//пример 1

<Window x:Class="WpfApplication2.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="MainWindow" Height="399.254" Width="269.775">

<Grid HorizontalAlignment="Left" Height="394" VerticalAlignment="Top" Width="268"

Margin="0,0,0,-2">

<TextBox x:Name="HeightBox" HorizontalAlignment="Left" Height="23" Margin="27,48,0,0" TextWrapping="Wrap" Text="TextBox" VerticalAlignment="Top" Width="199" TextChanged="HeightBox\_TextChanged"/>

<Label x:Name="HeightLabel" Content="Высота" HorizontalAlignment="Left" Margin="27,23,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="WidthBox" HorizontalAlignment="Left" Height="23" Margin="29,124,0,0" TextWrapping="Wrap" Text="TextBox" VerticalAlignment="Top" Width="197" TextChanged="WidthBox\_TextChanged"/>

<Label x:Name="WidthLabel" Content="Шырина" HorizontalAlignment="Left" Margin="27,100,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="AreaBox" HorizontalAlignment="Left" Height="23" Margin="29,208,0,0" TextWrapping="Wrap" Text="TextBox" VerticalAlignment="Top" Width="197"/>

<Label Content="Площадь" HorizontalAlignment="Left" Margin="29,187,0,0" VerticalAlignment="Top"/>

<Slider x:Name="FontSizeSlider" HorizontalAlignment="Left" Margin="26,265,0,0" VerticalAlignment="Top" Width="200"/>

<TextBlock x:Name="TextToChangeSize" HorizontalAlignment="Left" Margin="29,299,0,0" TextWrapping="Wrap" Text="Some Text" VerticalAlignment="Top"/>

</Grid>

</Window>

namespace WpfApplication2

{

class Rectangle

{

public Rectangle()

{ }

private double width;

private double height;

private double area;

public string Width

{

get { return Convert.ToString(width); }

set { width = Convert.ToDouble(value); }

}

public string Height

{

get { return Convert.ToString(height); }

set { height = Convert.ToDouble(value); }

}

public string Area

{

get

{

area = width \* height;

return Convert.ToString(area);

}

set

{

area = width \* height;

}

}

}

}

namespace WpfApplication2

{

class FontSizeValue

{

private int svalue;

public int Value

{

get

{

return Convert.ToInt32(50 \* Math.Sin(svalue));

}

set

{

svalue = value;

}

}

}

}

namespace WpfApplication2

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

Rectangle rect;

FontSizeValue fsv;

public MainWindow()

{

rect = new Rectangle();

fsv = new FontSizeValue();

Binding HeightBinding = new Binding();

HeightBinding.Path = new PropertyPath("Height");

HeightBinding.Mode = BindingMode.TwoWay;

HeightBinding.Source = rect;

HeightBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

Binding WidthBinding = new Binding();

WidthBinding.Path = new PropertyPath("Width");

WidthBinding.Mode = BindingMode.TwoWay;

WidthBinding.Source = rect;

WidthBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

Binding AreaBinding = new Binding();

AreaBinding.Path = new PropertyPath("Area");

AreaBinding.Mode = BindingMode.TwoWay;

AreaBinding.Source = rect;

AreaBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

Binding SliderBinding = new Binding();

SliderBinding.Path = new PropertyPath("Value");

SliderBinding.Mode = BindingMode.OneWayToSource;

SliderBinding.Source = fsv;

SliderBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

Binding FontSizeBinding = new Binding();

FontSizeBinding.Path = new PropertyPath("Value");

FontSizeBinding.Mode = BindingMode.OneWay;

FontSizeBinding.Source = fsv;

FontSizeBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

InitializeComponent();

HeightBox.SetBinding(TextBox.TextProperty, HeightBinding);

WidthBox.SetBinding(TextBox.TextProperty, WidthBinding);

AreaBox.SetBinding(TextBox.TextProperty, AreaBinding);

FontSizeSlider.SetBinding(Slider.ValueProperty, SliderBinding);

TextToChangeSize.SetBinding(TextBlock.FontSizeProperty, FontSizeBinding);

}

private void HeightBox\_TextChanged(object sender, TextChangedEventArgs e)

{

if (AreaBox != null) AreaBox.Text = "";

}

private void WidthBox\_TextChanged(object sender, TextChangedEventArgs e)

{

if (AreaBox != null) AreaBox.Text = "";

}

}

}

//пример 2

<Application x:Class="WpfApplication3.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<!-- StartupUri="View/MainWindowView.xaml"> -->

<Application.Resources>

</Application.Resources>

</Application>

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Threading.Tasks;

using System.Windows;

using WpfApplication3.ViewModel;

using WpfApplication3.View;

namespace WpfApplication3

{

/// <summary>

/// Interaction logic for App.xaml

/// </summary>

public partial class App : Application

{

public App()

{

var mw = new MainWindowView

{

DataContext = new MainWindowViewModel()

};

mw.Show();

}

}

}

<Window x:Class="WpfApplication3.View.MainWindowView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="MainWindow" Height="167.91" Width="399.627">

<Grid>

<TextBlock HorizontalAlignment="Left" VerticalAlignment="Top" Margin="12,12,0,0"

Text="First name"/>

<TextBlock HorizontalAlignment="Left" VerticalAlignment="Top" Margin="12,52,0,0"

Text="Last name"/>

<TextBox HorizontalAlignment="Left" VerticalAlignment="Top" Margin="90,9,0,0" Width="120"

Text="{Binding Path=People.FirstName, Mode=TwoWay}"/>

<TextBox HorizontalAlignment="Left" VerticalAlignment="Top" Margin="90,49,0,0" Width="120"

Text="{Binding Path=People.LastName, Mode=TwoWay}"/>

<TextBox HorizontalAlignment="Left" VerticalAlignment="Top" Margin="246,9,0,0" Width="120"

Text="{Binding Path=People.FirstName, Mode=TwoWay}" IsReadOnly="True" />

<TextBox HorizontalAlignment="Left" VerticalAlignment="Top" Margin="246,49,0,0" Width="120"

Text="{Binding Path=People.LastName, Mode=TwoWay}" IsReadOnly="True" />

<Button Content="Click me" HorizontalAlignment="Left" Margin="291,97,0,0" VerticalAlignment="Top" Width="75" Command="{Binding ClickCommand}" />

</Grid>

</Window>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using WpfApplication3.Model;

using System.Windows.Input;

namespace WpfApplication3.ViewModel

{

class MainWindowViewModel

{

#region Constructor

/// <summary>

/// Constructor.

/// </summary>

public MainWindowViewModel()

{

ClickCommand = new Command(arg => ClickMethod());

People = new PeopleModel

{

FirstName = "First name",

LastName = "Last name"

};

}

#endregion

#region Properties

/// <summary>

/// Get or set people.

/// </summary>

public PeopleModel People { get; set; }

#endregion

#region Commands

/// <summary>

/// Get or set ClickCommand.

/// </summary>

public ICommand ClickCommand { get; set; }

#endregion

#region Methods

/// <summary>

/// Click method.

/// </summary>

private void ClickMethod()

{

MessageBox.Show("This is click command.");

}

#endregion

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel;

namespace WpfApplication3.Model

{

class PeopleModel : INotifyPropertyChanged

{

#region Implement INotyfyPropertyChanged members

public event PropertyChangedEventHandler PropertyChanged;

protected virtual void OnPropertyChanged(string propertyName)

{

if (PropertyChanged != null)

{

PropertyChanged(this, new PropertyChangedEventArgs(propertyName));

}

}

#endregion

#region Fields

private string \_FirstName;

private string \_LastName;

#endregion

#region Properties

/// <summary>

/// Get or set first name.

/// </summary>

public string FirstName

{

get { return \_FirstName; }

set

{

if (\_FirstName != value)

{

\_FirstName = value;

OnPropertyChanged("FirstName");

}

}

}

/// <summary>

/// Get or set last name.

/// </summary>

public string LastName

{

get { return \_LastName; }

set

{

if (\_LastName != value)

{

\_LastName = value;

OnPropertyChanged("LastName");

}

}

}

#endregion

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Input;

namespace WpfApplication3.ViewModel

{

class Command : ICommand

{

#region Constructor

public Command(Action<object> action)

{

ExecuteDelegate = action;

}

#endregion

#region Properties

public Predicate<object> CanExecuteDelegate { get; set; }

public Action<object> ExecuteDelegate { get; set; }

#endregion

#region ICommand Members

public bool CanExecute(object parameter)

{

if (CanExecuteDelegate != null)

{

return CanExecuteDelegate(parameter);

}

return true;

}

public event EventHandler CanExecuteChanged

{

add { CommandManager.RequerySuggested += value; }

remove { CommandManager.RequerySuggested -= value; }

}

public void Execute(object parameter)

{

if (ExecuteDelegate != null)

{

ExecuteDelegate(parameter);

}

}

#endregion

}

}