Client for Attestation System

Создано системой Doxygen 1.13.2

Глава 1

Документация на клиентское приложение экзаменационной системы

Данный проект представляет собой клиентскую часть платформы для прохождения онлайн-экзаменов. Приложение разработано с использованием фреймворка Qt и взаимодействует с сервером по протоколу TCP. Интерфейс выполнен на основе виджетов Qt, обеспечивая удобную навигацию между экранами.

Клиент поддерживает следующие функции:

- регистрация и авторизация пользователя;
- прохождение экзаменов;
- просмотр статистики и результатов;
- редактирование профиля;
- смена пароля;
- взаимодействие с сервером в реальном времени через сокеты.

1.1 Основные компоненты:

- MainWindow — основное окно, управляющее всеми экранами (через QStackedWidget) и логикой переходов между ними.
- Окна пользовательского интерфейса:
  - LoginWindow — начальный экран: вход или регистрация;
  - AuthenticationWindow — ввод логина и пароля;
  - RegistrationWindow — создание нового пользователя;
  - WelcomeWindow — главное меню после входа;
  - ExamSelectionWindow — выбор доступного экзамена;
  - ExamWindow — прохождение экзамена (вопросы и варианты ответов);
  - ExamCompletionWindow — отображение результатов;
  - StatisticsWindow — просмотр и фильтрация статистики;
  - ProfileWindow — редактирование личных данных;
  - ChangePasswordWindow — смена пароля.

- Client — синглтон, реализующий TCP-соединение с сервером. Отвечает за:
  - регистрацию и авторизацию (REGISTER, LOGIN);
  - получение данных (GET_EXAMS, GET_QUESTIONS, GET_STATISTICS, GET_↩ PROFILE);
  - отправку (SAVE_RESULTS, UPDATE_PROFILE, CHANGE_PASSWORD);
  - приём JSON-ответов от сервера и генерацию сигналов для интерфейса.
- ExamQuestion — структура, описывающая один вопрос экзамена (текст, варианты, правильный ответ).

1.2 Архитектура

Приложение построено по принципам модульности и разделения ответственности:

- MainWindow управляет навигацией;
- Каждый UI-компонент отвечает за отдельную задачу (регистрация, экзамен, профиль и т.д.);
- Client обеспечивает взаимодействие с сервером и маршрутизацию данных к окнам.

Коммуникация между окнами осуществляется через Qt-сигналы и слоты, обеспечивая слабую связанность и расширяемость.

1.3 Сетевой протокол

Протокол обмена с сервером основан на отправке строковых команд, за которыми может следовать JSON:

- LOGIN <username> <password>
- REGISTER <username> <password>
- GET_EXAMS
- GET_QUESTIONS <exam_id>
- SAVE_RESULTS <json>
- GET_PROFILE
- UPDATE_PROFILE <json>
- CHANGE_PASSWORD <json>
- GET_STATISTICS

Сервер возвращает ответы либо в виде OK / ERROR, либо в формате JSON-массивов и объектов.

1.4 Документация

Данная документация сгенерирована с помощью Doxygen, и включает:

- подробное описание всех классов и методов;
- диаграммы связей классов и включаемых файлов;
- markdown-страницу в качестве основной;
- визуализацию архитектуры через Graphviz.

1.5 Сборка и запуск

- Язык: C++
- Фреймворк: Qt 5.12+
- Сборка: Qt Creator, qmake или cmake

Глава 2

Алфавитный указатель пространств имен

2.1 Пространства имен

Полный список пространств имен.

Глава 3

Иерархический список классов

3.1 Иерархия классов

Иерархия классов.

Глава 4

Алфавитный указатель структур данных

4.1 Структуры данных

Структуры данных с их кратким описанием.

Глава 5

Список файлов

5.1 Файлы

Полный список файлов.

Глава 6

Пространства имен

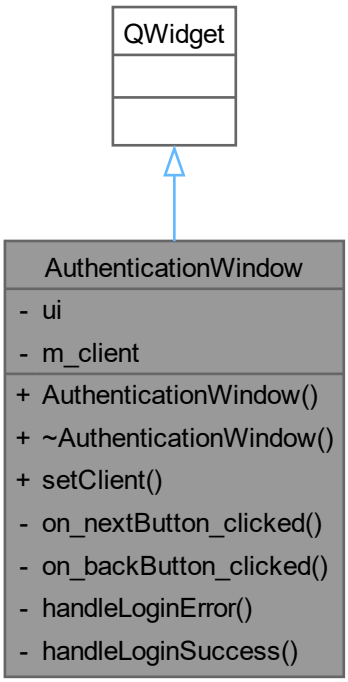
6.1 Пространство имен Ui

Глава 7

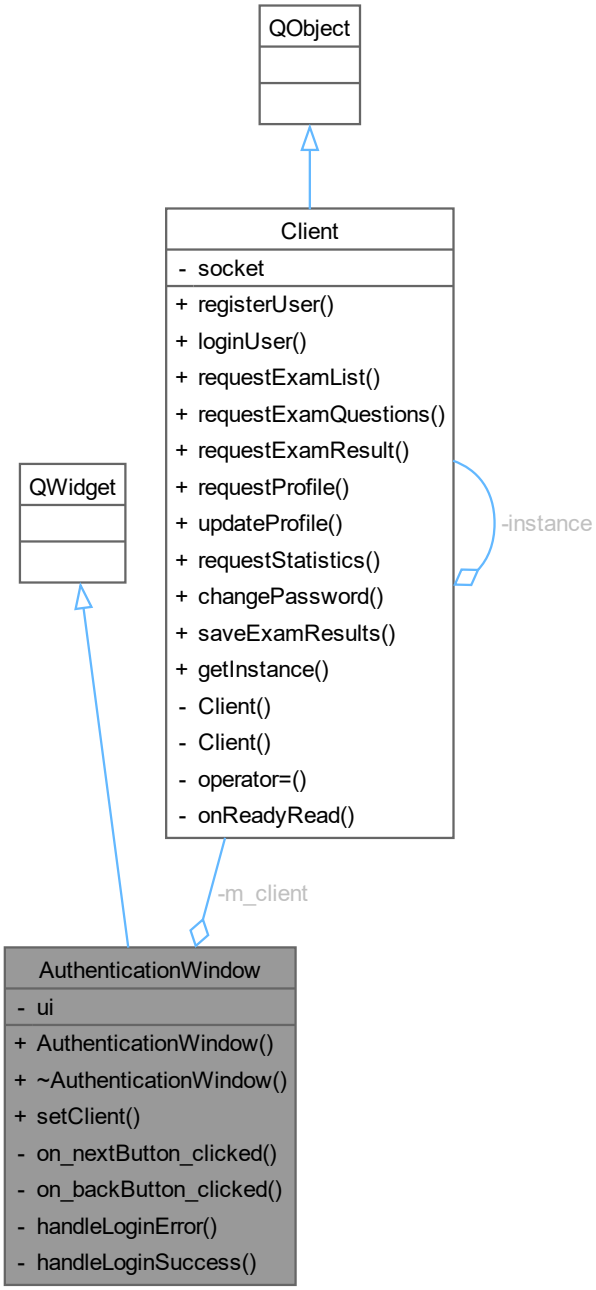
Структуры данных

7.1 Класс AuthenticationWindow

Класс AuthenticationWindow реализует окно аутентификации пользователя.
#include <AuthenticationWindow.h>
Граф наследования:AuthenticationWindow:



Граф связей класса AuthenticationWindow:



Сигналы

- void authenticationSucceeded ()

  Сигнал, испускаемый при успешной аутентификации.
- void backRequested ()

  Сигнал, испускаемый при нажатии кнопки "Назад".

Открытые члены

- AuthenticationWindow (QWidget *parent=nullptr)

  Конструктор класса AuthenticationWindow.
- ~AuthenticationWindow ()

  Деструктор класса AuthenticationWindow.
- void setClient (Client *client)

  Устанавливает указатель на объект клиента.

Закрытые слоты

- void on_nextButton_clicked ()

  Слот обработки нажатия кнопки "Войти".
- void on_backButton_clicked ()

  Слот обработки нажатия кнопки "Назад".
- void handleLoginError (const QString &errorMessage)

  Слот обработки ошибки входа.
- void handleLoginSuccess ()

  Слот обработки успешного входа.

Закрытые данные

- Ui::AuthenticationWindow * ui

  Указатель на объект интерфейса.
- Client * m_client

  Указатель на объект клиента для сетевого взаимодействия.

7.1.1 Подробное описание

Класс AuthenticationWindow реализует окно аутентификации пользователя.
Данный класс предоставляет интерфейс для ввода логина и пароля, а также механизмы отправки этих данных на сервер через объект клиента. Он обрабатывает сигналы об успешной или неуспешной попытке входа.

7.1.2 Конструктор(ы)

7.1.2.1 AuthenticationWindow()

AuthenticationWindow::AuthenticationWindow (
QWidget * parent = nullptr) [explicit]
Конструктор класса AuthenticationWindow.
Конструктор окна аутентификации.
Инициализирует пользовательский интерфейс.

Аргументы

parent	Родительский виджет, по умолчанию nullptr.

Инициализирует пользовательский интерфейс и устанавливает указатель клиента в nullptr.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

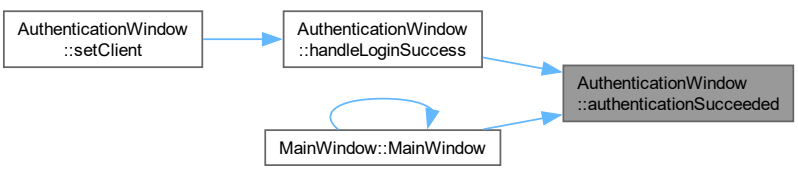
7.1.2.2 ~AuthenticationWindow()

AuthenticationWindow::~AuthenticationWindow ()
Деструктор класса AuthenticationWindow.
Деструктор окна аутентификации.
Освобождает ресурсы, связанные с UI.
Освобождает память, выделенную под интерфейс.

7.1.3 Методы

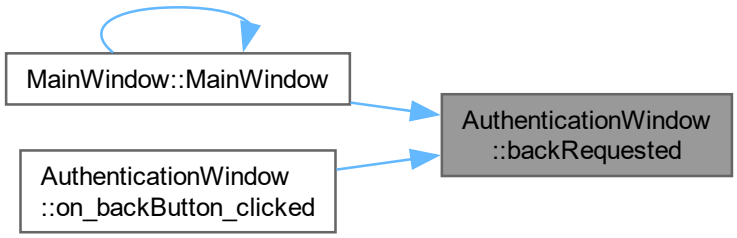
7.1.3.1 authenticationSucceeded

void AuthenticationWindow::authenticationSucceeded () [signal]
Сигнал, испускаемый при успешной аутентификации.
Используется для перехода к следующему окну после входа. Граф вызова функции:



7.1.3.2 backRequested

void AuthenticationWindow::backRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Назад".
Используется для возврата к предыдущему окну (например, окну входа). Граф вызова функции:



7.1.3.3 handleLoginError

void AuthenticationWindow::handleLoginError (
const QString & errorMessage) [private], [slot]
Слот обработки ошибки входа.
Обрабатывает неудачную попытку входа.
Отображает сообщение об ошибке пользователю.

Аргументы

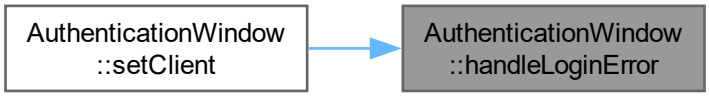
errorMessage	Текст ошибки от клиента.

Отображает сообщение об ошибке пользователю.

Аргументы

errorMessage	Текст ошибки, полученный от клиента.

Граф вызова функции:

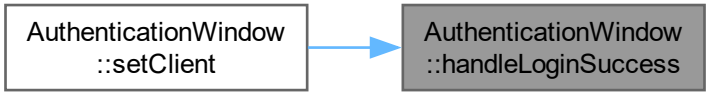


7.1.3.4 handleLoginSuccess

void AuthenticationWindow::handleLoginSuccess () [private], [slot]
Слот обработки успешного входа.

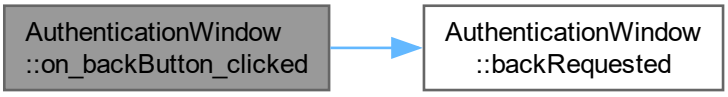
Обрабатывает успешную аутентификацию пользователя.
Испускает сигнал authenticationSucceeded().
Генерирует сигнал об успешной аутентификации, который обрабатывается внешним компонентом.
Граф вызовов:

Граф вызова функции:



7.1.3.5 on_backButton_clicked

void AuthenticationWindow::on_backButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Назад".
Слот для обработки нажатия кнопки "Назад".
Испускает сигнал backRequested().
Генерирует сигнал возврата к предыдущему окну. Граф вызовов:



7.1.3.6 on_nextButton_clicked

void AuthenticationWindow::on_nextButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Войти".
Слот для обработки нажатия кнопки "Далее".
Считывает введённые данные и передаёт их клиенту для аутентификации.
Отправляет имя пользователя и пароль в клиент для выполнения попытки входа в систему.

7.1.3.7 setClient()

void AuthenticationWindow::setClient (
Client * client)

Устанавливает указатель на объект клиента.
Устанавливает указатель на клиента и подключает слоты для обработки сигналов.
Подключает слоты обработки успешного входа и ошибки входа к сигналам клиента.

Аргументы

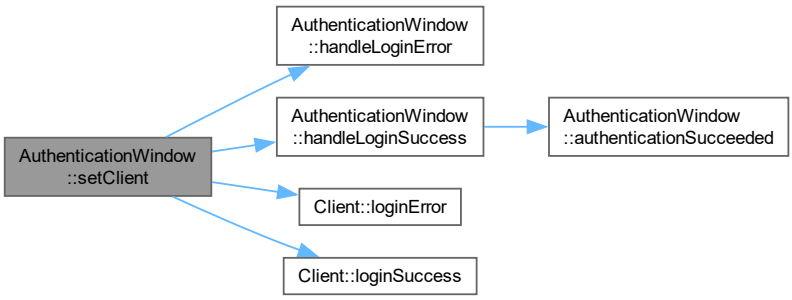
client	Указатель на объект класса Client.

Метод используется для инициализации взаимодействия между окном аутентификации и клиентским соединением.

Аргументы

client	Указатель на объект клиента.

Граф вызовов:



7.1.4 Поля

7.1.4.1 m_client

Client* AuthenticationWindow::m_client [private]
Указатель на объект клиента для сетевого взаимодействия.

7.1.4.2 ui

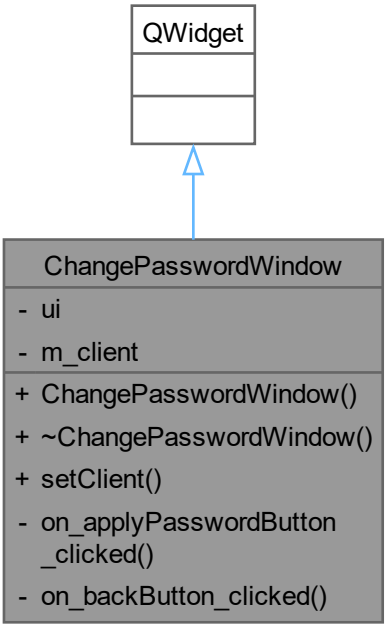
Ui::AuthenticationWindow* AuthenticationWindow::ui [private]
Указатель на объект интерфейса.
Объявления и описания членов классов находятся в файлах:

- ExamClient/AuthenticationWindow.h
- ExamClient/AuthenticationWindow.cpp

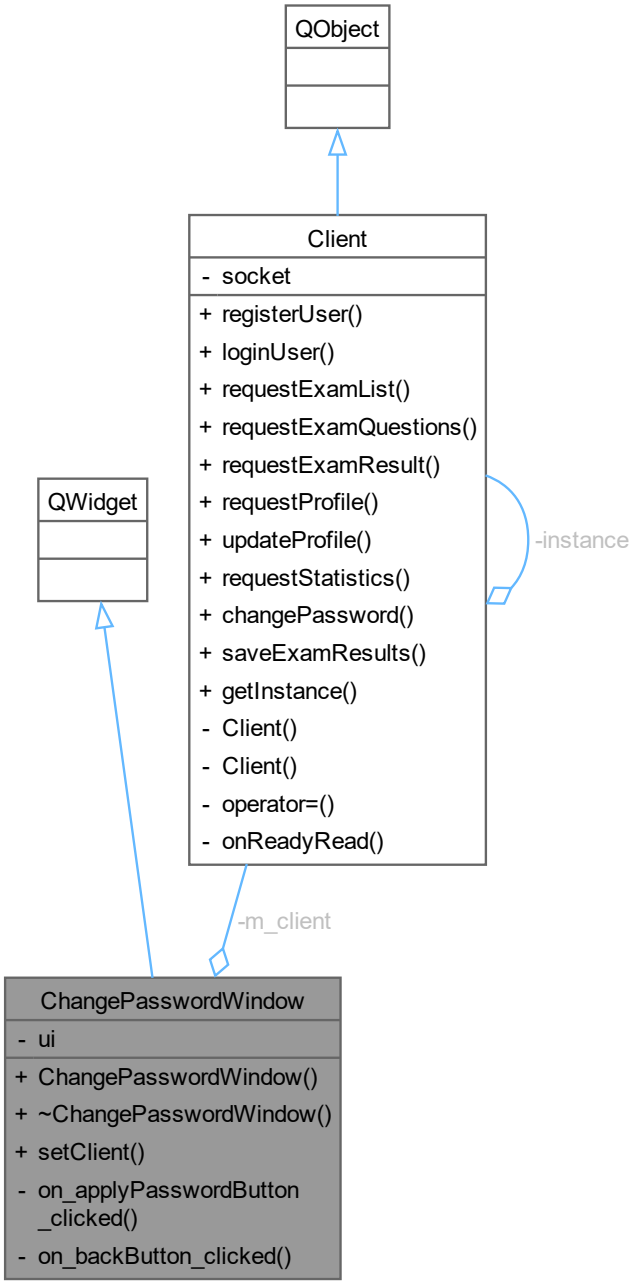
7.2 Класс ChangePasswordWindow

Класс ChangePasswordWindow предоставляет окно смены пароля пользователя.
#include <ChangePasswordWindow.h>

Граф наследования:ChangePasswordWindow:



Граф связей класса ChangePasswordWindow:



Сигналы

- void backRequested ()

  Сигнал, испускаемый при нажатии кнопки "Назад".

Открытые члены

- ChangePasswordWindow (QWidget *parent=nullptr)

  Конструктор класса ChangePasswordWindow.

- ~ChangePasswordWindow ()
  Деструктор класса.
- void setClient (Client *client)
  Устанавливает указатель на клиента для взаимодействия с сервером.

Закрытые слоты

- void on_applyPasswordButton_clicked ()
  Слот обработки нажатия кнопки "Сменить пароль".
- void on_backButton_clicked ()
  Слот обработки нажатия кнопки "Назад".

Закрытые данные

- Ui::ChangePasswordWindow * ui
  Указатель на сгенерированный интерфейс окна.
- Client * m_client
  Указатель на объект клиента для отправки команд на сервер.

7.2.1 Подробное описание

Класс ChangePasswordWindow предоставляет окно смены пароля пользователя.
Пользователь может ввести старый пароль, новый пароль и подтверждение нового пароля. При корректном вводе данные отправляются на сервер через объект клиента.

7.2.2 Конструктор(ы)

7.2.2.1 ChangePasswordWindow()

ChangePasswordWindow::ChangePasswordWindow (
QWidget * parent = nullptr) [explicit]
Конструктор класса ChangePasswordWindow.
Конструктор окна смены пароля.
Инициализирует интерфейс окна и настраивает поля ввода паролей.

Аргументы

parent	Родительский виджет, по умолчанию nullptr.

Инициализирует пользовательский интерфейс и устанавливает поля паролей в режим скрытия символов.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

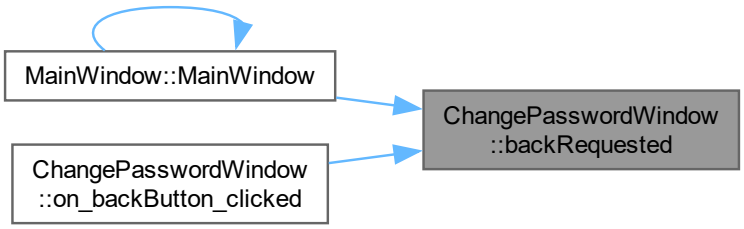
7.2.2.2 ~ChangePasswordWindow()

ChangePasswordWindow::~ChangePasswordWindow ()
Деструктор класса.
Деструктор окна смены пароля.
Освобождает память, выделенную под UI.
Освобождает ресурсы интерфейса.

7.2.3 Методы

7.2.3.1 backRequested

void ChangePasswordWindow::backRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Назад".
Используется для возврата к предыдущему экрану (например, в личный кабинет). Граф вызова функции:

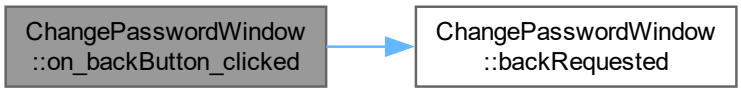


7.2.3.2 on_applyPasswordButton_clicked

void ChangePasswordWindow::on_applyPasswordButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Сменить пароль".
Слот, вызываемый при нажатии на кнопку "Применить".
Проверяет корректность введённых данных и инициирует смену пароля через клиент.
Выполняет проверку заполненности полей и совпадения нового пароля и подтверждения. При успешной валидации вызывает метод клиента для смены пароля.

7.2.3.3 on_backButton_clicked

void ChangePasswordWindow::on_backButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Назад".
Слот, вызываемый при нажатии на кнопку "Назад".
Испускает сигнал backRequested().
Генерирует сигнал для возврата на предыдущий экран. Граф вызовов:



7.2.3.4 setClient()

void ChangePasswordWindow::setClient (
Client * client)
Устанавливает указатель на клиента для взаимодействия с сервером.
Устанавливает клиент для взаимодействия с сервером.
Необходим для отправки команды смены пароля.

Аргументы

client	Указатель на объект класса Client.
client	Указатель на объект Client.

7.2.4 Поля

7.2.4.1 m_client

Client* ChangePasswordWindow::m_client [private]
Указатель на объект клиента для отправки команд на сервер.

7.2.4.2 ui

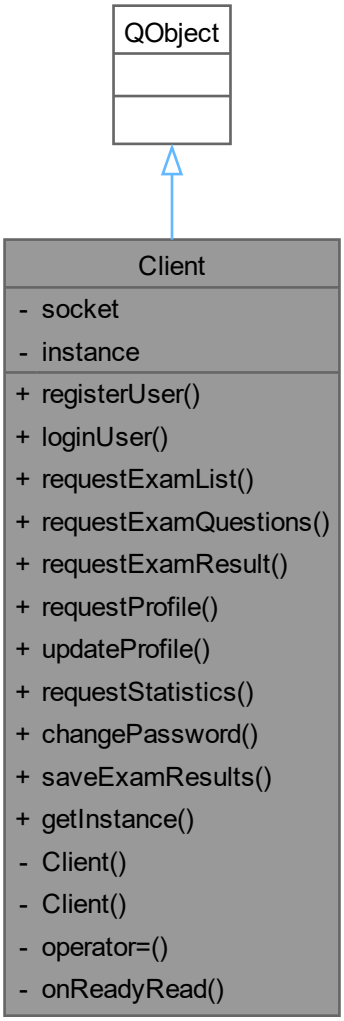
Ui::ChangePasswordWindow* ChangePasswordWindow::ui [private]
Указатель на сгенерированный интерфейс окна.
Объявления и описания членов классов находятся в файлах:

- ExamClient/ChangePasswordWindow.h
- ExamClient/ChangePasswordWindow.cpp

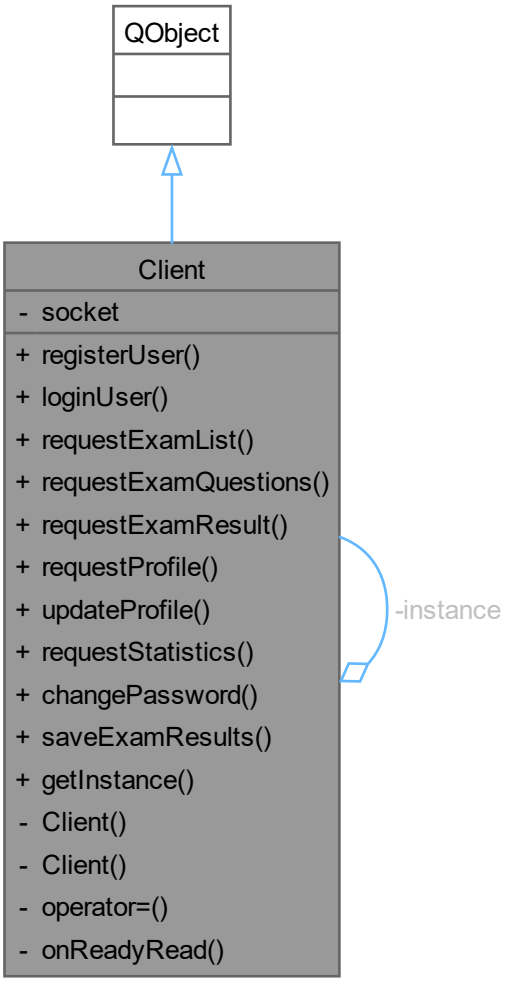
7.3 Класс Client

Класс Client — синглтон, обеспечивающий TCP-соединение с сервером.
#include <Client.h>

Граф наследования:Client:



Граф связей класса Client:



Сигналы

- void examQuestionsReceived (const QVector< ExamQuestion > &questions)

  Сигнал о получении вопросов экзамена.
- void examListReceived (const QJsonArray &examList)

  Сигнал о получении списка экзаменов.
- void examResultReceived (const QJsonObject &result)

  Сигнал о получении результата экзамена.
- void loginError (const QString &errorMessage)

  Сигнал о неудачном входе в систему.
- void loginSuccess ()

  Сигнал об успешном входе в систему.
- void registrationError (const QString &msg)

  Сигнал об ошибке при регистрации.
- void registrationSuccess ()

  Сигнал об успешной регистрации.

- void statisticsReceived (const QJsonArray &stats)

  Сигнал о получении статистики экзаменов.
- void examFinished (int score, const QVector< ExamQuestion > &questions, const QVector< int > &userAnswers)

  Сигнал о завершении экзамена.
- void profileReceived (const QJsonObject &profile)

  Сигнал о получении профиля пользователя.

Открытые члены

- void registerUser (const QString &username, const QString &password)

  Отправка команды на регистрацию пользователя.
- void loginUser (const QString &username, const QString &password)

  Отправка команды входа в систему.
- void requestExamList ()

  Запрос списка всех доступных экзаменов.
- void requestExamQuestions (int examId)

  Запрос вопросов конкретного экзамена по его идентификатору.
- void requestExamResult (int examId)

  Запрос результатов прохождения конкретного экзамена.
- void requestProfile ()

  Запрос информации о текущем пользователе (профиль).
- void updateProfile (const QJsonObject &profile)

  Отправка обновлённых данных профиля пользователя.
- void requestStatistics ()

  Запрос общей статистики по экзаменам пользователя.
- void changePassword (const QString &oldPassword, const QString &newPassword)

  Отправка запроса на смену пароля.
- void saveExamResults (int examId, int score, const QVector< ExamQuestion > &questions, const QVector< QString > &userAnswers)

  Сохраняет результаты завершённого экзамена.

Открытые статические члены

- static Client * getInstance ()

  Получение глобального экземпляра клиента.

Закрытые слоты

- void onReadyRead ()

  Слот обработки входящих данных от сервера.

Закрытые члены

- Client (QObject *parent=nullptr)

  Приватный конструктор клиента. Используется только внутри Singleton-реализации.
- Client (const Client &)=delete
- Client & operator= (const Client &)=delete

Закрытые данные

- QTcpSocket * socket

  TCP-сокет для подключения к серверу.

Закрытые статические данные

- static Client * instance = nullptr

  Единственный экземпляр клиента.

7.3.1 Подробное описание

Класс Client — синглтон, обеспечивающий TCP-соединение с сервером.
Отвечает за отправку команд на сервер и приём ответов, обрабатывая данные в формате JSON или текстовом виде. Используется клиентской частью приложения для регистрации, авторизации, получения экзаменов и статистики.

7.3.2 Конструктор(ы)

7.3.2.1 Client() [1/2]

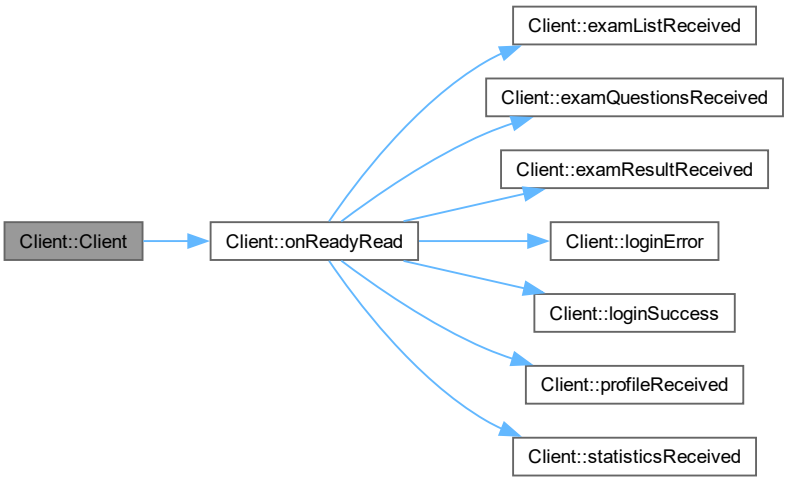
Client::Client (
QObject * parent = nullptr) [explicit], [private]

Приватный конструктор клиента. Используется только внутри Singleton-реализации.
Конструктор клиента.
Устанавливает соединение с сервером и подключает слот для обработки входящих данных.

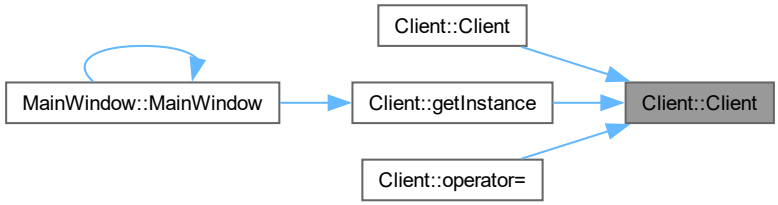
Аргументы

parent	Родительский объект (по умолчанию nullptr).

Граф вызовов:



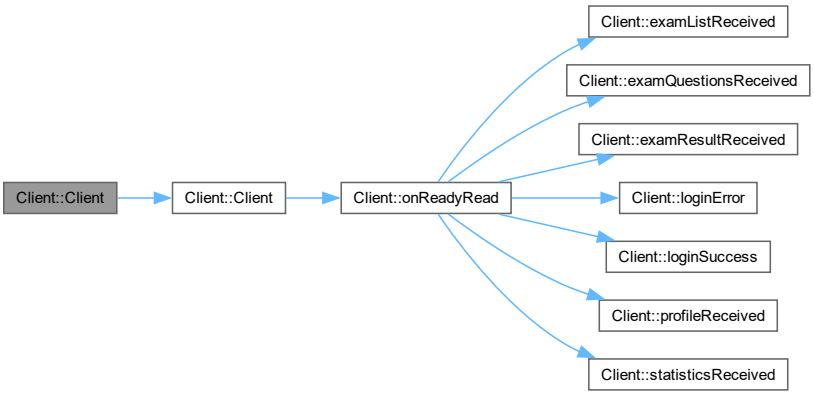
Граф вызова функции:



7.3.2.2 Client() [2/2]

Client::Client (
const Client &) [private], [delete]

Граф вызовов:



7.3.3 Методы

7.3.3.1 changePassword()

void Client::changePassword (
const QString & oldPassword,
const QString & newPassword)

Отправка запроса на смену пароля.
Отправка запроса на смену пароля пользователя.

Аргументы

oldPassword	Старый пароль.
newPassword	Новый пароль.

7.3.3.2 examFinished

void Client::examFinished (
int score,

const QVector< ExamQuestion > & questions,
const QVector< int > & userAnswers) [signal]

Сигнал о завершении экзамена.

Аргументы

score	Набранный балл.
questions	Список вопросов.
userAnswers	Ответы пользователя.

7.3.3.3 examListReceived

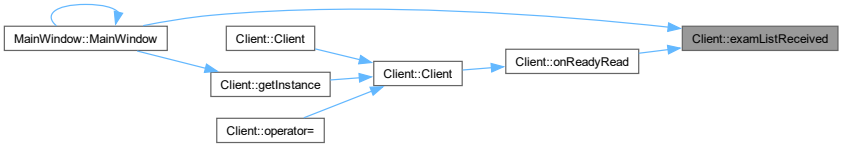
void Client::examListReceived (
const QJsonArray & examList) [signal]

Сигнал о получении списка экзаменов.

Аргументы

examList	JSON-массив с данными экзаменов.

Граф вызова функции:



7.3.3.4 examQuestionsReceived

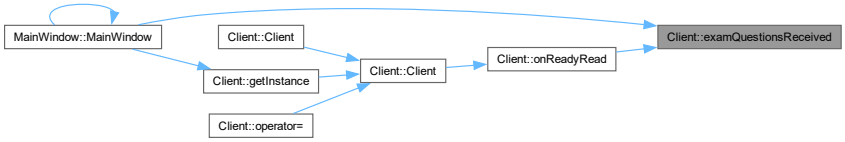
void Client::examQuestionsReceived (
const QVector< ExamQuestion > & questions) [signal]

Сигнал о получении вопросов экзамена.

Аргументы

questions	Список объектов ExamQuestion.

Граф вызова функции:



7.3.3.5 examResultReceived

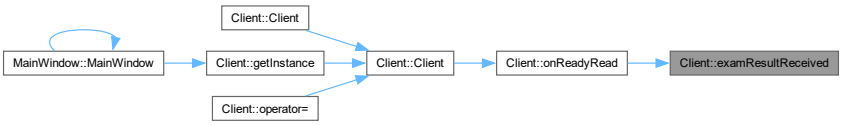
void Client::examResultReceived (
const QJsonObject & result) [signal]

Сигнал о получении результата экзамена.

Аргументы

result	JSON-объект с результатом.

Граф вызова функции:



7.3.3.6 getInstance()

Client * Client::getInstance () [static]

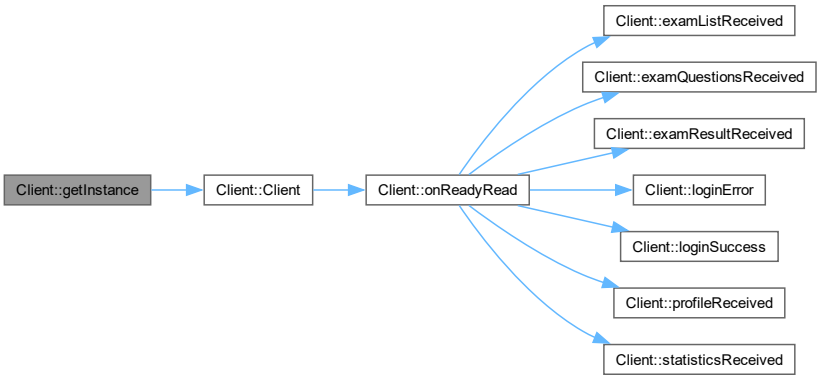
Получение глобального экземпляра клиента.
Получить единственный экземпляр клиента (паттерн Singleton).
Используется паттерн Singleton для обеспечения единого подключения к серверу.

Возвращает

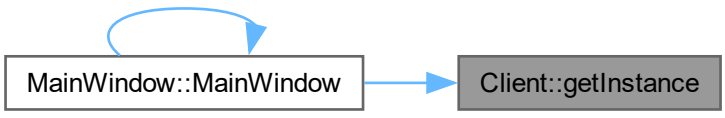
Указатель на единственный экземпляр Client.

Указатель на объект Client.

Граф вызовов:



Граф вызова функции:



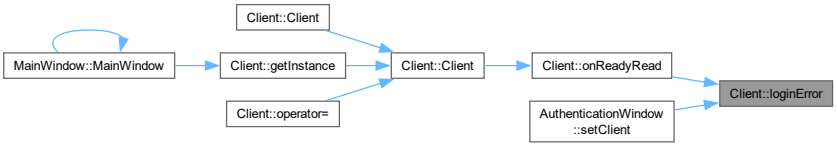
7.3.3.7 loginError

void Client::loginError (
const QString & errorMessage) [signal]
Сигнал о неудачном входе в систему.

Аргументы

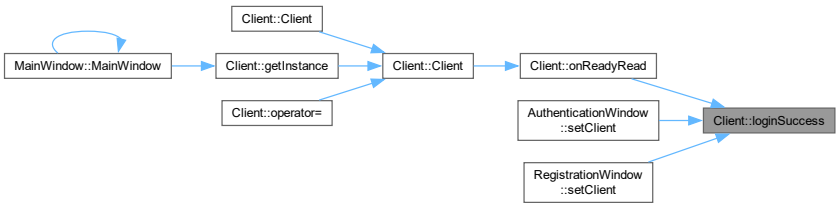
errorMessage	Сообщение об ошибке.

Граф вызова функции:



7.3.3.8 loginSuccess

void Client::loginSuccess () [signal]
Сигнал об успешном входе в систему.
Граф вызова функции:



7.3.3.9 loginUser()

void Client::loginUser (
const QString & username,
const QString & password)
Отправка команды входа в систему.
Отправка запроса входа в систему.

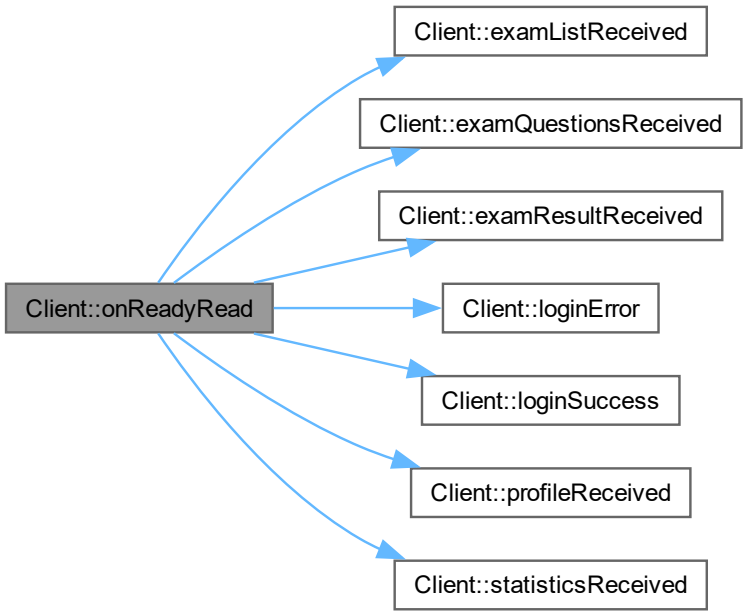
Аргументы

username	Имя пользователя.
password	Пароль.

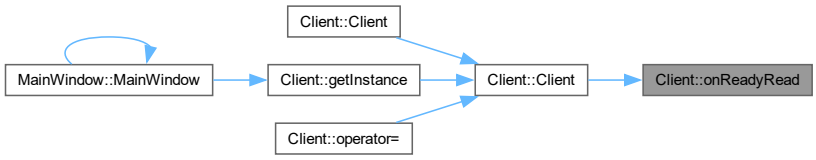
7.3.3.10 onReadyRead

void Client::onReadyRead () [private], [slot]
Слот обработки входящих данных от сервера.

Обработка входящих данных от сервера.
Определяет тип ответа (текст или JSON) и испускает соответствующие сигналы. Граф вызовов:



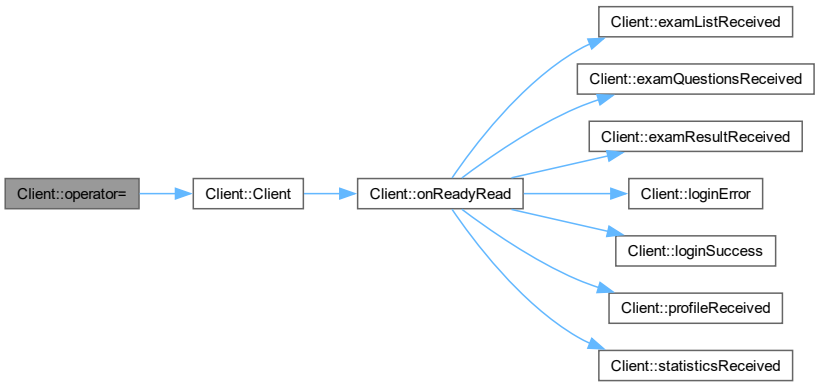
Граф вызова функции:



7.3.3.11 operator=()

Client & Client::operator= (
const Client &) [private], [delete]

Граф вызовов:



7.3.3.12 profileReceived

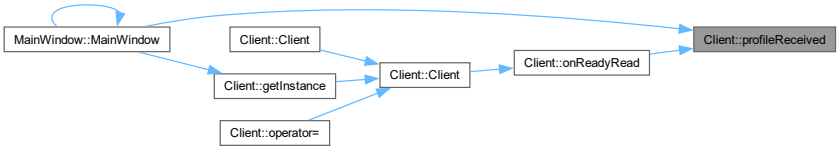
void Client::profileReceived (
const QJsonObject & profile) [signal]

Сигнал о получении профиля пользователя.

Аргументы

profile	JSON-объект с данными профиля.

Граф вызова функции:



7.3.3.13 registerUser()

void Client::registerUser (
const QString & username,
const QString & password)

Отправка команды на регистрацию пользователя.
Отправка запроса регистрации пользователя.

Аргументы

username	Имя пользователя.
password	Пароль.

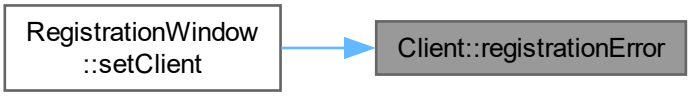
7.3.3.14 registrationError

void Client::registrationError (
const QString & msg) [signal]
Сигнал об ошибке при регистрации.

Аргументы

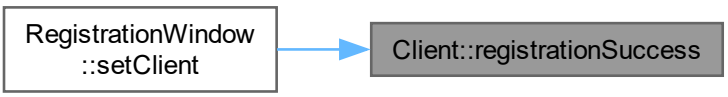
msg	Сообщение об ошибке.

Граф вызова функции:



7.3.3.15 registrationSuccess

void Client::registrationSuccess () [signal]
Сигнал об успешной регистрации.
Граф вызова функции:



7.3.3.16 requestExamList()

void Client::requestExamList ()
Запрос списка всех доступных экзаменов.

7.3.3.17 requestExamQuestions()

void Client::requestExamQuestions (
int examId)
Запрос вопросов конкретного экзамена по его идентификатору.
Запрос списка экзаменационных вопросов по ID экзамена.

Аргументы

examId	ID экзамена.
examId	Идентификатор экзамена.

7.3.3.18 requestExamResult()

void Client::requestExamResult (
int examId)
Запрос результатов прохождения конкретного экзамена.
Запрос результата конкретного экзамена.

Аргументы

examId	ID экзамена.
examId	Идентификатор экзамена.

7.3.3.19 requestProfile()

void Client::requestProfile ()
Запрос информации о текущем пользователе (профиль).
Запрос текущего профиля пользователя.

7.3.3.20 requestStatistics()

void Client::requestStatistics ()
Запрос общей статистики по экзаменам пользователя.
Запрос статистики по пройденным экзаменам.

7.3.3.21 saveExamResults()

void Client::saveExamResults (
int examId,
int score,
const QVector< ExamQuestion > & questions,
const QVector< QString > & userAnswers)
Сохраняет результаты завершённого экзамена.
Отправка результатов экзамена на сервер.

Аргументы

examId	ID экзамена.
score	Оценка (кол-во баллов).
questions	Список вопросов экзамена.
userAnswers	Ответы пользователя на вопросы.
examId	Идентификатор экзамена.
score	Количество набранных баллов.
questions	Вектор вопросов.
userAnswers	Вектор ответов пользователя.

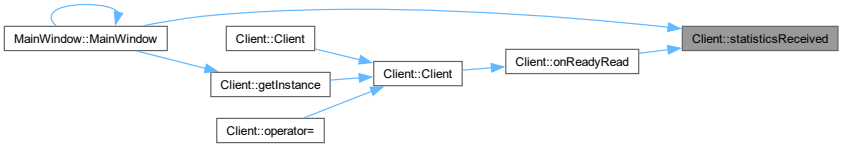
7.3.3.22 statisticsReceived

void Client::statisticsReceived (
const QJsonArray & stats) [signal]
Сигнал о получении статистики экзаменов.

Аргументы

stats	JSON-массив с результатами.

Граф вызова функции:



7.3.3.23 updateProfile()

void Client::updateProfile (
const QJsonObject & profile)
Отправка обновлённых данных профиля пользователя.
Обновление данных профиля пользователя.

Аргументы

profile	JSON-объект с данными профиля.

7.3.4 Поля

7.3.4.1 instance

Client * Client::instance = nullptr [static], [private]
Единственный экземпляр клиента.

7.3.4.2 socket

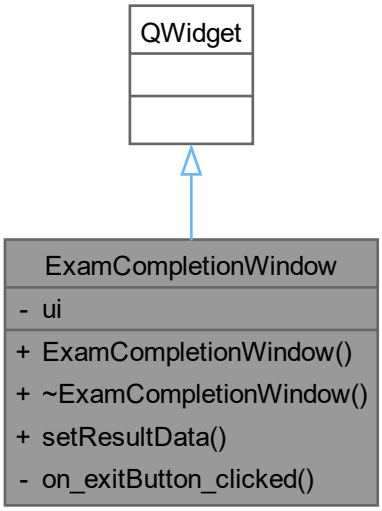
QTcpSocket* Client::socket [private]
TCP-сокет для подключения к серверу.
Объявления и описания членов классов находятся в файлах:

- ExamClient/Client.h
- ExamClient/Client.cpp

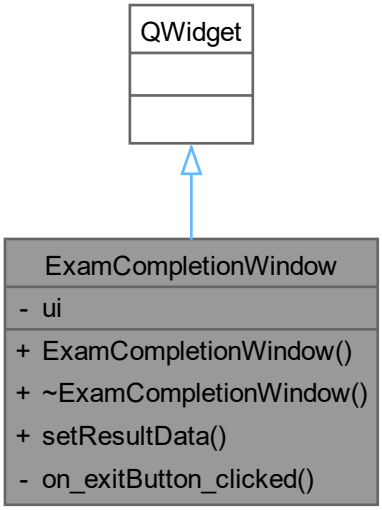
7.4 Класс ExamCompletionWindow

Класс ExamCompletionWindow реализует окно отображения результатов экзамена.
#include <ExamCompletionWindow.h>

Граф наследования:ExamCompletionWindow:



Граф связей класса ExamCompletionWindow:



Сигналы

- void exitToMenuRequested ()

  Сигнал, испускаемый при нажатии кнопки "Выйти".

Открытые члены

- ExamCompletionWindow (QWidget *parent=nullptr)

  Конструктор окна ExamCompletionWindow.
- ~ExamCompletionWindow ()

  Деструктор.
- void setResultData (int score, const QVector< ExamQuestion > &questions, const QVector< QString > &userAnswers)

  Устанавливает данные о результате экзамена и отображает их в окне.

Закрытые слоты

- void on_exitButton_clicked ()

  Слот обработки нажатия кнопки "Выйти".

Закрытые данные

- Ui::ExamCompletionWindow * ui

  Указатель на интерфейс окна.

7.4.1 Подробное описание

Класс ExamCompletionWindow реализует окно отображения результатов экзамена.
Это окно появляется после завершения экзамена и показывает пользователю:

- общее количество набранных баллов;
- список вопросов;
- ответы пользователя и правильные ответы.

7.4.2 Конструктор(ы)

7.4.2.1 ExamCompletionWindow()

ExamCompletionWindow::ExamCompletionWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна ExamCompletionWindow.
Конструктор окна завершения экзамена.
Инициализирует элементы пользовательского интерфейса.

Аргументы

parent	Родительский виджет (по умолчанию nullptr).

Инициализирует пользовательский интерфейс.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

7.4.2.2 ~ExamCompletionWindow()

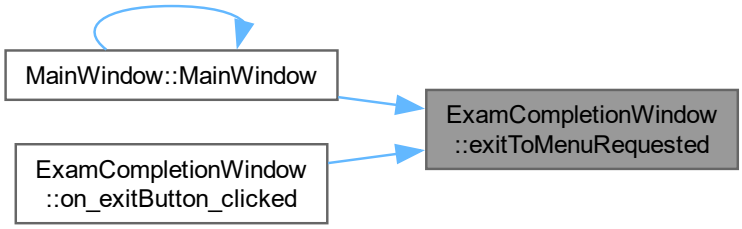
ExamCompletionWindow::~ExamCompletionWindow ()
Деструктор.
Деструктор окна завершения экзамена.
Освобождает память, выделенную под UI.
Освобождает ресурсы, выделенные для UI.

7.4.3 Методы

7.4.3.1 exitToMenuRequested

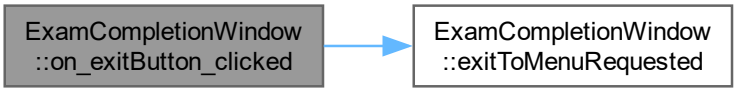
void ExamCompletionWindow::exitToMenuRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Выйти".

Используется для возврата в главное меню приложения. Граф вызова функции:



7.4.3.2 on_exitButton_clicked

void ExamCompletionWindow::on_exitButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Выйти".
Слот обработки нажатия кнопки "Выход".
Испускает сигнал exitToMenuRequested().
Генерирует сигнал возврата в главное меню. Граф вызовов:



7.4.3.3 setResultData()

void ExamCompletionWindow::setResultData (
int score,
const QVector< ExamQuestion > & questions,
const QVector< QString > & userAnswers)
Устанавливает данные о результате экзамена и отображает их в окне.
Устанавливает данные результата экзамена и отображает их.
Метод используется для отображения результатов после завершения экзамена. Включает расчёт итогового балла, отображение пользовательских и правильных ответов.

Аргументы

score	Количество правильных ответов, набранное пользователем.
questions	Список всех вопросов, заданных в экзамене.
userAnswers	Список текстовых ответов, выбранных пользователем.

Отображает общий счёт и подробности по каждому вопросу, включая текст вопроса, ответ пользователя и правильный ответ.

Аргументы

score	Количество набранных баллов.
questions	Список вопросов экзамена.
userAnswers	Ответы пользователя.

7.4.4 Поля

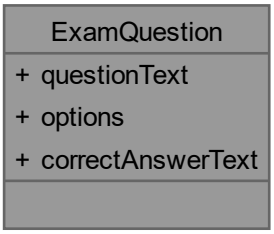
7.4.4.1 ui

Ui::ExamCompletionWindow* ExamCompletionWindow::ui [private]
Указатель на интерфейс окна.
Объявления и описания членов классов находятся в файлах:

- ExamClient/ExamCompletionWindow.h
- ExamClient/ExamCompletionWindow.cpp

7.5 Структура ExamQuestion

Структура ExamQuestion представляет собой модель одного вопроса экзамена.
#include <ExamQuestion.h>
Граф связей класса ExamQuestion:



Поля данных

- QString questionText

  Текст самого вопроса (например, "Сколько будет 2 + 2?")
- QStringList options

  Список всех вариантов ответа (например, ["2", "4", "8"])
- QString correctAnswerText

  Правильный ответ (например, "4")

7.5.1 Подробное описание

Структура ExamQuestion представляет собой модель одного вопроса экзамена.
Содержит текст вопроса, список всех возможных вариантов ответа и текст правильного ответа.
Используется как на стороне клиента, так и при обработке результатов экзамена.

7.5.2 Поля

7.5.2.1 correctAnswerText

QString ExamQuestion::correctAnswerText
Правильный ответ (например, "4")

7.5.2.2 options

QStringList ExamQuestion::options
Список всех вариантов ответа (например, ["2", "4", "8"])

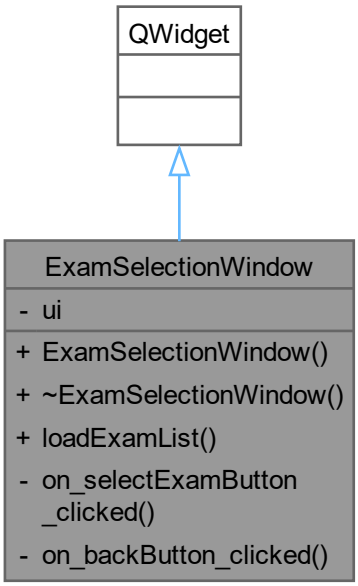
7.5.2.3 questionText

QString ExamQuestion::questionText
Текст самого вопроса (например, "Сколько будет 2 + 2?")
Объявления и описания членов структуры находятся в файле:

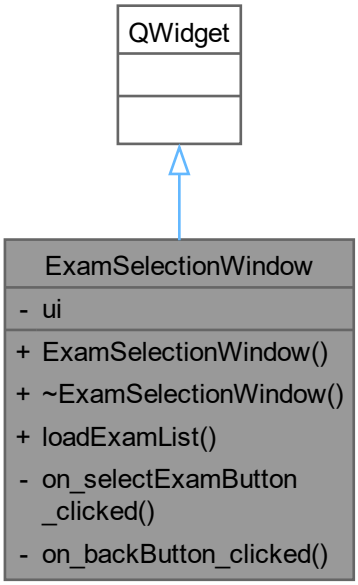
- ExamClient/ExamQuestion.h

7.6 Класс ExamSelectionWindow

Класс ExamSelectionWindow реализует окно выбора экзамена.
#include <ExamSelectionWindow.h>
Граф наследования:ExamSelectionWindow:



Граф связей класса ExamSelectionWindow:



Сигналы

- void examSelected (int examId)

  Сигнал, испускаемый при выборе экзамена пользователем.
- void backRequested ()

  Сигнал возврата на предыдущий экран.

Открытые члены

- ExamSelectionWindow (QWidget *parent=nullptr)

  Конструктор окна ExamSelectionWindow.
- ~ExamSelectionWindow ()

  Деструктор.
- void loadExamList (const QJsonArray &examList)

  Загружает список доступных экзаменов в интерфейс.

Закрытые слоты

- void on_selectExamButton_clicked ()

  Слот обработки нажатия кнопки "Выбрать экзамен".
- void on_backButton_clicked ()

  Слот обработки нажатия кнопки "Назад".

Закрытые данные

- Ui::ExamSelectionWindow * ui

  Указатель на интерфейс окна.

7.6.1 Подробное описание

Класс ExamSelectionWindow реализует окно выбора экзамена.
Окно отображает список доступных экзаменов, полученных от сервера, и позволяет пользователю выбрать один из них. После выбора испускается сигнал с ID выбранного экзамена.

7.6.2 Конструктор(ы)

7.6.2.1 ExamSelectionWindow()

ExamSelectionWindow::ExamSelectionWindow (
QWidget * parent = nullptr) [explicit]
Конструктор окна ExamSelectionWindow.
Конструктор окна выбора экзамена.
Инициализирует интерфейс выбора экзамена.

Аргументы

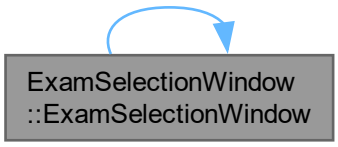
parent	Родительский виджет (по умолчанию nullptr).

Инициализирует пользовательский интерфейс и очищает список экзаменов.

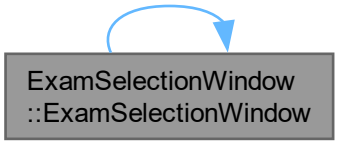
Аргументы

parent	Родительский виджет.

Граф вызовов:



Граф вызова функции:



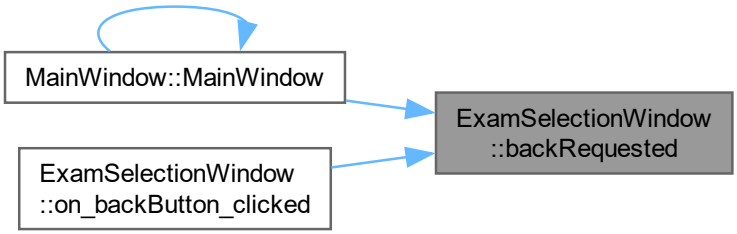
7.6.2.2 ~ExamSelectionWindow()

ExamSelectionWindow::~ExamSelectionWindow ()
Деструктор.
Деструктор окна выбора экзамена.
Освобождает ресурсы, выделенные под интерфейс.
Освобождает ресурсы, выделенные для UI.

7.6.3 Методы

7.6.3.1 backRequested

void ExamSelectionWindow::backRequested () [signal]
Сигнал возврата на предыдущий экран.
Граф вызова функции:



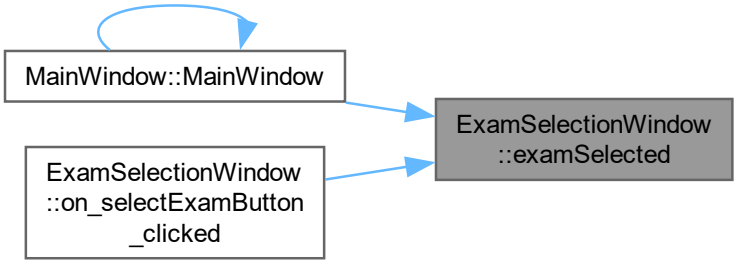
7.6.3.2 examSelected

void ExamSelectionWindow::examSelected (
int examId) [signal]
Сигнал, испускаемый при выборе экзамена пользователем.
Используется для перехода к следующему этапу — загрузке вопросов.

Аргументы

examId	Идентификатор выбранного экзамена.

Граф вызова функции:



7.6.3.3 loadExamList()

void ExamSelectionWindow::loadExamList (
const QJsonArray & examList)
Загружает список доступных экзаменов в интерфейс.
Загружает список экзаменов и отображает их в виджете.
Каждый элемент массива должен содержать поля "id" и "name". Эти данные используются для отображения и выбора экзамена.

Аргументы

examList	Массив JSON-объектов с описанием экзаменов.

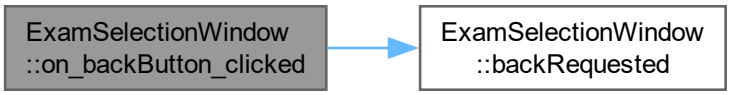
Получает JSON-массив экзаменов от клиента и добавляет каждый экзамен в список с отображением названия и сохранением ID в пользовательских данных.

Аргументы

examList	JSON-массив с экзаменами (каждый элемент содержит поля "id" и "name").

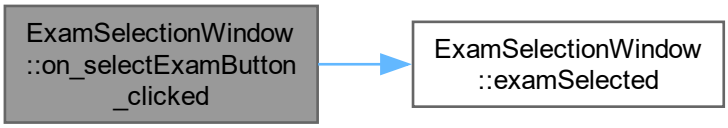
7.6.3.4 on_backButton_clicked

void ExamSelectionWindow::on_backButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Назад".
Испускает сигнал backRequested().
Испускает сигнал backRequested() для возврата к предыдущему окну. Граф вызовов:



7.6.3.5 on_selectExamButton_clicked

void ExamSelectionWindow::on_selectExamButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Выбрать экзамен".
Извлекает ID выбранного элемента и испускает сигнал examSelected().
Извлекает ID выбранного экзамена и испускает сигнал examSelected. Граф вызовов:



7.6.4 Поля

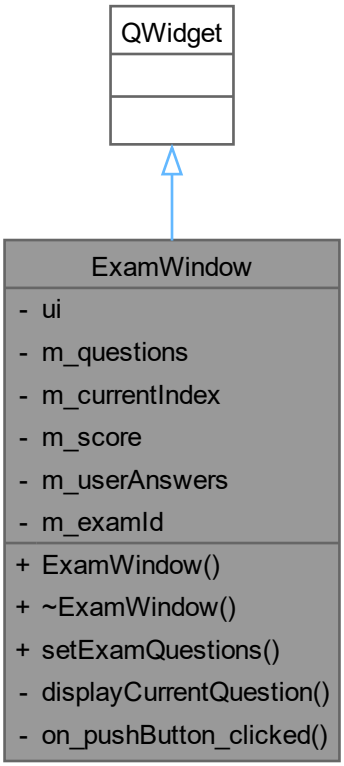
7.6.4.1 ui

Ui::ExamSelectionWindow* ExamSelectionWindow::ui [private]
Указатель на интерфейс окна.
Объявления и описания членов классов находятся в файлах:

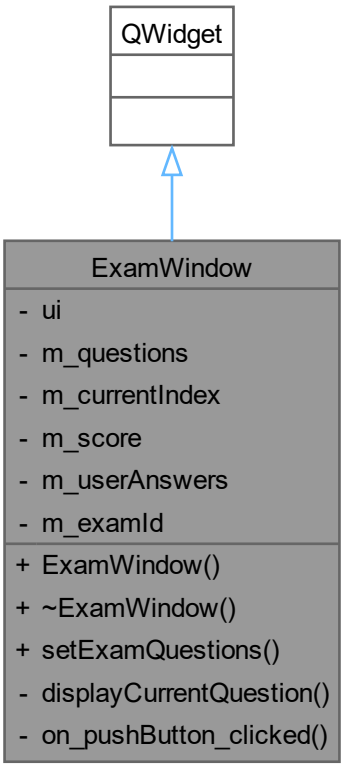
- ExamClient/ExamSelectionWindow.h
- ExamClient/ExamSelectionWindow.cpp

7.7 Класс ExamWindow

Класс ExamWindow предоставляет интерфейс для прохождения экзамена.
#include <ExamWindow.h>
Граф наследования:ExamWindow:



Граф связей класса ExamWindow:



Сигналы

- void examFinished (int examId, int score, const QVector< ExamQuestion > &questions, const QVector< QString > &userAnswers)

  Сигнал завершения экзамена.

Открытые члены

- ExamWindow (QWidget *parent=nullptr)

  Конструктор окна экзамена.
- ~ExamWindow ()

  Деструктор.
- void setExamQuestions (int examId, const QVector< ExamQuestion > &questions)

  Загружает вопросы экзамена и запускает его отображение.

Закрытые слоты

- void on_pushButton_clicked ()

  Обработка нажатия кнопки "Далее / Ответить".

Закрытые члены

- void displayCurrentQuestion ()

  Отображает текущий вопрос и доступные варианты ответа.

Закрытые данные

- Ui::ExamWindow * ui

  Указатель на интерфейс окна.
- QVector< ExamQuestion > m_questions

  Список всех вопросов экзамена.
- int m_currentIndex

  Индекс текущего вопроса.
- int m_score

  Количество правильных ответов.
- QVector< int > m_userAnswers

  Индексы выбранных ответов по каждому вопросу.
- int m_examId = -1

  Идентификатор текущего экзамена.

7.7.1 Подробное описание

Класс ExamWindow предоставляет интерфейс для прохождения экзамена.
Отображает по одному вопросу с вариантами ответов и отслеживает ответы пользователя. По завершении экзамена отправляет результаты через сигнал.

7.7.2 Конструктор(ы)

7.7.2.1 ExamWindow()

ExamWindow::ExamWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна экзамена.
Инициализирует UI-компоненты и внутреннее состояние.

Аргументы

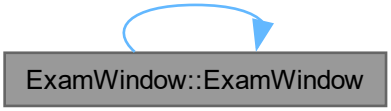
parent	Родительский виджет, по умолчанию nullptr.

Инициализирует интерфейс и устанавливает начальные значения для индекса вопроса и счёта.

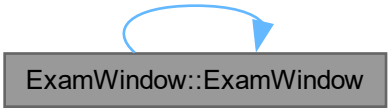
Аргументы

parent	Родительский виджет.

Граф вызовов:



Граф вызова функции:



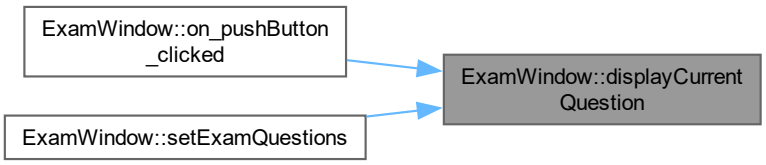
7.7.2.2 ~ExamWindow()

ExamWindow::~ExamWindow ()
Деструктор.
Деструктор окна экзамена.
Освобождает ресурсы, связанные с UI.
Освобождает ресурсы интерфейса.

7.7.3 Методы

7.7.3.1 displayCurrentQuestion()

void ExamWindow::displayCurrentQuestion () [private]
Отображает текущий вопрос и доступные варианты ответа.
Отображает текущий вопрос на интерфейсе.
Используется для обновления интерфейса при переходе к следующему вопросу.
Загружает текст вопроса и доступные варианты ответов в радиокнопки. Граф вызова функции:



7.7.3.2 examFinished

void ExamWindow::examFinished (
int examId,
int score,
const QVector< ExamQuestion > & questions,
const QVector< QString > & userAnswers) [signal]
Сигнал завершения экзамена.
Испускается, когда пользователь ответил на все вопросы.

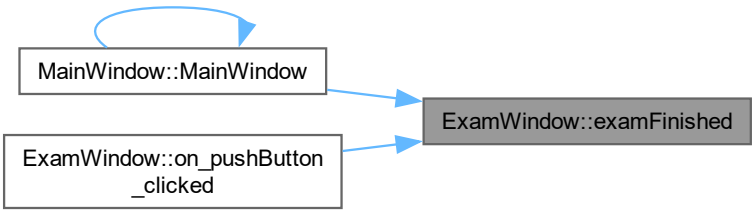
Аргументы

examId	Идентификатор экзамена.

Аргументы

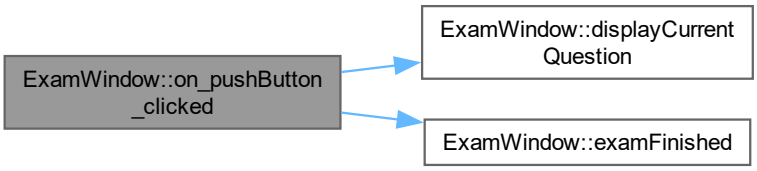
score	Количество правильных ответов.
questions	Все вопросы, заданные в экзамене.
userAnswers	Ответы пользователя в текстовом виде.

Граф вызова функции:



7.7.3.3 on_pushButton_clicked

void ExamWindow::on_pushButton_clicked () [private], [slot]
Обработка нажатия кнопки "Далее / Ответить".
Обработка нажатия кнопки "Ответить / Далее".
Проверяет выбранный ответ, сохраняет его и отображает следующий вопрос. При завершении — формирует результат и испускает сигнал examFinished().
Проверяет, выбран ли ответ, сохраняет его и обновляет счёт, если ответ верный. При завершении всех вопросов испускает сигнал examFinished(). Граф вызовов:



7.7.3.4 setExamQuestions()

void ExamWindow::setExamQuestions (
int examId,
const QVector< ExamQuestion > & questions)
Загружает вопросы экзамена и запускает его отображение.
Устанавливает список вопросов экзамена и инициализирует отображение первого.
Метод вызывается после получения списка вопросов от сервера. Обнуляет счёт, очищает старые ответы и отображает первый вопрос.

Аргументы

examId	Идентификатор экзамена.
questions	Вектор вопросов экзамена.

Также очищает старые данные, сбрасывает счёт и список пользовательских ответов. Если вопросов слишком много — отображается сообщение об ошибке.

Аргументы

examId	Идентификатор текущего экзамена.
questions	Список вопросов экзамена.

Граф вызовов:

7.7.4 Поля

7.7.4.1 m_currentIndex

int ExamWindow::m_currentIndex [private]
Индекс текущего вопроса.

7.7.4.2 m_examId

int ExamWindow::m_examId = -1 [private]
Идентификатор текущего экзамена.

7.7.4.3 m_questions

QVector<ExamQuestion> ExamWindow::m_questions [private]
Список всех вопросов экзамена.

7.7.4.4 m_score

int ExamWindow::m_score [private]
Количество правильных ответов.

7.7.4.5 m_userAnswers

QVector<int> ExamWindow::m_userAnswers [private]
Индексы выбранных ответов по каждому вопросу.

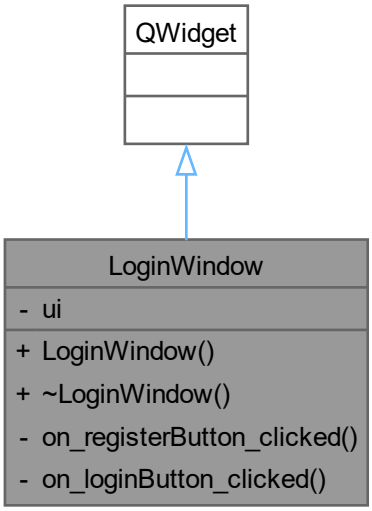
7.7.4.6 ui

Ui::ExamWindow* ExamWindow::ui [private]
Указатель на интерфейс окна.
Объявления и описания членов классов находятся в файлах:

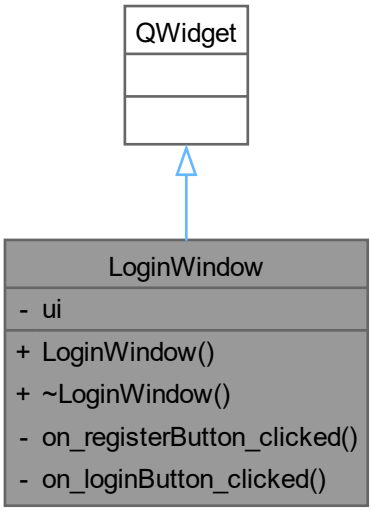
- ExamClient/ExamWindow.h
- ExamClient/ExamWindow.cpp

7.8 Класс LoginWindow

Класс LoginWindow реализует окно начального входа пользователя.
#include <LoginWindow.h>
Граф наследования:LoginWindow:



Граф связей класса LoginWindow:



Сигналы

- void registrationRequested ()

  Сигнал, испускаемый при нажатии кнопки "Регистрация".
- void loginRequested ()

  Сигнал, испускаемый при нажатии кнопки "Вход".

Открытые члены

- LoginWindow (QWidget *parent=nullptr)

  Конструктор LoginWindow.
- ~LoginWindow ()

  Деструктор.

Закрытые слоты

- void on_registerButton_clicked ()

  Слот обработки нажатия кнопки "Регистрация".
- void on_loginButton_clicked ()

  Слот обработки нажатия кнопки "Вход".

Закрытые данные

- Ui::LoginWindow * ui

  Указатель на интерфейс окна входа.

7.8.1 Подробное описание

Класс LoginWindow реализует окно начального входа пользователя.
Содержит две основные кнопки: для перехода к окну авторизации и к регистрации. Не содержит полей ввода — только действия выбора направления.

7.8.2 Конструктор(ы)

7.8.2.1 LoginWindow()

LoginWindow::LoginWindow (
QWidget * parent = nullptr) [explicit]

Конструктор LoginWindow.
Конструктор окна входа.
Инициализирует элементы интерфейса окна входа.

Аргументы

parent	Родительский виджет (по умолчанию nullptr).

Инициализирует интерфейс окна входа в систему.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

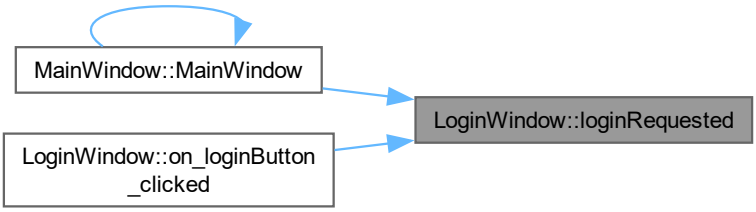
7.8.2.2 ~LoginWindow()

LoginWindow::~LoginWindow ()
Деструктор.
Деструктор окна входа.
Освобождает память, выделенную под UI.
Освобождает ресурсы, выделенные под интерфейс.

7.8.3 Методы

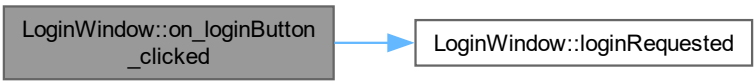
7.8.3.1 loginRequested

void LoginWindow::loginRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Вход".
Используется для перехода к окну авторизации. Граф вызова функции:



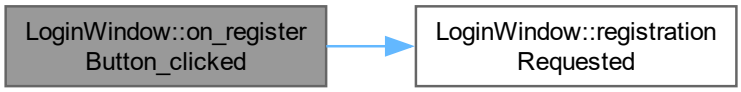
7.8.3.2 on_loginButton_clicked

void LoginWindow::on_loginButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Вход".
Слот, вызываемый при нажатии на кнопку "Войти".
Испускает сигнал loginRequested().
Генерирует сигнал loginRequested() для перехода к окну аутентификации. Граф вызовов:



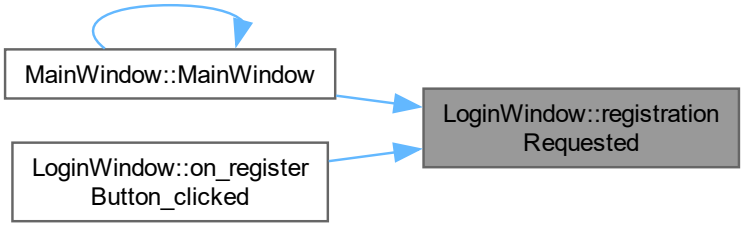
7.8.3.3 on_registerButton_clicked

void LoginWindow::on_registerButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Регистрация".
Слот, вызываемый при нажатии на кнопку "Регистрация".
Испускает сигнал registrationRequested().
Генерирует сигнал registrationRequested() для перехода к окну регистрации. Граф вызовов:



7.8.3.4 registrationRequested

void LoginWindow::registrationRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Регистрация".
Используется для перехода к окну регистрации. Граф вызова функции:



7.8.4 Поля

7.8.4.1 ui

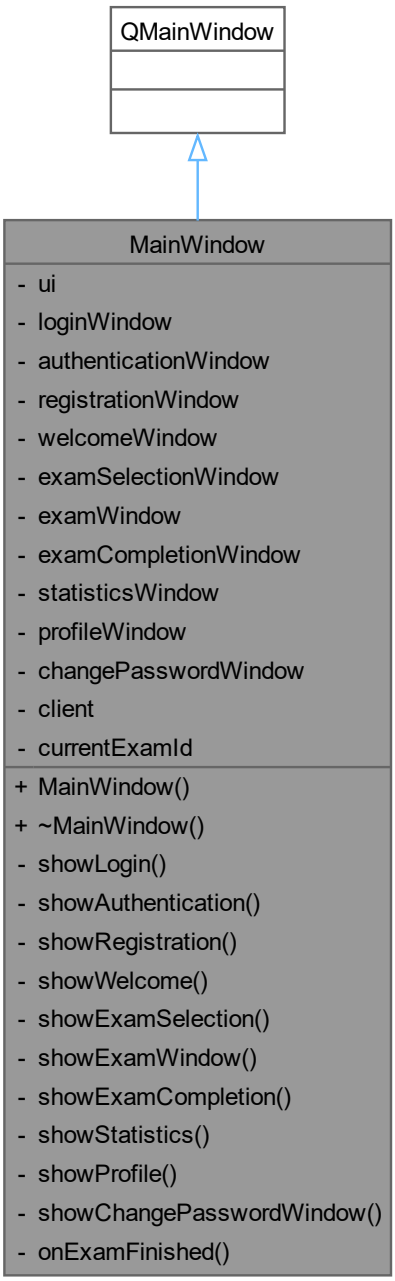
Ui::LoginWindow* LoginWindow::ui [private]
Указатель на интерфейс окна входа.
Объявления и описания членов классов находятся в файлах:

- ExamClient/LoginWindow.h
- ExamClient/LoginWindow.cpp

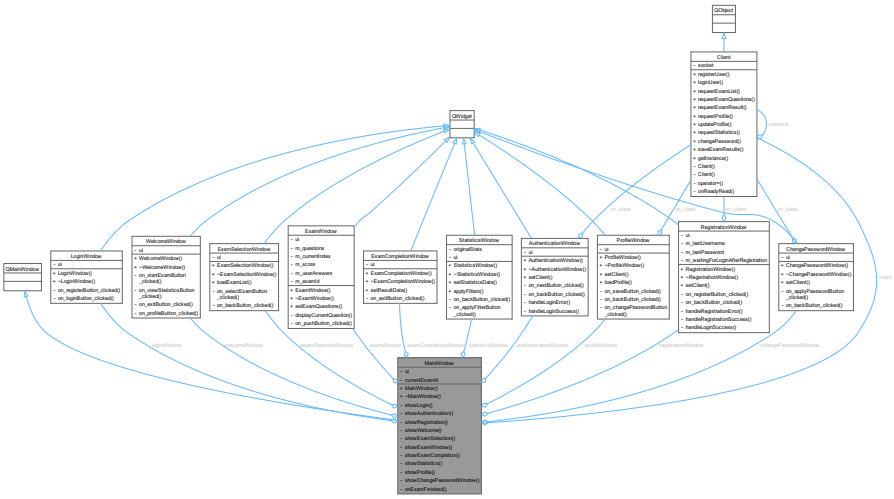
7.9 Класс MainWindow

Класс MainWindow управляет всеми экранами и логикой переходов в приложении.
#include <MainWindow.h>

Граф наследования:MainWindow:



Граф связей класса MainWindow:



Открытые члены

- MainWindow (QWidget *parent=nullptr)

  Конструктор основного окна.
- ~MainWindow ()

  Деструктор.

Закрытые слоты

- void showLogin ()

  Переход к окну входа.
- void showAuthentication ()

  Переход к окну авторизации.
- void showRegistration ()

  Переход к окну регистрации.
- void showWelcome ()

  Переход к главному экрану (welcome).
- void showExamSelection ()

  Переход к окну выбора экзамена.
- void showExamWindow (int examId)

  Переход к окну с вопросами выбранного экзамена.
- void showExamCompletion ()

  Переход к окну с результатами после завершения экзамена.
- void showStatistics ()

  Переход к экрану со статистикой.
- void showProfile ()

  Переход к окну профиля и загрузка данных пользователя.
- void showChangePasswordWindow ()

  Переход к окну смены пароля.
- void onExamFinished (int examId, int score, const QVector< ExamQuestion > &questions, const QVector< QString > &userAnswers)

  Слот, вызываемый по завершении экзамена.

Закрытые данные

- Ui::MainWindow * ui

  Указатель на UI интерфейс.
- LoginWindow * loginWindow
- AuthenticationWindow * authenticationWindow
- RegistrationWindow * registrationWindow
- WelcomeWindow * welcomeWindow
- ExamSelectionWindow * examSelectionWindow
- ExamWindow * examWindow
- ExamCompletionWindow * examCompletionWindow
- StatisticsWindow * statisticsWindow
- ProfileWindow * profileWindow
- ChangePasswordWindow * changePasswordWindow
- Client * client

  Указатель на синглтон-клиент (TCP-соединение).
- int currentExamId = -1

  Текущий идентификатор выбранного экзамена.

7.9.1 Подробное описание

Класс MainWindow управляет всеми экранами и логикой переходов в приложении.
Использует QStackedWidget для переключения между окнами. Подключает все дочерние окна и клиента, обрабатывает сигналы взаимодействия между компонентами.

7.9.2 Конструктор(ы)

7.9.2.1 MainWindow()

MainWindow::MainWindow (
QWidget * parent = nullptr) [explicit]

Конструктор основного окна.
Конструктор основного окна приложения.
Создаёт все дочерние окна, подключает их, и настраивает начальный экран.

Аргументы

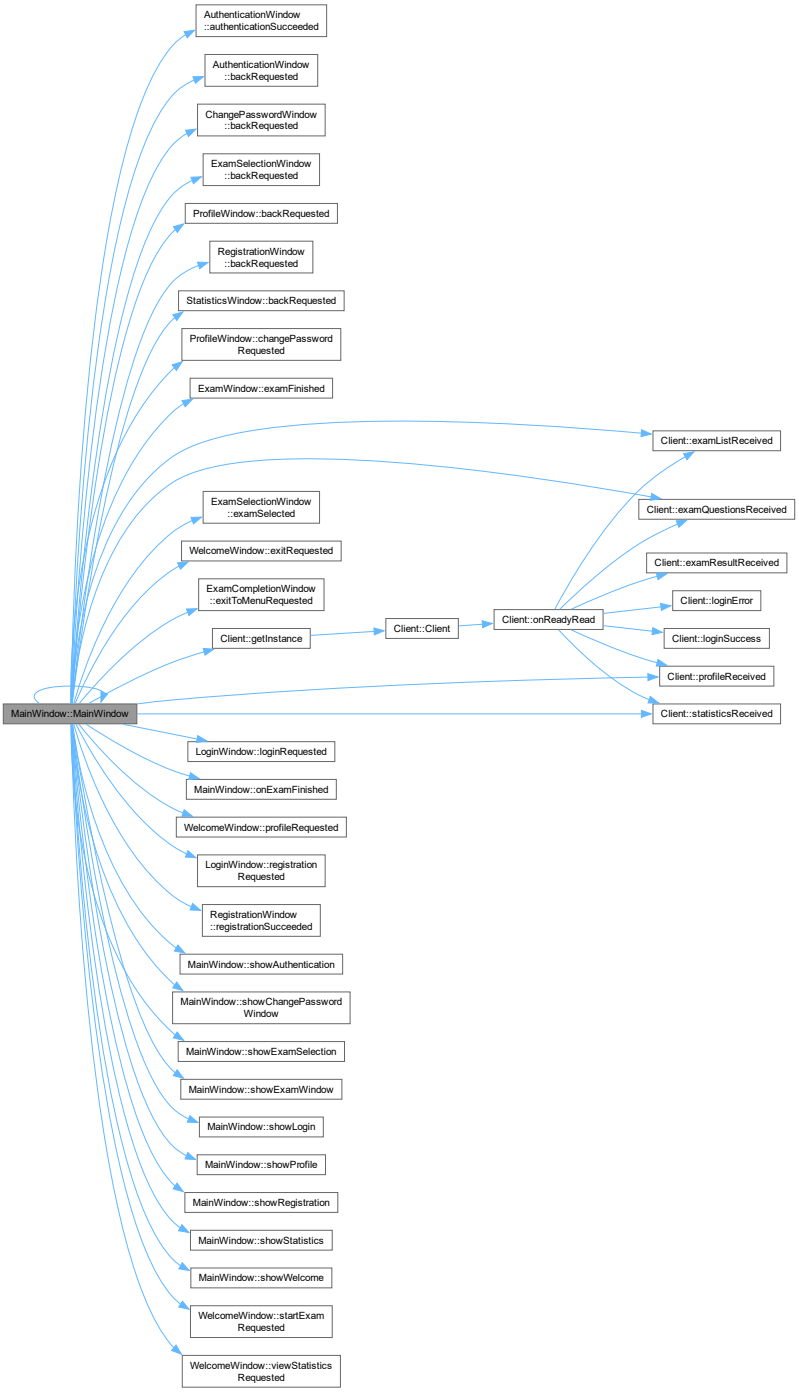
parent	Родительский виджет (по умолчанию nullptr).

Инициализирует все дочерние окна, устанавливает связи между сигналами и слотами и отображает начальный экран (login).

Аргументы

parent	Родительский виджет (обычно nullptr).

Граф вызовов:



Граф вызова функции:

7.9.2.2 ~MainWindow()

MainWindow::~MainWindow ()
Деструктор.
Деструктор основного окна.
Освобождает ресурсы, выделенные для интерфейса.

7.9.3 Методы

7.9.3.1 onExamFinished

void MainWindow::onExamFinished (
int examId,
int score,
const QVector< ExamQuestion > & questions,
const QVector< QString > & userAnswers) [private], [slot]
Слот, вызываемый по завершении экзамена.
Слот обработки завершения экзамена.
Сохраняет результаты и показывает окно с итогами.

Аргументы

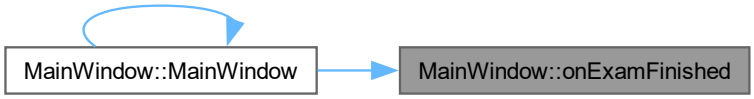
examId	ID экзамена.
score	Количество правильных ответов.
questions	Все вопросы экзамена.
userAnswers	Ответы пользователя в текстовом виде.

Отправляет результаты на сервер и переключает на окно с итогами.

Аргументы

examId	Идентификатор экзамена.
score	Количество правильных ответов.
questions	Список всех вопросов.
userAnswers	Ответы пользователя в текстовом виде.

Граф вызова функции:



7.9.3.2 showAuthentication

void MainWindow::showAuthentication () [private], [slot]
Переход к окну авторизации.
Отображает окно авторизации. Граф вызова функции:

7.9.3.3 showChangePasswordWindow

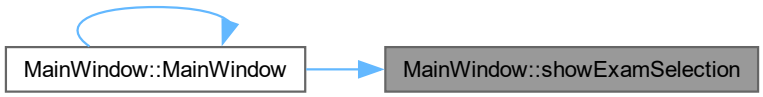
void MainWindow::showChangePasswordWindow () [private], [slot]
Переход к окну смены пароля.
Отображает окно смены пароля. Граф вызова функции:

7.9.3.4 showExamCompletion

void MainWindow::showExamCompletion () [private], [slot]
Переход к окну с результатами после завершения экзамена.
Отображает окно завершения экзамена.

7.9.3.5 showExamSelection

void MainWindow::showExamSelection () [private], [slot]
Переход к окну выбора экзамена.
Отображает окно выбора экзамена и запрашивает список экзаменов у сервера. Граф вызова функции:



7.9.3.6 showExamWindow

void MainWindow::showExamWindow (
int examId) [private], [slot]
Переход к окну с вопросами выбранного экзамена.
Отображает окно экзамена и запрашивает список вопросов.

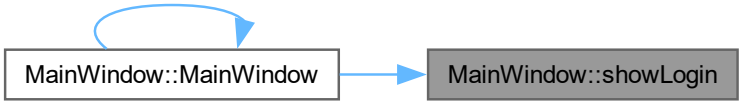
Аргументы

examId	Идентификатор выбранного экзамена.

Граф вызова функции:

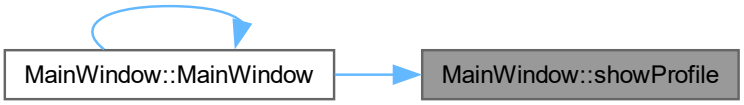
7.9.3.7 showLogin

void MainWindow::showLogin () [private], [slot]
Переход к окну входа.
Отображает окно входа. Граф вызова функции:



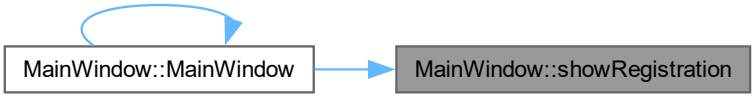
7.9.3.8 showProfile

void MainWindow::showProfile () [private], [slot]
Переход к окну профиля и загрузка данных пользователя.
Отображает окно профиля и запрашивает данные у сервера. Граф вызова функции:



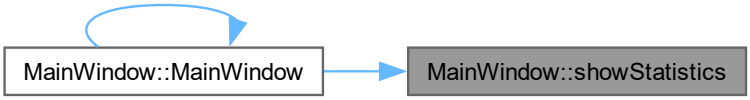
7.9.3.9 showRegistration

void MainWindow::showRegistration () [private], [slot]
Переход к окну регистрации.
Отображает окно регистрации. Граф вызова функции:



7.9.3.10 showStatistics

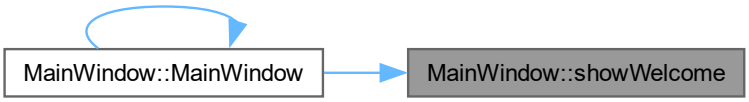
void MainWindow::showStatistics () [private], [slot]
Переход к экрану со статистикой.
Отображает окно статистики и запрашивает данные у сервера. Граф вызова функции:



7.9.3.11 showWelcome

void MainWindow::showWelcome () [private], [slot]
Переход к главному экрану (welcome).

Отображает главное меню (welcome). Граф вызова функции:



7.9.4 Поля

7.9.4.1 authenticationWindow

AuthenticationWindow* MainWindow::authenticationWindow [private]

7.9.4.2 changePasswordWindow

ChangePasswordWindow* MainWindow::changePasswordWindow [private]

7.9.4.3 client

Client* MainWindow::client [private]
Указатель на синглтон-клиент (TCP-соединение).

7.9.4.4 currentExamId

int MainWindow::currentExamId = -1 [private]
Текущий идентификатор выбранного экзамена.

7.9.4.5 examCompletionWindow

ExamCompletionWindow* MainWindow::examCompletionWindow [private]

7.9.4.6 examSelectionWindow

ExamSelectionWindow* MainWindow::examSelectionWindow [private]

7.9.4.7 examWindow

ExamWindow* MainWindow::examWindow [private]

7.9.4.8 loginWindow

LoginWindow* MainWindow::loginWindow [private]

7.9.4.9 profileWindow

ProfileWindow* MainWindow::profileWindow [private]

7.9.4.10 registrationWindow

RegistrationWindow* MainWindow::registrationWindow [private]

7.9.4.11 statisticsWindow

StatisticsWindow* MainWindow::statisticsWindow [private]

7.9.4.12 ui

Ui::MainWindow* MainWindow::ui [private]
Указатель на UI интерфейс.

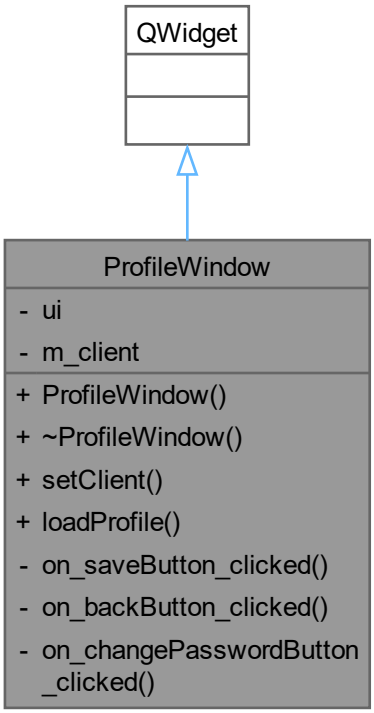
7.9.4.13 welcomeWindow

WelcomeWindow* MainWindow::welcomeWindow [private]
Объявления и описания членов классов находятся в файлах:

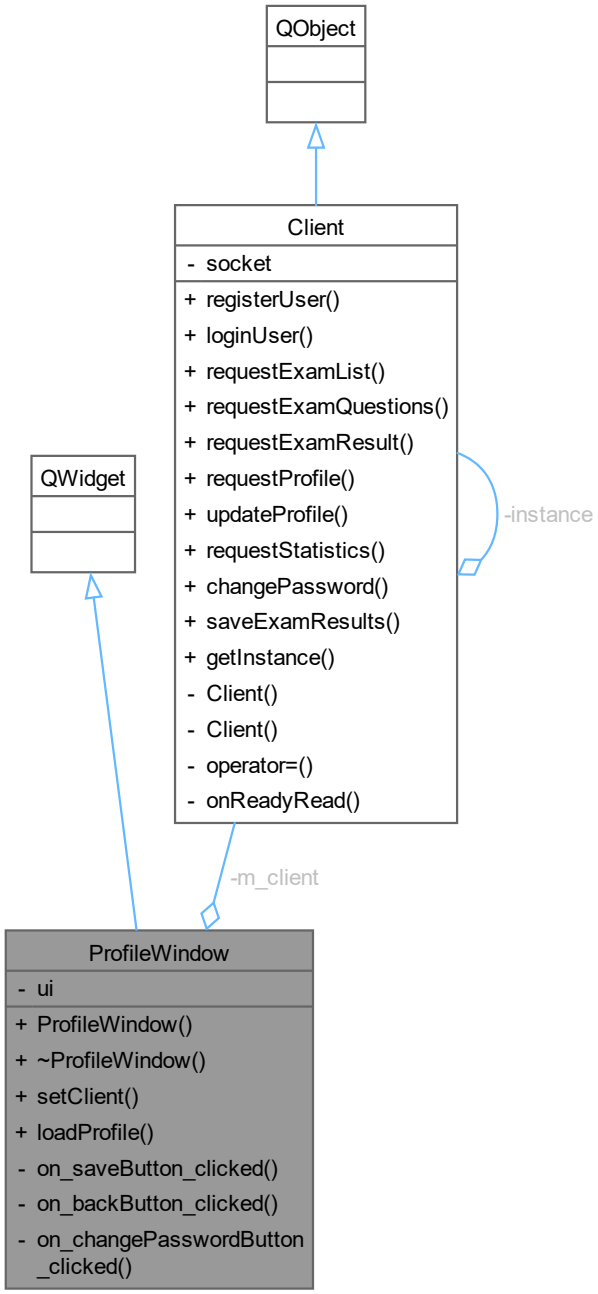
- ExamClient/MainWindow.h
- ExamClient/MainWindow.cpp

7.10 Класс ProfileWindow

Класс ProfileWindow реализует окно отображения и редактирования профиля пользователя.
#include <ProfileWindow.h>
Граф наследования:ProfileWindow:



Граф связей класса ProfileWindow:



Сигналы

- void backRequested ()

  Сигнал возврата на предыдущий экран.
- void changePasswordRequested ()

  Сигнал перехода к окну смены пароля.

Открытые члены

- ProfileWindow (QWidget *parent=nullptr)

  Конструктор окна профиля.
- ~ProfileWindow ()

  Деструктор.
- void setClient (Client *client)

  Устанавливает указатель на клиента для сетевого взаимодействия.
- void loadProfile (const QJsonObject &profile)

  Загружает данные профиля из JSON-объекта.

Закрытые слоты

- void on_saveButton_clicked ()

  Слот обработки нажатия кнопки "Сохранить".
- void on_backButton_clicked ()

  Слот обработки нажатия кнопки "Назад".
- void on_changePasswordButton_clicked ()

  Слот обработки нажатия кнопки "Сменить пароль".

Закрытые данные

- Ui::ProfileWindow * ui

  Указатель на UI форму профиля.
- Client * m_client

  Указатель на клиента для сетевого взаимодействия.

7.10.1 Подробное описание

Класс ProfileWindow реализует окно отображения и редактирования профиля пользователя. Позволяет пользователю просматривать и изменять свои персональные данные: ФИО, дату рождения и адрес электронной почты. Также предоставляет доступ к смене пароля.

7.10.2 Конструктор(ы)

7.10.2.1 ProfileWindow()

ProfileWindow::ProfileWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна профиля.
Конструктор окна профиля пользователя.
Инициализирует интерфейс и состояние.

Аргументы

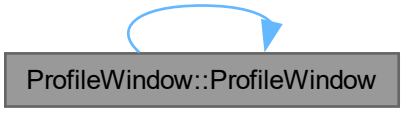
parent	Родительский виджет (по умолчанию nullptr).

Инициализирует интерфейс и сбрасывает указатель клиента.

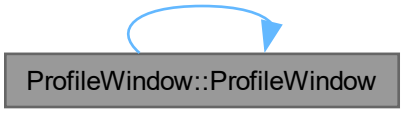
Аргументы

parent	Родительский виджет.

Граф вызовов:



Граф вызова функции:



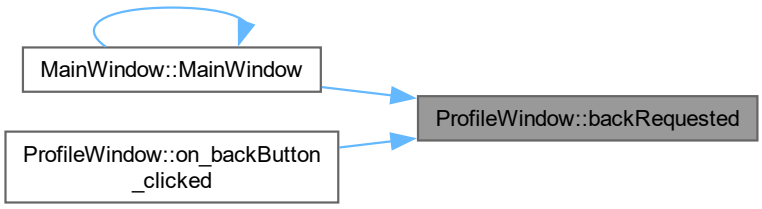
7.10.2.2 ~ProfileWindow()

ProfileWindow::~ProfileWindow ()
Деструктор.
Деструктор окна профиля.
Освобождает ресурсы, выделенные для UI.

7.10.3 Методы

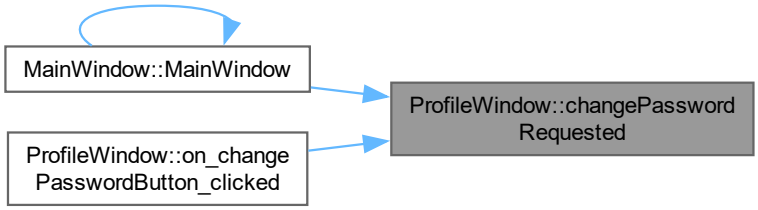
7.10.3.1 backRequested

void ProfileWindow::backRequested () [signal]
Сигнал возврата на предыдущий экран.
Граф вызова функции:



7.10.3.2 changePasswordRequested

void ProfileWindow::changePasswordRequested () [signal]
Сигнал перехода к окну смены пароля.
Граф вызова функции:



7.10.3.3 loadProfile()

void ProfileWindow::loadProfile (
const QJsonObject & profile)
Загружает данные профиля из JSON-объекта.
Загружает данные профиля в UI.
Используется при получении данных от сервера. Устанавливает значения в поля ввода на форме.

Аргументы

profile	JSON-объект с полями: full_name, birth_date, email.

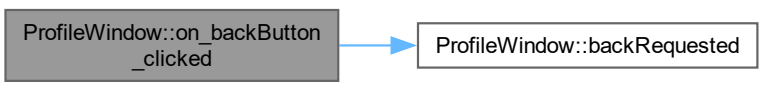
Устанавливает имя, дату рождения и email, полученные от сервера.

Аргументы

profile	JSON-объект с данными профиля.

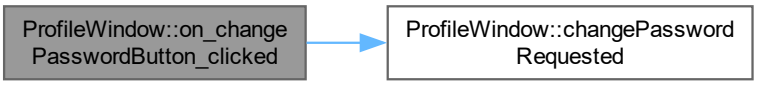
7.10.3.4 on_backButton_clicked

void ProfileWindow::on_backButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Назад".
Испускает сигнал backRequested().
Испускает сигнал для возврата к предыдущему экрану. Граф вызовов:



7.10.3.5 on_changePasswordButton_clicked

void ProfileWindow::on_changePasswordButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Сменить пароль".
Испускает сигнал changePasswordRequested().
Испускает сигнал для перехода к окну смены пароля. Граф вызовов:



7.10.3.6 on_saveButton_clicked

void ProfileWindow::on_saveButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Сохранить".
Отправляет изменения данных профиля на сервер.
Формирует JSON-объект с изменёнными данными профиля и отправляет его на сервер.

7.10.3.7 setClient()

void ProfileWindow::setClient (
Client * client)
Устанавливает указатель на клиента для сетевого взаимодействия.
Устанавливает указатель на клиент.
Необходим для отправки изменений на сервер.

Аргументы

client	Указатель на объект Client.

Используется для отправки обновлённых данных профиля на сервер.

Аргументы

client	Указатель на объект Client.

7.10.4 Поля

7.10.4.1 m_client

Client* ProfileWindow::m_client [private]
Указатель на клиента для сетевого взаимодействия.

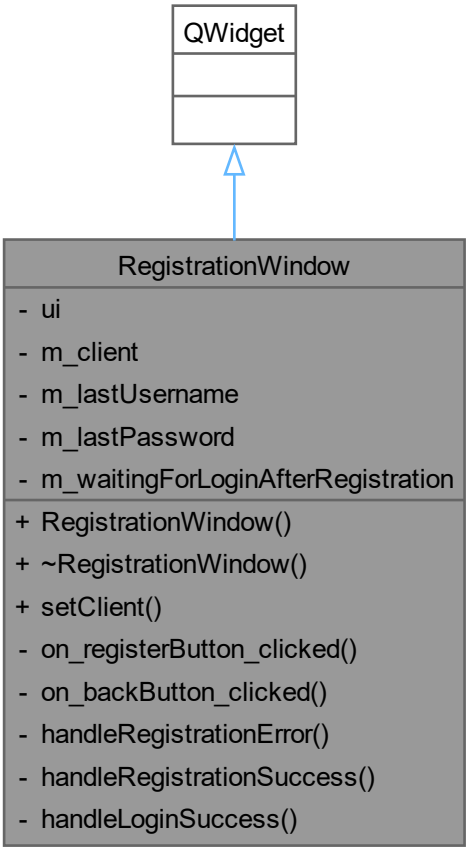
7.10.4.2 ui

Ui::ProfileWindow* ProfileWindow::ui [private]
Указатель на UI форму профиля.
Объявления и описания членов классов находятся в файлах:

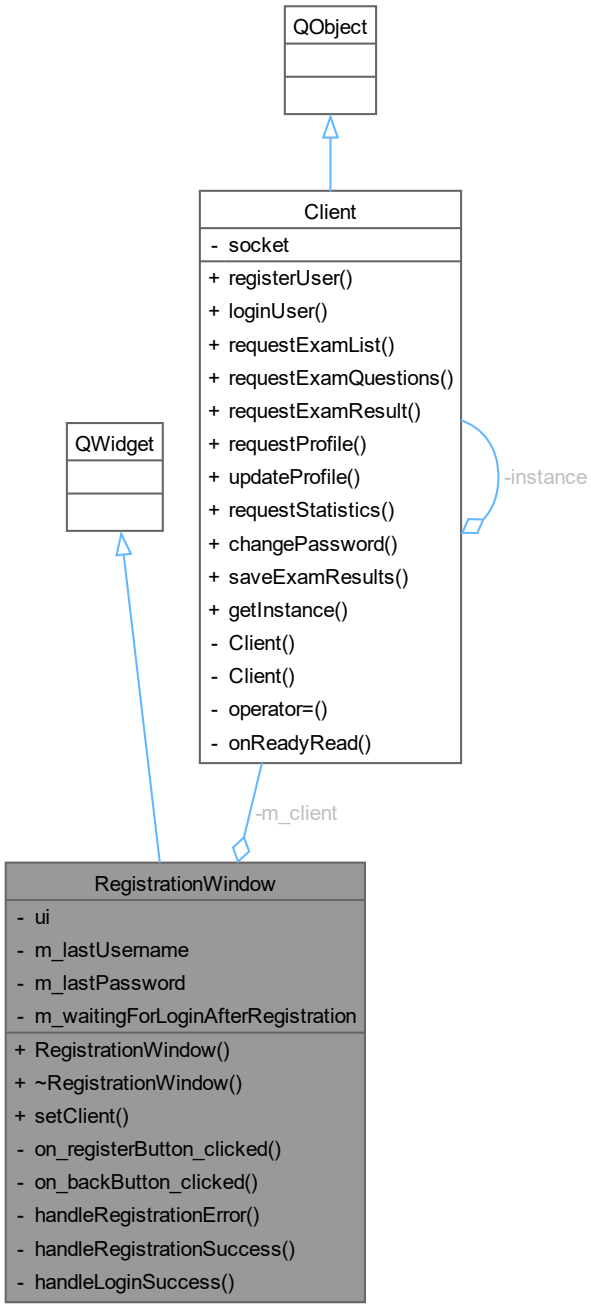
- ExamClient/ProfileWindow.h
- ExamClient/ProfileWindow.cpp

7.11 Класс RegistrationWindow

Класс RegistrationWindow реализует окно регистрации нового пользователя.
#include <RegistrationWindow.h>
Граф наследования:RegistrationWindow:



Граф связей класса RegistrationWindow:



Сигналы

- void registrationSucceeded ()

  Сигнал об успешной регистрации.
- void backRequested ()

  Сигнал возврата на экран входа.

Открытые члены

- RegistrationWindow (QWidget *parent=nullptr)

  Конструктор окна регистрации.
- ~RegistrationWindow ()

  Деструктор.
- void setClient (Client *client)

  Устанавливает клиент для регистрации и подключения сигналов.

Закрытые слоты

- void on_registerButton_clicked ()

  Обработка нажатия кнопки "Регистрация".
- void on_backButton_clicked ()

  Обработка нажатия кнопки "Назад".
- void handleRegistrationError (const QString &errorMessage)

  Обработка ошибки при регистрации.
- void handleRegistrationSuccess ()

  Обработка успешной регистрации.
- void handleLoginSuccess ()

  Обработка успешного входа сразу после регистрации.

Закрытые данные

- Ui::RegistrationWindow * ui

  Указатель на интерфейс.
- Client * m_client

  Указатель на клиента для сетевого взаимодействия.
- QString m_lastUsername

  Последнее введённое имя пользователя.
- QString m_lastPassword

  Последний введённый пароль.
- bool m_waitingForLoginAfterRegistration = false

  Флаг ожидания входа после регистрации.

7.11.1 Подробное описание

Класс RegistrationWindow реализует окно регистрации нового пользователя.
Предоставляет поля для ввода имени пользователя и пароля. Обеспечивает взаимодействие с сервером через клиент и обрабатывает события успеха/ошибки.

7.11.2 Конструктор(ы)

7.11.2.1 RegistrationWindow()

RegistrationWindow::RegistrationWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна регистрации.
Инициализирует интерфейс и внутренние переменные.

Аргументы

parent	Родительский виджет (по умолчанию nullptr).

Инициализирует интерфейс и сбрасывает указатель на клиент.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

7.11.2.2 ~RegistrationWindow()

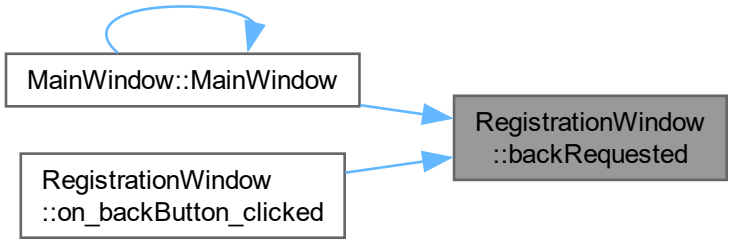
RegistrationWindow::~RegistrationWindow ()
Деструктор.
Деструктор окна регистрации.
Освобождает ресурсы, выделенные для UI.

7.11.3 Методы

7.11.3.1 backRequested

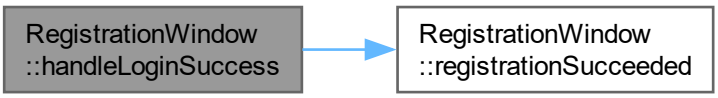
void RegistrationWindow::backRequested () [signal]
Сигнал возврата на экран входа.

Граф вызова функции:

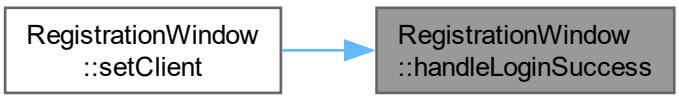


7.11.3.2 handleLoginSuccess

void RegistrationWindow::handleLoginSuccess () [private], [slot]
Обработка успешного входа сразу после регистрации.
Обработка успешного входа после регистрации.
Используется для автоматического входа после регистрации.
Завершает переход после автоматического входа пользователя. Граф вызовов:



Граф вызова функции:



7.11.3.3 handleRegistrationError

void RegistrationWindow::handleRegistrationError (
const QString & errorMessage) [private], [slot]
Обработка ошибки при регистрации.
Обработка ошибки регистрации.

Отображает сообщение об ошибке, полученное от сервера.

Аргументы

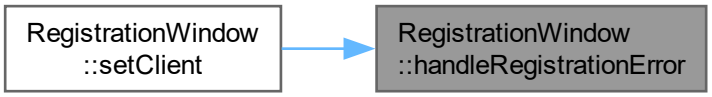
errorMessage	Текст ошибки.

Отображает сообщение об ошибке с текстом, полученным от клиента.

Аргументы

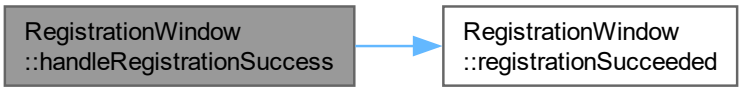
errorMessage	Сообщение об ошибке.

Граф вызова функции:

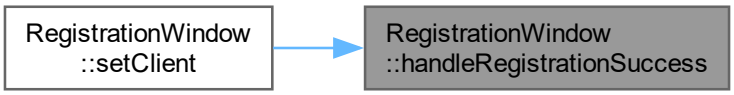


7.11.3.4 handleRegistrationSuccess

void RegistrationWindow::handleRegistrationSuccess () [private], [slot]
Обработка успешной регистрации.
Отображает сообщение и испускает сигнал перехода к следующему экрану.
Отображает уведомление и инициирует переход к следующему окну. Граф вызовов:

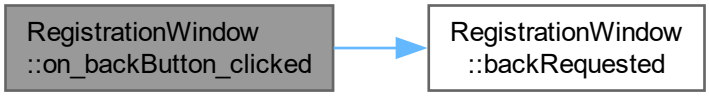


Граф вызова функции:



7.11.3.5 on_backButton_clicked

void RegistrationWindow::on_backButton_clicked () [private], [slot]
Обработка нажатия кнопки "Назад".
Слот обработки нажатия кнопки "Назад".
Возвращает на предыдущий экран.
Возвращает пользователя к предыдущему окну. Граф вызовов:

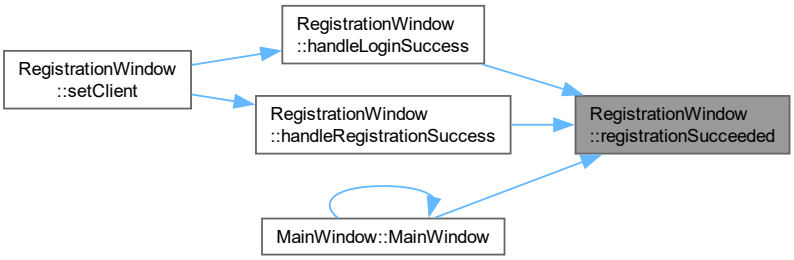


7.11.3.6 on_registerButton_clicked

void RegistrationWindow::on_registerButton_clicked () [private], [slot]
Обработка нажатия кнопки "Регистрация".
Слот, вызываемый при нажатии кнопки "Зарегистрироваться".
Считывает данные из полей ввода и отправляет их на сервер.
Отправляет введённые имя пользователя и пароль на сервер.

7.11.3.7 registrationSucceeded

void RegistrationWindow::registrationSucceeded () [signal]
Сигнал об успешной регистрации.
Используется для перехода к следующему окну после регистрации. Граф вызова функции:



7.11.3.8 setClient()

void RegistrationWindow::setClient (
Client * client)
Устанавливает клиент для регистрации и подключения сигналов.
Устанавливает клиента и подключает слоты для обработки событий.
Необходим для отправки регистрационных данных и получения ответа от сервера.

Аргументы

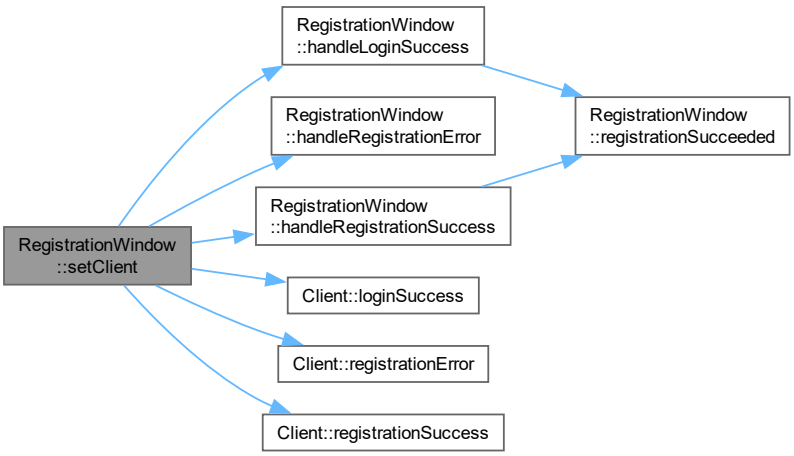
client	Указатель на объект Client.

Подключаются сигналы об ошибке и успехе регистрации, а также успешном входе в систему.

Аргументы

client	Указатель на объект Client.

Граф вызовов:



7.11.4 Поля

7.11.4.1 m_client

Client* RegistrationWindow::m_client [private]
Указатель на клиента для сетевого взаимодействия.

7.11.4.2 m_lastPassword

QString RegistrationWindow::m_lastPassword [private]
Последний введённый пароль.

7.11.4.3 m_lastUsername

QString RegistrationWindow::m_lastUsername [private]
Последнее введённое имя пользователя.

7.11.4.4 m_waitingForLoginAfterRegistration

bool RegistrationWindow::m_waitingForLoginAfterRegistration = false [private]
Флаг ожидания входа после регистрации.

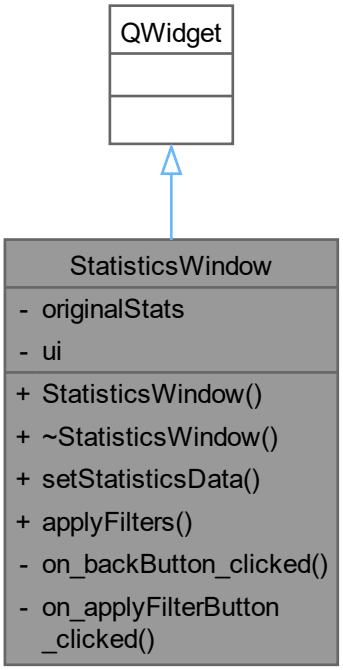
7.11.4.5 ui

Ui::RegistrationWindow* RegistrationWindow::ui [private]
Указатель на интерфейс.
Объявления и описания членов классов находятся в файлах:

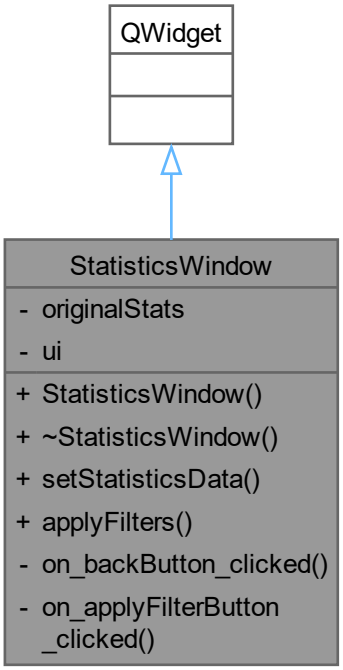
- ExamClient/RegistrationWindow.h
- ExamClient/RegistrationWindow.cpp

7.12 Класс StatisticsWindow

Класс StatisticsWindow реализует окно отображения результатов экзаменов пользователя.
#include <StatisticsWindow.h>
Граф наследования:StatisticsWindow:



Граф связей класса StatisticsWindow:



Сигналы

- void backRequested ()

  Сигнал возврата к предыдущему экрану.

Открытые члены

- StatisticsWindow (QWidget *parent=nullptr)

  Конструктор окна статистики.
- ~StatisticsWindow ()

  Деструктор.
- void setStatisticsData (const QJsonArray &stats)

  Устанавливает исходные данные по статистике экзаменов.
- void applyFilters ()

  Применяет фильтры, установленные пользователем (по дате и предмету).

Закрытые слоты

- void on_backButton_clicked ()

  Слот обработки нажатия кнопки "Назад".
- void on_applyFilterButton_clicked ()

  Слот обработки нажатия кнопки "Применить".

Закрытые данные

- QJsonArray originalStats

  Исходные необработанные данные статистики.
- Ui::StatisticsWindow * ui

  Указатель на пользовательский интерфейс окна.

7.12.1 Подробное описание

Класс StatisticsWindow реализует окно отображения результатов экзаменов пользователя. Предоставляет возможность фильтрации данных по дате и названию экзамена (предмету). Отображает отфильтрованные результаты в текстовом формате.

7.12.2 Конструктор(ы)

7.12.2.1 StatisticsWindow()

StatisticsWindow::StatisticsWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна статистики.
Инициализирует пользовательский интерфейс.

Аргументы

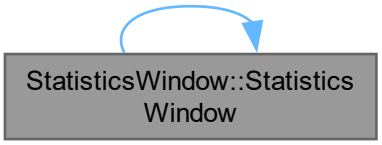
parent	Родительский виджет (по умолчанию nullptr).

Инициализирует интерфейс окна без данных.

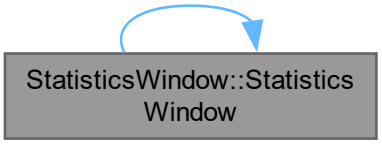
Аргументы

parent	Родительский виджет.

Граф вызовов:



Граф вызова функции:



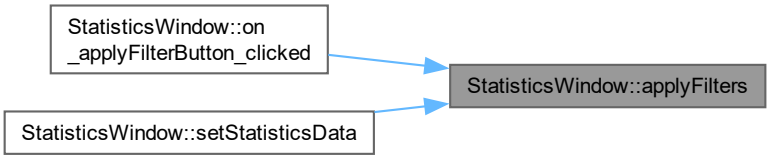
7.12.2.2 ~StatisticsWindow()

StatisticsWindow::~StatisticsWindow ()
Деструктор.
Деструктор окна статистики.
Освобождает ресурсы, выделенные для интерфейса.
Освобождает память, выделенную для интерфейса.

7.12.3 Методы

7.12.3.1 applyFilters()

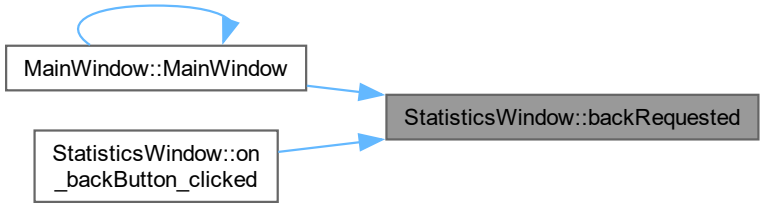
void StatisticsWindow::applyFilters ()
Применяет фильтры, установленные пользователем (по дате и предмету).
Применяет фильтры и отображает отфильтрованные результаты.
Фильтрация применяется к полю originalStats и результат отображается в QTextEdit.
Фильтрация производится по дате и названию экзамена. Отображает форматированную статистику в текстовом поле. Граф вызова функции:



7.12.3.2 backRequested

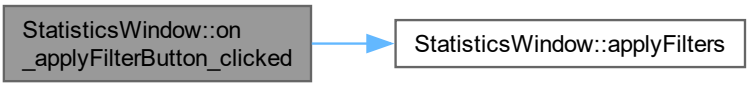
void StatisticsWindow::backRequested () [signal]
Сигнал возврата к предыдущему экрану.

Граф вызова функции:



7.12.3.3 on_applyFilterButton_clicked

void StatisticsWindow::on_applyFilterButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Применить".
Слот обработки кнопки "Применить фильтр".
Обновляет отображение результатов по текущим фильтрам.
Вызывает пересчёт фильтрации статистики. Граф вызовов:



7.12.3.4 on_backButton_clicked

void StatisticsWindow::on_backButton_clicked () [private], [slot]
Слот обработки нажатия кнопки "Назад".
Испускает сигнал backRequested().
Испускает сигнал возврата к предыдущему окну. Граф вызовов:

7.12.3.5 setStatisticsData()

void StatisticsWindow::setStatisticsData (
const QJsonArray & stats)
Устанавливает исходные данные по статистике экзаменов.

Загружает и отображает данные по статистике экзаменов.
Загружает JSON-массив с результатами экзаменов и подготавливает интерфейс к фильтрации.

Аргументы

stats	JSON-массив с объектами, содержащими данные экзамена (exam_name, score, passed_at и т.д.).

Заполняет комбобокс уникальными названиями экзаменов, устанавливает "Все" по умолчанию. После загрузки применяются текущие фильтры.

Аргументы

stats	JSON-массив с результатами экзаменов.

Граф вызовов:

7.12.4 Поля

7.12.4.1 originalStats

QJsonArray StatisticsWindow::originalStats [private]
Исходные необработанные данные статистики.

7.12.4.2 ui

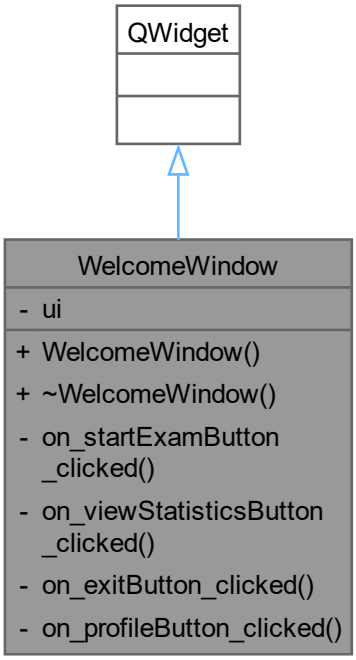
Ui::StatisticsWindow* StatisticsWindow::ui [private]
Указатель на пользовательский интерфейс окна.
Объявления и описания членов классов находятся в файлах:

- ExamClient/StatisticsWindow.h
- ExamClient/StatisticsWindow.cpp

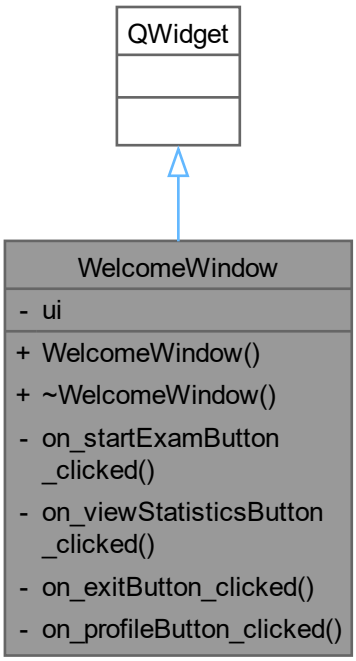
7.13 Класс WelcomeWindow

Класс WelcomeWindow представляет приветственное окно после входа пользователя.
#include <WelcomeWindow.h>

Граф наследования:WelcomeWindow:



Граф связей класса WelcomeWindow:



Сигналы

- void startExamRequested ()

  Сигнал, испускаемый при нажатии кнопки "Начать экзамен".
- void viewStatisticsRequested ()

  Сигнал, испускаемый при нажатии кнопки "Статистика".
- void exitRequested ()

  Сигнал, испускаемый при нажатии кнопки "Выход".
- void profileRequested ()

  Сигнал, испускаемый при нажатии кнопки "Профиль".

Открытые члены

- WelcomeWindow (QWidget *parent=nullptr)

  Конструктор окна WelcomeWindow.
- ~WelcomeWindow ()

  Деструктор.

Закрытые слоты

- void on_startExamButton_clicked ()

  Обработка нажатия кнопки "Начать экзамен".
- void on_viewStatisticsButton_clicked ()

  Обработка нажатия кнопки "Статистика".
- void on_exitButton_clicked ()

  Обработка нажатия кнопки "Выход".

- void on_profileButton_clicked ()

  Обработка нажатия кнопки "Профиль".

Закрытые данные

- Ui::WelcomeWindow * ui

  Указатель на пользовательский интерфейс окна.

7.13.1 Подробное описание

Класс WelcomeWindow представляет приветственное окно после входа пользователя.
Пользователь может:

- начать экзамен;
- просмотреть статистику;
- открыть профиль;
- выйти из системы.

7.13.2 Конструктор(ы)

7.13.2.1 WelcomeWindow()

WelcomeWindow::WelcomeWindow (
QWidget * parent = nullptr) [explicit]

Конструктор окна WelcomeWindow.
Конструктор окна приветствия.
Инициализирует пользовательский интерфейс.

Аргументы

parent	Родительский виджет (по умолчанию nullptr).

Инициализирует интерфейс главного меню после входа пользователя.

Аргументы

parent	Родительский виджет.

Граф вызовов:

Граф вызова функции:

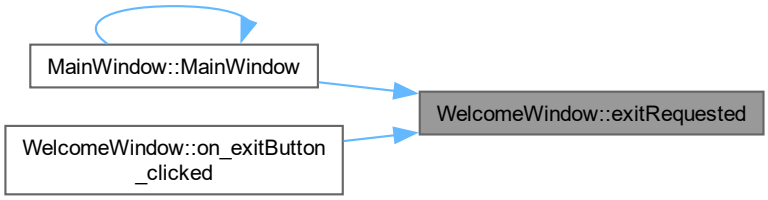
7.13.2.2 ~WelcomeWindow()

WelcomeWindow::~WelcomeWindow ()
Деструктор.
Деструктор окна приветствия.
Освобождает ресурсы интерфейса.

7.13.3 Методы

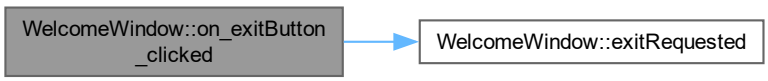
7.13.3.1 exitRequested

void WelcomeWindow::exitRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Выход".
Завершает сессию пользователя. Граф вызова функции:



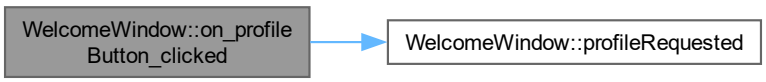
7.13.3.2 on_exitButton_clicked

void WelcomeWindow::on_exitButton_clicked () [private], [slot]
Обработка нажатия кнопки "Выход".
Слот обработки нажатия кнопки "Выход".
Испускает сигнал завершения работы приложения. Граф вызовов:



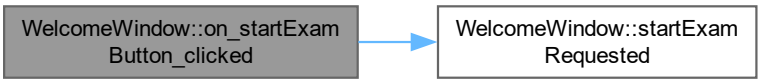
7.13.3.3 on_profileButton_clicked

void WelcomeWindow::on_profileButton_clicked () [private], [slot]
Обработка нажатия кнопки "Профиль".
Слот обработки нажатия кнопки "Профиль".
Испускает сигнал перехода к окну профиля пользователя. Граф вызовов:



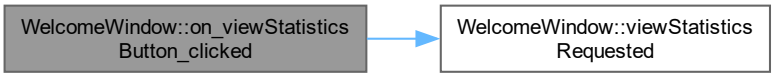
7.13.3.4 on_startExamButton_clicked

void WelcomeWindow::on_startExamButton_clicked () [private], [slot]
Обработка нажатия кнопки "Начать экзамен".
Слот обработки нажатия кнопки "Начать экзамен".
Испускает сигнал для перехода к выбору экзамена. Граф вызовов:



7.13.3.5 on_viewStatisticsButton_clicked

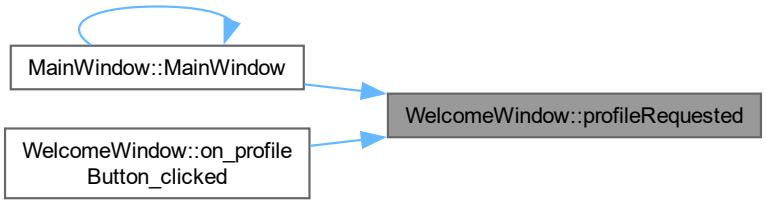
void WelcomeWindow::on_viewStatisticsButton_clicked () [private], [slot]
Обработка нажатия кнопки "Статистика".
Слот обработки нажатия кнопки "Статистика".
Испускает сигнал для перехода к окну статистики. Граф вызовов:



7.13.3.6 profileRequested

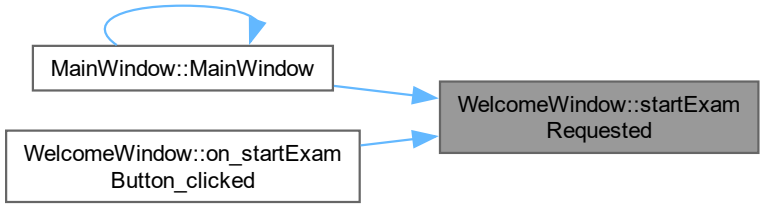
void WelcomeWindow::profileRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Профиль".

Открывает окно редактирования профиля. Граф вызова функции:



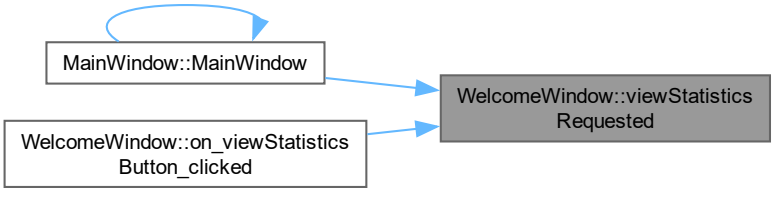
7.13.3.7 startExamRequested

void WelcomeWindow::startExamRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Начать экзамен".
Используется для перехода к экрану выбора экзамена. Граф вызова функции:



7.13.3.8 viewStatisticsRequested

void WelcomeWindow::viewStatisticsRequested () [signal]
Сигнал, испускаемый при нажатии кнопки "Статистика".
Используется для отображения результатов экзаменов. Граф вызова функции:



7.13.4 Поля

7.13.4.1 ui

Ui::WelcomeWindow* WelcomeWindow::ui [private]
Указатель на пользовательский интерфейс окна.
Объявления и описания членов классов находятся в файлах:

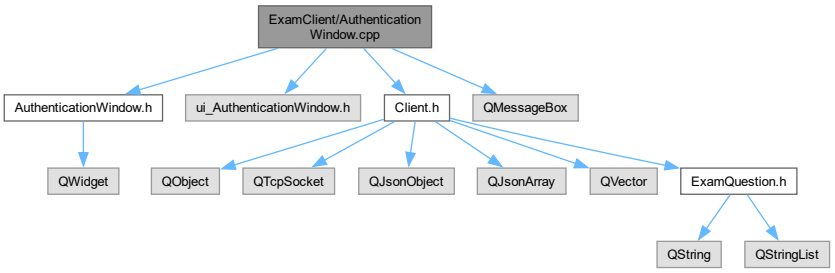
- ExamClient/WelcomeWindow.h
- ExamClient/WelcomeWindow.cpp

Глава 8

Файлы

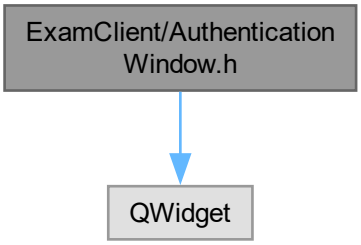
8.1 Файл ExamClient/AuthenticationWindow.cpp

#include "AuthenticationWindow.h"
#include "ui_AuthenticationWindow.h"
#include "Client.h"
#include <QMessageBox>
Граф включаемых заголовочных файлов для AuthenticationWindow.cpp:

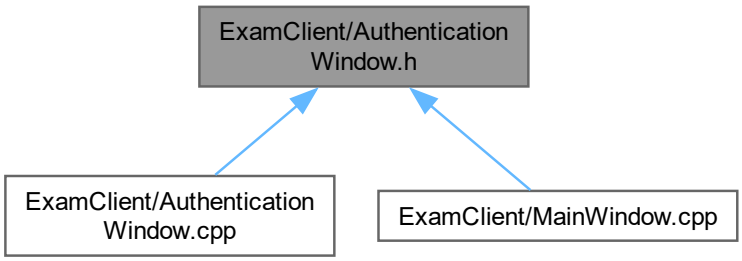


8.2 Файл ExamClient/AuthenticationWindow.h

#include <QWidget>
Граф включаемых заголовочных файлов для AuthenticationWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class AuthenticationWindow

  Класс AuthenticationWindow реализует окно аутентификации пользователя.

Пространства имен

- namespace Ui

8.3 AuthenticationWindow.h

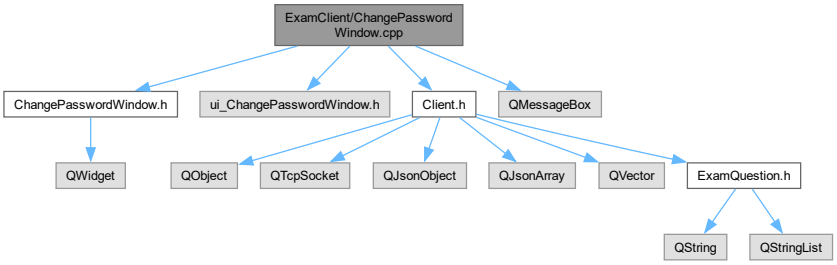
См. документацию.

```
00001 #ifndef AUTHENTICATIONWINDOW_H
00002 #define AUTHENTICATIONWINDOW_H
00003
00004 #include <QWidget>
00005
00006 namespace Ui {
00007 class AuthenticationWindow;
00008 }
00009
00010 class Client;
00011
00019 class AuthenticationWindow : public QWidget
00020 {
00021     Q_OBJECT
00022
00023 public:
00031     explicit AuthenticationWindow(QWidget *parent = nullptr);
00032
00038     ~AuthenticationWindow();
00039
00047     void setClient(Client *client);
00048
00049 signals:
00055     void authenticationSucceeded();
00056
00062     void backRequested();
00063
00064 private slots:
00070     void on_nextButton_clicked();
00071
00077     void on_backButton_clicked();
00078
00086     void handleLoginError(const QString &errorMessage);
00087
00093     void handleLoginSuccess();
00094
00095 private:
00096     Ui::AuthenticationWindow *ui;
00097     Client *m_client;
00098 };
00099
00100 #endif // AUTHENTICATIONWINDOW_H
```

8.4 Файл ExamClient/ChangePasswordWindow.cpp

```
#include "ChangePasswordWindow.h"
#include "ui_ChangePasswordWindow.h"
#include "Client.h"
#include <QMessageBox>
```

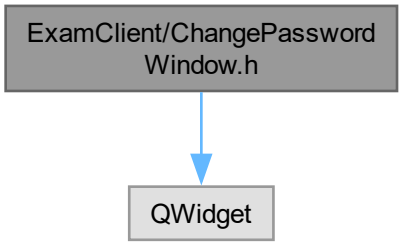
Граф включаемых заголовочных файлов для ChangePasswordWindow.cpp:



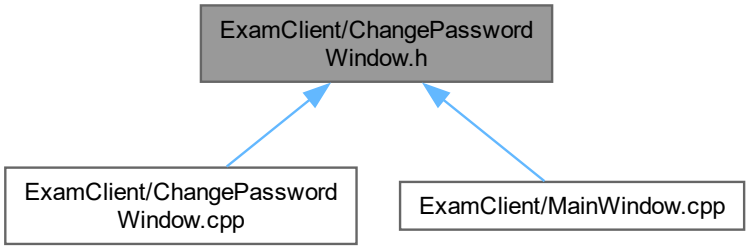
8.5 Файл ExamClient/ChangePasswordWindow.h

```
#include <QWidget>
```

Граф включаемых заголовочных файлов для ChangePasswordWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class ChangePasswordWindow

  Класс ChangePasswordWindow предоставляет окно смены пароля пользователя.

Пространства имен

- namespace Ui

8.6 ChangePasswordWindow.h

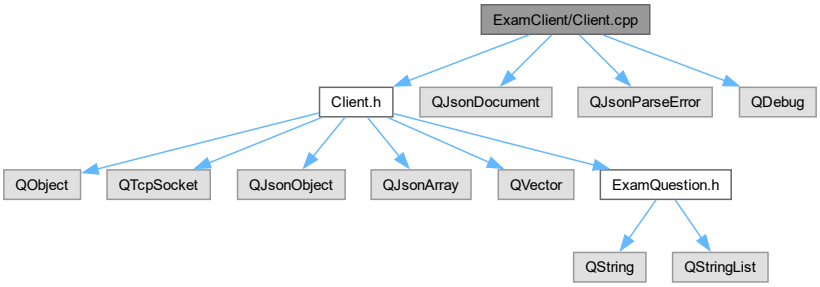
См. документацию.

```
00001 #ifndef CHANGEPASSWORDWINDOW_H
00002 #define CHANGEPASSWORDWINDOW_H
00003
00004 #include <QWidget>
00005
00006 class Client;
00007
00008 namespace Ui {
00009 class ChangePasswordWindow;
00010 }
00011
00018 class ChangePasswordWindow : public QWidget
00019 {
00020     Q_OBJECT
00021
00022 public:
00030     explicit ChangePasswordWindow(QWidget *parent = nullptr);
00031
00037     ~ChangePasswordWindow();
00038
00046     void setClient(Client *client);
00047
00048 signals:
00054     void backRequested();
00055
00056 private slots:
00062     void on_applyPasswordButton_clicked();
00063
00069     void on_backButton_clicked();
00070
00071 private:
00072     Ui::ChangePasswordWindow *ui;
00073     Client *m_client;
00074 };
00075
00076 #endif // CHANGEPASSWORDWINDOW_H
```

8.7 Файл ExamClient/Client.cpp

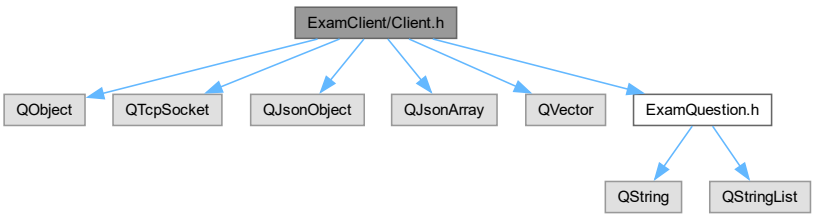
```
#include "Client.h"
#include <QJsonDocument>
#include <QJsonParseError>
```

#include <QDebug>
Граф включаемых заголовочных файлов для Client.cpp:

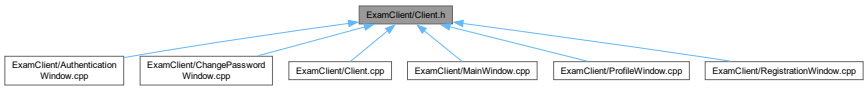


8.8 Файл ExamClient/Client.h

#include <QObject>
#include <QTcpSocket>
#include <QJsonObject>
#include <QJsonArray>
#include <QVector>
#include "ExamQuestion.h"
Граф включаемых заголовочных файлов для Client.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class Client

  Класс Client — синглтон, обеспечивающий TCP-соединение с сервером.

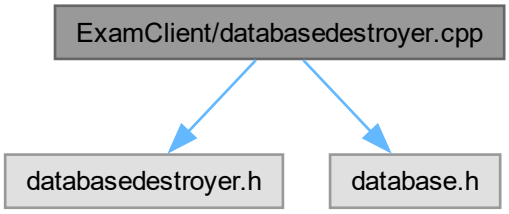
8.9 Client.h

См. документацию.
00001 #ifndef CLIENT_H

```
00002 #define CLIENT_H
00003
00004 #include <QObject>
00005 #include <QTcpSocket>
00006 #include <QJsonObject>
00007 #include <QJsonArray>
00008 #include <QVector>
00009 #include "ExamQuestion.h"
00010
00017 class Client : public QObject
00018 {
00019     Q_OBJECT
00020
00021 public:
00029     static Client* getInstance();
00030
00037     void registerUser(const QString &username, const QString &password);
00038
00045     void loginUser(const QString &username, const QString &password);
00046
00050     void requestExamList();
00051
00057     void requestExamQuestions(int examId);
00058
00064     void requestExamResult(int examId);
00065
00069     void requestProfile();
00070
00076     void updateProfile(const QJsonObject &profile);
00077
00081     void requestStatistics();
00082
00089     void changePassword(const QString &oldPassword, const QString &newPassword);
00090
00099     void saveExamResults(int examId, int score,
00100                          const QVector<ExamQuestion> &questions,
00101                          const QVector<QString> &userAnswers);
00102
00103 signals:
00109     void examQuestionsReceived(const QVector<ExamQuestion> &questions);
00110
00116     void examListReceived(const QJsonArray &examList);
00117
00123     void examResultReceived(const QJsonObject &result);
00124
00130     void loginError(const QString &errorMessage);
00131
00135     void loginSuccess();
00136
00142     void registrationError(const QString &msg);
00143
00147     void registrationSuccess();
00148
00154     void statisticsReceived(const QJsonArray &stats);
00155
00163     void examFinished(int score,
00164                       const QVector<ExamQuestion> &questions,
00165                       const QVector<int> &userAnswers);
00166
00172     void profileReceived(const QJsonObject &profile);
00173
00174 private slots:
00178     void onReadyRead();
00179
00180 private:
00185     explicit Client(QObject *parent = nullptr);
00186
00187     // Запрет копирования и присваивания
00188     Client(const Client&) = delete;
00189     Client& operator=(const Client&) = delete;
00190
00191     QTcpSocket *socket;
00192     static Client* instance;
00193 };
00194
00195 #endif // CLIENT_H
```

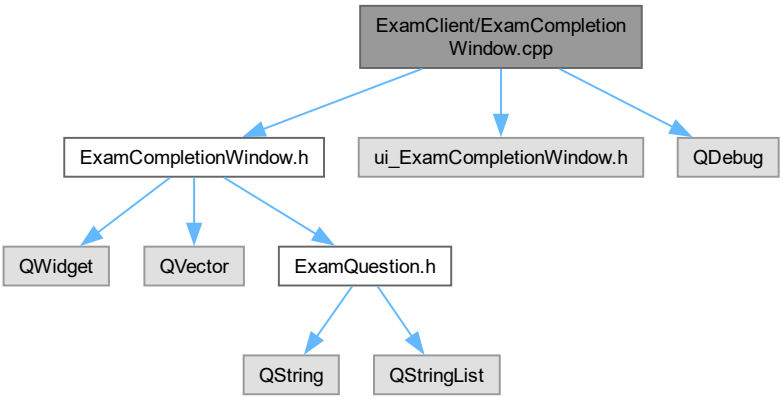
8.10 Файл ExamClient/databasedestroyer.cpp

#include "databasedestroyer.h"
#include "database.h"
Граф включаемых заголовочных файлов для databasedestroyer.cpp:



8.11 Файл ExamClient/ExamCompletionWindow.cpp

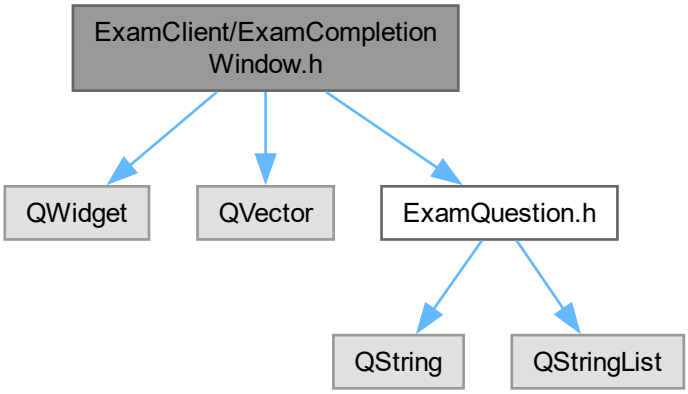
#include "ExamCompletionWindow.h"
#include "ui_ExamCompletionWindow.h"
#include <QDebug>
Граф включаемых заголовочных файлов для ExamCompletionWindow.cpp:



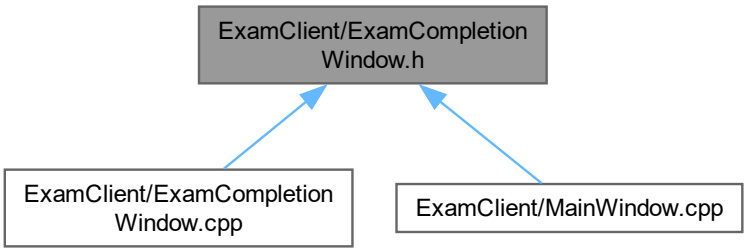
8.12 Файл ExamClient/ExamCompletionWindow.h

#include <QWidget>
#include <QVector>

#include "ExamQuestion.h"
Граф включаемых заголовочных файлов для ExamCompletionWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class ExamCompletionWindow

  Класс ExamCompletionWindow реализует окно отображения результатов экзамена.

Пространства имен

- namespace Ui

8.13 ExamCompletionWindow.h

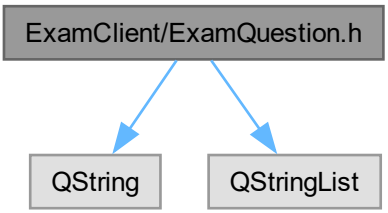
См. документацию.

```
00001 #ifndef EXAMCOMPLETIONWINDOW_H
00002 #define EXAMCOMPLETIONWINDOW_H
00003
00004 #include <QWidget>
00005 #include <QVector>
00006 #include "ExamQuestion.h"
00007
```

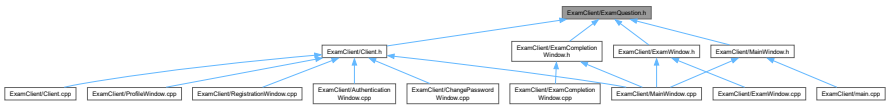
```
00008 namespace Ui {
00009 class ExamCompletionWindow;
00010 }
00011
00020 class ExamCompletionWindow : public QWidget
00021 {
00022     Q_OBJECT
00023
00024 public:
00032     explicit ExamCompletionWindow(QWidget *parent = nullptr);
00033
00039     ~ExamCompletionWindow();
00040
00051     void setResultData(int score,
00052                        const QVector<ExamQuestion> &questions,
00053                        const QVector<QString> &userAnswers);
00054
00055 signals:
00061     void exitToMenuRequested();
00062
00063 private slots:
00069     void on_exitButton_clicked();
00070
00071 private:
00072     Ui::ExamCompletionWindow *ui;
00073 };
00074
00075 #endif // EXAMCOMPLETIONWINDOW_H
```

8.14 Файл ExamClient/ExamQuestion.h

#include <QString>
#include <QStringList>
Граф включаемых заголовочных файлов для ExamQuestion.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- struct ExamQuestion

  Структура ExamQuestion представляет собой модель одного вопроса экзамена.

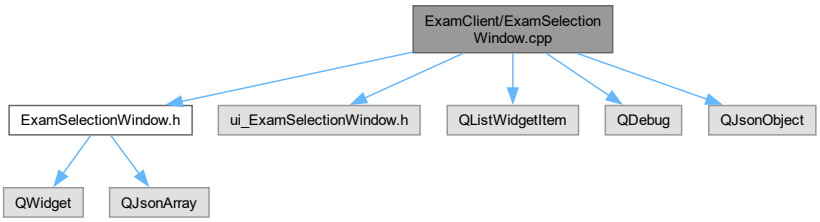
8.15 ExamQuestion.h

См. документацию.

```
00001 #ifndef EXAMQUESTION_H
00002 #define EXAMQUESTION_H
00003
00004 #include <QString>
00005 #include <QStringList>
00006
00013 struct ExamQuestion {
00014     QString questionText;
00015     QStringList options;
00016     QString correctAnswerText;
00017 };
00018
00019 #endif // EXAMQUESTION_H
```

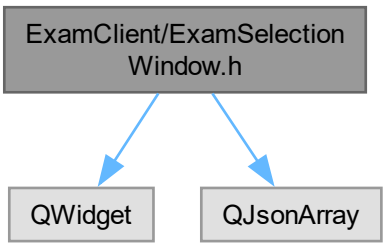
8.16 Файл ExamClient/ExamSelectionWindow.cpp

#include "ExamSelectionWindow.h"
#include "ui_ExamSelectionWindow.h"
#include <QListWidgetItem>
#include <QDebug>
#include <QJsonObject>
Граф включаемых заголовочных файлов для ExamSelectionWindow.cpp:

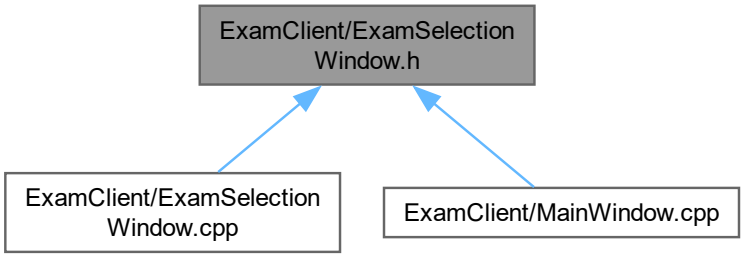


8.17 Файл ExamClient/ExamSelectionWindow.h

#include <QWidget>
#include <QJsonArray>
Граф включаемых заголовочных файлов для ExamSelectionWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class ExamSelectionWindow

  Класс ExamSelectionWindow реализует окно выбора экзамена.

Пространства имен

- namespace Ui

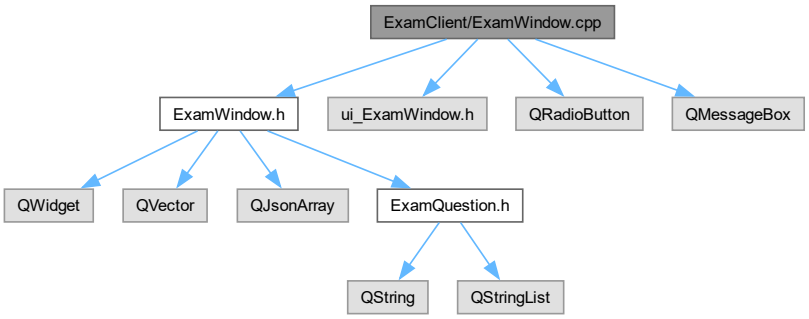
8.18 ExamSelectionWindow.h

См. документацию.

```
00001 #ifndef EXAMSELECTIONWINDOW_H
00002 #define EXAMSELECTIONWINDOW_H
00003
00004 #include <QWidget>
00005 #include <QJsonArray>
00006
00007 namespace Ui {
00008 class ExamSelectionWindow;
00009 }
00010
00017 class ExamSelectionWindow : public QWidget
00018 {
00019     Q_OBJECT
00020
00021 public:
00029     explicit ExamSelectionWindow(QWidget *parent = nullptr);
00030
00036     ~ExamSelectionWindow();
00037
00046     void loadExamList(const QJsonArray &examList);
00047
00048 signals:
00056     void examSelected(int examId);
00057
00061     void backRequested();
00062
00063 private slots:
00069     void on_selectExamButton_clicked();
00070
00076     void on_backButton_clicked();
00077
00078 private:
00079     Ui::ExamSelectionWindow *ui;
00080 };
00081
00082 #endif // EXAMSELECTIONWINDOW_H
```

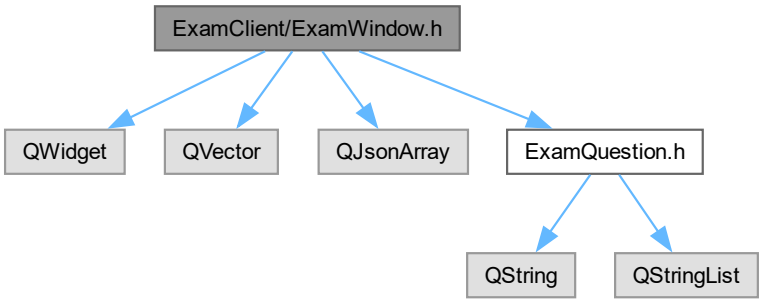
8.19 Файл ExamClient/ExamWindow.cpp

#include "ExamWindow.h"
#include "ui_ExamWindow.h"
#include <QRadioButton>
#include <QMessageBox>
Граф включаемых заголовочных файлов для ExamWindow.cpp:

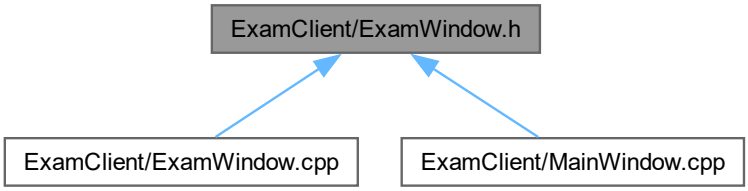


8.20 Файл ExamClient/ExamWindow.h

#include <QWidget>
#include <QVector>
#include <QJsonArray>
#include "ExamQuestion.h"
Граф включаемых заголовочных файлов для ExamWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class ExamWindow

  Класс ExamWindow предоставляет интерфейс для прохождения экзамена.

Пространства имен

- namespace Ui

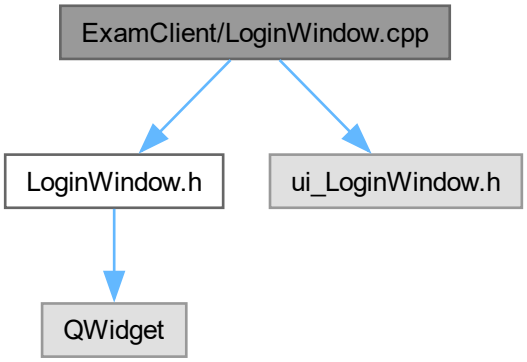
8.21 ExamWindow.h

См. документацию.

```
00001 #ifndef EXAMWINDOW_H
00002 #define EXAMWINDOW_H
00003
00004 #include <QWidget>
00005 #include <QVector>
00006 #include <QJsonArray>
00007 #include "ExamQuestion.h"
00008
00009 namespace Ui {
00010 class ExamWindow;
00011 }
00012
00019 class ExamWindow : public QWidget
00020 {
00021     Q_OBJECT
00022
00023 public:
00031     explicit ExamWindow(QWidget *parent = nullptr);
00032
00038     ~ExamWindow();
00039
00049     void setExamQuestions(int examId, const QVector<ExamQuestion> &questions);
00050
00051 signals:
00062     void examFinished(int examId,
00063                       int score,
00064                       const QVector<ExamQuestion> &questions,
00065                       const QVector<QString> &userAnswers);
00066
00067 private slots:
00074     void on_pushButton_clicked();
00075
00076 private:
00077     Ui::ExamWindow *ui;
00078
00079     QVector<ExamQuestion> m_questions;
00080     int m_currentIndex;
00081     int m_score;
00082     QVector<int> m_userAnswers;
00083     int m_examId = -1;
00084
00090     void displayCurrentQuestion();
00091 };
00092
00093 #endif // EXAMWINDOW_H
```

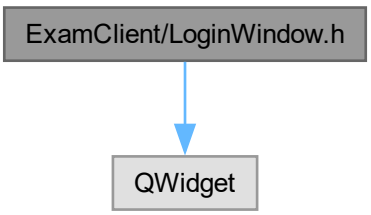
8.22 Файл ExamClient/LoginWindow.cpp

#include "LoginWindow.h"
#include "ui_LoginWindow.h"
Граф включаемых заголовочных файлов для LoginWindow.cpp:

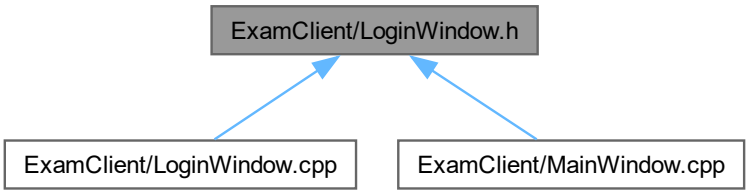


8.23 Файл ExamClient/LoginWindow.h

#include <QWidget>
Граф включаемых заголовочных файлов для LoginWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class LoginWindow

  Класс LoginWindow реализует окно начального входа пользователя.

Пространства имен

- namespace Ui

8.24 LoginWindow.h

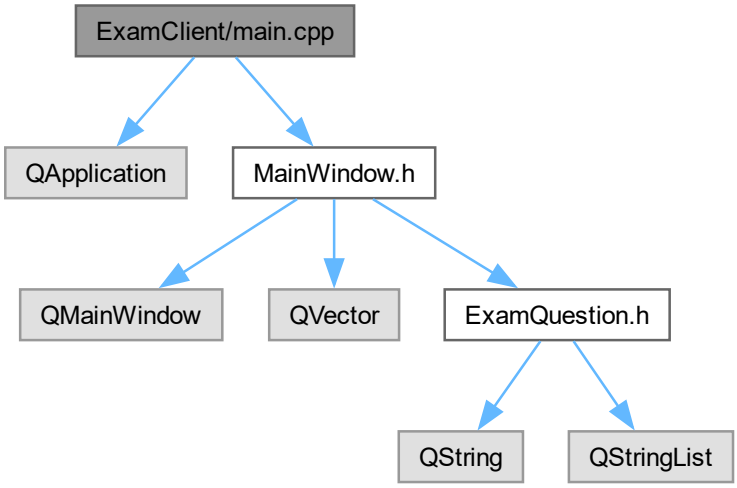
См. документацию.

```
00001 #ifndef LOGINWINDOW_H
00002 #define LOGINWINDOW_H
00003
00004 #include <QWidget>
00005
00006 namespace Ui {
00007 class LoginWindow;
00008 }
00009
00016 class LoginWindow : public QWidget
00017 {
00018     Q_OBJECT
00019
00020 public:
00028     explicit LoginWindow(QWidget *parent = nullptr);
00029
00035     ~LoginWindow();
00036
00037 signals:
00043     void registrationRequested();
00044
00050     void loginRequested();
00051
00052 private slots:
00058     void on_registerButton_clicked();
00059
00065     void on_loginButton_clicked();
00066
00067 private:
00068     Ui::LoginWindow *ui;
00069 };
00070
00071 #endif // LOGINWINDOW_H
```

8.25 Файл ExamClient/main.cpp

Точка входа в приложение ExamSem.

#include <QApplication>
#include "MainWindow.h"
Граф включаемых заголовочных файлов для main.cpp:



Функции

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

  Главная функция, запускающая Qt-приложение.

8.25.1 Подробное описание

Точка входа в приложение ExamSem.

8.25.2 Функции

8.25.2.1 main()

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

Главная функция, запускающая Qt-приложение.

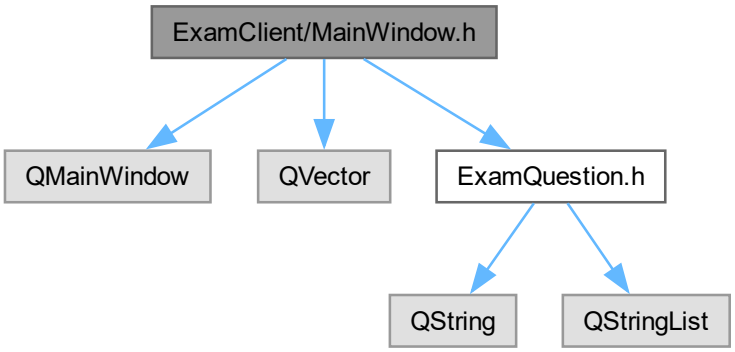
8.26 Файл ExamClient/MainWindow.cpp

#include "MainWindow.h"
#include "ui_MainWindow.h"
#include "LoginWindow.h"
#include "AuthenticationWindow.h"
#include "RegistrationWindow.h"
#include "WelcomeWindow.h"
#include "ExamSelectionWindow.h"
#include "ExamWindow.h"
#include "ExamCompletionWindow.h"
#include "StatisticsWindow.h"
#include "Client.h"

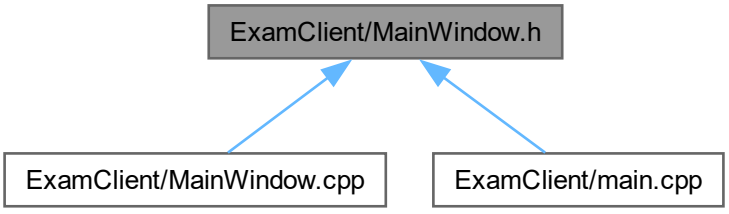
#include "ProfileWindow.h"
#include "ChangePasswordWindow.h"
Граф включаемых заголовочных файлов для MainWindow.cpp:

8.27 Файл ExamClient/MainWindow.h

#include <QMainWindow>
#include <QVector>
#include "ExamQuestion.h"
Граф включаемых заголовочных файлов для MainWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class MainWindow

  Класс MainWindow управляет всеми экранами и логикой переходов в приложении.

Пространства имен

- namespace Ui

8.28 MainWindow.h

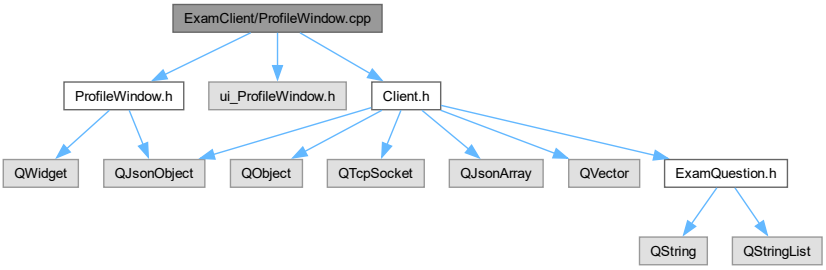
См. документацию.

```
00001 #ifndef MAINWINDOW_H
00002 #define MAINWINDOW_H
00003
00004 #include <QMainWindow>
00005 #include <QVector>
00006 #include "ExamQuestion.h"
00007
00008 class LoginWindow;
00009 class AuthenticationWindow;
00010 class RegistrationWindow;
00011 class WelcomeWindow;
00012 class ExamSelectionWindow;
00013 class ExamWindow;
00014 class ExamCompletionWindow;
00015 class StatisticsWindow;
00016 class Client;
00017 class ChangePasswordWindow;
00018 class ProfileWindow;
00019
00020 QT_BEGIN_NAMESPACE
00021 namespace Ui { class MainWindow; }
00022 QT_END_NAMESPACE
00023
00030 class MainWindow : public QMainWindow
00031 {
00032     Q_OBJECT
00033
00034 public:
00042     explicit MainWindow(QWidget *parent = nullptr);
00043
00049     ~MainWindow();
00050
00051 private slots:
00055     void showLogin();
00056
00060     void showAuthentication();
00061
00065     void showRegistration();
00066
00070     void showWelcome();
00071
00075     void showExamSelection();
00076
00082     void showExamWindow(int examId);
00083
00087     void showExamCompletion();
00088
00092     void showStatistics();
00093
00097     void showProfile();
00098
00102     void showChangePasswordWindow();
00103
00114     void onExamFinished(int examId,
00115                         int score,
00116                         const QVector<ExamQuestion> &questions,
00117                         const QVector<QString> &userAnswers);
00118
00119 private:
00120     Ui::MainWindow *ui;
00121
00122     // Все окна интерфейса
00123     LoginWindow          *loginWindow;
00124     AuthenticationWindow *authenticationWindow;
00125     RegistrationWindow    *registrationWindow;
00126     WelcomeWindow         *welcomeWindow;
00127     ExamSelectionWindow   *examSelectionWindow;
00128     ExamWindow            *examWindow;
00129     ExamCompletionWindow *examCompletionWindow;
00130     StatisticsWindow      *statisticsWindow;
00131     ProfileWindow         *profileWindow;
00132     ChangePasswordWindow *changePasswordWindow;
00133
00134     Client *client;
00135     int currentExamId = -1;
00136 };
00137
```

00138 #endif // MAINWINDOW_H

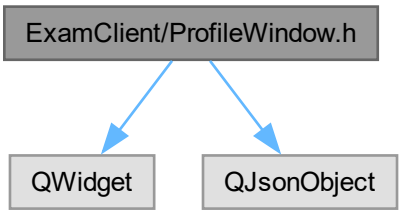
8.29 Файл ExamClient/ProfileWindow.cpp

#include "ProfileWindow.h"
#include "ui_ProfileWindow.h"
#include "Client.h"
Граф включаемых заголовочных файлов для ProfileWindow.cpp:

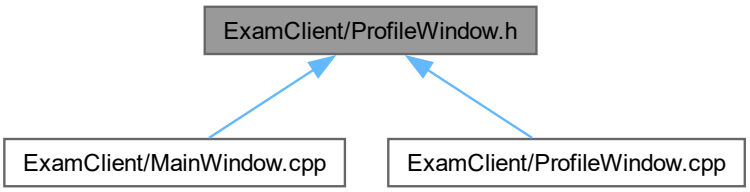


8.30 Файл ExamClient/ProfileWindow.h

#include <QWidget>
#include <QJsonObject>
Граф включаемых заголовочных файлов для ProfileWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class ProfileWindow

  Класс ProfileWindow реализует окно отображения и редактирования профиля пользователя.

Пространства имен

- namespace Ui

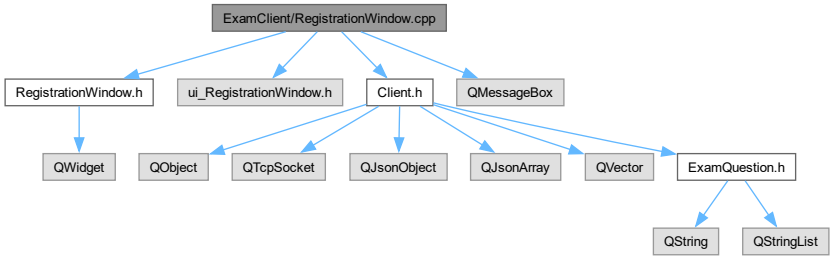
8.31 ProfileWindow.h

См. документацию.

```
00001 #ifndef PROFILEWINDOW_H
00002 #define PROFILEWINDOW_H
00003
00004 #include <QWidget>
00005 #include <QJsonObject>
00006
00007 class Client;
00008
00009 namespace Ui {
00010 class ProfileWindow;
00011 }
00012
00019 class ProfileWindow : public QWidget
00020 {
00021     Q_OBJECT
00022
00023 public:
00031     explicit ProfileWindow(QWidget *parent = nullptr);
00032
00038     ~ProfileWindow();
00039
00047     void setClient(Client *client);
00048
00057     void loadProfile(const QJsonObject &profile);
00058
00059 signals:
00063     void backRequested();
00064
00068     void changePasswordRequested();
00069
00070 private slots:
00076     void on_saveButton_clicked();
00077
00083     void on_backButton_clicked();
00084
00090     void on_changePasswordButton_clicked();
00091
00092 private:
00093     Ui::ProfileWindow *ui;
00094     Client *m_client;
00095 };
00096
00097 #endif // PROFILEWINDOW_H
```

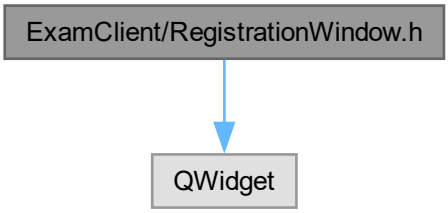
8.32 Файл ExamClient/RegistrationWindow.cpp

#include "RegistrationWindow.h"
#include "ui_RegistrationWindow.h"
#include "Client.h"
#include <QMessageBox>
Граф включаемых заголовочных файлов для RegistrationWindow.cpp:

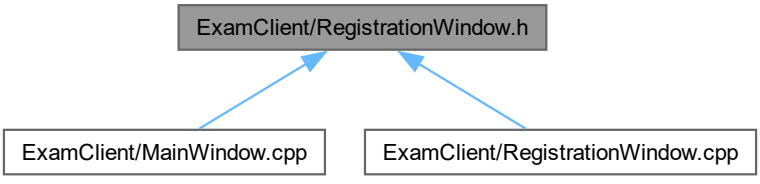


8.33 Файл ExamClient/RegistrationWindow.h

#include <QWidget>
Граф включаемых заголовочных файлов для RegistrationWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class RegistrationWindow

Класс RegistrationWindow реализует окно регистрации нового пользователя.

Пространства имен

- namespace Ui

8.34 RegistrationWindow.h

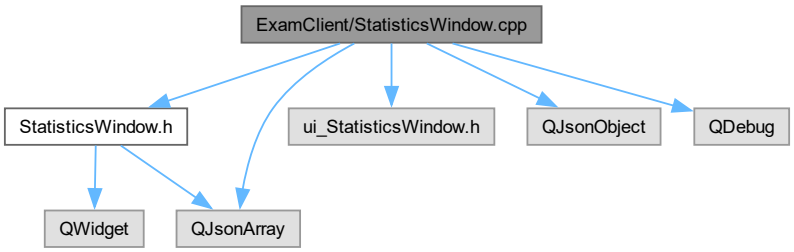
См. документацию.

```
00001 #ifndef REGISTRATIONWINDOW_H
00002 #define REGISTRATIONWINDOW_H
00003
00004 #include <QWidget>
00005
00006 namespace Ui {
00007 class RegistrationWindow;
00008 }
00009
00010 class Client;
00011
00018 class RegistrationWindow : public QWidget
00019 {
00020     Q_OBJECT
00021
00022 public:
00030     explicit RegistrationWindow(QWidget *parent = nullptr);
00031
00037     ~RegistrationWindow();
00038
00046     void setClient(Client *client);
00047
00048 signals:
00054     void registrationSucceeded();
00055
00059     void backRequested();
00060
00061 private slots:
00067     void on_registerButton_clicked();
00068
00074     void on_backButton_clicked();
00075
00083     void handleRegistrationError(const QString &errorMessage);
00084
00090     void handleRegistrationSuccess();
00091
00097     void handleLoginSuccess();
00098
00099 private:
00100     Ui::RegistrationWindow *ui;
00101     Client *m_client;
00102     QString m_lastUsername;
00103     QString m_lastPassword;
00104     bool m_waitingForLoginAfterRegistration = false;
00105 };
00106
00107 #endif // REGISTRATIONWINDOW_H
```

8.35 Файл ExamClient/StatisticsWindow.cpp

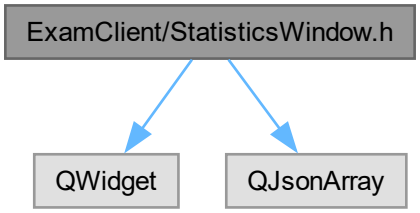
```
#include "StatisticsWindow.h"
#include "ui_StatisticsWindow.h"
#include <QJsonObject>
#include <QJsonArray>
```

#include <QDebug>
Граф включаемых заголовочных файлов для StatisticsWindow.cpp:

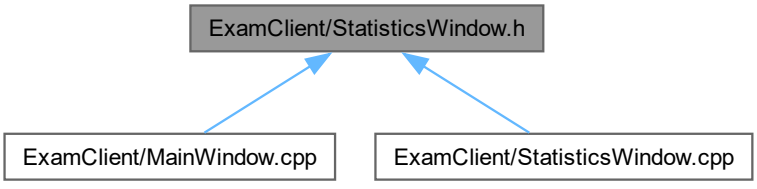


8.36 Файл ExamClient/StatisticsWindow.h

#include <QWidget>
#include <QJsonArray>
Граф включаемых заголовочных файлов для StatisticsWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class StatisticsWindow

  Класс StatisticsWindow реализует окно отображения результатов экзаменов пользователя.

Пространства имен

- namespace Ui

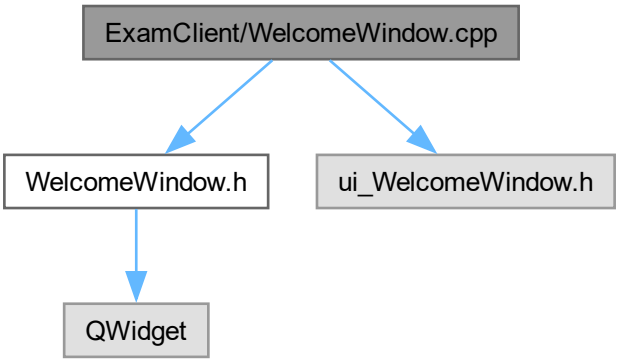
8.37 StatisticsWindow.h

См. документацию.

```
00001 #ifndef STATISTICSWINDOW_H
00002 #define STATISTICSWINDOW_H
00003
00004 #include <QWidget>
00005 #include <QJsonArray>
00006
00007 namespace Ui {
00008 class StatisticsWindow;
00009 }
00010
00017 class StatisticsWindow : public QWidget
00018 {
00019     Q_OBJECT
00020
00021 public:
00029     explicit StatisticsWindow(QWidget *parent = nullptr);
00030
00036     ~StatisticsWindow();
00037
00045     void setStatisticsData(const QJsonArray &stats);
00046
00052     void applyFilters();
00053
00054 signals:
00058     void backRequested();
00059
00060 private slots:
00066     void on_backButton_clicked();
00067
00073     void on_applyFilterButton_clicked();
00074
00075 private:
00076     QJsonArray originalStats;
00077     Ui::StatisticsWindow *ui;
00078 };
00079
00080 #endif // STATISTICSWINDOW_H
```

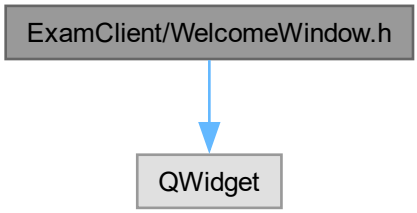
8.38 Файл ExamClient/WelcomeWindow.cpp

#include "WelcomeWindow.h"
#include "ui_WelcomeWindow.h"
Граф включаемых заголовочных файлов для WelcomeWindow.cpp:

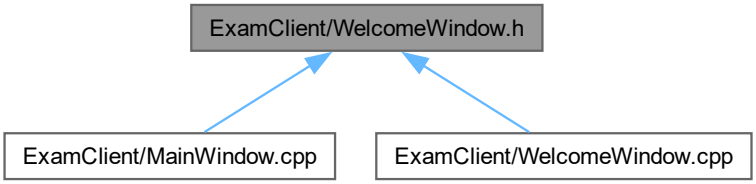


8.39 Файл ExamClient/WelcomeWindow.h

#include <QWidget>
Граф включаемых заголовочных файлов для WelcomeWindow.h:



Граф файлов, в которые включается этот файл:



Структуры данных

- class WelcomeWindow

  Класс WelcomeWindow представляет приветственное окно после входа пользователя.

Пространства имен

- namespace Ui

8.40 WelcomeWindow.h

См. документацию.

```
00001 #ifndef WELCOMEWINDOW_H
00002 #define WELCOMEWINDOW_H
00003
00004 #include <QWidget>
00005
00006 namespace Ui {
00007 class WelcomeWindow;
00008 }
00009
00019 class WelcomeWindow : public QWidget
00020 {
00021     Q_OBJECT
00022
00023 public:
00031     explicit WelcomeWindow(QWidget *parent = nullptr);
00032
```

```
00038     ~WelcomeWindow();
00039
00040 signals:
00046     void startExamRequested();
00047
00053     void viewStatisticsRequested();
00054
00060     void exitRequested();
00061
00067     void profileRequested();
00068
00069 private slots:
00073     void on_startExamButton_clicked();
00074
00078     void on_viewStatisticsButton_clicked();
00079
00083     void on_exitButton_clicked();
00084
00088     void on_profileButton_clicked();
00089
00090 private:
00091     Ui::WelcomeWindow *ui;
00092 };
00093
00094 #endif // WELCOMEWINDOW_H
```

8.41 Файл mainpage.md