Projeto Final TP1

Lucas Gonçalves Ramalho

[1]Dep. Ciência da Computação – Universidade de Brasília (UnB)

ramalho.lucas@aluno.unb.br

Resumo. *Nesse relatório será explanado o processo de contrução do projeto do jogo SpaceInvaders, detalhando os seus métodos, telas e diagrama de classes*

1. Introduction

Antes de começarmos a falar sobre o projeto em si, vamos falar um pouco sobre o jogo que inspirou esse projeto. SpaceInvaders é um jogo de 1987, ele teve uma grande fama principalmente por ser um dos primeiros jogos bidimensionais. A ideia base desse jogo é que se tem uma nave aliada, constrolada pelo jogador, e algumas naves inimigas. O objetivo do jogo é destruir todas as naves alienigenas antes que chegassem até você.

2. Objetivo

Replicar a estrutura do jogo da maneira que for possível utilizando conceitos da programação orientada a objetos nos aproveitando das ferramentas que são disponibilizadas pela linguagem Java.

3. Descrição do problema

Somos uma nave aliada que muita vontade de defender a terra de uma invasão alienígena, porém é preciso construir uma arquitetura em Java para que nossa nave possa se movimentar no espaço, atirar nos inimigos e destruir as naves inimigas que ameaçam a vida na terra.

4. Regras de negócio

Aqui vamos tocar nas regras de negócio do nosso jogo.

1. Controle de Status de telas – A apliacação deverá gerir qual tela deve estar visível no momento apropriado;
2. Controle de itens na tela – A aplicação deverá controlar posição dos objetos em tela e decetar eventos de teclado.
3. Garantir que os elementos da tela não passem dos seus limites
4. Regular o jogo a partir da dificuldade selecionada
5. Manter um hitórico das pontuações do jogador, além de organizar de forma decrescente.
6. Gerir colisões
7. Gerir quantidade inimigos
8. Gerir atividade da nave aliada, controlando a sua movimentação, quantidade de tiros e frequência de tiros

5. Diagrama de classes (UML)

Na imagem abaixo podemos conferir o diagram de classes do projeto

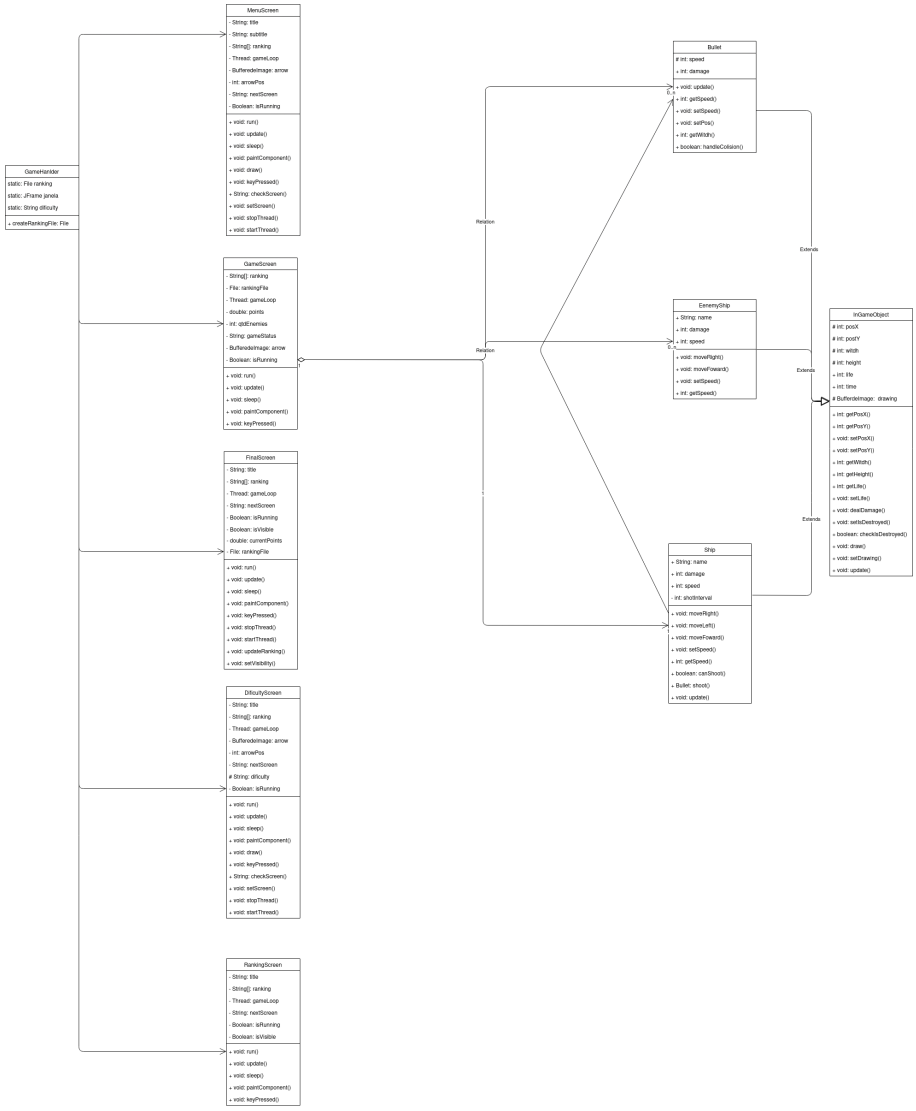

Figure 1. Diagrama de Classes

Na classe ”GameHandler” no seu método main é feito o carregamento do Frame e e controle das telas através da variável ”screen”. Além de carregar ou criar o arquivo de ranking através do método ”createRanking” no qual verifica se já existe um arquivo com nome ”ranking”, caso não haja esse arquivo é criado, caso contrário carrega esse arquivo para ser utilizado.

Enquanto o jogo está sendo executado o tempo todo é monitorado qual tela deve estar aparecendo utilizando o loop ”while”

Nas demais classes que controlam as telas todas extendem a class ”JPanel” da biblioteca gráfica ”awt”. Nelas são implementados o método ”paintComponent” que ”desenha” o que deve ser posto na imagem;

O método ”run” mantém as atualizações sendo feitas e faz o controle do frameRate da tela;

O método ”sleep” faz com que a thread ”duarma” por um certo período de tempo, esse tempo é definido pelo desempenho da máquina que está rodando a aplicação.

Também em todas as telas é implementado o método ”Keylistener” que escuta entradas de teclado em tempo de execução.

Agora na class ”InGameObject” temos os método getPosX, getPosY, getWitdh, getHeight e getLife que retornam os valores das respectivas variáveis, assim como todos esses método tem o seu equivalente com o prefixo ”set” em que é possível alterar o valor dessas variáveis.

Também temos o método ”dealDamage” que recebe um valor inteiro que é deduzido da vida do objeto. Complementar a esse método temos o método ”setIsDestroyed” que junto com o ”checkIsDestroyed” fazem o controle da vida do objeto e define se ele deve ser destruido, ou seja que seja removido da tela.

Ainda em ”InGameObject” temos o método ”setDrawing” que define quase é o desenho que deve ser utilizado por aquele objeto, e o método ”draw” que utiliza as variáveis de altura, largura e de posições além da imagem pra de fato colocar aquele objeto em tela.

Cada tela que se extende da classe ”InGameObject” implementa da forma que é necessária para funcionar de acordo com as suas especificações. Por exemplo a class ”Ship” precisa instâciar classes do tipo ”Bullet” através do método ”shoot”.

A classe ”Ship” pode se mover somente no eixo X. Durante a execução do programa essa classe utiliza o método ”canShoot” para verificar se o intervalo de tempo em que pode instânciar um objeto do tipo ”Bullet” passou, caso esteja no interavalo de tempo correto chama o método ”shoot”.

6. Telas de aplicação

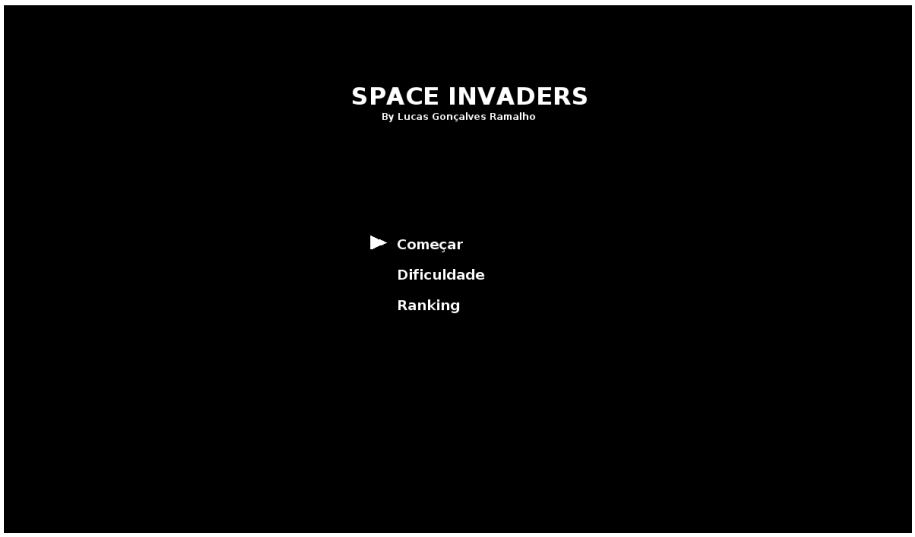

Figure 2. Tela de Menu

Na tela 2 é possível navegar nas 3 diferente telas, Ranking(6) onde é possível visualizar ranking das top 10 pontuações.
Começar um jogo, apartir da opção ”começar”, em que o usuário é direcionado pra a tela 3. Nela temos a nave aliada, as naves inimigas e a pontuação do jogador. A nave aliada pode utilizar as teclas ”A” e ”D” do teclado para se movimentar respectivamente para esquerda e direita.

Ao derrotar todos os inimigos ou os inimigos chegarem ao final da tela, o usuário é direcionado para a tela 4 onde se pode ver a pontuação atual do jogador e o ranking das ultimas partidas.

Figure 3. Tela do Jogo

Figure 4. Tela de fim de jogo

Figure 5. Tela de seleção de dificuldade

Na tela 5 é possível escolher a dificuldade entre Fácil, média e difícil, em que a depender da dificuldade escolhida é alterado a quantidade de inimigos e a velocidade deles.

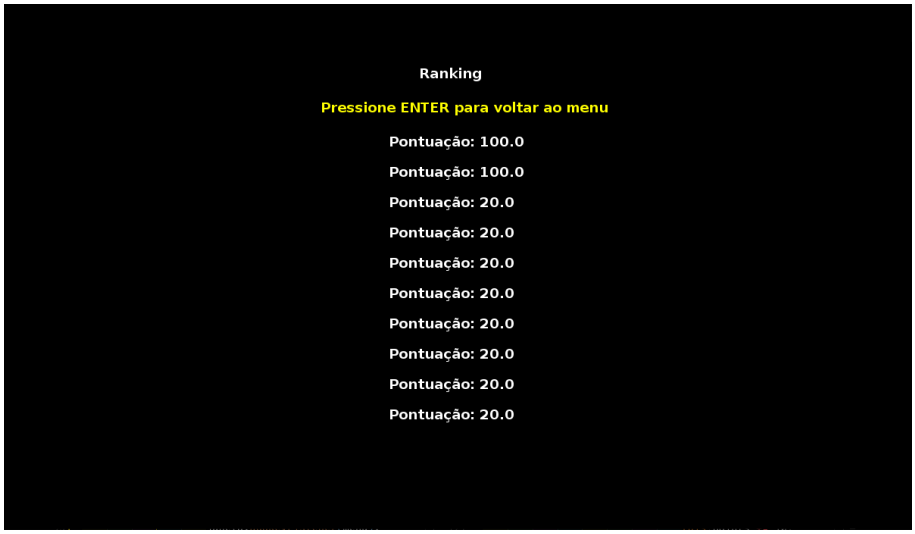

Figure 6. Tela de Ranking