**Fundação valeparaibana de ensino**colégios univap – unidade centro

Curso técnico em informática

LEONARDO MARTINELLI DE OLIVEIRA LIMA

**LISTA DE EXERCÍCIOS 3º bimestre**

DESENVOLVIMENTO DE JOGOS

Lista apresentada ao Curso Técnico de informática como composição de nota da disciplina Desenvolvimento de Jogos.

Prof. Me. Hélio Lourenço Esperidião Ferreira

SÃO JOSÉ DOS CAMPOS

2023

LISTA DE DESENVOLVIMENTO DE JOGOS 1º BIMESTRE

1. O que são Sprites?

É um método rápido de animação utilizando várias imagens agrupadas numa tela.

Qual a diferença entre animator e Animation?

O Animation permite configurar os assets que são parte de uma animação. Ao criar uma animação também é criado automaticamente um animador. Ele pode ser encontrado em meio a seus assets por meio da imagem ao lado

1. O que é uma transição em animação?

É a transição de uma animação para outra.

1. Como fazer uma transição no animator?

Abre o animator já com suas animações, e terá varias caixas, depois tem que ligar as animações e configurar as condições de ativação

1. O que são parâmetros no animator e para que servem?

São variáveis criadas para ativar/desativar animações com condições.

1. Como criar um parâmetro?

Do lado esquerdo da janela do animator existe a ferramenta de criação e manipulação de parâmetros (Parameters), ao clicar no botão de “+” irá ser criado um parâmetro.

1. O que são condições de transição?

São condições necessárias para ativar ou desativar uma animação.

1. O que a linha de código abaixo faz?

personagemAnimator = gameObject.GetComponent ();

“personagemAnimator” recebe o objeto do jogo que o código foi colocado.

1. O que as linhas de código abaixo fazem?  
   personagemAnimator.SetBool ("andando", true);

A animação “andando” será executada ao receber o valor “true”.

personagemAnimator.SetBool ("andando", false);

A animação “andando” será desativada/parada ao receber o valor “false”.

personagemAnimator.SetBool ("pulando", true);

A animação “pulando” será executada ao receber o valor “true”.

1. O que é um temporizador?

É o tempo de ativação de uma animação.

1. Explique como funciona o Time.deltaTime.

É o intervalo de tempo entre uma execução de um frame para outro.

1. Explique o funcionamento do código abaixo:

| void VerificarBonusPular(){          if (BonusVelocidadePular == true) {                           if (TempoBonusVelocidadePular >= 0) {                 TempoBonusVelocidadePular = TempoBonusVelocidadePular - Time.deltaTime;                 print ("Tempo Bonus Pular: " + TempoBonusVelocidadePular);             } else {                 BonusVelocidadePular = false;                 TempoBonusVelocidadePular = 10;                 VelocidadePular = 5;             }         }     } |
| --- |

A função “VerificarBonusPular() verifica se “BonusVelocidadePular” é verdadeiro ou falso, se for verdadeiro verifica se o TempoBonusVelocidadePular >= 0, se for verdadeiro TempoBonusVelocidadePular será subtraído pelo Time.deltaTime e imprime a mensagem “Tempo Bonus Pular : “ mais o valor da variavel TempoBonusVelocidadePular, mas se TempoBonusVelocidadePular for menor do que 0, BonusVelocidadePular recebe falso, TempoBonusVelocidadePular recebe 10 e VelocidadePular recebe 5.

1. Qual a funcionalidade do **Invoke**?

É outro método dentro do script em um determinado tempo através do seu nome que deve ser passado como uma string

1. Explique o que a linha de código abaixo faz?

Invoke ("metodo" , 2);

Será executado que um tempo passar

1. Qual recurso do Unity é responsável por todas as entradas de dados

Input

1. Detalhe o funcionamento do GetMouseButtonDown(). Quais números representam quais botões do mouse?
   1. 1 Botão esquerdo
   2. 2 Botão direito
   3. 3 Scroll
2. Explique o funcionamento do GetKeyDown().

É utilizado pra identificar de uma tecla esta sendo pressionada ou não.

1. O que é um KeyCode?

KeyCode é o código da função de cada tecla

1. Qual a diferença entre o KeyCode() e o GetKeyDown()?

GeKeyDow é utilizado pra identificar de uma tecla esta sendo pressionada ou não

KeyCode éo código da função de cada tecla

1. O que é o transform do Unity?

É o nome da classe (Script) que o Unity criou que contém toda a informação de acesso como métodos, variáveis disponíveis

1. O que a linha de código abaixo faz?

Vector2 posicaoDestino = GameObject.FindGameObjectWithTag ("portal02").GetComponent ().position;

Que se ele entrar no portal o seu destino vai ser que o programador desejar, tipo entrou no portal ele tele transporta para ou outro lado da ponte.

1. Explique detalhadamente o código abaixo:

| void OnTriggerEnter2D(Collider2D objetoTriggerTocado){         string tagTocadaTrigger = objetoTriggerTocado.gameObject.tag;         if (tagTocadaTrigger == "portal01") {             Vector2 posicaoDestino = GameObject.FindGameObjectWithTag ("portal02").GetComponent<Transform> ().position;             float xDestino = posicaoDestino.x;             float yDestino = posicaoDestino.y;             xDestino = xDestino + 1.5f;             transform.position = new Vector2(xDestino,yDestino);         }          if (tagTocadaTrigger == "portal02") {             Vector2 posicaoDestino = GameObject.FindGameObjectWithTag ("portal01").GetComponent<Transform> ().position;             float xDestino = posicaoDestino.x;             float yDestino = posicaoDestino.y;             xDestino = xDestino - 1.5f;             transform.position = new Vector2(xDestino,yDestino);         }     } |
| --- |

Se entrar na área da tag “portal01” será tele portado para frente, se entrar na área da tag “portal02” será tele portado para trás.

1. Qual a funcionalidade das linhas de código abaixo e o que pode ser feito com esse código?

Rigidbody2D objeto;

objeto = GameObject.FindGameObjectWithTag ("tag").GetComponent ();

objeto é um corpo rigido e receberá uma tag.

1. Explique detalhadamente o código abaixo:

| void OnTriggerEnter2D(Collider2D objetoTriggerTocado){         string tagTocadaTrigger = objetoTriggerTocado.gameObject.tag;         if (tagTocadaTrigger == "tagA") {             GameObject[] vetorObjetos;             vetorObjetos= GameObject.FindGameObjectsWithTag ("TagX");             int qtdObjetos = vetorObjetos.Length;             for (int i = 0; i < qtdObjetos; i++) {                 Rigidbody2D corpoRigidoBola;                 corpoRigidoBola = vetorObjetos[i].GetComponent<Rigidbody2D> ();                 corpoRigidoBola.gravityScale = 0.2f;             }         }     } |
| --- |

1. Qual a diferença entre GameObject.FindGameObjectsWithTag  eGameObject.FindGameObjectWithTag?

Um pega um objeto em especifico e o outro pega todos com a tag

1. Explique detalhadamente o funcionamento do código abaixo:

| void metodo(){         Vx = CorpoRigidoInimigo.velocity.x;         Vy = 0;         if (Vx == 0) {             Vx = VelocidadeAndar \* DirecaoHorizontal;             Vector2 movimento = new Vector2 (Vx, Vy);             CorpoRigidoInimigo.velocity = movimento;             DirecaoHorizontal = DirecaoHorizontal \* -1;         }     } |
| --- |

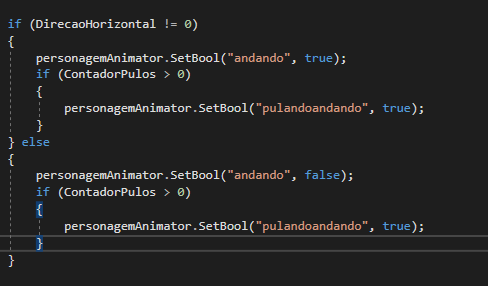
A função método faz o movimento do personagem ser automático e ter um padrão.

1. Explique a diferença entre objeto tocado e filho do objeto tocado no código abaixo:

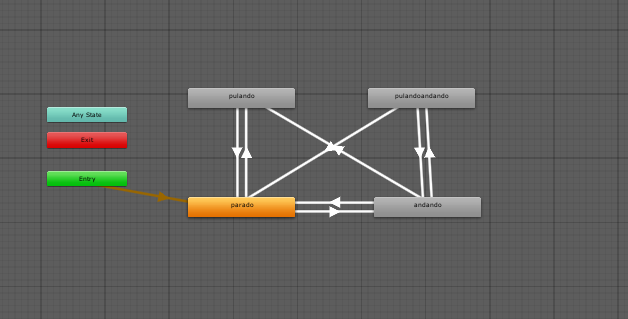
| void OnCollisionEnter2D(Collision2D objetoTocado){         GameObject filhoObjetoTocado = objetoTocado.collider.gameObject;         string tagFilhoObjetoTocado = filhoObjetoTocado.tag;         string tagObjetoTocado = objetoTocado.gameObject.tag;      } |
| --- |

O objeto tocado é todo o objeto, enquanto o filho é o objeto ligado ao objeto tocado.

1. Faça um jogo com no mínimo 4 animações. Apresente o código, o print do animator e um print do jogo funcionando.







1. Faça um jogo que ao coletar uma maça permita bônus de velocidade por 4 segundos. Ao coletar uma banana o personagem anda automaticamente para frente por 2 segundos. Ao coletar um tomate permita pulo triplo por 3 segundos. Ao coletar uma cenoura permita bônus de altura do pulo por 5 segundos.

using System.Collections;

using UnityEngine;

public class PlayerController : MonoBehaviour

{

private bool velocidadeBonusAtivo = false;

private bool andarAutomaticoAtivo = false;

private bool puloTriploAtivo = false;

private bool bonusAlturaPuloAtivo = false;

private void Update()

{

if (velocidadeBonusAtivo)

{

// Aumentar a velocidade do personagem aqui

}

if (andarAutomaticoAtivo)

{

// Fazer o personagem andar automaticamente para frente aqui

}

if (puloTriploAtivo)

{

// Permitir pulo triplo do personagem aqui

}

if (bonusAlturaPuloAtivo)

{

// Aumentar a altura do pulo do personagem aqui

}

}

private IEnumerator DesativarBonus(float duracao)

{

yield return new WaitForSeconds(duracao);

// Desativar os bônus correspondentes aqui

velocidadeBonusAtivo = false;

andarAutomaticoAtivo = false;

puloTriploAtivo = false;

bonusAlturaPuloAtivo = false;

}

}

public class AppleCollector : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

PlayerController playerController = other.GetComponent<PlayerController>();

playerController.velocidadeBonusAtivo = true;

playerController.StartCoroutine(playerController.DesativarBonus(4f));

Destroy(gameObject);

}

}

}

public class BananaCollector : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

PlayerController playerController = other.GetComponent<PlayerController>();

playerController.andarAutomaticoAtivo = true;

playerController.StartCoroutine(playerController.DesativarBonus(2f));

Destroy(gameObject);

}

}

}

public class TomatoCollector : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

PlayerController playerController = other.GetComponent<PlayerController>();

playerController.puloTriploAtivo = true;

playerController.StartCoroutine(playerController.DesativarBonus(3f));

Destroy(gameObject);

}

}

}

public class CarrotCollector : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

PlayerController playerController = other.GetComponent<PlayerController>();

playerController.bonusAlturaPuloAtivo = true;

playerController.StartCoroutine(playerController.DesativarBonus(5f));

Destroy(gameObject);

}

}

}

1. Faça um jogo que permita que o personagem se movimente para direita com o click do botão direito do mouse, se movimente para direita com o click do botão direito do mouse e para de se movimentar o click do botão do meio do mouse.
2. Crie pelo menos 4 portais.

using UnityEngine;

public class PlayerController : MonoBehaviour

{

private Rigidbody2D rb;

private bool movimentoAtivo = false;

public float forcaMovimento = 5f;

private void Awake()

{

rb = GetComponent<Rigidbody2D>();

}

private void Update()

{

if (Input.GetMouseButtonDown(0))

{

// Botão esquerdo do mouse não utilizado neste exemplo.

}

else if (Input.GetMouseButtonDown(1))

{

// Botão direito do mouse pressionado, ativa o movimento para a direita.

movimentoAtivo = true;

}

else if (Input.GetMouseButtonDown(2))

{

// Botão do meio do mouse pressionado, desativa o movimento.

movimentoAtivo = false;

}

}

private void FixedUpdate()

{

if (movimentoAtivo)

{

// Aplica uma força para mover o personagem para a direita.

rb.AddForce(Vector2.right \* forcaMovimento);

}

}

}

1. Permita que ao coletar 4 chaves uma porta seja destruída.

using UnityEngine;

using UnityEngine.SceneManagement;

public class Portal : MonoBehaviour

{

public string cenaDestino;

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Player"))

{

SceneManager.LoadScene(cenaDestino);

}

}

}

1. Faça um inimigo que pule de 5 em 5 segundos.

using UnityEngine;

public class PlayerController : MonoBehaviour

{

public int numChavesColetadas = 0;

public int numChavesNecessarias = 4;

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Key"))

{

// Aumentar o contador de chaves no jogador

numChavesColetadas++;

// Destruir a chave

Destroy(other.gameObject);

}

else if (other.CompareTag("Door"))

{

// Verificar se o jogador coletou todas as chaves necessárias

if (numChavesColetadas >= numChavesNecessarias)

{

// Destruir a porta

Destroy(other.gameObject);

}

}

}

}

public class Key : MonoBehaviour

{

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Player"))

{

// Aumentar o contador de chaves no jogador

PlayerController playerController = other.GetComponent<PlayerController>();

playerController.numChavesColetadas++;

// Destruir a chave

Destroy(gameObject);

}

}

}

1. Faça um inimigo que vá para direita, vá para esquerda e pule. Repita os movimentos infinitamente.

using UnityEngine;

public class EnemyController : MonoBehaviour

{

private Rigidbody rb;

public float intervaloPulo = 5f;

public float forcaSalto = 5f;

private void Start()

{

rb = GetComponent<Rigidbody>();

InvokeRepeating("Pular", intervaloPulo, intervaloPulo);

}

private void Pular()

{

rb.AddForce(Vector3.up \* forcaSalto, ForceMode.Impulse);

}

}