Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский университет
«Высшая школа экономики»

Кафедра «Компьютерная безопасность»

ОТЧЕТ
К ЛАБОРАТОРНОЙ РАБОТЕ №24

по дисциплине

«Языки программирования»

Работу выполнил
студент группы СКБ-222 _____ Д.А. Спиридонов
подпись, дата

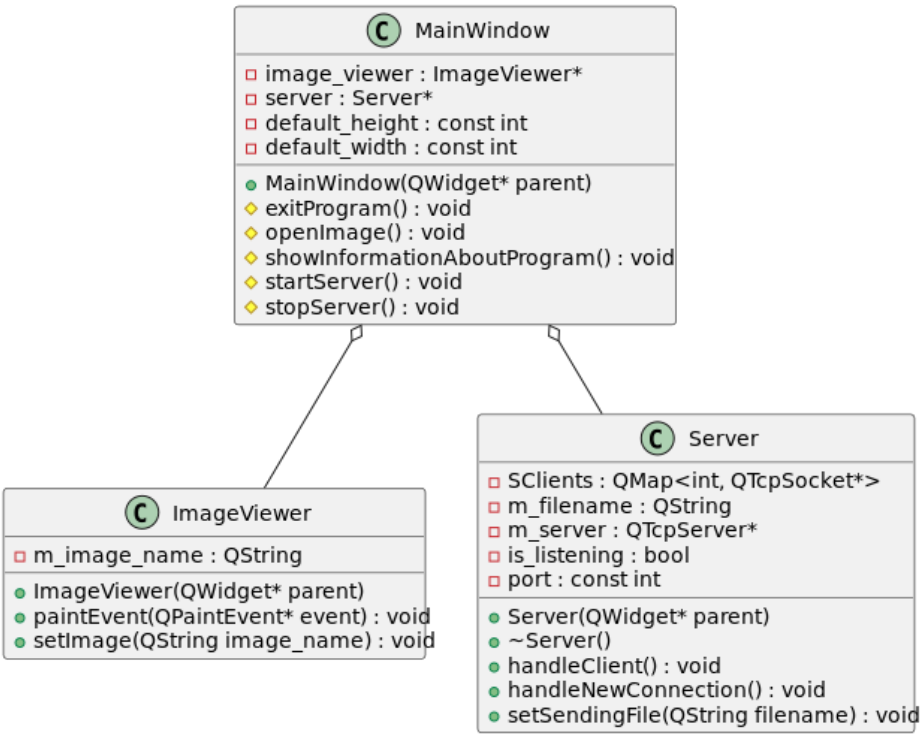
Работу проверил _____ С.А. Булгаков
подпись, дата

Москва 2023

Содержание

Приложение А

А.1 UML-диаграммы классов MainWindow, Server и ImageViewer



Приложение B

B.1 Исходный код ImageViewer.h

```
#ifndef __IMAGE_VIEWER_H__
#define __IMAGE_VIEWER_H__

#include <QWidget>
#include <QImage>
#include <QPainter>
#include <QtGlobal>
#include <QString>

class ImageViewer : public QWidget
{
public:
    ImageViewer(QWidget* parent=nullptr);
    void setImage(QString image_name);
    void paintEvent(QPaintEvent* event=nullptr);
private:
    QString m_image_name;
};

#endif // __IMAGE_VIEWER_H__
```

B.2 Исходный код ImageViewer.cpp

```
#include "ImageViewer.h"


ImageViewer::ImageViewer(QWidget* parent)
    : QWidget(parent), m_image_name("")
{
    this->show();
}


void ImageViewer::setImage(QString image_name)
{
    m_image_name = image_name;
    this->update();
}


void ImageViewer::paintEvent(QPaintEvent* event)
{
    if(!m_image_name.isEmpty())
    {
        QPainter painter(this);
        painter.drawImage(
            QRectF(
                (qreal) (this->width()) / 2 - (qreal) (this->width()) / 3,
                0,
                (qreal) (this->width()  * 2) / 3,
                (qreal) (this->height() * 2) / 3
            ),
            QImage(m_image_name)
        );
    }
}
```

B.3 Исходный код MainWindow.h

```
#ifndef __MAIN_WINDOW_H__
#define __MAIN_WINDOW_H__

#include <QMainWindow>
#include <QWidget>
#include <QVBoxLayout>
#include <QFileDialog>
#include <QMenuBar>
#include <QMessageBox>

#include "ImageViewer.h"
#include "Server.h"


class MainWindow : public QMainWindow
{
    Q_OBJECT
public:
    MainWindow(QWidget* parent=nullptr);
protected:
    // Actions
    void startServer();
    void stopServer();
    void exitProgram();
    void showInformationAboutProgram();
    void openImage();
private:
    const int default_height = 480;
    const int default_width  = 640;

    ImageViewer* image_viewer;
    Server* server = nullptr;
};

#endif // __MAIN_WINDOW_H__
```

B.4 Исходный код MainWindow.cpp

```
#include "MainWindow.h"


MainWindow::MainWindow(QWidget* parent)
    : QMainWindow(parent)
{
    QWidget *ui_area = new QWidget;
    this->setCentralWidget(ui_area);

    this->setWindowTitle(tr("ImageViewer"));

    QVBoxLayout* layout = new QVBoxLayout(this);
    image_viewer = new ImageViewer;
    layout->addWidget(image_viewer);

    QAction* startServerAction = new QAction(tr("Запустить сервер"));
    connect(startServerAction, &QAction::triggered, this, &MainWindow::startServer);

    QAction* stopServerAction = new QAction(tr("Остановить сервер"));
    connect(stopServerAction, &QAction::triggered, this, &MainWindow::stopServer);

    QAction* openImageAction = new QAction(tr("Открыть изображение"));
    connect(openImageAction, &QAction::triggered, this, &MainWindow::openImage);

    QAction* exitProgramAction = new QAction(tr("Выход"));
    connect(exitProgramAction, &QAction::triggered, this, &MainWindow::exitProgram);

    QAction* showInfoAboutProgramAction = new QAction(tr("О программе"));
    connect(showInfoAboutProgramAction, &QAction::triggered, this, &MainWindow::showInformationAboutProgram)

    auto fileMenu = menuBar()->addMenu(tr("Файлы"));
    fileMenu->addAction(startServerAction);
    fileMenu->addAction(stopServerAction);
    fileMenu->addAction(openImageAction);
    fileMenu->addAction(exitProgramAction);

    auto helpMenu = menuBar()->addMenu(tr("Помощь"));
    helpMenu->addAction(showInfoAboutProgramAction);

    ui_area->setLayout(layout);
    this->resize(default_width, default_height);
}


void MainWindow::startServer()
{
    server = new Server(this);
}


void MainWindow::stopServer()
{
    if(server) delete server;
    server = nullptr;
}


void MainWindow::exitProgram()
{
    this->close();
}
```

```
void MainWindow::showInformationAboutProgram()
{
    QMessageBox msg;
    msg.setText("Автор: Даниил Спиридонов\nВерсия библиотеки Qt: 5.15");
    msg.exec();
}


void MainWindow::openImage()
{
    QString filename = QFileDialog::getOpenFileName(this, tr("Open Image"), "./", tr("Image Files (*.jpg)"))
    image_viewer->setImage(filename);
    if(server) server->setSendingFile(filename);
}
```

B.5 Исходный код Server.h

```
#ifndef __SERVER_H__
#define __SERVER_H__

#include <QTcpServer>
#include <QTcpSocket>
#include <QWidget>
#include <QObject>
#include <QFile>

class Server : public QObject
{
    Q_OBJECT
public:
    Server(QWidget* parent=nullptr);
    ~Server();
    void setSendingFile(QString filename);
public slots:
    void handleNewConnection();
    void handleClient();
private:
    const int port = 31337;

    QTcpServer* m_server;
    bool is_listening;
    QMap<int,QTcpSocket*> SClients;

    QString m_filename = "";
};

#endif // __SERVER_H__
```

B.6 Исходный код Server.cpp

```
#include "Server.h"


Server::Server(QWidget* parent)
{
    m_server = new QTcpServer(parent);
    connect(m_server, &QTcpServer::newConnection, this, &Server::handleNewConnection);
    if(m_server->listen(QHostAddress::Any, port) || is_listening)
    {
        is_listening = true;
        qDebug() << "Server started!";
    }
    else
    {
        qDebug() << "Unable to start the server";
    }
}


Server::~Server()
{
    if(is_listening)
    {
        foreach(int i, SClients.keys())
        {
            SClients[i]->close();
            SClients.remove(i);
        }
        m_server->close();
        is_listening = false;
    }
    delete m_server;
}


void Server::setSendingFile(QString filename)
{
    m_filename = filename;
}


void Server::handleNewConnection()
{
    if(is_listening)
    {
        QTcpSocket* client = m_server->nextPendingConnection();
        int idusersocs = client->socketDescriptor();
        SClients[idusersocs] = client;
        connect(SClients[idusersocs], &QTcpSocket::readyRead, this, &Server::handleClient);
    }
}


void Server::handleClient()
{
    QTcpSocket* client = (QTcpSocket*)sender();
    int idusersocs = client->socketDescriptor();

    QTextStream os(client);
    os.setAutoDetectUnicode(true);
    if(m_filename.isEmpty())
    {
        os << "HTTP/1.0 200 Ok\r\n"
```

```
            << "Content-Type: text/html; charset=\"utf-8\"\r\n"
            << "\r\n"
            << "<h1>Nothing to see here</h1>";
    }
    else
    {
        client->write(
            "HTTP/1.1 200 OK\r\n"
            "Content-Type: image/jpg; charset=\"utf-8\"\r\n"
            "\r\n"
        );

        QFile file(m_filename);
        file.open(QIODevice::ReadOnly);
        QByteArray file_data=file.readAll();
        client->write(file_data);
    }
    os << "\r\n\r\n";
    client->close();
    SClients.remove(idusersocs);
}
```

B.7 Исходный код main.cpp

```
#include <QApplication>
#include "MainWindow.h"


int main(int argc, char** argv)
{
    QApplication app(argc, argv);
    QCoreApplication::setAttribute(Qt::AA_DontUseNativeMenuBar);
    MainWindow main_window;
    main_window.show();
    return app.exec();
}
```