Сервер для курсовой работы

Создано системой Doxygen 1.9.4

Глава 1

Алфавитный указатель пространств имен

1.1 Пространства имен

Полный список документированных пространств имен.

Глава 2

Иерархический список классов

2.1 Иерархия классов

Иерархия классов.

Глава 3

Алфавитный указатель классов

3.1 Классы

Классы с их кратким описанием.

Глава 4

Список файлов

4.1 Файлы

Полный список документированных файлов.

Глава 5

Пространства имен

5.1 Пространство имен CPP

Пространство имён библиотеки Crypto++.

5.1.1 Подробное описание

Пространство имён библиотеки Crypto++.

5.2 Пространство имен fs

Пространство имен библиотеки boost::filesystem. Используется для проверки существования файлов

5.2.1 Подробное описание

Пространство имен библиотеки boost::filesystem. Используется для проверки существования файлов

5.3 Пространство имен po

Пространство имен библиотеки boost::program_options. Используется для парсинга командной строки

5.3.1 Подробное описание

Пространство имен библиотеки boost::program_options. Используется для парсинга командной строки

Глава 6

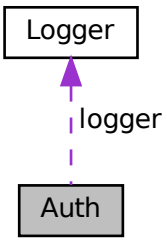
Классы

6.1 Класс Auth

Класс аутентификации

#include <auth.h>

Граф связей класса Auth:



Открытые члены

- Auth (int socket, std::string file)

  Конструктор класса Auth.
- ~Auth ()

  Деструктор класса Auth освобождает динамическую память, выделенную после создания объекта класса SHA1.
- void authentication (int socket, std::string database)

  Функция аутентификации

Закрытые члены

- std::string read_buffer ()

  Функция чтения данных из буфера методом удвоения в цикле
- void sendMessage (int socket, const std::string &message)

  Функция отправки данных клиенту
- std::string calculateHash (const std::string &pass, const std::string &salt)

  Функция вычисления хэша
- void processVectors (int client_socket)

  Функция обработки векторов
- std::string getPass (const std::string &filename, const std::string &username)

  Функция поиска пароля по логину

Закрытые данные

- int clientSocket

  Клиентский сокет
- std::string database

  Строка, содержащая название файла с базой клиентов
- std::string logfile

  Строка, содержащая название файла журнала
- Logger logger

  Объект класса Logger.
- CryptoPP::HashTransformation * hash

6.1.1 Подробное описание

Класс аутентификации

В конструкторе устанавливается клиентский сокет и название файла журнала для логирования. Класс содержит главную функцию аутентификации, метод чтений данных из буфера, метод поиска пароля из базы данных, метод отправки сообщения клиенту, метод вычисления хэша и метод приема и обработки векторов.

6.1.2 Конструктор(ы)

6.1.2.1 Auth()

```
Auth::Auth (
            int socket,
            std::string file )
```

Конструктор класса Auth.

Констурктор устанавливает клиентский сокет, название файла журнала, вызывает конструктор класса Logger и создает новый объект класса SHA1 для метода вычисления хэша

Аргументы

in	socket	клиентский сокет, file название файла журнала

6.1.3 Методы

6.1.3.1 authentication()

```
void Auth::authentication (
                int socket,
                std::string database )
```

Функция аутентификации

Принимает сообщение от клиента и извлекает из него логин, соль и хэш. Затем по извлеченному логину в базе данных ищется пароль вычисляется хэш и сравнивается с хэшем, полученным от клиента. Если хэши совпадают, то клиенту отправляется сообщение "OK", в противном случае клиенту отправляется сообщение "ERR" и возбуждается исключение auth_error.

Аргументы

in	socket	клиентский сокет
in	database	название файла с базой данных

Исключения

FileError,если	файл не открывается или его не существует
auth_error,если	из сообщения не удалось извлечь логин или если пароли не совпадают
std::system_error,при	ошибках, связанных с приемом или отправкой данных клиенту

6.1.3.2 calculateHash()

```
std::string Auth::calculateHash (
                const std::string & pass,
                const std::string & salt )   [private]
```

Функция вычисления хэша

Аргументы

in	pass	пароль
in	salt	соль

Возвращает

hash_16 строка, содержащая хэш

Исключения

auth_error,если	вычисленный хэш и хэш, полученный от клиента не совпадают

6.1.3.3 getPass()

```
std::string Auth::getPass (
            const std::string & filename,
            const std::string & username )   [private]
```

Функция поиска пароля по логину

Аргументы

in	filename	файл с базой клиетов
in	username	логин

Возвращает

pass строка, содержащая пароль, соответствующий логину

Исключения

FileError,если	невозможно открыть файл или его не существует
auth_error,если	логин не найден или строки базы данных не соответствуют формату

6.1.3.4 processVectors()

```
void Auth::processVectors (
            int client_socket )   [private]
```

Функция обработки векторов

Функция принимает количество векторов, длину вектора и его значения и вычисляет сумму квадратов для каждого вектора и отправляет результат клиенту. В случае переполнения клиенту отправляется максимальное значения для типа uint32_t.

Аргументы

in	client_socket	клиентский сокет

Исключения

std::system_error,при	ошибках, связанных с приемом или отправкой данных клиенту
vector_error,если	длина вектора больше максимального значения для uint32_t или длина вектора отрицательная

6.1.3.5 read_buffer()

std::string Auth::read_buffer ( ) [private]

Функция чтения данных из буфера методом удвоения в цикле

Возвращает

res Данные из буфера

Исключения

std::system_error,если	буфер пустой или при ошибках приема данных от клиента

6.1.3.6 sendMessage()

void Auth::sendMessage (
int socket,
const std::string & message) [private]

Функция отправки данных клиенту

Аргументы

in	socket	клиентский сокет
in	message	сообщение, отправляемое клиенту

Исключения

std::system_error,при	ошибках оправки данных клиенту

Объявления и описания членов классов находятся в файлах:

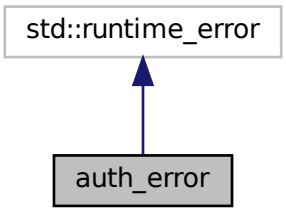
- auth.h
- auth.cpp

6.2 Класс auth_error

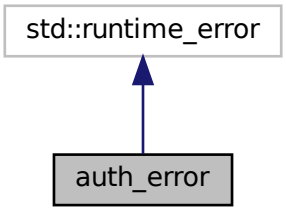
Класс отлова исключений, связанных с аутентификацией

#include <serv_error.h>

Граф наследования:auth_error:



Граф связей класса auth_error:



Открытые члены

- auth_error (const std::string &s)
- auth_error (const char *s)

6.2.1 Подробное описание

Класс отлова исключений, связанных с аутентификацией

Отлавливает исключения, которые возбуждаются в результате ошибок функции аутентификации

Объявления и описания членов класса находятся в файле:

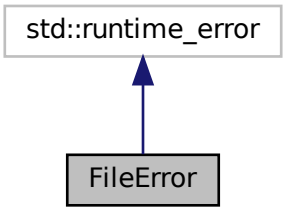
- serv_error.h

6.3 Класс FileError

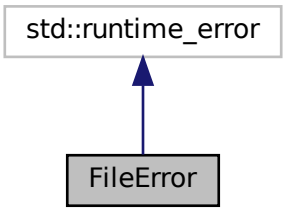
Класс отлова исключений, связанных с работой с файлами

#include <serv_error.h>

Граф наследования:FileError:



Граф связей класса FileError:



Открытые члены

- FileError (const std::string &s)

6.3.1 Подробное описание

Класс отлова исключений, связанных с работой с файлами

Отлавливает исключения, которые возбуждаются в результате ошибок работы с файлами

Объявления и описания членов класса находятся в файле:

- serv_error.h

6.4 Класс Logger

Класс логирования

#include <server.h>

Открытые члены

- Logger (const std::string &filename)
  Конструктор класса Logger.
- void logError (const std::string &message, bool critical)
  Метод записи информации об ошибках в файл журнала

Закрытые члены

- std::string getCurrentDateTime ()
  Метод получения текущей даты и времени

Закрытые данные

- std::ofstream logFile
  Поток для записи информации об ошибках в файл журнала
- std::string logFilename
  Название файла журнала

6.4.1 Подробное описание

Класс логирования

В конструкторе устанавливается название файла журнала для логирования. Класс содержит метод записи информации об ошибках в файл журнала и метод получения времени возбуждения исключения.

6.4.2 Конструктор(ы)

6.4.2.1 Logger()

```
Logger::Logger (
            const std::string & filename )
```

Конструктор класса Logger.

Конструктор открывает файл журнала

Аргументы

in	filename	назвние файла журнала

Исключения

FileError,если	файл не открывается или его не существует

6.4.3 Методы

6.4.3.1 getCurrentDateTime()

std::string Logger::getCurrentDateTime ( ) [private]

Метод получения текущей даты и времени

Возвращает

ss.str() строка из потока с текущей датой и временем

6.4.3.2 logError()

void Logger::logError (
const std::string & message,
bool critical)

Метод записи информации об ошибках в файл журнала

Записывается дата и время возникновения ошибки, критичность ошибки и информация об ошибке.

Аргументы

in	message	сообщение об ошибке, critical критичность ошибки

Объявления и описания членов классов находятся в файлах:

- server.h
- server.cpp

6.5 Структура Params

Структура для хранения значений, полученных из командной строки

#include <UserInterface.h>

Открытые атрибуты

- std::string logfile

  Строка, содержащая название файла журнала
- std::string database

  Строка, содержащая название файла с базой клиентов
- unsigned short int port

  Значение сетевого порта, на котором работает сервер

6.5.1 Подробное описание

Структура для хранения значений, полученных из командной строки

Объявления и описания членов структуры находятся в файле:

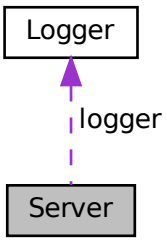
- UserInterface.h

6.6 Класс Server

Класс сетевого взаимодействия

#include <server.h>

Граф связей класса Server:



Открытые члены

- Server (const std::string &database, unsigned short int port, const std::string &logfile)

  Конструктор класса Server.
- ~Server ()

  Деструктор класса Server. Закрывает серверный сокет
- void run ()

  Метод запуска сервера

Закрытые данные

- std::string database

  Название файла с базой клиентов
- unsigned short int port

  Значение сетевого порта, на котором работает сервер
- std::string logfile

  Название файла журнала
- Logger logger

  Объект класса Logger.
- int serverSocket

  Идентификатор серверного сокета
- std::unique_ptr< sockaddr_in > self_addr

  Умный указатель серверного адреса
- std::unique_ptr< sockaddr_in > foreign_addr

  Умный указатель адреса клиента

6.6.1 Подробное описание

Класс сетевого взаимодействия

В конструкторе устанавливается название файла журнала для логирования название файла с базой клиентов и сетевой порт, создается серверный сокет, подготавливаются сетевые адреса и привязывается адрес к серверному сокету Класс содержит метод run для запуска сервера.

6.6.2 Конструктор(ы)

6.6.2.1 Server()

```
Server::Server (
            const std::string & database,
            unsigned short int port,
            const std::string & logfile )
```

Конструктор класса Server.

Конструктор устанавливает название файла журнала, название файла с базой клиентов и порт, вызывает конструктор класса Logger. Создает серверный сокет, настравиваются сетевые адреса и привязывается адрес к серверному порту.

Аргументы

in	database	файл с базой клиентов
in	port	сетевой порт, на котором будет работать сервер
in	logfile	название файла журнала

Исключения

server_error,при	возникновения ошибок создания сокета или привязки адреса к серверному сокету

6.6.3 Методы

6.6.3.1 run()

void Server::run ( )

Метод запуска сервера

Переводит сервер в режим прослушивания, устанавливает соединения с клиентом и вызывает функцию аутентификации, если соединение установлено.

Исключения

server_error,при	возникновения ошибок прослушивания или установки соединения с клиентом

Объявления и описания членов классов находятся в файлах:

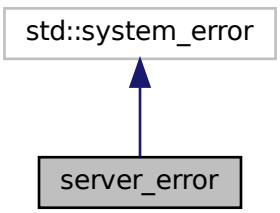
- server.h
- server.cpp

6.7 Класс server_error

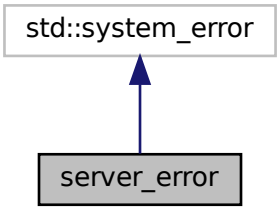
Класс отлова исключений, связанных с сетевым взаимодействием

#include <serv_error.h>

Граф наследования:server_error:



Граф связей класса server_error:



Открытые члены

- server_error (int code, const std::string &s)

6.7.1 Подробное описание

Класс отлова исключений, связанных с сетевым взаимодействием

Отлавливает исключения, которые возбуждаются в результате ошибок в сетевом взаимодействии

Объявления и описания членов класса находятся в файле:

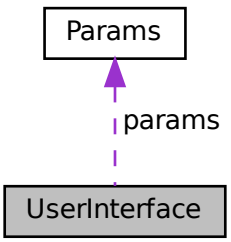
- serv_error.h

6.8 Класс UserInterface

Класс UserInterface.

#include <UserInterface.h>

Граф связей класса UserInterface:



Открытые члены

- UserInterface ()
  Конструктор класса UserInterface.
- bool Parser (int argc, const char **argv)
  Метод разбора значений командной строки
- bool fileExists (const std::string &filename)
  Метод проверки существования файлов
- std::string getDescription ()
  Метод получения справки
- Params getParams ()
  Метод получения параметров из структуры Params.

Закрытые данные

- po::options_description desc
  Объект, содержащий справку к пользовательскому интерфейсу
- po::variables_map vm
  Ассоциативный массив, который используется для хранений знчений командной строки
- Params params
  Объект структуры Params.

6.8.1 Подробное описание

Класс UserInterface.

6.8.2 Конструктор(ы)

6.8.2.1 UserInterface()

```
UserInterface::UserInterface ( )
```

Конструктор класса UserInterface.

Конструктор формирует справку к пользовательскому интерфейсу

6.8.3 Методы

6.8.3.1 fileExists()

```
bool UserInterface::fileExists (
            const std::string & filename )
```

Метод проверки существования файлов

Аргументы

in	filename	название файла

Возвращает

false, если файл не существует

true

6.8.3.2 getDescription()

```
std::string UserInterface::getDescription ( )
```

Метод получения справки

Возвращает

ss.str(), строка из потока со справкой

6.8.3.3 getParams()

```
Params UserInterface::getParams ( )  [inline]
```

Метод получения параметров из структуры Params.

Возвращает

params, объект структуры Params

6.8.3.4 Parser()

```
bool UserInterface::Parser (
            int argc,
            const char ** argv )
```

Метод разбора значений командной строки

Аргументы

in	argc	количество параметров
in	argv	значения параметров

Возвращает

false, при возникновении ошибок

true

Исключения

po::required_option,если	отсутствуют обязательные параметры
std::invalid_argument,при	передачи недопустимых значений
FileError,если	файл(ы) не существуют

Объявления и описания членов классов находятся в файлах:

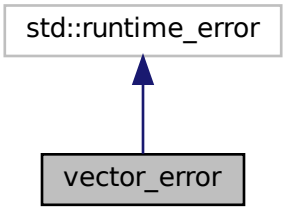
- UserInterface.h
- UserInterface.cpp

6.9 Класс vector_error

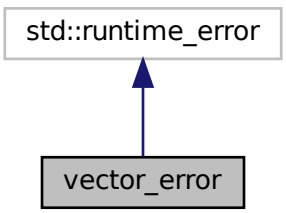
Класс отлова исключений, связанных с векторами

#include <serv_error.h>

Граф наследования:vector_error:



Граф связей класса vector_error:



Открытые члены

- vector_error (const std::string &s)
- vector_error (const char *s)

6.9.1 Подробное описание

Класс отлова исключений, связанных с векторами

Отлавливает исключения, которые возбуждаются в результате ошибок, связанных с std::vector

Объявления и описания членов класса находятся в файле:

- serv_error.h

Глава 7

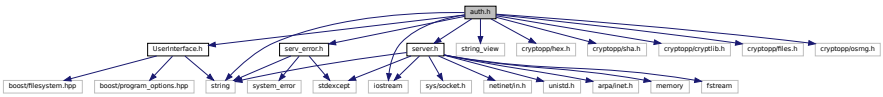
Файлы

7.1 Файл auth.h

Заголовочный файл для модуля аутентификации auth.h.

```
#include <iostream>
#include <string>
#include <string_view>
#include <cryptopp/hex.h>
#include <cryptopp/sha.h>
#include <cryptopp/cryptlib.h>
#include <cryptopp/files.h>
#include <cryptopp/osrng.h>
#include "UserInterface.h"
#include "serv_error.h"
#include "server.h"
```

Граф включаемых заголовочных файлов для auth.h:



Классы

- class Auth

  Класс аутентификации

Пространства имен

- namespace CPP

  Пространство имён библиотеки Crypto++.

Макросы

- #define BUFLEN 5024

  Макрос, который определяет максимальную длину буфера

7.1.1 Подробное описание

Заголовочный файл для модуля аутентификации auth.h.

Автор

Храбров А.А.

Версия

1.0

Дата

18.12.2024

Авторство

ИБСТ ПГУ

Предупреждения

Тестовый

7.2 auth.h

См. документацию.

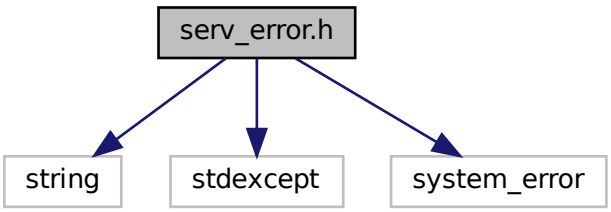
```
1
10 #pragma once
11 // #define private   public // Для модульного тестирования приватных методов класса
12 #include <iostream>
13 #include <string>
14 #include <string_view>
15 #include <cryptopp/hex.h>
16 #include <cryptopp/sha.h>
17 #include <cryptopp/cryptlib.h>
18 #include <cryptopp/files.h>
19 #include <cryptopp/osrng.h>
20
21 /*
22 * @file UserInterface.h
23 * @brief Этот файл содержит объявления для функций, связанных с пользовательским интерфейсом
24 */
25
26 #include "UserInterface.h"
27
28 /*
29 * @file serv_error.h
30 * @brief Этот файл содержит классы возбуждаемых исключений
31 */
32
33 #include "serv_error.h"
34
35 /*
36 * @file server.h
37 * @brief Этот файл содержит объявления для функций, связанных с сетевым взаимодействием
38 */
39
40 #include "server.h"
41
42 #define BUFLEN 5024
43
47 namespace CPP = CryptoPP;
48
```

```
55 class Auth {
56 public:
63     Auth(int socket, std::string file);
64
65     ~Auth();
66
79     void authentication(int socket, std::string database);
80
81 private:
82     int clientSocket;
83     std::string database;
84     std::string logfile;
85     Logger logger;
86     CryptoPP::HashTransformation*  hash;
87
93     std::string read_buffer();
94
101     void sendMessage(int socket, const std::string& message);
102
110     std::string calculateHash(const std::string& pass, const std::string& salt);
111
122     void processVectors(int client_socket);
123
132     std::string getPass(const std::string& filename, const std::string& username);
133 };
134
135 // #undef private // Для модульного тестирования приватных методов класса
```

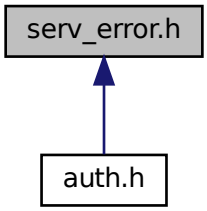
7.3 Файл serv_error.h

Заголовочный файл для модуля исключений serv_error.h.

#include <string>
#include <stdexcept>
#include <system_error>
Граф включаемых заголовочных файлов для serv_error.h:



Граф файлов, в которые включается этот файл:



Классы

- class server_error

  Класс отлова исключений, связанных с сетевым взаимодействием
- class auth_error

  Класс отлова исключений, связанных с аутентификацией
- class vector_error

  Класс отлова исключений, связанных с векторами
- class FileError

  Класс отлова исключений, связанных с работой с файлами

7.3.1 Подробное описание

Заголовочный файл для модуля исключений serv_error.h.

Автор

Храбров А.А.

Версия

1.0

Дата

18.12.2024

Авторство

ИБСТ ПГУ

Предупреждения

Тестовый

7.4 serv_error.h

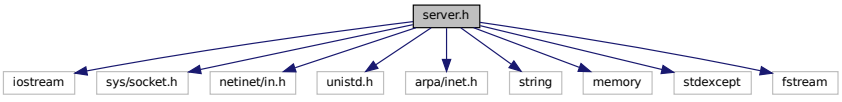
См. документацию.

```
1
10 #pragma once
11 #include <string>
12 #include <stdexcept>
13 #include <system_error>
14
19 class server_error : public std::system_error {
20 public:
21     server_error(int code, const std::string& s) : std::system_error(code, std::generic_category(), s) {}
22 };
23
28 class auth_error: public std::runtime_error {
29 public:
30     auth_error(const std::string& s) : std::runtime_error(s) {}
31     auth_error(const char * s) : std::runtime_error(s) {}
32 };
33
38 class vector_error: public std::runtime_error {
39 public:
40     vector_error(const std::string& s) : std::runtime_error(s) {}
41     vector_error(const char * s) : std::runtime_error(s) {}
42 };
43
48 class FileError : public std::runtime_error {
49 public:
50     FileError(const std::string& s) : std::runtime_error(s) {}
51 };
```

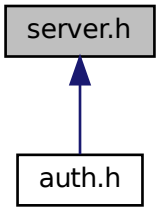
7.5 Файл server.h

Заголовочный файл для модуля сетевого взаимодействия server.h.

#include <iostream>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <string>
#include <memory>
#include <stdexcept>
#include <fstream>
Граф включаемых заголовочных файлов для server.h:



Граф файлов, в которые включается этот файл:



Классы

- class Logger

  Класс логирования
- class Server

  Класс сетевого взаимодействия

7.5.1 Подробное описание

Заголовочный файл для модуля сетевого взаимодействия server.h.

Автор

Храбров А.А.

Версия

1.0

Дата

18.12.2024

Авторство

ИБСТ ПГУ

Предупреждения

Тестовый

7.6 server.h

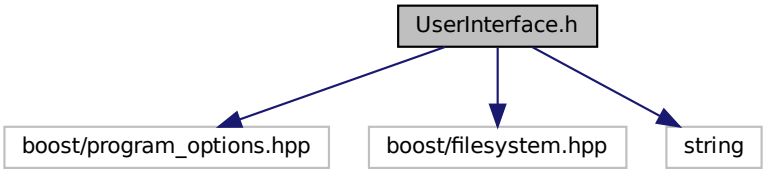
См. документацию.

```
1
10 #pragma once
11 #include <iostream>
12 #include <sys/socket.h>
13 #include <netinet/in.h>
14 #include <unistd.h>
15 #include <arpa/inet.h>
16 #include <string>
17 #include <memory>
18 #include <stdexcept>
19 #include <fstream>
20
27 class Logger {
28 public:
35     Logger(const std::string& filename);
36
43     void logError(const std::string& message, bool critical);
44 private:
45     std::ofstream logFile;
46     std::string logFilename;
47
52     std::string getCurrentDateTime();
53 };
54
62 class Server {
63 public:
74     Server(const std::string& database, unsigned short int port, const std::string& logfile);
75     ~Server();
76
83     void run();
84
85 private:
86     std::string database;
87     unsigned short int port;
88     std::string logfile;
89     Logger logger;
90     int serverSocket;
91     std::unique_ptr<sockaddr_in> self_addr;
92     std::unique_ptr<sockaddr_in> foreign_addr;
93 };
```

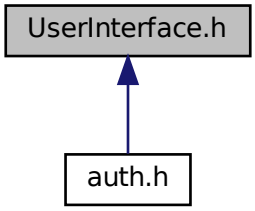
7.7 Файл UserInterface.h

Заголовочный файл для модуля пользовательского интерфейса UserInterface.h.

#include <boost/program_options.hpp>
#include <boost/filesystem.hpp>
#include <string>
Граф включаемых заголовочных файлов для UserInterface.h:



Граф файлов, в которые включается этот файл:



Классы

- struct Params

  Структура для хранения значений, полученных из командной строки
- class UserInterface

  Класс UserInterface.

Пространства имен

- namespace po

  Пространство имен библиотеки boost::program_options. Используется для парсинга командной строки
- namespace fs

  Пространство имен библиотеки boost::filesystem. Используется для проверки существования файлов

7.7.1 Подробное описание

Заголовочный файл для модуля пользовательского интерфейса UserInterface.h.

Автор

Храбров А.А.

Версия

1.0

Дата

18.12.2024

Авторство

ИБСТ ПГУ

Предупреждения

Тестовый

7.8 UserInterface.h

См. документацию.

```
1
10 #pragma once
11 #include <boost/program_options.hpp>
12 #include <boost/filesystem.hpp>
13 #include <string>
14
18 namespace po = boost::program_options;
19
23 namespace fs = boost::filesystem;
24
28 struct Params {
29     std::string logfile;
30     std::string database;
31     unsigned short int port;
32 };
33
36 class UserInterface {
37 private:
38     po::options_description desc;
39     po::variables_map vm;
40     Params params;
41
42 public:
47     UserInterface();
48
59     bool Parser(int argc, const char** argv);
60
67     bool fileExists(const std::string& filename);
68
73     std::string getDescription();
74
79     Params getParams() {
80         return params;
81     };
82 };
```

Предметный указатель