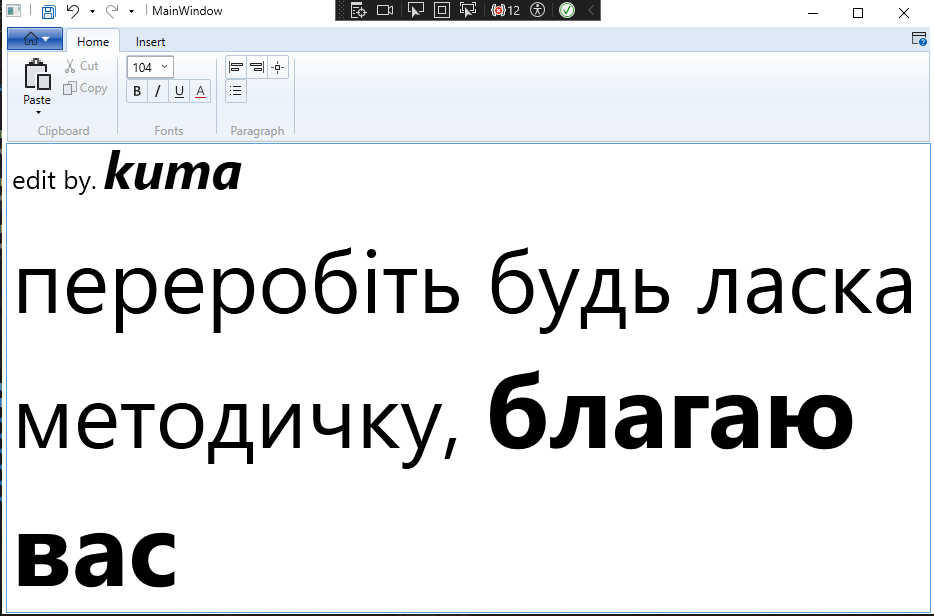
Лабораторна робота № 13-15

Тема роботи: Реалізація інтерфейсу текстового редактору у WPF.



| MainWindow.xaml |
| --- |
| <RibbonWindow x:Class="WpfApp12.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:WpfApp12"  mc:Ignorable="d"  Title="MainWindow" Height="350" Width="525">  <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="Auto"/>  <RowDefinition Height="\*"/>  </Grid.RowDefinitions>  <Ribbon Grid.Row="0" Margin="1,1,1,1">  <!--btn Help-->  <Ribbon.HelpPaneContent>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\foto\Help\help.png" KeyTip="F" />  </Ribbon.HelpPaneContent>  <!--Home menu-->  <Ribbon.ApplicationMenu>  <RibbonApplicationMenu SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\foto\home\Home.png">  <RibbonApplicationMenuItem Header="Create" ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Document.png"  KeyTip="C" Click="btnOpen\_Click"/>  <RibbonApplicationMenuItem Header="Open" ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\OpenFolder.png"  KeyTip="O" Click="btnOpen\_Click"/>  <RibbonApplicationMenuItem Header="Print" ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\PrintDialog.png"  KeyTip="P"/>  <RibbonApplicationMenuItem Header="Properties" ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\PropertyPublic.png"/>  <RibbonApplicationMenuItem Header="Close" ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\CloseDocument.png"  KeyTip="C" Click="btnClose\_Click"/>  </RibbonApplicationMenu>  </Ribbon.ApplicationMenu>  <!--Fast menu-->  <Ribbon.QuickAccessToolBar>  <RibbonQuickAccessToolBar>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\SaveAll.png" Click="btnSave\_Click"/>  <RibbonSplitButton x:Name="Undo" SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Undo.png"  Command="{x:Static ApplicationCommands.Undo}"  CommandTarget="{Binding ElementName=\_richTextBox}">  <RibbonSplitMenuItem Header="Undo1"></RibbonSplitMenuItem>  </RibbonSplitButton>  <RibbonSplitButton x:Name="Redo" SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Redo.png"  Command="{x:Static ApplicationCommands.Redo}"  CommandTarget="{Binding ElementName=\_richTextBox}">  <RibbonSplitMenuItem Header="Redo1"></RibbonSplitMenuItem>  </RibbonSplitButton>  </RibbonQuickAccessToolBar>  </Ribbon.QuickAccessToolBar>  <!--ribbon tabs-->  <RibbonTab Header="Home">  <!--Home/Clip.group-->  <RibbonGroup Header="Clipboard">  <RibbonMenuButton LargeImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\paste.png" Label="Paste" KeyTip="V">  <RibbonMenuItem ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Paste.png" Header="Keep Text Only" KeyTip="T"  Command="{x:Static ApplicationCommands.Paste}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonMenuItem ImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Paste.png" Header="Paste Special..."  KeyTip="S"/>  </RibbonMenuButton>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Cut.png" Label="Cut" KeyTip="X"  Command="{x:Static ApplicationCommands.Cut}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Copy.png" Label="Copy" KeyTip="C"  Command="{x:Static ApplicationCommands.Copy}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  </RibbonGroup>  <!--Home/Colors group-->  <RibbonGroup x:Name="fonts" Header="Fonts" Width="Auto">  <RibbonControlGroup>  <ComboBox ItemsSource="{Binding Source={x:Static Fonts.SystemFontFamilies}}"  SelectionChanged="FontSize\_SelectionChange" x:Name="\_fontSize"  Text="Size..." IsEditable="True"></ComboBox>  </RibbonControlGroup>  <RibbonControlGroup>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Bold.png" KeyTip="B"  Command="{x:Static EditingCommands.ToggleBold}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Italic.png" KeyTip="I"  Command="{x:Static EditingCommands.ToggleItalic}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Underline.png" KeyTip="U"  Command="{x:Static EditingCommands.ToggleUnderline}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Fontcolor.png" KeyTip="A"/>  </RibbonControlGroup>  </RibbonGroup>  <RibbonGroup x:Name="paragraph" Header="Paragraph">  <RibbonControlGroup>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\AlignLeft.png"  Command="{x:Static EditingCommands.AlignLeft}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\AlignRight.png"  Command="{x:Static EditingCommands.AlignRight}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\AlignBoth.png"  Command="{x:Static EditingCommands.AlignJustify}"  CommandTarget="{Binding ElementName=\_richTextBox}"/>  </RibbonControlGroup>  <RibbonControlGroup>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\BulletList.png"/>  </RibbonControlGroup>  </RibbonGroup>    </RibbonTab>  <RibbonTab Header="Insert" Margin="-8,0,8,0" UseLayoutRounding="True"  ScrollViewer.VerticalScrollBarVisibility="Auto">  <RibbonGroup x:Name="objects" Header="Objects">  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Table.png" Label="Table"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\ApplicationAccess.png"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Rectangle.png"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\LineChart.png"/>  <RibbonButton SmallImageSource="E:\Project VS\C#\WPF practices\WPF-practices\WpfApp12\WpfApp12\Images\Image.png"/>  </RibbonGroup>  </RibbonTab>  </Ribbon>  <RichTextBox x:Name="doc1" Grid.Row="1">  </RichTextBox>    </Grid>  </RibbonWindow> |

| MainWindow.xaml.cs |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  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;  using System.Windows.Controls.Ribbon;  using System.Windows.Forms;  using System.IO;  namespace WpfApp12  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : RibbonWindow  {  public bool IsSaved = false;  public MainWindow()  {  InitializeComponent();  \_fontSize.ItemsSource = FontSizes;  }  public double[] FontSizes  {  get  {  return new double[] { 3.0, 4.0, 5.0, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0,  9.5, 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0,  15.0, 16.0, 17.0, 18.0, 19.0, 20.0, 22.0, 24.0, 26.0, 28.0,  30.0, 32.0, 34.0, 36.0, 38.0, 40.0, 44.0, 48.0, 52.0, 56.0,  60.0, 64.0, 68.0, 72.0, 76.0, 80.0, 88.0, 96.0, 104.0, 112.0,  120.0, 128.0, 136.0, 144.0 };  }  }  private void btnOpen\_Click(object sender, RoutedEventArgs e)  {  //OpenFileDialog dlg = new OpenFileDialog();  Microsoft.Win32.OpenFileDialog dlg = new Microsoft.Win32.OpenFileDialog();  dlg.Filter = "Document files (\*.rtf)|\*.rtf";  //DialogResult result = dlg.ShowDialog();  Nullable<bool> result = dlg.ShowDialog();  if (result == true)// result.Value  {  TextRange tr = new TextRange(doc1.Document.ContentStart, doc1.Document.ContentEnd);  FileStream file = new FileStream(dlg.FileName, FileMode.Open);  tr.Load(file, System.Windows.DataFormats.Rtf);  }  }  private void btnCreate\_Click(object sender, RoutedEventArgs e)  {  //TextRange tr = new TextRange(doc1.Document.ContentStart, doc1.Document.ContentEnd);  //tr.Text = "";  }  private void btnSave\_Click(object sender, RoutedEventArgs e)  {  //SaveFileDialog saveFile = new SaveFileDialog();  Microsoft.Win32.SaveFileDialog saveFile = new Microsoft.Win32.SaveFileDialog();  saveFile.FileName = "unknown.doc";  // set filters - this can be done in properties as well  saveFile.Filter = "Document files (\*.rtf)|\*.rtf";  if (saveFile.ShowDialog() == true)  {  TextRange tr = new TextRange(doc1.Document.ContentStart, doc1.Document.ContentEnd);  this.Title = this.Title + " - " + saveFile.FileName;  FileStream file = new FileStream(saveFile.FileName, FileMode.Create);  tr.Save(file, System.Windows.DataFormats.Rtf);  file.Close();  }  IsSaved = true;  }  private void btnClose\_Click(object sender, RoutedEventArgs e)  {  if (!IsSaved)  {  if (System.Windows.MessageBox.Show("Do you want save changes ?", "Message", MessageBoxButton.YesNo) == MessageBoxResult.Yes)  {  //If the Yes button was pressed, call the Save method  this.btnSave\_Click(sender, e);  }  }  this.Close();  }  void ApplyPropertyValueToSelectedText(DependencyProperty formattingProperty, object value)  {  if (value == null)  return;  doc1.Selection.ApplyPropertyValue(formattingProperty, value);  }  private void FontFamili\_SelectionChange(object sender, SelectionChangedEventArgs e)  {  try  {  if (e.AddedItems.Count > 0 && e.AddedItems[0] is FontFamily)  {  FontFamily editValue = (FontFamily)e.AddedItems[0];  ApplyPropertyValueToSelectedText(TextElement.FontFamilyProperty, editValue);  }  }  catch (Exception ex)  {  System.Windows.MessageBox.Show($"An error occurred: {ex.Message}");  }  }  private void FontSize\_SelectionChange(object sender, SelectionChangedEventArgs e)  {  try  {  ApplyPropertyValueToSelectedText(TextElement.FontSizeProperty, e.AddedItems[0]);  }  catch (Exception) { }  }  }  } |