Referência da Classe CadastroDeJogadores

`#include <CadastroDeJogadores.hpp>`

Membros Públicos

void **criarArquivoJogadores** () const
Essa função cria um arquivo chamado jogadores.txt se ele não existir. Ela verifica se o arquivo já está presente usando a função std::filesystem::exists. Se o arquivo existir, uma mensagem informando que ele já existe é exibida. Caso contrário, um novo arquivo é criado. Se houver algum erro ao criar o arquivo, uma mensagem de erro será exibida.

void **adicionarJogadorNoArquivo** (const **Jogador** &jogador)
Essa função adiciona um novo jogador ao arquivo jogadores.txt. Ela primeiro garante que o arquivo existe chamando criar_arquivo_jogadores(). Em seguida, carrega todos os jogadores existentes do arquivo para evitar duplicatas. Se o jogador já estiver presente, uma mensagem será exibida e a função terminará. Se o jogador não existir, ele será adicionado à lista e o arquivo será atualizado para incluir o novo jogador.

void **carregarJogadoresDeArquivo** (const std::string &nome_arquivo)
Essa função carrega a lista de jogadores a partir do arquivo "jogadores.txt". Ela abre o arquivo para leitura, limpa a lista atual de jogadores e, em seguida, lê cada nome de jogador do arquivo, adicionando-o à lista de jogadores em memória. Se o arquivo não puder ser aberto, uma mensagem de erro é exibida.

bool **buscarJogadorNoArquivo** (const std::string &nome)
Essa função verifica se um jogador específico está presente no arquivo jogadores.txt. Ela carrega a lista de jogadores e percorre a lista para verificar se o nome do jogador existe.

void **removerJogadorDoArquivo** (const std::string &nome)
Essa função remove um jogador específico do arquivo "jogadores.txt". Ela carrega a lista de jogadores, procura o jogador com o nome especificado, e se encontrado, o remove da lista. O arquivo é então atualizado para refletir essa remoção. Se o jogador não for encontrado, uma mensagem informando isso é exibida.

void **atualizarNomeDoJogador** (const std::string &nome_atual, const std::string &novo_nome)
Essa função atualiza o nome de um jogador específico no arquivo jogadores.txt. Ela lê o arquivo e substitui o nome atual pelo novo nome, mantendo o restante do conteúdo inalterado. Se o nome atual não for encontrado, uma mensagem é exibida. Após a atualização, o arquivo é salvo novamente.

void **listarJogadoresDoArquivo** ()
Essa função lista todos os jogadores presentes no arquivo jogadores.txt. Ela lê o conteúdo do arquivo linha por linha e exibe cada nome de jogador. Se o arquivo estiver vazio, uma mensagem informando que a lista está vazia é exibida. Se o arquivo não puder ser aberto, uma mensagem de erro será exibida.

void **atualizarArquivo** () const
Essa função atualiza o conteúdo do arquivo jogadores.txt com a lista atual de jogadores. Ela sobrescreve o arquivo existente, escrevendo cada nome de jogador na lista. Se o arquivo não

puder ser aberto para escrita, uma mensagem de erro será exibida.

void	**limparArquivo** () Essa função limpa o conteúdo do arquivo "jogadores.txt". Após limpar o arquivo, ele é atualizado chamando a função atualizar_arquivo(). Se houver um erro ao abrir o arquivo, uma mensagem de erro será exibida.
Jogador *	**getJogadorPorNome** (const std::string &nome) Essa função retorna um ponteiro para um objeto **Jogador** com base no nome fornecido. Ela carrega a lista de jogadores do arquivo e procura um jogador com o nome especificado. Se o jogador for encontrado, um ponteiro para esse jogador é retornado. Caso contrário, um ponteiro nulo é retornado.

Documentação das funções

◆ adicionarJogadorNoArquivo()

void CadastroDeJogadores::adicionarJogadorNoArquivo (const **Jogador** & jogador)

Essa função adiciona um novo jogador ao arquivo jogadores.txt. Ela primeiro garante que o arquivo existe chamando criar_arquivo_jogadores(). Em seguida, carrega todos os jogadores existentes do arquivo para evitar duplicatas. Se o jogador já estiver presente, uma mensagem será exibida e a função terminará. Se o jogador não existir, ele será adicionado à lista e o arquivo será atualizado para incluir o novo jogador.

Parâmetros

jogador Objeto do tipo **Jogador** que será adicionado ao arquivo.

```
29                                                                  {
30      criarArquivoJogadores();
31
32      carregarJogadoresDeArquivo("jogadores.txt");
33
34
35      bool jogador_existe = std::any_of(jogadores.begin(), jogadores.end(),
36          [&jogador](const Jogador& j) { return j.getNome() ==
    jogador.getNome(); });
37
38      if (jogador_existe) {
39          std::cout << "O jogador " << jogador.getNome() << " já existe." <<
    std::endl;
40          return;
41      }
42
43
44      jogadores.push_back(jogador);
45      atualizarArquivo();
46
47      std::cout << "Jogador " << jogador.getNome() << " adicionado ao
    arquivo." << std::endl;
48  }
```

◆ atualizarArquivo()

void CadastroDeJogadores::atualizarArquivo ( ) const

Essa função atualiza o conteúdo do arquivo jogadores.txt com a lista atual de jogadores. Ela sobrescreve o arquivo existente, escrevendo cada nome de jogador na lista. Se o arquivo não puder ser aberto para escrita, uma mensagem de erro será exibida.

```
155                                                 {
156     std::string nome_arquivo = "jogadores.txt";
157     std::ofstream arquivo(nome_arquivo, std::ios::trunc);
158     if (arquivo.is_open()) {
159         for (const auto& jogador : jogadores) {
160             arquivo << std::quoted(jogador.getNome()) << std::endl;
161         }
162         arquivo.close();
163         std::cout << "Arquivo " << nome_arquivo << " atualizado com
    sucesso." << std::endl;
164     } else {
165         std::cout << "Não foi possível abrir o arquivo " << nome_arquivo <<
    " para atualizar." << std::endl;
166     }
167 }
```

◆ atualizarNomeDoJogador()

void CadastroDeJogadores::atualizarNomeDoJogador (const std::string & nome_atual, const std::string & novo_nome)

Essa função atualiza o nome de um jogador específico no arquivo jogadores.txt. Ela lê o arquivo e substitui o nome atual pelo novo nome, mantendo o restante do conteúdo inalterado. Se o nome atual não for encontrado, uma mensagem é exibida. Após a atualização, o arquivo é salvo novamente.

Parâmetros

nome_atual Nome atual do jogador que será atualizado.

novo_nome Novo nome que substituirá o nome atual do jogador.

```
169 
    {
170     std::string nome_arquivo = "jogadores.txt";
171 
172     std::ifstream arquivo(nome_arquivo);
173 
174     if (!arquivo) {
175         std::cerr << "Erro ao abrir o arquivo " << nome_arquivo << " para
    leitura." << std::endl;
176         return;
177     }
178 
179     std::string conteudo_atualizado;
180     std::string linha;
181 
182     bool encontrado = false;
183 
184     while (std::getline(arquivo, linha)) {
185 
186         if (!linha.empty() && linha.front() == '\"' && linha.back() == '\"')
    {
187             linha = linha.substr(1, linha.length() - 2);
188         }
189 
190         if (linha == nome_atual) {
191             conteudo_atualizado = novo_nome ;
192             encontrado = true;
193         } else {
194             conteudo_atualizado = linha;
195         }
196     }
197     arquivo.close();
198 
199     if (!encontrado) {
200         std::cout << "Nome atual não encontrado no arquivo." << std::endl;
201         return;
202     }
203 
204     std::ofstream arquivo_atualizado(nome_arquivo, std::ios::trunc);
205     if (!arquivo_atualizado) {
206         std::cerr << "Erro ao abrir o arquivo " << nome_arquivo << " para
    escrita." << std::endl;
207         return;
208     }
209 
210     arquivo_atualizado << conteudo_atualizado;
211     arquivo_atualizado.close();
212 
213     std::cout << "Nome do jogador atualizado com sucesso." << std::endl;
```

```
214 }
```

buscarJogadorNoArquivo()

bool CadastroDeJogadores::buscarJogadorNoArquivo (const std::string & nome)

Essa função verifica se um jogador específico está presente no arquivo jogadores.txt. Ela carrega a lista de jogadores e percorre a lista para verificar se o nome do jogador existe.

Parâmetros

nome Nome do jogador a ser buscado no arquivo.

Valores Retornados

TRUE Exibe uma mensagem confirmando que o jogador foi encontrado.

FALSE Exibe uma mensagem informando que o jogador não foi encontrado.

```
121                                                                 {
122     const std::string nome_arquivo = "jogadores.txt";
123     carregarJogadoresDeArquivo(nome_arquivo);
124 
125     for(const auto& jogador : jogadores){
126         if(jogador.getNome()==nome){
127             std::cout << "Jogador "<< nome <<"encontrado no arquivo."
    <<std::endl;
128             return true;
129         }
130     }
131     std::cout << "Jogador " << nome << " não encontrado no arquivo." <<
    std::endl;
132     return false;
133 }
```

carregarJogadoresDeArquivo()

void CadastroDeJogadores::carregarJogadoresDeArquivo (const std::string & nome_arquivo)

Essa função carrega a lista de jogadores a partir do arquivo "jogadores.txt". Ela abre o arquivo para leitura, limpa a lista atual de jogadores e, em seguida, lê cada nome de jogador do arquivo, adicionando-o à lista de jogadores em memória. Se o arquivo não puder ser aberto, uma mensagem de erro é exibida.

Parâmetros

nome_arquivo Nome do arquivo de onde os jogadores serão carregados.

```
50 
   {
51     std::ifstream arquivo(nome_arquivo);
52 
53     if(arquivo.is_open()){
54         std:: string nome;
55         jogadores.clear();
56 
57         while (arquivo >> std::quoted(nome)) {
58             jogadores.emplace_back(nome);
59         }
60 
61         arquivo.close();
62         std::cout << "Jogadores carregados do arquivo: "<< nome_arquivo
   <<"."<< std::endl;
63     }else{
64         std::cout << "Não foi possível abrir o arquivo "<< nome_arquivo
   <<"."<< std::endl;
65     }
66 }
```

◆ criarArquivoJogadores()

void CadastroDeJogadores::criarArquivoJogadores ( ) const

Essa função cria um arquivo chamado jogadores.txt se ele não existir. Ela verifica se o arquivo já está presente usando a função std::filesystem::exists. Se o arquivo existir, uma mensagem informando que ele já existe é exibida. Caso contrário, um novo arquivo é criado. Se houver algum erro ao criar o arquivo, uma mensagem de erro será exibida.

```
13                                                      {
14      const std::string nome_arquivo = "jogadores.txt";
15
16      std::ifstream ifile(nome_arquivo.c_str());
17      if (ifile) {
18          std::cout << "O arquivo " << nome_arquivo << " já existe." <<
    std::endl;
19      } else {
20          std::ofstream arquivo(nome_arquivo);
21          if(!arquivo){
22              std::cerr <<"Erro ao criar o arquivo: "<< nome_arquivo
    <<std::endl;
23          } else {
24              std::cout << "Arquivo " << nome_arquivo << " criado com
    sucesso." << std::endl;
25          }
26      }
27  }
```

◆ getJogadorPorNome()

Jogador * CadastroDeJogadores::getJogadorPorNome (const std::string & nome)

Essa função retorna um ponteiro para um objeto **Jogador** com base no nome fornecido. Ela carrega a lista de jogadores do arquivo e procura um jogador com o nome especificado. Se o jogador for encontrado, um ponteiro para esse jogador é retornado. Caso contrário, um ponteiro nulo é retornado.

Parâmetros

nome Nome do jogador a ser buscado.

Retorna

Ponteiro para o objeto **Jogador** encontrado ou nulo se o jogador não for encontrado.

```
216                                                                     {
217     for (auto& jogador : jogadores) {
218         if (jogador.getNome() == nome) {
219             return &jogador;
220         }
221     }
222     return nullptr;
223 }
```

◆ limparArquivo()

void CadastroDeJogadores::limparArquivo ( )

Essa função limpa o conteúdo do arquivo "jogadores.txt". Após limpar o arquivo, ele é atualizado chamando a função atualizar_arquivo(). Se houver um erro ao abrir o arquivo, uma mensagem de erro será exibida.

```
68                                          {
69
70
71      std::string nome_arquivo = "jogadores.txt";
72
73      std::ofstream arquivo(nome_arquivo, std::ios::trunc);
74
75      atualizarArquivo();
76
77      if(!arquivo){
78          std::cerr <<"Erro ao abrir o arquivo "<< nome_arquivo <<"."<<
    std::endl;
79          return;
80      }
81      arquivo.close();
82
83      std::cout <<"Arquivo limpo com sucesso." << std::endl;
84  }
```

◆ listarJogadoresDoArquivo()

void CadastroDeJogadores::listarJogadoresDoArquivo ( )

Essa função lista todos os jogadores presentes no arquivo jogadores.txt. Ela lê o conteúdo do arquivo linha por linha e exibe cada nome de jogador. Se o arquivo estiver vazio, uma mensagem informando que a lista está vazia é exibida. Se o arquivo não puder ser aberto, uma mensagem de erro será exibida.

```
86                                                  {
87      std::string nome_arquivo = "jogadores.txt";
88
89      std::ifstream arquivo(nome_arquivo);
90
91
92
93      if(!arquivo){
94          std::cerr <<"Erro ao abrir o arquivo "<< nome_arquivo <<"."<<
    std::endl;
95          return;
96      }
97
98      std::string linha;
99      bool lista_vazia = true;
100
101     std::cout << "Lista de jogadores:" << std::endl;
102
103     while (std::getline(arquivo, linha)) {
104             if (!linha.empty()) {
105
106                 if (linha.front() == '\"' && linha.back() == '\"') {
107                     linha = linha.substr(1, linha.length() - 2);
108                 }
109                 std::cout << linha << std::endl;
110                 lista_vazia = false;
111             }
112         }
113     arquivo.close();
114
115     if(lista_vazia){
116         std::cout << "A lista de jogadores está vazia." << std::endl;
117     }
118
119 }
```

◆ removerJogadorDoArquivo()

void CadastroDeJogadores::removerJogadorDoArquivo (const std::string & nome)

Essa função remove um jogador específico do arquivo "jogadores.txt". Ela carrega a lista de jogadores, procura o jogador com o nome especificado, e se encontrado, o remove da lista. O arquivo é então atualizado para refletir essa remoção. Se o jogador não for encontrado, uma mensagem informando isso é exibida.

Parâmetros

nome Nome do jogador a ser removido do arquivo.

```
135                                                                 {
136     const std::string nome_arquivo = "jogadores.txt";
137     carregarJogadoresDeArquivo(nome_arquivo);
138 
139     auto it = std::remove_if(jogadores.begin(), jogadores.end(),
140                              [&nome](const Jogador& jogador) {
141                                  return jogador.getNome() == nome;
142                              });
143 
144     if (it != jogadores.end()) {
145 
146         jogadores.erase(it, jogadores.end());
147         atualizarArquivo();
148         std::cout << "Jogador " << nome << " foi removido com sucesso." <<
    std::endl;
149 
150     } else {
151 
152         std::cout << "Jogador com nome " << nome << " não encontrado no
    arquivo." << std::endl;
153     }
154 }
```

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

- /Users/iangodoi/Desktop/TP-jogosTabuleiro-cpp-desenvolvimento/include/**CadastroDeJogadores.hpp**
- /Users/iangodoi/Desktop/TP-jogosTabuleiro-cpp-desenvolvimento/src/**CadastroDeJogadores.cpp**