

Concordo

1

Gerado por Doxygen 1.9.7



<b>1 README</b>	<b>1</b>
<b>2 Índice Hierárquico</b>	<b>3</b>
2.1 Hierarquia de Classes	3
<b>3 Índice dos Componentes</b>	<b>5</b>
3.1 Lista de Classes	5
<b>4 Índice dos Arquivos</b>	<b>7</b>
4.1 Lista de Arquivos	7
<b>5 Classes</b>	<b>9</b>
5.1 Referência da Classe Channel	9
5.1.1 Descrição detalhada	9
5.1.2 Construtores e Destrutores	10
5.1.2.1 Channel()	10
5.2 Referência da Classe Message	10
5.2.1 Descrição detalhada	11
5.2.2 Construtores e Destrutores	11
5.2.2.1 Message()	11
5.2.3 Documentação das funções	11
5.2.3.1 getContent()	11
5.2.3.2 getSentAt()	11
5.2.3.3 getSentBy()	11
5.2.3.4 operator=()	12
5.2.3.5 operator==()	12
5.3 Referência da Classe Server	12
5.3.1 Descrição detalhada	14
5.3.2 Construtores e Destrutores	14
5.3.2.1 Server()	14
5.3.3 Documentação das funções	14
5.3.3.1 add_TextChannel()	14
5.3.3.2 add_VoiceChannel()	15
5.3.3.3 addMember()	15
5.3.3.4 create_channel()	15
5.3.3.5 find_channel()	15
5.3.3.6 getDescription()	16
5.3.3.7 getInviteCode()	16
5.3.3.8 getInviteCodeEnabled()	16
5.3.3.9 getMembers()	17
5.3.3.10 getName()	17
5.3.3.11 getOwner()	17
5.3.3.12 getTextChannels()	17
5.3.3.13 getVoiceChannels()	18

5.3.3.14 list_channels()	18
5.3.3.15 listMembers()	18
5.3.3.16 memberExists()	18
5.3.3.17 operator==( )	19
5.3.3.18 removeInviteCode()	19
5.3.3.19 removeMember()	19
5.3.3.20 server_clear()	19
5.3.3.21 setDescription()	19
5.3.3.22 setInviteCode()	20
5.3.3.23 setOwner()	20
5.4 Referência da Classe System	20
5.4.1 Descrição detalhada	22
5.4.2 Documentação das funções	22
5.4.2.1 carregar()	22
5.4.2.2 carregarServidores()	22
5.4.2.3 carregarUsuarios()	22
5.4.2.4 change_server_description()	22
5.4.2.5 create_server()	23
5.4.2.6 create_user()	23
5.4.2.7 disconnect()	23
5.4.2.8 enter_channel()	23
5.4.2.9 enter_server()	23
5.4.2.10 executable()	24
5.4.2.11 get_current_channel()	24
5.4.2.12 get_current_server()	24
5.4.2.13 increase_l_id()	24
5.4.2.14 join_server()	24
5.4.2.15 leave_server()	25
5.4.2.16 list_servers()	25
5.4.2.17 logged_email()	25
5.4.2.18 logged_user_name()	25
5.4.2.19 login()	25
5.4.2.20 remove_server()	26
5.4.2.21 salvar()	26
5.4.2.22 salvarServidores()	26
5.4.2.23 salvarUsuarios()	26
5.4.2.24 send_message()	26
5.4.2.25 set_server_invite_code()	27
5.5 Referência da Classe TextChannel	27
5.5.1 Descrição detalhada	28
5.5.2 Construtores e Destrutores	28
5.5.2.1 TextChannel() [1/2]	28

5.5.2.2 TextChannel() [2/2]	28
5.5.3 Documentação das funções	29
5.5.3.1 add_message()	29
5.5.3.2 getMessages()	29
5.5.3.3 list_messages()	29
5.5.3.4 send()	29
5.6 Referência da Classe User	30
5.6.1 Descrição detalhada	31
5.6.2 Construtores e Destrutores	31
5.6.2.1 User()	31
5.6.3 Documentação das funções	31
5.6.3.1 getEmail()	31
5.6.3.2 getId()	31
5.6.3.3 getName()	32
5.6.3.4 getPassword()	32
5.6.3.5 operator==()	32
5.6.3.6 setId()	32
5.6.3.7 updateEmail()	33
5.6.3.8 updateName()	33
5.6.3.9 updatePassword()	33
5.7 Referência da Classe VoiceChannel	34
5.7.1 Descrição detalhada	35
5.7.2 Construtores e Destrutores	35
5.7.2.1 VoiceChannel()	35
5.7.3 Documentação das funções	35
5.7.3.1 add_message()	35
5.7.3.2 getLastMessage()	35
5.7.3.3 list_messages()	35
5.7.3.4 send()	36
<b>6 Arquivos</b>	<b>37</b>
6.1 Referência do Arquivo include/Channel.h	37
6.1.1 Descrição detalhada	37
6.2 Channel.h	37
6.3 Referência do Arquivo include/Message.h	38
6.3.1 Descrição detalhada	38
6.4 Message.h	38
6.5 Referência do Arquivo include/Server.h	38
6.5.1 Descrição detalhada	39
6.6 Server.h	39
6.7 Referência do Arquivo include/System.h	40
6.7.1 Descrição detalhada	40

---

6.8 System.h . . . . .	40
6.9 Referência do Arquivo include/TextChannel.h . . . . .	41
6.9.1 Descrição detalhada . . . . .	41
6.10 TextChannel.h . . . . .	41
6.11 Referência do Arquivo include/User.h . . . . .	41
6.11.1 Descrição detalhada . . . . .	42
6.12 User.h . . . . .	42
6.13 Referência do Arquivo include/VoiceChannel.h . . . . .	42
6.13.1 Descrição detalhada . . . . .	43
6.14 VoiceChannel.h . . . . .	43
6.15 Referência do Arquivo source/Channel.cpp . . . . .	43
6.15.1 Descrição detalhada . . . . .	43
6.16 Referência do Arquivo source/main.cpp . . . . .	43
6.16.1 Descrição detalhada . . . . .	44
6.17 Referência do Arquivo source/Message.cpp . . . . .	44
6.17.1 Descrição detalhada . . . . .	44
6.18 Referência do Arquivo source/Server.cpp . . . . .	44
6.18.1 Descrição detalhada . . . . .	44
6.19 Referência do Arquivo source/System.cpp . . . . .	44
6.19.1 Descrição detalhada . . . . .	45
6.20 Referência do Arquivo source/TextChannel.cpp . . . . .	45
6.20.1 Descrição detalhada . . . . .	45
6.21 Referência do Arquivo source/User.cpp . . . . .	45
6.21.1 Descrição detalhada . . . . .	45
6.22 Referência do Arquivo source/VoiceChannel.cpp . . . . .	45
6.22.1 Descrição detalhada . . . . .	45
<b>Índice Remissivo</b>	<b>47</b>

# Capítulo 1

## README

**Esse é o projeto Concorde da disciplina Linguagem de Programação 1 ministrada pelo professor Sidemar na UFRN para o curso de BTI.**

*Autor: Artur Revorêdo Pinto*

Horário da turma: *M56*

**Para compilar o código basta rodar os seguintes comandos**

- `cmake -B build`
- `cmake --build build`

**Para rodar o programa basta utilizar o comando**

- `./build/concorde`

Nesse projeto existem alguns comandos possíveis, caso você não esteja logado os comandos são:

- `create-user email senha nome`  
Nesse comando email nem senha podem conter espaços, o nome no entanto pode.
- `login email senha`  
Entra com um usuário no sistema.
- `quit`  
Sai do programa.

Caso o login seja efetuado então outros comandos ficarão disponíveis para o usuário, esses são.

- `disconnect`  
Desconecta o usuário atual do sistema.
- `create-server nome`  
Em que nome é o nome do servidor.

- `set-server-desc nome descrição`

Nesse o nome é o nome do servidor e descrição é a descrição que você deseja inserir no servidor.

- `set-server-invite-code nome invite-code`

Nesse o nome é o nome do servidor o qual o invite code será mudado e invite code é um código sem espaços para ser inserido no servidor

- `list-servers`

Apenas lista os servidores existentes

- `remove-server`

Deleta um servidor do sistema.

- `join-server nome invite code`

Nesse o usuário tentará ser adicionado no servidor de nome "nome" e digitará o código de convite.

- `enter-server nome`

Nesse o usuário entra em um servidor do qual já é membro.

- `leave-server`

Nesse o usuário sai do servidor, não deixando de ser um participante mas apenas para navegar por outros servidores.

- `create-channel nome`

Nesse o usuário cria um canal no servidor em que está no momento com o nome dado no comando.

- `enter-channel nome`

Nesse o usuário entrará no canal de nome informado no comando caso exista no servidor atual.

- `leave-channel`

Nesse o usuário sairá do canal atual.

- `send-message mensagem`

Nesse o usuário envia uma mensagem nova no canal atual se estiver em um.

- `list-messages`

Nesse o usuário lista as mensagens do canal atual.



## Capítulo 2

# Índice Hierárquico

### 2.1 Hierarquia de Classes

Esta lista de hierarquias está parcialmente ordenada (ordem alfabética):

Channel . . . . .	9
TextChannel . . . . .	27
VoiceChannel . . . . .	34
Message . . . . .	10
Server . . . . .	12
System . . . . .	20
User . . . . .	30



## Capítulo 3

# Índice dos Componentes

### 3.1 Lista de Classes

Aqui estão as classes, estruturas, uniões e interfaces e suas respectivas descrições:

<a href="#">Channel</a>	Classe que representa um canal . . . . .	9
<a href="#">Message</a>	Classe que representa uma mensagem . . . . .	10
<a href="#">Server</a>	Classe que representa um servidor . . . . .	12
<a href="#">System</a>	Classe que representa o sistema . . . . .	20
<a href="#">TextChannel</a>	Classe que representa um canal de texto . . . . .	27
<a href="#">User</a>	Classe que representa um usuário . . . . .	30
<a href="#">VoiceChannel</a>	Classe que representa um canal de voz . . . . .	34



## Capítulo 4

# Índice dos Arquivos

### 4.1 Lista de Arquivos

Esta é a lista de todos os arquivos documentados e suas respectivas descrições:

include/Channel.h	
Definição da classe <a href="#">Channel</a>	37
include/Message.h	
Definição da classe <a href="#">Message</a>	38
include/Server.h	
Definição da classe <a href="#">Server</a>	38
include/System.h	
Definição da classe <a href="#">System</a>	40
include/TextChannel.h	
Classe que representa um canal de texto	41
include/User.h	
Definição da classe <a href="#">User</a>	41
include/VoiceChannel.h	
Classe que representa um canal de voz	42
source/Channel.cpp	
Implementação da classe <a href="#">Channel</a>	43
source/main.cpp	
Execução do programa	43
source/Message.cpp	
Implementação da classe <a href="#">Message</a>	44
source/Server.cpp	
Implementação da classe <a href="#">Server</a>	44
source/System.cpp	
Implementação dos métodos da classe <a href="#">System</a>	44
source/TextChannel.cpp	
Implementação da classe <a href="#">TextChannel</a>	45
source/User.cpp	
Implementação da classe <a href="#">User</a>	45
source/VoiceChannel.cpp	
Implementação da classe <a href="#">VoiceChannel</a>	45



# Capítulo 5

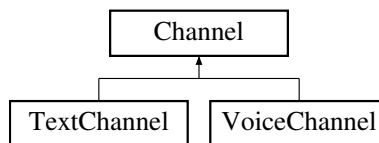
## Classes

### 5.1 Referência da Classe Channel

Classe que representa um canal.

```
#include <Channel.h>
```

Diagrama de hierarquia da classe Channel:



#### Membros Públicos

- **Channel** (std::string **name**)  
*Construtor da classe Channel.*
- std::string **getName** ()  
*Método que retorna o nome do canal.*
- void **send\_message** (std::string message, int logged\_user\_id)  
*Método que envia uma mensagem no canal.*
- void **list\_messages** ()  
*Método que lista as mensagens do canal.*

#### Atributos Protegidos

- std::string **name**  
*Variável que guarda o nome do canal.*

#### 5.1.1 Descrição detalhada

Classe que representa um canal.

## 5.1.2 Construtores e Destrutores

### 5.1.2.1 Channel()

```
Channel::Channel (
    std::string name )
```

Construtor da classe [Channel](#).

Construtor de [Channel](#).

Parâmetros

<i>name</i>	
-------------	--

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- include/[Channel.h](#)
- source/[Channel.cpp](#)

## 5.2 Referência da Classe Message

Classe que representa uma mensagem.

```
#include <Message.h>
```

### Membros Públicos

- [Message](#) ()  
*Construtor de [Message](#).*
- **Message** (std::string sentAt, int sentBy, std::string content)
- std::string [getSentAt](#) ()  
*Retorna a data de envio da mensagem.*
- int [getSentBy](#) ()  
*Retorna o id de quem enviou a mensagem.*
- std::string [getContent](#) ()  
*Retorna o conteúdo da mensagem.*
- void **setSentAt** (std::string sentAt)
- void **setSentBy** (int sentBy)
- void **setContent** (std::string content)
- bool [operator==](#) (const [Message](#) &other)  
*Sobrecarga do operador de comparação.*
- [Message](#) [operator=](#) (const [Message](#) &other)  
*Sobrecarga do operador de atribuição.*

### Atributos Privados

- std::string **sentAt**
- int **sentBy**
- std::string **content**



### 5.2.1 Descrição detalhada

Classe que representa uma mensagem.

### 5.2.2 Construtores e Destrutores

#### 5.2.2.1 Message()

```
Message::Message ( )
```

Construtor de [Message](#).

Parâmetros

<i>sentAt</i>	
<i>sentBy</i>	
<i>content</i>	

### 5.2.3 Documentação das funções

#### 5.2.3.1 getContent()

```
std::string Message::getContent ( )
```

Retorna o conteúdo da mensagem.

Retorna

std::string

#### 5.2.3.2 getSentAt()

```
std::string Message::getSentAt ( )
```

Retorna a data de envio da mensagem.

Retorna

std::string

#### 5.2.3.3 getSentBy()

```
int Message::getSentBy ( )
```

Retorna o id de quem enviou a mensagem.

Retorna

int

#### 5.2.3.4 operator=()

```
Message Message::operator= (
    const Message & other )
```

Sobrecarga do operador de atribuição.

Parâmetros

<i>sentBy</i>	
---------------	--

#### 5.2.3.5 operator==()

```
bool Message::operator== (
    const Message & other )
```

Sobrecarga do operador de comparação.

Parâmetros

<i>sent</i> ↔	
<i>At</i>	

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- include/[Message.h](#)
- source/[Message.cpp](#)

## 5.3 Referência da Classe Server

Classe que representa um servidor.

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

### Membros Públicos

- **Server** ()  
*Construtor da classe [Server](#).*
- **Server** (std::string [name](#))  
*Construtor da classe [Server](#).*
- **Server** (int owner, std::string [name](#), std::string [description](#), std::string [invite\\_code](#))  
*Construtor da classe [Server](#).*
- void [server\\_clear](#) ()  
*Método que limpa o servidor.*
- int [getOwner](#) ()  
*Método que retorna o id do dono do servidor.*
- std::string [getName](#) ()

- Método que retorna o nome do servidor.*
- `std::string getDescription ()`
- Método que retorna a descrição do servidor.*
- `std::string getInviteCode ()`
- Método que retorna o código de convite do servidor.*
- `std::vector< int > getMembers ()`
- Método que retorna os ids dos membros do servidor.*
- `std::vector< VoiceChannel > getVoiceChannels ()`
- Método que retorna os canais do servidor.*
- `std::vector< TextChannel > getTextChannels ()`
- Método que retorna os canais do servidor.*
- `bool getInviteCodeEnabled ()`
- Método que retorna se o código de convite está habilitado.*
- `void setOwner (int id)`
- Método que altera o id do dono do servidor.*
- `void setDescription (std::string description)`
- Método que altera a descrição do servidor.*
- `void setInviteCode (std::string invite_code)`
- Método que altera o código de convite do servidor.*
- `void removeInviteCode ()`
- Método que remove o código de convite do servidor.*
- `void addMember (int id)`
- Método que adiciona um membro ao servidor.*
- `void removeMember (int id)`
- Método que remove um membro do servidor.*
- `void listMembers ()`
- Método que lista os membros do servidor.*
- `bool memberExists (int id)`
- Método que verifica se um membro existe no servidor.*
- `void create_channel (std::string name, std::string type)`
- Método que adiciona um canal ao servidor.*
- `void list_channels ()`
- Método que lista os canais do servidor.*
- `bool find_channel (std::string name)`
- Método que encontra um canal do servidor.*
- `void add_TextChannel (TextChannel channel)`
- Método que adiciona um canal de texto ao servidor.*
- `void add_VoiceChannel (VoiceChannel channel)`
- Método que adiciona um canal de voz ao servidor.*
- `bool operator== (Server server)`

### Atributos Protegidos

- `int owner_id`  
*Variável que guarda o id do dono do servidor.*
- `bool invite_code_enabled = false`  
*Variável que guarda se o código de convite está habilitado.*
- `std::string name = ""`  
*Variável que guarda o nome do servidor.*
- `std::string description = ""`

*Variável que guarda a descrição do servidor.*

- `std::string invite_code = ""`

*Variável que guarda o código de convite do servidor.*

- `std::vector< int > members`

*Vetor que guarda os ids dos membros do servidor.*

- `std::vector< VoiceChannel > voice_channels`

*Vetor que guarda os canais do servidor.*

- `std::vector< TextChannel > text_channels`

*Vetor que guarda os canais do servidor.*

### 5.3.1 Descrição detalhada

Classe que representa um servidor.

### 5.3.2 Construtores e Destrutores

#### 5.3.2.1 Server()

```
Server::Server (
    std::string name )
```

Construtor da classe `Server`.

Construtor de `Server`.

Parâmetros

<code>name</code>	nome do server
-------------------	----------------

### 5.3.3 Documentação das funções

#### 5.3.3.1 add\_TextChannel()

```
void Server::add_TextChannel (
    TextChannel channel )
```

Método que adiciona um canal de texto ao servidor.

Adiciona um channel ao servidor.

Parâmetros

<code>channel</code>	
----------------------	--

### 5.3.3.2 add\_VoiceChannel()

```
void Server::add_VoiceChannel (
    VoiceChannel channel )
```

Método que adiciona um canal de voz ao servidor.

Adiciona um channel ao servidor.

#### Parâmetros

<i>channel</i>	
----------------	--

### 5.3.3.3 addMember()

```
void Server::addMember (
    int id )
```

Método que adiciona um membro ao servidor.

Adiciona um membro ao servidor.

#### Parâmetros

<i>id</i>	
-----------	--

### 5.3.3.4 create\_channel()

```
void Server::create_channel (
    std::string name,
    std::string type )
```

Método que adiciona um canal ao servidor.

Adiciona um channel ao servidor.

#### Parâmetros

<i>channel</i>	
----------------	--

### 5.3.3.5 find\_channel()

```
bool Server::find_channel (
    std::string name )
```

Método que encontra um canal do servidor.

Verifica se um channel existe no servidor.

**Parâmetros**

<i>name</i>	
-------------	--

**Retorna**

Channel\*

**5.3.3.6 getDescription()**

```
std::string Server::getDescription ( )
```

Método que retorna a descrição do servidor.

Retorna a descrição do servidor.

**Retorna**

std::string

**5.3.3.7 getInviteCode()**

```
std::string Server::getInviteCode ( )
```

Método que retorna o código de convite do servidor.

Retorna o código de convite do servidor.

**Retorna**

std::string

**5.3.3.8 getInviteCodeEnabled()**

```
bool Server::getInviteCodeEnabled ( )
```

Método que retorna se o código de convite está habilitado.

Retorna se o código de convite está habilitado.

**Retorna**

true

false

#### 5.3.3.9 getMembers()

```
std::vector< int > Server::getMembers ( )
```

Método que retorna os ids dos membros do servidor.

Retorna os membros do servidor.

Retorna

```
std::vector<int>
```

#### 5.3.3.10 getName()

```
std::string Server::getName ( )
```

Método que retorna o nome do servidor.

Retorna o nome do servidor.

Retorna

```
std::string
```

#### 5.3.3.11 getOwner()

```
int Server::getOwner ( )
```

Método que retorna o id do dono do servidor.

Retorna o id do dono do servidor.

Retorna

```
int
```

#### 5.3.3.12 getTextChannels()

```
std::vector< TextChannel > Server::getTextChannels ( )
```

Método que retorna os canais do servidor.

Retorna os text channels do servidor.

Retorna

```
std::vector<TextChannel *>
```

#### 5.3.3.13 getVoiceChannels()

```
std::vector< VoiceChannel > Server::getVoiceChannels ( )
```

Método que retorna os canais do servidor.

Retorna voice channels do servidor.

Retorna

```
std::vector<VoiceChannel *>
```

#### 5.3.3.14 list\_channels()

```
void Server::list_channels ( )
```

Método que lista os canais do servidor.

Lista os channels do servidor.

#### 5.3.3.15 listMembers()

```
void Server::listMembers ( )
```

Método que lista os membros do servidor.

Lista os membros do servidor.

#### 5.3.3.16 memberExists()

```
bool Server::memberExists (
    int id )
```

Método que verifica se um membro existe no servidor.

Verifica se um membro existe no servidor.

Parâmetros

<i>id</i>	
-----------	--

Retorna

true

false



#### 5.3.3.17 operator==()

```
bool Server::operator== (
    Server server ) [inline]
```

##### Parâmetros

<i>server</i>	Operador de igualdade.
---------------	------------------------

#### 5.3.3.18 removeInviteCode()

```
void Server::removeInviteCode ( )
```

Método que remove o código de convite do servidor.

Remove o código de convite do servidor.

#### 5.3.3.19 removeMember()

```
void Server::removeMember (
    int id )
```

Método que remove um membro do servidor.

Remove um membro do servidor.

##### Parâmetros

<i>id</i>	
-----------	--

#### 5.3.3.20 server\_clear()

```
void Server::server_clear ( )
```

Método que limpa o servidor.

Limpa o servidor de channels e members.

#### 5.3.3.21 setDescription()

```
void Server::setDescription (
    std::string description )
```

Método que altera a descrição do servidor.

Seta a descrição do servidor.

**Parâmetros**

<i>description</i>	
--------------------	--

**5.3.3.22 setInviteCode()**

```
void Server::setInviteCode (
    std::string invite_code )
```

Método que altera o código de convite do servidor.

Seta o código de convite do servidor.

**Parâmetros**

<i>invite_code</i>	
--------------------	--

**5.3.3.23 setOwner()**

```
void Server::setOwner (
    int id )
```

Método que altera o id do dono do servidor.

Seta o id do dono do servidor.

**Parâmetros**

<i>id</i>	
-----------	--

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

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

## 5.4 Referência da Classe System

Classe que representa o sistema.

```
#include <System.h>
```

**Membros Públicos**

- [Channel \\* get\\_current\\_channel \(\)](#)  
*Retorna o ponteiro para o canal atual.*

- `Server * get_current_server ()`  
*Retorna o ponteiro para o servidor atual.*
- `std::string logged_user_name ()`  
*Retorna o nome do usuário logado.*
- `std::string logged_email ()`  
*Retorna o email do usuário logado.*
- `int increase_l_id ()`  
*Aumenta o id do próximo usuário a ser criado.*
- `void executable ()`  
*Método que executa o sistema.*
- `void create_user (std::string email, std::string password, std::string name)`  
*Método que cria um usuário.*
- `void login (std::string email, std::string password)`  
*Método que faz o login de um usuário.*
- `void disconnect ()`  
*Método que faz o logout de um usuário.*
- `void create_server (std::string name)`  
*Método que cria um servidor.*
- `void change_server_description (std::string name, std::string description)`  
*Método que altera a descrição de um servidor.*
- `void set_server_invite_code (std::string name, std::string invite_code)`  
*Método que altera o código de convite de um servidor.*
- `void list_servers ()`  
*Método que lista os servidores.*
- `void remove_server (std::string name)`  
*Método que remove um servidor.*
- `void join_server (std::string name, std::string invite_code)`  
*Método que adiciona o usuário em um servidor.*
- `void enter_server (std::string name)`  
*Método que entra em um servidor.*
- `void leave_server ()`  
*Método que sai de um servidor.*
- `void enter_channel (std::string name)`  
*Método que entra em um canal.*
- `void send_message ()`  
*Método que envia uma mens:gem.*
- `void salvarUsuarios ()`  
*Método que salva os usuários em um arquivo.*
- `void salvarServidores ()`  
*Método que salva os servidores em um arquivo.*
- `void salvar ()`  
*Método que salva os usuários e os servidores em arquivos.*
- `void carregarUsuarios ()`  
*Método que carrega os usuários de um arquivo.*
- `void carregarServidores ()`  
*Método que carrega os servidores de um arquivo.*
- `void carregar ()`  
*Método que carrega os usuários e os servidores de arquivos.*

### Atributos Privados

- `int l_id_assigned = 1`  
*Variável que guarda o id do próximo usuário a ser criado.*
- `int logged_user = -1`  
*Variável que guarda o id do usuário logado.*
- `std::vector< User > users`  
*Vetor que guarda todos os usuários do sistema.*
- `std::vector< Server > servers`  
*Vetor que guarda todos os servidores do sistema.*
- `Server current_server = Server()`  
*Ponteiro para o servidor atual.*
- `Server * update_server`  
*Ponteiro para o servidor a ser atualizado.*
- `Channel * current_channel`  
*Ponteiro para o canal atual.*

### 5.4.1 Descrição detalhada

Classe que representa o sistema.

### 5.4.2 Documentação das funções

#### 5.4.2.1 carregar()

```
void System::carregar ( )
```

Método que carrega os usuários e os servidores de arquivos.

Carrega os dados salvos em arquivos de texto.

#### 5.4.2.2 carregarServidores()

```
void System::carregarServidores ( )
```

Método que carrega os servidores de um arquivo.

Carrega os servidores do sistema.

#### 5.4.2.3 carregarUsuarios()

```
void System::carregarUsuarios ( )
```

Método que carrega os usuários de um arquivo.

Carrega os usuários do sistema.

#### 5.4.2.4 change\_server\_description()

```
void System::change_server_description (
    std::string name,
    std::string description )
```

Método que altera a descrição de um servidor.

Altera a descrição do servidor.

## Parâmetros

<i>name</i>	nome do servidor
<i>description</i>	nova descrição do servidor

**5.4.2.5 create\_server()**

```
void System::create_server (
    std::string name )
```

Método que cria um servidor.

Cria um novo servidor.

## Parâmetros

<i>name</i>	nome do servidor
-------------	------------------

**5.4.2.6 create\_user()**

```
void System::create_user (
    std::string email,
    std::string password,
    std::string name )
```

Método que cria um usuário.

Cria o usuário e o adiciona na lista de usuários do sistema.

**5.4.2.7 disconnect()**

```
void System::disconnect ( )
```

Método que faz o logout de um usuário.

Desconecta o usuário logado.

**5.4.2.8 enter\_channel()**

```
void System::enter_channel (
    std::string name )
```

Método que entra em um canal.

entra em um canal do servidor atual

**5.4.2.9 enter\_server()**

```
void System::enter_server (
    std::string name )
```

Método que entra em um servidor.

Entra em um servidor.

## Parâmetros

<i>name</i>	nome do servidor.
-------------	-------------------

**5.4.2.10 executable()**

```
void System::executable ( )
```

Método que executa o sistema.

A função chamada para executar o sistema.

**5.4.2.11 get\_current\_channel()**

```
Channel * System::get_current_channel ( )
```

Retorna o ponteiro para o canal atual.

Retona o ponteiro para o canal atual.

Retorna

Channel\*

**5.4.2.12 get\_current\_server()**

```
Server * System::get_current_server ( )
```

Retorna o ponteiro para o servidor atual.

Retorna

Server\*

**5.4.2.13 increase\_l\_id()**

```
int System::increase_l_id ( )
```

Aumenta o id do próximo usuário a ser criado.

Atualiza o id que deve ser implementado em um novo usuário.

Retorna

int

**5.4.2.14 join\_server()**

```
void System::join_server (
    std::string name,
    std::string invite_code )
```

Método que adiciona o usuário em um servidor.

Adiciona o usuário ao servidor name.

## Parâmetros

<i>name</i>	nome do servidor
<i>invite_code</i>	código de convite do servidor

**5.4.2.15 leave\_server()**

```
void System::leave_server ( )
```

Método que sai de um servidor.

Sai do servidor atual.

**5.4.2.16 list\_servers()**

```
void System::list_servers ( )
```

Método que lista os servidores.

Lista os servidores do sistema.

**5.4.2.17 logged\_email()**

```
std::string System::logged_email ( )
```

Retorna o email do usuário logado.

Retorna

std::string

**5.4.2.18 logged\_user\_name()**

```
std::string System::logged_user_name ( )
```

Retorna o nome do usuário logado.

Retorna

std::string

**5.4.2.19 login()**

```
void System::login (
    std::string email,
    std::string password )
```

Método que faz o login de um usuário.

Realiza o login do usuário.

**Parâmetros**

<i>email</i>	do usuário tentando login
<i>password</i>	do usuário tentando login

**5.4.2.20 remove\_server()**

```
void System::remove_server (
    std::string name )
```

Método que remove um servidor.

Remove um servidor do sistema.

**Parâmetros**

<i>name</i>	nome do servidor
-------------	------------------

**5.4.2.21 salvar()**

```
void System::salvar ( )
```

Método que salva os usuários e os servidores em arquivos.

Salva os dados do sistema.

**5.4.2.22 salvarServidores()**

```
void System::salvarServidores ( )
```

Método que salva os servidores em um arquivo.

Salva os servidores do sistema.

**5.4.2.23 salvarUsuarios()**

```
void System::salvarUsuarios ( )
```

Método que salva os usuários em um arquivo.

Salva os usuários do sistema.

**5.4.2.24 send\_message()**

```
void System::send_message ( )
```

Método que envia uma mensagem.

Envia uma mensagem para o canal atual.



## 5.4.2.25 set\_server\_invite\_code()

```
void System::set_server_invite_code (
    std::string name,
    std::string invite_code )
```

Método que altera o código de convite de um servidor.

Altera o código de convite do servidor.

## Parâmetros

<i>name</i>	nome do servidor
<i>invite_code</i>	novο código de convite

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

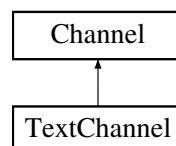
- [include/System.h](#)
- [source/System.cpp](#)

## 5.5 Referência da Classe TextChannel

Classe que representa um canal de texto.

```
#include <TextChannel.h>
```

Diagrama de hierarquia da classe TextChannel:



## Membros Públicos

- [TextChannel \(\)](#)  
*Construtor de [TextChannel](#).*
- `std::vector< Message > getMessages ()`  
*Método que retorna todas as mensagens enviadas no canal.*
- [TextChannel \(std::string name\)](#)  
*Construtor da classe [TextChannel](#).*
- `void send (std::string message, int logged_user_id)`  
*Método que envia uma mensagem no canal.*
- `void list_messages ()`  
*Lista todas as mensagens enviadas no canal.*
- `void add_message (Message message)`  
*Adiciona uma mensagem ao canal.*

## Membros Públicos herdados de [Channel](#)

- [Channel](#) (std::string [name](#))  
*Construtor da classe [Channel](#).*
- std::string [getName](#) ()  
*Método que retorna o nome do canal.*
- void [send\\_message](#) (std::string message, int logged\_user\_id)  
*Método que envia uma mensagem no canal.*
- void [list\\_messages](#) ()  
*Método que lista as mensagens do canal.*

## Atributos Privados

- std::vector< [Message](#) > [messages](#)  
*Vetor que guarda todas as mensagens enviadas no canal.*

## Outros membros herdados

## Atributos Protegidos herdados de [Channel](#)

- std::string [name](#)  
*Variável que guarda o nome do canal.*

### 5.5.1 Descrição detalhada

Classe que representa um canal de texto.

### 5.5.2 Construtores e Destrutores

#### 5.5.2.1 [TextChannel](#)() [1/2]

```
TextChannel::TextChannel ( )
```

Construtor de [TextChannel](#).

#### 5.5.2.2 [TextChannel](#)() [2/2]

```
TextChannel::TextChannel (
    std::string name = "" )
```

Construtor da classe [TextChannel](#).

Construtor de [TextChannel](#).

#### Parâmetros

<a href="#">name</a>	
----------------------	--

### 5.5.3 Documentação das funções

#### 5.5.3.1 add\_message()

```
void TextChannel::add_message (
    Message message )
```

Adiciona uma mensagem ao canal.

##### Parâmetros

<i>message</i>	
----------------	--

#### 5.5.3.2 getMessages()

```
std::vector< Message > TextChannel::getMessages ( )
```

Método que retorna todas as mensagens enviadas no canal.

Retorna todas as mensagens enviadas no canal.

##### Retorna

std::vector<Message>

#### 5.5.3.3 list\_messages()

```
void TextChannel::list_messages ( )
```

Lista todas as mensagens enviadas no canal.

#### 5.5.3.4 send()

```
void TextChannel::send (
    std::string message,
    int logged_user_id )
```

Método que envia uma mensagem no canal.

Envia uma mensagem no canal.

##### Parâmetros

<i>message</i>	
<i>logged_user↵ _id</i>	

pega o tempo atual

converte para o tempo local

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- include/TextChannel.h
- source/TextChannel.cpp

## 5.6 Referência da Classe User

Classe que representa um usuário.

```
#include <User.h>
```

### Membros Públicos

- **User** (std::string name, std::string email, std::string password)  
*Construtor da classe User.*
- **User** (int id, std::string name, std::string email, std::string password)  
*Construtor da classe User.*
- int getId ()  
*Método que retorna o id do usuário.*
- void setId (int id)  
*Método que altera o id do usuário.*
- std::string getName ()  
*Método que retorna o nome do usuário.*
- std::string getEmail ()  
*Método que retorna o email do usuário.*
- std::string getPassword ()  
*Método que retorna a senha do usuário.*
- void updateName (std::string name)  
*Método que altera o nome do usuário.*
- void updateEmail (std::string email)  
*Método que altera o email do usuário.*
- void updatePassword (std::string password)  
*Método que altera a senha do usuário.*
- bool operator== (const User &other)  
*Sobrecarga do operador de igualdade.*

### Atributos Privados

- int id  
*Variável que guarda o id do usuário.*
- std::string name  
*Variável que guarda o nome do usuário.*
- std::string email  
*Variável que guarda o email do usuário.*
- std::string password  
*Variável que guarda a senha do usuário.*

### 5.6.1 Descrição detalhada

Classe que representa um usuário.

### 5.6.2 Construtores e Destrutores

#### 5.6.2.1 User()

```
User::User (
    std::string name,
    std::string email,
    std::string password )
```

Construtor da classe [User](#).

Construtor de [User](#).

##### Parâmetros

<i>name</i>	
<i>email</i>	
<i>password</i>	

### 5.6.3 Documentação das funções

#### 5.6.3.1 getEmail()

```
std::string User::getEmail ( )
```

Método que retorna o email do usuário.

Retorna o email do usuário.

##### Retorna

std::string

#### 5.6.3.2 getId()

```
int User::getId ( )
```

Método que retorna o id do usuário.

Retorna o id do usuário.

##### Retorna

int

#### 5.6.3.3 getName()

```
std::string User::getName ( )
```

Método que retorna o nome do usuário.

Retorna o nome do usuário.

Retorna

std::string

#### 5.6.3.4 getPassword()

```
std::string User::getPassword ( )
```

Método que retorna a senha do usuário.

Retorna a senha do usuário.

Retorna

std::string

#### 5.6.3.5 operator==()

```
bool User::operator== (
    const User & other )
```

Sobrecarga do operador de igualdade.

Sobrecarga do operador de comparação.

Parâmetros

<i>other</i>	
--------------	--

Retorna

true

false

#### 5.6.3.6 setId()

```
void User::setId (
    int id )
```

Método que altera o id do usuário.

Seta o id do usuário.

**Parâmetros**

<i>id</i>	
-----------	--

**5.6.3.7 updateEmail()**

```
void User::updateEmail (
    std::string email )
```

Método que altera o email do usuário.

Atualiza o email do usuário.

**Parâmetros**

<i>email</i>	
--------------	--

**5.6.3.8 updateName()**

```
void User::updateName (
    std::string name )
```

Método que altera o nome do usuário.

Atualiza o nome do usuário.

**Parâmetros**

<i>name</i>	
-------------	--

**5.6.3.9 updatePassword()**

```
void User::updatePassword (
    std::string password )
```

Método que altera a senha do usuário.

Atualiza a senha do usuário.

**Parâmetros**

<i>password</i>	
-----------------	--

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

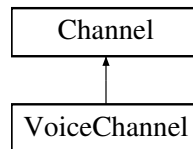
- include/[User.h](#)
- source/[User.cpp](#)

## 5.7 Referência da Classe VoiceChannel

Classe que representa um canal de voz.

```
#include <VoiceChannel.h>
```

Diagrama de hierarquia da classe VoiceChannel:



### Membros Públicos

- [VoiceChannel](#) (std::string [name](#))  
*Construtor da classe [VoiceChannel](#).*
- [Message](#) [getLastMessage](#) ()  
*Método que retorna a última mensagem enviada no canal.*
- void [send](#) (std::string message, int logged\_user\_id)  
*Método que envia uma mensagem no canal.*
- void [list\\_messages](#) ()  
*Lista a última mensagem enviada no canal.*
- void [add\\_message](#) ([Message](#) message)  
*Adiciona uma mensagem ao canal.*

### Membros Públicos herdados de [Channel](#)

- [Channel](#) (std::string [name](#))  
*Construtor da classe [Channel](#).*
- std::string [getName](#) ()  
*Método que retorna o nome do canal.*
- void [send\\_message](#) (std::string message, int logged\_user\_id)  
*Método que envia uma mensagem no canal.*
- void [list\\_messages](#) ()  
*Método que lista as mensagens do canal.*

### Atributos Privados

- [Message](#) [last\\_message](#)  
*Variável que guarda a última mensagem enviada no canal.*

### Outros membros herdados

### Atributos Protegidos herdados de [Channel](#)

- std::string [name](#)  
*Variável que guarda o nome do canal.*



### 5.7.1 Descrição detalhada

Classe que representa um canal de voz.

### 5.7.2 Construtores e Destrutores

#### 5.7.2.1 VoiceChannel()

```
VoiceChannel::VoiceChannel (
    std::string name = "" )
```

Construtor da classe [VoiceChannel](#).

Construtor de [VoiceChannel](#).

Parâmetros

<i>name</i>	
-------------	--

### 5.7.3 Documentação das funções

#### 5.7.3.1 add\_message()

```
void VoiceChannel::add_message (
    Message message )
```

Adiciona uma mensagem ao canal.

Parâmetros

<i>message</i>	
----------------	--

#### 5.7.3.2 getLastMessage()

```
Message VoiceChannel::getLastMessage ( )
```

Método que retorna a última mensagem enviada no canal.

Retorna a última mensagem enviada no canal.

Retorna

[Message](#)

#### 5.7.3.3 list\_messages()

```
void VoiceChannel::list_messages ( )
```

Lista a última mensagem enviada no canal.

#### 5.7.3.4 send()

```
void VoiceChannel::send (
    std::string message,
    int logged_user_id )
```

Método que envia uma mensagem no canal.

Envia uma mensagem no canal.

##### Parâmetros

<i>message</i>	
<i>logged_user_id</i>	

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [include/VoiceChannel.h](#)
- [source/VoiceChannel.cpp](#)

## Capítulo 6

# Arquivos

### 6.1 Referência do Arquivo include/Channel.h

Definição da classe [Channel](#).

```
#include <iostream>
#include <string>
```

#### Componentes

- class [Channel](#)

*Classe que representa um canal.*

#### 6.1.1 Descrição detalhada

Definição da classe [Channel](#).

### 6.2 Channel.h

[Ir para a documentação desse arquivo.](#)

```
00001
00007 #include <iostream>
00008 #include <string>
00009 #ifndef CHANNEL_H
00010 #define CHANNEL_H
00011
00016 class Channel
00017 {
00018 protected:
00019     std::string name;
00020 public:
00021     Channel(std::string name);
00022     std::string getName();
00023     void send_message(std::string message, int logged_user_id){std::cout << "place holder";}
00024     void list_messages(){std::cout << "place holder";}
00025 };
00026
00027 #endif
```

## 6.3 Referência do Arquivo include/Message.h

Definição da classe [Message](#).

```
#include <iostream>
#include <string>
```

### Componentes

- class [Message](#)

*Classe que representa uma mensagem.*

### 6.3.1 Descrição detalhada

Definição da classe [Message](#).

## 6.4 Message.h

[Ir para a documentação desse arquivo.](#)

```
00001 #ifndef MESSAGE_H
00002 #define MESSAGE_H
00003
00009 #include <iostream>
00010 #include <string>
00015 class Message
00016 {
00017 private:
00018     std::string sentAt; //!< Variável que guarda o horário de envio da mensagem.
00019     int sentBy;         //!< Variável que guarda o id do usuário que enviou a mensagem.
00020     std::string content; //!< Variável que guarda o conteúdo da mensagem.
00021
00022 public:
00023     Message();
00024     Message(std::string sentAt, int sentBy, std::string content); //!< Construtor da classe Message.
00025     std::string getSentAt(); //!< Método que retorna o horário de envio da mensagem.
00026     int getSentBy();        //!< Método que retorna o id do usuário que enviou a mensagem.
00027     std::string getContent(); //!< Método que retorna o conteúdo da mensagem.
00028     void setSentAt(std::string sentAt); //!< Método que seta o horário de envio da mensagem.
00029     void setSentBy(int sentBy);        //!< Método que seta o id do usuário que enviou a mensagem.
00030     void setContent(std::string content); //!< Método que seta o conteúdo da mensagem.
00031     bool operator==(const Message &other); //!< Sobrecarga do operador de comparação.
00032     Message operator=(const Message &other); //!< Sobrecarga do operador de atribuição.
00033 };
00034
00035 #endif
```

## 6.5 Referência do Arquivo include/Server.h

Definição da classe [Server](#).

```
#include <bits/stdc++.h>
#include "User.h"
#include "Channel.h"
#include "VoiceChannel.h"
#include "TextChannel.h"
```

## Componentes

- class [Server](#)

*Classe que representa um servidor.*

### 6.5.1 Descrição detalhada

Definição da classe [Server](#).

## 6.6 Server.h

[Ir para a documentação desse arquivo.](#)

```

00001
00006 #include <bits/stdc++.h>
00007 #include "User.h"
00008 #include "Channel.h"
00009 #include "VoiceChannel.h"
00010 #include "TextChannel.h"
00011
00012 #ifndef SERVER_H
00013 #define SERVER_H
00014
00019 class Server
00020 {
00021     protected:
00022         int owner_id;
00023         bool invite_code_enabled = false;
00024         std::string name = "";
00025         std::string description = "";
00026         std::string invite_code = "";
00027         std::vector<int> members;
00028         std::vector<VoiceChannel> voice_channels;
00029         std::vector<TextChannel> text_channels;
00030     public:
00031         Server();
00032         Server(std::string name);
00033         Server(int owner, std::string name, std::string description, std::string invite_code);
00034         void server_clear();
00035         int getOwner();
00036         std::string getName();
00037         std::string getDescription();
00038         std::string getInviteCode();
00039         std::vector<int> getMembers();
00040         std::vector<VoiceChannel> getVoiceChannels();
00041         std::vector<TextChannel> getTextChannels();
00042         bool getInviteCodeEnabled();
00043         void setOwner(int id);
00044         void setDescription(std::string description);
00045         void setInviteCode(std::string invite_code);
00046         void removeInviteCode();
00047         void addMember(int id);
00048         void removeMember(int id);
00049         void listMembers();
00050         bool memberExists(int id);
00051         void create_channel(std::string name, std::string type);
00052         void list_channels();
00053         bool find_channel(std::string name);
00054         void add_TextChannel(TextChannel channel);
00055         void add_VoiceChannel(VoiceChannel channel);
00056         bool operator ==(Server server)
00057         {
00058             return (this->name == server.name);
00059         }
00060 };
00061
00062 #endif

```

## 6.7 Referência do Arquivo include/System.h

Definição da classe [System](#).

```
#include <bits/stdc++.h>
#include "User.h"
#include "Channel.h"
#include "Server.h"
```

### Componentes

- class [System](#)

*Classe que representa o sistema.*

### 6.7.1 Descrição detalhada

Definição da classe [System](#).

## 6.8 System.h

[Ir para a documentação desse arquivo.](#)

```
00001
00008 #include <bits/stdc++.h>
00009 #include "User.h"
00010 #include "Channel.h"
00011 #include "Server.h"
00012
00013 #ifndef SYSTEM_H
00014 #define SYSTEM_H
00015
00016
00022 class System
00023 {
00024     private:
00025         int l_id_assigned = 1;
00026         int logged_user = -1;
00027         std::vector<User> users;
00028         std::vector<Server> servers;
00029         Server current_server = Server();
00030         Server* update_server;
00031         Channel* current_channel;
00032     public:
00033         Channel* get_current_channel();
00034         Server* get_current_server();
00035         std::string logged_user_name();
00036         std::string logged_email();
00037         int increase_l_id();
00038         void executable();
00039         void create_user(std::string email, std::string password, std::string name);
00040         void login(std::string email, std::string password);
00041         // here begins the logged in methods
00042         void disconnect();
00043         void create_server(std::string name);
00044         void change_server_description(std::string name, std::string description);
00045         void set_server_invite_code(std::string name, std::string invite_code);
00046         void list_servers();
00047         void remove_server(std::string name);
00048         void join_server(std::string name, std::string invite_code);
00049         void enter_server(std::string name);
00050         void leave_server();
00051         void enter_channel(std::string name);
00052         void send_message();
00053         void salvarUsuarios();
00054         void salvarServidores();
00055         void salvar();
00056         void carregarUsuarios();
00057         void carregarServidores();
00058         void carregar();
00059 };
00060
00061 #endif
00062
```

## 6.9 Referência do Arquivo include/TextChannel.h

Classe que representa um canal de texto.

```
#include <iostream>
#include <string>
#include <vector>
#include "Channel.h"
#include "Message.h"
#include <ctime>
```

### Componentes

- class [TextChannel](#)

*Classe que representa um canal de texto.*

### 6.9.1 Descrição detalhada

Classe que representa um canal de texto.

## 6.10 TextChannel.h

[Ir para a documentação desse arquivo.](#)

```
00001
00006 #ifndef TEXTCHANNEL_H
00007 #define TEXTCHANNEL_H
00008
00009 #include <iostream>
00010 #include <string>
00011 #include <vector>
00012 #include "Channel.h"
00013 #include "Message.h"
00014 #include <ctime>
00015
00020 class TextChannel : public Channel
00021 {
00022     private:
00023         std::vector<Message> messages;
00024     public:
00025         TextChannel();
00026         std::vector<Message> getMessages();
00027         TextChannel(std::string name);
00028         void send(std::string message, int logged_user_id);
00029         void list_messages();
00030         void add_message(Message message);
00031 };
00032
00033 #endif
```

## 6.11 Referência do Arquivo include/User.h

Definição da classe [User](#).

```
#include <iostream>
#include <string>
```

## Componentes

- class [User](#)

*Classe que representa um usuário.*

### 6.11.1 Descrição detalhada

Definição da classe [User](#).

## 6.12 User.h

[Ir para a documentação desse arquivo.](#)

```
00001
00006 #include <iostream>
00007 #include <string>
00008 #ifndef USER_H
00009 #define USER_H
00010
00015 class User
00016 {
00017 private:
00018     int id;
00019     std::string name;
00020     std::string email;
00021     std::string password;
00022
00023 public:
00024     User(std::string name, std::string email, std::string password);
00025     User(int id, std::string name, std::string email, std::string password);
00026     int getId();
00027     void setId(int id);
00028     std::string getName();
00029     std::string getEmail();
00030     std::string getPassword();
00031     void updateName(std::string name);
00032     void updateEmail(std::string email);
00033     void updatePassword(std::string password);
00034     bool operator==(const User &other);
00035 };
00036
00037 #endif
```

## 6.13 Referência do Arquivo include/VoiceChannel.h

Classe que representa um canal de voz.

```
#include <iostream>
#include <string>
#include "Channel.h"
#include "Message.h"
```

## Componentes

- class [VoiceChannel](#)

*Classe que representa um canal de voz.*



### 6.13.1 Descrição detalhada

Classe que representa um canal de voz.

## 6.14 VoiceChannel.h

[Ir para a documentação desse arquivo.](#)

```
00001
00006 #ifndef VOICECHANNEL_H
00007 #define VOICECHANNEL_H
00008
00009 #include <iostream>
00010 #include <string>
00011 #include "Channel.h"
00012 #include "Message.h"
00013
00018 class VoiceChannel : public Channel
00019 {
00020     private:
00021         Message last_message;
00022     public:
00023         VoiceChannel(std::string name);
00024         Message getLastMessage();
00025         void send(std::string message, int logged_user_id);
00026         void list_messages();
00027         void add_message(Message message);
00028 };
00029
00030 #endif
```

## 6.15 Referência do Arquivo source/Channel.cpp

Implementação da classe [Channel](#).

```
#include <iostream>
#include <string>
#include "Channel.h"
```

### 6.15.1 Descrição detalhada

Implementação da classe [Channel](#).

## 6.16 Referência do Arquivo source/main.cpp

Execução do programa.

```
#include <bits/stdc++.h>
#include "System.h"
```

### Funções

- `int main ()`

### 6.16.1 Descrição detalhada

Execução do programa.

Autor

your name (Artur Revorêdo Pinto)

## 6.17 Referência do Arquivo source/Message.cpp

Implementação da classe [Message](#).

```
#include <iostream>
#include <string>
#include "Message.h"
```

### 6.17.1 Descrição detalhada

Implementação da classe [Message](#).

## 6.18 Referência do Arquivo source/Server.cpp

Implementação da classe [Server](#).

```
#include "Server.h"
#include "Channel.h"
#include "VoiceChannel.h"
#include "TextChannel.h"
```

### 6.18.1 Descrição detalhada

Implementação da classe [Server](#).

## 6.19 Referência do Arquivo source/System.cpp

Implementação dos métodos da classe [System](#).

```
#include <bits/stdc++.h>
#include <fstream>
#include "System.h"
```

### 6.19.1 Descrição detalhada

Implementação dos métodos da classe [System](#).

## 6.20 Referência do Arquivo source/TextChannel.cpp

Implementação da classe [TextChannel](#).

```
#include <iostream>
#include <string>
#include "TextChannel.h"
#include <ctime>
```

### 6.20.1 Descrição detalhada

Implementação da classe [TextChannel](#).

## 6.21 Referência do Arquivo source/User.cpp

Implementação da classe [User](#).

```
#include <iostream>
#include <string>
#include "User.h"
```

### 6.21.1 Descrição detalhada

Implementação da classe [User](#).

## 6.22 Referência do Arquivo source/VoiceChannel.cpp

Implementação da classe [VoiceChannel](#).

```
#include <iostream>
#include <string>
#include <iomanip>
#include "VoiceChannel.h"
#include "Message.h"
#include <ctime>
```

### 6.22.1 Descrição detalhada

Implementação da classe [VoiceChannel](#).



# Índice Remissivo

- add\_message
  - TextChannel, [29](#)
  - VoiceChannel, [35](#)
- add\_TextChannel
  - Server, [14](#)
- add\_VoiceChannel
  - Server, [14](#)
- addMember
  - Server, [15](#)
- carregar
  - System, [22](#)
- carregarServidores
  - System, [22](#)
- carregarUsuarios
  - System, [22](#)
- change\_server\_description
  - System, [22](#)
- Channel, [9](#)
  - Channel, [10](#)
- create\_channel
  - Server, [15](#)
- create\_server
  - System, [23](#)
- create\_user
  - System, [23](#)
- disconnect
  - System, [23](#)
- enter\_channel
  - System, [23](#)
- enter\_server
  - System, [23](#)
- executable
  - System, [24](#)
- find\_channel
  - Server, [15](#)
- get\_current\_channel
  - System, [24](#)
- get\_current\_server
  - System, [24](#)
- getContent
  - Message, [11](#)
- getDescription
  - Server, [16](#)
- getEmail
  - User, [31](#)
- getId
  - User, [31](#)
- getInviteCode
  - Server, [16](#)
- getInviteCodeEnabled
  - Server, [16](#)
- getLastMessage
  - VoiceChannel, [35](#)
- getMembers
  - Server, [16](#)
- getMessages
  - TextChannel, [29](#)
- getName
  - Server, [17](#)
  - User, [31](#)
- getOwner
  - Server, [17](#)
- getPassword
  - User, [32](#)
- getSentAt
  - Message, [11](#)
- getSentBy
  - Message, [11](#)
- getTextChannels
  - Server, [17](#)
- getVoiceChannels
  - Server, [17](#)
- include/Channel.h, [37](#)
- include/Message.h, [38](#)
- include/Server.h, [38](#), [39](#)
- include/System.h, [40](#)
- include/TextChannel.h, [41](#)
- include/User.h, [41](#), [42](#)
- include/VoiceChannel.h, [42](#), [43](#)
- increase\_l\_id
  - System, [24](#)
- join\_server
  - System, [24](#)
- leave\_server
  - System, [25](#)
- list\_channels
  - Server, [18](#)
- list\_messages
  - TextChannel, [29](#)
  - VoiceChannel, [35](#)
- list\_servers
  - System, [25](#)

- listMembers
  - Server, [18](#)
- logged\_email
  - System, [25](#)
- logged\_user\_name
  - System, [25](#)
- login
  - System, [25](#)
- memberExists
  - Server, [18](#)
- Message, [10](#)
  - getContent, [11](#)
  - getSentAt, [11](#)
  - getSentBy, [11](#)
  - Message, [11](#)
  - operator=, [11](#)
  - operator==, [12](#)
- operator=
  - Message, [11](#)
- operator==
  - Message, [12](#)
  - Server, [18](#)
  - User, [32](#)
- README, [1](#)
- remove\_server
  - System, [26](#)
- removeInviteCode
  - Server, [19](#)
- removeMember
  - Server, [19](#)
- salvar
  - System, [26](#)
- salvarServidores
  - System, [26](#)
- salvarUsuarios
  - System, [26](#)
- send
  - TextChannel, [29](#)
  - VoiceChannel, [35](#)
- send\_message
  - System, [26](#)
- Server, [12](#)
  - add\_TextChannel, [14](#)
  - add\_VoiceChannel, [14](#)
  - addMember, [15](#)
  - create\_channel, [15](#)
  - find\_channel, [15](#)
  - getDescription, [16](#)
  - getInviteCode, [16](#)
  - getInviteCodeEnabled, [16](#)
  - getMembers, [16](#)
  - getName, [17](#)
  - getOwner, [17](#)
  - getTextChannels, [17](#)
  - getVoiceChannels, [17](#)
  - list\_channels, [18](#)
  - listMembers, [18](#)
  - memberExists, [18](#)
  - operator==, [18](#)
  - removeInviteCode, [19](#)
  - removeMember, [19](#)
  - Server, [14](#)
  - server\_clear, [19](#)
  - setDescription, [19](#)
  - setInviteCode, [20](#)
  - setOwner, [20](#)
  - server\_clear
    - Server, [19](#)
  - set\_server\_invite\_code
    - System, [26](#)
  - setDescription
    - Server, [19](#)
  - setId
    - User, [32](#)
  - setInviteCode
    - Server, [20](#)
  - setOwner
    - Server, [20](#)
  - source/Channel.cpp, [43](#)
  - source/main.cpp, [43](#)
  - source/Message.cpp, [44](#)
  - source/Server.cpp, [44](#)
  - source/System.cpp, [44](#)
  - source/TextChannel.cpp, [45](#)
  - source/User.cpp, [45](#)
  - source/VoiceChannel.cpp, [45](#)
  - System, [20](#)
    - carregar, [22](#)
    - carregarServidores, [22](#)
    - carregarUsuarios, [22](#)
    - change\_server\_description, [22](#)
    - create\_server, [23](#)
    - create\_user, [23](#)
    - disconnect, [23](#)
    - enter\_channel, [23](#)
    - enter\_server, [23](#)
    - executable, [24](#)
    - get\_current\_channel, [24](#)
    - get\_current\_server, [24](#)
    - increase\_l\_id, [24](#)
    - join\_server, [24](#)
    - leave\_server, [25](#)
    - list\_servers, [25](#)
    - logged\_email, [25](#)
    - logged\_user\_name, [25](#)
    - login, [25](#)
    - remove\_server, [26](#)
    - salvar, [26](#)
    - salvarServidores, [26](#)
    - salvarUsuarios, [26](#)
    - send\_message, [26](#)
    - set\_server\_invite\_code, [26](#)
  - TextChannel, [27](#)

- add\_message, [29](#)
- getMessages, [29](#)
- list\_messages, [29](#)
- send, [29](#)
- TextChannel, [28](#)

updateEmail

- User, [33](#)

updateName

- User, [33](#)

updatePassword

- User, [33](#)

User, [30](#)

- getEmail, [31](#)
- getId, [31](#)
- getName, [31](#)
- getPassword, [32](#)
- operator==, [32](#)
- setId, [32](#)
- updateEmail, [33](#)
- updateName, [33](#)
- updatePassword, [33](#)
- User, [31](#)

VoiceChannel, [34](#)

- add\_message, [35](#)
- getLastMessage, [35](#)
- list\_messages, [35](#)
- send, [35](#)
- VoiceChannel, [35](#)