Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Факультет электронно-информационных систем

Кафедра ИИТ

Лабораторная работа №5

за 5 семестр

По дисциплине: «ОСиСП»

Выполнила:

студентка 3 курса

группы ПО-4(1)

Боровский М.В.

Проверила:

Дряпко А.В.

2021

Лабораторная работа №5

Цель работы: познакомиться с возможностями, предлагаемыми фреймворком Qt, для разработки многопоточных приложений.

Вариант 3

1) Основное задание заключается в доработке функционала обновления, разработка которого производилась 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 архитектуры);

2) Проверка обновления должна осуществляться автоматически по таймеру (QTimer) либо по

непосредственному запросу пользователя. Предусмотреть выбор из меню политики обновления (с пользовательским подтверждением, без подтверждения/автоматически);

3) Сам процесс обновления должен осуществляться с использованием отдельного потока (QThread) с минимальной вовлечённостью пользователя;

4) Необходимо отображать прогресс обновления (для этого можно использовать строку состояния – QStatusBar);

5) Для демонстрации процесса обновления и независимой работы основного и вспомогательного потоков приложения осуществлять передачу с сервера обновления помимо основных обновляемых компонентов (в соответствии с вариантом задания) одного-двух крупных файлов с произвольным содержимым (например, видео).

6) Обновляемые компоненты по вариантам (ЛР No4):

7) Процесс обновления логируется. При завершении обновления пользователю выдается соответствующее сообщение.

**Текст программы:**

**private void Send()**

{

string IPAddress = "127.0.0.1";

int Port = 9000;

string Filename = @"D:\\Server\\Update";

int bufferSize = 1024;

byte[] buffer = null;

byte[] header = null;

FileStream fs = new FileStream(Filename, FileMode.Open);

bool read = true;

int bufferCount = Convert.ToInt32(Math.Ceiling((double)fs.Length / (double)bufferSize));

TcpClient tcpClient = new TcpClient(IPAddress, Port);

tcpClient.SendTimeout = 600000;

tcpClient.ReceiveTimeout = 600000;

string headerStr = "Content-length:" + fs.Length.ToString() + "\r\nFilename:" + @"D:\\Update" + "Update\r\n";

header = new byte[bufferSize];

Array.Copy(Encoding.ASCII.GetBytes(headerStr), header, Encoding.ASCII.GetBytes(headerStr).Length);

tcpClient.Client.Send(header);

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

{

buffer = new byte[bufferSize];

int size = fs.Read(buffer, 0, bufferSize);

tcpClient.Client.Send(buffer, size, SocketFlags.Partial);

}

tcpClient.Client.Close();

fs.Close();

}

**public void ReciveData()**

    {

        int Port = 9000;

        TcpListener listener = new TcpListener(IPAddress.Any, Port);

        listener.Start();

        Socket socket = listener.AcceptSocket();

        int bufferSize = 1024;

        byte[] buffer = null;

        byte[] header = null;

        string headerStr = "";

        string filename = "";

        int filesize = 0;

        header = new byte[bufferSize];

        socket.Receive(header);

        headerStr = Encoding.ASCII.GetString(header);

        string[] splitted = headerStr.Split(new string[] { "\r\n" }, StringSplitOptions.None);

        Dictionary<string, string> headers = new Dictionary<string, string>();

        foreach (string s in splitted)

        {

            if (s.Contains(":"))

            {

                headers.Add(s.Substring(0, s.IndexOf(":")), s.Substring(s.IndexOf(":") + 1));

            }

        }

        //Get filesize from header

        filesize = Convert.ToInt32(headers["Content-length"]);

        //Get filename from header

        filename = headers["Filename"];

        int bufferCount = Convert.ToInt32(Math.Ceiling((double)filesize / (double)bufferSize));

        FileStream fs = new FileStream(filename, FileMode.OpenOrCreate);

        while (filesize > 0)

        {

            buffer = new byte[bufferSize];

            int size = socket.Receive(buffer, SocketFlags.Partial);

            fs.Write(buffer, 0, size);

            filesize -= size;

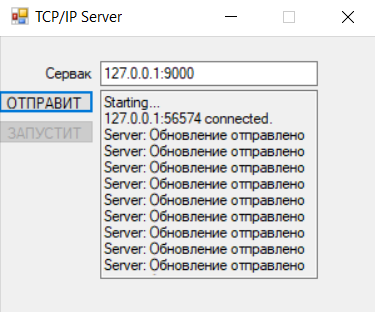
        }

        fs.Close();

    }

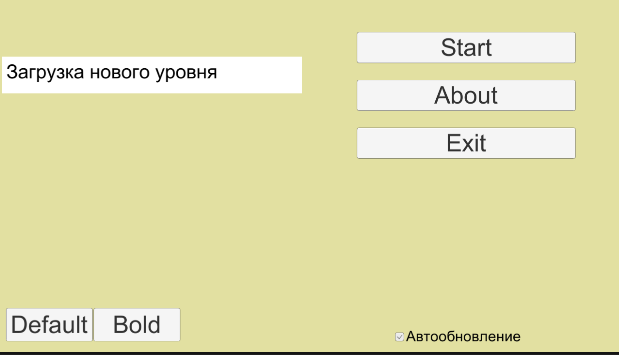
**Результаты тестирования программы:**

Отправление с сервера:

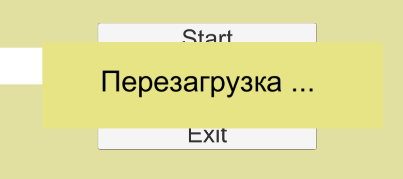




Выводим какие файлы загружаем:

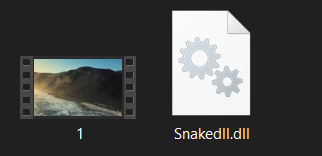


Перезагружаем игру:



Файлы которые перекинулись в игру:





После перезагрузки выводиться окно с информацией обновления: ****