

Сетевой сервер

Создано системой Doxygen 1.9.4

| | |
|--------------------------------|----|
| 1 Иерархический список классов | 1 |
| 1.1 Иерархия классов | 1 |
| 2 Алфавитный указатель классов | 3 |
| 2.1 Классы | 3 |
| 3 Список файлов | 5 |
| 3.1 Файлы | 5 |
| 4 Классы | 7 |
| 4.1 Класс base | 7 |
| 4.1.1 Подробное описание | 7 |
| 4.1.2 Методы | 7 |
| 4.1.2.1 connect() | 7 |
| 4.2 Класс calc | 8 |
| 4.2.1 Подробное описание | 8 |
| 4.2.2 Конструктор(ы) | 8 |
| 4.2.2.1 calc() | 8 |
| 4.3 Класс communicator | 9 |
| 4.3.1 Подробное описание | 9 |
| 4.3.2 Методы | 9 |
| 4.3.2.1 connection() | 9 |
| 4.3.2.2 generate_salt() | 10 |
| 4.3.2.3 sha224() | 10 |
| 4.4 Класс crit_err | 11 |
| 4.4.1 Подробное описание | 11 |
| 4.5 Класс interface | 12 |
| 4.5.1 Подробное описание | 12 |
| 4.5.2 Методы | 12 |
| 4.5.2.1 get_port() | 12 |
| 4.5.2.2 parser() | 12 |
| 4.5.2.3 setup_connection() | 13 |
| 4.5.2.4 spravka() | 13 |
| 4.6 Класс logger | 13 |
| 4.6.1 Подробное описание | 14 |
| 4.6.2 Методы | 14 |
| 4.6.2.1 gettime() | 14 |
| 4.6.2.2 set_path() | 14 |
| 4.6.2.3 writelog() | 15 |
| 4.7 Класс no_crit_err | 15 |
| 4.7.1 Подробное описание | 16 |
| 5 Файлы | 17 |
| 5.1 Файл base.h | 17 |
| 5.1.1 Подробное описание | 18 |

| | |
|-------------------------------------|----|
| 5.2 base.h | 18 |
| 5.3 Файл calc.h | 18 |
| 5.3.1 Подробное описание | 19 |
| 5.4 calc.h | 19 |
| 5.5 Файл communicator.h | 19 |
| 5.5.1 Подробное описание | 20 |
| 5.6 communicator.h | 20 |
| 5.7 Файл error.h | 21 |
| 5.7.1 Подробное описание | 22 |
| 5.8 error.h | 22 |
| 5.9 Файл interface.h | 22 |
| 5.9.1 Подробное описание | 23 |
| 5.10 interface.h | 23 |
| 5.11 Файл log.h | 23 |
| 5.11.1 Подробное описание | 24 |
| 5.12 log.h | 24 |
| Предметный указатель | 25 |

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

| | |
|------------------------|----|
| base | 7 |
| calc | 8 |
| communicator | 9 |
| interface | 12 |
| logger | 13 |
| std::runtime_error | |
| crit_err | 11 |
| no_crit_err | 15 |

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

| | | |
|------------------------------|--|----|
| base | Класс для чтения базы данных | 7 |
| calc | Класс для операции среднего арифметического элементов вектора | 8 |
| communicator | Класс коммуникатора | 9 |
| crit_err | Класс для возбуждения критических ошибок Возбуждает критические ошибки . | 11 |
| interface | Класс интерфейса | 12 |
| logger | Класс для журнала лога | 13 |
| no_crit_err | Класс для возбуждения некритических ошибок Возбуждает некритические ошибки | 15 |

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

| | | |
|--------------------------------|--|----|
| base.h | Заголовочный файл для модуля базы данных | 17 |
| calc.h | Заголовочный файл для модуля вычислений | 18 |
| communicator.h | Заголовочный файл для коммуникатора сервера | 19 |
| error.h | Заголовочный файл модуля возбуждения ошибок | 21 |
| interface.h | Заголовочный файл для интерфейса | 22 |
| log.h | Заголовочный файл для установки журнала лога | 23 |

Глава 4

Классы

4.1 Класс base

Класс для чтения базы данных

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

Открытые члены

- void `connect` (std::string f)
Установка соединения с базой данных
- std::map< std::string, std::string > `get_data` ()
Получить базу данных

Закрытые данные

- std::map< std::string, std::string > `data_base`
Контейнер "логин+пароль".

4.1.1 Подробное описание

Класс для чтения базы данных

Контейнер `data_base` хранит в себе логин и пароль пользователя Для получения базы используется метод `get_data()`

4.1.2 Методы

4.1.2.1 `connect()`

```
void base::connect (  
    std::string f )
```

Установка соединения с базой данных

Читает из файла строку базы данных

Аргументы

| | | |
|----|------|---------------------|
| in | Путь | к файлу базы данных |
|----|------|---------------------|

Исключения

| | |
|-----------------------|---|
| <code>crit_err</code> | Если файл не открывается, либо несоответствие формату строки "логин:пароль" |
|-----------------------|---|

Объявления и описания членов классов находятся в файлах:

- `base.h`
- `base.cpp`

4.2 Класс calc

Класс для операции среднего арифметического элементов вектора

```
#include <calc.h>
```

Открытые члены

- `calc` (`std::vector< float >chisla`)
Вычисление
- `float send_res ()`
Метод для отправки результата

Открытые атрибуты

- `float res`
Переменная, в которую будет записан результат

4.2.1 Подробное описание

Класс для операции среднего арифметического элементов вектора

Вектор указывается в параметрах конструктора. Для получения результата вычислений используется метод `send_res`.

4.2.2 Конструктор(ы)

4.2.2.1 calc()

```
calc::calc (
    std::vector< float > chisla )
```

Вычисление

Аргументы

| | | |
|----|--------|--|
| in | chisla | Вектор данных. Не должен быть пустой. Тип данных float |
|----|--------|--|

Исключения

| | |
|------------------|--------------|
| no_crit_err,если | вектор пуст. |
|------------------|--------------|

Объявления и описания членов классов находятся в файлах:

- [calc.h](#)
- [calc.cpp](#)

4.3 Класс communicator

Класс коммуникатора

```
#include <communicator.h>
```

Открытые члены

- int [connection](#) (int port, std::map< std::string, std::string > [base](#), [logger](#) *l)
Соединение с сервером
- std::string [sha224](#) (std::string input_str)
Формирование хэша методом SHA224.
- std::string [generate_salt](#) ()
Формирование соли

4.3.1 Подробное описание

Класс коммуникатора

Устанавливает соединение с сервером, производит авторизацию клиента В качестве метода хеширования выбран sha224.

4.3.2 Методы

4.3.2.1 connection()

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

Соединение с сервером

Производит соединение с сервером, авторизует пользователя Передает вектор с данными для вычисления в класс calc.

Аргументы

| | | |
|----|-------|--|
| in | Номер | порта, контейнер с базой данных, переменная типа logger для записи всех событий в журнал событий |
|----|-------|--|

Исключения

| | |
|------------------------------|---|
| <code>crit_err</code> , если | произошел сбой на этапе соединения с сервером, на этапе авторизации или отправки данных |
|------------------------------|---|

4.3.2.2 generate_salt()

```
std::string communicator::generate_salt ( )
```

Формирование соли

Производит формирование соли ПОсновано на библиотеке boost

Аргументы

| | | |
|----|-----------|--|
| in | input_str | |
|----|-----------|--|

Возвращает

сформированная соль

4.3.2.3 sha224()

```
std::string communicator::sha224 (
    std::string input_str )
```

Формирование хэша методом SHA224.

Производит формирование хэша методом sha224 библиотеки cryptopp Формирует хэш соли и пароля

Аргументы

| | | |
|----|-----------|--|
| in | input_str | |
|----|-----------|--|

Возвращает

результат хэширования

Объявления и описания членов классов находятся в файлах:

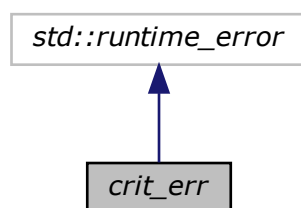
- [communicator.h](#)
- `communicator.cpp`

4.4 Класс `crit_err`

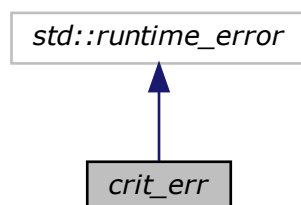
Класс для возбуждения критических ошибок Возбуждает критические ошибки

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

Граф наследования:`crit_err`:



Граф связей класса `crit_err`:



Открытые члены

- `crit_err (const std::string &s)`

4.4.1 Подробное описание

Класс для возбуждения критических ошибок Возбуждает критические ошибки

Объявления и описания членов класса находятся в файле:

- [error.h](#)

4.5 Класс interface

Класс интерфейса

```
#include <interface.h>
```

Открытые члены

- `interface ()`
конструктор по умолчанию
- `bool parser (int argc, const char **argv)`
Парсер
- `void setup_connection (const std::string &basefile, const std::string &logfile)`
Установка соединения с базой данных и журналом лога
- `void spravka (const boost::program_options::options_description &opts)`
Справка
- `int get_port () const`
Получение порта
- `std::string get_base ()`
- `std::string get_log ()`

Закрытые данные

- `int port`
Переменная с номером порта
- `string basefile`
путь к файлу базы данных
- `string logfile`
путь к файлу журнала

4.5.1 Подробное описание

Класс интерфейса

Получает порт, по умолчанию 33333 Парсер выполняет чтение операндов ком. строки Устанавливается соединение с базой данных и журналом лога Выполняется вызов справки

4.5.2 Методы

4.5.2.1 get_port()

```
int interface::get_port ( ) const [inline]
```

Получение порта

Этот метод передает значение порта в коммуникатор

4.5.2.2 parser()

```
bool interface::parser (
    int argc,
    const char ** argv )
```

Парсер

Читает операнды ком.строки В случае передачи операнда -h производится вызов справки

Аргументы

| | | |
|----|--------|-------------------------------|
| in | Кол-во | операндов, значение операндов |
|----|--------|-------------------------------|

Исключения

| | |
|--------------------------|--|
| crit_err | в случае передачи некорректного значения порта |
|--------------------------|--|

Возвращает

true или false в случае корректной или некорретной передачи аргументов

4.5.2.3 setup_connection()

```
void interface::setup_connection (
    const std::string & basefile,
    const std::string & logfile )
```

Установка соединения с базой данных и журналом лога

Устанавливает соединение с базой банных и журналом лога

Аргументы

| | | |
|----|------|------------------------------------|
| in | путь | к файлу базы данных и журналу лога |
|----|------|------------------------------------|

4.5.2.4 spravka()

```
void interface::spravka (
    const boost::program_options::options_description & opts )
```

Справка

Вызов справки

Объявления и описания членов классов находятся в файлах:

- [interface.h](#)
- [interface.cpp](#)

4.6 Класс logger

Класс для журнала лога

```
#include <log.h>
```

Открытые члены

- `logger ()`
конструктор по умолчанию
- `logger (const std::string &path)`
конструктор с параметром
- `int set_path (const std::string &path_file)`
Установка пути к файлу лога
- `int writelog (const std::string &message)`
Запись события в журнал
- `std::string get_path () const`
Получение пути к файлу лога (для модульного тестирования)

Закрытые члены

- `std::string gettime ()`
Получение текущего времени

Закрытые данные

- `std::string path_to_logfile`
Путь к файлу лога

4.6.1 Подробное описание

Класс для журнала лога

4.6.2 Методы

4.6.2.1 `gettime()`

```
std::string logger::gettime ( ) [private]
```

Получение текущего времени

Позволяет получить время вместе с датой

4.6.2.2 `set_path()`

```
int logger::set_path (
    const std::string & path_file )
```

Установка пути к файлу лога

Устанавливает путь к файлу лога

Аргументы

| | | |
|----|------|--------------|
| in | Путь | к файлу лога |
|----|------|--------------|

Исключения

| | |
|--------------------------|--------------------------|
| crit_err | Если файл не открывается |
|--------------------------|--------------------------|

4.6.2.3 writelog()

```
int logger::writelog (
    const std::string & message )
```

Запись события в журнал

Записывает событие в лог

Аргументы

| | | |
|----|-----------|------------|
| in | Сообщение | для записи |
|----|-----------|------------|

Исключения

| | |
|--------------------------|------------------------------------|
| crit_err | Если файл не открывается на запись |
|--------------------------|------------------------------------|

Объявления и описания членов классов находятся в файлах:

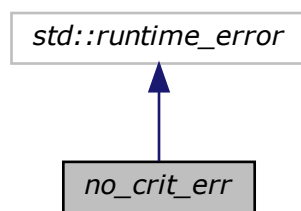
- [log.h](#)
- log.cpp

4.7 Класс no_crit_err

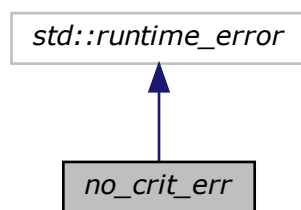
Класс для возбуждения некритических ошибок Возбуждает некритические ошибки

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

Граф наследования: `no_crit_err`:



Граф связей класса `no_crit_err`:



Открытые члены

- `no_crit_err` (`const std::string s`)

4.7.1 Подробное описание

Класс для возбуждения некритических ошибок Возбуждает некритические ошибки

Объявления и описания членов класса находятся в файле:

- [error.h](#)

Глава 5

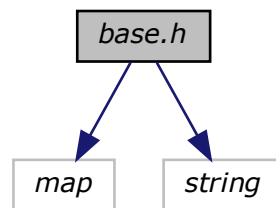
Файлы

5.1 Файл base.h

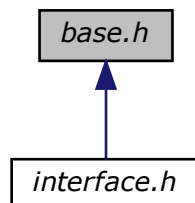
Заголовочный файл для модуля базы данных

```
#include <map>
#include <string>
```

Граф включаемых заголовочных файлов для base.h:



Граф файлов, в которые включается этот файл:



Классы

- class [base](#)

Класс для чтения базы данных

5.1.1 Подробное описание

Заголовочный файл для модуля базы данных

Автор

Маштаков Д.С.

Версия

1.0

5.2 base.h

[См. документацию.](#)

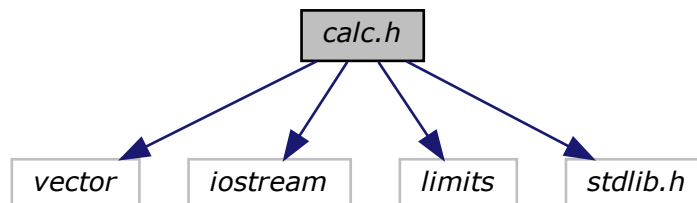
```
1 #pragma once
2 #include <map>
3 #include <string>
13 class base
14 {
15 private:
16     std::map<std::string, std::string> data_base;
17 public:
26     void connect(std::string f);
30     std::map<std::string, std::string> get_data()
31     {
32         return data_base;
33     }
34 };
```

5.3 Файл calc.h

Заголовочный файл для модуля вычислений

```
#include <vector>
#include <iostream>
#include <limits>
#include <stdlib.h>
```

Граф включаемых заголовочных файлов для calc.h:



Классы

- class `calc`

Класс для операции среднего арифметического элементов вектора

5.3.1 Подробное описание

Заголовочный файл для модуля вычислений

Автор

Маштаков Д.С.

Версия

1.0

5.4 calc.h

[См. документацию.](#)

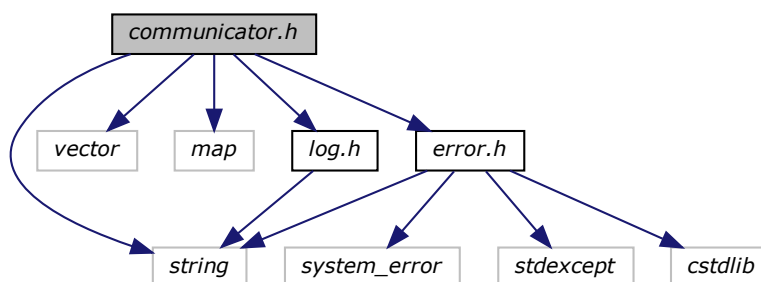
```
1 #pragma once
2 #include <vector>
3 #include <iostream>
4 #include <vector>
5 #include <limits>
6 #include <stdlib.h>
16 class calc
17 {
18     public:
19     float res;
26     calc(std::vector<float>chisla);
27     float send_res();
28 };
```

5.5 Файл communicator.h

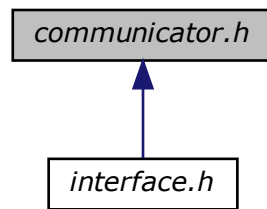
Заголовочный файл для коммуникатора сервера

```
#include <string>
#include <vector>
#include <map>
#include "log.h"
#include "error.h"
```

Граф включаемых заголовочных файлов для communicator.h:



Граф файлов, в которые включается этот файл:



Классы

- class `communicator`
Класс коммуникатора

5.5.1 Подробное описание

Заголовочный файл для коммуникатора сервера

Автор

Маштаков Д.С.

Версия

1.0

5.6 `communicator.h`

[См. документацию.](#)

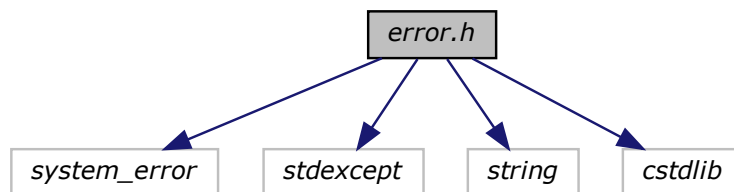
```
1 #pragma once
2 #include <string>
3 #include <vector>
4 #include <map>
5 #include "log.h"
6 #include "error.h"
7
8 using namespace std;
18 class communicator
19 {
20     public:
29     int connection(int port, std::map<std::string, std::string> base, logger* l);
38     std::string sha224(std::string input_str);
47     std::string generate_salt();
48 };
```


5.7 Файл error.h

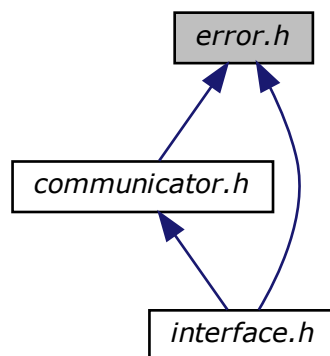
Заголовочный файл модуля возбуждения ошибок

```
#include <system_error>
#include <stdexcept>
#include <string>
#include <cstdlib>
```

Граф включаемых заголовочных файлов для error.h:



Граф файлов, в которые включается этот файл:



Классы

- class `crit_err`
Класс для возбуждения критических ошибок Возбуждает критические ошибки
- class `no_crit_err`
Класс для возбуждения некритических ошибок Возбуждает некритические ошибки

5.7.1 Подробное описание

Заголовочный файл модуля возбуждения ошибок

Автор

Маштаков Д.С.

Версия

1.0

5.8 error.h

[См. документацию.](#)

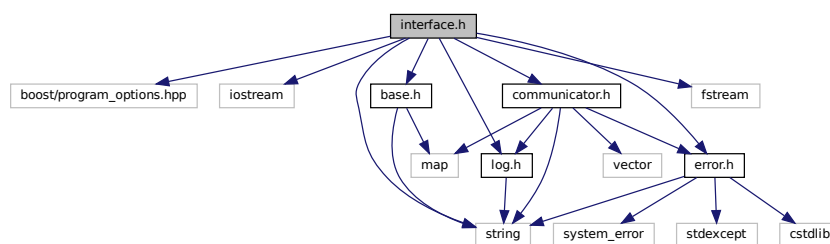
```
1 #pragma once
2 #include <system_error>
3 #include <stdexcept>
4 #include <string>
5 #include <cstdlib>
14 class crit_err:public std::runtime_error
15 {
16     public:
17     crit_err(const std::string& s):std::runtime_error(s){}
18 };
19 };
23 class no_crit_err:public std::runtime_error
24 {
25     public:
26     no_crit_err(const std::string s): std::runtime_error(s){}
27 };
```

5.9 Файл interface.h

Заголовочный файл для интерфейса

```
#include <boost/program_options.hpp>
#include <iostream>
#include <string>
#include <fstream>
#include "log.h"
#include "base.h"
#include "communicator.h"
#include "error.h"
```

Граф включаемых заголовочных файлов для interface.h:



Классы

- class `interface`
Класс интерфейса

5.9.1 Подробное описание

Заголовочный файл для интерфейса

Автор

Маштаков Д.С.

Версия

1.0

5.10 interface.h

[См. документацию.](#)

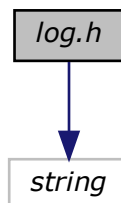
```
1 #pragma once
2 #include <boost/program_options.hpp>
3 #include <iostream>
4 #include <string>
5 #include <fstream>
6 #include "log.h"
7 #include "base.h"
8 #include "communicator.h"
9 #include "error.h"
10
11 class interface {
12     int port;
13     string basefile;
14     string logfile;
15
16 public:
17     interface() : port(33333) {}
18     bool parser(int argc, const char** argv);
19     void setup_connection(const std::string& basefile, const std::string& logfile);
20     void spravka(const boost::program_options::options_description& opts);
21     int get_port() const { return port; }
22     std::string get_base() { return basefile; }
23     std::string get_log() { return logfile; }
24 };
```

5.11 Файл log.h

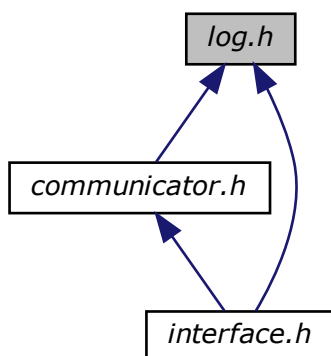
Заголовочный файл для установки журнала лога

```
#include <string>
```

Граф включаемых заголовочных файлов для log.h:



Граф файлов, в которые включается этот файл:



Классы

- class `logger`

Класс для журнала лога

5.11.1 Подробное описание

Заголовочный файл для установки журнала лога

Автор

Маштаков Д.С.

Версия

1.0

5.12 log.h

[См. документацию.](#)

```
1 #pragma once
2 #include <string>
10 class logger {
11 private:
18     std::string gettime();
19     std::string path_to_logfile;
20
21 public:
22     logger();
23
24     logger(const std::string& path);
25
34     int set_path(const std::string& path_file);
35
44     int writelog(const std::string& message);
45
50     std::string get_path() const;
51 };
```

Предметный указатель

- base, [7](#)
 - connect, [7](#)
- base.h, [17](#)
- calc, [8](#)
 - calc, [8](#)
- calc.h, [18](#)
- communicator, [9](#)
 - connection, [9](#)
 - generate_salt, [10](#)
 - sha224, [10](#)
- communicator.h, [19](#)
- connect
 - base, [7](#)
- connection
 - communicator, [9](#)
- crit_err, [11](#)
- error.h, [21](#)
- generate_salt
 - communicator, [10](#)
- get_port
 - interface, [12](#)
- gettime
 - logger, [14](#)
- interface, [12](#)
 - get_port, [12](#)
 - parser, [12](#)
 - setup_connection, [13](#)
 - spravka, [13](#)
- interface.h, [22](#)
- log.h, [23](#)
- logger, [13](#)
 - gettime, [14](#)
 - set_path, [14](#)
 - writelog, [15](#)
- no_crit_err, [15](#)
- parser
 - interface, [12](#)
- set_path
 - logger, [14](#)
- setup_connection
 - interface, [13](#)
- sha224
 - communicator, [10](#)
- spravka
 - interface, [13](#)
- writelog
 - logger, [15](#)