Etapa 01: Projeto

Juliana Zambon[1]

[1]Departamento de Informática
Programação 02 – CI1002
Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brasil

{jz22}@inf.ufpr.br

Abstract. *Development of an alternative version of Space Impact, inspired by Space Impact 303, preserving the original game's mechanics and dynamics.*

Resumo. *Desenvolvimento de uma versão alternativa de Space Impact, inspirada no Space Impact 303, mantendo as mesmas mecânicas e dinâmicas do jogo original.*

1. Introdução

Os requisitos mínimos do jogo incluem a presença de duas fases distintas, cada uma com seu próprio cenário. A nave do jogador deve ter um sprite próprio e animações para os movimentos laterais, além de um sistema de HP que, ao se esgotar, encerra o jogo. A nave também deve dispor de um ataque simples sempre ativo, com possibilidade de adquirir ataques especiais temporários ao longo do mapa.

Em cada fase, é necessário incluir pelo menos dois tipos de inimigos normais (totalizando quatro no jogo). Dois desses inimigos devem ser capazes de lançar projéteis, enquanto apenas um pode ser imóvel; os demais devem seguir padrões de movimentação, ainda que simples. Cada inimigo deve ter um sprite exclusivo, sem repetições.

Cada fase deve durar no mínimo trinta segundos e culminar em um chefe único. Cada chefe deve ter um sprite exclusivo, dois ataques especiais, incluindo ao menos um que lance projéteis, e ser consideravelmente mais resistente que os inimigos normais. Além disso, os chefes devem representar um desafio significativo ao jogador, como mudanças de comportamento que aumentem a dificuldade conforme sua vida diminui.

Por fim, a nave do jogador deve possuir dois tipos de ataques especiais, que podem ser adquiridos temporariamente no cenário, com duração máxima de cinco segundos cada.

2. Segmentação do Trabalho

2.1. Módulo Principal: `main.c` e `configuracoes.h`

O módulo principal, main.c, coordena o fluxo do jogo e integra os demais módulos. O arquivo configuracoes.h define parâmetros gerais e constantes do jogo, facilitando a configuração e manutenção do código.

2.2. Módulo de Controle do Joystick: `joystick.c` e `joystick.h`

Responsável pela inicialização, atualização e finalização do joystick, esse módulo interpreta a entrada do jogador e traduz os comandos para a movimentação da nave e controle de ações, como ataques.

2.3. Fases do jogo: `fase1.c`, `fase2.c`, e `fases.h`

Cada fase possui um módulo dedicado (fase1.c e fase2.c) para definir o cenário, a duração e a sequência de eventos. Enquanto fases.h centraliza as funções comuns a ambas as fases, como inicialização, controle do tempo e transição para o chefe ao final de cada fase.

2.4. Módulo do Jogador: `jogador.c` e `jogador.h`

Centraliza o controle da nave do jogador, incluindo o gerenciamento de movimentação, sistema de HP, e ataques básicos e especiais. Esse módulo simplifica o controle da nave, tornando a integração com o joystick eficiente.

2.5. Módulo de Inimigos: `inimigos.c` e `inimigos.h`

Gerencia a criação e o comportamento dos inimigos normais em cada fase. Inclui a movimentação, os ataques, e a lógica de atualização para garantir uma experiência consistente em ambas as fases.

2.6. Módulo de Chefes: `chefes.c` e `chefes.h`

Dedicado aos chefes finais de cada fase, esse módulo define suas características únicas, como padrões de ataque e resistências, garantindo que cada chefe proporcione um desafio adequado ao jogador.

2.7. Módulo de Colisões e Interações: `colisoes.c` e `colisoes.h`

Gerencia as colisões entre a nave do jogador, inimigos, projéteis e chefes, aplicando danos e atualizando o HP do jogador e dos chefes. Centraliza as interações após colisões, incluindo efeitos de dano e remoção de inimigos ou projéteis após o impacto.

3. Dependências dos Módulos

3.1. Módulo Principal (`main.c`)

- **Dependências:**
  - `configuracoes.h`: Para acesso às constantes e configurações globais do jogo.
  - `joystick.h`: Para capturar entradas do joystick que controlam a nave.
  - `jogador.h`: Para gerenciar a nave do jogador e suas interações.
  - `fase1.h` e `fase2.h`: Para iniciar e gerenciar as fases do jogo.
  - `inimigos.h`: Para acessar funções relacionadas à criação e controle dos inimigos.
  - `chefes.h`: Para gerenciar a lógica dos chefes ao final de cada fase.
  - `colisoes.h`: Para detecção de colisões e interações entre os elementos do jogo.

3.2. Módulo de Controle do Joystick (`joystick.c`)

- **Dependências:**
  - `jogador.h`: Para atualizar a posição da nave e ativar ataques baseados na entrada do joystick.

3.3. Módulo da Nave do Jogador (`jogador.c`)

- **Dependências:**
  - `configuracoes.h`: Para acessar configurações de HP e limites do jogo.
  - `colisoes.h`: Para gerenciar a aplicação de danos e efeitos relacionados à nave.
  - `joystick.h`: Para receber comandos de movimentação e ataque.

3.4. Módulo de Fases (`fase1.c` e `fase2.c`)

- **Dependências:**
  - `jogador.h`: Para instanciar e controlar a nave do jogador durante a fase.
  - `inimigos.h`: Para gerar e gerenciar inimigos específicos para cada fase.
  - `chefes.h`: Para iniciar o chefe da fase ao final dela.
  - `colisoes.h`: Para implementar a lógica de colisão entre a nave, inimigos e projéteis.

3.5. Módulo de Inimigos (`inimigos.c`)

- **Dependências:**
  - `colisoes.h`: Para detectar colisões entre os inimigos e a nave do jogador, assim como seus projéteis.
  - `jogador.h`: Para referenciar o HP da nave e aplicar danos ao jogador em caso de colisão.

3.6. Módulo de Chefes (`chefes.c`)

- **Dependências:**
  - `jogador.h`: Para interagir com a nave do jogador e aplicar dano.
  - `colisoes.h`: Para gerenciar as interações entre os ataques do chefe e a nave do jogador.
  - `configuracoes.h`: Para acessar informações sobre a resistência dos chefes e seus padrões de ataque.

3.7. Módulo de Colisões (`colisoes.c`)

- **Dependências:**
  - `jogador.h`: Para aplicar efeitos de colisão à nave do jogador.
  - `inimigos.h`: Para interagir com os inimigos e determinar o resultado das colisões.
  - `chefes.h`: Para gerenciar colisões entre os ataques dos chefes e a nave do jogador.

4. Ordem de Desenvolvimento dos Módulos

Levando em consideração as dependências e a lógica do jogo:

1. **Módulo de Configurações (`configuracoes.h`)**
   - Contém constantes e configurações globais que serão usadas em todo o jogo.
2. **Módulo de Colisões (`colisoes.c` e `colisoes.h`)**

- Para garantir que os elementos do jogo possam interagir adequadamente, essencial para a nave, inimigos e chefes.

3. **Módulo da Nave do Jogador (`jogador.c` e `jogador.h`)**
   - Funcionalidades básicas da nave, como movimentação, sistema de HP e ataque simples, para o jogador interagir com o jogo.
4. **Módulo de Controle do Joystick (`joystick.c` e `joystick.h`)**
   - Após a nave estar implementada, adicionar seu controle via joystick.
5. **Módulo de Inimigos (`inimigos.c` e `inimigos.h`)**
   - Criação dos inimigos com suas mecânicas de movimento e ataques.
6. **Módulo de Chefes (`chefes.c` e `chefes.h`)**
   - A lógica para os chefes, padrões de ataque e resistência.
7. **Módulo da Fase 1 (`fase1.c` e `fase1.h`)**
   - Desenvolvimento da primeira fase do jogo, integrando a nave, inimigos e chefe.
8. **Módulo da Fase 2 (`fase2.c` e `fase2.h`)**
   - Implementação da segunda fase, semelhante à primeira, mas com novo cenário e inimigos.
9. **Módulo Principal (`main.c`)**
   - **Observação:** O desenvolvimento do módulo principal ocorrerá em paralelo com os demais módulos, permitindo a realização de testes ao longo do processo.
   - Desenvolvimento do módulo principal, que será responsável por iniciar o jogo, gerenciar a lógica de transição entre fases e controlar o fluxo geral do jogo.

4.1. Cronograma de Desenvolvimento

5. Cronograma de Desenvolvimento

Data	Atividade
05 a 07 de Nov	Módulo de Configurações (`configuracoes.h`)
08 a 10 de Nov	Módulo de Colisões (`colisoes.c` e `colisoes.h`)
11 a 15 de Nov	Módulo da Nave do Jogador (`jogador.c` e `jogador.h`)
16 a 19 de Nov	Módulo de Controle do Joystick (`joystick.c` e `joystick.h`)
20 a 24 de Nov	Módulo de Inimigos (`inimigos.c` e `inimigos.h`)
25 a 28 de Nov	Módulo de Chefes (`chefes.c` e `chefes.h`)
29 de Nov a 01 de Dez	Módulo da Fase 1 (`fase1.c` e `fase1.h`)
02 de Dez	Módulo da Fase 2 (`fase2.c` e `fase2.h`)
03 de Dez	Módulo Principal (`main.c`) e Testes Finais

Tabela 1. Cronograma de Desenvolvimento do Jogo

5.1. Considerações Finais

A data final no cronograma está estabelecida para o dia 3 de dezembro, permitindo uma margem para eventuais atrasos ou imprevistos. Ressalta-se que tanto o cronograma quanto o projeto estão sujeitos a alterações conforme necessário.

Referências

ALLEGRO 5. Disponível em: https://liballeg.org/.
DOCUMENTAÇÃO do Allegro 5. Disponível em: https://liballeg.org/a5docs/trunk/.