Курсовая работа

1.0

Создано системой Doxygen 1.9.1

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

Глава 4

Классы

4.1 Класс calc

Прием и обработка вектора чисел типа float.

#include <calc.h>

Открытые члены

- float countoper (const std::vector< float > &vec)
  Метод, производящий математические вычисления

4.1.1 Подробное описание

Прием и обработка вектора чисел типа float.

Производимая над числами операция - сумма квадратов

4.1.2 Методы

4.1.2.1 countoper()

```
float calc::countoper (
                const std::vector< float > & vec )
```

Метод, производящий математические вычисления

Принимает вектор чисел типа float и производит математическую операцию "сумма квадратов". Возвращает сумму квадратов полученных чисел

Аргументы

vec	Ссылка на вектор чисел типа float

Объявления и описания членов классов находятся в файлах:

- calc.h
- calc.cpp

4.2 Класс criticalerr

Класс для обработки критических ошибок

#include <error.h>

Граф наследования:criticalerr:

Граф связей класса criticalerr:

Открытые члены

- criticalerr (const std::string &s)
  Конструктор, принимающий константную ссылку на строку.

4.2.1 Подробное описание

Класс для обработки критических ошибок

Этот класс наследуется от std::runtime_error и используется для обработки ошибок и исключений.

4.2.2 Конструктор(ы)

4.2.2.1 criticalerr()

criticalerr::criticalerr (
const std::string & s) [inline]

Конструктор, принимающий константную ссылку на строку.

Аргументы

s	Константная ссылка на строку, сообщение об ошибке в виде строки.

Объявления и описания членов класса находятся в файле:

- error.h

4.3 Класс interface

Прием и обработка аргументов командной строки

#include <interface.h>

Открытые члены

- void get_args_of_comline (int argc, const char *argv[])
  Первичный прием аргументов командной строки
- std::tuple< std::string, std::string, std::string, std::string > check_args (int argc, const char *argv[])
  Проверка аргументов командной строки
- bool check_path_to_logfile (std::string pathtologfile)
  Проверка пути до лог-файла
- std::tuple< bool, std::map< std::string, std::string > > check_path_to_database_and_get_database (std::string pathtodatabase, logtxt *logger)
  Проверка пути до базы данных и формирование ассоциативного массива
- std::tuple< bool, int > check_port (std::string portstr, logtxt *logger)
  Проверка порта

4.3.1 Подробное описание

Прием и обработка аргументов командной строки

В private секцию устанавливается ассоциативный массив "логин : пароль" Устанавливает порт и пути до лог-файла и базы данных. Порождает объекты других классов и передает им параметры Для обработки ключей и их параметров используются отдельные функции

4.3.2 Методы

4.3.2.1 check_args()

```
std::tuple< std::string, std::string, std::string, std::string > interface::check_args (
            int argc,
            const char * argv[] )
```

Проверка аргументов командной строки

Проверяет введенные пользователем флаги и их параметры

Аргументы

argc	Количество аргументов командной строки
argv	Массив строк, содержащий аргументы командной строки

Исключения

criticalerr	при вводе несуществующего ключа или ключа без параметра

4.3.2.2 check_path_to_database_and_get_database()

```
std::tuple< bool, std::map< std::string, std::string > > interface::check_path_to_database_and_get_database (
        std::string pathtodatabase,
        logtxt * logger )
```

Проверка пути до базы данных и формирование ассоциативного массива

Проверяет существование пути до базы данных и формирует ассоциативный массив из логинов и паролей, устанавливает путь до базы данных

Аргументы

pathtodatabase	Путь до базы данных
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

criticalerr	при вводе несуществующего пути до базы данных
noncriticalerr	при неверном формате строки

4.3.2.3 check_path_to_logfile()

```
bool interface::check_path_to_logfile (
        std::string pathtologfile )
```

Проверка пути до лог-файла

Проверяет существование лог-файла по указанному пути, устанавливает путь до лог-файла

Аргументы

pathtologfile	Путь до лог-файла

Исключения

criticalerr	при вводе несуществующего пути до лог-файла

4.3.2.4 check_port()

```
std::tuple< bool, int > interface::check_port (
                std::string portstr,
                logtxt * logger )
```

Проверка порта

Проверяет введенный пользователем порт на корректность и приводит его к типу int, устанавливает порт

Аргументы

portstr	Строка, содержащая порт
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

criticalerr	при вводе пустого или некорректного порта

4.3.2.5 get_args_of_comline()

```
void interface::get_args_of_comline (
                int argc,
                const char * argv[] )
```

Первичный прием аргументов командной строки

Порождает все последующие методы и объекты других классов, а именно объект logger класса logtxt, объект startconnect класса server

Аргументы

argc	Количество аргументов командной строки
argv	Массив строк, содержащий аргументы командной строки

Объявления и описания членов классов находятся в файлах:

- interface.h
- interface.cpp

4.4 Класс logtxt

Прием и запись ошибок в лог-файл

#include <log.h>

Открытые члены

- void setpath (std::string path)
  Установка пути до лог-файла
- void writeerr (const std::string &error)
  Запись ошибки в лог-файл

4.4.1 Подробное описание

Прием и запись ошибок в лог-файл

В private секцию устанавливается путь до лог-файла

4.4.2 Методы

4.4.2.1 setpath()

void logtxt::setpath (
std::string path)

Установка пути до лог-файла

Устанавливает путь до лог-файла

Аргументы

path	Строка, представляющая собой путь до лог-файла

4.4.2.2 writeerr()

void logtxt::writeerr (
const std::string & error)

Запись ошибки в лог-файл

Вычисляет дату и время происхождения ошибки, форматирует их и записывает вместе с ошбкой в лог-файл

Аргументы

error	Константная ссылка на строку, представляющая собой ошибку

Объявления и описания членов классов находятся в файлах:

- log.h
- log.cpp

4.5 Класс noncriticalerr

Класс для обработки некритических ошибок

#include <error.h>

Граф наследования:noncriticalerr:

Граф связей класса noncriticalerr:

Открытые члены

- noncriticalerr (const std::string &s)
  Конструктор, принимающий константную ссылку на строку.

4.5.1 Подробное описание

Класс для обработки некритических ошибок

Этот класс наследуется от std::runtime_error и используется для обработки ошибок и исключений.

4.5.2 Конструктор(ы)

4.5.2.1 noncriticalerr()

```
noncriticalerr::noncriticalerr (
            const std::string & s )   [inline]
```

Конструктор, принимающий константную ссылку на строку.

Аргументы

s	Константная ссылка на строку, сообщение об ошибке в виде строки.

Объявления и описания членов класса находятся в файле:

- error.h

4.6 Класс server

Создание соединения для работы с клиентом

#include <server.h>

Открытые члены

- int connection (int port, std::map< std::string, std::string > base, logtxt *logger)
  Основной метод, порождающий другие методы
- std::tuple< bool, int, sockaddr_in > create_socket (int port, logtxt *logger)
  Создание сокета
- std::tuple< bool, sockaddr_in > bind_socket (int server_socket, sockaddr_in server_addr, logtxt *logger)
  Привязка сокета
- std::tuple< bool, sockaddr_in > listen_socket (int port, int server_socket, sockaddr_in server_addr, logtxt *logger)
  Прослушивание сокета
- void threadclient (int client_socket, std::map< std::string, std::string > base, logtxt *logger)
  Метод, отвечающий за взаимодействие с клиентом
- std::string sha256 (const std::string &message)
  Хеширование пароля

4.6.1 Подробное описание

Создание соединения для работы с клиентом

Создает сокет для взимодействия с клиентом В private секции находится объект cout_mutex типа mutex

4.6.2 Методы

4.6.2.1 bind_socket()

```
std::tuple< bool, sockaddr_in > server::bind_socket (
        int server_socket,
        sockaddr_in server_addr,
        logtxt * logger )
```

Привязка сокета

Привязывает сокет к адресу и порту, на которых он будет работать

Аргументы

server_socket	Дескриптор сокета, использующийся для прослушивания входящих соединений
server_addr	Адрес сокета
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

criticalerr	при ошибке привязки сокета

4.6.2.2 connection()

```
int server::connection (
                int port,
                std::map< std::string, std::string > base,
                logtxt * logger )
```

Основной метод, порождающий другие методы

Принимает параметры, которые будут передаваться другим методам, порождает методы по созданию, привязке и прослушиванию сокета. Запускает цикл, в котором принимает клиентов.

Аргументы

port	Порт, на котором будет запускаться сервер
base	Ассоциативный массив с логинами и паролями
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

noncriticalerr	при ошибке подкдючения клиента

4.6.2.3 create_socket()

```
std::tuple< bool, int, sockaddr_in > server::create_socket (
                int port,
                logtxt * logger )
```

Создание сокета

Создает сокет, который в дальнейшем будет использоваться для работы с клиентом

Аргументы

port	Порт, на котором будет запускаться сервер
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

criticalerr	при ошибке создания сокета

4.6.2.4 listen_socket()

```
std::tuple< bool, sockaddr_in > server::listen_socket (
                int port,
                int server_socket,
                sockaddr_in server_addr,
                logtxt * logger )
```

Прослушивание сокета

Прослушивает сервер

Аргументы

port	Порт, на котором запускается сервер
server_socket	Дескриптор сокета, использующийся для прослушивания входящих соединений
server_addr	Адрес сокета
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

criticalerr	при ошибке прослушивания сокета

4.6.2.5 sha256()

```
std::string server::sha256 (
                const std::string & message )
```

Хеширование пароля

Хэширует строку, содержащую соль и пароль

Аргументы

message	Константная ссылка на строку, содержащую сообщение из соли и пароля

4.6.2.6 threadclient()

```
void server::threadclient (
                int client_socket,
                std::map< std::string, std::string > base,
                logtxt * logger )
```

Метод, отвечающий за взаимодействие с клиентом

Данный метод выполняет следующие действия: принимает от клиента строку, содержащую логин, соль и хэш; передает соль и пароль из базы данных в метод хэшироавния; сравнивает захэшированный пароль с хэшем от клиента; получает количество, длину и содержимое векторов; порождает объект calculator класса calc и передает ему векторы; отправляет клиенту посчитанный результат. После 30 секунд метод закрывает сокет

Аргументы

client_socket	Дескриптор сокета, представляющий собой соединение с конкретным клиентом
base	Ассоциативный массив с логинами и паролями
logger	Указатель на объект класса logtxt, необходимый для записи ошибок в лог-файл

Исключения

noncriticalerr	при следующих ошибках: Ошибка получения сообщения от клиента; Неверная длина сообщения; Ошибка отправки "ERR" при неизвестном логине; Неизвестный логин; Ошибка отправки "ERR" при ошибке авторизации; Ошибка авторизации для логина; Ошибка отправки "OK"; Ошибка получения количества векторов; Ошибка получения длины n-го вектора; Ошибка получения n-го вектора; Ошибка отправки n-го результата

Объявления и описания членов классов находятся в файлах:

- server.h
- server.cpp

Глава 5

Файлы

5.1 Файл calc.h

Заголовочный файл для модуля calc.

#include <vector>
Граф включаемых заголовочных файлов для calc.h:

5.2 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 criticalerr

  Класс для обработки критических ошибок
- class noncriticalerr

  Класс для обработки некритических ошибок

5.2.1 Подробное описание

Заголовочный файл для модуля error.

Автор

Невзоров Т.В.

Версия

1.0

Дата

20.12.2024

Авторство

ИБСТ ПГУ

5.3 Файл interface.h

Заголовочный файл для модуля interface.

#include "log.h"
#include "error.h"
#include <filesystem>
#include <string>
#include <map>
#include <tuple>
Граф включаемых заголовочных файлов для interface.h:

Классы

- class interface

  Прием и обработка аргументов командной строки

5.3.1 Подробное описание

Заголовочный файл для модуля interface.

Автор

Невзоров Т.В.

Версия

1.0

Дата

20.12.2024

Авторство

ИБСТ ПГУ

5.4 Файл log.h

Заголовочный файл для модуля log.

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

Классы

- class logtxt

  Прием и запись ошибок в лог-файл

5.4.1 Подробное описание

Заголовочный файл для модуля log.

Автор

Невзоров Т.В.

Версия

1.0

Дата

20.12.2024

Авторство

ИБСТ ПГУ

5.5 Файл server.h

Заголовочный файл для модуля server.

```
#include <string>
#include <map>
#include <mutex>
#include <tuple>
#include <sstream>
#include <thread>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <cryptopp/sha.h>
#include <cryptopp/hex.h>
#include "log.h"
```

Граф включаемых заголовочных файлов для server.h:

Классы

- class server

  Создание соединения для работы с клиентом

5.5.1 Подробное описание

Заголовочный файл для модуля server.

Автор

Невзоров Т.В.

Версия

1.0

Дата

20.12.2024

Авторство

ИБСТ ПГУ

Предметный указатель