**27 КОМПОНОВКА И ЭЛЕМЕНТЫ УПРАВЛЕНИЯ В WPF**

Задание 1. В работе 24 «Язык разметки XAML». Измените интерфейс проекта WPF – XmlTaskWPF.

Листинг программы:

using System;

using System.Windows;

using System.Windows.Controls;

using System.Xml;

namespace z1

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

static string path = string.Empty;

static XmlDocument xDoc = new XmlDocument();

public MainWindow()

{

InitializeComponent();

}

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

{

path = textBoxPath.Text;

CheckerChooseView(path);

}

private void CheckerChooseView(string localPath)

{

RadioButton textBoxRadioButton = TextBoxRadioButton;

RadioButton treeViewRadioButton = TreeViewRadioButton;

RadioButton listBoxRadioButton = ListBoxRadioButton;

textBoxContent.Text = string.Empty;

if ((bool)textBoxRadioButton.IsChecked)

{

MessageBox.Show("Входит в textBox");

TextBoxView(path);

}

if ((bool)treeViewRadioButton.IsChecked)

{

MessageBox.Show("Входит в treeView");

TreeView(path);

}

if ((bool)listBoxRadioButton.IsChecked)

{

MessageBox.Show("Входит в listBox");

ListBoxView(path);

}

}

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

{

PritnSomeXmlElement(path);

}

private void PritnSomeXmlElement(string path)

{

if (path != string.Empty)

{

if (path == textBoxPath.Text)

{

int count = 0;

textBoxResult.Text = string.Empty;

xDoc.Load(path);

XmlElement xRoot = xDoc.DocumentElement;

foreach (XmlNode xnode in xRoot)

{

// получаем атрибут name

if (xnode.Attributes.Count > 0)

{

XmlNode attr = xnode.Attributes.GetNamedItem("name");

if (attr != null && attr.Value == textBoxSearch.Text)

{

count++;

textBoxResult.Text += attr.Value + " | ";

foreach (XmlNode childnode in xnode.ChildNodes)

{

// если узел - company

if (childnode.Name == "author")

{

textBoxResult.Text += "Автор: " + childnode.InnerText + " | ";

}

// если узел age

if (childnode.Name == "publishingYear")

{

textBoxResult.Text += "Год: " + childnode.InnerText + '\n';

}

}

}

}

}

if (count == 0)

{

textBoxResult.Text = "Совпадений не найдено";

}

}

else MessageBox.Show("Путь изменён, сначала откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

else MessageBox.Show("Откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

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

{

if (path != string.Empty)

{

if (path == textBoxPath.Text)

{

xDoc.Load(path);

XmlElement xRoot = xDoc.DocumentElement;

string deleteName = textBoxDelete.Text;

try

{

DeleteNodeByAttName(deleteName, xDoc);

xDoc.Save(path);

textBoxResult.Text = $"Книга {deleteName} успешно удаленa";

CheckerChooseView(path);

}

catch (Exception)

{

if (deleteName == String.Empty)

textBoxResult.Text = $"Книга не найдена";

else textBoxResult.Text = $"Книга {deleteName} не найдена";

}

}

else MessageBox.Show("Путь изменён, сначала откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

else MessageBox.Show("Откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

private void DeleteNodeByAttName(string name, XmlDocument xmlDoc)

{

XmlNode node = xmlDoc.SelectSingleNode(String.Format("/library/book[@name='{0}']", name));

if (node != null)

{

xmlDoc.SelectSingleNode("library").RemoveChild(node);

}

else throw new Exception();

}

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

{

Application.Current.Shutdown();

}

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

{

if (path == textBoxPath.Text)

{

if (path != string.Empty)

{

xmlAddWindows xmlAddWindows = new xmlAddWindows();

xmlAddWindows.path = textBoxPath.Text;

xmlAddWindows.ShowDialog();

}

else MessageBox.Show("Введите путь к файлу!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

else if (path == "") MessageBox.Show("Откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

else MessageBox.Show("Путь изменён, сначала откройте файл!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

static TextBox localTextBoxContent;

static TreeView localTreeViewContent;

static ListBox localListBoxContent;

private void RadioButton\_Checked(object sender, RoutedEventArgs e)

{

localTextBoxContent = textBoxContent;

localTreeViewContent = treeViewContent;

localListBoxContent = listBoxContent;

localTextBoxContent.Visibility = Visibility.Visible;

localTreeViewContent.Visibility = Visibility.Collapsed;

localListBoxContent.Visibility = Visibility.Collapsed;

}

private void TreeViewRadioButton\_Checked(object sender, RoutedEventArgs e)

{

localTextBoxContent = textBoxContent;

localTreeViewContent = treeViewContent;

localListBoxContent = listBoxContent;

localTextBoxContent.Visibility = Visibility.Collapsed;

localTreeViewContent.Visibility = Visibility.Visible;

localListBoxContent.Visibility = Visibility.Collapsed;

}

private void ListBoxRadioButton\_Checked(object sender, RoutedEventArgs e)

{

localTextBoxContent = textBoxContent;

localTreeViewContent = treeViewContent;

localListBoxContent = listBoxContent;

localTextBoxContent.Visibility = Visibility.Collapsed;

localTreeViewContent.Visibility = Visibility.Collapsed;

localListBoxContent.Visibility = Visibility.Visible;

}

private void TextBoxView(string localPath)

{

try

{

xDoc.Load(localPath);

XmlElement xRoot = xDoc.DocumentElement;

// обход всех узлов в корневом элементе

foreach (XmlNode xnode in xRoot)

{

// получаем атрибут name

if (xnode.Attributes.Count > 0)

{

XmlNode attr = xnode.Attributes.GetNamedItem("name");

if (attr != null)

textBoxContent.Text += attr.Value + " | ";

}

// обходим все дочерние узлы элемента user

foreach (XmlNode childnode in xnode.ChildNodes)

{

// если узел - company

if (childnode.Name == "author")

{

textBoxContent.Text += "Автор: " + childnode.InnerText + " | ";

}

// если узел age

if (childnode.Name == "publishingYear")

{

textBoxContent.Text += "Год: " + childnode.InnerText + '\n';

}

}

}

}

catch (Exception)

{

path = String.Empty;

MessageBox.Show("Неверный путь к файлу!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

static TreeViewItem item = new TreeViewItem();

static private void TreeView(string localPath)

{

try

{

xDoc.Load(localPath);

XmlElement xRoot = xDoc.DocumentElement;

// обход всех узлов в корневом элементе

foreach (XmlNode xnode in xRoot)

{

// получаем атрибут name

if (xnode.Attributes.Count > 0)

{

XmlNode attr = xnode.Attributes.GetNamedItem("name");

if (attr != null)

{

item = new TreeViewItem();

item.Header = attr.Value;

localTreeViewContent.Items.Add(item);

}

}

// обходим все дочерние узлы элемента user

foreach (XmlNode childnode in xnode.ChildNodes)

{

// если узел - company

if (childnode.Name == "author")

{

item.Items.Add(childnode.InnerText);

}

// если узел age

if (childnode.Name == "publishingYear")

{

item.Items.Add(childnode.InnerText);

}

}

}

}

catch (Exception)

{

path = String.Empty;

MessageBox.Show("Неверный путь к файлу!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void ListBoxView(string localPath)

{

try

{

xDoc.Load(localPath);

XmlElement xRoot = xDoc.DocumentElement;

// обход всех узлов в корневом элементе

foreach (XmlNode xnode in xRoot)

{

string bookInfo = string.Empty;

// получаем атрибут name

if (xnode.Attributes.Count > 0)

{

XmlNode attr = xnode.Attributes.GetNamedItem("name");

if (attr != null)

bookInfo += attr.Value + " | ";

}

// обходим все дочерние узлы элемента user

foreach (XmlNode childnode in xnode.ChildNodes)

{

// если узел - company

if (childnode.Name == "author")

{

bookInfo += "Автор: " + childnode.InnerText + " | ";

}

// если узел age

if (childnode.Name == "publishingYear")

{

bookInfo += childnode.InnerText;

}

}

listBoxContent.Items.Add(bookInfo);

}

}

catch (Exception)

{

path = String.Empty;

MessageBox.Show("Неверный путь к файлу!", "Error!", MessageBoxButton.OK, MessageBoxImage.Error);

}}

}

}

Таблица 27.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Путь к файлу | Harry Potter | Автор: lk | Год: 09988  Harry Potter 3 | Автор: J. K. Rowling | Год: 2000  Harry Potter 5 | Автор: Shumel | Год: 2020  Harry Potter | Автор: lk | Год: 09988  Harry Potter 3 | Автор: J. K. Rowling | Год: 2000  Harry Potter 5 | Автор: ShumeAl | Год: 2020 |

Анализ результатов:

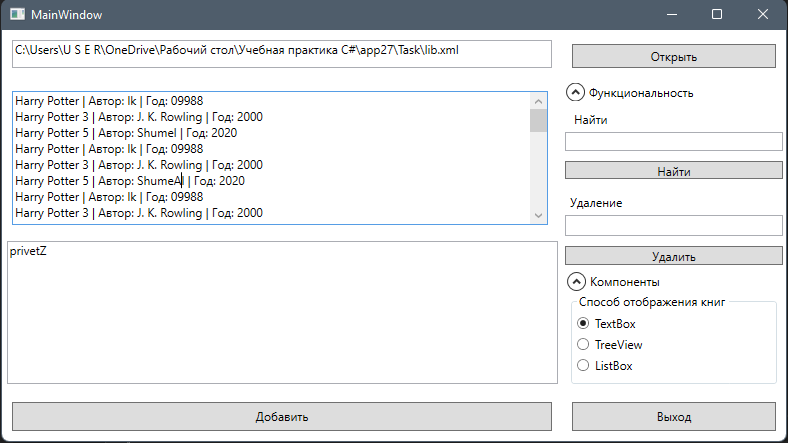


Рисунок 27.1 – Результат работы программы

Источник – собственная разработка