My Project

Создано системой Doxygen 1.9.4

Глава 1

Алфавитный указатель классов

1.1 Классы

Классы с их кратким описанием.

Глава 2

Список файлов

2.1 Файлы

Полный список документированных файлов.

Глава 3

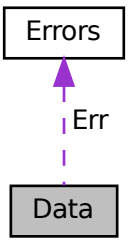
Классы

3.1 Класс Data

Класс Data для управления данными, связанными с файлами и клиентами.

#include <Data.h>

Граф связей класса Data:



Открытые члены

- string get_FileReader ()
  Возвращает имя файла, которое используется для чтения данных.
- void set_FileReader (string file)
- pair< vector< string >, vector< string > > getClient ()

Закрытые данные

- string FileReader
  Строка для хранения имени файла, которое используется для чтения данных.
- Errors Err
  Объект класса Errors для обработки ошибок.

3.1.1 Подробное описание

Класс Data для управления данными, связанными с файлами и клиентами.

Этот класс предназначен для хранения и обработки информации, связанной с чтением данных из файла, а также для получения информации о клиентах, хранимой в двух векторах.

3.1.2 Методы

3.1.2.1 get_FileReader()

string Data::get_FileReader ( )

Возвращает имя файла, которое используется для чтения данных.

Возвращает

std::string Имя файла, установленное с помощью set_FileReader.

Объявления и описания членов классов находятся в файлах:

- Data.h
- Data.cpp

3.2 Класс Errors

Класс для обработки и записи ошибок.

#include <Errors.h>

Открытые члены

- string get_File_Log ()
  Возвращает имя файла лога ошибок.
- void set_File_Log (string file)
  Устанавливает имя файла лога ошибок.
- void error_recording (string flag, string info)
  Записывает информацию об ошибке в лог-файл.

Закрытые данные

- string File_Log
  Имя файла лога ошибок.

3.2.1 Подробное описание

Класс для обработки и записи ошибок.

Класс Errors предоставляет функциональность для записи информации об ошибках в лог-файл, а также для управления именем файла лога.

3.2.2 Методы

3.2.2.1 error_recording()

```
void Errors::error_recording (
                string flag,
                string info )
```

Записывает информацию об ошибке в лог-файл.

Аргументы

flag	Тип ошибки
info	Описание ошибки

3.2.2.2 get_File_Log()

```
string Errors::get_File_Log ( )
```

Возвращает имя файла лога ошибок.

Возвращает

Строка, содержащая имя файла лога ошибок.

3.2.2.3 set_File_Log()

```
void Errors::set_File_Log (
                string file )
```

Устанавливает имя файла лога ошибок.

Аргументы

file	Имя файла лога ошибок.

Объявления и описания членов классов находятся в файлах:

- Errors.h
- Errors.cpp

3.3 Класс interface

Класс interface для обработки аргументов командной строки и хранения конфигурационных данных.

#include <Interface.h>

Открытые члены

- interface (int argc, char **argv)
  Конструктор, обрабатывающий аргументы командной строки.
- interface ()=delete
  Запрещает создание объектов класса без аргументов.
- string get_ip ()
  Возвращает IP-адрес.
- int get_port ()
  Возвращает номер порта.
- int help ()
  Возвращает справку и завершает работу.

Открытые атрибуты

- string ip
  IP-адрес
- int port
  Порт
- string database
  Название файла с базой данных
- string logfile
  Название файла, куда записываются ошибки

3.3.1 Подробное описание

Класс interface для обработки аргументов командной строки и хранения конфигурационных данных.

Этот класс предназначен для обработки аргументов командной строки, полученных при запуске программы, и хранения конфигурационных данных, таких как IP-адрес, номер порта, имя файла базы данных и имя файла журнала ошибок.

3.3.2 Конструктор(ы)

3.3.2.1 interface()

```
interface::interface (
            int argc,
            char ** argv )
```

Конструктор, обрабатывающий аргументы командной строки.

Аргументы

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

3.3.3 Методы

3.3.3.1 get_ip()

string interface::get_ip ( ) [inline]

Возвращает IP-адрес.

Возвращает

IP-адрес.

3.3.3.2 get_port()

int interface::get_port ( ) [inline]

Возвращает номер порта.

Возвращает

int Номер порта.

3.3.3.3 help()

int interface::help ( )

Возвращает справку и завершает работу.

Возвращает

int Возвращает справку.

Объявления и описания членов классов находятся в файлах:

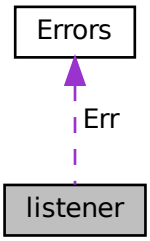
- Interface.h
- Interface.cpp

3.4 Класс listener

Класс listener, реализующий серверную часть приложения. Этот класс отвечает за установление соединения с клиентами, обработку их запросов и отправку ответов.

#include <listener.h>

Граф связей класса listener:



Открытые члены

- int interaction (string database, string logFile)

  Устанавливает взаимодействие с клиентами, обрабатывает их запросы и отправляет ответы.
- uint64_t sredn ()

  Вычисляет среднее арифметическое значений в векторе vec.
- string get_address ()
- void set_address (string address1)
- int get_port ()

  Устанавливает номер порта сервера.
- void set_port (int port1)
- vector< uint64_t > get_vec ()
- void set_vec (vector< uint64_t > v)
- string get_salt ()
- void set_salt ()
- pair< vector< string >, vector< string > > get_DB_clients ()
- void set_DB_clients (vector< string > login, vector< string > password)

Закрытые данные

- string address
- int port
- vector< uint64_t > vec
- string salt
- pair< vector< string >, vector< string > > DB_clients
- Errors Err

3.4.1 Подробное описание

Класс listener, реализующий серверную часть приложения. Этот класс отвечает за установление соединения с клиентами, обработку их запросов и отправку ответов.

3.4.2 Методы

3.4.2.1 get_port()

```
int listener::get_port ( )
```

Устанавливает номер порта сервера.

Аргументы

port1	Номер порта, который нужно установить.

3.4.2.2 interaction()

```
int listener::interaction (
                string database,
                string logFile )
```

Устанавливает взаимодействие с клиентами, обрабатывает их запросы и отправляет ответы.

Аргументы

database	Путь к файлу базы данных.
logFile	Путь к файлу логов.

Возвращает

0, если взаимодействие успешно установлено; иначе -1.

!!!!!!!!!

3.4.2.3 sredn()

```
uint64_t listener::sredn ( )
```

Вычисляет среднее арифметическое значений в векторе vec.

Возвращает

Среднее арифметическое значений в векторе.

3.4.3 Данные класса

3.4.3.1 address

string listener::address [private]

IP-адрес сервера.

3.4.3.2 DB_clients

pair<vector<string>, vector<string> > listener::DB_clients [private]

Пара векторов: логины и пароли пользователей.

3.4.3.3 Err

Errors listener::Err [private]

Объект для обработки ошибок.

3.4.3.4 port

int listener::port [private]

Номер порта сервера.

3.4.3.5 salt

string listener::salt [private]

Сгенерированная соль.

3.4.3.6 vec

vector<uint64_t> listener::vec [private]

Вектор значений.

Объявления и описания членов классов находятся в файлах:

- listener.h
- listener.cpp

3.5 Класс User

Класс, представляющий пользователя системы.

#include <User.h>

Открытые члены

- bool CheckLogin (vector< string > Db_ID)

  Проверяет, существует ли ID пользователя в базе данных.
- bool CheckPassword (vector< string > Db_hash, vector< string > Db_ID, string SALT)

  Проверяет пароль пользователя.
- string get_ID ()

  Возвращает ID пользователя.
- void set_ID (string ID1)

  Устанавливает ID пользователя.
- string get_hash ()

  Возвращает хэш пароля пользователя.
- void set_hash (string hash1)

  Устанавливает хэш пароля пользователя.

Закрытые данные

- string ID

  Идентификатор пользователя.
- string hash

  Хэш пароля пользователя.

3.5.1 Подробное описание

Класс, представляющий пользователя системы.

Класс User хранит информацию о пользователе (ID и хэш пароля) и предоставляет методы для проверки подлинности пользователя. Обратите внимание, что хранение хэшей паролей напрямую в коде - небезопасная практика. В реальных приложениях необходимо использовать более безопасные методы, такие как bcrypt или Argon2, и хранить хэши в защищенной базе данных.

3.5.2 Методы

3.5.2.1 CheckLogin()

```
bool User::CheckLogin (
            vector< string > Db_ID )
```

Проверяет, существует ли ID пользователя в базе данных.

Функция проверяет наличие переданного ID пользователя в векторе ID из базы данных.

Аргументы

Db_ID	Вектор строк, содержащий ID пользователей из базы данных.

Возвращает

true, если ID пользователя найден в базе данных, false в противном случае.

3.5.2.2 CheckPassword()

```
bool User::CheckPassword (
            vector< string > Db_password,
            vector< string > Db_ID,
            string SALT )
```

Проверяет пароль пользователя.

Функция проверяет корректность пароля пользователя, сравнивая его хэш с хэшем из базы данных. Сначала по ID пользователя находится соответствующий пароль в базе данных. Затем вычисляется хэш переданного пароля с добавлением соли и сравнивается с хэшем из базы данных.

Аргументы

Db_password	Вектор строк, содержащий пароли пользователей из базы данных.
Db_ID	Вектор строк, содержащий ID пользователей из базы данных.
SALT	Строка, содержащая соль для хеширования.

Возвращает

true, если пароль верный, false в противном случае.

3.5.2.3 get_hash()

```
string User::get_hash ( )
```

Возвращает хэш пароля пользователя.

Возвращает

Строка, содержащая хэш пароля пользователя.

3.5.2.4 get_ID()

```
string User::get_ID ( )
```

Возвращает ID пользователя.

Возвращает

Строка, содержащая ID пользователя.

3.5.2.5 set_hash()

```
void User::set_hash (
            string hash1 )
```

Устанавливает хэш пароля пользователя.

Аргументы

hash1	Новый хэш пароля пользователя.

3.5.2.6 set_ID()

```
void User::set_ID (
            string ID1 )
```

Устанавливает ID пользователя.

Аргументы

ID1	Новый ID пользователя.

Объявления и описания членов классов находятся в файлах:

- User.h
- User.cpp

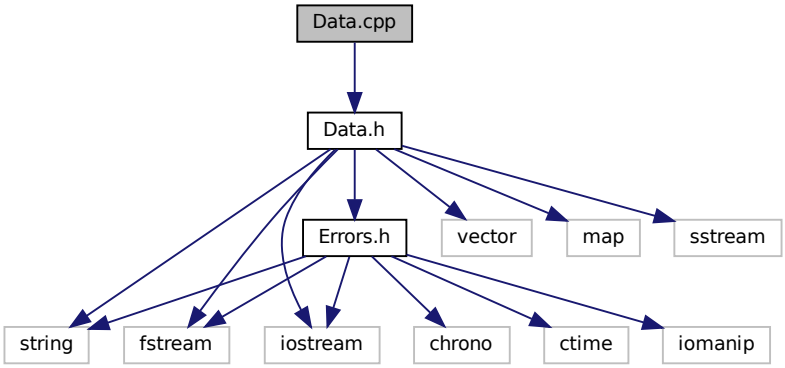
Глава 4

Файлы

4.1 Файл Data.cpp

Класс Data для считывания данных клиентов из файла.

#include "Data.h"
Граф включаемых заголовочных файлов для Data.cpp:



4.1.1 Подробное описание

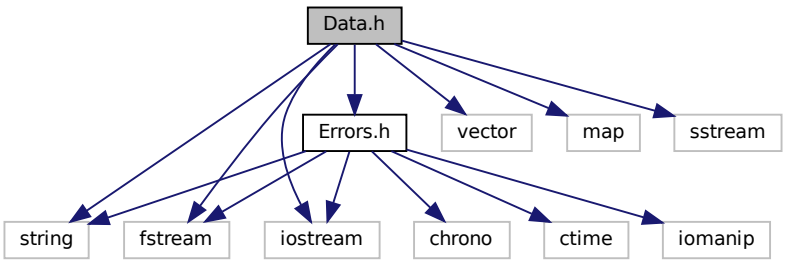
Класс Data для считывания данных клиентов из файла.

4.2 Файл Data.h

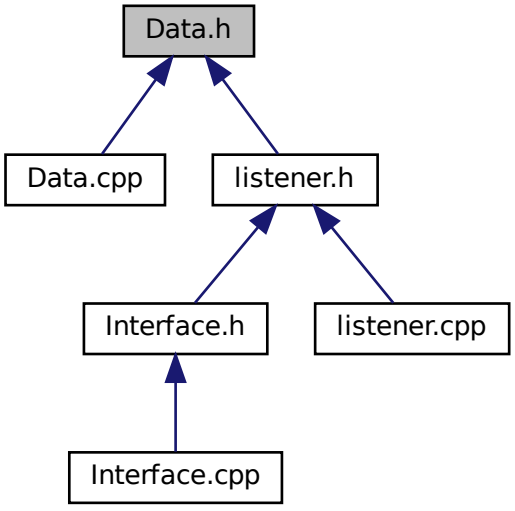
Класс Data для управления данными, связанными с файлами и клиентами.

#include <string>
#include <vector>

#include <map>
#include <fstream>
#include <sstream>
#include <iostream>
#include "Errors.h"
Граф включаемых заголовочных файлов для Data.h:



Граф файлов, в которые включается этот файл:



Классы

- class Data

  Класс Data для управления данными, связанными с файлами и клиентами.

4.2.1 Подробное описание

Класс Data для управления данными, связанными с файлами и клиентами.

4.3 Data.h

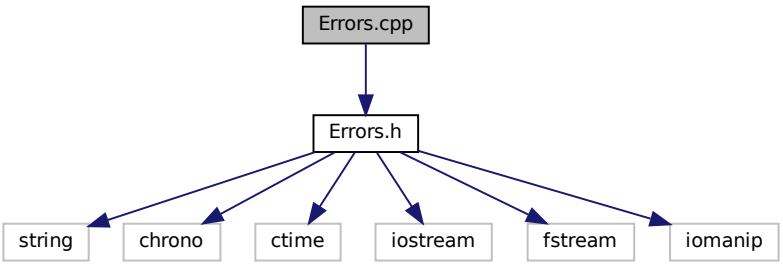
См. документацию.

```
1
5 #include <string>
6 #include <vector>
7 #include <map>
8 #include <fstream>
9 #include <sstream>
10 #include <iostream>
11
12 #include "Errors.h"
13
14 using namespace std;
15
22 class Data{
23     public:
29         string get_FileReader();
30         void set_FileReader(string file);
31
32         pair<vector<string>, vector<string» getClient();
33     private:
37         string FileReader;
41         Errors Err;
42 };
```

4.4 Файл Errors.cpp

Класс Errors для обработки ошибок и их записи в файл.

#include "Errors.h"
Граф включаемых заголовочных файлов для Errors.cpp:



4.4.1 Подробное описание

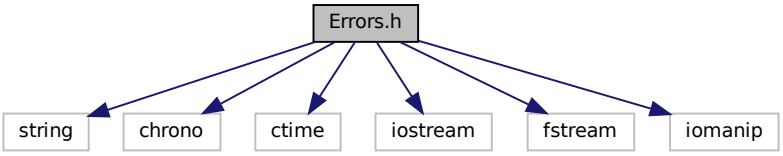
Класс Errors для обработки ошибок и их записи в файл.

4.5 Файл Errors.h

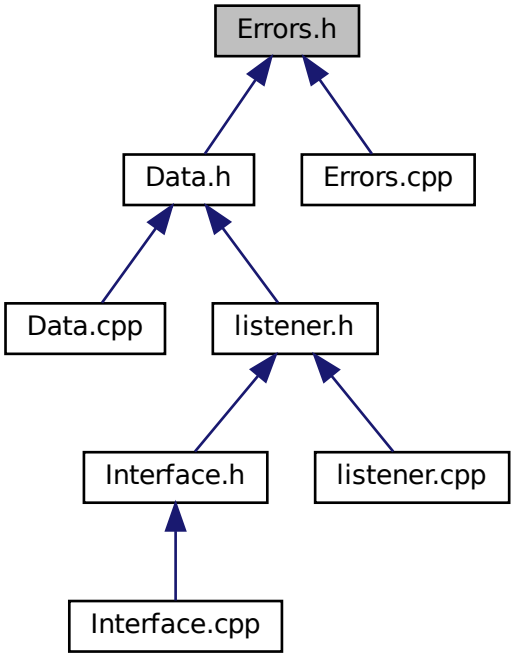
Класс Errors для обработки и записи ошибок.

#include <string>
#include <chrono>

#include <ctime>
#include <iostream>
#include <fstream>
#include <iomanip>
Граф включаемых заголовочных файлов для Errors.h:



Граф файлов, в которые включается этот файл:



Классы

- class Errors

  Класс для обработки и записи ошибок.

4.5.1 Подробное описание

Класс Errors для обработки и записи ошибок.

4.6 Errors.h

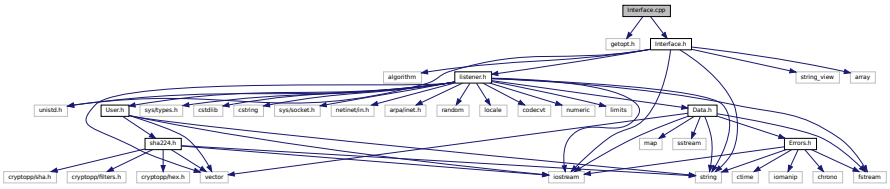
См. документацию.

```
1
5 #include <string>
6 #include <chrono>
7 #include <ctime>
8 #include <iostream>
9 #include <fstream>
10 #include <iomanip>
11
12 using namespace std;
13
20 class Errors {
21 public:
27     string get_File_Log();
28
34     void set_File_Log(string file);
35
42     void error_recording(string flag, string info);
43
44 private:
46     string File_Log;
47 };
48
```

4.7 Файл Interface.cpp

Класс Interface для обрабатки аргументов командной строки.

#include <getopt.h>
#include "Interface.h"
Граф включаемых заголовочных файлов для Interface.cpp:



4.7.1 Подробное описание

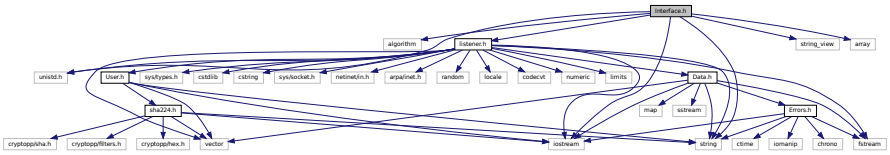
Класс Interface для обрабатки аргументов командной строки.

4.8 Файл Interface.h

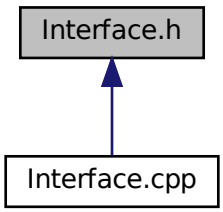
Класс Interface для обработки аргументов командной строки и хранения конфигурационных данных.

#include <algorithm>
#include <iostream>
#include <string>
#include <string_view>
#include <array>
#include <unistd.h>

#include "listener.h"
Граф включаемых заголовочных файлов для Interface.h:



Граф файлов, в которые включается этот файл:



Классы

- class interface

  Класс interface для обработки аргументов командной строки и хранения конфигурационных данных.

4.8.1 Подробное описание

Класс Interface для обработки аргументов командной строки и хранения конфигурационных данных.

4.9 Interface.h

См. документацию.

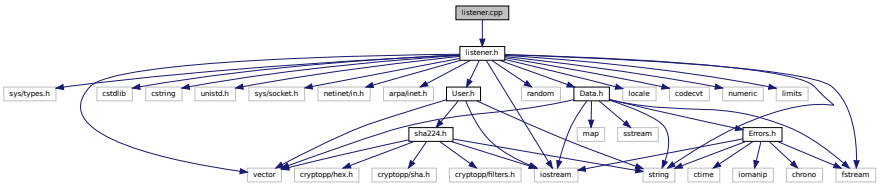
```
1
5 #pragma once
6 #include <algorithm>
7 #include <iostream>
8 #include <string>
9 #include <string_view>
10 #include <array>
11 #include <unistd.h>
12
13 #include "listener.h"
14
15 using namespace std;
16
24 class interface {
25 public:
```

```
27     string ip;
29     int port;
31     string database;
33     string logfile;
34
41     interface(int argc, char** argv);
42
46     interface() = delete;
47
53     string get_ip() {
54         return ip;
55     }
56
62     int get_port() {
63         return port;
64     }
69     int help();
70 };
71 //-----------------------------------------------------------------------------
```

4.10 Файл listener.cpp

Класс listener для вычисления среднего арифметического значений в векторе.

#include "listener.h"
Граф включаемых заголовочных файлов для listener.cpp:



4.10.1 Подробное описание

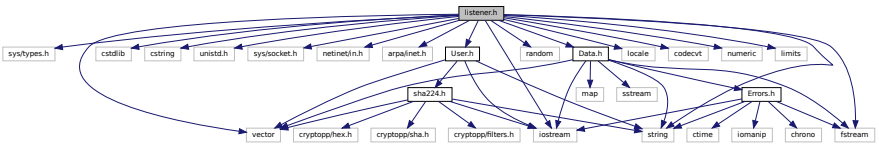
Класс listener для вычисления среднего арифметического значений в векторе.

4.11 Файл listener.h

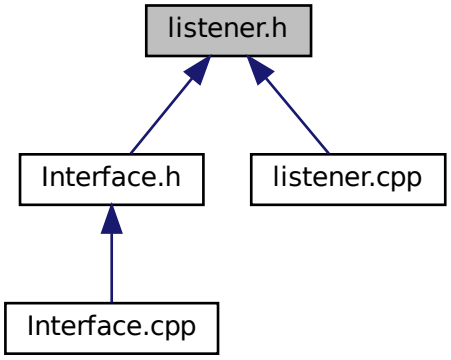
Класс listener для вычисления среднего арифметического значений в векторе.

#include <sys/types.h>
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fstream>
#include <vector>
#include <random>
#include <string>
#include <locale>

#include <codecvt>
#include <numeric>
#include <limits>
#include "User.h"
#include "Data.h"
Граф включаемых заголовочных файлов для listener.h:



Граф файлов, в которые включается этот файл:



Классы

- class listener

  Класс listener, реализующий серверную часть приложения. Этот класс отвечает за установление соединения с клиентами, обработку их запросов и отправку ответов.

4.11.1 Подробное описание

Класс listener для вычисления среднего арифметического значений в векторе.

4.12 listener.h

См. документацию.

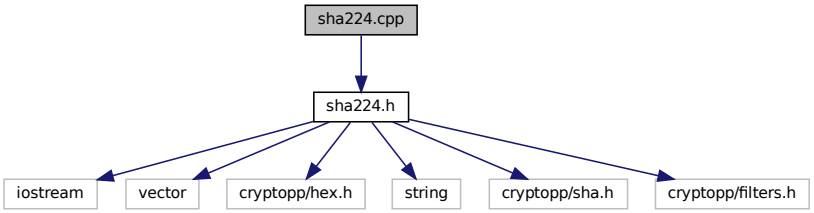
```
1
5 #pragma one
6 #include <sys/types.h>
```

```
7 #include <iostream>
8 #include <cstdlib>
9 #include <cstring>
10 #include <unistd.h>
11 #include <sys/socket.h>
12 #include <netinet/in.h>
13 #include <arpa/inet.h>
14 #include <fstream>
15 #include <vector>
16 #include <random>
17 #include <string>
18 #include <locale>
19 #include <codecvt>
20 #include <numeric>
21 #include <limits>
22
23 #include "User.h"
24 #include "Data.h"
25
26 using namespace std;
27
32 class listener{
33    public:
40       int interaction(string database, string logFile);
41
46       uint64_t sredn();
47
48       string get_address();
49       void set_address(string address1);
54       int get_port();
55       void set_port(int port1);
56
57       vector<uint64_t> get_vec();
58       void set_vec(vector<uint64_t> v);
59
60       string get_salt();
61       void set_salt();
62
63       pair<vector<string>, vector<string» get_DB_clients();
64       void set_DB_clients(vector<string> login, vector<string> password);
65
66    private:
67       string address;
68       int port;
69       vector<uint64_t> vec;
70       string salt;
71       pair<vector<string>, vector<string»DB_clients;
72       Errors Err;
73 };
```

4.13 Файл sha224.cpp

Вычисляет SHA224 хэш для заданной строки.

#include "sha224.h"
Граф включаемых заголовочных файлов для sha224.cpp:



Функции

- std::string SHA224_hash (std::string msg)

4.13.1 Подробное описание

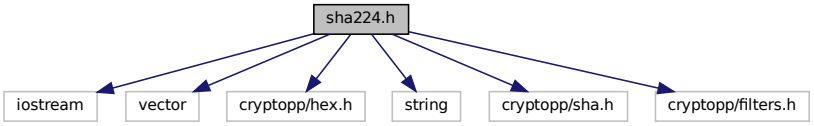
Вычисляет SHA224 хэш для заданной строки.

4.14 Файл sha224.h

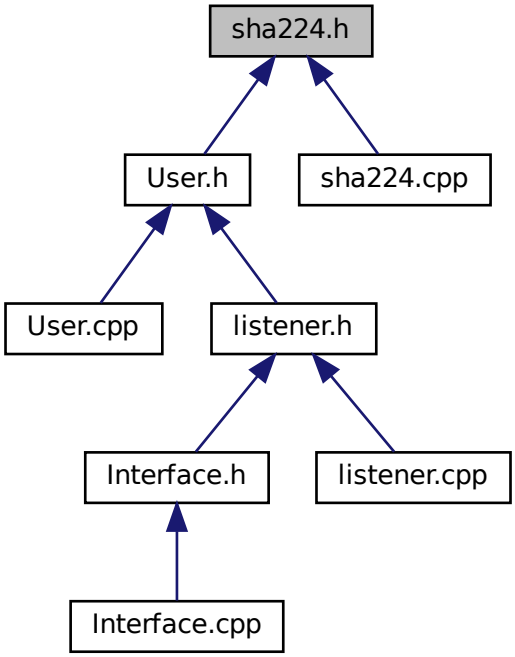
Вычисляет SHA224 хэш для заданной строки.

```
#include <iostream>
#include <vector>
#include <cryptopp/hex.h>
#include <string>
#include <cryptopp/sha.h>
#include <cryptopp/filters.h>
```

Граф включаемых заголовочных файлов для sha224.h:



Граф файлов, в которые включается этот файл:



Макросы

- #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1

Функции

- std::string SHA224_hash (std::string msg)

4.14.1 Подробное описание

Вычисляет SHA224 хэш для заданной строки.

4.15 sha224.h

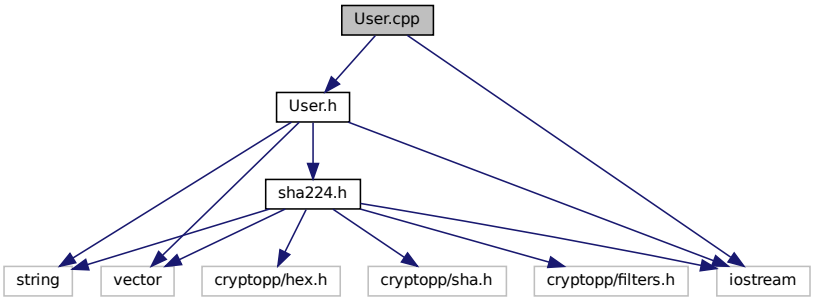
См. документацию.

```
1
5 #include <iostream>
6 #include <vector>
7 #include <cryptopp/hex.h>
8 #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1
9 #include <string>
10 #include <cryptopp/sha.h>
11 #include <cryptopp/hex.h>
12 #include <cryptopp/filters.h>
13
14 std::string SHA224_hash(std::string msg);
```

4.16 Файл User.cpp

Класс User, представляющий пользователя системы.

#include "User.h"
#include <iostream>
Граф включаемых заголовочных файлов для User.cpp:



Функции

- void foo ()

  Пустая глобальная функция, для Doxygen.

4.16.1 Подробное описание

Класс User, представляющий пользователя системы.

4.16.2 Функции

4.16.2.1 foo()

void foo ( )

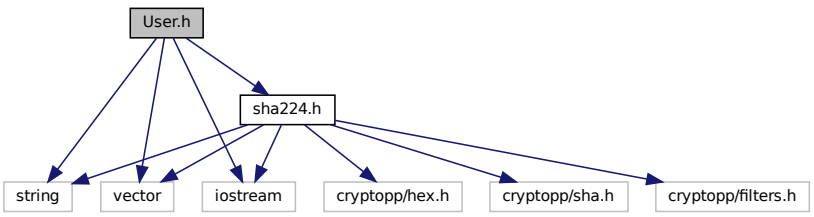
Пустая глобальная функция, для Doxygen.

•

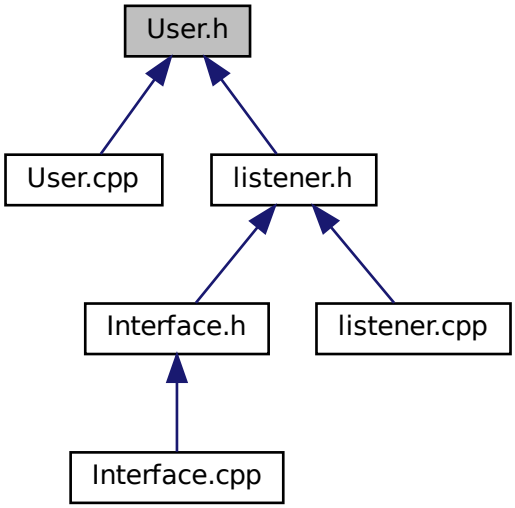
4.17 Файл User.h

Класс User, представляющий пользователя системы.

#include <string>
#include <vector>
#include <iostream>
#include "sha224.h"
Граф включаемых заголовочных файлов для User.h:



Граф файлов, в которые включается этот файл:



Классы

- class User

  Класс, представляющий пользователя системы.

4.17.1 Подробное описание

Класс User, представляющий пользователя системы.

4.18 User.h

См. документацию.

```
1
5 #include <string>
6 #include <vector>
7 #include <iostream>
8
9 #include "sha224.h"
10
11 using namespace std;
12
21 class User {
22 private:
24     string ID;
26     string hash;
27
28 public:
29     bool CheckLogin(vector<string> Db_ID);
30     bool CheckPassword(vector<string> Db_hash, vector<string> Db_ID, string SALT);
31
32     string get_ID();
33     void set_ID(string ID1);
34
35     string get_hash();
36     void set_hash(string hash1);
37 };
38
```

Предметный указатель