SERVER

Создано системой Doxygen 1.9.1

1 Иерархический список классов		1
1.1 Иерархия классов		 1
2 Алфавитный указатель классов		3
2.1 Классы		 3
		_
3 Классы		5
3.1 Knacc authorization		5
3.1.1 Подробное описание		5
3.1.2 Конструктор(ы)		6
3.1.2.1 authorization()		6
3.1.3 Методы		6
$3.1.3.1 \text{ compare_hash}() \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots$		6
$3.1.3.2 \text{ generate_salt}() \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots$		6
3.2 Класс base		7
3.2.1 Подробное описание		7
3.2.2 Конструктор(ы)	•	 7
3.2.2.1 base()		 7
3.2.3 Методы		 8
$3.2.3.1 \mathrm{check_user}() \ldots \ldots \ldots \ldots \ldots \ldots \ldots$		 8
3.3 Класс calc		 8
3.3.1 Подробное описание		 8
3.3.2 Методы		 8
$3.3.2.1 \text{ calculation}() \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots$		 8
3.4 Класс connection		 9
3.4.1 Подробное описание		 9
3.4.2 Конструктор(ы)		 9
$3.4.2.1 \text{ connection}() \dots \dots$		 9
3.4.3 Методы		 10
3.4.3.1 Recv()		10
$3.4.3.2 \text{ Send}() \dots \dots$		 10
3.5 Класс error_processing		 11
3.5.1 Подробное описание		 11
3.5.2 Методы		11
3.5.2.1 write log()		11
3.6 Класс name error		 11
- 3.6.1 Подробное описание		12
3.6.2 Конструктор(ы)		12
3.6.2.1 name error() [1/2]		13
$3.6.2.2 \text{ name_error}() [2/2] \dots \dots$		13
3.6.3 Методы		13
3.6.3.1 get_status()		13
3.7 Класс server		13
3.7.1 Подробное описание		14
· · · · · · · · · · · · · · · · · · ·		

3.7.2 Конс	груктор(ы)	 										 			14
3.7.	2.1 server()	 										 			14
Предметный указа	тель														15

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

authorization	
base	7
calc	8
connection	9
error_processing	11
invalid_argument	
name_error	11
server	13

перархический список классов	Иерархический	список	классов
------------------------------	---------------	--------	---------

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

authorization	
Класс для аутентификации клиента на сервере	5
base	
Класс для работы с базой данных	7
m calc	
Класс для вычислений по вектору	8
connection	
Класс соединения с клиентом	9
error_processing	
Класс для обработки ошибок и записи в журнал	11
name_error	
Класс для обработки ошибок	11
server	
Класс для получения параметров коммандной строки	13

	Алфавитный	указатель	классов
--	------------	-----------	---------

Глава 3

Классы

3.1 Класс authorization

Класс для аутентификации клиента на сервере

#include <authorization.h>

Открытые члены

- authorization (string Id, string Password)
 - Конструктор для установки идентификатора и пароля клиента
- void generate salt ()

Генерация случайной соли для вычисления хэша

- bool compare hash (string client hash)
 - Сравнение хэша, присылаемого клиентом и хэша, вычисляемого внутри метода
- string get salt ()
- std::string get id ()
- std::string get password ()
- std::string get hash ()

Открытые атрибуты

- $char err [3] = \{'E', 'R', 'R'\}$
 - Сообщение, отсылаемое клиенту при ошибке его обрабоки
- char ok $[2] = \{'O', 'K'\}$

Сообщение, отсылаемое клиенту при успешной авторизации

3.1.1 Подробное описание

Класс для аутентификации клиента на сервере

Функция вычисления хэша MD5

3.1.2 Конструктор(ы)

3.1.2.1 authorization()

```
\label{eq:authorization:authorization} \begin{tabular}{l} authorization:: authorization ( & tring Id, & tring Password ) \end{tabular}
```

Конструктор для установки идентификатора и пароля клиента

Аргументы

in	Id,идентификатор	клиента, string.
in	Password,пароль	клиента, string.

3.1.3 Методы

3.1.3.1 compare_hash()

Сравнение хэша, присылаемого клиентом и хэша, вычисляемого внутри метода

Аргументы

in client_hash,xэш	клиента, tring
--------------------	----------------

3.1.3.2 generate_salt()

```
{\tt void\ authorization::generate\_salt\ (\ )}
```

Генерация случайной соли для вычисления хэша

Аргументы

in	salt,соль,string.	
in	size,размер	соли,
		size_t.

3.2 Класс base 7

Объявления и описания членов классов находятся в файлах:

- \bullet authorization.h
- authorization.cpp

3.2 Класс base

```
Класс для работы с базой данных
```

```
#include <read base.h>
```

Открытые члены

• base (string base name)

Конструктор, считывающий базу данных и сохраняющий её в словарь

• bool check_user (string user)

Проверка наличия идентификатора клиента в базе данных

Открытые атрибуты

• map< string, string > DataBase Словарь с парами идентификатор:пароль

3.2.1 Подробное описание

Класс для работы с базой данных

3.2.2 Конструктор(ы)

```
3.2.2.1 base()
```

```
base::base (
string base name)
```

Конструктор, считывающий базу данных и сохраняющий её в словарь

Аргументы

in	base name,путь	к файлу с базой данных, string.

3.2.3 Методы

```
3.2.3.1 check_user()
bool base::check_user(
string user)
```

Проверка наличия идентификатора клиента в базе данных

Аргументы

in user,идентификатора	клиента, string
------------------------	-----------------

Объявления и описания членов классов находятся в файлах:

- $read_base.h$
- $read_base.cpp$

3.3 Класс calc

Класс для вычислений по вектору

```
#include <calculation.h>
```

Открытые члены

```
• int16_t * calculation (vector< int16_t > vecto) 
Конструктор без параметров
```

3.3.1 Подробное описание

Класс для вычислений по вектору

Операция - произведение

3.3.2 Методы

```
3.3.2.1 calculation()
```

```
int
16_t* calc::calculation ( \label{eq:calculation} vector < int
16_t > vecto \ ) \quad [inline]
```

Конструктор без параметров

Вычисляет произведение вектоов

3.4 Класс connection 9

Аргументы

```
in vector, вектор, vector < int 16_t>
```

Объявления и описания членов класса находятся в файле:

• calculation.h

3.4 Класс connection

```
Класс соединения с клиентом
```

```
#include <connection.h>
```

Открытые члены

```
• connection (int port)
```

Конструктор

• int Bind ()

Привязка сокета к адресу

• void Listen ()

Установка сокета в пассивный режим ожидания

• int Accept ()

Приём соединения

• int Recv (int s, void *buf, int size)

Приём данных от клиента

• void Send (int s, void *buf, int sizeb)

Отправка данных клиенту

• void Close ()

Закрытие сокета

3.4.1 Подробное описание

Класс соединения с клиентом

Для соединения использованы сокеты

3.4.2 Конструктор(ы)

3.4.2.1 connection()

```
connection::connection (
    int port )
```

Конструктор

Устанавливает порт, инициализирует основной сокет и структуру sockaddr_in

Аргументы

in рогt,порт,на котором работает сервер	, int.
---	--------

3.4.3 Методы

3.4.3.1 Recv()

```
int connection::Recv (
    int s,
    void * buf,
    int size )
```

Приём данных от клиента

Аргументы

i	n	sock,coket,int	
i	n	buf,буфер	для данных, void*
i	n	size,размер	буфера, int

3.4.3.2 Send()

```
\label{eq:connection::Send} \begin{tabular}{ll} \mbox{int s,} \\ \mbox{void} * \mbox{buf,} \\ \mbox{int sizeb} \end{tabular}
```

Отправка данных клиенту

Аргументы

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

Объявления и описания членов классов находятся в файлах:

- connection.h
- connection.cpp

3.6 Kaacc name error

3.5 Класс error processing

Класс для обработки ошибок и записи в журнал

```
#include <error processing.h>
```

Открытые члены

```
• string get_log_path (string log_path)
Конструктор без параметров
```

• void write_log (string why, bool exit)

Функция, выдающая путь к файлу с журналом ошибок

3.5.1 Подробное описание

Класс для обработки ошибок и записи в журнал

3.5.2 Методы

```
3.5.2.1 write_log()
```

Функция, выдающая путь к файлу с журналом ошибок

Функция записи ошибки в журнал

Функция записывает время, суть и критичность ошибки

Аргументы

in	why,суть	ошибки, string
in	exit,критичность	ошибки (Критическая - true, Некритическая - false), string

Объявления и описания членов классов находятся в файлах:

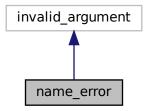
- · error processing.h
- error processing.cpp

3.6 Класс name error

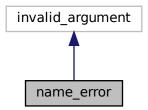
Класс для обработки ошибок

#include <error_processing.h>

Граф наследования:name_error:



Граф связей класса name_error:



Открытые члены

- name_error (const string &whyy, bool exitt=false)
 Конструктор ошибок с строкой в качестве параметра
- name_error (const char *whyy, bool exitt=false) Конструктор ошибок с си-строкой в качестве параметра
- bool get_status () const

3.6.1 Подробное описание

Класс для обработки ошибок

Наследует от класса invalid argument

3.6.2 Конструктор(ы)

3.7 Knacc server 13

```
3.6.2.1 name_error() [1/2]
```

```
\label{eq:const_string} \begin{split} name\_error::name\_error \; ( \\ const \; string \; \& \; whyy, \\ bool \; exitt = false \; ) \quad [inline], \; [explicit] \end{split}
```

Конструктор ошибок с строкой в качестве параметра

Аргументы

in	whyy,тип	ошибки, const string.
in	exitt,критическа	ошибка - true, некритическая ошибка - false, bool

```
3.6.2.2 name error() [2/2]
```

```
\label{eq:const_char} \begin{split} name\_error::name\_error \; ( \\ const \; char * whyy, \\ bool \; exitt = false \; ) \quad [inline], \; [explicit] \end{split}
```

Конструктор ошибок с си-строкой в качестве параметра

Аргументы

in	whyy,тип	ошибки, const char*.
in	exitt,критическа	ошибка - true, штатная - false, bool

3.6.3 Методы

```
3.6.3.1 get status()
```

```
bool name_error::get_status ( ) const [inline]
```

<Возвращает статус критичности ошибки

Объявления и описания членов класса находятся в файле:

• error_processing.h

3.7 Kласс server

Класс для получения параметров коммандной строки

#include <server.h>

Открытые члены

```
server (int argc, char **argv)
Конструктор, внутри которого считываются параметры командной строки
string get_lose ()
string get_log ()
int get_port ()
```

3.7.1 Подробное описание

Класс для получения параметров коммандной строки

3.7.2 Конструктор(ы)

```
3.7.2.1 server()
server::server(
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)-р Порт, на котором работает сервер, необязательный 4)-h вызов подсказки 5)? вызов подсказки При ошибках в параметрах или запуске без них вызывается справка и программа завершает работу

Аргументы

in	int	argc
in	char	**argv

Объявления и описания членов классов находятся в файлах:

- server.h
- server.cpp

Предметный указатель

```
authorization, 5
     authorization, 6
     compare hash, 6
     generate salt, 6
base, 7
     base, 7
    check user, 8
{\rm calc,\,8}
     calculation, 8
calculation
    calc, 8
{\rm check\_user}
     base, 8
compare\_hash
     authorization, 6
connection, 9
     connection, 9
     Recv, 10
     Send, 10
error_processing, 11
     {\rm write\_log,}\ {\color{red}11}
generate\_salt
     authorization, 6
{\tt get\_status}
     name_error, 13
{\rm name\_error,}~11
     get_status, 13
     name\_error,\, 12,\, 13
Recv
    connection, 10
Send
    connection, 10
server, 13
    server, 14
write log
    error_processing, 11
```