.Dock cho phép bạn gán các giá trị: Left, Right, Top, Bottom. Các giá trị tương ứng với mỗi phần không gian trong DockPanel, phần còn lại ở giữa sẽ được dùng để chứa các control khác
- ❑ Thuộc tính LastChildFill có giá trị mặc định true nhằm xác định các control thêm vào sau sẽ lấp đầy khoảng trống còn lại. Nếu bạn muốn giữ lại khoảng trống này, hãy gán giá trị của nó trở thành false.

□ DockPanel:

```
<DockPanel LastChildFill="True">
  <Button DockPanel.Dock="Top">Top 1</Button>
  <Button DockPanel.Dock="Right">Right 1</Button>
  <Button DockPanel.Dock="Top">Top 2</Button>
  <Button DockPanel.Dock="Bottom">Bottom 1</Button>
  <Button DockPanel.Dock="Left" VerticalAlignment="Top" Height="50">Left, VAlign="Top"</Button>
  <Button DockPanel.Dock="Bottom" HorizontalAlignment="Left" Height="50">Bottom 2, HAlign="Left"</Button>
  <Button>Add more controls here</Button>
</DockPanel>
```



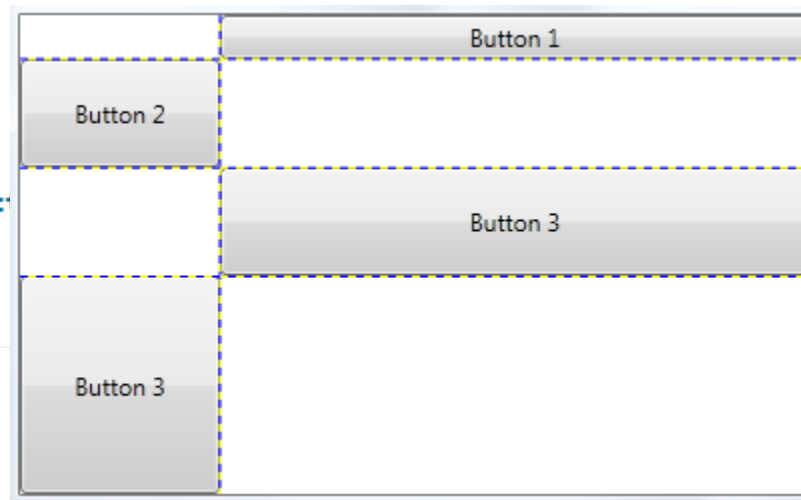
□ Grid

- ❖ Control này sắp xếp các control con trong một lưới giống như table trong html. Grid sẽ chia không gian thành các phần với kích thước tùy ý được ngăn cách bởi các đường lưới.
- ❖ Cần định nghĩa cấu trúc của Grid thông qua hai collection là Grid.RowDefinitions và Grid.ColumnDefinitions. Với mỗi dòng, cột bạn cần thêm một đối tượng RowDefinition/ColumnDefinition vào collection tương ứng
- ❖ Với mỗi dòng/cột, bạn có thể thiết lập kích thước cho chúng bằng cách dùng giá trị cố định, tự động (Auto) hoặc tỷ lệ.

- ❑ Ví dụ sau chia Grid làm 4 dòng, trong đó dòng 1 và dòng 2 bằng nhau, dòng 3 có kích thước gấp đôi dòng 1. Vì dòng đầu tiên có chiều cao là 100, nên chiều cao của dòng 1 sẽ bằng chiều cao của Grid chia cho 4.

```
<Grid ShowGridLines="True">
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="100"/>
    <ColumnDefinition/>
  </Grid.ColumnDefinitions>
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto"/>
    <RowDefinition Height="*/>
    <RowDefinition Height="*/>
    <RowDefinition Height="2*/>
  </Grid.RowDefinitions>
  <Button Grid.Column="1" Grid.Row="0" Height="50">Button 1</Button>
  <Button Grid.Column="0" Grid.Row="1">Button 2</Button>
  <Button Grid.Column="1" Grid.Row="2">Button 3</Button>
  <Button Grid.Column="0" Grid.Row="3">Button 4</Button>
</Grid>
```

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="50" />
    <RowDefinition Height="*" />
    <RowDefinition Height="*" />
    <RowDefinition Height="2*" />
  </Grid.RowDefinitions>
  ...
</Grid>
```





DEMO

-Hiện thực các ví dụ





LẬP TRÌNH C# 3

BÀI 8: WINDOWS PRESENTATION FOUNDATION(P2)

- Cho phép bạn hiển thị văn bản trên màn hình, giống như một Label, nhưng đơn giản và ít tốn tài nguyên hơn

```
<StackPanel>
  <TextBlock Margin="10" Foreground="Red">
    This is a TextBlock control<LineBreak />
    with multiple lines of text.
  </TextBlock>
  <TextBlock Margin="10" TextTrimming="CharacterEllipsis" Foreground="Green">
    This is a TextBlock control with text that may not be rendered completely,
    which will be indicated with an ellipsis.
  </TextBlock>
  <TextBlock Margin="10" TextWrapping="Wrap" Foreground="Blue">
    This is a TextBlock control with automatically wrapped text,
    using the TextWrapping property.
  </TextBlock>
</StackPanel>
```

- Các cách ngắt dòng: <LineBreak />, TextTrimming="CharacterEllipsis", TextWrapping="Wrap"

- ❑ Là control cơ bản để nhập liệu trong WPF, nó cho phép người dùng nhập văn bản đơn giản chỉ gồm các ký tự trên một dòng như hộp thoại nhập liệu, hoặc trên nhiều dòng như trình soạn thảo

```
<Grid Margin="10">  
    <TextBox AcceptsReturn="True" TextWrapping="Wrap" />  
</Grid>
```

- ❖ Thuộc tính AcceptReturn biến TextBox thành 1 control nhiều dòng (multi-line)
- ❖ thuộc tính TextWrapping cho phép văn bản tự động xuống dòng khi chạm điểm cuối của TextBox

- ❑ Hỗ trợ xử lý các sự kiện tương tác bởi người dùng

```
<Button Click="HelloWorldButton_Click">Hello, World!</Button>
```

- ❑ Trong phần Code-behind, bạn sẽ cần 1 hàm tương ứng để xử lý sự kiện click:

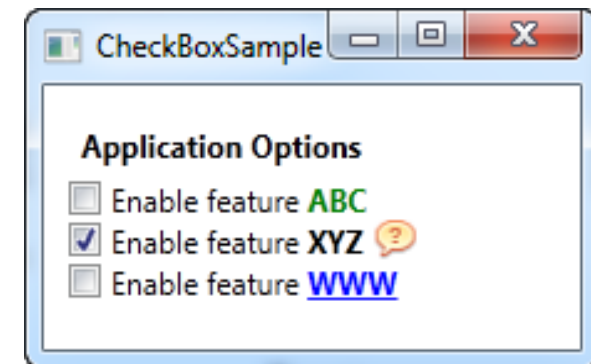
```
private void HelloWorldButton_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("Hello, world!");
}
```

- ❑ có thể thêm một vài text control với những định dạng khác nhau vào button

```
<Button>
    <StackPanel Orientation="Horizontal">
        <TextBlock>Formatted </TextBlock>
        <TextBlock Foreground="Blue" FontWeight="Bold" Margin="2,0">Button</TextBlock>
        <TextBlock Foreground="Gray" FontStyle="Italic">[Various]</TextBlock>
    </StackPanel>
</Button>
```


- ❑ CheckBox cho phép người dùng lựa chọn trạng thái tắt hặc mở (on or off), thường để phản ánh giá trị Boolean trong Code-behind

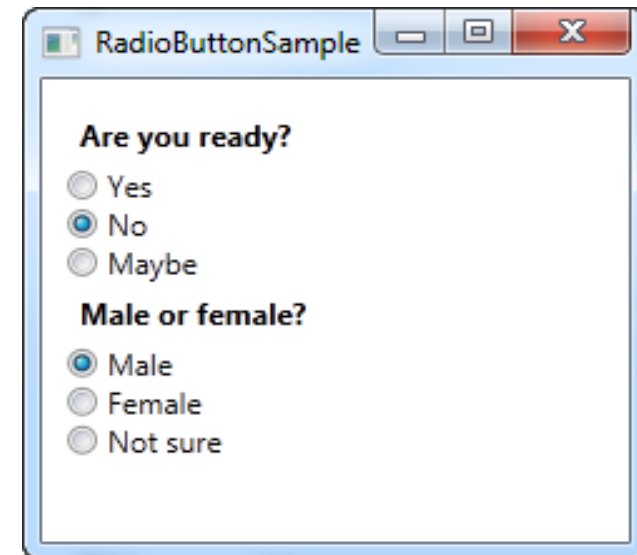
```
<StackPanel Margin="10">
    <Label FontWeight="Bold">Application Options</Label>
    <CheckBox>
        <TextBlock>
            Enable feature <Run Foreground="Green" FontWeight="Bold">ABC</Run>
        </TextBlock>
    </CheckBox>
    <CheckBox IsChecked="True">
        <WrapPanel>
            <TextBlock>
                Enable feature <Run FontWeight="Bold">XYZ</Run>
            </TextBlock>
            <Image Source="/WpfTutorialSamples;component/Images/question.png"
                Width="16" Height="16" Margin="5,0" />
        </WrapPanel>
    </CheckBox>
    <CheckBox>
        <TextBlock>
            Enable feature <Run Foreground="Blue" TextDecorations="Underline"
                FontWeight="Bold">WWW</Run>
        </TextBlock>
    </CheckBox>
</StackPanel>
```



- ❑ cho phép bạn đưa ra các danh sách các tùy chọn dành cho user, và chỉ được chọn một tùy chọn cùng một thời điểm

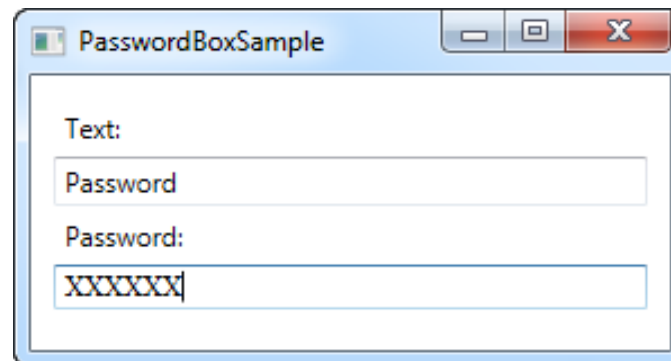
```
<StackPanel Margin="10">
    <Label FontWeight="Bold">Are you ready?</Label>
    <RadioButton GroupName="ready">Yes</RadioButton>
    <RadioButton GroupName="ready">No</RadioButton>
    <RadioButton GroupName="ready" IsChecked="True">Maybe</RadioButton>

    <Label FontWeight="Bold">Male or female?</Label>
    <RadioButton GroupName="sex">Male</RadioButton>
    <RadioButton GroupName="sex">Female</RadioButton>
    <RadioButton GroupName="sex" IsChecked="True">Not sure</RadioButton>
</StackPanel>
```



- hiển thị nội dung định dạng khác với các ký tự thực tế nhập vào mật khẩu, để bảo vệ các ký tự khỏi bị nhìn lén

```
<StackPanel Margin="10">  
    <Label>Text:</Label>  
    <TextBox />  
    <Label>Password:</Label>  
    <PasswordBox MaxLength="6" PasswordChar="X" />  
</StackPanel>
```

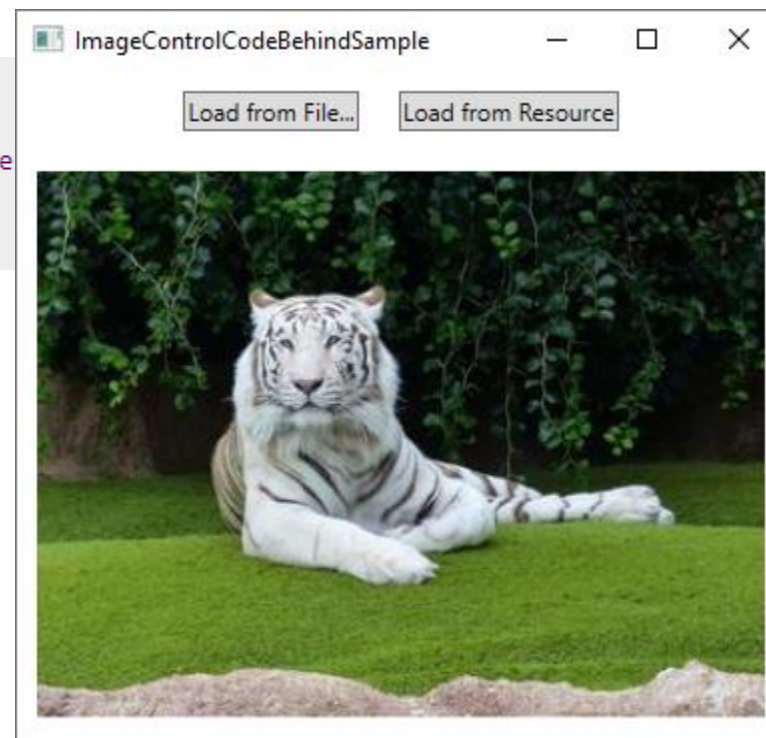


□ hiển thị nội dung là hình ảnh

```
<StackPanel>
    <WrapPanel Margin="10" HorizontalAlignment="Center">
        <Button Name="btnLoadFromFile" Margin="0,0,20,0"
            Click="BtnLoadFromFile_Click">Load from File...</Button>
        <Button Name="btnLoadFromResource" Click="BtnLoadFromResource_Click">Load from Resource</Button>
    </WrapPanel>
    <Image Name="imgDynamic" Margin="10" />
</StackPanel>
```

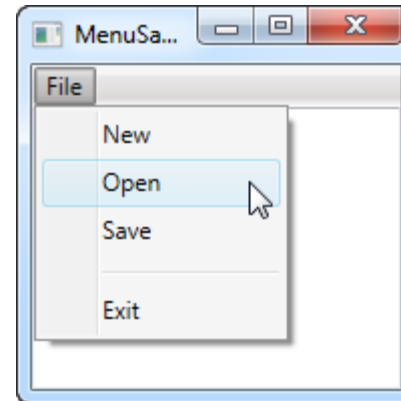
```
private void BtnLoadFromResource_Click(object sender, RoutedEventArgs e)
{
    Uri resourceUri = new Uri("/Images/white_bengal_tiger.jpg", UriKind.Relative);
    imgDynamic.Source = new BitmapImage(resourceUri);
}
```

```
private void BtnLoadFromFile_Click(object sender, RoutedEventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    if(openFileDialog.ShowDialog() == true)
    {
        Uri fileUri = new Uri(openFileDialog.FileName);
        imgDynamic.Source = new BitmapImage(fileUri);
    }
}
```



- cung cấp rất nhiều tùy chọn, chiếm rất ít không gian, chỉ cần thêm các phần tử MenuItem.

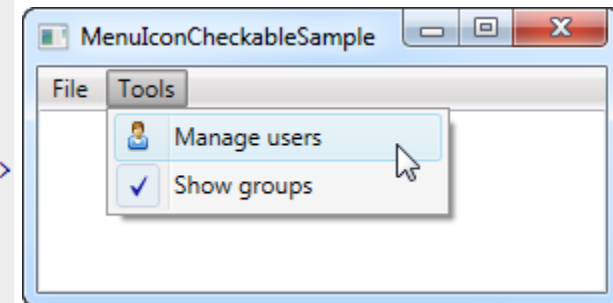
```
<DockPanel>
  <Menu DockPanel.Dock="Top">
    <MenuItem Header="_File">
      <MenuItem Header="_New" />
      <MenuItem Header="_Open" />
      <MenuItem Header="_Save" />
      <Separator />
      <MenuItem Header="_Exit" />
    </MenuItem>
  </Menu>
  <TextBox AcceptsReturn="True" />
</DockPanel>
```



- thuộc tính Header để định nghĩa tên của mục menu và bạn nên chú ý dấu gạch dưới trước ký tự đầu tiên của mỗi tên. Nó báo cho WPF sử dụng ký tự đó làm phím shortcut, có nghĩa là người dùng có thể nhấn phím Alt + ký tự đã cho để kích hoạt mục menu

- ❑ Icons và checkboxes: Hai tính năng phổ biến của một mục menu là biểu tượng, thường dùng để dễ xác định mục menu, xem nó là gì, và để dễ nhận biết được mục menu, có thể bật và tắt một tính năng cụ thể

```
<DockPanel>
  <Menu DockPanel.Dock="Top">
    <MenuItem Header="_File">
      <MenuItem Header="_Exit" />
    </MenuItem>
    <MenuItem Header="_Tools">
      <MenuItem Header="_Manage users">
        <MenuItem.Icon>
          <Image Source="/WpfTutorialSamples;component/Images/user.png" />
        </MenuItem.Icon>
      </MenuItem>
      <MenuItem Header="_Show groups" IsCheckable="True" IsChecked="True" />
    </MenuItem>
  </Menu>
  <TextBox AcceptsReturn="True" />
</DockPanel>
```



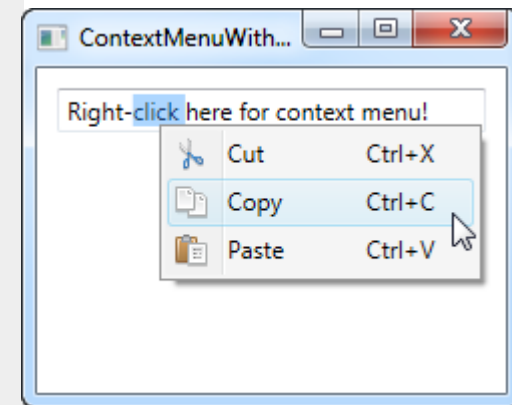
- ❑ Sự kiện Clicks: Khi người dùng click vào menu item, thường bạn sẽ muốn nó thực hiện chuyện gì đó. Cách đơn giản nhất là thêm một sự kiện cho MenuItem

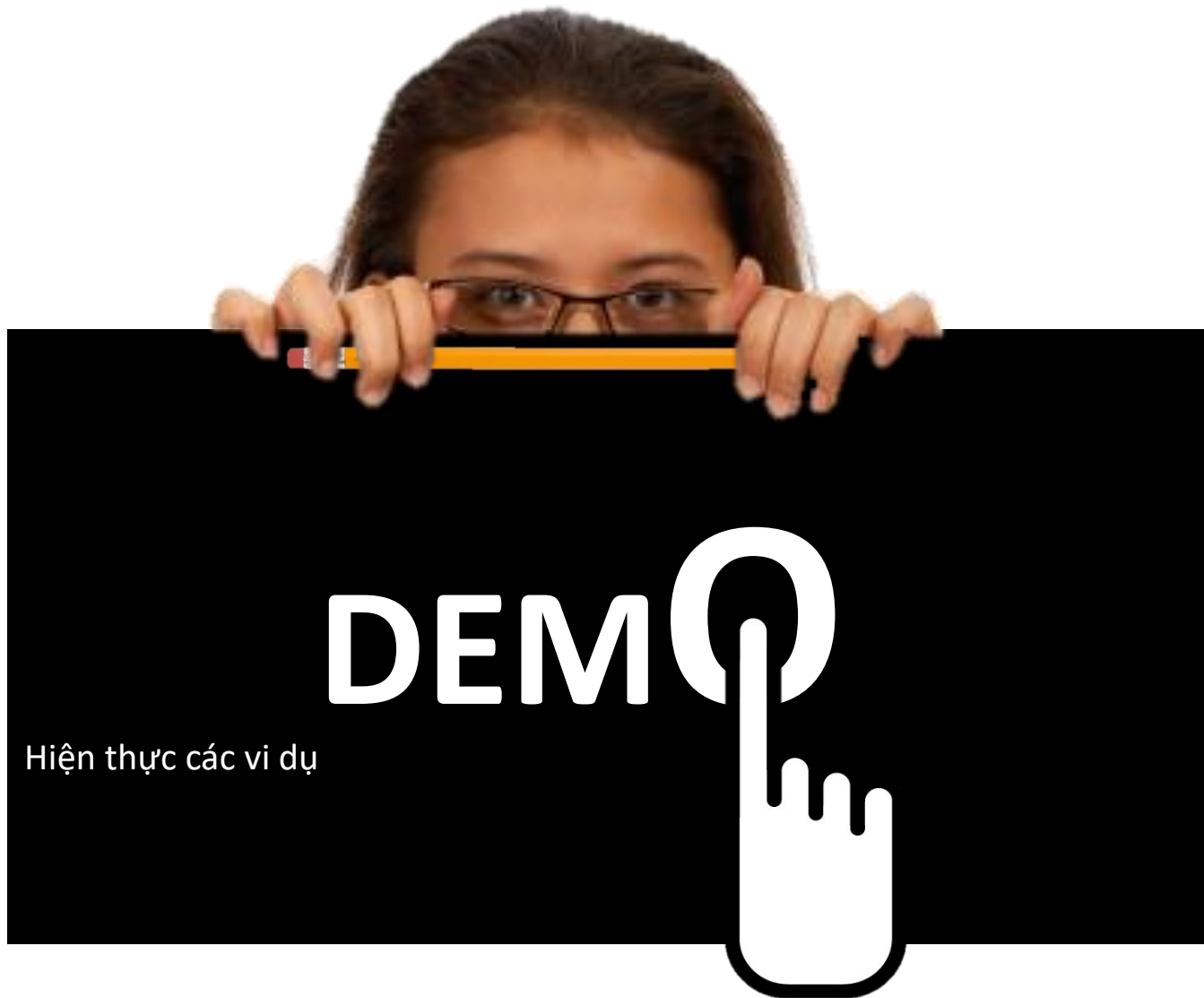
```
<MenuItem Header="_New" Click="mnuNew_Click" />
```

```
private void mnuNew_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("New");
}
```

- Thường xuất hiện khi user click phải chuột và popup hoặc trình đơn bật lên

```
<StackPanel Margin="10">
    <TextBox Text="Right-click here for context menu!">
        <TextBox.ContextMenu>
            <ContextMenu>
                <MenuItem Command="Cut">
                    <MenuItem.Icon>
                        <Image Source="/WpfTutorialSamples;component/Images/cut.png" />
                    </MenuItem.Icon>
                </MenuItem>
                <MenuItem Command="Copy">
                    <MenuItem.Icon>
                        <Image Source="/WpfTutorialSamples;component/Images/copy.png" />
                    </MenuItem.Icon>
                </MenuItem>
                <MenuItem Command="Paste">
                    <MenuItem.Icon>
                        <Image Source="/WpfTutorialSamples;component/Images/paste.png" />
                    </MenuItem.Icon>
                </MenuItem>
            </ContextMenu>
        </TextBox.ContextMenu>
    </TextBox>
</StackPanel>
```





Tổng kết bài học

- ◎ Windows Presentation Foundation
- ◎ Các control cơ bản WPF





KẾT THÚC