

Разработка сервера

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 Список файлов	5
3.1 Файлы	5
4 Классы	7
4.1 Класс ArgsDecodeException	7
4.1.1 Подробное описание	8
4.1.2 Конструктор(ы)	8
4.1.2.1 ArgsDecodeException()	8
4.2 Класс AuthException	9
4.2.1 Подробное описание	10
4.2.2 Конструктор(ы)	10
4.2.2.1 AuthException()	10
4.3 Класс DataDecodeException	10
4.3.1 Подробное описание	11
4.3.2 Конструктор(ы)	12
4.3.2.1 DataDecodeException()	12
4.4 Класс Decoder	12
4.4.1 Подробное описание	13
4.4.2 Конструктор(ы)	13
4.4.2.1 Decoder()	13
4.4.3 Методы	13
4.4.3.1 getAddress()	13
4.4.3.2 getDataBasePath()	13
4.4.3.3 getLogPath()	14
4.4.3.4 getPort()	14
4.4.3.5 parse()	14
4.4.3.6 showHelp()	14
4.5 Класс Exception	15
4.5.1 Подробное описание	16
4.5.2 Конструктор(ы)	16
4.5.2.1 Exception()	16
4.5.3 Методы	17
4.5.3.1 logException()	17
4.5.3.2 what()	17
4.5.4 Данные класса	17
4.5.4.1 critical	17
4.5.4.2 func	17
4.5.4.3 log_path	17

4.5.4.4 message . . . . .	18
4.5.4.5 name . . . . .	18
4.6 Класс IOException . . . . .	18
4.6.1 Подробное описание . . . . .	19
4.6.2 Конструктор(ы) . . . . .	19
4.6.2.1 IOException() . . . . .	19
4.7 Класс Network . . . . .	20
4.7.1 Подробное описание . . . . .	20
4.7.2 Конструктор(ы) . . . . .	20
4.7.2.1 Network() . . . . .	20
4.7.3 Методы . . . . .	21
4.7.3.1 auth() . . . . .	21
4.7.3.2 getAddress() . . . . .	21
4.7.3.3 getDatabase() . . . . .	21
4.7.3.4 getPort() . . . . .	22
4.7.3.5 start() . . . . .	22
4.7.3.6 stop() . . . . .	22
4.7.3.7 sum() . . . . .	22
4.7.3.8 wait() . . . . .	22
4.8 Класс NetworkException . . . . .	23
4.8.1 Подробное описание . . . . .	24
4.8.2 Конструктор(ы) . . . . .	24
4.8.2.1 NetworkException() . . . . .	24
5 Файлы . . . . .	25
5.1 Файл Desktop/sasha/decoder.cpp . . . . .	25
5.2 Файл Desktop/sasha/decoder.h . . . . .	25
5.2.1 Подробное описание . . . . .	26
5.3 Файл Desktop/sasha/exceptions.cpp . . . . .	27
5.4 Файл Desktop/sasha/exceptions.h . . . . .	27
5.4.1 Подробное описание . . . . .	28
5.5 Файл Desktop/sasha/hash.cpp . . . . .	29
5.5.1 Функции . . . . .	29
5.5.1.1 getHash() . . . . .	29
5.5.1.2 getSalt() . . . . .	29
5.6 Файл Desktop/sasha/hash.h . . . . .	30
5.6.1 Подробное описание . . . . .	31
5.6.2 Функции . . . . .	31
5.6.2.1 getHash() . . . . .	31
5.6.2.2 getSalt() . . . . .	31
5.7 Файл Desktop/sasha/main.cpp . . . . .	32
5.7.1 Функции . . . . .	32
5.7.1.1 getDataBase() . . . . .	32

---

5.7.1.2 loop()	33
5.7.1.3 main()	33
5.7.2 Переменные	34
5.7.2.1 LOG_PATH	34
5.8 Файл Desktop/sasha/main1.cpp	34
5.8.1 Функции	34
5.8.1.1 main()	34
5.8.1.2 TEST() [1/9]	35
5.8.1.3 TEST() [2/9]	35
5.8.1.4 TEST() [3/9]	35
5.8.1.5 TEST() [4/9]	35
5.8.1.6 TEST() [5/9]	35
5.8.1.7 TEST() [6/9]	35
5.8.1.8 TEST() [7/9]	35
5.8.1.9 TEST() [8/9]	36
5.8.1.10 TEST() [9/9]	36
5.9 Файл Desktop/sasha/network.cpp	36
5.10 Файл Desktop/sasha/network.h	36
5.10.1 Подробное описание	37
Предметный указатель	39



# Глава 1

## Иерархический список классов

### 1.1 Иерархия классов

Иерархия классов.

Decoder . . . . .	12
std::exception	
Exception . . . . .	15
ArgsDecodeException . . . . .	7
AuthException . . . . .	9
DataDecodeException . . . . .	10
IOException . . . . .	18
NetworkException . . . . .	23
Network . . . . .	20





## Глава 2

# Алфавитный указатель классов

### 2.1 Классы

Классы с их кратким описанием.

<a href="#">ArgsDecodeException</a>	Класс для исключений при декодировании аргументов . . . . .	7
<a href="#">AuthException</a>	Класс для исключений аутентификации . . . . .	9
<a href="#">DataDecodeException</a>	Класс для исключений при декодировании данных . . . . .	10
<a href="#">Decoder</a>	Класс для управления интерфейсом программы . . . . .	12
<a href="#">Exception</a>	Базовый класс для исключений . . . . .	15
<a href="#">IOException</a>	Класс для исключений ввода-вывода . . . . .	18
<a href="#">Network</a>	Класс для управления сетевым подключением и взаимодействием . . . . .	20
<a href="#">NetworkException</a>	Класс для сетевых исключений . . . . .	23



## Глава 3

# Список файлов

### 3.1 Файлы

Полный список файлов.

Desktop/sasha/ <a href="#">decoder.cpp</a> . . . . .	25
Desktop/sasha/ <a href="#">decoder.h</a>	
Определение класса интерфейса . . . . .	25
Desktop/sasha/ <a href="#">exceptions.cpp</a> . . . . .	27
Desktop/sasha/ <a href="#">exceptions.h</a>	
Определение классов исключений . . . . .	27
Desktop/sasha/ <a href="#">hash.cpp</a> . . . . .	29
Desktop/sasha/ <a href="#">hash.h</a>	
Определения вспомогательных функций для криптографических операций . . . . .	30
Desktop/sasha/ <a href="#">main.cpp</a> . . . . .	32
Desktop/sasha/ <a href="#">main1.cpp</a> . . . . .	34
Desktop/sasha/ <a href="#">network.cpp</a> . . . . .	36
Desktop/sasha/ <a href="#">network.h</a>	
Определения классов для управления сетевым взаимодействием . . . . .	36



## Глава 4

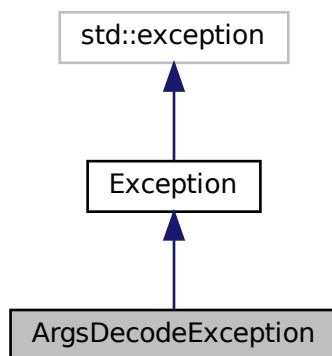
# Классы

### 4.1 Класс ArgsDecodeException

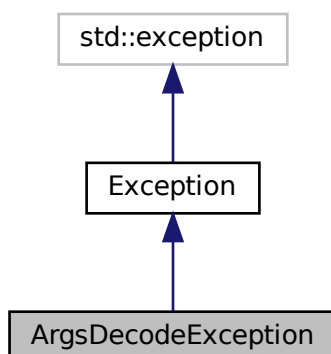
Класс для исключений при декодировании аргументов.

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

Граф наследования:ArgsDecodeException:



Граф связей класса ArgsDecodeException:



## Открытые члены

- [ArgsDecodeException](#) (const string &[message](#), const string &[func](#), const string &[log\\_path](#), bool [critical](#))

Конструктор класса [ArgsDecodeException](#).

## Дополнительные унаследованные члены

### 4.1.1 Подробное описание

Класс для исключений при декодировании аргументов.

### 4.1.2 Конструктор(ы)

#### 4.1.2.1 ArgsDecodeException()

```

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

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:

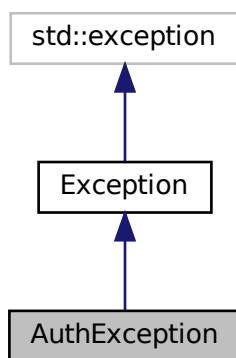
- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

## 4.2 Класс AuthException

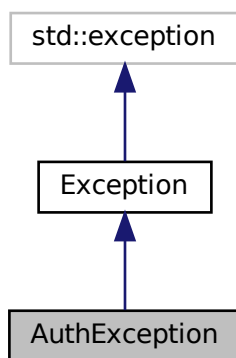
Класс для исключений аутентификации.

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

Граф наследования:AuthException:



Граф связей класса AuthException:



## Открытые члены

- [AuthException](#) (const string &message, const string &func, const string &log\_path, bool critical)  
Конструктор класса [AuthException](#).

## Дополнительные унаследованные члены

### 4.2.1 Подробное описание

Класс для исключений аутентификации.

### 4.2.2 Конструктор(ы)

#### 4.2.2.1 AuthException()

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

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:

- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

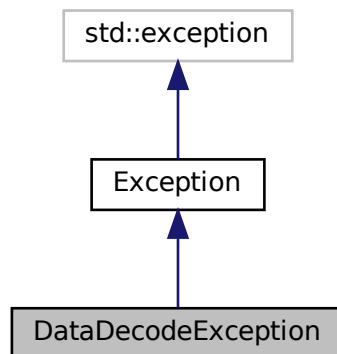
## 4.3 Класс DataDecodeException

Класс для исключений при декодировании данных.

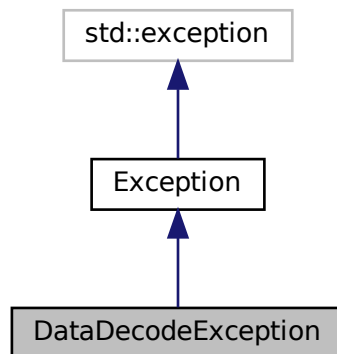
```
#include <exceptions.h>
```



Граф наследования:DataDecodeException:



Граф связей класса DataDecodeException:



Открытые члены

- [DataDecodeException](#) (const string &[message](#), const string &[func](#), const string &[log\\_path](#), bool [critical](#))

Конструктор класса [DataDecodeException](#).

Дополнительные унаследованные члены

#### 4.3.1 Подробное описание

Класс для исключений при декодировании данных.

### 4.3.2 Конструктор(ы)

#### 4.3.2.1 DataDecodeException()

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

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:

- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

## 4.4 Класс Decoder

Класс для управления интерфейсом программы.

```
#include <decoder.h>
```

Открытые члены

- [Decoder](#) ()  
Конструктор принимает параметры командной строки.
- void [parse](#) (int argc, char \*argv[])  
Метод для парсинга аргументов командной строки.
- void [showHelp](#) () const  
Метод для получения справки.
- string & [getDataBasePath](#) ()  
Метод для получения пути к базе данных.
- string & [getLogPath](#) ()  
Метод для получения пути к файлу логов.
- string & [getAddres](#) ()  
Метод для получения адреса сервера.
- int & [getPort](#) ()  
Метод для получения порта сервера.

#### 4.4.1 Подробное описание

Класс для управления интерфейсом программы.

#### 4.4.2 Конструктор(ы)

##### 4.4.2.1 Decoder()

```
Decoder::Decoder ( )
```

Конструктор принимает параметры командной строки.

Исключения

<a href="#">ArgsDecodeException</a>	Если аргументы командной строки некорректны.
-------------------------------------	--

#### 4.4.3 Методы

##### 4.4.3.1 getAddress()

```
string & Decoder::getAddress ( )
```

Метод для получения адреса сервера.

Возвращает

Ссылка на строку с адресом сервера.

##### 4.4.3.2 getDataBasePath()

```
string & Decoder::getDataBasePath ( )
```

Метод для получения пути к базе данных.

Возвращает

Ссылка на строку с путем к базе данных.

#### 4.4.3.3 getLogPath()

```
string & Decoder::getLogPath ( )
```

Метод для получения пути к файлу логов.

Возвращает

Ссылка на строку с путем к файлу логов.

#### 4.4.3.4 getPort()

```
int & Decoder::getPort ( )
```

Метод для получения порта сервера.

Возвращает

Ссылка на целое число с номером порта сервера.

#### 4.4.3.5 parse()

```
void Decoder::parse (
    int argc,
    char * argv[] )
```

Метод для парсинга аргументов командной строки.

Аргументы

argc	Количество аргументов командной строки.
argv	Аргументы командной строки.

Исключения

<a href="#">ArgsDecodeException</a>	Если аргументы командной строки некорректны.
-------------------------------------	--

#### 4.4.3.6 showHelp()

```
void Decoder::showHelp ( ) const
```

Метод для получения справки.

Возвращает

Справка по использованию.

Объявления и описания членов классов находятся в файлах:

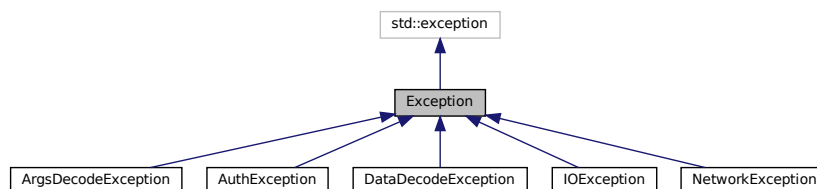
- Desktop/sasha/[decoder.h](#)
- Desktop/sasha/[decoder.cpp](#)

## 4.5 Класс Exception

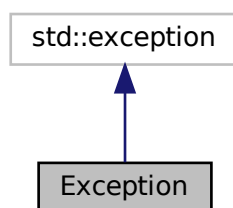
Базовый класс для исключений.

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

Граф наследования:Exception:



Граф связей класса Exception:



Открытые члены

- [Exception](#) (const string &[name](#), const string &[message](#), const string &[func](#), const string &[log\\_path](#), bool [critical](#))  
Конструктор класса [Exception](#).
- const char \* [what](#) () const noexcept override  
Метод для получения сообщения об ошибке.

## Защищенные члены

- void `logException` () const  
Метод для логирования исключения.

## Защищенные данные

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

### 4.5.1 Подробное описание

Базовый класс для исключений.

### 4.5.2 Конструктор(ы)

#### 4.5.2.1 Exception()

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

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.

### 4.5.3 Методы

#### 4.5.3.1 logException()

```
void Exception::logException ( ) const [protected]
```

Метод для логирования исключения.

#### 4.5.3.2 what()

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

Метод для получения сообщения об ошибке.

Возвращает

Сообщение об ошибке.

### 4.5.4 Данные класса

#### 4.5.4.1 critical

```
bool Exception::critical [protected]
```

Флаг критичности исключения.

#### 4.5.4.2 func

```
string Exception::func [protected]
```

Имя функции, в которой возникло исключение.

#### 4.5.4.3 log\_path

```
string Exception::log_path [protected]
```

Путь к файлу журнала.

#### 4.5.4.4 message

```
string Exception::message [mutable], [protected]
```

Сообщение об ошибке.

#### 4.5.4.5 name

```
string Exception::name [protected]
```

Имя исключения.

Объявления и описания членов классов находятся в файлах:

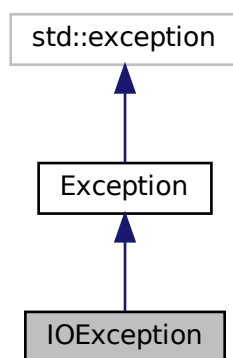
- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

## 4.6 Класс IOException

Класс для исключений ввода-вывода.

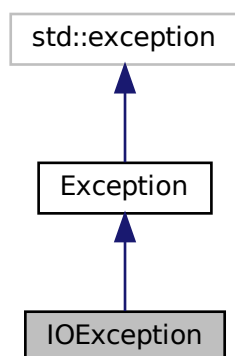
```
#include <exceptions.h>
```

Граф наследования:IOException:





Граф связей класса IOException:



Открытые члены

- [IOException](#) (const string &message, const string &func, const string &log\_path, bool critical)  
Конструктор класса [IOException](#).

Дополнительные унаследованные члены

#### 4.6.1 Подробное описание

Класс для исключений ввода-вывода.

#### 4.6.2 Конструктор(ы)

##### 4.6.2.1 IOException()

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

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:

- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

## 4.7 Класс Network

Класс для управления сетевым подключением и взаимодействием.

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

Открытые члены

- [Network](#) (const string &address, uint16\_t port, const map< string, string > &database, const string &log\_path)  
Конструктор класса [Network](#).
- string & [getAddress](#) ()  
Метод для получения адреса сервера.
- uint16\_t & [getPort](#) ()  
Метод для получения порта сервера.
- map< string, string > & [getDatabase](#) ()  
Метод для получения базы данных пользователей.
- void [start](#) ()  
Метод для старта работы сервера.
- void [stop](#) ()  
Метод для закрытия основного соединения.
- void [wait](#) ()  
Метод для ожидания соединений.
- void [auth](#) ()  
Метод для аутентификации пользователя.
- void [sum](#) ()  
Метод для вычисления суммы значений в векторах.

### 4.7.1 Подробное описание

Класс для управления сетевым подключением и взаимодействием.

### 4.7.2 Конструктор(ы)

#### 4.7.2.1 Network()

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

Конструктор класса [Network](#).

## Аргументы

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

## 4.7.3 Методы

## 4.7.3.1 auth()

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

Метод для аутентификации пользователя.

## Исключения

<a href="#">NetworkException</a>	Если не удалось прочитать логин, найти логин в базе данных, отправить или получить данные от клиента.
----------------------------------	---

## 4.7.3.2 getAddress()

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

Метод для получения адреса сервера.

Возвращает

Адрес сервера.

## 4.7.3.3 getDatabase()

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

Метод для получения базы данных пользователей.

Возвращает

База данных пользователей.

#### 4.7.3.4 getPort()

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

Метод для получения порта сервера.

Возвращает

Порт сервера.

#### 4.7.3.5 start()

```
void Network::start ( )
```

Метод для старта работы сервера.

#### 4.7.3.6 stop()

```
void Network::stop ( )
```

Метод для закрытия основного соединения.

#### 4.7.3.7 sum()

```
void Network::sum ( )
```

Метод для вычисления суммы значений в векторах.

Исключения

<a href="#">NetworkException</a>	Если не удалось прочитать данные от клиента или отправить результат.
----------------------------------	--

#### 4.7.3.8 wait()

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

Метод для ожидания соединений.

Исключения

<a href="#">NetworkException</a>	Если не удалось создать, привязать или слушать сокет.
----------------------------------	---

Объявления и описания членов классов находятся в файлах:

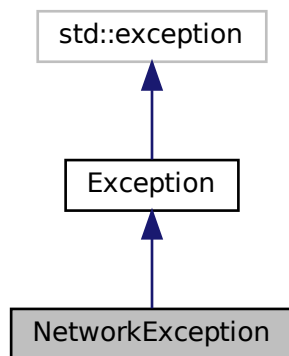
- Desktop/sasha/[network.h](#)
- Desktop/sasha/[network.cpp](#)

## 4.8 Класс NetworkException

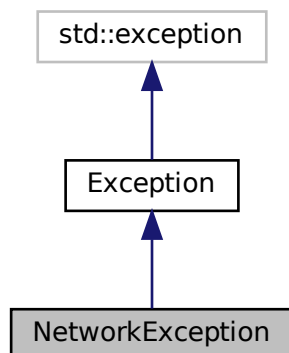
Класс для сетевых исключений.

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

Граф наследования:NetworkException:



Граф связей класса NetworkException:



## Открытые члены

- [NetworkException](#) (const string &message, const string &func, const string &log\_path, bool critical)

Конструктор класса [NetworkException](#).

## Дополнительные унаследованные члены

### 4.8.1 Подробное описание

Класс для сетевых исключений.

### 4.8.2 Конструктор(ы)

#### 4.8.2.1 NetworkException()

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

Конструктор класса [NetworkException](#).

#### Аргументы

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

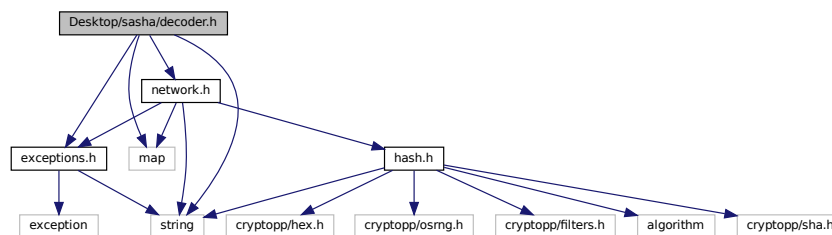
Объявления и описания членов классов находятся в файлах:

- Desktop/sasha/[exceptions.h](#)
- Desktop/sasha/[exceptions.cpp](#)

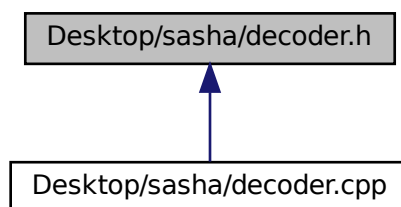


```
#include "network.h"
```

Граф включаемых заголовочных файлов для decoder.h:



Граф файлов, в которые включается этот файл:



## Классы

- class [Decoder](#)

Класс для управления интерфейсом программы.

### 5.2.1 Подробное описание

Определение класса интерфейса.

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

Дата

01.12.2025

Версия

1.0

Автор

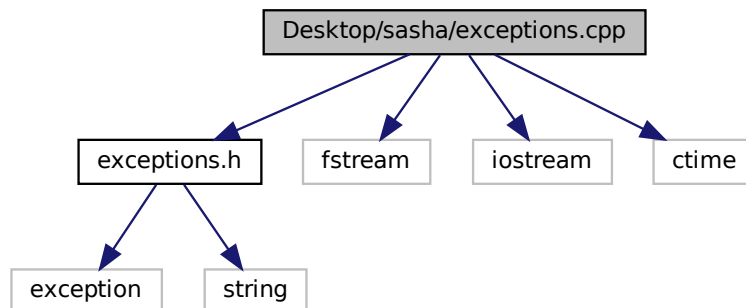
Лынев А. С.



### 5.3 Файл Desktop/sasha/exceptions.cpp

```
#include "exceptions.h"  
#include <fstream>  
#include <iostream>  
#include <ctime>
```

Граф включаемых заголовочных файлов для exceptions.cpp:

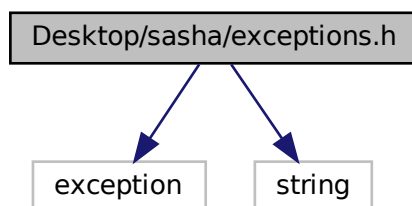


### 5.4 Файл Desktop/sasha/exceptions.h

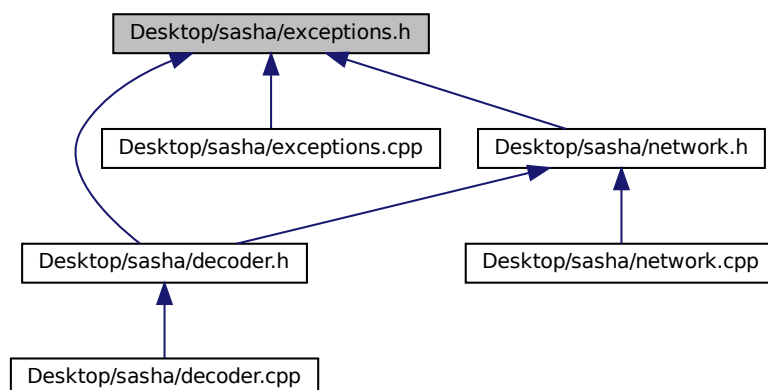
Определение классов исключений.

```
#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.

### 5.4.1 Подробное описание

Определение классов исключений.

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

Дата

01.12.2025

Версия

1.0

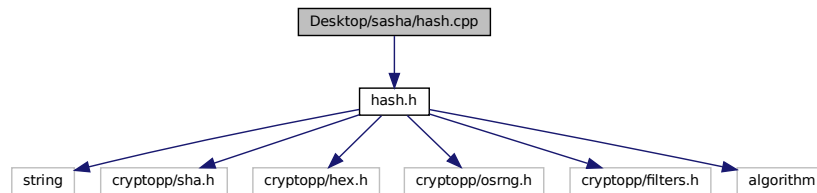
Автор

Лынев А. С.

## 5.5 Файл Desktop/sasha/hash.cpp

```
#include "hash.h"
```

Граф включаемых заголовочных файлов для hash.cpp:



### Функции

- string `getSalt` ()  
Функция для генерации соли.
- string `getHash` (const string &data)  
Функция для вычисления хеша.

#### 5.5.1 Функции

##### 5.5.1.1 getHash()

```
string getHash (
    const string & data )
```

Функция для вычисления хеша.

Аргументы

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

Возвращает

Хеш в виде строки.

##### 5.5.1.2 getSalt()

```
string getSalt ( )
```

Функция для генерации соли.

Возвращает

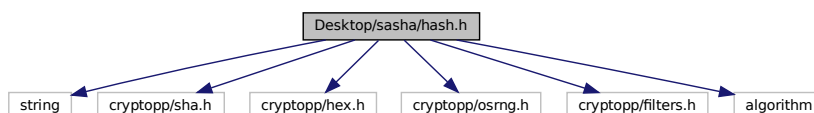
Соль в виде строки.

## 5.6 Файл Desktop/sasha/hash.h

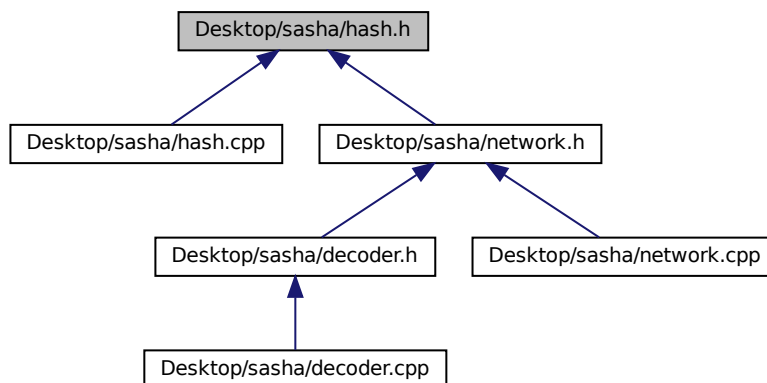
Определения вспомогательных функций для криптографических операций.

```
#include <string>
#include <cryptopp/sha.h>
#include <cryptopp/hex.h>
#include <cryptopp/osrng.h>
#include <cryptopp/filters.h>
#include <algorithm>
```

Граф включаемых заголовочных файлов для hash.h:



Граф файлов, в которые включается этот файл:



### Функции

- string `getSalt` ()  
Функция для генерации соли.
- string `getHash` (const string &data)  
Функция для вычисления хеша.

### 5.6.1 Подробное описание

Определения вспомогательных функций для криптографических операций.

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

Дата

01.12.2025

Версия

1.0

Автор

Лынев А. С.

### 5.6.2 Функции

#### 5.6.2.1 getHash()

```
string getHash (  
    const string & data )
```

Функция для вычисления хеша.

Аргументы

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

Возвращает

Хеш в виде строки.

#### 5.6.2.2 getSalt()

```
string getSalt ( )
```

Функция для генерации соли.

Возвращает

Соль в виде строки.

## 5.7 Файл Desktop/sasha/main.cpp

```
#include "../Kurovaya/source/headers/network.h"
#include "../Kurovaya/source/headers/decoder.h"
#include "../Kurovaya/source/headers/exceptions.h"
#include <iostream>
#include <fstream>
#include <map>
#include <string>
```

Граф включаемых заголовочных файлов для main.cpp:



### Функции

- `map< string, string > getDataBase (string &path)`  
Читает файл базы данных и заполняет карту парами "имя пользователя - пароль".
- `void loop (Network *network)`  
Основной цикл для обработки сетевых операций. Эта функция запускает сетевой сервис и входит в бесконечный цикл, в котором она ожидает входящих соединений, аутентифицирует пользователей и выполняет операции. Любые сетевые исключения обрабатываются и регистрируются. Если возникает неожиданное исключение, сетевой сервис останавливается.
- `int main (int argc, char *argv[])`  
Главная точка входа в приложение.

### Переменные

- `string LOG_PATH`

#### 5.7.1 Функции

##### 5.7.1.1 getDataBase()

```
map<string, string> getDataBase (
    string & path )
```

Читает файл базы данных и заполняет карту парами "имя пользователя - пароль".

#### Аргументы

path	Ссылка на строку, представляющую путь к файлу базы данных.
------	--

Возвращает

Карта, содержащая пары "имя пользователя - пароль"

Исключения

<a href="#">IOException</a>	Если файл не может быть открыт.
<a href="#">DataDecodeException</a>	Если строка в файле не соответствует ожидаемому формату или если произошла ошибка при чтении из файла.

#### 5.7.1.2 loop()

```
void loop (
    Network * network )
```

Основной цикл для обработки сетевых операций. Эта функция запускает сетевой сервис и входит в бесконечный цикл, в котором она ожидает входящих соединений, аутентифицирует пользователей и выполняет операции. Любые сетевые исключения обрабатываются и регистрируются. Если возникает неожиданное исключение, сетевой сервис останавливается.

Аргументы

network	Указатель на объект <a href="#">Network</a> , который обрабатывает сетевые операции.
---------	--

#### 5.7.1.3 main()

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

Главная точка входа в приложение.

Эта функция инициализирует объект [Decoder](#) для парсинга аргументов командной строки, получает путь к журналу и базу данных из указанной конфигурации, инициализирует объект [Network](#) с этими настройками и входит в цикл для обработки сетевых операций. Также включает обработку ошибок для сетевых исключений.

Аргументы

argc	Количество аргументов командной строки.
argv	Массив строк аргументов командной строки.

Возвращает

Целое число, представляющее статус завершения программы (0 для успешного завершения).

## 5.7.2 Переменные

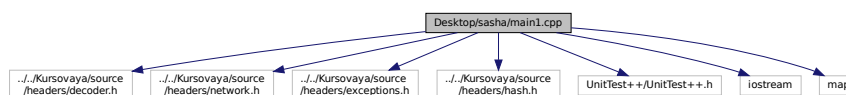
### 5.7.2.1 LOG\_PATH

```
string LOG_PATH
```

## 5.8 Файл Desktop/sasha/main1.cpp

```
#include "../Kurovaya/source/headers/decoder.h"
#include "../Kurovaya/source/headers/network.h"
#include "../Kurovaya/source/headers/exceptions.h"
#include "../Kurovaya/source/headers/hash.h"
#include <UnitTest++/UnitTest++.h>
#include <iostream>
#include <map>
```

Граф включаемых заголовочных файлов для main1.cpp:



## Функции

- **TEST** (DecoderConstructorAndGetters)
- **TEST** (DecoderParseArguments)
- **TEST** (DecoderShowHelp)
- **TEST** (DecoderInvalidArguments)
- **TEST** (DecoderInvalidPort)
- **TEST** (GetSalt)
- **TEST** (GetHash)
- **TEST** (SameInputSameHash)
- **TEST** (NetworkConstructorAndGetters)
- **int** **main** ()

### 5.8.1 Функции

#### 5.8.1.1 main()

```
int main ( )
```



## 5.8.1.2 TEST() [1/9]

```
TEST (
    DecoderConstructorAndGetters )
```

## 5.8.1.3 TEST() [2/9]

```
TEST (
    DecoderInvalidArguments )
```

## 5.8.1.4 TEST() [3/9]

```
TEST (
    DecoderInvalidPort )
```

## 5.8.1.5 TEST() [4/9]

```
TEST (
    DecoderParseArguments )
```

## 5.8.1.6 TEST() [5/9]

```
TEST (
    DecoderShowHelp )
```

## 5.8.1.7 TEST() [6/9]

```
TEST (
    GetHash )
```

## 5.8.1.8 TEST() [7/9]

```
TEST (
    GetSalt )
```

## 5.8.1.9 TEST() [8/9]

TEST (

NetworkConstructorAndGetters )

## 5.8.1.10 TEST() [9/9]

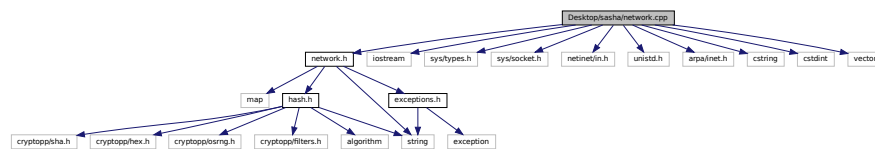
TEST (

SameInputSameHash )

## 5.9 Файл Desktop/sasha/network.cpp

```
#include "network.h"
#include <iostream>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <cstring>
#include <cstdint>
#include <vector>
```

Граф включаемых заголовочных файлов для network.cpp:

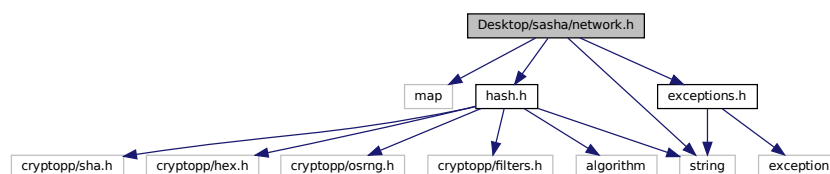


## 5.10 Файл Desktop/sasha/network.h

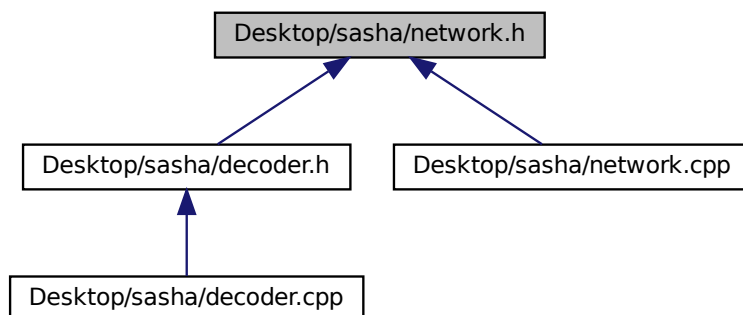
Определения классов для управления сетевым взаимодействием.

```
#include <map>
#include <string>
#include "hash.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.

### 5.10.1 Подробное описание

Определения классов для управления сетевым взаимодействием.

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

Дата

01.12.2025

Версия

1.0

Автор

Лынев А. С.



# Предметный указатель

- ArgsDecodeException, 7
  - ArgsDecodeException, 8
- auth
  - Network, 21
- AuthException, 9
  - AuthException, 10
- critical
  - Exception, 17
- DataDecodeException, 10
  - DataDecodeException, 12
- Decoder, 12
  - Decoder, 13
  - getAddress, 13
  - getDataBasePath, 13
  - getLogPath, 13
  - getPort, 14
  - parse, 14
  - showHelp, 14
- Desktop/sasha/decoder.cpp, 25
- Desktop/sasha/decoder.h, 25
- Desktop/sasha/exceptions.cpp, 27
- Desktop/sasha/exceptions.h, 27
- Desktop/sasha/hash.cpp, 29
- Desktop/sasha/hash.h, 30
- Desktop/sasha/main.cpp, 32
- Desktop/sasha/main1.cpp, 34
- Desktop/sasha/network.cpp, 36
- Desktop/sasha/network.h, 36
- Exception, 15
  - critical, 17
  - Exception, 16
  - func, 17
  - log\_path, 17
  - logException, 17
  - message, 17
  - name, 18
  - what, 17
- func
  - Exception, 17
- getAddress
  - Decoder, 13
- getAddress
  - Network, 21
- getDataBase
  - main.cpp, 32
- getDatabase
  - Network, 21
- getDataBasePath
  - Decoder, 13
- getHash
  - hash.cpp, 29
  - hash.h, 31
- getLogPath
  - Decoder, 13
- getPort
  - Decoder, 14
  - Network, 21
- getSalt
  - hash.cpp, 29
  - hash.h, 31
- hash.cpp
  - getHash, 29
  - getSalt, 29
- hash.h
  - getHash, 31
  - getSalt, 31
- IOException, 18
  - IOException, 19
- LOG\_PATH
  - main.cpp, 34
- log\_path
  - Exception, 17
- logException
  - Exception, 17
- loop
  - main.cpp, 33
- main
  - main.cpp, 33
  - main1.cpp, 34
- main.cpp
  - getDatabase, 32
  - LOG\_PATH, 34
  - loop, 33
  - main, 33
- main1.cpp
  - main, 34
  - TEST, 34–36
- message
  - Exception, 17
- name

- Exception, [18](#)
- Network, [20](#)
  - auth, [21](#)
  - getAddress, [21](#)
  - getDatabase, [21](#)
  - getPort, [21](#)
  - Network, [20](#)
  - start, [22](#)
  - stop, [22](#)
  - sum, [22](#)
  - wait, [22](#)
- NetworkException, [23](#)
  - NetworkException, [24](#)
- parse
  - Decoder, [14](#)
- showHelp
  - Decoder, [14](#)
- start
  - Network, [22](#)
- stop
  - Network, [22](#)
- sum
  - Network, [22](#)
- TEST
  - main1.cpp, [34–36](#)
- wait
  - Network, [22](#)
- what
  - Exception, [17](#)