Programação em Linguagem Java

Projeto Intercalar - Rogue - Mais informações

Outubro/Novembro 2020

1 Introdução

De forma a complementar e melhorar o enunciado projecto disponibilizado, este documento clarifica alguns detalhes e apresenta as funcionalidades necessários para cada um dos níveis de avaliação.

2 Movimentação das personagens

No pacote inicial distribuído uma das personagens existentes é o "Thief" esta personagem deve ter uma movimentação especifica sendo semelhante à peça de xadrez do bispo, ou seja, a sua movimentação deve ser na diagonal.

3 Lançamento de Bolas de Fogo

Interface `FireTile` descreve dois métodos: o método `setPosition()` e o `validateImpact()` que é chamado cada vez que a bola de fogo se move. Este método deve ser responsável por ver se a bola de fogo atingiu algum objeto e fazê-la "explodir". Também deve ser responsável por fazer o inimigo perder a vida, caso a bola de fogo o atinja.

`FireBallThread` recebe uma direção e uma `FireTile` e tem como objetivo mudar a posição da bola de fogo de 300 ms em 300 ms. Cada vez que a bola de fogo se move, o método `validateImpact()` da `FireTile` é invocado. Caso o método `validateImpact()` devolva `true`, então o processo continua normalmente. Caso devolva `false`, o ciclo termina e a bola de fogo é eliminada da Gui.

4 Avaliação

O trabalho será classificado com A, B ou C. A avaliação será feita do seguinte modo:

Serão **excluídos** (não aceites para discussão e classificados com zero valores) os trabalhos que:

- Apresentem erros de sintaxe;
- Apresentem um funcionamento muito limitado, sem que componentes essenciais do projeto estejam implementadas. Por exemplo, se a gestão de colisões dos objetos não está implementada é considerado que o trabalho não está utilizável;
- Não é utilizada herança (por exemplo no comportamento e características das criaturas) e coleções (por exemplo para guardar os vários objetos presentes numa cena);
- O comportamento de todo do programa é concentrado num método, numa classe ou não demonstrem que sabem usar corretamente classes e objetos para modularizar o código.

Serão classificados com **C**, os trabalhos que:

- Forem jogáveis numa sala, com uma personagem que se move de acordo com as decisões do utilizador e (pelo menos) dois tipos diferentes de criaturas que exibem algum tipo de comportamento diferente;
- Usem corretamente classes, herança e coleções;
- O comportamento do programa não está maioritariamente concentrado num ou dois métodos / classes.

Serão classificados com **B**, os trabalhos com uma estrutura interna razoável, nomeadamente:

- Classes desenhadas de acordo com as boas práticas;
  - utilização das coleções adequadas para cada situação;
  - boa proteção dos atributos de cada classe;
  - métodos de tamanho razoável com um único objetivo;
- Que permitam, em geral, jogar um jogo até ao fim, em que a personagem tenha meios de causar dano às criaturas à distância, com pelo mais de duas salas (idealmente deveria ter cerca de seis salas em dois níveis diferentes).

Serão classificados com **A** os trabalhos que:

- Tenham uma estrutura interna que, além dos pontos anteriores, permita a implementação do jogo de modo realista (i.e. que resista bem ao escalamento para muitas salas e níveis distintos). A demonstração não precisa de ter mais de seis salas em dois níveis;
- Gravem e recuperem todas as salas por onde o personagem passou até ao momento, bem como o estado do jogo em cada sala, nomeadamente a saúde das criaturas em jogo;
- Gravem e carreguem de um ficheiro as pontuações máximas alcançadas no jogo.
- Devem saber usar adequadamente exceções para tratar erros de programação e situações de exceção;
- Elementos extra.