SERVER Андреев

Создано системой Doxygen 1.9.4

1 Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

2 Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

3 Список файлов

3.1 Файлы

Полный список документированных файлов.

4 Классы

4.1 Класс base_reader

Класс базы данных

#include <read_base.h>

Граф связей класса base_reader:

Открытые члены

- std::vector< std::string > get_logins ()
  Геттер вектора с логинами
- std::vector< std::string > get_passwords ()
  Геттер вектора с паролями
- void read_base (std::ifstream &b)
  Метод чтения базы данных
- base_reader (std::string base_loc, std::string log_loc)
  Конструктор инициализации
- ~base_reader ()
  Деструктор модуля

Открытые атрибуты

- logger log
  Объекта класса loger для записей в журнал
- std::string log_location
  Расположение лог файла
- std::ifstream cl_base
  Объект потока ifstream для чтения базы данных

Закрытые члены

- std::vector< std::string > handle_id_s (std::vector< std::string > logins)
  Метод разбора логинов
- std::vector< std::string > handle_password_s (std::vector< std::string > pswds)
  Метод разбора паролей

Закрытые данные

- std::vector< std::string > logins
  Вектор для хранения логинов
- std::vector< std::string > passwords
  Вектор для хранения паролей
- std::vector< std::string > clients
  Вектор для хранения логинов/паролей

4.1.1 Подробное описание

Класс базы данных

Осуществляется работа с базой данных. Разбор данных, получение логинов и паролей

4.1.2 Конструктор(ы)

4.1.2.1 base_reader() base_reader::base_reader (
std::string base_loc,
std::string log_loc)

Конструктор инициализации

Открывается файл базы данных, при удачном открытии происходит чтение содерджимого файла

Аргументы

in	base_loc	Расположения файла базы данных
in	log_loc	Расположения лог файла

Исключения

critical_error,если	не удалось открыть файл с базой данных

4.1.2.2 ~base_reader() base_reader::~base_reader ( )

Деструктор модуля

Закрывается файл с базой данных

4.1.3 Методы

4.1.3.1 handle_id_s() std::vector< std::string > base_reader::handle_id_s (std::vector< std::string > logins) [private]

Метод разбора логинов

Из строки логин:пароль выбирается логин

Аргументы

in	logins	Вектор для хранения логинов

4.1.3.2 handle_password_s() std::vector< std::string > base_reader::handle_password_s (std::vector< std::string > pswds) [private]

Метод разбора паролей

Из строки логин:пароль выбирается пароль

Аргументы

in	pswds	Вектор для хранения паролей

4.1.3.3 read_base() void base_reader::read_base (std::ifstream & b)

Метод чтения базы данных

Построчно считывается содержимое базы данных в формате логин:пароль

Аргументы

in	base	Объект потока ifstream с открытым файлом базы данных

Исключения

critical_error,если	содержимое базы данных не соответствует формату логин:пароль

Объявления и описания членов классов находятся в файлах:

- read_base.h
- read_base.cpp

4.2 Класс critical_error

Класс ошибок

#include <error.h>

Граф наследования:critical_error:

Граф связей класса critical_error:

Открытые члены

- critical_error (const std::string &s)
  Конструктор ошибки

4.2.1 Подробное описание

Класс ошибок

Используется для отлова специфических ошибок, возникающих в ходе работы модулей В конструкторе указывается строка с сообщением ошибки

4.2.2 Конструктор(ы)

4.2.2.1 critical_error() critical_error::critical_error (const std::string & s) [inline]

Конструктор ошибки

Аргументы

in	s	Сообщение об ошибке

Объявления и описания членов класса находятся в файле:

- error.h

4.3 Класс logger

Класс записи сообщений в журнал

#include <logger.h>

Открытые члены

- int write_log (std::string log_loc, std::string message)
  Метод записи сообщения в лог файл

Открытые атрибуты

- std::ofstream log
  Объект ofstream для открытия файла журнала

4.3.1 Подробное описание

Класс записи сообщений в журнал

Запись сообщений в лог файл в методе write_log

4.3.2 Методы

4.3.2.1 write_log()

```
int logger::write_log (
        std::string log_loc,
        std::string message )
```

Метод записи сообщения в лог файл

Получается текущее время, затем сообщение записывается в файл в формате время/сообщение. Файл открывается, закрывается каждый раз для записи сообщения

Аргументы

in	log_loc	Расположение лог файла
in	message	Сообщение для записи

Исключения

critical_error,если	введеное расположение файла указывает на несуществующий объект

Объявления и описания членов классов находятся в файлах:

- logger.h
- logger.cpp

4.4 Класс server

Класс сервера

#include <server.h>

Граф связей класса server:

Открытые члены

- void connect_to_cl ()
  Соединение с клиентом
- int client_auth ()
  Аутентификация клиента
- std::string SALT_generate ()
  Генерация соли
- std::string convert_to_hex (uint64_t)
  Конвертирование числа в 16-ричный формат
- void send_data (std::string data, std::string msg)
  Отправка сообщения клиенту
- std::string recv_data (std::string messg)
  Прием данных от клиента
- std::string hash_gen (std::string &salt, std::string &password)
  Генерация хеша
- void close_sock ()
  Метод закрытия соединения с клиентом
- void start ()
  Запуск сервера
- int64_t calculation (int64_t number1, int64_t number2)
  Калькулятор
- int handle_data ()
  Обработка данных от клиента
- void work (UI &intf)
  Работа сервера

Открытые атрибуты

- logger log
  Объекта класса loger для записи сообщений в журнал
- uint32_t nums
  Количество векторов
- int serverSocket
  Переменная для сокетов сервера и клиента
- int clientSocket
- std::string cl_id
  Айди клиента, расположение лог файла
- std::string log_location

Закрытые данные

- struct sockaddr_in serverAddr clientAddr
  Структуры адреса сервера и клиента
- socklen_t addr_size
  Размер адреса
- std::string base_location
  Расположение базы данных
- std::vector< std::string > cl_ids
  Вектора с логинами и паролями
- std::vector< std::string > cl_passes
- size_t buflen =1024
  Размер буффера
- std::unique_ptr< char[]> buffer {new char[buflen]}
  Unique_ptr для приема и отправки сообщений
- uint port
  Порт
- std::string digits [16] = {"0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"}
  Набор 16-ричных символов для генерации SALT.

4.4.1 Подробное описание

Класс сервера

Порт, расположение логфайла и базы данных устанавливаются в методе work с помощью методов объекта класса UI(Передается по ссылке)
Работу сервера начинает метод work Сетевое взаимодействие реализовано через методы client_auth, connect_to_cl, handle_data, recv_data, send_data

4.4.2 Методы

4.4.2.1 calculation()

```
int64_t server::calculation (
        int64_t number1,
        int64_t number2 )
```

Калькулятор

Суммирует элементы вектора

Аргументы

in	number1	Первое слагаемое int64_t
in	number2	Второе слагаемое int64_t

Предупреждения

В случае переполнения при сумме возвращается максимум или минимум типа данных, так же если второе слагаемое имеет значение больше или меньше лимита типа данных

Возвращает

Функция возвращает сумму двух чисел

4.4.2.2 client_auth() int server::client_auth ( )

Аутентификация клиента

Проверка присланного хеша со сгенерированным на основе присланного логина и соответствующего пароля из базы данных

Возвращает

Функция возвращает 1, если не удалось аутентифицировать клиента

4.4.2.3 connect_to_cl() void server::connect_to_cl ( )

Соединение с клиентом

Осуществляется установление соединения с клиентом. Сервер ожидает подключения

Возвращает

Функция ничего не возвращает

Исключения

critical_error,если	не удалось встать в режим ожидания подключения

4.4.2.4 convert_to_hex() std::string server::convert_to_hex (uint64_t x)

Конвертирование числа в 16-ричный формат

Число в цикле конвертируется в 16-ричное

Аргументы

in	d_salt	Случайно сгенерированное чилсо для конвертации

Возвращает

Функция возвращает сгенерированное 64-разрядное число

4.4.2.5 handle_data() int server::handle_data ( )

Обработка данных от клиента

Прием количества векторов и их значение, выполнение суммы элементов векторов

Предупреждения

В случае приема неверного типа данных соединение с клиентом обрывается

4.4.2.6 hash_gen() std::string server::hash_gen (std::string & salt, std::string & password)

Генерация хеша

Хеш генерируется на основе соли и пароля

Аргументы

in	salt	Соль
in	pswd	Пароль клиента

Возвращает

Функция возвращает сгенерированный хеш

4.4.2.7 recv_data() std::string server::recv_data (std::string messg)

Прием данных от клиента

Внутри в цикле обрабатывается ситуация когда размер присылаемого сообщенния больше буффера

Аргументы

in	msg	Строка для записи принятого сообщения

Исключения

В	случае ошибки при приеме закрывается соединение с клиентом

4.4.2.8 SALT_generate() std::string server::SALT_generate ( )

Генерация соли

Генерируется 64-разрядное число, затем конвертируется в 16-ричное с дополнением 0 до длины 16

Возвращает

Сгенерированная соль

4.4.2.9 send_data() void server::send_data (std::string data, std::string msg)

Отправка сообщения клиенту

Осуществляется отправка данных клиенту с помощью unique_ptr

Аргументы

in	data	Строка, которую необходимо отправить
in	log_msg	Сообщение,записываемое в журнал при возникновении ошибки

Исключения

В	случае ошибки при отправке закрывается соединение с клиентом

4.4.2.10 start() void server::start ( )

Запуск сервера

Создание сокета сервера и его привязка к локальному адресу, чтение базы данных клиентов

Исключения

critical_error,если	не удалось создать или привязать сокет

4.4.2.11 work() void server::work (UI & intf)

Работа сервера

Осуществляется запуск сервера, происходит соединение с клиентом, аутентификация и обработка данных, присланных от клиента

Аргументы

in	intf	Объект класса UI для получения расположения лог файла, базы данных и порта при разборе комстроки

Исключения

В	случае ошибки на всех этапах работы сервера: аутентификация, старт, соединение с клиентом, обработка данных

Объявления и описания членов классов находятся в файлах:

- server.h
- server.cpp

4.5 Класс UI

Класс пользовательского интерфейса

#include <ui.h>

Граф связей класса UI:

5 Файлы

5.1 Файл error.h

Заголовочный файл для модуля error.

#include <stdexcept>
#include <string>
Граф включаемых заголовочных файлов для error.h: Граф файлов, в которые включается этот файл:

Классы

- class critical_error
  Класс ошибок

5.1.1 Подробное описание

Заголовочный файл для модуля error.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.2 error.h

См. документацию.

```
1
8 #pragma once
9 #include <stdexcept>
10 #include <string>
15 class critical_error:public std::runtime_error{
16     public:
20     critical_error(const std::string& s):std::runtime_error(s){}
21 };
```

5.3 Файл logger.h

Заголовочный файл модуля logger.

#include <vector>
#include <string>
#include <fstream>
#include <iostream>
#include <chrono>
#include <cstring>
#include "error.h"
#include <boost/filesystem.hpp>
Граф включаемых заголовочных файлов для logger.h: Граф файлов, в которые включается этот файл:

Классы

- class logger

  Класс записи сообщений в журнал

5.3.1 Подробное описание

Заголовочный файл модуля logger.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.4 logger.h

См. документацию.

```
1
8 #pragma once
9 #include <vector>
10 #include <string>
11 #include <fstream>
12 #include <iostream>
13 #include <chrono>
14 #include <cstring>
15 #include "error.h"
16 #include <boost/filesystem.hpp>
20 class logger{
21     public:
24         std::ofstream log;
33         int write_log(std::string log_loc,std::string message);
34 };
```

5.5 Файл main.cpp

Главный файл проекта

#include "ui.h"
#include "read_base.h"
#include "server.h"
#include "logger.h"
#include "error.h"
Граф включаемых заголовочных файлов для main.cpp:

Функции

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

5.5.1 Подробное описание

Главный файл проекта

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.6 Файл read_base.cpp

Исполняемый файл модуля read_base.

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

5.6.1 Подробное описание

Исполняемый файл модуля read_base.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.7 Файл read_base.h

Заголовочный файл модуля read_base.

#include <vector>
#include <string>
#include <fstream>
#include <iostream>
#include <algorithm>
#include "logger.h"
#include "error.h"
Граф включаемых заголовочных файлов для read_base.h: Граф файлов, в которые включается этот файл:

Классы

- class base_reader
  Класс базы данных

5.7.1 Подробное описание

Заголовочный файл модуля read_base.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.8 read_base.h

См. документацию.

```
1
8 #pragma once
9 #include <vector>
10 #include <string>
11 #include <fstream>
12 #include <iostream>
13 #include <algorithm>
14 #include "logger.h"
15 #include "error.h"
19 class base_reader{
20     private:
23         std::vector<std::string> logins;
26         std::vector<std::string> passwords;
29         std::vector<std::string> clients;
35         std::vector<std::string> handle_id_s(std::vector<std::string> logins);
41         std::vector<std::string> handle_password_s(std::vector<std::string> pswds);
42     public:
46         logger log;
50         std::string log_location;
54         std::ifstream cl_base;
58         std::vector<std::string> get_logins();
62         std::vector<std::string> get_passwords();
69         void read_base(std::ifstream &b);
77         base_reader(std::string base_loc,std::string log_loc);
82         ~base_reader();
83 };
```

5.9 Файл server.cpp

Исполняемый файл модуля server.

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

5.9.1 Подробное описание

Исполняемый файл модуля server.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.10 Файл server.h

Заголовочный файл модуля server.

#include <iostream>
#include <string>
#include <vector>
#include <cstring>
#include <algorithm>
#include <memory>
#include <stdlib.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <random>
#include <chrono>
#include <thread>
#include <limits>
#include "read_base.h"
#include "logger.h"
#include "error.h"
#include "ui.h"
#include <cryptopp/cryptlib.h>
#include <cryptopp/hex.h>
#include <cryptopp/files.h>
#include <cryptopp/md5.h>
#include <cryptopp/filters.h>
#include <cryptopp/osrng.h>
Граф включаемых заголовочных файлов для server.h: Граф файлов, в которые включается этот файл:

Классы

- class server

  Класс сервера

Макросы

- #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1

5.10.1 Подробное описание

Заголовочный файл модуля server.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.11 server.h

См. документацию.

```
1
8 #pragma once
9 #include <iostream>
10 #include <string>
11 #include <vector>
12 #include <cstring>
13 #include <algorithm>
14 #include <memory>
15 #include <stdlib.h>
16 #include <unistd.h>
17 #include <arpa/inet.h>
18 #include <sys/socket.h>
19 #include <netdb.h>
20 #include <netinet/in.h>
21
22 #include <random>
23 #include <chrono>
24 #include <thread>
25 #include <limits>
26 #include "read_base.h"
27 #include "logger.h"
28 #include "error.h"
29 #include "ui.h"
30 #include <cryptopp/cryptlib.h>
31 #include <cryptopp/hex.h>
32 #include <cryptopp/files.h>
33 #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1
34 #include <cryptopp/md5.h>
35 #include <cryptopp/filters.h>
```

```
36 #include <cryptopp/osrng.h>
41 class server{
42     private:
46         struct sockaddr_in serverAddr, clientAddr;
50         socklen_t addr_size;
54         std::string base_location;
58         std::vector<std::string> cl_ids,cl_passes;
62         size_t buflen =1024;
66         std::unique_ptr <char[]> buffer{new char[buflen]};
70         uint port;
74         std::string digits[16] = {"0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"};
75     public:
79         logger log;
83         uint32_t nums;
87         int serverSocket, clientSocket;
91         std::string cl_id,log_location;
98         void connect_to_cl();
104         int client_auth();
110         std::string SALT_generate();
117         std::string convert_to_hex(uint64_t);
125         void send_data(std::string data,std::string msg);
132         std::string recv_data(std::string messg);
140         std::string hash_gen(std::string &salt,std::string &password);
144         void close_sock();
150         void start();
159         int64_t calculation(int64_t number1, int64_t number2);
165         int handle_data();
172         void work(UI &intf);
173 };
```

5.12 Файл ui.cpp

Исполняемый файл для модуля UI.

#include "ui.h"
#include <boost/program_options.hpp>
Граф включаемых заголовочных файлов для ui.cpp:

5.12.1 Подробное описание

Исполняемый файл для модуля UI.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.13 Файл ui.h

Заголовочный файл для модуля UI.

#include <boost/program_options.hpp>
#include <iostream>
#include <string>
#include <vector>
#include "error.h"
#include "logger.h"
Граф включаемых заголовочных файлов для ui.h: Граф файлов, в которые включается этот файл:

Классы

- class UI

  Класс пользовательского интерфейса

5.13.1 Подробное описание

Заголовочный файл для модуля UI.

Автор

Андреев И.В.

Версия

1.0

Дата

17.08.2024

Авторство

ИБСТ ПГУ

5.14 ui.h

См. документацию.

```
1
8 #pragma once
9 #include <boost/program_options.hpp>
10 #include <iostream>
11 #include <string>
12 #include <vector>
13
14 #include "error.h"
15 #include "logger.h"
16 namespace po = boost::program_options;
20 class UI{
21     public:
24         logger log;
27         po::options_description desc;
30         po::variables_map vm;
37         UI(int argc, char* argv[]);
44         uint get_port();
50         std::string get_base_loc();
56         std::string get_log_loc();
57 };
```

Предметный указатель