**КОД ПРИЛОЖЕНИЯ**

**MainViewModels**

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Diagnostics;  using System.Linq;  using System.Runtime.CompilerServices;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Media;  using System.Xml.Linq;  namespace Paramonov\_Prakt1.ViewModels  {  internal class MainViewModel : INotifyPropertyChanged  {  public event PropertyChangedEventHandler? PropertyChanged;  public void OnPropertyChanged([CallerMemberName] string name = "")  {  PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));  }  private byte \_R;  public byte R  {  get  {  return \_R;  }  set  {  \_R = value;  OnPropertyChanged();  UpdateColor();  }  }  private byte \_G;  public byte G  {  get  {  return \_G;  }  set  {  \_G = value;  OnPropertyChanged();  UpdateColor();  }  }  private byte \_B;  public byte B  {  get  {  return \_B;  }  set  {  \_B = value;  OnPropertyChanged();  UpdateColor();  }  }  private Brush? \_brush;  public Brush Brush  {  get  {  return \_brush;  }  set  {  \_brush = value;  OnPropertyChanged();  }  }  public void UpdateColor()  {  Brush = new SolidColorBrush(Color.FromRgb(\_R, \_G, \_B));  }  public DelegateCommand UpdateClipboardCommand  {  get  {  return new DelegateCommand(O => { UpdateClipboard(); });  }  }  private void UpdateClipboard()  {  Clipboard.SetText(Color.FromRgb(\_R, \_G, \_B).ToString());  }  }  } |

**MainWindow.xaml**

|  |
| --- |
| <Window x:Class="Paramonov\_Prakt1.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:Paramonov\_Prakt1"  xmlns:vm="clr-namespace:Paramonov\_Prakt1.ViewModels"  mc:Ignorable="d"  Title="MainWindow" Height="450" Width="800">  <Window.DataContext>  <vm:MainViewModel/>  </Window.DataContext>  <StackPanel Margin="57,0,57,-6">  <StackPanel.Background>  <ImageBrush ImageSource="/1671285025\_new\_preview\_48001b7a8907be5cf675cbb33ec298fa.jpg"/>  </StackPanel.Background>  <Label Content="R" Width="20"/>  <Label Content="{Binding R}" Width="20"/>  <Button Command="{Binding UpdateClipboardCommand}" Width="100" Height="100"/>  <Label Content="G" Width="20"/>  <Label Background="{Binding Brush}" Width="545" Height="63"/>  <Label Content="{Binding G}" Width="20"/>  <Slider Value="{Binding R}" Maximum="255" Width="558"/>  <Label Content="B" Width="20"/>  <Slider Value="{Binding G}" Maximum="255" Width="510" RenderTransformOrigin="0.498,0.676"/>  <Label Content="{Binding B}" Width="20"/>  <Slider Value="{Binding B}" Maximum="255" Width="460"/>  </StackPanel>  </Window> |

**MainWindow.xaml.cs**

|  |
| --- |
| using System.Text;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace Paramonov\_Prakt1  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public MainWindow()  {  InitializeComponent();  }  private void Slider\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)  {  }  }  } |

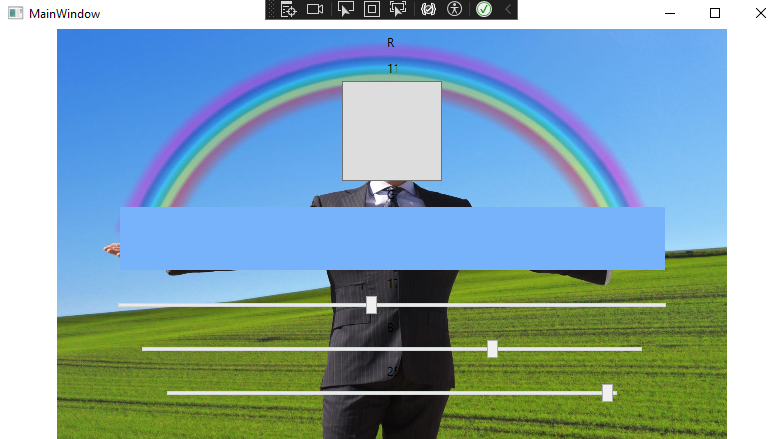
**DelegateCommand**

|  |
| --- |
| using System;  using System.Windows.Input;  namespace Paramonov\_Prakt1  {  public class DelegateCommand : ICommand  {  private Action<object> execute;  private Func<object, bool> canExecute;  public event EventHandler CanExecuteChanged  {  add { CommandManager.RequerySuggested += value; }  remove { CommandManager.RequerySuggested -= value; }  }  public DelegateCommand(Action<object> execute, Func<object, bool> canExecute = null)  {  this.execute = execute;  this.canExecute = canExecute;  }  public bool CanExecute(object parameter)  {  return this.canExecute == null || this.canExecute(parameter);  }  public void Execute(object parameter)  {  this.execute(parameter);  }  }  } |

**AssemblyInfo**

|  |
| --- |
| using System.Windows;  [assembly: ThemeInfo(  ResourceDictionaryLocation.None, //where theme specific resource dictionaries are located  //(used if a resource is not found in the page,  // or application resource dictionaries)  ResourceDictionaryLocation.SourceAssembly //where the generic resource dictionary is located  //(used if a resource is not found in the page,  // app, or any theme specific resource dictionaries)  )] |

**РАБОТОСПОСОБНОСТЬ**

****