

CENTRO UNIVERSITÁRIO FEI

Hugo Linhares Oliveira

João Pedro Rosa Cezarino

Thales de Oliveira Lacerda

Vitor Martins Oliveira

GRUPO C: Hero's Journey

São Bernardo do Campo

2023

Hugo Linhares Oliveira

João Pedro Rosa Cesarino

Thales de Oliveira Lacerda

Vitor Martins Oliveira

GRUPO C: Hero's Journey

Trabalho de Desenvolvimento de Jogos apresentado ao Centro Universitário FEI, como parte dos requisitos necessários para aprovação na disciplina de Desenvolvimento de Jogos Digitais do curso de Ciência da Computação, orientado pelo Prof. Dr. Fagner de Assis Moura Pimentel.

São Bernardo do Campo

2023

SUMÁRIO

1	INTRODUÇÃO	4
1.1	ESTRUTURA DO TRABALHO	4
2	PÚBLICO ALVO	5
3	ESTÉTICA	6
4	DINÂMICA	7
5	MECÂNICA	8
5.0.1	Algoritmos	8
5.0.2	Atores e seus componentes	24
5.0.3	Sprites	25
5.0.4	Backgrounds	26
6	PROTOTIPAÇÃO NO PAPEL	27
7	DESENVOLVIMENTO	30
8	TESTES	31
8.0.1	Relato 1	31
8.0.2	Relato 2	31
8.0.3	Relato 3	31
9	RESULTADO FINAL	33

RESUMO

Este trabalho tem como objetivo apresentar uma análise do jogo Hero's Journey, um jogo de aventura em 2D com vista lateral, que coloca o jogador na pele do guerreiro Axel, em uma missão para resgatar Raven, uma nobre guerreira raptada pelo Clã Blood Wolves. O jogo apresenta uma série de desafios e inimigos, incluindo bosses principais de cada fase, como Morgath, Draven, Ravenna e Lilith. Além disso, o jogo oferece uma variedade de cenários, desde florestas até o castelo inimigo, bem como personagens místicos, como Zephyrus, o Rei do Clã Moonlight Sentinels. A análise deste jogo permitirá compreender melhor as características dos jogos de aventura em 2D com vista lateral e como Hero's Journey se destaca dentro deste gênero.

1 INTRODUÇÃO

Os jogos de aventura em 2D com vista lateral, como Metal Slug e Mario, são bastante populares entre jogadores de todas as idades. Hero's Journey é um desses jogos, que coloca o jogador na pele de Axel, um guerreiro que precisa resgatar Raven, uma nobre guerreira raptada pelo Clã Blood Wolves. O jogo oferece uma experiência de ação intensa, com cenários variados e personagens místicos que adicionam elementos de fantasia à história.

1.1 ESTRUTURA DO TRABALHO

O restante deste trabalho é dividido da seguinte maneira:

No capítulo 2, é apresentado o público alvo para o qual o jogo está sendo desenvolvido.

No capítulo 3, é apresentada toda a história do jogo e o impacto que deve ter sobre o jogador.

No capítulo 4, são apresentados os elementos de dinâmica do jogo.

No capítulo 5, são apresentados os elementos de mecânica do jogo.

No capítulo 6, São apresentados os elementos de prototipação em papel.

No capítulo 7, é apresentada a implementação do jogo.

No capítulo 8, são apresentados os testes realizados com os jogadores.

No capítulo 9, é apresentado o resultado final do trabalho.

2 PÚBLICO ALVO

O jogo deve agradar principalmente as pessoas que gostam de jogos de plataforma como, por exemplo, jogadores de Mario e Metal Slug. Portanto, deve agradar o público majoritariamente mais velho que busca um RPG que evoca os gráficos de jogos nostálgicos. Acreditamos que ao ressuscitar jogos que têm perdido popularidade e que são considerados "antigos" ou "ultrapassados", conseguiremos trazer de volta uma comunidade que está sendo progressivamente esquecida.

3 ESTÉTICA

O jogo Hero's Journey é um jogo de aventura em 2D com vista lateral, com inspiração em jogos clássicos como Metal Slug e Mario. A história se passa em um reino governado pelo Rei Zephyrus, onde o Clã Moonlight Sentinels vive pacificamente até que a nobre guerreira do clã é raptada pelos integrantes do Clã Blood Wolves. O personagem principal, Axel, é um guerreiro nórdico em busca de resgatar sua companheira.

Durante a jogabilidade, os sentidos aguçados serão o visual e auditivo, com gráficos e efeitos sonoros nostálgicos de jogos de plataforma. A história do jogo evolui conforme Axel avança nas fases, enfrentando inimigos e chefões. Há a possibilidade de um final alternativo, onde após o resgate de sua guerreira, Axel descobre que Raven faz parte do clã inimigo e há uma traição.

Para orientar os jogadores, o jogo terá formas de orientação diretas e indiretas. Na orientação direta, haverá uma caixa de texto no início do jogo informando os controles básicos. Além disso, um diagrama visual mostrará ao jogador a possibilidade de interação com o mouse quando um novo item for coletado. Já na orientação indireta, setas indicarão os itens do cenário com possibilidade de interação. O design de áudio também será utilizado para transmitir sensações aos jogadores.

Essas formas de orientação visam proporcionar uma experiência mais fluida e agradável aos jogadores, além de garantir que eles possam compreender e aproveitar ao máximo tudo o que o jogo tem a oferecer. Com isso, esperamos que os jogadores possam se divertir e explorar cada detalhe do jogo Hero's Journey.

4 DINÂMICA

“Hero’s Journey” apresentará decisões discerníveis desde o início, ou seja, decisões que deixarão claro para o jogador as consequências de suas escolhas. No entanto, ao se aproximar do final, será incluída uma decisão de dois gumes, oferecendo duas opções que levarão o jogador a um final feliz ou triste. Isso criará uma situação em que a escolha do jogador terá um impacto significativo na história e no desfecho do jogo.

5 MECÂNICA

5.0.1 Algoritmos

Codigos/ArmaScript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ArmaScript : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }

    void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "Inimigo")
        {
            Destroy(collision.gameObject);
        }
        else if (collision.gameObject.tag == "Player")
        {
            GameManager.instance.DecreaseLife(1);
        }
    }
}
```

```
}
```

Codigos/ArmaScriptBoss.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class ArmaScriptBoss : MonoBehaviour
{
    private int life = 5;

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }

    void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "Boss")
        {
            if ((life - 1) == 0)
            {
                life = 0;
                Destroy(collision.gameObject);

                string currentScene = SceneManager.GetActiveScene
                    ().name;
            }
        }
    }
}
```

```

        switch(currentScene) {
            case "FinalAlternativo":
                SceneManager.LoadScene("EndSceneAlternativo");
                break;
            case "Final1":
                SceneManager.LoadScene("EndScene1");
                break;
            default:
                break;
        }
    }
    else
    {
        life--;
    }
}
else if (collision.gameObject.tag == "Player")
{
    GameManager.instance.DecreaseLife(1);
}
}
}
}

```

Codigos/CoinManager.cs

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CoinManager : MonoBehaviour
{
    public GameObject coinPrefab;

    private void Start()

```

```
{  
    GenerateCoin();  
}  
  
public void GenerateCoin()  
{  
    GameObject coin = Instantiate(coinPrefab);  
    coin.transform.position = new Vector2(Random.Range(-5f,  
        5f), Random.Range(0f, 0f));  
    coin.SetActive(true);  
}  
}
```

Codigos/CoinScript.cs

```
using System.Collections;  
using System.Collections.Generic;  
using UnityEngine;  
  
public class CoinScript : MonoBehaviour  
{  
    public int coinValue = 1;  
  
    private void OnTriggerEnter2D(Collider2D other)  
    {  
        if (other.gameObject.CompareTag("Player"))  
        {  
            gameObject.SetActive(false);  
            GameManager.instance.IncreaseScore(coinValue);  
        }  
    }  
  
    void Start()  
    {
```

```
}

void Update()
{

}

}
```

Codigos/CutSceneManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class CutSceneManager : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {
        Invoke("LoadGameScene", 10f);
    }

    // Load the Game scene
    void LoadGameScene()
    {
        string currentScene = SceneManager.GetActiveScene().name
        ;
        switch(currentScene) {
            case "Final1CutScene":
                SceneManager.LoadScene("Final1");
                break;
            case "FinalAlternativoCutScene":
                SceneManager.LoadScene("FinalAlternativo");
                break;
        }
    }
}
```

```
        default:
        case "CutScene":
            SceneManager.LoadScene("Game");
            break;
    }
}

// Update is called once per frame
void Update()
{
}

}
```

Codigos/GameManager.cs

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour
{
    public static GameManager instance;

    public int score = 0;
    public Text scoreText;

    public int life = 15;
    public Text lifeText;

    public void Start()
    {
        Invoke("HideInstructions", 15f);
    }
}
```

```
private void HideInstructions()
{
    GameObject instructions = GameObject.Find("Instrucoes");
    if (instructions != null) instructions.SetActive(false);
}

private void Awake()
{
    if (instance == null)
    {
        instance = this;
    }
    else if (instance != this)
    {
        Destroy(gameObject);
    }
    DontDestroyOnLoad(gameObject);
}

public void IncreaseScore(int value)
{
    score += value;
    scoreText.text = score.ToString();
}

public void DecreaseLife(int value)
{
    if (life - value == 0) {
        // reset life
        life = 15;
        lifeText.text = life.ToString();
        // reset score
        score = 0;
        scoreText.text = score.ToString();
        // reset scene
    }
}
```

```
        SceneManager.LoadScene("Game");

        return;
    }

    life -= value;
    lifeText.text = life.ToString();
}

}
```

Codigos/InimigoScript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;
using System.Threading;

public class InimigoScript : MonoBehaviour
{

    private Animator playerAnimator;
    private Rigidbody2D playerRb;
    private SpriteRenderer spriteRenderer;
    public bool Grounded;
    public int idAnimation;
    public Transform groundCheck; //detectar se o personagem
        esta pisando no chao

    private float h, v;
    public float speed;
    public float jumpForce;
    public Collider2D standing, crouching; //colisor em pe e
        agachado

    //ARMAS
    public GameObject[] armas;
```

```
public Camera minhaCamera;  
// Start is called before the first frame update  
void Start()  
{  
    playerAnimator = GetComponent<Animator>();  
    playerRb = GetComponent< Rigidbody2D >();  
    spriteRenderer = GetComponent< SpriteRenderer >();  
  
}  
  
void FixedUpdate()  
{  
}  
  
// Update is called once per frame  
void Update()  
{  
  
    Input.GetButtonDown("Fire1");  
    playerAnimator.SetTrigger("attack");  
    idAnimation = 0;  
    playerAnimator.SetBool("grounded", true);  
}  
  
void controleArma(int id)  
{  
    foreach (GameObject o in armas)  
    {  
        o.SetActive(false);  
    }  
  
    if (id >= 0 && id < armas.Length)  
    {  
        armas[id].SetActive(true);  
    }  
}
```

```
    }

}

void OnCollisionEnter2D(Collision2D col)
{
}

void OnCollisionExit2D(Collision2D col)
{
}

}
```

Codigos/MenuManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MenuManager : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

```

public void PlayGame()
{
    SceneManager.LoadScene("CutScene");
}
}

```

Codigos/ObjectPersist.cs

```

using UnityEngine;

public class ObjectPersist : MonoBehaviour
{
    private static ObjectPersist instance;

    private void Awake()
    {
        if (instance == null)
        {
            instance = this;
            DontDestroyOnLoad(transform.gameObject);
        }
        else
        {
            Destroy(gameObject);
        }
    }
}

```

Codigos/playerScript.cs

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;
using System.Threading;
using UnityEngine.SceneManagement;

```

```
public class playerScript : MonoBehaviour
{
    private Animator playerAnimator;
    private Rigidbody2D playerRb;
    private SpriteRenderer spriteRenderer;
    public bool Grounded;
    public int idAnimation;
    public Transform groundCheck; //detectar se o personagem
        esta pisando no chao

    private float h, v;
    public float speed;
    public float jumpForce;
    public Collider2D standing, crouching; //colisor em pé e
        agachado

    //ARMAS
    public GameObject[] armas;

    public Camera minhaCamera;
    // Start is called before the first frame update
    void Start()
    {
        playerAnimator = GetComponent<Animator>();
        playerRb = GetComponent<Rigidbody2D>();
        spriteRenderer = GetComponent<SpriteRenderer>();

        foreach(GameObject o in armas)
        {
            o.SetActive(false);
        }
    }
}
```

```
void FixedUpdate()
{
    Grounded = Physics2D.OverlapCircle(groundCheck.position,
        0.02f);
    playerRb.velocity = new Vector2(h * speed, playerRb.
        velocity.y);
}

// Update is called once per frame
void Update()
{
    h = Input.GetAxisRaw("Horizontal");
    v = Input.GetAxisRaw("Vertical");

    if (v < 0)
    {
        if (h >= 0)
        {
            idAnimation = 2;
            spriteRenderer.flipX = false;
            h = 0;
        }
    }
    else
    {
        idAnimation = 2;
        spriteRenderer.flipX = true;
        h = 0;
    }
}
else if (h > 0)
{
    idAnimation = 1;
    spriteRenderer.flipX = false;
```

```
    }

    else if (h < 0)
    {

        idAnimation = 1;
        spriteRenderer.flipX = true;
    }

    else
    {

        idAnimation = 0;
    }

    if (Input.GetButtonDown("Fire1"))
    {

        if (spriteRenderer.flipX == true)
        {

            foreach (GameObject o in armas)
            {

                o.GetComponent<SpriteRenderer>().flipX =
                    true;
            }

            playerAnimator.SetTrigger("atack");
            idAnimation = 0;
        }

        else
        {

            foreach (GameObject o in armas)
            {

                o.GetComponent<SpriteRenderer>().flipX =
                    false;
            }

            playerAnimator.SetTrigger("atack");
            idAnimation = 0;
        }
    }
}
```

```
        if (Input.GetButtonDown("Jump") && Grounded)
        {
            playerRb.AddForce(new Vector2(0, 200));
        }

        if (v < 0 && Grounded)
        {
            crouching.enabled = true;
            standing.enabled = false;
        }
        else if (v >= 0 && Grounded)
        {
            crouching.enabled = false;
            standing.enabled = true;
        }

        playerAnimator.SetBool("grounded", Grounded);
        playerAnimator.SetInteger("idAnimation", idAnimation);
        playerAnimator.SetFloat("speedY", playerRb.velocity.y);
    }

    void controlArma(int id)
    {
        foreach (GameObject o in armas)
        {
            o.SetActive(false);
        }

        if (id >= 0 && id < armas.Length)
        {
            armas[id].SetActive(true);
        }
    }
}
```

```
void OnCollisionEnter2D(Collision2D col)
{
    if (col.gameObject.tag == "Espinhos")
    {
        GameManager.instance.DecreaseLife(1);
    }

    if (col.gameObject.tag == "Portal")
    {
        String currentScene = SceneManager.GetActiveScene().name;
        if (currentScene == "Game")
        {
            SceneManager.LoadScene("Game2");
        } else if (currentScene == "Game2" && GameManager.instance.score <= 8)
        {
            SceneManager.LoadScene("Final1CutScene");
        } else if (currentScene == "Game2" && GameManager.instance.score > 8)
        {
            SceneManager.LoadScene("FinalAlternativoCutScene");
        }
    }
}

void OnCollisionExit2D(Collision2D col)
{
}
```

5.0.2 Atores e seus componentes

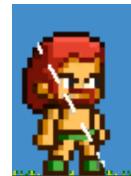


Figura 1 – Axel



Figura 2 – Raven

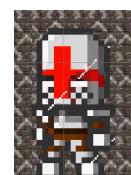


Figura 3 – Boss



Figura 4 – Goblin

5.0.3 Sprites



Figura 5 – Espinhos



Figura 6 – Coração (indicador de vida)

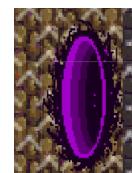


Figura 7 – Portal (transição entre fases)

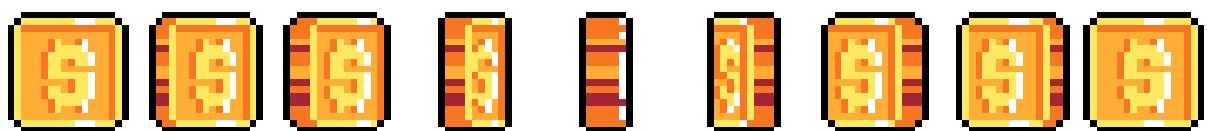


Figura 8 – Moedas

5.0.4 Backgrounds

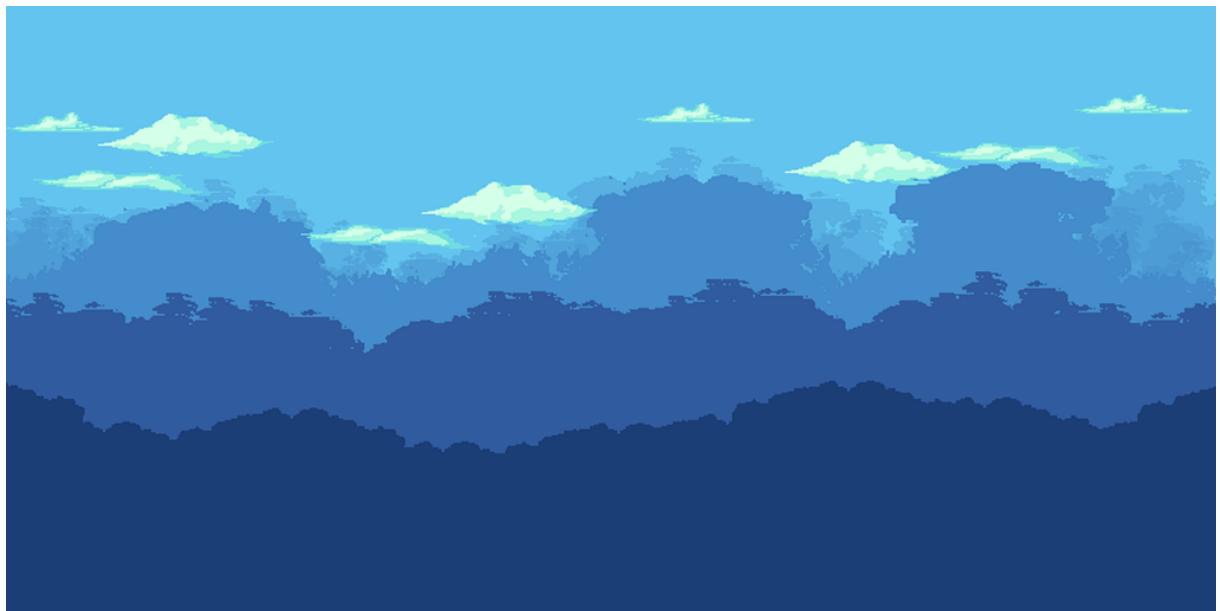


Figura 9 – Background 1



Figura 10 – Árvores para o background 1

6 PROTOTIPAÇÃO NO PAPEL

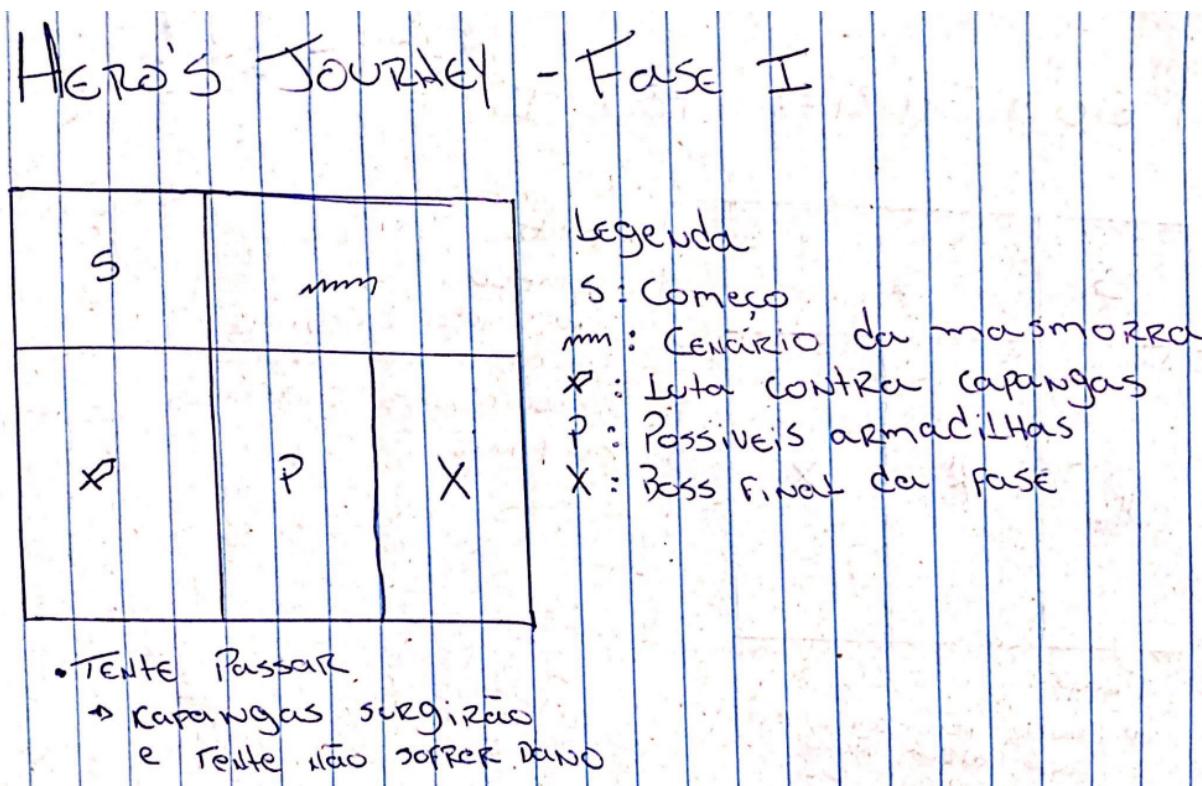


Figura 11 – Protótipo da fase 1

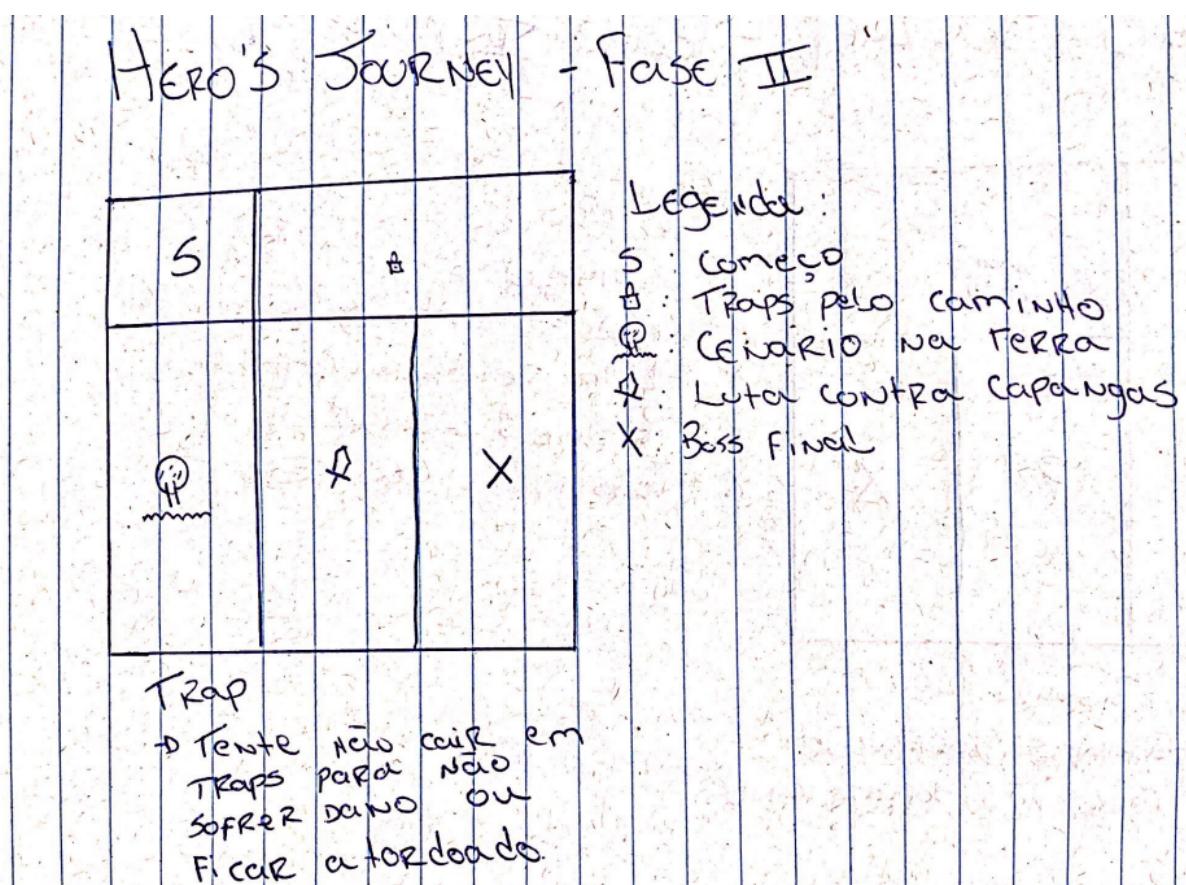


Figura 12 – Protótipo da fase 2

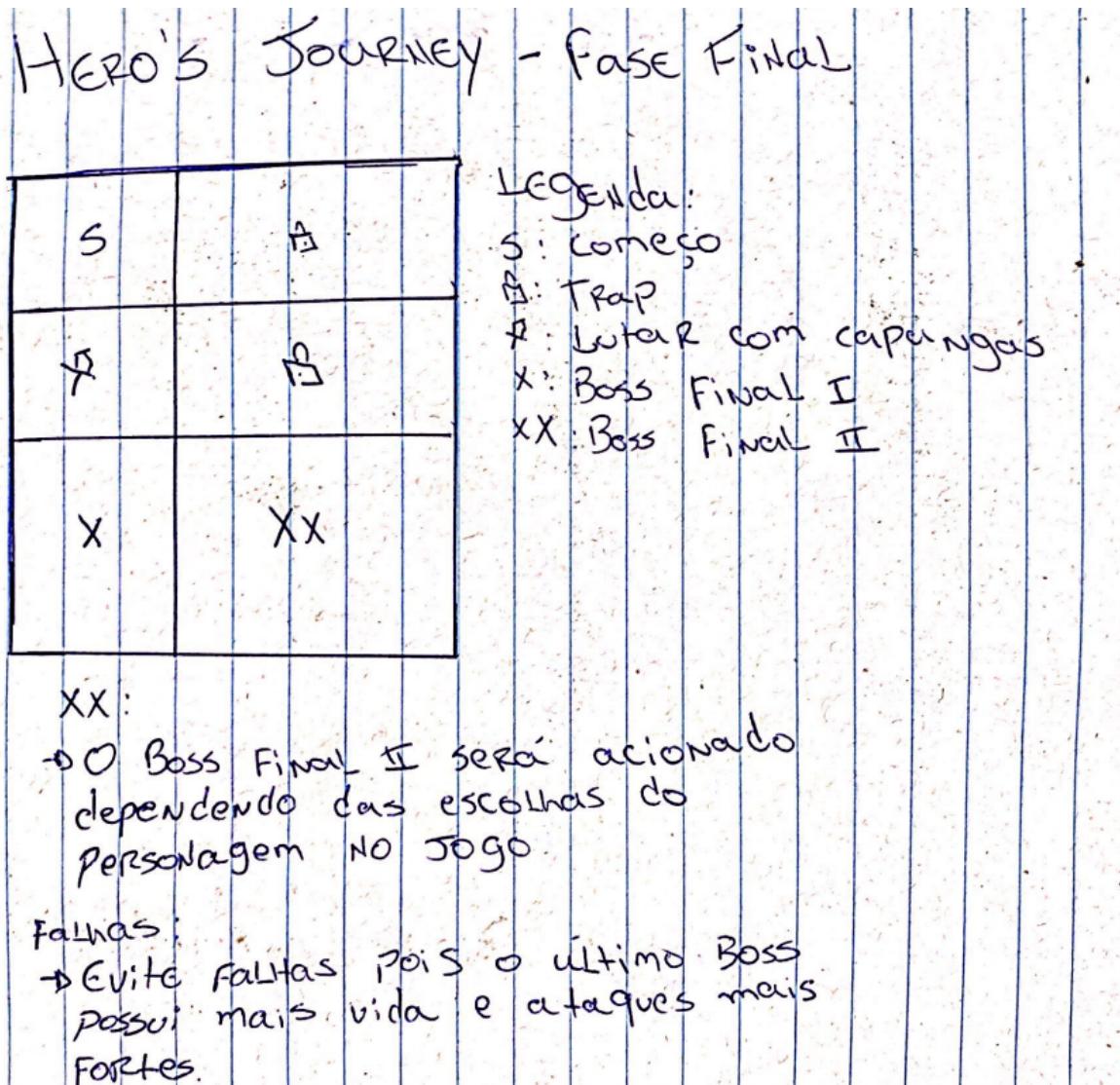


Figura 13 – Protótipo da fase final

7 DESENVOLVIMENTO

O jogo Hero's Journey foi implementado usando o Unity 2D como plataforma de desenvolvimento. O Unity 2D é uma poderosa ferramenta que permite a criação de jogos em duas dimensões, oferecendo uma ampla gama de recursos e funcionalidades.

Os scripts do jogo foram escritos em C#, uma linguagem de programação amplamente utilizada no desenvolvimento de jogos no Unity. A escolha do C# permitiu uma programação eficiente e estruturada, possibilitando a implementação das mecânicas de jogo, interações e lógica necessárias para proporcionar uma experiência divertida e imersiva aos jogadores.

No que diz respeito aos recursos visuais e sonoros, foram utilizados sprites, músicas e imagens de fundo obtidos de fontes gratuitas disponíveis no itch.io. O itch.io é uma plataforma popular que oferece uma ampla variedade de recursos gratuitos para desenvolvedores de jogos independentes. A utilização desses recursos permitiu a criação de uma atmosfera única para o jogo, adicionando elementos visuais e sonoros que complementam a história e a jogabilidade.

A combinação do Unity 2D, a linguagem C# e os recursos obtidos do itch.io permitiram a criação do jogo Hero's Journey, proporcionando aos jogadores uma experiência envolvente em um mundo de fantasia repleto de desafios e emoções.

8 TESTES

Durante o desenvolvimento da primeira fase do jogo, foram realizados testes com um grupo de jogadores para coletar feedback e avaliar a experiência do jogo. Os relatos e feedbacks coletados foram os seguintes:

8.0.1 Relato 1

Um dos testadores relatou ter se sentido empolgado e desafiado durante a jogatina. Ele destacou que as interações com os personagens proporcionaram uma maior imersão na história. No entanto, ele mencionou que a falta de variedade de movimentos acabou tornando o jogo monótono em alguns momentos, o que o deixou entediado. Esse feedback indica a necessidade de adicionar mais elementos de jogabilidade e diversificação de movimentos para manter o interesse do jogador ao longo do jogo.

8.0.2 Relato 2

Outro testador afirmou que o jogo apresenta um bom cenário e uma jogabilidade satisfatória, destacando a facilidade de uso da mecânica de machado e a movimentação responsiva do personagem principal. No entanto, ele ressaltou a falta de instruções claras para que o jogador compreenda completamente as habilidades e objetivos do jogo. Esse testador sugeriu que uma explicação mais detalhada dos objetivos e mecânicas, juntamente com indicações claras de progressão para a próxima fase ou nível, seriam úteis para melhorar a experiência do jogador.

8.0.3 Relato 3

Um terceiro testador elogiou o cenário bem detalhado do jogo, que proporcionou uma atmosfera envolvente. Ele também elogiou a movimentação suave e responsiva do personagem principal. O testador destacou que a história cativante despertou seu interesse em saber mais sobre o universo do jogo e os personagens. No entanto, ele expressou a sensação de que a curta duração do jogo deixou um gostinho de quero mais. Além disso, a falta de missões secundárias fez com que ele sentisse que poderia ter explorado mais o mundo do jogo. Esses comentários sugerem a possibilidade de adicionar conteúdo adicional, como missões secundárias, para prolongar a experiência do jogador e explorar mais o universo do jogo.

Com base nos feedbacks coletados, foram identificados pontos fortes e áreas de melhoria no jogo. Essas informações serão utilizadas para aprimorar a jogabilidade, adicionar instruções mais claras e considerar a inclusão de conteúdo adicional para proporcionar uma experiência mais completa e envolvente aos jogadores.

9 RESULTADO FINAL

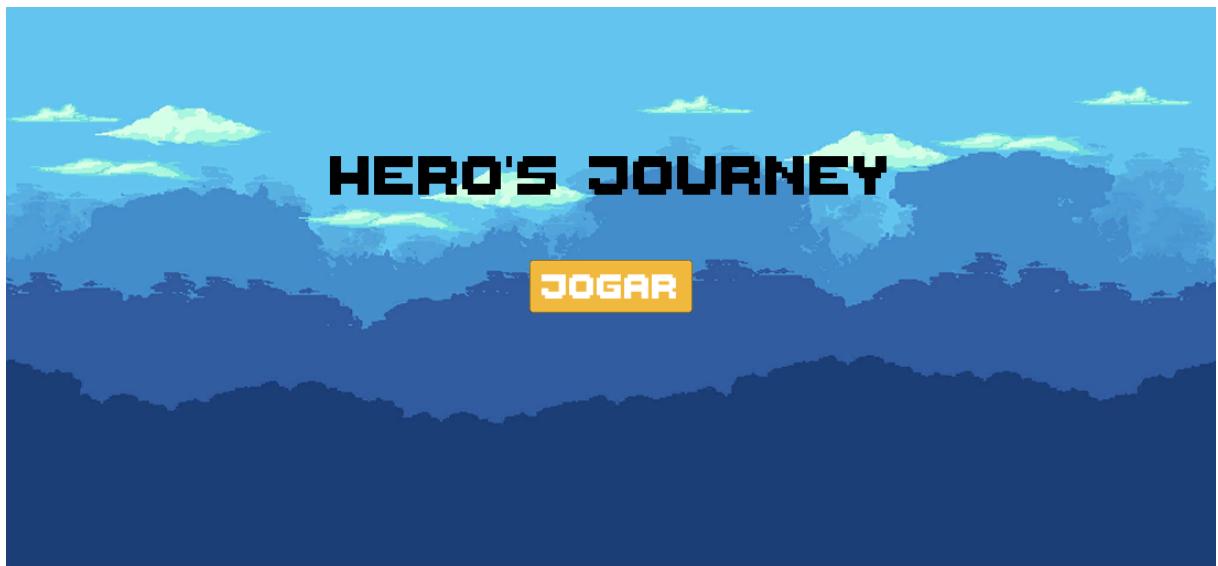


Figura 14 – Menu

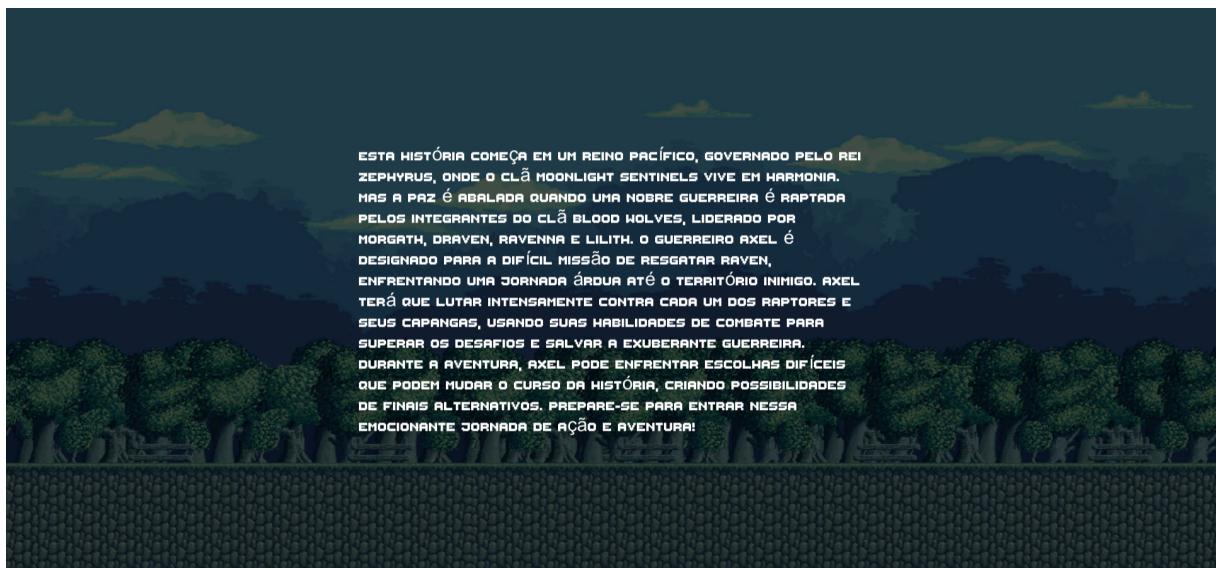


Figura 15 – CutScene Inicial

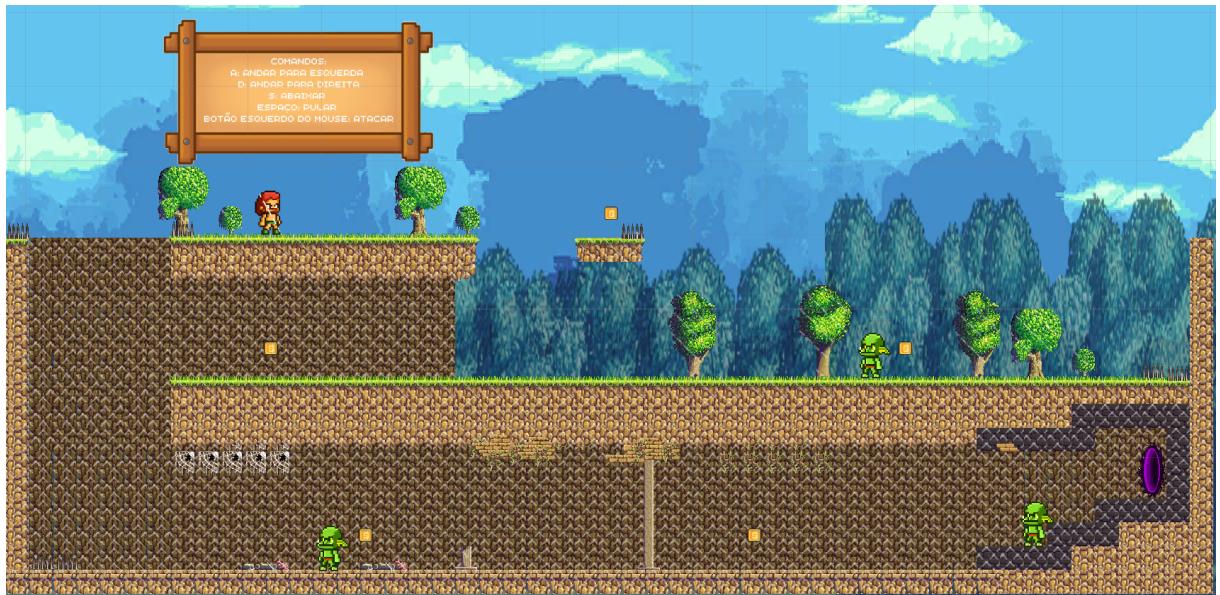


Figura 16 – Fase 1



Figura 17 – Fase 2

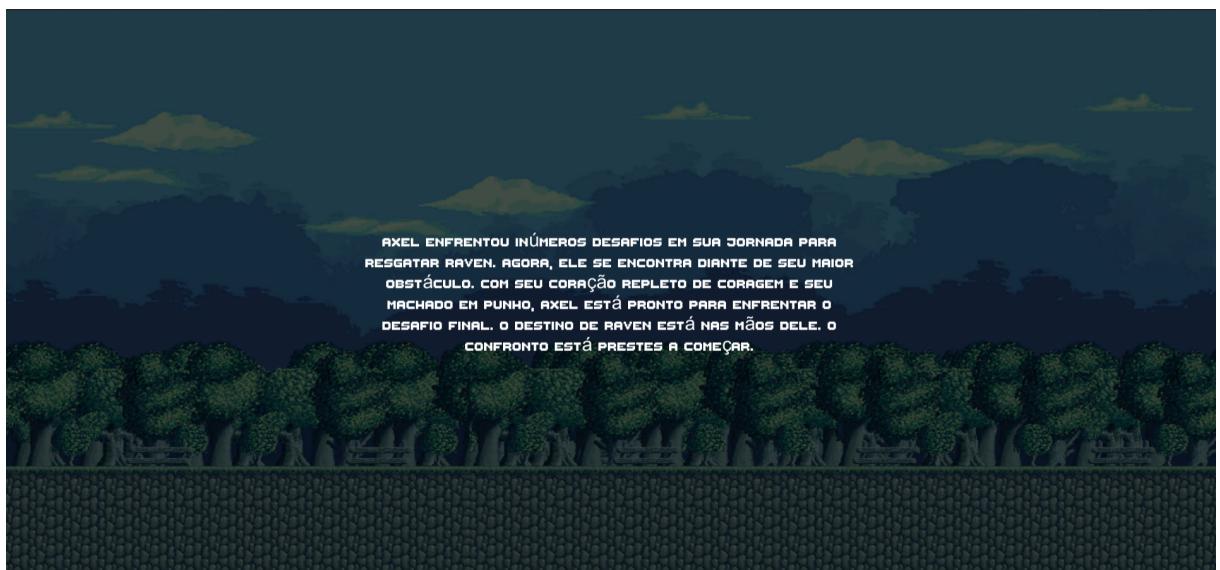


Figura 18 – CutScene Final 1

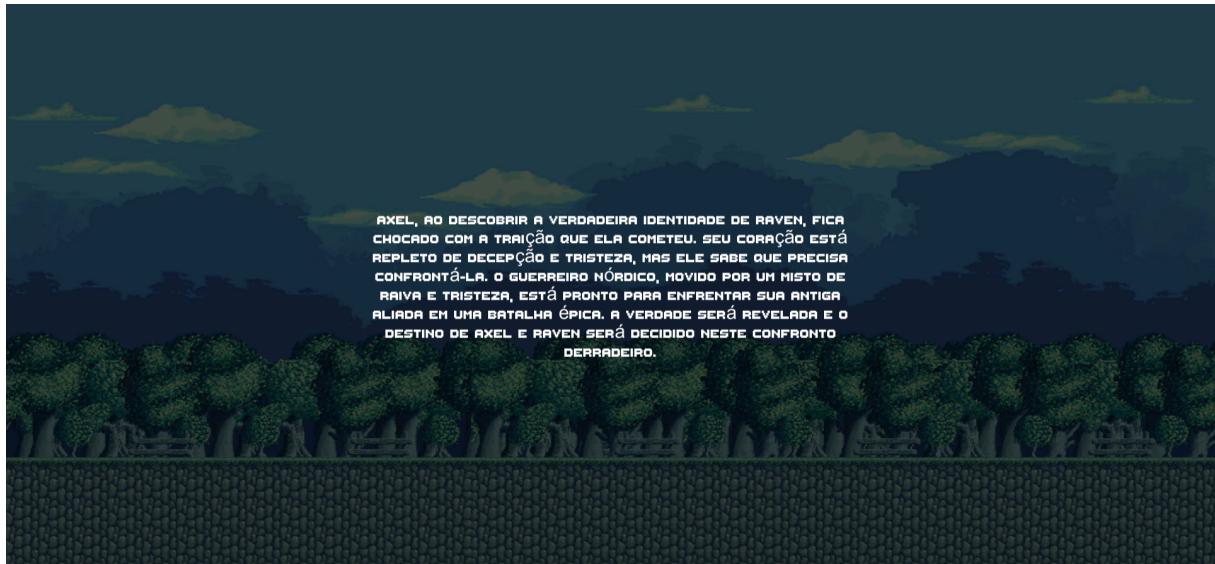


Figura 19 – CutScene Final Alternativo

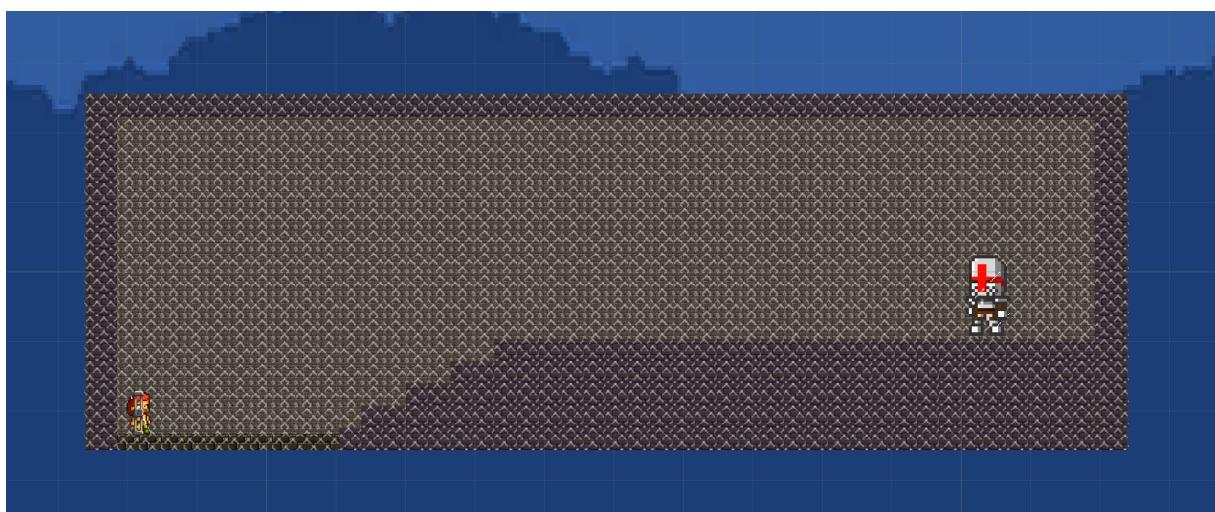


Figura 20 – Final 1

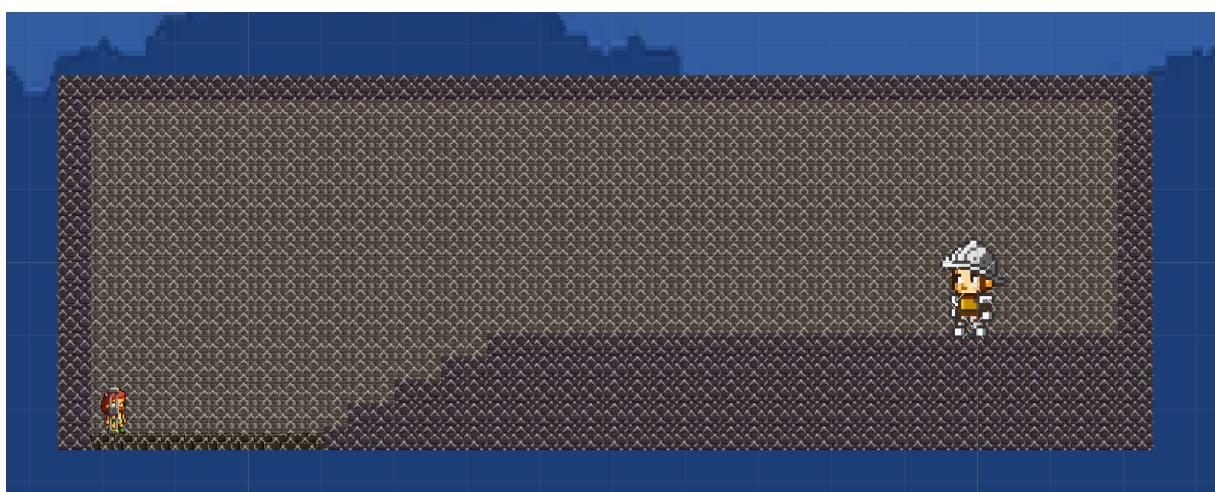


Figura 21 – Final Alternativo



Figura 22 – EndScene Final 1

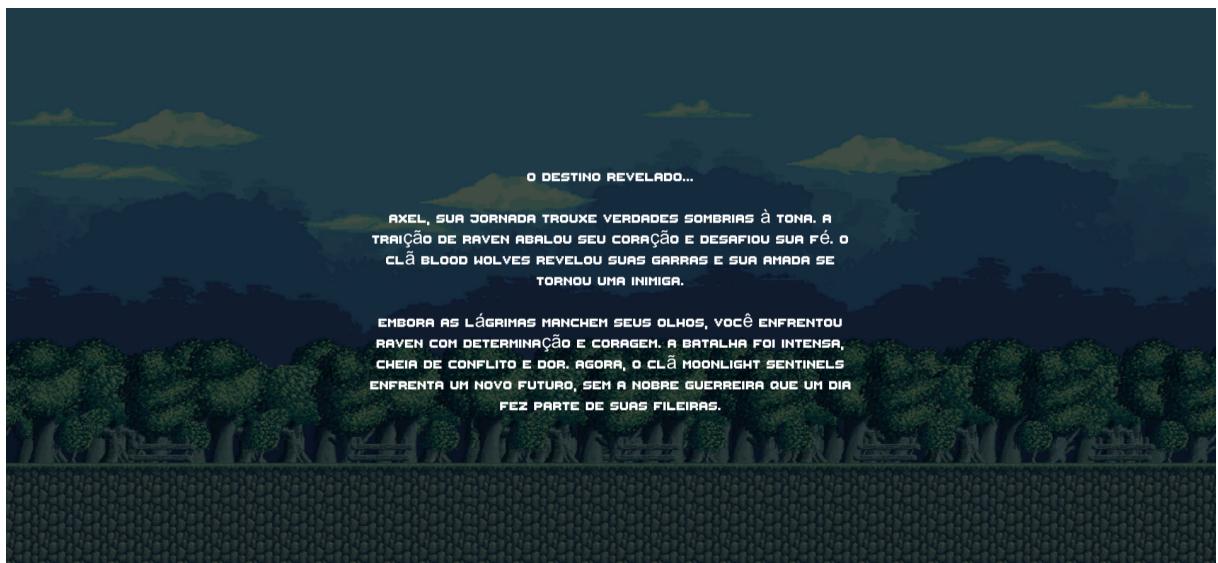


Figura 23 – EndScene Final Alternativo