

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Факультет Информатики   
Кафедра Программных систем

**ОТЧЕТ**  
  
по лабораторному практикуму по дисциплине

«Компьютерные сети»

Лабораторная работа № 2

Вариант 3

Студент В.Д. Гижевская

Руководитель А. С. Пигусов

Самара 2020

**Цель работы**

Освоение принципов работы протокола прикладного уровня HTTP, приобретение навыков формирования и обработки запросов информации по протоколу HTTP.

**Задание**

Создать сетевое клиентское приложение, реализующее обращения к www-серверу по протоколу HTTP и производящее обработку полученных данных. Приложение должно выполнять функции, изложенные в задании, и обладать дружественным к пользователю интерфейсом.

Для формирования запросов и получения ответов сервера запрещается использование стандартных либо реализованных сторонними разработчиками компонент и функций, исполняющих протоколы прикладного уровня.

Адрес www-сервера и номер порта (по умолчанию 80) вводятся пользователем. Адрес может быть записан как в виде IP-адреса, так и в символьной форме (например, 89.186.244.16 или www.ssau.ru).

Приложение должно включать контроль ошибок при вводе и обработке запросов.

**Вариант задания**

Определить и вывести суммарный размер всех страниц www-сервера и их количество. Вывести адреса страниц, имеющих наибольший и наименьший размеры

**Листинг программы**

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HttpParser

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

**Form1.cs**

using System;

using System.Collections.Generic;

using System.Net;

using System.Windows.Forms;

using System.Linq;

namespace HttpParser

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

tbInfo.Clear();

tbLinks.Clear();

tbStatus.Clear();

int deepCount = (int)nUDDeep.Value;

int port = (int)nUDPort.Value;

string url = tBLink.Text;

if (!url.StartsWith("http"))

{

url = "http://" + url;

}

if (Uri.TryCreate(url, UriKind.Absolute, out var uri) &&

uri.Scheme == Uri.UriSchemeHttp)//проверка что вначале http

{

ParsingInfo info = ProcessPages(uri, port, deepCount);

tbInfo.Text +=

"Количество обработанных страниц: " + (info.PagesCount - 1) + "\n" +

"Общий размер: " + info.TotalSize + " байт\n" +

"Минимальный размер: " + info.MinURL + "\n(" + info.MinSize + " байт)\n" +

"Максимальный размер: " + info.MaxURL + "\n(" + info.MaxSize + " байт)\n";

tbStatus.Text += info.Status;

}

else tbStatus.Text = (" Не удаётся обработать сайт.Проверьте корректность ввода.");

}

ParsingInfo ProcessPages(Uri startURL, int port, int maxLevel)//обработка страницы

{

List<Uri> urisList = new List<Uri>();// Общий промежуточный список

List<Uri> totalUriList = new List<Uri>();// Общий список

string minURL = "";

string maxURL = "";

int minSize = 0;

int maxSize = 0;

int pagesCount = 1;

int totalSize = 0;

urisList.Add(startURL);//добавляю стартовую ссылку

totalUriList.Add(startURL);

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

{

List<Uri> processingURLs = new List<Uri>(); //Список страниц на парсинг

foreach (var uri in urisList)

{

ResultStatus get\_page;//Результат обработки страницы (получаю путь)

string curPageResponse = "";//Ответ сервера по странице (сюда будет записываться ответ сервера)

try

{

var path = uri.ToString().Split('/'); //выделяем путь

get\_page = HttpRequest.GetPage(uri.Host, "/" + path[3], port);//передаём хост и путь

if (get\_page.Length == -1 || get\_page.Result == null)//проверка

{

return new ParsingInfo(

minSize,

maxSize,

minURL,

maxURL,

pagesCount,

totalSize,

get\_page.Status);

}

curPageResponse = get\_page.Result;//ответ сервера

totalSize += get\_page.Length;//прибавляем размер страницы

if (minSize > get\_page.Length || minSize == 0)//нахождение минимальной и максимальной

{

minSize = get\_page.Length;

minURL = uri.ToString();

}

if (maxSize < get\_page.Length)

{

maxSize = get\_page.Length;

maxURL = uri.ToString();

}

}

catch (Exception ex)

{

return new ParsingInfo(

minSize,

maxSize,

minURL,

maxURL,

pagesCount,

totalSize,

ex.Message);

}

var PageProcessingResult = HtmlParse.GetLinksHTML(curPageResponse, uri, out List<Uri> curLinks);//передаю на обработку ответ сервера, адрес и выходной список ссылок

if (curLinks == null)

{

return new ParsingInfo(

minSize,

maxSize,

minURL,

maxURL,

pagesCount,

totalSize,

PageProcessingResult);

}

else

{

foreach (var curLink in curLinks)//проверка на совпадения и принадлежность ссылки к серверу

{

if (!processingURLs.Contains(curLink) && startURL.Host == curLink.Host && !totalUriList.Contains(curLink))

{

processingURLs.Add(curLink);//добавляем

tbLinks.Text += (pagesCount++) + ". " + curLink.ToString() + "\n";//выводим ссылку

}

}

}

}

urisList = processingURLs;//проходимся по новым ссылкам

totalUriList.AddRange(processingURLs);//добавляем полученные ссылки в общий список

}

return new ParsingInfo(

minSize,

maxSize,

minURL,

maxURL,

pagesCount,

totalSize,

"Страница успешно обработана!");

}

private void Form1\_Load(object sender, EventArgs e)

{

}

}

}

**HtmlRequest.cs**

using System;

using System.Net;

using System.Net.Sockets;

using System.Text;

namespace HttpParser

{

class HttpRequest//отправляет запрос на сервер и получает ответ

{

const int BUFFER\_SIZE = 1024;

//Открытие сокета

static string ConnectSocket(string address, int port, out Socket socket)//открывает сокеты, и если всё нормально и всё подключается к серверу, то продолжаем

{

socket = null;

Socket skt = new Socket(

AddressFamily.InterNetwork,

SocketType.Stream,

ProtocolType.Tcp);

try

{

skt.Connect(address, port);

}

catch (Exception e)

{

return e.Message;

}

if (skt.Connected)

{

socket = skt;

return "OK";

}

return "Not Connected";

;

}

//Отправка запроса,получение ответа от сервера

public static ResultStatus GetPage(string host,string path, int port)//получает ответ с сервера через открытый сокет

{

var request = GetRequest(host, path);

string page = "";

byte[] data = Encoding.Default.GetBytes(request);//массив байтов

byte[] bytesReveived = new byte[BUFFER\_SIZE];//сда будем записывать ответ

string SocketConnectionStatus = ConnectSocket(host, port,out Socket socket);//возвращаем сокет

if (socket == null)

{

return new ResultStatus(page, SocketConnectionStatus,-1);

}

try

{

int bytesSent = socket.Send(data);//отправляем запрос серверу

int bytes = 0;

do

{

bytes = socket.Receive(bytesReveived,0, bytesReveived.Length,0);//получение ответа построчно (количество полученных байтов)

page += Encoding.Default.GetString(bytesReveived,0,bytes);//накапливаем ответ сервера построчно

}

while (bytes > 0);

socket.Shutdown(SocketShutdown.Both);//закрываем сокеты

socket.Close();

int pageSize = page.Length - page.IndexOf("Content-Type: ") - 14;//получаем размер страницы (по количеству символов)

}

catch (Exception e)

{

return new ResultStatus(page, e.Message,-1);

}

if (page == null)

return new ResultStatus(page, " Страница не была загружена ", 0);

return new ResultStatus(page, " Страница успешно загружена ", page.Length);

}

private static string GetRequest(string host,string path)//формирует запрос к серверу (сервер, страница на сервере)

{

string rn = "\r\n";

string request = "";

request = path != null ?

"GET " + path + " HTTP/1.1" + rn +

"Host: " + host + rn +

"Connection: Close" + rn + rn :

"GET / HTTP/1.1" + rn +

"Host: " + host + rn +

"Connection: Close" + rn + rn;

return request;

}

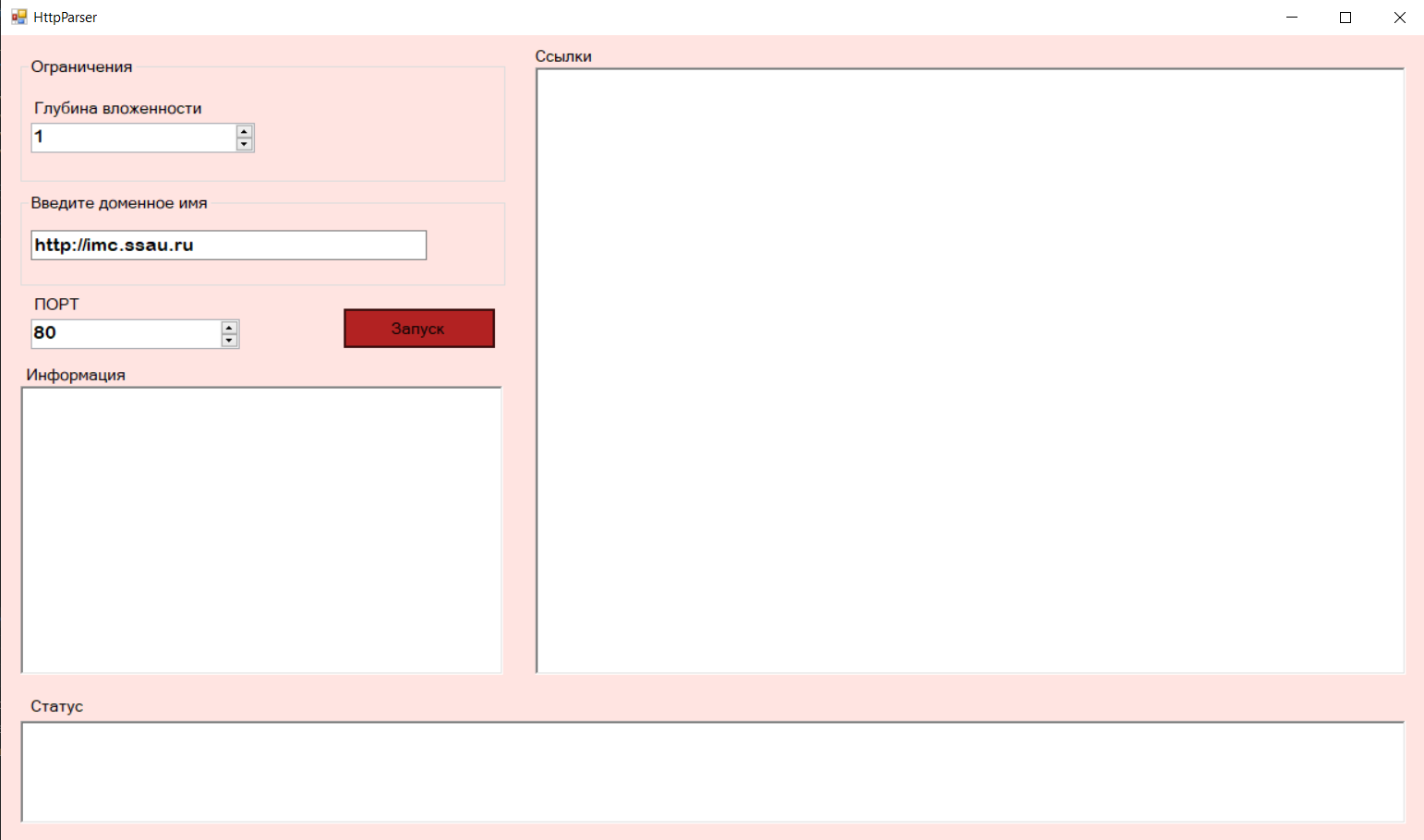
}

}

**Результаты работы программы**

Данная программа определяет и выводит суммарный размер всех страниц www-сервера и их количество. И выводит адреса страниц, имеющих наибольший и наименьший размеры

На рисунке 1 представлен интерфейс программы.

  
Рисунок 1 – Интерфейс программы

На рисунке 2 представлен результат работы программы. В качестве входных параметров были заданы:

* Адрес www-сервера: http://imc.ssau.ru/ (по умолчанию);
* Количество ссылок: 80;

В качестве выходных параметров было получено:

* Найденные ссылки:

1. http://imc.ssau.ru/index.php

2. http://imc.ssau.ru/home

3. http://imc.ssau.ru/resursy

4. http://imc.ssau.ru/uslugi

5. http://imc.ssau.ru/partnery

6. http://imc.ssau.ru/poisk

7. http://imc.ssau.ru/kontakty

8. http://imc.ssau.ru/po-mediatsentra

9. <http://imc.ssau.ru/konferents-zaly>

* Найденные характеристики:

Количество обработанных страниц: 9

Общий размер: 17511 байт

Минимальный размер: <http://imc.ssau.ru/> (17511 байт)

Максимальный размер: <http://imc.ssau.ru/> (17511 байт)

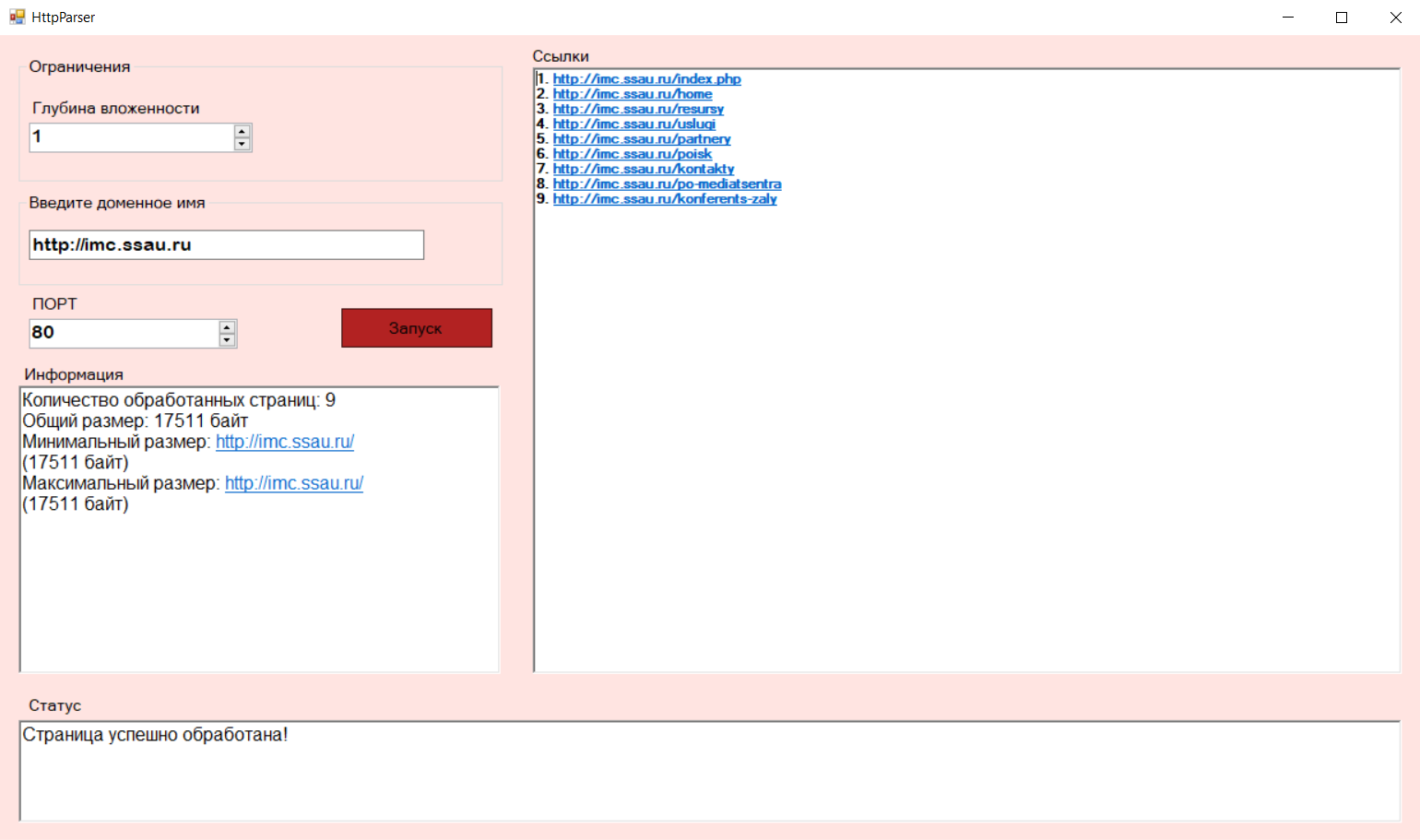


Рисунок 2 – Результат работы программы