

 **GAME DESIGN DOCUMENT (GDD)** **Nome do Jogo:** 1soMath **Conceito Principal:**

Um puzzle game que mistura **lógica matemática** com uma **jornada emocional**, onde Pit, o guardião da Torre da Lógica, precisa restaurar o equilíbrio das Runas para salvar seu mundo do Vazio.

 **Versão do Documento:** v1.0 **Desenvolvedor:** Eder Jr **Data de Criação:** 26/04/2025 **Conteúdo do Documento:**

Este GDD contém todas as informações sobre o universo de 1soMath, incluindo:

- História, personagens e ambientação
- Mecânicas de gameplay (Runas, puzzles, HUD)
- Progressão narrativa (Jornada do Herói)
- Estrutura visual e artística
- Etapas de desenvolvimento e cronograma

 **Confidencialidade:**

Este documento é **confidencial** e de uso exclusivo da equipe de desenvolvimento do jogo **1soMath**. Nenhuma parte pode ser reproduzida ou divulgada sem autorização prévia.



**SUMÁRIO**

1. Conceito do Jogo (Game Concept) .....	7
1.1. Definição do Foco .....	7
1.2. Introdução/Background.....	7
1.3. Postos-chave .....	7
1.4. Gênero.....	7
1.5. Cultura .....	7
1.6. Tamanho e Escopo .....	8
1.7. Requisitos Técnicos .....	8
1.8. Cronograma do Projeto.....	8
2. História e Roteiro do Jogo .....	9
2.1. História .....	9
2.1.1. Etapas da História Contempladas .....	9
2.2. Roteiro .....	10
3. Mecanismos do Jogo.....	11
3.1. Regras .....	11
3.2. Funcionamento.....	11
3.2.1. Modelos Conceituais de Funcionamento .....	11
3.2.2. Modos de Jogo .....	12
3.2.3. Número de Jogadores .....	12
3.2.4. Controles do Jogo .....	12
3.2.5. Movimentos do Personagem Jogável.....	13
3.3. Objetos Interativos .....	13
3.4. Cenário .....	13
3.4.1. Quantidade.....	13
3.4.2. Colisões .....	13
3.5. Controle do Usuário .....	14
3.5.1. Ajuda ou Tutorial .....	14
3.5.2. Save Points .....	14
3.5.3. Continue .....	14
3.5.4. Dificuldade .....	14
3.6. Visão do Jogador .....	14
3.7. Informações da Tela Principal do Jogo.....	15
3.8. Sidequests.....	15
4. Elementos do Jogo .....	17

4.1. Personagens .....	17
4.1.1. Principal .....	17
4.1.2. Inimigo Conceitual.....	18
4.2. Cenários .....	19
4.2.1. Definição Conceitual dos Cenários.....	19
4.2.2. Level Design .....	20
4.3. Objetos Interativos .....	21
4.4. Menus do Jogo.....	22
4.4.1. Estrutura dos Menus .....	22
4.4.2. Visual das Telas de Menu .....	23
4.5. Som .....	25
4.5.1. Cenários.....	25
4.5.2. Menus e Transições .....	25
4.6. Animações.....	25
4.6.1. Personagem (Pit).....	25
4.6.2. Inimigos (O Vazio) .....	25
4.6.3. Objetos .....	25
4.6.4. Cenários.....	25
4.6.5. Menus e Telas .....	26
4.7. Textos do Jogo .....	26
4.7.1. Idiomas Disponíveis .....	26
4.7.2. Exemplos de Textos Iniciais: .....	26
4.7.3. Menu: .....	26
5. Inteligência Artificial .....	27
5.1. Necessidade no Jogo .....	27
5.2. Algoritmo Utilizado .....	27
5.2.1. Técnica de IA por Eventos Condicionados.....	27
5.2.2. Justificativa do Uso .....	27
5.3. Aplicação da IA no Jogo .....	28
5.3.1. Eventos Controlados por IA .....	28
5.3.2. Vetores de Influência no Jogo .....	29
5.3.3. Balanceamento da Dificuldade .....	29
5.4. Resumo da Implementação .....	29
6. Implementação do Jogo .....	31
6.1. Recursos Tecnológicos.....	31

6.1.1. Hardware.....	31
6.1.2. Software .....	31
6.2. Modelagem de Classes.....	32
6.2.1. Motor do Jogo – Estrutura de Classes Sugerida.....	32
7. Monetização e Estratégia de Publicação.....	39
7.1. Monetização .....	39
7.1.1. Modelo de Receita: .....	39
7.2. Estratégia de Publicação .....	40
7.2.1. Plataformas-Alvo: .....	40
7.2.2. Requisitos para Publicação: .....	40
7.2.3. Marketing Inicial:.....	41
8. Plano de Testes (Quality Assurance).....	43
8.1. Objetivos do Plano de Testes .....	43
8.2. Tipos de Testes Realizados .....	43
8.2.1. Teste Funcional .....	43
8.2.2. Teste de Interface (UI/UX).....	44
8.2.3. Teste de Desempenho .....	44
8.2.4. Teste de Compatibilidade.....	44
8.2.5. Teste de Monetização .....	45
8.2.6. Teste de Usabilidade e Experiência.....	45
8.3. Cronograma de Testes.....	46
8.4. Ferramentas Utilizadas.....	46
8.5. Critérios de Aceitação .....	46



## 1. CONCEITO DO JOGO (GAME CONCEPT)

O conceito do jogo consiste em um projeto que integra **raciocínio lógico e emoções humanas**, trazendo uma abordagem **matemática e filosófica** por meio de puzzles e uma jornada introspectiva. Serão apresentados o foco do projeto, introdução narrativa, postos-chave, gênero, ambientação cultural, escopo técnico e cronograma de desenvolvimento do jogo **1soMath**.

### 1.1. DEFINIÇÃO DO FOCO

O objetivo do projeto é criar uma experiência interativa que une **desafios matemáticos** a uma narrativa profunda sobre **equilíbrio, lógica e autoconhecimento**. O público-alvo principal são pessoas a partir dos **12 anos**, com interesse em **puzzles inteligentes e histórias emocionais**, jogando em dispositivos **mobile**.

### 1.2. INTRODUÇÃO/BACKGROUND

**1soMath** é um jogo de puzzles em **perspectiva isométrica**, onde o jogador controla **Pit**, o guardião da **Torre da Lógica**. A torre está ameaçada por uma entidade conhecida como **O Vazio**, que fragmenta as **Runas**, responsáveis por manter a estrutura lógica do mundo.

Ao resolver puzzles matemáticos e lógicos, Pit recupera as Runas e restaura o equilíbrio do mundo. Cada Runa possui uma função única e se desgasta com o uso, exigindo do jogador estratégia na sua gestão.

A jornada de Pit é tanto **mental quanto emocional**, refletindo a luta interna entre ordem e caos. O jogo se destaca por seu **visual artístico minimalista**, trilha sonora imersiva e a originalidade do sistema de desgaste e recarga das Runas.

### 1.3. POSTOS-CHAVE

- **Runas:** Mecânica central de desgaste e recarga.
- **Puzzles matemáticos:** Variedade de desafios baseados em lógica, álgebra, geometria.
- **Narrativa emocional:** Jornada de autoconhecimento e superação.
- **Estilo visual:** Inspirado em Monument Valley e Gris, com ambientação mística e simbólica.
- **Desenvolvimento estratégico:** Gestão de recursos (Runas) e tempo.

### 1.4. GÊNERO

Puzzle / Aventura / Narrativo Interativo

### 1.5. CULTURA

O jogo trata de conceitos universais como **lógica, equilíbrio e resiliência emocional**, representados por elementos gráficos e narrativos baseados em símbolos matemáticos e filosóficos. Não possui ligações com culturas específicas, focando em uma experiência **atemporal e universal**.

## 1.6. TAMANHO E ESCOPO

- Tamanho estimado: **200 MB** (compatível com dispositivos Android e iOS modernos).
- Estrutura:
  - **12 puzzles principais**, interligados com narrativa.
  - HUD simples e intuitiva.
  - Sistema de anúncios não-invasivo para recarga de Runas.

## 1.7. REQUISITOS TÉCNICOS

### PLATAFORMAS:

- Android (versão 8.0 ou superior)
- iOS (versão 12 ou superior)

### RESOLUÇÕES COMPATÍVEIS:

- 1280x720 (HD)
- 1920x1080 (Full HD)
- Escalável para dispositivos 4K com otimização.

### MOTOR GRÁFICO SUGERIDO:

- **Unity** (suporte completo para 2.5D, animações leves, sistema de anúncios integrado)

### PERFORMANCE MÍNIMA ESPERADA:

- Processador quad-core
- 2 GB de RAM
- GPU com suporte OpenGL ES 3.0 ou superior

## 1.8. CRONOGRAMA DO PROJETO

O cronograma do desenvolvimento do **1soMath** segue uma sequência de etapas, desde o conceito inicial até a publicação. Veja detalhes completos na **Seção 10** deste GDD.

## 2. HISTÓRIA E ROTEIRO DO JOGO

A história tem como foco **envolver emocionalmente o jogador**, proporcionando uma jornada que vai além da lógica, explorando temas como **autoconhecimento, equilíbrio e superação**. O roteiro aqui apresentado dá uma **visão geral da estrutura narrativa**, sem entrar em detalhes técnicos de fluxograma. A progressão segue a **Jornada do Herói**, conectada aos puzzles e à mecânica das Runas.

### 2.1. HISTÓRIA

Onde começa o equilíbrio? De onde surgem as leis que regem a lógica e a emoção?

Em um mundo estruturado pela matemática e pela harmonia, **Pit**, o guardião da **Torre da Lógica**, é responsável por manter as **Runas** – fragmentos da **Grande Equação**, que asseguram a estabilidade do mundo.

Porém, uma falha surge: o **Vazio**, uma força de entropia, começa a corroer as Runas, fragmentando-as e ameaçando mergulhar tudo no caos. Pit, sentindo a conexão com as Runas enfraquecer, precisa embarcar em uma jornada para restaurá-las, enfrentando desafios lógicos e emocionais, refletidos nos puzzles e nos próprios dilemas internos.

Cada puzzle resolvido aproxima Pit de uma nova verdade sobre si mesmo e sobre o mundo que protege. As Runas, ao serem restauradas, não apenas o ajudam, mas também o testam, desgastando-se com o uso e exigindo escolhas difíceis. A luta de Pit é silenciosa, mas grandiosa: ele precisa aprender a lidar com os limites de suas próprias forças para reconstruir a Torre e, com ela, a ordem do mundo.

#### 2.1.1. ETAPAS DA HISTÓRIA CONTEMPLADAS

As etapas são baseadas na teoria da **Jornada do Herói** de Joseph Campbell, adaptadas para a narrativa introspectiva de 1soMath.

- **Herói:** Pit
- **Mundo Comum:** Torre da Lógica
- **Mundo Especial:** Domínio das Runas / Enfrentamento do Vazio

#### ETAPAS DA JORNADA:

##### 1. MUNDO COMUM:

Pit vive em equilíbrio na Torre da Lógica, cuidando das Runas que mantêm a ordem do mundo.

##### 2. CHAMADO À AVENTURA:

A conexão com as Runas enfraquece, e uma névoa escura surge – o Vazio começa a agir.

##### 3. RECUSA AO CHAMADO:

Pit dúvida de sua capacidade de enfrentar o caos. Questiona se poderá restaurar algo tão grande.

##### 4. ENCONTRO COM O MENTOR (INTERIOR):

Meditações revelam a sabedoria dos antigos guardiões: Pit precisa confiar nas Runas e em si.

##### 5. TRAVESSIA DO PRIMEIRO LIMIAR:

Pit ativa a primeira Runa restaurada e acessa a sala do primeiro puzzle.

##### 6. TESTES, ALIADOS E INIMIGOS:

Pit resolve puzzles que exigem lógica e estratégia. As Runas ajudam, mas se desgastam.

**7. APROXIMAÇÃO DA CAVERNA OCULTA:**

Pit descobre a verdadeira natureza do Vazio – a falha da própria lógica absoluta.

**8. PROVAÇÃO SUPREMA:**

Desafio final com todas as Runas integradas. Pit quase cede ao Vazio, mas supera seus limites.

**9. RECOMPENSA:**

Pit restaura a Runa Prateada, símbolo da harmonia, e ganha nova clareza.

**10. CAMINHO DE VOLTA:**

Retorna para estabilizar as Runas restauradas, enfrentando os últimos resquícios do Vazio.

**11. RESSURREIÇÃO:**

Confronto final com o Vazio, agora como um reflexo interno. Aceitação e superação.

**12. RETORNO COM O ELIXIR:**

A Torre da Lógica é restaurada. Pit se torna um guardião pleno, símbolo do equilíbrio entre razão e emoção.

## 2.2. ROTEIRO

O jogo possui uma **narrativa interativa contínua**, que guia o jogador através de pequenos textos e reflexões internas de Pit ao longo dos puzzles.

- **Textos curtos** aparecem ao início e fim de cada puzzle, reforçando o estado emocional de Pit.
- As **Runas** emitem mensagens simbólicas, reveladas ao serem restauradas.
- Inscrições antigas nos cenários expandem a Lore do mundo e a história dos guardiões anteriores.

**Nota:**

Os diálogos e textos narrativos específicos serão detalhados nas seções:

- **5.2 – Tipos de Puzzles**
- **5.4 – HUD e Interface**
- **6.1 – Jornada do Herói – Etapas**

### 3. MECANISMOS DO JOGO

Esta seção detalha as **regras**, o **funcionamento interno**, os **elementos interativos**, e o **controle do jogador** sobre o mundo de 1soMath. Inclui colisões, lógica de Runas, HUD, e a visão geral de como a mecânica das Runas e puzzles guiará o fluxo do jogo. A perspectiva do jogador é central para manter uma imersão contínua.

#### 3.1. REGRAS

O objetivo principal do jogo é ajudar **Pit** a restaurar as **Runas**, resolvendo puzzles em cada estágio da **Torre da Lógica**, enfrentando a ameaça do **Vazio**. As Runas são obtidas após cada puzzle e são essenciais para acessar novas áreas e resolver desafios subsequentes.

##### **Regras básicas:**

- O uso de cada Runa consome energia (desgaste).
- Runas recarregam em **8 horas** ou instantaneamente via anúncios.
- O jogador precisa usar estrategicamente as Runas disponíveis para resolver os puzzles.
- Não é possível usar uma Runa que esteja recarregando, mesmo que parcialmente carregada.
- Cada puzzle liberado desbloqueia uma nova Runa.
- A progressão segue obrigatoriamente a sequência: **Puzzle → Runa → Novo Puzzle**.

##### **Como Fazer:**

- Criar um **sistema de progressão linear** baseado na liberação de Runas.
- Usar uma **tabela interna** para controle de desgaste e recarga (com timers em background).
- Sistema de **verificação de bloqueio**: se a Runa necessária estiver indisponível, o jogador deve trocar.

#### 3.2. FUNCIONAMENTO

Esta seção descreve como a lógica do jogo opera, as modalidades, o controle do jogador, e o comportamento do personagem principal (Pit).

##### 3.2.1. MODELOS CONCEITUAIS DE FUNCIONAMENTO

- **Novo Jogo:**
  1. Introdução narrativa e apresentação da Torre.
  2. Primeiro puzzle (sequência simples) desbloqueia a Runa Branca.
  3. A cada puzzle resolvido, Pit obtém uma nova Runa e acesso ao próximo puzzle.
  4. Runas gastam energia a cada uso e devem ser trocadas quando descarregadas.
- **Fluxo básico do puzzle:**
  1. Entrada no puzzle.
  2. Seleção de Runa (ativa na HUD).

3. Interação com o ambiente.
4. Verificação de sucesso.
5. Cutscene curta de restauração da Runa.
6. Desbloqueio da próxima área/puzzle.

**Como Fazer:**

- Criar **fluxogramas visuais** com Unity Visual Scripting ou desenhar no Figma.
- Programar o loop: **Verificar Runa → Ativar Puzzle → Resolver → Atualizar Estado**.

### 3.2.2. MODOS DE JOGO

- **Modo História** (único modo principal):
  - Progressão linear por 12 puzzles.
  - Sistema de desgaste e recarga das Runas.
  - Narrativa interativa ao longo do caminho.
- **Modo Replay de Puzzles** (após terminar o jogo):
  - Jogador pode repetir puzzles específicos para bater tempo ou explorar variações.

**Como Fazer:**

- Estruturar um **sistema de checkpoints** com as Runas desbloqueadas.
- Criar uma **tela de seleção** para puzzles já resolvidos.

### 3.2.3. NÚMERO DE JOGADORES

- **Single Player**.
  - O jogador controla apenas Pit, mas interage com diferentes Runas e ambientes.

### 3.2.4. CONTROLES DO JOGO

- **Toque ou Swipe (Mobile)**:
  - **Toque simples**: Interagir com elementos do puzzle.
  - **Swipe**: Movimentar Pit entre áreas exploráveis.
  - **Toque longo**: Ativar/trocar Runa.

**Como Fazer:**

- Utilizar **Input System** do Unity (ou plugin mobile) para gestos.
- **Designar áreas ativas** da tela para diferentes tipos de toque.

### 3.2.5. MOVIMENTOS DO PERSONAGEM JOGÁVEL

- Pit se move lateralmente em cenários isométricos (esquerda/direita).
- Movimentos automáticos durante transições e cutscenes.
- Jogador não precisa controlar velocidade – foco está na **interação lógica**.

### 3.3. OBJETOS INTERATIVOS

- **Runas:** Ativadas via HUD, interagem com o ambiente.
- **Elementos dos puzzles:** Botões, alavancas, reflexos de luz, plataformas.
- **Portais de Transição:** Abrem após resolução do puzzle.
- **Inscrições Antigas:** Tocando nelas, revelam fragmentos da Lore.

#### Como Fazer:

- Criar cada objeto como um **Prefab** reutilizável.
- Scripts simples de interação: **OnClick → Trigger Action**.

### 3.4. CENÁRIO

#### 3.4.1. QUANTIDADE

- **12 ambientes únicos**, cada um correspondente a um puzzle e a uma Runa.
- Cada ambiente possui um tema visual e mecânico baseado na Runa associada.

#### 3.4.2. COLISÕES

- Pit só se move em áreas definidas como **caminháveis**.
- Colisão com:
  - **Paredes invisíveis:** delimitam áreas jogáveis.
  - **Obstáculos de puzzles:** bloqueiam ou redirecionam ações.

#### Como Fazer:

- Utilizar **colliders 2D/3D** para delimitação.
- Scripts de **evento ao colidir**: exibir mensagem, bloquear ação, etc.

### 3.5. CONTROLE DO USUÁRIO

#### 3.5.1. AJUDA OU TUTORIAL

- Tutorial inicial integrado nos 3 primeiros puzzles.
- Menu de ajuda acessível, explicando o sistema de Runas e HUD.

#### Como Fazer:

- Criação de **Tooltips** que aparecem nas primeiras interações.
- Script que verifica se o jogador já passou por um tutorial.

#### 3.5.2. SAVE POINTS

- Progresso salvo automaticamente após cada puzzle.
- **Sistema de salvamento local:** Runas obtidas, desgaste atual, puzzles resolvidos.

#### Como Fazer:

- Usar **PlayerPrefs** ou **sistema de arquivos locais**.
- Sincronizar com Firebase (opcional) para armazenamento em nuvem.

#### 3.5.3. CONTINUE

- Se o jogador fechar o jogo, retorna ao último puzzle resolvido, com o estado das Runas preservado.

#### 3.5.4. DIFICULDADE

- Progressiva: puzzles vão ficando mais complexos, exigindo combinação de Runas.
- **Não ajustável manualmente**, mas a gestão de Runas oferece um nível estratégico natural.

### 3.6. VISÃO DO JOGADOR

- Visão **isométrica 2.5D**, com movimentação lateral e profundidade visual.
- **Iluminação suave** e efeitos visuais com foco nas Runas e suas interações.

#### Como Fazer:

- Utilizar **câmera ortográfica** com ajuste de ângulo.
- Luzes **dinâmicas** para destacar Runas ativas e áreas resolvidas.

### 3.7. INFORMAÇÕES DA TELA PRINCIPAL DO JOGO

- **HUD:**

- Círculos com Runas disponíveis.
- Barra de desgaste.
- Timer de recarga.
- Botão de troca.

- **Ambiente:**

- Puzzle visual ao fundo.
- Indicadores visuais de progresso.

### 3.8. SIDEQUESTS

- **Bater recordes de tempo** em puzzles resolvidos.
- **Descobrir todas as inscrições** para desbloquear uma **Runa Secreta** visual.
- **Modo Desafio:** resolver puzzles sem usar determinada Runa.



## 4. ELEMENTOS DO JOGO

Nesta etapa são definidos todos os elementos fundamentais do jogo: **Personagens, Cenários, Objetos, Menus, Som, Animações e Textos**. Cada parte contribui para a imersão e funcionamento do mundo de **1soMath**.

### 4.1. PERSONAGENS

Aqui estão os personagens do jogo, com foco no protagonista e os elementos que desafiam sua jornada.

#### 4.1.1. PRINCIPAL

- **Pit:** O guardião da **Torre da Lógica**, Pit é o único personagem controlável pelo jogador. Ele não possui combate direto, mas interage com o mundo através das **Runas**, resolvendo puzzles e restaurando a harmonia.
- **Características:**
  - Visual leve, aparência serena, olhos brilhantes, corpo fluido.
  - Carrega consigo uma **aura luminosa** que reflete o estado atual das Runas (brilho forte = Runas em alta, brilho fraco = Runas desgastadas).
  - Movimentos suaves, baseados em gestos de toque.

{ imagem aqui!!! }

---

#### 4.1.2. INIMIGO CONCEITUAL

- **O Vazio:**
  - Entidade sem forma definida, que se manifesta como **sombra**s e **distorções** no ambiente.
  - Não ataca diretamente, mas corrompe puzzles e fragmenta Runas, criando obstáculos emocionais e lógicos.
- **Elementos do Vazio:**
  - Névoa escura que bloqueia caminhos.
  - Ecos e reflexos distorcidos de Pit que aparecem em momentos críticos.

{ imagem aqui!!! }

## 4.2. CENÁRIOS

Os cenários de 1soMath são ambientes **místicos e abstratos**, representando diferentes aspectos da mente lógica e emocional.

### 4.2.1. DEFINIÇÃO CONCEITUAL DOS CENÁRIOS

1. **Sala da Sequência:** Ambientes onde a repetição e o padrão visual dominam. Representa a **ordem inicial** da lógica.
2. **Templo da Luz:** Espaço iluminado por Runas flutuantes. Reflexão e iluminação são os principais desafios.
3. **Câmara do Equilíbrio:** Plataformas móveis e pesos. Simboliza a necessidade de **balancear** ações.
4. **Espelhos do Eu:** Cenário repleto de reflexos distorcidos. Enfatiza o **autoconhecimento**.
5. **Passagem da Água:** Correntes que alteram o ambiente em tempo real. Teste de **resiliência**.

{ imagem aqui!!! }

#### 4.2.2. LEVEL DESIGN

Cada cenário contém:

- **Áreas de movimentação restritas** (caminhos definidos para Pit).
- **Objetos interativos** (Runas, puzzles, inscrições).
- **Camadas visuais** (backgrounds estáticos e dinâmicos).
- **Limitações físicas**: bordas, plataformas, zonas bloqueadas pela névoa do Vazio.

{ imagem aqui!!! }

#### 4.3. OBJETOS INTERATIVOS

- **Runas:** Principais objetos interativos. Ativadas pelo jogador para manipular o cenário.
- **Inscrições:** Textos antigos que revelam fragmentos da lore.
- **Portais:** Transportam Pit entre as salas após completar os puzzles.

{ imagem aqui!!! }

## 4.4. MENUS DO JOGO

### 4.4.1. ESTRUTURA DOS MENUS

- **Tela Inicial:**

- Novo Jogo
- Continuar
- Configurações
- Créditos
- Sair

- **Menu Configurações:**

- Som: On/Off
- Idioma: português
- Dificuldade: Normal (única)

- **HUD durante o jogo:**

- Roda de Runas (seleção rápida).
- Barra de desgaste da Runa ativa.
- Indicador de tempo para recarga.

#### 4.4.2. VISUAL DAS TELAS DE MENU

- **Menu Principal:** Ambiente abstrato com Runas flutuantes.

{ imagem aqui!!! }

- **Menu HUD:** Elementos minimalistas, com ícones luminosos.

{ imagem aqui!!! }

## 4.5. SOM

### 4.5.1. CENÁRIOS

- **Trilha Sonora Ambiental:**
  - Música calma, com sons ambientes (ecos, vibrações).
  - Variação conforme o estado emocional de Pit.
- **Eventos Específicos:**
  - Ativação de Runa: som suave de cristal quebrando e recompondo.
  - Recarga Completa: brilho sonoro crescente.

### 4.5.2. MENUS E TRANSIÇÕES

- Menus: sons leves ao navegar.
- Transições entre puzzles: sons etéreos e sutil mudança na trilha.

## 4.6. ANIMAÇÕES

### 4.6.1. PERSONAGEM (PIT)

- **Andar:** deslizamento contínuo, com efeitos leves de luz ao redor.
- **Interagir com Runas:** animação de absorção de energia.
- **Desgaste:** aura enfraquece, postura mais curvada.

### 4.6.2. INIMIGOS (O VAZIO)

- Movimento da névoa: constante e imprevisível.
- Reflexos de Pit: surgem e somem com distorções.

### 4.6.3. OBJETOS

- Runas: giram lentamente, brilham conforme energia.
- Inscrições: aparecem com brilho ao serem tocadas.

### 4.6.4. CENÁRIOS

- Animação de parallax: camadas se movem conforme Pit avança.

- Efeitos de luz/sombra dinâmicos nas áreas de puzzle.

---

#### 4.6.5. MENUS E TELAS

- Botões: brilham ao toque.
- Runas no menu: flutuam e giram ao fundo.

---

### 4.7. TEXTOS DO JOGO

---

#### 4.7.1. IDIOMAS DISPONÍVEIS

- **Português**

---

#### 4.7.2. EXEMPLOS DE TEXTOS INICIAIS:

- **Início do Jogo:**
  - "A Torre enfraquece. As Runas clamam por equilíbrio."
- **Tutorial:**
  - "Toque em uma Runa para ativá-la. Use-a com sabedoria."
- **Ao Restaurar uma Runa:**
  - "A clareza retorna. Um passo a menos para o caos."

---

#### 4.7.3. MENU:

Novo Jogo

Continuar

Configurações

Sair

Som

## 5. INTELIGÊNCIA ARTIFICIAL

### 5.1. NECESSIDADE NO JOGO

A IA em **1soMath** é necessária para:

- Tornar o mundo **reativo** às escolhas do jogador.
- Aumentar a **imersão** por meio de mudanças visuais e desafios dinâmicos.
- Controlar o **desgaste das Runas** e a **interferência do Vazio** nos puzzles.

#### Como Fazer:

- Criar uma **estrutura de eventos** ligada ao comportamento do jogador (ex.: acertos, erros, tempo).
- Desenvolver scripts que monitoram:
  - Quantidade de erros por puzzle.
  - Uso de cada Runa.
  - Presença de Pit em áreas específicas.

### 5.2. ALGORITMO UTILIZADO

#### 5.2.1. TÉCNICA DE IA POR EVENTOS CONDICIONADOS

- Sistema baseado em **condições simples** que ativam ou modificam elementos do jogo.
- Exemplo de eventos:
  - **Se** Runa usada 5 vezes em 1 puzzle → **então** desgaste +10%.
  - **Se** Pit erra 3 vezes → **então** iluminar dica visual;

#### Como Fazer:

- Criar scripts com **condições lógicas** (If/Else) que verificam os eventos em tempo real.
- Utilizar **listas de estados** ou **booleanas** para ativar/desativar eventos.
- Criar um **Event Manager** que centraliza todas as respostas da IA.

#### 5.2.2. JUSTIFICATIVA DO USO

- Simples de implementar.
- Baixo custo computacional (ideal para mobile).
- Permite personalização rápida e fácil balanceamento.

## 5.3. APLICAÇÃO DA IA NO JOGO

### 5.3.1. EVENTOS CONTROLADOS POR IA

#### 1. Desgaste das Runas:

- Aumenta com o uso repetitivo ou uso incorreto.
- Pode acelerar em puzzles mais difíceis.

 **Como Fazer:**

- A cada uso, reduzir um valor na variável **energia Runa**.
- Se **energia Runa <= 0**, desativar Runa e iniciar **timer de recarga**.

#### 2. Adaptação dos Puzzles:

- Modifica elementos dependendo do desempenho do jogador.

 **Como Fazer:**

- Implementar **Checkpoints** dentro do puzzle.
- Se o jogador demora, ativar pequenas **alterações** (ex.: mudar posição de um bloco, alterar tempo de reação).

#### 3. Ações do Vazio:

- Gera distorções visuais, sons e bloqueios com base em progresso.

 **Como Fazer:**

- Criar uma variável **nível Vazio**, que aumenta ao longo do jogo.
- Acionar efeitos (shader, som, partículas) conforme **nível Vazio** sobe.

---

### 5.3.2. VETORES DE INFLUÊNCIA NO JOGO

- **Prioridade do ambiente** (puzzles) em dificultar ou aliviar com base em:
  - Erros seguidos.
  - Tempo gasto.
  - Runas em recarga.

 **Como Fazer:**

- Definir pesos:
  - Ex: Erros = +2, Tempo = +1, Runas = +3.
- Somar os pesos e ativar eventos conforme o valor total.

---

### 5.3.3. BALANCEAMENTO DA DIFICULDADE

- Progressão **natural**, sem escolha direta de fácil/médio/difícil.
- A IA responde de forma a manter o jogador **desafiado, mas não frustrado**.

 **Como Fazer:**

- Dividir puzzles por zonas:
  - Zona 1 (puzzles 1-4): baixa reatividade.
  - Zona 2 (puzzles 5-8): média reatividade.
  - Zona 3 (puzzles 9-12): alta reatividade + Vazio constante.
  -

---

## 5.4. RESUMO DA IMPLEMENTAÇÃO

 **Como Fazer:**

1. **Estrutura de IA Reativa:**
  - Criar um **Game Manager** que monitora:
    - Uso de Runas.
    - Estado dos puzzles.

- Eventos do Vazio.

**2. Event Triggers:**

- Associar eventos a objetos:
  - Exemplo: ao tocar uma Runa -> evento desgaste.
  - Ao completar um puzzle -> evento redução do Vazio.

**3. Debug e Ajuste:**

- Testar com **logs visuais** e ajustar os pesos conforme o comportamento do jogador real.

{ imagem aqui!!! }

*(Fluxograma: Uso da Runa → Desgaste → Evento do Vazio → Feedback Visual)*

## 6. IMPLEMENTAÇÃO DO JOGO

Esta seção define os **recursos tecnológicos**, o **motor de jogo** e a **estrutura de classes** necessárias para a criação de **1soMath**. Embora o jogo não tenha modo multiplayer, detalharemos a arquitetura para **jogo single player**, com foco em **puzzles, Runas, IA reativa e HUD interativa**.

### 6.1. RECURSOS TECNOLÓGICOS

#### 6.1.1. HARDWARE

- **Computadores:**

- Processador **i5** ou superior, 8 GB RAM (Programação e Arte)
- GPU integrada (Intel UHD) ou dedicada (GeForce GTX 1050 ou superior)
- Armazenamento SSD (mínimo 256 GB)

- **Celulares para Teste:**

- Android 8.0 ou superior
- iOS 12 ou superior

- **Periféricos:**

- Mesa digitalizadora (opcional, para artes)
- Monitor Full HD

#### Como Fazer:

- Utilizar um notebook ou desktop **intermediário** com suporte a **Unity/Unreal**.
- Testar builds em pelo menos **1 dispositivo Android e 1 iOS** para garantir compatibilidade.

---

#### 6.1.2. SOFTWARE

- **Motor de Jogo:**

- **Unity 2021 LTS** ou superior (preferencial para 2.5D e mobile)
- Visual Studio Code (IDE para scripts C#)

- **Design Visual:**

- Adobe Photoshop / Illustrator ou Figma (mockups)

- Spine 2D / Unity Animator (animações leves)

- **Gerenciamento de Projeto:**

- Trello / Notion (organização de tarefas)
- GitHub (controle de versões)

- **Áudio:**

- Audacity (edição de áudio)
- Bfxr (geração de efeitos sonoros simples)

- **Outros:**

- Google AdMob SDK (anúncios para recarga de Runas)
- Firebase (salvamento de progresso online, opcional)

## Como Fazer:

- Instalar **Unity** com suporte para **Android/iOS build**.
- Configurar **AdMob** e **Firebase SDK** (se necessário).
- Criar projeto com pasta organizada: **Assets → Scripts, Prefabs, Art, Audio**.

## 6.2. MODELAGEM DE CLASSE

### 6.2.1. MOTOR DO JOGO – ESTRUTURA DE CLASSES SUGERIDA

#### 1. GAME MANAGER

- Controla o fluxo geral do jogo (início, progresso, fim).

##### Atributos:

- progressoAtual
- runasDesbloqueadas[]
- nivelVazio

##### Métodos:

- IniciarJogo()

- AtualizarProgresso()
- CarregarCena(cena)

#### Como Fazer:

- Criar como **singleton** em Unity.
- Controlar transições entre puzzles, salvar progresso.

## 2. PLAYERCONTROLLER

- Controla o personagem Pit e sua interação com o mundo.

#### Atributos:

- posicao
- runaAtiva
- energiaRuna

#### Métodos:

- Mover()
- UsarRuna()
- TrocarRuna()

#### Como Fazer:

- Usar **Input System** do Unity para gestos.
- Adicionar **colisores** para interação com objetos/puzzles.

## 3. RUNA

- Define comportamento de cada Runa.

#### Atributos:

- cor
- energia

- status (ativa, recarregando)

**Métodos:**

- Ativar()
- Desgastar()
- Recarregar()

 **Como Fazer:**

- Criar **ScriptableObject** com os dados das Runas.
- Anexar métodos ao HUD.

**4. PUZZLEMANAGER**

- Controla a lógica dos puzzles.

**Atributos:**

- estadoPuzzle (incompleto/completo)
- dicasDisponiveis

**Métodos:**

- IniciarPuzzle()
- VerificarSolução()
- LiberarProximo()

 **Como Fazer:**

- Criar scripts específicos por tipo de puzzle.
- Usar eventos de Unity para gatilhos visuais.

**5. VAZIOCONTROLLER**

- Gerencia eventos e distorções do Vazio.

**Atributos:**

- intensidade

- tempoUltimaInterferencia

**Métodos:**

- AtivarInterferencia()
- AtualizarVazio()

 **Como Fazer:**

- Utilizar **Shaders** e **post-process effects** para distorções visuais.
- Integrar com GameManager para ativação progressiva.

**6. HUDMANAGER**

- Controla os elementos visuais da tela.

**Atributos:**

- barraEnergiaRuna
- botaoTrocaRuna
- temporizadorRecarga

**Métodos:**

- AtualizarHUD()
- MostrarMensagem()
- TrocarHUSRuna()

 **Como Fazer:**

- Utilizar **Canvas UI** do Unity.
- Scripts de binding direto com atributos do **PlayerController**.

---

## 7. AUDIOMANAGER

- Gerencia trilha sonora e efeitos.

**Atributos:**

- trilhaAtual
- efeitosSonoros[]

**Métodos:**

- TocarTrilha(trilha)
- TocarEfeito(efeito)

 **Como Fazer:**

- Criar **singleton** com  **AudioSource** e  **AudioClips**.
- Chamar métodos via eventos de gameplay (interações, transições).

---

## 8. SAVEMANAGER

- Salva e carrega o progresso.

**Atributos:**

- dadosSalvos

**Métodos:**

- SalvarJogo()
- CarregarJogo()

 **Como Fazer:**

- Utilizar  **PlayerPrefs** ou salvar em arquivo JSON.
- Opcional: integrar com  **Firebase** para nuvem.

{ imagem aqui!!! }

(Diagrama de Classes: *GameManager* → *PlayerController* → *PuzzleManager / HUDManager / AudioManager*)



## 7. MONETIZAÇÃO E ESTRATÉGIA DE PUBLICAÇÃO

### 7.1. MONETIZAÇÃO

O modelo de monetização de **1soMath** será **gratuito com anúncios opcionais**, focando em manter a experiência fluida e **não-invasiva**, respeitando a imersão do jogador.

---

#### 7.1.1. MODELO DE RECEITA:

- **Anúncios Recompensados:**

- Assistir a um anúncio para **recarregar uma Runa** instantaneamente.
- Frequência: no máximo 1 a cada 10 minutos.
- Plataforma: **Google AdMob** ou **Unity Ads**.

- **Compras Internas (opcional):**

- Pacote de **Runas Ilimitadas**: remove o desgaste e recarga.
- Skins visuais para Pit e interface (ex.: tema escuro, cores alternativas de Runas).
- Compra para **remover anúncios**.

 **Como Fazer:**

- **Para anúncios:**

1. Criar conta no **Google AdMob** ou usar **Unity Ads** (mais direto via Unity).
2. Integrar SDK no projeto (Unity → Services → Ads).
3. Inserir **Ad Placements** nos momentos de recarga.

- **Para compras internas (IAP):**

1. Ativar **Unity IAP** no projeto.
2. Criar os produtos: "runas\_unlimited", "remove\_ads", "skin\_darkmode".
3. Script simples para compra:
4. `IAPManager.BuyProductID("runas_unlimited");`

## 7.2. ESTRATÉGIA DE PUBLICAÇÃO

### 7.2.1. PLATAFORMAS-ALVO:

- **Google Play Store** (Android)
- **Apple App Store** (iOS)

### 7.2.2. REQUISITOS PARA PUBLICAÇÃO:

- **Conta de Desenvolvedor:**

- Google Play: US\$ 25 (pagamento único)
  - Apple Store: US\$ 99/ano

- **Documentação Necessária:**

- Descrição do jogo.
  - Screenshots (padrão 1242x2208 px, 16:9).
  - Ícone do App (512x512 px).
  - Política de Privacidade (pode ser simples, focando em uso de anúncios e dados básicos).

### Como Fazer:

- **Google Play:**

1. Criar conta em [Google Play Console](#).
2. Preencher dados do app, políticas e faixa etária.
3. Fazer build **.apk ou .aab** e enviar para revisão.
4. Acompanhar relatórios e feedbacks.

- **Apple Store:**

1. Criar conta em [Apple Developer](#).
2. Usar **Xcode** (para builds iOS, mesmo se usando Unity).
3. Fazer build **.ipa**.
4. Submeter via **App Store Connect**.

---

### 7.2.3. MARKETING INICIAL:

- **Landing Page** simples com:
  - Trailer curto (30s).
  - Link direto para download.
  - Descrição emocional do jogo.
- **Social Media:**
  - Instagram com posts mostrando arte conceitual, curiosidades das Runas.
  - Lançamento com contagem regressiva + teaser.
- **Parcerias com microinfluenciadores:**
  - Jogadores de puzzle games.
  - Oferecer acesso antecipado.

#### Como Fazer:

- Criar **trailer** usando Unity Recorder + software de edição (ex: CapCut, Premiere).
- Criar perfis sociais e usar **hashtags**: #PuzzleGame #IndieDev #LogicGame
- Enviar **versão beta** para amigos e capturar feedback.

#### Resumo Final da Monetização e Publicação:

- **Ads Recompensados**: monetização leve e integrada ao gameplay.
- **Publicação** em Google Play / App Store, com baixo custo inicial.
- **Marketing simples e direto**, aproveitando a força visual e conceitual do jogo.



## 8. PLANO DE TESTES (QUALITY ASSURANCE)

Esta seção define as estratégias, tipos de testes, ferramentas e processos para garantir que **1soMath** seja lançado com **qualidade técnica e experiência imersiva**, sem falhas críticas.

### 8.1. OBJETIVOS DO PLANO DE TESTES

- Garantir que **todas as mecânicas** (Runas, puzzles, IA, HUD) funcionem corretamente.
- Identificar **bugs**, falhas de desempenho e problemas visuais.
- Validar que a **monetização** (ads, compras internas) esteja funcionando.
- Testar a **compatibilidade** em diferentes dispositivos Android e iOS.
- Avaliar a **usabilidade** (intuitividade dos controles e menus).

### 8.2. TIPOS DE TESTES REALIZADOS

#### 8.2.1. TESTE FUNCIONAL

- Verifica se **cada função** do jogo opera corretamente.
- Exemplo:
  - Runas desgastam e recarregam conforme esperado.
  - Puzzles só liberam após completar corretamente.

#### Como Fazer:

- Criar uma **lista de todas as funções** do jogo.
- Testar individualmente cada item (ex.: menu abre? runa ativa? HUD atualiza?).
- Usar ferramentas como **Unity Console** para logs.

---

#### 8.2.2. TESTE DE INTERFACE (UI/UX)

- Confere se **HUD** e **menus** estão funcionando visualmente e são intuitivos.
- Avalia **respostas ao toque** e **transições visuais**.

 **Como Fazer:**

- Testar em dispositivos reais: diferentes tamanhos de tela.
- Verificar se botões estão responsivos e visíveis.
- Pedir para um jogador externo usar o menu sem instruções (teste de navegação).

---

#### 8.2.3. TESTE DE DESEMPENHO

- Verifica **FPS (frames por segundo)**, carregamento e uso de memória.
- Testa **lag, travamentos** ou quedas de performance.

 **Como Fazer:**

- Ativar **stats de performance** no Unity.
- Usar **Profiler** para medir FPS e consumo.
- Testar em **dispositivos fracos e fortes**.

---

#### 8.2.4. TESTE DE COMPATIBILIDADE

- Garante que o jogo roda em diferentes:
  - Sistemas Android/iOS.
  - Resoluções de tela.
  - Configurações de hardware.

 **Como Fazer:**

- Compilar builds para **APK e IPA**.
- Testar em:
  - Android 8.0, 9, 10+.
  - iOS 12, 13+.
- Ajustar o **Canvas UI** para ser escalável.

**8.2.5. TESTE DE MONETIZAÇÃO**

- Confere se:
  - Anúncios carregam e recompensam.
  - Compras internas processam corretamente.

 **Como Fazer:**

- Usar **modo teste** do AdMob ou Unity Ads.
- Simular compras via **Google Play / Apple Sandbox**.

**8.2.6. TESTE DE USABILIDADE E EXPERIÊNCIA**

- Avaliar se o jogo é **intuitivo** e agradável.
- Receber feedback de **testadores reais**.

 **Como Fazer:**

- Fazer um **beta fechado** com 5-10 jogadores.
- Fornecer um **formulário** com perguntas como:
  - Teve dificuldades? Onde?
  - Qual puzzle foi mais interessante?
  - O sistema de Runas ficou claro?

### 8.3. CRONOGRAMA DE TESTES

Fase	Tipo de Teste
Teste Interno Inicial	Funcional, UI
Teste de Desempenho	Performance
Teste em Dispositivos	Compatibilidade
Teste de Monetização	Ads e Compras Internas
Beta com Feedback	Usabilidade, Bugs

### 8.4. FERRAMENTAS UTILIZADAS

- **Unity Profiler** – para medir desempenho.
- **Google Play Console** – testes beta Android e **TestFlight** – beta iOS.
- **Google Forms** – coleta de feedback.
- **Trello** – organização de bugs e melhorias.

### 8.5. CRITÉRIOS DE ACEITAÇÃO

- 100% dos puzzles e Runas funcionam sem bugs.
- Nenhum **crash** ou travamento em dispositivos testados.
- Ads e IAP testados com sucesso.
- Jogadores conseguem jogar **sem instruções extras**.
- Feedback positivo sobre controles e narrativa.