Post Mortem do Projeto PIXELs

Matheus Chaves Ferreira
RA: 823210801
USJT

Introdução

O projeto PIXELs foi iniciado em 15 de junho de 2024 e concluído em 21 de junho de 2024. O objetivo inicial era criar um jogo 2D de fases em que o jogador coleta maçãs e chega nos checkpoints para mudar de fase.

Visão Geral do Projeto

PIXELs é um jogo de plataforma 2D desenvolvido na Unity, onde os jogadores devem coletar maçãs, passar de nível nos checkpoints e evitar perigos como espinhos e serras. O jogo utiliza trampolins e apoios de suporte para ajudar o player a progredir.

Principais características do jogo incluem:

- Coleta de maçãs.
- Passagem de nível em checkpoints.
- Evitar perigos como espinhos e serras.
- Uso de trampolins e apoios de suporte.
- Implementação de pulo duplo e pulo simples.
- Animações nos itens de maçã, coleta, personagens, serras, trampolins, entre outros.

O Que Deu Certo

- **Mecânicas de Jogo:** As mecânicas de coleta de maçãs e troca de fase funcionaram bem, proporcionando uma experiência divertida e desafiadora.
- **Animações:** As animações dos itens e personagens foram bem recebidas, adicionando uma camada de imersão e dinamismo ao jogo.
- **Desenvolvimento na Unity:** O uso da Unity facilitou a implementação de diversas funcionalidades, como física e animações.

O Que Deu Errado

- **Criação do Chefão:** A criação de um chefão (MASK) encontrou problemas devido à lentidão do computador utilizado, o que impediu a finalização e implementação adequada do chefão. Ele foi substituído por uma serra (saw) como desafio final. A lógica do chefão está implementada no código, mas não foi possível testá-la completamente.

- **Problemas de Desempenho:** A lentidão do computador impactou negativamente a produtividade e a capacidade de testar novas funcionalidades de maneira eficiente.

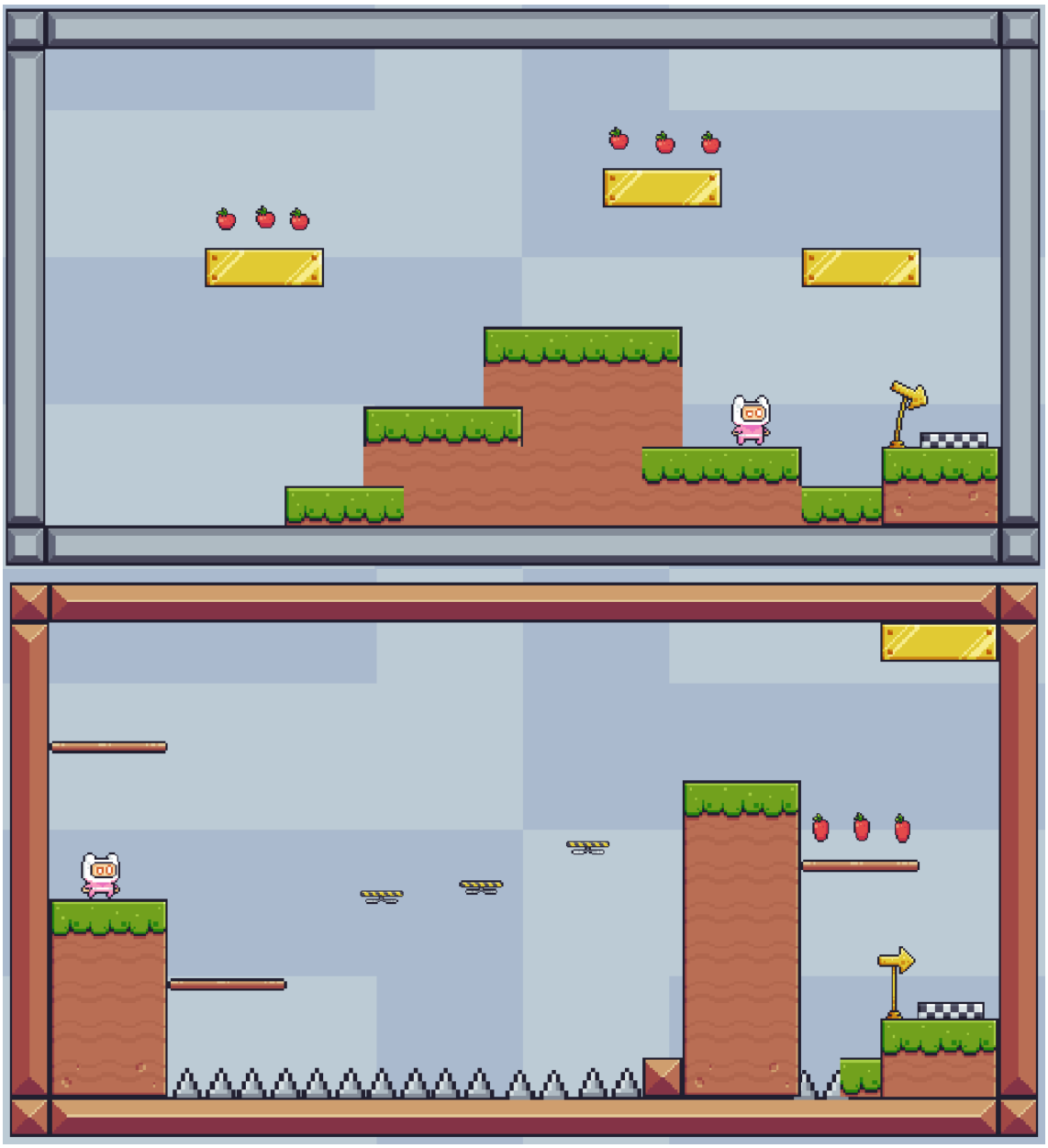
Lições Aprendidas

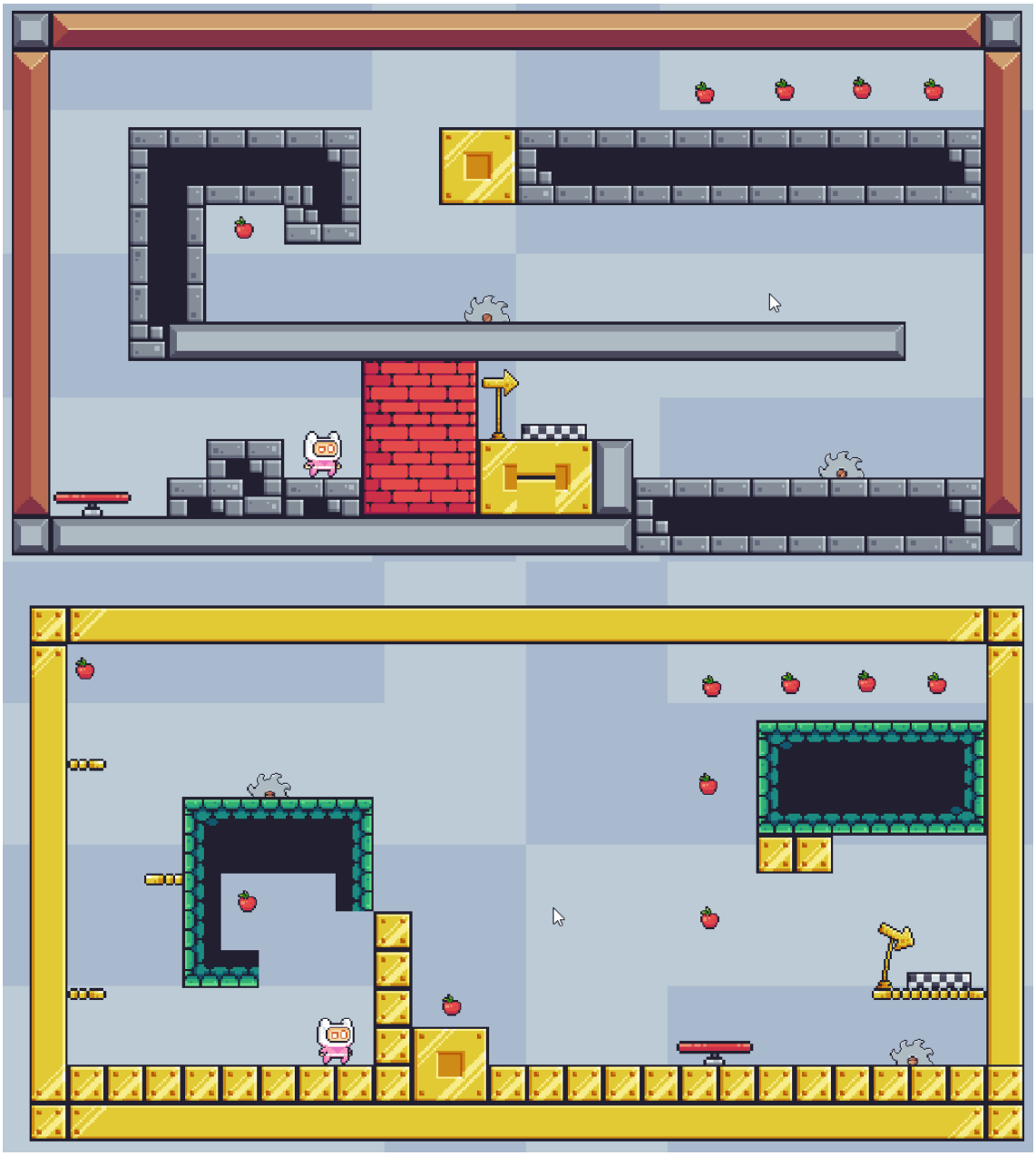
- **Metodologias da Unity:** Durante o desenvolvimento do PIXELs, aprendi muito sobre as metodologias da Unity, desde a criação de mecânicas de jogo até a implementação de animações e física. A experiência com projetos simples na Unity mostrou a importância de planejamento e testes contínuos para garantir a qualidade do jogo.
- **Planejamento de Recursos:** A importância de ter um hardware adequado para desenvolvimento ficou evidente, especialmente em projetos que exigem muitos recursos. Planejar melhor o uso dos recursos e garantir que a infraestrutura de desenvolvimento esteja à altura das necessidades do projeto é crucial para evitar atrasos e problemas.

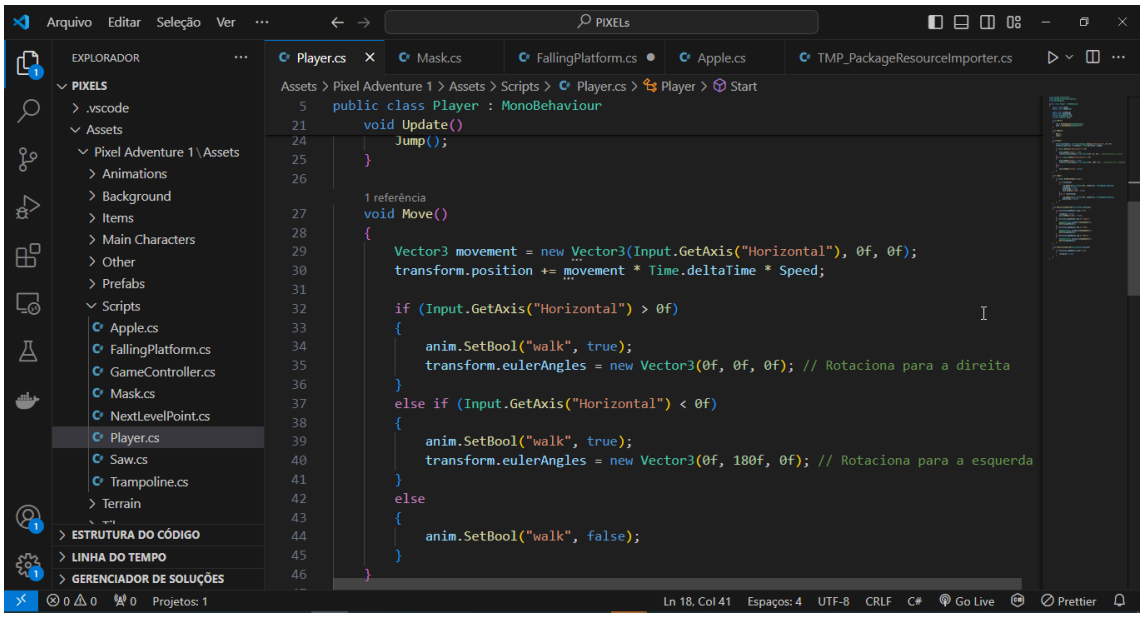
Agradecimentos

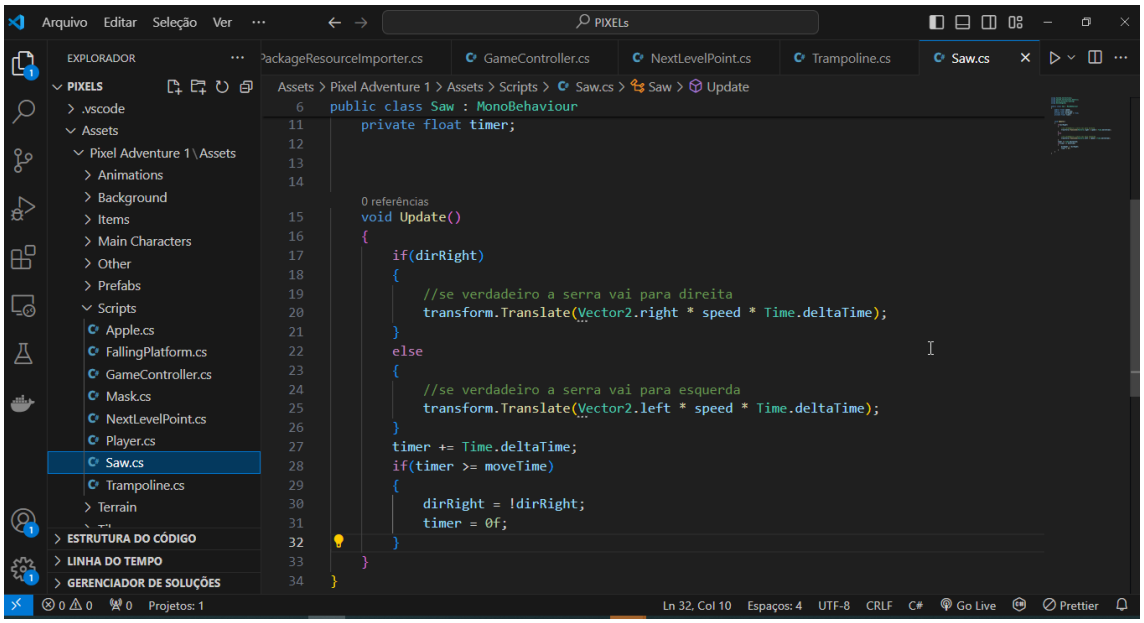
Gostaria de agradecer ao professor pelo conhecimento compartilhado no período letivo e elogiar pela solicitação de trabalho, pois nos incentiva a aprender mais no dia a dia.

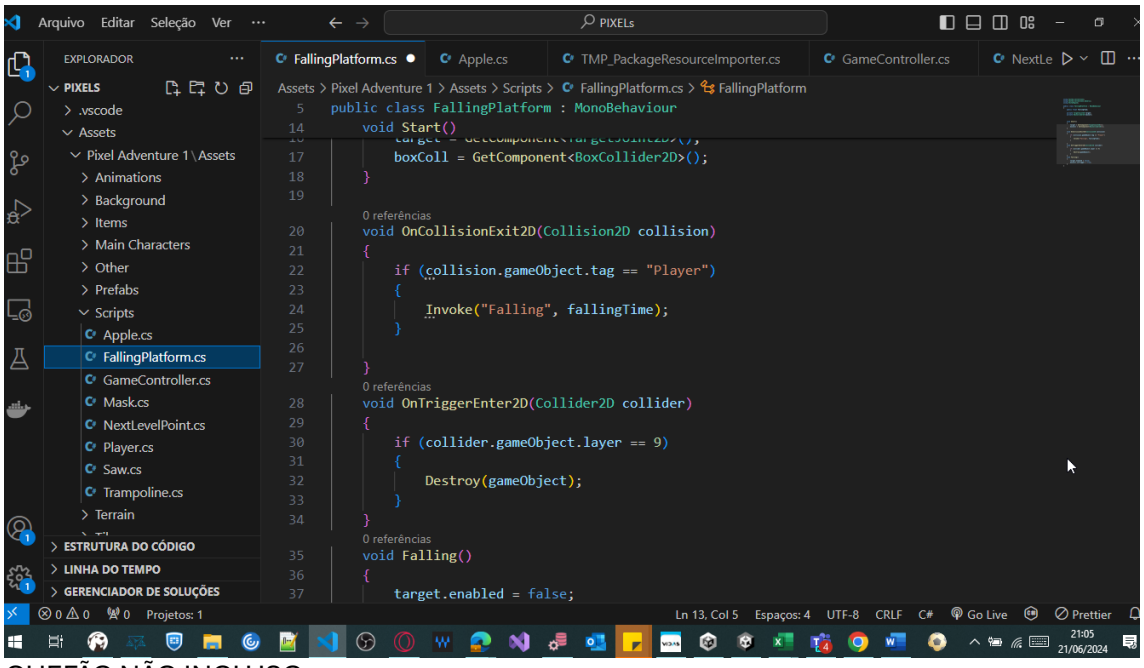
Abaixo vou deixar imagens do projeto, as fases do game, link do jogo no GIT e partes dos códigos utilizadas.






Arquivo Editar Seleção Ver
PIXELs
EXPLORADOR
Player.cs
Mask.cs
FallingPlatform.cs
Apple.cs
TMP_PackageResourceImporter.cs
Assets > Pixel Adventure 1 > Assets > Scripts > Player.cs > Player > Start
PIXELS
.vscode
Assets
Pixel Adventure 1 \ Assets
Animations
Background
Items
Main Characters
Other
Prefabs
Scripts
Apple.cs
FallingPlatform.cs
GameController.cs
Mask.cs
NextLevelPoint.cs
Player.cs
Saw.cs
Trampoline.cs
Terrain
ESTRUTURA DO CÓDIGO
LINHA DO TEMPO
GERENCIADOR DE SOLUÇÕES
public class Player : MonoBehaviour
void Update()
Jump();
1 referência
void Move()
Vector3 movement = new Vector3(Input.GetAxis("Horizontal"), 0f, 0f);
transform.position += movement * Time.deltaTime * Speed;
if (Input.GetAxis("Horizontal") > 0f)
anim.SetBool("walk", true);
transform.eulerAngles = new Vector3(0f, 0f, 0f); // Rotaciona para a direita
else if (Input.GetAxis("Horizontal") < 0f)
anim.SetBool("walk", true);
transform.eulerAngles = new Vector3(0f, 180f, 0f); // Rotaciona para a esquerda
else
anim.SetBool("walk", false);
Ln 18, Col 41 Espaços: 4 UTF-8 CRLF C# Go Live Prettier
Projetos: 1


Arquivo Editar Seleção Ver
PIXELs
EXPLORADOR
GameController.cs
NextLevelPoint.cs
Trampoline.cs
Saw.cs
Assets > Pixel Adventure 1 > Assets > Scripts > Saw.cs > Saw > Update
PIXELS
.vscode
Assets
Pixel Adventure 1 \ Assets
Animations
Background
Items
Main Characters
Other
Prefabs
Scripts
Apple.cs
FallingPlatform.cs
GameController.cs
Mask.cs
NextLevelPoint.cs
Player.cs
Saw.cs
Trampoline.cs
Terrain
ESTRUTURA DO CÓDIGO
LINHA DO TEMPO
GERENCIADOR DE SOLUÇÕES
public class Saw : MonoBehaviour
private float timer;
0 referências
void Update()
if(dirRight)
//se verdadeiro a serra vai para direita
transform.Translate(Vector2.right * speed * Time.deltaTime);
else
//se verdadeiro a serra vai para esquerda
transform.Translate(Vector2.left * speed * Time.deltaTime);
timer += Time.deltaTime;
if(timer >= moveTime)
dirRight = !dirRight;
timer = 0f;
Ln 32, Col 10 Espaços: 4 UTF-8 CRLF C# Go Live Prettier
Projetos: 1



CHEFÃO NÃO INCLUSO no game