Windows Presentation Foundation

WPF, C#, Desktop

Orientation

- 실제 Layout & Event 수정을 위한 파일은 아래 두 가지이다.
  - Layout : MainWindow.xaml
  - Logic : Mainwindow.xaml.cs

예시

- 클릭하면 TextBox가 "Hello World!" 및 0부터 1씩 증가.
- 추가적으로 배경색을 공유하고, 체크박스에 버튼 내용(Text)를 가져온다.

```
<Window x:Class="WpfApplication1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Window.Resources>
        <SolidColorBrush x:Key="bg" Color="red" />
    </Window.Resources>
    <StackPanel>
        <Button x:Name="btn1" Click="MyButtonClicked"
Background="{StaticResource bg}">Hello World</Button> <!-- 배경 공유 -->
        <Button x:Name="btn2" Click="MyButtonClicked"
Background="{StaticResource bg}">Reset</Button> <!-- 배경 공유 -->
        <TextBox x:Name="text1">This is Textbox</TextBox>
        <CheckBox Content="{Binding ElementName=btn1, Path=Content}"/> <!--
Button Text 가져오기 -->
    </StackPanel>
</Window>
```

```
using System;
using System.Windows;

namespace WpfApplication1
{
    public partial class MainWindow : Window
    {

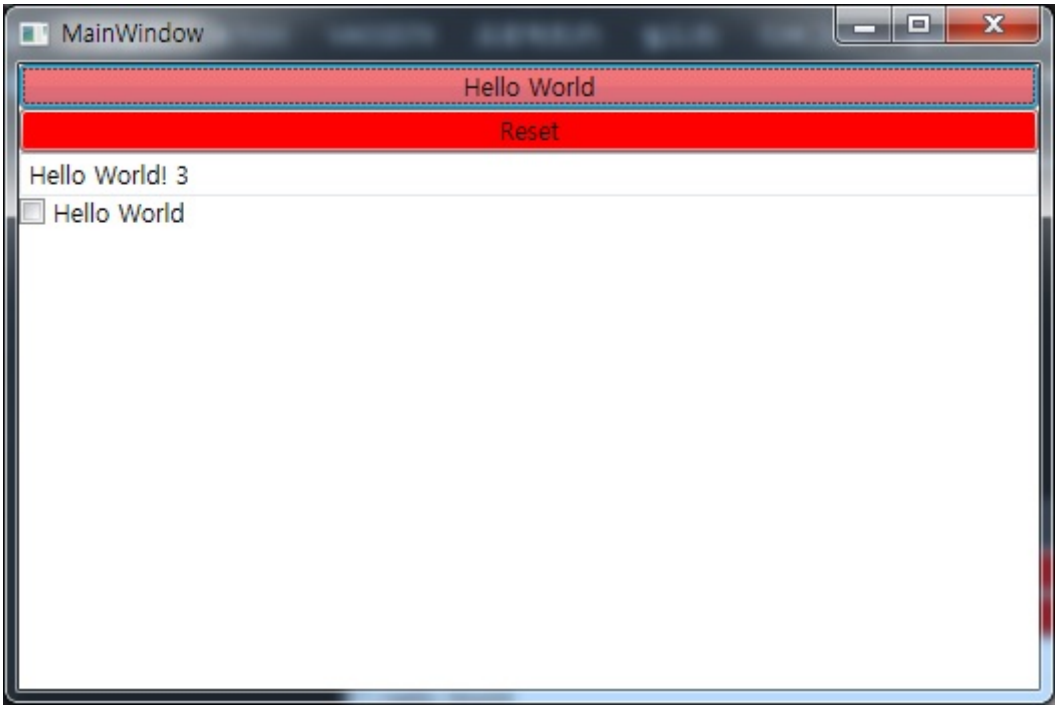
        private static int count = 0;

        public MainWindow()
        {
```

```
            InitializeComponent();
        }

        void MyButtonClicked(object sender, RoutedEventArgs e)
        {
            Button btn = (Button) sender;

            if (btn.Name.ToString().Equals("btn1"))
            {
                count++;
                text1.Text = "Hello World! " + count;
            }
            else
            {
                text1.Text = "This is Textbox";
                count = 0;
            }
        }
    }
}
```



Layout

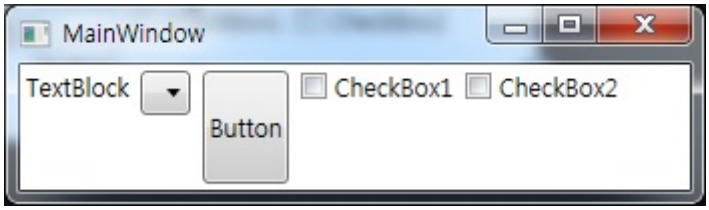
- WPF에서 Layout은 **Panel**들에 의해 관리된다.

StackPanel

- 가로 또는 세로 방향으로 지정할 수 있는 한 줄에 자식 요소를 정렬
- Android의 LinearLayout과 같은 개념. (선형 배치)

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="350" Height="100">
    <StackPanel Orientation="Horizontal">
        <TextBlock Margin="3">TextBlock</TextBlock>
        <ComboBox Margin="3" VerticalAlignment="top" />
        <Button Margin="3">Button</Button>
        <CheckBox Margin="3">CheckBox1</CheckBox>
        <CheckBox Margin="3">CheckBox2</CheckBox>
    </StackPanel>
</Window>
```

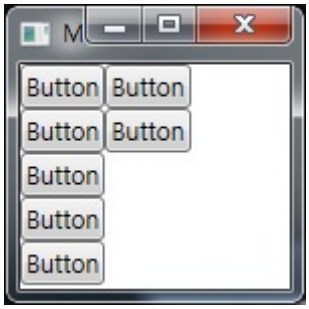
- Orientation : Pannel 안 자식 요소의 **배치 방향** 변경
    - Vertical (Default), Horizontal
- VerticalAlignment : 세로 크기를 내용물만큼 조정하고, 상대적인 위치를 속성값에 따라 배치시킨다.
    - stretch (Default), center, top, bottom
- HorizontalAlignment : 가로 크기를 내용물만큼 조정하고, M상대적인 위치를 속성값에 따라 배치시킨다.
    - stretch (Default), center, left, right



WrapPanel

- 자식 요소들을 지정 방향에 따라 순서대로 배치하고, 너비를 넘을 때는 적절하게 다음 줄로 이동시킨다.

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="150" Height="150">
    <WrapPanel Orientation="Vertical"> <!-- default : Horizontal -->
        <Button>Button</Button>
        <Button>Button</Button>
        <Button>Button</Button>
        <Button>Button</Button>
        <Button>Button</Button>
        <Button>Button</Button>
        <Button>Button</Button>
    </WrapPanel>
</Window>
```



DockPanel

- 레이아웃 컨테이너의 가장자리를 기준으로 자식 콘텐츠의 위치를 지정하는 데 사용.

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">
    <DockPanel>
        <Button DockPanel.Dock="Top">Top</Button>
        <Button DockPanel.Dock="Bottom">Bottom</Button>
        <Button DockPanel.Dock="Left">Left</Button>
        <Button DockPanel.Dock="Right">Right</Button>
        <Button>Fill</Button>
    </DockPanel>
</Window>
```



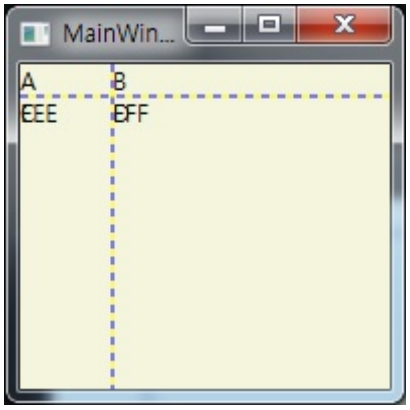
Grid

- 열과 행으로 자식 요소 배치
- 컬럼 형태를 정의하고, 내용을 채운다. Index를 넘어가는(Overflow) 경우, 마지막 Index에 맞춤.

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">

    <Grid Background="Beige" ShowGridLines="True">
        <Grid.RowDefinitions>
            <!-- Auto는 내용물에 맞춤. -->
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
     <!-- "숫자*"을 붙이면 상대적 배율 속성. 1:3 -->
            <ColumnDefinition Width="1*" />
            <ColumnDefinition Width="3*"/>
        </Grid.ColumnDefinitions>

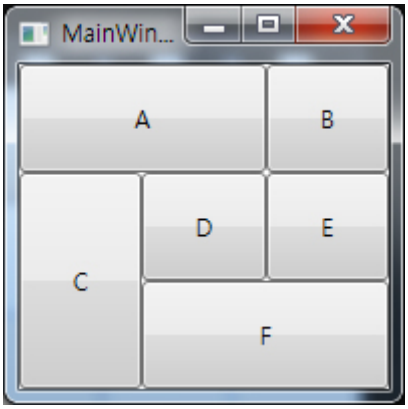

        <TextBlock Grid.Row="0" Grid.Column="0">A</TextBlock>
        <TextBlock Grid.Row="0" Grid.Column="1">B</TextBlock>
        <TextBlock Grid.Row="1" Grid.Column="0">C</TextBlock>
        <TextBlock Grid.Row="1" Grid.Column="1" >D</TextBlock>
        <TextBlock Grid.Row="2" Grid.Column="0" >EEE</TextBlock>
   <!-- 마지막 열에 배치 -->
        <TextBlock Grid.Row="2" Grid.Column="1">FFF</TextBlock>
   <!-- 마지막 행에 배치 -->
     </Grid>
</Window>
```



여러 행/열 걸치기 및 병합 (RowSpan, ColumnSpan)

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="200" Width="200">
    <Grid>
```

```
        <Grid.RowDefinitions>
            <RowDefinition />
            <RowDefinition />
            <RowDefinition />
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition />
            <ColumnDefinition />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Button Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="2">A</Button>
        <Button Grid.Row="0" Grid.Column="2">B</Button>
        <Button Grid.Row="1" Grid.Column="0" Grid.RowSpan="2">C</Button>
        <Button Grid.Row="1" Grid.Column="1">D</Button>
        <Button Grid.Row="1" Grid.Column="2">E</Button>
        <Button Grid.Row="2" Grid.Column="1" Grid.ColumnSpan="2">F</Button>
    </Grid>
</Window>
```




```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">

    <Grid Background="Beige">
        <Grid.Resources>
            <Style TargetType="TextBlock">
                <Setter Property="Margin" Value="5,3" />
            </Style>
        </Grid.Resources>

        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
```

```
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>

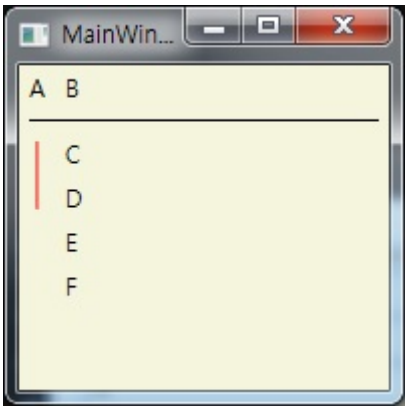
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>


        <TextBlock Grid.Row="0" Grid.Column="0">A</TextBlock>
        <TextBlock Grid.Row="0" Grid.Column="1">B</TextBlock>

        <Rectangle Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="2"
Margin="5" Height="1" Fill="Black" />

        <Rectangle Grid.Row="2" Grid.Column="0" Grid.RowSpan="2" Margin="5"
Width="1" Fill="Red" />
        <TextBlock Grid.Row="2" Grid.Column="1">C</TextBlock>
        <TextBlock Grid.Row="3" Grid.Column="1">D</TextBlock>
        <TextBlock Grid.Row="4" Grid.Column="1">E</TextBlock>
        <TextBlock Grid.Row="5" Grid.Column="1">F</TextBlock>

    </Grid>
</Window>
```



동적 Grid 생성

```
/* xaml에서 grid에 "myGrid"라는 Name 부여. 그리고 loaded 연결.*/

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

namespace DynamicallyGrid
{
    public partial class MainWindow : Window
```

```
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            myGrid.ShowGridLines = true;

            for(int row=0 ; row<3 ; row++)
                myGrid.RowDefinitions.Add(new RowDefinition() { Height = new
GridLength(50) });

            for (int column = 0; column < 4; column++)
                myGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width
= new GridLength(50) });

            SetLocation(new TextBlock() { Text = "Hello" }, 0, 0);
            SetLocation(new TextBlock() { Name = "name2", Text = "olleh" },
1, 1);
            SetLocation(new TextBlock() { Text = "WPF" }, 2, 2);
            SetLocation(new TextBlock() { Text = "QWERTY" }, 1, 3);

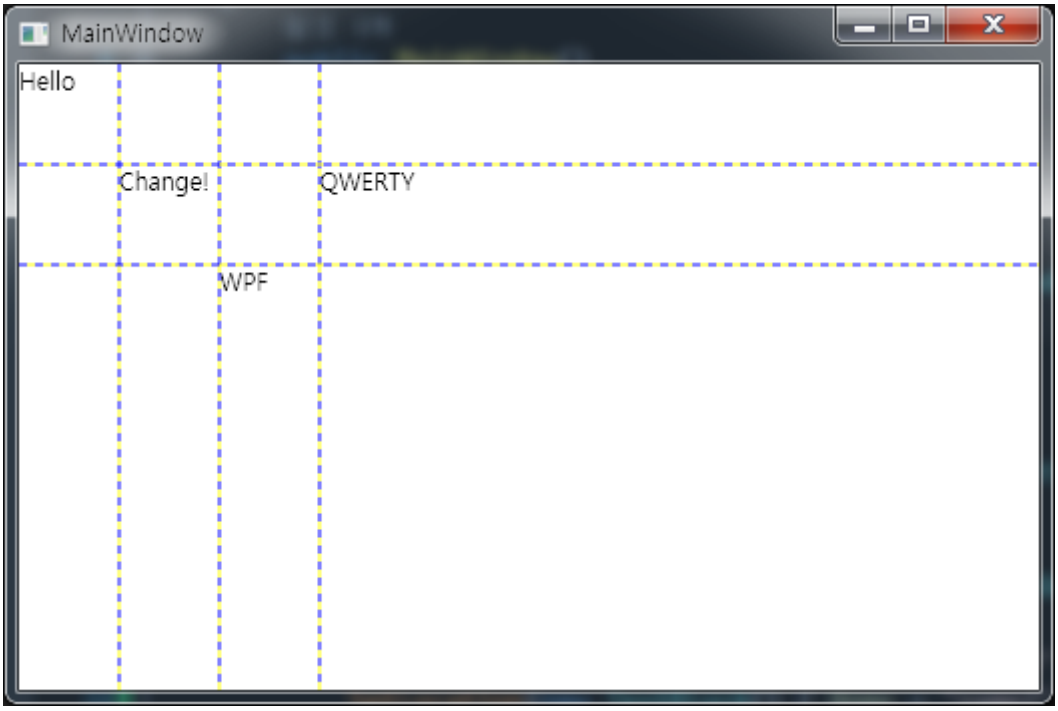


        }

        private void SetLocation(TextBlock tb, int row, int column)
        {
    //이름 추적이 가능한 지 시험.
            if(tb.Name == "name2")
                tb.Text = "Change!";

            Grid.SetRow(tb, row);
            Grid.SetColumn(tb, column);

            myGrid.Children.Add(tb);
        }
    }
}
```

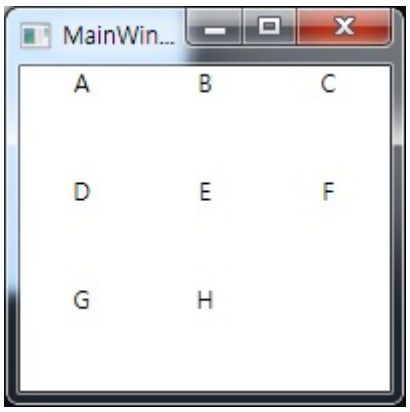


UniformGrid

- 모든 칸이 똑같은 크기로 Grid안에 정렬된다.
- 균형있게 순차 배치.

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">

    <UniformGrid TextBlock.TextAlignment="Center">
        <TextBlock Text="A" />
        <TextBlock Text="B" />
        <TextBlock Text="C" />
        <TextBlock Text="D" />
        <TextBlock Text="E" />
        <TextBlock Text="F" />
        <TextBlock Text="G" />
        <TextBlock Text="H" />
    </UniformGrid>
</Window>
```

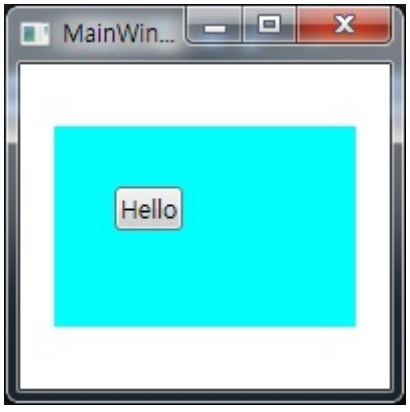


Canvas

- 절대좌표로 자식 요소 위치 지정

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">

    <Canvas Background="Cyan" Width="150" Height="100">
        <Button Canvas.Left="30" Canvas.Top="30">Hello</Button> <!-- Canvas
크기 기준으로 배치 -->
    </Canvas>
</Window>
```



Viewbox

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="500" Height="200">
```

```
    <Viewbox Stretch="Fill"> <!-- Default : Uniform (창 크기에 맞추기) -->
        <Canvas Width="18" Height="18" VerticalAlignment="Center">
            <Ellipse Canvas.Left="1" Canvas.Top="1" Width="16" Height="16"
Fill="Yellow" Stroke="Black" />
            <Ellipse Canvas.Left="4.5" Canvas.Top="5" Width="2.5" Height="3"
Fill="Black" />
            <Ellipse Canvas.Left="11" Canvas.Top="5" Width="2.5" Height="3"
Fill="Black" />
            <Path Data="M 5,10 A 3,3 90 0 0 13,10" Stroke="Black" />
        </Canvas>
    </Viewbox>
</Window>
```



ScrollViewer

안쪽에 다른 Layout/Control을 삽입하여 사용

```
<Window x:Class="Layout1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="500" Height="200">
    <ScrollViewer HorizontalScrollBarVisibility="Auto">
        <Ellipse Fill="Cyan" Height="1000" Width="2000" />
    </ScrollViewer>

</Window>
```



Custom Layout

Panel 클래스를 상속받아 Method Overriding 한다.

Control

Buttons

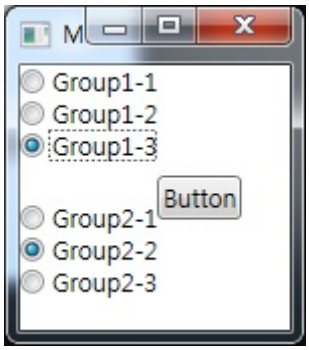
```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="170" Width="150">
    <StackPanel Orientation="Horizontal">

        <StackPanel HorizontalAlignment="Left">
            <StackPanel Margin="0,0,0,20">
                <RadioButton GroupName="group1">Group1-1</RadioButton>
                <RadioButton GroupName="group1">Group1-2</RadioButton>
                <RadioButton GroupName="group1">Group1-3</RadioButton>
            </StackPanel>

            <StackPanel>
                <RadioButton GroupName="group2">Group2-1</RadioButton>
                <RadioButton GroupName="group2">Group2-2</RadioButton>
                <RadioButton GroupName="group2">Group2-3</RadioButton>
            </StackPanel>
        </StackPanel>


        <Button VerticalAlignment="Center">
            <AccessText>Button</AccessText> <!-- 이런 형태로 Text 삽입 가능 -->
        </Button>
    </StackPanel>
```

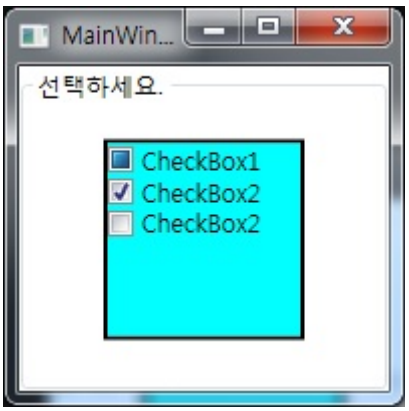
```
</Window>
```



CheckBox

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="200" Width="200">

    <GroupBox>
        <GroupBox.Header>선택하세요.</GroupBox.Header>
        <Border Name="border" BorderBrush="Black" Background="Cyan"
BorderThickness="2" Width="100" Height="100">
            <StackPanel>
                <CheckBox IsChecked="{x:Null}">CheckBox1</CheckBox>
                <CheckBox IsChecked="True">CheckBox2</CheckBox>
                <CheckBox IsChecked="False">CheckBox2</CheckBox>
            </StackPanel>
        </Border>
    </GroupBox>
</Window>
```



Slider & Scrollbar

Slider vs. ScrollBar ⇒ **구간 설정 여부**

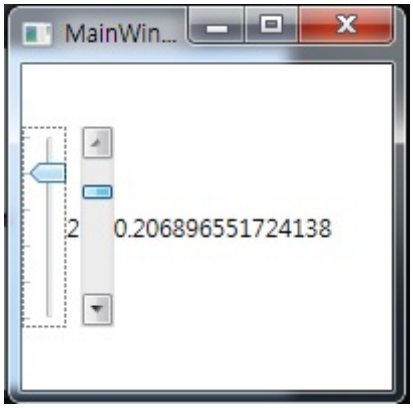
```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="200" Width="200">

    <!--
        IsSnapToTickEnabled : Drag시 딱딱 끊어질 것인 지 여부 (구간 설정)
        TickPlacement : 구간 눈금 표시
        TickFrequency : 구간 이동 주기 (Maximum/n)
        AutoToolTipPlacement : Drag시 Tooltip 표시 여부
        AutoToolTipPrecision : Tooltip 표시할 때 소수점 몇 자리까지 보일 지 설정.
        IsDirectionReversed : Slider 방향 반전 여부
    -->

    <StackPanel Orientation="Horizontal">
        <!-- Slider -->
        <StackPanel Orientation="Horizontal">
        <Slider Name="mySlider" Orientation="Vertical" Height="100"
Minimum="0"  Maximum="10"
            IsSnapToTickEnabled="True" TickPlacement="TopLeft"
TickFrequency="2"
            AutoToolTipPlacement="BottomRight" AutoToolTipPrecision="3"
            IsDirectionReversed="True"/>

        <TextBlock Text="{Binding ElementName=mySlider, Path=Value}"
VerticalAlignment="Center" ></TextBlock>
        </StackPanel>
        <!-- ScrollBar -->
        <StackPanel Orientation="Horizontal">
            <ScrollBar Name="sb" Orientation="Vertical"
Height="100"></ScrollBar>
        </StackPanel>
        <TextBlock Text="{Binding ElementName=sb, Path=Value}"
VerticalAlignment="Center" ></TextBlock>
    </StackPanel>

</Window>
```

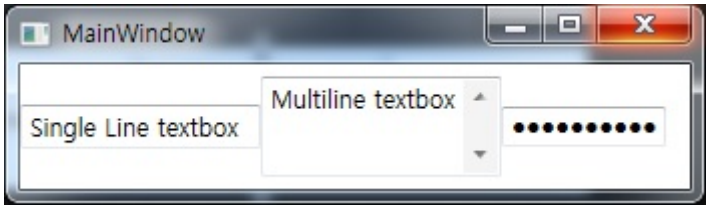


ProgressBar

- 예제 생략.
  ⇒ 보통 Progressbar는 동적으로 활용하기 때문에 정적인 예제가 필요 없음.

Text Controls

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="100" Width="350">
    <StackPanel Orientation="Horizontal">
        <TextBox Width="120" VerticalAlignment="Center" Text="Single Line
textbox" />
        <!--
            AcceptsReturn : Enter Key 개행 여부
        -->
        <TextBox Width="120" AcceptsReturn="True" Height="50"
                 VerticalScrollBarVisibility="Visible"
VerticalAlignment="Center" Text="Multiline textbox" />
        <PasswordBox VerticalAlignment="Center"  Password="HelloWorld" />
    </StackPanel>
</Window>
```



TextBox 커서 글자 맨 끝으로 스크롤

```
public static void tbox_cursor_TextChanged(object sender,
TextChangedEventArgs e)
{
    TextBox tbox = sender as TextBox;
    if (tbox == null)
        return;

    var rect = tbox.GetRectFromCharacterIndex(tbox.Text.Length);
    tbox.ScrollToHorizontalOffset(rect.Right);
}
```

소수만 입력 받는 **Textbox**

```
    public class NumberTextbox : TextBox
    {
        protected override void OnPreviewTextInput(TextCompositionEventArgs
e)
        {
            if (((e.Source as TextBox).Text.Contains(".") &&
e.Text.Equals(".")) || (!e.Text.Equals(".") &&
!Char.IsDigit(e.Text.ToCharArray()[0])))
                e.Handled = true;
            else
                e.Handled = !AreAllValidNumericChars(e.Text);

            base.OnPreviewTextInput(e);
        }

        bool AreAllValidNumericChars(string str)
        {
            bool ret = true;

            if (str ==
System.Globalization.NumberFormatInfo.CurrentInfo.CurrencyDecimalSeparator |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.CurrencyGroupSeparator |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.CurrencySymbol |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.NegativeSign |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.NegativeInfinitySymbol |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.NumberDecimalSeparator |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.NumberGroupSeparator |
                str ==
```

```
System.Globalization.NumberFormatInfo.CurrentInfo.PercentDecimalSeparator |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.PercentGroupSeparator |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.PercentSymbol |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.PerMilleSymbol |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.PositiveInfinitySymbol |
                str ==
System.Globalization.NumberFormatInfo.CurrentInfo.PositiveSign)
                return ret;

            for (int i = 0; i < str.Length; i++)
            {
                char ch = str[i];
                ret &= Char.IsDigit(ch);
            }

            return ret;
        }
    }
```

```
<Window ...>
    ...
        <local:NumberTextbox x:Name="tbox_age" Grid.Row="1" Grid.Column="1"
VerticalContentAlignment="Center" />
    ...
</Window>
```

ToolTip

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="100" Width="200">
    <StackPanel>
        <TextBox Width="100">
            <TextBox.ToolTip>
                <ToolTip Content="여기에 입력하세요." />
            </TextBox.ToolTip>
        </TextBox>

        <!-- Customize -->
        <TextBox Width="100">
```

```
            <TextBox.ToolTip>
                <TextBlock FontSize="25">
                    <Ellipse Fill="Orange" Width="20" Height="20" />
                    Plain text is <Italic>so</Italic>
                    <Span FontFamily="Old English Text MT">last
century</Span>
                    <Ellipse Fill="Orange" Width="20" Height="20" />
                </TextBlock>
            </TextBox.ToolTip>
        </TextBox>
    </StackPanel>
</Window>
```



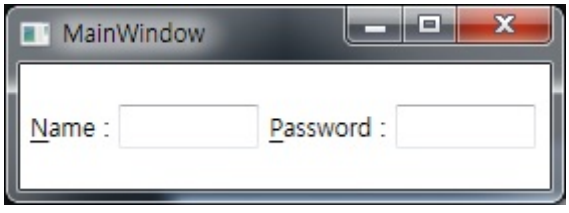
Label

- 단순한 Text 출력 표시용

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="100" Width="280">

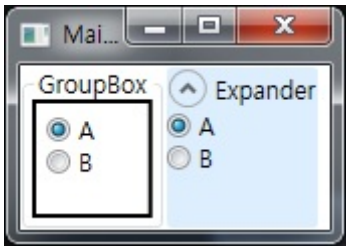
    <StackPanel Orientation="Horizontal">
        <Label Target="{Binding ElementName=nameText}"
VerticalAlignment="Center">_Name :</Label>
        <TextBox x:Name="nameText" Width="70"  VerticalAlignment="Center"/>

        <Label Target="{Binding ElementName=pwText}"
VerticalAlignment="Center">_Password :</Label>
        <TextBox x:Name="pwText" Width="70"  VerticalAlignment="Center"/>
    </StackPanel>
</Window>
```



GroupBox and Expander

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="120" Width="170">
    <StackPanel Orientation="Horizontal">
        <GroupBox Header="GroupBox">
            <Border BorderBrush="Black" BorderThickness="2" Padding="5">
                <StackPanel>
                    <RadioButton Content="A" IsChecked="True" />
                    <RadioButton Content="B" />
                </StackPanel>
            </Border>
        </GroupBox>
        <Expander Header="Expander" Background="#def" IsExpanded="True">
            <Border>
                <StackPanel>
                    <RadioButton Content="A" IsChecked="True" />
                    <RadioButton Content="B" />
                </StackPanel>
            </Border>
        </Expander>
    </StackPanel>
</Window>
```



List Controls

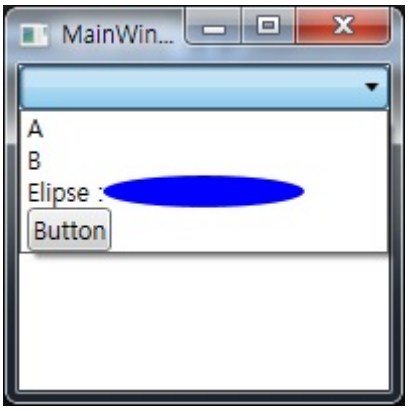
ComboBox

클릭하면 스크롤하여 목록 표시. (= Android의 Spinner)

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="200" Width="200">

    <StackPanel>
```

```
        <ComboBox IsDropDownOpen="True"> <!-- 자동 펼침 -->
            <ComboBoxItem>A</ComboBoxItem>
            <ComboBoxItem>B</ComboBoxItem>
            <!-- ComboBoxItem외의 Control 삽입도 가능!-->
            <StackPanel Orientation="Horizontal">
                <TextBlock>Elipse : </TextBlock>
                <Ellipse Fill="Blue" Width="100" />
            </StackPanel>
            <Button>Button</Button>
        </ComboBox>
    </StackPanel>
</Window>
```



ComboBoxItem의 "Content"로 선택 아이템 초기화

```
<Window x:Class="WpfApplication1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <!-- SelectedValuePath="Content"가 핵심!-->
        <ComboBox Name="cbox"  Width="100"  Height="50"
SelectedValuePath="Content">
            <ComboBoxItem Content="2 번" />
            <ComboBoxItem Content="4 번" />
        </ComboBox>
    </Grid>
</Window>
```

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

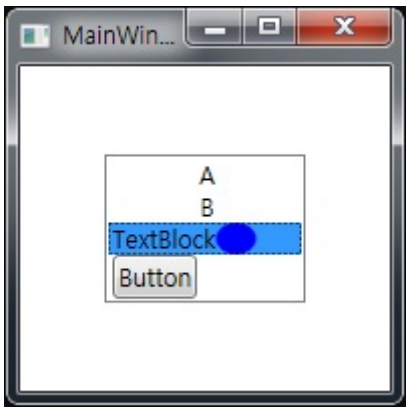
```
namespace WpfApplication1
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();

            cbox.SelectedValue = 4 + " 번";
        }
    }
}
```

ListBox

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="200" Width="200">
    <Grid VerticalAlignment="Center" HorizontalAlignment="Center">
        <ListBox Width="100">
            <ListBox.Items>
                <ListBoxItem
HorizontalContentAlignment="Center">A</ListBoxItem>
                <ListBoxItem
HorizontalContentAlignment="Center">B</ListBoxItem>

                <!-- ComboBoxItem외의 Control 삽입도 가능! -->
                <StackPanel Orientation="Horizontal">
                    <TextBlock>TextBlock</TextBlock>
                    <Ellipse Fill="Blue" Width="20" />
                </StackPanel>
                <Button>Button</Button>
            </ListBox.Items>
        </ListBox>
    </Grid>
</Window>
```



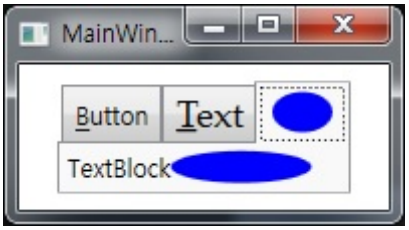
TabControl

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="110" Width="200">
    <Grid VerticalAlignment="Center" HorizontalAlignment="Center">
        <TabControl>
            <TabItem Header="_Button">
                <Button>Click!</Button>
            </TabItem>

            <TabItem>
                <TabItem.Header>
                    <TextBlock FontSize="18" FontFamily="Palatino Linotype">
                        <AccessText>_Text</AccessText>
                    </TextBlock>
                </TabItem.Header>

                <TextBlock>Hello, World!</TextBlock>
            </TabItem>

            <TabItem>
                <TabItem.Header>
                    <Ellipse Fill="Blue" Width="30" Height="20" />
                </TabItem.Header>
                <StackPanel Orientation="Horizontal">
                    <TextBlock>TextBlock</TextBlock>
                    <Ellipse Fill="Blue" Width="70" />
                </StackPanel>
            </TabItem>
        </TabControl>
    </Grid>
</Window>
```

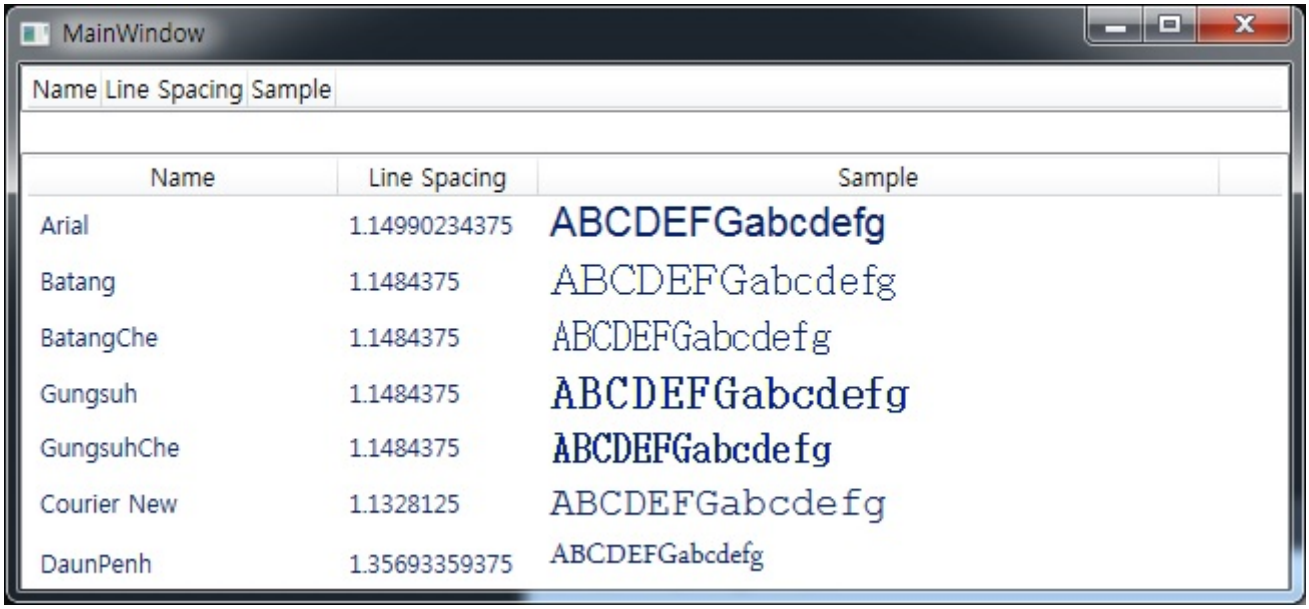


ListView

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="650" Height="300">

    <StackPanel>
   <!-- 기본형태 -->
        <ListView>
            <ListView.View>
                <GridView AllowsColumnReorder="True">
                    <GridViewColumn Header="Name" />
                    <GridViewColumn Header="Line Spacing"/>
                    <GridViewColumn Header="Sample" />
                </GridView>
            </ListView.View>
        </ListView>


        <!-- StackPanel 밖에 있으면 자동 스크롤 가능.-->
        <ListView ItemsSource="{x:Static Fonts.SystemFontFamilies}"
Margin="0,20,0,0">
            <ListView.View>
                <GridView AllowsColumnReorder="True">
                    <GridViewColumn Header="Name"
DisplayMemberBinding="{Binding Source}"/>
                    <GridViewColumn Header="Line Spacing"
DisplayMemberBinding="{Binding LineSpacing}" />
                    <GridViewColumn Header="Sample">
                        <GridViewColumn.CellTemplate>
                            <DataTemplate>
                                <TextBlock FontFamily="{Binding}"
FontSize="20" Text="ABCDEFGabcdefg" />
                            </DataTemplate>
                        </GridViewColumn.CellTemplate>
                    </GridViewColumn>
                </GridView>
            </ListView.View>
        </ListView>
    </StackPanel>
</Window>
```




```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="200">


    <Grid HorizontalAlignment="Center" VerticalAlignment="Center">
        <Grid.Resources>
            <XmlDataProvider x:Key="src" XPath="/Root">
                <x:XData>
                    <Root xmlns="">
                        <Item id="One" flag="True" value="A" />
                        <Item id="Two" flag="True" value="B" />
                        <Item id="Three" flag="False" value="C" />
                        <Item id="Four" flag="True" value="D" />
                    </Root>
                </x:XData>
            </XmlDataProvider>
        </Grid.Resources>

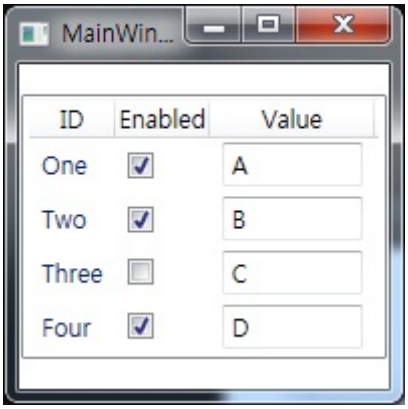
        <ListView DataContext="{StaticResource src}" ItemsSource="{Binding
XPath=Item}">
            <ListView.View>
                <GridView>
                    <GridViewColumn Header="ID"
DisplayMemberBinding="{Binding XPath=@id}" />

                    <GridViewColumn Header="Enabled">
                        <GridViewColumn.CellTemplate>
                            <DataTemplate>
```

```
                                <CheckBox IsChecked="{Binding XPath=@flag}"
/>
                            </DataTemplate>
                        </GridViewColumn.CellTemplate>
                    </GridViewColumn>

                    <GridViewColumn Header="Value">
                        <GridViewColumn.CellTemplate>
                            <DataTemplate>
                                <TextBox Text="{Binding XPath=@value}"
Width="70" />
                            </DataTemplate>
                        </GridViewColumn.CellTemplate>
                    </GridViewColumn>

                </GridView>
            </ListView.View>
        </ListView>
    </Grid>
</Window>
```

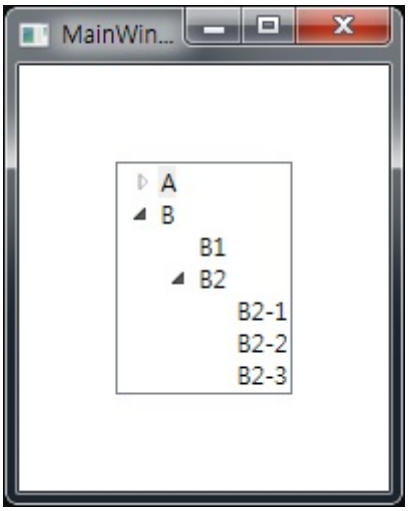


TreeView

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="250">


    <Grid HorizontalAlignment="Center" VerticalAlignment="Center">
        <TreeView>
            <TreeViewItem Header="A" IsSelected="True">
                <TreeViewItem Header="A1" />
                <TreeViewItem Header="A2" />
                <TreeViewItem Header="A3" />
            </TreeViewItem>
```

```
            <TreeViewItem Header="B" IsExpanded="True">
                <TreeViewItem Header="B1" />
                <TreeViewItem Header="B2" IsExpanded="True">
                    <TreeViewItem Header="B2-1" />
                    <TreeViewItem Header="B2-2" />
                    <TreeViewItem Header="B2-3" />
                </TreeViewItem>
            </TreeViewItem>
        </TreeView>
    </Grid>
</Window>
```



Menus

Menu

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="150">

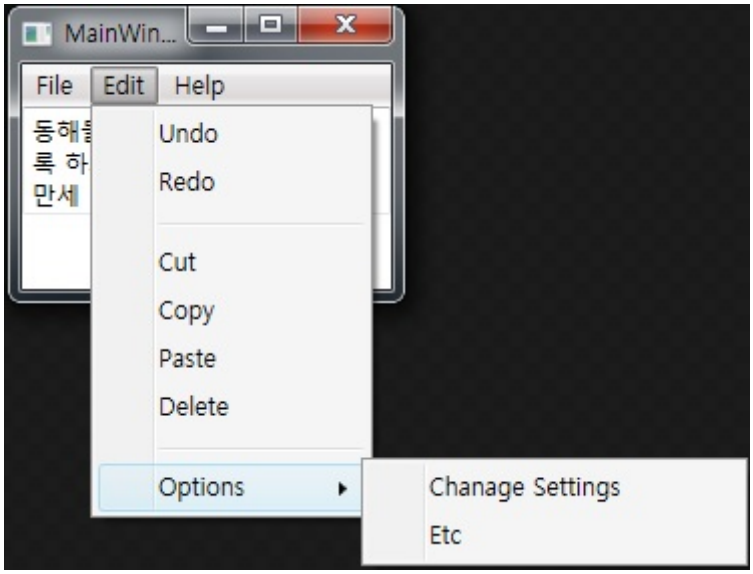

    <Grid VerticalAlignment="Top">

        <Grid.RowDefinitions>
            <RowDefinition Height="*" />
            <RowDefinition Height="*" />
        </Grid.RowDefinitions>
        <Menu>
            <MenuItem Header="_File">
                <MenuItem Header="_New" />
                <MenuItem Header="_Open" />
```

```
                <MenuItem Header="_Save" />
                <MenuItem Header="Sa_ve As..." />
                <Separator />
                <MenuItem Header="E_xit" />
            </MenuItem>
            <MenuItem Header="_Edit">
                <MenuItem Header="_Undo" />
                <MenuItem Header="_Redo" />
                <Separator />

                <MenuItem Header="Cu_t" />
                <MenuItem Header="_Copy" />
                <MenuItem Header="_Paste" />
                <MenuItem Header="_Delete" />
                <Separator />
                <MenuItem Header="_Options" >
                    <MenuItem Header="Chanage Settings" />
                    <MenuItem Header="Etc" />
                </MenuItem>
            </MenuItem>

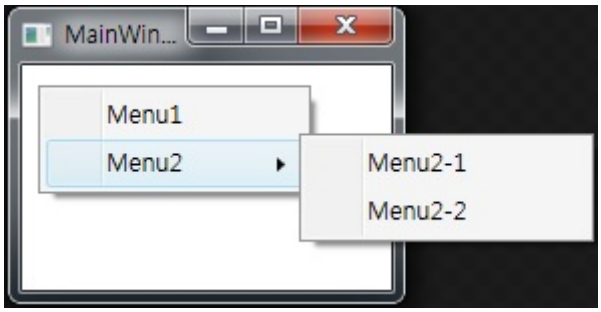
            <MenuItem Header="_Help">
                <MenuItem Header="_About..." />
            </MenuItem>
        </Menu>
        <TextBox Name="textBox1" Grid.Row="1" TextWrapping="Wrap">
    동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세
        </TextBox>
    </Grid>
</Window>
```



ContextMenu

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="200" Height="150">

    <Grid Background="Transparent">
        <Grid.ContextMenu>
            <ContextMenu>
                <MenuItem Header="Menu1" />
                <MenuItem Header="Menu2" >
                    <MenuItem Header="Menu2-1" />
                    <MenuItem Header="Menu2-2" />
                </MenuItem>
            </ContextMenu>
        </Grid.ContextMenu>
    </Grid>
</Window>
```



Toolbar

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="350" Height="100">
    <ToolBarTray>
        <ToolBar>
            <Button>Button1-1</Button>
            <Button>Buttton1-2</Button>
        </ToolBar>

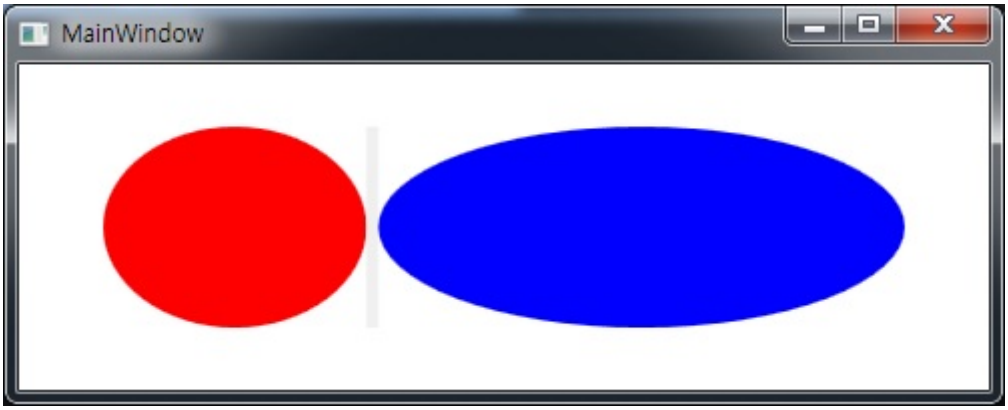
        <ToolBar>
            <Button>Button2-1</Button>
            <Button>Button2-2</Button>
        </ToolBar>
    </ToolBarTray>
</Window>
```



GridSplitter

Grid의 행/열 재조정.

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Width="500" Height="200">
    <Grid Height="100" Width="400">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="1*" />
            <ColumnDefinition Width="6" />
            <ColumnDefinition Width="2*" />
        </Grid.ColumnDefinitions>
        <Ellipse Grid.Column="0" Fill="Red" />
        <GridSplitter Grid.Column="1" HorizontalAlignment="Stretch" />
        <Ellipse Grid.Column="2" Fill="Blue" />
    </Grid>
</Window>
```



Calendar & DatePicker

http://www.dotnetperls.com/calendar http://www.dotnetperls.com/datepicker

---

Input

Routed Events

- Event가 어떻게 제어되는 지 살펴보자.

```
<Window x:Class="Control1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="70" Width="150">
    <StackPanel>
        <Button PreviewMouseDown="Button_PreviewMouseDown"
MouseDown="Button_MouseDown">
            <Grid PreviewMouseDown="Grid_PreviewMouseDown"
MouseDown="Grid_MouseDown">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition />
                    <ColumnDefinition />
                </Grid.ColumnDefinitions>
                <Canvas PreviewMouseDown="Canvas_PreviewMouseDown"
MouseDown="Canvas_MouseDown"
                        Width="20" Height="18" VerticalAlignment="Center">
                    <Ellipse PreviewMouseDown="Ellipse_PreviewMouseDown"
MouseDown="Ellipse_MouseDown"
                             x:Name="myEllipse" Canvas.Left="1"
Canvas.Top="1" Width="16" Height="16" Fill="Yellow" Stroke="Black" />

                    <Ellipse Canvas.Left="4.5" Canvas.Top="5" Width="2.5"
Height="3" Fill="Black" />
                    <Ellipse Canvas.Left="11" Canvas.Top="5" Width="2.5"
Height="3" Fill="Black" />

                    <Path Data="M 5,10 A 3,3 0 0 0 13,10" Stroke="Black" />
                </Canvas>
                <TextBlock Grid.Column="1">Click!</TextBlock>
            </Grid>
        </Button>
    </StackPanel>
</Window>
```

```
using System;
using System.Windows;
using System.Windows.Input;
using System.Diagnostics;

namespace Control1
{
    public partial class MainWindow : Window
```

```
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Button_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            Debug.WriteLine("1-1 Button_PreviewMouseDown");
        }

        private void Button_MouseDown(object sender, MouseButtonEventArgs e)
        {
            Debug.WriteLine("1-2 Button_MouseDown");
        }

        private void Grid_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            Debug.WriteLine("2-1 Grid_PreviewMouseDown");
        }

        private void Grid_MouseDown(object sender, MouseButtonEventArgs e)
        {
            Debug.WriteLine("2-2 Grid_MouseDown");
        }

        private void Canvas_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            Debug.WriteLine("3-1 Canvas_PreviewMouseDown");
        }

        private void Canvas_MouseDown(object sender, MouseButtonEventArgs e)
        {
            Debug.WriteLine("3-2 Canvas_MouseDown");
        }

        private void Ellipse_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            Debug.WriteLine("4-1 Ellipse_PreviewMouseDown");
        }

        private void Ellipse_MouseDown(object sender, MouseButtonEventArgs
e)
        {
            Debug.WriteLine("4-2 Ellipse_MouseDown");
        }
    }
```

```
}
```



- 이렇게 단계적으로 호출되지 않고, **Button 자체에만** Event를 주려면

```
//중략
private void Button_PreviewMouseDown(object sender, MouseButtonEventArgs e)
{
    Debug.WriteLine("1-1 Button_PreviewMouseDown");
    e.Handled = true;
}
//중략
```

- 중복 호출을 방지하려면 "+=" 이용.

Mouse Input

- xaml은 그대로 이용.

```
using System;
using System.Windows;
using System.Windows.Input;
using System.Diagnostics;
using System.Windows.Shapes;

namespace Control1
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();

            myEllipse.MouseDown += myEllipse_MouseDown;
            myEllipse.MouseMove += myEllipse_MouseMove;
```

```
            myEllipse.MouseUp += myEllipse_MouseUp;
        }

        private void myEllipse_MouseDown(object sender, MouseButtonEventArgs
e)
        {
            Mouse.Capture(myEllipse);
        }

        private void myEllipse_MouseUp(object sender, MouseButtonEventArgs
e)
        {
            Mouse.Capture(null);
        }

        private void myEllipse_MouseMove(object sender, MouseEventArgs e)
        {
            Debug.WriteLine(Mouse.GetPosition(myEllipse));
        }



        private void Button_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            //Debug.WriteLine("1-1 Button_PreviewMouseDown");
            //e.Handled = true;
        }

        private void Button_MouseDown(object sender, MouseButtonEventArgs e)
        {
            //Debug.WriteLine("1-2 Button_MouseDown");
        }

        private void Grid_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            //Debug.WriteLine("2-1 Grid_PreviewMouseDown");
        }

        private void Grid_MouseDown(object sender, MouseButtonEventArgs e)
        {
            //Debug.WriteLine("2-2 Grid_MouseDown");
        }

        private void Canvas_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            //Debug.WriteLine("3-1 Canvas_PreviewMouseDown");
        }
```

```
        private void Canvas_MouseDown(object sender, MouseButtonEventArgs e)
        {
            //Debug.WriteLine("3-2 Canvas_MouseDown");
        }

        private void Ellipse_PreviewMouseDown(object sender,
MouseButtonEventArgs e)
        {
            //Debug.WriteLine("4-1 Ellipse_PreviewMouseDown");
        }

        private void Ellipse_MouseDown(object sender, MouseButtonEventArgs
e)
        {
            //Debug.WriteLine("4-2 Ellipse_MouseDown");
        }
    }
}
```

- 결과적으로 Ellipse 위에 올릴 때, 움직일 때, 클릭할 때 좌표값을 추적한다.

Keyboard Input

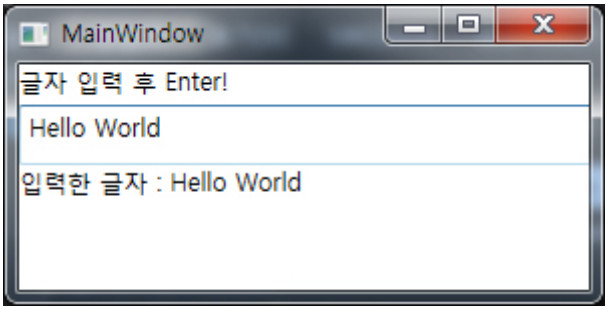
```
<Window x:Class="Input1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="150" Width="300">
    <StackPanel>
        <TextBlock Width="300" Height="20">글자 입력 후 Enter!</TextBlock>
        <TextBox Width="300" Height="30" Name="textBox1"
KeyDown="textBox1_KeyDown"/>
        <TextBlock Width="300" Height="100" Name="textBlock1"/>
    </StackPanel>
</Window>
```

```
using System;
using System.Windows;
using System.Windows.Input;

namespace Input1
{
    public partial class MainWindow : Window
    {
        public MainWindow()
```

```
        {
            InitializeComponent();
        }

        private void textBox1_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.Key == Key.Return)
            {
                textBlock1.Text = "입력한 글자 : " + textBox1.Text;
            }
        }
    }
}
```



Ink Input

- 터치 스크린에서의 입력.

Commands

- Ctrl+X 같은 특수 명령.

Simple Data Binding

Without Data Binding

- getter/setter 클래스를 따로 구현해야 한다.

```
<Window x:Class="SimpleDataBinding.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="WithoutBinding" Height="100" Width="200">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition />
```

```
            <RowDefinition />
            <RowDefinition />
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="1*"/>
            <ColumnDefinition Width="3*"/>
        </Grid.ColumnDefinitions>
        <TextBlock Grid.Row="0" Grid.Column="0"> Name:</TextBlock>
        <TextBox Name="tb_name" Grid.Row="0" Grid.Column="1" />

        <TextBlock Grid.Row="1" Grid.Column="0">Age:</TextBlock>
        <TextBox Name="tb_age" Grid.Row="1" Grid.Column="1" />
        <Button Name="btn_birthday" Grid.Row="2"
Grid.Column="1">Birthday</Button>
    </Grid>
</Window>
```

```
//새로 생성한 getter/setter Class

using System;

namespace SimpleDataBinding
{
    class Person
    {
        string name;
        int age;

        public Person(string name, int age)
        {
            this.name = name;
            this.age = age;
        }

        public string Name
        {
            get {return this.name;}
            set {this.name = value;}
        }

        public int Age
        {
            get {return this.age;}
            set {this.age = value;}
        }
    }
}
```

```
using System.Windows;

namespace SimpleDataBinding
{
    public partial class MainWindow : Window
    {
        Person person = new Person("Tom", 11);

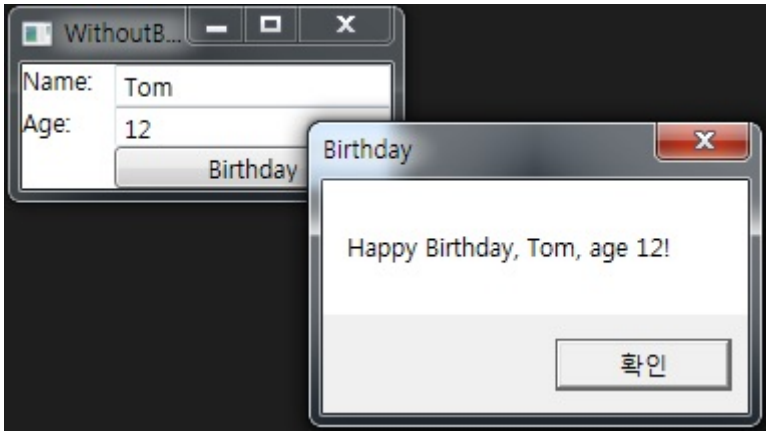
        public MainWindow()
        {
            InitializeComponent();

            this.tb_name.Text = person.Name;
            this.tb_age.Text = person.Age.ToString();

            this.btn_birthday.Click += btn_birthday_Click;
        }

        private void btn_birthday_Click(object sender, RoutedEventArgs e)
        {
            ++person.Age;
            this.tb_age.Text = person.Age.ToString();

            MessageBox.Show(string.Format("Happy Birthday, {0}, age {1}!",
person.Name, person.Age), "Birthday");
        }
    }
}
```



Data Binding

Debuging Data Binding

Binding to List Data

(임시) ComboBox Data Binding

```
<Window x:Class="WpfApplication3.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:local="clr-namespace:WpfApplication3"
        Title="MainWindow" Height="110" Width="200"
        Loaded="Window_Loaded">

    <StackPanel>

        <ComboBox Name="cbox1" />

        <ComboBox Name="cbox2">
            <ComboBox.ItemTemplate>
                <DataTemplate>
                    <StackPanel Orientation="Horizontal">
                        <TextBlock Text="{Binding Age}" />
                        <TextBlock Text="{Binding Name}" />
                    </StackPanel>
                </DataTemplate>
            </ComboBox.ItemTemplate>
        </ComboBox>

        <ComboBox Name="cbox3">
            <ComboBox.ItemTemplate>
                <DataTemplate>
                    <StackPanel Orientation="Horizontal">
                        <Rectangle Fill="{Binding Name}" Width="16"
Height="16" Margin="0,2,5,2" />
                        <TextBlock Text="{Binding Name}" />
                    </StackPanel>
                </DataTemplate>
            </ComboBox.ItemTemplate>
        </ComboBox>

    </StackPanel>
</Window>
```

```
using System;
```

```
using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media;

namespace WpfApplication3
{
    public partial class MainWindow : Window
    {
        List<string> myList1 = new List<string>();
        List<Person> myList2 = new List<Person>();

        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
    /*단순 String Binding */
            myList1.Add("String1");
            myList1.Add("String2");
            myList1.Add("String3");
            cbox1.ItemsSource = myList1;
            cbox1.SelectedIndex = 0;


    /*복합 Data Binding 1 */
            myList2.Add(new Person(1, "a", "OK"));
            myList2.Add(new Person(2, "b", "Hello"));
            myList2.Add(new Person(3, "c", "Hi"));
            myList2.Add(new Person(4, "d", "hehehe"));
            myList2.Add(new Person(5, "e", "hohoho"));
            cbox2.ItemsSource = myList2;

            myList2.Add(new Person(77, "XX", "Wooooh")); //추가 삽입
            cbox2.ItemsSource = null;
            cbox2.ItemsSource = myList2;
            cbox2.SelectedIndex = 0; //Default Index 0 지정.
            cbox2.SelectionChanged += cbox2_SelectionChanged; //Index 변경 후
Event 걸기


    /*복합 Data Binding 2 */
            cbox3.ItemsSource = typeof(Colors).GetProperties();
            cbox3.SelectedIndex = 0;
        }

        private void cbox2_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
```

```
            if (!IsLoaded)
                return;

            if (cbox2.SelectedIndex < 0)
                return;

            MessageBox.Show(myList2[cbox2.SelectedIndex].Message);
        }

    }


    public class Person
    {
        public int Age { get; set; }
        public string Name { get; set; }

        public string Message { get; set; }

        public Person(int age, string name, string message)
        {
            this.Age = age;
            this.Name = name;
            this.Message = message;
        }
    }
}
```

Binding to List Data

Data Source Providers

Master-Detail Binding

Hierarchical Binding

Styles

Without Styles

Inline Styles

Named Styles

Element-Typed Styles

Data Templates and Styles

Triggers

Control Templates

Beyound Styles

Logical and Visual Trees

Data-Driven UI

Windows and Dialogs

Window

Dialogs

Application

Windows

- 최상위 화면 요소. (WinAPI의 HWND, MFC의 CWnd, WinForms의 Form과 같은 개념)

Life Cycles (생명/수명 주기)

- 생명주기 중 가장 많이 사용되는 Event
  1. Loaded : Window가 나타나기 직전
  2. Closing : Window가 닫히기 직전
  3. Closed : Window가 닫혔을 때

```
<Window x:Class="Windows1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525"
        Loaded="Window_Loaded"
        Closing="Window_Closing"
        Closed="Window_Closed">
</Window>
```

```
using System;
using System.Windows;

namespace Windows1
{

    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            Title = "Hello World! " + DateTime.Now;
        }

        private void Window_Closing(object sender,
System.ComponentModel.CancelEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("닫겠습니까?", "TITLE",
MessageBoxButton.YesNo);

            if (result == MessageBoxResult.No)
                e.Cancel = true;
        }

        private void Window_Closed(object sender, EventArgs e)
        {
            MessageBox.Show("닫혔습니다.");
        }
    }
}
```

Windows 표시

- Modaless : 다른 Window와의 상호 작용 가능.

    - Show() Method
    - Visibility 속성(Property)을 “visible”
- Modal : 다른 Window와의 상호 작용 **불가**.
    - ShowDialog() Method

```
<Window x:Class="Windows2.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">

        <StackPanel>
            <Button Click="ShowMethod">ShowMethod</Button>
            <Button
Click="UseVisibilityProperty">UseVisibilityProperty</Button>
            <Button Click="ShowDialogMethod">ShowDialogMethod</Button>
        </StackPanel>
</Window>
```

```
using System.Windows;

namespace Windows2
using System.Windows;

namespace Windows2
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void ShowMethod(object sender, RoutedEventArgs e)
        {
            Window w = new Window();
            w.Owner = this;
            w.Show();
        }

        private void UseVisibilityProperty(object sender, RoutedEventArgs e)
        {
            Window w = new Window();
            w.Visibility = Visibility.Visible;
            w.ShowInTaskbar = false;
            w.Show();
        }
```

```
        private void ShowDialogMethod(object sender, RoutedEventArgs e)
        {
            Window w = new Window();
            w.Owner = this; // 소유권 지정. 가장 맨 앞 출력 + 최소화시 같이 최소화
            w.SizeToContent = SizeToContent.WidthAndHeight; // 내용물에 맞게 창
크기 조정
            w.WindowStartupLocation = WindowStartupLocation.CenterOwner; //
창 초기 위치 지정. 소유자 중앙.
            w.ShowInTaskbar = false; //작업 표시줄 표시X
            w.ShowDialog();
        }
    }
}
```

From:
https://wiki.ledyx.xyz/ - **Ledyx WIKI**

Permanent link:
https://wiki.ledyx.xyz/doku.php?id=desktop:windows_presentation_foundation

Last update: **2021/02/07 03:55**