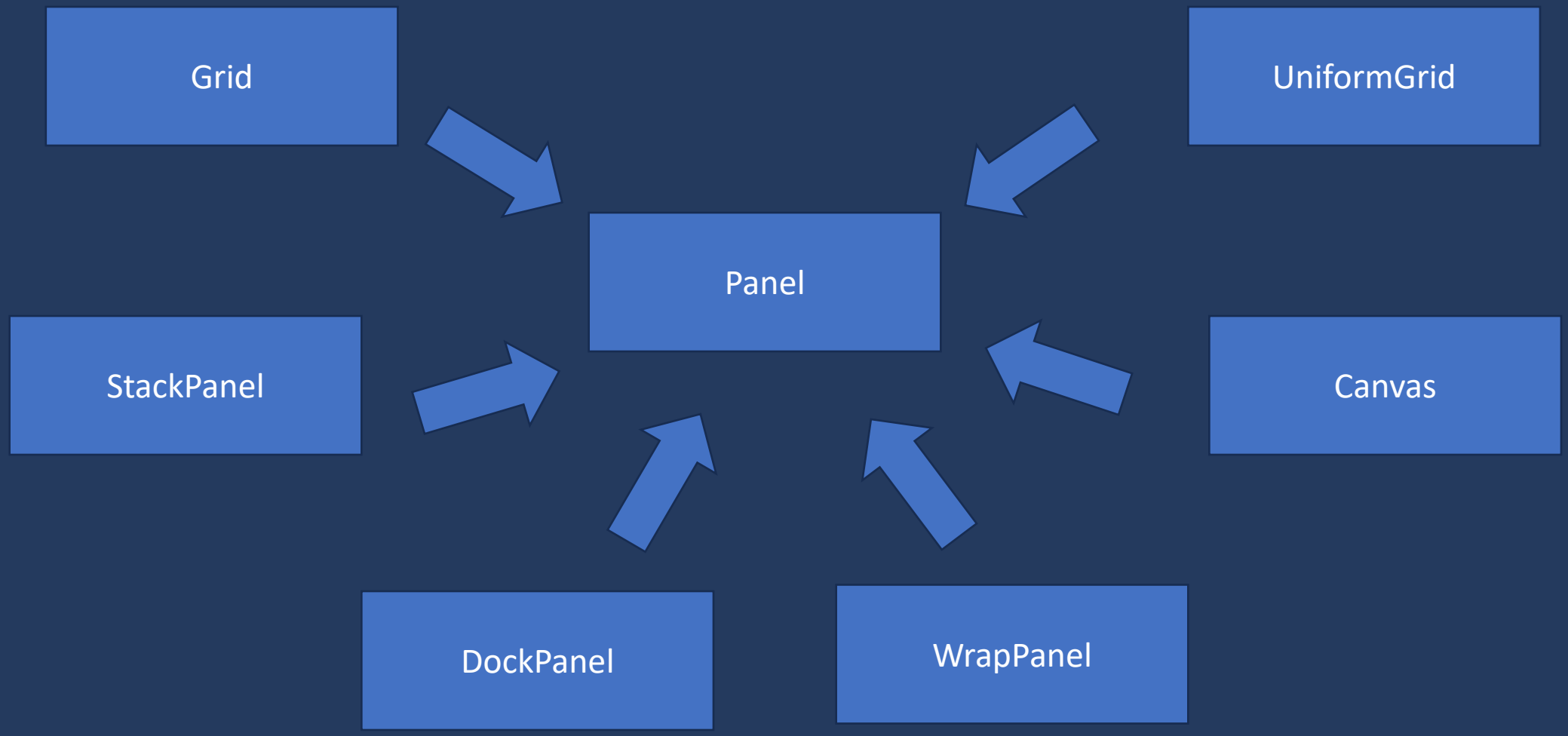
布局系统

张典
An ordinary Avalonia Developer | IRIHI staff

布局控件

常见的布局控件


Grid
UniformGrid
Panel
StackPanel
Canvas
DockPanel
WrapPanel

Panel

```
namespace Avalonia.Controls
{
    Base class for controls that can contain multiple
    children.
    Remarks:  Controls can be added to a Panel by
              adding them to its Children collection.
              All children are layed out to fill the
              panel.
    775 用法   30 继承者   Steven Kirk +6   0+87 扩展方法   9+31 公开 API
    public class Panel : Control, IChildIndexProvider
```

Panel

```
namespace System.Windows.Controls
{
    Base class for all layout panels.
    [Localizability(LocalizationCategory.Ignore)]
    [ContentProperty(name:"Children")]
    109 用法    28 继承者    Vatsan Madhavan +3    10+14 公开 API
    public abstract class Panel : FrameworkElement, IAddChild
```

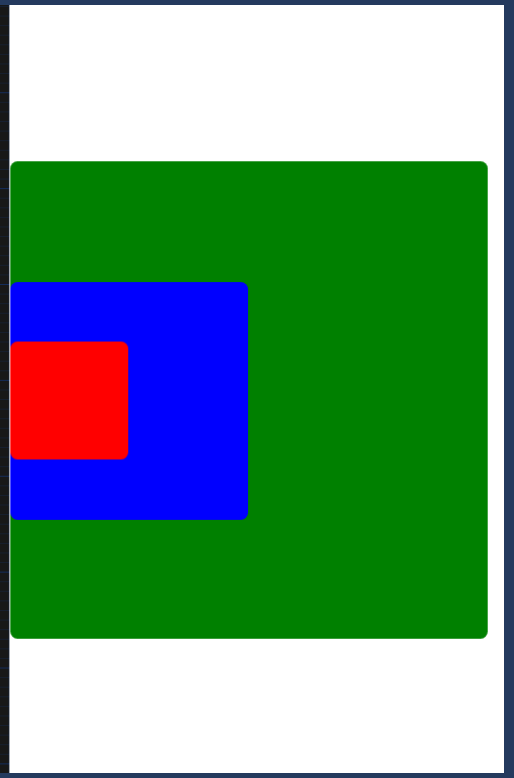
WPF

```
namespace Avalonia.Controls
{
    Base class for controls that can contain multiple
    children.
    Remarks:  Controls can be added to a Panel by
              adding them to its Children collection.
              All children are layed out to fill the
              panel.
    775 用法    30 继承者    Steven Kirk +6    0+87 扩展方法    9+31 公开 API
    public class Panel : Control, IChildIndexProvider
```

Avalonia

Panel

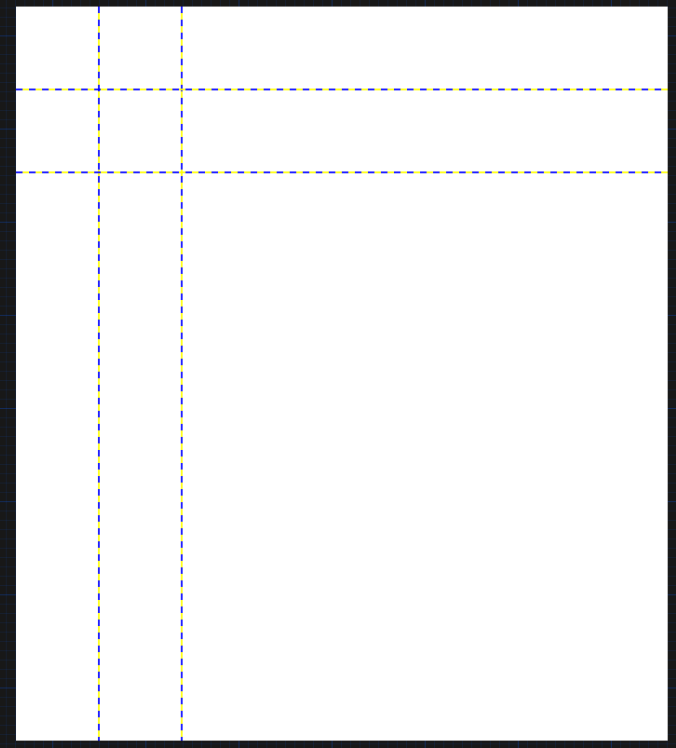
```
<Grid>
    <Button
        Width="200"
        Height="200"
        Background="Green" />
    <Button
        Width="100"
        Height="100"
        Background="Blue" />
    <Button
        Width="50"
        Height="50"
        Background="Red" />
</Grid>
```



使用Grid代替Panel的作用

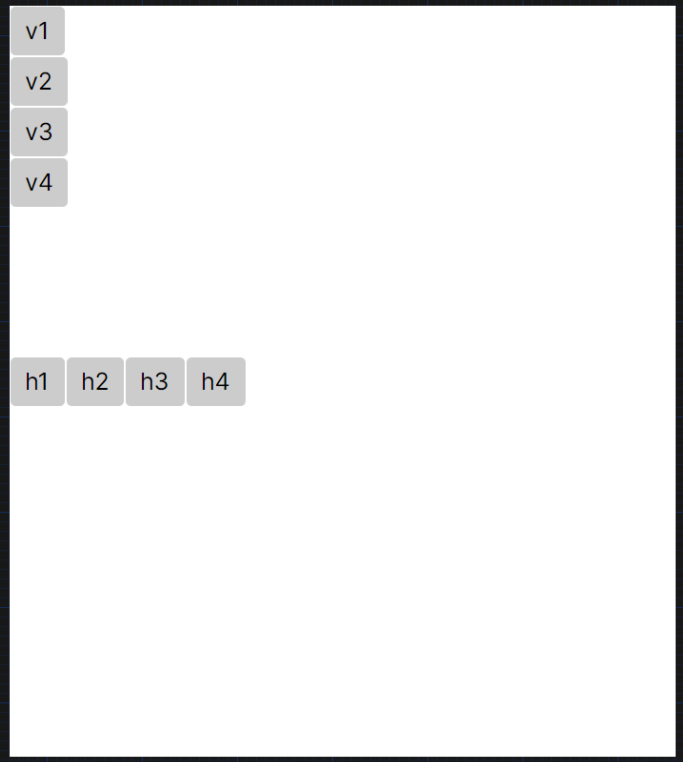
Grid

```
    <!--          Background="Blue" /> -->
    <!--      <Button -->
    <!--          Width="50" -->
    <!--          Height="50" -->
    <!--          Background="Red" /> -->
    <!-- </Grid> -->

    <Grid ColumnDefinitions="50,50,50"
          RowDefinitions="50,50,50"
          ShowGridLines="True" />
</Window>
```



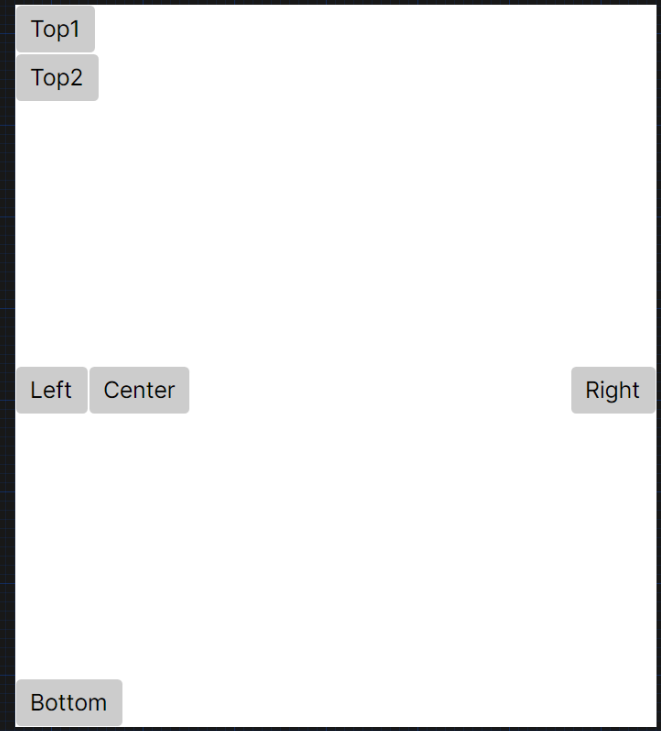
StackPanel

```
<Panel>
    <StackPanel Orientation="Vertical">
        <Button Content="v1" />
        <Button Content="v2" />
        <Button Content="v3" />
        <Button Content="v4" />
    </StackPanel>
    <StackPanel Orientation="Horizontal">
        <Button Content="h1" />
        <Button Content="h2" />
        <Button Content="h3" />
        <Button Content="h4" />
    </StackPanel>
</Panel>
```



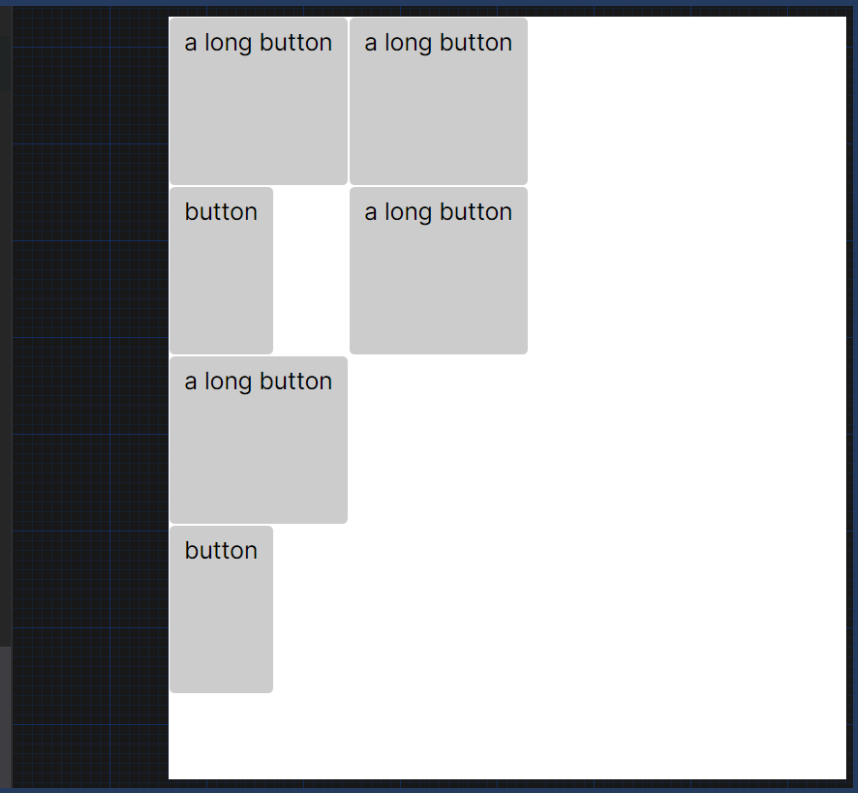
DockPanel

```
<!--     </StackPanel> -->
<!-- </Panel> -->

<DockPanel>
    <Button DockPanel.Dock="Top" Content="Top1" />
    <Button DockPanel.Dock="Top" Content="Top2" />
    <Button DockPanel.Dock="Bottom" Content="Bottom" />
    <Button DockPanel.Dock="Left" Content="Left" />
    <Button DockPanel.Dock="Right" Content="Right" />
    <Button Content="Center" />
</DockPanel>
Window>
```



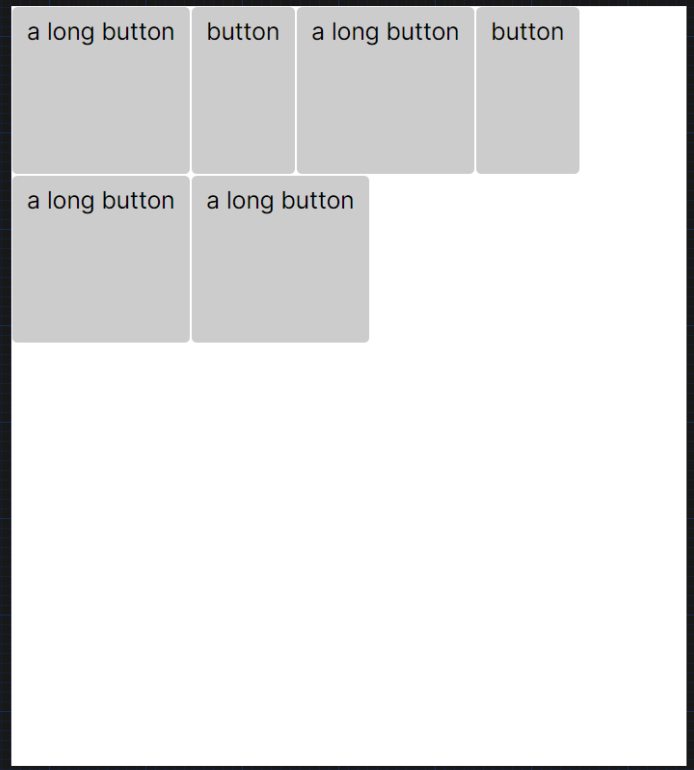
WrapPanel

```
<!-- </DockPanel> -->

<WrapPanel Orientation="Vertical">
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="a long button" />
</WrapPanel>

Window>
```



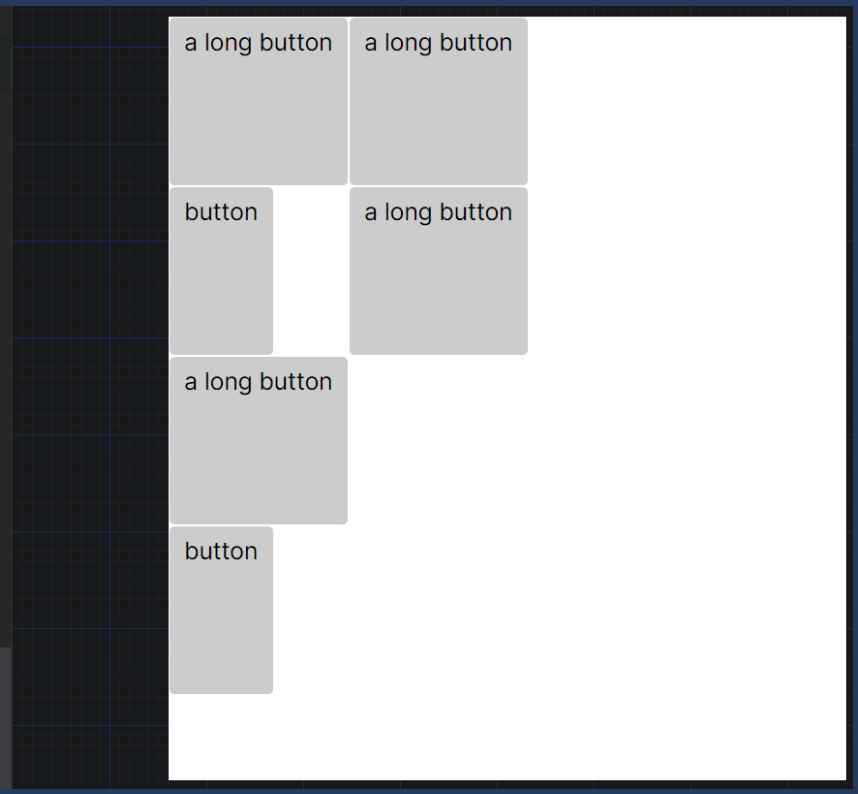
WrapPanel

```
<!-- </DockPanel> -->

<WrapPanel>
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="a long button" />
</WrapPanel>

</Window>
```



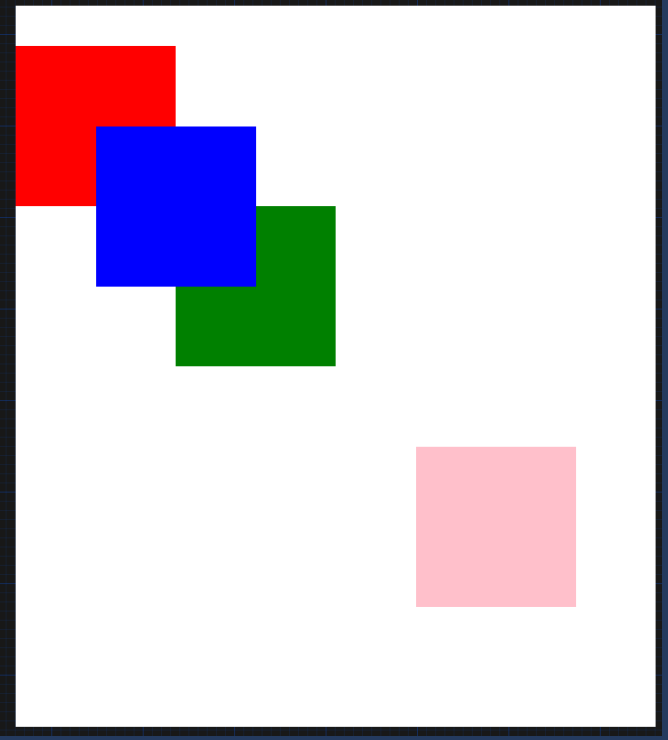
WrapPanel

```
<!-- </DockPanel> -->

<WrapPanel Orientation="Vertical">
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="button" />
    <Button Height="100" Content="a long button" />
    <Button Height="100" Content="a long button" />
</WrapPanel>

Window>
```



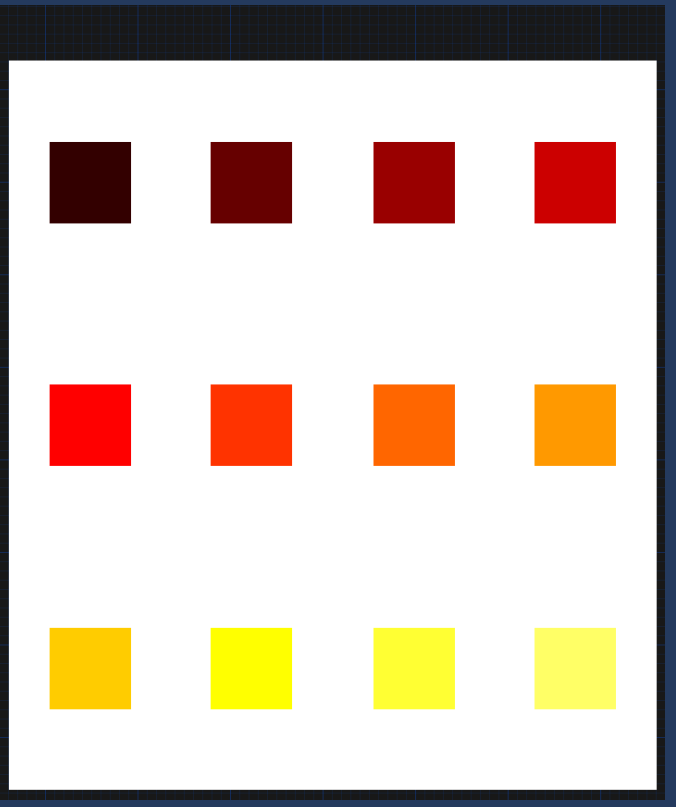
Canvas

```
<Canvas Height="400" Width="400">
    <Panel Height="100" Width="100"
           Canvas.Top="0" Canvas.Left="0"
           Background="Red" />
    <Panel Height="100" Width="100"
           Canvas.Top="100" Canvas.Left="100"
           Background="Green" />
    <Panel Height="100" Width="100"
           Canvas.Top="50" Canvas.Left="50"
           Background="Blue" />
    <Panel Height="100" Width="100"
           Canvas.Bottom="50" Canvas.Right="50"
           Background="Pink" />
</Canvas>
indow>
```



UniformGrid

```
<UniformGrid Rows="3" Columns="4">
    <Rectangle Width="50" Height="50" Fill="#330000"/>
    <Rectangle Width="50" Height="50" Fill="#660000"/>
    <Rectangle Width="50" Height="50" Fill="#990000"/>
    <Rectangle Width="50" Height="50" Fill="#CC0000"/>
    <Rectangle Width="50" Height="50" Fill="#FF0000"/>
    <Rectangle Width="50" Height="50" Fill="#FF3300"/>
    <Rectangle Width="50" Height="50" Fill="#FF6600"/>
    <Rectangle Width="50" Height="50" Fill="#FF9900"/>
    <Rectangle Width="50" Height="50" Fill="#FFCC00"/>
    <Rectangle Width="50" Height="50" Fill="#FFFF00"/>
    <Rectangle Width="50" Height="50" Fill="#FFFF33"/>
    <Rectangle Width="50" Height="50" Fill="#FFFF66"/>
</UniformGrid>
ndow>
```



Alignment, Margin & Padding

Alignment

Defines how a control aligns itself horizontally in its parent control.

250 用法 Steven Kirk 15 公开 API

```
public enum HorizontalAlignment
{
    The control stretches to fill the width of the parent control.
    Stretch,

    The control aligns itself to the left of the parent control.
    Left,

    The control centers itself in the parent control.
    Center,

    The control aligns itself to the right of the parent control.
    Right,
}
```

Defines how a control aligns itself vertically in its parent control.

262 用法 Steven Kirk 15 公开 API

```
public enum VerticalAlignment
{
    The control stretches to fill the height of the parent control.
    Stretch,

    The control aligns itself to the top of the parent control.
    Top,

    The control centers itself within the parent control.
    Center,

    The control aligns itself to the bottom of the parent control.
    Bottom,
}
```

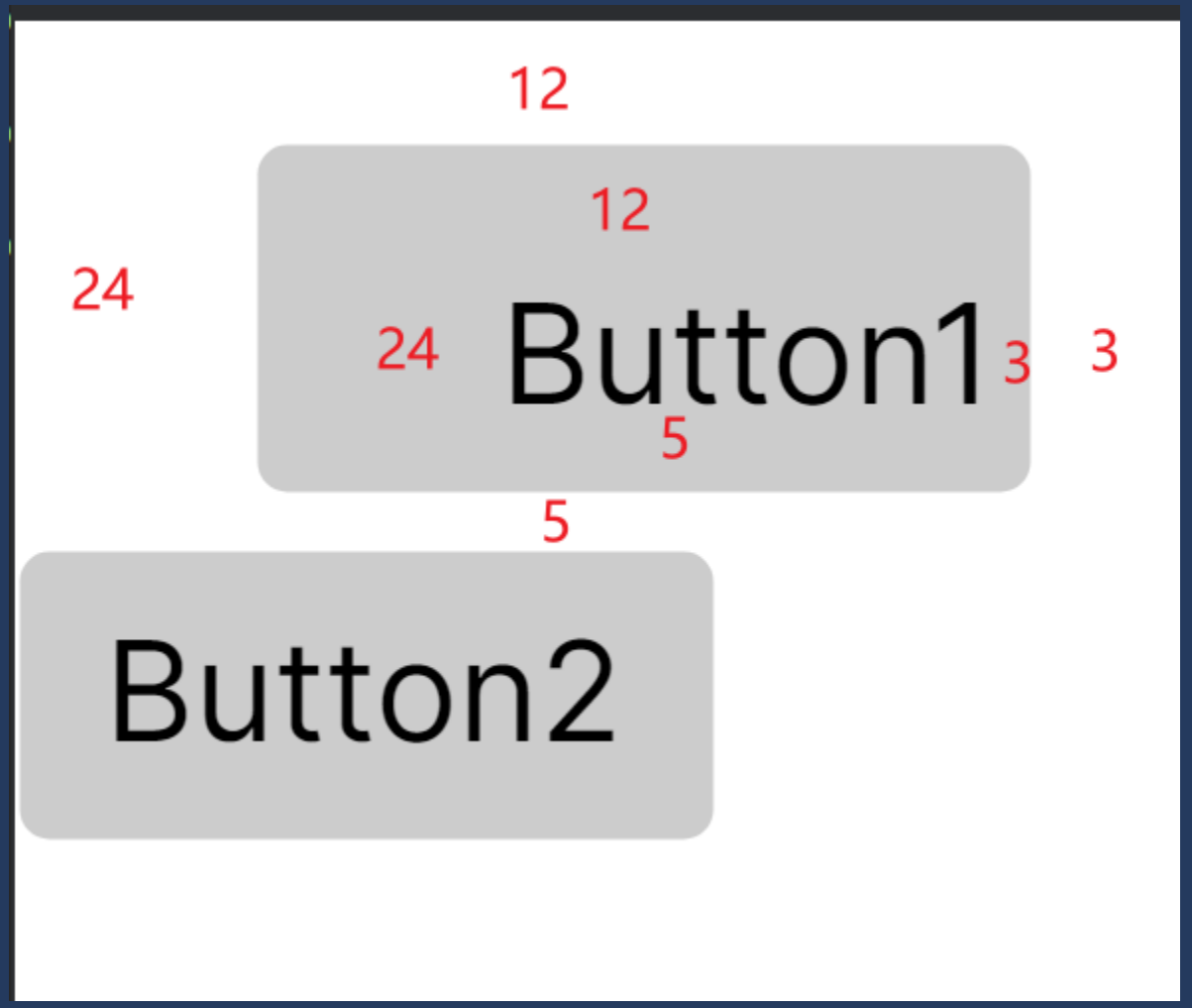
Margin & Padding

```
Gets or sets the margin around the
element.
679 用法    Steven Kirk +1
public Thickness Margin
{
    ◇
    get { return GetValue(MarginProperty); }
    ◇
    set { SetValue(MarginProperty, value); }
}
```

```
Gets or sets the padding to place around
the Child control.
362 用法    Steven Kirk +1
public Thickness Padding
{
    get ⇒ GetValue(PaddingProperty);
    set ⇒ SetValue(PaddingProperty, value);
}
```

Margin & Padding

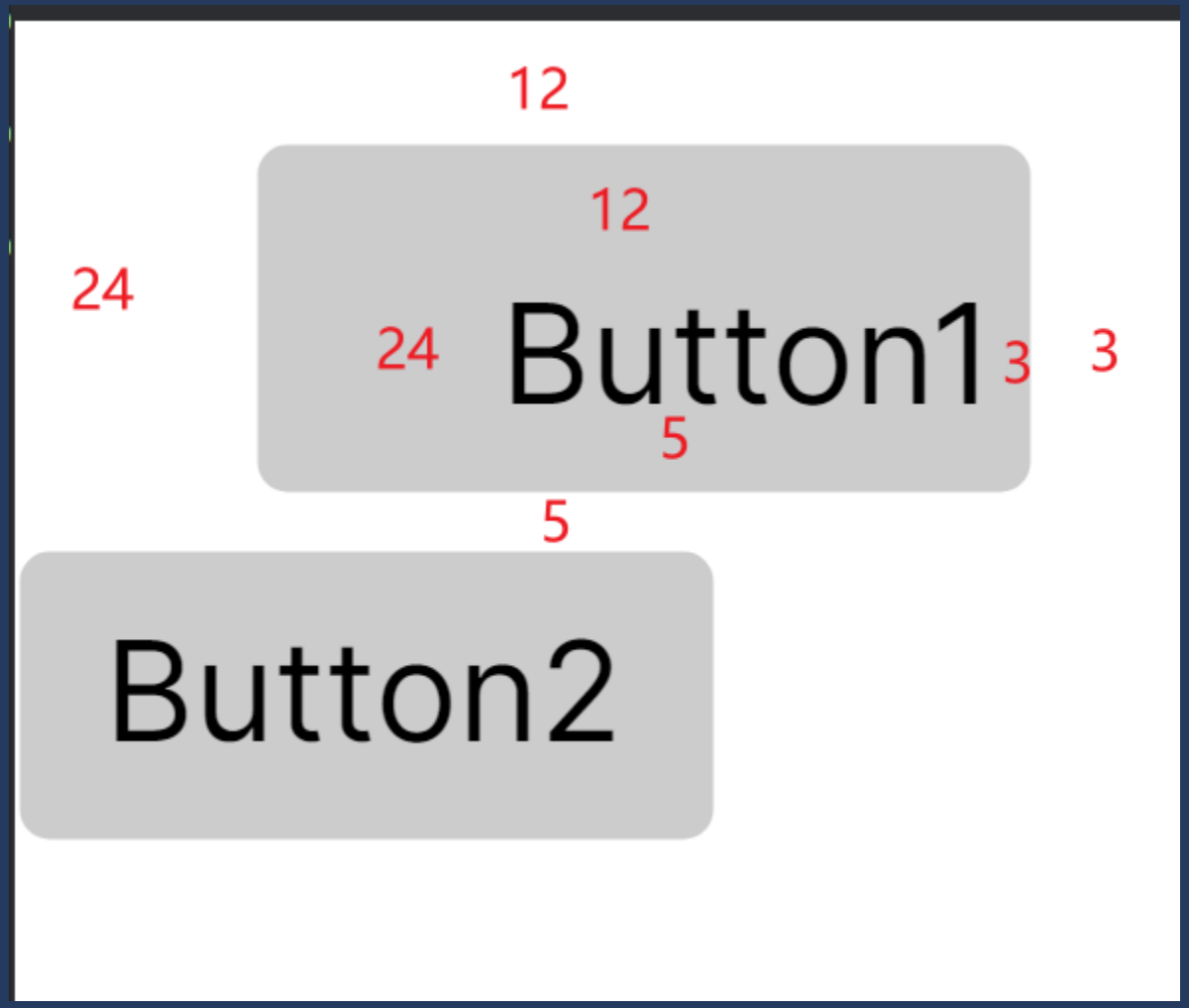
```
<StackPanel>
    <Button
        Margin="[l:]24 [t:]12 [r:]3 [b:]5"
        Padding="[l:]24 [t:]12 [r:]3 [b:]5"
        Content="Button1" />
    <Button Content="Button2" />
</StackPanel>
```



Margin & Padding

```
<StackPanel>
    <Button
        Margin=" [l:] 24  [t:] 12  [r:] 3  [b:] 5"
        Padding=" [l:] 24  [t:] 12  [r:] 3  [b:] 5"
        Content="Button1" />
    <Button Content="Button2" />
</StackPanel>
```

```
padding: 24px 12px 3px 5px;
  padding-top: 24px;
  padding-right: 12px;
  padding-bottom: 3px;
  padding-left: 5px;
```



Margin & Padding

```
static Layoutable()
{
    AffectsMeasure<Layoutable>(
        params properties:WidthProperty,
        HeightProperty,
        MinWidthProperty,
        MaxWidthProperty,
        MinHeightProperty,
        MaxHeightProperty,
        MarginProperty,
        HorizontalAlignmentProperty,
        VerticalAlignmentProperty);
}
```

Thank you.