목차

SECTION 1.

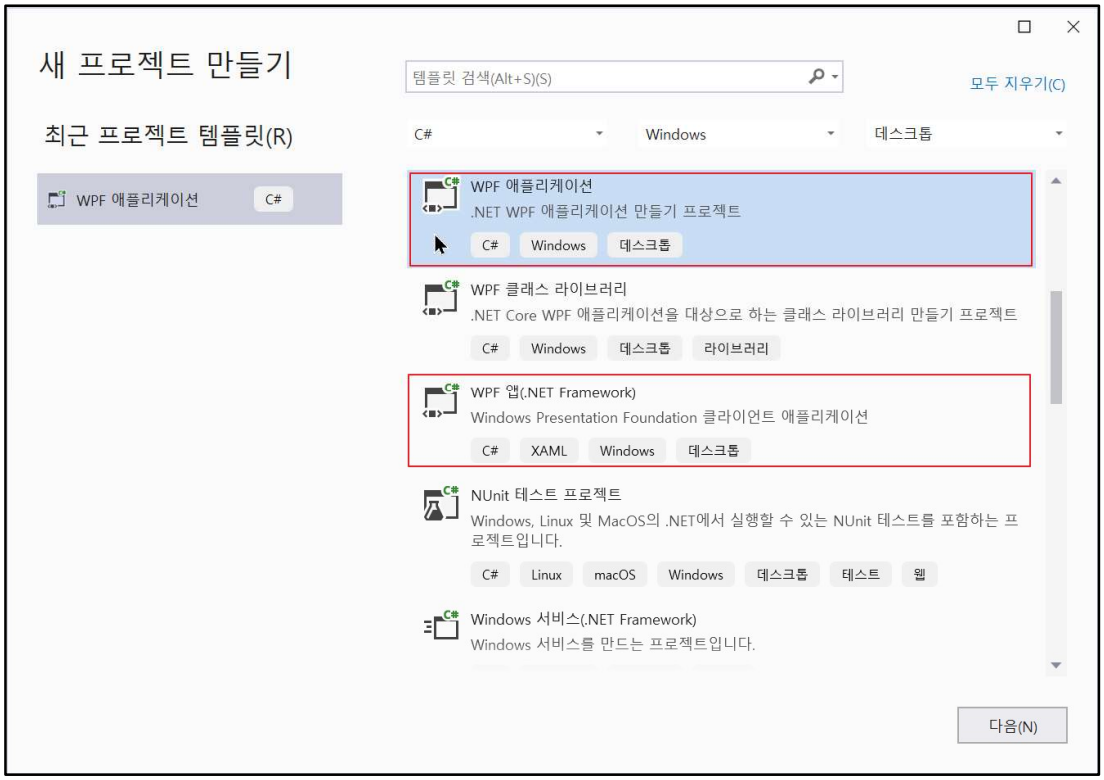
WPF Program 구조

❑ 주요 학습 내용

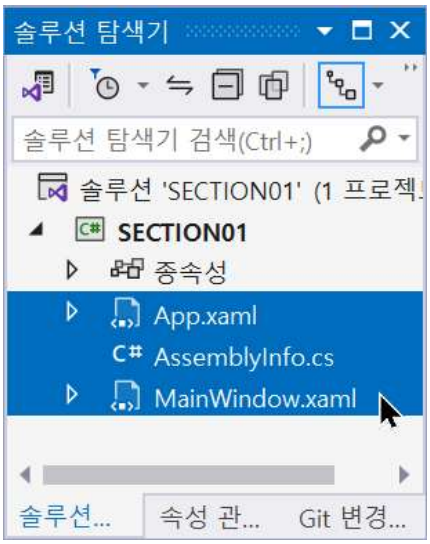
> Create Project
>
> Window
>
> Application
>
> Event handling
>
> Grid Layout
>
> Contents Property
>
> XAML

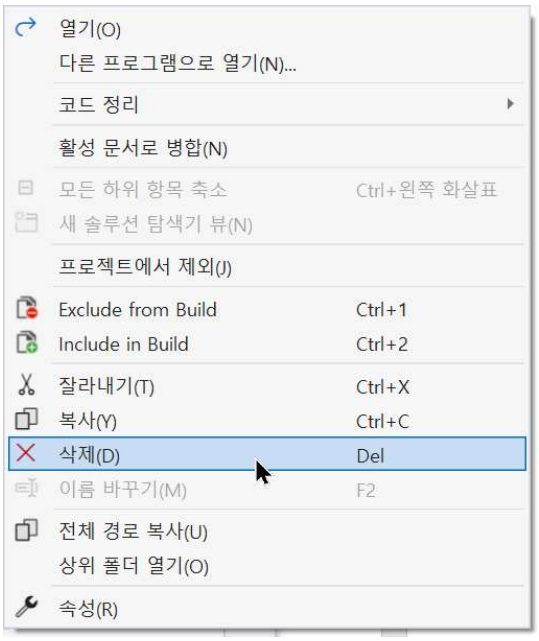
WPF Project

Create Project



Remove Auto generated Files





Project Files(.csproj) - Console App vs WPF App

Console App .csproj

```
<Project Sdk="Microsoft.NET.Sdk">
    <PropertyGroup>
        <OutputType>Exe</OutputType>
        <TargetFramework>net8.0</TargetFramework>
        <ImplicitUsings>enable</ImplicitUsings>
        <Nullable>enable</Nullable>
    </PropertyGroup>
</Project>
```

WPF App .csproj

```
<Project Sdk="Microsoft.NET.Sdk">
    <PropertyGroup>
        <OutputType>WinExe</OutputType>
        <TargetFramework>net8.0-windows</TargetFramework>
        <Nullable>enable</Nullable>
        <ImplicitUsings>enable</ImplicitUsings>
        <UseWPF>true</UseWPF>
    </PropertyGroup>
</Project>
```

Hello, WPF

MessageBox.Show()

```
using System;
using System.Windows;

class Program
{
    public static void Main()
    {
        MessageBox.Show("Hello, WPF");
    }
}
```

Make Window

System.Windows

```
using System;
using System.Windows;

class Program
{
    [STAThread]
    public static void Main()
    {
        Window win = new Window();
        win.Show();
        MessageBox.Show("Hello, WPF");
    }
}
```

Application Object

Event Loop

```
using System;
using System.Windows;

class Program
{
    [STAThread]
    public static void Main()
    {
        Application app = new Application();
        Window win = new Window();

        // win.Show();
        // app.MainWindow = win;

        app.Run(win);
    }
}
```

Window’s properties

Window’s Properties

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

class Program
{
    [STAThread]
    public static void Main()
    {
        Application app = new Application();

        Window win = new Window();

        win.Width   = 400;
        win.Height  = 400;
        win.Title   = "EX12";
        win.Topmost = true;
        win.Background = new SolidColorBrush(Colors.Green);

        app.Run(win);
    }
}
```

Windows Event Handling #1

Override virtual function

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

public class MainWindow : System.Windows.Window
{
    protected override void OnMouseDown(MouseButtonEventArgs e)
    {
        base.OnMouseDown(e);
        Point pt = e.GetPosition(this);
        this.Title = $"{pt.X}, {pt.Y}";
    }
}
class Program
{
    [STAThread]
    public static void Main()
    {
        Application app = new Application();
        MainWindow win = new MainWindow();
        app.Run(win);
    }
}
```

Windows Event Handling #2

Register Event callback method

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

public class MainWindow : System.Windows.Window
{
}

class Program
{
    [STAThread]
    public static void Main()
    {
        Application app = new Application();
        MainWindow win = new MainWindow();
        win.MouseDown += Win_MouseDown;
        app.Run(win);
    }
    private static void Win_MouseDown(object sender,
                            MouseButtonEventArgs e)
    {
        Window win = (Window)sender;
        Point pt = e.GetPosition(win);
        win.Title = $"{pt.X}, {pt.Y}";
    }
}
```

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

public class MainWindow : System.Windows.Window
{
    public MainWindow()
    {
        this.MouseDown += MainWindow_MouseDown;
    }
    private void MainWindow_MouseDown(object sender,
                                      MouseButtonEventArgs e)
    {
        Window win = (Window)sender;
        Point pt = e.GetPosition(this);
        win.Title = $"{pt.X}, {pt.Y}";
    }
}
class Program
{
    [STAThread]
    public static void Main()
    {
        Application app = new Application();
        MainWindow win = new MainWindow();
        app.Run(win);
    }
}
```

Application Event Handling #1

Override virtual function

```
using System;
using System.Windows;

public class MainWindow : System.Windows.Window
{
}
public class App : System.Windows.Application
{
    protected override void OnStartup(StartupEventArgs e)
    {
        base.OnStartup(e);  Console.WriteLine("OnStartup");
    }
    protected override void OnExit(ExitEventArgs e)
    {
        base.OnExit(e);     Console.WriteLine("OnExit");
    }
}
class Program
{
    [STAThread]
    public static void Main()
    {
        App app = new App();
        MainWindow win = new MainWindow();
        app.Run(win);
    }
}
```

Application Event Handling #2

Register Event callback method

```
using System;
using System.Windows;

public class MainWindow : System.Windows.Window
{
}
public class App : System.Windows.Application
{
    public App() { this.Startup += App_Startup; }

    private void App_Startup(object sender, StartupEventArgs e)
    {
        Console.WriteLine("App_Startup");
    }
}
class Program
{
    [STAThread]
    public static void Main()
    {
       App app = new App();
       MainWindow win = new MainWindow();
       app.Run(win);
    }
}
```

WPF Program 기본 구조

WPF 프로그램의 기본 구조

```
using System;
using System.Windows;

public class MainWindow : System.Windows.Window
{
    public MainWindow()
    {
    }
}

public class App : System.Windows.Application
{
    public App()
    {
    }
    [STAThread]
    public static void Main()
    {
        App app = new App();
        MainWindow win = new MainWindow();
        app.Run(win);
    }
}
```

Get Application Reference

Application.Current

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

public class MainWindow : System.Windows.Window
{
    protected override void OnMouseDown(MouseButtonEventArgs e)
    {
        base.OnMouseDown(e);

        //Application.Current.Shutdown();
        ((App)Application.Current).Fn();
    }
}

public class App : System.Windows.Application
{
    public void Fn() { Console.WriteLine("Fn"); }

    [STAThread]
    public static void Main()
    {
        App app = new App();

        MainWindow win = new MainWindow();

        app.Run(win);
    }
}
```

Get MainWindow Reference

app.MainWindow property

```
// ex41.cs
using System;
using System.Windows;
using System.Windows.Input;

public class MainWindow : System.Windows.Window
{
    public void Fn()
    {
        Console.WriteLine("MainWindow Fn");
    }
}
public class App : System.Windows.Application
{
    protected override void OnStartup(StartupEventArgs e)
    {
        base.OnStartup(e);
        ((MainWindow)this.MainWindow).Fn();
    }
    protected override void OnExit(ExitEventArgs e)
    {
        base.OnExit(e);
        //((MainWindow)this.MainWindow).Fn();
        Console.WriteLine($"{this.MainWindow == null}");
    }
    [STAThread]
    public static void Main()
    {
        App app = new App();
        MainWindow win = new MainWindow();
        //app.MainWindow = win;
        app.Run(win);
    }
}
```

Contents Property #1

Text & button

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

public class MainWindow : System.Windows.Window
{
    public MainWindow()
    {
        //    this.Content = "Hello";

        Button btn = new Button();

        btn.Content = "OK";

        this.Content = btn;
    }
}

public class App : System.Windows.Application
{
    [STAThread]
    public static void Main()
    {
        App app = new App();

        MainWindow win = new MainWindow();

        app.Run(win);
    }
}
```

Contents Property #2

bitmap

```
using System;
using System.Runtime.InteropServices;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Media.Imaging;

public class MainWindow : System.Windows.Window
{
    public MainWindow()
    {
        BitmapImage bitmap = new BitmapImage();
        bitmap.BeginInit();
        bitmap.UriSource = new Uri("D:\\totoro.jpg");
        bitmap.EndInit();

        Image img = new Image();
        img.Source = bitmap;
        img.Stretch = System.Windows.Media.Stretch.Fill;
        this.Content = img;
    }
}
public class App : System.Windows.Application
{
    [STAThread]
    public static void Main()
    {
        App app = new App();
        MainWindow win = new MainWindow();
        app.Run(win);
    }
}
```

Grid Layout & Event Handling

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

public class MainWindow : System.Windows.Window
{
    private Grid grid = null;

    protected void InitializeComponent()
    {
        grid = new Grid();

        grid.RowDefinitions.Add(new RowDefinition());
        grid.RowDefinitions.Add(new RowDefinition());

        grid.ColumnDefinitions.Add(new ColumnDefinition());
        grid.ColumnDefinitions.Add(new ColumnDefinition());

        this.Content = grid;

        Button btn1 = new Button();
        Button btn2 = new Button();
        Button btn3 = new Button();
        Button btn4 = new Button();
        btn1.Content = "확인 1";
        btn2.Content = "확인 2";
        btn3.Content = "확인 3";
        btn4.Content = "확인 4";

        Grid.SetRow(btn1, 0); Grid.SetColumn(btn1, 0);
        Grid.SetRow(btn2, 0); Grid.SetColumn(btn2, 1);
        Grid.SetRow(btn3, 1); Grid.SetColumn(btn3, 0);
        Grid.SetRow(btn4, 1); Grid.SetColumn(btn4, 1);

        grid.Children.Add(btn1);
        grid.Children.Add(btn2);
        grid.Children.Add(btn3);
        grid.Children.Add(btn4);

        btn1.Click += Btn1_Click;
        btn2.Click += Btn2_Click;
        btn3.Click += Btn3_Click;
        btn4.Click += Btn3_Click;
    }
```

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

    }

    private void Btn3_Click(object sender, RoutedEventArgs e)
    {
        Button btn = sender as Button;
        string s = btn.Content as string;

        Console.WriteLine($"{s} Click");
    }

    private void Btn2_Click(object sender, RoutedEventArgs e)
    {
        Console.WriteLine("확인 2 Click");
    }

    private void Btn1_Click(object sender, RoutedEventArgs e)
    {
        Console.WriteLine("확인 1 Click");
    }
}


public class App : System.Windows.Application
{
    [STAThread]
    public static void Main()
    {
        App app = new App();

        MainWindow win = new MainWindow();

        app.Run(win);
    }
}
```

Using Xaml #1 – make button

```
<Grid xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <Button Name= "button1" >확인</Button>
</Grid>
```

```
public class MainWindow : System.Windows.Window
{
    protected void InitializeComponent()
    {
        Grid grid = null;

        using(FileStream fs = new FileStream("ex70.xaml", FileMode.Open))
        {
            grid = (Grid)XamlReader.Load(fs);
        }

        this.Content = grid;

        Button btn = (Button)grid.FindName("button1");

        btn.Click += Btn_Click;

    }

    private void Btn_Click(object sender, RoutedEventArgs e)
    {
        Console.WriteLine("button click");
    }

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

Using Xaml #2 – make Window

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">

</Window>
```

```
public class App : System.Windows.Application
{
    [STAThread]
    public static void Main()
    {
        App app = new App();

        Window win = null;

        using(FileStream fs = new FileStream("ex71.xaml", FileMode.Open))
        {
            win = (Window)XamlReader.Load(fs);
        }

        app.Run(win);
    }
}
```

Using Xaml #3 – make MainWindow

```
<local:MainWindow
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
                  xmlns:local="clr-namespace:;assembly=SECTION1"
          Title="ex72" Width="400" Height="400">

</local:MainWindow>
```

```
using System;
using System.IO;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Markup;

public class MainWindow : System.Windows.Window
{
    protected override void OnMouseDown(MouseButtonEventArgs e)
    {
        base.OnMouseDown(e);
        Console.WriteLine("MouseDown");
    }
}

public class App : System.Windows.Application
{
    [STAThread]
    public static void Main()
    {
        App app = new App();
        MainWindow win = null;
        using (FileStream fs = new FileStream("ex72.xaml", FileMode.Open))
        {
            win = (MainWindow)XamlReader.Load(fs);
        }
        app.Run(win);
    }
}
```

SECTION 2.

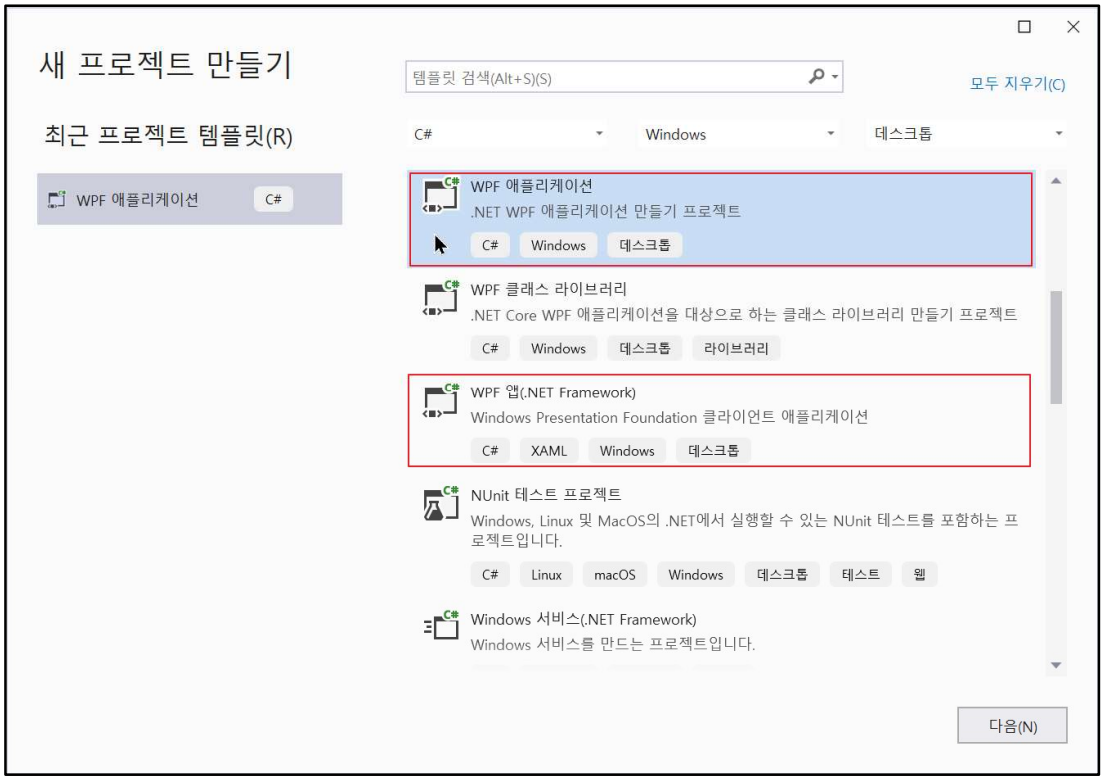
CODE + XAML

❑ 주요 학습 내용

> Auto Generated Files
>
> XAML
>
> XAML namespace
>
> Custom Markup
>
> Sliding Puzzle

Auto Generated Files

Create Project



Auto generated Files

Application	App.Xaml App.Xaml.cs App.g.i.cs
MainWindow	MainWindow.Xaml MainWindow.Xaml.cs MainWindow.g.i.cs

Window Style & Event Handling

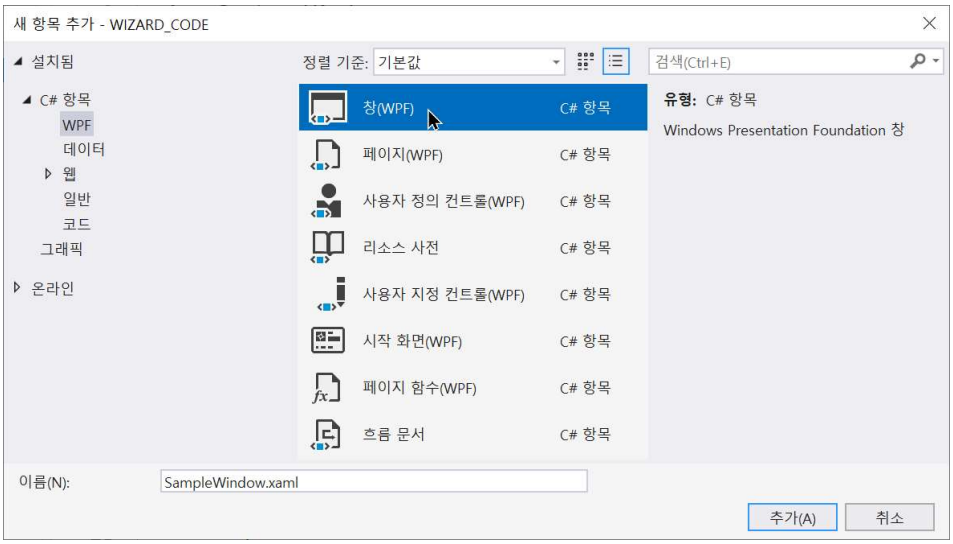
MainWindow.xaml

```
<Window x:Class="WIZARD_CODE.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:WIZARD_CODE"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800"
        Background="Blue" MouseDown="Window_MouseDown">
    <Grid>

    </Grid>
</Window>>
```

MainWindow.xaml.cs

```
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
    }
    private void Window_MouseDown(object sender, MouseButtonEventArgs e)
    {
        MessageBox.Show("Hello, WPF");
    }
}
```

Multiple Window



MainWindow.xaml.cs

```
private void Window_MouseDown(object sender, MouseButtonEventArgs e)
{
    SampleWindow sw = new SampleWindow();
    // sw.Show();
    sw.ShowDialog();
}
```

App.xaml StartupUri

App.xaml

```
<Application x:Class="WIZARD_CODE.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="clr-namespace:WIZARD_CODE"
    StartupUri="SampleWindow.xaml">
    <Application.Resources>
    </Application.Resources>
</Application>
```

Xaml #1

MainWindow.xaml

```
<Window x:Class="XAML.Ex1Window"
        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:XAML"
        mc:Ignorable="d"
        Name="win" Title="Hello">

    <Window.Width>300</Window.Width>
    <Window.Height>300</Window.Height>

</Window>
```

Xaml #2

MainWindow.xaml

```
<Window.Background>
    <LinearGradientBrush StartPoint="0,0.5" EndPoint="1,0.5">
        <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>

</Window.Background>
```

MainWindow.xaml.cs

```
public partial class Ex2Window : Window
{
    public Ex2Window()
    {
        InitializeComponent();

        LinearGradientBrush brush = new LinearGradientBrush();
        brush.StartPoint = new Point(0, 0);
        brush.EndPoint = new Point(1, 1);

        brush.GradientStops.Add(new GradientStop(Colors.Yellow, 0.0));
        brush.GradientStops.Add(new GradientStop(Colors.Red, 0.25));
        brush.GradientStops.Add(new GradientStop(Colors.Blue, 0.75));
        brush.GradientStops.Add(new GradientStop(Colors.LimeGreen, 1.0));

        //this.Background = brush;
    }
}
```

Xaml #3

MainWindow.xaml

```
<StackPanel>

        <Button Content="OK1"/>
        <Button Content="OK2"/>

</StackPanel>
```

Grid & Button in xaml

MainWindow.xaml

```
<Grid>
    <Button   x:Name="button1" Content="Button"
              HorizontalAlignment="Left"    Margin="148,129,0,0"
              VerticalAlignment="Top" Width="240" Height="46"
              Background="#FFCA4444" Click="button1_Click"/>
</Grid>
```

MainWindow.xaml.cs

```
private void button1_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("button click");
}
```

Xaml namespace

MainWindow.xaml

```
<ListBox>
    <ListBox.Items>
        <Button>ok1</Button>
        <Label>label</Label>
        <System:String>hello</System:String>
    </ListBox.Items>
</ListBox>
```

Custom Markup #1

MainWindow.xaml

```
<Button Content="Hello" Width="100" Height="100"
        Background="{local:Null}"/>
```

MainWindow.xaml.cs

```
public class Null : MarkupExtension
{
    public override object ProvideValue(IServiceProvider
serviceProvider)
    {
        //    return null;

        return new SolidColorBrush(Colors.Yellow);
    }
}

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

Custom Markup #2

MainWindow.xaml

```
<Button Content="Hello" Width="100" Height="100"
        Background="{local:Header Background}"  FontSize="{local:Header
FontSize}"/>
```

MainWindow.xaml.cs

```
public class Header : MarkupExtension
{
    private string key;

    public Header(string s) { key = s; }

    public override object ProvideValue(IServiceProvider
serviceProvider)
    {
        switch (key)
        {
            case "FontSize": return (Double)30;
            case "Background": return new SolidColorBrush(Colors.Yellow);
        }
        return null;
    }
}
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
    }
}
```

Custom Markup #3

MainWindow.xaml

```
<Button Content="Hello" Width="100" Height="100"
        Background="{local:FontInfo Element=Title, Key=Background}"/>
```

MainWindow.xaml.cs

```
public class FontInfo : MarkupExtension
{
    public string Element { get; set; }
    public string Key { get; set; }

    public override object ProvideValue(IServiceProvider
serviceProvider)
    {
        if ( Element == "Title")
        {
            switch(Key)
            {
                case "FontSize": return (Double)30;
                case "Background":return new
SolidColorBrush(Colors.Yellow);
            }
        }
        return null;
    }
}
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
    }
}
```

Example - Sliding Puzzle

MainWindow.xaml

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

    <TextBlock Grid.Row="0" FontSize="30" HorizontalAlignment="Center"
VerticalAlignment="Center">Sliding Puzzlie</TextBlock>
    <Button Grid.Row="1" Width="100" Height="30">Shuffle</Button>

    <Grid Grid.Row="2" Name="gameGrid"> </Grid>
</Grid>
```

MainWindow.xaml.cs

```
    public partial class MainWindow : Window
    {

        // Step 1.
        private const int COUNT = 5;
        private const int EMPTY = COUNT * COUNT - 1;

        public void InitGameGrid()
        {
            for (int i = 0; i < COUNT; i++)
            {
                gameGrid.RowDefinitions.Add(new RowDefinition());
                gameGrid.ColumnDefinitions.Add(new ColumnDefinition());
            }
        }

        private int[,] board = new int[COUNT, COUNT];


        public void InitBoard()
        {
            for (int y = 0; y < COUNT; y++)
            {
                for (int x = 0; x < COUNT; x++)
```

```
                {
                    board[y, x] = y * COUNT + x;
                }
            }
        }


        private double width = 0;
        private double height = 0;

        public void DrawGameGrid()
        {
            BitmapImage bitmap = new BitmapImage();
            bitmap.BeginInit();
            bitmap.UriSource = new Uri("D:\\totoro.jpg");
            bitmap.EndInit();

            width = bitmap.Width / COUNT;
            height = bitmap.Height / COUNT;

            for (int y = 0; y < COUNT; y++)
            {
                for (int x = 0; x < COUNT; x++)
                {
                    if (board[y, x] != EMPTY)
                    {
                        int no = board[y, x];
                        int bx = no % COUNT;
                        int by = no / COUNT;

                        CroppedBitmap cb = new CroppedBitmap(bitmap,
                                                new Int32Rect((int)(bx *
width), (int)(by * height),
                                                        (int)width,
(int)height));

                        Image img = new Image();
                        //img.Source = bitmap; // 전체 그림.
                        img.Source = cb; // 한개 블럭 그림
                        img.Stretch = Stretch.Fill;
                        img.Margin = new Thickness(0.5);

                        Grid.SetRow(img, y);
                        Grid.SetColumn(img, x);

                        gameGrid.Children.Add(img);
                    }
                }
```

```
            }
        }

        public MainWindow()
        {
            InitializeComponent();
            InitBoard();
            InitGameGrid();
            DrawGameGrid();
        }

        protected override void
OnMouseLeftButtonDown(MouseButtonEventArgs e)
        {
            base.OnMouseLeftButtonDown(e);

            //Point pt = e.GetPosition(this); // MainWindow 기준 좌표
            Point pt = e.GetPosition(gameGrid);

            int bx = (int)(pt.X / (gameGrid.ActualWidth / COUNT));
            int by = (int)(pt.Y / (gameGrid.ActualHeight / COUNT));

            if (bx < 0 || by < 0 || bx >= COUNT || by >= COUNT) return;

            // 상하좌우 조사
            if (bx < COUNT - 1 && board[by, bx + 1] == EMPTY) // RIGHT 가
empty
            {
                SwapBlock(bx, by, bx + 1, by);
            }
            else if (bx > 0 && board[by, bx - 1] == EMPTY) // Left 가
empty
            {
                SwapBlock(bx, by, bx - 1, by);
            }
            else if (by < COUNT - 1 && board[by + 1, bx] == EMPTY)
            {
                SwapBlock(bx, by, bx, by + 1);
            }
            else if (by > 0 && board[by - 1, bx] == EMPTY)
            {
                SwapBlock(bx, by, bx, by - 1);
            }
            else
            {
                SystemSounds.Beep.Play();
                return;
            }
            // 다 맞추었는지 확인
```

```
        }

        public void SwapBlock(int x1, int y1, int x2, int y2)
        {
            // 배열값 교환
            int temp = board[y1, x1];
            board[y1, x1] = board[y2, x2];
            board[y2, x2] = temp;

            // grid 내부의 image 교환
            Image img1 = gameGrid.Children.Cast<Image>().FirstOrDefault(n
=> Grid.GetRow(n) == y1 && Grid.GetColumn(n) == x1);
            Image img2 = gameGrid.Children.Cast<Image>().FirstOrDefault(n
=> Grid.GetRow(n) == y2 && Grid.GetColumn(n) == x2);

            if (img1 != null)
            {
                Grid.SetRow(img1, y2);
                Grid.SetColumn(img1, x2);
            }

            if (img2 != null)
            {
                Grid.SetRow(img2, y1);
                Grid.SetColumn(img2, x1);
            }
        }
    }
```

SECTION 3.

Layout

❑ 주요 학습 내용

> Canvas Layout
>
> Sketch Example
>
> StackPanel
>
> DockPanel
>
> Dialog Example
>
> Grid Layout
>
> GridSplitter

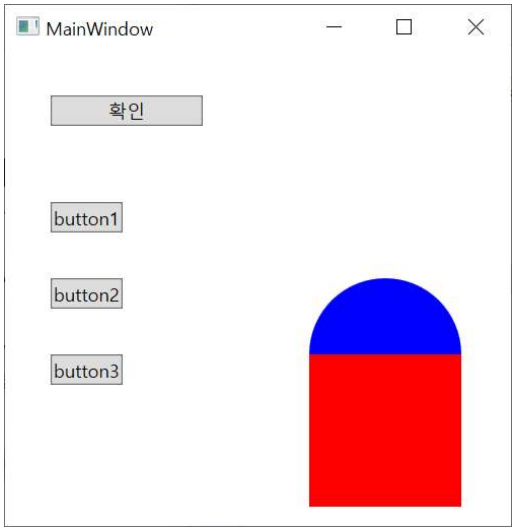
Canvas Layout

MainWindow.xaml

```
<Canvas Name="canvas">
    <Button Canvas.Top="100" Canvas.Left="30">button1</Button>
    <Button Canvas.Top="150" Canvas.Left="30">button2</Button>
    <Button Canvas.Top="200" Canvas.Left="30">button3</Button>

    <Ellipse     Panel.ZIndex="1"   Canvas.Top="150" Canvas.Left="200"
                 Width="100" Height="100" Fill="Blue"></Ellipse>
    <Rectangle   Panel.ZIndex="2" Canvas.Top="200" Canvas.Left="200"
                 Width="100" Height="100" Fill="Red"></Rectangle>

</Canvas>
```

MainWindow.xaml.cs

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

    //Canvas v = new Canvas();

    Button btn = new Button();
    btn.Content = "확인";
    btn.Width = 100;

    Canvas.SetTop(btn, 30);
    Canvas.SetLeft(btn, 30);

    canvas.Children.Add(btn);
}
```



Sketch Example

MainWindow.xaml

```
<Canvas Name="canvas" Background="white"  MouseMove="canvas_MouseMove"
                      MouseLeftButtonDown="canvas_MouseLeftButtonDown">
</Canvas>
```

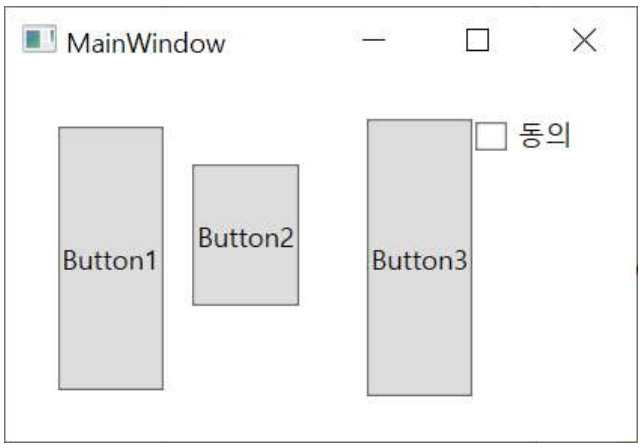
MainWindow.xaml.cs

```
public partial class MainWindow : Window
{
    private Point ptFrom;

    public MainWindow()
    {
        InitializeComponent();
    }

    private void canvas_MouseMove(object sender, MouseEventArgs e)
    {
        if (e.LeftButton == MouseButtonState.Pressed)
        {
            Point ptTo = e.GetPosition(this);

            Line line = new Line();
            line.Stroke = new SolidColorBrush(Colors.Red);

            line.X1 = ptFrom.X;
            line.Y1 = ptFrom.Y;
            line.X2 = ptTo.X;
            line.Y2 = ptTo.Y;
            canvas.Children.Add(line);
            ptFrom = ptTo;
        }
    }
    private void canvas_MouseLeftButtonDown(object sender,
                                            MouseButtonEventArgs e)
    {
        ptFrom = e.GetPosition(this);
    }
}
```

StackPanel

MainWindow.xaml

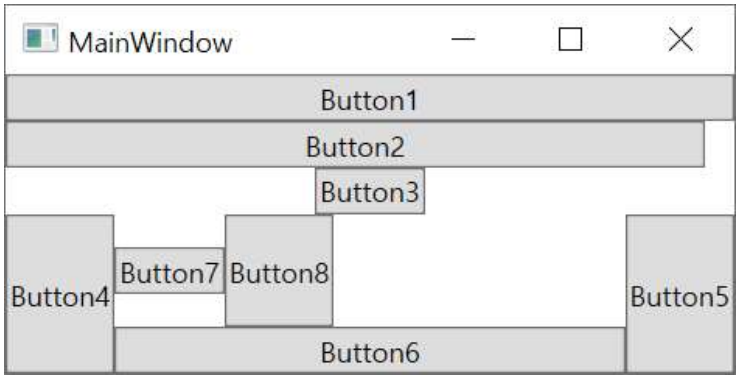
```
<StackPanel Margin="20" Orientation="Horizontal">
    <Button Margin="3">Button1</Button>
    <Button Margin="10,20,30,40">Button2</Button>
    <Button>Button3</Button>
    <CheckBox HorizontalAlignment="Center">동의</CheckBox>
</StackPanel>
```



DockPanel

MainWindow.xaml

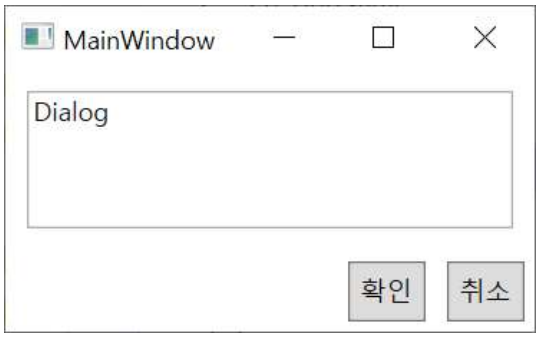
```
<DockPanel LastChildFill="False">
    <Button DockPanel.Dock="Top">Button1</Button>
    <Button DockPanel.Dock="Top" HorizontalAlignment="Left"
                                 Width="300">Button2</Button>
    <Button   DockPanel.Dock="Top"
              HorizontalAlignment="center">Button3</Button>

    <Button DockPanel.Dock="Left">Button4</Button>
    <Button DockPanel.Dock="Right">Button5</Button>
    <Button DockPanel.Dock="Bottom">Button6</Button>
    <Button VerticalAlignment="center">Button7</Button>
    <Button>Button8</Button>
</DockPanel>
```



Dialog Example

MainWindow.xaml

```
<DockPanel LastChildFill="True">
    <StackPanel DockPanel.Dock="Bottom" Orientation="Horizontal"
                HorizontalAlignment="Right">
        <Button Margin="5" Padding="5">확인</Button>
        <Button Margin="5" Padding="5">취소</Button>
    </StackPanel>
    <TextBox DockPanel.Dock="Top" Margin="10">Dialog</TextBox>
</DockPanel>
```



array.Cast<int>()

Console App

```
using System;
using System.Collections.Generic;
using System.Linq;

public class Program
{
    static void Main(string[] args)
    {
        //int[] arr = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

        //var result = arr.Where(e => e % 2 == 0);

        object[] arr = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

        var result = arr.Cast<int>().Where(e => e % 2 == 0);

        foreach ( var e in result)
        {
            Console.WriteLine(e);
        }

        object[] arr2 = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

        //var result2 = arr.Cast<int>().First(e => e % 12 == 0);

        var result2 = arr.Cast<int>().FirstOrDefault(e => e % 12 == 0);

        Console.WriteLine(result2);

    }
}
```

Grid Layout

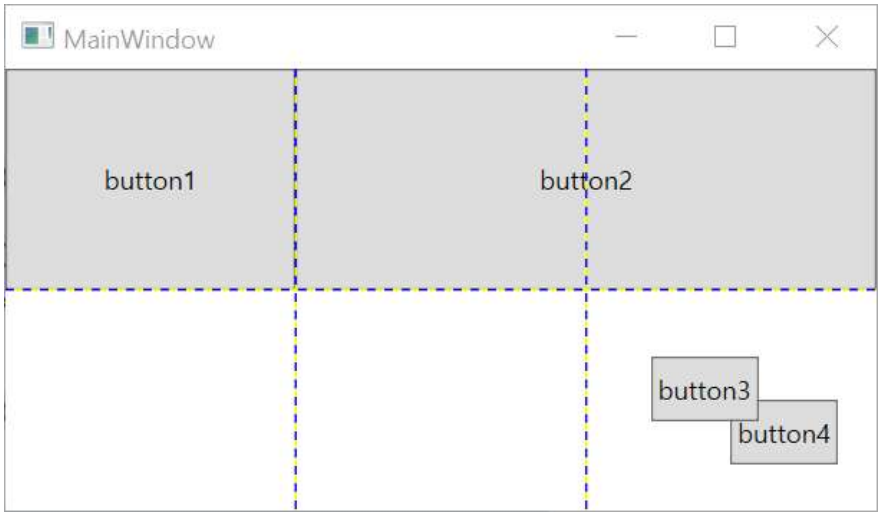
MainWindow.xaml

```
<Grid ShowGridLines="True" Name="grid">
    <Grid.RowDefinitions>
        <RowDefinition Height="50*"/>
        <RowDefinition Height="50*"/>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
    </Grid.ColumnDefinitions>

    <Button Name="button1" Grid.Row="0" Grid.Column="0"
                           Click="button1_Click" >button1</Button>
    <Button Name="button2" Grid.Row="0" Grid.Column="1"
          Grid.ColumnSpan="2" Click="button2_Click">button2</Button>

    <Button Name="button3" Grid.Row="1" Grid.Column="2" Width="50"
Height="30" Margin="29,26,54,36" Panel.ZIndex="1">button3</Button>
    <Button Name="button4" Grid.Row="1" Grid.Column="2" Width="50"
Height="30" Margin="66,46,17,16">button4</Button>
</Grid>
```

MainWindow.xaml.cs

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

    }

    private void button1_Click(object sender, RoutedEventArgs e)
    {
        Grid.SetRow(button1, 1);
        Grid.SetColumn(button1, 0);
    }

    private void button2_Click(object sender, RoutedEventArgs e)
    {
```

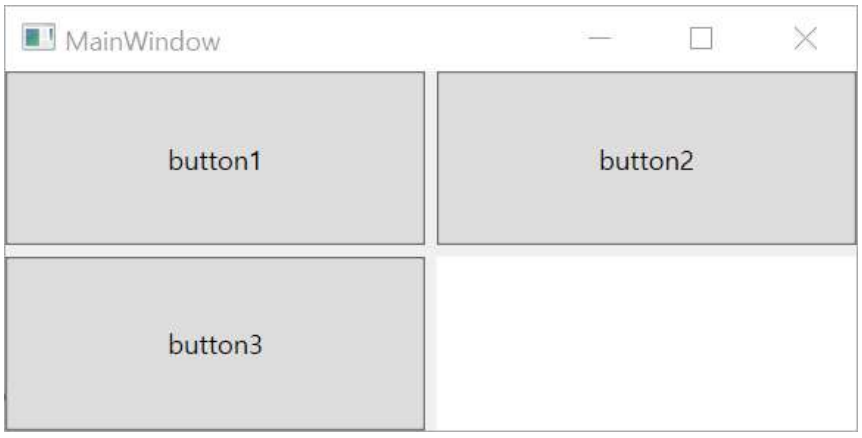
```
        // Grid 에 col=0, row=1 에 어떤 컨트롤이 있는 알고 싶다...
        Button button = grid.Children.Cast<Button>().FirstOrDefault(
                        btn => Grid.GetRow(btn) == 1 &&
Grid.GetColumn(btn) ==0);

        if (button != null)
        {
            Grid.SetRow(button, 0);
            Grid.SetColumn(button, 0);
        }
        else
        {
            MessageBox.Show("0,1 에 버튼이 없음");
        }
    }
}
```



GridSplitter

MainWindow.xaml

```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition />
        <RowDefinition Height="auto"/>
        <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition Width="auto"/>
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <Button Grid.Row="0" Grid.Column="0">button1</Button>
    <Button Grid.Row="0" Grid.Column="2">button2</Button>
    <Button Grid.Row="2" Grid.Column="0">button3</Button>
    <GridSplitter Grid.Row="0" Grid.Column="1" Grid.RowSpan="3"
Width="5" HorizontalAlignment="Center" VerticalAlignment="Stretch"/>
    <GridSplitter Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="3"
Height="5" VerticalAlignment="Center" HorizontalAlignment="Stretch"/>
</Grid>
```



SECTION 4.

Routed Event

❑ 주요 학습 내용

> Bubbling Event
>
> Tunneling Event
>
> Routed Event
>
> Sender, Tag
>
> Keypress Event

Bubbled Event & Tunelling Event

MainWindow.xaml

```
<Grid Name="grid"
        MouseLeftButtonDown="Grid_MouseLeftButtonDown"
        PreviewMouseLeftButtonDown="Grid_PreviewMouseLeftButtonDown">

    <Image Source="images/cloud.png"
             MouseLeftButtonDown="Image_MouseLeftButtonDown"
       PreviewMouseLeftButtonDown="Image_PreviewMouseLeftButtonDown"/>

</Grid>
```

MainWindow.xaml.cs

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

    }

    private void Image_MouseLeftButtonDown(object sender,
                                           MouseButtonEventArgs e)
    {
        Console.WriteLine("Image_MouseLeftButtonDown");
    }

    private void Grid_MouseLeftButtonDown(object sender,
                                           MouseButtonEventArgs e)
    {
        Console.WriteLine("Grid_MouseLeftButtonDown");
        //  e.Handled = true;
    }

    private void Window_MouseLeftButtonDown(object sender,
                                           MouseButtonEventArgs e)
    {
        Console.WriteLine("Window_MouseLeftButtonDown");
    }

    private void Image_PreviewMouseLeftButtonDown(object sender,
                                           MouseButtonEventArgs e)
    {
        Console.WriteLine("Image_PreviewMouseLeftButtonDown");
    }
```

```
    private void Grid_PreviewMouseLeftButtonDown(object sender,
                                                 MouseButtonEventArgs e)
    {
        Console.WriteLine("Grid_PreviewMouseLeftButtonDown");

        // e.Handled = true;
    }

    private void Window_PreviewMouseLeftButtonDown(object sender,
                                                   MouseButtonEventArgs e)
    {
        Console.WriteLine("Window_PreviewMouseLeftButtonDown");
    }
}
```

Background Event

MainWindow.xaml

```
<Canvas Background="{x:Null}"
MouseLeftButtonDown="Canvas_MouseLeftButtonDown">

    <Rectangle Canvas.Left="50" Canvas.Top="50"
               Width="100" Height="100" Fill="Red"
               MouseLeftButtonDown="Rectangle_MouseLeftButtonDown"/>
</Canvas>
```

MainWindow.xaml.cs

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

    private void Window_MouseLeftButtonDown(object sender,
                                            MouseButtonEventArgs e)
    {
        Console.WriteLine("Window_MouseLeftButtonDown");
    }

    private void Canvas_MouseLeftButtonDown(object sender,
                                            MouseButtonEventArgs e)
    {
        Console.WriteLine("Canvas_MouseLeftButtonDown");
    }

    private void Rectangle_MouseLeftButtonDown(object sender,
                                            MouseButtonEventArgs e)
    {
        Console.WriteLine("Rectangle_MouseLeftButtonDown");
    }
}
```

Sender, Tag

MainWindow.xaml

```
<StackPanel ButtonBase.Click="StackPanel_Click" >

    <Button Click="btn_Click" x:Name="btn1"
                              Content="button1" Tag="1" FontSize="30"/>
    <Button Click="btn_Click" x:Name="btn2"
                              Content="button2" Tag="2" FontSize="30"/>
    <Button Click="btn_Click" x:Name="btn3"
                              Content="button3" Tag="3" FontSize="30"/>
</StackPanel>
```

MainWindow.xaml.cs

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

    private void btn_Click(object sender, RoutedEventArgs e)
    {
    //    Button btn = (Button)sender;

        //   Console.WriteLine($"{btn.Name}, {btn.Content},{btn.Tag}");
    }

    private void StackPanel_Click(object sender, RoutedEventArgs e)
    {
        //Button btn = (Button)sender; // stack panel 자체의 참조

        Button btn = (Button)e.Source;

        Console.WriteLine($"{btn.Name}, {btn.Content},{btn.Tag}");
    }
}
```

KeyPressEvent

MainWindow.xaml

```
<StackPanel>
    <TextBox PreviewKeyDown = "KeyEvent" KeyDown="KeyEvent"
                  PreviewKeyUp="KeyEvent" KeyUp="KeyEvent"
                  PreviewTextInput="textInput"
                  TextChanged="TextChanged">
    </TextBox>
</StackPanel>
```

MainWindow.xaml.cs

```
private void KeyEvent(object sender, KeyEventArgs e)
{
    string message = "(KeyEvent)Event: " + e.RoutedEvent + " "
                                        + " Key: " + e.Key;
    Console.WriteLine(message);
}
private void textInput(object sender, TextCompositionEventArgs e)
{
    string message = "(textInput)Event: " + e.RoutedEvent + " "
                                         + " Text: " + e.Text;
    Console.WriteLine(message);
}
private void TextChanged(object sender, TextChangedEventArgs e)
{
    string message = "(TextChanged)Event: " + e.RoutedEvent;
    Console.WriteLine(message);
}
```

Example - NumericTextBox

MainWindow.xaml

```
<StackPanel Margin = "5" PreviewTextInput="pnl_PreviewTextInput"
PreviewKeyDown="pnl_PreviewKeyDown">
    <TextBox Margin = "3" AcceptsTab="False"></TextBox>
    <TextBox Margin = "3" ></ TextBox >
    < TextBox Margin="3"></TextBox>
</StackPanel>
```

MainWindow.xaml.cs

```
private void pnl_PreviewTextInput(object sender,
TextCompositionEventArgs e)
{
    short val;
    if (!Int16.TryParse(e.Text, out val))
    {
        e.Handled = true;
    }
}

private void pnl_PreviewKeyDown(object sender, KeyEventArgs e)
{
    if (e.Key == Key.Space)
    {
        e.Handled = true;
    }
}
```

AddHandler

MainWindow.xaml

```
<Grid>
    <Button Name = "cmd" Content="Button" MouseDown="Button_MouseDown"
              MouseUp="Button_MouseUp" HorizontalAlignment="Left"
              Margin="178,134,0,0" VerticalAlignment="Top" Width="212"
              Height="58"/>
    <Label Content = "Label" MouseDown="Label_MouseDown"
           MouseUp="Label_MouseUp" HorizontalAlignment="Left"
           Margin="183,231,0,0" VerticalAlignment="Top" Width="219"/>
</Grid>
```

MainWindow.xaml.cs

```
private void Button_MouseDown(object sender, MouseButtonEventArgs e)
{
    Console.WriteLine("MouseDown");
}
private void Button_MouseUp(object sender, MouseButtonEventArgs e)
{
    Console.WriteLine("MouseUp");
}
private void Label_MouseDown(object sender, MouseButtonEventArgs e)
{
    Console.WriteLine("label MouseDown");
}
private void Label_MouseUp(object sender, MouseButtonEventArgs e)
{
    Console.WriteLine("label MouseUp");
}
private void Backdoor(object sender, RoutedEventArgs e)
{
    Console.WriteLine("button mouseup");
}
```

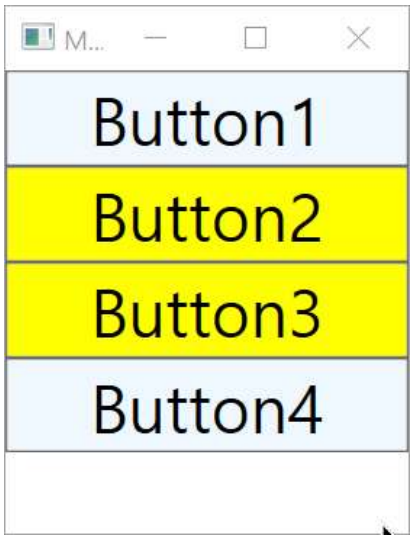
SECTION 5.

Resource & Style

❑ 주요 학습 내용

> Resource
>
> Style

Static Resource vs Dynamic Resource



MainWindow.xaml

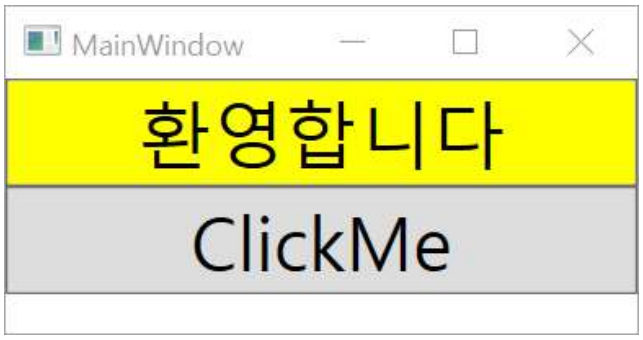
```
<Window.Resources>
    <SolidColorBrush x:Key="MyBrush" Color="Yellow"/>
</Window.Resources>
<StackPanel>

    <Button Background="AliceBlue" Content="Button1" FontSize="30"/>

    <Button Background="{StaticResource MyBrush}"
                              Content="Button2" FontSize="30"/>

    <Button Background="{DynamicResource MyBrush}"
                         Content="Button3" FontSize="30"/>

    <Button Background="AliceBlue" Content="Button4" FontSize="30"
            x:Name="button4" Click="button4_Click"/>

</StackPanel>
```

MainWindow.xaml.cs

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

```
    {
        //this.Resources["MyBrush"] = new SolidColorBrush(Colors.Yellow);

        InitializeComponent();

        //this.Resources["MyBrush"] = new SolidColorBrush(Colors.Yellow);
    }

    private void button4_Click(object sender, RoutedEventArgs e)
    {
        this.Resources["MyBrush"] = new SolidColorBrush(Colors.Red);
    }
}
```

Create User Define Object in Resource



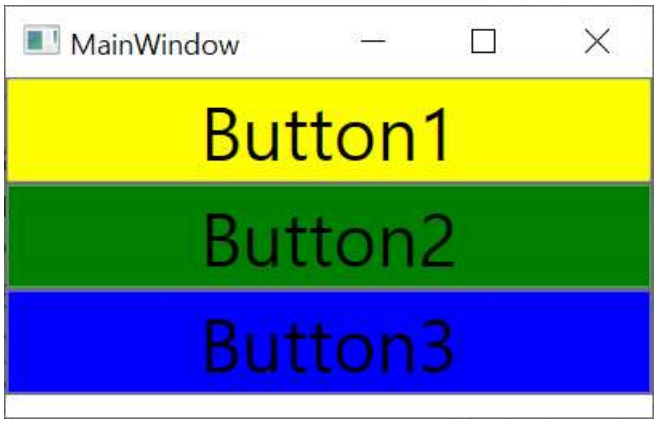
MainWindow.xaml

```
<Window x:Class="RESOURCE2.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:RESOURCE2"
        xmlns:system="clr-namespace:System;assembly=mscorlib"
        mc:Ignorable="d"
        Title="MainWindow" Height="200" Width="300">

    <Window.Resources>
        <SolidColorBrush x:Key="MyBrush" Color="Yellow"/>
        <system:String   x:Key="msg">환영합니다</system:String>
        <local:Car x:Key="mycar"/>

    </Window.Resources>

    <StackPanel>

        <Button Background="{StaticResource MyBrush}" FontSize="30"
                Content="{StaticResource msg}" />

        <Button Click="Button_Click" Content="ClickMe" FontSize="30"/>

    </StackPanel>
</Window>
```

MainWindow.xaml.cs

```
public class Car
{
    public void Go()
    {
        Console.WriteLine("Car Go");
    }
}

public partial class MainWindow : Window
{
    public MainWindow()
    {
    //    this.Resources["msg"] = "환영합니다";

        InitializeComponent();
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        Car c = (Car)this.Resources["mycar"];

        c.Go();
    }
}
```

Application Resource



MyResource.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
    <SolidColorBrush x:Key="brush3" Color="Blue"/>

</ResourceDictionary>
```

App.xaml

```
<Application x:Class="RESOURCE3.App"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:local="clr-namespace:RESOURCE3"
             StartupUri="MainWindow.xaml">
    <Application.Resources>

        <ResourceDictionary>
            <SolidColorBrush x:Key="brush2" Color="Green"/>

            <ResourceDictionary.MergedDictionaries>
                <ResourceDictionary Source="MyResource.xaml"/>
            </ResourceDictionary.MergedDictionaries>
        </ResourceDictionary>

    </Application.Resources>
</Application>
```

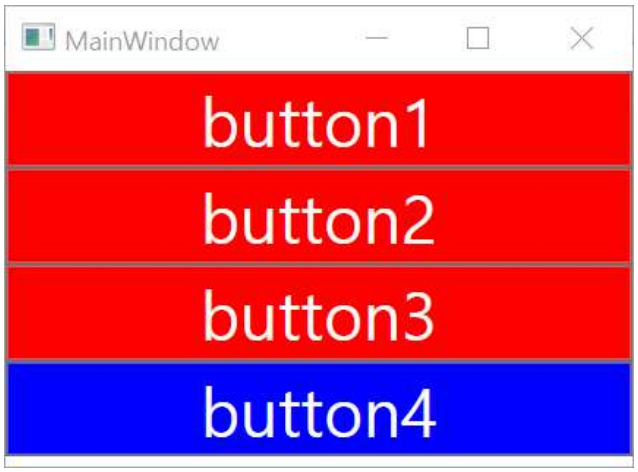
MainWindow.xaml

```
<Window.Resources>
    <SolidColorBrush x:Key="brush1" Color="Yellow"/>
</Window.Resources>

<StackPanel>
    <Button Background="{DynamicResource brush1}" Content="Button1"
FontSize="30"/>
    <Button Background="{DynamicResource brush2}" Content="Button2"
FontSize="30"/>
    <Button Background="{DynamicResource brush3}" Content="Button3"
FontSize="30"/>
</StackPanel>
```

MainWindow.xaml.cs

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

        var b1 = (SolidColorBrush)(this.Resources["brush1"]);
        var b2 =
(SolidColorBrush)(Application.Current.Resources["brush2"]);
        var b3 =
(SolidColorBrush)(Application.Current.Resources["brush3"]);
    }
}
```

Style #1



MainWindow.xaml

```
<Window.Resources>

    <Style x:Key="bluestyle">
        <Setter Property="Control.Foreground" Value="white"/>
        <Setter Property="Control.Background" Value="blue"/>
        <Setter Property="Control.FontSize" Value="30"/>
    </Style>


</Window.Resources>


<StackPanel>
    <Button Content="button1" FontSize="30" Background="Red"
Foreground="White"/>

    <Button x:Name="button2" Content="button2" />
    <Button x:Name="button3" Content="button3"
            Style="{StaticResource redstyle}"/>

    <Button x:Name="button4" Content="button4"
            Style="{StaticResource bluestyle}"/>

</StackPanel>
```

MainWindow.xaml.cs

```
public partial class MainWindow : Window
{
    public MainWindow()
    {
        Style style = new Style();

        style.Setters.Add(new Setter(Control.ForegroundProperty,
                                     new SolidColorBrush(Colors.White)));

        style.Setters.Add(new Setter(Control.BackgroundProperty,
                                     new SolidColorBrush(Colors.Red)));

        style.Setters.Add(new Setter(Control.FontSizeProperty,
                        (Double)30));

        this.Resources["redstyle"] = style;

        InitializeComponent();

        button2.Style = style;
    }
}
```

Style #2 - example

MainWindow.xaml

```
<Window.Resources>
    <Style x:Key="BigFontStyle">
        <Setter Property="Control.FontFamily" Value="Times New Roman" />
        <Setter Property="Control.FontSize"   Value="30" />
        <Setter Property="Control.FontWeight" Value="Bold" />
    </Style>
    <!-- A Style that affects all Labels -->
    <Style TargetType="Label">
        <Setter Property="Control.Foreground" Value="Red" />
        <Setter Property="Control.Background" Value="Yellow" />
        <Setter Property="Control.FontSize"   Value="30" />
    </Style>
</Window.Resources>

<StackPanel>
    <Button Content="Welcome" Style="{StaticResource BigFontStyle}"/>
    <Label Content="Label1"/>
    <Label Content="Lable2"/>
</StackPanel>
```

MainWindow.xaml

```
<Window.Resources>
    <Style x:Key="BaseFontStyle">
        <Setter Property="Control.FontFamily" Value="Times New Roman" />
        <Setter Property="Control.FontSize"   Value="30" />
        <Setter Property="Control.FontWeight" Value="Bold" />
    </Style>

    <!-- 핵심 : Style 은 확장(Extended)할수도 있다 -->
    <Style  x:Key="TitleFontStyle" BasedOn="{StaticResource
BaseFontStyle}">
        <Setter Property="Control.Foreground" Value="White" />
        <Setter Property="Control.Background" Value="DarkBlue" />
    </Style>

</Window.Resources>

<StackPanel>
    <Button Content="Welcome" Style="{StaticResource TitleFontStyle}"/>
</StackPanel>
```

MainWindow.xaml

```
<Window.Resources>
    <Style x:Key="MouseOverHighlightStyle1">
        <Setter Property="Control.FontSize" Value="30" />
        <EventSetter Event="FrameworkElement.MouseEnter"
Handler="element_MouseEnter" />
        <EventSetter Event="FrameworkElement.MouseLeave"
Handler="element_MouseLeave" />
    </Style>

    <Style x:Key="MouseOverHighlightStyle2">
        <Setter Property="Control.FontSize" Value="30" />

        <Style.Triggers>
            <Trigger Property="Control.IsMouseOver" Value="True">
                <Setter Property="Control.Background" Value="Yellow" />
            </Trigger>
            <Trigger Property="Control.IsMouseOver" Value="False">
                <Setter Property="Control.Background" Value="{x:Null}" />
            </Trigger>
        </Style.Triggers>
    </Style>
</Window.Resources>
<StackPanel>
    <Label Content="Label1" Style="{StaticResource
MouseOverHighlightStyle1}"/>
    <Label Content="Lable2" Style="{StaticResource
MouseOverHighlightStyle2}"/>

</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Example3 : Window
{
    public Example3()
    {
        InitializeComponent();
```

```
    }

    private Brush highlightBrush = new SolidColorBrush(Colors.Yellow);

    private void element_MouseEnter(object sender, MouseEventArgs e)
    {
        Label lb = (Label)sender;

        lb.Background = highlightBrush;
    }
    private void element_MouseLeave(object sender, MouseEventArgs e)
    {
        Label lb = (Label)sender;

        lb.Background = null;
    }
}
```

Trigger

MainWindow.xaml

```
<Window.Resources>
    <Style x:Key="BigFontButton">
        <Style.Setters>
            <Setter Property = "Control.FontFamily" Value="Times New
Roman" />
            <Setter Property = "Control.FontSize" Value="18" />
        </Style.Setters>

        <Style.Triggers>
            <Trigger Property = "Control.IsMouseOver" Value="True">
                <Setter Property = "Control.Foreground" Value="Red" />
            </Trigger>
            <Trigger Property = "Control.IsFocused" Value="True">
                <Setter Property = "Control.Foreground" Value="Yellow" />
            </Trigger>

            <Trigger Property = "Button.IsPressed" Value="True">
                <Setter Property = "Control.Foreground" Value="Blue" />
            </Trigger>

            <EventTrigger RoutedEvent = "Mouse.MouseEnter" >
                < EventTrigger.Actions >
                    < BeginStoryboard >
                        < Storyboard >
                            < DoubleAnimation Duration="0:0:0.2"
Storyboard.TargetProperty="FontSize" To="22"/>
                        </Storyboard>
                    </BeginStoryboard>
                </EventTrigger.Actions>
            </EventTrigger>

            <EventTrigger RoutedEvent = "Mouse.MouseLeave" >
                < EventTrigger.Actions >
                    < BeginStoryboard >
                        < Storyboard >
                            < DoubleAnimation  Duration="0:0:1"
Storyboard.TargetProperty="FontSize"  />
                        </Storyboard>
                    </BeginStoryboard>
```

```
            </EventTrigger.Actions>
          </EventTrigger>
       </Style.Triggers>
    </Style>
</Window.Resources>

<StackPanel Margin = "5" >
    < Button Padding="5" Margin="5">Normal Button</Button>
       <Button Padding = "5" Margin= "5"  Style= "{StaticResource
BigFontButton}" > A Customized Button</Button>
</StackPanel>
```

SECTION 6.

Control

❑ 주요 학습 내용

> Control Style
>
> Control Event
>
> VisualTree
>
> Custom Control

Control Example

MainWindow.xaml

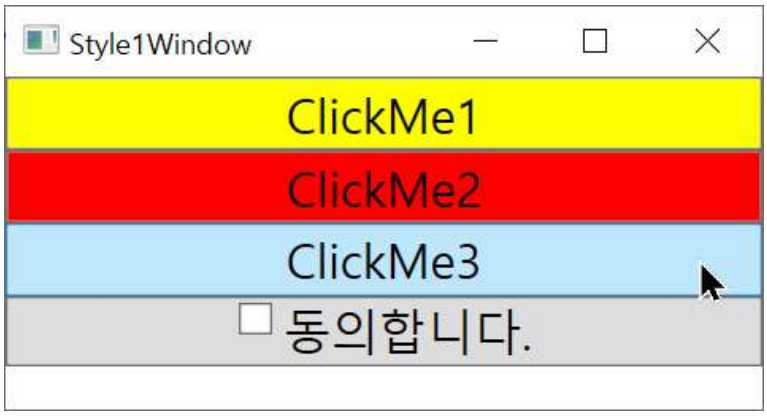
```
< Grid >
    < Grid.RowDefinitions >
        < RowDefinition Height = "114*" />
        < RowDefinition Height = "201*" />
        < RowDefinition Height = "107*" />
    </ Grid.RowDefinitions >
    < Grid.ColumnDefinitions >
        < ColumnDefinition Width = "128*" />
        < ColumnDefinition Width = "145*" />
        < ColumnDefinition Width = "161*" />
        < ColumnDefinition Width = "179*" />
        < ColumnDefinition Width = "181*" />
    </ Grid.ColumnDefinitions >
    < TextBox Name = "textbox" Margin = "20" Grid.Row = "1" TextWrapping
= "Wrap" Text = "TextBox" />
    < Button Name = "button1" Click = "Button1_Click"  Margin = "20"
Content = "Button" Grid.Column = "1" Grid.Row = "1" />
    < ListBox Name = "listbox" Margin = "20" Grid.Column = "2" Grid.Row =
"1" />
    < Button Name = "button2" Click = "Button2_Click" Margin = "20"
Content = "Button" Grid.Column = "3" Grid.Row = "1" />
    < ComboBox Name = "combobox" Margin = "20" Grid.Column = "4" Grid.Row
= "1" Height = "50" />

</ Grid >
```

MainWindow.xaml.cs

```
public partial class Ex4_ControlExample1 : Window
{
    public Ex4_ControlExample1()
    {
        InitializeComponent();
    }

    private void Button1_Click(object sender, RoutedEventArgs e)
    {
        string s = textbox.Text;
```

```
        listbox.Items.Add(s);

        textbox.Clear();
    }

    private void Button2_Click(object sender, RoutedEventArgs e)
    {
        string s = listbox.SelectedItem.ToString();

        combobox.Items.Add(s);

        int idx = listbox.SelectedIndex;

        if (idx != -1)
            listbox.Items.RemoveAt(idx);
    }
}
```

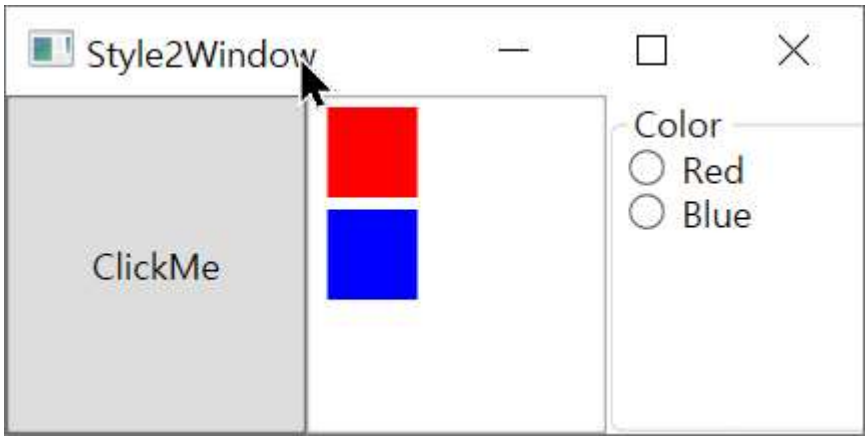
Control Style #1



MainWindow.xaml

```
<StackPanel>
    <Button Background="Yellow"  Content="ClickMe1"
FontSize="20"></Button>
    <Button Content="ClickMe2" FontSize="20">
        <Button.Background>
            <SolidColorBrush Color="Red"/>
        </Button.Background>
    </Button>
    <Button Content="ClickMe3" FontSize="20">
        <Button.Background>
            <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
                <GradientStop Color="Black" Offset="0"/>
                <GradientStop Color="#FFEDDADA" Offset="1"/>
                <GradientStop Color="#FF040404" Offset="0.491"/>
            </LinearGradientBrush>
        </Button.Background>
    </Button>
    <Button  FontSize="20">
        <StackPanel>
            <CheckBox Content="동의합니다."/>
        </StackPanel>
    </Button>
</StackPanel>
```

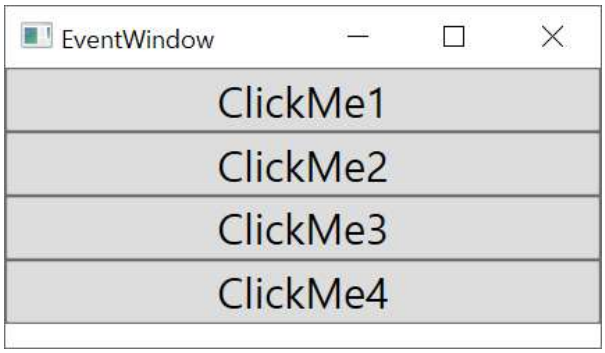
Control Style #2



MainWindow.xaml

```
<StackPanel Orientation="Horizontal">

    <Button Content="ClickMe" Width="100"/>

    <ListBox Width="100">
        <ListBox.Items>
            <Rectangle Fill="Red" Width="30" Height="30"/>
            <Rectangle Fill="Blue" Width="30" Height="30"/>
        </ListBox.Items>
    </ListBox>

    <GroupBox Header="Color" Width="100">
        <StackPanel>
            <RadioButton Content="Red"/>
            <RadioButton Content="Blue"/>
        </StackPanel>
    </GroupBox>

</StackPanel>
```

Control Event



MainWindow.xaml

```
<Window.Resources>

    <BeginStoryboard x:Key="mystoryboard">
        <Storyboard>
            <DoubleAnimation Storyboard.TargetName ="button1"
                   Storyboard.TargetProperty="Width"
                          From="100" To="200" Duration="0:0:5"/>
        </Storyboard>
    </BeginStoryboard>

</Window.Resources>

<StackPanel>

    <Button Click="button1_Click" x:Name="button1"  Content="ClickMe1"
FontSize="20"></Button>

    <Button Click="Clicked" Content="ClickMe2" FontSize="20">
        <x:Code>
            <![CDATA[
            void Clicked(object sender, RoutedEventArgs e)
            {
                button1.Content = "Hello World";
            }
            ]]>
        </x:Code>
    </Button>
```

```
    <Button Content="ClickMe3" FontSize="20">
        <Button.Triggers>
            <EventTrigger RoutedEvent="Button.Click">
                <EventTrigger.Actions>

                    <BeginStoryboard>
                        <Storyboard>
                            <DoubleAnimation Storyboard.TargetName
="button1"
                 Storyboard.TargetProperty="Width"
                 From="100" To="200" Duration="0:0:5"/>
                        </Storyboard>
                    </BeginStoryboard>

                </EventTrigger.Actions>
            </EventTrigger>
        </Button.Triggers>

    </Button>


    <Button Content="ClickMe4" FontSize="20">

        <Button.Triggers>
            <EventTrigger RoutedEvent="Button.Click">

                <StaticResource ResourceKey="mystoryboard"/>

            </EventTrigger>
        </Button.Triggers>
    </Button>

</StackPanel>
```

VisualTreeHelper

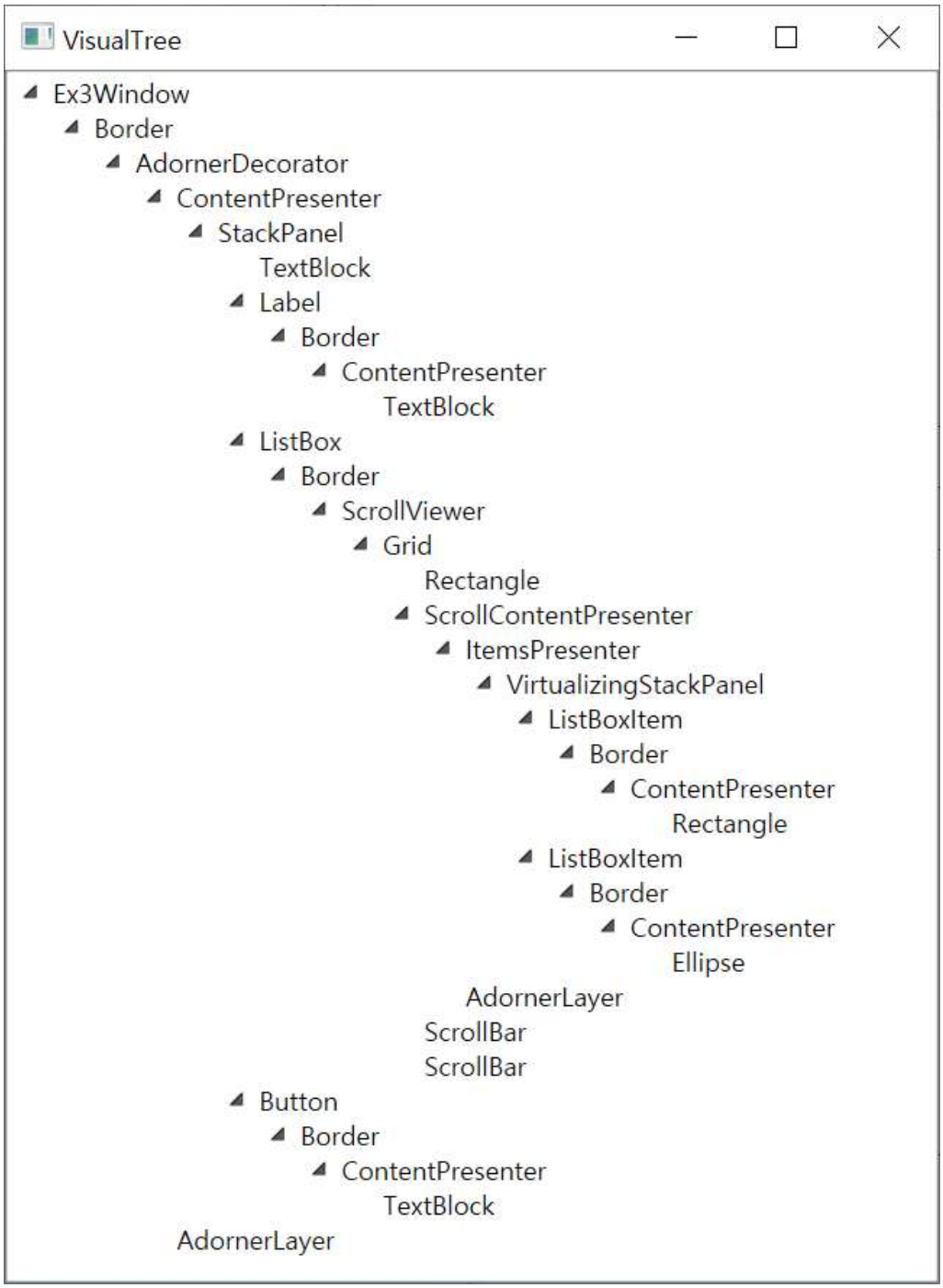
MainWindow.xaml

```
<Grid>
    <Button Content="ClickMe" Click="Button_Click" Width="100"
Height="100"/>
</Grid>
```

MainWindow.xaml.cs

```
public partial class Ex2Window : Window
{
    public Ex2Window()
    {
        InitializeComponent();
    }
    private void Button_Click(object sender, RoutedEventArgs e)
    {
        ShowChild("", this);
    }
    private void ShowChild(string sep, DependencyObject obj)
    {
        Console.WriteLine($"{sep}{obj.GetType()}");

        int cnt = VisualTreeHelper.GetChildrenCount(obj);

        for (int i = 0; i < cnt; i++)
        {
            var child = VisualTreeHelper.GetChild(obj, i);

            ShowChild(sep + " ", child);
        }
    }
}
```

Visual Tree


VisualTree
Ex3Window
Border
AdornerDecorator
ContentPresenter
StackPanel
TextBlock
Label
Border
ContentPresenter
TextBlock
ListBox
Border
ScrollViewer
Grid
Rectangle
ScrollContentPresenter
ItemsPresenter
VirtualizingStackPanel
ListBoxItem
Border
ContentPresenter
Rectangle
ListBoxItem
Border
ContentPresenter
Ellipse
AdornerLayer
ScrollBar
ScrollBar
Button
Border
ContentPresenter
TextBlock
AdornerLayer

VisualTree.xaml

```
<Grid>
    <TreeView Name="treeView"></TreeView>
</Grid>
```

VisualTree.xaml.cs

```
public partial class VisualTree : Window
{
    public VisualTree()
    {
        InitializeComponent();
    }
    public void Process(DependencyObject root)
    {
        treeView.Items.Clear();
        ProcessElement(root, null);
    }

    private void ProcessElement(DependencyObject element, TreeViewItem
previousItem)
    {
        TreeViewItem item = new TreeViewItem();
        item.Header = element.GetType().Name;

        item.IsExpanded = true;

        if (previousItem == null)
            treeView.Items.Add(item);
        else
            previousItem.Items.Add(item);

        for (int i = 0; i < VisualTreeHelper.GetChildrenCount(element);
i++)
        {
            ProcessElement(VisualTreeHelper.GetChild(element, i), item);
        }
    }
}
```

Using Visual Tree

UsingVisualTree.xaml

```
<StackPanel>
    <TextBlock Text="TextBlock"/>
    <Label Content="Label"/>
    <ListBox>
        <Rectangle Fill="Red" Width="100" Height="30"/>
        <Ellipse Fill="Blue" Width="100" Height="30"/>
    </ListBox>
    <Button Content="Button" Click="Button_Click"/>
</StackPanel>
```

UsingVisualTree.xaml.cs

```
public partial class Ex3Window : Window
{
    public Ex3Window()
    {
        InitializeComponent();
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        VisualTree vt = new VisualTree();

        vt.Process(this);

        vt.Show();
    }
}
```

Custom Control

MainWindow.xaml

```
< Grid >
    < Button Content = "Button" HorizontalAlignment = "Left" Margin =
"51,85,0,0" VerticalAlignment = "Top" Width = "75" />
    < Button HorizontalAlignment = "Left" Margin = "202,86,0,0"
VerticalAlignment = "Top" Width = "75" >
        < StackPanel >
            < CheckBox Content = "동의합니다." ></ CheckBox >
            < ListBox >
                < ListBoxItem > Item1 </ ListBoxItem >
                < ListBoxItem > Item2 </ ListBoxItem >
                < ListBoxItem > Item3 </ ListBoxItem >
            </ ListBox >
        </ StackPanel >
    </ Button >

    < Button HorizontalAlignment = "Left" Margin = "352,82,0,0"
VerticalAlignment = "Top" Width = "75" >
        < Ellipse Fill = "Red" Width = "30" Height = "30" />
    </ Button >

    < Button Content = "Button4" Background = "blue" HorizontalAlignment =
"Left" Margin = "510,91,0,0" VerticalAlignment = "Top" Width = "75" >
        < Button.Template >
            < ControlTemplate TargetType = "{x:Type Button}" >
                < Grid >
                    < Ellipse Fill = "Red" Width = "30" Height = "30" />
                    < Label Content = "{TemplateBinding Content}"
HorizontalAlignment = "Center" VerticalAlignment = "Center" ></ Label >
                </ Grid >
            </ ControlTemplate >
        </ Button.Template >
    </ Button >
    < Button Content = "Button" HorizontalAlignment = "Left" Margin =
"620,83,0,0" VerticalAlignment = "Top" Width = "75" Click = "Button_Click" />
</ Grid >
```

SECTION 7.

Binding

❑ 주요 학습 내용

> Element Binding
>
> Data Binding

No Binding

MainWindow.xaml

```
<StackPanel>
    <Label  Name="label" Content="Label" FontSize="30"/>
    <Slider Name="slider" Minimum="10" Maximum="100" Value="30"
            ValueChanged="slider_ValueChanged"/>
    <Button Name="button" Content="ClickMe" Click="button_Click"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Ex1Window : Window
{
    public Ex1Window()
    {
        InitializeComponent();
    }
    private void button_Click(object sender, RoutedEventArgs e)
    {
        label.FontSize = 30;
        slider.Value = 30;
    }
    private void slider_ValueChanged(object sender,
                            RoutedPropertyChangedEventArgs<double> e)
    {
        label.FontSize = slider.Value;
    }
}
```

Element Binding using Code

MainWindow.xaml

```
<StackPanel>
    <Label Name="label" Content="Label" FontSize="30"/>
    <Slider Name="slider" Minimum="10" Maximum="100" Value="30"/>
    <Button Name="button" Content="ClickMe" Click="button_Click"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Ex2Window : Window
{
    public Ex2Window()
    {
        InitializeComponent();

        Binding b = new Binding();
        b.Source = slider;
        b.Path = new PropertyPath("Value");
        b.Mode = BindingMode.TwoWay;

        label.SetBinding(System.Windows.Controls.Label.FontSizeProperty,
b);
    }

    private void button_Click(object sender, RoutedEventArgs e)
    {
        label.FontSize = 30;
    }
}
```

Element Binding using Xaml

MainWindow.xaml

```
<StackPanel>
    <Label Name="label" Content="Label"
           FontSize="{Binding ElementName=slider, Path=Value,
Mode=TwoWay}"/>

    <Slider Name="slider" Minimum="10" Maximum="100" Value="30"/>
    <Button Name="button" Content="ClickMe" Click="button_Click"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Ex3Window : Window
{
    public Ex3Window()
    {
        InitializeComponent();
    }

    private void button_Click(object sender, RoutedEventArgs e)
    {
        label.FontSize = 30;
    }
}
```

Data Binding #1 – Property Path

MainWindow.xaml

```
<StackPanel x:Name="stackpanel">
    <TextBox Text="{Binding Path=Name}" FontSize="30"/>
    <TextBox Text="{Binding Path=Address}" FontSize="30"/>
    <Button Content="ClickMe" Click="Button_Click"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Ex1Window : Window
{
    public Person person;

    public Ex1Window()
    {
        InitializeComponent();

        person = new Person();
        person.Name = "kim";
        person.Address = "seoul";

        stackpanel.DataContext = person;
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        person.Name = "lee";
        person.Address = "jeju";
    }
}
```

Data Binding #2 – ItemSource

MainWindow.xaml

```
<StackPanel>
    <ListBox x:Name="listbox" DisplayMemberPath="Name" FontSize="20" />
    <Button Content="button1" Click="Button1_Click"/>
    <Button Content="button2" Click="Button2_Click"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class Ex2Window : Window
{
    //public List<Person> st = new List<Person>();

    public ObservableCollection<Person> st =
                                new ObservableCollection<Person>();

    public Ex2Window()
    {
        InitializeComponent();

        st.Add(new Person { Name = "kim1", Address = "jeju1" });
        st.Add(new Person { Name = "kim2", Address = "jeju2" });

        listbox.ItemsSource = st;
    }

    private void Button1_Click(object sender, RoutedEventArgs e)
    {
        st[0].Name = "lee";
    }
    private void Button2_Click(object sender, RoutedEventArgs e)
    {
        st.Add(new Person { Name = "kim3", Address = "jeju3" });
    }
}
```

Data Binding #3

MainWindow.xaml

```
<Grid>
    <ListView Name="listview" FontSize="30">
        <ListView.View>
            <GridView>
                <GridViewColumn Header="Name" Width="100"
                            DisplayMemberBinding="{Binding Path=Name}"/>

                <GridViewColumn Header="Address" Width="150"
                            DisplayMemberBinding="{Binding
Path=Address}"/>
            </GridView>
        </ListView.View>
    </ListView>
</Grid>
```

MainWindow.xaml.cs

```
public partial class Ex3Window : Window
{
    public ObservableCollection<Person> st =
                                new ObservableCollection<Person>();

    public Ex3Window()
    {
        InitializeComponent();

        st.Add(new Person { Name = "kim1", Address = "jeju1" });
        st.Add(new Person { Name = "kim2", Address = "jeju2" });
        st.Add(new Person { Name = "kim3", Address = "jeju3" });
        st.Add(new Person { Name = "kim4", Address = "jeju4" });

        listview.ItemsSource = st;
    }
}
```

Data Binding #4 - DataGrid

Person.cs

```
public class Person : INotifyPropertyChanged
{
    private string name;
    private string address;

    public string Name
    {
        get { return name; }
        set { name = value;
            if (PropertyChanged != null)
                PropertyChanged(this,
                                new PropertyChangedEventArgs("Name"));
        }
    }

    public string Address
    {
        get { return address; }
        set { address = value;

            if (PropertyChanged != null)
                PropertyChanged(this,
                                new PropertyChangedEventArgs("Address"));

        }
    }

    public event PropertyChangedEventHandler PropertyChanged;
}
```

MainWindow.xaml

```
<DataGrid x:Name="dataGrid" FontSize="30">

</DataGrid>
```

MainWindow.xaml.cs

```
public partial class Ex4Window : Window
```

```
{
    public ObservableCollection<Person> st =
                                new ObservableCollection<Person>();

    public Ex4Window()
    {
        InitializeComponent();

        st.Add(new Person { Name = "kim1", Address = "jeju1" });
        st.Add(new Person { Name = "kim2", Address = "jeju2" });
        st.Add(new Person { Name = "kim3", Address = "jeju3" });
        st.Add(new Person { Name = "kim4", Address = "jeju4" });

        dataGrid.ItemsSource = st;
    }
}
```

SECTION 8.

Command & Menu

❑ 주요 학습 내용

> Command
>
> Menu

Command #1

MainWindow.xaml

```
<StackPanel>
    <TextBox x:Name="txtBox" Margin="10" FontSize="30"/>
    <Button  Click="button1_Click"  x:Name="button1" Margin="10"
                                FontSize="30" Content="button1"/>
    <Button  Command="local:MyCommand.cmdAction" x:Name="button2"
                      Margin="10" FontSize="30" Content="button2"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public class ActionCommand : ICommand
{
    public event EventHandler CanExecuteChanged;

    public bool CanExecute(object parameter)
    {
        return true;
        //return false;
    }
    public void Execute(object parameter)
    {
        MessageBox.Show("ActionCommand");
    }
}
static class MyCommand
{
    public static ActionCommand cmdAction = new ActionCommand();
}
public partial class Command2Window : Window
{
    public Command2Window()
    {
        InitializeComponent();
    }
    private void button1_Click(object sender, RoutedEventArgs e) {}
}
```

Command #2 - CanExecute

MainWindow.xaml

```
<StackPanel>
    <TextBox TextChanged="txtBox_TextChanged" x:Name="txtBox"
                                        Margin="10" FontSize="30"/>
    <Button  Command="local:MyCommand2.cmdAction" x:Name="button1"
                    Margin="10" FontSize="30" Content="button1"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public class ActionCommand2 : ICommand
{
    public event EventHandler CanExecuteChanged;

    public TextBox txtBox = null;

    public void FireCanExecute()
    {
        // CanExecute 를 다시 호출해 달라는 의미.
        CanExecuteChanged(this, EventArgs.Empty);
    }

    public bool CanExecute(object parameter)
    {
        Console.WriteLine("CanExecute");

        if (txtBox == null)
            txtBox =
((Command3Window)Application.Current.MainWindow).txtBox;

        return !string.IsNullOrEmpty(txtBox.Text);
    }

    public void Execute(object parameter)
    {
        MessageBox.Show(txtBox?.Text);
    }
}

static class MyCommand2
{
    public static ActionCommand2 cmdAction = new ActionCommand2();
}

public partial class Command3Window : Window
```

```
{
    public Command3Window()
    {
        InitializeComponent();
    }

    private void txtBox_TextChanged(object sender,
                                    TextChangedEventArgs e)
    {
        MyCommand2.cmdAction.FireCanExecute();
    }
}
```

Command #3

MainWindow.xaml

```
<StackPanel>
    <TextBox x:Name="txtBox" Margin="10" FontSize="30"/>
    <Button  Command="local:MyCommand3.cmdAction" x:Name="button1"
       Margin="10" FontSize="30" Content="button1"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public class ActionCommand3 : ICommand
{

    public event EventHandler CanExecuteChanged
    {
        add
        {
            CommandManager.RequerySuggested += value;
        }

        remove
        {
            CommandManager.RequerySuggested -= value;
        }
    }
    public TextBox txtBox = null;


    public bool CanExecute(object parameter)
    {
        Console.WriteLine("CanExecute");

        if (txtBox == null)
            txtBox =
((Command4Window)Application.Current.MainWindow).txtBox;

        return !string.IsNullOrEmpty(txtBox.Text);
    }
    public void Execute(object parameter)
    {
        MessageBox.Show(txtBox?.Text);
```

```
    }
}

static class MyCommand3
{
    public static ActionCommand3 cmdAction = new ActionCommand3();
}

public partial class Command4Window : Window
{
    public Command4Window()
    {
        InitializeComponent();
    }
}
```

Routed Command #1

MainWindow.xaml

```
<StackPanel>
    <TextBox x:Name="txtBox" Margin="10" FontSize="30"/>
    <Button  Command="local:MyCommand.cmdAction" x:Name="button"
Margin="10" FontSize="30" Content="button1"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public static class MyCommand
{
    public static RoutedCommand cmdAction = new RoutedCommand();

}

public partial class RoutedCommand1 : Window
{
    public RoutedCommand1()
    {
        InitializeComponent();

        CommandBinding cmd = new CommandBinding(MyCommand.cmdAction);

        cmd.Executed += Cmd_Executed;
        cmd.CanExecute += Cmd_CanExecute;

        this.CommandBindings.Add(cmd);

        InputBinding ibFind = new InputBinding(MyCommand.cmdAction,
                    new KeyGesture(Key.R, ModifierKeys.Control));
        this.InputBindings.Add(ibFind);
    }

    private void Cmd_CanExecute(object sender, CanExecuteRoutedEventArgs
e)
    {
        e.CanExecute = !string.IsNullOrEmpty(txtBox.Text);
    }

    private void Cmd_Executed(object sender, ExecutedRoutedEventArgs e)
```

```
    {
        MessageBox.Show(txtBox.Text);
    }
}
```

Routed Command #2

MainWindow.xaml

```
<Window.Resources>
    <RoutedCommand x:Key="cmdAction" />
</Window.Resources>
<Window.CommandBindings>
    <CommandBinding Command="{StaticResource ResourceKey=cmdAction}"
       Executed="Cmd_Executed" CanExecute="Cmd_CanExecute"/>
</Window.CommandBindings>
<Window.InputBindings>
    <KeyBinding Key="R" Modifiers="Control"
                Command="{StaticResource ResourceKey=cmdAction}"/>
</Window.InputBindings>
<StackPanel>
    <TextBox x:Name="txtBox" Margin="10" FontSize="30"/>
    <Button  Command="{StaticResource ResourceKey=cmdAction}"
       x:Name="button" Margin="10" FontSize="30" Content="button1"/>
</StackPanel>
```

MainWindow.xaml.cs

```
public partial class RoutedCommand2 : Window
{
    public RoutedCommand2()
    {
        InitializeComponent();
    }
    private void Cmd_CanExecute(object sender,
                                        CanExecuteRoutedEventArgs e)
    {
        e.CanExecute = !string.IsNullOrEmpty(txtBox.Text);
    }
    private void Cmd_Executed(object sender, ExecutedRoutedEventArgs e)
    {
        MessageBox.Show(txtBox.Text);
    }
}
```

Menu & ToolBar

MainWindow.xaml

```
<Window.Resources>
    <RoutedCommand x:Key="cmdAction" />
</Window.Resources>
<Window.CommandBindings>
    <CommandBinding Command="{StaticResource ResourceKey=cmdAction}"
Executed="MenuItem_Click"/>
</Window.CommandBindings>
<Window.InputBindings>
    <KeyBinding Key="O" Modifiers="Control" Command="{StaticResource
ResourceKey=cmdAction}"/>
</Window.InputBindings>
<StackPanel>
    <Menu>
        <MenuItem Header="File">
            <MenuItem Header="One" InputGestureText="Ctrl+O"
                 Command="{StaticResource ResourceKey=cmdAction}">
                <MenuItem.Icon>
                    <Image Source="images/melon.png" />
                </MenuItem.Icon>
            </MenuItem>

            <MenuItem Header="Two" Click="MenuItem_Click"/>
            <Separator/>
            <MenuItem Header="Exit"/>
        </MenuItem>

        <MenuItem Header="Help">
            <MenuItem Header="Three"/>
            <MenuItem Header="Four"/>
        </MenuItem>
    </Menu>

    <ToolBarTray >
        <ToolBar >
            <Button  Command="{StaticResource ResourceKey=cmdAction}">
                <StackPanel>
                    <Image Source="images/melon.png" />
                    <Label>OK</Label>
                </StackPanel>
```

```
            </Button>
            <Button>Two</Button>
            <Button>Three</Button>
        </ToolBar>
    </ToolBarTray >

</StackPanel>
```

MainWindow.xaml.cs

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

    private void MenuItem_Click(object sender, RoutedEventArgs e)
    {
        MessageBox.Show("menu click");
    }
}
```

SECTION 9.

MISCELLANEOUS

❑ 주요 학습 내용

> Code Animation
>
> Xaml Animation
>
> File Dialog
>
> Threading
>
> Play Audio
>
> Play Video
>
> Culture

Code Animation

MainWindow.xaml

```
<Canvas>
        <Ellipse Name="RedBall" Fill="Red" Width="100" Height="100"
                 Canvas.Left="132" Canvas.Top="62"/>
        <Button Name="button1" Click="Button1_Click" Content="Button"
                 Canvas.Left="144" Canvas.Top="200" Width="75"/>
        <Button Name="button2" Click="Button2_Click" Content="Button"
                 Canvas.Left="144" Canvas.Top="241" Width="75"/>
        <Button Name="button3" Click="Button3_Click" Content="Button"
                 Canvas.Left="148" Canvas.Top="280" Width="75"/>
        <Button Name="button4" Click="Button4_Click" Content="Button"
                 Canvas.Left="148" Canvas.Top="323" Width="75"/>
        <Button Name="button5" Click="Button5_Click" Content="Button"
                 Canvas.Left="148" Canvas.Top="353" Width="75"/>
        <Button Name="button6" Click="Button6_Click" Content="Button"
                 Canvas.Left="248" Canvas.Top="353" Width="75"/>
        <Button Name="button7" Click="Button5_Click" Content="Button"
                 Canvas.Left="148" Canvas.Top="400" Width="75"/>
        <Button Name="button8" Click="Button6_Click" Content="Button"
                 Canvas.Left="248" Canvas.Top="400" Width="75"/>
    </Canvas>
```

MainWindow.xaml.cs

```
public partial class Ex1_CodeAnimation : Window
{
    public Ex1_CodeAnimation()
    {
        InitializeComponent();
    }

    private void Animation_Completed(object sender, EventArgs e)
    {
        Console.WriteLine("Animation Completed");
    }

    private void Button1_Click(object sender, RoutedEventArgs e)
    {
        // Width 속성 변경
        DoubleAnimation anim1 = new DoubleAnimation();
```

```
        anim1.From = RedBall.Width;
        anim1.To = RedBall.Width + 100;
        anim1.Duration = new TimeSpan(0, 0, 3);

        //   anim1.Completed += Animation_Completed;

        RedBall.BeginAnimation(Ellipse.WidthProperty, anim1);
    }

    private void Button2_Click(object sender, RoutedEventArgs e)
    {
        DoubleAnimation anim1 = new DoubleAnimation();
        anim1.Duration = TimeSpan.FromSeconds(3);
        RedBall.BeginAnimation(Ellipse.WidthProperty, anim1);
    }

    private void Button3_Click(object sender, RoutedEventArgs e)
    {
        DoubleAnimation anim1 = new DoubleAnimation();
        anim1.From = (double)RedBall.GetValue(Canvas.LeftProperty);
        anim1.To = (double)RedBall.GetValue(Canvas.LeftProperty) + 100;
        anim1.Duration = TimeSpan.FromSeconds(3);

        RedBall.BeginAnimation(Canvas.LeftProperty, anim1);
    }
    private void Button4_Click(object sender, RoutedEventArgs e)
    {
        DoubleAnimation anim1 = new DoubleAnimation();
        anim1.By = -30;
        anim1.Duration = TimeSpan.FromSeconds(1);

        RedBall.BeginAnimation(Canvas.LeftProperty, anim1);
    }

    // DispatcherTimer 사용
    private DispatcherTimer MyTimer = new DispatcherTimer();

    private void Button5_Click(object sender, RoutedEventArgs e)
    {
        MyTimer.Interval = new TimeSpan(30); // 30 nano
        MyTimer.Tick += UpdateRedBall;
        MyTimer.Start();
    }

    private double xPos = 0;
```

```
    void UpdateRedBall(object sender, EventArgs e)
    {
        Canvas.SetLeft(RedBall, xPos);
        if (xPos > this.Width)
            xPos = 0;
        else
            xPos += 0.1;
        Thread.Sleep(1);
    }
    private void Button6_Click(object sender, RoutedEventArgs e)
    {
        MyTimer.Stop();
    }

    private void Button7_Click(object sender, RoutedEventArgs e)
    {
        CompositionTarget.Rendering += new
EventHandler(CompositionTarget_Rendering);
    }

    private void Button8_Click(object sender, RoutedEventArgs e)
    {
        CompositionTarget.Rendering -= new
EventHandler(CompositionTarget_Rendering);
    }
    void CompositionTarget_Rendering(object sender, EventArgs e)
    {
        Canvas.SetLeft(RedBall, xPos);
        if (xPos > this.Width)
            xPos = 0;
        else
            xPos += 0.5;
    }
}
```

Xaml Animation

MainWindow.xaml

```
<Canvas>
       <Ellipse Name="RedBall" Fill="Red" Width="100" Height="100"
Canvas.Left="132" Canvas.Top="62"/>
       <Button  Name="button1" Click="Button1_Click" Content="Button"
Canvas.Left="144" Canvas.Top="200" Width="75">
       <Button.Triggers>
           <EventTrigger RoutedEvent="Button.Click">
                  <BeginStoryboard Name="MyStoryBoard">
                      <Storyboard>
                          <!-- Width 프라퍼티는 직접, Canvas.Left 는 ()로
묶어야 한다. -->
                          <DoubleAnimation Storyboard.TargetName
="RedBall" Storyboard.TargetProperty="(Canvas.Left)"
                                  From="0" To="300"
Duration="0:0:15"></DoubleAnimation>
                      </Storyboard>
                  </BeginStoryboard>
              </EventTrigger>
           </Button.Triggers>
       </Button>

       <Button  Name="button2" Content="Button" Canvas.Left="144"
Canvas.Top="250" Width="75">
           <Button.Triggers>
               <EventTrigger RoutedEvent="Button.Click">
                   <StopStoryboard  BeginStoryboardName="MyStoryBoard"/>
               </EventTrigger>
           </Button.Triggers>
       </Button>
   </Canvas>
```

File Dialog #1

MainWindow.xaml

```
< Grid >
    < Button Content = "Button" HorizontalAlignment = "Left" Margin =
"145,114,0,0" VerticalAlignment = "Top" Width = "75" Click = "Button_Click" />
</ Grid >
```

MainWindow.xaml.cs

```
public partial class Ex1_FileDialog : Window
{
    public Ex1_FileDialog()
    {
        InitializeComponent();
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        // Configure open file dialog box
        Microsoft.Win32.OpenFileDialog dlg = new
Microsoft.Win32.OpenFileDialog();
        dlg.FileName = "Document"; // Default file name
        dlg.DefaultExt = ".txt"; // Default file extension
        dlg.Filter = "Text documents (.txt)|*.txt"; // Filter files by extension

        // Show open file dialog box
        Nullable<bool> result = dlg.ShowDialog();

        // Process open file dialog box results
        if (result == true)
        {
            // Open document
            string filename = dlg.FileName;
        }
    }
}
```

File Dialog #2

MainWindow.xaml

```
< DockPanel Margin = "10" >
    < WrapPanel HorizontalAlignment = "Center"
                DockPanel.Dock = "Top" Margin = "0,0,0,10" >
        < Button Name = "btnSaveFile"
                 Click = "btnSaveFile_Click" > Save file </ Button >
    </ WrapPanel >
    < TextBox Name = "txtEditor" TextWrapping = "Wrap"
              AcceptsReturn = "True"
              ScrollViewer.VerticalScrollBarVisibility = "Auto" />
</ DockPanel >
```

MainWindow.xaml.cs

```
public partial class SaveFileDialogSample : Window
{
    public SaveFileDialogSample()
    {
        InitializeComponent();
    }
    private void btnSaveFile_Click(object sender, RoutedEventArgs e)
    {
        SaveFileDialog saveFileDialog = new SaveFileDialog();

        saveFileDialog.Filter = "Text file (*.txt)|*.txt|C# file
(*.cs)|*.cs";
        saveFileDialog.InitialDirectory = @"c:\temp\";
        saveFileDialog.InitialDirectory =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments);
        if (saveFileDialog.ShowDialog() == true)
            File.WriteAllText(saveFileDialog.FileName, txtEditor.Text);
    }
}
```

Threading

MainWindow.xaml

```
< Grid >
    < Grid.RowDefinitions >
        < RowDefinition Height = "50*" />
        < RowDefinition Height = "50*" />
    </ Grid.RowDefinitions >
    < Button x: Name = "button" Grid.Row = "0"
        Width = "100"
        Height = "30"
        Content = "테스트" />
    < TextBox x: Name = "textBox" Grid.Row = "1"
        Width = "100"
        Height = "25"
        HorizontalContentAlignment = "Center"
        VerticalContentAlignment = "Center"
        BorderBrush = "Black"
        BorderThickness = "1" />
</ Grid >
```

MainWindow.xaml.cs

```
public partial class MainWindow : Window
{

    public MainWindow()
    {
        InitializeComponent();

        this.button.Click += button_Click;
    }

    private void button_Click(object sender, RoutedEventArgs e)
    {
        IsEnabled = false;

        Thread thread = new Thread(new ThreadStart(ProcessTest));

        thread.Start();
    }
    private void ProcessTest()
    {
        for (int i = 1; i <= 10; i++)
        {
            Thread.Sleep(100);
```

```
            DispatcherOperation dispatcherOperation =
Dispatcher.BeginInvoke
            (
                DispatcherPriority.Normal, new Action<string, int>(
                    (message, value) => {
                        this.textBox.Text = string.Format("{0} : {1}",
message, value);
                    }
                ),
                "Test",
                i
            );

            DispatcherOperationStatus dispatcherOperationStatus =
dispatcherOperation.Status;

            while (dispatcherOperationStatus !=
DispatcherOperationStatus.Completed)
            {
                dispatcherOperationStatus =
dispatcherOperation.Wait(TimeSpan.FromMilliseconds(1000));

                if (dispatcherOperationStatus ==
DispatcherOperationStatus.Aborted)
                {
                }
            }
        }
        Dispatcher.BeginInvoke(DispatcherPriority.Normal, new Action(()
=> { IsEnabled = true; }));
    }
}
```

Play Audio

MainWindow.xaml

```
< Window x: Class =
"WpfTutorialSamples.Audio_and_Video.SystemSoundsSample"
       xmlns =
"http://schemas.microsoft.com/winfx/2006/xaml/presentation"
       xmlns: x = "http://schemas.microsoft.com/winfx/2006/xaml"
       Title = "SystemSoundsSample" Height = "200" Width = "150" >
    < StackPanel Margin = "10" HorizontalAlignment = "Center"
                               VerticalAlignment = "Center" >
       <Button Name = "btnAsterisk" Click = "btnAsterisk_Click" >
                                          Asterisk </ Button >
       <Button Name = "btnBeep" Margin = "0,5" Click = "btnBeep_Click" >
                                    Beep </ Button >
       <Button Name = "btnExclamation" Click = "btnExclamation_Click" >
                                    Exclamation </ Button >
       <Button Name = "btnHand" Margin = "0,5" Click = "btnHand_Click" >
                                    Hand </ Button >
       < Button Name = "btnQuestion" Click = "btnQuestion_Click" >
                                    Question </ Button >
    </ StackPanel >
</ Window >
```

MainWindow.xaml.cs

```
public partial class SystemSoundsSample : Window
{
    public SystemSoundsSample()
    {
        InitializeComponent();
    }

    private void btnAsterisk_Click(object sender, RoutedEventArgs e)
    {
        SystemSounds.Asterisk.Play();
    }

    private void btnBeep_Click(object sender, RoutedEventArgs e)
    {
        SystemSounds.Beep.Play();
    }
```

```
    private void btnExclamation_Click(object sender, RoutedEventArgs e)
    {
        SystemSounds.Exclamation.Play();
    }

    private void btnHand_Click(object sender, RoutedEventArgs e)
    {
        SystemSounds.Hand.Play();
    }

    private void btnQuestion_Click(object sender, RoutedEventArgs e)
    {
        SystemSounds.Question.Play();
    }
}
```

Play Video

MainWindow.xaml

```
< Grid Margin = "10" >
    < Grid.RowDefinitions >
        < RowDefinition Height = "*" />
        < RowDefinition Height = "Auto" />
    </ Grid.RowDefinitions >
    < MediaElement Source = "test.mpg" LoadedBehavior = "Manual" Name =
"mePlayer" />
    < StackPanel Grid.Row = "1" >
        < Label Name = "lblStatus" Content = "Not playing..."
HorizontalContentAlignment = "Center" Margin = "5" />
        < WrapPanel HorizontalAlignment = "Center" >
            < Button Name = "btnPlay" Click = "btnPlay_Click" > Play </
Button >
            < Button Name = "btnPause" Margin = "5,0" Click =
"btnPause_Click" > Pause </ Button >
            < Button Name = "btnStop" Click = "btnStop_Click" > Stop </
Button >
        </ WrapPanel >
    </ StackPanel >
</ Grid >
```

MainWindow.xaml.cs

```
public partial class MediaPlayerVideoControlSample : Window
{
    public MediaPlayerVideoControlSample()
    {
        InitializeComponent();

        DispatcherTimer timer = new DispatcherTimer();
        timer.Interval = TimeSpan.FromSeconds(1);
        timer.Tick += timer_Tick;
        timer.Start();
    }

    void timer_Tick(object sender, EventArgs e)
    {
        if (mePlayer.Source != null)
        {
```

```
            if (mePlayer.NaturalDuration.HasTimeSpan)
                lblStatus.Content = String.Format("{0} / {1}",
mePlayer.Position.ToString(@"mm\:ss"),
mePlayer.NaturalDuration.TimeSpan.ToString(@"mm\:ss"));
        }
        else
            lblStatus.Content = "No file selected...";
    }

    private void btnPlay_Click(object sender, RoutedEventArgs e)
    {
        mePlayer.Play();
    }

    private void btnPause_Click(object sender, RoutedEventArgs e)
    {
        mePlayer.Pause();
    }

    private void btnStop_Click(object sender, RoutedEventArgs e)
    {
        mePlayer.Stop();
    }
}
```

Culture

MainWindow.xaml

```
< StackPanel Margin = "20" >
    < Grid >
        < Grid.RowDefinitions >
            < RowDefinition Height = "Auto" />
            < RowDefinition Height = "Auto" />
        </ Grid.RowDefinitions >
        < Grid.ColumnDefinitions >
            < ColumnDefinition Width = "*" />
            < ColumnDefinition Width = "*" />
        </ Grid.ColumnDefinitions >
        < Label > Number:</ Label >
        < Label Name = "lblNumber" Grid.Column = "1" />
        < Label Grid.Row = "1" > Date:</ Label >
        < Label Name = "lblDate" Grid.Row = "1" Grid.Column = "1" />
    </ Grid >
    < StackPanel Orientation = "Horizontal" HorizontalAlignment =
"Center" Margin = "0,20" >
        < Button Tag = "en-US" Click = "CultureInfoSwitchButton_Click"
HorizontalContentAlignment = "Stretch" > English(US) </ Button >
        < Button Tag = "de-DE" Click = "CultureInfoSwitchButton_Click"
HorizontalContentAlignment = "Stretch" Margin = "10,0" > German(DE) </
Button >
        < Button Tag = "sv-SE" Click = "CultureInfoSwitchButton_Click"
HorizontalContentAlignment = "Stretch" > Swedish(SE) </ Button >
    </ StackPanel >
</ StackPanel >
```

MainWindow.xaml.cs

```
public partial class ApplicationCultureSwitchSample : Window
{
    public ApplicationCultureSwitchSample()
    {
        InitializeComponent();
    }

    private void CultureInfoSwitchButton_Click(object sender,
RoutedEventArgs e)
    {
```

```
        Thread.CurrentThread.CurrentCulture = new CultureInfo((sender as
Button).Tag.ToString());
        lblNumber.Content = (123456789.42d).ToString("N2");
        lblDate.Content = DateTime.Now.ToString();
    }
}
```

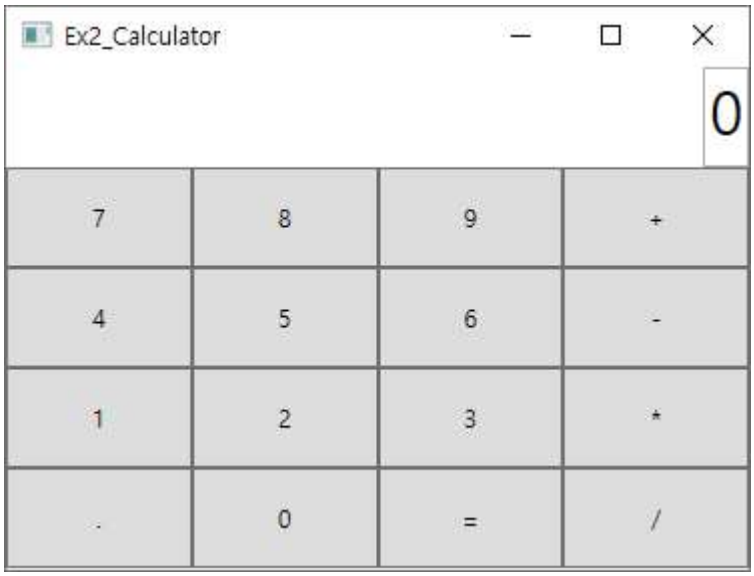
SECTION 10.

Example

❑ 주요 학습 내용

> Calculator
>
> Puzzle Game

Calculator



MainWindow.xaml

```
< StackPanel >
    < TextBox Height = "50" Name = "txtResult" FontSize = "30" HorizontalAlignment =
"Right" > 0 </ TextBox >
    < Grid Height = "200" >
        < Grid.ColumnDefinitions >
            < ColumnDefinition Width = "25*" />
            < ColumnDefinition Width = "25*" />
            < ColumnDefinition Width = "25*" />
            < ColumnDefinition Width = "25*" />
        </ Grid.ColumnDefinitions >
        < Grid.RowDefinitions >
            < RowDefinition Height = "25*" />
            < RowDefinition Height = "25*" />
            < RowDefinition Height = "25*" />
            < RowDefinition Height = "25*" />
        </ Grid.RowDefinitions >
        < Button Content = "7" Grid.Column = "0" Grid.Row = "0" Click = "num_click" />
        < Button Content = "4" Grid.Column = "0" Grid.Row = "1" Click = "num_click" />
        < Button Content = "1" Grid.Column = "0" Grid.Row = "2" Click = "num_click" />
        < Button Content = "." Grid.Column = "0" Grid.Row = "3" Click = "dot_click" />
        < Button Content = "8" Grid.Column = "1" Grid.Row = "0" Click = "num_click" />
        < Button Content = "5" Grid.Column = "1" Grid.Row = "1" Click = "num_click" />
        < Button Content = "2" Grid.Column = "1" Grid.Row = "2" Click = "num_click" />
        < Button Content = "0" Grid.Column = "1" Grid.Row = "3" Click = "num_click" />
        < Button Content = "9" Grid.Column = "2" Grid.Row = "0" Click = "num_click" />
        < Button Content = "6" Grid.Column = "2" Grid.Row = "1" Click = "num_click" />
        < Button Content = "3" Grid.Column = "2" Grid.Row = "2" Click = "num_click" />
        < Button Content = "=" Grid.Column = "2" Grid.Row = "3" Click ="equal_click" />
        < Button Content = "+" Grid.Column = "3" Grid.Row = "0" Click = "op_click" />
        < Button Content = "-" Grid.Column = "3" Grid.Row = "1" Click = "op_click" />
        < Button Content = "*" Grid.Column = "3" Grid.Row = "2" Click = "op_click" />
```

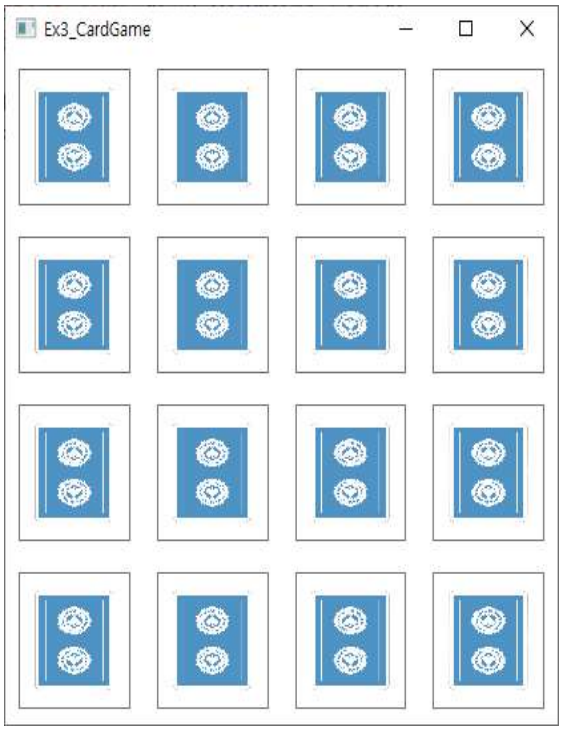
```
        < Button Content = "/" Grid.Column = "3" Grid.Row = "3" Click = "op_click" />
    </ Grid >
</ StackPanel >
```

MainWindow.xaml.cs

```
public partial class Ex2_Calculator : Window
{
    public Ex2_Calculator()
    {
        InitializeComponent();
    }

    private double savedValue = 0;
    private char op;
    private bool newButton = false;

    private void num_click(object sender, RoutedEventArgs e)
    {
        Button btn = sender as Button;

        string number = btn.Content.ToString();

        if (txtResult.Text == "0" || newButton == true)
        {
            txtResult.Text = number;
            newButton = false;
        }
        else
            txtResult.Text = txtResult.Text + number;
    }

    private void dot_click(object sender, RoutedEventArgs e)
    {
        if (txtResult.Text.Contains(".") == false)
            txtResult.Text += ".";
    }

    private void op_click(object sender, RoutedEventArgs e)
    {
        Button btn = sender as Button;

        savedValue = double.Parse(txtResult.Text);// string의 첫번째 요소
값
        op = btn.Content.ToString()[0];
        newButton = true;
    }

    private void equal_click(object sender, RoutedEventArgs e)
    {
        if (op == '+')
            txtResult.Text = (savedValue +
```

```
double.Parse(txtResult.Text)).ToString();
        else if (op == '-')
            txtResult.Text = (savedValue -
double.Parse(txtResult.Text)).ToString();
        else if (op == '*')
            txtResult.Text = (savedValue *
double.Parse(txtResult.Text)).ToString();
        else if (op == '/')
            txtResult.Text = (savedValue /
double.Parse(txtResult.Text)).ToString();
    }
}
```

Puzzle Game



MainWindow.xaml

```
<UniformGrid Name="board">
</UniformGrid>
```

MainWindow.xaml.cs

```
public partial class Ex3_CardGame : Window
{
    Button first;
    Button second;
    string[] strImageName = { "AH", "2H", "3H", "4H", "5H", "6H", "7H",
"8H" };
    DispatcherTimer myTimer = new DispatcherTimer();
    int matched = 0;

    Image[] imgCard = new Image[8];
    Image imgBack = null;

    public Ex3_CardGame()
    {
        InitializeComponent();
```

```
        boardSet();
        LoadImage();
        // 타이머 객체 생성
        myTimer.Interval = new TimeSpan(0, 0, 0, 0, 750); // 0.75초
        myTimer.Tick += MyTimer_Tick;
    }
    private void LoadImage()
    {
        imgBack = MakeImage("/Image/blue_back.jpg");

        for (int i = 0; i < 8; i++)
        {
            imgCard[i] = MakeImage("/Image/" + strImageName[i] + ".jpg");
        }
    }
    private void MyTimer_Tick(object sender, EventArgs e)
    {
        myTimer.Stop();

        first.Content = imgBack;
        second.Content = imgBack;
        first = null;
        second = null;
    }
    //
    private void boardSet()
    {
        for (int i = 0; i < 16; i++)
        {
            Button c = new Button();
            c.Background = Brushes.White;
            c.Margin = new Thickness(10);
            c.Content = MakeImage("/Image/blue_back.jpg");
            c.Tag = TagSet();   //  그림의 인덱스
            c.Click += Card_Click;
            board.Children.Add(c);
        }
    }
    private Image MakeImage(string v)
    {
        BitmapImage bi = new BitmapImage();
        bi.BeginInit();
        bi.UriSource = new Uri(v, UriKind.Relative);
        bi.EndInit();

        Image myImage = new Image();
        myImage.Margin = new Thickness(10);
        myImage.Stretch = Stretch.Fill;
        myImage.Source = bi;

        return myImage;
```

```
    }
    //--------------------------------
    // 0~7사이의 정수를 만들어 리턴하는 함수
    // 0~15사이의 랜덤값이 중복되지 않게 만들어지고
    // 이를 8로 나눈 나머지 값을 리턴합니다.

    int[] rnd = new int[16];  // 랜덤 숫자가 중복되는지 체크

    private int TagSet()
    {
        int i;
        Random r = new Random();
        while (true)
        {
            i = r.Next(16); // 0~15까지

            if (rnd[i] == 0)
            {
                rnd[i] = 1;
                break;
            }
        }
        return i % 8; // 태그는 0~7까지, 8개의 그림을 표시
    }
    private void Card_Click(object sender, RoutedEventArgs e)
    {
        Button btn = sender as Button;

        // 버튼에 따른 그림 Load
        string imgNo = strImageName[(int)btn.Tag];

        if (first == null) // 이 버튼은 첫번째 버튼
        {
            first = btn;
            btn.Content = MakeImage("/Image/" + imgNo + ".jpg");
            return;
        }
        else if (second == null)
        {
            second = btn;
            btn.Content = MakeImage("/Image/" + imgNo + ".jpg");
        }
        else   // 이미 두개의 버튼이 열렸는데 3번째 버튼을 눌렀을 경우, 아무 일
안하고 리턴
            return;

        // 매치가 되었을 때 (2)
        if ((int)first.Tag == (int)second.Tag)  // (3) 같다면
```

```
        {
            first = null;
            second = null;
            matched += 2;
            if (matched >= 16)
            {
                MessageBoxResult res = MessageBox.Show(
                       "성공! 다시 하시겠습니까?", "Success",
MessageBoxButton.YesNo);
                if (res == MessageBoxResult.Yes)
                {
                    resetRnd();
                    boardReset();
                    boardSet();
                    matched = 0;
                }
            }
        }
        // 매치가 안되었을 때는 다시 덮어주어야 한다 (4)
        else
        {
            myTimer.Start();
        }
    }
    // 게임 초기화
    private void boardReset()
    {
        board.Children.Clear();
    }
    // rnd[] 배열 초기화
    private void resetRnd()
    {
        for (int i = 0; i < 16; i++)
            rnd[i] = 0;
    }
}
```