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 Communicator

Класс для управления сетевым соединением с сервером

#include <connection.h>

Открытые статические члены

- static int conn (const Params *p)
  Установка соединения с сервером

3.1.1 Подробное описание

Класс для управления сетевым соединением с сервером

Обеспечивает установку соединения и аутентификацию

3.1.2 Методы

3.1.2.1 conn()

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

Установка соединения с сервером

Аргументы

	in	p	Указатель на параметры соединения

Возвращает

0 при успешном выполнении

Исключения

	system_error	при ошибках сетевого взаимодействия

Аргументы

	in	p	Указатель на параметры соединения

Возвращает

0 при успешном выполнении

Исключения

	system_error	при ошибках сетевого взаимодействия

Метод выполняет:

- Создание сокета и установку соединения
- Аутентификацию с использованием соли и хеширования пароля
- Вызов DataReader для передачи данных

< Семейство адресов IPv4

< Автоматический выбор порта

< Автоматический выбор адреса

< Семейство адресов IPv4

< Порт сервера

< IP-адрес сервера

< Логин пользователя

< Пароль пользователя

Объявления и описания членов классов находятся в файлах:

- connection.h
- connection.cpp

3.2 Класс DataReader

Класс для чтения и передачи данных через сетевое соединение

#include <connection.h>

Открытые статические члены

- static int datareader (const Params *p, int s)
  Чтение данных из файла и передача через сокет

3.2.1 Подробное описание

Класс для чтения и передачи данных через сетевое соединение

Обрабатывает файлы с векторами и передает их на сервер

3.2.2 Методы

3.2.2.1 datareader()

```
int DataReader::datareader (
                const Params * p,
                int s )   [static]
```

Чтение данных из файла и передача через сокет

Аргументы

	in	p	Указатель на параметры с именами файлов
	in	s	Дескриптор сокета для передачи данных

Возвращает

0 при успешном выполнении

Исключения

	system_error	при ошибках ввода-вывода или сетевого взаимодействия

Аргументы

	in	p	Указатель на параметры с именами файлов
	in	s	Дескриптор сокета для передачи данных

Возвращает

0 при успешном выполнении

Исключения

system_error	при ошибках ввода-вывода или сетевого взаимодействия

Метод выполняет:

- Чтение количества векторов из файла
- Передачу размеров и элементов каждого вектора на сервер
- Получение результатов от сервера и запись в файл

< Количество векторов для обработки

< Размер текущего вектора

< Вектор числовых значений

< Число в сетевом порядке байт

< Результат обработки от сервера

Объявления и описания членов классов находятся в файлах:

- connection.h
- connection.cpp

3.3 Структура Params

Структура для хранения параметров командной строки

#include <interface.h>

Открытые атрибуты

- string inFileName
  Имя входного файла с данными
- string inFileResult
  Имя файла для результатов
- string inFileData
  Имя файла с данными аутентификации
- int Port
  Порт сервера
- string Address
  Адрес сервера

3.3.1 Подробное описание

Структура для хранения параметров командной строки

Объявления и описания членов структуры находятся в файле:

- interface.h

3.4 Класс UserInterface

Класс для обработки параметров командной строки

#include <interface.h>

Открытые члены

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

3.4.1 Подробное описание

Класс для обработки параметров командной строки

Обеспечивает парсинг и валидацию входных параметров

3.4.2 Конструктор(ы)

3.4.2.1 UserInterface()

UserInterface::UserInterface ( )

Конструктор класса UserInterface.

Инициализирует опции командной строки с обязательными параметрами < Опция помощи

< Входной файл

< Файл результатов

< Файл с данными аутентификации

< Порт сервера

< Адрес сервера

3.4.3 Методы

3.4.3.1 getDescription()

std::string UserInterface::getDescription ( )

Получение описания параметров

Возвращает

Строка с описанием поддерживаемых опций

3.4.3.2 getParams()

Params UserInterface::getParams ( ) [inline]

Получение параметров

Возвращает

Структура Params с распарсенными значениями

3.4.3.3 Parser()

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

Парсинг аргументов командной строки

Аргументы

	in	argc	Количество аргументов
	in	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

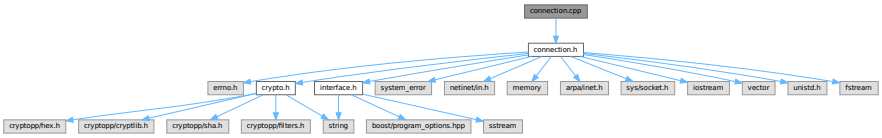
Глава 4

Файлы

4.1 Файл connection.cpp

Реализация классов Communicator и DataReader для сетевого взаимодействия

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



4.1.1 Подробное описание

Реализация классов Communicator и DataReader для сетевого взаимодействия

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

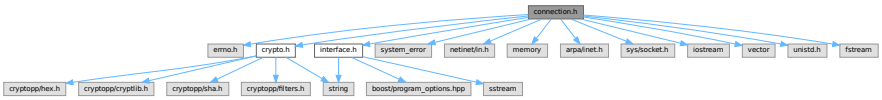
ИБСТ ПГУ

4.2 Файл connection.h

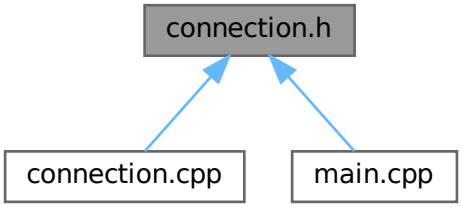
Заголовочный файл классов Communicator и DataReader для сетевого взаимодействия

```
#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 <vector>
#include <unistd.h>
#include <fstream>
```

Граф включаемых заголовочных файлов для connection.h:



Граф файлов, в которые включается этот файл:



Классы

- class Communicator

  Класс для управления сетевым соединением с сервером
- class DataReader

  Класс для чтения и передачи данных через сетевое соединение

Макросы

- #define BUFFER_SIZE 1024

  Размер буфера для сетевого обмена

4.2.1 Подробное описание

Заголовочный файл классов Communicator и DataReader для сетевого взаимодействия

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

4.3 connection.h

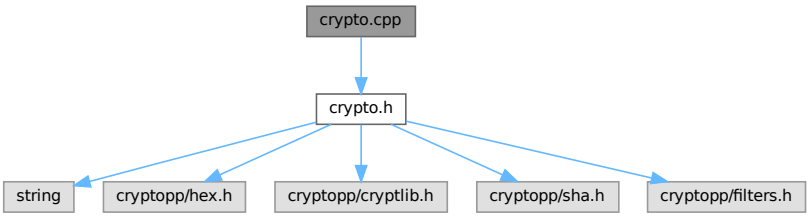
См. документацию.

```
00001
00010 #pragma once
00011 #include "errno.h"
00012 #include "crypto.h"
00013 #include "interface.h"
00014 #include <system_error>
00015 #include <netinet/in.h>
00016 #include <memory>
00017 #include <arpa/inet.h>
00018 #include <sys/socket.h>
00019 #include <iostream>
00020 #include <vector>
00021 #include <unistd.h>
00022 #include <fstream>
00023 using namespace std;
00024
00025 #define BUFFER_SIZE 1024
00026
00032 class Communicator{
00033 private:
00034     static string salt;
00035
00036 public:
00043     static int conn(const Params* p);
00044 };
00045
00051 class DataReader{
00052 public:
00060     static int datareader(const Params* p, int s);
00061 };
```

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)
  Функция аутентификации с использованием хеширования

4.4.1 Подробное описание

Реализация криптографических функций

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Предупреждения

Используется библиотека CryptoPP

4.4.2 Функции

4.4.2.1 auth()

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

Функция аутентификации с использованием хеширования

Аргументы

	in	salt	Соль для хеширования
	in	pass	Пароль пользователя

Возвращает

Хеш-строка в hex-формате

Использует алгоритм SHA224 для создания хеша комбинации соли и пароля < Объект для вычисления SHA224

< Результирующий хеш

< Конкатенация соли и пароля

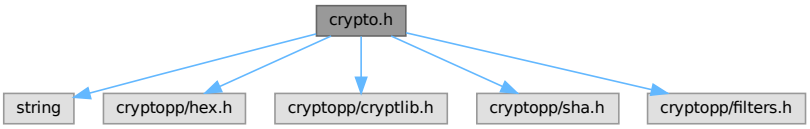
< Возврат хеша в hex-формате

4.5 Файл crypto.h

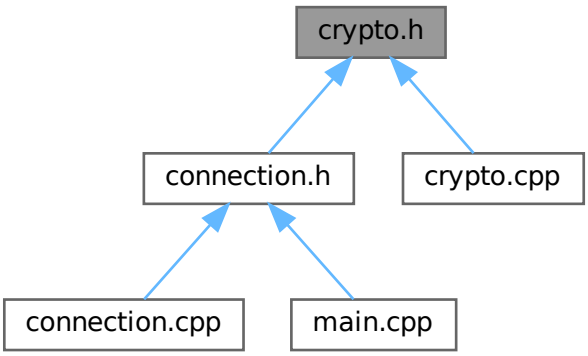
Заголовочный файл для криптографических функций

```
#include <string>
#include <cryptopp/hex.h>
#include <cryptopp/cryptlib.h>
#include <cryptopp/sha.h>
#include <cryptopp/filters.h>
```

Граф включаемых заголовочных файлов для crypto.h:



Граф файлов, в которые включается этот файл:



Макросы

- #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1

Функции

- string auth (string salt, string pass)
  Функция аутентификации с использованием хеширования

4.5.1 Подробное описание

Заголовочный файл для криптографических функций

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Предупреждения

Используется библиотека CryptoPP

4.5.2 Функции

4.5.2.1 auth()

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

Функция аутентификации с использованием хеширования

Аргументы

	in	salt	Соль для хеширования
	in	pass	Пароль пользователя

Возвращает

Хеш-строка в hex-формате

Использует алгоритм SHA224 для создания хеша соли и пароля

Аргументы

	in	salt	Соль для хеширования
	in	pass	Пароль пользователя

Возвращает

Хеш-строка в hex-формате

Использует алгоритм SHA224 для создания хеша комбинации соли и пароля < Объект для вычисления SHA224

< Результирующий хеш

< Конкатенация соли и пароля

< Возврат хеша в hex-формате

4.6 crypto.h

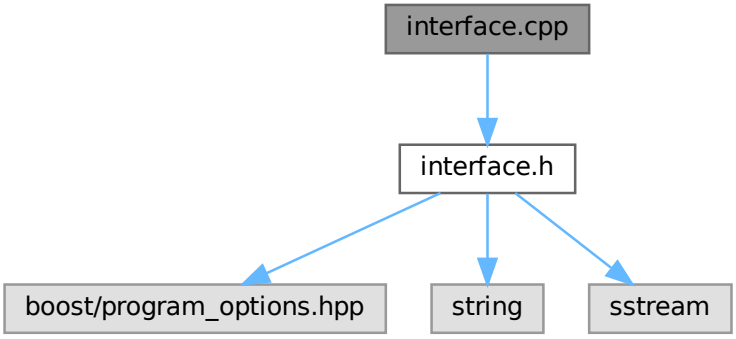
См. документацию.

```
00001
00011 #pragma once
00012 #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1
00013 #include <string>
00014 #include <cryptopp/hex.h>
00015 #include <cryptopp/cryptlib.h>
00016 #include <cryptopp/sha.h>
00017 #include <cryptopp/filters.h>
00018 using namespace std;
00019 namespace CPP = CryptoPP;
00020
00028 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

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

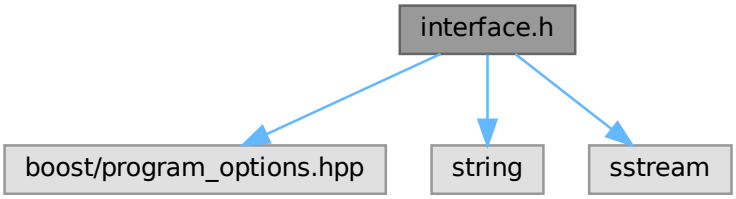
ИБСТ ПГУ

4.8 Файл interface.h

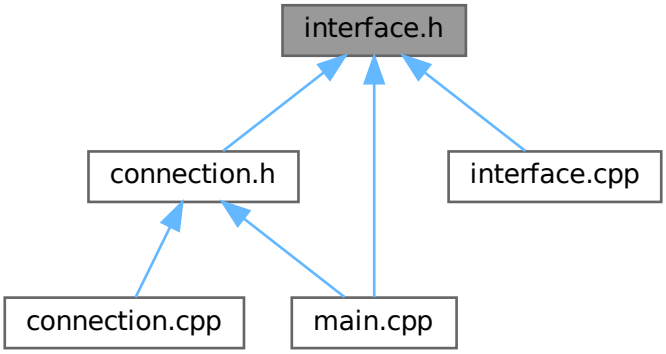
Заголовочный файл пользовательского интерфейса

#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 командной строки
- class UserInterface

  Класс для обработки параметров командной строки

4.8.1 Подробное описание

Заголовочный файл пользовательского интерфейса

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Использует boost::program_options для парсинга аргументов командной строки

4.9 interface.h

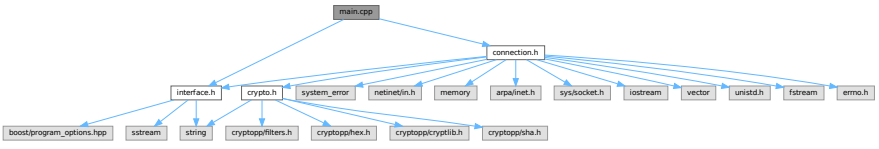
См. документацию.

```
00001
00011 #pragma once
00012 #include <boost/program_options.hpp>
00013 #include <string>
00014 #include <sstream>
00015 using namespace std;
00016 namespace po = boost::program_options;
00017
00022 struct Params {
00023     string inFileName;
00024     string inFileResult;
00025     string inFileData;
00026     int Port;
00027     string Address;
00028 };
00029
00035 class UserInterface {
00036 private:
00037     po::options_description desc;
00038     po::variables_map vm;
00039     Params params;
00040
00041 public:
00045     UserInterface();
00046
00053     bool Parser(int argc, const char** argv);
00054
00059     string getDescription();
00060
00065     Params getParams() {
00066         return params;
00067     };
00068 };
```

4.10 Файл main.cpp

Главный файл приложения

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



Функции

- int main (int argc, const char **argv)
  Главная функция приложения

4.10.1 Подробное описание

Главный файл приложения

Автор

Клименко Г.А.

Версия

1.0

Дата

2025

Авторство

ИБСТ ПГУ

Точка входа в программу, обработка параметров и запуск соединения

4.10.2 Функции

4.10.2.1 main()

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

Главная функция приложения

Аргументы

	in	argc	Количество аргументов командной строки
	in	argv	Массив аргументов командной строки

Возвращает

0 при успешном выполнении, 1 при ошибке

< Объект пользовательского интерфейса