МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Брестский Государственный Технический Университет
Кафедра интеллектуальных информационных технологий

Лабораторная работа №5
по дисциплине «ОсиСП» за 5 семестр
на тему «Возможности, предлагаемыми фреймворком Qt, для разработки многопоточных приложений»

Выполнила: студентка
2 курса
Факультета ЭИС
Андросюк Мария
Михайловна
Проверила:
Дряпко А.В.

Брест, 2021

Ход работы:

myserver.h

```
#ifndef MYSERVER_H
#define MYSERVER_H

#include <QTcpServer>
#include <QTcpSocket>
#include <QDir>
#include <QFile>

class MyServer: public QTcpServer{
    Q_OBJECT
public:
    MyServer();
    ~MyServer();
    QTcpSocket * socket;
    QByteArray Data;
    QString path_to_Download;

public slots:
    void StartServer();
    void incomingConnection(int socketDeskriptor);
    void sockReady();
    void sockDisc();
};


#endif // MYSERVER_H
```

myserver.cpp

```
#include "myserver.h"

MyServer::MyServer(){}
MyServer::~MyServer(){}

void MyServer::StartServer(){
    if(this->listen(QHostAddress::Any, 1919)) {
       qDebug() << "Listening";
    }
    else {
        qDebug() << "Not Listening";
    }
}

void MyServer::incomingConnection(int socketDeskriptor) {
    socket = new QTcpSocket(this);
    socket->setSocketDescriptor(socketDeskriptor);
    connect(socket,SIGNAL(readyRead()), this, SLOT(sockReady()));
    connect(socket,SIGNAL(disconnected()),this,SLOT(sockDisc()));
    qDebug()<<socketDeskriptor<<"Client connected";
    qDebug()<<"Send client connect status - YES";
}

void MyServer::sockReady() {
    Data = socket->readAll();
    qDebug() << "Select from Client" << Data;


    if(!Data.isEmpty()) {

        QDir server_version(QDir::currentPath());
        QStringList filter;
```

```
        filter << "*.json";
        foreach(QFileInfo info, server_version.entryInfoList(filter)) {
            filter.clear();
            filter << info.absoluteFilePath();
        }
        qDebug() << filter;

        QFile file(filter.back());
        if (!file.open(QIODevice::ReadOnly))
            return;
        path_to_Download = file.readAll();
        if(QString(Data) == path_to_Download) {
            qDebug() << "Send to Client" << "Actuale";
            socket->write("Actuale");
        }

        else if(Data == "Yes") {
            Data.clear();
            QDir dir_server(QDir::currentPath());

            Data.append(dir_server.path()+' ' + "Colors.txt " +
path_to_Download);

            qDebug() << "Send to Client" << Data;
            socket->write(Data);
        }

        else {
            qDebug() << "Send to Client" << "Need update";
            socket->write("Need update");
        }
        socket->waitForBytesWritten(100);
    }
}

void MyServer::sockDisc(){
    qDebug()<<"Disconnect";
    socket->deleteLater();
}
```

main.cpp

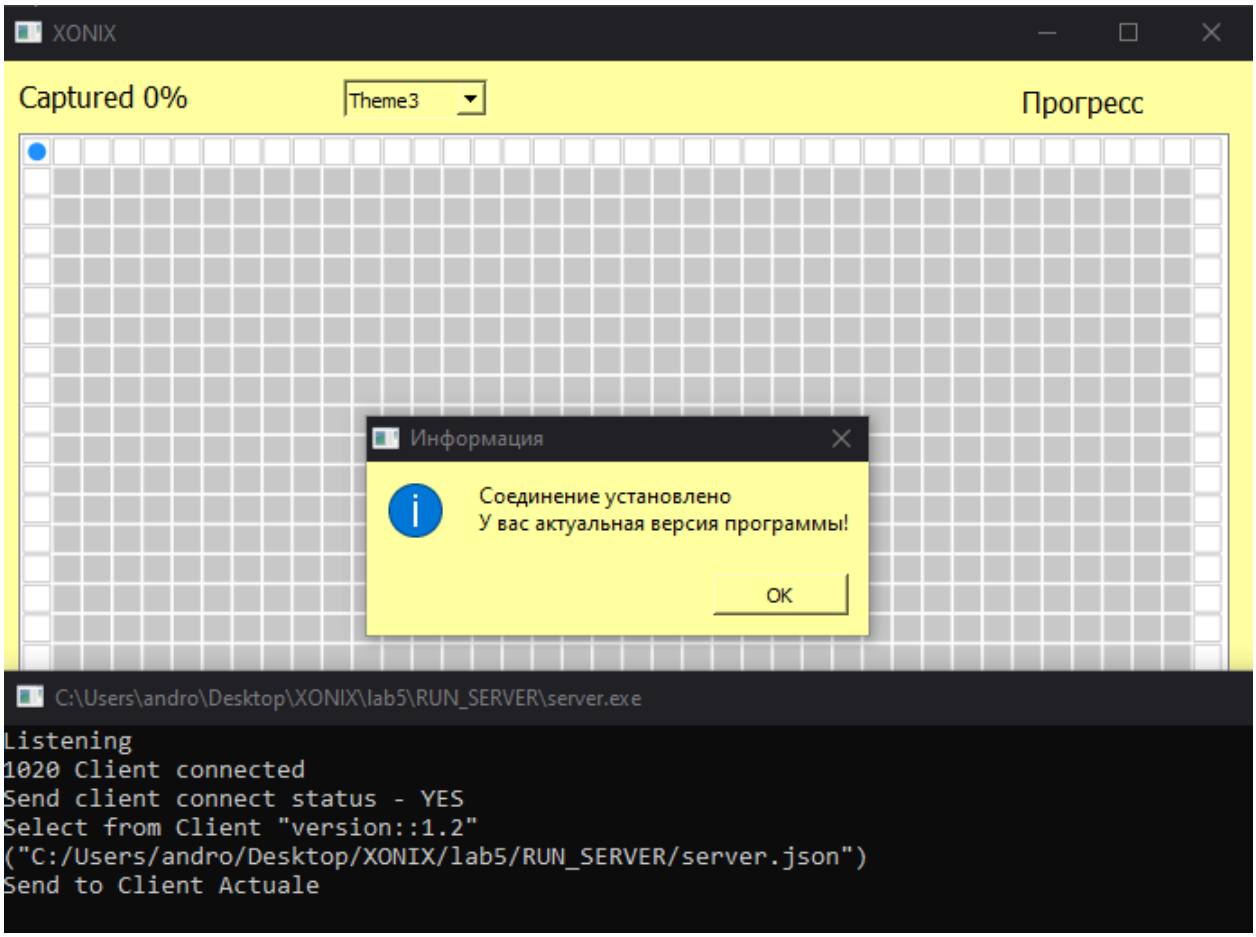
```
#include <QCoreApplication>
#include <myserver.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    MyServer Server;
    Server.StartServer();

    return a.exec();
}
```

Результат работы программы:



Вывод: изучила озможности, предлагаемыми фреймворком Qt, для разработки многопоточных приложений. В результате работы было добавлено предупреждение пользователя о наличии новой версии программы и разрешение на установку, а также дополнительная тема.