ПРИЛОЖЕНИЕ 1. ШАБЛОН ОТЧЕТА

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Казанский национальный исследовательский технический университет

им. А.Н. Туполева – КАИ»

Институт компьютерных технологий и защиты информации

Отделение СПО ИКТЗИ (Колледж информационных технологий)

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине

Разработка программных модулей

Тема: «Основные элементы управления WPF»

Работу выполнил

Студент гр.4232

Салахов К.И

Принял

Преподаватель Лоповок Е.Е

Казань 2024

**Цель**

Научиться пользоваться основными элементами управления WPF.

**Задания на лабораторную работу**

Задание 1:

Разработать WPF-приложение с меню, панелью инструментов и строкой состояния. С помощью пунктов меню пользователь может изменять цвет фона окна, получить информацию о разработчике, а также закрыть окно. Кнопки панели инструментов дублируют команды меню. При наведении на пункты меню или кнопки панели инструментов в строке состояния отображается информация об этих элементах управления.

Задание 2:

Разработать WPF-приложение «Графический редактор» с выпадающим списком для выбора цвета кисти, ползунком для выбора размеров кисти и зависимыми переключателями для выбора режима работы: «рисование», «редактирование», «удаление».

**Результат выполнения работы**

Задание 1(Рисунок 1-6):

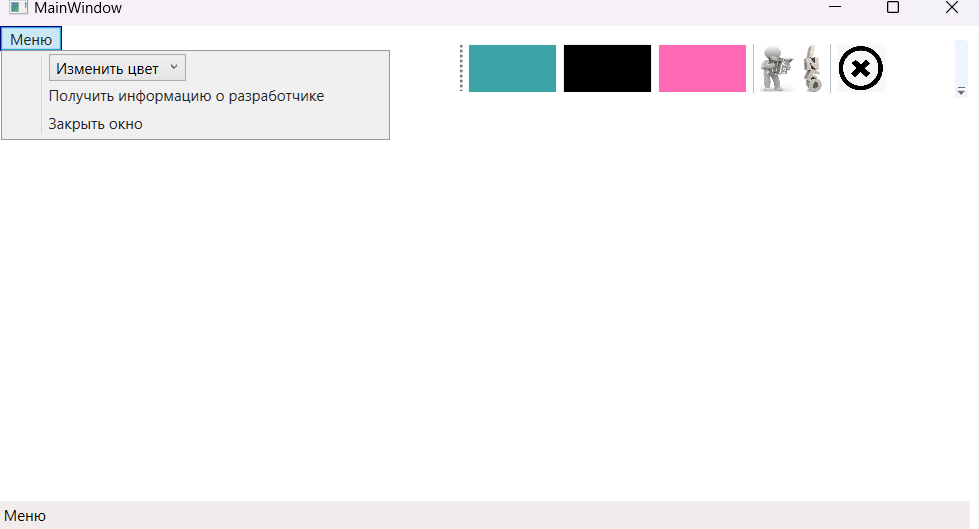
****

Рисунок 1 - Меню

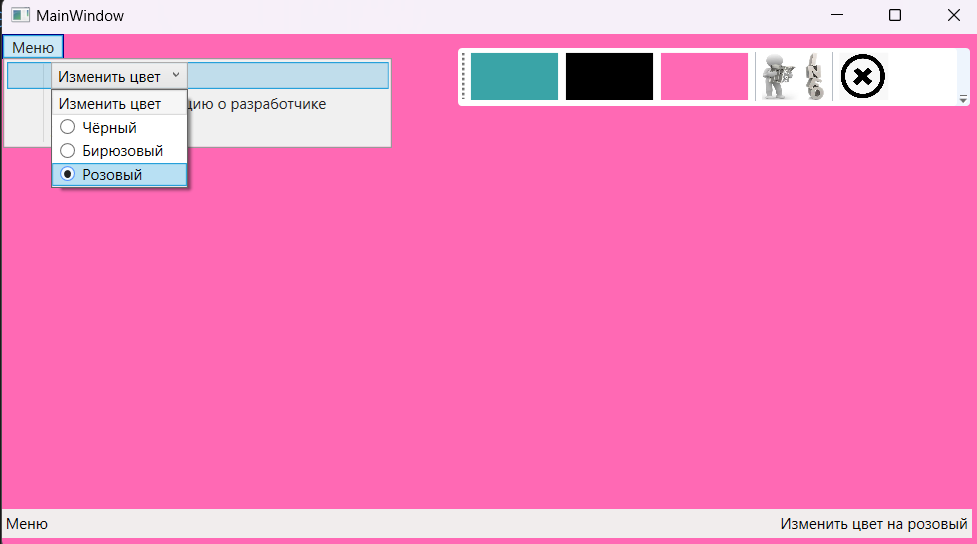
****

Рисунок 2 – Изменение цвета через меню

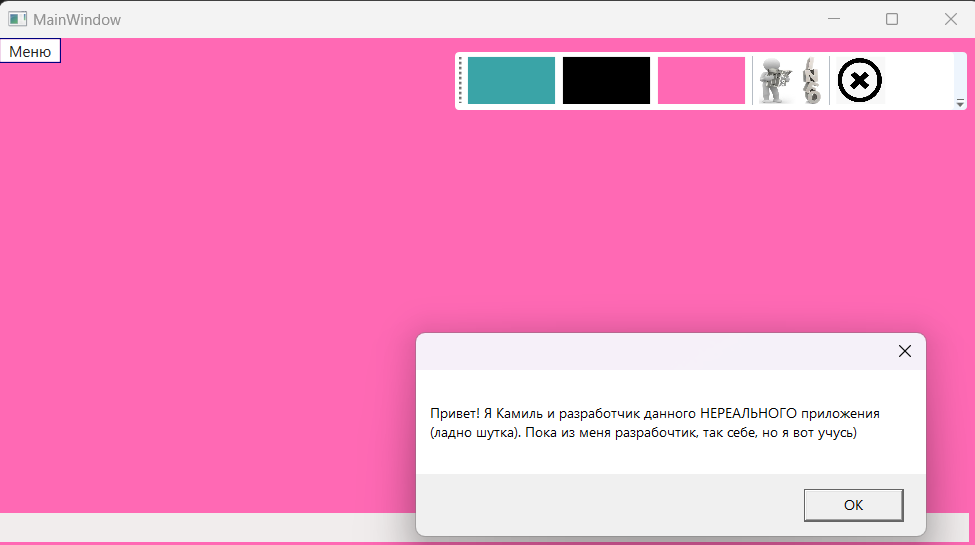
****

Рисунок 3 – Информация разработчика через меню

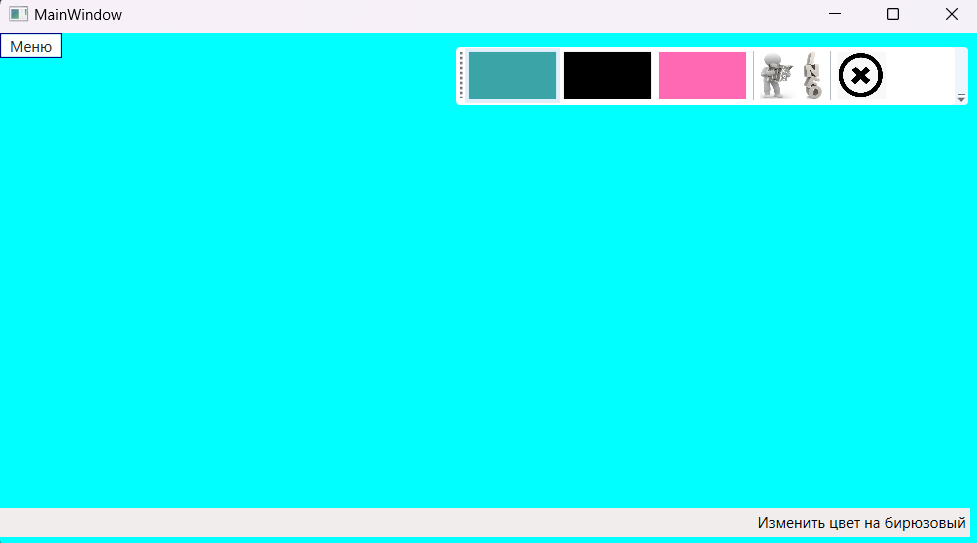
****

Рисунок 4 – Изменение цвета через панель инструментов

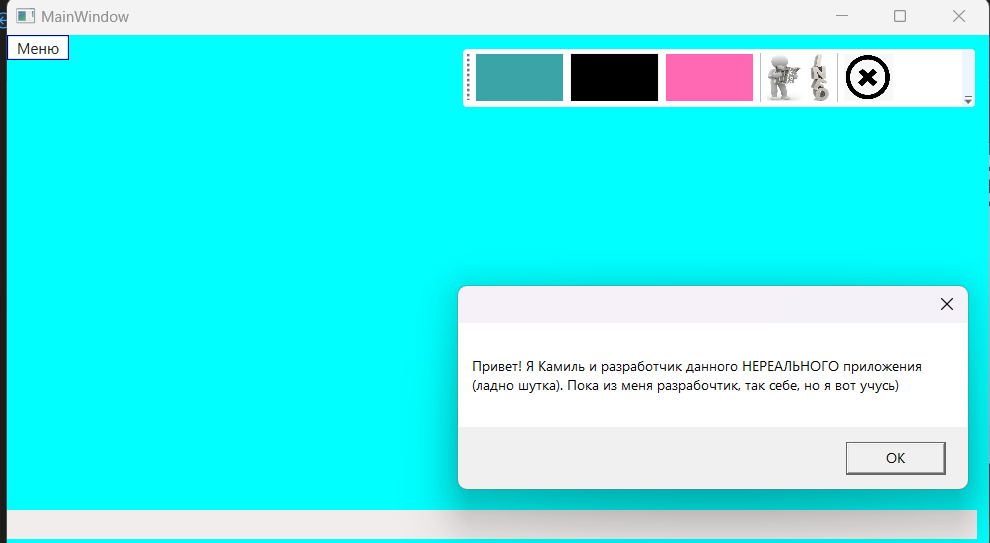
****

Рисунок 5 – Информация разработчика через панель инструментов

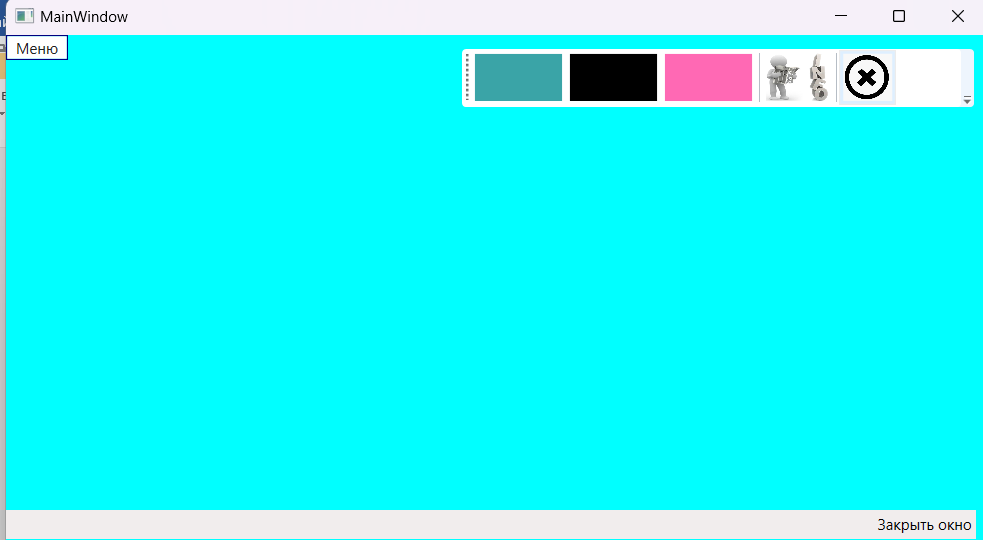
****

Рисунок 6 – Закрытие окна

Задание 2 (Рисунок 7-13):

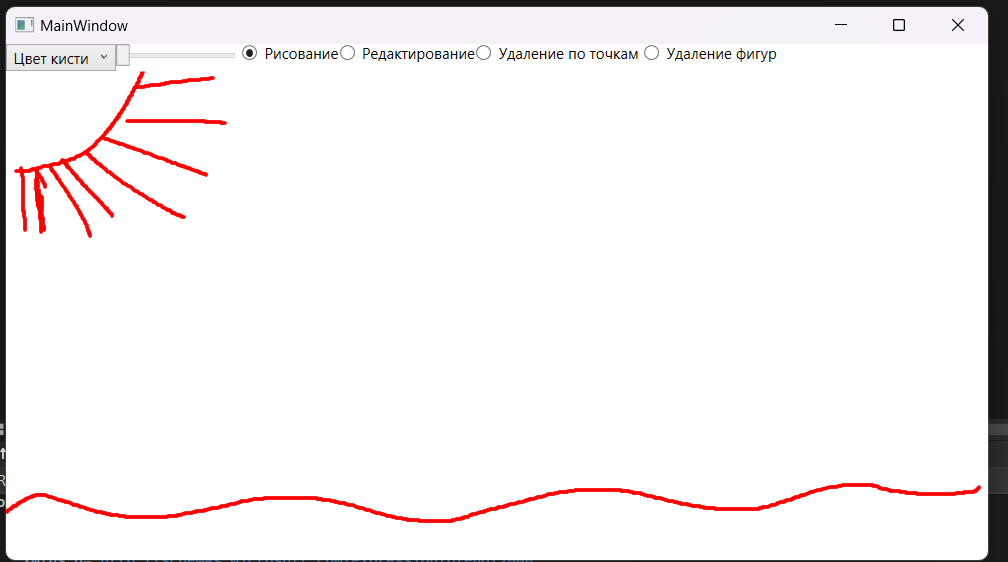


Рисунок 7 – Режим “рисование”



Рисунок 8 – Выбираем в выпадающем списке другой цвет кисти

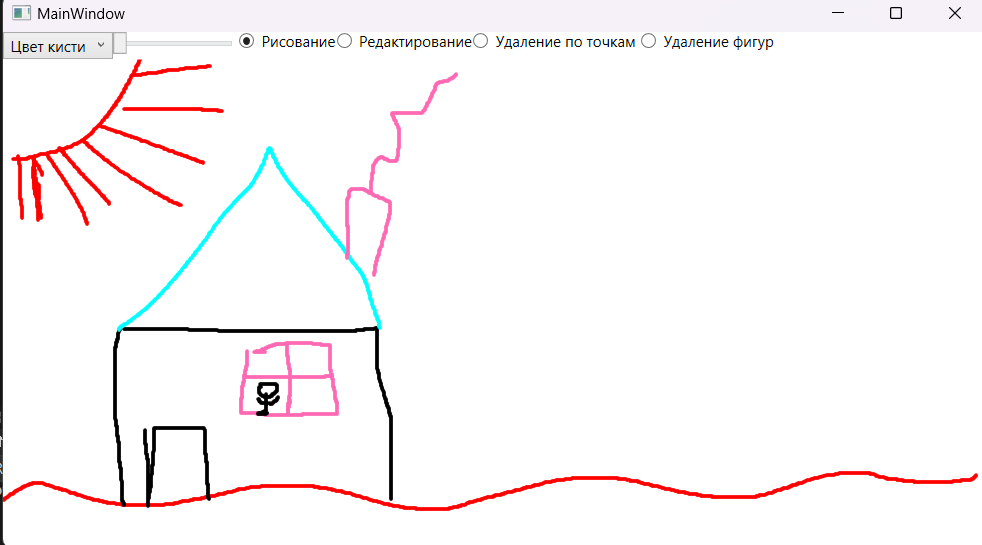


Рисунок 9 – Рисование разными цветами кисти



Рисунок 10 – Изменение размера кисти с помощью ползунка

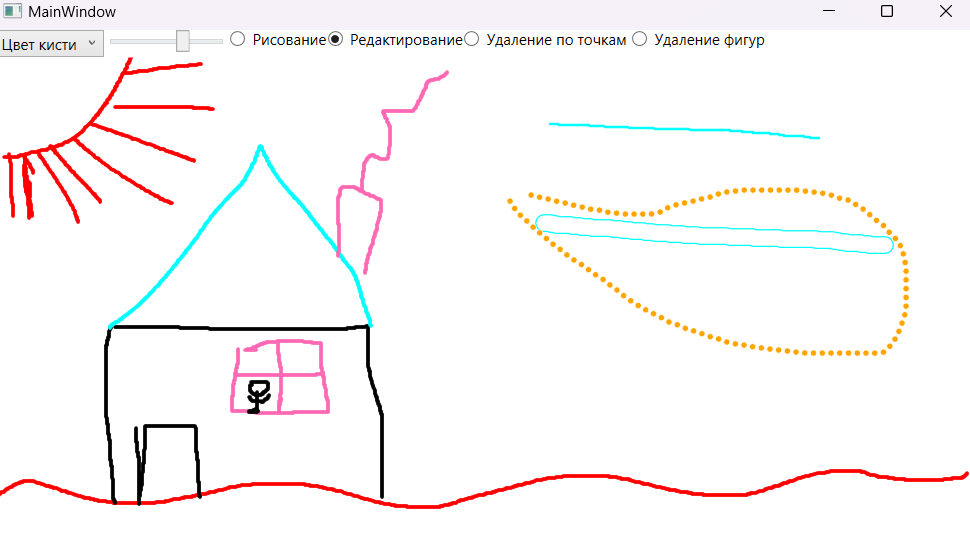


Рисунок 11 – Режим “Редактирование”



Рисунок 12 – Режим “Удаление по точкам”



Рисунок 11 – Режим “Удаление фигур”

**Листинг**

**Задание 1:**

Xaml:

<Window x:Class="РПМ\_3.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:РПМ\_3"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800">

<Grid>

<WrapPanel>

<Menu Background="White" BorderBrush="Navy" BorderThickness="1" Grid.ColumnSpan="2">

<MenuItem Header="\_Меню" MouseMove="MenuItem\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<ComboBox SelectedIndex="0">

<ComboBoxItem Content="Изменить цвет" MouseMove="Color\_changed\_MouseMove" MouseLeave="STROKA\_MouseLeave"/>

<RadioButton Content="Чёрный" GroupName="color" x:Name="color1" Click="color\_Click" MouseMove="black\_MouseMove" MouseLeave="STROKA\_MouseLeave"></RadioButton>

<RadioButton Content="Бирюзовый" GroupName="color" x:Name="color2" Click="color\_Click" MouseMove="turquoise\_MouseMove" MouseLeave="STROKA\_MouseLeave"></RadioButton>

<RadioButton Content="Розовый" GroupName="color" x:Name="color3" Click="color\_Click" MouseMove="Hot\_Pink\_MouseMove" MouseLeave="STROKA\_MouseLeave"></RadioButton>

</ComboBox>

<MenuItem Header="\_Получить информацию о разработчике" Click="MenuItem\_Click\_1" MouseMove="Info\_MouseMove" MouseLeave="STROKA\_MouseLeave"/>

<MenuItem Header="\_Закрыть окно" Click="MenuItem\_Click\_2" MouseLeave="STROKA\_MouseLeave" MouseMove="Close\_MouseMove"/>

</MenuItem>

</Menu>

</WrapPanel>

<ToolBar Margin="362,10,10,354" Background="White">

<Button Click="Button\_Click" MouseMove="turquoise\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<Image Source="C:\Users\Камиль\OneDrive\Изображения\Снимки экрана\бирюзовый.png" Height="70" Width="70"></Image>

</Button>

<Button Click="Button\_Click\_1" MouseMove="black\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<Image Source="C:\Users\Камиль\OneDrive\Изображения\Снимки экрана\Черный.png" Height="70" Width="70"></Image>

</Button>

<Button Click="Button\_Click\_2" MouseMove="Hot\_Pink\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<Image Source="C:\Users\Камиль\OneDrive\Изображения\Снимки экрана\Хот пинк.png" Height="70" Width="70"></Image>

</Button>

<Separator/>

<Button Click="MenuItem\_Click\_1" MouseMove="Info\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<Image Source="C:\Users\Камиль\OneDrive\Изображения\Снимки экрана\инфа.png"></Image>

</Button>

<Separator/>

<Button Click="MenuItem\_Click\_2" MouseMove="Close\_MouseMove" MouseLeave="STROKA\_MouseLeave">

<Image Source="C:\Users\Камиль\OneDrive\Изображения\Снимки экрана\закрыть.png"></Image>

</Button>

</ToolBar>

<StatusBar DockPanel.Dock="Bottom" Margin="0,380,10,10">

<TextBlock x:Name="str\_left" Text="" />

<StatusBarItem HorizontalAlignment="Right" >

<TextBlock x:Name="stroka" Text=""/>

</StatusBarItem>

</StatusBar>

</Grid>

</Window>

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 РПМ\_3

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void MenuItem\_Click\_1(object sender, RoutedEventArgs e)

{

MessageBox.Show("Привет! Я Камиль и разработчик данного НЕРЕАЛЬНОГО приложения (ладно шутка). Пока из меня разрабочтик, так себе, но я вот учусь)");

}

private void MenuItem\_Click\_2(object sender, RoutedEventArgs e)

{

Close();

}

private void color\_Click(object sender, RoutedEventArgs e)

{

if (color1.IsChecked == true)

{

this.Background = Brushes.Black;

}

else if (color2.IsChecked == true)

{

this.Background = Brushes.Aqua;

}

else if (color3.IsChecked == true)

{

this.Background = Brushes.HotPink;

}

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

this.Background=Brushes.Aqua;

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

this.Background = Brushes.Black;

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

this.Background = Brushes.HotPink;

}

private void STROKA\_MouseLeave(object sender, MouseEventArgs e)

{

this.stroka.Text = "";

this.str\_left.Text = "";

}

private void menu\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Меню";

}

private void turquoise\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Изменить цвет на бирюзовый";

}

private void Color\_changed\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Изменить цвет";

}

private void MenuItem\_MouseMove(object sender, MouseEventArgs e)

{

this.str\_left.Text = "Меню";

}

private void black\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Изменить цвет на чёрный";

}

private void Hot\_Pink\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Изменить цвет на розовый";

}

private void Info\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Получить информацию от разработчика";

}

private void Close\_MouseMove(object sender, MouseEventArgs e)

{

this.stroka.Text = "Закрыть окно";

}

}

}

**Задание 2:**

Xaml:

<Window x:Class="РПМ3\_2\_ЗАДАНИЕ\_.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:РПМ3\_2\_ЗАДАНИЕ\_"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal" Grid.Row="0">

<ComboBox SelectedIndex="0">

<ComboBoxItem Content="Цвет кисти" />

<RadioButton x:Name = "Black" Content="Чёрный" Click="Black\_Click"></RadioButton>

<RadioButton x:Name="Turquoise" Content="Бирюзовый" Click="Turquoise\_Click"></RadioButton>

<RadioButton x:Name="Pink" Content="Розовый" Click="Pink\_Click"></RadioButton>

</ComboBox>

<Slider x:Name="brushSizeSlider" Minimum="1" Maximum="20" Width="100" ValueChanged="brushSizeSlider\_ValueChanged"/>

<RadioButton GroupName="mode" Content="Рисование" IsChecked="True" x:Name="drawRadioButton" Click="drawRadioButton\_Click"/>

<RadioButton GroupName="mode" Content="Редактирование" x:Name="SelectRadioButton" Click="SelectButton\_Click"/>

<RadioButton GroupName="mode" Content="Удаление по точкам " x:Name="EraseByPointRadioButton" Click="EraseByPoint\_Click"/>

<RadioButton GroupName="mode" Content="Удаление фигур" x:Name="EraseByStrokeRadioButton" Click="EraseByStrokeRadioButton\_Click"/>

</StackPanel>

<InkCanvas Margin="0,0,0,10" Grid.Row="1" x:Name="inkCanvas">

<InkCanvas.DefaultDrawingAttributes>

<DrawingAttributes Color="Red" Height="3" Width="3"/>

</InkCanvas.DefaultDrawingAttributes>

</InkCanvas>

</Grid>

</Window>

Xaml.cs:

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Ink;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace РПМ3\_2\_ЗАДАНИЕ\_

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Black\_Click(object sender, RoutedEventArgs e)

{

SolidColorBrush brush = new SolidColorBrush(Colors.Black);

inkCanvas.DefaultDrawingAttributes.Color = brush.Color;

}

private void Turquoise\_Click(object sender, RoutedEventArgs e)

{

SolidColorBrush brush = new SolidColorBrush(Colors.Aqua);

inkCanvas.DefaultDrawingAttributes.Color = brush.Color;

}

private void Pink\_Click(object sender, RoutedEventArgs e)

{

SolidColorBrush brush = new SolidColorBrush(Colors.HotPink);

inkCanvas.DefaultDrawingAttributes.Color = brush.Color;

}

private void brushSizeSlider\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

{

if (inkCanvas != null)

{

inkCanvas.DefaultDrawingAttributes.Height = brushSizeSlider.Value;

inkCanvas.DefaultDrawingAttributes.Width = brushSizeSlider.Value;

}

}

private void EraseByPoint\_Click(object sender, RoutedEventArgs e)

{

if (inkCanvas != null)

{

inkCanvas.EditingMode = InkCanvasEditingMode.EraseByPoint;

}

}

private void EraseByStrokeRadioButton\_Click(object sender, RoutedEventArgs e)

{

if (inkCanvas != null)

{

inkCanvas.EditingMode = InkCanvasEditingMode.EraseByStroke;

}

}

private void SelectButton\_Click(object sender, RoutedEventArgs e)

{

if (inkCanvas != null)

{

inkCanvas.EditingMode = InkCanvasEditingMode.Select;

}

}

private void drawRadioButton\_Click(object sender, RoutedEventArgs e)

{

if (inkCanvas != null)

{

inkCanvas.EditingMode = InkCanvasEditingMode.Ink;

}

}

}

}