УО «Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

По дисциплине: “Технологии разработки интерфейса программных средств„

Тема: «Использование стилей в WPF-приложениях».

Вариант 6

Выполнил: Комиссаров А. Е.

Брест 2021 Группа ПО-7, 2 курс.

Проверила: Войцехович О.Ю.

Цель: Ознакомиться с принципами использования стилей в WPF-приложениях.

Ход работы:

**Задание 1.**  Проверьте, какое значение имеет больший приоритет: значение свойства, указанное в стиле, или значение атрибута элемента.

Содержимое файла MainWindow.xaml:

<Window.Resources>

<Style TargetType="Button" x:Key="DocButton">

**<Setter Property="Background" Value="DarkCyan" />**

</Style>

</Window.Resources>

<StackPanel Orientation="Horizontal" VerticalAlignment="Top">

<Button Style="{StaticResource DocButton}" **Background="Black**"> Что главнее? </Button>

</StackPanel>

При запуске данного кода в окне отображается чёрная кнопка, следовательно, атрибуты элемента важнее его свойств, указанных в стиле.

Результат работы программы:

Рис. 1 – Задание 1.

**Задание 2.**  Модифицируйте WPF-приложение, разработанное в 3-ей лабораторной работе: используйте стили для однотипных элементов управления.

Содержимое файла MainWindow.xaml:

<Window.Resources>

<x:Array x:Key="MyEditingModes" Type="{x:Type InkCanvasEditingMode}">

<x:Static Member="InkCanvasEditingMode.Ink"/>

<x:Static Member="InkCanvasEditingMode.EraseByPoint"/>

<x:Static Member="InkCanvasEditingMode.EraseByStroke"/>

</x:Array>

<x:Array x:Key="MyDrawingAttributes" Type="{x:Type DrawingAttributes}">

<DrawingAttributes Color="Red" Width="2"/>

<DrawingAttributes Color="Green" Width="2"/>

<DrawingAttributes Color="Blue" Width="2"/>

</x:Array>

<Style x:Key="MenuBorder" TargetType="Menu">

<Setter Property="Background" Value="DarkBlue"/>

</Style>

<Style x:Key="BrushColors" TargetType="MenuItem">

<Setter Property="BorderBrush" Value="Black"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="FontStyle" Value="Italic"/>

<Setter Property="FontSize" Value="15"/>

<Setter Property="Background" Value="Cyan"/>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Foreground" Value="White"/>

</Trigger>

</Style.Triggers>

</Style>

<Style BasedOn="{StaticResource BrushColors}" x:Key="BrushThickness" TargetType="MenuItem">

<Setter Property="FontStyle" Value="Normal"/>

<Setter Property="FontSize" Value="13"/>

<Setter Property="Background" Value="DarkCyan"/>

</Style>

<Style BasedOn="{StaticResource BrushColors}" x:Key="BrushMode" TargetType="MenuItem">

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontSize" Value="10"/>

<Setter Property="Background" Value="BlueViolet"/>

</Style>

<Style x:Key="BrushThicknessItem" TargetType="Slider">

<Setter Property="Background" Value="DarkOrange"/>

</Style>

<Style x:Key="BrushModeItem" TargetType="ListBox">

<Setter Property="Background" Value="Gold"/>

</Style>

</Window.Resources>

<Grid>

<InkCanvas Height="420" x:Name="Drawboard" DefaultDrawingAttributes="{Binding ElementName=lbColors, Path=SelectedValue}" EditingMode="{Binding ElementName=lbEditingModes, Path=SelectedValue}"/>

<Menu Style="{StaticResource MenuBorder}" Height="22" VerticalAlignment="Top" HorizontalAlignment="Stretch" >

<MenuItem Header="\_Цвет кисти" ToolTip="Выбор цвета." Style="{StaticResource BrushColors}">

<ListBox x:Name="lbColors" ItemsSource="{StaticResource MyDrawingAttributes}">

<ListBox.ItemTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=Color}">

<TextBlock.Background>

<SolidColorBrush Color="{Binding Path=Color}"/>

</TextBlock.Background>

</TextBlock>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

</MenuItem>

<MenuItem Style="{StaticResource BrushThickness}" Header="\_Толщина кисти">

<Slider Name="Slider111" Height="25" Width="100" Minimum="1" Maximum="15" Value="2" ValueChanged="Slider111\_ValueChanged" Style="{StaticResource BrushThicknessItem}"/>

</MenuItem>

<MenuItem Style="{StaticResource BrushMode}" Header="\_Режим работы">

<ListBox x:Name="lbEditingModes" ItemsSource="{StaticResource MyEditingModes}" Style="{StaticResource BrushModeItem}"/>

</MenuItem>

</Menu>

</Grid>

Содержимое файла MainWindow.xaml.CS:

private void Slider111\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

{

Drawboard.DefaultDrawingAttributes.Width = Slider111.Value;

Drawboard.DefaultDrawingAttributes.Height = Slider111.Value;

}

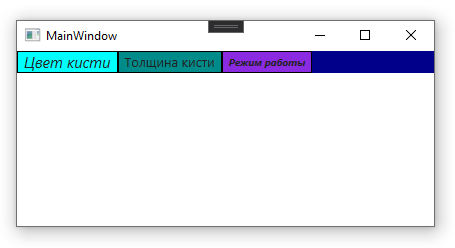
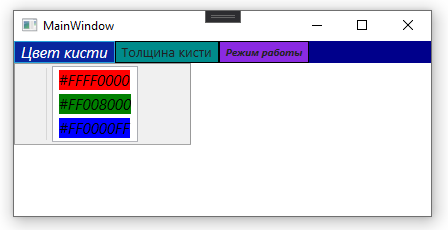
Результат работы программы:

Рис. 2 – Открытие программы

При открытии программы стили можно заметить сразу – меню имеет задний фон цвета “DarkBlue”, кнопки Цвет кисти – Толщина кисти – Режим работы “Cyan” – “DarkCyan” – “BlueViolet” соответственно.

Рис. 3 – Подпункты меню

Подпункты меню также имеют свои собственные стили.

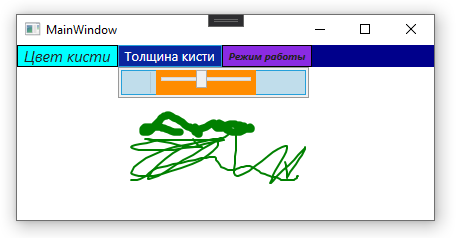


Рис. 4 – Работа программы, подпункты меню

**Задание 3.**  Разработайте приложение MultiEdit для одновременной работы с несколькими текстами. Окно должно быть разделено на две части с одинаковыми градиентами. В каждой части окна должно быть несколько многострочных текстовых полей: одно из них большого размера с крупным шрифтом, а остальные маленького размера с мелким шрифтом. То текстовое окно, в котором пользователь набирает текст, должно быть большим, остальные текстовые поля должны быть маленькими. Внешний вид однотипных элементов управления должен определяться с помощью стилей.

Содержимое файла MainWindow.xaml:

<Window.Resources>

<Style TargetType="TextBox" x:Key="SmallTextBox">

<Setter Property="TextWrapping" Value="Wrap"/>

<Setter Property="AcceptsReturn" Value="True"/>

<Setter Property="AcceptsTab" Value="True"/>

<Setter Property="Margin" Value="12"/>

<Setter Property="Height" Value="72"/>

<Setter Property="FontSize" Value="10"/>

<Setter Property="HorizontalAlignment" Value="Stretch"/>

<Style.Triggers>

<Trigger Property="IsFocused" Value="True">

<Setter Property="Margin" Value="8"/>

<Setter Property="Height" Value="110"/>

<Setter Property="FontSize" Value="18"/>

</Trigger>

</Style.Triggers>

</Style>

</Window.Resources>

<Grid>

<Grid.RowDefinitions>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<!-- Левый столбец -->

<Rectangle Grid.Column="0">

<Rectangle.Fill>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<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>

</Rectangle.Fill>

</Rectangle>

<TextBox Grid.Column="0" VerticalAlignment="Top" Style="{StaticResource SmallTextBox}">DefaultText (Top text)</TextBox>

<TextBox Grid.Column="0" VerticalAlignment="Center" Style="{StaticResource SmallTextBox}">DefaultText (Center text)</TextBox>

<TextBox Grid.Column="0" VerticalAlignment="Bottom" Style="{StaticResource SmallTextBox}">DefaultText (Bottom text)</TextBox>

<!-- Правый столбец -->

<Rectangle Grid.Column="1">

<Rectangle.Fill>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<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>

</Rectangle.Fill>

</Rectangle>

<TextBox Grid.Column="1" VerticalAlignment="Top" Style="{StaticResource SmallTextBox}">DefaultText TOP TOP TOP TOP TOP TOP</TextBox>

<TextBox Grid.Column="1" VerticalAlignment="Center" Style="{StaticResource SmallTextBox}">DefaultText CENTER CENTER CENTER CENTER</TextBox>

<TextBox Grid.Column="1" VerticalAlignment="Bottom" Style="{StaticResource SmallTextBox}">DefaultText BOTTOM BOTTOM BOTTOM BOTTOM</TextBox>

</Grid>

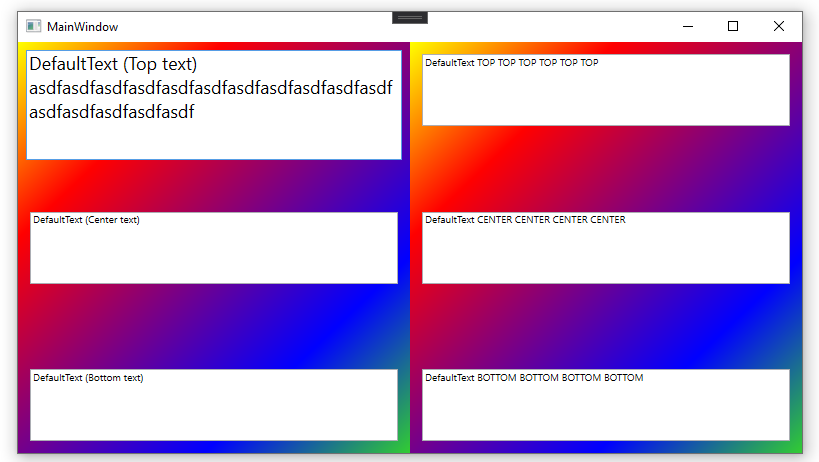
Результат выполнения программы:

Рис. 5 – Окно программы

При запуске программы в окне видны два одинаковых градиента, разделяющих программу на две половины. Также в каждой части окна можно увидеть несколько многострочных текстовых полей.

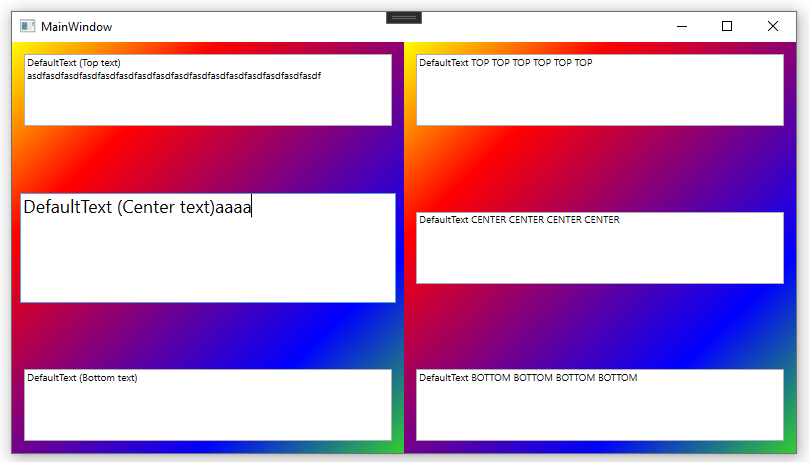


Рис. 6 – Окно программы при выделении другого текстового поля

То текстовое окно, в котором пользователь набирает текст, становится большим, размер шрифта в нём увеличивается, остальные текстовые поля остаются маленькими. Внешний вид однотипных элементов управления определяется с помощью стилей.

Вывод: Я ознакомился с принципами использования стилей в WPF-приложениях.