Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики, математики и электроники

Факультет информатики

Кафедра технической кибернетики

Отчет по лабораторной работе №6

Дисциплина: «Операционные системы»

Выполнил: Абрамов Д.О.

Группа: 6208-010302D

Самара, 2021

**Задание на лабораторную работу**

Реализовать графический интерфейс пользователя для функционала межпроцессного взаимодействия ранее сделанной лабораторной работы № 4 или для лабораторной работы № 5. В качестве получения дополнительного бонуса допускается реализация обеих лабораторных работ. Все происходящие события должны документироваться: должен писаться лог, то есть журнал событий – сколько процессов запущено, когда, что передано каким клиентом/процессом на сервер и т.п. Должно быть предусмотрено сохранение лога в файл, а также возможность просмотра лога предыдущей запущенной сессии (допустимо использование для этого разных файлов). Ограничений по языку программирования, технологиям, операционной системе нет.

**Текст программы**

Сервер

Bee.h

#ifndef BEE\_H

#define BEE\_H

#include "mainwindow.h"

#include "ui\_mainwindow.h"

class **Bee** : public QObject

{

Q\_OBJECT

private:

bool \*WinnieThePooh\_sleep\_got\_enough\_sleep;

bool \*Pyatochek\_sleep\_got\_enough\_sleep;

bool \*WinnieThePooh\_sleep;

MainWindow\* mainWindow;

bool \*Pyatochek\_sleep;

int \*barrel\_fullness;

QFile\* file;

QMutex\* mutex;

int index;

int H;

public:

**Bee**(bool&, bool&, bool&, bool&, int&, int, int, MainWindow&, QMutex&, QFile&);

~***Bee***();

**Bee**();

public slots:

void **run**();

signals:

void **finished**();

};

#endif // BEE\_H

mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QFile>

#include <QDebug>

#include <QMutex>

#include <windows.h>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

Ui::MainWindow \*ui;

QFile\* file;

bool flag = false;

private slots:

void on\_pushButton\_clicked();

};

#endif // MAINWINDOW\_H

Pyatochek.h

#ifndef PYATOCHEK\_H

#define PYATOCHEK\_H

#include "mainwindow.h"

#include "ui\_mainwindow.h"

class **Pyatochek** : public QObject

{

Q\_OBJECT

public:

**Pyatochek**(bool&, bool&, int&, int, MainWindow&, QMutex&, QFile&);

~***Pyatochek***();

private:

int \*barrel\_fullness;

int H;

QMutex\* mutex;

QFile\* file;

MainWindow\* mainWindow;

QTextStream\* textStream;

bool \*Pyatochek\_sleep;

bool \*Pyatochek\_sleep\_got\_enough\_sleep;

public slots:

void **run**();

signals:

void **finished**();

};

#endif // PYATOCHEK\_H

SpecialThread.h

#ifndef SPECIALTHREAD\_H

#define SPECIALTHREAD\_H

#include <QThread>

#include "Bee.h"

#include "WinnieThePooh.h"

#include "Pyatochek.h"

class SpecialThread : public QThread

{

public:

SpecialThread();

~SpecialThread();

void setTestClass(Bee\*);

void setWinnieThePooh(WinnieThePooh\*);

void setPyatochek(Pyatochek\*);

private:

Bee\* pTestObject;

WinnieThePooh\* pTestObjectW;

Pyatochek\* pTestObjectP;

public:

void run();

};

#endif // SPECIALTHREAD\_H

tcpserver.h

#ifndef TCPSERVER\_H

#define TCPSERVER\_H

#include <QTcpServer>

#include <QTcpSocket>

#include "mainwindow.h"

#include "ui\_mainwindow.h"

class **TcpServer** : public QObject

{

Q\_OBJECT

public:

explicit **TcpServer**(QObject \*parent = nullptr);

~***TcpServer***();

public slots:

void **onNewConnection**();

void **onReadyRead**();

void **onClientDisconnected**();

private:

QString **getClientKey**(const QTcpSocket \*client) const;

public:

QTcpServer \_server;

MainWindow\* mainWindow;

QFile\* file;

QHash<QString, QTcpSocket\*> \_clients;

int n;

int H;

};

#endif // TCPSERVER\_H

WinnieThePooh.h

#ifndef WINNIETHEPOOH\_H

#define WINNIETHEPOOH\_H

#include "mainwindow.h"

#include "ui\_mainwindow.h"

class WinnieThePooh : public QObject

{

Q\_OBJECT

public:

WinnieThePooh(bool&, bool&, int&, int, MainWindow&, QMutex&, QFile&);

~WinnieThePooh();

private:

int \*barrel\_fullness;

int H;

QMutex\* mutex;

MainWindow\* mainWindow;

QFile\* file;

QTextStream\* textStream;

bool \*WinnieThePooh\_sleep;

bool \*WinnieThePooh\_sleep\_got\_enough\_sleep;

public slots:

void run();

signals:

void finished();

};

#endif // WINNIETHEPOOH\_H

Bee.cpp

#include "Bee.h"

Bee::Bee(){}

Bee::Bee(bool &ws, bool &ps, bool &wsges, bool &psges, int &fullness, int h, int i, MainWindow &mw, QMutex& m, QFile& f)

{

WinnieThePooh\_sleep\_got\_enough\_sleep = &wsges;

Pyatochek\_sleep\_got\_enough\_sleep = &psges;

barrel\_fullness = &fullness;

WinnieThePooh\_sleep = &ws;

Pyatochek\_sleep = &ps;

mainWindow = &mw;

mutex = &m;

index = i;

H = h;

file = &f;

}

Bee::~Bee()

{

delete WinnieThePooh\_sleep\_got\_enough\_sleep;

delete Pyatochek\_sleep\_got\_enough\_sleep;

delete WinnieThePooh\_sleep;

delete Pyatochek\_sleep;

delete barrel\_fullness;

delete mainWindow;

delete mutex;

delete file;

}

void Bee::run() {

while (1)

{

Sleep(1000);

mutex->lock();

if(\*barrel\_fullness < H)

{

++\*barrel\_fullness;

qDebug() << "Пчела " + QString::number(index) + " принесла каплю";

mainWindow->ui->listWidget->addItem("Пчела " + QString::number(index) + " принесла каплю");

if (file->open(QIODevice::Append)) {file->write("Пчела " + QByteArray::number(index) + " принесла каплю\n");file->close();}

if (\*barrel\_fullness == H)

{

while (!\*WinnieThePooh\_sleep\_got\_enough\_sleep && !\*Pyatochek\_sleep\_got\_enough\_sleep) Sleep(1000);

if (\*WinnieThePooh\_sleep\_got\_enough\_sleep && \*Pyatochek\_sleep\_got\_enough\_sleep)

if (rand() & 1)

{

qDebug() << "Пчела " + QString::number(index) + " разбудила Пятачка";

mainWindow->ui->listWidget->addItem("Пчела " + QString::number(index) + " разбудила Пятачка");

if (file->open(QIODevice::Append)){ file->write("Пчела " + QByteArray::number(index) + " разбудила Пятачка\n");file->close();}

\*Pyatochek\_sleep = false;

\*Pyatochek\_sleep\_got\_enough\_sleep = false;

}

else

{

qDebug() << "Пчела " + QString::number(index) + " разбудила Винни-Пуxа";

mainWindow->ui->listWidget->addItem("Пчела " + QString::number(index) + " разбудила Винни-Пуxа");

if (file->open(QIODevice::Append)) {file->write("Пчела " + QByteArray::number(index) + " разбудила Винни-Пуxа\n");file->close();}

\*WinnieThePooh\_sleep = false;

\*WinnieThePooh\_sleep\_got\_enough\_sleep = false;

}

else if (\*WinnieThePooh\_sleep\_got\_enough\_sleep)

{

qDebug() << "Пчела " + QString::number(index) + " разбудила Винни-Пуxа";

mainWindow->ui->listWidget->addItem("Пчела " + QString::number(index) + " разбудила Винни-Пуxа");

if (file->open(QIODevice::Append)) {file->write("Пчела " + QByteArray::number(index) + " разбудила Винни-Пуxа\n");file->close();}

\*WinnieThePooh\_sleep = false;

\*WinnieThePooh\_sleep\_got\_enough\_sleep = false;

}

else

{

qDebug() << "Пчела " + QString::number(index) + " разбудила Пятачка";

mainWindow->ui->listWidget->addItem("Пчела " + QString::number(index) + " разбудила Пятачка");

if (file->open(QIODevice::Append)) {file->write("Пчела " + QByteArray::number(index) + " разбудила Пятачка\n");file->close();}

\*Pyatochek\_sleep = false;

\*Pyatochek\_sleep\_got\_enough\_sleep = false;

}

}

}

mutex->unlock();

}

emit finished();

}

main.cpp

#include "SpecialThread.h"

#include "tcpserver.h"

#include <QFile>

int main(int argc, char \*argv[])

{

QApplication qApplication2(argc, argv);

MainWindow mainWindow2;

QFile file("log.txt");

if (file.open(QIODevice::Append)) file.write("\n/////////////////////////////////////////////////////\n\nОжидание клиента...\n");

file.close();

TcpServer server;

server.mainWindow = &mainWindow2;

server.file = &file;

mainWindow2.setWindowIcon(QIcon(":/resources/img/icon.png"));

mainWindow2.ui->label->setPixmap(QPixmap(":resources/img/Pyatochek.png"));

mainWindow2.ui->label\_2->setPixmap(QPixmap(":resources/img/WinnieThePooh.png"));

mainWindow2.setWindowTitle("Задача о Винни-Пухе и Пятачке");

mainWindow2.setWindowFlags(Qt::MSWindowsFixedSizeDialogHint);

mainWindow2.file = &file;

mainWindow2.show();

mainWindow2.ui->listWidget->addItem("Ожидание клиента...");

QMutex mutex;

QMutex mutex2;

qApplication2.exec();

if(server.mainWindow->flag)

{

QApplication qApplication(argc, argv);

MainWindow mainWindow;

mainWindow.flag = true;

mainWindow.ui->pushButton->setText("Стоп");

mainWindow.ui->label->setPixmap(QPixmap(":resources/img/Pyatochek.png"));

mainWindow.ui->label\_2->setPixmap(QPixmap(":resources/img/WinnieThePooh.png"));

mainWindow.setWindowIcon(QIcon(":/resources/img/icon.png"));

mainWindow.setWindowTitle("Задача о Винни-Пухе и Пятачке");

mainWindow.setWindowFlags(Qt::MSWindowsFixedSizeDialogHint);

mainWindow.show();

srand(time(0));

bool WinnieThePooh\_sleep = true, WinnieThePooh\_sleep\_got\_enough\_sleep = false;

bool Pyatochek\_sleep = true, Pyatochek\_sleep\_got\_enough\_sleep = false;

int barrel\_fullness = 0;

const int n = server.n, H = server.H;

SpecialThread\* bee = new SpecialThread[n];

SpecialThread winniethePooh;

SpecialThread pyatochek;

Bee\*\* thread1 = new Bee\*[n];

for(int i = 0; i - n; ++i) thread1[i] = new Bee(WinnieThePooh\_sleep, Pyatochek\_sleep, WinnieThePooh\_sleep\_got\_enough\_sleep,

Pyatochek\_sleep\_got\_enough\_sleep, barrel\_fullness, H, i + 1, mainWindow, mutex2, file);

WinnieThePooh thread2(WinnieThePooh\_sleep, WinnieThePooh\_sleep\_got\_enough\_sleep, barrel\_fullness, H, mainWindow, mutex, file);

Pyatochek thread3(Pyatochek\_sleep, Pyatochek\_sleep\_got\_enough\_sleep, barrel\_fullness, H, mainWindow, mutex, file);

for(int i = 0; i - n; ++i) bee[i].setTestClass(thread1[i]);

winniethePooh.setWinnieThePooh(&thread2);

pyatochek.setPyatochek(&thread3);

for(int i = 0; i - n; ++i) bee[i].start();

winniethePooh.start();

pyatochek.start();

qApplication.exec();

delete[] bee;

for(int i = 0; i - n; delete thread1[i++]);

file.close();

}

}

mainwindow.cpp

#include "SpecialThread.h"

#include "tcpserver.h"

#include <QFile>

int main(int argc, char \*argv[])

{

QApplication qApplication2(*argc*, *argv*);

MainWindow mainWindow2;

QFile file("log.txt");

if (file.*open*(QIODevice::Append)) file.write("\n/////////////////////////////////////////////////////\n\nОжидание клиента...\n");

file.*close*();

TcpServer server;

server.mainWindow = &mainWindow2;

server.file = &file;

mainWindow2.setWindowIcon(QIcon(":/resources/img/icon.png"));

mainWindow2.ui->label->setPixmap(QPixmap(":resources/img/Pyatochek.png"));

mainWindow2.ui->label\_2->setPixmap(QPixmap(":resources/img/WinnieThePooh.png"));

mainWindow2.setWindowTitle("Задача о Винни-Пухе и Пятачке");

mainWindow2.setWindowFlags(Qt::MSWindowsFixedSizeDialogHint);

mainWindow2.file = &file;

mainWindow2.show();

mainWindow2.ui->listWidget->addItem("Ожидание клиента...");

QMutex mutex;

QMutex mutex2;

qApplication2.exec();

if(server.mainWindow->flag)

{

QApplication qApplication(*argc*, *argv*);

MainWindow mainWindow;

mainWindow.flag = true;

mainWindow.ui->pushButton->setText("Стоп");

mainWindow.ui->label->setPixmap(QPixmap(":resources/img/Pyatochek.png"));

mainWindow.ui->label\_2->setPixmap(QPixmap(":resources/img/WinnieThePooh.png"));

mainWindow.setWindowIcon(QIcon(":/resources/img/icon.png"));

mainWindow.setWindowTitle("Задача о Винни-Пухе и Пятачке");

mainWindow.setWindowFlags(Qt::MSWindowsFixedSizeDialogHint);

mainWindow.show();

srand(time(0));

bool WinnieThePooh\_sleep = true, WinnieThePooh\_sleep\_got\_enough\_sleep = false;

bool Pyatochek\_sleep = true, Pyatochek\_sleep\_got\_enough\_sleep = false;

int barrel\_fullness = 0;

const int n = server.n, H = server.H;

SpecialThread\* bee = new SpecialThread[n];

SpecialThread winniethePooh;

SpecialThread pyatochek;

Bee\*\* thread1 = new Bee\*[n];

for(int i = 0; i - n; ++i) thread1[i] = new Bee(*WinnieThePooh\_sleep*, *Pyatochek\_sleep*, *WinnieThePooh\_sleep\_got\_enough\_sleep*,

*Pyatochek\_sleep\_got\_enough\_sleep*, *barrel\_fullness*, H, i + 1, *mainWindow*, *mutex2*, *file*);

WinnieThePooh thread2(*WinnieThePooh\_sleep*, *WinnieThePooh\_sleep\_got\_enough\_sleep*, *barrel\_fullness*, H, *mainWindow*, *mutex*, *file*);

Pyatochek thread3(*Pyatochek\_sleep*, *Pyatochek\_sleep\_got\_enough\_sleep*, *barrel\_fullness*, H, *mainWindow*, *mutex*, *file*);

for(int i = 0; i - n; ++i) bee[i].setTestClass(*thread1[i]*);

winniethePooh.setWinnieThePooh(*&thread2*);

pyatochek.setPyatochek(*&thread3*);

for(int i = 0; i - n; ++i) bee[i].start();

winniethePooh.start();

pyatochek.start();

qApplication.exec();

delete[] bee;

for(int i = 0; i - n; delete thread1[i++]);

file.*close*();

}

}

Pyatochek.cpp

#include "Pyatochek.h"

Pyatochek::**Pyatochek**(bool &ps, bool &psges, int &fullness, int glotki, MainWindow &mw, QMutex& m, QFile& f)

{

barrel\_fullness = &fullness;

H = glotki;

mainWindow = &mw;

Pyatochek\_sleep = &ps;

Pyatochek\_sleep\_got\_enough\_sleep = &psges;

file = &f;

mutex = &m;

}

Pyatochek::~***Pyatochek***()

{

delete barrel\_fullness;

delete mainWindow;

delete Pyatochek\_sleep;

delete Pyatochek\_sleep\_got\_enough\_sleep;

delete mutex;

delete file;

}

void Pyatochek::**run**() {

while (1)

{

if(!\*Pyatochek\_sleep)

{

Sleep(1000);

qDebug() <<"Пятачёк съел 1/4 горшка";

mainWindow->ui->listWidget->addItem("Пятачёк съел 1/4 горшка");

if (file->*open*(QIODevice::Append)) {

mutex->lock();

file->write("Пятачёк съел 1/4 горшка\n");

file->*close*();

mutex->unlock();

}

\*barrel\_fullness = \*barrel\_fullness \* 0.75;

\*Pyatochek\_sleep = true;

\*Pyatochek\_sleep\_got\_enough\_sleep = false;

}

if (!\*Pyatochek\_sleep\_got\_enough\_sleep)

{

qDebug() <<"Пятачёк лёг спать";

mainWindow->ui->listWidget->addItem("Пятачёк лёг спать");

if (file->*open*(QIODevice::Append)) {

mutex->lock();

file->write("Пятачёк лёг спать\n");

file->*close*();

mutex->unlock();

}

Sleep(3000);

qDebug() <<"Пятачёк выспался";

mainWindow->ui->listWidget->addItem("Пятачёк выспался");

if (file->*open*(QIODevice::Append)) {

mutex->lock();

file->write("Пятачёк выспался\n");

file->*close*();

mutex->unlock();

}

\*Pyatochek\_sleep\_got\_enough\_sleep = true;

}

Sleep(1000);

}

emit finished();

}

SpecialThread.cpp

#include "SpecialThread.h"

SpecialThread::**SpecialThread**()

{

pTestObject = 0;

pTestObjectW = 0;

pTestObjectP = 0;

}

SpecialThread::~***SpecialThread***()

{

delete pTestObject;

delete pTestObjectW;

delete pTestObjectP;

}

void SpecialThread::**setTestClass**(Bee \*pTO) {

pTestObject = pTO;

}

void SpecialThread::**setWinnieThePooh**(WinnieThePooh \*pTO) {

pTestObjectW = pTO;

}

void SpecialThread::**setPyatochek**(Pyatochek \*pTO) {

pTestObjectP = pTO;

}

void SpecialThread::***run***() {

if (pTestObject) {

pTestObject->run();

}

if (pTestObjectW) {

pTestObjectW->run();

}

if (pTestObjectP) {

pTestObjectP->run();

}

}

tcpserver.cpp

#include "tcpserver.h"

TcpServer::TcpServer(QObject \*parent) : QObject(parent)

{

connect(&\_server, &QTcpServer::newConnection, this, &TcpServer::onNewConnection);

if(\_server.listen(QHostAddress::Any, 55555)) qDebug() << "Ожидание клиента...";

}

TcpServer::~TcpServer()

{

delete mainWindow;

delete file;

}

void TcpServer::onNewConnection()

{

const auto client = \_server.nextPendingConnection();

if(client == nullptr) return;

qDebug() << "Клиент подключился!";

mainWindow->ui->listWidget->addItem("Клиент подключился!");

if (file->open(QIODevice::Append)) file->write("Клиент подключился!\n");

file->close();

\_clients.insert(this->getClientKey(client), client);

connect(client, &QTcpSocket::readyRead, this, &TcpServer::onReadyRead);

connect(client, &QTcpSocket::disconnected, this, &TcpServer::onClientDisconnected);

}

void TcpServer::onReadyRead()

{

const auto client = qobject\_cast<QTcpSocket\*>(sender());

if(!client) return;

QString message = client->readAll();

QString nstr = "", Hstr = "";

int i = 0;

bool flag = true;

for(; message[i] != " "; nstr += message[i], ++i)

if(message[i] < "0" || message[i] > "9") flag = false;

for(++i; i - message.size(); Hstr += message[i], ++i)

if(message[i] < "0" || message[i] > "9") flag = false;

if(flag){

n = nstr.toUInt();

H = Hstr.toUInt();

qDebug() << "Данные получены от клиента";

qDebug() << "Количество пчёл: " + nstr;

qDebug() << "Количество глотков горшке: " + Hstr;

mainWindow->ui->listWidget->addItem("Данные получены от клиента");

mainWindow->ui->listWidget->addItem("Количество пчёл: " + nstr);

mainWindow->ui->listWidget->addItem("Количество глотков горшке: " + Hstr);

if (file->open(QIODevice::Append)) {

file->write("Данные получены от клиента\n");

file->write(("Количество пчёл: " + nstr + "\n").toUtf8());

file->write(("Количество глотков горшке: " + Hstr + "\n").toUtf8());

file->close();

}

mainWindow->flag = true;

}

else {

qDebug() << "Данные клиента введены некорректно";

mainWindow->ui->listWidget->addItem("Данные клиента введены некорректно");

if (file->open(QIODevice::Append)) file->write("Данные клиента введены некорректно\n");

file->close();

}

}

void TcpServer::onClientDisconnected()

{

const auto client = qobject\_cast<QTcpSocket\*>(sender());

if(!client) return;

\_clients.remove(this->getClientKey(client));

qDebug() << "Клиент отключился";

mainWindow->ui->listWidget->addItem("Клиент отключился");

if (file->open(QIODevice::Append)) file->write("Клиент отключился\n");

file->close();

}

QString TcpServer::getClientKey(const QTcpSocket \*client) const

{

return client->peerAddress().toString().append(":").append(QString::number(client->peerPort()));

}

WinnieThePooh.cpp

#include "tcpserver.h"

TcpServer::TcpServer(QObject \*parent) : QObject(parent)

{

connect(&\_server, &QTcpServer::newConnection, this, &TcpServer::onNewConnection);

if(\_server.listen(QHostAddress::Any, 55555)) qDebug() << "Ожидание клиента...";

}

TcpServer::~TcpServer()

{

delete mainWindow;

delete file;

}

void TcpServer::onNewConnection()

{

const auto client = \_server.nextPendingConnection();

if(client == nullptr) return;

qDebug() << "Клиент подключился!";

mainWindow->ui->listWidget->addItem("Клиент подключился!");

if (file->open(QIODevice::Append)) file->write("Клиент подключился!\n");

file->close();

\_clients.insert(this->getClientKey(client), client);

connect(client, &QTcpSocket::readyRead, this, &TcpServer::onReadyRead);

connect(client, &QTcpSocket::disconnected, this, &TcpServer::onClientDisconnected);

}

void TcpServer::onReadyRead()

{

const auto client = qobject\_cast<QTcpSocket\*>(sender());

if(!client) return;

QString message = client->readAll();

QString nstr = "", Hstr = "";

int i = 0;

bool flag = true;

for(; message[i] != " "; nstr += message[i], ++i)

if(message[i] < "0" || message[i] > "9") flag = false;

for(++i; i - message.size(); Hstr += message[i], ++i)

if(message[i] < "0" || message[i] > "9") flag = false;

if(flag){

n = nstr.toUInt();

H = Hstr.toUInt();

qDebug() << "Данные получены от клиента";

qDebug() << "Количество пчёл: " + nstr;

qDebug() << "Количество глотков горшке: " + Hstr;

mainWindow->ui->listWidget->addItem("Данные получены от клиента");

mainWindow->ui->listWidget->addItem("Количество пчёл: " + nstr);

mainWindow->ui->listWidget->addItem("Количество глотков горшке: " + Hstr);

if (file->open(QIODevice::Append)) {

file->write("Данные получены от клиента\n");

file->write(("Количество пчёл: " + nstr + "\n").toUtf8());

file->write(("Количество глотков горшке: " + Hstr + "\n").toUtf8());

file->close();

}

mainWindow->flag = true;

}

else {

qDebug() << "Данные клиента введены некорректно";

mainWindow->ui->listWidget->addItem("Данные клиента введены некорректно");

if (file->open(QIODevice::Append)) file->write("Данные клиента введены некорректно\n");

file->close();

}

}

void TcpServer::onClientDisconnected()

{

const auto client = qobject\_cast<QTcpSocket\*>(sender());

if(!client) return;

\_clients.remove(this->getClientKey(client));

qDebug() << "Клиент отключился";

mainWindow->ui->listWidget->addItem("Клиент отключился");

if (file->open(QIODevice::Append)) file->write("Клиент отключился\n");

file->close();

}

QString TcpServer::getClientKey(const QTcpSocket \*client) const

{

return client->peerAddress().toString().append(":").append(QString::number(client->peerPort()));

}

Клиент

mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QTcpSocket>

#include <QMainWindow>

#include <QFile>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

private:

Ui::MainWindow \*ui;

public:

QTcpSocket\* \_socket;

QByteArray ba;

QFile\* file;

};

#endif // MAINWINDOW\_H

tcpclient.h

#ifndef TCPCLIENT\_H

#define TCPCLIENT\_H

#include <QObject>

#include <QTcpSocket>

#include <QFile>

class **TcpClient** : public QObject

{

Q\_OBJECT

public:

explicit **TcpClient**(QObject \*parent = nullptr);

~***TcpClient***();

public slots:

void **connectToServer**(const QString &ip, const QString &port);

private slots:

void **onConnected**();

void **onErrorOccurred**(QAbstractSocket::SocketError error);

public:

QTcpSocket \_socket;

QFile\* file;

};

#endif // TCPCLIENT\_H

main.cpp

#include "mainwindow.h"

#include "tcpclient.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(*argc*, *argv*);

MainWindow w;

TcpClient client;

QFile file("log.txt");

client.file = &file;

w.file = &file;

w.\_socket = &client.\_socket;

client.connectToServer("127.1.2.3", "55555");

w.setWindowTitle("Задача о Винни-Пухе и Пятачке");

w.setWindowIcon(QIcon(":/resources/img/icon.png"));

w.setWindowFlags(Qt::MSWindowsFixedSizeDialogHint);

w.show();

return a.exec();

}

mainwindow.cpp

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

delete \_socket;

delete file;

}

void MainWindow::on\_pushButton\_clicked()

{

if(0 < ui->textEdit->document()->toPlainText().toInt() && 0 < ui->textEdit\_2->document()->toPlainText().toInt()){

qDebug() << "Данные введены корректно";

qDebug() << "Количество пчёл: " << ui->textEdit->document()->toPlainText();

qDebug() << "Количество глотков горшке: " << ui->textEdit\_2->document()->toPlainText();

if (file->open(QIODevice::Append)) {

file->write("Данные введены корректно\n");

file->write(("Количество пчёл: " + ui->textEdit->document()->toPlainText() + "\n").toUtf8());

file->write(("Количество глотков горшке: " + ui->textEdit\_2->document()->toPlainText() + "\n").toUtf8());

file->close();

}

}

else {

qDebug() << "Данные введены некорректно";

if (file->open(QIODevice::Append)) file->write("Данные введены некорректно\n");

file->close();

}

ba = (ui->textEdit->document()->toPlainText() + " " + ui->textEdit\_2->document()->toPlainText()).toUtf8();

\_socket->write(ba);

}

tcpclient.cpp

#include "tcpclient.h"

TcpClient::**TcpClient**(QObject \*parent) : QObject(*parent*)

{

connect(&\_socket, &QTcpSocket::connected, this, &TcpClient::onConnected);

connect(&\_socket, &QTcpSocket::errorOccurred, this, &TcpClient::onErrorOccurred);

}

TcpClient::~***TcpClient***(){

delete file;

}

void TcpClient::**connectToServer**(const QString &ip, const QString &port)

{

\_socket.*connectToHost*(ip, port.toUInt());

}

void TcpClient::**onConnected**()

{

qInfo() << "Подключение к серверу произошло успешно!";

if (file->*open*(QIODevice::Append)) file->write("\n/////////////////////////////////////////////////////\n\nПодключение к серверу произошло успешно!\n");

file->*close*();

}

void TcpClient::**onErrorOccurred**(QAbstractSocket::SocketError error)

{

qWarning() << "Error: " << error;

if (file->*open*(QIODevice::Append)) file->write(("\n/////////////////////////////////////////////////////\n\nError: Ошибка подключения к серверу\n"));

file->*close*();

}