## Лабораторная работа 8

## Процессы и потоки

### Подготовка к лабораторной работе

Для выполнения лабораторной работы необходимо ознакомиться с примерами, запустить их на выполнение и объяснить результаты работы.

# Создание потоков и их синхронизация

using System.Threading;

namespace WFA\_inf\_potoki

{ public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

**Пример 1.**

## // Создание вторичного потока (на основе метода класса Potok)

Potok tc1;

Thread t1;

int flag = 0;

private void button2\_Click(object sender, EventArgs e)

{ tc1 = new Potok();

t1 = new Thread(new ThreadStart(tc1.Execute));

t1.Name = "Вторичный поток";

t1.Start();

MessageBox.Show("Вторичный поток запущен",

"Сообщение первичного потока");

timer1.Start();

}

// На форму вставить компонент Таймер

private void timer1\_Tick(object sender, EventArgs e)

{

if (tc1 != null)

label1.Text = "Сумма: " + tc1.a.ToString();

if (flag == 0)

{

flag = 1;

MessageBox.Show("Имя домена: " + Thread.GetDomain().FriendlyName +

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

"\nИмя потока: " + t1.Name + "\nЗапущен ли поток? " + t1.IsAlive.ToString()

+ "\nУровень приоритета: " + t1.Priority.ToString() +

"\nСостояние потока: " + t1.ThreadState.ToString());

}

}

private void button3\_Click(object sender, EventArgs e)

{

if (tc1 != null)

tc1.a = tc1.a + 100;

label1.Text = "Сумма: " + tc1.a.ToString();

}

//------------------------------------------------------------------------------------------

**Пример 2**

// Запуск пяти потоков на основе одной функции и использование общих ресурсов

// Запустить и объяснить как это работает (с комментариями и без)

private void button5\_Click(object sender, EventArgs e)

{

Thread[] threads = new Thread[5];

for (int i = 0; i < 5; i++)

{

threads[i] = new Thread(new ThreadStart(myPr));

threads[i].Name = "Поток " + i.ToString();

}

foreach (Thread t in threads)

t.Start();

}

object thlock = new object();

int a = 1;

int sum = 0;

public void myPr()

{ string str="";

// lock (thlock)

{

{

for (int i = 0; i < 10; i++)

{

sum = sum + a;

// str = str + i.ToString() + " ";

Thread.Sleep(5);

//Thread.Sleep(20);

// Thread.Sleep(50);

}

a++;

str = sum.ToString();

MessageBox.Show(str,Thread.CurrentThread.Name);

}

}

}

//------------------------------------------------------------------------------------------------------------------------------

// класс метод, которого является началом потока

public class Potok

{

public int a=0;

public void Execute()

{

int i;

for (i = 1; i <= 10; i++)

{

a = a + i;

Thread.Sleep(1000);

}

Thread p = Thread.CurrentThread;

MessageBox.Show("Имя домена: " + Thread.GetDomain().FriendlyName + "\nИдентификатор контекста: " + Thread.CurrentContext.ContextID +

"\nИмя потока: " + p.Name + "\nЗапущен ли поток? " + p.IsAlive.ToString() + "\nУровень приоритета: " + p.Priority.ToString() + "\nСостояние потока: " + p.ThreadState.ToString());

MessageBox.Show("a=", a.ToString());

}

}

}

## Задание

Для выполнения работы создать приложение на языке C#.

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

Thread mainThread = Thread.CurrentThread;

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

Thread.CurrentContext.ContextID + "\nЗапущен ли поток? " +

mainThread.IsAlive.ToString() + "\nУровень приоритета: " + mainThread.Priority.ToString() + "\nСостояние потока: " +

mainThread.ThreadState.ToString() + "\n";

Thread thread1 = new Thread(new ThreadStart(GenerateAndDrawFirstPermutation));

Thread thread2 = new Thread(new ThreadStart(GenerateAndDrawSecondPermutation));  
 Thread thread3 = new Thread(new ThreadStart(CollectInfoAboutSystem));

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

void DrawImageByPermutation(Graphics graphics, List<Bitmap> image, List<int> ind)

{

int width = image[0].Width;

int height = image[0].Height;

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

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

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

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

}

private void MakePermutation(ref List<int> ind)

{

Random rnd = new Random();

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

{

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

int tmp = ind[i];

ind[i] = ind[j];

ind[j] = tmp;

}

}

private void GenerateAndDrawFirstPermutation()

{

permutation1.AddRange(truePermutation);

MakePermutation(ref permutation1);

while (!permutation1.SequenceEqual(truePermutation))

{

MakePermutation(ref permutation1);

DrawImageByPermutation(picBox1Graphics, image1Parts, permutation1);

Thread.Sleep(50);

}

}

private void GenerateAndDrawSecondPermutation()

{

permutation2.AddRange(truePermutation);

MakePermutation(ref permutation2);

while (!permutation2.SequenceEqual(truePermutation))

{

MakePermutation(ref permutation2);

DrawImageByPermutation(picBox2Graphics, image2Parts, permutation2);

Thread.Sleep(100);

}

}

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).

private string CollectInfoAboutSystem()

{

StringBuilder output = new StringBuilder();

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

output.AppendFormat("64 Bit операционная система? : {0}\n", Environment.Is64BitOperatingSystem ? "Yes" : "No");

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("UserDomainName: {0}\n", Environment.UserDomainName);

output.AppendFormat("UserInteractive: {0}\n", Environment.UserInteractive);

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

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

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

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

return output.ToString();

}

private void outputInfoAboutSystem()

{

string info = CollectInfoAboutSystem();

thread1.Join();

watchThread1.Stop();

thread2.Join();

watchThread2.Stop();

MakePresentation(info);

}

private void MakePresentation(string info)

{

Graphics graphicWnd = this.CreateGraphics();

Font font = new Font("Times New Roman", 11, FontStyle.Bold);

graphicWnd.Clear(Color.White);

graphicWnd.DrawString(info, font, Brushes.Black, 10, 580);

}

}

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

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

{

DrawImage = !DrawImage;

if (DrawImage)

Main();

else

{

permutation1.Clear();

permutation2.Clear();

mainThreadInfo.Text = "";

timeInfo.Text = "";

Refresh();

}

}

private void Main()

{

Thread mainThread = Thread.CurrentThread;

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

Thread.CurrentContext.ContextID + "\nЗапущен ли поток? " +

mainThread.IsAlive.ToString() + "\nУровень приоритета: " + mainThread.Priority.ToString() + "\nСостояние потока: " +

mainThread.ThreadState.ToString() + "\n";

thread1 = new Thread(new ThreadStart(GenerateAndDrawFirstPermutation));

thread2 = new Thread(new ThreadStart(GenerateAndDrawSecondPermutation));

thread3 = new Thread(new ThreadStart(outputInfoAboutSystem));

watchThread1.Start();

thread1.Start();

watchThread2.Start();

thread2.Start();

watchThread3.Start();

thread3.Start();

thread3.Join();

watchThread3.Stop();

TimeSpan ts1 = watchThread1.Elapsed;

TimeSpan ts2 = watchThread2.Elapsed;

TimeSpan ts3 = watchThread3.Elapsed;

string threadsTime;

threadsTime = String.Format("Thread1: {0:00}s.{1:000}ms\n", ts1.Seconds, ts1.Milliseconds);

threadsTime += String.Format("Thread2: {0:00}s.{1:000}ms\n", ts2.Seconds, ts2.Milliseconds);

threadsTime += String.Format("Thread3: {0:00}s.{1:000}ms\n", ts3.Seconds, ts3.Milliseconds);

timeInfo.Text = threadsTime;

}