Shinobivania

O objetivo deste documento é listar os principais elementos necessários para o game, bem como suas funcionalidades e requisitos.

# Elementos do Game

Abaixo, uma lista com os principais elementos de um game:

**Jogador**

**Game Controller**

**Camera**

**Inimigos e NPCs**

**Menus**

**Configurações**

**Cenas**

**Level**

# Descrição Detalhada

## Jogador

Principal elemento do game, representado pelo avatar do personagem protagonista.

### Mecânica

Diz respeito a forma como o jogador interage com o jogador, com o cenário, com outras entidades e com a física do jogo.

**Criar um componente ControlePersonagem que será responsável por funções básicas de movimentação e mecânica.  
Preferência por uso alternado entre RigidBody2D com controle da velocidade (RigidBody2D.velocity) e Root Motion**

### Skills

Habilidades, magias, poderes especiais que poderão ser usadas pelo jogador

**Criar um componente Skill que controlará a ativação das skills do personagem.  
Criar a função TentarSkill() que verificará se todos os requisitos para uso dessa skill foram atendidos.**

### Entrada

Teclado / Touch Screen. Controlar quando o jogador terá ou não o controle sobre o personagem

**Componente PlayerInput que se comunicará diretamente com o ControlePersonagem recebendo comandos via teclado/joypad/touchscreen.**

### Vida/Status

Controlar a situação do jogador, seu nível de vida, controle de dano, limitações mecânicas, fraquezas etc.

**Componente essencial para TODOS as entidades do jogo que possam sofrer danos ou até mesmo serem destruídas.**

## Game Controller

Responsável por controlar o estado, os eventos e o fluxo do game.

### Máquina de Estados (FSM)

Será controlado por uma FSM:

1. Apresentação: exibe uma cena de introdução ao jogador, bem como informações sobre o level atual. (Tempo normal, controle de personagem desabilitado, HUD opcional, tema opcional)
2. Gameplay: momento em que a ação do game ocorre. (Tempo normal, controle de personagem habilitado, HUD vivível, tema do level)
3. Pausa: para o jogo e exibe o menu de pausa. (Tempo parado, controle de personagem desabilitado, HUD invisível, elementos de interface do menu pausa, tema do level com volume reduzido)
4. Game Over: momento após a morte ou derrota do jogador. (Tempo normal, controle de personagem desabilitado, HUD invisível, tema de game over)
5. Conclusão: momento após o sucesso do jogado (Tempo normal, controle de personagem desabilitado, HUD invisível, tema opcional, menu opcional)

**Criar um componente GameController que controlará o estado do jogo.  
Esse componente terá funções essenciais para o controle do fluo da gameplay. Tains como PararTempo(), PausarControles(), MudarMusica(), VolumeMusica() etc.**

**Criar também uma classe abstrata chamada SubControle para cada estado acima que fará requisições ao GameController para alterar a gameplay.**

## Câmera

Controla o posicionamento e configuração da câmera do jogo

**Criar um componente ControleCamera**

### Alvo

Para qual objeto a câmera deverá olhar e como transitará de um alvo para outro. Controlar o enquadramento dos objetos.

**MudarAlvo(Transform novoAlvo)**

### Paralaxe

Como o background responderá à movimentação da câmera

**Componente Paralaxe() que terá uma referência apontando para a câmera e mover-se-á de acordo com o movimento dessa câmera.**

### Configuração

Como as propriedades da câmera poderão ser modificadas ou resetadas. Configuração PADRÃO.

**Funções específicas para esse fim.**

### Efeitos

Controlar os efeitos visuais, filtros, chacoalhar, tremor etc.

**Funções específicas para esse fim.**

## Inimigos

Controle dos agentes (inimigos e NPCs) do jogo

### Consistência Mecânica

A mecânica dos agentes deve ser parecida com a do jogador. Exceto por habilidades específicas de alguns agentes como levitar, atravessar paredes etc.

**Substituir o componente PlayerInput por um controlador de IA**

### Inteligência Artificial (IA)

Uso de técnicas de IA.

**FSM, Árvore de decisão, Árvore de comportamento**

## Menus

### Layout

Posicionamento e responsividade dos elementos da interface gráfica

**Usar o sistema de GUI da Unity.**

### IHC

Poder de controle do jogador.

**Heurísticas de Nielsen**

## Configurações

Controle sobre as configurações gerais do jogo

* Audio: volume dos sons, da música, da narração etc.
* Vídeo: nível de qualidade visual, taxa de frames (fps), modo daltônico etc.
* Controle de violência: mimimi

### Arquitetura da Informação

Como as configurações serão organizadas (estruturas de dados)

### Persistência

Armazenar as configurações e salvar o jogo

### Perfis de Usuário

Possibilidade de guardar mais de uma configuração.

### Criptografia

Como e quais dados serão criptografados

## Cenas

Controle sobre as diversas cenas do jogo

### Transição

Como serão (ou SE ocorrerão) as transições entre uma cena e outra. Efeitos de transição.

### Loading

Como se dará o processo de carregamento das cenas, seja explícito ou em background

### Cutscenes

Cenas recorrentes entre ou durante os levels:

* Cena de abertura
* Cena entre um level e outro
* Cena de interrupção (ex. quando o jogador ganha um novo item)

## Level

Controle sobre os elementos dos mapas do jogo

### Spawn Points

Pontos estratégicos de onde objetos surgirão (ex.: um portal de onde saem inimigos)

### Zonas do Mapa

Áreas do mapa que disparam determinados eventos.

### Hazards

Objetos que causam danos ou matam o jogador ao contato.

### Pontos Estratégicos

* Ponto de Início: onde o jogador começa
* Pontos de Entrada/Saída: portas, canos (Mario feelings) etc.
* Savepoints
* Pontos de Cura
* Etc.

## HUD

Interface gráfica (GUI)

### Estado do Jogador

Painel com informações sobre o jogador: vida, energia, munição etc.

### Estado dos Inimigos

Painel com informações sobre um inimigo.