

server

Создано системой Doxygen 1.9.4

1	Алфавитный указатель пространств имен	1
1.1	Пространства имен	1
2	Иерархический список классов	3
2.1	Иерархия классов	3
3	Алфавитный указатель классов	5
3.1	Классы	5
4	Список файлов	7
4.1	Файлы	7
5	Пространства имен	9
5.1	Пространство имен <code>crutils</code>	9
5.1.1	Подробное описание	9
5.1.2	Функции	9
5.1.2.1	<code>calculateHash()</code>	9
5.1.2.2	<code>generateSalt()</code>	10
6	Классы	11
6.1	Класс <code>ArgsDecodeException</code>	11
6.1.1	Подробное описание	12
6.1.2	Конструктор(ы)	12
6.1.2.1	<code>ArgsDecodeException()</code>	12
6.2	Класс <code>AuthException</code>	13
6.2.1	Подробное описание	14
6.2.2	Конструктор(ы)	14
6.2.2.1	<code>AuthException()</code>	14
6.3	Класс <code>DataDecodeException</code>	14
6.3.1	Подробное описание	15
6.3.2	Конструктор(ы)	16
6.3.2.1	<code>DataDecodeException()</code>	16
6.4	Класс <code>Exception</code>	16
6.4.1	Подробное описание	17
6.4.2	Конструктор(ы)	17
6.4.2.1	<code>Exception()</code>	18
6.4.3	Методы	18
6.4.3.1	<code>what()</code>	18
6.5	Класс <code>Interface</code>	18
6.5.1	Подробное описание	20
6.5.2	Конструктор(ы)	20
6.5.2.1	<code>Interface()</code>	20
6.5.3	Методы	20
6.5.3.1	<code>getDbPath()</code>	20
6.5.3.2	<code>getLogPath()</code>	20

6.5.3.3	<code>getNetwork()</code>	21
6.5.3.4	<code>parseArgs()</code>	21
6.5.3.5	<code>readDB()</code>	21
6.5.3.6	<code>run()</code>	21
6.6	Класс <code>IOException</code>	22
6.6.1	Подробное описание	23
6.6.2	Конструктор(ы)	23
6.6.2.1	<code>IOException()</code>	23
6.7	Класс <code>Network</code>	24
6.7.1	Подробное описание	25
6.7.2	Конструктор(ы)	25
6.7.2.1	<code>Network()</code>	25
6.7.3	Методы	25
6.7.3.1	<code>auth()</code>	25
6.7.3.2	<code>calc()</code>	25
6.7.3.3	<code>getAddress()</code>	27
6.7.3.4	<code>getDatabase()</code>	27
6.7.3.5	<code>getPort()</code>	27
6.7.3.6	<code>wait()</code>	27
6.8	Класс <code>NetworkException</code>	28
6.8.1	Подробное описание	29
6.8.2	Конструктор(ы)	29
6.8.2.1	<code>NetworkException()</code>	29
7	Файлы	31
7.1	Файл <code>crutils.h</code>	31
7.1.1	Подробное описание	32
7.2	<code>crutils.h</code>	33
7.3	Файл <code>exceptions.h</code>	33
7.3.1	Подробное описание	34
7.4	<code>exceptions.h</code>	35
7.5	Файл <code>interface.h</code>	36
7.5.1	Подробное описание	37
7.6	<code>interface.h</code>	37
7.7	Файл <code>main.cpp</code>	38
7.7.1	Подробное описание	38
7.7.2	Функции	39
7.7.2.1	<code>main()</code>	39
7.8	Файл <code>network.h</code>	39
7.8.1	Подробное описание	40
7.9	<code>network.h</code>	41
	Предметный указатель	43

Глава 1

Алфавитный указатель пространств имен

1.1 Пространства имен

Полный список документированных пространств имен.

crutils	
Пространство имен для криптографических утилит	9

Глава 2

Иерархический список классов

2.1 Иерархия классов

Иерархия классов.

std::exception	
Exception	16
ArgsDecodeException	11
AuthException	13
DataDecodeException	14
IOException	22
NetworkException	28
Interface	18
Network	24

Глава 3

Алфавитный указатель классов

3.1 Классы

Классы с их кратким описанием.

ArgsDecodeException	Класс для исключений при декодировании аргументов	11
AuthException	Класс для исключений аутентификации	13
DataDecodeException	Класс для исключений при декодировании данных	14
Exception	Базовый класс для исключений	16
Interface	Класс для управления интерфейсом программы	18
IOException	Класс для исключений ввода-вывода	22
Network	Класс для управления сетевым подключением и взаимодействием	24
NetworkException	Класс для сетевых исключений	28

Глава 4

Список файлов

4.1 Файлы

Полный список документированных файлов.

crutils.h	Определения вспомогательных функций для криптографических операций	31
exceptions.h	Определение классов исключений	33
interface.h	Определение класса интерфейса	36
main.cpp	Главный файл программы	38
network.h	Определения классов для управления сетевым взаимодействием	39

Глава 5

Пространства имен

5.1 Пространство имен crutils

Пространство имен для криптографических утилит.

Функции

- `std::string generateSalt ()`
Функция для генерации соли.
- `std::string calculateHash (const std::string &data)`
Функция для вычисления хеша.

5.1.1 Подробное описание

Пространство имен для криптографических утилит.

5.1.2 Функции

5.1.2.1 calculateHash()

```
std::string crutils::calculateHash (  
    const std::string & data )
```

Функция для вычисления хеша.

Аргументы

data	Данные для хеширования.
------	-------------------------

Возвращает

Хеш в виде строки.

5.1.2.2 generateSalt()

`std::string crutils::generateSalt ()`

Функция для генерации соли.

Возвращает

Соль в виде строки.

Глава 6

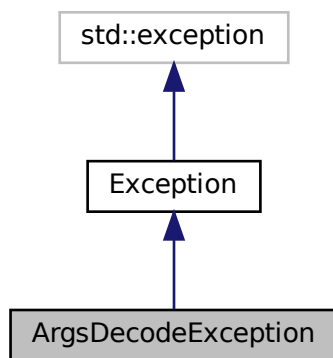
Классы

6.1 Класс ArgsDecodeException

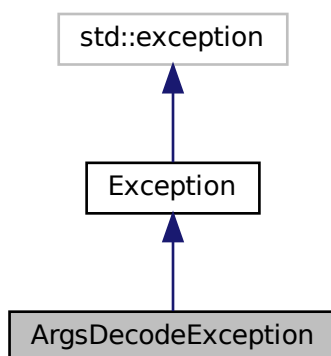
Класс для исключений при декодировании аргументов.

```
#include <exceptions.h>
```

Граф наследования:ArgsDecodeException:



Граф связей класса ArgsDecodeException:



Открытые члены

- [ArgsDecodeException](#) (const std::string &message, const std::string &func, const std::string &log_path, bool critical=false)

Конструктор класса [ArgsDecodeException](#).

6.1.1 Подробное описание

Класс для исключений при декодировании аргументов.

6.1.2 Конструктор(ы)

6.1.2.1 ArgsDecodeException()

```
ArgsDecodeException::ArgsDecodeException (  
    const std::string & message,  
    const std::string & func,  
    const std::string & log_path,  
    bool critical = false )
```

Конструктор класса [ArgsDecodeException](#).

Аргументы

message	Сообщение об ошибке.
func	Имя функции, в которой возникло исключение.
log_path	Путь к файлу журнала.
critical	Флаг критичности исключения.

Объявления и описания членов классов находятся в файлах:

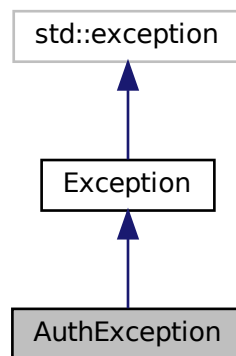
- [exceptions.h](#)
- exceptions.cpp

6.2 Класс AuthException

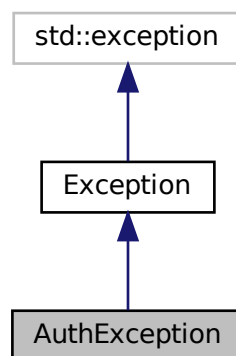
Класс для исключений аутентификации.

```
#include <exceptions.h>
```

Граф наследования:AuthException:



Граф связей класса AuthException:



Открытые члены

- [AuthException](#) (const std::string &message, const std::string &func, const std::string &log_path, bool critical=false)

Конструктор класса [AuthException](#).

6.2.1 Подробное описание

Класс для исключений аутентификации.

6.2.2 Конструктор(ы)

6.2.2.1 AuthException()

```
AuthException::AuthException (  
    const std::string & message,  
    const std::string & func,  
    const std::string & log_path,  
    bool critical = false )
```

Конструктор класса [AuthException](#).

Аргументы

message	Сообщение об ошибке.
func	Имя функции, в которой возникло исключение.
log_path	Путь к файлу журнала.
critical	Флаг критичности исключения.

Объявления и описания членов классов находятся в файлах:

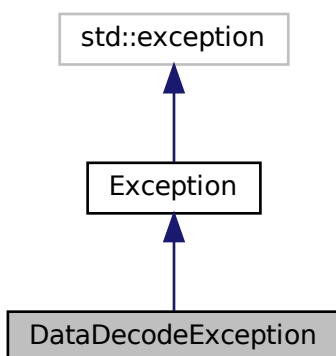
- [exceptions.h](#)
- [exceptions.cpp](#)

6.3 Класс DataDecodeException

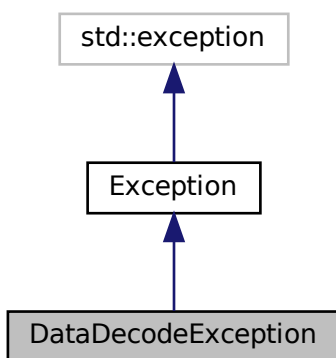
Класс для исключений при декодировании данных.

```
#include <exceptions.h>
```

Граф наследования:DataDecodeException:



Граф связей класса DataDecodeException:



Открытые члены

- [DataDecodeException](#) (const std::string &message, const std::string &func, const std::string &log_path, bool critical=false)
Конструктор класса [DataDecodeException](#).

6.3.1 Подробное описание

Класс для исключений при декодировании данных.

6.3.2 Конструктор(ы)

6.3.2.1 DataDecodeException()

```
DataDecodeException::DataDecodeException (
    const std::string & message,
    const std::string & func,
    const std::string & log_path,
    bool critical = false )
```

Конструктор класса [DataDecodeException](#).

Аргументы

message	Сообщение об ошибке.
func	Имя функции, в которой возникло исключение.
log_path	Путь к файлу журнала.
critical	Флаг критичности исключения.

Объявления и описания членов классов находятся в файлах:

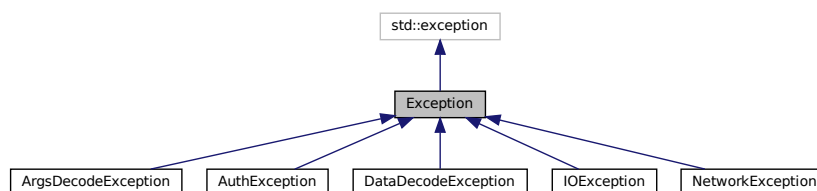
- [exceptions.h](#)
- [exceptions.cpp](#)

6.4 Класс Exception

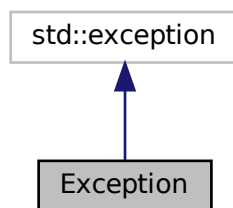
Базовый класс для исключений.

```
#include <exceptions.h>
```

Граф наследования:Exception:



Граф связей класса Exception:



Открытые члены

- `Exception` (const std::string &name, const std::string &message, const std::string &func, const std::string &log_path, bool critical=false)
Конструктор класса `Exception`.
- const char * `what` () const noexcept override
Метод для получения сообщения об ошибке.
- void `logException` () const
Метод для логирования исключения.

Закрытые данные

- std::string name
Имя исключения.
- std::string func
Имя функции, в которой возникло исключение.
- std::string log_path
Путь к файлу журнала.
- bool critical
Флаг критичности исключения.
- std::string message
Сообщение об ошибке.

6.4.1 Подробное описание

Базовый класс для исключений.

6.4.2 Конструктор(ы)

6.4.2.1 Exception()

```
Exception::Exception (
    const std::string & name,
    const std::string & message,
    const std::string & func,
    const std::string & log_path,
    bool critical = false )
```

Конструктор класса [Exception](#).

Аргументы

name	Имя исключения.
message	Сообщение об ошибке.
func	Имя функции, в которой возникло исключение.
log_path	Путь к файлу журнала.
critical	Флаг критичности исключения.

6.4.3 Методы

6.4.3.1 what()

```
const char * Exception::what ( ) const    [override], [noexcept]
```

Метод для получения сообщения об ошибке.

Возвращает

Сообщение об ошибке.

Объявления и описания членов классов находятся в файлах:

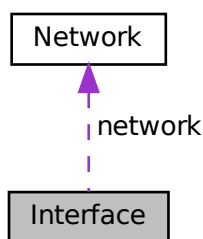
- [exceptions.h](#)
- exceptions.cpp

6.5 Класс Interface

Класс для управления интерфейсом программы.

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

Граф связей класса Interface:



Открытые члены

- [Interface](#) (int argc, char *argv[])
Конструктор принимает параметры командной строки.
- void [parseArgs](#) (int argc, char *argv[])
Метод для парсинга аргументов командной строки.
- void [readDB](#) ()
Метод для чтения базы данных паролей и записи в атрибут map.
- void [run](#) ()
Метод для запуска сервера.
- void [showHelp](#) () const
Метод для вывода справки.
- [Network](#) * [getNetwork](#) ()
Метод для получения объекта [Network](#).
- std::string & [getDbPath](#) ()
Метод для получения пути к файлу базы данных.
- std::string & [getLogPath](#) ()
Метод для получения пути к файлу логов.

Закрытые данные

- [Network](#) * network
Указатель на объект класса [Network](#).
- std::string db_path
Путь к файлу базы данных.
- std::string log_path
Путь к файлу логов.
- int port
Порт.
- std::string address
Адрес.
- std::map< std::string, std::string > database
Словарь для хранения базы данных паролей.

6.5.1 Подробное описание

Класс для управления интерфейсом программы.

6.5.2 Конструктор(ы)

6.5.2.1 Interface()

```
Interface::Interface (
    int argc,
    char * argv[] )
```

Конструктор принимает параметры командной строки.

Аргументы

argc	Количество аргументов командной строки.
argv	Аргументы командной строки.

Исключения

ArgsDecodeException	Если аргументы командной строки некорректны.
-------------------------------------	--

6.5.3 Методы

6.5.3.1 getDbPath()

```
std::string & Interface::getDbPath ( )
```

Метод для получения пути к файлу базы данных.

Возвращает

Путь к файлу базы данных.

6.5.3.2 getLogPath()

```
std::string & Interface::getLogPath ( )
```

Метод для получения пути к файлу логов.

Возвращает

Путь к файлу логов.

6.5.3.3 getNetwork()

```
Network * Interface::getNetwork ( )
```

Метод для получения объекта [Network](#).

Возвращает

Указатель на объект класса [Network](#).

6.5.3.4 parseArgs()

```
void Interface::parseArgs (
    int argc,
    char * argv[] )
```

Метод для парсинга аргументов командной строки.

Аргументы

argc	Количество аргументов командной строки.
argv	Аргументы командной строки.

Исключения

ArgsDecodeException	Если аргументы командной строки некорректны.
-------------------------------------	--

6.5.3.5 readDB()

```
void Interface::readDB ( )
```

Метод для чтения базы данных паролей и записи в атрибут map.

Исключения

IOException	Если не удалось открыть или прочитать файл базы данных.
DataDecodeException	Если строки в базе данных имеют некорректный формат или произошла ошибка при чтении файла.

6.5.3.6 run()

```
void Interface::run ( )
```

Метод для запуска сервера.

Исключения

NetworkException	Если возникли ошибки при запуске сервера.
----------------------------------	---

Объявления и описания членов классов находятся в файлах:

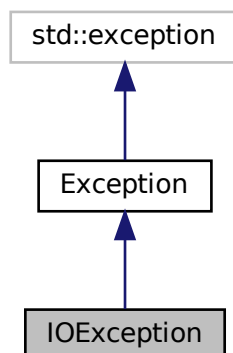
- [interface.h](#)
- [interface.cpp](#)

6.6 Класс IOException

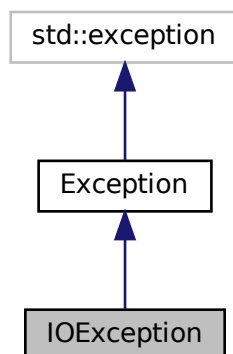
Класс для исключений ввода-вывода.

```
#include <exceptions.h>
```

Граф наследования:IOException:



Граф связей класса IOException:



Открытые члены

- [IOException](#) (const std::string &message, const std::string &func, const std::string &log_path, bool critical=false)

Конструктор класса [IOException](#).

6.6.1 Подробное описание

Класс для исключений ввода-вывода.

6.6.2 Конструктор(ы)

6.6.2.1 IOException()

```

IOException::IOException (
    const std::string & message,
    const std::string & func,
    const std::string & log_path,
    bool critical = false )
  
```

Конструктор класса [IOException](#).

Аргументы

message	Сообщение об ошибке.
func	Имя функции, в которой возникло исключение.
log_path	Путь к файлу журнала.
critical	Флаг критичности исключения.

Объявления и описания членов классов находятся в файлах:

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

6.7 Класс Network

Класс для управления сетевым подключением и взаимодействием.

```
#include <network.h>
```

Открытые члены

- [Network](#) (const std::string &[address](#), uint16_t [port](#), const std::map< std::string, std::string > &[database](#), const std::string &[log_path](#))
Конструктор класса [Network](#).
- std::string & [getAddress](#) ()
Метод для получения адреса сервера.
- uint16_t & [getPort](#) ()
Метод для получения порта сервера.
- std::map< std::string, std::string > & [getDatabase](#) ()
Метод для получения базы данных пользователей.
- void [wait](#) ()
Метод для ожидания соединений.
- void [auth](#) ()
Метод для аутентификации пользователя.
- void [calc](#) ()
Метод для вычисления суммы значений в векторах.
- void [close](#) ()
Метод для закрытия клиентского соединения.
- void [quit](#) ()
Метод для закрытия основного соединения.

Закрытые данные

- std::string [address](#)
Адрес сервера.
- uint16_t [port](#)
Порт сервера.
- std::map< std::string, std::string > [database](#)
База данных пользователей.
- int [socket](#)
Сокет сервера.
- int [client_socket](#)
Сокет клиента.
- std::string [log_path](#)
Путь к файлу логов.

6.7.1 Подробное описание

Класс для управления сетевым подключением и взаимодействием.

6.7.2 Конструктор(ы)

6.7.2.1 Network()

```
Network::Network (
    const std::string & address,
    uint16_t port,
    const std::map< std::string, std::string > & database,
    const std::string & log_path )
```

Конструктор класса [Network](#).

Аргументы

address	Адрес сервера.
port	Порт сервера.
database	База данных пользователей.
log_path	Путь к файлу логов.

6.7.3 Методы

6.7.3.1 auth()

```
void Network::auth ( )
```

Метод для аутентификации пользователя.

Исключения

NetworkException	Если не удалось прочесть логин, найти логин в базе данных, отправить или получить данные от клиента.
----------------------------------	--

6.7.3.2 calc()

```
void Network::calc ( )
```

Метод для вычисления суммы значений в векторах.

Исключения

NetworkException	Если не удалось прочитать данные от клиента или отправить результат.
----------------------------------	--

6.7.3.3 getAddress()

```
std::string & Network::getAddress ( )
```

Метод для получения адреса сервера.

Возвращает

Адрес сервера.

6.7.3.4 getDatabase()

```
std::map< std::string, std::string > & Network::getDatabase ( )
```

Метод для получения базы данных пользователей.

Возвращает

База данных пользователей.

6.7.3.5 getPort()

```
uint16_t & Network::getPort ( )
```

Метод для получения порта сервера.

Возвращает

Порт сервера.

6.7.3.6 wait()

```
void Network::wait ( )
```

Метод для ожидания соединений.

Исключения

NetworkException	Если не удалось создать, привязать или слушать сокет.
----------------------------------	---

Объявления и описания членов классов находятся в файлах:

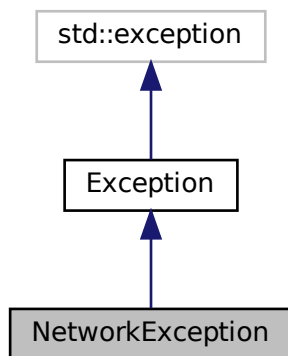
- [network.h](#)
- [network.cpp](#)

6.8 Класс NetworkException

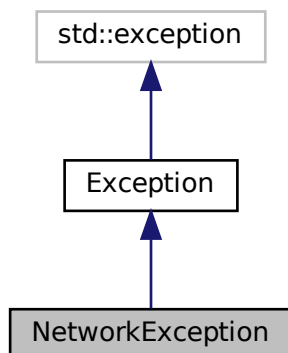
Класс для сетевых исключений.

```
#include <exceptions.h>
```

Граф наследования:NetworkException:



Граф связей класса NetworkException:



Открытые члены

- `NetworkException` (`const std::string &message`, `const std::string &func`, `const std::string &log_path`, `bool critical=false`)

Конструктор класса `NetworkException`.

6.8.1 Подробное описание

Класс для сетевых исключений.

6.8.2 Конструктор(ы)

6.8.2.1 `NetworkException()`

```
NetworkException::NetworkException (  
    const std::string & message,  
    const std::string & func,  
    const std::string & log_path,  
    bool critical = false )
```

Конструктор класса `NetworkException`.

Аргументы

<code>message</code>	Сообщение об ошибке.
<code>func</code>	Имя функции, в которой возникло исключение.
<code>log_path</code>	Путь к файлу журнала.
<code>critical</code>	Флаг критичности исключения.

Объявления и описания членов классов находятся в файлах:

- `exceptions.h`
- `exceptions.cpp`

Глава 7

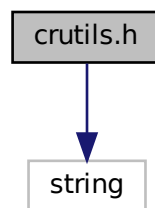
Файлы

7.1 Файл crutils.h

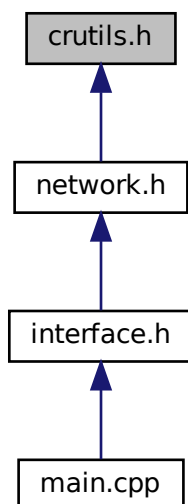
Определения вспомогательных функций для криптографических операций.

```
#include <string>
```

Граф включаемых заголовочных файлов для crutils.h:



Граф файлов, в которые включается этот файл:



Пространства имен

- namespace `crutils`
Пространство имен для криптографических утилит.

Функции

- `std::string crutils::generateSalt ()`
Функция для генерации соли.
- `std::string crutils::calculateHash (const std::string &data)`
Функция для вычисления хеша.

7.1.1 Подробное описание

Определения вспомогательных функций для криптографических операций.

Этот файл содержит определения функций для генерации соли и вычисления хеша.

Дата

23.11.2024

Версия

1.0

Автор

Мамелин Д. А.

7.2 crutils.h

[См. документацию.](#)

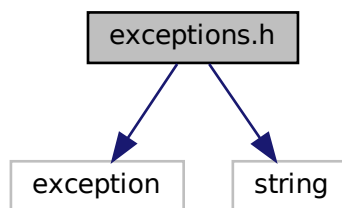
```
1 #ifndef CRUTILS_H
2 #define CRUTILS_H
3
4 #include <string>
5
18 namespace crutils {
23     std::string generateSalt();
24
30     std::string calculateHash(const std::string &data);
31 }
32
33 #endif // CRUTILS_H
```

7.3 Файл exceptions.h

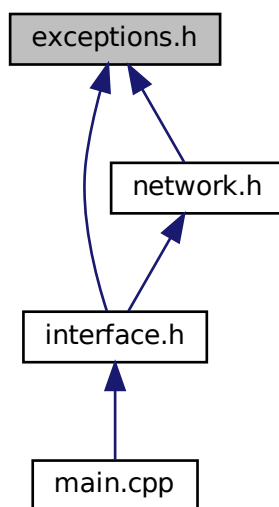
Определение классов исключений.

```
#include <exception>
#include <string>
```

Граф включаемых заголовочных файлов для exceptions.h:



Граф файлов, в которые включается этот файл:



Классы

- class [Exception](#)
Базовый класс для исключений.
- class [IOException](#)
Класс для исключений ввода-вывода.
- class [ArgsDecodeException](#)
Класс для исключений при декодировании аргументов.
- class [DataDecodeException](#)
Класс для исключений при декодировании данных.
- class [AuthException](#)
Класс для исключений аутентификации.
- class [NetworkException](#)
Класс для сетевых исключений.

7.3.1 Подробное описание

Определение классов исключений.

Этот файл содержит определения классов исключений для обработки различных ошибок в программе.

Дата

23.11.2024

Версия

1.0

Автор

Мамелин Д. А.

7.4 exceptions.h

[См. документацию.](#)

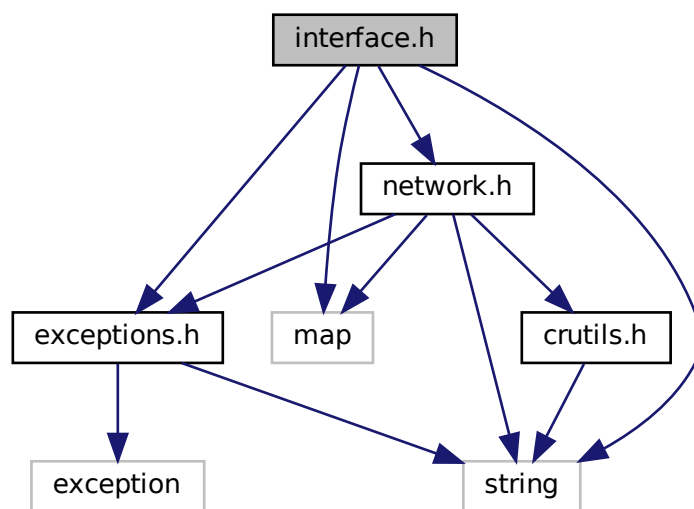
```
1 #ifndef EXCEPTIONS_H
2 #define EXCEPTIONS_H
3
4 #include <exception>
5 #include <string>
6
19 class Exception : public std::exception {
20 public:
29     Exception(const std::string &name,
30               const std::string &message,
31               const std::string &func,
32               const std::string &log_path,
33               bool critical = false);
34
39     const char* what() const noexcept override;
40
44     void logException() const;
45
46 private:
47     std::string name;
48     std::string func;
49     std::string log_path;
50     bool critical;
51     mutable std::string message;
52 };
53
57 class IOException : public Exception {
58 public:
66     IOException(const std::string &message,
67                 const std::string &func,
68                 const std::string &log_path,
69                 bool critical = false);
70 };
71
75 class ArgsDecodeException : public Exception {
76 public:
84     ArgsDecodeException(const std::string &message,
85                         const std::string &func,
86                         const std::string &log_path,
87                         bool critical = false);
88 };
89
93 class DataDecodeException : public Exception {
94 public:
102     DataDecodeException(const std::string &message,
103                        const std::string &func,
104                        const std::string &log_path,
105                        bool critical = false);
106 };
107
111 class AuthException : public Exception {
112 public:
120     AuthException(const std::string &message,
121                  const std::string &func,
122                  const std::string &log_path,
123                  bool critical = false);
124 };
125
129 class NetworkException : public Exception {
130 public:
138     NetworkException(const std::string &message,
139                     const std::string &func,
140                     const std::string &log_path,
141                     bool critical = false);
142 };
143
144 #endif // EXCEPTIONS_H
```

7.5 Файл interface.h

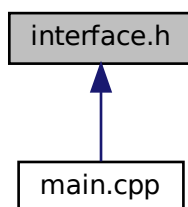
Определение класса интерфейса.

```
#include <map>
#include <string>
#include "exceptions.h"
#include "network.h"
```

Граф включаемых заголовочных файлов для interface.h:



Граф файлов, в которые включается этот файл:



Классы

- class [Interface](#)

Класс для управления интерфейсом программы.

7.5.1 Подробное описание

Определение класса интерфейса.

Этот файл содержит определения классов для обработки параметров командной строки, чтения базы данных паролей и запуска сервера.

Дата

23.11.2024

Версия

1.0

Автор

Мамелин Д. А.

7.6 interface.h

[См. документацию.](#)

```
1 #ifndef INTERFACE_H
2 #define INTERFACE_H
3
4 #include <map>
5 #include <string>
6 #include "exceptions.h"
7 #include "network.h"
8
9
10
11
12
13
14
15
16
17
18
19
20
21 class Interface
22 {
23 public:
24     Interface(int argc, char *argv[]);
25
26     void parseArgs(int argc, char *argv[]);
27
28     void readDB();
29
30     void run();
31
32     void showHelp() const;
33
34     Network *getNetwork();
35
36     std::string &getDbPath();
37
38     std::string &getLogPath();
39
40 private:
41     Network *network;
42     std::string db_path;
43     std::string log_path;
44     int port;
45     std::string address;
46     std::map<std::string, std::string> database;
47 };
48
49 #endif // INTERFACE_H
```

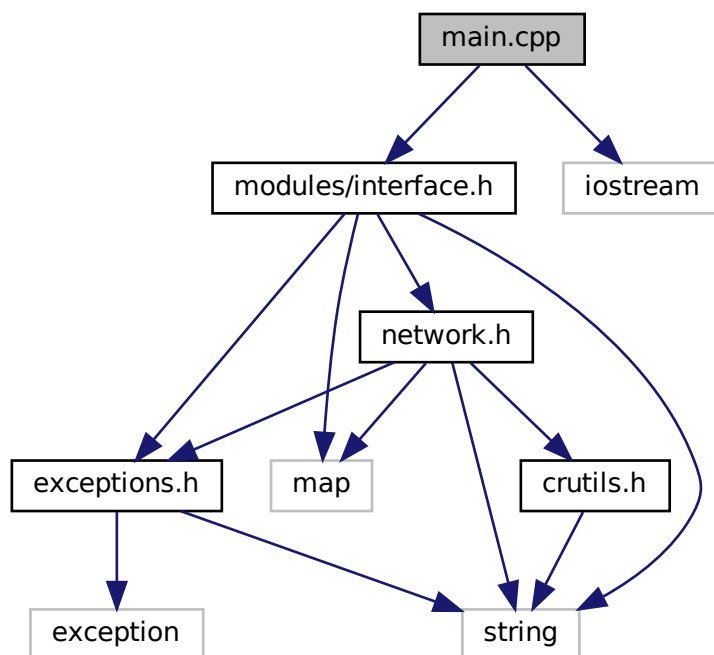
7.7 Файл main.cpp

Главный файл программы.

```
#include "modules/interface.h"
```

```
#include <iostream>
```

Граф включаемых заголовочных файлов для main.cpp:



Функции

- int `main` (int argc, char *argv[])

Главная функция программы.

7.7.1 Подробное описание

Главный файл программы.

Этот файл содержит функцию `main`, которая инициализирует интерфейс и запускает сервер.

Дата

23.11.2024

Версия

1.0

Автор

Мамелин Д. А.

7.7.2 Функции

7.7.2.1 main()

```
int main (
    int argc,
    char * argv[] )
```

Главная функция программы.

Инициализирует объект [Interface](#) и запускает его. Обрабатывает все исключения, возникающие во время выполнения программы.

Аргументы

argc	Количество аргументов командной строки.
argv	Аргументы командной строки.

Возвращает

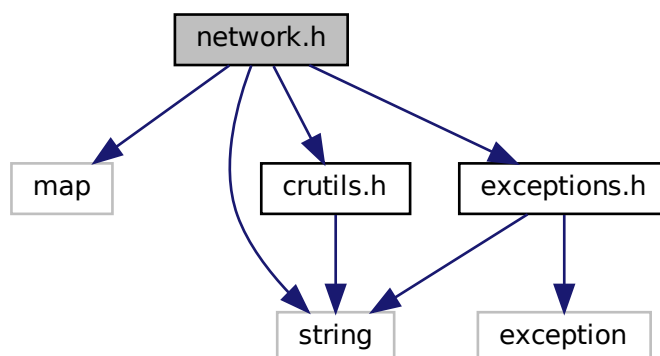
Код завершения программы. 0 - успешное завершение, 1 - ошибка.

7.8 Файл network.h

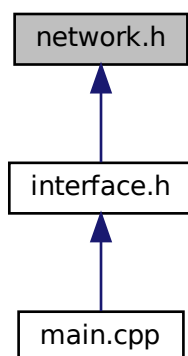
Определения классов для управления сетевым взаимодействием.

```
#include <map>
#include <string>
#include "crutils.h"
#include "exceptions.h"
```

Граф включаемых заголовочных файлов для network.h:



Граф файлов, в которые включается этот файл:



Классы

- class [Network](#)

Класс для управления сетевым подключением и взаимодействием.

7.8.1 Подробное описание

Определения классов для управления сетевым взаимодействием.

Этот файл содержит определения классов для управления сетевыми подключениями и передачей данных.

Дата

23.11.2024

Версия

1.0

Автор

Мамелин Д. А.

7.9 network.h

[См. документацию.](#)

```
1 #ifndef NETWORK_H
2 #define NETWORK_H
3
4 #include <map>
5 #include <string>
6 #include "crutils.h"
7 #include "exceptions.h"
8
9 class Network
10 {
11 public:
12     Network(
13         const std::string &address,
14         uint16_t port,
15         const std::map<std::string, std::string> &database,
16         const std::string &log_path);
17
18     std::string &getAddress();
19
20     uint16_t &getPort();
21
22     std::map<std::string, std::string> &getDatabase();
23
24     void wait();
25
26     void auth();
27
28     void calc();
29
30     void close();
31
32     void quit();
33 private:
34     std::string address;
35     uint16_t port;
36     std::map<std::string, std::string> database;
37     int socket;
38     int client_socket;
39     std::string log_path;
40 };
41
42 #endif // NETWORK_H
```


Предметный указатель

- ArgsDecodeException, [11](#)
 - ArgsDecodeException, [12](#)
- auth
 - Network, [25](#)
- AuthException, [13](#)
 - AuthException, [14](#)
- calc
 - Network, [25](#)
- calculateHash
 - crutils, [9](#)
- crutils, [9](#)
 - calculateHash, [9](#)
 - generateSalt, [10](#)
- crutils.h, [31](#)
- DataDecodeException, [14](#)
 - DataDecodeException, [16](#)
- Exception, [16](#)
 - Exception, [17](#)
 - what, [18](#)
- exceptions.h, [33](#)
- generateSalt
 - crutils, [10](#)
- getAddress
 - Network, [27](#)
- getDatabase
 - Network, [27](#)
- getDbPath
 - Interface, [20](#)
- getLogPath
 - Interface, [20](#)
- getNetwork
 - Interface, [20](#)
- getPort
 - Network, [27](#)
- Interface, [18](#)
 - getDbPath, [20](#)
 - getLogPath, [20](#)
 - getNetwork, [20](#)
 - Interface, [20](#)
 - parseArgs, [21](#)
 - readDB, [21](#)
 - run, [21](#)
- interface.h, [36](#)
- IOException, [22](#)
 - IOException, [23](#)
- main
 - main.cpp, [39](#)
- main.cpp, [38](#)
 - main, [39](#)
- Network, [24](#)
 - auth, [25](#)
 - calc, [25](#)
 - getAddress, [27](#)
 - getDatabase, [27](#)
 - getPort, [27](#)
 - Network, [25](#)
 - wait, [27](#)
- network.h, [39](#)
- NetworkException, [28](#)
 - NetworkException, [29](#)
- parseArgs
 - Interface, [21](#)
- readDB
 - Interface, [21](#)
- run
 - Interface, [21](#)
- wait
 - Network, [27](#)
- what
 - Exception, [18](#)