Monstrinhos

Gerado por Doxygen 1.11.0

Capítulo 1

Trabalho-Final-PDS2

1.1 Introdução

O projeto desenvolvido, com inspiração no clássico Pokemón Stadium, se baseia em um jogo de luta de arena de player vs bot.

1.2 Como jogar

1. **Início do Jogo**:
    - Abrindo o jogo insira o seu nome.
2. **Escolha dos Monstrinhos**:
    - Você começa com 12 moedas.
    - Use as moedas para escolher 4 monstrinhos, cada um com um custo entre 1 e 5 moedas.
    - Os monstrinhos são os seus aliados na batalha!
3. **Enfrentando o Treinador**:
    - Após escolher seus monstrinhos, você enfrentará um treinador com seu próprio time de 4 monstrinhos.
    - A batalha começa e você pode escolher entre as seguintes opções:
        - **Atacar**: Escolha um dos 4 ataques do seu monstrinho para atacar o inimigo.

Mudar de Monstrinho: Troque o monstrinho atual para um de seus outros monstrinhos para atacar.

- **Usar um Item**: Use um item especial em seu monstrinho atual para melhorar suas chances na batalha.

Ataques dos Monstrinhos:

- Cada monstrinho possui 4 ataques diferentes.
- Escolha o ataque que você deseja usar para causar dano ao inimigo.

Itens do jogo:

- **Estamina**: Recupera a estamina de um ataque!
- **Revive**: Revive um monstrinho já derrotado!
- **Cura**: Cura o HP de um monstrinho!

Fim do Jogo:

- O jogo termina quando todos os monstrinhos de um dos times são derrotados.

1.3 Dicas

- **Moedas**: Gerencie suas moedas com sabedoria para garantir que você tenha os melhores monstrinhos para a batalha.
- **Estratégia**: Use a estratégia para escolher quando mudar de monstrinho ou usar itens para obter vantagem na batalha.
- **Tipos**: Lembre-se que dependendo do tipo do seu ataque e do tipo do monstrinho inimigo você pode ter vantagem!

1.4 Instruções de Compilação e Execução

Clone o repositório utilizando o link
```
https://github.com/Lucaszioli/Trabalho-Final-PDS2
```

Crie um diretório de build
```
mkdir build
```

Acesse o diretório criado
```
cd build
```

Execute o comando
```
cmake ..
```

Compile o projeto usando o makefile gerado
```
make
```

Execute o programa criado
```
./Monstrinhos
```

1.5 Autores

- Henrique Dias
- Lucas Zioli
- Malu Lauar
- Matheus Gregor
- Otávio Serafim

Capítulo 2

Índice Hierárquico

2.1 Hierarquia de Classes

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

Capítulo 3

Índice dos Componentes

3.1 Lista de Classes

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

Capítulo 4

Índice dos Arquivos

4.1 Lista de Arquivos

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

Capítulo 5

Classes

5.1 Referência da Classe Ataque

Representa um ataque em um jogo.

`#include <Ataque.hpp>`

Membros Públicos

- Ataque (int ID, string nome, string tipo, int dano, string descricao, int quantidade, double chanceAcerto)

  Constrói um novo objeto Ataque.
- int getID ()

  Retorna o ID do ataque.
- string getNome ()

  Retorna o nome do ataque.
- string getTipo ()

  Retorna o tipo do ataque.
- int getDano ()

  Retorna o dano causado pelo ataque.
- string getDescricao ()

  Retorna a descrição do ataque.
- int getQuantidade ()

  Retorna a quantidade máxima de vezes que o ataque pode ser usado.
- int getQuantidadeAtual ()

  Retorna a quantidade atual de vezes que o ataque pode ser usado.
- void setQuantidadeAtual (int valor)

  Define a quantidade atual de vezes que o ataque pode ser usado.
- double getChanceAcerto ()

  Retorna a chance de acerto do ataque.
- bool fazerAtaque (Monstrinho &inimigo)

  Realiza um ataque em um monstrinho.

Membros públicos estáticos

- static vector< Ataque > construirAtaques ()

  Constrói uma lista de ataques a partir de um arquivo CSV.
- static double calcularEfetividade (string tipoAtaque, vector< string > tiposMonstrinho)

  Calcula a efetividade de um ataque.

5.1.1 Descrição detalhada

Representa um ataque em um jogo.

Cada ataque tem um ID, nome, tipo, dano, descrição e quantidade.

5.1.2 Construtores e Destrutores

5.1.2.1 Ataque()

```
Ataque::Ataque (
            int ID,
            string nome,
            string tipo,
            int dano,
            string descricao,
            int quantidade,
            double chanceAcerto)  [inline]
```

Constrói um novo objeto Ataque.

Parâmetros

ID	O ID do ataque.
nome	O nome do ataque.
tipo	O tipo do ataque.
dano	O dano causado pelo ataque.
descricao	A descrição do ataque.
quantidade	A quantidade de vezes que o ataque pode ser usado.
chanceAcerto	A chance de acerto do ataque.

5.1.3 Documentação das funções

5.1.3.1 calcularEfetividade()

```
double Ataque::calcularEfetividade (
            string tipoAtaque,
            vector< string > tiposMonstrinho)  [static]
```

Calcula a efetividade de um ataque.

Parâmetros

tipoAtaque	O tipo do ataque.
tiposMonstrinho	Os tipos do monstrinho que será atacado.

Retorna

double A efetividade do ataque.

5.1.3.2 construirAtaques()

```
vector< Ataque > Ataque::construirAtaques ()  [static]
```

Constrói uma lista de ataques a partir de um arquivo CSV.

Retorna

vector<Ataque> Um vetor com todos os ataques do arquivo CSV.

5.1.3.3 fazerAtaque()

```
bool Ataque::fazerAtaque (
            Monstrinho & inimigo)
```

Realiza um ataque em um monstrinho.

Parâmetros

inimigo	O monstrinho que será atacado.

Retorna

bool true se o monstrinho conseguiu realizar o ataque, false caso contrário.

5.1.3.4 getChanceAcerto()

```
double Ataque::getChanceAcerto ()
```

Retorna a chance de acerto do ataque.

Retorna

double A chance de acerto do ataque.

5.1.3.5 getDano()

```
int Ataque::getDano ()
```

Retorna o dano causado pelo ataque.

Retorna

int O dano causado pelo ataque.

5.1.3.6 getDescricao()

```
string Ataque::getDescricao ()
```

Retorna a descrição do ataque.

Retorna

string A descrição do ataque.

5.1.3.7 getID()

```
int Ataque::getID ()
```

Retorna o ID do ataque.

Retorna

int O ID do ataque.

5.1.3.8 getNome()

```
string Ataque::getNome ()
```

Retorna o nome do ataque.

Retorna

string O nome do ataque.

5.1.3.9 getQuantidade()

```
int Ataque::getQuantidade ()
```

Retorna a quantidade máxima de vezes que o ataque pode ser usado.

Retorna

int A quantidade de vezes que o ataque pode ser usado.

5.1.3.10 getQuantidadeAtual()

```
int Ataque::getQuantidadeAtual ()
```

Retorna a quantidade atual de vezes que o ataque pode ser usado.

Retorna

int A quantidade atual de vezes que o ataque pode ser usado.

5.1.3.11 getTipo()

```
string Ataque::getTipo ()
```

Retorna o tipo do ataque.

Retorna

string O tipo do ataque.

5.1.3.12 setQuantidadeAtual()

```
void Ataque::setQuantidadeAtual (
            int valor)
```

Define a quantidade atual de vezes que o ataque pode ser usado.

Parâmetros

valor	A nova quantidade atual de vezes que o ataque pode ser usado.

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

- HPPs/Ataque.hpp
- CPPs/Ataque.cpp

5.2 Referência da Classe AtaqueError

Diagrama de hierarquia da classe AtaqueError:

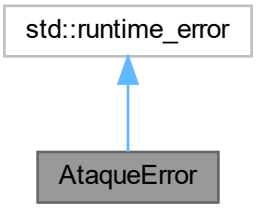
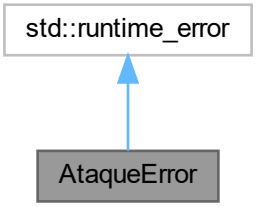


Diagrama de colaboração para AtaqueError:



Membros Públicos

- **AtaqueError** (const std::string &message)

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/AtaqueError.hpp

5.3 Referência da Classe BadRequestError

Diagrama de hierarquia da classe BadRequestError:

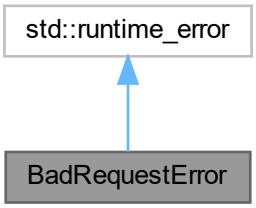
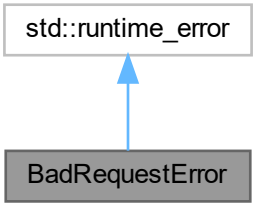


Diagrama de colaboração para BadRequestError:



Membros Públicos

- **BadRequestError** (const std::string &message)

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/BadRequestError.hpp

5.4 Referência da Classe Bot

Representa um Bot em um jogo.

```
#include <Bot.hpp>
```

Diagrama de hierarquia da classe Bot:

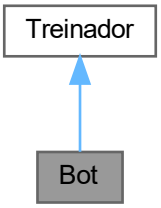
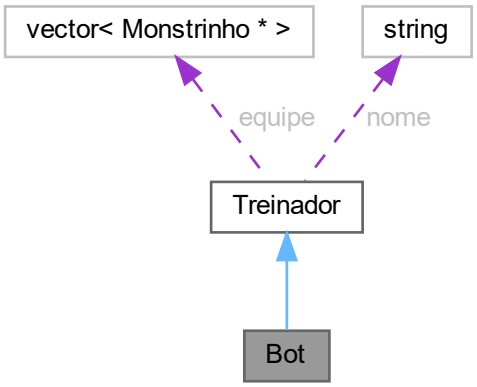


Diagrama de colaboração para Bot:



Membros Públicos

- Bot (int ID, std::string nome, vector< Monstrinho * > equipe, vector< std::string > fala)

  Constrói um novo objeto Bot.
- vector< std::string > getFala ()

  Obtém o discurso do Bot.
- bool mudaEquipe () override

  Altera a equipe de objetos Monstrinho.

Membros Públicos herdados de Treinador

- Treinador (int ID, std::string nome, vector< Monstrinho * > equipe)

  Construtor da classe Treinador.
- int getID ()

Obtém o identificador do treinador.

- std::string getNome ()

  Obtém o nome do treinador.

- vector< Monstrinho * > getEquipe ()

  Obtém a equipe de monstrinhos do treinador.

- bool verificaEquipe ()

  Verifica se a equipe de monstrinhos do treinador está completa.

- void **imprimeEquipe** ()

  Imprime a equipe de monstrinhos do treinador.

- void colocaMonstrinho (Monstrinho monstrinho)

  Coloca um monstrinho na equipe do treinador.

Outros membros herdados

Atributos Protegidos herdados de Treinador

- int ID
- vector< Monstrinho * > equipe
- std::string nome

5.4.1 Descrição detalhada

Representa um Bot em um jogo.

A classe Bot é uma classe derivada da classe Treinador e representa um Bot em um jogo. Ela contém informações sobre o ID do Bot, nome, equipe de objetos Monstrinho e discurso.

5.4.2 Construtores e Destrutores

5.4.2.1 Bot()

```
Bot::Bot (
            int ID,
            std::string nome,
            vector< Monstrinho * > equipe,
            vector< std::string > fala)
```

Constrói um novo objeto Bot.

Parâmetros

ID	O ID do Bot.
nome	O nome do Bot.
equipe	A equipe de objetos Monstrinho.
fala	O discurso do Bot.

5.4.3 Documentação das funções

5.4.3.1 getFala()

```
vector< std::string > Bot::getFala ()
```

Obtém o discurso do Bot.

Retorna

O discurso do Bot.

5.4.3.2 mudaEquipe()

```
bool Bot::mudaEquipe ()  [override], [virtual]
```

Altera a equipe de objetos Monstrinho.

Esta função substitui a função mudaEquipe() da classe base.

Implementa Treinador.

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

- HPPs/Bot.hpp
- CPPs/Bot.cpp

5.5 Referência da Classe Cura

Classe que representa o item Cura.

```
#include <Cura.hpp>
```

Diagrama de hierarquia da classe Cura:

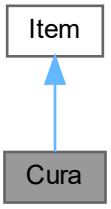
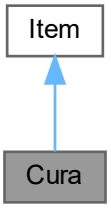


Diagrama de colaboração para Cura:



Membros Públicos

- void pegarItem () override

  Método para pegar o item.
- bool usarItem (Monstrinho ∗monstro) override

  Usa um item selecionado.
- void setCura (int cura)

  Define a quantidade de cura do item.
- int getCura ()

  Obtém a quantidade de cura do item.

Membros Públicos herdados de Item

- string getTipo ()

  Obtém o tipo do item.
- void setTipo (string tipo)

  Define o tipo do item.
- string getRaridade ()

  Obtém a raridade do item.
- void setRaridade (string raridade)

  Define a raridade do item.
- string getNome ()

  Obtém o nome do item.
- void setNome (string nome)

  Define o nome do item.
- string getDescricao ()

  Obtém a descrição.
- void setDescricao (string descricao)

  Define a descrição do item.

5.5.1 Descrição detalhada

Classe que representa o item Cura.

5.5.2 Documentação das funções

5.5.2.1 getCura()

```
int Cura::getCura ()
```

Obtém a quantidade de cura do item.

Retorna

A quantidade de cura.

5.5.2.2 pegarItem()

```
void Cura::pegarItem ()  [override], [virtual]
```

Método para pegar o item.

Implementa Item.

5.5.2.3 setCura()

```
void Cura::setCura (
            int cura)
```

Define a quantidade de cura do item.

Parâmetros

cura	A quantidade de cura.

5.5.2.4 usarItem()

```
bool Cura::usarItem (
            Monstrinho * monstro)  [override], [virtual]
```

Usa um item selecionado.

Parâmetros

Monstro	O monstrinho em que o item será utilizado

Retorna

true se o item foi usado com sucesso, false se não foi

Implementa Item.

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

- HPPs/Cura.hpp
- CPPs/Cura.cpp

5.6 Referência da Classe CuraError

Diagrama de hierarquia da classe CuraError:

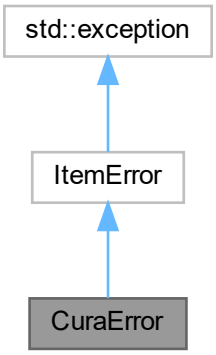
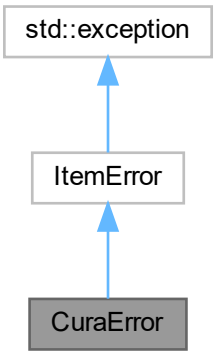


Diagrama de colaboração para CuraError:



Membros Públicos

- **CuraError** (const std::string &message)

Membros Públicos herdados de ItemError

- **ItemError** (const std::string &message)
- const char * **what** () const noexcept override

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/CuraError.hpp

5.7 Referência da Classe EscolhaError

Diagrama de hierarquia da classe EscolhaError:

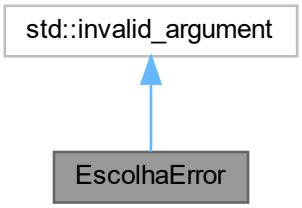
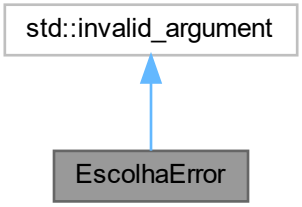


Diagrama de colaboração para EscolhaError:



Membros Públicos

- **EscolhaError** (const std::string &message)

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/EscolhaError.hpp

5.8 Referência da Classe Estamina

Classe que representa o item Estamina.

`#include <Estamina.hpp>`

Diagrama de hierarquia da classe Estamina:

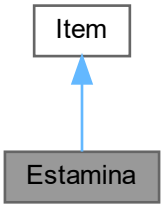
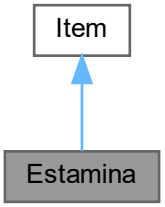


Diagrama de colaboração para Estamina:



Membros Públicos

- void pegarItem () override

  Método para pegar o item.
- int **getEnergia** ()
- void **setEnergia** (int energia)
- bool usarItem (Monstrinho ∗monstro) override

  Usa um item selecionado.

Membros Públicos herdados de Item

- string getTipo ()

  Obtém o tipo do item.
- void setTipo (string tipo)

  Define o tipo do item.
- string getRaridade ()

  Obtém a raridade do item.
- void setRaridade (string raridade)

  Define a raridade do item.

- string getNome ()

  Obtém o nome do item.
- void setNome (string nome)

  Define o nome do item.
- string getDescricao ()

  Obtém a descrição.
- void setDescricao (string descricao)

  Define a descrição do item.

5.8.1 Descrição detalhada

Classe que representa o item Estamina.

5.8.2 Documentação das funções

5.8.2.1 pegarItem()

```
void Estamina::pegarItem ()  [override], [virtual]
```

Método para pegar o item.

Implementa Item.

5.8.2.2 usarItem()

```
bool Estamina::usarItem (
            Monstrinho * monstro)  [override], [virtual]
```

Usa um item selecionado.

Parâmetros

Monstro	O monstrinho em que o item será utilizado

Retorna

true se o item foi usado com sucesso, false se não foi

Implementa Item.

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

- HPPs/Estamina.hpp
- CPPs/Estamina.cpp

5.9 Referência da Classe EstaminaError

Diagrama de hierarquia da classe EstaminaError:

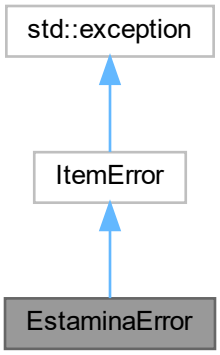
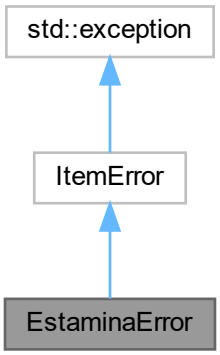


Diagrama de colaboração para EstaminaError:



Membros Públicos

- **EstaminaError** (const std::string &message)

Membros Públicos herdados de ItemError

- **ItemError** (const std::string &message)
- const char * **what** () const noexcept override

A documentação para essa classe foi gerada a partir do seguinte arquivo:

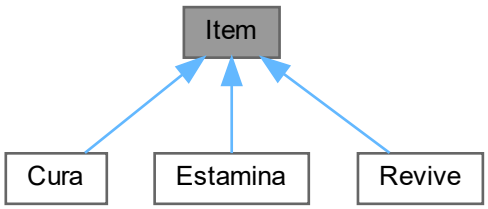
- ERR/EstaminaError.hpp

5.10 Referência da Classe Item

Classe que representa um item.

`#include <Item.hpp>`

Diagrama de hierarquia da classe Item:



Membros Públicos

- virtual bool usarItem (Monstrinho ∗monstro)=0
  Usa um item selecionado.
- string getTipo ()
  Obtém o tipo do item.
- void setTipo (string tipo)
  Define o tipo do item.
- virtual void pegarItem ()=0
  Método para pegar o item.
- string getRaridade ()
  Obtém a raridade do item.
- void setRaridade (string raridade)
  Define a raridade do item.
- string getNome ()
  Obtém o nome do item.
- void setNome (string nome)
  Define o nome do item.
- string getDescricao ()
  Obtém a descrição.
- void setDescricao (string descricao)
  Define a descrição do item.

5.10.1 Descrição detalhada

Classe que representa um item.

A classe Item representa um item com suas caraterísticas, como nome, descrição, tipo e raridade Ela também possui métodos para acessar e modificar essas características.

5.10.2 Documentação das funções

5.10.2.1 getDescricao()

```
string Item::getDescricao ()
```

Obtém a descrição.

Retorna

string A descrição

5.10.2.2 getNome()

```
string Item::getNome ()
```

Obtém o nome do item.

Retorna

string O nome do item

5.10.2.3 getRaridade()

```
string Item::getRaridade ()
```

Obtém a raridade do item.

Retorna

string A raridade do item

5.10.2.4 getTipo()

```
string Item::getTipo ()
```

Obtém o tipo do item.

Retorna

string O tipo do item

5.10.2.5 pegarItem()

```
virtual void Item::pegarItem ()  [pure virtual]
```

Método para pegar o item.

Implementado por Cura, Estamina e Revive.

5.10.2.6 setDescricao()

```
void Item::setDescricao (
            string descricao)
```

Define a descrição do item.

Parâmetros

descricao	A descrição do item

5.10.2.7 setNome()

```
void Item::setNome (
            string nome)
```

Define o nome do item.

Parâmetros

nome	O nome do item

5.10.2.8 setRaridade()

```
void Item::setRaridade (
            string raridade)
```

Define a raridade do item.

Parâmetros

raridade	A raridade do item

5.10.2.9 setTipo()

```
void Item::setTipo (
            string tipo)
```

Define o tipo do item.

Parâmetros

tipo	O tipo do item

5.10.2.10 usarItem()

```
virtual bool Item::usarItem (
            Monstrinho * monstro)  [pure virtual]
```

Usa um item selecionado.

Parâmetros

Monstro	O monstrinho em que o item será utilizado

Retorna

true se o item foi usado com sucesso, false se não foi

Implementado por Cura, Estamina e Revive.

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

- HPPs/Item.hpp
- CPPs/Item.cpp

5.11 Referência da Classe ItemError

Diagrama de hierarquia da classe ItemError:

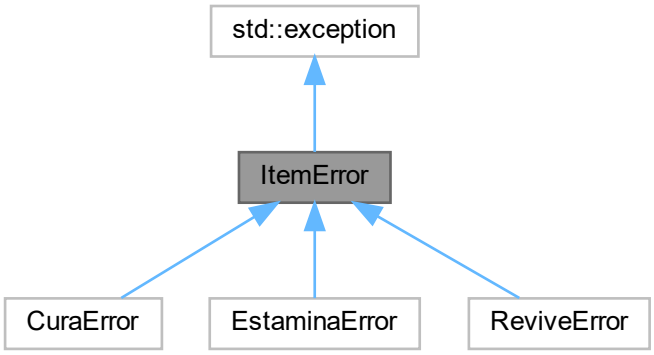
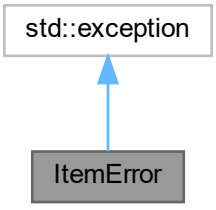


Diagrama de colaboração para ItemError:



Membros Públicos

- **ItemError** (const std::string &message)
- const char * **what** () const noexcept override

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/ItemError.hpp

5.12 Referência da Classe Jogador

Classe que representa um jogador no jogo.

`#include <Jogador.hpp>`

Diagrama de hierarquia da classe Jogador:

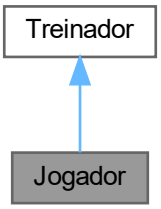
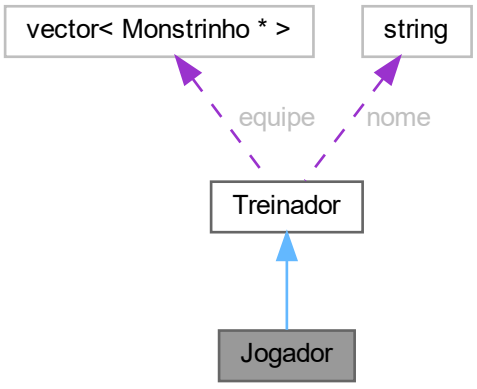


Diagrama de colaboração para Jogador:



Membros Públicos

- Jogador (int ID, string nome, vector< Monstrinho * > equipe)

  Construtor da classe Jogador.
- bool mudaEquipe () override

  Função que permite ao jogador mudar sua equipe de monstrinhos.
- void **receberItem** ()

  Função que permite o jogador receber um item aleatório ao fim de uma rodada.
- vector< Item * > **getInventario** ()

  Função que permiter pegar o inventario do Jogador.
- void adicionarItem (Item *item)

  Adiciona um Item ao inventario do jogador.
- void **removerItem** (int item)
- bool **usarItem** ()

Membros Públicos herdados de Treinador

- Treinador (int ID, std::string nome, vector< Monstrinho * > equipe)

  Construtor da classe Treinador.
- int getID ()

  Obtém o identificador do treinador.
- std::string getNome ()

  Obtém o nome do treinador.
- vector< Monstrinho * > getEquipe ()

  Obtém a equipe de monstrinhos do treinador.
- bool verificaEquipe ()

  Verifica se a equipe de monstrinhos do treinador está completa.
- void **imprimeEquipe** ()

  Imprime a equipe de monstrinhos do treinador.
- void colocaMonstrinho (Monstrinho monstrinho)

  Coloca um monstrinho na equipe do treinador.

Outros membros herdados

Atributos Protegidos herdados de Treinador

- int ID
- vector< Monstrinho * > equipe
- std::string nome

5.12.1 Descrição detalhada

Classe que representa um jogador no jogo.

Esta classe herda da classe Treinador e representa um jogador no jogo. Um jogador possui um ID, um nome e uma equipe de monstrinhos.

5.12.2 Construtores e Destrutores

5.12.2.1 Jogador()

```
Jogador::Jogador (
            int ID,
            string nome,
            vector< Monstrinho * > equipe)
```

Construtor da classe Jogador.

Parâmetros

ID	O ID do jogador.
nome	O nome do jogador.
equipe	A equipe de monstrinhos do jogador.

5.12.3 Documentação das funções

5.12.3.1 adicionarItem()

```
void Jogador::adicionarItem (
            Item * item)
```

Adiciona um Item ao inventario do jogador.

Parâmetros

item	Item adicionado no inventario

5.12.3.2 mudaEquipe()

```
bool Jogador::mudaEquipe ()  [override], [virtual]
```

Função que permite ao jogador mudar sua equipe de monstrinhos.

Esta função é uma sobrescrita da função mudaEquipe da classe Treinador. Ela permite ao jogador mudar sua equipe de monstrinhos.

Implementa Treinador.

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

- HPPs/Jogador.hpp
- CPPs/Jogador.cpp

5.13 Referência da Classe Jogo

Classe que representa toda a lógica do jogo.

```
#include <Jogo.hpp>
```

Membros Públicos

- void **iniciar** ()

  Função que inicia o jogo.
- void geraTurno (Jogador *jogador, Bot *bot)

  Função que gera um turno de batalha.
- vector< Monstrinho * > escolherMonstrinho (Jogador *jogador)

  Cria a equipe para o jogador.
- vector< Monstrinho * > criaEquipeBot (Bot *bot)

  Cria a equipe para o bot.

5.13.1 Descrição detalhada

Classe que representa toda a lógica do jogo.

A classe cria as equipes de cada treinador e gera os turnos de batalha Em cada turno você pode escolher qual monstrinho atacar e qual ataque usar, bem como trocar de monstrinho e usar itens ganhos nas vitórias

5.13.2 Documentação das funções

5.13.2.1 criaEquipeBot()

```
vector< Monstrinho * > Jogo::criaEquipeBot (
            Bot * bot)
```

Cria a equipe para o bot.

Parâmetros

bot	Ponteiro para o bot

Retorna

Retorna um vetor de ponteiros do tipo Monstrinho

5.13.2.2 escolherMonstrinho()

```
vector< Monstrinho * > Jogo::escolherMonstrinho (
            Jogador * jogador)
```

Cria a equipe para o jogador.

Parâmetros

jogador	Ponteiro para o jogador

Retorna

Retorna um vetor de ponteiros do tipo Monstrinho

5.13.2.3 geraTurno()

```
void Jogo::geraTurno (
            Jogador * jogador,
            Bot * bot)
```

Função que gera um turno de batalha.

Parâmetros

jogador	Ponteiro para o jogador
bot	Ponteiro para o bot

Retorna

void

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

- HPPs/Jogo.hpp
- CPPs/Jogo.cpp

5.14 Referência da Classe Monstrinho

Classe que representa um monstrinho.

`#include <Monstrinho.hpp>`

Membros Públicos

- Monstrinho (int ID, string nome, vector< string > tipo, string descricao, int HP, int HPAtual, int velocidade, int tier, vector< Ataque > ataques)

  Construtor da classe Monstrinho.
- int getID ()

  Obtém o ID do monstrinho.
- string getNome ()

  Obtém o nome do monstrinho.
- string getDescricao ()

  Obtém a descrição do monstrinho.
- vector< string > getTipo ()

  Obtém o tipo do monstrinho.
- int getHP ()

  Obtém a quantidade de HP do monstrinho.
- int getHPAtual ()

  Obtém a quantidade de HP atual do monstrinho.
- void setHPAtual (int HPAtual)

  Define a quantidade de HP atual do monstrinho.
- int getVelocidade ()

  Obtém a velocidade do monstrinho.
- int getTier ()

  Obtém o tier do monstrinho.
- vector< Ataque > & getAtaques ()

  Obtém os ataques do monstrinho.
- bool atacar (Monstrinho *monstroAtacado, int escolha)

  Ataca um monstrinho.
- int escolhaAtaque ()

  Escolhe um ataque para ser utilizado.

Membros públicos estáticos

- static vector< Monstrinho > construirMonstrinhos ()

  Constrói uma lista de monstrinhos a partir de um arquivo CSV.

5.14.1 Descrição detalhada

Classe que representa um monstrinho.

A classe Monstrinho representa um monstrinho com suas características, como ID, nome, descrição, tipo, HP, HP atual, velocidade e ataques. Ela também possui métodos para acessar e modificar essas características. Além disso, possui um método estático para construir uma lista de monstrinhos a partir de um arquivo CSV.

5.14.2 Construtores e Destrutores

5.14.2.1 Monstrinho()

```
Monstrinho::Monstrinho (
            int ID,
            string nome,
            vector< string > tipo,
            string descricao,
            int HP,
            int HPAtual,
            int velocidade,
            int tier,
            vector< Ataque > ataques)  [inline]
```

Construtor da classe Monstrinho.

Parâmetros

ID	O ID do monstrinho.
nome	O nome do monstrinho.
descricao	A descrição do monstrinho.
tipo	O tipo do monstrinho.
HP	A quantidade de HP do monstrinho.
HPAtual	A quantidade de HP atual do monstrinho.
velocidade	A velocidade do monstrinho.
ataques	Os ataques do monstrinho.

5.14.3 Documentação das funções

5.14.3.1 atacar()

```
bool Monstrinho::atacar (
            Monstrinho * monstroAtacado,
            int escolha)
```

Ataca um monstrinho.

Parâmetros

monstroAtacado	O monstrinho que está sendo atacado.
escolha	Valor da escolha feita pelo usuário

5.14.3.2 construirMonstrinhos()

```
vector< Monstrinho > Monstrinho::construirMonstrinhos ()  [static]
```

Constrói uma lista de monstrinhos a partir de um arquivo CSV.

Retorna

Um vetor com todos os monstrinhos do arquivo CSV.

5.14.3.3 escolhaAtaque()

```
int Monstrinho::escolhaAtaque ()
```

Escolhe um ataque para ser utilizado.

Retorna

O valor de escolha de 0 a 4

5.14.3.4 getAtaques()

```
vector< Ataque > & Monstrinho::getAtaques ()
```

Obtém os ataques do monstrinho.

Retorna

Os ataques do monstrinho.

5.14.3.5 getDescricao()

```
string Monstrinho::getDescricao ()
```

Obtém a descrição do monstrinho.

Retorna

A descrição do monstrinho.

5.14.3.6 getHP()

```
int Monstrinho::getHP ()
```

Obtém a quantidade de HP do monstrinho.

Retorna

A quantidade de HP do monstrinho.

5.14.3.7 getHPAtual()

```
int Monstrinho::getHPAtual ()
```

Obtém a quantidade de HP atual do monstrinho.

Retorna

A quantidade de HP atual do monstrinho.

5.14.3.8 getID()

```
int Monstrinho::getID ()
```

Obtém o ID do monstrinho.

Retorna

O ID do monstrinho.

5.14.3.9 getNome()

```
string Monstrinho::getNome ()
```

Obtém o nome do monstrinho.

Retorna

O nome do monstrinho.

5.14.3.10 getTier()

```
int Monstrinho::getTier ()
```

Obtém o tier do monstrinho.

Retorna

O tier do monstrinho.

5.14.3.11 getTipo()

```
vector< string > Monstrinho::getTipo ()
```

Obtém o tipo do monstrinho.

Retorna

O tipo do monstrinho.

5.14.3.12 getVelocidade()

```
int Monstrinho::getVelocidade ()
```

Obtém a velocidade do monstrinho.

Retorna

A velocidade do monstrinho.

5.14.3.13 setHPAtual()

```
void Monstrinho::setHPAtual (
            int HPAtual)
```

Define a quantidade de HP atual do monstrinho.

Parâmetros

HPAtual	A quantidade de HP atual do monstrinho.

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

- HPPs/Monstrinho.hpp
- CPPs/Monstrinho.cpp

5.15 Referência da Classe Revive

Classe que representa o item Revive.

`#include <Revive.hpp>`

Diagrama de hierarquia da classe Revive:

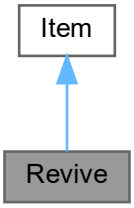
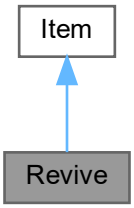


Diagrama de colaboração para Revive:



Membros Públicos

- void pegarItem () override

  Método para pegar o item.
- bool usarItem (Monstrinho ∗monstro) override

  Usa um item selecionado.

Membros Públicos herdados de Item

- string getTipo ()

  Obtém o tipo do item.
- void setTipo (string tipo)

  Define o tipo do item.

- string getRaridade ()
  Obtém a raridade do item.
- void setRaridade (string raridade)
  Define a raridade do item.
- string getNome ()
  Obtém o nome do item.
- void setNome (string nome)
  Define o nome do item.
- string getDescricao ()
  Obtém a descrição.
- void setDescricao (string descricao)
  Define a descrição do item.

5.15.1 Descrição detalhada

Classe que representa o item Revive.

5.15.2 Documentação das funções

5.15.2.1 pegarItem()

```
void Revive::pegarItem ()  [override], [virtual]
```

Método para pegar o item.

Implementa Item.

5.15.2.2 usarItem()

```
bool Revive::usarItem (
            Monstrinho * monstro)  [override], [virtual]
```

Usa um item selecionado.

Parâmetros

Monstro	O monstrinho em que o item será utilizado

Retorna

true se o item foi usado com sucesso, false se não foi

Implementa Item.

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

- HPPs/Revive.hpp
- CPPs/Revive.cpp

5.16 Referência da Classe ReviveError

Diagrama de hierarquia da classe ReviveError:

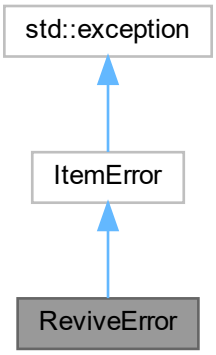
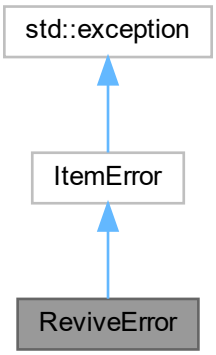


Diagrama de colaboração para ReviveError:



Membros Públicos

- **ReviveError** (const std::string &message)

Membros Públicos herdados de ItemError

- **ItemError** (const std::string &message)
- const char * **what** () const noexcept override

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- ERR/ReviveError.hpp

5.17 Referência da Classe Treinador

Classe que representa um treinador de monstrinhos.

`#include <Treinador.hpp>`

Diagrama de hierarquia da classe Treinador:

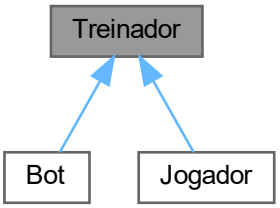
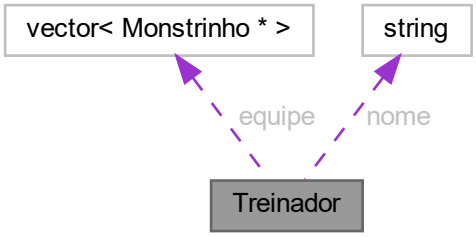


Diagrama de colaboração para Treinador:



Membros Públicos

- Treinador (int ID, std::string nome, vector< Monstrinho * > equipe)
  Construtor da classe Treinador.
- int getID ()
  Obtém o identificador do treinador.
- std::string getNome ()
  Obtém o nome do treinador.
- vector< Monstrinho * > getEquipe ()
  Obtém a equipe de monstrinhos do treinador.
- bool verificaEquipe ()
  Verifica se a equipe de monstrinhos do treinador está completa.
- virtual bool mudaEquipe ()=0

Método virtual para mudar a equipe de monstrinhos do treinador.

- void **imprimeEquipe** ()

  Imprime a equipe de monstrinhos do treinador.

- void colocaMonstrinho (Monstrinho monstrinho)

  Coloca um monstrinho na equipe do treinador.

Atributos Protegidos

- int ID
- vector< Monstrinho * > equipe
- std::string nome

5.17.1 Descrição detalhada

Classe que representa um treinador de monstrinhos.

A classe Treinador possui um identificador, um nome e uma equipe de monstrinhos. Ela fornece métodos para obter o identificador, o nome e a equipe do treinador, verificar se a equipe está completa, mudar a equipe e imprimir a equipe.

5.17.2 Construtores e Destrutores

5.17.2.1 Treinador()

```
Treinador::Treinador (
            int ID,
            std::string nome,
            vector< Monstrinho * > equipe)
```

Construtor da classe Treinador.

Parâmetros

ID	O identificador do treinador.
nome	O nome do treinador.
equipe	A equipe de monstrinhos do treinador.

5.17.3 Documentação das funções

5.17.3.1 colocaMonstrinho()

```
void Treinador::colocaMonstrinho (
            Monstrinho monstrinho)
```

Coloca um monstrinho na equipe do treinador.

Parâmetros

monstrinho	O monstrinho a ser colocado na equipe.

5.17.3.2 getEquipe()

```
vector< Monstrinho * > Treinador::getEquipe ()
```

Obtém a equipe de monstrinhos do treinador.

Retorna

A equipe de monstrinhos do treinador.

5.17.3.3 getID()

```
int Treinador::getID ()
```

Obtém o identificador do treinador.

Retorna

O identificador do treinador.

5.17.3.4 getNome()

```
string Treinador::getNome ()
```

Obtém o nome do treinador.

Retorna

O nome do treinador.

5.17.3.5 mudaEquipe()

```
virtual bool Treinador::mudaEquipe ()  [pure virtual]
```

Método virtual para mudar a equipe de monstrinhos do treinador.

Este método deve ser implementado nas classes derivadas.

Implementado por Bot e Jogador.

5.17.3.6 verificaEquipe()

```
bool Treinador::verificaEquipe ()
```

Verifica se a equipe de monstrinhos do treinador está completa.

Retorna

true se a equipe estiver completa, false caso contrário.

5.17.4 Atributos

5.17.4.1 equipe

```
vector<Monstrinho *> Treinador::equipe  [protected]
```

Equipe de monstrinhos do treinador

5.17.4.2 ID

```
int Treinador::ID  [protected]
```

Identificador do treinador

5.17.4.3 nome

```
std::string Treinador::nome  [protected]
```

Nome do treinador

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

- HPPs/Treinador.hpp
- CPPs/Treinador.cpp

Capítulo 6

Arquivos

6.1 AtaqueError.hpp

```
00001 #ifndef ATAQUEERRO
00002 #define ATAQUEERRO
00003 #include <stdexcept>
00004
00005 class AtaqueError : public std::runtime_error{
00006     public:
00007         AtaqueError(const std::string& message): std::runtime_error(message){};
00008 };
00009
00010 #endif
```

6.2 BadRequestError.hpp

```
00001 #ifndef BADREQUESTERRO
00002 #define BADREQUESTERRO
00003
00004 #include <stdexcept>
00005
00006 class BadRequestError : public std::runtime_error{
00007     public:
00008         BadRequestError(const std::string& message): std::runtime_error(message){};
00009 };
00010
00011
00012 #endif
```

6.3 CuraError.hpp

```
00001 #ifndef CURAERRO
00002 #define CURAERRO
00003
00004 #include <stdexcept>
00005 #include <string>
00006 #include "ItemError.hpp"
00007
00008 class CuraError : public ItemError {
00009 public:
00010     CuraError(const std::string& message)
00011         : ItemError(message) {}
00012 };
00013
00014 #endif
```

6.4 EscolhaError.hpp

```
00001 #ifndef ESCOLHAERRO
00002 #define ESCOLHAERRO
00003 #include <stdexcept>
00004
00005 class EscolhaError : public std::invalid_argument{
00006     public:
00007         EscolhaError(const std::string& message): std::invalid_argument(message){};
00008 };
00009
00010
00011
00012 #endif
```

6.5 EstaminaError.hpp

```
00001 #ifndef ESTAMINAERRO
00002 #define ESTAMINAERRO
00003
00004 #include <stdexcept>
00005 #include <string>
00006 #include "ItemError.hpp"
00007
00008 class EstaminaError : public ItemError {
00009 public:
00010     EstaminaError(const std::string& message)
00011         : ItemError(message) {}
00012 };
00013
00014 #endif
```

6.6 ItemError.hpp

```
00001 #ifndef ITEMERRO
00002 #define ITEMERRO
00003
00004 #include <stdexcept>
00005 #include <string>
00006
00007 class ItemError : public std::exception{
00008 public:
00009     explicit ItemError(const std::string& message)
00010         : msg_(message) {}
00011     const char* what() const noexcept override {
00012     return msg_.c_str();
00013 }
00014     private:
00015         std::string msg_;
00016 };
00017
00018 #endif
```

6.7 ReviveError.hpp

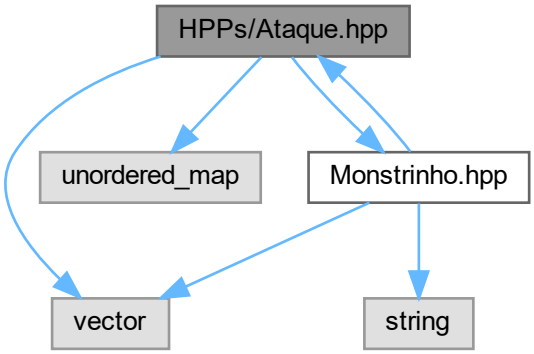
```
00001 #ifndef REVIVEERRO
00002 #define REVIVEERRO
00003
00004 #include <stdexcept>
00005 #include <string>
00006 #include "ItemError.hpp"
00007
00008 class ReviveError : public ItemError {
00009 public:
00010     ReviveError(const std::string& message)
00011         : ItemError(message) {}
00012 };
00013
00014 #endif
```

6.8 Referência do Arquivo HPPs/Ataque.hpp

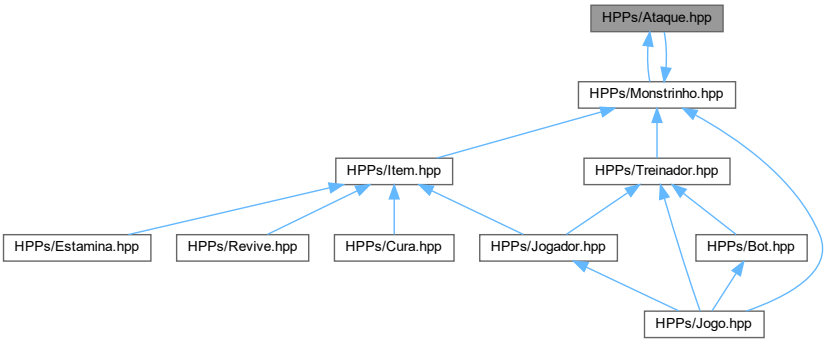
Definição da classe Ataque e seus métodos.

```
#include <vector>
#include <unordered_map>
#include "Monstrinho.hpp"
```

Gráfico de dependência de inclusões para Ataque.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Ataque

  Representa um ataque em um jogo.

6.8.1 Descrição detalhada

Definição da classe Ataque e seus métodos.

6.9 Ataque.hpp

Ir para a documentação desse arquivo.

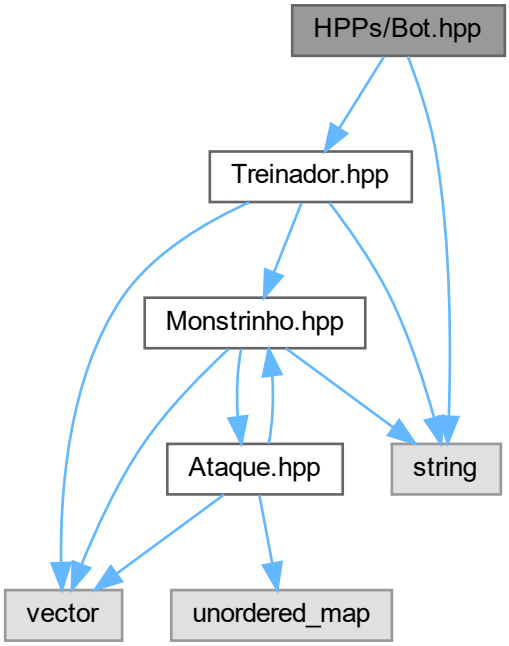
```
00001 #ifndef ATAQUE_HPP
00002 #define ATAQUE_HPP
00003
00004 #include <vector>
00005 #include <unordered_map>
00006 #include "Monstrinho.hpp"
00007
00008 using namespace std;
00014 class Monstrinho; // Declaração antecipada da classe Monstrinho para evitar dependência circular
00015
00022 class Ataque {
00023 private:
00024     int ID;
00025     string nome;
00026     string tipo;
00027     int dano;
00028     string descricao;
00029     int quantidade;
00030     int quantidadeAtual;
00031     double chanceAcerto;
00032     static unordered_map<string, unordered_map<string, double» TabelaEfetividade;
00033     static unordered_map<string, unordered_map<string, double» gerarTabelaEfetividade();
00034
00035
00036 public:
00048     Ataque(int ID, string nome, string tipo, int dano, string descricao, int quantidade, double
      chanceAcerto)
00049         : ID(ID), nome(nome), tipo(tipo), dano(dano), descricao(descricao), quantidade(quantidade),
      quantidadeAtual(quantidade), chanceAcerto(chanceAcerto) {}
00050
00056     int getID();
00057
00063     string getNome();
00064
00070     string getTipo();
00071
00077     int getDano();
00078
00084     string getDescricao();
00085
00091     int getQuantidade();
00092
00098     int getQuantidadeAtual();
00099
00105     void setQuantidadeAtual(int valor);
00111     double getChanceAcerto();
00112
00118     static vector<Ataque> construirAtaques();
00119
00126     bool fazerAtaque(Monstrinho &inimigo);
00127
00135     static double calcularEfetividade(string tipoAtaque, vector<string> tiposMonstrinho);
00136 };
00137
00138
00139 #endif
```

6.10 Referência do Arquivo HPPs/Bot.hpp

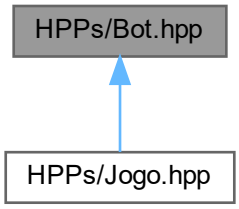
Uma Classe herdando da Classe Treinador que representa o Bot.

```
#include "Treinador.hpp"
#include <string>
```

Gráfico de dependência de inclusões para Bot.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Bot

  Representa um Bot em um jogo.

6.10.1 Descrição detalhada

Uma Classe herdando da Classe Treinador que representa o Bot.

6.11 Bot.hpp

Ir para a documentação desse arquivo.

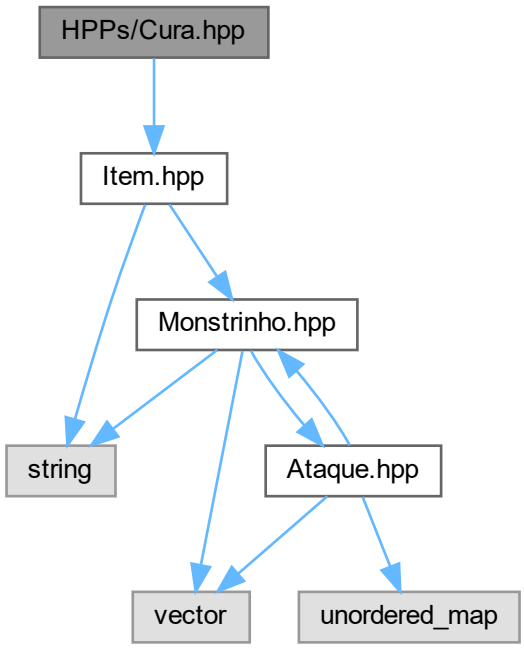
```
00001 #ifndef BOT_HPP
00002 #define BOT_HPP
00003
00004 #include "Treinador.hpp"
00005 #include <string>
00006
00019 class Bot : public Treinador
00020 {
00021 private:
00022     vector<std::string> fala;
00024 public:
00033     Bot(int ID, std::string nome, vector<Monstrinho *> equipe, vector<std::string> fala);
00034
00040     vector<std::string> getFala();
00041
00047     bool mudaEquipe() override;
00048
00049 };
00050 ;
00051
00052 #endif
```

6.12 Referência do Arquivo HPPs/Cura.hpp

Uma Classe herdando da Classe Item que representa o item Cura.

`#include "Item.hpp"`

Gráfico de dependência de inclusões para Cura.hpp:



Componentes

- class Cura

  Classe que representa o item Cura.

6.12.1 Descrição detalhada

Uma Classe herdando da Classe Item que representa o item Cura.

6.13 Cura.hpp

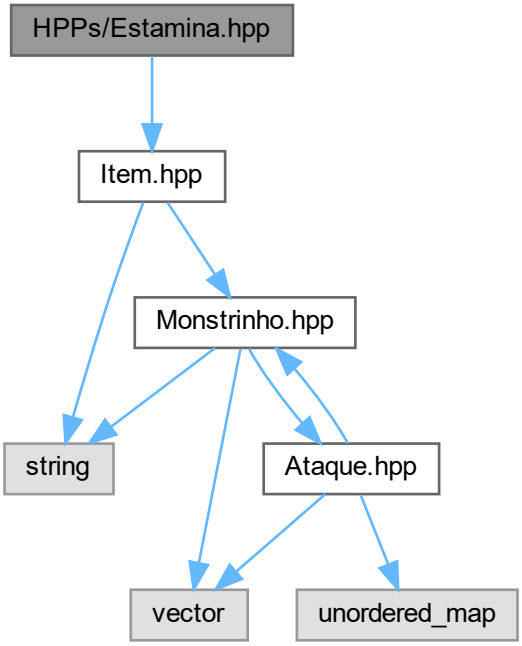
Ir para a documentação desse arquivo.

```
00001 #ifndef CURA
00002 #define CURA
00003
00004 #include "Item.hpp"
00005
00017 class Cura: public Item{
00018     private:
00019         int cura;
00020     public:
00024         void pegarItem() override;
00025
00033         bool usarItem(Monstrinho* monstro) override;
00034
00039         void setCura(int cura);
00040
00045         int getCura();
00046 };
00047
00048 #endif
```

6.14 Referência do Arquivo HPPs/Estamina.hpp

Uma Classe herdando da Classe Item que representa o item Estamina.

`#include "Item.hpp"`
Gráfico de dependência de inclusões para Estamina.hpp:



Componentes

- class Estamina

  Classe que representa o item Estamina.

6.14.1 Descrição detalhada

Uma Classe herdando da Classe Item que representa o item Estamina.

6.15 Estamina.hpp

Ir para a documentação desse arquivo.

```
00001 #ifndef ESTAMINA
00002 #define ESTAMINA
00003
00004 #include "Item.hpp"
00005
00018 class Estamina : public Item{
00019     private:
00020         int energia;
00021     public:
00025         void pegarItem() override;
00026
00027
00028         int getEnergia();
```

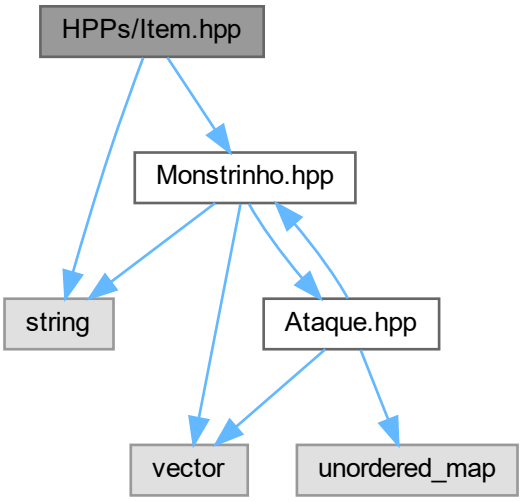
```
00029
00030
00031         void setEnergia(int energia);
00032
00040         bool usarItem(Monstrinho* monstro) override;
00041 };
00042
00043 #endif
```

6.16 Referência do Arquivo HPPs/Item.hpp

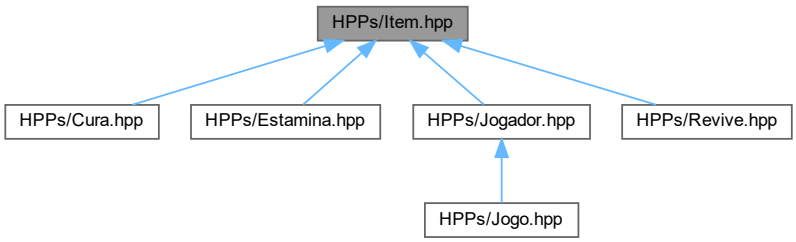
Classe que representa um item.

```
#include "Monstrinho.hpp"
#include <string>
```

Gráfico de dependência de inclusões para Item.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Item

  Classe que representa um item.

6.16.1 Descrição detalhada

Classe que representa um item.

6.17 Item.hpp

Ir para a documentação desse arquivo.

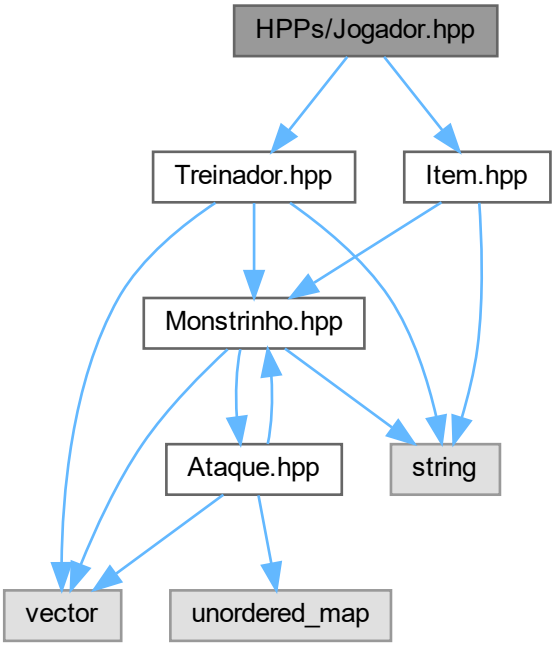
```
00001 #ifndef ITEM
00002 #define ITEM
00003
00004 #include "Monstrinho.hpp"
00005 #include <string>
00006 using std::string;
00007
00021 class Item{
00022     private:
00023         string nome;
00024         string descricao;
00025         string tipo;
00026         string raridade;
00027     public:
00035         virtual bool usarItem(Monstrinho* monstro) = 0; //mudar dependendo se for ponteiro ou não
00036
00041         string getTipo();
00042
00047         void setTipo(string tipo);
00048
00052         virtual void pegarItem() = 0;
00053
00058         string getRaridade();
00059
00064         void setRaridade(string raridade);
00065
00070         string getNome();
00071
00076         void setNome(string nome); // <= Muda o nome
00077
00082         string getDescricao();
00083
00088         void setDescricao(string descricao); //<= Muda a descrição
00089
00090
00091 };
00092
00093 #endif
```

6.18 Referência do Arquivo HPPs/Jogador.hpp

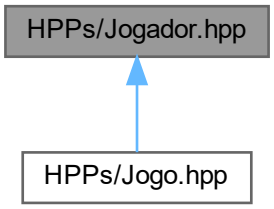
Uma Classe herdando da Classe Treinador que representa o Jogador.

```
#include "Treinador.hpp"
#include "Item.hpp"
```

Gráfico de dependência de inclusões para Jogador.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Jogador

  Classe que representa um jogador no jogo.

6.18.1 Descrição detalhada

Uma Classe herdando da Classe Treinador que representa o Jogador.

6.19 Jogador.hpp

Ir para a documentação desse arquivo.

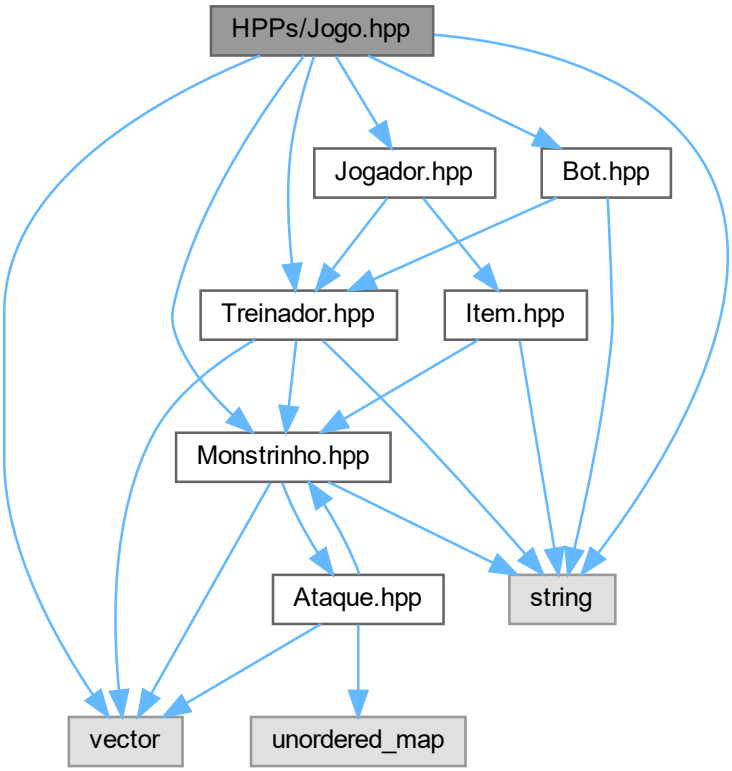
```
00001 #ifndef JOGADOR_HPP
00002 #define JOGADOR_HPP
00003
00004 #include "Treinador.hpp"
00005 #include "Item.hpp"
00006
00018 class Jogador : public Treinador
00019 {
00020 private:
00021     // Implementar inventário
00022     std::vector<Item*> inventario;
00023 public:
00031     Jogador(int ID, string nome, vector<Monstrinho *> equipe);
00032
00039     bool mudaEquipe() override;
00043     void receberItem();
00044
00048     vector<Item*> getInventario();
00049
00055     void adicionarItem(Item* item);
00056     void removerItem(int item);
00057     bool usarItem();
00058 };
00059
00060 #endif
```

6.20 Referência do Arquivo HPPs/Jogo.hpp

Definição da classe Jogo e seus métodos.

```
#include <string>
#include <vector>
#include "Monstrinho.hpp"
#include "Treinador.hpp"
#include "Jogador.hpp"
#include "Bot.hpp"
```

Gráfico de dependência de inclusões para Jogo.hpp:



Componentes

- class Jogo

  Classe que representa toda a lógica do jogo.

6.20.1 Descrição detalhada

Definição da classe Jogo e seus métodos.

6.21 Jogo.hpp

Ir para a documentação desse arquivo.

```
00001 #ifndef JOGO_HPP
00002 #define JOGO_HPP
00003
00004 #include <string>
00005 #include <vector>
00006 #include "Monstrinho.hpp"
00007 #include "Treinador.hpp"
00008 #include "Jogador.hpp"
00009 #include "Bot.hpp"
```

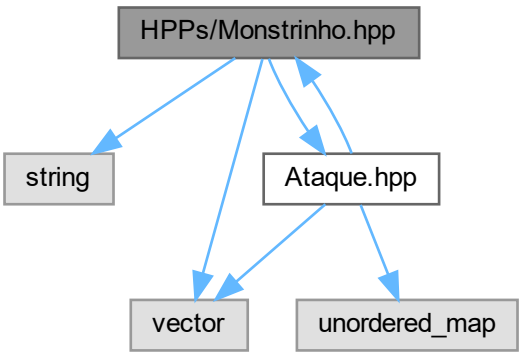
```
00010
00011 using std::vector;
00012 using std::string;
00013
00027 class Jogo{
00028     public:
00033         void iniciar();
00034
00042         void geraTurno(Jogador* jogador, Bot* bot);
00043
00050         vector<Monstrinho*> escolherMonstrinho(Jogador* jogador);
00051
00058         vector<Monstrinho*> criaEquipeBot(Bot* bot);
00059
00060
00061
00062 };
00063
00064 #endif
```

6.22 Referência do Arquivo HPPs/Monstrinho.hpp

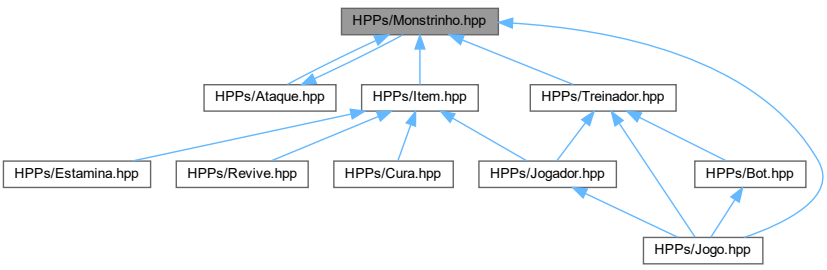
Definição da classe Monstrinho e seus métodos.

```
#include <string>
#include <vector>
#include "Ataque.hpp"
```

Gráfico de dependência de inclusões para Monstrinho.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Monstrinho

  Classe que representa um monstrinho.

6.22.1 Descrição detalhada

Definição da classe Monstrinho e seus métodos.

6.23 Monstrinho.hpp

Ir para a documentação desse arquivo.

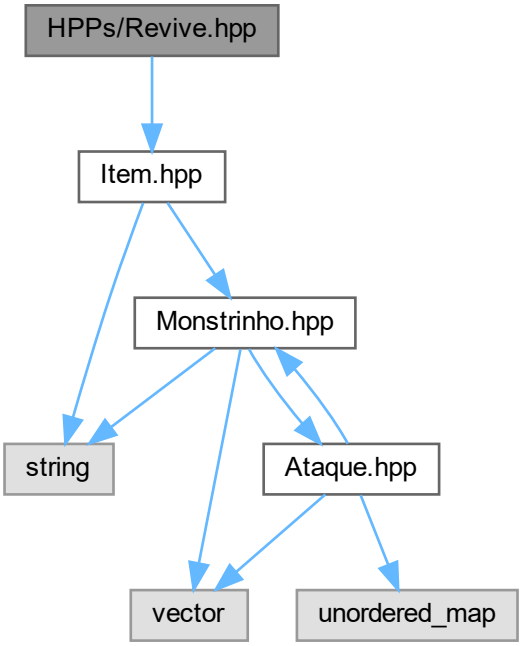
```
00001 #ifndef MONSTRINHO_HPP
00002 #define MONSTRINHO_HPP
00003
00004 #include <string>
00005 #include <vector>
00006 #include "Ataque.hpp"
00007
00008 using namespace std;
00009
00015 class Ataque; // Declaração antecipada da classe Ataque para evitar dependência circular.
00016
00026 class Monstrinho {
00027 private:
00028     int ID;
00029     string nome;
00030     string descricao;
00031     vector<string> tipo;
00032     int HP;
00033     int HPAtual;
00034     int velocidade;
00035     int tier;
00036     vector<Ataque> ataques;
00037
00038 public:
00052     Monstrinho(int ID, string nome, vector<string> tipo,string descricao, int HP, int HPAtual, int
      velocidade, int tier, vector<Ataque> ataques)
00053         : ID(ID), nome(nome), descricao(descricao), tipo(tipo), HP(HP), HPAtual(HPAtual),
      velocidade(velocidade), tier(tier), ataques(ataques) {}
00054
00060     int getID();
00061
00067     string getNome();
00068
00074     string getDescricao();
00075
00081     vector<string> getTipo();
00082
00088     int getHP();
00089
00095     int getHPAtual();
00096
00102     void setHPAtual(int HPAtual);
00103
00109     int getVelocidade();
00110
00116     int getTier();
00117
00123     vector<Ataque>& getAtaques();
00124
00131     bool atacar(Monstrinho* monstroAtacado, int escolha);
00132
00138     static vector<Monstrinho> construirMonstrinhos();
00139
00145     int escolhaAtaque();
00146 };
00147
00148 #endif
```

6.24 Referência do Arquivo HPPs/Revive.hpp

Uma Classe herdando da Classe Item que representa o item Revive.

```
#include "Item.hpp"
```

Gráfico de dependência de inclusões para Revive.hpp:



Componentes

- class Revive

  Classe que representa o item Revive.

6.24.1 Descrição detalhada

Uma Classe herdando da Classe Item que representa o item Revive.

6.25 Revive.hpp

Ir para a documentação desse arquivo.

```
00001 #ifndef REVIVE
00002 #define REVIVE
00003 #include "Item.hpp"
00004
00016 class Revive : public Item{
00017     private:
```

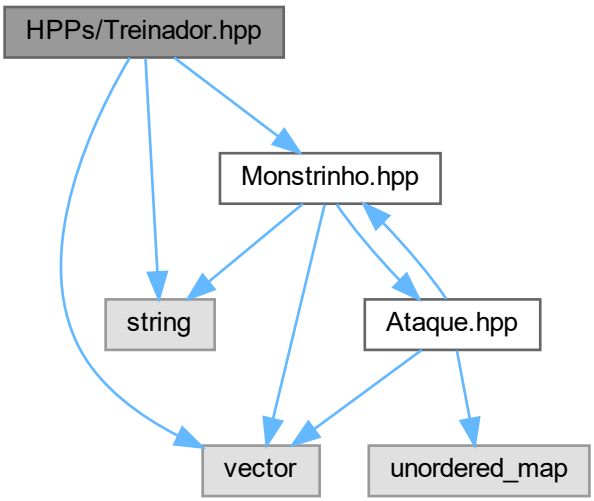
```
00018         string raridade;
00019     public:
00023         void pegarItem() override;
00024
00032         bool usarItem(Monstrinho* monstro) override;
00033 };
00034
00035
00036
00037
00038 #endif
```

6.26 Referência do Arquivo HPPs/Treinador.hpp

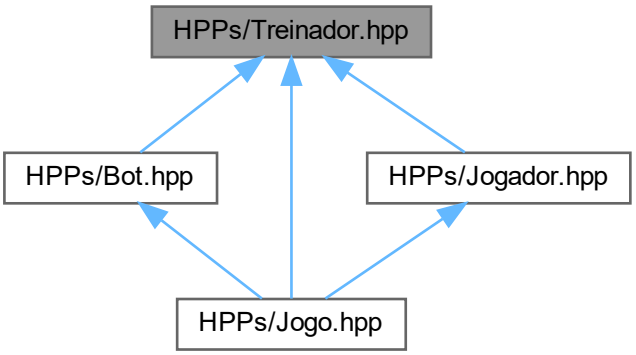
Definição da classe Treinador e seus métodos.

```
#include <string>
#include <vector>
#include "Monstrinho.hpp"
```

Gráfico de dependência de inclusões para Treinador.hpp:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Componentes

- class Treinador

  Classe que representa um treinador de monstrinhos.

6.26.1 Descrição detalhada

Definição da classe Treinador e seus métodos.

6.27 Treinador.hpp

Ir para a documentação desse arquivo.

```
00001 #ifndef TREINADOR_HPP
00002 #define TREINADOR_HPP
00003
00004 #include <string>
00005 #include <vector>
00006 #include "Monstrinho.hpp"
00007
00022 class Treinador
00023 {
00024 protected:
00025     int ID;
00026     vector<Monstrinho *> equipe;
00027     std::string nome;
00029 public:
00037     Treinador(int ID, std::string nome, vector<Monstrinho *> equipe);
00038
00044     int getID();
00045
00051     std::string getNome();
00052
00058     vector<Monstrinho *> getEquipe();
00059
00065     bool verificaEquipe();
00066
00072     virtual bool mudaEquipe() = 0;
00073
00077     void imprimeEquipe();
00078
00084     void colocaMonstrinho(Monstrinho monstrinho);
00085 };
00086
00087 #endif
```