My Project

Создано системой Doxygen 1.9.8

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 Calculator

Класс для математических вычислений

#include <Calculator.h>

Открытые статические члены

- static uint16_t calculateSumOfSquares (const std::vector< uint16_t > &data)
  Вычисление суммы квадратов элементов вектора

3.1.1 Подробное описание

Класс для математических вычислений

Содержит статические методы для вычислений над векторами данных

3.1.2 Методы

3.1.2.1 calculateSumOfSquares()

uint16_t Calculator::calculateSumOfSquares (
const std::vector< uint16_t > & data) [static]

Вычисление суммы квадратов элементов вектора

Аргументы

in	data	Вектор значений типа uint16_t

Возвращает

Результат вычислений (uint16_t)

Заметки

При переполнении возвращается максимальное значение uint16_t

См. также

Требования к функции обработки данных (ТЗ п.4.2.5)

Аргументы

in	data	Вектор значений типа uint16_t

Возвращает

Результат вычислений (uint16_t)

Заметки

При переполнении возвращается максимальное значение uint16_t

Алгоритм соответствует требованиям ТЗ п.4.2.5:

- Проверка переполнения при суммировании
- Возврат 2^15-1 при переполнении вверх
- Возврат 0 при переполнении вниз

Объявления и описания членов классов находятся в файлах:

- Calculator.h
- Calculator.cpp

3.2 Класс Logger

Класс для ведения журнала работы сервера

#include <Logger.h>

Открытые члены

- void logError (const std::string &severity, const std::string &message, const std::string &params="")
  Запись ошибки в журнал
- ~Logger ()
  Деструктор

Открытые статические члены

- static Logger & getInstance (const std::string &filename="/var/log/scale.log")
  Получение экземпляра логгера

3.2.1 Подробное описание

Класс для ведения журнала работы сервера

Реализует паттерн Singleton для глобального доступа к логгеру Потокобезопасен благодаря использованию мьютекса

3.2.2 Конструктор(ы)

3.2.2.1 ~Logger()

Logger::~Logger ( )

Деструктор

Закрывает файл журнала

3.2.3 Методы

3.2.3.1 getInstance()

Logger & Logger::getInstance (
const std::string & filename = "/var/log/scale.log") [static]

Получение экземпляра логгера

Аргументы

in	filename	Имя файла журнала (по умолчанию /var/log/scale.log)

Возвращает

Ссылка на единственный экземпляр логгера

Аргументы

in	filename	Имя файла журнала (по умолчанию /var/log/scale.log)

Возвращает

Ссылка на единственный экземпляр логгера

Заметки

Создает экземпляр при первом вызове

3.2.3.2 logError()

```
void Logger::logError (
                const std::string & severity,
                const std::string & message,
                const std::string & params = "" )
```

Запись ошибки в журнал

Аргументы

in	severity	Уровень серьезности ошибки
in	message	Текст сообщения
in	params	Дополнительные параметры ошибки
in	severity	Уровень серьезности ошибки
in	message	Текст сообщения
in	params	Дополнительные параметры ошибки

Формат записи: время | уровень | сообщение | параметры Соответствует требованиям ТЗ п.4.2.6

Объявления и описания членов классов находятся в файлах:

- Logger.h
- Logger.cpp

3.3 Класс Server

Основной класс сетевого сервера

#include <Server.h>

Открытые члены

- Server ()
  Конструктор по умолчанию
- void parseCommandLine (int argc, char **argv)
  Разбор параметров командной строки
- void start ()
  Запуск сервера

3.3.1 Подробное описание

Основной класс сетевого сервера

Обрабатывает параметры командной строки и запускает сервер

3.3.2 Конструктор(ы)

3.3.2.1 Server()

```
Server::Server ( )
```

Конструктор по умолчанию

Инициализирует порт значением по умолчанию (33333)

3.3.3 Методы

3.3.3.1 parseCommandLine()

```
void Server::parseCommandLine (
                int argc,
                char ** argv )
```

Разбор параметров командной строки

Аргументы

in	argc	Количество аргументов
in	argv	Массив аргументов
in	argc	Количество аргументов
in	argv	Массив аргументов

Поддерживает параметры: -p, –port PORT - установка порта сервера -h, –help - вывод справки

3.3.3.2 start()

```
void Server::start ( )
```

Запуск сервера

Запускает сервер на указанном порту

Запускает сервер на указанном порту

Заметки

В текущей реализации содержит заглушку для демонстрации

Объявления и описания членов классов находятся в файлах:

- Server.h
- Server.cpp

3.4 Класс Session

Класс для обработки клиентской сессии

#include <Session.h>

Открытые члены

- Session (int socket, UserDatabase &db, Logger &log)
  Конструктор сессии
- void handleSession ()
  Обработка клиентской сессии

3.4.1 Подробное описание

Класс для обработки клиентской сессии

Управляет всем жизненным циклом соединения с клиентом: аутентификация, прием данных, вычисления, отправка результатов

3.4.2 Конструктор(ы)

3.4.2.1 Session()

```
Session::Session (
            int socket,
            UserDatabase & db,
            Logger & log )
```

Конструктор сессии

Аргументы

in	socket	Дескриптор клиентского сокета
in	db	Ссылка на базу пользователей
in	log	Ссылка на логгер

3.4.3 Методы

3.4.3.1 handleSession()

```
void Session::handleSession ( )
```

Обработка клиентской сессии

Выполняет всю последовательность обработки клиента

Выполняет всю последовательность обработки клиента:

1. Аутентификация
2. Обработка векторов данных
3. Закрытие соединения

   Заметки

   В текущей реализации содержатся заглушки для демонстрации

Объявления и описания членов классов находятся в файлах:

- Session.h
- Session.cpp

3.5 Класс UserDatabase

Класс для работы с базой данных пользователей

#include <UserDatabase.h>

Открытые члены

- UserDatabase ()

  Конструктор по умолчанию
- bool loadFromFile (const std::string &filename)

  Загрузка базы пользователей из файла
- bool checkUser (const std::string &login)

  Проверка наличия пользователя в базе
- std::string getPassword (const std::string &login)

  Получение пароля пользователя

3.5.1 Подробное описание

Класс для работы с базой данных пользователей

Хранит пары "логин:пароль" и предоставляет методы для проверки

3.5.2 Методы

3.5.2.1 checkUser()

```
bool UserDatabase::checkUser (
            const std::string & login )
```

Проверка наличия пользователя в базе

Аргументы

in	login	Логин пользователя

Возвращает

true - пользователь существует, false - не существует

3.5.2.2 getPassword()

```
std::string UserDatabase::getPassword (
            const std::string & login )
```

Получение пароля пользователя

Аргументы

in	login	Логин пользователя

Возвращает

Пароль пользователя или пустую строку, если пользователь не найден

3.5.2.3 loadFromFile()

```
bool UserDatabase::loadFromFile (
            const std::string & filename )
```

Загрузка базы пользователей из файла

Аргументы

in	filename	Имя файла с базой пользователей

Возвращает

true - успешная загрузка, false - ошибка

Формат файла: логин:пароль (по одной паре на строку)

Аргументы

in	filename	Имя файла с базой пользователей

Возвращает

true - успешная загрузка, false - ошибка

Формат файла: логин:пароль (по одной паре на строку)

Заметки

Соответствует требованиям ТЗ п.4.2.2

Объявления и описания членов классов находятся в файлах:

- UserDatabase.h
- UserDatabase.cpp

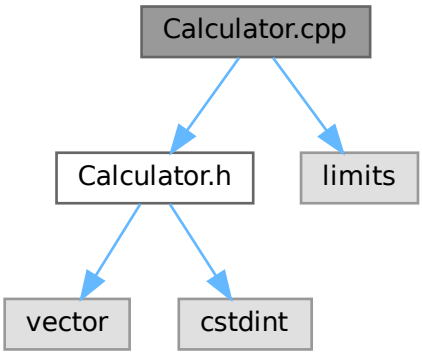
Глава 4

Файлы

4.1 Файл Calculator.cpp

Реализация класса Calculator.

#include "Calculator.h"
#include <limits>
Граф включаемых заголовочных файлов для Calculator.cpp:



4.1.1 Подробное описание

Реализация класса Calculator.

Автор

Назарова Софья

Версия

1.0

Дата

2025

Авторство

WECT ПГУ

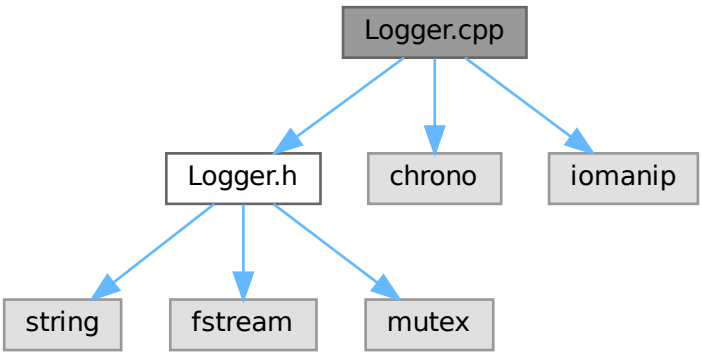
4.2 Calculator.h

```
00001 #ifndef CALCULATOR_H
00002 #define CALCULATOR_H
00003
00004 #include <vector>
00005 #include <cstdint>
00006
00012 class Calculator {
00013 public:
00021     static uint16_t calculateSumOfSquares(const std::vector<uint16_t>& data);
00022 };
00023
00024 #endif
```

4.3 Файл Logger.cpp

Реализация класса Logger.

#include "Logger.h"
#include <chrono>
#include <iomanip>
Граф включаемых заголовочных файлов для Logger.cpp:



4.3.1 Подробное описание

Реализация класса Logger.

Автор

Назарова Софья

Версия

1.0

Дата

2025

Авторство

WECT ПГУ

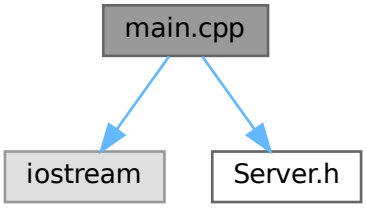
4.4 Logger.h

```
00001 #ifndef LOGGER_H
00002 #define LOGGER_H
00003
00004 #include <string>
00005 #include <fstream>
00006 #include <mutex>
00007
00014 class Logger {
00015 private:
00016     static Logger* instance;
00017     std::ofstream logFile;
00018     std::mutex logMutex;
00019
00024     Logger(const std::string& filename);
00025
00026 public:
00032     static Logger& getInstance(const std::string& filename = "/var/log/scale.log");
00033
00040     void logError(const std::string& severity, const std::string& message,
00041                   const std::string& params = "");
00042
00047     ~Logger();
00048 };
00049
00050 #endif
```

4.5 Файл main.cpp

Точка входа серверного приложения

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



Функции

- int main (int argc, char **argv)
  Основная функция программы

4.5.1 Подробное описание

Точка входа серверного приложения

Автор

Назарова Софья

Версия

1.0

Дата

2025

Создает экземпляр сервера, разбирает параметры командной строки и запускает сервер

Авторство

WECT ПГУ

4.5.2 Функции

4.5.2.1 main()

int main (
int argc,
char ** argv)

Основная функция программы

Аргументы

in	argc	Количество аргументов командной строки
in	argv	Массив аргументов командной строки

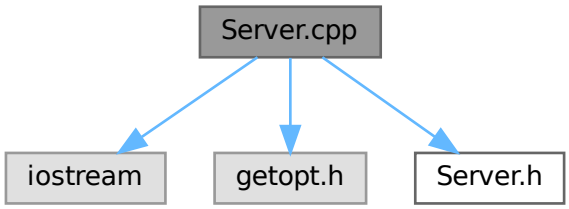
Возвращает

Код завершения программы (0 - успех, 1 - ошибка)

4.6 Файл Server.cpp

Реализация класса Server.

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



4.6.1 Подробное описание

Реализация класса Server.

Автор

Назарова Софья

Версия

1.0

Дата

2025

Авторство

WECT ПГУ

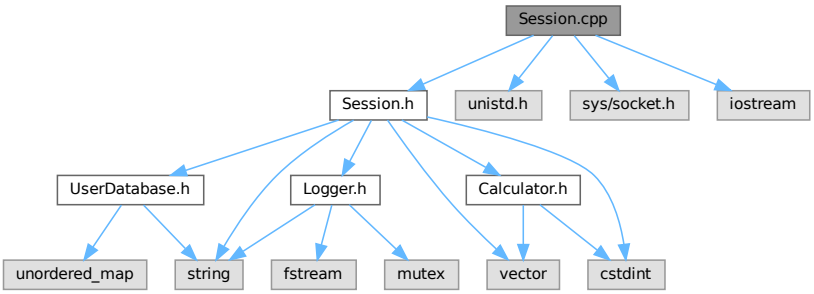
4.7 Server.h

```
00001 #ifndef SERVER_H
00002 #define SERVER_H
00003
00009 class Server {
00010 private:
00011     int port;
00012
00013 public:
00018     Server();
00019
00025     void parseCommandLine(int argc, char** argv);
00026
00031     void start();
00032 };
00033
00034 #endif
```

4.8 Файл Session.cpp

Реализация класса Session.

#include "Session.h"
#include <unistd.h>
#include <sys/socket.h>
#include <iostream>
Граф включаемых заголовочных файлов для Session.cpp:



4.8.1 Подробное описание

Реализация класса Session.

Автор

Назарова Софья

Версия

1.0

Дата

2025

Авторство

WECT ПГУ

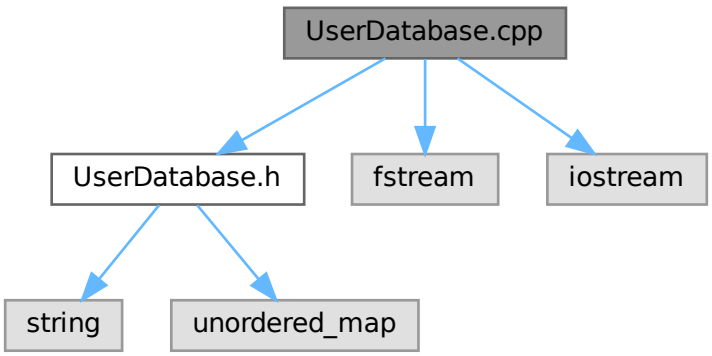
4.9 Session.h

```
00001 #ifndef SESSION_H
00002 #define SESSION_H
00003
00004 #include "UserDatabase.h"
00005 #include "Logger.h"
00006 #include "Calculator.h"
00007 #include <cstdint>
00008 #include <vector>
00009 #include <string>
00010
00017 class Session {
00018 private:
00019     int clientSocket;
00020     UserDatabase& userDB;
00021     Logger& logger;
00022
00028     bool authenticate();
00029
00035     bool processVectors();
00036
00043     bool receiveExactly(void* buffer, size_t len);
00044
00050     bool sendResult(uint16_t result);
00051
00052 public:
00059     Session(int socket, UserDatabase& db, Logger& log);
00060
00065     void handleSession();
00066 };
00067
00068 #endif
```

4.10 Файл UserDatabase.cpp

Реализация класса UserDatabase.

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



4.10.1 Подробное описание

Реализация класса UserDatabase.

Автор

Назарова Софья

Версия

1.0

Дата

2025

Авторство

WECT ПГУ

4.11 UserDatabase.h

```
00001 #ifndef USERDATABASE_H
00002 #define USERDATABASE_H
00003
00004 #include <string>
00005 #include <unordered_map>
00006
00012 class UserDatabase {
00013 private:
00014     std::unordered_map<std::string, std::string> users;
00015
00016 public:
00020     UserDatabase();
00021
00028     bool loadFromFile(const std::string& filename);
00029
00035     bool checkUser(const std::string& login);
00036
00042     std::string getPassword(const std::string& login);
00043 };
00044
00045 #endif
```

Предметный указатель