

Luis Felipe Carneiro Pereira<sup>1</sup> – Fatec Carapicuíba  
Andre Gueiros Nogueira<sup>2</sup> – Fatec Carapicuíba  
Victor Lopes Domingues<sup>3</sup> – Fatec Carapicuíba  
Larissa Barreto Storck da Costa<sup>4</sup> – Fatec Carapicuíba  
Dr. Luiz Sergio de Souza<sup>5</sup> – Fatec Carapicuíba (Orientador)

## GDD: Projeto Blitz



---

<sup>1</sup> - Tecnólogo em Eletrônica – e-mail: luis.pereira20@fatec.sp.gov.br

<sup>2</sup> - Segundo grau completo – e-mail: victor.domingues@fatec.sp.gov.br

<sup>3</sup> - Graduado em Sistemas da Informação – e-mail: andre.nogueira01@fatec.sp.gov.br

<sup>4</sup> - Segundo grau completo – e-mail: larissa.costa01@fatec.sp.gov.br

<sup>5</sup> - Doutor em Engenharia Eletrica – e-mail: luiz.souza71@fatec.sp.gov.br – Orientador

## **1. GAME DESIGN**

### **1.1. Gameplay**

#### **1.1.1. Imersão**

A imersão utilizada neste jogo consiste em explorar a curiosidade do jogador. Com uma interface simples e clara, para que qualquer tipo de jogador pudesse reconhecer o gênero do jogo, e mesmo não reconhecendo, sendo rápido em aprender e não interferir no comportamento do jogador.

#### **1.1.2. Estrutura de Missões e Desafios**

Fase 1: Subterrâneos do Central Park

- Derrotar os inimigos para coletar energia.
- Se esquivar das armadilhas.
- Procurar por itens nos objetos do cenário.
- Derrotar todos os inimigos da sala para desbloquear o caminho.
- Coletar as energias.
- Utilizar a energia coletada para desbloquear habilidades.
- Alternar de tiro ao final de cada sala.
- Explorar as salas que compõem os andares.

#### **1.1.3. Objetivos do jogo**

O objetivo principal do jogo, é que o personagem percorra os 10 andares e recupere a fonte de energia que se encontrar escondido no labirinto.

O objetivo de cada fase (Andar), é derrotar todos os inimigos.

Derrotando inimigos ou destruindo objetos no cenário é possível encontrar energia, que pode ser utilizada para adquirir novas habilidades.

#### **1.1.4. Fluxo do Jogo**

As salas são alocadas de maneira randômica sempre que uma nova partida se inicia ou o jogador avança de andar.

Os inimigos são distribuídos aleatoriamente por sala conforme o nível de dificuldade e tamanho da sala. Cada sala possui um número mínimo e máximo de inimigos.

### **1.2. Mecânica do Jogo**

#### **1.2.1. Regras Implícitas e Explícitas no Jogo**

- Em todas as salas devem existir inimigos aleatórios para o jogador enfrentar.
- Inimigos quando derrotados devem desaparecer da sala e deixar energia para o jogador coletar.
- O jogador pode coletar e equipar diferentes tiros.
- Derrotar todos os inimigos de uma sala permite que o jogador avance para a próxima.
- Desbloquear habilidades consome energia.
- As habilidades do jogador possuem tempo de recarga entre utilizações.
- Uma vida por partida.
- É possível se esconder dos inimigos atrás de objetos e paredes.
- Objetos no cenário podem ser destruídos para revelar itens escondidos.
- Atirar enquanto se movimenta.

#### **1.2.2. Física**

A física do jogo irá simular o ambiente real. Por exemplo: para que o jogador quebre uma das caixas, ele precisa atirar nelas.

### **1.2.3. Movimentação dos Personagens**

O personagem controlável é o robô Blitz, que possui 8 sentidos de movimentação.

Com a câmera na visão top down ele pode se movimentar para cima, para baixo, para o lado esquerdo, para o lado direito, para diagonal direita para cima, diagonal direita para baixo, diagonal esquerda para cima e diagonal direita para baixo.

### **1.2.4. Objetos**

Energia: São as moedas do jogo, que podem ser adquiridas toda vez quando um inimigo é derrotado, ou quando as caixas de madeira ou vasos de porcelana são quebrados.

Tiro inicial: Tiro inicial do jogador, possui poder e velocidade balanceados.

Tiro Triplo: Esse tiro dispara 3 tiros na direção que o jogador, cobre uma área maior com mais velocidade, porém com baixo poder.

Bola de fogo: O tiro de fogo possui um poder elevado com velocidade baixa, requer precisão para ser utilizado.

Vida: Pode ser adquirida dentro das caixas de madeira ou vasilhas de porcelana.

### **1.2.5. Mecânica e Combate**

Para derrotar os inimigos o jogador deve atirar neles até que a “barra de vida” deste se esvazie. E deve evitar ao máximo ficar próximo dos inimigos, ou sofrerá danos.

### **1.2.6. Economia e Mecânica de Troca**

Ao destruir as caixas, vasos de porcelanas ou derrotar os inimigos o jogador ganhará energia (moeda do jogo) que podem ser trocadas por habilidades.

Sempre que uma sala for concluída um tiro aleatório aparecerá na sala vazia para o jogar, que possui o poder de escolha se deseja ou não utilizar este tiro.

### 1.3. Projeto de Fases (Level Design)

Fase 1: Subterrâneo do Central Park

- **OBJETIVOS:** levar o jogador a explorar todas as salas, derrotar todos os inimigos dentro delas e adquirir energia para desbloquear as habilidades.
- **NÍVEL DE DIFICULDADE:** Médio. Os monstros nas salas possuem níveis de agressividade médio e baixo.

### 1.4. Projeto de Interface

#### 1.4.1. Sistema Visual

a) HUD (*Head-Up Display*): O elemento mostrado na tela do jogo será:

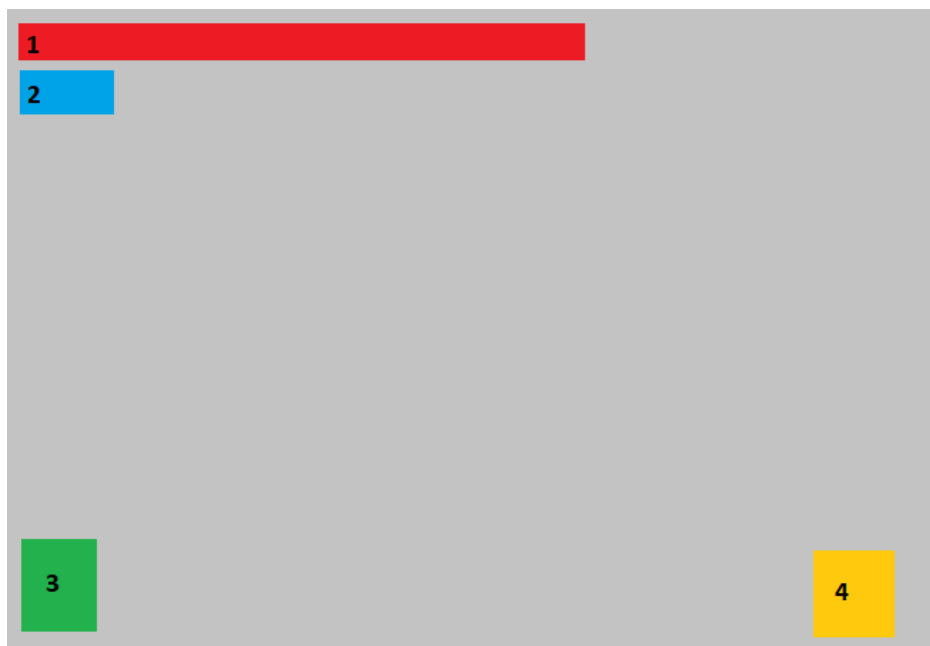


Figura 01 - Elementos fixos na tela

1. Barra de vida do jogador.

2. Quantidade de energia coletada.
3. Tiro equipado.
4. Habilidade equipada para uso rápido.

b) Menu *in game*: No menu serão mostradas as seguintes opções:

- Habilidades: Restaurar, Drenar, Clone, Escudo, Armadilha de fogo e Onda de Choque.
- Quadro explicativo habilidades: contém a explicação da skill, quantidade de energia para comprá-la e quanto tempo dura quando usada.

c) Sistema de renderização (*pipeline* de renderização): O jogo possui dois sistemas de renderização: um contexto simples em 2D e o WebGL, sendo o último geralmente mais rápido, além de possuir um número maior de recursos e suporte à maioria dos navegadores, à exceção do Internet Explorer.

Caso o WebGL se encontre desativado ou sem suporte no navegador, o jogo será automaticamente adaptado para a renderização de contexto simples.

d) Câmera: A câmera é ortográfica e mostra uma perspectiva top down ao jogador. O jogador não pode controlar seu ângulo de visão. A câmera irá seguir o jogador com um amortecimento em seu deslocamento e respeitando os limites de cada mapa.

#### **1.4.2. Sistema de Controle**

Os comandos de controle do jogo serão feitos através de um teclado QWERTY, e especificamente o controle de movimentos da personagem será feito através das setas de navegação do mesmo teclado ou nas teclas A, W, S e D. Sendo:

- A - Movimentar para a esquerda.
- D - Movimentar para a direita.
- S - Movimentar para a esquerda.
- W - Movimentar para cima.

Para acessar o menu in-game, usa-se a tecla espaço.

Para pausar/retornar o jogo, usa-se a tecla enter.

Para mirar, usa-se o ponteiro do mouse.

Para atirar, usa-se o botão esquerdo do mouse.

Para usar a skill adquirida, usa-se o botão direito do mouse.

#### 1.4.3. Fluxo de Telas

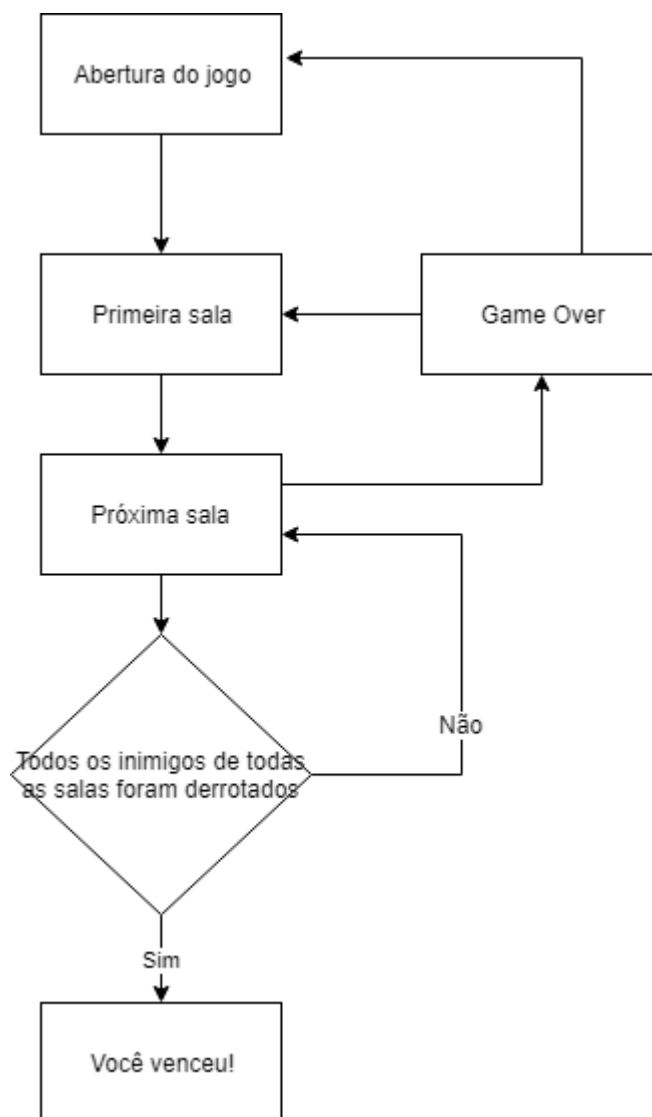


Figura 02 - Fluxo das Telas

- **ABERTURA DO JOGO:** Ao ser executado, o jogo trará uma tela onde será possível escolher entre as opções:
  - Jogar: Inicia o jogo.

- Opções: o jogador pode escolher qual música irá tocar no game, controlar o volume da música e dos efeitos sonoros, e escolher o idioma do jogo, que se encontra nas opções português ou inglês.
- Créditos: Informações sobre os desenvolvedores.
- Controles: mostra quais as teclas de ação do jogo.
- Sobre: História do jogo.
- Sair: Fecha o jogo.
- PRIMEIRA SALA: Sala segura, onde o jogador pode testar os controles do jogo sem nenhuma surpresa.
- PRÓXIMA SALA: Sala gerada aleatoriamente para o jogador explorar e enfrentar os desafios.
- GAME OVER: Aparece uma mensagem em caixa alta na tela escrito “Game Over”, o jogador não consegue mais se movimentar na tela e pode escolher “Reiniciar o Jogo” e voltar para a PRIMEIRA SALA ou “Voltar para o Menu”, será redirecionado para a ABERTURA DO JOGO.

## **1.5. Sistema de Inteligência Artificial**

### **1.5.1. Inimigos**

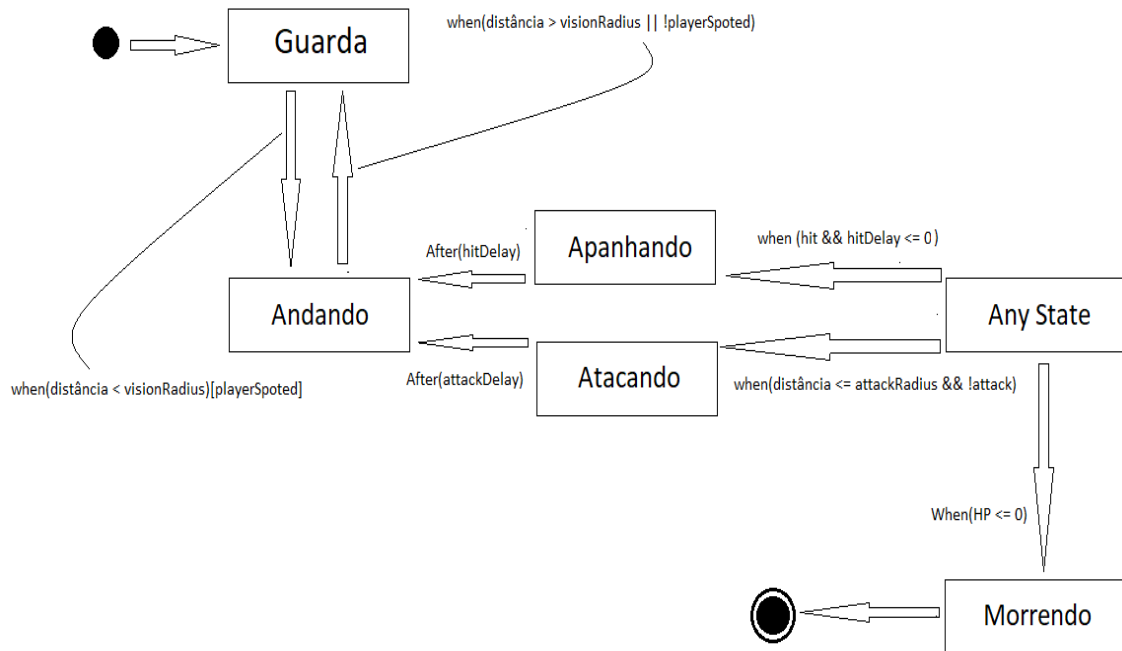
Os inimigos são controlados por uma máquina de estados.

A ativação dos inimigos acontece quando o jogador está dentro do raio de visão dos inimigos, para isso foi utilizado um raycast para calcular a distância, mesmo princípio é utilizado para habilitar o ataque contra o jogador.

Os inimigos possuem um ponto de partida fixo em cada sala, quando perdem o jogador de vista se movimentam para esse ponto e ficam em estado de espera.

Para a colisão é utilizado colisores no corpo do inimigos para representar o ponto que o jogador pode infligir dano e pequenos colisores que são instanciados somente durante o movimento de ataque dos inimigos para checar se ocorreu contato com o jogador.





Float distância: cálculo da distância entre o jogador e o inimigo.

Float visionRadius: Raio de visão do inimigo.

Float attackRadius: Raio de ataque do inimigo.

Bool playerSpoted: Marca que o jogador está dentro da visão do raio e está visível para o inimigo.

Float HP: Quantidade dano que inimigo consegue suportar antes de morrer.

Bool attack: Controla o intervalo entre os ataques.

Bool hit: Informa quando o inimigo sofre dano.

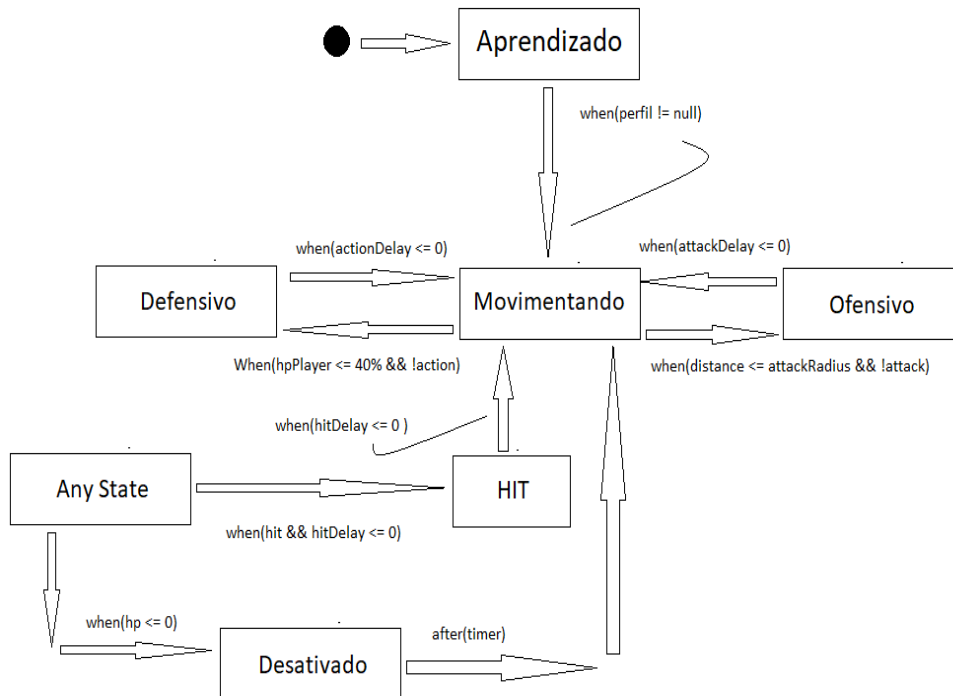
Float attackDelay: Tempo de execução da ação de ataque.

Float hitDelay: Controla a execução da animação e o tempo de invencibilidade após receber dano.

### 1.5.2. Inteligência Artificial Auxiliar

O npc ajudante H6 é controlado por uma máquina de estados que possui pequenas alterações conforme a classificação do perfil do jogador utilizando a técnica de IA KNN.

O H6 calcula a distância do jogador para se movimentar e a dos inimigos para atacar utilizando um cálculo de distância de vetores.



String perfil: Variável que define a classificação do comportamento do jogador, utiliza técnica knn com os parâmetros (quantidade de tiros, quantidade de acerto, tempo gasto na sala, quantidade de caixas destruídas, quantidade de dano causado, quantidade de hp perdido, quantidade de itens de cura utilizados).

Float attackDelay: controla o tempo do intervalo entre os ataques.

Float actionDelay: controla o tempo do intervalo entre o uso das habilidades.

Float distância: cálculo da distância entre o jogador e o inimigo.

Float attackRadius: Raio de ataque do inimigo.

Bool attack: indica que o npc está executando ação de ataque.

Bool action: indica que o npc está executando uma ação de habilidade.

Float hpPlyer: Quantidade de hp o jogador.

Bool hit: indica quando o npc foi acertado por um inimigo.

Float hitDelay: Controla a execução da animação e o tempo de invencibilidade após receber dano.

Float HP: Quantidade dano que o npc consegue suportar antes de desativar

Float timer: tempo que o npc permanece desativado.

## 2. ARTE

### 2.1. Arte Conceitual



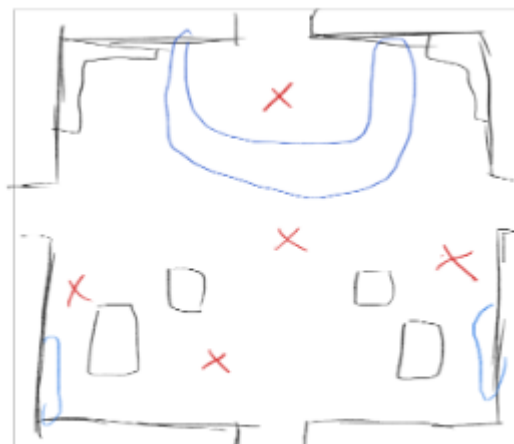
Figura 03 - Esboço logo do jogo



Figura 04 - Processo de criação do robô Blitz



X - SPALL DE INIMIGOS  
O - QUEBRÁVEIS








X - SPALL DE INIMIGOS  
O - QUEBRÁVEIS

Figura 05 - Esboço salas com localização dos inimigos e objetos quebráveis

### 2.2. Asset List

### 2.2.1. Personagens

 <p>Figura 06 - Robô Blitz</p>	 <p>Figura 07 - Slime</p>
 <p>Figura 08 - H6</p>	 <p>Figura 10 - Torreia</p>
 <p>Figura 09 - Orc</p>	

### 2.2.2. Ambiente



Figura 11 - Abertura do jogo



Figura 12 - Sala do jogo

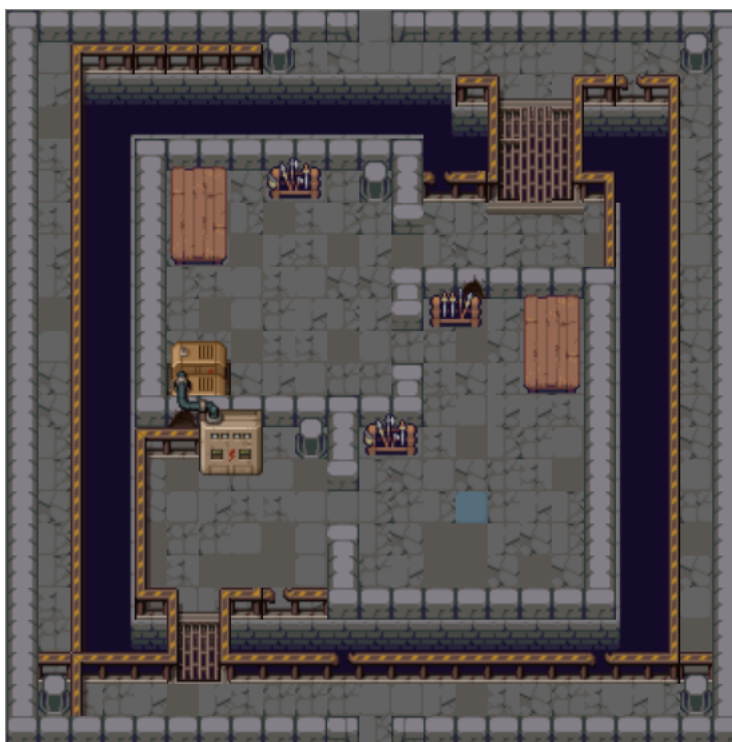


Figura 13 - Sala do jogo

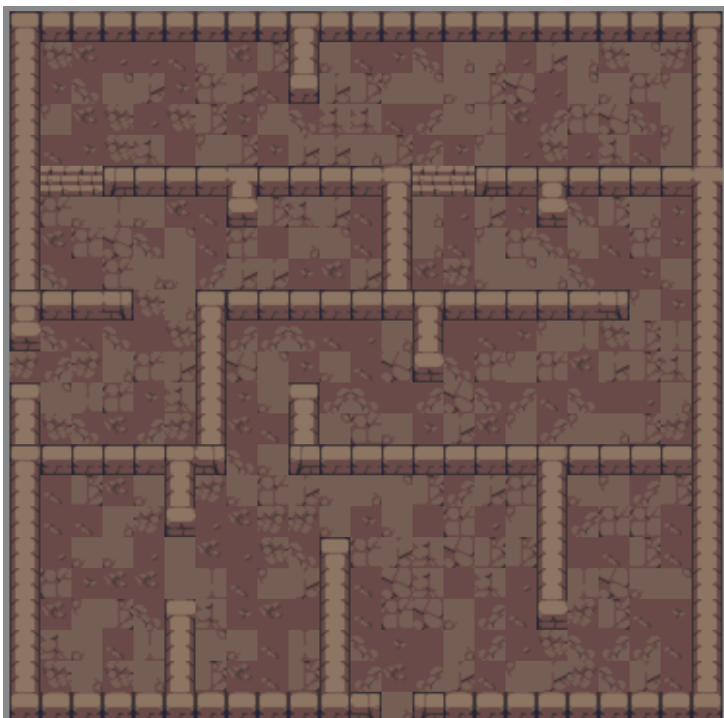


Figura 14 - Sala do jogo

### 2.2.3. Animações



Figura 15 - Frames de animação: Blitz

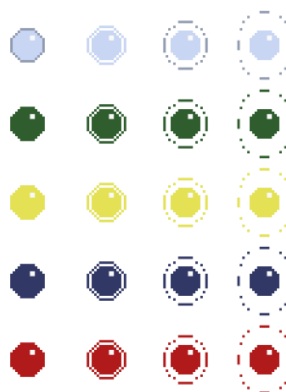


Figura 16 - Frames de animação: H6

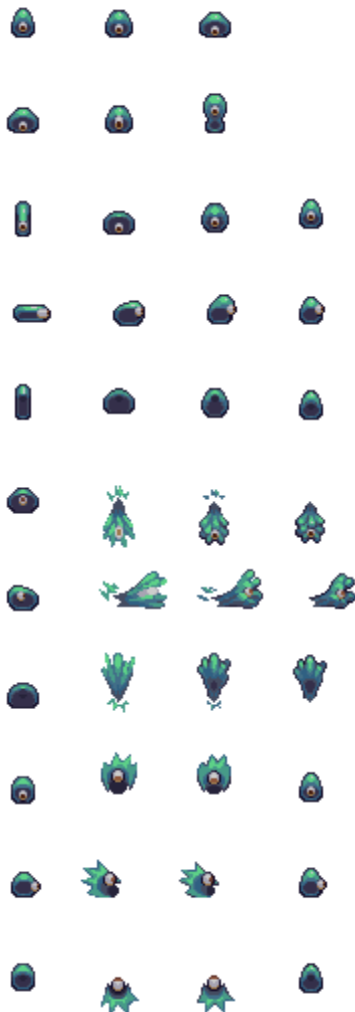


Figura 17 - Frames de animação: Slime

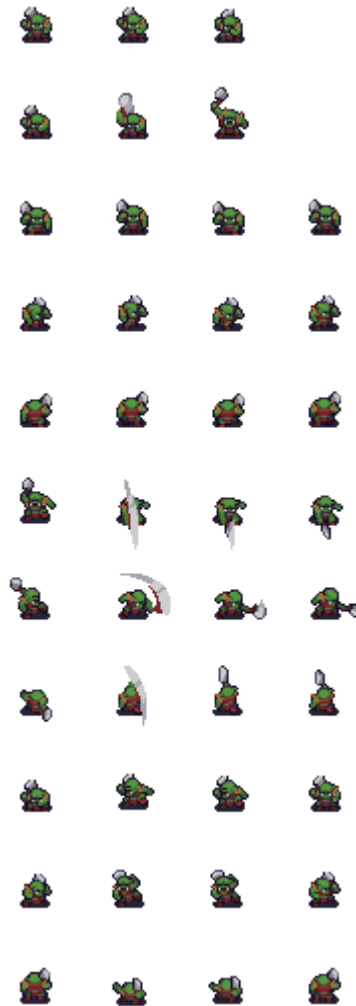


Figura 18 - Frames de animação: Orc



Figura 19 - Frames de animação: Torreta

#### 5.2.4. Interfaces



Figura 20 - Barra de Vida



Figura 21 - Quantidade de energia coletada

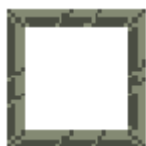


Figura 22 - Slot de tiro e skill vazio

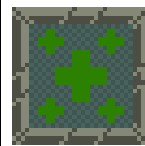


Figura 23 - Slot de skill equipado com

	restauração
 <p>Figura 24 - Slot de skill equipado com escudo</p>	 <p>Figura 25 - Slot de skill equipado com onda de choque</p>
 <p>Figura 26 - Slot de skill equipado com roubo de vida</p>	 <p>Figura 27 - Slot de skill equipado com clone</p>
 <p>Figura 28 - Slot de skill equipado com armadilha de fogo</p>	 <p>Figura 29 - Slot de tiro equipado com o tiro triplo</p>
 <p>Figura 30 - Slot de tiro equipado com o tiro padrão</p>	 <p>Figura 31 - Slot de tiro equipado com o tiro de fogo</p>

### 2.2.5. Outros





 <p>Figura 32 - Animação de restauração de vida sobre o personagem</p>	 <p>Figura 33 - Frames de animação: Tiro de fogo</p>
 <p>Figura 34 - Frames de animação: Tiro padrão</p>	 <p>Figura 35 - Frames de animação: Energia</p>






 Figura 36 - Frames de animação: Núcleo reparador de vida	 Figura 37 - Vaso de porcelana
 Figura 39 - Frames de animação: morte dos inimigos	 Figura 38 - Caixa de madeira

### 2.3. Guia de Cores e Estilos Gráficos

O aspecto visual do jogo deve ser com cores escuras, pois são características de cavernas e ambientes sombrios. Com relação às cores, segue abaixo a paleta usada no desenvolvimento do jogo:

#### Robô Blitz

 Figura 40 - Paleta da hélice <ul style="list-style-type: none"> <li>• #22263B, #44485F, #232846, #141934, #0A0E27</li> </ul>	 Figura 41 - Paleta do corpo <ul style="list-style-type: none"> <li>• #0A0E27, #232846, #141934, #343B61</li> </ul>
 Figura 42 - Paleta olhos e boca <ul style="list-style-type: none"> <li>• #000, #818288</li> </ul>	

<b>Salas</b>	<b>Inimigos</b>
--------------	-----------------



Figura 43 - Paleta sala

- #140B28, #FFF, #636363, #4D403F, #4A403F, #6B635C, #595652, #444841, #323C39, #B5B7BB, #847E87, #473C47, #757375, #695A5D, #3D3334, #694A48, #663931, #A33939, #755E54, #876158, #8F6F64, #786757, #8C7562, #856A56, #795642, #956854, #8F563B, #BD7453, #B33A12, #AC7751, #CF8C4E, #DF7126, #DC3F0B, #9A8464, #CCBAA9, #787160, #847C68, #B3A370, #C0B505, #FFF000, #FBF236, #60B34F, #60B34F, #75D661, #4173AA, #5DB3FF, #222034, #26233A, #7C3EB7, #9D61D6, #45283C, #5E4854, #826068, #966F74.



Figura 48 - paleta inimigos

- #140B28, #636363, #CCC, #CFCFCF, #444841, #232826, #323C39, #A7A4B3, #847E87, #663931, #521B1B, #843131, #742525, #933434, #9D2323, #F7C6C6, #523521, #63412A, #8F563B, #9D794D, #8E5D3A, #8F5E39, #411A0C, #B7733F, #426336, #497E36, #589740, #5DD188, #479B88, #426E88, #306082, #444C61, #26233A, #342F4A.

**Explosão dos inimigos**

**Tiros**



Figura 45 - Paleta explosão dos inimigos

- #3E231A, #8A4A37, #B95234, #D77B49, #FB9C68



Figura 46 - Paleta tiros

- #858876, #4C4E44, #702020, #861515, #F9BAA1, #59310F, #6A370C, #C65E2C, #C69B2C, #282D6D, #2E358E, #565DBA, #878EDB, #544663, #635275

#### Barra de vida



Figura 47 - paleta barra de vida

- #858876, #696A64, #4C4E44, #6B2222, #A15252, #882424, #DA7575.

#### Abertura do Jogo



Figura 44 - Paleta abertura do jogo

- #474646, #5A5A5A, #968781, #5C534F, #453F3C, #68625C, #5C564F, #99A2AB, #7E2F2F, #6A5C56, #472E24, #382118, #7B6157, #2A160F, #AB6740, #333A2E, #394531, #324724, #1C374D, #457CA7, #A7C7E1, #22263B, #44485F, #232846, #141934.

### 3. DOCUMENTAÇÃO TÉCNICA

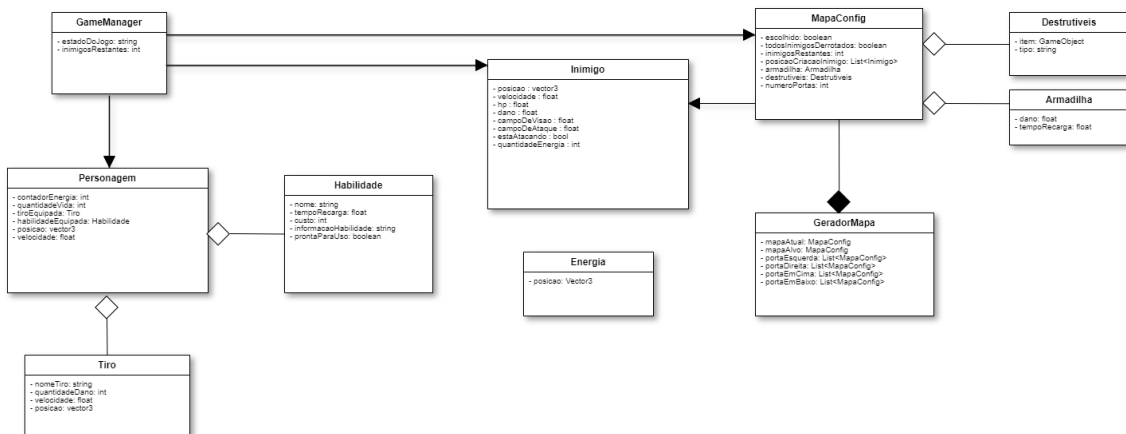
#### 3.1. Requisitos de sistema

Para jogar, o usuário deve estar online, e possuir instalado em seu computador, um navegador web que execute HTML5.

#### 3.2. Engenharia de Software

##### 3.2.1. Diagramas de Classe Conceitual

O diagrama de classes a seguir, representa a estrutura conceitual do Jogo Projeto Blitz:



Descrição das Classes:

A classe *Personagem* possui os atributos abaixo:

- ContadorEnergia: Contabiliza as energia coletadas - Tipo: Inteiro;
- quantidadeVida: Quantidade de vidas do jogador - Tipo: Inteiro;
- tiroEquipado: Tipo de tiro que o jogador está usando no momento - Tipo: Tiro;
- habilidadeEquipada: Tipo de habilidade que o jogador está usando no momento - Tipo: Habilidade;
- posicao: Posicao atual do jogador - Tipo: Vector3;
- velocidade: velocidade de deslocamento do personagem - Tipo: Flutuante;

A classe *Inimigo* possui os atributos abaixo:

- posicao: posicao atual do inimigo - Tipo: Vector3;
- velocidade: velocidade de deslocamento do inimigo - Tipo: Flutuante;
- hp: quantidade de vida do inimigo - Tipo: Flutuante;
- dano: quantidade de dano sofrido - Tipo: Flutuante;
- campoDeVisao: raio do campo de visão do inimigo -Tipo: Flutuante;
- campoDeAtaque: raio do campo de ataque do inimigo - Tipo: Flutuante;
- estaAtacando: verifica se o inimigo está em modo de ataque - Tipo: Booleano;
- quantidadeEnergia: quantidade de energia que o inimigo deixa quando morre - Tipo: Inteiro;

A classe *Habilidade* possui os atributos abaixo:

- nome: nome da habilidade - Tipo: String;
- tempoRecarga: quantidade de segundos para o jogador conseguir usar a habilidade novamente - Tipo: Flutuante;
- custo: quantidade de energia para a habilidade ser comprada - Tipo: Inteiro;
- informacaoHabilidade: descrição da habilidade - Tipo: String;
- prontaParaUso: verifica se a habilidade pode ser utilizada - Tipo: Booleano;

A classe *Tiro* possui os atributos abaixo:

- nome: nome tiro - Tipo: String
- quantidadeDano: quantidade de dano do tiro - Tipo: Inteiro
- velocidade: velocidade que o tiro percorre -Tipo: Flutuante
- posicao: posicao do ícone do tiro para ser equipado - Tipo: Vector3

A classe *Energia* possui os atributos abaixo:

- posicao: posicao da energia no mapa - Tipo: Vector3

A classe *Armadilha* possui os atributos abaixo:

- dano: quantidade de dano que armadilha da quanto está em contato com o jogador - Tipo: Flutuante;
- tempoRecarga: tempo em segundos para a armadilha funcionar novamente - Tipo: Flutuante;

A classe *Destrutíveis* possui os atributos abaixo:

- item: tipo de item que possui dentro do objeto destrutível - Tipo: GameObject;
- tipo: tipo do objeto destrutível - Tipo: String;

A classe *GeradorMapa* possui os atributos abaixo:

- mapaAtual: sala atual - Tipo: MapaConfig;
- mapaAlvo: proxima sala - Tipo: MapaConfig;
- portaEsquerda: mapas que possuem entrada pela porta esquerda - Tipo: List<MapaConfig>;
- portaDireita: mapas que possuem entrada pela porta direita - Tipo: List<MapaConfig>;
- portaEmCima: mapas que possuem entrada pela porta de cima - Tipo: List<MapaConfig>;
- portaEmBaixo: mapas que possuem entrada pela porta de baixo - Tipo: List<MapaConfig>;

A classe *MapaConfig* possui os atributos abaixo:

- escolhido: verifica se algum mapa já foi escolhido -Tipo: Booleano
- todosInimigosDerrotados: verifica se todos os inimigos da sala foram derrotados - Tipo: Booleano
- inimigosRestantes: verifica os inimigos restantes Tipo: Inteiro
- posicaoCriacaoInimigo: posição dos inimigos na sala - List<Inimigo>
- armadilha: posição das armadilhas criadas na sala - Tipo: Armadilha

- destrutíveis: posição dos objetos destrutíveis criados na sala -  
Tipo: Destrutíveis
- numeroPortas: numero de portas da sala - Tipo: Inteiro

### 3.2.2. Diagramas de Caso de Uso

Caso de Uso 1 – Jogar primeiro andar

Tabela 01 - Caso de Uso 1

<b>ID do Caso de Uso:</b>	1
<b>Nome do Caso de Uso:</b>	Jogar na masmorra
<b>Criado por:</b>	Storck, Larissa
<b>Data de criação:</b>	
<b>Última atualização realizada por:</b>	Storck, Larissa
<b>Data da última atualização:</b>	10/05/2019
<b>Atores</b>	Jogador
<b>Propósito:</b>	O jogador deve explorar a sala e derrotar todos os inimigos, coletar as energias para comprar habilidades, e ir para a próxima sala
<b>Pós-condições:</b>	1. Jogador passa para a próxima fase ou perde vida e reinicia o jogo.

Tabela 02 - Fluxo Básico de Eventos

Fluxo básico de eventos	
<b>Ações do Ator (Jogador):</b>	<b>Ações do Sistema:</b>
1.O jogador exterminar todos os inimigos da sala	
2. O jogador coleta as energias	
3. Jogador quebra os vasos ou caixas	4. O sistema contabiliza os pontos adquiridos com a coleta das energias
5. O jogador compra a habilidade que acredita que pode mais ajudar nos combates	
	6. O sistema exibe uma tela com as habilidades que o jogador pode adquirir
9. O jogador passa para a próxima sala	

	7. O sistema retira pontos de energia correspondentes a habilidade comprada pelo jogador
	8. O sistema libera as portas da sala quando todos os inimigos são derrotados

### 3.3. Software(s) Secundário(s)

Para o desenvolvimento do projeto serão necessários os *softwares* secundários listados abaixo:

- Microsoft Word
- Microsoft Excel
- Piskel
- Photoshop
- Tiled
- Visual Studio 2017

### 3.4. Game Engine

Utilizamos a engine Unity 2018.

### 3.5. Programação

A escolha da programação C# foi devido a familiaridade do grupo pela linguagem e pela facilidade de utilizar uma engine (unity) que possui funcionalidade multiplataforma com apenas uma codificação.

### 3.6. Scripting

Abaixo seguem as definições da linguagem C#, utilizadas pela engine Unity.

A linguagem usada no Unity é chamada C # (pronuncia-se C-sharp). Todas as linguagens com as quais o Unity opera são linguagens de script orientadas a objeto. Como qualquer idioma, as linguagens de script têm sintaxe ou partes da fala, e as partes principais são chamadas de variáveis, funções e classes.