Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

По дисциплине: «ОСиСП»

Выполнил:

Студент 3 курса

Группы ПО-4(1)

Елисеев С. Г.

Проверила:

Дряпко А.В.

Брест 2021

Лабораторная работа № 5

Вариант 3

Цель: научиться разрабатывать и использовать динамические библиотеки (DLL)

Задание:

Основное задание заключается в доработке функционала обновления, разработка которого про-

изводилась в ЛР No4. Нужно интегрировать указанную функцию в само приложение, без использования стороннего клиента. При этом серверная часть приложения остается без изменений (возможны некоторые доработки сервера, без изменения общей клиент-серверной архитекту-

ры)

Проверка обновления осуществляться по непосредственному запросу пользователя. Предусмотреть выбор из меню политики обновления

(с пользовательским подтверждением, без подтверждения/автоматически);

Решение содержит 4 проекта:

1. Lab1
2. Aboutdll
3. Server
4. ClientLayer

Код:

**Lab1.Mainwindow.xaml.cs**

using System;

using System.Collections.Generic;

using System.IO;

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.Globalization;

using Microsoft.Win32;

using lab1;

namespace TRIPS\_2

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void DrawingModeSelection(object sender, RoutedEventArgs e)

{

Ink.EditingMode = InkCanvasEditingMode.Ink;

}

private void DeletingModeSelection(object sender, RoutedEventArgs e)

{

Ink.EditingMode = InkCanvasEditingMode.EraseByStroke;

}

private void FormatingModeSelection(object sender, RoutedEventArgs e)

{

Ink.EditingMode = InkCanvasEditingMode.EraseByPoint;

}

private void ColorsList\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

ComboBoxItem selectedItem = (ComboBoxItem)(ColorsList.SelectedValue);

string value = (string)(selectedItem.Content);

Ink.DefaultDrawingAttributes.Color = (Color)ColorConverter.ConvertFromString(value);

}

private void BrushWidthChange(object sender, RoutedPropertyChangedEventArgs<double> e)

{

Ink.DefaultDrawingAttributes.Width = ChooseWidth.Value;

Ink.DefaultDrawingAttributes.Height = ChooseWidth.Value;

}

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

{

Close();

}

private void SaveControlImage(

Visual baseElement, int imageWidth, int imageHeight, string pathToOutputFile)

{

// 1) get current dpi

var pSource = PresentationSource.FromVisual(Application.Current.MainWindow);

Matrix m = pSource.CompositionTarget.TransformToDevice;

double dpiX = m.M11 \* 96;

double dpiY = m.M22 \* 96;

// 2) create RenderTargetBitmap

var elementBitmap = new RenderTargetBitmap(imageWidth, imageHeight, dpiX, dpiY, PixelFormats.Default);

// 3) undo element transformation

var drawingVisual = new DrawingVisual();

using (DrawingContext drawingContext = drawingVisual.RenderOpen())

{

var visualBrush = new VisualBrush(baseElement);

drawingContext.DrawRectangle(

visualBrush,

null,

new Rect(new Point(0, 0), new Size(imageWidth / m.M11, imageHeight / m.M22)));

}

// 4) draw element

elementBitmap.Render(drawingVisual);

// 5) create PNG image

var encoder = new PngBitmapEncoder();

encoder.Frames.Add(BitmapFrame.Create(elementBitmap));

// 6) save image to file

using (var imageFile = new FileStream(pathToOutputFile, FileMode.Create, FileAccess.Write))

{

encoder.Save(imageFile);

imageFile.Flush();

imageFile.Close();

}

}

private string GetOutputFileName()

{

var dlgSave = new SaveFileDialog

{

DefaultExt = "png",

Title = "Choose output file",

Filter = "PNG Files|\*.png"

};

if (dlgSave.ShowDialog() == true)

return dlgSave.FileName;

return string.Empty;

}

private void Btn\_Save\_Click(object sender, RoutedEventArgs e)

{

// get output file name

string fileName = this.GetOutputFileName();

if (fileName == string.Empty)

return;

// select visual element

Visual element = null;

var elementWidth = 0;

var elementHeight = 0;

element = this.Ink;

elementWidth = (int)this.Ink.ActualWidth;

elementHeight = (int)this.Ink.ActualHeight;

// save selected element to file

this.SaveControlImage(element, elementWidth, elementHeight, fileName);

}

private void Btn\_About\_Click(object sender, RoutedEventArgs e)

{

string A = "Елисеев Сергей, 3-й курс, ПО-4(1)\r\n" +

"Моё приложение позволит вам реализовать свои фантазии на холсте";

aboutdll.Class1.Show\_About(A);

}

private void Btn\_toUpdate(object sender, RoutedEventArgs e)

{

string path1 = @"C:\Users\Sergei\source\repos\lab1\lab1\MainWindow.xaml.cs";

string path2 = @"C:\Users\Sergei\source\repos\lab4\lab1\MainWindow.xaml.cs";

FileStream fs1;

FileStream fs2;

ClientLayer.ClientServer.Main();

fs1 = new FileStream(path1, FileMode.Open, FileAccess.Read);

fs2 = new FileStream(path2, FileMode.Open, FileAccess.Read);

if (fs1.Length != fs2.Length)

{

fs1.Close();

fs2.Close();

Window1 UpWindow = new Window1();

UpWindow.Show();

this.Close();

}

else

{

fs1.Close();

fs2.Close();

MessageBox.Show("We have no new version:(");

}

}

}

}

**Lab1.MainWindow.xaml**

<Window x:Class="TRIPS\_2.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:TRIPS\_2"

mc:Ignorable="d"

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

<Grid Name="baseGrid">

<InkCanvas Name="Ink" >

<InkCanvas.DefaultDrawingAttributes>

<DrawingAttributes/>

</InkCanvas.DefaultDrawingAttributes>

</InkCanvas>

<Grid Name="TopPanel" VerticalAlignment="Top" HorizontalAlignment="Left" Width="800" Height="99" Background="Gainsboro" Margin="0,0,-6.4,0">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="423\*"/>

<ColumnDefinition Width="377\*"/>

</Grid.ColumnDefinitions>

<TextBlock FontFamily="Arial" FontSize="18" Margin="20, 10, 0, 0" Grid.ColumnSpan="2"><Run Text="Choose color:"/></TextBlock>

<ComboBox FontFamily="Arial" FontSize="16" Name="ColorsList" Width="200" Height="30" HorizontalAlignment="Left" Margin="20,32,0,18" SelectionChanged="ColorsList\_SelectionChanged">

<ComboBoxItem Content="Black" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Red" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Orange" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Yellow" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Green" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Cyan" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Blue" FontFamily="Arial" FontSize="16"/>

<ComboBoxItem Content="Purple" FontFamily="Arial" FontSize="16"/>

</ComboBox>

<TextBlock Text="Choose mode:" FontFamily="Arial" FontSize="18" Margin="280,10,0,0" Grid.ColumnSpan="2"/>

<RadioButton Name="DrawingMode" Checked="DrawingModeSelection" Margin="280, 30, 0, 0" FontFamily="Arial" FontSize="12" Content="кисть&#xD;&#xA;" Grid.ColumnSpan="2">

</RadioButton>

<RadioButton Name="DeletingMode" Checked="DeletingModeSelection" Margin="280, 45, 0, 0" FontFamily="Arial" FontSize="12" Content="удалить объект&#xD;&#xA;" Grid.ColumnSpan="2">

</RadioButton>

<RadioButton Name="FormatingMode" Checked="FormatingModeSelection" Margin="280, 60, 0, 0" FontFamily="Arial" FontSize="12" Content="ластик" Grid.ColumnSpan="2">

</RadioButton>

<TextBlock FontFamily="Arial" FontSize="18" Margin="34.6,10,0,0" Grid.Column="1"><Run Text="Choose width:"/><InlineUIContainer>

</InlineUIContainer></TextBlock>

<Button x:Name="Btn\_Save" Content="Сохранить" Click="Btn\_Save\_Click" HorizontalAlignment="Right" VerticalAlignment="Bottom" Width="71" Height="35" Margin="0,0,10,0" Grid.Column="1"/>

<Slider Name="ChooseWidth" Maximum="100" Value="1" TickPlacement="Both" TickFrequency="10" Margin="76.6,35,100,0" Width="200" Foreground="Black" ValueChanged="BrushWidthChange" Grid.Column="1"/>

<Button Name="CloseWindow" Width="40" Height="40" BorderThickness="0" VerticalAlignment="Top" HorizontalAlignment="Right" Background="Gainsboro" Click="CloseWindow\_Click" Grid.Column="1" Margin="0,0,10,0">

<Image Source="https://emojio.ru/images/apple-b/274c.png" Height="11" Width="20"></Image>

</Button>

<Button x:Name="Btn\_About" Content="О&#xD;&#xA; программе" Click="Btn\_About\_Click" HorizontalAlignment="Right" VerticalAlignment="Bottom" Width="71" Height="35" Margin="0,0,10,30" Grid.Column="1"/>

<Button x:Name="Btn\_Update" Content="Обновление" Click="Btn\_toUpdate" HorizontalAlignment="Right" VerticalAlignment="Bottom" Width="71" Height="22" Margin="0,0,100,0" Grid.Column="1"/>

</Grid>

</Grid>

</Window>

**Lab1.Updates.xaml**

<Window x:Class="lab1.Window1"

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:lab1"

mc:Ignorable="d"

Title="Window1" Height="350" Width="500">

<Grid>

<Button Content="Да" HorizontalAlignment="Left" Margin="95,258,0,0" VerticalAlignment="Top" Width="94" Height="30" Click="Button\_Yes"/>

<Button Content="Нет" HorizontalAlignment="Left" Margin="316,258,0,0" VerticalAlignment="Top" Width="94" Height="30" Click="Button\_No"/>

<Label Content="Хотите обновить версию?" HorizontalAlignment="Left" Margin="168,105,0,0" VerticalAlignment="Top" Height="29" Width="159"/>

</Grid>

</Window>

**Lab1.Updates.xaml.cs**

using System;

using System.Collections.Generic;

using System.IO;

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.Shapes;

using TRIPS\_2;

namespace lab1

{

/// <summary>

/// Логика взаимодействия для Window1.xaml

/// </summary>

public partial class Window1 : Window

{

public Window1()

{

InitializeComponent();

}

public static void ReplaceFile(string path1, string path2)

{

File.Delete(path2);

File.Copy(path1, path2);

}

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

{

string path1 = @"C:\Users\Sergei\source\repos\lab1\lab1\MainWindow.xaml.cs";

string path2 = @"C:\Users\Sergei\source\repos\lab4\lab1\MainWindow.xaml.cs";

ReplaceFile(path1, path2);

MainWindow newWindow = new MainWindow();

Application.Current.MainWindow = newWindow;

newWindow.Show();

MessageBox.Show("Вы обновились!");

Close();

}

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

{

MessageBox.Show("Ну и оставайтесь на старой версии -\_-");

MainWindow newWindow = new MainWindow();

Application.Current.MainWindow = newWindow;

newWindow.Show();

Close();

}

}

}

**Aboutdll.Class1(cs):**

using System;

using System.Windows;

namespace aboutdll

{

public static class Class1

{

public static void Show\_About(string a)

{

MessageBox.Show(a);

}

}

}

**Server.Program:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Sockets;

using System.Text;

using System.Threading.Tasks;

namespace Server

{

class Program

{

static int port = 8005; // порт для приема входящих запросов

static void Main(string[] args)

{

// получаем адреса для запуска сокета

IPEndPoint ipPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"), port);

// создаем сокет

Socket listenSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

try

{

// связываем сокет с локальной точкой, по которой будем принимать данные

listenSocket.Bind(ipPoint);

// начинаем прослушивание

listenSocket.Listen(10);

Console.WriteLine("Сервер запущен. Ожидание подключений...");

while (true)

{

Socket handler = listenSocket.Accept();

// получаем сообщение

StringBuilder builder = new StringBuilder();

int bytes = 0; // количество полученных байтов

byte[] data = new byte[256]; // буфер для получаемых данных

do

{

bytes = handler.Receive(data);

builder.Append(Encoding.Unicode.GetString(data, 0, bytes));

}

while (handler.Available > 0);

Console.WriteLine(DateTime.Now.ToShortTimeString() + ": " + builder.ToString());

// отправляем ответ

string message = "ваше сообщение доставлено";

data = Encoding.Unicode.GetBytes(message);

handler.Send(data);

// закрываем сокет

handler.Shutdown(SocketShutdown.Both);

handler.Close();

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

**ClientLayer.ClientServer:**

using System;

using System.Text;

using System.Net;

using System.Net.Sockets;

using System.IO;

namespace ClientLayer

{

public class ClientServer

{

// адрес и порт сервера, к которому будем подключаться

static int port = 8005; // порт сервера

static string address = "127.0.0.1"; // адрес сервера

public static void Main()

{

try

{

IPEndPoint ipPoint = new IPEndPoint(IPAddress.Parse(address), port);

Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

// подключаемся к удаленному хосту

socket.Connect(ipPoint);

string path1 = @"C:\Users\Sergei\source\repos\lab1\lab1\MainWindow.xaml.cs";

string path2 = @"C:\Users\Sergei\source\repos\lab4\lab1\MainWindow.xaml.cs";

FileStream fs1;

FileStream fs2;

fs1 = new FileStream(path1, FileMode.Open, FileAccess.Read);

fs2 = new FileStream(path2, FileMode.Open, FileAccess.Read);

string message;

if (fs1.Length != fs2.Length)

{

fs1.Close();

fs2.Close();

message = "Доступно обновление";

}

else

{

message = "Обновления.нет";

}

byte[] data = Encoding.Unicode.GetBytes(message);

socket.Send(data);

// получаем ответ

data = new byte[256]; // буфер для ответа

StringBuilder builder = new StringBuilder();

int bytes = 0; // количество полученных байт

do

{

bytes = socket.Receive(data, data.Length, 0);

builder.Append(Encoding.Unicode.GetString(data, 0, bytes));

}

while (socket.Available > 0);

Console.WriteLine("ответ сервера: " + builder.ToString());

// закрываем сокет

socket.Shutdown(SocketShutdown.Both);

socket.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Console.Read();

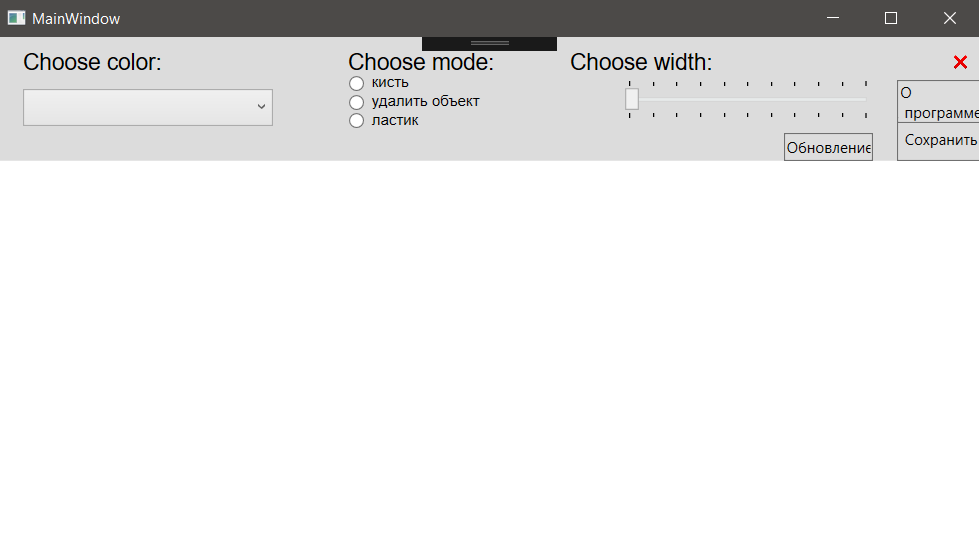
}

}

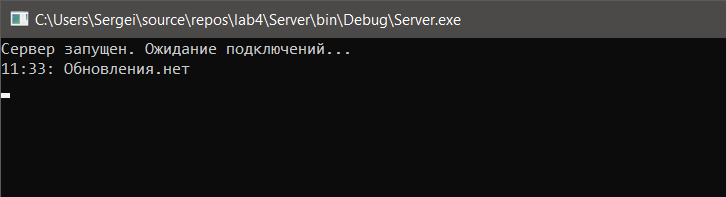
}

Результат:

Client:



Server:



Вывод: в ходе работы была доработано клиент-серверное взаимодействие.