**INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA**

Engenharia Informática e de Computadores

**Computação Gráfica**

**1º Trabalho Prático**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Membros do Grupo** | | | | | |
| **31768** | **Ricardo Romão** | **31923** | **Nuno Sousa** | **26657** | **Ricardo Neto** |

**Batalha Naval**

**Regras do Jogo**

O jogo consiste numa “batalha” entre 2 jogadores em que, o primeiro a afundar a frota do adversário é o vencedor. Um dos jogadores é “humano” e outro representado pelo computador.

O primeiro passo para se dar inicio ao jogo é proceder à colocação dos barcos no tabuleiro de jogo respeitando as seguintes quantidades:

|  |  |  |  |
| --- | --- | --- | --- |
| Representação Gráfica | Nome | Nº de Canos | Quantidade |
|  | *Aircraft Carrier* | 5 | 1 |
|  | *BattleShip* | 4 | 1 |
|  | *Destroyer* | 3 | 2 |
|  | *Patrol Boat* | 2 | 3 |
|  | *Submarine* | 1 | 4 |

*Pontos de âncora marcados a amarelo \**

Além das regras em cima, que validam a quantidade de barcos presentes no tabuleiro por tipo, a colocação de peças também respeita outras regras, nomeadamente, não podem haver barcos posicionados sem uma quadricula de distância de outro barco e, naturalmente, os barcos terão que estar nos limites do tabuleiro.

O tabuleiro de jogo apresenta-se numa estrutura quadricular, sendo as linhas identificadas por letras e as colunas por números, conforme segue:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| alnilam_camouflage_2.pngA |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |
| E |  |  |  |  |  |  |  |  |  |  |
| F |  |  |  |  |  |  |  |  |  |  |
| G |  |  |  |  |  |  |  |  |  |  |
| H |  |  |  |  |  |  |  |  |  |  |
| I |  |  |  |  |  |  |  |  |  |  |
| J |  |  |  |  |  |  |  |  |  |  |

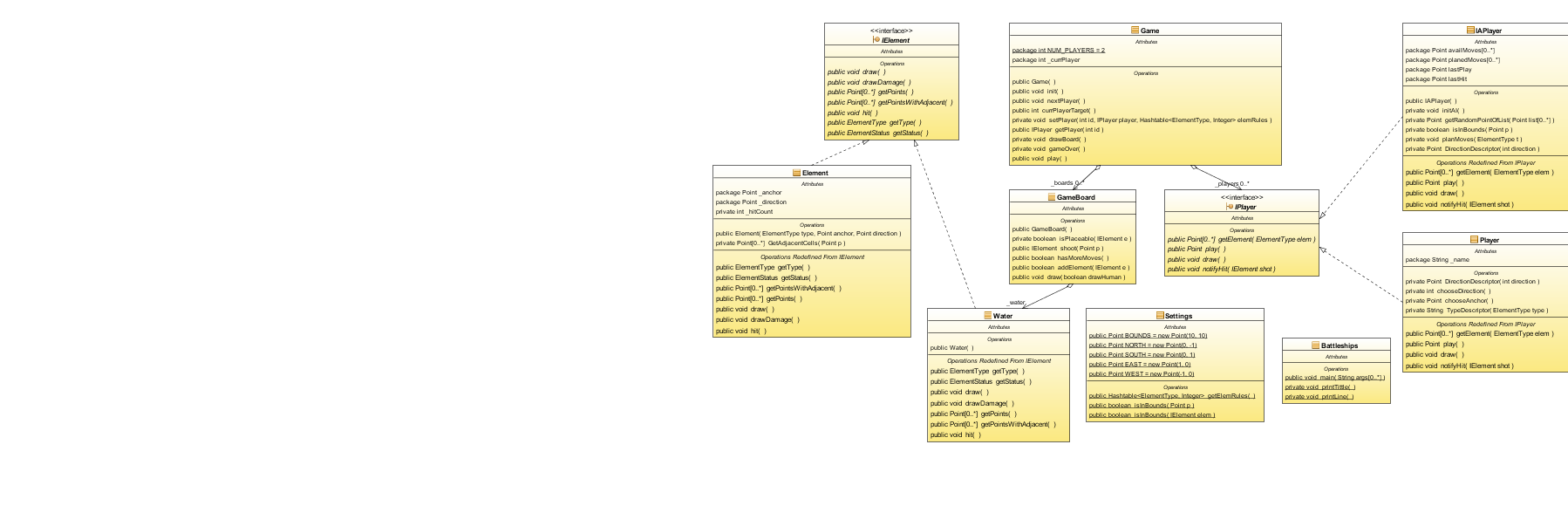
Após a fase de definição do tabuleiro, o jogo está pronto a começar.

A acção de jogo é mediada pelo gestor do jogo. Esta entidade será a responsável por solicitar ao jogador uma coordenada para a jogada, efectivando-a no tabuleiro do adversário e, retornando informação acerca de qual o alvo atingido.

O processo de jogo é feito de igual forma para ambos os jogadores, i.e., é solicitado um ponto para a jogada e é devovida informação aceca do resultado da mesma. Esta informação devolvida pelo gestor de jogo é crucial para o jogador do computador, uma vez que, as suas próximas jogadas são calculadas com base no resultado da anterior.

O jogo termina quando num dos tabuleiros não existirem mais elementos sem danos, apresentando-se todos afundados.

**Diagrama de Classes**



Todas as classes e respectivas funcionalidades, encontram-se detralhadas no formato Javadoc, anexado a este documento.

**Opções de Implementação**

**Referências para Tabuleiros**

Apesar de um tabuleiro pertencer a um jogador, considerou-se que Game é quem guarda referências para tabuleiros, associando-os a jogadores.

Desta forma, os jogadores “montam” um tabuleiro de jogo e jogam sobre o tabuleiro do outro jogador, sem que tenham a responsabilidade da gestão de pedidos aos seus tabuleiros.

**Associação de Jogadores a Tabuleiros**

No que respeita a estas entidades, considerou-se ser da responsabilidade de Game guardar referências para as mesmas, uma vez que, sendo Game a entidade responsável por agilizar todas as acções do jogo, faz sentido que tenha na sua posse referências para os intervenientes, de forma a direccionar as acções.

**Tabuleiro e estruturas de Elementos**

Um tabuleiro é composto por elementos e, como tal, guarda referências para os mesmos. No que respeita às estruturas que possibilitam associar um ponto a um elemento, optou-se pela utilização de Hashtables, uma vez que, dada a chave Point, obtém-se o valor IElement.

Uma vez que um elemento pode ocupar vários pontos, o tabuleiro guarda a mesma referência do elemento para todos os pontos ocupados por este, uma vez que, é da sua responsabilidade fazer a gestão das posições ocupadas.

No que respeita às posições livres, as mesmas são determinadas quando para um dado ponto não existe elemento associado.

Considerou-se necessário ter duas estruturas como as referidas em cima: uma para elementos posicionados ainda não atingidos e outra para pontos já atingidos.

Esta necessidade prende-se com o facto de se pretender ordenar o desenho de um elemento com dano ou sem dano, bem como, ordenar o desenho de um tiro na água.

Cada tabuleiro tem um objecto estático Water, para que, todas as posições que já foram jogadas e que resultaram num tiro sem sucesso, façam referência a esse objecto.

Com esta implementação, evita-se a ocupação inicial de memória com referências para água.

**Jogadores**

Considerou-se necessário definir uma interface que permitisse, no futuro, criar outro tipo de jogadores (e.g. PDAPlayer), contudo, percebeu-se que o âmbito da aplicação não iria abranger tais cenários, uma vez que, sendo uma aplicação local (sem componente servidor) seria sempre necessário um jogador do computador e um jogador humano.

Apesar dessa percepção, houve certos detalhes de implementação que ficaram comprometidos desde o início, nomeadamente, o facto das referências para jogadores serem guardadas em Game num array.

**Posicionamento de Elementos**

Uma vez que é necessário garantir o número correcto de elementos por tipo no tabuleiro de jogo, delegou-se a responsabilidade de garantir tal facto a Game.

Para tal, Game itera sobre um objecto do tipo Hashtable que guarda quantidades por tipo de elemento, garantindo a correcta construção do tabuleiro, por parte do utilizador.

Esta opção invalida a possibilidade do utilizador escolher qual a peça que vai posicionar.

**Inteligência do jogador do computador**

O jogador representado por uma instância da classe IAPlayer apresenta a capacidade de prever e ponderar sobre a relativa posição dos elementos no tabuleiro, para tal, tem como base a implementação de um padrão Observer, no qual o jogador é tratado como um “listener” do estado do tabuleiro. Num estado inicial o jogador “inteligente” terá como uníca hipótese a escolha aleatória de um qualquer Ponto no tabuleiro, assim que o jogador acerte num elemento do tabuleiro, é notificado de tal ocorrência, sendo-lhe passado como parâmetro o elemento, desta forma e sabendo o tipo do elemento é possível ao jogador ponderar sobre as localizações prováveis do restante elemento, sendo as suas próximas tentativas em Pontos nessas localizações. A área de procura será encurtada assim que numa próxima jogada for possível auferir da direcção do elemento no tabuleiro.