VServer

Создано системой Doxygen 1.9.4

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

Глава 4

Классы

4.1 Класс AcceptConnectionError

Ошибка принятия соединения.

#include <errors.h>

Граф наследования:AcceptConnectionError:

4.2 Класс BindSocketError

Ошибка привязки сокета.

#include <errors.h>

Граф наследования:BindSocketError:

Граф связей класса BindSocketError:

Открытые члены

- BindSocketError (const string &func, bool is_critical=false)
  Конструктор BindSocketError.
- BindSocketError (const string &func, bool is_critical=false)
  Конструктор BindSocketError.

Дополнительные унаследованные члены

4.2.1 Подробное описание

Ошибка привязки сокета.

4.2.2 Конструктор(ы)

4.2.2.1 BindSocketError() [1/2]

```
BindSocketError::BindSocketError (
                const string & func,
                bool is_critical = false )
```

Конструктор BindSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.2.2.2 BindSocketError() [2/2]

```
BindSocketError::BindSocketError (
            const string & func,
            bool is_critical = false )
```

Конструктор BindSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.3 Класс CreateSocketError

Ошибка создания сокета.

#include <errors.h>

Граф наследования:CreateSocketError:

Граф связей класса CreateSocketError:

Открытые члены

- CreateSocketError (const string &func, bool is_critical=false)
  Конструктор CreateSocketError.
- CreateSocketError (const string &func, bool is_critical=false)
  Конструктор CreateSocketError.

Дополнительные унаследованные члены

4.3.1 Подробное описание

Ошибка создания сокета.

4.3.2 Конструктор(ы)

4.3.2.1 CreateSocketError() [1/2]

```
CreateSocketError::CreateSocketError (
            const string & func,
            bool is_critical = false )
```

Конструктор CreateSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.3.2.2 CreateSocketError() [2/2]

```
CreateSocketError::CreateSocketError (
            const string & func,
            bool is_critical = false )
```

Конструктор CreateSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.4 Класс InvalidArgumentError

Ошибка некорректного аргумента.

#include <errors.h>

Граф наследования:InvalidArgumentError:

Граф связей класса InvalidArgumentError:

Открытые члены

- InvalidArgumentError (const string &func, bool is_critical=false)
  Конструктор InvalidArgumentError.
- InvalidArgumentError (const string &func, bool is_critical=false)
  Конструктор InvalidArgumentError.

Дополнительные унаследованные члены

4.4.1 Подробное описание

Ошибка некорректного аргумента.

4.4.2 Конструктор(ы)

4.4.2.1 InvalidArgumentError() [1/2]

```
InvalidArgumentError::InvalidArgumentError (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidArgumentError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.4.2.2 InvalidArgumentError() [2/2]

```
InvalidArgumentError::InvalidArgumentError (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidArgumentError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.5 Класс InvalidAuthDataFormatError

Ошибка некорректного формата данных аутентификации.

#include <errors.h>

Граф наследования:InvalidAuthDataFormatError:

Граф связей класса InvalidAuthDataFormatError:

Открытые члены

- InvalidAuthDataFormatError (const string &func, bool is_critical=false)
  Конструктор InvalidAuthDataFormatError.
- InvalidAuthDataFormatError (const string &func, bool is_critical=false)
  Конструктор InvalidAuthDataFormatError.

Дополнительные унаследованные члены

4.5.1 Подробное описание

Ошибка некорректного формата данных аутентификации.

4.5.2 Конструктор(ы)

4.5.2.1 InvalidAuthDataFormatError() [1/2]

```
InvalidAuthDataFormatError::InvalidAuthDataFormatError (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidAuthDataFormatError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.5.2.2 InvalidAuthDataFormatError() [2/2]

```
InvalidAuthDataFormatError::InvalidAuthDataFormatError (
        const string & func,
        bool is_critical = false )
```

Конструктор InvalidAuthDataFormatError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.6 Класс InvalidLineFormat

Ошибка некорректного формата строки.

#include <errors.h>

Граф наследования:InvalidLineFormat:

Граф связей класса InvalidLineFormat:

Открытые члены

- InvalidLineFormat (const string &func, bool is_critical=false)
  Конструктор InvalidLineFormat.
- InvalidLineFormat (const string &func, bool is_critical=false)
  Конструктор InvalidLineFormat.

Дополнительные унаследованные члены

4.6.1 Подробное описание

Ошибка некорректного формата строки.

4.6.2 Конструктор(ы)

4.6.2.1 InvalidLineFormat() [1/2]

```
InvalidLineFormat::InvalidLineFormat (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidLineFormat.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.6.2.2 InvalidLineFormat() [2/2]

```
InvalidLineFormat::InvalidLineFormat (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidLineFormat.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.7 Класс InvalidLoginOrPasswordError

Ошибка некорректного логина или пароля.

#include <errors.h>

Граф наследования:InvalidLoginOrPasswordError:

Граф связей класса InvalidLoginOrPasswordError:

Открытые члены

- InvalidLoginOrPasswordError (const string &func, bool is_critical=false)
  Конструктор InvalidLoginOrPasswordError.
- InvalidLoginOrPasswordError (const string &func, bool is_critical=false)
  Конструктор InvalidLoginOrPasswordError.

Дополнительные унаследованные члены

4.7.1 Подробное описание

Ошибка некорректного логина или пароля.

4.7.2 Конструктор(ы)

4.7.2.1 InvalidLoginOrPasswordError() [1/2]

```
InvalidLoginOrPasswordError::InvalidLoginOrPasswordError (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidLoginOrPasswordError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.7.2.2 InvalidLoginOrPasswordError() [2/2]

```
InvalidLoginOrPasswordError::InvalidLoginOrPasswordError (
            const string & func,
            bool is_critical = false )
```

Конструктор InvalidLoginOrPasswordError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.8 Класс ListenSocketError

Ошибка прослушивания сокета.

#include <errors.h>

Граф наследования:ListenSocketError:

Граф связей класса ListenSocketError:

Открытые члены

- ListenSocketError (const string &func, bool is_critical=false)
  Конструктор ListenSocketError.
- ListenSocketError (const string &func, bool is_critical=false)
  Конструктор ListenSocketError.

Дополнительные унаследованные члены

4.8.1 Подробное описание

Ошибка прослушивания сокета.

4.8.2 Конструктор(ы)

4.8.2.1 ListenSocketError() [1/2]

```
ListenSocketError::ListenSocketError (
                const string & func,
                bool is_critical = false )
```

Конструктор ListenSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.8.2.2 ListenSocketError() [2/2]

```
ListenSocketError::ListenSocketError (
        const string & func,
        bool is_critical = false )
```

Конструктор ListenSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.9 Класс Parser

Класс для парсинга аргументов командной строки и хранения параметров.

#include <arguments.h>

Открытые члены

- Parser ()
  Конструктор по умолчанию. Устанавливает пути по умолчанию и параметры.
- void parse (int argc, char *argv[])
  Парсит аргументы командной строки.
- void help () const
  Выводит сообщение помощи.
- std::string & getPathToUsers ()
  Возвращает путь к файлу с пользователями.
- std::string & getPathToLog ()
  Возвращает путь к файлу логов.
- std::string & getAddress ()
  Возвращает адрес.

- int getPort () const
  Возвращает порт.
- Parser ()
  Конструктор по умолчанию. Устанавливает пути по умолчанию и параметры.
- void parse (int argc, char *argv[])
  Парсит аргументы командной строки.
- void help () const
  Выводит сообщение помощи.
- std::string & getPathToUsers ()
  Возвращает путь к файлу с пользователями.
- std::string & getPathToLog ()
  Возвращает путь к файлу логов.
- std::string & getAddress ()
  Возвращает адрес.
- int getPort () const
  Возвращает порт.

4.9.1 Подробное описание

Класс для парсинга аргументов командной строки и хранения параметров.

4.9.2 Методы

4.9.2.1 getAddress() [1/2]

std::string & Parser::getAddress ( )

Возвращает адрес.

Возвращает

std::string& Адрес.

4.9.2.2 getAddress() [2/2]

std::string & Parser::getAddress ( )

Возвращает адрес.

Возвращает

std::string& Адрес.

4.9.2.3 getPathToLog() [1/2]

std::string & Parser::getPathToLog ( )

Возвращает путь к файлу логов.

Возвращает

std::string& Путь к файлу логов.

4.9.2.4 getPathToLog() [2/2]

std::string & Parser::getPathToLog ( )

Возвращает путь к файлу логов.

Возвращает

std::string& Путь к файлу логов.

4.9.2.5 getPathToUsers() [1/2]

std::string & Parser::getPathToUsers ( )

Возвращает путь к файлу с пользователями.

Возвращает

std::string& Путь к файлу с пользователями.

4.9.2.6 getPathToUsers() [2/2]

std::string & Parser::getPathToUsers ( )

Возвращает путь к файлу с пользователями.

Возвращает

std::string& Путь к файлу с пользователями.

4.9.2.7 getPort() [1/2]

int Parser::getPort ( ) const

Возвращает порт.

Возвращает

int Порт.

4.9.2.8 getPort() [2/2]

int Parser::getPort ( ) const

Возвращает порт.

Возвращает

int Порт.

4.9.2.9 parse() [1/2]

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

Парсит аргументы командной строки.

Аргументы

argc	Количество аргументов.
argv	Массив аргументов.

Исключения

InvalidArgumentError	если указан неверный аргумент.

4.9.2.10 parse() [2/2]

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

Парсит аргументы командной строки.

Аргументы

argc	Количество аргументов.
argv	Массив аргументов.

Исключения

InvalidArgumentError	если указан неверный аргумент.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/arguments.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/arguments.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/arguments.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/arguments.cpp

4.10 Класс ReadWriteError

Ошибка чтения/записи.

#include <errors.h>

Граф наследования:ReadWriteError:

Граф связей класса ReadWriteError:

Открытые члены

- ReadWriteError (const string &func, bool is_critical=false)
  Конструктор ReadWriteError.
- ReadWriteError (const string &func, bool is_critical=false)
  Конструктор ReadWriteError.

Дополнительные унаследованные члены

4.10.1 Подробное описание

Ошибка чтения/записи.

4.10.2 Конструктор(ы)

4.10.2.1 ReadWriteError() [1/2]

```
ReadWriteError::ReadWriteError (
            const string & func,
            bool is_critical = false )
```

Конструктор ReadWriteError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.10.2.2 ReadWriteError() [2/2]

```
ReadWriteError::ReadWriteError (
            const string & func,
            bool is_critical = false )
```

Конструктор ReadWriteError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.11 Класс RecvDataError

Ошибка получения данных.

#include <errors.h>

Граф наследования:RecvDataError:

Граф связей класса RecvDataError:

Открытые члены

- RecvDataError (const string &func, bool is_critical=false)
  Конструктор RecvDataError.
- RecvDataError (const string &func, bool is_critical=false)
  Конструктор RecvDataError.

Дополнительные унаследованные члены

4.11.1 Подробное описание

Ошибка получения данных.

4.11.2 Конструктор(ы)

4.11.2.1 RecvDataError() [1/2]

```
RecvDataError::RecvDataError (
            const string & func,
            bool is_critical = false )
```

Конструктор RecvDataError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.11.2.2 RecvDataError() [2/2]

```
RecvDataError::RecvDataError (
            const string & func,
            bool is_critical = false )
```

Конструктор RecvDataError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.12 Класс SendDataError

Ошибка отправки данных.

#include <errors.h>

Граф наследования:SendDataError:

Граф связей класса SendDataError:

Открытые члены

- SendDataError (const string &func, bool is_critical=false)
  Конструктор SendDataError.
- SendDataError (const string &func, bool is_critical=false)
  Конструктор SendDataError.

Дополнительные унаследованные члены

4.12.1 Подробное описание

Ошибка отправки данных.

4.12.2 Конструктор(ы)

4.12.2.1 SendDataError() [1/2]

```
SendDataError::SendDataError (
            const string & func,
            bool is_critical = false )
```

Конструктор SendDataError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.12.2.2 SendDataError() [2/2]

```
SendDataError::SendDataError (
            const string & func,
            bool is_critical = false )
```

Конструктор SendDataError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.13 Класс Server

Класс для представления сервера, способного обрабатывать подключения и аутентификацию пользователей.

#include <server.h>

Открытые члены

- Server (const string &address, uint16_t port, const map< string, string > &users)
  Конструктор Server.
- string & getAddress ()
  Возвращает адрес сервера.
- uint16_t & getPort ()
  Возвращает порт сервера.
- map< string, string > & getUsers ()
  Возвращает логины и пароли пользователей.
- void startup ()
  Запускает сервер.
- void shutdown ()
  Останавливает сервер.
- void connect ()
  Ожидает подключения клиента.
- void authentication ()
  Аутентифицирует клиента.
- void averange ()
  Вычисляет среднее арифметическое значений, полученных от клиента.
- Server (const string &address, uint16_t port, const map< string, string > &users)
  Конструктор Server.
- string & getAddress ()
  Возвращает адрес сервера.
- uint16_t & getPort ()
  Возвращает порт сервера.
- map< string, string > & getUsers ()

Возвращает логины и пароли пользователей.

- void startup ()

  Запускает сервер.

- void shutdown ()

  Останавливает сервер.

- void connect ()

  Ожидает подключения клиента.

- void authentication ()

  Аутентифицирует клиента.

- void averange ()

  Вычисляет среднее арифметическое значений, полученных от клиента.

4.13.1 Подробное описание

Класс для представления сервера, способного обрабатывать подключения и аутентификацию пользователей.

4.13.2 Конструктор(ы)

4.13.2.1 Server() [1/2]

```
Server::Server (
            const string & address,
            uint16_t port,
            const map< string, string > & users )
```

Конструктор Server.

Аргументы

address	Адрес сервера.
port	Порт сервера.
users	Логины и пароли пользователей.

4.13.2.2 Server() [2/2]

```
Server::Server (
            const string & address,
            uint16_t port,
            const map< string, string > & users )
```

Конструктор Server.

Аргументы

address	Адрес сервера.
port	Порт сервера.
users	Логины и пароли пользователей.

4.13.3 Методы

4.13.3.1 authentication() [1/2]

void Server::authentication ( )

Аутентифицирует клиента.

Исключения

RecvDataError	Если данные не могут быть получены от клиента.
InvalidAuthDataFormatError	Если формат данных аутентификации некорректен.
InvalidLoginOrPasswordError	Если логин или пароль некорректны.

4.13.3.2 authentication() [2/2]

void Server::authentication ( )

Аутентифицирует клиента.

Исключения

RecvDataError	Если данные не могут быть получены от клиента.
InvalidAuthDataFormatError	Если формат данных аутентификации некорректен.
InvalidLoginOrPasswordError	Если логин или пароль некорректны.

4.13.3.3 averange() [1/2]

void Server::averange ( )

Вычисляет среднее арифметическое значений, полученных от клиента.

Исключения

RecvDataError	Если данные не могут быть получены от клиента.
SendDataError	Если данные не могут быть отправлены клиенту.

4.13.3.4 averange() [2/2]

void Server::averange ( )

Вычисляет среднее арифметическое значений, полученных от клиента.

Исключения

RecvDataError	Если данные не могут быть получены от клиента.
SendDataError	Если данные не могут быть отправлены клиенту.

4.13.3.5 connect() [1/2]

void Server::connect ( )

Ожидает подключения клиента.

Исключения

AcceptConnectionError	Если подключение не может быть принято.

4.13.3.6 connect() [2/2]

void Server::connect ( )

Ожидает подключения клиента.

Исключения

AcceptConnectionError	Если подключение не может быть принято.

4.13.3.7 getAddress() [1/2]

string & Server::getAddress ( )

Возвращает адрес сервера.

Возвращает

string& Адрес сервера.

4.13.3.8 getAddress() [2/2]

string & Server::getAddress ( )

Возвращает адрес сервера.

Возвращает

string& Адрес сервера.

4.13.3.9 getPort() [1/2]

uint16_t & Server::getPort ( )

Возвращает порт сервера.

Возвращает

uint16_t& Порт сервера.

4.13.3.10 getPort() [2/2]

uint16_t & Server::getPort ( )

Возвращает порт сервера.

Возвращает

uint16_t& Порт сервера.

4.13.3.11 getUsers() [1/2]

map< string, string > & Server::getUsers ( )

Возвращает логины и пароли пользователей.

Возвращает

map<string, string>& Логины и пароли пользователей.

4.13.3.12 getUsers() [2/2]

map< string, string > & Server::getUsers ( )

Возвращает логины и пароли пользователей.

Возвращает

map<string, string>& Логины и пароли пользователей.

4.13.3.13 shutdown() [1/2]

void Server::shutdown ( )

Останавливает сервер.

Исключения

ShutdownSocketError	Если сокет не может быть корректно закрыт.

4.13.3.14 shutdown() [2/2]

void Server::shutdown ( )

Останавливает сервер.

Исключения

ShutdownSocketError	Если сокет не может быть корректно закрыт.

4.13.3.15 startup() [1/2]

void Server::startup ( )

Запускает сервер.

Исключения

CreateSocketError	Если сокет не может быть создан.
BindSocketError	Если сокет не может быть привязан к адресу.
ListenSocketError	Если сокет не может быть переведен в режим прослушивания.

4.13.3.16 startup() [2/2]

void Server::startup ( )

Запускает сервер.

Исключения

CreateSocketError	Если сокет не может быть создан.
BindSocketError	Если сокет не может быть привязан к адресу.
ListenSocketError	Если сокет не может быть переведен в режим прослушивания.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/server.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/server.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/server.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/server.cpp

4.14 Класс ShutdownSocketError

Ошибка завершения работы сокета.

#include <errors.h>

Граф наследования:ShutdownSocketError:

Граф связей класса ShutdownSocketError:

Открытые члены

- ShutdownSocketError (const string &func, bool is_critical=false)
  Конструктор ShutdownSocketError.
- ShutdownSocketError (const string &func, bool is_critical=false)
  Конструктор ShutdownSocketError.

Дополнительные унаследованные члены

4.14.1 Подробное описание

Ошибка завершения работы сокета.

4.14.2 Конструктор(ы)

4.14.2.1 ShutdownSocketError() [1/2]

```
ShutdownSocketError::ShutdownSocketError (
            const string & func,
            bool is_critical = false )
```

Конструктор ShutdownSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.14.2.2 ShutdownSocketError() [2/2]

```
ShutdownSocketError::ShutdownSocketError (
            const string & func,
            bool is_critical = false )
```

Конструктор ShutdownSocketError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

4.15 Класс UnknownError

Базовый класс для всех ошибок.

#include <errors.h>

Граф наследования:UnknownError:

Граф связей класса UnknownError:

Открытые члены

- UnknownError (const string &func, bool is_critical=false)
  Конструктор UnknownError.
- string getName () const
  Получает имя ошибки.
- string getFunc () const
  Получает имя функции, в которой возникла ошибка.
- string getMessage () const
  Получает сообщение об ошибке.
- time_t getTime () const
  Получает время возникновения ошибки.
- bool isCritical () const
  Проверяет, является ли ошибка критической.
- const char * what () const noexcept override
  Получает сообщение об ошибке для вывода.
- UnknownError (const string &func, bool is_critical=false)
  Конструктор UnknownError.
- string getName () const
  Получает имя ошибки.
- string getFunc () const
  Получает имя функции, в которой возникла ошибка.
- string getMessage () const
  Получает сообщение об ошибке.
- time_t getTime () const
  Получает время возникновения ошибки.
- bool isCritical () const
  Проверяет, является ли ошибка критической.
- const char * what () const noexcept override
  Получает сообщение об ошибке для вывода.

Защищенные данные

- string name = "UnknownError"
  Имя ошибки.

4.15.1 Подробное описание

Базовый класс для всех ошибок.

4.15.2 Конструктор(ы)

4.15.2.1 UnknownError() [1/2]

```
UnknownError::UnknownError (
            const string & func,
            bool is_critical = false )
```

Конструктор UnknownError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.15.2.2 UnknownError() [2/2]

```
UnknownError::UnknownError (
            const string & func,
            bool is_critical = false )
```

Конструктор UnknownError.

Аргументы

func	Имя функции, в которой возникла ошибка.
is_critical	Флаг критичности ошибки.

4.15.3 Методы

4.15.3.1 getFunc() [1/2]

```
string UnknownError::getFunc ( ) const
```

Получает имя функции, в которой возникла ошибка.

Возвращает

Имя функции.

4.15.3.2 getFunc() [2/2]

string UnknownError::getFunc ( ) const

Получает имя функции, в которой возникла ошибка.

Возвращает

Имя функции.

4.15.3.3 getMessage() [1/2]

string UnknownError::getMessage ( ) const

Получает сообщение об ошибке.

Возвращает

Сообщение об ошибке.

4.15.3.4 getMessage() [2/2]

string UnknownError::getMessage ( ) const

Получает сообщение об ошибке.

Возвращает

Сообщение об ошибке.

4.15.3.5 getName() [1/2]

string UnknownError::getName ( ) const

Получает имя ошибки.

Возвращает

Имя ошибки.

4.15.3.6 getName() [2/2]

string UnknownError::getName ( ) const

Получает имя ошибки.

Возвращает

Имя ошибки.

4.15.3.7 getTime() [1/2]

time_t UnknownError::getTime ( ) const

Получает время возникновения ошибки.

Возвращает

Время возникновения ошибки.

4.15.3.8 getTime() [2/2]

time_t UnknownError::getTime ( ) const

Получает время возникновения ошибки.

Возвращает

Время возникновения ошибки.

4.15.3.9 isCritical() [1/2]

bool UnknownError::isCritical ( ) const

Проверяет, является ли ошибка критической.

Возвращает

true, если ошибка критическая, иначе false.

4.15.3.10 isCritical() [2/2]

bool UnknownError::isCritical ( ) const

Проверяет, является ли ошибка критической.

Возвращает

true, если ошибка критическая, иначе false.

4.15.3.11 what() [1/2]

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

Получает сообщение об ошибке для вывода.

Возвращает

Сообщение об ошибке.

4.15.3.12 what() [2/2]

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

Получает сообщение об ошибке для вывода.

Возвращает

Сообщение об ошибке.

Объявления и описания членов классов находятся в файлах:

- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.h
- /home/student/Документы/busigin/MUBypass/doxygen/code/errors.cpp
- /home/student/Документы/busigin/MUBypass/doxygen/unit/errors.cpp

Глава 5

Файлы

5.1 errors.h

```
1 #ifndef ERRORS_H
2 #define ERRORS_H
3
4 #include <string>
5 #include <ctime>
6 #include <exception>
7
8 using namespace std;
9
14 class UnknownError : public exception
15 {
16 public:
22     UnknownError(const string &func, bool is_critical = false);
23
28     string getName() const;
29
34     string getFunc() const;
35
40     string getMessage() const;
41
46     time_t getTime() const;
47
52     bool isCritical() const;
53
58     const char *what() const noexcept override;
59
60 protected:
61     string name = "UnknownError";
62
63 private:
64     string func;
65     time_t time;
66     bool is_critical;
67     mutable string message;
68 };
69
74 class InvalidArgumentError : public UnknownError
75 {
76 public:
82     InvalidArgumentError(const string &func, bool is_critical = false);
83 };
84
85 // Ошибки чтения/записи
86
91 class ReadWriteError : public UnknownError
92 {
93 public:
99     ReadWriteError(const string &func, bool is_critical = false);
100 };
101
106 class InvalidLineFormat : public UnknownError
107 {
108 public:
114     InvalidLineFormat(const string &func, bool is_critical = false);
115 };
116
117 // Ошибки сокета
118
```

```
123 class CreateSocketError : public UnknownError
124 {
125 public:
131     CreateSocketError(const string &func, bool is_critical = false);
132 };
133
138 class BindSocketError : public UnknownError
139 {
140 public:
146     BindSocketError(const string &func, bool is_critical = false);
147 };
148
153 class ListenSocketError : public UnknownError
154 {
155 public:
161     ListenSocketError(const string &func, bool is_critical = false);
162 };
163
168 class ShutdownSocketError : public UnknownError
169 {
170 public:
176     ShutdownSocketError(const string &func, bool is_critical = false);
177 };
178
183 class AcceptConnectionError : public UnknownError
184 {
185 public:
191     AcceptConnectionError(const string &func, bool is_critical = false);
192 };
193
198 class RecvDataError : public UnknownError
199 {
200 public:
206     RecvDataError(const string &func, bool is_critical = false);
207 };
208
213 class SendDataError : public UnknownError
214 {
215 public:
221     SendDataError(const string &func, bool is_critical = false);
222 };
223
224 // Ошибки аутентификации
225
230 class InvalidAuthDataFormatError : public UnknownError
231 {
232 public:
238     InvalidAuthDataFormatError(const string &func, bool is_critical = false);
239 };
240
245 class InvalidLoginOrPasswordError : public UnknownError
246 {
247 public:
253     InvalidLoginOrPasswordError(const string &func, bool is_critical = false);
254 };
255
256 #endif // ERRORS_H
```

5.2 errors.h

```
1 #ifndef ERRORS_H
2 #define ERRORS_H
3
4 #include <string>
5 #include <ctime>
6 #include <exception>
7
8 using namespace std;
9
14 class UnknownError : public exception
15 {
16 public:
22     UnknownError(const string &func, bool is_critical = false);
23
28     string getName() const;
29
34     string getFunc() const;
35
40     string getMessage() const;
41
46     time_t getTime() const;
47
52     bool isCritical() const;
53
```

```
58     const char *what() const noexcept override;
59
60 protected:
61     string name = "UnknownError";
62
63 private:
64     string func;
65     time_t time;
66     bool is_critical;
67     mutable string message;
68 };
69
74 class InvalidArgumentError : public UnknownError
75 {
76 public:
82     InvalidArgumentError(const string &func, bool is_critical = false);
83 };
84
85 // Ошибки чтения/записи
86
91 class ReadWriteError : public UnknownError
92 {
93 public:
99     ReadWriteError(const string &func, bool is_critical = false);
100 };
101
106 class InvalidLineFormat : public UnknownError
107 {
108 public:
114     InvalidLineFormat(const string &func, bool is_critical = false);
115 };
116
117 // Ошибки сокета
118
123 class CreateSocketError : public UnknownError
124 {
125 public:
131     CreateSocketError(const string &func, bool is_critical = false);
132 };
133
138 class BindSocketError : public UnknownError
139 {
140 public:
146     BindSocketError(const string &func, bool is_critical = false);
147 };
148
153 class ListenSocketError : public UnknownError
154 {
155 public:
161     ListenSocketError(const string &func, bool is_critical = false);
162 };
163
168 class ShutdownSocketError : public UnknownError
169 {
170 public:
176     ShutdownSocketError(const string &func, bool is_critical = false);
177 };
178
183 class AcceptConnectionError : public UnknownError
184 {
185 public:
191     AcceptConnectionError(const string &func, bool is_critical = false);
192 };
193
198 class RecvDataError : public UnknownError
199 {
200 public:
206     RecvDataError(const string &func, bool is_critical = false);
207 };
208
213 class SendDataError : public UnknownError
214 {
215 public:
221     SendDataError(const string &func, bool is_critical = false);
222 };
223
224 // Ошибки аутентификации
225
230 class InvalidAuthDataFormatError : public UnknownError
231 {
232 public:
238     InvalidAuthDataFormatError(const string &func, bool is_critical = false);
239 };
240
245 class InvalidLoginOrPasswordError : public UnknownError
246 {
247 public:
```

```
253     InvalidLoginOrPasswordError(const string &func, bool is_critical = false);
254 };
255
256 #endif // ERRORS_H
```

5.3 arguments.h

```
1 #ifndef PARSER_H
2 #define PARSER_H
3
4 #include <string>
5 #include <cstring>
6 #include <iostream>
7
8 #include "errors.h"
9
10 using namespace std;
11
16 class Parser
17 {
18 public:
22     Parser();
23
31     void parse(int argc, char *argv[]);
32
36     void help() const;
37
43     std::string& getPathToUsers();
44
50     std::string& getPathToLog();
51
57     std::string& getAddress();
58
64     int getPort() const;
65
66 private:
67     std::string pathToUsers;
68     std::string pathToLog;
69     std::string address;
70     int port;
71 };
72
73 #endif // PARSER_H
```

5.4 arguments.h

```
1 #ifndef PARSER_H
2 #define PARSER_H
3
4 #include <string>
5 #include <cstring>
6 #include <iostream>
7
8 #include "errors.h"
9
10 using namespace std;
11
16 class Parser
17 {
18 public:
22     Parser();
23
31     void parse(int argc, char *argv[]);
32
36     void help() const;
37
43     std::string& getPathToUsers();
44
50     std::string& getPathToLog();
51
57     std::string& getAddress();
58
64     int getPort() const;
65
66 private:
67     std::string pathToUsers;
68     std::string pathToLog;
69     std::string address;
70     int port;
71 };
```

```
72
73 #endif // PARSER_H
```

5.5 rw.h

```
1 #ifndef RW_H
2 #define RW_H
3
4 #include <map>
5 #include <string>
6
7 #include "errors.h"
8
9 using namespace std;
10
19 map<string, string> readUsers(const string &path);
20
28 void writeLog(const string &path, const UnknownError &e);
29
30 #endif // RW_H
```

5.6 rw.h

```
1 #ifndef RW_H
2 #define RW_H
3
4 #include <map>
5 #include <string>
6
7 #include "errors.h"
8
9 using namespace std;
10
19 map<string, string> readUsers(const string &path);
20
28 void writeLog(const string &path, const UnknownError &e);
29
30 #endif // RW_H
```

5.7 sha1.h

```
1 #ifndef SHA1_H
2 #define SHA1_H
3
4 #include <string>
5 #include <cryptopp/sha.h>
6 #include <cryptopp/hex.h>
7 #include <cryptopp/osrng.h>
8 #include <cryptopp/filters.h>
9
10 using namespace std;
11
18 string SHA1(const string &data);
19
20 #endif // SHA1_H
```

5.8 sha1.h

```
1 #ifndef SHA1_H
2 #define SHA1_H
3
4 #include <string>
5 #include <cryptopp/sha.h>
6 #include <cryptopp/hex.h>
7 #include <cryptopp/osrng.h>
8 #include <cryptopp/filters.h>
9
10 using namespace std;
11
18 string SHA1(const string &data);
19
20 #endif // SHA1_H
```

5.9 server.h

```
1 #ifndef SERVER_H
2 #define SERVER_H
3
4 #include "sha1.h"
5 #include "errors.h"
6
7 #include <map>
8 #include <string>
9 #include <vector>
10 #include <cstring>
11 #include <cstdint>
12 #include <iostream>
13
14 #include <unistd.h>
15 #include <arpa/inet.h>
16 #include <sys/types.h>
17 #include <sys/socket.h>
18 #include <netinet/in.h>
19
20 using namespace std;
21
26 class Server
27 {
28 public:
35     Server(const string &address, uint16_t port, const map<string, string> &users);
36
41     string& getAddress();
42
47     uint16_t& getPort();
48
53     map<string, string>& getUsers();
54
61     void startup();
62
67     void shutdown();
68
73     void connect();
74
81     void authentication();
82
88     void averange();
89
90 private:
91     string address;
92     uint16_t port;
93     map<string, string> users;
94     int socket;
95     int client;
96 };
97
98 #endif // SERVER_H
```

5.10 server.h

```
1 #ifndef SERVER_H
2 #define SERVER_H
3
4 #include "sha1.h"
5 #include "errors.h"
6
7 #include <map>
8 #include <string>
9 #include <vector>
10 #include <cstring>
11 #include <cstdint>
12 #include <iostream>
13
14 #include <unistd.h>
15 #include <arpa/inet.h>
16 #include <sys/types.h>
17 #include <sys/socket.h>
18 #include <netinet/in.h>
19
20 using namespace std;
21
26 class Server
27 {
28 public:
35     Server(const string &address, uint16_t port, const map<string, string> &users);
36
41     string& getAddress();
```

```
42
47     uint16_t& getPort();
48
53     map<string, string>& getUsers();
54
61     void startup();
62
67     void shutdown();
68
73     void connect();
74
81     void authentication();
82
88     void averange();
89
90 private:
91     string address;
92     uint16_t port;
93     map<string, string> users;
94     int socket;
95     int client;
96 };
97
98 #endif // SERVER_H
```

Предметный указатель