My Project

Создано системой Doxygen 1.9.8

Глава 1

Алфавитный указатель классов

1.1 Классы

Классы с их кратким описанием.

Глава 2

Список файлов

2.1 Файлы

Полный список документированных файлов.

Глава 3

Классы

3.1 Класс Connection

Класс для управления сетевыми соединениями сервера

#include <connection.h>

Открытые статические члены

- static int conn (const Params *p)
  Установка соединения и обработка клиентов

3.1.1 Подробное описание

Класс для управления сетевыми соединениями сервера

Обеспечивает установку соединения, аутентификацию клиентов и обработку данных

3.1.2 Методы

3.1.2.1 conn()

int Connection::conn (
const Params * p) [static]

Установка соединения и обработка клиентов

Основной метод установки соединения и обработки клиентов

Аргументы

	p	Указатель на параметры соединения

Возвращает

Код завершения (0 - успех, 1 - ошибка)

Исключения

	std::system_error	при ошибках сетевых операций

Аргументы

	p	Указатель на параметры соединения

Возвращает

Код завершения (0 - успех, 1 - ошибка)

Исключения

	std::system_error	при ошибках сетевых операций

Выполняет полный цикл работы сервера: создание сокета, привязка, прослушивание, аутентификация клиента и обработка данных

Объявления и описания членов классов находятся в файлах:

- connection.h
- connection.cpp

3.2 Структура Params

Структура для хранения параметров программы

#include <interface.h>

Открытые атрибуты

- string inFileName
  Имя файла с пользователями
- string inFileJournal
  Имя журнального файла
- string inFileData
  Имя файла данных
- string logFile
  Имя файла лога
- int Port
  Порт сервера
- string Address
  Адрес сервера

3.2.1 Подробное описание

Структура для хранения параметров программы

Объявления и описания членов структуры находятся в файле:

- interface.h

3.3 Класс UserInterface

Класс для обработки аргументов командной строки

#include <interface.h>

Открытые члены

- UserInterface ()
  Конструктор по умолчанию
- bool Parser (int argc, const char **argv)
  Парсинг аргументов командной строки
- string getDescription ()
  Получение описания параметров
- Params getParams ()
  Получение параметров программы

3.3.1 Подробное описание

Класс для обработки аргументов командной строки

Использует boost::program_options для парсинга параметров

3.3.2 Конструктор(ы)

3.3.2.1 UserInterface()

UserInterface::UserInterface ( )

Конструктор по умолчанию

Конструктор UserInterface.

Инициализирует описание параметров командной строки

3.3.3 Методы

3.3.3.1 getDescription()

std::string UserInterface::getDescription ( )

Получение описания параметров

Возвращает

Строка с описанием параметров

3.3.3.2 getParams()

Params UserInterface::getParams ( ) [inline]

Получение параметров программы

Возвращает

Структура с параметрами

3.3.3.3 Parser()

bool UserInterface::Parser (
int argc,
const char ** argv)

Парсинг аргументов командной строки

Аргументы

	argc	Количество аргументов
	argv	Массив аргументов

Возвращает

true если парсинг успешен, false если требуется показать справку

Объявления и описания членов классов находятся в файлах:

- interface.h
- interface.cpp

Глава 4

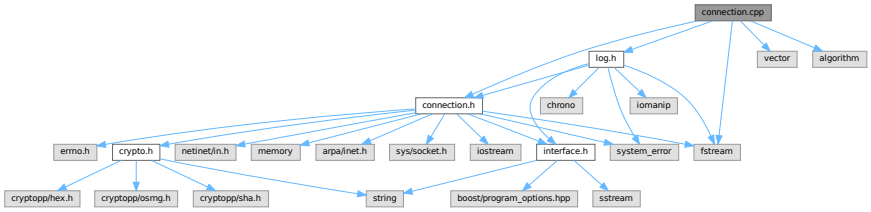
Файлы

4.1 Файл connection.cpp

Реализация сетевого соединения сервера

```
#include "connection.h"
#include "log.h"
#include <fstream>
#include <vector>
#include <algorithm>
```

Граф включаемых заголовочных файлов для connection.cpp:



Функции

- bool findUserInFile (const std::string &filename, const std::string &username, std::string &password)

  Поиск пользователя в файле по логину с кэшированием
- std::string generateSalt (size_t length=16)

  Генерация случайной соли
- uint32_t processVector (int client_socket, uint32_t vector_size, const Params *p)

  Обработка одного вектора с проверкой на переполнение
- void safeRecv (int socket, void *buffer, size_t size, const Params *p, const std::string &context)

  Безопасное получение данных фиксированного размера
- void safeSend (int socket, const void *data, size_t size, const Params *p, const std::string &context)

  Безопасная отправка данных

4.1.1 Подробное описание

Реализация сетевого соединения сервера

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Содержит функции для установки соединения, аутентификации и обработки данных от клиентов

4.1.2 Функции

4.1.2.1 findUserInFile()

```
bool findUserInFile (
        const std::string & filename,
        const std::string & username,
        std::string & password )
```

Поиск пользователя в файле по логину с кэшированием

Аргументы

	filename	Имя файла с пользователями
	username	Логин для поиска
	password	Найденный пароль (выходной параметр)

Возвращает

true если пользователь найден, false если нет

Предупреждения

Файл пользователей загружается только при первом вызове

4.1.2.2 generateSalt()

```
std::string generateSalt (
                size_t length = 16 )
```

Генерация случайной соли

Аргументы

	length	Длина соли (по умолчанию 16)

Возвращает

Случайная соль

4.1.2.3 processVector()

```
uint32_t processVector (
                int client_socket,
                uint32_t vector_size,
                const Params * p )
```

Обработка одного вектора с проверкой на переполнение

Аргументы

	client_socket	Сокет клиента
	vector_size	Размер вектора
	p	Параметры соединения

Возвращает

Результат обработки вектора (произведение элементов)

Исключения

	std::system_error	при ошибках сетевых операций

Предупреждения

Проверяет переполнение и ограничивает размер вектора

4.1.2.4 safeRecv()

```
void safeRecv (
                int socket,
                void * buffer,
```

```
size_t size,
const Params * p,
const std::string & context )
```

Безопасное получение данных фиксированного размера

Аргументы

socket	Сокет
buffer	Буфер для данных
size	Размер данных
p	Параметры соединения
context	Контекст для сообщения об ошибке

Исключения

std::system_error	при ошибках получения данных

4.1.2.5 safeSend()

```
void safeSend (
        int socket,
        const void * data,
        size_t size,
        const Params * p,
        const std::string & context )
```

Безопасная отправка данных

Аргументы

socket	Сокет
data	Данные для отправки
size	Размер данных
p	Параметры соединения
context	Контекст для сообщения об ошибке

Исключения

std::system_error	при ошибках отправки данных

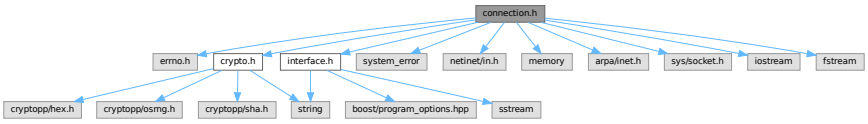
4.2 Файл connection.h

Заголовочный файл для класса соединения

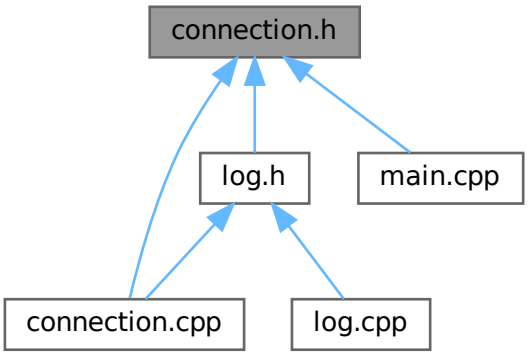
```
#include "errno.h"
#include "crypto.h"
#include "interface.h"
#include <system_error>
```

```
#include <netinet/in.h>
#include <memory>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <iostream>
#include <fstream>
```

Граф включаемых заголовочных файлов для connection.h:



Граф файлов, в которые включается этот файл:



Классы

- class Connection

  Класс для управления сетевыми соединениями сервера

Макросы

- #define BUFFER_SIZE 1024

  Размер буфера для сетевых операций

4.2.1 Подробное описание

Заголовочный файл для класса соединения

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Определяет класс Connection для управления сетевыми соединениями

4.3 connection.h

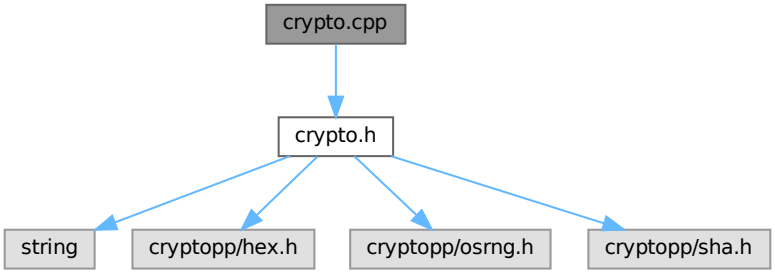
См. документацию.

```
00001
00011 #pragma once
00012 #include "errno.h"
00013 #include "crypto.h"
00014 #include "interface.h"
00015 #include <system_error>
00016 #include <netinet/in.h>
00017 #include <memory>
00018 #include <arpa/inet.h>
00019 #include <sys/socket.h>
00020 #include <iostream>
00021 #include <fstream>
00022
00023 #define BUFFER_SIZE 1024
00024
00025 using namespace std;
00026
00032 class Connection{
00033 public:
00040     static int conn(const Params* p);
00041 };
```

4.4 Файл crypto.cpp

Реализация криптографических функций

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



Функции

- string auth (string salt, string pass)
  Вычисление хеша аутентификации

4.4.1 Подробное описание

Реализация криптографических функций

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Содержит функцию для вычисления хеша аутентификации

4.4.2 Функции

4.4.2.1 auth()

```
string auth (
            string salt,
            string pass )
```

Вычисление хеша аутентификации

Функция аутентификации

Аргументы

	salt	Соль для хеширования
	pass	Пароль пользователя

Возвращает

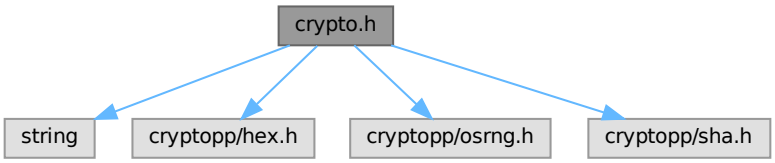
Хеш SHA-256 в hex-формате

Использует алгоритм SHA-256 из библиотеки CryptoPP

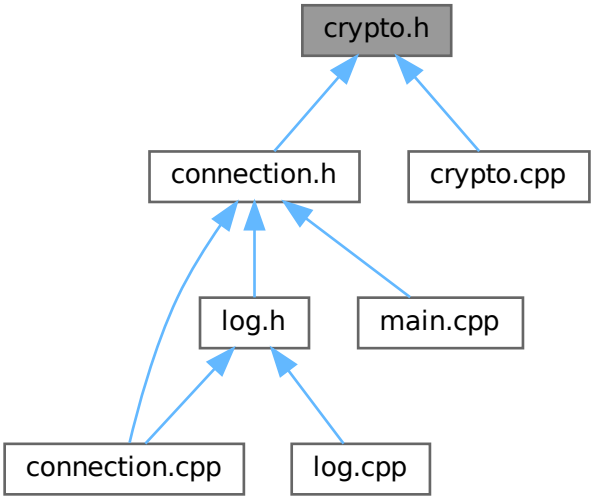
4.5 Файл crypto.h

Заголовочный файл для криптографических функций

#include <string>
#include <cryptopp/hex.h>
#include <cryptopp/osrng.h>
#include <cryptopp/sha.h>
Граф включаемых заголовочных файлов для crypto.h:



Граф файлов, в которые включается этот файл:



Функции

- string auth (string salt, string pass)

  Функция аутентификации

4.5.1 Подробное описание

Заголовочный файл для криптографических функций

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Определяет функцию аутентификации с использованием CryptoPP

4.5.2 Функции

4.5.2.1 auth()

```
string auth (
            string salt,
            string pass )
```

Функция аутентификации

Аргументы

	salt	Соль для хеширования
	pass	Пароль пользователя

Возвращает

Хеш SHA-256 от соли и пароля

Функция аутентификации

Аргументы

	salt	Соль для хеширования
	pass	Пароль пользователя

Возвращает

Хеш SHA-256 в hex-формате

Использует алгоритм SHA-256 из библиотеки CryptoPP

4.6 crypto.h

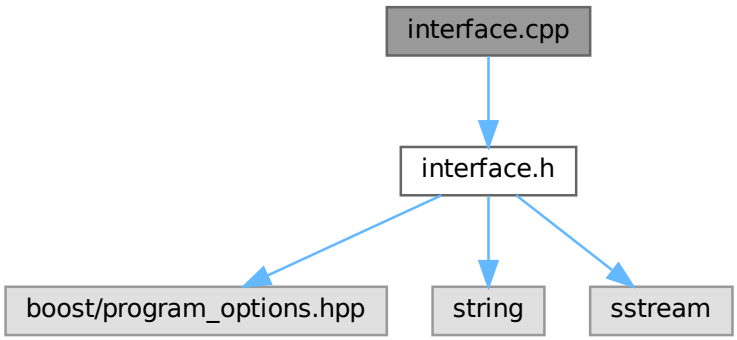
См. документацию.

```
00001
00011 #pragma once
00012 #include <string>
00013 #include <cryptopp/hex.h>
00014 #include <cryptopp/osrng.h>
00015 #include <cryptopp/sha.h>
00016
00017 using namespace std;
00018 namespace CPP = CryptoPP;
00019
00026 string auth(string salt, string pass);
```

4.7 Файл interface.cpp

Реализация пользовательского интерфейса

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



4.7.1 Подробное описание

Реализация пользовательского интерфейса

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

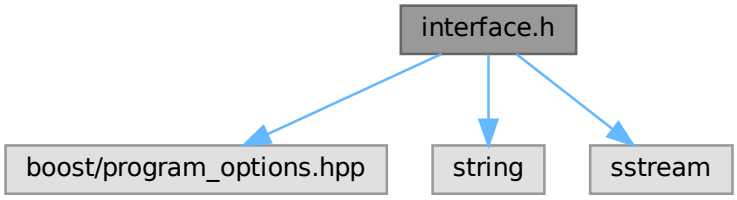
ИБСТ ПГУ

Содержит функции для парсинга и обработки аргументов командной строки

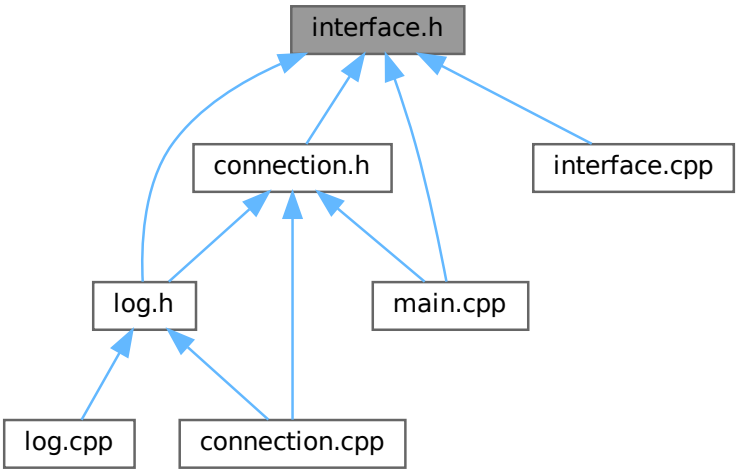
4.8 Файл interface.h

Заголовочный файл для пользовательского интерфейса

#include <boost/program_options.hpp>
#include <string>
#include <sstream>
Граф включаемых заголовочных файлов для interface.h:



Граф файлов, в которые включается этот файл:



Классы

- struct Params

  Структура для хранения параметров программы
- class UserInterface

  Класс для обработки аргументов командной строки

4.8.1 Подробное описание

Заголовочный файл для пользовательского интерфейса

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Определяет структуру параметров и класс для обработки аргументов командной строки

4.9 interface.h

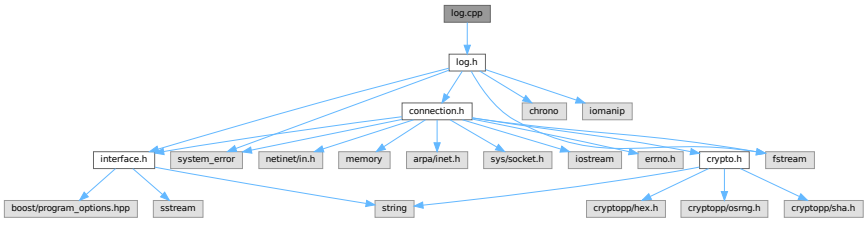
См. документацию.

```
00001
00011 #pragma once
00012 #include <boost/program_options.hpp>
00013 #include <string>
00014 #include <sstream>
00015
00016 using namespace std;
00017 namespace po = boost::program_options;
00018
00023 struct Params {
00024     string inFileName;
00025     string inFileJournal;
00026     string inFileData;
00027     string logFile;
00028     int Port;
00029     string Address;
00030 };
00031
00037 class UserInterface {
00038 private:
00039     po::options_description desc;
00040     po::variables_map vm;
00041     Params params;
00042
00043 public:
00047     UserInterface();
00048
00055     bool Parser(int argc, const char** argv);
00056
00061     string getDescription();
00062
00067     Params getParams() {
00068         return params;
00069     };
00070 };
```

4.10 Файл log.cpp

Реализация функций логирования

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



Функции

- std::string getCurrentTime ()

  Получение текущего времени в формате строки
- void logError (const std::string &logFile, const std::string &errorMessage)

  Запись ошибки в лог-файл

4.10.1 Подробное описание

Реализация функций логирования

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Содержит функции для записи логов с временными метками

4.10.2 Функции

4.10.2.1 getCurrentTime()

```
std::string getCurrentTime ( )
```

Получение текущего времени в формате строки

Возвращает

Строка с текущим временем в формате "ГГГГ-ММ-ДД ЧЧ:ММ:СС.мс"

4.10.2.2 logError()

```
void logError (
            const std::string & logFile,
            const std::string & errorMessage )
```

Запись ошибки в лог-файл

Аргументы

	logFile	Имя файла лога
	errorMessage	Сообщение об ошибке

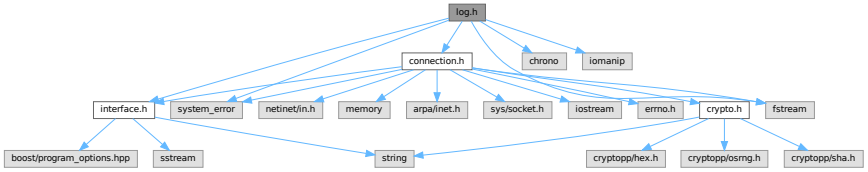
Добавляет временную метку и записывает сообщение в файл

4.11 Файл log.h

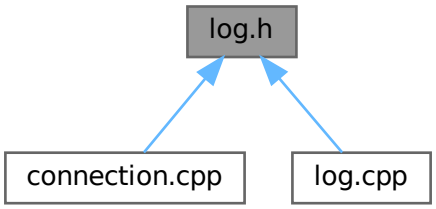
Заголовочный файл для функций логирования

```
#include "connection.h"
#include "interface.h"
#include <fstream>
#include <system_error>
#include <chrono>
#include <iomanip>
```

Граф включаемых заголовочных файлов для log.h:



Граф файлов, в которые включается этот файл:



Функции

- std::string getCurrentTime ()

  Получение текущего времени в формате строки
- void logError (const std::string &logFile, const std::string &errorMessage)

  Запись ошибки в лог-файл

4.11.1 Подробное описание

Заголовочный файл для функций логирования

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Определяет функции для работы с системой логирования

4.11.2 Функции

4.11.2.1 getCurrentTime()

```
std::string getCurrentTime ( )
```

Получение текущего времени в формате строки

Возвращает

Строка с текущим временем в формате "ГГГГ-ММ-ДД ЧЧ:ММ:СС.мс"

4.11.2.2 logError()

```
void logError (
            const std::string & logFile,
            const std::string & errorMessage )
```

Запись ошибки в лог-файл

Аргументы

	logFile	Имя файла лога
	errorMessage	Сообщение об ошибке
	logFile	Имя файла лога
	errorMessage	Сообщение об ошибке

Добавляет временную метку и записывает сообщение в файл

4.12 log.h

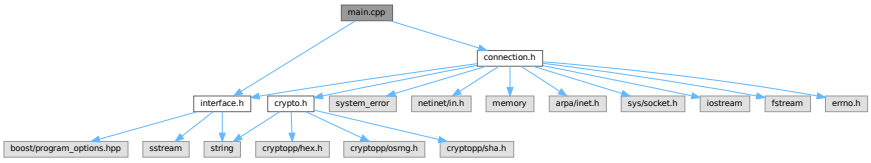
См. документацию.

```
00001
00011 #include "connection.h"
00012 #include "interface.h"
00013 #include <fstream>
00014 #include <system_error>
00015 #include <chrono>
00016 #include <iomanip>
00017
00022 std::string getCurrentTime();
00023
00029 void logError(const std::string& logFile, const std::string& errorMessage);
```

4.13 Файл main.cpp

Главный модуль серверного приложения

#include "connection.h"
#include "interface.h"
Граф включаемых заголовочных файлов для main.cpp:



Функции

- int main (int argc, const char **argv)

  Главная функция программы

4.13.1 Подробное описание

Главный модуль серверного приложения

Автор

Мураев Н.Д.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Основная точка входа в программу, обрабатывающая аргументы командной строки и запускающая сервер

4.13.2 Функции

4.13.2.1 main()

```
int main (
                int argc,
                const char ** argv )
```

Главная функция программы

Аргументы

	argc	Количество аргументов командной строки
	argv	Массив аргументов командной строки

Возвращает

Код завершения программы (0 - успех, 1 - ошибка)

Предупреждения

Для работы программы необходимо указать обязательные параметры