

server

Создано системой Doxygen 1.9.4



|                                |    |
|--------------------------------|----|
| 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 Calculator           | 7  |
| 4.1.1 Подробное описание       | 7  |
| 4.1.2 Конструктор(ы)           | 7  |
| 4.1.2.1 Calculator()           | 7  |
| 4.1.3 Методы                   | 8  |
| 4.1.3.1 send_res()             | 8  |
| 4.2 Класс Communicate          | 8  |
| 4.2.1 Подробное описание       | 8  |
| 4.2.2 Методы                   | 9  |
| 4.2.2.1 connection()           | 9  |
| 4.2.2.2 generate_salt()        | 9  |
| 4.2.2.3 sha1()                 | 9  |
| 4.3 Класс Connector            | 10 |
| 4.3.1 Подробное описание       | 10 |
| 4.3.2 Методы                   | 10 |
| 4.3.2.1 connect()              | 10 |
| 4.3.2.2 get_data()             | 11 |
| 4.4 Класс crit_err             | 11 |
| 4.4.1 Подробное описание       | 12 |
| 4.4.2 Конструктор(ы)           | 12 |
| 4.4.2.1 crit_err()             | 12 |
| 4.5 Класс Interface            | 12 |
| 4.5.1 Подробное описание       | 13 |
| 4.5.2 Методы                   | 13 |
| 4.5.2.1 comm_proc()            | 13 |
| 4.6 Класс Logger               | 14 |
| 4.6.1 Подробное описание       | 14 |
| 4.6.2 Конструктор(ы)           | 14 |
| 4.6.2.1 Logger()               | 14 |
| 4.6.3 Методы                   | 15 |
| 4.6.3.1 set_path()             | 15 |
| 4.6.3.2 writelog()             | 15 |
| 4.7 Класс no_crit_err          | 15 |
| 4.7.1 Подробное описание       | 16 |

---

|                           |    |
|---------------------------|----|
| 4.7.2 Конструктор(ы)      | 16 |
| 4.7.2.1 no_crit_err()     | 16 |
| 5 Файлы                   | 19 |
| 5.1 Файл Calculator.h     | 19 |
| 5.1.1 Подробное описание  | 20 |
| 5.2 Calculator.h          | 20 |
| 5.3 Файл Communicate.h    | 21 |
| 5.3.1 Подробное описание  | 21 |
| 5.4 Communicate.h         | 22 |
| 5.5 Файл Connector.h      | 22 |
| 5.5.1 Подробное описание  | 23 |
| 5.6 Connector.h           | 24 |
| 5.7 Файл Errors.h         | 24 |
| 5.7.1 Подробное описание  | 25 |
| 5.8 Errors.h              | 26 |
| 5.9 Includer.h            | 26 |
| 5.10 Файл Interface.h     | 26 |
| 5.10.1 Подробное описание | 27 |
| 5.11 Interface.h          | 27 |
| 5.12 Файл Logger.h        | 28 |
| 5.12.1 Подробное описание | 28 |
| 5.13 Logger.h             | 29 |
| Предметный указатель      | 31 |

# Глава 1

## Иерархический список классов

### 1.1 Иерархия классов

Иерархия классов.

|                       |    |
|-----------------------|----|
| Calculator . . . . .  | 7  |
| Communicate . . . . . | 8  |
| Connector . . . . .   | 10 |
| Interface . . . . .   | 12 |
| Logger . . . . .      | 14 |
| std::runtime_error    |    |
| crit_err . . . . .    | 11 |
| no_crit_err . . . . . | 15 |



## Глава 2

# Алфавитный указатель классов

### 2.1 Классы

Классы с их кратким описанием.

|                             |  |    |
|-----------------------------|--|----|
| <a href="#">Calculator</a>  | Класс для выполнения расчетов . . . . .  | 7  |
| <a href="#">Communicate</a> | Класс для управления коммуникациями . . . . .  | 8  |
| <a href="#">Connector</a>   | Класс для подключения к базе данных . . . . .  | 10 |
| <a href="#">crit_err</a>    | Класс для критических ошибок. Этот класс используется для обработки критических ошибок в программе . . . . . | 11 |
| <a href="#">Interface</a>   | Класс интерфейса пользователя . . . . .  | 12 |
| <a href="#">Logger</a>      | Класс для ведения логов . . . . .  | 14 |
| <a href="#">no_crit_err</a> | Класс для некритических ошибок . . . . .   | 15 |





## Глава 3

# Список файлов

### 3.1 Файлы

Полный список документированных файлов.

|  |    |
|--|----|
| <a href="#">Calculator.h</a>                             |    |
| Заголовочный файл для класса <a href="#">Calculator</a>  | 19 |
| <a href="#">Communicate.h</a>                            |    |
| Заголовочный файл для класса <a href="#">Communicate</a> | 21 |
| <a href="#">Connector.h</a>                              |    |
| Заголовочный файл для класса <a href="#">Connector</a>   | 22 |
| <a href="#">Errors.h</a>                                 |    |
| Заголовочный файл для классов ошибок                     | 24 |
| <a href="#">Includer.h</a>                               | ?? |
| <a href="#">Interface.h</a>                              |    |
| Заголовочный файл для класса <a href="#">Interface</a>   | 26 |
| <a href="#">Logger.h</a>                                 |    |
| Заголовочный файл для класса <a href="#">Logger</a>      | 28 |



## Глава 4

# Классы

### 4.1 Класс Calculator

Класс для выполнения расчетов.

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

Открытые члены

- `Calculator` (`std::vector< double > input_data`)  
Конструктор, который инициализирует класс с входными данными.
- `double send_res ()`  
Метод для получения результата расчетов.

Закрытые данные

- `double results = 0`

#### 4.1.1 Подробное описание

Класс для выполнения расчетов.

Данный класс принимает вектор входных данных и предоставляет метод для получения результата.

#### 4.1.2 Конструктор(ы)

##### 4.1.2.1 Calculator()

```
Calculator::Calculator (  
    std::vector< double > input_data )
```

Конструктор, который инициализирует класс с входными данными.

## Аргументы

|            |                                    |
|------------|------------------------------------|
| input_data | Вектор входных данных типа double. |
|------------|------------------------------------|

## 4.1.3 Методы

## 4.1.3.1 send\_res()

```
double Calculator::send_res ( )
```

Метод для получения результата расчетов.

Возвращает

Возвращает результат типа double.

Объявления и описания членов классов находятся в файлах:

- [Calculator.h](#)
- [Calculator.cpp](#)

## 4.2 Класс Communicate

Класс для управления коммуникациями.

```
#include <Communicate.h>
```

## Открытые члены

- `int connection (int port, std::map< std::string, std::string > database, Logger *l1)`  
Метод для установки соединения.

## Открытые статические члены

- `static std::string sha1 (std::string input_str)`  
Генерация SHA1 хеша из строки.
- `static std::string generate\_salt ()`  
Генерация соли для хеширования.

## 4.2.1 Подробное описание

Класс для управления коммуникациями.

Этот класс предоставляет методы для установления соединений и генерации хешей.

## 4.2.2 Методы

### 4.2.2.1 connection()

```
int Communicate::connection (
    int port,
    std::map< std::string, std::string > database,
    Logger * l1 )
```

Метод для установки соединения.

Аргументы

|          |   |
|----------|---|
| port     | Порт для соединения.  |
| database | Карта с данными базы.                                       |
| l1       | Указатель на объект <a href="#">Logger</a> для логирования. |

Возвращает

Возвращает статус соединения (0 - успех, другой код - ошибка).

### 4.2.2.2 generate\_salt()

```
std::string Communicate::generate_salt ( ) [static]
```

Генерация соли для хеширования.

Возвращает

Возвращает сгенерированную соль в виде строки.

### 4.2.2.3 sha1()

```
std::string Communicate::sha1 (
    std::string input_str ) [static]
```

Генерация SHA1 хеша из строки.

Аргументы

|           |                                 |
|-----------|---------------------------------|
| input_str | Входная строка для хеширования. |
|-----------|---------------------------------|

Возвращает

Возвращает хеш в виде строки.

Объявления и описания членов классов находятся в файлах:

- [Communicate.h](#)
- [Communicate.cpp](#)

## 4.3 Класс Connector

Класс для подключения к базе данных.

```
#include <Connector.h>
```

Открытые члены

- `int connect (std::string base_file="test_files/auth.txt")`  
Метод для подключения к базе данных.
- `std::map< std::string, std::string > get\_data ()`  
Метод для получения данных из базы данных.

Закрытые данные

- `std::map< std::string, std::string > data_base`

### 4.3.1 Подробное описание

Класс для подключения к базе данных.

Этот класс управляет подключением к файлу базы данных и предоставляет доступ к данным.

### 4.3.2 Методы

#### 4.3.2.1 `connect()`

```
int Connector::connect (  
    std::string base_file = "test_files/auth.txt" )
```

Метод для подключения к базе данных.

## Аргументы

|           |   |
|-----------|---|
| base_file | Имя файла базы данных (по умолчанию "test_files/auth.txt"). |
|-----------|---|

## Возвращает

Возвращает статус подключения (0 - успех, другой код - ошибка).

## 4.3.2.2 get\_data()

```
std::map< std::string, std::string > Connector::get_data ( )
```

Метод для получения данных из базы данных.

## Возвращает

Возвращает карту с данными базы.

Объявления и описания членов классов находятся в файлах:

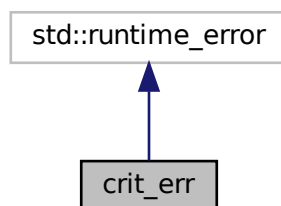
- [Connector.h](#)
- [Connector.cpp](#)

## 4.4 Класс crit\_err

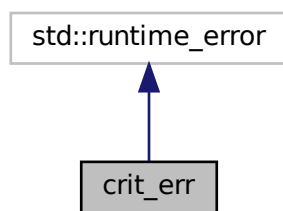
Класс для критических ошибок. Этот класс используется для обработки критических ошибок в программе.

```
#include <Errors.h>
```

Граф наследования:crit\_err:



Граф связей класса `crit_err`:



### Открытые члены

- `crit_err` (`const std::string &s`)  
Конструктор класса `crit_err`.

#### 4.4.1 Подробное описание

Класс для критических ошибок. Этот класс используется для обработки критических ошибок в программе.

#### 4.4.2 Конструктор(ы)

##### 4.4.2.1 `crit_err()`

```
crit_err::crit_err (  
    const std::string & s ) [inline]
```

Конструктор класса `crit_err`.

Аргументы

|   |                      |
|---|----------------------|
| s | Сообщение об ошибке. |
|---|----------------------|

Объявления и описания членов класса находятся в файле:

- `Errors.h`

### 4.5 Класс Interface

Класс интерфейса пользователя.



```
#include <Interface.h>
```

### Открытые члены

- Interface ()  
Конструктор класса [Interface](#).
- int [comm\\_proc](#) (int argc, const char \*\*argv)  
Метод обработки командной строки.

### Закрытые данные

- int PORT

#### 4.5.1 Подробное описание

Класс интерфейса пользователя.

Этот класс отвечает за обработку команд и взаимодействие с пользователем.

#### 4.5.2 Методы

##### 4.5.2.1 comm\_proc()

```
int Interface::comm_proc (  
    int argc,  
    const char ** argv )
```

Метод обработки командной строки.

Аргументы

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

Возвращает

Возвращает статус выполнения (0 - успех, другой код - ошибка).

Объявления и описания членов классов находятся в файлах:

- [Interface.h](#)
- Interface.cpp

## 4.6 Класс Logger

Класс для ведения логов.

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

### Открытые члены

- `int writelog (std::string s)`  
Метод записи сообщения в лог-файл.
- `int set_path (std::string path_file)`  
Метод установки пути к лог-файлу.
- `Logger ()`  
Конструктор по умолчанию класса `Logger`.
- `Logger (std::string s)`  
Конструктор класса `Logger` с указанием пути к файлу лога.

### Закрытые статические члены

- `static std::string getCurrentDateTime (std::string s)`

### Закрытые данные

- `std::string path_to_logfile`

#### 4.6.1 Подробное описание

Класс для ведения логов.

Этот класс управляет записью логов в файл и предоставляет методы для работы с ними.

#### 4.6.2 Конструктор(ы)

##### 4.6.2.1 Logger()

```
Logger::Logger (  
    std::string s )
```

Конструктор класса `Logger` с указанием пути к файлу лога.

Аргументы

|   |                    |
|---|--------------------|
| s | Путь к файлу лога. |
|---|--------------------|

### 4.6.3 Методы

#### 4.6.3.1 set\_path()

```
int Logger::set_path (
    std::string path_file )
```

Метод установки пути к лог-файлу.

Аргументы

|           |                    |
|-----------|--------------------|
| path_file | Путь к файлу лога. |
|-----------|--------------------|

Возвращает

Возвращает статус установки пути (0 - успех, другой код - ошибка).

#### 4.6.3.2 writelog()

```
int Logger::writelog (
    std::string s )
```

Метод записи сообщения в лог-файл.

Аргументы

|   |  |
|---|--|
| s | Сообщение, которое нужно записать в лог. |
|---|--|

Возвращает

Возвращает статус записи (0 - успех, другой код - ошибка).

Объявления и описания членов классов находятся в файлах:

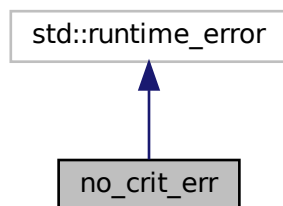
- [Logger.h](#)
- [Logger.cpp](#)

## 4.7 Класс no\_crit\_err

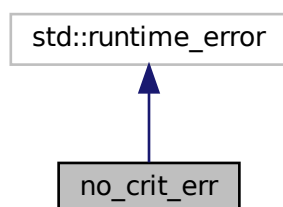
Класс для некритических ошибок.

```
#include <Errors.h>
```

Граф наследования: `no_crit_err`:



Граф связей класса `no_crit_err`:



## Открытые члены

- `no_crit_err` (`const std::string s`)  
Конструктор класса `no_crit_err`.

### 4.7.1 Подробное описание

Класс для некритических ошибок.

Этот класс используется для обработки некритических ошибок в программе.

### 4.7.2 Конструктор(ы)

#### 4.7.2.1 `no_crit_err()`

```
no_crit_err::no_crit_err (  
    const std::string s ) [inline]
```

Конструктор класса `no_crit_err`.

Аргументы

|   |                      |
|---|----------------------|
| s | Сообщение об ошибке. |
|---|----------------------|

Объявления и описания членов класса находятся в файле:

- [Errors.h](#)



## Глава 5

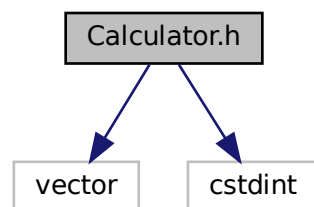
# Файлы

### 5.1 Файл Calculator.h

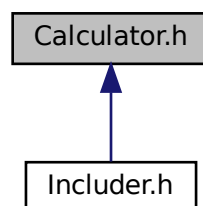
Заголовочный файл для класса [Calculator](#).

```
#include <vector>
#include <cstdint>
```

Граф включаемых заголовочных файлов для Calculator.h:



Граф файлов, в которые включается этот файл:



## Классы

- class [Calculator](#)

Класс для выполнения расчетов.

### 5.1.1 Подробное описание

Заголовочный файл для класса [Calculator](#).

Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот класс предназначен для выполнения расчетов на основе входных данных.

## 5.2 Calculator.h

[См. документацию.](#)

```
1
11 #pragma once
12 #include <vector>
13 #include <stdint>
14
20 class Calculator {
21     double results = 0;
22
23 public:
27     Calculator(std::vector<double> input_data);
28
32     double send_res();
33 };
```



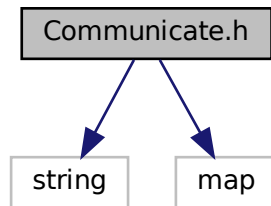
## 5.3 Файл Communicate.h

Заголовочный файл для класса `Communicate`.

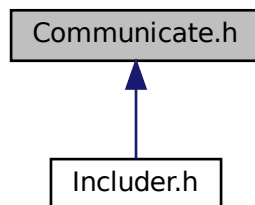
```
#include <string>
```

```
#include <map>
```

Граф включаемых заголовочных файлов для Communicate.h:



Граф файлов, в которые включается этот файл:



### Классы

- class `Communicate`

Класс для управления коммуникациями.

#### 5.3.1 Подробное описание

Заголовочный файл для класса `Communicate`.

Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот класс отвечает за установление соединения и обработку коммуникаций.

## 5.4 Communicate.h

[См. документацию.](#)

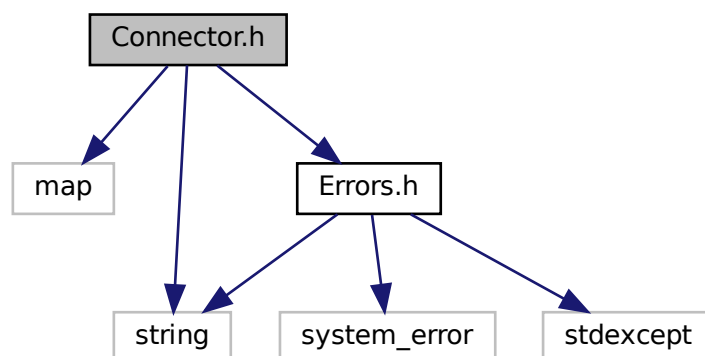
```
1
11 #pragma once
12 #include <string>
13 #include <map>
14
15 class Logger;
16
22 class Communicate {
23 public:
30     int connection(int port, std::map<std::string, std::string> database, Logger* l1);
31
36     static std::string sha1(std::string input_str);
37
41     static std::string generate_salt();
42 };
```

## 5.5 Файл Connector.h

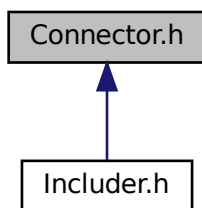
Заголовочный файл для класса [Connector](#).

```
#include <map>
#include <string>
#include "Errors.h"
```

Граф включаемых заголовочных файлов для Connector.h:



Граф файлов, в которые включается этот файл:



## Классы

- class [Connector](#)

Класс для подключения к базе данных.

### 5.5.1 Подробное описание

Заголовочный файл для класса [Connector](#).

Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот класс предназначен для подключения к базе данных и получения данных.

## 5.6 Connector.h

См. документацию.

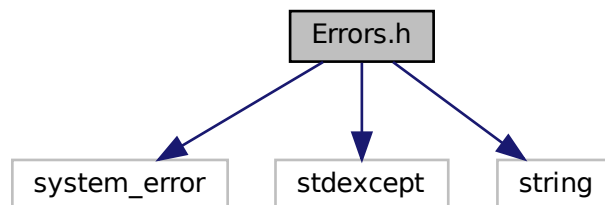
```
1
11 #pragma once
12 #include <map>
13 #include <string>
14 #include "Errors.h"
15
21 class Connector {
22 private:
23     std::map<std::string, std::string> data_base;
24
25 public:
30     int connect(std::string base_file = "test_files/auth.txt");
31
35     std::map<std::string, std::string> get_data();
36 };
```

## 5.7 Файл Errors.h

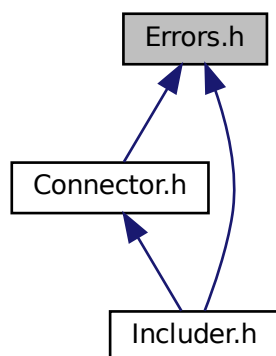
Заголовочный файл для классов ошибок

```
#include <system_error>
#include <stdexcept>
#include <string>
```

Граф включаемых заголовочных файлов для Errors.h:



Граф файлов, в которые включается этот файл:



## Классы

- class `crit_err`  
Класс для критических ошибок. Этот класс используется для обработки критических ошибок в программе.
- class `no_crit_err`  
Класс для некритических ошибок.

### 5.7.1 Подробное описание

Заголовочный файл для классов ошибок

Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот файл содержит определения классов для обработки критических и некритических ошибок.

## 5.8 Errors.h

См. документацию.

```

1
11 #pragma once
12 #include <system_error>
13 #include <stdexcept>
14 #include <string>
15
20 class crit_err: public std::runtime_error {
21 public:
25     crit_err(const std::string& s): std::runtime_error(s) {}
26 };
27
33 class no_crit_err: public std::runtime_error {
34 public:
38     no_crit_err(const std::string s): std::runtime_error(s) {}
39 };
40

```

## 5.9 Includer.h

```

1 #include "Connector.h"
2 #include "Communicate.h"
3 #include "Calculator.h"
4 #include "Interface.h"
5 #include "Logger.h"
6 #include "Errors.h"

```

## 5.10 Файл Interface.h

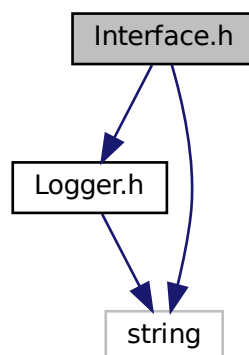
Заголовочный файл для класса [Interface](#).

```

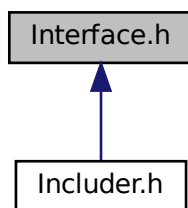
#include "Logger.h"
#include <string>

```

Граф включаемых заголовочных файлов для Interface.h:



Граф файлов, в которые включается этот файл:



## Классы

- class [Interface](#)

Класс интерфейса пользователя.

### 5.10.1 Подробное описание

Заголовочный файл для класса [Interface](#).

Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот класс управляет взаимодействием с пользователем через интерфейс командной строки.

## 5.11 Interface.h

[См. документацию.](#)

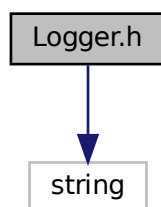
```
1
11 #pragma once
12 #include "Logger.h"
13 #include <string>
14
20 class Interface {
21     int PORT;
22
23 public:
26     Interface() {}
27
33     int comm_proc(int argc, const char** argv);
34 };
```

## 5.12 Файл Logger.h

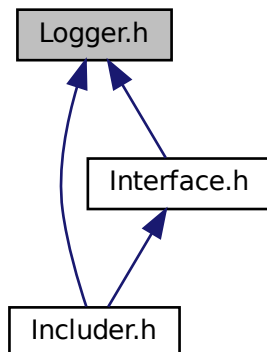
Заголовочный файл для класса [Logger](#).

```
#include <string>
```

Граф включаемых заголовочных файлов для Logger.h:



Граф файлов, в которые включается этот файл:



### Классы

- class [Logger](#)

Класс для ведения логов.

#### 5.12.1 Подробное описание

Заголовочный файл для класса [Logger](#).



Автор

Рыбаков Е.М.

Версия

1.0

Дата

23.12.2024

Авторство

ИБСТ ПГУ

Этот класс отвечает за ведение логов событий в программе.

## 5.13 Logger.h

[См. документацию.](#)

```
1
11 #pragma once
12 #include <string>
13
19 class Logger {
20     static std::string getCurrentDateTime(std::string s);
21
22     std::string path_to_logfile;
23
24 public:
29     int writelog(std::string s);
30
35     int set_path(std::string path_file);
36
39     Logger();
40
44     Logger(std::string s);
45 };
```



# Предметный указатель

- Calculator, [7](#)
  - Calculator, [7](#)
  - send\_res, [8](#)
- Calculator.h, [19](#)
- comm\_proc
  - Interface, [13](#)
- Communicate, [8](#)
  - connection, [9](#)
  - generate\_salt, [9](#)
  - sha1, [9](#)
- Communicate.h, [21](#)
- connect
  - Connector, [10](#)
- connection
  - Communicate, [9](#)
- Connector, [10](#)
  - connect, [10](#)
  - get\_data, [11](#)
- Connector.h, [22](#)
- crit\_err, [11](#)
  - crit\_err, [12](#)
- Errors.h, [24](#)
- generate\_salt
  - Communicate, [9](#)
- get\_data
  - Connector, [11](#)
- Interface, [12](#)
  - comm\_proc, [13](#)
- Interface.h, [26](#)
- Logger, [14](#)
  - Logger, [14](#)
  - set\_path, [15](#)
  - writelog, [15](#)
- Logger.h, [28](#)
- no\_crit\_err, [15](#)
  - no\_crit\_err, [16](#)
- send\_res
  - Calculator, [8](#)
- set\_path
  - Logger, [15](#)
- sha1
  - Communicate, [9](#)
- writelog
  - Logger, [15](#)