UI like VisualStudio

Vzhled VS s jednim SideBarem

<Controls:MetroWindow

x:Class="MIR.Media.Changing2.Gui.Views.MainView"

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

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

xmlns:Controls="clr-namespace:MahApps.Metro.Controls;assembly=MahApps.Metro"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:gui="clr-namespace:MIR.Media.Changing2.Gui"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

d:DesignHeight="1000"

d:DesignWidth="1000"

WindowState="Maximized"

mc:Ignorable="d">

<Window.Resources>

<BitmapImage x:Key="Pin" UriSource="/MIR.Media.Changing2.Gui;component/Icons/Pin.png" />

</Window.Resources>

<DockPanel>

<!-- Svisly button -->

<StackPanel DockPanel.Dock="Right" Orientation="Horizontal">

<StackPanel.LayoutTransform>

<RotateTransform Angle="90" />

</StackPanel.LayoutTransform>

<Button

Name="BarButton"

Content="{x:Static gui:GuiResources.MotivletFilters}"

MouseEnter="Pane1Button\_MouseEnter"

Width="300"/>

</StackPanel>

<Grid Name="ParentGrid" Grid.IsSharedSizeScope="True">

<!-- spodni vrstva -->

<Grid Name="Layer0" MouseEnter="Layer0\_MouseEnter">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto" />

<ColumnDefinition />

</Grid.ColumnDefinitions>

<GroupBox Grid.Column="1" Header="Spodni vrstva">

<ListBox>

<ListBoxItem>1hoiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiihhh</ListBoxItem>

</ListBox>

</GroupBox>

</Grid>

<!-- horni vrstva se SidePanelem -->

<Grid Name="Layer1" Visibility="Collapsed">

<Grid.ColumnDefinitions>

<ColumnDefinition />

<ColumnDefinition Width="auto" SharedSizeGroup="column1" />

</Grid.ColumnDefinitions>

<!-- obsah sidePanelu -->

<Grid Grid.Column="1" Background="{DynamicResource {x:Static SystemColors.ActiveCaptionBrushKey}}">

<Grid.RowDefinitions>

<RowDefinition Height="auto" />

<RowDefinition />

</Grid.RowDefinitions>

<!-- header sidePanelu -->

<DockPanel Grid.Row="0">

<Button

Name="SidePanelPin"

Width="26"

Click="Pane1Pin\_Click"

DockPanel.Dock="Right"

Margin="5">

<Image Name="PinImage" Source="{StaticResource Pin}" />

</Button>

<TextBlock

Padding="8"

DockPanel.Dock="Left"

Foreground="{DynamicResource {x:Static SystemColors.ActiveCaptionTextBrushKey}}"

Text="{x:Static gui:GuiResources.MotivletFilters}"

TextTrimming="CharacterEllipsis" />

</DockPanel>

<ListBox Grid.Row="1" Padding="10" Background="LightGray">

<ListBoxItem MinWidth="400">CONTENT: USERCONTROL</ListBoxItem>

</ListBox>

</Grid>

<GridSplitter

Grid.Column="1"

Width="5"

HorizontalAlignment="Left" />

</Grid>

</Grid>

</DockPanel>

</Controls:MetroWindow>

Codebehind:

using System;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Media.Imaging;

using MahApps.Metro.Controls;

namespace MIR.Media.Changing2.Gui.Views

{

/// <summary>

/// Layer0 => The lower layer without SidePanel

/// Layer1 => The upper layer with Sidepanel

/// </summary>

public partial class MainView : MetroWindow

{

private readonly ColumnDefinition m\_column1CloneForLayer0;

public MainView()

{

InitializeComponent();

// Initialize the dummy columns used when docking:

m\_column1CloneForLayer0 = new ColumnDefinition

{

SharedSizeGroup = "column1"

};

}

// Toggle between docked and undocked states (Pane 1)

private void Pane1Pin\_Click(object sender, RoutedEventArgs e)

{

if (BarButton.Visibility == Visibility.Collapsed)

{

UndockPane(1);

}

else

{

DockPane(1);

}

}

// Show Pane 1 when hovering over its button

private void Pane1Button\_MouseEnter(object sender, RoutedEventArgs e)

{

Layer1.Visibility = Visibility.Visible;

}

// Hide any undocked panes when the mouse enters Layer 0

private void Layer0\_MouseEnter(object sender, RoutedEventArgs e)

{

if (BarButton.Visibility == Visibility.Visible)

{

Layer1.Visibility = Visibility.Collapsed;

}

}

// Docks a pane, which hides the corresponding pane button

private void DockPane(int paneNumber)

{

if (paneNumber == 1)

{

BarButton.Visibility = Visibility.Collapsed;

PinImage.Source = new BitmapImage(new Uri("/Icons/Pin.png", UriKind.Relative));

// Add the cloned column to layer 0:

Layer0.ColumnDefinitions.Add(m\_column1CloneForLayer0);

}

}

// Undocks a pane, which reveals the corresponding pane button

private void UndockPane(int paneNumber)

{

if (paneNumber == 1)

{

Layer1.Visibility = Visibility.Collapsed;

BarButton.Visibility = Visibility.Visible;

PinImage.Source = new BitmapImage(new Uri("HorizontalPin.png", UriKind.Relative));

// Remove the cloned columns from layers 0 and 1:

Layer0.ColumnDefinitions.Remove(m\_column1CloneForLayer0);

}

}

}

}