Курсовая работа по дисциплине Технологии и методы программирования

Создано системой Doxygen 1.9.4

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 ClientHandler

Класс для обработки запросов от клиента.

#include <ClientHandler.h>

Открытые члены

- void handleRequest (int socket, const std::string &dbFile, const std::string &logFile)
  Обработка запроса от клиента.

3.1.1 Подробное описание

Класс для обработки запросов от клиента.

Этот класс предназначен для обработки запросов от клиентов через сокеты, включая аутентификацию, обработку векторов и отправку результатов обратно клиенту.

Основные функции класса включают:

- аутентификацию пользователя,
- обработку векторов, включая проверку на правильность данных,
- отправку результатов клиенту.

3.1.2 Методы

3.1.2.1 handleRequest()

```
void ClientHandler::handleRequest (
            int socket,
            const std::string & dbFile,
            const std::string & logFile )
```

Обработка запроса от клиента.

Класс для обработки запросов от клиентов.

Этот метод выполняет все необходимые шаги для обработки запроса от клиента, включая аутентификацию пользователя, прием и обработку векторов, а также отправку результатов обратно клиенту.

Аргументы

socket	Дескриптор сокета для связи с клиентом.
dbFile	Путь к файлу базы данных для аутентификации пользователя.
logFile	Путь к файлу для записи логов.

Этот класс обрабатывает запросы клиентов по определенному сокету, включая аутентификацию пользователя, прием и обработку векторов, а также отправку результатов обратно клиенту.

Обработка запроса от клиента.

Этот метод обрабатывает запрос клиента, включая аутентификацию, прием векторов и выполнение операции. Он также управляет обменом данными с клиентом через сокет.

Аргументы

socket	Дескриптор сокета для связи с клиентом.
dbFile	Путь к файлу базы данных для аутентификации пользователя.
logFile	Путь к файлу для записи логов.

< Буфер для приема данных от клиента.

< Преобразование полученных данных в строку.

< Отправка успешного ответа клиенту.

< Отправка ошибки аутентификации клиенту.

< Закрытие сокета после неудачной аутентификации.

< Количество векторов для обработки.

< Экземпляр класса для обработки векторов.

< Размер текущего вектора.

< Ограничение на размер вектора.

< Отправка ошибки о слишком большом векторе.

< Вектор для хранения значений.

< Отправка ошибки о пустом векторе.

< Вывод значений вектора.

< Отправка результата клиенту.

Объявления и описания членов классов находятся в файлах:

- ClientHandler.h
- ClientHandler.cpp

3.2 Класс DatabaseConnector

Класс для взаимодействия с базой данных пользователей.

#include <DatabaseConnector.h>

Открытые члены

- bool verifyUser (const std::string &login, const std::string &salt, const std::string &hash, const std::string &dbPath)

  Проверка аутентификации пользователя.

3.2.1 Подробное описание

Класс для взаимодействия с базой данных пользователей.

Класс предоставляет методы для аутентификации пользователей, проверяя их логин и хэш пароля. Аутентификация выполняется с использованием соли и хэширования пароля.

3.2.2 Методы

3.2.2.1 verifyUser()

```
bool DatabaseConnector::verifyUser (
                const std::string & login,
                const std::string & salt,
                const std::string & hash,
                const std::string & dbPath )
```

Проверка аутентификации пользователя.

Проверяет пользователя в базе данных.

Метод проверяет, существует ли пользователь в базе данных, и совпадает ли хэш пароля с сохраненным хэшем в базе данных. Хэширование пароля выполняется с использованием MD5 и соли.

Аргументы

login	Логин пользователя.
salt	Соль, используемая при хэшировании пароля.
hash	Хэш пароля, который передан для проверки.
dbPath	Путь к файлу базы данных.

Возвращает

Возвращает true, если пользователь найден и хэш пароля совпадает, иначе false.

Метод проверяет логин и хеш пароля с солью в базе данных. Сначала из базы данных извлекаются логин и хеш пароля, после чего вычисляется хеш пароля клиента с использованием соли и сравнивается с хешом из базы данных.

Аргументы

login	Логин пользователя для проверки.
salt	Соль, используемая для хеширования пароля.
hash	Хеш пароля, полученный от клиента.
dbPath	Путь к файлу базы данных, содержащей логины и пароли.

Возвращает

Возвращает true, если логин и хеш пароля совпадают с данными в базе данных, иначе возвращает false.

Объявления и описания членов классов находятся в файлах:

- DatabaseConnector.h
- DatabaseConnector.cpp

3.3 Класс Log

Класс для записи ошибок в файл лога и вывода их в консоль.

#include <Log.h>

Открытые статические члены

- static void recordError (const std::string &message, bool critical=false)
  Записывает сообщение об ошибке в лог и выводит в консоль.

Статические открытые данные

- static std::string logFile = "server_log.txt"
  Путь к файлу логов.

3.3.1 Подробное описание

Класс для записи ошибок в файл лога и вывода их в консоль.

Этот класс предоставляет методы для записи сообщений об ошибках в файл и консоль, а также поддерживает возможность указания критичности ошибки.

3.3.2 Методы

3.3.2.1 recordError()

```
void Log::recordError (
                const std::string & message,
                bool critical = false )   [static]
```

Записывает сообщение об ошибке в лог и выводит в консоль.

Записывает сообщение об ошибке в лог-файл и выводит в консоль.

Этот метод записывает переданное сообщение в файл лога с отметкой времени и выводит его в консоль. Также можно указать, является ли ошибка критической (по умолчанию — нет).

Аргументы

message	Сообщение, которое будет записано в лог.
critical	Флаг критичности ошибки. По умолчанию — false.

Метод записывает сообщение в файл лога, а также выводит его в консоль. При этом добавляется отметка времени и указание на критичность сообщения.

Аргументы

message	Сообщение, которое необходимо записать в лог.
critical	Флаг, указывающий на критичность ошибки. Если true, будет указано, что ошибка критична.

< Открытие файла для записи в конец

< Проверка на успешное открытие файла

< Получаем текущую дату и время

< Записываем сообщение в лог

< Выводим сообщение на консоль

3.3.3 Данные класса

3.3.3.1 logFile

std::string Log::logFile = "server_log.txt" [static]

Путь к файлу логов.

Класс для записи логов в файл и вывода ошибок в консоль.

Это статический член класса, который определяет путь к файлу, куда будут записываться логи. Значение по умолчанию — "server_log.txt". Путь к файлу логов

Класс позволяет записывать сообщения об ошибках в файл и выводить их в консоль. Также поддерживается отметка времени для каждого сообщения. Путь к файлу логов (по умолчанию "server↩ _log.txt")

Объявления и описания членов классов находятся в файлах:

- Log.h
- Log.cpp

3.4 Класс VectorProcessor

Класс для обработки векторов целых чисел типа int16_t.

#include <VectorProcessor.h>

Открытые члены

- int16_t computeProduct (const std::vector< int16_t > &vector)
  Вычисляет произведение всех элементов вектора.

3.4.1 Подробное описание

Класс для обработки векторов целых чисел типа int16_t.

Этот класс содержит методы для работы с векторами целых чисел типа int16_t. Он предоставляет функциональность для вычисления произведения всех элементов вектора с учетом переполнений.

3.4.2 Методы

3.4.2.1 computeProduct()

int16_t VectorProcessor::computeProduct (
const std::vector< int16_t > & vector)

Вычисляет произведение всех элементов вектора.

Вычисляет произведение элементов вектора.

Метод принимает вектор целых чисел типа int16_t и вычисляет его произведение. Если в процессе вычисления происходит переполнение (значение выходит за пределы диапазона int16_t), возвращается максимальное или минимальное значение для типа int16_t.

Аргументы

vector	Вектор целых чисел типа int16_t, элементы которого необходимо перемножить.

Возвращает

Возвращает результат произведения элементов вектора, приведенный к типу int16_t. В случае переполнения возвращается максимально возможное значение для типа int16_t или минимально возможное, в зависимости от направления переполнения.

Этот метод принимает вектор целых чисел типа int16_t и вычисляет его произведение. Если в процессе вычисления происходит переполнение (значение выходит за пределы диапазона int16_t), возвращается максимальное или минимальное значение для типа int16_t.

Аргументы

vector	Вектор целых чисел типа int16_t, элементы которого необходимо перемножить.

Возвращает

Возвращает результат произведения элементов вектора, приведенный к типу int16_t. В случае переполнения возвращается максимально возможное значение для типа int16_t или минимально возможное, в зависимости от направления переполнения.

< Переменная для хранения произведения элементов вектора.

< Умножение текущего элемента на произведение.

< Возврат максимального значения, если переполнение вверх.

< Возврат минимального значения, если переполнение вниз.

< Приведение результата к типу int16_t и возврат.

Объявления и описания членов классов находятся в файлах:

- VectorProcessor.h
- VectorProcessor.cpp

Глава 4

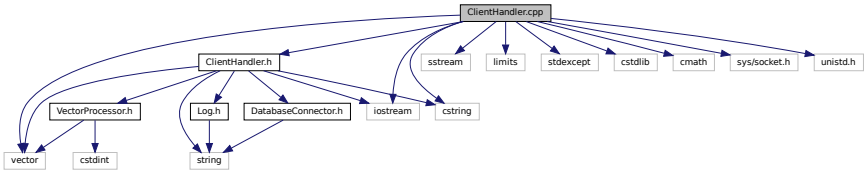
Файлы

4.1 Файл ClientHandler.cpp

Реализация обработки запросов от клиентов, включая аутентификацию, обработку векторов и отправку результатов.

```
#include "ClientHandler.h"
#include <iostream>
#include <vector>
#include <sstream>
#include <limits>
#include <stdexcept>
#include <cstring>
#include <cstdlib>
#include <cmath>
#include <sys/socket.h>
#include <unistd.h>
```

Граф включаемых заголовочных файлов для ClientHandler.cpp:



4.1.1 Подробное описание

Реализация обработки запросов от клиентов, включая аутентификацию, обработку векторов и отправку результатов.

Автор

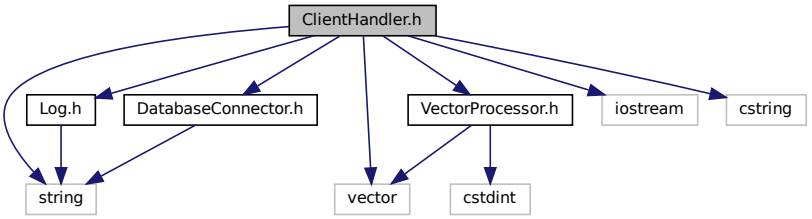
Тришкин В.Д.

4.2 Файл ClientHandler.h

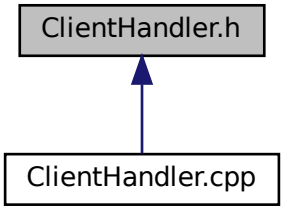
Заголовочный файл для класса ClientHandler, который обрабатывает запросы от клиента.

```
#include <string>
#include <vector>
#include "Log.h"
#include "DatabaseConnector.h"
#include "VectorProcessor.h"
#include <iostream>
#include <cstring>
```

Граф включаемых заголовочных файлов для ClientHandler.h:



Граф файлов, в которые включается этот файл:



Классы

- class ClientHandler

  Класс для обработки запросов от клиента.

4.2.1 Подробное описание

Заголовочный файл для класса ClientHandler, который обрабатывает запросы от клиента.

Автор

Тришкин В.Д.

4.3 ClientHandler.h

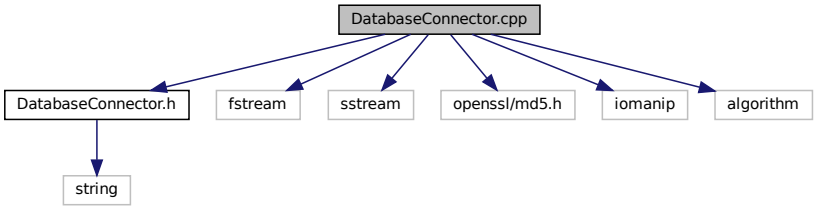
См. документацию.

```
1 #ifndef CLIENTHANDLER_H
2 #define CLIENTHANDLER_H
3
4 #include <string>
5 #include <vector>
6 #include "Log.h"
7 #include "DatabaseConnector.h"
8 #include "VectorProcessor.h"
9 #include <iostream>
10 #include <cstring>
11
29 class ClientHandler {
30 public:
41     void handleRequest(int socket, const std::string& dbFile, const std::string& logFile);
42
43 private:
53     bool validateVectorType(const std::vector<int16_t>& vec);
54 };
55
56 #endif // CLIENTHANDLER_H
57
```

4.4 Файл DatabaseConnector.cpp

Реализация класса DatabaseConnector для работы с базой данных.

#include "DatabaseConnector.h"
#include <fstream>
#include <sstream>
#include <openssl/md5.h>
#include <iomanip>
#include <algorithm>
Граф включаемых заголовочных файлов для DatabaseConnector.cpp:



4.4.1 Подробное описание

Реализация класса DatabaseConnector для работы с базой данных.

Автор

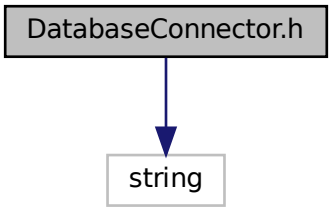
Тришкин В.Д.

Этот файл содержит реализацию методов для работы с базой данных пользователей, включая проверку пользователя по логину, соли и хешу пароля.

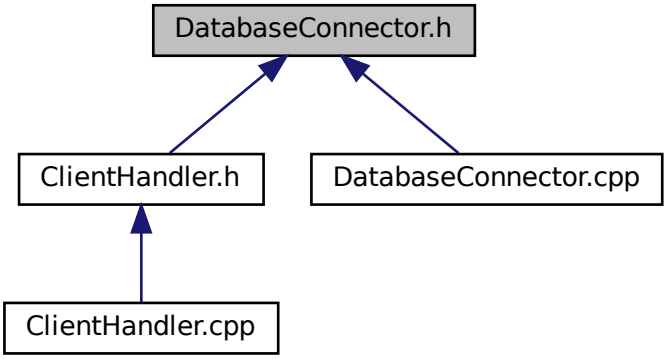
4.5 Файл DatabaseConnector.h

Заголовочный файл для работы с базой данных пользователей.

#include <string>
Граф включаемых заголовочных файлов для DatabaseConnector.h:



Граф файлов, в которые включается этот файл:



Классы

- class DatabaseConnector

  Класс для взаимодействия с базой данных пользователей.

4.5.1 Подробное описание

Заголовочный файл для работы с базой данных пользователей.

Автор

Тришкин В.Д.

4.6 DatabaseConnector.h

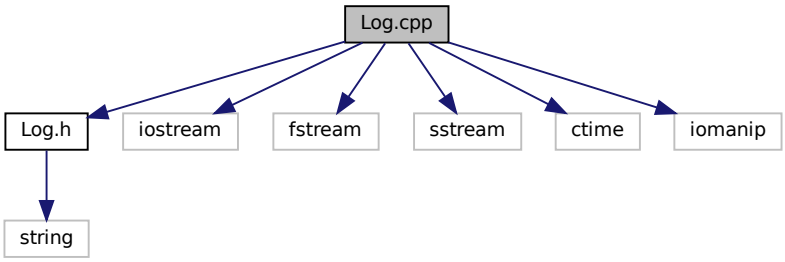
См. документацию.

```
1 #ifndef DATABASECONNECTOR_H
2 #define DATABASECONNECTOR_H
3
4 #include <string>
5
18 class DatabaseConnector {
19 public:
32     bool verifyUser(const std::string& login, const std::string& salt, const std::string& hash, const std::string& dbPath);
33
34 private:
44     std::string generateHash(const std::string& password, const std::string& salt);
45
55     bool compareHashes(const std::string& serverHash, const std::string& clientHash);
56 };
57
58 #endif // DATABASECONNECTOR_H
59
```

4.7 Файл Log.cpp

Реализация логирования сообщений об ошибках.

#include "Log.h"
#include <iostream>
#include <fstream>
#include <sstream>
#include <ctime>
#include <iomanip>
Граф включаемых заголовочных файлов для Log.cpp:



4.7.1 Подробное описание

Реализация логирования сообщений об ошибках.

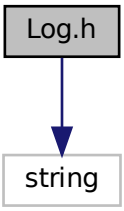
Автор

Тришкин В.Д.

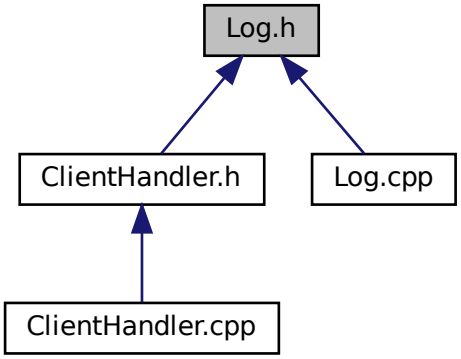
4.8 Файл Log.h

Интерфейс класса для логирования ошибок.

#include <string>
Граф включаемых заголовочных файлов для Log.h:



Граф файлов, в которые включается этот файл:



Классы

- class Log

  Класс для записи ошибок в файл лога и вывода их в консоль.

4.8.1 Подробное описание

Интерфейс класса для логирования ошибок.

Автор

Тришкин В.Д.

4.9 Log.h

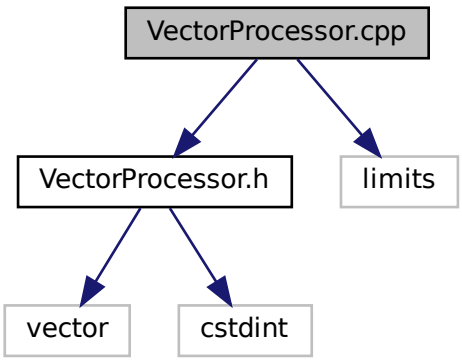
См. документацию.

```
1 #ifndef LOG_H
2 #define LOG_H
3
4 #include <string>
5
18 class Log {
19 public:
26     static std::string logFile;
27
37     static void recordError(const std::string& message, bool critical = false);
38
39 private:
47     static std::string getCurrentTime();
48 };
49
50 #endif // LOG_H
51
```

4.10 Файл VectorProcessor.cpp

Реализация класса VectorProcessor для обработки векторов целых чисел.

#include "VectorProcessor.h"
#include <limits>
Граф включаемых заголовочных файлов для VectorProcessor.cpp:



4.10.1 Подробное описание

Реализация класса VectorProcessor для обработки векторов целых чисел.

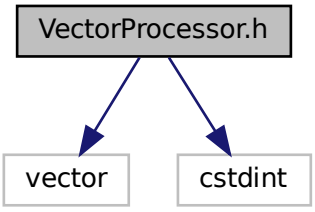
Автор

Тришкин В.Д.

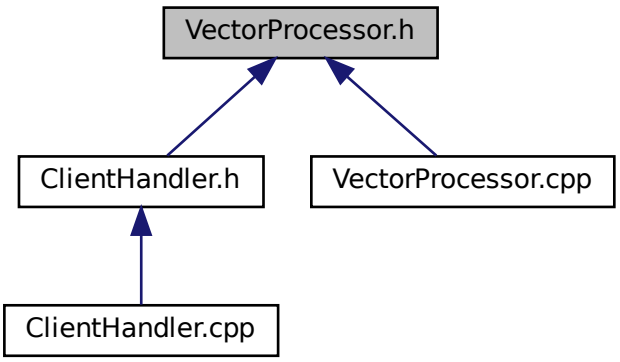
4.11 Файл VectorProcessor.h

Заголовочный файл для класса VectorProcessor.

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



Граф файлов, в которые включается этот файл:



Классы

- class VectorProcessor

  Класс для обработки векторов целых чисел типа int16_t.

4.11.1 Подробное описание

Заголовочный файл для класса VectorProcessor.

Автор

Тришкин В.Д.

Класс VectorProcessor предоставляет методы для обработки векторов целых чисел типа int16_t, в частности, для вычисления произведения всех элементов вектора.

4.12 VectorProcessor.h

См. документацию.

```
1 #ifndef VECTORPROCESSOR_H
2 #define VECTORPROCESSOR_H
3
4 #include <vector>
5 #include <cstdint>
6
20 class VectorProcessor {
21 public:
34     int16_t computeProduct(const std::vector<int16_t>& vector);
35 };
36
37 #endif // VECTORPROCESSOR_H
38
```

Предметный указатель