

Server

Создано системой Doxygen 1.9.4

1	Алфавитный указатель классов	1
1.1	Классы	1
2	Список файлов	3
2.1	Файлы	3
3	Классы	5
3.1	Класс Calculator	5
3.1.1	Подробное описание	5
3.1.2	Методы	5
3.1.2.1	Calculate()	5
3.2	Класс Communicator	6
3.2.1	Подробное описание	6
3.2.2	Методы	6
3.2.2.1	CreateSocket()	6
3.2.2.2	isAuthorized()	7
3.2.2.3	WaitClient()	7
3.3	Класс Logger	8
3.3.1	Подробное описание	8
3.3.2	Методы	8
3.3.2.1	Error()	8
3.3.2.2	Info()	9
3.3.2.3	Initialize()	9
3.3.2.4	Warning()	9
3.4	Класс Server	10
3.4.1	Подробное описание	10
3.4.2	Методы	10
3.4.2.1	Initialize()	10
3.4.2.2	ParseCmdLine()	10
3.4.2.3	Run()	11
3.5	Класс UsersBase	11
3.5.1	Подробное описание	12
3.5.2	Методы	12
3.5.2.1	AddUser()	12
3.5.2.2	getPassword()	12
3.5.2.3	isHasUser()	13
3.5.2.4	ParseFile()	13
4	Файлы	15
4.1	Файл include/Calculator.h	15
4.1.1	Подробное описание	15
4.2	Calculator.h	16
4.3	Файл include/Communicator.h	16
4.4	Communicator.h	16

4.5 Файл include/Logger.h	16
4.5.1 Подробное описание	17
4.6 Logger.h	17
4.7 Файл include/Server.h	17
4.7.1 Подробное описание	18
4.8 Server.h	18
4.9 Файл include/UsersBase.h	19
4.9.1 Подробное описание	19
4.10 UsersBase.h	19
Предметный указатель	21

Глава 1

Алфавитный указатель классов

1.1 Классы

Классы с их кратким описанием.

Calculator		
Класс Calculator	5
Communicator		
Класс Communicator	6
Logger		
Класс Logger	8
Server		
Класс Server	10
UsersBase		
Класс UsersBase	11

Глава 2

Список файлов

2.1 Файлы

Полный список документированных файлов.

include/	Calculator.h	
	Заголовочный файл для модуля Calculator	15
include/	Communicator.h	
	Заголовочный файл для модуля Communicator	16
include/	Logger.h	
	Заголовочный файл для модуля Logger	16
include/	Server.h	
	Заголовочный файл для модуля Server	17
include/	UsersBase.h	
	Заголовочный файл для модуля UsersBase	19

Глава 3

Классы

3.1 Класс Calculator

Класс `Calculator`.

```
#include <Calculator.h>
```

Граф связей класса Calculator:

Открытые статические члены

- static bool `Calculate` (const int _clientSocket)
Метод Calculate для выполнения операции над данными и их отправки

3.1.1 Подробное описание

Класс `Calculator`.

Для выполнения операции над данными и их отправки предназначен метод `Calculate`

Предупреждения

Реализация только для типа данных `uint32_t`

3.1.2 Методы

3.1.2.1 `Calculate()`

```
bool Calculator::Calculate (  
    const int _clientSocket ) [static]
```

Метод `Calculate` для выполнения операции над данными и их отправки

Аргументы

in	_clientSocket	сокет
----	---------------	-------

Возвращает

Метод ничего не возвращает

Объявления и описания членов классов находятся в файлах:

- include/[Calculator.h](#)
- src/[Calculator.cpp](#)

3.2 Класс Communicator

Класс [Communicator](#).

```
#include <Communicator.h>
```

Граф связей класса Communicator:

Открытые члены

- int [CreateSocket](#) (const int _port)
Создание сокета сервера
- int [WaitClient](#) (const int _socket)
Создание сокета клиента
- bool [isAuthorized](#) (int work_sock, [UsersBase](#) _usersBase)
Авторизация

3.2.1 Подробное описание

Класс [Communicator](#).

Для создания сокета сервера и слушания запросов предназначен метод CreateSocket

Для создания сокета клиента и подключения его к серверу метод WaitClient

Для авторизации клиента предназначен метод isAuthorized

3.2.2 Методы

3.2.2.1 CreateSocket()

```
int Communicator::CreateSocket (
    const int _port )
```

Создание сокета сервера

Аргументы

in	_port	порт
----	-------	------

Возвращает

Сокет

3.2.2.2 isAuthorized()

```
bool Communicator::isAuthorized (
    int work_sock,
    UsersBase _usersBase )
```

Авторизация

Аргументы

in	work_sock	сокет клиента
in	UsersBase	база данных пользователя

Возвращает

Сокет клиента

3.2.2.3 WaitClient()

```
int Communicator::WaitClient (
    const int _socket )
```

Создание сокета клиента

Аргументы

in	_socket	сокет сервера
----	---------	---------------

Возвращает

Сокет клиента

Объявления и описания членов классов находятся в файлах:

- include/[Communicator.h](#)
- src/[Communicator.cpp](#)

3.3 Класс Logger

Класс `Logger`.

```
#include <Logger.h>
```

Граф связей класса `Logger`:

Открытые члены

- `bool Initialize` (`const std::string _pathToLogFile`)
Проверка доступности файла
- `void Info` (`std::string _log`)
Выводы в логи информации
- `void Error` (`std::string _log`)
Выводы в логи ошибки
- `void Warning` (`std::string _log`)
Выводы в логи предупреждения

3.3.1 Подробное описание

Класс `Logger`.

Для записи ошибок предназначен методы `Info`, `Error`, `Warning`

3.3.2 Методы

3.3.2.1 `Error()`

```
void Logger::Error (  
    std::string _log )
```

Выводы в логи ошибки

Аргументы

in	_log	запись в файл
----	------	---------------

Возвращает

Метод ничего не возвращает

3.3.2.2 Info()

```
void Logger::Info (
    std::string _log )
```

Выводы в логи информации

Аргументы

in	_log	запись в файл
----	------	---------------

Возвращает

Метод ничего не возвращает

3.3.2.3 Initialize()

```
bool Logger::Initialize (
    const std::string _pathToLogFile )
```

Проверка доступности файла

Аргументы

in	_pathToLogFile	путь до файла
----	----------------	---------------

Возвращает

Метод возвращает false, если файл недоступен для записи

3.3.2.4 Wargning()

```
void Logger::Wargning (
    std::string _log )
```

Выводы в логи предупреждения

Аргументы

in	_log	запись в файл
----	------	---------------

Возвращает

Метод ничего не возвращает

Объявления и описания членов классов находятся в файлах:

- `include/Logger.h`
- `src/Logger.cpp`

3.4 Класс Server

Класс `Server`.

```
#include <Server.h>
```

Граф связей класса `Server`:

Открытые члены

- `bool ParseCmdLine (int argc, char **argv)`
Метод `ParseCmdLine` для парсинга
- `bool Initialize ()`
Метод `Initialize` для инициализации
- `int Run ()`
Метод `Run` для запуска и прослушки сообщений

3.4.1 Подробное описание

Класс `Server`.

Для управления сервером предназначены методы: `ParseCmdLine` для парсинга командной строки и `Initialize` для инициализации и `Run` для запуска сервера

3.4.2 Методы

3.4.2.1 `Initialize()`

```
bool Server::Initialize ( )
```

Метод `Initialize` для инициализации

Возвращает

Метод возвращает `false`, если указан неверный параметр

3.4.2.2 `ParseCmdLine()`

```
bool Server::ParseCmdLine (
    int argc,
    char ** argv )
```

Метод `ParseCmdLine` для парсинга

Аргументы

Параметры	командной строки
-----------	------------------

Возвращает

Метод возвращает false, если нарушена правильность написания аргументов

Предупреждения

Вызов до инициализации

3.4.2.3 Run()

```
int Server::Run ( )
```

Метод Run для запуска и прослушки сообщений

Возвращает

Метод ничего не возвращает

Объявления и описания членов классов находятся в файлах:

- include/[Server.h](#)
- src/[Server.cpp](#)

3.5 Класс UsersBase

Класс [UsersBase](#).

```
#include <UsersBase.h>
```

Граф связей класса UsersBase:

Открытые члены

- bool [ParseFile](#) (const std::string &_pathToFile)
Метод ParseFile для обработки базы данных пользователей
- bool [AddUser](#) (const std::string &_login, const std::string &_password)
Метод AddUser для добавления пользователя в базу данных
- std::string [getPassword](#) (const std::string &_login)
Метод getPassword возвращает пароль для логина
- const bool [isHasUser](#) (const std::string &_login)
Метод isHasUser для проверки наличия пользователя

3.5.1 Подробное описание

Класс [UsersBase](#).

Для чтения базы данных предназначен метод `ParseFile` Для проверки наличия клиента в базе данных предназначен метод `isHasUser`

3.5.2 Методы

3.5.2.1 `AddUser()`

```
bool UsersBase::AddUser (
    const std::string & _login,
    const std::string & _password )
```

Метод `AddUser` для добавления пользователя в базу данных

Аргументы

in	login	- логин
in	password	- пароль

Возвращает

Метод возвращает `false`, если пользователь уже существует

3.5.2.2 `getPassword()`

```
std::string UsersBase::getPassword (
    const std::string & _login )
```

Метод `getPassword` возвращает пароль для логина

Аргументы

in	login	- логин
----	-------	---------

Возвращает

Метод возвращает пустую строку, если пользователь отсутствует

3.5.2.3 isHasUser()

```
const bool UsersBase::isHasUser (
    const std::string & _login )
```

Метод isHasUser для проверки наличия пользователя

Аргументы

in	login	- логин
----	-------	---------

Возвращает

Метод возвращает false, если пользователь отсутствует

3.5.2.4 ParseFile()

```
bool UsersBase::ParseFile (
    const std::string & _pathToFile )
```

Метод ParseFile для обработки базы данных пользователей

Аргументы

in	_pathToUser	base - путь до базы данных пользователей
----	-------------	--

Возвращает

Метод возвращает false, если файл недоступен для чтения и если содержится ошибка в базе данных

Объявления и описания членов классов находятся в файлах:

- [include/UsersBase.h](#)
- [src/UsersBase.cpp](#)

Глава 4

Файлы

4.1 Файл `include/Calculator.h`

Заголовочный файл для модуля `Calculator`.

Классы

- `class Calculator`
Класс `Calculator`.

4.1.1 Подробное описание

Заголовочный файл для модуля `Calculator`.

Автор

Спирин И.А.

Версия

1.0

Дата

18.12.2023

Авторство

ИБСТ ПГУ

Предупреждения

Курсовая работа студента

4.2 Calculator.h

[См. документацию.](#)

```
1 #pragma once
2
17 class Calculator
18 {
19 public:
25     static bool Calculate(const int _clientSocket);
26 };
```

4.3 Файл include/Communicator.h

Заголовочный файл для модуля [Communicator](#).

```
#include "UsersBase.h"
```

Граф включаемых заголовочных файлов для Communicator.h:

4.4 Communicator.h

[См. документацию.](#)

```
1 #pragma once
2
12 #include "UsersBase.h"
13
19 class Communicator
20 {
21 public:
27     int CreateSocket(const int _port);
33     int WaitClient(const int _socket);
40     bool isAuthorized(int work_sock, UsersBase _usersBase);
41 };
```

4.5 Файл include/Logger.h

Заголовочный файл для модуля [Logger](#).

```
#include <iostream>
```

Граф включаемых заголовочных файлов для Logger.h: Граф файлов, в которые включается этот файл:

Классы

- class [Logger](#)

Класс [Logger](#).

4.5.1 Подробное описание

Заголовочный файл для модуля [Logger](#).

Автор

Спирин И.А.

Версия

1.0

Дата

18.12.2023

Авторство

ИБСТ ПГУ

Предупреждения

Курсовая работа студента

4.6 Logger.h

[См. документацию.](#)

```
1 #pragma once
2
12 #include <iostream>
13
17 class Logger{
18
19     public:
20
21         Logger() = default;
22
28         bool Initialize(const std::string _pathToLogFile);
29
35         void Info(std::string _log);
41         void Error(std::string _log);
47         void Warning(std::string _log);
48     private:
49
50         std::string m_pathToFile;
51 };
```

4.7 Файл include/Server.h

Заголовочный файл для модуля [Server](#).

```
#include "Logger.h"
#include "Comunicator.h"
#include "UsersBase.h"
```

Граф включаемых заголовочных файлов для Server.h:

Классы

- class [Server](#)
Класс [Server](#).

4.7.1 Подробное описание

Заголовочный файл для модуля [Server](#).

Автор

Спирин И.А.

Версия

1.0

Дата

18.12.2023

Авторство

ИБСТ ПГУ

Предупреждения

Курсовая работа студента

4.8 Server.h

[См. документацию.](#)

```
1 #pragma once
2
3 #include "Logger.h"
4 #include "Communicator.h"
5 #include "UsersBase.h"
6
7
8 class Server
9 {
10 public:
11     bool ParseCmdLine(int argc, char **argv);
12     bool Initialize();
13     int Run();
14
15 private:
16     std::string m_usersBasePath;
17     int m_targetPort;
18     std::string m_logFilePath;
19     UsersBase m_usersBase;
20     Communicator m_communicator;
21     Logger m_logger;
22
23     std::string getNextArg(int argc, char** argv);
24     bool isNumber(std::string _str);
25     int lastArg;
26 };
```

4.9 Файл include/UsersBase.h

Заголовочный файл для модуля [UsersBase](#).

```
#include <iostream>
#include <unordered_map>
```

Граф включаемых заголовочных файлов для UsersBase.h: Граф файлов, в которые включается этот файл:

Классы

- class [UsersBase](#)
Класс [UsersBase](#).

4.9.1 Подробное описание

Заголовочный файл для модуля [UsersBase](#).

Автор

Спирин И.А.

Версия

1.0

Дата

18.12.2023

Авторство

ИБСТ ПГУ

Предупреждения

Курсовая работа студента

4.10 UsersBase.h

[См. документацию.](#)

```
1 #pragma once
2
12 #include <iostream>
13 #include <unordered_map>
14
19 class UsersBase{
20
21     public:
22
28         bool ParseFile(const std::string& _pathToFile);
35         bool AddUser(const std::string& _login, const std::string& _password);
41         std::string getPassword(const std::string& _login);
47         const bool isHasUser(const std::string& _login);
48
49     private:
50         std::unordered_map<std::string, std::string> m_users;
51 };
```


Предметный указатель

- AddUser
 - UsersBase, [12](#)
- Calculate
 - Calculator, [5](#)
- Calculator, [5](#)
 - Calculate, [5](#)
- Communicator, [6](#)
 - CreateSocket, [6](#)
 - isAuthorized, [7](#)
 - WaitClient, [7](#)
- CreateSocket
 - Communicator, [6](#)
- Error
 - Logger, [8](#)
- getPassword
 - UsersBase, [12](#)
- include/Calculator.h, [15](#), [16](#)
- include/Communicator.h, [16](#)
- include/Logger.h, [16](#), [17](#)
- include/Server.h, [17](#), [18](#)
- include/UsersBase.h, [19](#)
- Info
 - Logger, [8](#)
- Initialize
 - Logger, [9](#)
 - Server, [10](#)
- isAuthorized
 - Communicator, [7](#)
- isHasUser
 - UsersBase, [12](#)
- Logger, [8](#)
 - Error, [8](#)
 - Info, [8](#)
 - Initialize, [9](#)
 - Wargning, [9](#)
- ParseCmdLine
 - Server, [10](#)
- ParseFile
 - UsersBase, [13](#)
- Run
 - Server, [11](#)
- Server, [10](#)
 - Initialize, [10](#)
- ParseCmdLine, [10](#)
- Run, [11](#)
- UsersBase, [11](#)
 - AddUser, [12](#)
 - getPassword, [12](#)
 - isHasUser, [12](#)
 - ParseFile, [13](#)
- WaitClient
 - Communicator, [7](#)
- Wargning
 - Logger, [9](#)