|  |
| --- |
| **Especialização em IA para Engenharia de Testes de Software - IARTES** |
| Caroline do Amaral ● Hassan Said ● Mauricio Calheiro ● Uilhans Oliveira |

**Projeto Final de Orientação a Objetos com Python**

# Introdução

O contexto deste trabalho é o mundo dos jogos de RPG, onde os jogadores estão imersos em mundos de fantasia. O foco principal são os personagens, suas características e ações.

Foram separados dois grupos de personagens: os heróis (aqueles que o jogador controla) e os monstros (os inimigos). Representando os heróis, temos guerreiros e magos, enquanto os antagonistas são compostos por orcs e dragões.

Figura : Representação dos personagens do jogo.

|  |  |  |  |
| --- | --- | --- | --- |
| Human Fighter Warrior - Pathfinder PFRPG DND D&D d20 fantasy | Classes de  rpg, Cavaleiros medievais, Medieval rpg  Guerreiro | Wizard – d20PFSRD  Mago | Orc | Dragão |

Essencialmente, o jogo consiste em explorar o mapa do mundo e travar batalhas. A mecânica dos confrontos é baseada em turnos, onde cada personagem e inimigo faz uma ação a cada turno, podendo atacar, defender e realizar outras ações que serão melhor descritas mais à frente.

# Hierarquia

A classe Personagem é abstrata e possui os atributos nome e nível. O nível inicial de um personagem é 1 e o máximo é 10. Além disso, possui alguns métodos abstratos, que devem ser implementados em suas subclasses: atacar, defender, lutar e mover.

A classe Heroi herda personagem e possui alguns atributos e métodos a mais (descritos a seguir). Além disso, possui os *getters* e *setters* dos atributos.

Tabela : Descrição da classe Heroi

|  |  |
| --- | --- |
| Heroi | |
| Atributos | **raca**: indica a raça do herói, visto que esta é uma escolha comum neste tipo de jogo.  Alguns exemplos são: humano, elfo, anão, mas este atributo aceita qualquer *string*, portanto o jogador é livre para decidir sua raça conforme preferencias. |
| **atk** (ataque): quantidade de dano que causará ao inimigo.  Valor inicial é 1, para um herói nível 1. E aumenta 0.50 a cada evolução |
| **hp\_max** (saúde): quantidade de dano que o herói pode sofrer até ser derrotado em batalha.  Valor inicial é 10 para um herói nível 1. E aumenta 1 a cada evolução. |
| **hp** (saúde): é o estado atual da saúde do herói.  O valor inicial é igual a hp\_max e diminui a cada dano sofrido. Sempre que o herói evolui, todo o hp é restaurado. |
| **nome** e **nível** são atributos que todo personagem tem. Já explanados na classe Personagem. |
| Métodos | **atacar**: ataca um inimigo se o sucesso do ataque estiver no intervalo de 6 a 10.  O hp do inimigo diminui o valor do atk do herói.  No jogo, o sucesso seria gerado aleatoriamente sempre que necessário, com **randint(0, 10)**, simulando um lançamento de dados. |
| **defender**: defende-se de um ataque do inimigo se o sucesso da defesa estiver no intervalo de 7 a 10.  O hp do heroi diminui o valor do atk do inimigo. |
| **evoluir**: evolui o herói, melhorando seus atributos.  Nível é acrescido de 1;  Ataque é acrescido de 0.50;  Hp\_max é acrescido de 1;  Hp é restaurado até o novo valor máximo. |
| **fugir**: o herói pode fugir da batalha se o sucesso da fuga estiver no intervalo entre o nível do inimigo e 10.  Não é possível fugir de 'chefes’, independente no nível. |
| **lutar**: lutar chama os métodos atacar, defender e fugir dependendo da ação escolhida.  A ação é escolhida pela entrada do usuário e pode ser 1, 2 ou 3. Caso seja diferente disso, o método lutar é chamado novamente. |
| **mover**: move o personagem pelo mundo, através de pares de coordenadas.  A ideia é que o par de coordenadas sejam obtidos por um click no local para onde deseja se deslocar. |

A classe Monstro também herda personagem, mas possui algumas diferenças com relação a Heroi. Seus atributos são calculados de maneira diferente e ao invés de fugir, o monstro pode curar-se (restaurar saúde).

Tabela 2: Descrição da classe Monstro

|  |  |
| --- | --- |
| Heroi | |
| Atributos | **tipo**: monstros podem ser inimigos comuns ou chefes. |
| **atk** (ataque): quantidade de dano que causará ao inimigo.  Valor inicial é **nível\*0.50**. |
| **hp\_max** (saúde): quantidade de dano que o herói pode sofrer até ser derrotado em batalha.  Valor inicial é **10 + nivel**. |
| **hp** (saúde): é o estado atual da saúde do herói.  O valor inicial é igual a hp\_max e diminui a cada dano sofrido. |
| **nome** e **nível** são atributos que todo personagem tem.  Contudo, o nível de um monstro é especificado quando é instanciado, não sendo 1 por padrão. |
| Métodos | **atacar**: similar ao de herói, mas com intervalo de 5 a 10. |
| **defender**: similar ao de herói, mas com intervalo de 6 a 10. |
| **curar**: restaura a o HP do monstro em 25% do seu hp\_max.  Se o novo HP for maior que o máximo, é restaurado somente até o máximo. |
| **lutar**: similar ao de herói, mas a ação seria gerada aleatoriamente com **randint(0, 10)**, pois não é possível controlar o monstro. |
| **mover**: similar ao de herói, mas a escolha da coordenada seria calculada pelo computador, pois não é possível controlar o monstro. |

# Cobertura do código

Quando o código estava 50% pronto...

Quando o código estava 80% pronto...

Quando o código estava 100% pronto...

# Conclusões

Aplicar TDD torna a validação do código mais simples e garante alta cobertura.

Alguns testes precisaram ser reescritos ao fazer refatoracao no código, pois percebemos uma maneira melhor de codificar as classes (adicionando atributos e modificando métodos).

As classes finais foram mais fáceis de escrever e testar apresentam menos código, uma vez que herdam muitas coisas das demais.