WPF
(windows Presentation Foundation)

إعداد

م. لمى السبع

WPF

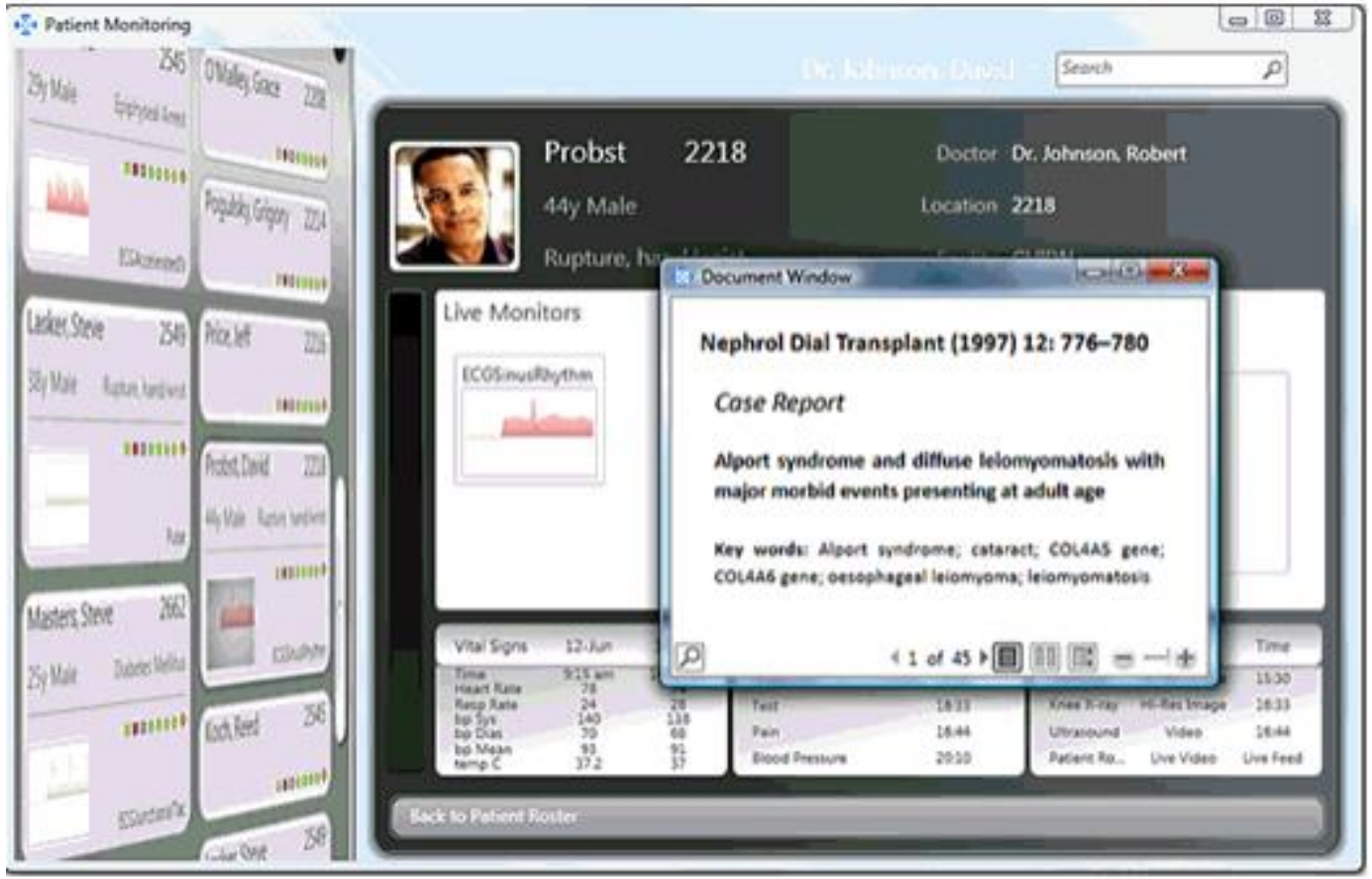
هو نظام جديد لبناء تطبيقات مع واجهات رسومية متطورة.

باستخدام wpf يمكن ان ننشئ تطبيقات من كلا النوعين windows application و web-based application.

من البرامج المطورة باستخدام wpf :

Yahoo! Messenger

Contoso HealthCare Sample Application



البرمجة باستخدام wpf

- Wpf هي جزء من بيئة عمل .NET. تستخدم فضاء الأسماء System.window.
- تشبه Asp.net , windows form بالإضافة أنها تقوم بتعريف صفوف ومتحولات واستدعاء توابع ومعالجة الأحداث وذلك باستخدام C# أو VB.

Wpf vs. windows Form

- محاسن wpf

١. أحدث لذلك فهي تتوافق مع المعايير .
٢. تستخدمها Microsoft لعدة تطبيقات مثل visual studio.
٣. مرونة عالية حيث يمكن تطوير أدوات تحكم كثيرة دون الحاجة لجهد في كتابتها أو شرائها.
٤. تستخدم wpf لغة xaml والذي يجعل من السهل إنشاء وتعديل الواجهات دون تغيير سلوك التطبيق، لأنّ xaml تفصل عمل المصمم (xaml) عن المبرمج (c#,vb).
٥. تحقق Databinding وهو فصل البيانات والعمليات عليها عن الواجهات والتصميمات الرسومية.
٦. تستخدم wpf لتطوير windows application and web based application.

- محاسن windows Form

١. أقدم أي أنها مجربة أكثر وتمّ اختبارها في عدة تطبيقات.
٢. يمكن الحصول على أدوات تحكم مطورة عن طريق شرائها أو يمكن الحصول عليها مجاناً .

أدوات التحكم في wpf

Buttons : Button , RepeatButton

Digital Ink : InkCanvas , InkPresenter

Documents : DocumentViewer , FlowDocumentPageViewer , FlowDocumentReader , FlowDocumentScrollViewer , StickyNoteControl

Input : TextBox , RichTextBox PasswordBox

Layout : Border , BulletDecorator , DockPanel , Canvas , Expander , Grid , GridView , GridSplitter , GroupBox , Panel , ResizeGrip ,Separator , ScrollBar ,ScrollViewer , StackPanel , Thumb , Viewbox , VirtualizingStackPanel , Window , WrapPanel

Media : Image , MediaElement , SoundPlayerAction

Menus : ContextMenu , Menu , ToolBar

Navigation : Frame , Hyperlink , Page , NavigationWindow , TabControl

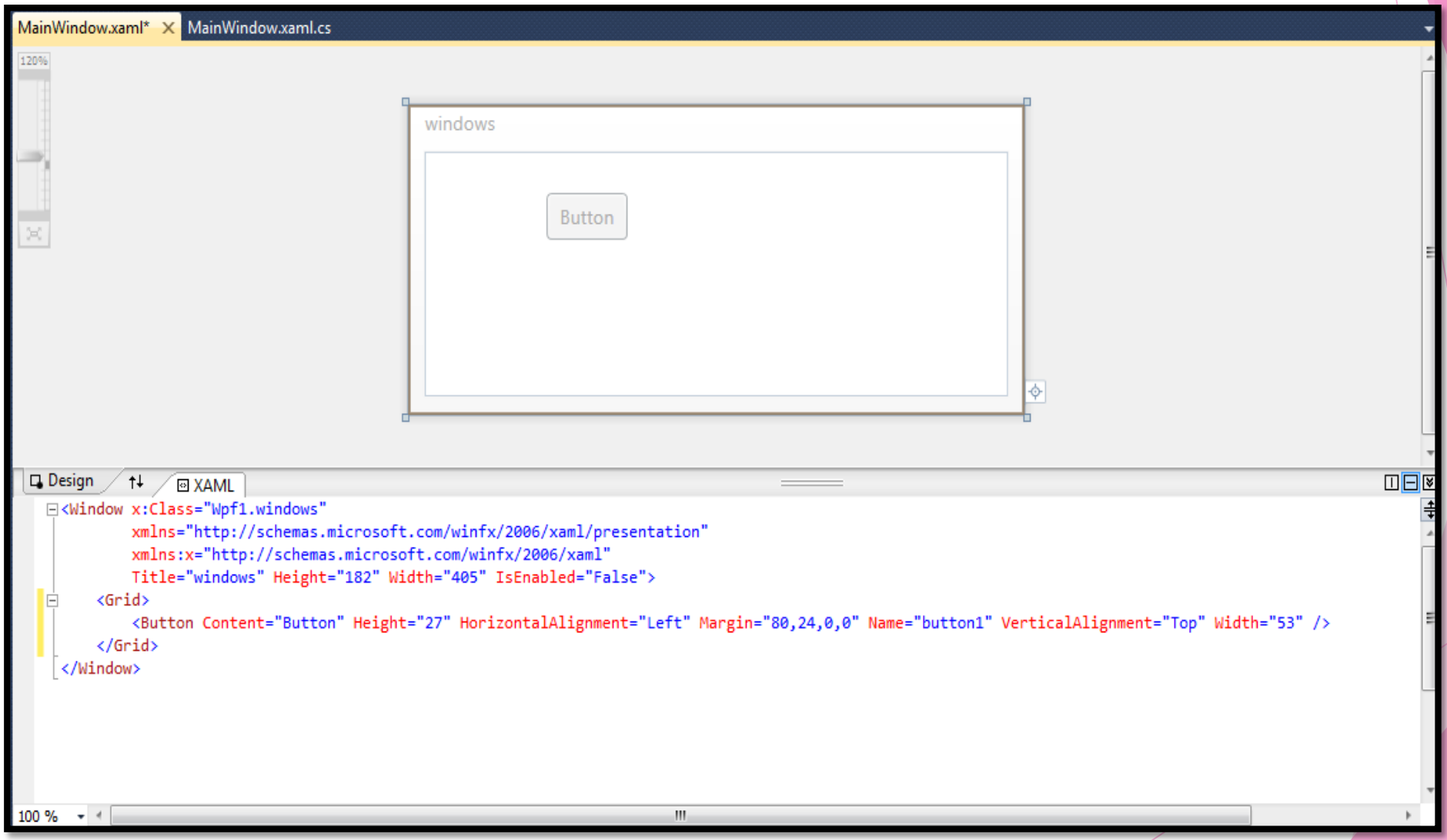
Selection : CheckBox , ComboBox , ListBox , TreeView , RadioButton , Slider

User Information : AccessText , Label , Popup , ProgressBar , StatusBar , TextBlock , ToolTip

XAML
(Extensible Application Markup Language)

- هي لغة xml تستخدم لتطوير واجهات التطبيقات.

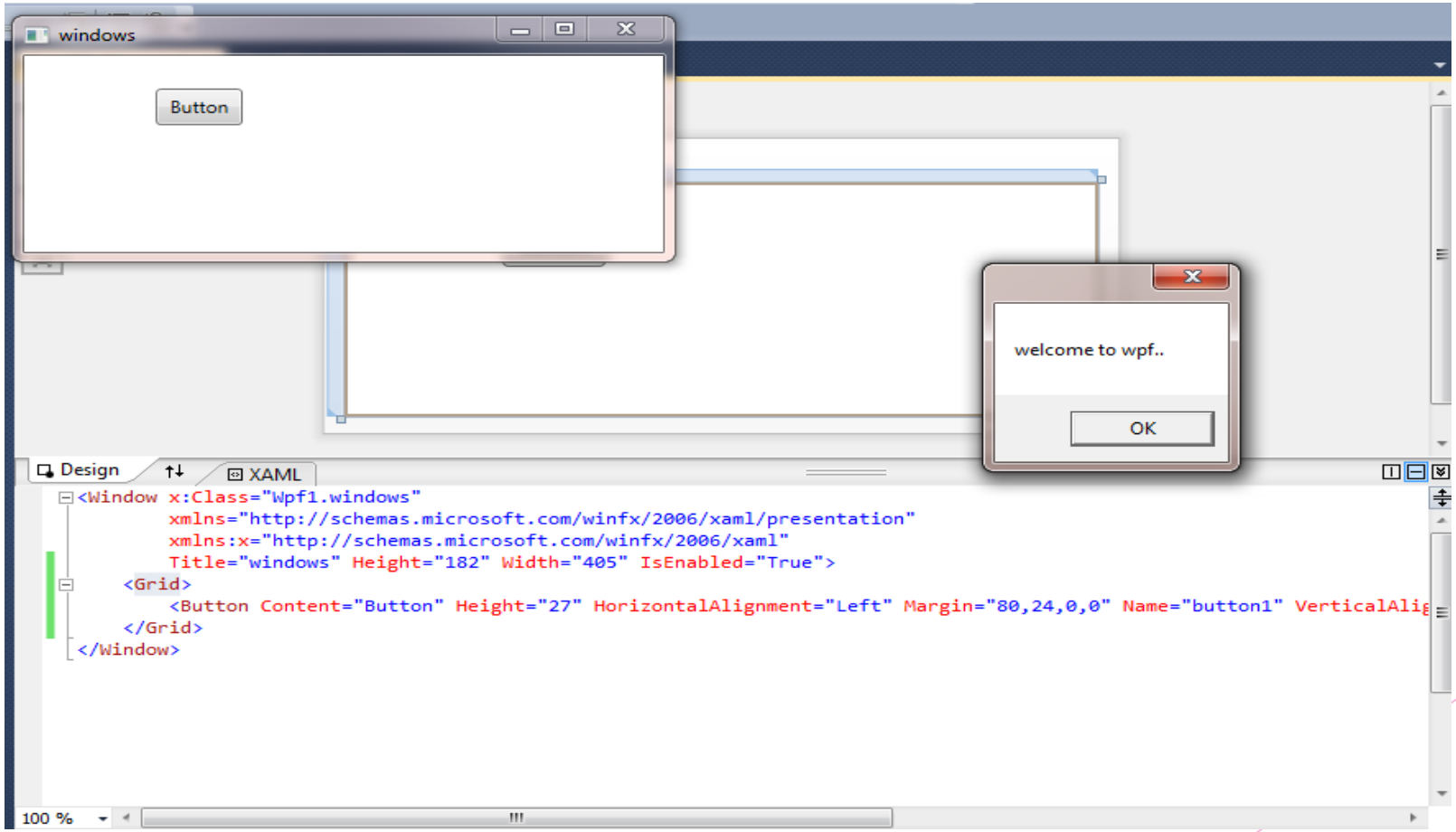
التمرين الأول



- Xml namespace
- Attribute (title, width, Height)
- Grid

التمرين الثاني

- بناء واجهة تحوي زر عند الضغط عليه تظهر رسالة "welcome to wpf.."
- نستخدم MessageBox.show()



Layout && Border

إعداد

م. لمى السبع

الجلسة الثانية

Xaml

- XAML اختصار لـ Extensible Application Markup Language وتلفظ "زامل"، وهي لغة وسوم لتمثيل الكائنات في الدوت نت، تم إنشائها خصيصاً من اجل WPF وتستخدم في تقنيات اخرى مثل WF (Workflow Foundation)و Silverlight .
- تستخدم XAML في WPF اساساً من اجل تعريف واجهات المستخدم، والغاية من هذا هو فصل الأجزاء المتعلقة بتعريف الواجهة عن الكود في التطبيق.
- تعتمد لغة XAML اساسا على لغة XML، لذا فإن اي مستند XAML في الواقع هو ملف XML ، ويخضع لنفس القواعد والشروط، مثلها مثل اي لغة وسوم اخرى، إلا ان XAML اكثر صرامة من غيرها حيث ان كل عنصر فيها يتم تحويله إلى كائن في الدوت نت.
- وكونها تعتمد على XML، فإن هذا اضاف سهولة ومرونة كبيرة، فيمكن لأكثر من اداة ان تتعامل مع نفس المستند في نفس الوقت، ومن السهل جدا قراءة مستندات XMLونقلها او حملها من جهاز إلى آخر.

- حتى نفهم XAML ، علينا ان نتذكر القواعد التالية:

١. كل عنصر في مستند XAML يتم ترجمته إلى مثيل من فئة في الدوت نت. ودائما يكون اسم العنصر مطابقا لاسم فئة معينة، مثلا العنصر <Button> يمثل امر XAML من اجل إنشاء كائن من الفئة Button.

٢. كما هو الحال في اي مستند XML، يمكن للعناصر ان تتداخل، وتقوم XAML بتفسير هذا التداخل على انه احتواء، فإذا وجدت عنصر Button داخل عنصر Grid، فإن هذا يعني ان الواجهة ستتضمن Grid يحتوي على زر.

٣. يمكن تعيين الخصائص من خلال الواصفات (Attributes) ، لكن في بعض الحالات تكون الواصفة غير كافية لتحديد قيمة الخاصية، لذا فإننا نستخدم صيغة اخرى تسمى بالعنصر الخاصية (Property Element).

مثال

XAML Code:

```
1 <Window x:Class="WpfApplication1.Window1"
2 xmlns="http://schemas.microsoft.com/winfx/2...l/presentation"
3 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4 Title="Window1" Height="300" Width="300">
5   <Grid>

6   </Grid>
7 </Window>
```

Layout

هي المكان (الحاوية) تحوي مجموعة من العناصر (أزرار ونصوص...).

يفضل أن نقوم بتصميم الواجهة بحيث تتناسب مع حجم أي شاشة سيتم العرض عليها.

أنواع Layout

الاسم	التوصيف
StackPanel	يتم تكديس العناصر فوق بعض بشكل أفقي أو عمودي.
WrapPanel	ترتيب الأدوات من اليسار إلى اليمين مع إمكانية التفاف الأدوات إلى سطر جديد عند عدم وجود مساحة كافية في السطر الحالي.
DockPanel	ترتيب الأدوات على حواف النافذة.
Grid	يتم تحديد مواضع العناصر حسب الصف والعمود كما يمكن تحديد موضع أي عنصر ضمن أي خلية.
UniformGrid	يتم ترتيب العناصر بصفوف وأعمدة ويتم ترتيب العناصر حسب ترتيب إضافتها، كل الخلايا بنفس الحجم.
Canvas	يتم ترتيب العناصر بإحداثيات ثابتة.

خصائص Layout

الاسم	التوصيف
HorizontalAlignment	يتم تحديد موضع العناصر (center,left,right,stretch)
VerticalAlignment	يتم تحديد موضع العناصر (center,top,bottom,stretch)
Margin	يحدد مسافة حول العنصر من جميع حوافه (top,bottom,left,right)
MinWidth and MinHeight	يحدد أصغر قيمة لأبعاد العنصر،ففي حال كان حجم العنصر كبير بالنسبة للذي يحتويه يتم تصغيره.
MaxWidth and MaxHeight	يحدد أكبر قيمة لأبعاد العنصر
Width and Height	يحدد قيمة الطول والعرض للعنصر.

Margin

- التصميم الجيد لواجهة ما لا يحدد أبعاد العناصر فقط وإنما نحدد المسافات حول العناصر أيضاً والتي تحدد المسافات بين العناصر.
- لتحديد Margin يوجد عدة طرق :

١. نحدد قيمة ثابتة للمسافات حول كل الحواف

```
<Button Margin="5">Button 3</Button>
```

٢. تحديد قيم مختلفة للمسافة حول الحواف (left, Top, right, bottom)

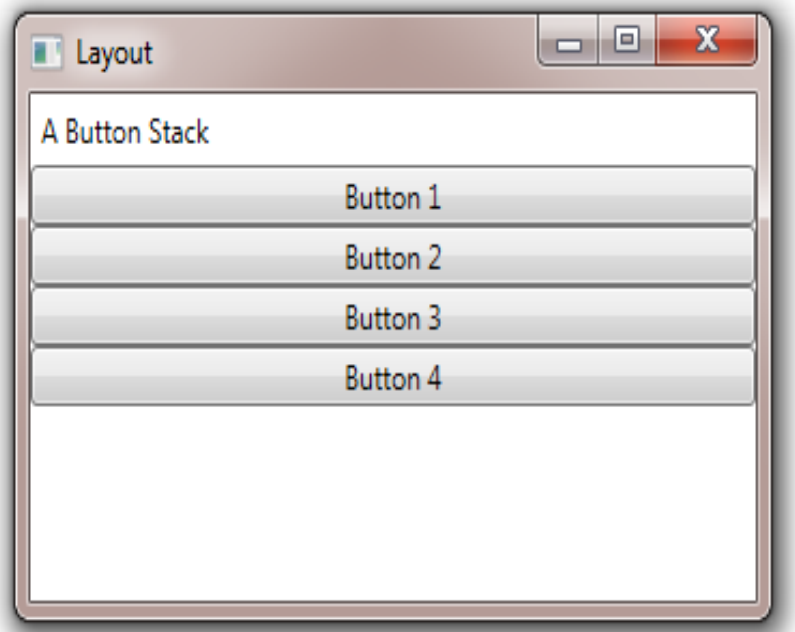
```
<Button Margin="5,10,5,10">Button 3</Button>
```

- ملاحظة:

يمكن أن نضيف خاصية Margin لل Layout.

التمرين الأول
StackPanel

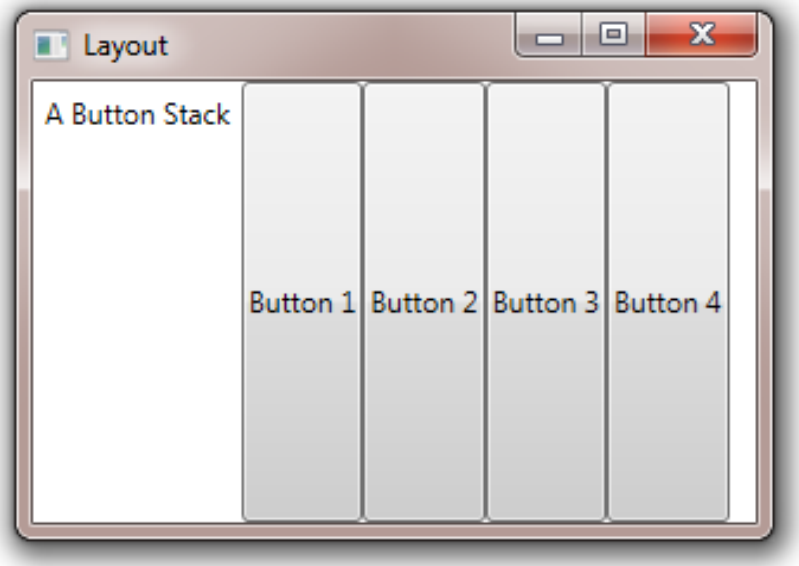
```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <StackPanel>
        <Label>A Button Stack</Label>
        <Button>Button 1</Button>
        <Button>Button 2</Button>
        <Button>Button 3</Button>
        <Button>Button 4</Button>
    </StackPanel>
</Window>
```



تتمة التمرين الأول

عدّل التمرين السابق بحيث تصبح محاذاة المحتوى أفقية

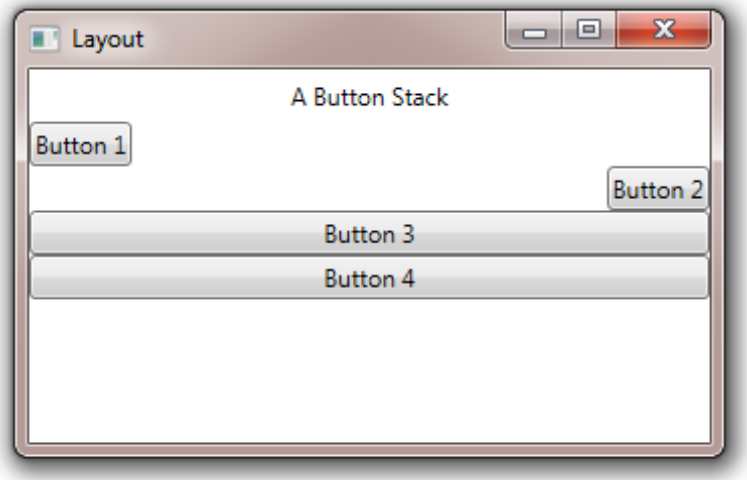
```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <StackPanel Orientation="Horizontal">
        <Label>A Button Stack</Label>
        <Button>Button 1</Button>
        <Button>Button 2</Button>
        <Button>Button 3</Button>
        <Button>Button 4</Button>
    </StackPanel>
</Window>
```



تتمة التمرين الأول

القيمة الافتراضية HorizontalAlignment=“Stretch”

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <StackPanel>
        <Label HorizontalAlignment="Center">A Button Stack</Label>
        <Button HorizontalAlignment="Left">Button 1</Button>
        <Button HorizontalAlignment="Right">Button 2</Button>
        <Button HorizontalAlignment="Stretch">Button 3</Button>
        <Button>Button 4</Button>
    </StackPanel>
</Window>
```

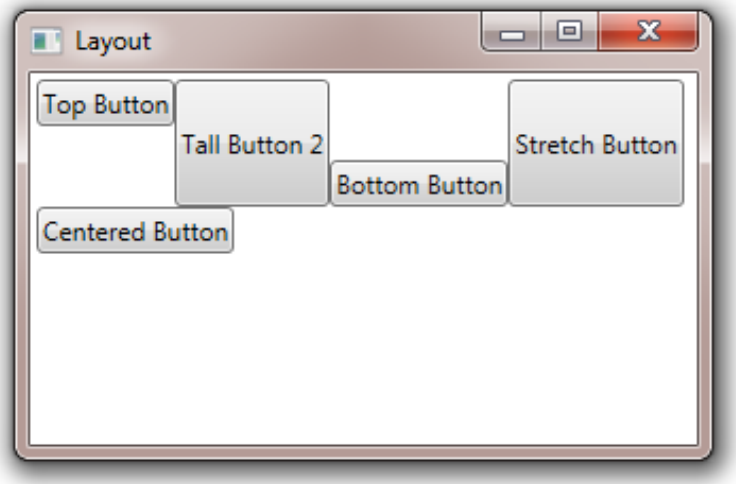


التمرين الثاني

wrapPanel

الحالة الافتراضية ترتب العناصر بشكل أفقي

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <WrapPanel Margin="3">
        <Button VerticalAlignment="Top">Top Button</Button>
        <Button MinHeight="60">Tall Button 2</Button>
        <Button VerticalAlignment="Bottom">Bottom Button</Button>
        <Button>Stretch Button</Button>
        <Button VerticalAlignment="Center">Centered Button</Button>
    </WrapPanel>
</Window>
```



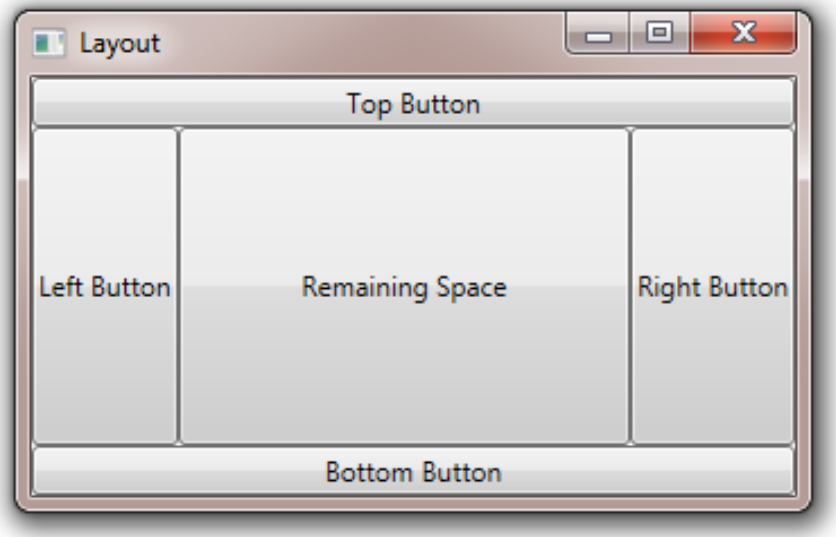
ملاحظات

١. لاحظ الخاصية MinHeight :حاول تعديل حجم النافذة يبقى طول الزر ذاته.

٢. لاحظ verticalAlignment

التمرين الثالث
DockPanel

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <DockPanel LastChildFill="True">
        <Button DockPanel.Dock="Top">Top Button</Button>
        <Button DockPanel.Dock="Bottom">Bottom Button</Button>
        <Button DockPanel.Dock="Left">Left Button</Button>
        <Button DockPanel.Dock="Right">Right Button</Button>
        <Button>Remaining Space</Button>
    </DockPanel>
</Window>
```

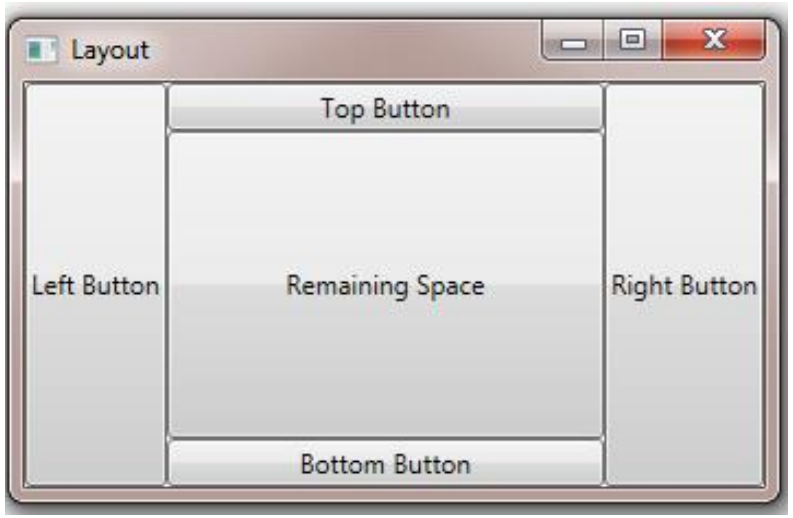


ملاحظات

١) الخاصية Dock تحدد العنصر بأي حافة موجود.

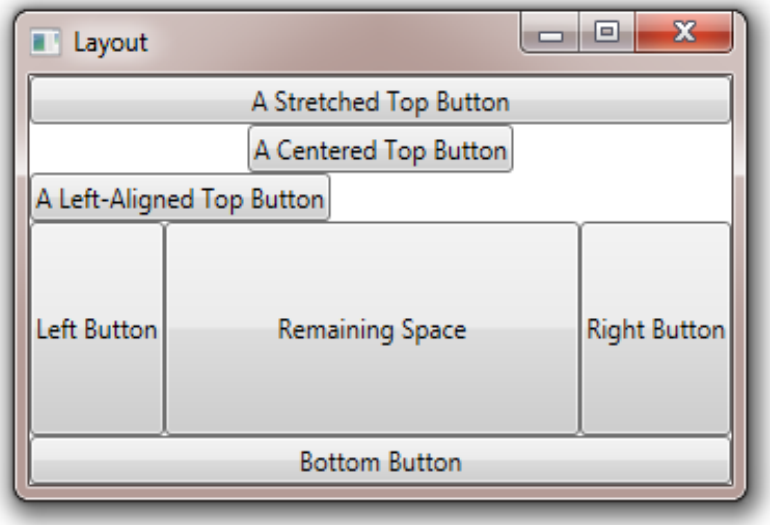
٢) الخاصية LastChildFill=true تعني المساحة المتبقية من النافذة تخص العنصر الأخير.

٣) ترتيب العناصر مهم: حاول أن تجعل زر اليمين واليسار معرفين قبل زري الأعلى والأسفل عندها نلاحظ أنّ اليمين واليسار امتدا على كل الحافة.



- عدّل على المثال السابق بحيث تضع في الحافة العليا ٣ أزرار (الأول ممتد، الثاني بالوسط، الثالث عاليسار) وذلك باستخدام HorizontalAlignment
- بالنظر للشكل المجاور نلاحظ أنّه يرتب عناصر الحافة العليا كما StackPanel

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354">
    <DockPanel LastChildFill="True">
        <Button DockPanel.Dock="Top">A Stretched Top Button</Button>
        <Button DockPanel.Dock="Top" HorizontalAlignment="Center">
            A Centered Top Button</Button>
        <Button DockPanel.Dock="Top" HorizontalAlignment="Left">
            A Left-Aligned Top Button</Button>
        <Button DockPanel.Dock="Bottom">Bottom Button</Button>
        <Button DockPanel.Dock="Left">Left Button</Button>
        <Button DockPanel.Dock="Right">Right Button</Button>
        <Button>Remaining Space</Button>
    </DockPanel>
</Window>
```



التمرين الرابع
Grid

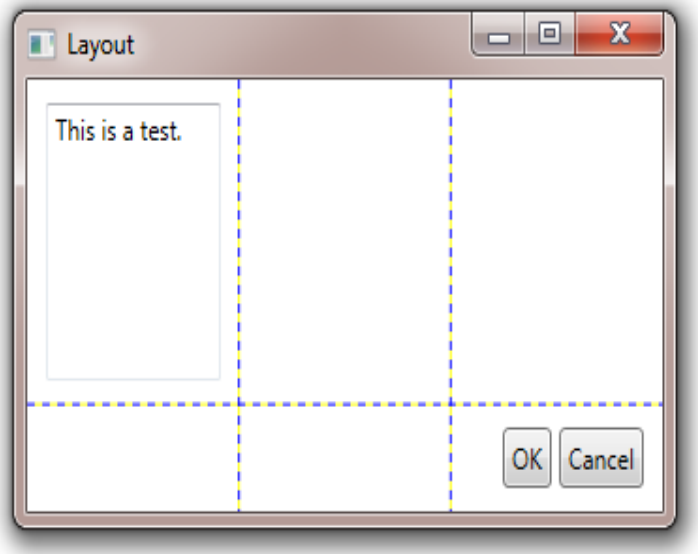
- لإنشاء Grid نتبع الخطوات التالية :

١. نحدد عدد الأسطر والأعمدة.

٢. نسند العناصر للخلايا.

- الخاصية ShowGridLines=“true” تظهر الحواف التي تفصل بين الخلايا.

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354"
    >
   <Grid ShowGridLines="True">
  <Grid.RowDefinitions>
    <RowDefinition Height="*"></RowDefinition>
    <RowDefinition Height="Auto"></RowDefinition>
  </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <TextBox Margin="10" Grid.Row="0">This is a test.</TextBox>
  <StackPanel Grid.Row="1" Grid.Column="2" HorizontalAlignment="right" Orientation="Horizontal">
            <Button Margin="10,10,2,10" Padding="3">OK</Button>
            <Button Margin="2,10,10,10" Padding="3">Cancel</Button>
  </StackPanel>
</Grid>
</Window>
```



ملاحظات

- حجم الخلايا في Grid متساوية افتراضياً ولتغيير الحجم هناك عدة طرق:

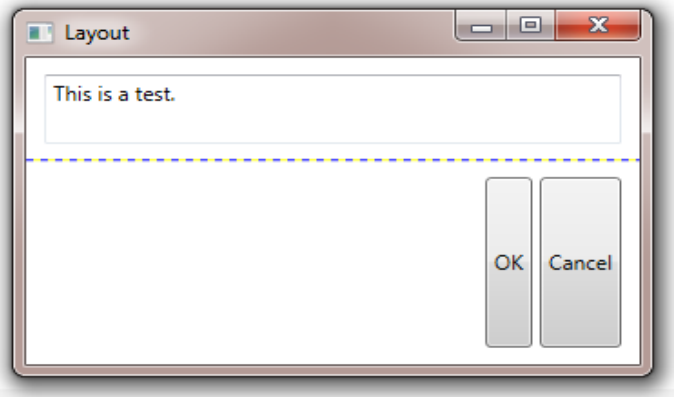
١. الحجم الثابت

٢. الحجم الأوتوماتيكي Auto :كل سطر أو عمود يأخذ الحجم الذي يحتاجه.

٣. الحجم Proportion :الحجم يقسم بين الخلايا ويزداد حجم الخلايا كلما زاد حجم النافذة.

مثال

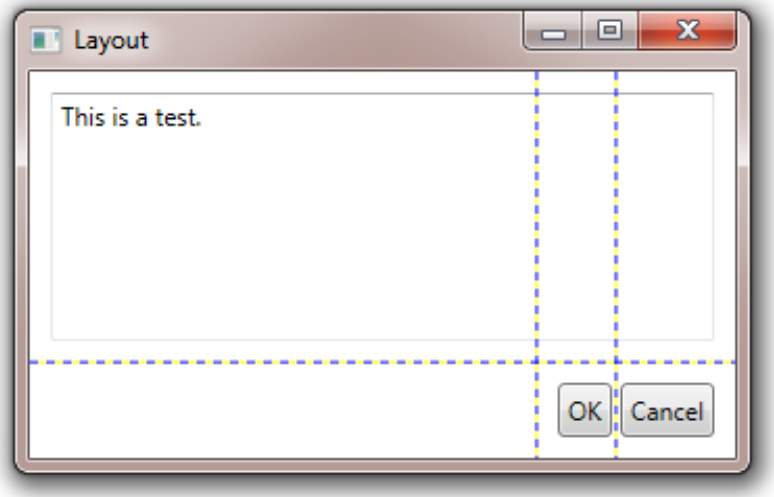
```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354"
    >
   <Grid ShowGridLines="True">
  <Grid.RowDefinitions>
    <RowDefinition Height="*"></RowDefinition>
    <RowDefinition Height="2*"></RowDefinition>
  </Grid.RowDefinitions>

        <TextBox Margin="10" Grid.Row="0">This is a test.</TextBox>
  <StackPanel Grid.Row="1"  HorizontalAlignment="right" Orientation="Horizontal">
            <Button Margin="10,10,2,10" Padding="3">OK</Button>
            <Button Margin="2,10,10,10" Padding="3">Cancel</Button>
  </StackPanel>
</Grid>
</Window>
```



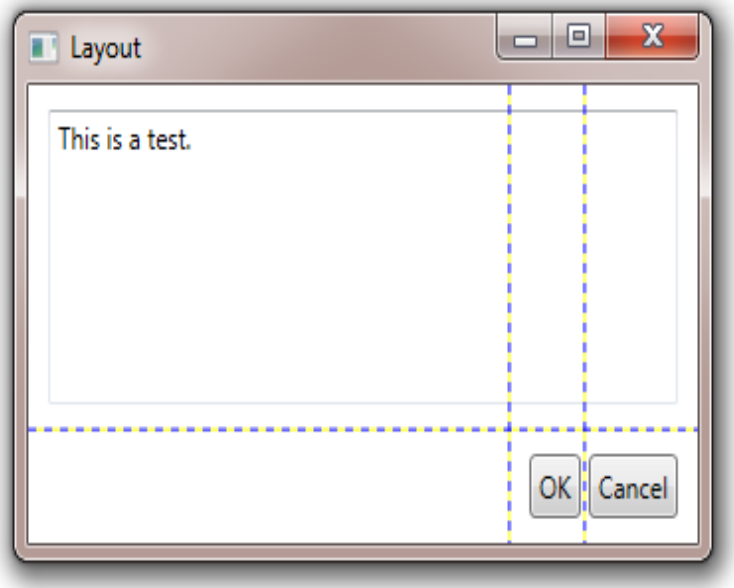
RowSpan && ColumnSpan

- مثال

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354"
    >
    <Grid ShowGridLines="True">
        <Grid.RowDefinitions>
            <RowDefinition Height="*"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="*"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <TextBox Margin="10" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="3">
            This is a test.</TextBox>
        <Button Margin="10,10,2,10" Padding="3"
    Grid.Row="1" Grid.Column="1">OK</Button>
        <Button Margin="2,10,10,10" Padding="3"
    Grid.Row="1" Grid.Column="2">Cancel</Button>
    </Grid>
</Window>
```

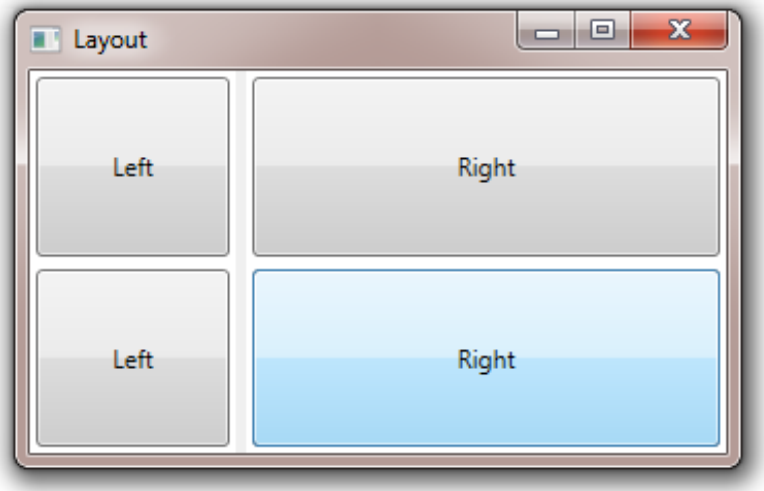


```xaml
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354"
    >
    <Grid ShowGridLines="True">
        <Grid.RowDefinitions>
            <RowDefinition Height="*"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="*"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <TextBox Margin="10" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="3">
            This is a test.</TextBox>
        <Button Margin="10,10,2,10" Padding="3"
    Grid.Row="1" Grid.Column="1">OK</Button>
        <Button Margin="2,10,10,10" Padding="3"
    Grid.Row="1" Grid.Column="2">Cancel</Button>
    </Grid>
</Window>
```

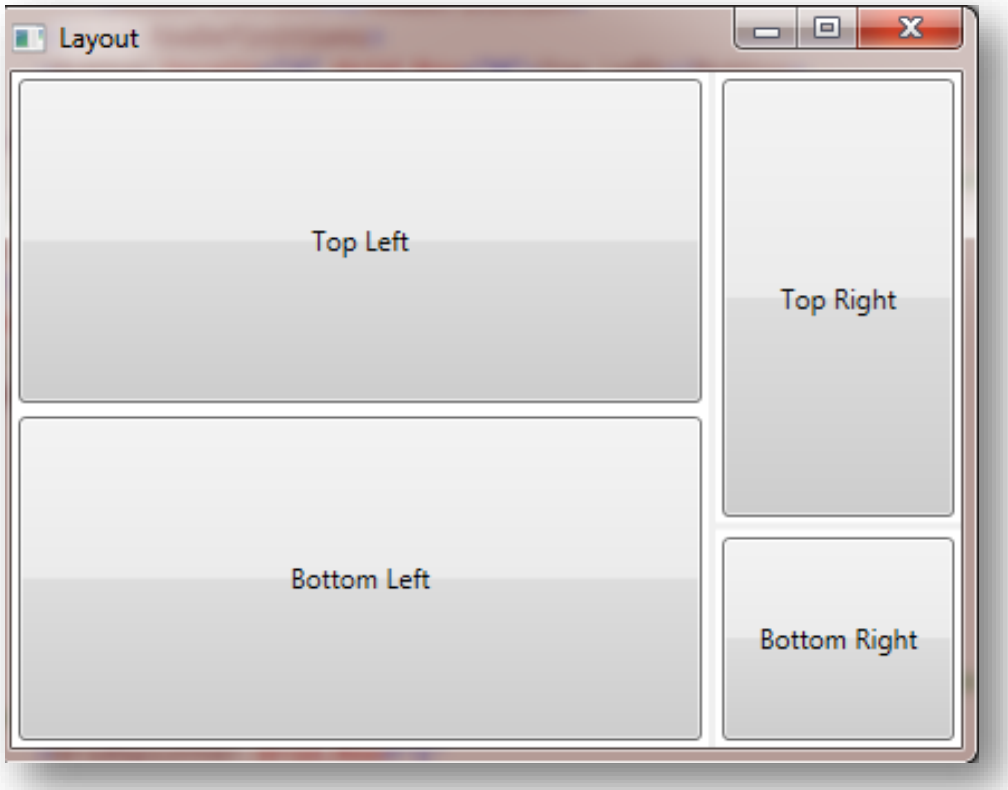


الحد الفاصل

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354"
    >
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition ></RowDefinition>
            <RowDefinition ></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition MinWidth="100"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
            <ColumnDefinition MinWidth="50"></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Button Grid.Row="0" Grid.Column="0" Margin="3">Left</Button>
        <Button Grid.Row="0" Grid.Column="2" Margin="3">Right</Button>
        <Button Grid.Row="1" Grid.Column="0" Margin="3">Left</Button>
        <Button Grid.Row="1" Grid.Column="2" Margin="3">Right</Button>

        <GridSplitter Grid.Row="0" Grid.Column="1" Grid.RowSpan="2"
   Width="5" VerticalAlignment="Stretch" HorizontalAlignment="center"
   ShowsPreview="False"></GridSplitter>

    </Grid>
</Window>
```



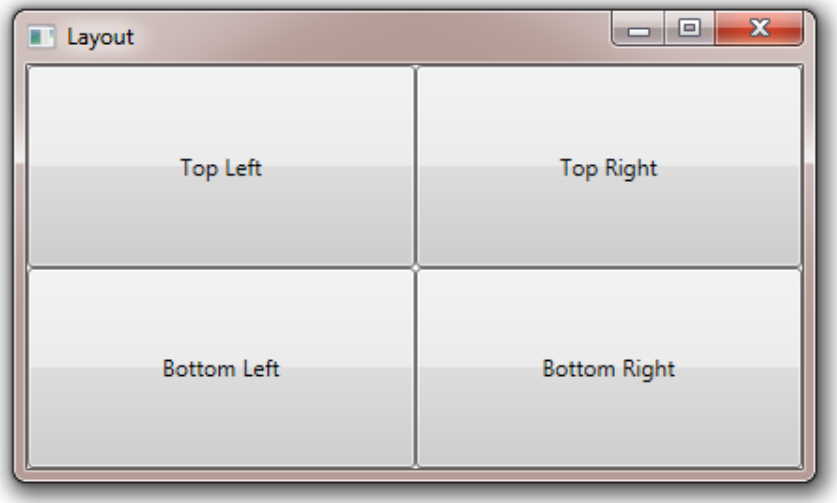
وظيفة



التمرين الخامس

- UniformGrid

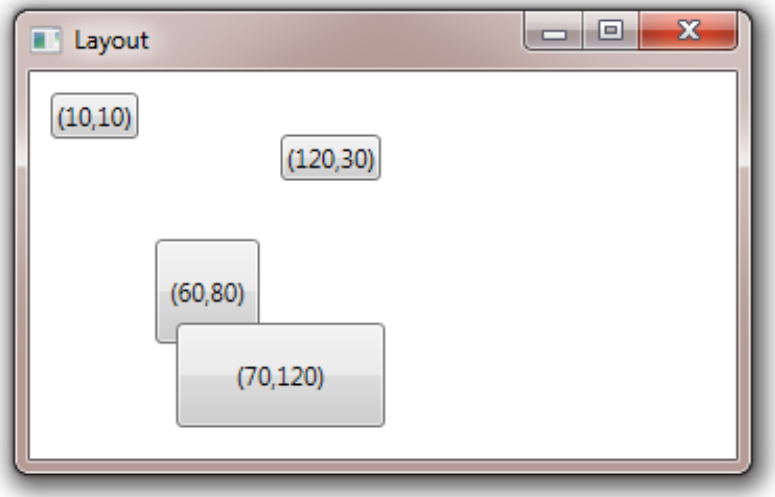
```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354" >
    <UniformGrid Rows="2" Columns="2">
        <Button>Top Left</Button>
        <Button>Top Right</Button>
        <Button>Bottom Left</Button>
        <Button>Bottom Right</Button>
    </UniformGrid>
</Window>
```



التمرين السادس

- Canvas

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354" >
    <Canvas>
        <Button Canvas.Left="10" Canvas.Top="10">(10,10)</Button>
        <Button Canvas.Left="120" Canvas.Top="30">(120,30)</Button>
        <Button Canvas.Left="60" Canvas.Top="80" Width="50" Height="50">
            (60,80)</Button>
        <Button Canvas.Left="70" Canvas.Top="120" Width="100" Height="50">
            (70,120)</Button>
    </Canvas>
</Window>
```



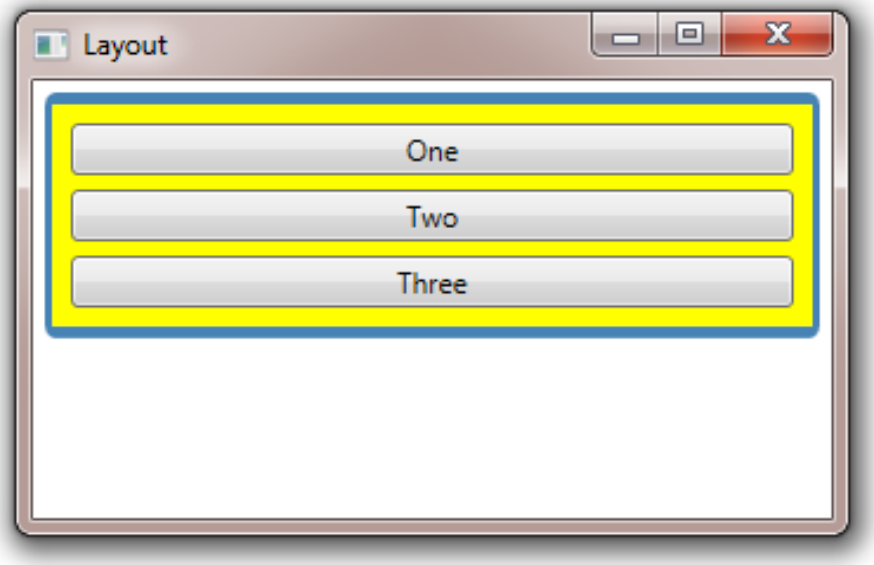
الحدود

خصائص الحدود

الاسم	التوصيف
Background	تحدد لون الخلفية للعناصر التي داخل الحدود
BorderBrush and BorderThickness	تحدد لون الحدود وسماكتها
CornerRadius	تحدد درجة انحناء حواف الحدود
Padding	تحدد المسافة التي تفصل محتوى الحدود عن الحواف

تمرین

```
<Window x:Class="Wpf1.windows"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Layout" Height="223" Width="354" >
    <Border Margin="5" Padding="5" Background="Yellow"
 BorderBrush="SteelBlue" BorderThickness="3,5,3,5" CornerRadius="3"
 VerticalAlignment="Top">
        <StackPanel>
            <Button Margin="3">One</Button>
            <Button Margin="3">Two</Button>
            <Button Margin="3">Three</Button>
        </StackPanel>
    </Border>
</Window>
```



Technical Institute Computer IN HAMA

جامعة حماه

المعهد التقاني للحاسوب TIC

السنة الثانية

MultiMedia

Lab3

Eng : M.Mahdi Alkhaled

WPF controls

Controls : هي عناصر تفاعل مع المستخدم .

الخصائص المشتركة :

- Margin : البعد عن حواف التخطيط الموجود ضمنه .
- Name : الاسم البرمجي للمتحكم .
- Content : المحتوى للمتحكم. يمكن استخدامه :

<Button Content=" "> </Button>

<Button>المحتوى هنا</Button>

<Button> <Button.content> الحتوى هنا </Button.content> </Button >

مع ملاحظة أنه يوجد ابن وحيد للمحتوى فقط .

- Padding : تباعد محتوى المتحكم عن الحواف .
- horizontalContentAlignment : انزياح أفقي لمحتوى المتحكم.
- horizontalContentAlignment : انزياح عمودي لمحتوى المتحكم.
- Click : يحدد التابع البرمجي الذي سيعالج ضغط المتحكم.
- MinWidth : أصغر عرض للتوسيع عند تصغير النافذة .
- MaxWidth : أكبر عرض للتوسيع عند تكبير النافذة .
- MinHeigh : أصغر ارتفاع للتوسيع عند تصغير النافذة .
- MaxHeigh : أكبر ارتفاع للتوسيع عند تكبير النافذة

Button :

<Button> </Button>

له الخصائص التالية أيضا :

- IsCancel : أي أنه زر لإغلاق النافذة عندما نضع قيمة True .
- IsDefault : أي أنه الزر الافتراضي للنافذة عندما نضع قيمة True .

مع ملاحظة أنه يجب وضع زر إغلاق وحيد للنافذة و زر افتراضي وحيد لها.

CheckBox :

لتحديد الاختيار أو عدم الاختيار .

<CheckBox> المحتوى (اسم الزر) </CheckBox>

له نفس الخصائص المشتركة إضافة إلى :

- IsChecked : لتحديد الحالة الابتدائية للزر.

RadioButton :

يحدد اختيار من عدة خيارات موجودة بحيث يجب ربط الأزرار المترابطة باسم مجموعة واحد .

لتحديد مجموعة محددة للأزرار عبر الوسم :

<GroupBox name =" اسم المجموعة" Header="عنوان المجموعة "></GroupBox>

أو عبر تحديد اسم المجموعة ضمن الزر :

< RadioButton GroupName="….." > </RadioButton>

ToolTips :

هي خاصية لكل الوسوم في لغة WPF حيث أنها هذه الخاصية تقوم بعرض تعليق عن الوسم التي تم إضافته لها عند مرور مؤشر الفأرة عليه . مثل :

<CheckBox>

<CheckBox.ToolTip> التعليق </CheckBox.ToolTip>

</CheckBox>

ScrollViewer :

يستخدم هذا الوسم لإضافة شريط تمرير للنافذة أو مساحة محددة منها و ذلك عندما يكون المحتوى أكبر من أن نكون قادرين على عرضه في النافذة المحددة.

<ScrollViewer> </ScrollViewer>

Tab :

تستخدم عند وجود الكثير من المعلومات و الأدوات التي تحويها النافذة

```
<TabControl >
        <TabItem Header="Tab One">

        </TabItem>
        <TabItem Header="Tab Two">
           ...
        </TabItem>
 </TabControl>
```

Expander :

يستخدم هذا المتحكم لإضافة محتوى مخفي يتم عرضه عبر الضغط على السهم :

```
<Expander Header="اسمه"> المحتوى </Expander>
```

خصائصه :

- ExpanderDirection تحدد جهة فتح التوسيع للعنصر .
- SizeToContent :لتكون الشاشة ملائمة عند التوسيع للعنصر .

TextBox :

عنصر لإدخال نصي

```
<TextBox >   </TextBox>
```

له الخصائص :

- AcceptsTab : تحدد عند التفعيل أن زر ال Tab من لوحة المفاتيح سوف ينقل المؤشر مسافة و ليس الانتقال إلى العنصر التالي بالنافذة.
- AcceptsReturn :عند تفعيله تحدد أنه زر Enter سوف ينزل المؤشر سطر و ليس الضغط على الزر الافتراضي في النافذة .
- MaxLines :تحدد عدد السطور الأعظمي.
- TextWrapping : تحدد التفاف النص عند الوصول لنهاية السطر .
- MaxLength : تحدد عدد المحارف الأعظمي.
- IsReadOnly : لجعل النص الموجود ضمنه غير قابل للتعديل .
- IsEnable : لجعل النص جامد لايمكن نسخه .

PasswordBox :

يستخدم عند الحاجة لإدخال كلمة مرور وهو لا يدعم نسخ المحتوى.

```
<PasswordBox > </PasswordBox>
```

خصائصه :

- PasswordChar : تحدد نوع المحرف الذي سيظهر بدلا من إدخال المستخدم.

: ListBox

يستخدم لإضافة مجموعة من الخيارات التي يمكن اختيار أحد منها فقط.

```
<ListBox>
        <ListBoxItem>……</ListBoxItem>
         <ListBoxItem>……</ListBoxItem>
 </ListBox>
```

: ComboBox

لاختيار عنصر وحيد من قائمة منسدلة .

```
<ComboBox>
            <ComboBoxItem >1</ComboBoxItem>
            <ComboBoxItem >2</ComboBoxItem>
</ComboBox>
```

:Slider

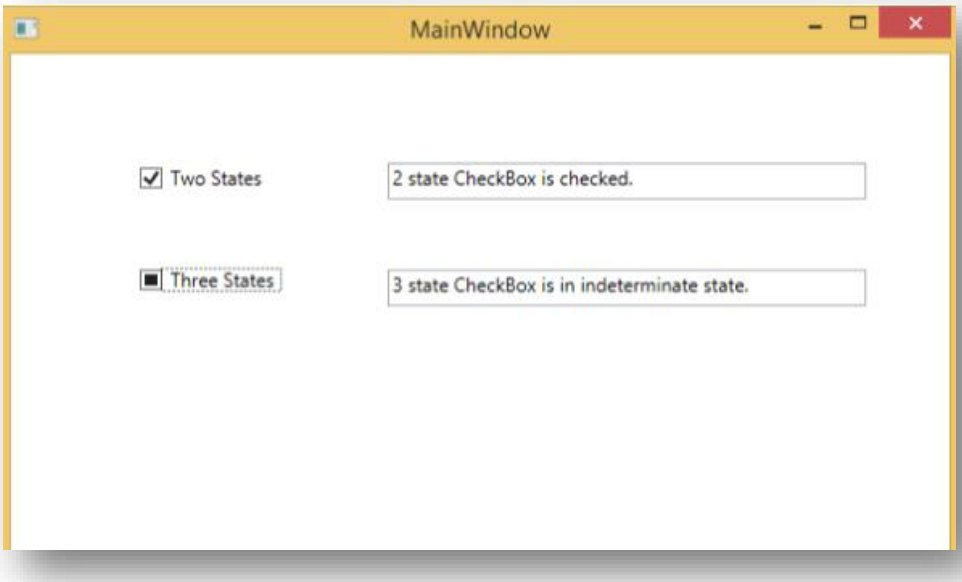
```
<Slider>       </Slider>
```

الخصائص :

- TickPlacement : تحديد تموضع المؤشر .
- Maximum : قيمة السلايدر .
- TickFrequency : تحديد المسافة لوضع النقاط .

الأحداث في WPF

- **Checked & UnChecked** يستخدم هذان الحدثان مع Checkbox & radio button & Toggle Button

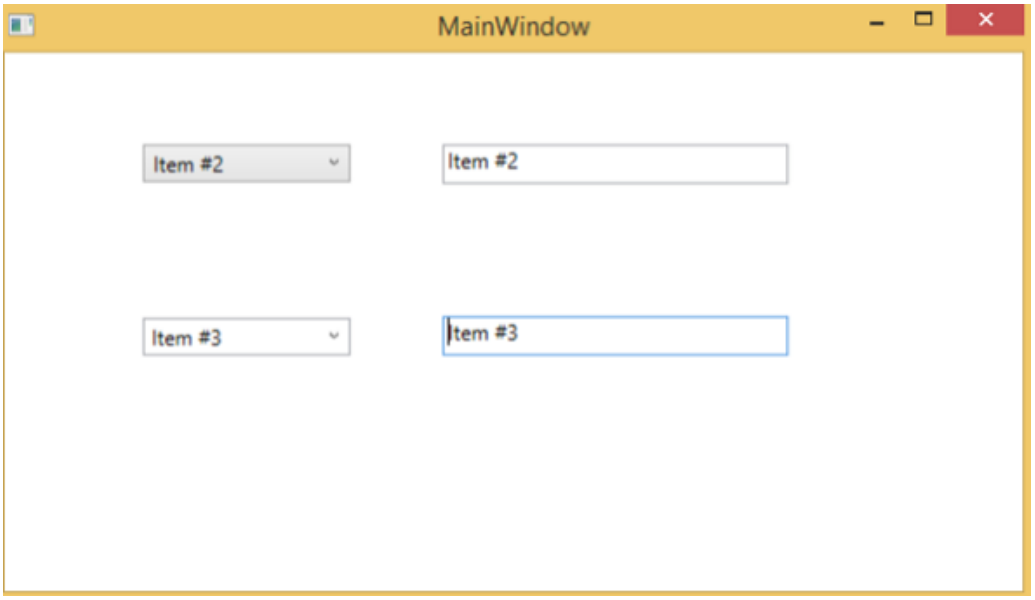



```
<Window x:Class = "WPFCheckBoxControl.MainWindow"
   xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d = "http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc = "http://schemas.openxmlformats.org/markup-compatibility/2006"
   xmlns:local = "clr-namespace:WPFCheckBoxControl"
   mc:Ignorable = "d" Title = "MainWindow" Height = "350" Width = "604">

   <Grid>
      <CheckBox x:Name = "checkBox1" Content = "Two States" HorizontalAlignment = "Le
         Margin = "80,70,0,0" VerticalAlignment = "Top" Checked = "HandleCheck"
         Unchecked = "HandleUnchecked" Width = "90"/>

      <CheckBox x:Name = "checkBox2" Content = "Three States"
         HorizontalAlignment = "Left" Margin = "80,134,0,0" VerticalAlignment = "Top"
         Width = "90" IsThreeState = "True" Indeterminate = "HandleThirdState"
         Checked = "HandleCheck" Unchecked = "HandleUnchecked"/>

      <TextBox x:Name = "textBox1" HorizontalAlignment = "Left"
         Height = "23" Margin = "236,68,0,0" TextWrapping = "Wrap"
         VerticalAlignment = "Top" Width = "300"/>

      <TextBox x:Name = "textBox2" HorizontalAlignment = "Left"
         Height = "23" Margin = "236,135,0,0" TextWrapping = "Wrap"
         VerticalAlignment = "Top" Width = "300"/>
   </Grid>

</Window>
```

وكود C# المقابل

```
using System.Windows;
using System.Windows.Controls;

namespace WPFCheckBoxControl {
   /// <summary>
      /// Interaction logic for MainWindow.xaml
   /// </summary>

   public partial class MainWindow : Window {

      public MainWindow() {
         InitializeComponent();
      }

      private void HandleCheck(object sender, RoutedEventArgs e) {
         CheckBox cb = sender as CheckBox;

         if (cb.Name == "checkBox1") textBox1.Text = "2 state CheckBox is checked.";
         else textBox2.Text = "3 state CheckBox is checked.";
      }

      private void HandleUnchecked(object sender, RoutedEventArgs e) {
         CheckBox cb = sender as CheckBox;

         if (cb.Name == "checkBox1") textBox1.Text = "2 state CheckBox is unchecked."
         else textBox2.Text = "3 state CheckBox is unchecked.";
      }

      private void HandleThirdState(object sender, RoutedEventArgs e) {
         CheckBox cb = sender as CheckBox;
         textBox2.Text = "3 state CheckBox is in indeterminate state.";
      }

   }
}
```

- ***SelectionChanged***



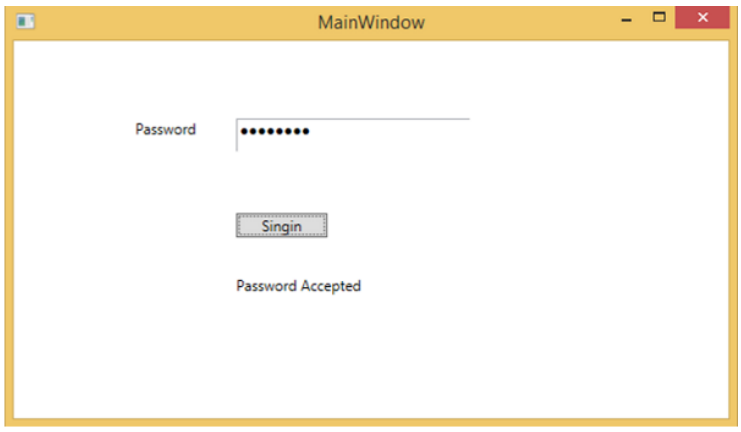
الـواجهة

```
<Window x:Class = "WPFComboBoxControl.MainWindow"
   xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d = "http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc = "http://schemas.openxmlformats.org/markup-compatibility/2006"
   xmlns:local = "clr-namespace:WPFComboBoxControl"
   mc:Ignorable = "d" Title = "MainWindow" Height = "350" Width = "604">

   <Grid>

      <ComboBox x:Name = "comboBox" HorizontalAlignment = "Left"
         Margin = "80,53,0,0" VerticalAlignment = "Top" Width = "120"
         SelectionChanged = "Combo_SelectionChanged">

         <ComboBoxItem Content = "Item #1" />
         <ComboBoxItem Content = "Item #2" />
         <ComboBoxItem Content = "Item #3" />
      </ComboBox>

      <ComboBox x:Name = "comboBox1" HorizontalAlignment = "Left"
         Margin = "80,153,0,0" VerticalAlignment = "Top" Width = "120"
         IsEditable = "True"
         SelectionChanged = "Combo1_SelectionChanged">

         <ComboBoxItem Content = "Item #1" />
         <ComboBoxItem Content = "Item #2" />
         <ComboBoxItem Content = "Item #3" />
      </ComboBox>

      <TextBox x:Name = "textBox" HorizontalAlignment = "Left"
         Height = "23" Margin = "253,53,0,0" TextWrapping = "Wrap"
         VerticalAlignment = "Top" Width = "200" />

      <TextBox x:Name = "textBox1" HorizontalAlignment = "Left"
         Height = "23" Margin = "253,152,0,0" TextWrapping = "Wrap"
         VerticalAlignment = "Top" Width = "200" />

   </Grid>

</Window>
```

كـود C# الـمقـابـل

```
using System.Windows;
using System.Windows.Controls;

namespace WPFComboBoxControl {
   /// <summary>
      /// Interaction logic for MainWindow.xaml
   /// </summary>

   public partial class MainWindow : Window {

      public MainWindow() {
         InitializeComponent();
      }

      private void Combo_SelectionChanged(object sender, SelectionChangedEventArgs e)
         textBox.Text = comboBox.SelectedItem.ToString();
      }

      private void Combo1_SelectionChanged(object sender, SelectionChangedEventArgs e
         textBox1.Text = comboBox1.SelectedItem.ToString();
      }

   }
}
```

- **Click**



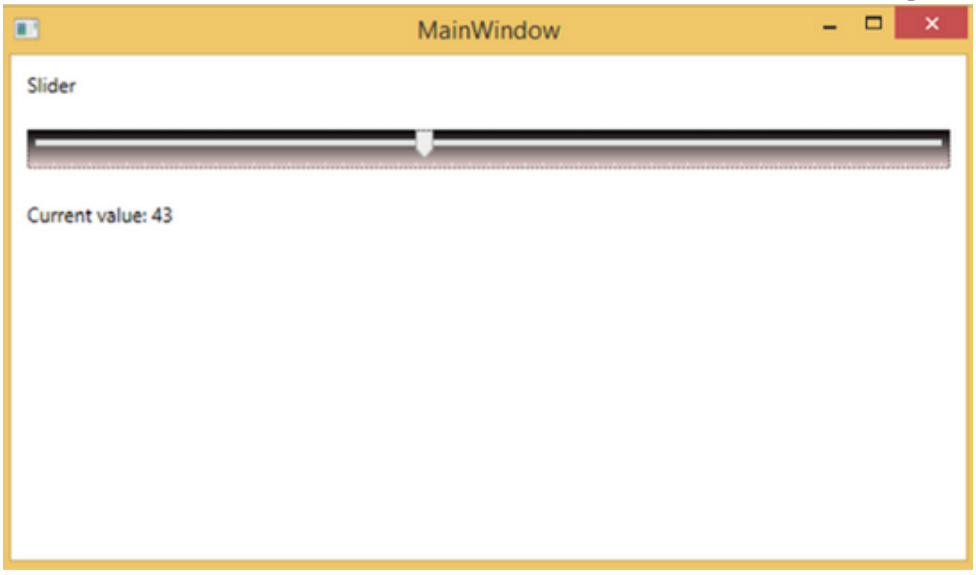
الـواجهة

```
<Window x:Class = "PasswordBox.MainWindow"
   xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml"
   Title = "MainWindow" Height = "350" Width = "604">

   <Grid >
      <PasswordBox x:Name = "pwBox" Height = "35" Width = "200"
         MaxLength = "8" Margin = "159,55,158,229" />
      <Label Content = "Password" HorizontalAlignment = "Left" Margin = "108,61,0,0"
         VerticalAlignment = "Top" Width = "70" />
      <Button Content = "Ok" HorizontalAlignment = "Left" Margin = "406,64,0,0"
         VerticalAlignment = "Top" Width = "75" Click = "Button_Click"/>
      <Label Name = "statusText" HorizontalAlignment = "Left" Margin = "159,128,0,0"
         VerticalAlignment = "Top" Width = "200" Height = "38"/>
   </Grid>

</Window>
```

كود c# المقابل

```
using System.Windows;

namespace WPFPasswordBoxControl {
   /// <summary>
      /// Interaction logic for MainWindow.xaml
   /// </summary>

   public partial class MainWindow : Window {

      public MainWindow() {
         InitializeComponent();
      }

      private void Button_Click(object sender, RoutedEventArgs e) {

         if (pwBox.Password.ToString() == "wpf12345")
            statusText.Text = "Password Accepted";
         else
            statusText.Text = "Wrong Password";
      }

   }
}
```

- **ValueChanged**



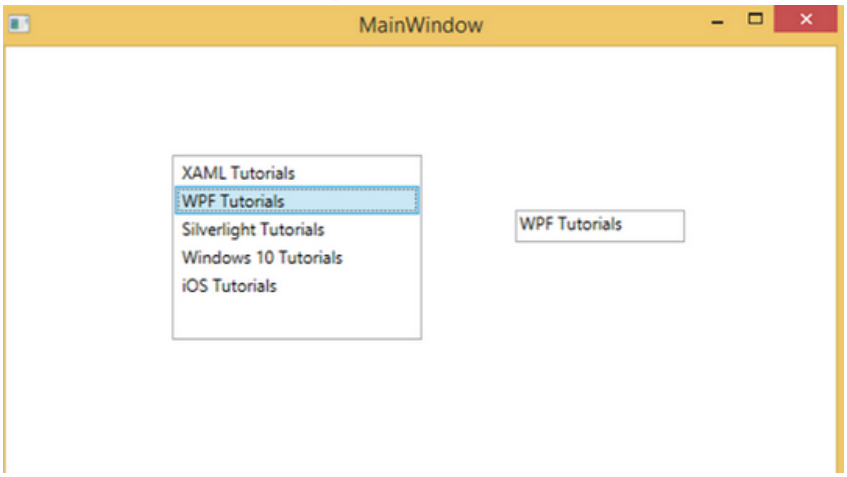
كود الواجهة

```
<Window x:Class = "WPFSliderControl.MainWindow"
   xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d = "http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc = "http://schemas.openxmlformats.org/markup-compatibility/2006"
   xmlns:local = "clr-namespace:WPFSliderControl"
   mc:Ignorable = "d" Title = "MainWindow" Height = "350" Width = "604">

   <StackPanel>
      <TextBlock Text = "Slider" Margin = "10" />

      <Slider x:Name = "slider2" Minimum = "0" Maximum = "100" TickFrequency = "2"
         TickPlacement = "BottomRight" ValueChanged = "slider2_ValueChanged" Margin =
"10">
         <Slider.Background>
            <LinearGradientBrush EndPoint = "0.5,1" StartPoint = "0.5,0">
               <GradientStop Color = "Black" Offset = "0" />
               <GradientStop Color = "#FFF5DCDC" Offset = "1" />
            </LinearGradientBrush>
         </Slider.Background>
      </Slider>

      <TextBlock x:Name = "textBlock1" Margin = "10" Text = "Current value: 0" />
   </StackPanel>

</Window>
```

كود #c المقابل

```
using System;
using System.Windows;

namespace WPFSliderControl {

   public partial class MainWindow : Window {

      public MainWindow() {
         InitializeComponent();
      }

      private void slider2_ValueChanged(object sender,
RoutedPropertyChangedEventArgs<double> e) {
         int val = Convert.ToInt32(e.NewValue);
         string msg = String.Format("Current value: {0}", val);
         this.textBlock1.Text = msg;
      }

   }
}
```

Binding Event




```
<Window x:Class = "WPFListBoxControl.MainWindow"
   xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d = "http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc = "http://schemas.openxmlformats.org/markup-compatibility/2006"
   xmlns:local = "clr-namespace:WPFListBoxControl"
   mc:Ignorable = "d" Title = "MainWindow" Height = "350" Width = "604">

   <Grid>
      <ListBox Name = "listbox" Margin = "118,77,293,103">
         <ListBoxItem Content = "XAML Tutorials" />
         <ListBoxItem Content = "WPF Tutorials" />
         <ListBoxItem Content = "Silverlight Tutorials" />
         <ListBoxItem Content = "Windows 10 Tutorials" />
         <ListBoxItem Content = "iOS Tutorials" />
      </ListBox>

      <TextBox Height = "23" x:Name = "textBox1" Width = "120" Margin = "361,116,0,0"
         HorizontalAlignment = "Left" VerticalAlignment = "Top"
         Text="{Binding SelectedItem.Content, ElementName=listbox}" />
   </Grid>

</Window>
```

جامعة حماه

المعهد التقاني للحاسوب TIC

السنة الثانية

MultiMedia

Lab4

Eng : M.Mahdi Alkhaled

Shapes, Brushes, and Transforms

Shapes :

lines, ellipses, rectangles, and polygons

لها الخصائص المشتركة التالية :

Fill	لتعبئة الشكل .
Stroke	لتلوين الحواف.
StrokeThickness	تحديد ثخانة الحواف
StrokeStartLineCap and StrokeEndLineCap	تحديد محيط الشكل
StrokeDashArray، StrokeDashOffset, and StrokeDashCap	لتحديد حواف مخططة للشكل .
Stretch	لتحديد امتداد الشكل ضمن الحاوية .
DefiningGeometry	لتحديد تنسيق و قياس الشكل .
GeometryTransform	لتحديد إعادة تموضع ودوران و انحراف الشكل

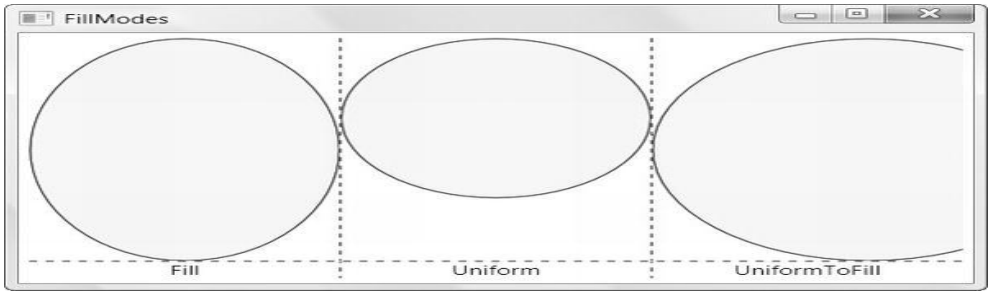
كما يمكن تحديد اسم و حدث للشكل .

Rectangle and Ellipse:

ليظهر الشكل يجب تحديد أحد الخصائص Fill أو Stroke

كما أنه يمكن تحديد الخصائص التالية :

- MinWidth - HorizontalAlignment – VerticalAlignment - - MinHeight Margin.
- RadiusX and RadiusY هي خاصة لل Rectangle حيث أنها تحدد التفاف الزوايا للشكل .
- Stretch : لها القيم المحددة التالية :
  i) Fill : الشكل سوف يمتد على الحاوية كاملة .
  ii) None : الشكل لن يمتد و سوف يظهر بتحديد الأبعاد به .
  iii) Uniform : الشكل سوف يمتد بشكل متساوي للعرض و الارتفاع .
  iv) UniformToFill : الشكل سوف يمتد حتى يلامس حواف الحاوية الموجود ضمنها .



التخطيط Canvas هو المناسب الأفضل للشكال .

نضع الأشكال ضمن العنصر Viewbox و هذا يؤدي إلى تغيير حجم الأشكال بحسب تكبير أو تصغير النافذة .

: Line

```
<Line Stroke="Blue" X1="0" Y1="0" X2="10" Y2="100"></Line>
```

حيث الخصائص المحددة تحدد بداية ونهاية الإحداثيات للمستقيم .

: Polyline

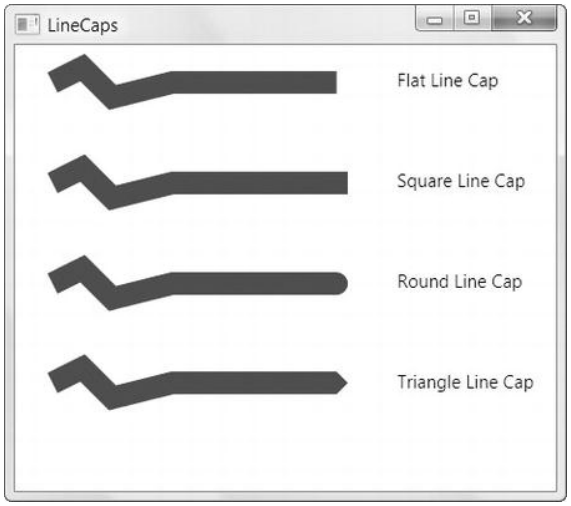
```
<Canvas>
        <Polyline Stroke="Blue" StrokeThickness="5" Points="10,150 30,140 50,160
        70,130
        90,170 110,120 130,180 150,110 170,190 190,100 210,240" >
        </Polyline>
</Canvas>
```

يتم تزويد الوسم بمجموعة أحداثيات للمستقيمات المرسومة .

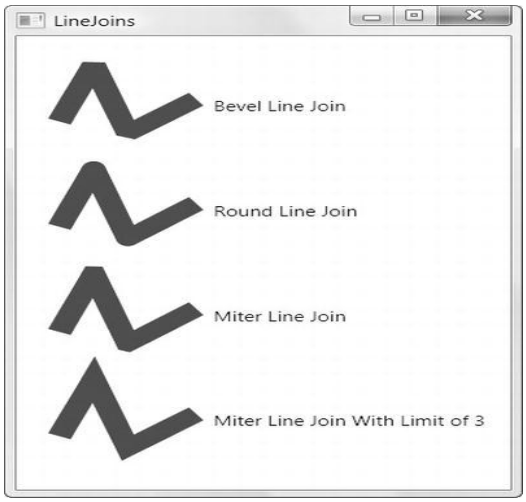
الخاصية "FillRule="Nonzero لتعبئة ما داخل الشكل كامل.

: Line Caps and Line Joins

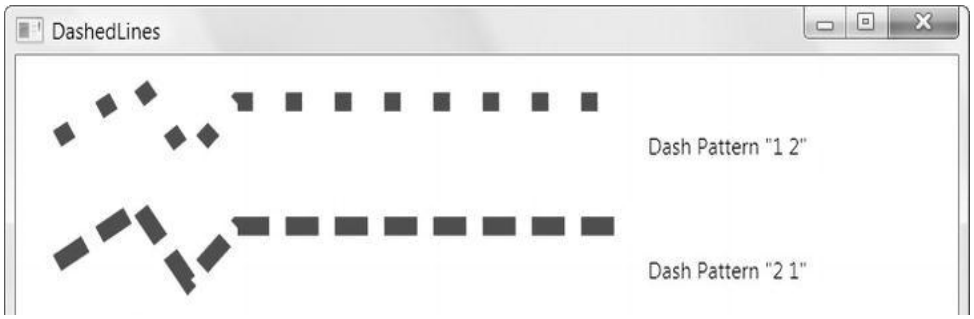
StartLineCap EndLineCap تحدد هذه الخصائص بداية ونهاية الخط المرسوم :



StrokeLineJoin خاصية تحدد طريقة ربط نقاط التقاء المستقيمات :



StrokeDashArray تحدد الخطوط للمستقيمات ك dashes نحدد الفجوة و الخط المرسوم :



Brushes :

أداة الفرشاة لها الصفوف التالية :

SolidColorBrush	الرسم بلون واحد مستمر .
LinearGradientBrush	الرسم بألوان متدرجة
RadialGradientBrush	الرسم بألوان متدرجة لكن بشكل شعاعي .
ImageBrush	الرسم عبر صورة في المنطقة المحددة .
DrawingBrush	الرسم عبر شكل محدد.
VisualBrush	الرسم بعنصر مثل انعكاس لزر.
BitmapCacheBrush	نفس السابقة و لكن إعادة استخدام العنصر للرسم .

: SolidColorBrush

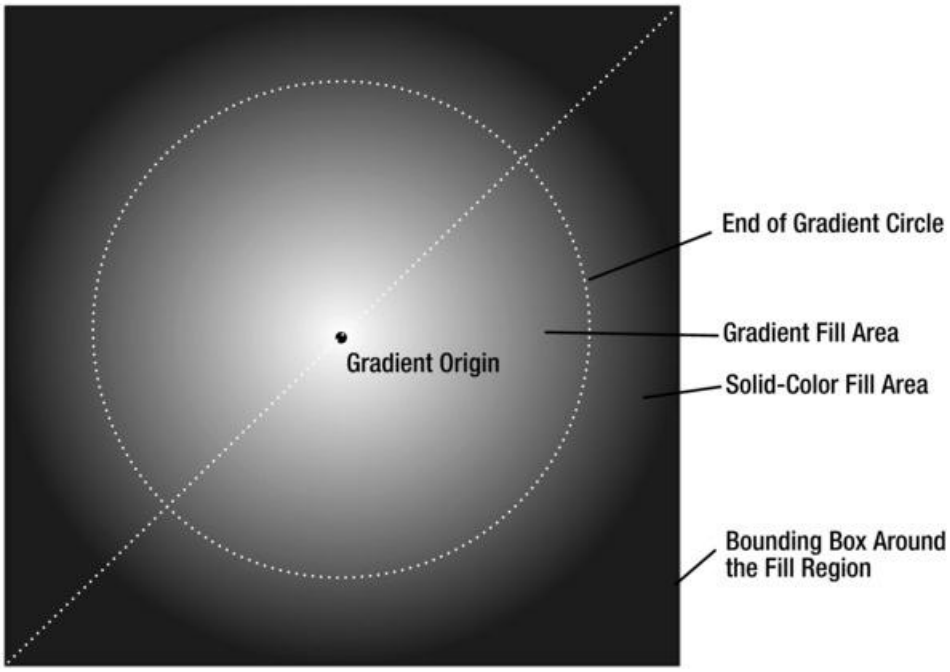
```
<Button>A Button
   <Button.Background>
         <SolidColorBrush Color="Red" />
  </Button.Background>
</Button>
```

: The LinearGradientBrush

```
<Rectangle Width="150" Height="100">
   <Rectangle.Fill>
        <LinearGradientBrush StartPoint="0,0" EndPoint="1,1">
                <GradientStop Color="Yellow" Offset="0.0" />
                <GradientStop Color="Red" Offset="0.25" />
                <GradientStop Color="Blue" Offset="0.75" />
                <GradientStop Color="LimeGreen" Offset="1.0" />
       </LinearGradientBrush>
  </Rectangle.Fill>
</Rectangle>
```

: The RadialGradientBrush

تحديد النقطة المركزية بالخاصية GradientOrigin ، افتراضيا هي ٠.٥ ، ٠.٥




```
<Ellipse Margin="5" Stroke="Black" StrokeThickness="1" Width="200"
Height="200">
        <Ellipse.Fill>
                        <RadialGradientBrush RadiusX="1" RadiusY="1"
                        GradientOrigin="0.7,0.3">
                                <GradientStop Color="White" Offset="0" />
                                <GradientStop Color="Blue" Offset="1" />
                        </RadialGradientBrush>
        </Ellipse.Fill>
</Ellipse>
```

: The ImageBrush

```
<Grid>
<Grid.Background>
<ImageBrush ImageSource="....."></ImageBrush>
</Grid.Background>
```

Viewbox خاصية لتحديد منطقة الفرشاة المستخدمة لرسم الصورة .

Stretch : لتحديد امتداد الرسم .

: The VisualBrush

```
<Button Name="cmd" Margin="3" Padding="5">Is this a real button?</Button>
<Rectangle Margin="3" Height="100">
        <Rectangle.Fill>
        <VisualBrush Visual="{Binding ElementName=cmd}"></VisualBrush>
        </Rectangle.Fill>
</Rectangle>
```

:Transforms

التحويلات على العناصر لها الصفوف التالية :

الصف	عمله	الخصائص
TranslateTransform	للإزاحة	X, Y
RotateTransform	دوران حول نقطة مركزية	Angle, CenterX، CenterY
ScaleTransform	لعمل توسيع للعنصر	ScaleX, ScaleY، CenterX، CenterY
SkewTransform	التواء العنصر	AngleX, AngleY، CenterX، CenterX

نختار الصفوف بحسب الخاصية RenderTransform للعناصر .

: Transparency

Opacity الخاصية لتحديد الشفافية بين ٠ و ١

: **Geometries and Drawings**

الصفوف الهندسية :

LineGeometry	مشابه لLine
RectangleGeometry	مشابه لRectangle
EllipseGeometry	مشابه لEllipse
GeometryGroup	لإضافة مجموعة أشكال هندسية
CombinedGeometry	لدمج مجموعة أشكال هندسية

PathGeometry	تعرض ميزات معقدة إضافية للأشكال

: Line, Rectangle, and Ellipse Geometries

عبر العنصر Path و الخاصية Data له :

```
<Path Fill="Yellow" Stroke="Blue">
        <Path.Data>
                        <RectangleGeometry Rect="0,0
                        100,50"></RectangleGeometry>
        </Path.Data>
</Path>
```

: Combining Shapes with GeometryGroup

```
<Path Fill="Yellow" Stroke="Blue" Canvas.Top="10" Canvas.Left="10" >
        <Path.Data>
                <GeometryGroup>
                        <RectangleGeometry Rect="0,0
                        100,100"></RectangleGeometry>
                        <EllipseGeometry Center="150,50" RadiusX="35"
                        RadiusY="25"></EllipseGeometry>
                </GeometryGroup>
        </Path.Data>
</Path>
```

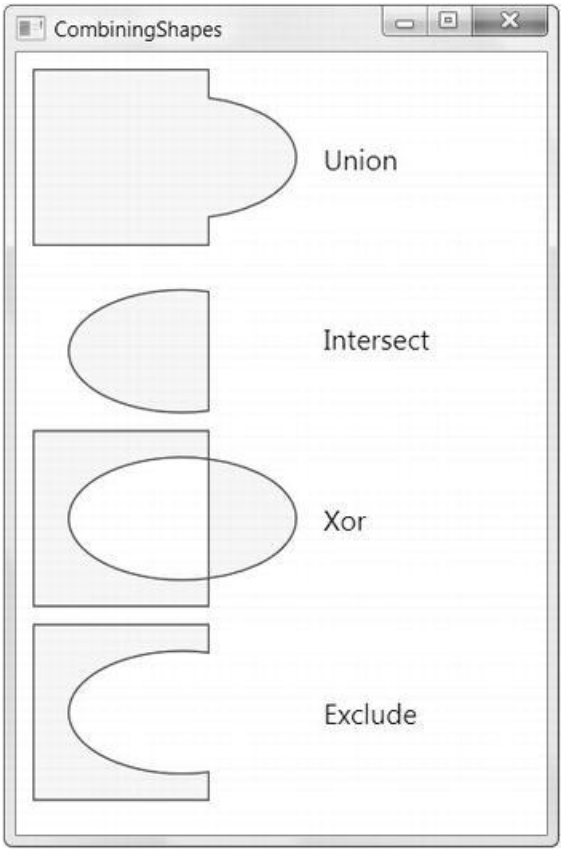
: CombinedGeometry

دمج شكلين فقط عبر هذا الصف و استخدام الخاصية GeometryCombineMode التي لها القيم التالية :

- Union : الشكل يحوي كامل المنطقة للشكلين .
- Intersect : المنطقة المشتركة للشكلين .
- Xor : المنطقة للأول و غير موجودة بالثاني و موجودة بالشكل الثاني غير موجودة بالأول.
- Exclude : المنطقة الموجودة بالشكل الأول و غير الموجودة بالشكل الثاني .

```
<Path Fill="Yellow" Stroke="Blue" Margin="5">
        <Path.Data>
```

```
            <CombinedGeometry GeometryCombineMode="Union">
                <CombinedGeometry.Geometry1>
                ….
                </CombinedGeometry.Geometry1>
                <CombinedGeometry.Geometry2>
                ……
                </CombinedGeometry.Geometry2>
            </CombinedGeometry>
        </Path.Data>
</Path>
```



Curves and Lines with PathGeometry :

من خلال الصف PathFigure الذي له الخصائص التالية :

- StartPoint : نقطة تحدد بداية الخط.
- Segments : مجموعة ال PathSegment لرسم الشكل .
- IsClosed : إذا كانت مفعلة فيتم وصل نهاية وبداية الشكل.
- IsFilled : عند التفعيل يقوم بتعبئة المنطقة في الشكل عبر Path.Fill .

صفوف ال PathSegment :

- LineSegment : رسم خط مستقيم بين البداية والنهاية .
- ArcSegment : شكل بيضوي بين نقطتين .
- PolyLineSegment : لرسم مجموعة خطوط .
- PolyBezierSegment : لرسم مجموعة منحنيات بيزية .

Straight Lines

```
<Path Stroke="Blue">
        <Path.Data>
                <PathGeometry>
                <PathFigure IsClosed="True" StartPoint="10,100">
                        <LineSegment Point="100,100" />
                        <LineSegment Point="100,50" />
                </PathFigure>
                </PathGeometry>

        <Path.Data/>

</Path>
```

Arcs

```
<Path Stroke="Blue" StrokeThickness="3">
        <Path.Data>
                <PathGeometry>
                        <PathFigure IsClosed="False" StartPoint="10,100" >
                        <ArcSegment Point="250,150" Size="200,300" />
                        </PathFigure>
                </PathGeometry>
        </Path.Data>
</Path>
```

:Clipping with Geometry

الخاصية Clip لجميع العناصر تحدد قص هذا العنصر و شكله .

```
<Window.Resources>
<GeometryGroup x:Key="clipGeometry" FillRule="Nonzero">
<EllipseGeometry RadiusX="75" RadiusY="50"
Center="100,150"></EllipseGeometry>
<EllipseGeometry RadiusX="100" RadiusY="25"
Center="200,150"></EllipseGeometry>
```

```
<EllipseGeometry RadiusX="75" RadiusY="130"
Center="140,140"></EllipseGeometry>
</GeometryGroup>
</Window.Resources>

<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition></ColumnDefinition>
<ColumnDefinition></ColumnDefinition>
</Grid.ColumnDefinitions>
<Button Clip="{StaticResource clipGeometry}">A button</Button>
<Image Grid.Column="1" Clip="{StaticResource clipGeometry}"
Stretch="None" Source="creek.jpg"></Image>
</Grid>
```