Uma imagem com interior, monitor, animal

Descrição gerada automaticamente

Faculdade de Ciências Exatas e de Engenharia

Programação Orientada por Objetos 2019/2020

Alterações Climáticas



*Diego Andrés da Silva Briceño 2043818*

*Sílvia da Silva Fernandes 2043118*

*Rúben José Gouveia Rodrigues 2046018*

*Funchal 14/03/2020*

1. Introdução

Este relatório tem por objetivo demonstrar como foram aplicados os conhecimentos adquiridos na unidade de Programação Orientada por Objetos para atingir a meta proposta pelos docentes, criar um jogo cooperativo usando a plataforma *Greenfoot* cujo tema fosse as Alterações Climáticas.

Este relatório explicará a ideia principal deste grupo tal como a forma de implementação e os procedimentos realizados para tal fim e a justificação pela qual decidiu-se implementá-los.

1. Ideia Principal do Jogo

A ideia principal do grupo foi criar um jogo dividido em 3 fases que têm que ser completados sequencialmente para ganhar o jogo. A primeira fase está relacionada com a emissão de gases tóxicos para a atmosfera, principalmente pelas grandes indústrias. A segunda fase está relacionada com os efeitos maliciosos de ditos gases na atmosfera do planeta, em particular a camada de ozono. A terceira e última fase está relacionada com o degelo das calotas polares, consequência das alterações climáticas provocadas pela poluição e não só.

1. Procedimento e implementação do código
   1. Menus e Escolhas
      1. Classe *World*
         1. Subclasse “MenuInicial”

Este mundo permite aos utilizadores iniciar o jogo (depois de escolher os nomes e cores), redefinir os controlos ou sair do jogo.

Nesta subclasse tem-se implementado 2 construtores, um não recebe parâmetros enquanto que outro recebe um parâmetro booleano. Estes construtores só diferem na forma de definir a variável de instância reset: no construtor sem parâmetros esta variável é automaticamente true e no construtor com parâmetro booleano, reset é igual ao parâmetro passado.

Ambos os construtores criam um mundo de tamanho 1200 células por 700 células, em que cada célula mede 1 pixel por 1 pixel.

O método **prepare** instancia um objeto da classe “Play”, um objeto da classe “Options” e um objeto da classe “Exit”, todos nas posições devidas. O método prepare chama outro método chamado **resetStaticVariables** que, como o nome indica, vai repor aos valores “iniciais” as variáveis estáticas relacionadas com os jogadores.

Por fim, esta subclasse tem um método público **getMusica** que permite outros objetos parar a música de ambiente tocada.

3.1.1.2. Subclasse “Opções”

Este mundo permite aos utilizadores alterarem os controlos que utilizarão no jogo.

O construtor desta subclasse define um mundo de tamanho igual ao tamanho do MenuInicial.

O método **prepare** os botões relacionados com a mudança dos controlos e os textos que explicam a tecla e a instrução relacionadas a cada controlo.

3.1.1.3. Subclasse “EscolhaNomes”

Este mundo permite aos utilizadores registarem os seus nomes.

Nesta subclasse implementou-se um construtor que define as variáveis necessárias (nome é um string vazio, flag é true e recebeuNomeP1 é false) e cria um mundo das mesmas dimensões que o MenuInicial.

O método **prepare** instancia os objetos caixa, da classe Texto, que é um simples retângulo onde o objeto displayNome, também da classe Texto, mostra o nome que o utilizador está inserindo, um objeto Back, para retornar ao menu inicial, e outros textos informando o utilizador o que deve fazer. Ao utilizar este método no construtor, o mundo inicializa com os objetos nas coordenadas decididas automaticamente.

O método **escritaNome** é o método responsável por definir os nomes dos jogadores, estes nomes têm um limite de 10 carateres e o utilizador submete o seu nome carregando na tecla “enter”, como é explicado pelo mundo.

3.1.1.4. Subclasse “EscolherCor”

Este mundo permite aos utilizadores escolherem as suas cores.

Nesta subclasse implementou-se um construtor que cria um mundo do mesmo tamanho que os anteriores e define que, após a escolha das cores, será apresentado o mundo HowToPlay com a explicação do primeiro nível.

O método **prepare** instancia os objetos relacionados com as cores (Azul, Amarelo, Verde, Vermelho e Preto) e um botão Back para regressar ao mundo EscolhaNomes.

3.1.1.5. Subclasse “HowToPlay”

Este mundo simplesmente informa os utilizadores sobre o nível que jogarão ao clicar no botão Start.

Como nas subclasses anteriores, o construtor cria um mundo do tamanho anteriormente referido. Neste caso o método **prepare** demonstra a explicação (com recurso a objetos da classe Texto) e cria um botão da classe Start.

* + 1. Classe Actor e Classe Menus

Na classe Menus encontra-se o método **moveMouse** que recebe 2 imagens como parâmetros e alterna a imagem do objeto consoante o rato passe por cima do objeto, que irá ser chamado em todas as subclasses descritas abaixo. Também apresenta o método **playClick** que é chamada em todas as subclasses para reproduzir um som.

3.1.2.1. Subclasse “Play”

Esta subclasse tem um método **clickMouse** que regista se o utilizador selecionou o objeto e cria um mundo da classe EscolhaNomes. O método **act** chama o método clickMouse e o método moveMouse herdado da superclasse Menus.

3.1.2.2. Subclasse “Options”

Esta subclasse também tem um método **clickMouse**, só que quando regista que o utilizador selecionou o objeto cria um mundo da classe Opções e o método **act** é semelhante ao da subclasse anterior.

3.1.2.3. Subclasse “Exit”

Esta subclasse é semelhante às anteriores, só que o método **clickMouse** também regista se o utilizador clicou a tecla “Escape” e para a simulação.

3.1.2.4. Subclasse “Controlos”

Esta subclasse está encarregue de mudar os controlos consoante o utilizador deseje. O seu construtor recebe 2 inteiros, um designa qual o player a mudar o controlo e o outro designa qual controlo. O método **clickMouse** desta subclasse regista o click no objeto e, depois de despejar a última tecla digitada, fica à espera da tecla que será o novo controlo, e atualiza o texto que demonstra para o utilizador ficar informado. O método **inicio** insere o texto informando qual o controlo atual do jogador.

3.1.2.5. Subclasse “Texto”

Esta subclasse é usada puramente para escrever texto no jogo. Tem o método **updateText**, que como o nome indica é usado para atualizar o texto.

3.1.2.6. Subclasse “Back”

Esta subclasse é semelhante à subclasse “Exit”, pois o método **clickMouse** regista tanto o click do rato no objeto como regista o utilizador usar a tecla Escape para criar um novo mundo MenuInicial, sem fazer reset dos controlos.

* 1. Jogo 1

Instruções:

Nesta fase, o objetivo dos jogadores é destruir a máquina que está poluindo a atmosfera, dentro do limite de tempo de noventa segundos.

O jogo inicia com um jogador de cada lado do mundo. Cada jogador só pode permanecer no seu lado, não podendo ultrapassar a máquina e ir para o lado do outro jogador. Cada jogador tem 10 vidas, representadas por 5 corações. Os jogadores podem mover-se para a esquerda, direita, saltar, e também podem disparar, com as teclas predefinidas no jogo ou com aquelas que eles escolheram no menu “Controls”.

Para destruir a máquina os jogadores têm de atingi-la com as balas disparadas e ao mesmo tempo têm de se desviar do granizo na primeira etapa (que dura até a máquina atingir metade da sua vida) e depois na segunda etapa, têm de se desviar dos relâmpagos.

Antes de ser atingido por um relâmpago, o jogador é avisado através de um “alvo”, que aparece por debaixo dele, durante 3 segundos. Caso o relâmpago atinja o jogador, ele perde 2 vidas (1 coração).

Caso o granizo atinja o jogador, ele perde 1 vida.

Ao longo desta fase, para ajudar os jogadores, caem corações que lhes devolvem duas vidas.

Caso a máquina não seja destruída até o tempo acabar, os jogadores passam para a segunda fase se ambos ainda tiverem vidas. Caso contrário, como o jogo é cooperativo, basta apenas um jogador ficar sem vidas para o jogo terminar (game over).

Quanto mais vida a máquina tiver quando a fase acabar, maior será a dificuldade da fase seguinte.

Classe “World”

3.2.1.1 Subclasse “Jogo1”

Este mundo corresponde à primeira fase do jogo. O seu construtor

* 1. Jogo 2

Instruções:

Nesta fase, o objetivo principal é evitar que os gases tóxicos, libertados pela máquina na primeira fase, não atinjam a camada de ozono, durante 2 minutos.

O jogo inicia com duas naves com a mesma vida com que os jogadores terminaram a fase anterior. As naves têm as mesmas cores que os jogadores escolheram inicialmente. Elas podem mover-se na horizontal, para esquerda e para a direita, entre os limites do mundo.

Ao longo do jogo são lançados misseis teleguiados com a mesma cor da nave que eles pretendem atingir. Ao ser atingido por um míssil (de qualquer cor), o jogador perde duas vidas e 100 pontos.

Assim como na primeira fase, a camada de ozono tem uma barra a indicar a sua vida que irá diminuir caso os gases a atinjam.

Caso a vida da camada de ozono chegue a 0 ou uma das naves seja destruída, o jogo termina.

No decorrer do jogo, aparecem corações para os jogadores recuperarem vidas.

Quanto menor for a vida da camada de ozono no final desta fase, maior será a dificuldade da última fase.

* + 1. Classe World
       1. Subclasse Jogo2

O construtor deste nível cria um mundo com o tamanho referido anteriormente, define o relógio em 2 minutos, toca o som ambiente do nível, inicializa a variável quantoGas que será mencionada mais à frente e invoca o método prepare que inicializa todos os objetos necessários.

No método **prepare**, os objetos inicializados são as naves dos jogadores (*Nave1* e *Nave2*), os displays das vidas (*Vida\_player1* e *Vida\_player2*), a camada de ozono (*CamadaOzono*) e a sua vida (*VidaCamadaOzono*) e os textos que visualizam o relógio, os nomes dos jogadores e a sua pontuação.

Este mundo tem o método **atualizaRelogio** já mencionado no Jogo1, o método **libertarGases** que, como o nome indica, invoca objetos da classe Gas com uma probabilidade 3/(100 – quantoGas), de modo a que a quantidade de Gas libertado seja tanto maior quanto mais vida a máquina acabou no nível 1.

O método **vidas** invoca objetos da classe Vida\_jogo2 com uma probabilidade de 1/500. O método **disparaMissil**, como o nome indica, invoca objetos da classe Missil com uma certa probabilidade e, dependendo do número aleatório gerado, o míssil tem como alvo a nave do jogador 1 ou a nave do jogador 2.

O método **gameOver** recebe como parâmetros a vida da CamadaOzono e as vidas dos jogadores pois se algum destes parâmetros chegar a 0, os jogadores perdem o jogo.

O método **geral** invoca os métodos mencionados anteriormente, exceto o gameOver, enquanto a variável control é falsa, que indica que os jogadores ainda não perderam o jogo e é chamado no **act** do mundo, junto com o método gameOver.

* + 1. Classe Actor
       1. Subclasses Nave1 e Nave2

Estas subclasses são quase idênticas, a única diferença é que Nave1 é subclasse de Player1 e Nave2 é subclasse de Player2.

O construtor destas subclasses carregam a imagem da nave dependendo da cor que os jogadores escolheram e carrega as imagens relacionadas à explosão da nave, caso aconteça, num array denominado explosão.

As subclasses têm dois métodos: **moveNave**, que é o método relacionado com o movimento da nave, esta só pode mover-se para a esquerda e direita enquanto um dos jogadores tem vida, e o método **removeGas** que, como o nome indica, remove objetos da classe Gas quando a nave toca em dito objeto e adiciona 10 pontos ao jogador que removeu o Gas.

No método **act** é chamado os 2 métodos da subclasse e os métodos **perdeVidas** e **naveDestruida**, quando o jogador perdeu toda a sua vida, herdados da superclasse Players.

* + - 1. Subclasse Gas

Esta subclasse é simples, não apresenta um particular construtor e apresenta 2 métodos, ambos chamados no **act**.

O método **ultrapassaLimite** é simples, elimina o objeto se este chegou ao limite do mundo, só útil no Jogo1, e o método **movimentoGas** move o Gas para cima uma célula e possivelmente para a esquerda ou direita, um número aleatório de células.

* + - 1. Subclasse CamadaOzono

No construtor desta subclasse é definida a vida da camada (125) e definida a transparência da imagem da camada.

O método **act** só chama o método **mudarTransparencia** que diminui a vida e redefine a transparência da camada de acordo com a vida da camada, se um objeto da classe Gas toca na camada e remove o objeto da classe Gas.

A subclasse também tem um método estático **getVida** para obter o valor da vida da camada de ozono.

* + - 1. Subclasse VidaCamadaOzono

Esta subclasse é praticamente idêntica à subclasse VidaMáquina, a única diferença as dimensões da barra e a cor da barra.

* + - 1. Subclasse Missil

O construtor da subclasse Missil recebe como parâmetros um inteiro valor, que define a variável de instância limite, que define até qual y o míssil é teleguiado, um boolean alvoENave2, que indica se o alvo do míssil será a nave do jogador 2 ou a nave do jogador 1, e referências às duas naves no mundo. O construtor também define a cor do míssil dependendo da cor do alvo e carrega as imagens relativas à explosão do míssil num array.

Esta subclasse apresenta o método **teleguiado**, alvo de overloading de modo a que receba ou uma referência a um objeto da Nave1 ou da Nave2, que aponta o míssil ao alvo e move o míssil 2 células, se este míssil ainda estiver abaixo do limite.

O método **chegouAoFim** remove o míssil se este chegar a um limite do mundo, sem que este expluda e o método **atingiuAlvo** regista se o míssil está a tocar uma das Naves e retira uma vida a esse jogador e retira 100 pontos a esse jogador; também define a variável explosao como true de modo a que se possa chamar o método **animaExplosao** que, como o nome indica, simplesmente anima a explosão do míssil e, no fim da explosão, remove o objeto.

No método **act**, se explosao é false, os métodos teleguiado, com o alvo como parâmetro, atingiuAlvo e chegouAoFim são chamados; caso contrário, se o míssil atingiu uma nave, é animada a sua explosão.

* 1. Jogo 3

Nesta última fase, o objetivo principal é chegar à meta, saltando entre plataformas de gelo que se movem continuamente para a esquerda.

O jogo inicia com os jogadores (esquimós) numa plataforma inicial que se encontra parada até as próximas plataformas chegarem perto dos jogadores.

Nesta fase, os jogadores podem mover-se para a esquerda, direita e saltar.

Ao longo do jogo aparecem pinguins em cima das plataformas de gelo. Os jogadores para salvá-los têm ambos de tocar no pinguim ao mesmo tempo, ganhando assim 25 pontos cada. Caso contrário, ao chegar ao limite esquerdo do mundo, os pinguins caem da plataforma e os jogadores perdem ??? pontos. SCORE?

A largura das plataformas depende da fase anterior.

A fase é concluída quando ambos os jogadores chegam à plataforma final. Se um dos jogadores cair no mar, o jogo termina (game over).