|  |
| --- |
| **МИНОБРНАУКИ РОССИИ**  федеральное государственное бюджетное образовательное  учреждение высшего образования  **«Национальный исследовательский университет «МЭИ»** |

|  |  |
| --- | --- |
| **Институт** | ИВТИ |
| **Кафедра** | ПМИИ |

|  |  |
| --- | --- |
|  |  |
| **Дисциплина: «Системное программирование»**  **Отчет по лабораторной работе №8**  **«Процессы и потоки»**  **Выполнил: студент группы А-13а-19**  **Рамазанов Н. М.**  **Преподаватель: Меньшикова К. Г.**  **Москва, 2021** | |

**Выполнение работы.**

Для выполнения работы создал приложение на языке C#.

1. Программа должна выдать информацию о первичном (главном) потоке, создать еще три потока.

Необходимые переменные:

private Thread CatThread, DogThread, SystemInfoThread;

Все делаем по кнопке:

private void button1\_Click(object sender, EventArgs e)

{

var mainThread = Thread.CurrentThread;

mainThreadInfo.Text = "Имя домена: " + Thread.GetDomain().FriendlyName +

"\nУровень приоритета: " + mainThread.Priority +

"\nСостояние потока: " + mainThread.ThreadState +

"\nИдентификатор контекста: " + Thread.CurrentContext.ContextID +

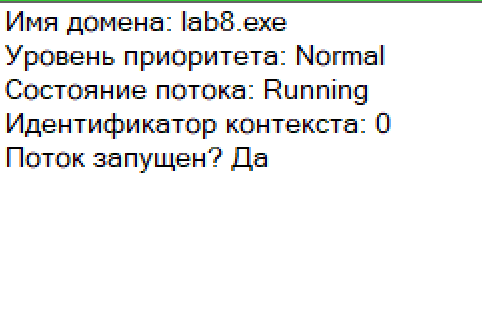
"\nПоток запущен? " + (mainThread.IsAlive ? "Да" : "Нет");

CatThread = new Thread(CatProcess);

DogThread = new Thread(DogProcess);

SystemInfoThread = new Thread(ShowSystemInfo);

}



Функции процессов приведу ниже (в пунктах 2, 4).

1. Два вторичных потока должны быть созданы на основе двух разных функций, каждая из которых реализует метод случайного представления фрагментов одной из двух картинок (**каждая картинка режется на квадратики**). Поток должен завершиться, если «соберет» свою картинку в правильном порядке.

Необходимые переменные:

private readonly List<Bitmap> CatImgs = new List<Bitmap>();

private readonly List<Bitmap> DogImgs = new List<Bitmap>();

private List<int> CatReshuffle = new List<int>();

private List<int> DogReshuffle = new List<int>();

private readonly Graphics CatGraph, DogGraph;

private readonly List<int> RightReshuffle = new List<int> { 0, 1, 2, 3 };

private readonly Stopwatch CatSW = new Stopwatch();

private readonly Stopwatch DogSW = new Stopwatch();

private readonly Stopwatch SystemInfoSW = new Stopwatch();

private TimeSpan CatTS, DogTS, SystemInfoTS;

Реализуем две различных функции для перестановок картинок: обычная перестановка с помощью вспомогательной переменной, и перестановка, основанная на использовании структуры Set (множество) и его свойств.

private void MakeReshuffleBySet(ref List<int> ind)

{

var rnd = new Random();

var set = new HashSet<int>(RightReshuffle);

for (int i = 0; i < ind.Count; ++i)

{

var index = rnd.Next(set.Count);

var enumerator = set.GetEnumerator();

for (int j = 0; j <= index; ++j)

{

enumerator.MoveNext();

}

ind[i] = enumerator.Current;

set.Remove(enumerator.Current);

enumerator.Dispose();

}

}

private void MakeReshuffle(ref List<int> ind)

{

var rnd = new Random();

for (var i = ind.Count - 1; i >= 0; i--)

{

var j = rnd.Next(i + 1);

var tmp = ind[i];

ind[i] = ind[j];

ind[j] = tmp;

}

}

Непосредственно функции процессов:

private void CatProcess()

{

CatSW.Restart();

CatReshuffle.Clear();

CatReshuffle.AddRange(RightReshuffle);

do

{

MakeReshuffle(ref CatReshuffle);

DrawImageCat(CatGraph, CatImgs, CatReshuffle);

Thread.Sleep(50);

} while (!CatReshuffle.SequenceEqual(RightReshuffle));

CatTS = CatSW.Elapsed;

}

private void DogProcess()

{

DogSW.Restart();

DogReshuffle.Clear();

DogReshuffle.AddRange(RightReshuffle);

do

{

MakeReshuffleBySet(ref DogReshuffle);

DrawImageDog(DogGraph, DogImgs, DogReshuffle);

Thread.Sleep(10);

} while (!DogReshuffle.SequenceEqual(RightReshuffle));

DogTS = DogSW.Elapsed;

}

1. Приложение должно выводить на экран эти «разрезанные» картинки на экран (в выбранном в потоке порядке).

Функция загрузки картинок:

private void LoadImages()

{

for (var i = 1; i <= 4; i++)

{

CatImgs.Add(new Bitmap(@"D:\учеба\5th\_semester\СистПрог\lab8\cat" + i + ".bmp"));

DogImgs.Add(new Bitmap(@"D:\учеба\5th\_semester\СистПрог\lab8\dog" + i + ".bmp"));

}

}

В конструктор формы Form1() добавим загрузку картинок и создание объектов Graphics в PictureBox’ах:

LoadImages();

CatGraph = pictureBox1.CreateGraphics();

DogGraph = pictureBox2.CreateGraphics();

Функции вывода картинок:

private static void DrawImageCat(Graphics graphics, IReadOnlyList<Bitmap> image, IReadOnlyList<int> reshuffle)

{

var width = image[0].Width;

var height = image[0].Height;

graphics.DrawImage(image[reshuffle[0]], new Point(0, 0));

graphics.DrawImage(image[reshuffle[1]], new Point(width, 0));

graphics.DrawImage(image[reshuffle[2]], new Point(0, height));

graphics.DrawImage(image[reshuffle[3]], new Point(width, height));

}

private static void DrawImageDog(Graphics graphics, IReadOnlyList<Bitmap> image, IReadOnlyList<int> reshuffle)

{

var width = image[0].Width;

var height = image[0].Height;

graphics.DrawImage(image[reshuffle[0]], new Point(0, 0));

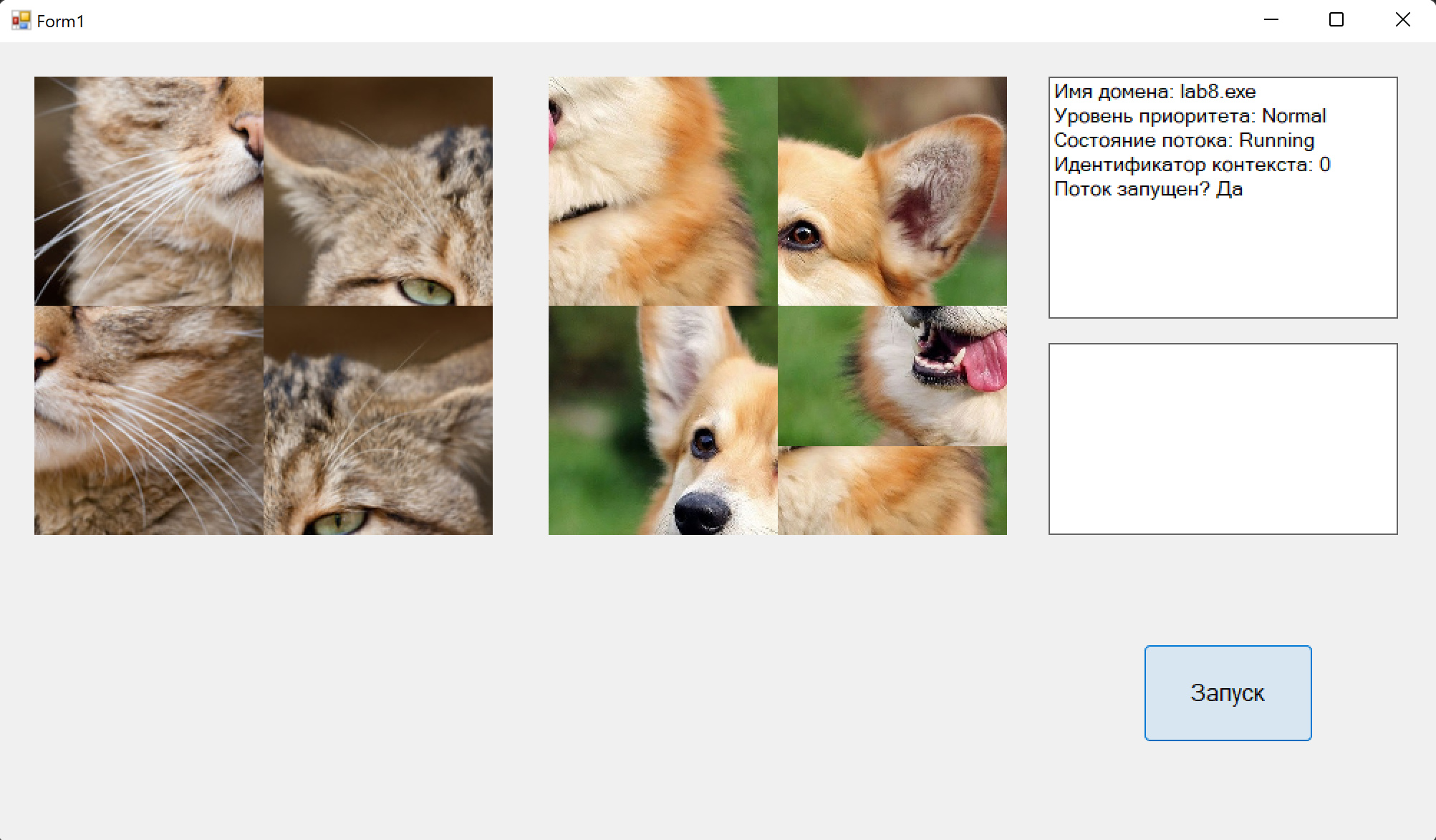
graphics.DrawImage(image[reshuffle[1]], new Point(width / 2, 0));

graphics.DrawImage(image[reshuffle[2]], new Point(0, height / 2));

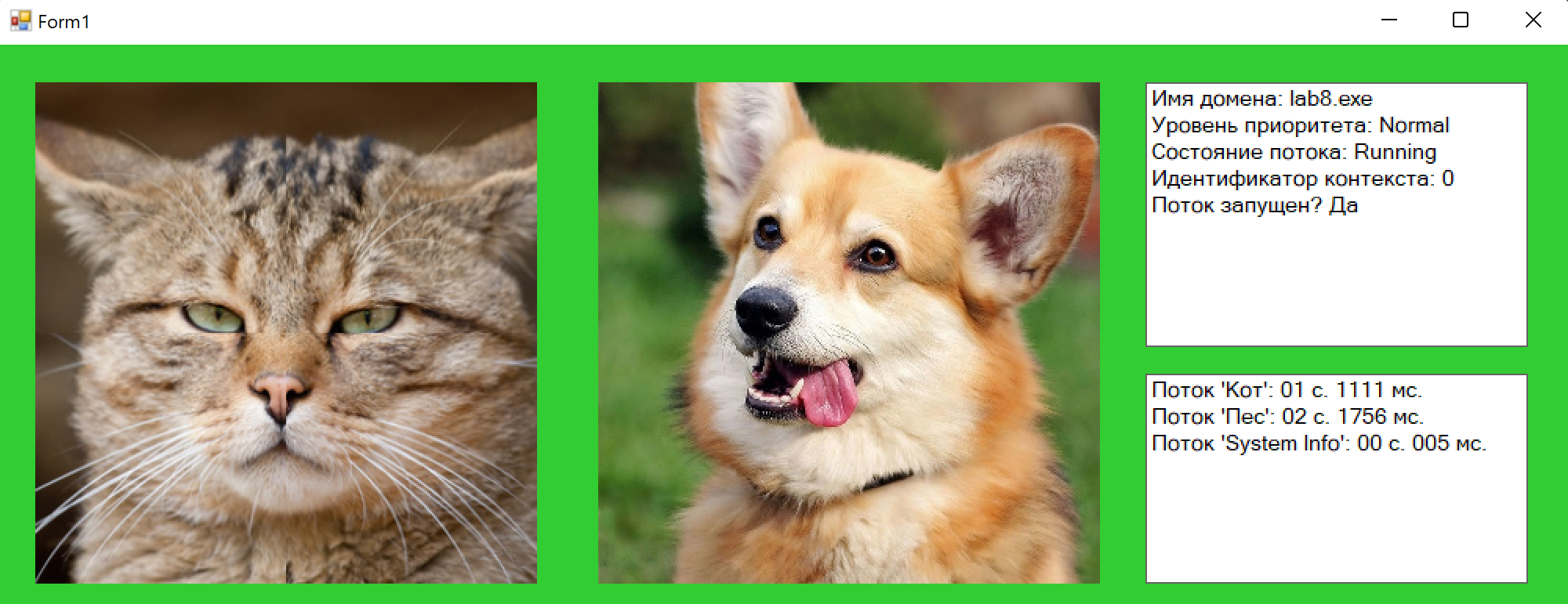
graphics.DrawImage(image[reshuffle[3]], new Point(width/2, height / 2));

}

Демонстрация работы во время перестановок:



После перестановок:



1. Третий поток должен собрать информацию о компьютерной системе (не менее 10-ти параметров), подготовить картинку для демонстрации этой информации на экране (**в виде рекламы или презентации**) и ждать события о завершении двух предыдущих потоков (информация о системе должна быть представлена пользователю, когда произойдет завершение двух потоков).

Для того чтобы информация о системе вывелась только после того, как закончатся 2 предыдущих потока, добавляем в функцию вывода информации ShowSystemInfo перед непосредственным выводом строки:

CatThread.Join();

DogThread.Join();

Эти команды блокируют дальнейшее выполнение кода, пока оба потока не будут завершены.

private string GetSystemInfo()

{

SystemInfoSW.Restart();

var output = new StringBuilder();

output.AppendFormat("Версия Windows: {0}\n", Environment.OSVersion);

output.AppendFormat("64 Bit операционная система? {0}\n", Environment.Is64BitOperatingSystem ? "Да" : "Нет");

output.AppendFormat("Имя компьютера: {0}\n", Environment.MachineName);

output.AppendFormat("Число процессоров: {0}\n", Environment.ProcessorCount);

output.AppendFormat("Системная папка: {0}\n", Environment.SystemDirectory);

output.AppendFormat("Текущая директория: {0}\n", Environment.CurrentDirectory);

output.AppendFormat("Имя домена текущего пользователя: {0}\n", Environment.UserDomainName);

output.AppendFormat("Текущий процесс в режиме взаимодействия с пользователем? {0}\n",

Environment.UserInteractive ? "Да" : "Нет");

output.AppendFormat("Объем физической памяти: {0}\n", Environment.WorkingSet);

output.AppendFormat("Логические диски: {0}\n",

string.Join(", ", Environment.GetLogicalDrives())

.Replace("\\", string.Empty));

SystemInfoTS = SystemInfoSW.Elapsed;

return output.ToString();

}

private void ShowSystemInfo()

{

var info = GetSystemInfo();

CatThread.Join();

DogThread.Join();

ShowInfo(info);

}

private void ShowInfo(string info)

{

var graphicWnd = CreateGraphics();

var font = new Font("Arial", 11, FontStyle.Bold);

graphicWnd.Clear(Color.LimeGreen);

Bitmap info\_bmp = new Bitmap(@"D:\учеба\5th\_semester\СистПрог\lab8\info1.bmp");

graphicWnd.DrawImage(info\_bmp, new Point(10, 350));

graphicWnd.DrawString(info, font, Brushes.Black, 25, 370);

}

Демонстрация – после завершения работы потоков выводится рамочка, внутри нее выводится информация о системе, а фон окна окрашивается в зеленый цвет, символизирующий успешное завершение работы.



1. Программа должна также предоставить пользователю информацию о времени, в течении которого выполнялись потоки.

Дополним код кнопки (после создания трех потоков):

CatThread.Start();

DogThread.Start();

SystemInfoThread.Start();

CatThread.Join();

DogThread.Join();

SystemInfoThread.Join();

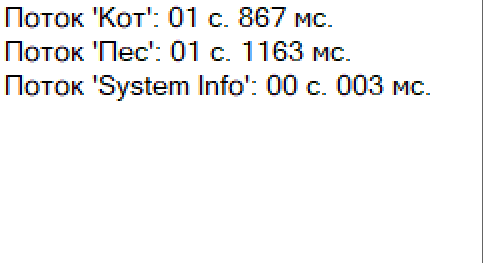
var threadsTime = $"Поток 'Кот': {CatTS.TotalSeconds:00} c. {CatTS.TotalMilliseconds:000} мс.\n";

threadsTime += $"Поток 'Пес': {DogTS.TotalSeconds:00} с. {DogTS.TotalMilliseconds:000} мс.\n";

threadsTime += $"Поток 'System Info': {SystemInfoTS.TotalSeconds:00} с. {SystemInfoTS.TotalMilliseconds:000} мс.\n";

timeInfo.Text = threadsTime;

Демонстрация:



Полный код программы доступен по ссылке: <https://github.com/programzan/SystemProgramming/blob/lab8/Form1.cs>

Здесь же размещены все сопутствующие файлы: <https://github.com/programzan/SystemProgramming/tree/lab8>