Desenvolvimento com Linguagens de Programação Disruptivas II

Docente: Marcus Vinícius Chiulle Pinheiro

Alunos: Gabriel Antônio Dias Lanfranca Maida

Leonardo Peron Krause

Luis Felipe Barbosa

Pedro Henrique Sardá

N1 - Documentação - Block Smasher - Linguagem Lua - ADS41

1. Introdução
    - **Nome do Projeto:** Block Smasher
    - **Versão do Documento:** 1.0.0
    - **Data:** 13/04/2025
    - **Autores:** Gabriel Antônio, Leonardo Peron, Luis Felipe, Pedro Sardá
    - **Objetivo:** Descrever os Requisitos Funcionais, Não-Funcionais e Casos de Uso para o desenvolvimento do projeto do Block Smasher

2. Visão Geral
    - **Descrição do Sistema:** Block Smasher será um jogo arcade desenvolvido em Lua no estilo “Quebra-Tijolos”. A principal funcionalidade dele é implementar conceitos de programação distribuída e paralelismo, os quais serão representados através da concorrência de execução entre a barra e as bolinhas. O jogo consistirá em uma barra e vários blocos posicionados para serem destruídos. Ao apertar a tecla de espaço, a barra jogará uma bola em direção aos blocos. (preferencialmente em ângulo, senão as bolas apenas se moverão em um eixo). O objetivo é destruir todos os blocos.
    - **Stakeholders:** Jogadores, patrocinadores, professor

3. Requisitos Funcionais
    - **RF-01:** Movimentação da Barra: O jogador deve ser capaz de mover a barra horizontalmente (esquerda e direita) usando as setas do teclado
    - **RF-02:** Lançamento da Bola: O jogador deve ser capaz de lançar uma bola com a tecla espaço, preferencialmente em um ângulo para evitar movimento linear

- **RF-03:** Múltiplas Bolas Ativas: Deve ser possível ter mais de uma bola simultaneamente no jogo para simular o aspecto distribuído
- **RF-04:** Colisão da Bola: As bolas devem ricochetear ao colidirem com a barra, as paredes ou blocos
- **RF-05:** Destruição de Blocos e Pontuação: Ao atingir os blocos, estes devem ser destruídos, e o jogador deve ganhar pontos. A pontuação deve ser exibida na tela em tempo real
- **RF-06:** Condição de Vitória: O jogo deve terminar e indicar a vitória quando todos os blocos forem destruídos
- **RF-07:** Pontuação em Tela: A pontuação do jogador será exibida dinamicamente durante o jogo, mas não será armazenada após o término da partida
- **RF-08:** Sincronização entre Bolas: O estado das bolas deve ser sincronizado entre processos distribuídos para garantir consistência durante a partida
- **RF-09:** Paralelismo de entidades: As movimentações das bolas e da barra devem ser independentes e ocorrer de forma paralela

4. Requisitos Não-Funcionais

- **RNF-01:** Desempenho: O jogo deve rodar de maneira fluida e o tempo de resposta rápido
- **RNF-02:** Modularidade: O código deve ser modular para suportar facilmente novas implementações e escalabilidade
- **RNF-03:** Usabilidade: A interface deve ser simples e intuitiva, sem a necessidade de instruções complexas
- **RNF-04:** Clareza do Código: O código deve seguir boas práticas de programação, incluindo comentários e organização lógica, para facilitar a manutenção e futuras colaborações

5. Casos de Uso

- **Caso de Uso 1:** Movimentação da Barra
    - **Descrição:** O jogador move a barra no eixo horizontal utilizando as setas do teclado
    - **Ator(es):** Jogador
    - **Fluxo Principal:**
        i. O jogador pressiona a seta para a direita ou esquerda

ii. A barra se move na direção correspondente, respeitando os limites da tela
iii. O jogador ajusta a posição da barra para interceptar a bola

- **Caso de Uso 2:** Lançamento da Bola
  - **Descrição:** O jogador lança uma bola em direção aos blocos pressionando a tecla espaço
  - **Ator(es):** Jogador
  - **Fluxo Principal:**
    i. O jogador posiciona a barra
    ii. O jogador pressiona a tecla espaço
    iii. A bola é lançada em um ângulo inicial, movendo-se em direção aos blocos

- **Caso de Uso 3:** Vitória do Jogador
  - **Descrição:** O jogador vence o jogo ao destruir todos os blocos presentes no cenário
  - **Ator(es):** Jogador
  - **Fluxo Principal:**
    i. O jogador acerta os blocos com a(s) bola(s)
    ii. Quando todos os blocos são destruídos, o sistema exibe uma mensagem de vitória
    iii. O jogo termina e o jogador tem a opção de reiniciar a partida

- **Caso de Uso 4:** Sincronização de Bolas (Programação Distribuída)
  - **Descrição:** As bolas são sincronizadas entre diferentes processos ou dispositivos conectados
  - **Ator(es):** Servidor e Clientes
  - **Fluxo Principal:**
    i. O jogador lança uma bola
    ii. O estado da bola (posição, direção, velocidade) é compartilhado entre os processos ou máquinas conectadas
    iii. As posições das bolas são atualizadas em tempo real nos dispositivos conectados, garantindo consistência

6. Restrições
    - **Simplicidade Intencional:** Por ser um projeto acadêmico, funcionalidades complexas foram excluídas do escopo propositalmente
    - **Tempo de Desenvolvimento:** O tempo para o desenvolvimento é limitado a duração do período de aulas e, consequentemente, de avaliações
    - **Falta de Persistência dos Dados:** Inicialmente, não há previsões de adicionar persistência dos dados entre as sessões de jogo, desta forma, o progresso não será armazenado
    - **Dependência do Framework LÖVE:** O jogo requer o framework instalado para funcionar corretamente

7. Tecnologias Utilizadas
    - Linguagem Lua
    - Framework LÖVE2D
    - Git / GitHub
    - VS Code