# Курсовая работа. Программная реализация сетевого сервера.

1.0

Создано системой Doxygen 1.9.1

# 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 Auth

Класс для аутентификации клиента на сервере

#include <Auth.h>

### Открытые члены

- Auth (std::string ID, std::string pass)
  Конструктор для установки идентификатора и пароля клиента
- void GenSALT ()
  Генерация случайной соли для вычисления хэша
- bool CompareHashes (std::string ClientHash)
  Сравнение хэша, присылаемого клиентом и хэша, вычисляемого внутри метода
- std::string getSALT ()
- std::string getId ()
- std::string getpass ()
- std::string getstrHash ()

### Открытые атрибуты

- char ERRmsg [3] = {'E', 'R','R'}
  Сообщение, отсылаемое клиенту при ошибке его обрабоки
- char OKmsg [2] = {'O', 'K'}
  Сообщение, отсылаемое клиенту при успешной авторизации

### Закрытые данные

- std::string SALT
  Соль для вычисления хэша
- std::string Id
  Идентификатор клиента
- std::string password
  Пароль клиента
- std::string strHash
  Хэш в виде шестнадцатиричных цифр

### 4.1.1 Подробное описание

Класс для аутентификации клиента на сервере

### 4.1.2 Конструктор(ы)

#### 4.1.2.1 Auth() Auth::Auth ( std::string ID, std::string pass )

Конструктор для установки идентификатора и пароля клиента

Аргументы

| | | |
|---|---|---|
| in | ID,идентификатор | клиента, std::string. |
| in | pass,пароль | клиента, std::string. |

### 4.1.3 Методы

#### 4.1.3.1 CompareHashes() bool Auth::CompareHashes ( std::string ClientHash )

Сравнение хэша, присылаемого клиентом и хэша, вычисляемого внутри метода

Вычисляет MD5 хэш от строки SALT+password и сравнивает его с хэшем, который присылает клиент

Аргументы

| | | |
|---|---|---|
| in | ClientHash,хэш | клиента, std::string |

Возвращает

bool, если хэши совпадают - true, иначе false

Исключения

| | |
|---|---|
| std::server_error | в случае несовпадения хэшей, штатная type = invalid_argument, what ="Invalid hash" |

#### 4.1.3.2 GenSALT() void Auth::GenSALT ( )

Генерация случайной соли для вычисления хэша

Соль - 64-х разрядное число, представленное в виде строки из 16-ти шестнадцатиричных цифр

Объявления и описания членов классов находятся в файлах:

- Auth.h
- Auth.cpp

## 4.2 Класс Counter

Класс для вычислений по вектору

#include <Counter.h>

Открытые члены

- int16_t * mean (std::vector< int16_t > arr)
  Конструктор без параметров

Закрытые данные

- int16_t result [1]
  Результат вычислений

### 4.2.1 Подробное описание

Класс для вычислений по вектору

### 4.2.2 Методы

#### 4.2.2.1 mean() int16_t* Counter::mean ( std::vector< int16_t > arr ) [inline]

Конструктор без параметров

Вычисляет среднее арифметическое по вектору

Аргументы

| | | |
|---|---|---|
| in | arr,вектор,std::vector<int16_t> | |

Возвращает

указатель на массив с результатом, int16_t *

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, критическая<br>type = invalid_argument, what ="Count Error" |

Объявления и описания членов класса находятся в файле:

- Counter.h

## 4.3 Класс DB

Класс для работы с базой данных пользователей

#include <DataBase.h>

Открытые члены

- DB (std::string DBName)
  Конструктор, в котором считывается база данных и сохраняется в словарь
- bool IDcheck (std::string login)
  Проверка наличия идентификатора клиента в базе данных

Открытые атрибуты

- std::map< std::string, std::string > DataBaseP
  Словарь с парами идентификатор:пароль

Закрытые данные

- char sep = ' '
  Разделитель идентификатора и пароля в базе данных

### 4.3.1 Подробное описание

Класс для работы с базой данных пользователей

### 4.3.2 Конструктор(ы)

#### 4.3.2.1 DB() DB::DB ( std::string DBName )

Конструктор, в котором считывается база данных и сохраняется в словарь

Аргументы

| | | |
|---|---|---|
| in | DBName,путь | к файлу с базой данных, std::string. |

Исключения

| | |
|---|---|
| std::server_error | в случае проблем с файлом базы данных, критическая |

### 4.3.3 Методы

#### 4.3.3.1 IDcheck() bool DB::IDcheck ( std::string login )

Проверка наличия идентификатора клиента в базе данных

Аргументы

| | | |
|---|---|---|
| in | login,идентификатора | клиента, std::string |

Возвращает

bool, если идентификатор есть в базе - true, иначе false

Исключения

| | |
|---|---|
| std::server_error | в случае отсутствия идентификатора в базе, штатная type = invalid_argument, what ="Invalid ID" |

Объявления и описания членов классов находятся в файлах:

- DataBase.h
- DataBase.cpp

## 4.4 Класс ErrorTracker

Класс для обработки ошибок

#include <ErrorTracker.h>

Открытые члены

- void setLogName (std::string LogName)
  Конструктор без параметров
- void write_log (std::string what, bool Critical)
  Запись ошибки в лог

Закрытые данные

- std::string LogFileName

  Путь к файлу с логом ошибок

### 4.4.1 Подробное описание

Класс для обработки ошибок

### 4.4.2 Методы

#### 4.4.2.1 setLogName() void ErrorTracker::setLogName ( std::string LogName )

Конструктор без параметров

Функция, устанавливающая путь к файлу с логом ошибок

#### 4.4.2.2 write_log() void ErrorTracker::write_log ( std::string what, bool Critical )

Запись ошибки в лог

Записывает время, тип и критичность ошибки

Аргументы

| | | |
|---|---|---|
| in | what,тип | ошибки, std::string |
| in | Critical,критичность | ошибки (Критическая - true, Штатная - false), std::string |

Объявления и описания членов классов находятся в файлах:

- ErrorTracker.h
- ErrorTracker.cpp

## 4.5 Класс Opts

Класс для получения параметров коммандной строки

#include <interface.h>

Открытые члены

- Opts (int argc, char **argv)

  Конструктор, внутри которого считываются параметры командной строки
- bool CheckFiles ()

  Проверка роботоспособности файлов базы данных и лога
- string getDataBaseName ()
- string getLogFileName ()
- int getPort ()

Закрытые члены

- void usage (const char *progName)

  вывод подсказки и останов

Закрытые данные

- string DataBaseName = "DB.txt"

  Путь к файлу с базой данных
- string LogFileName = "log.txt"

  Путь к файлу для записи логов
- int Port = 33333

  Порт, на котором работает сервер

### 4.5.1 Подробное описание

Класс для получения параметров коммандной строки

### 4.5.2 Конструктор(ы)

#### 4.5.2.1 Opts() Opts::Opts ( int argc, char ** argv )

Конструктор, внутри которого считываются параметры командной строки

Параметры командной строки: 1)-b Путь к файлу с базой данных, необязательный 2)-l Путь к файлу для записи логов, необязательный 3)-p Порт, на котором работает сервер, необязательный 4)-h вызов подсказки При ошибках в параметрах вызывается справка и программа завершает работу

Аргументы

| | | |
|---|---|---|
| in | int | argc |
| in | char | **argv |

### 4.5.3 Методы

#### 4.5.3.1 CheckFiles() bool Opts::CheckFiles ( )

Проверка роботоспособности файлов базы данных и лога

Возвращает

bool, если нет ошибок в фалах - true, иначе false

Исключения

| | |
|---|---|
| std::invalid_argument | в случае проблем с файлами, критическая<br>type = invalid_argument, what ="Wrong DB File Name" или what ="Wrong Log File Name" |

Объявления и описания членов классов находятся в файлах:

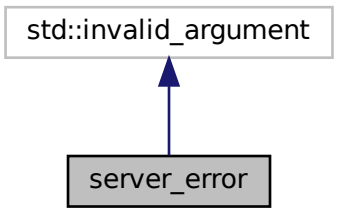
- interface.h
- interface.cpp

## 4.6 Класс server_error

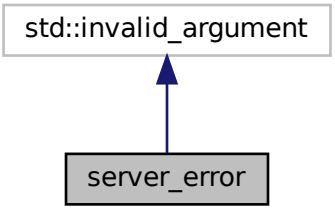
Класс ошибок

#include <ErrorTracker.h>

Граф наследования:server_error:



Граф связей класса server_error:




Открытые члены

- server_error (const std::string &what_arg, bool critical=false)
  Конструктор ошибок с строкой в качестве параметра
- server_error (const char *what_arg, bool critical=false)
  Конструктор ошибок с си-строкой в качестве параметра
- bool getState () const
  Возвращает статус критичности ошибки

Закрытые данные

- bool State = false
  Статус критичности ошибки

### 4.6.1 Подробное описание

Класс ошибок

Наследует от класса std::invalid_argument

### 4.6.2 Конструктор(ы)

#### 4.6.2.1 server_error() [1/2]

server_error::server_error (
const std::string & what_arg,
bool critical = false ) [inline], [explicit]

Конструктор ошибок с строкой в качестве параметра

Аргументы

| | | |
|---|---|---|
| in | what_arg,тип | ошибки, const std::string. |
| in | critical,критическа | ошибка - true, штатная - false, bool |

#### 4.6.2.2 server_error() [2/2] server_error::server_error ( const char * what_arg, bool critical = false ) [inline], [explicit]

Конструктор ошибок с си-строкой в качестве параметра

Аргументы

| | | |
|---|---|---|
| in | what_arg,тип | ошибки, const char*. |
| in | critical,критическа | ошибка - true, штатная - false, bool |

Объявления и описания членов класса находятся в файле:

- ErrorTracker.h

## 4.7 Класс WebManager

Класс, обеспечивающий работу с сокетами и сетовое взаимодействие

#include <WebManager.h>

Открытые члены

- WebManager (unsigned int port)
  Конструктор
- void start_listening ()
  Установка сокета в режим ожидания
- int new_bind ()
  Привязка сокета к адресу
- int accepting ()
  Приём соединения
- int receiving (int sock, void *buf, int size)
  Приём данных
- void sending (int sock, void *buf, int sizeb)
  Отправка данных

Закрытые данные

- unsigned int Port
  Порт, на котором работает сервер
- const char * Adress = "127.0.0.1"
  Сетевой адрес сервера
- struct sockaddr_in addr
  Структура sockaddr_in.
- int listener
  Основной сокет

### 4.7.1 Подробное описание

Класс, обеспечивающий работу с сокетами и сетовое взаимодействие

### 4.7.2 Конструктор(ы)

#### 4.7.2.1 WebManager() WebManager::WebManager ( unsigned int port )

Конструктор

Устанавливает порт, инициализирует основной сокет и структуру sockaddr_in

Аргументы

| | | |
|---|---|---|
| in | port,порт,на | котором работает сервер, int. |

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, критическая<br>type = invalid_argument, what ="Socket creation error" |

### 4.7.3 Методы

#### 4.7.3.1 accepting() int WebManager::accepting ( )

Приём соединения

Возвращает

код сокета, int

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, штатная<br>type = invalid_argument, what ="Accepting error" |

#### 4.7.3.2 new_bind() int WebManager::new_bind ( )

Привязка сокета к адресу

Возвращает

код сокета, int

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, критическая<br>type = invalid_argument, what ="Socket bind error" |

#### 4.7.3.3 receiving()

```
int WebManager::receiving (
        int sock,
        void * buf,
        int size )
```

Приём данных

Аргументы

| | | |
|---|---|---|
| in | sock,сокет,int | |
| in | buf,буфер | для данных, void* |
| in | size,размер | буфера, int |

Возвращает

количество полученных байт, int

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, штатная<br>type = invalid_argument, what ="Receiving error" |

#### 4.7.3.4 sending()

```
void WebManager::sending (
        int sock,
        void * buf,
        int sizeb )
```

Отправка данных

Аргументы

| | | |
|---|---|---|
| in | sock,сокет,int | |
| in | buf,буфер | с данными, void* |
| in | sizeb,количество | отправляемых байт, int |

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, штатная<br>type = invalid_argument, what ="Sending error" |

#### 4.7.3.5 start_listening() void WebManager::start_listening ( )

Установка сокета в режим ожидания

Исключения

| | |
|---|---|
| std::server_error | в случае ошибки, критическая<br>type = invalid_argument, what ="Listening error" |

Объявления и описания членов классов находятся в файлах:

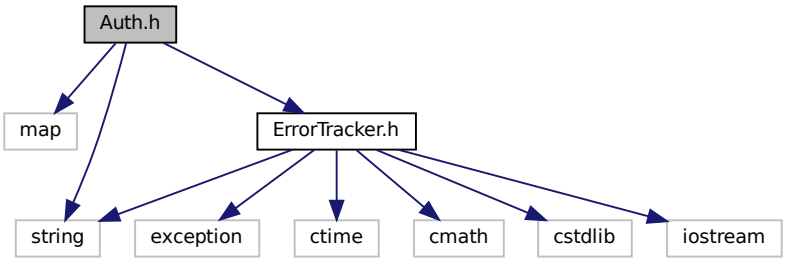
- WebManager.h
- WebManager.cpp

# 5 Файлы

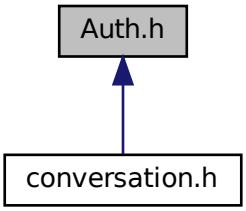
## 5.1 Файл Auth.h

Класс для аутентификации клиента на сервере

#include <map>
#include <string>
#include "ErrorTracker.h"
Граф включаемых заголовочных файлов для Auth.h:



Граф файлов, в которые включается этот файл:




Классы

- class Auth

  Класс для аутентификации клиента на сервере

### 5.1.1 Подробное описание

Класс для аутентификации клиента на сервере

Автор

Соколенко Н.С.

Версия

1.0

Дата

18.12.2022

Авторство

ИБСТ ПГУ

# Предметный указатель