МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ОДЕССКИЙ НАЦИОНАЛЬНЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИНСТИТУТ КОМПЬЮТЕРНЫХ СИСТЕМ

КАФЕДРА КОМПЬЮТЕРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ И СЕТЕЙ

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

по дисциплине:

«Системное программирование»

Выполнил студент группы АЕ-145

Михайлов О.М.

Проверил

Олещук О.В.

Одесса 2017

СОДЕРЖАНИЕ

[Введение 3](#_Toc452568078)

[1. Ввыбор варианта задания 4](#_Toc452568079)

[2.Описание программы 5](#_Toc452568080)

[2.1 Таймер 5](#_Toc452568081)

[2.2 Графики 5](#_Toc452568082)

[2.3 Реестр 6](#_Toc452568083)

[3.Ппример работы программы 7](#_Toc452568084)

[Вывод 9](#_Toc452568085)

[Список литературы 10](#_Toc452568086)

[Приложение 1. Код программы 11](#_Toc452568087)

# ВВЕДЕНИЕ

Целью данной работы является знакомство с созданием оконных приложений WPF, а также улучшение навыков написания программ на языке программирования С#. Данная расчетно-графическая работа охватывает очень многие аспекты работы с WPF и разработки приложений под операционную систему Windows.

В ходе выполнения работы студенты обучатся работе с реестром Windows, а также работой с графикой, также работе с таймерами и обработке событий, поступающих в главное и дочерние окна. Студенты также научатся создавать дочерние окна, задавать им размер и менять их положение, форму и их внешний вид.

Ещё одним полезным навыком для студентов будет обучение работе с диалоговыми окнами, выводом текста в них, добавлением картинок и другим действиям с диалоговыми окнами.

# 1. ВЫБОР ВАРИАНТА ЗАДАНИЯ

1. Создать оконное приложение на базе WPF.

2. Установить таймер, срабатывающий каждые 3 секунды.

3. Отобразить два графика. На каждом из них отображаются последние 40 значений. Значения считываются по событиям таймера. Оба графика отображаются в главном окне.

Вид графика - точечный.

Содержимое графика №1 - Объем свободной оперативной памяти.

Содержимое графика №2 – Объем свободного места на диске D:\.

4. Создать диалоговое окно, в котором отображается задание и информация об авторе с фотографией.

5. Прочесть информацию из реестра Windows, относящуюся к самостоятельно выбранному аппаратному и программному обеспечению, а также к настройкам, относящимся к компьютеру в целом и к отдельному пользователю.

6. Прочесть информацию из реестра Windows, определяющую ассоциации типа документа. Тип документа - DOC.

# 2. ОПИСАНИЕ ПРОГРАММЫ

## 2.1 Таймер

Чтобы данные на графиках отрисовывались каждые 3 секунды – задан таймер:

Paint PixelRam = new Paint(Graphic\_1, FreeRam);

Paint PixelMemory = new Paint(Graphic\_2, FreeMemory);

DispatcherTimer Timer = new DispatcherTimer();

Timer.Interval = TimeSpan.FromSeconds(3);

Timer.Tick += PixelRam.SetPixelRam;

Timer.Tick += PixelMemory.SetPixelMemory;

Timer.Start();

При срабатывании таймера, в функции обработки сообщений оконных функции передаётся соответствующее сообщение.

## 2.2 Графики

В данной расчетно-графической работе необходимо создать два графика в которых выводятся данные о количестве свободной оперативной памяти, а также свободного места на диске D:\.

На первом этапе создаётся сетка графика, внутри которой будут выводится нужные значение:

for (int y = 485; y >= 0; y -= 15)

{

Line OX = new Line();

if (y % 2 == 0)

{

Label text = new Label();

text.Content = Convert.ToString(Math.Round((8192.0 / 500) \* (y + 15)));

int marginTop = 485 - y - 9;

text.Margin = new Thickness(25, marginTop, 0, 0);

Graphic\_1.Children.Add(text);

}

OX.X1 = 61;

OX.X2 = 540;

OX.Y1 = y;

OX.Y2 = y;

OX.Stroke = Brushes.Gray;

Graphic\_1.Children.Add(OX);

}

Осуществляется это при помощи двух циклов, где рисуются горизонтальные и вертикальные линии. Одновременно с рисованием горизонтальных линий, выводится шкала всей физической и оперативной памяти.

На втором этапе рисуются данные на графиках, получаемые по событию таймера. В данной работе вид графика точечный, потому данные на графиках отображаются путём создания элипса(он же по размерам как пиксель, только заметнее), вместо функции SetPixel():

Ellipse PixelMemory = new Ellipse();

PixelMemory.Height = 2;

PixelMemory.Width = 2;

PixelMemory.Stroke = Brushes.Red;

PixelMemory.Fill = Brushes.Red;

graphic.Children.Add(PixelRam);

Размещается пиксель в области Canvas graphic.

Максимальное значение данных на графиках равно 40. После достижения этой цифры, происходит перерисовка пикселей, выведенных ранее.

## 2.3 Реестр

В данной расчетно-графической работе осуществляется считывание различных значений из ключей реестра Widows. Вначале осуществляется переход в нужный ключ функцией Registry.GetValue(). Синтаксис этой функции:

Registry.GetValue(

string keyName, //Полный путь к разделу реестра

string valueName, //Имя в паре "имя-значение"

object defaultValue //Возвращаемое значение, если значение

параметра valueName не существует

)

Например получение из регистра данных о процессоре системы:  
CPU.Content = "Мой процессор: " + Registry.GetValue(@"HKEY\_LOCAL\_MACHINE\HARDWARE\DESCRIPTION\System\CentralProcessor\0", "ProcessorNameString", 0);

# 3. ПРИМЕР РАБОТЫ ПРОГРАММЫ

После запуска исполняемого файла создается окно. В окне выводится график свободной оперативной памяти, а также график свободной памяти на диске D:\, информацию, относящуюся к самостоятельно выбранному аппаратному и программному обеспечению, а также к настройкам, относящимся к компьютеру в целом и к отдельному пользователю.

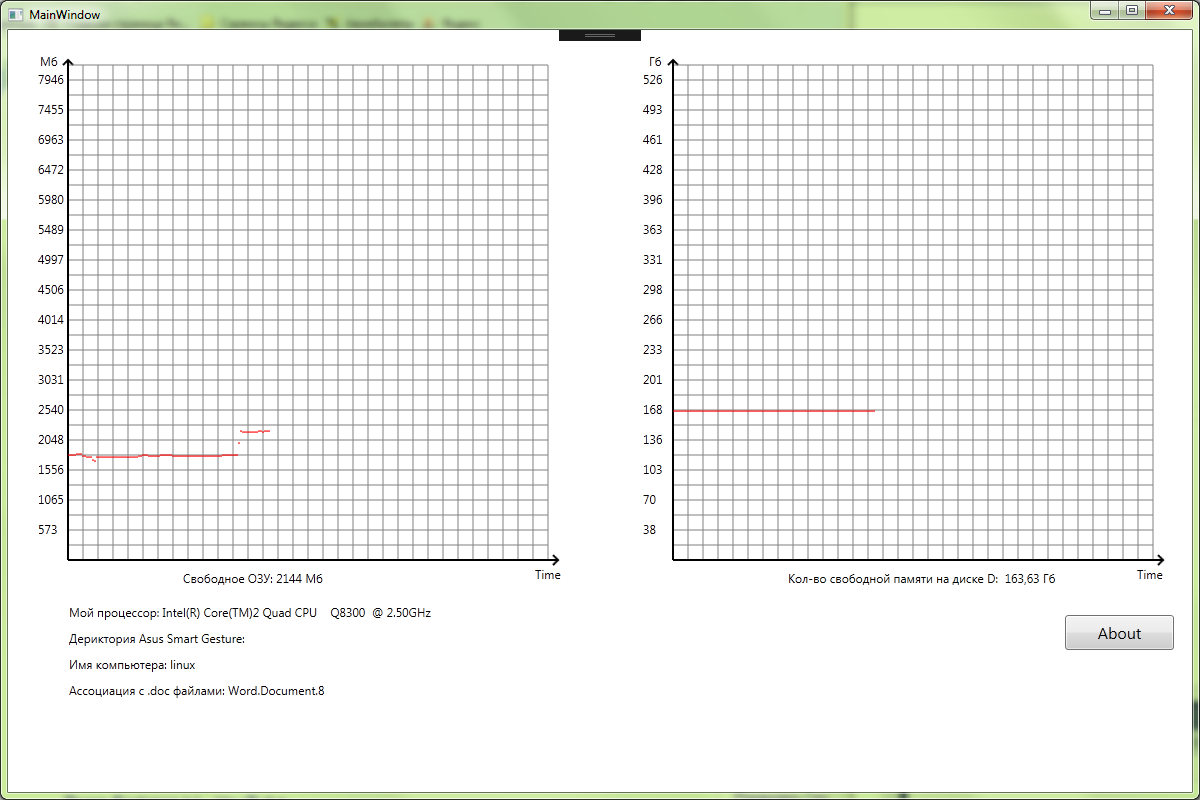


Рисунок 1 - Главное окно

После нажатия кнопки About будет выведено диалоговое окно с заданием и фотографией автора.

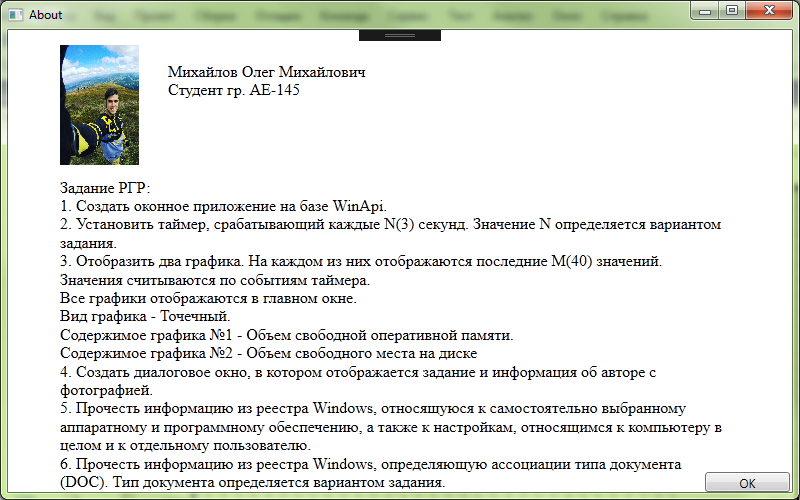


Рисунок 2 - Диалоговое окно

# ВЫВОД

В ходе расчетно-графической работы были получены навыки программирования оконных приложений WPF под операционной системой Windows. Были получены сведения о реестре Windows, а также выполнены задания, связанные с ним. Таким образом, в качестве информации об аппаратной части компьютера используется название центрального процессора, в качестве информации о программной части компьютера выводится расположениеAsus Smart Gesture. Информацией о всём компьютере считается его название, а информацией о текущем пользователе – язык системы. Также проверяется ассоциация файлов .doc.

Также выполняется отрисовка двух графиков – доступного места на диске и свободном объеме оперативной памяти. Оба графика выводятся в главном окне. Также был получен навык работы с диалоговыми окнами, добавлением туда текста и картинок. Таким образом, все задания расчетно-графической работы выполнены и работают без ошибок.

# СПИСОК ЛІТЕРАТУРИ

1. msdn.microsoft.com
2. Герберт Шилдт – C# 4.0 Полное руководство
3. Натан Адам – WPF 4 Подробное руководство

# Приложение 1. Код программы

**Главное окно:**

<Window x:Class="MikhailovRGR.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:MikhailovRGR"

mc:Ignorable="d"

Title="MainWindow" Height="800" Width="1200">

<Grid>

<Canvas x:Name="Graphic\_1" Height="550" Width="600" Margin="0,30,593.6,190.4">

<Line X1 = "60" Y1="0" X2="60" Y2="500" Stroke="Black" StrokeThickness="2" />

<Line X1 = "60" Y1="500" X2="550" Y2="500" Stroke="Black" StrokeThickness="2" />

<Polyline Points = "55,5 60,0 65,5" Stroke="Black" StrokeThickness="2"/>

<Polyline Points = "545,495 550,500 545,505" Stroke="Black" StrokeThickness="2"/>

<Label x:Name="label" Content="Мб" Canvas.Left="27" Canvas.Top="-12"/>

<Label x:Name="label\_Copy" Content="Гб" Canvas.Left="636" Canvas.Top="-12"/>

<Label x:Name="label1" Content="Time" Canvas.Left="522" Canvas.Top="501"/>

</Canvas>

<Canvas x:Name="Graphic\_2" Height="550" Width="600" Margin="605,30,-11.4,190.4">

<Line X1 = "60" Y1="0" X2="60" Y2="500" Stroke="Black" StrokeThickness="2" />

<Line X1 = "60" Y1="500" X2="550" Y2="500" Stroke="Black" StrokeThickness="2" />

<Polyline Points = "55,5 60,0 65,5" Stroke="Black" StrokeThickness="2"/>

<Polyline Points = "545,495 550,500 545,505" Stroke="Black" StrokeThickness="2"/>

<Label x:Name="label1\_Copy" Content="Time" Canvas.Left="519" Canvas.Top="501"/>

</Canvas>

<Button x:Name="button" Content="About" HorizontalAlignment="Left" Margin="1057,585,0,0" VerticalAlignment="Top" Width="109" Height="35" FontSize="16" Click="button\_Click"/>

<Label x:Name="CPU" Content="CPU" HorizontalAlignment="Left" Margin="56,569,0,0" VerticalAlignment="Top"/>

<Label x:Name="Dir" Content="Dir" HorizontalAlignment="Left" Margin="56,595,0,0" VerticalAlignment="Top"/>

<Label x:Name="NamePC" Content="NamePC" HorizontalAlignment="Left" Margin="56,621,0,0" VerticalAlignment="Top"/>

<Label x:Name="Asocoation" Content="Asociation" HorizontalAlignment="Left" Margin="56,647,0,0" VerticalAlignment="Top"/>

</Grid>

</Window>

using System;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Media;

using System.Windows.Shapes;

using System.Diagnostics;

using System.Windows.Threading;

using System.IO;

using Microsoft.Win32;

namespace MikhailovRGR

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

class Paint

{

Canvas graphic;

Label Free;

int x = 60;

public Paint(Canvas graph, Label label)

{

this.graphic = graph;

this.Free = label;

Free.Margin = new Thickness(170, 505, 0, 0);

}

public void SetPixelRam(Object source, EventArgs e)

{

//Кол-во свободного ОЗУ

PerformanceCounter ramFree = new PerformanceCounter("Memory", "Available MBytes");

var freeRam = ramFree.NextValue();

Free.Content = "Свободное ОЗУ: " + Convert.ToString(freeRam) + " Мб";

//Создаем эллипс

Ellipse PixelRam = new Ellipse();

PixelRam.Height = 2;

PixelRam.Width = 2;

PixelRam.Stroke = Brushes.Red;

PixelRam.Fill = Brushes.Red;

//Размещение эллипса

int y1 = (int)((freeRam / 8192) \* 500);

int marginLeft = x;

int marginTop = 500 - y1;

PixelRam.Margin = new Thickness(marginLeft, marginTop, 0, 0);

graphic.Children.Add(PixelRam);

x += 2;

}

public void SetPixelMemory(Object source, EventArgs e)

{

//Кол-во свободной памяти на диске D:\

DriveInfo memory = new DriveInfo(@"D:\");

var freeMemory = memory.AvailableFreeSpace;

Free.Content = "Кол-во свободной памяти на диске D: " + Convert.ToString(Math.Round(freeMemory / Math.Pow(1024, 3), 2)) + " Гб";

//Создаем пиксель

Ellipse PixelMemory = new Ellipse();

PixelMemory.Height = 2;

PixelMemory.Width = 2;

PixelMemory.Stroke = Brushes.Red;

PixelMemory.Fill = Brushes.Red;

//Размещение эллипса

int y = (int)(((freeMemory / Math.Pow(1024, 3)) / 542) \* 500);

int marginLeft = x;

int marginTop = 500 - y;

PixelMemory.Margin = new Thickness(marginLeft, marginTop, 0, 0);

graphic.Children.Add(PixelMemory);

x += 2;

}

}

public MainWindow()

{

InitializeComponent();

//Рисуем сетку координат для Graphic\_1

//Ось OX

for (int y = 485; y >= 0; y -= 15)

{

Line OX = new Line();

if (y % 2 == 0)

{

Label text = new Label();

text.Content = Convert.ToString(Math.Round((8192.0 / 500) \* (y + 15)));

int marginTop = 485 - y - 9;

text.Margin = new Thickness(25, marginTop, 0, 0);

Graphic\_1.Children.Add(text);

}

OX.X1 = 61;

OX.X2 = 540;

OX.Y1 = y;

OX.Y2 = y;

OX.Stroke = Brushes.Gray;

Graphic\_1.Children.Add(OX);

}

//Ось OY

for (int x = 75; x <= 550; x += 15)

{

Line OY = new Line();

OY.X1 = x;

OY.X2 = x;

OY.Y1 = 5;

OY.Y2 = 499;

OY.Stroke = Brushes.Gray;

Graphic\_1.Children.Add(OY);

}

//Конец разметки графика Graphic\_1

//Рисуем сетку координат для Graphic\_2

//Ось OX

for (int y = 485; y >= 0; y -= 15)

{

Line OX = new Line();

if (y % 2 == 0)

{

Label text = new Label();

text.Content = Convert.ToString(Math.Round((542.0 / 500.0) \* (y + 15)));

int marginTop = 485 - y - 9;

text.Margin = new Thickness(25, marginTop, 0, 0);

Graphic\_2.Children.Add(text);

}

OX.X1 = 61;

OX.X2 = 540;

OX.Y1 = y;

OX.Y2 = y;

OX.Stroke = Brushes.Gray;

Graphic\_2.Children.Add(OX);

}

//Ось OY

for (int x = 75; x <= 550; x += 15)

{

Line OY = new Line();

OY.X1 = x;

OY.X2 = x;

OY.Y1 = 5;

OY.Y2 = 499;

OY.Stroke = Brushes.Gray;

Graphic\_2.Children.Add(OY);

}

//Конец разметки графика Graphic\_2

Label FreeRam = new Label();

Graphic\_1.Children.Add(FreeRam);

Label FreeMemory = new Label();

Graphic\_2.Children.Add(FreeMemory);

//Работа с таймером

Paint PixelRam = new Paint(Graphic\_1, FreeRam);

Paint PixelMemory = new Paint(Graphic\_2, FreeMemory);

DispatcherTimer Timer = new DispatcherTimer();

Timer.Interval = TimeSpan.FromSeconds(3);

Timer.Tick += PixelRam.SetPixelRam;

Timer.Tick += PixelMemory.SetPixelMemory;

Timer.Start();

//Вывод данных из реестра

CPU.Content = "Мой процессор: " + Registry.GetValue(@"HKEY\_LOCAL\_MACHINE\HARDWARE\DESCRIPTION\System\CentralProcessor\0", "ProcessorNameString", 0);

Dir.Content = "Дериктория Asus Smart Gesture: " + Registry.GetValue(@"HKEY\_LOCAL\_MACHINE\SOFTWARE\Asus\Asus Smart Gesture", "InstalledDir", 0);

NamePC.Content = "Имя компьютера: " + (Registry.GetValue(@"HKEY\_CURRENT\_USER\Volatile Environment", "USERNAME", 0));

Asocoation.Content = "Ассоциация с .doc файлами: " + (Registry.GetValue(@"HKEY\_CLASSES\_ROOT\.doc", "", 0));

}

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

{

AboutWin okno = new AboutWin();

okno.Show();

}

}

}

**Окно About:**

<Window x:Name="WindowAbout" x:Class="MikhailovRGR.AboutWin"

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:MikhailovRGR"

mc:Ignorable="d"

Title="About" Height="500" Width="800" Visibility="Visible">

<Grid Margin="0,0,-0.4,-1.6">

<Image x:Name="tmpwinjpg" Margin="52,15,653.4,328.6" Source="tmpwin.jpg" Stretch="Fill"/>

<TextBlock x:Name="textBlock" HorizontalAlignment="Left" Margin="52,148,0,-0.4" TextWrapping="Wrap" VerticalAlignment="Top" Height="324" Width="663" FontFamily="Times New Roman" FontSize="16"><Run Text="Задание РГР:"/><LineBreak/><Run Text="1. Создать оконное приложение на базе WinApi."/><LineBreak/><Run Text="2. Установить таймер, срабатывающий каждые N"/><Run Text="(3)"/><Run Text=" секунд. Значение N определяется вариантом задания."/><LineBreak/><Run Text="3. Отобразить два графика. На каждом из них отображаются последние M"/><Run Text="(40)"/><Run Text=" значений. Значения считываются по событиям таймера. "/><LineBreak/><Run Text="Все графики отображаются в главном окне."/><LineBreak/><Run Text="Вид графика "/><Run Text="- "/><Run Text="Точечный."/><LineBreak/><Run Text="Содержимое графика №1 "/><Run Text="- "/><Run Text="Объем свободной оперативной памяти."/><LineBreak/><Run Text="Содержимое графика №2"/><Run Text=" - "/><Run Text="Объем свободного места на диске"/><LineBreak/><Run Text="4. Создать диалоговое окно, в котором отображается задание и информация об авторе с фотографией."/><LineBreak/><Run Text="5. Прочесть информацию из реестра Windows, относящуюся к самостоятельно выбранному аппаратному и программному обеспечению, а также к настройкам, относящимся к компьютеру в целом и к отдельному пользоват"/><Run Text="е"/><Run Text="лю."/><LineBreak/><Run Text="6. Прочесть информацию из реестра Windows, определяющую ассоциации типа документа"/><Run Text="("/><Run Text="DOC"/><Run Text=")"/><Run Text=". Тип документа определяется вариантом задания."/><LineBreak/><Run Text=""/></TextBlock>

<Button x:Name="Ok" Content="OK" HorizontalAlignment="Left" Margin="697,442,0,0" VerticalAlignment="Top" Width="85" Height="20" Click="Ok\_Click"/>

<TextBlock x:Name="InfoAboutStudent" HorizontalAlignment="Left" Margin="160,32,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Height="48" Width="228" FontFamily="Times New Roman" FontSize="16"><Run Text="Михайлов Олег Михайлович"/><LineBreak/><Run Text="Студент гр. АЕ-145"/></TextBlock>

</Grid>

</Window>

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.Shapes;

namespace MikhailovRGR

{

/// <summary>

/// Логика взаимодействия для AboutWin.xaml

/// </summary>

public partial class AboutWin : Window

{

public AboutWin()

{

InitializeComponent();

}

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

{

WindowAbout.Visibility = Visibility.Collapsed;

}

}

}