MECÂNICAS ELABORADAS DE MOVIMENTAÇÃO

Prof. Me. Hélio Esperidião

Atributos

```
public class tempBimestre2 : MonoBehaviour {

    int TotalPulos;
    string TagTriggerStay;
    string TagTriggerEnter;
    string TagTriggerExit;

    string TagOjetoTocando;
    string TagObjetoParouTocar;

    int ContadorPulos;
    SpriteRenderer Renderer;
    string TagObjetoTocado;
    bool PegouChave01;
    bool ApertouBotaoPular; // para saber se o botão de pular foi apertado
    float VelocidadePuloSimples; // variável que guarda a velocidade de pulo do personagem
    Collider2D Colisor2dPersonagem;  //variável que armazena o colisor do personagem
    bool EstaTocandoAlgumColisor; //Variável que verificar se o personagem está tocando um colisor
    // Variáveis para controlar a velocidade do personagem
    float VelocidadeX; // Velocidade no eixo X (horizontal)
    float VelocidadeY; // Velocidade no eixo Y (vertical)
    float VelocidadeHorizontalMaxima; //Velocidade máxima que o personagem pode atingir no eixo X
    float DirecaoHorizontal; //Direção do movimento horizontal (-1 para esquerda, 1 para direita, 0 para parado)
    float VelocidadeVerticalMaxima; //Variável que armazena a velocidade Maxima Vertical do personagem
    float DirecaoVertical; // Variável que armazena a direção vertical do personamagem
    Vector2 VetorVelocidadePersonagem; // Vetor que armazena a velocidade do personagem em X e Y
    Rigidbody2D CorpoRigidoPersonagem;// Referência ao componente Rigidbody2D do personagem
```

Update ()

```
// Método chamado uma vez por frame (quadro)
    void Update () {
        MovimentoHorizontalComReducaoAoPular ();
        MovimentoPuloMultiploComReducaoVelocidade ();
        //MovimentoHorizontalFlip();
        //MovimentoPuloMultiplo ();
        //MovimentoPuloDuplo ();
        //MovimentoHorizontal ();
        //MovimentoPuloUnico();
        //MovimentoVertical ();
        //MovimentoPuloSimples();
    }
```

void Start ()

```
void Start () {
        TotalPulos = 2;
        TagTriggerStay = "";
        TagTriggerEnter = "";
        TagTriggerExit = "";
        TagOjetoTocando="";
        TagObjetoParouTocar="";
        ContadorPulos = 0;
        Renderer = GetComponent<SpriteRenderer> ();
        PegouChave01 = false;
        TagObjetoTocado = "";
        ApertouBotaoPular = false; // inicia a variavel com false;
        EstaTocandoAlgumColisor = false; // inicializa a variavel como falso;
        VelocidadePuloSimples = 10.0f; // determina o valor da velocidade do Pulo;
        // Obtém o componente Rigidbody2D do objeto ao qual este script está anexado
        CorpoRigidoPersonagem = GetComponent<Rigidbody2D> ();
        Colisor2dPersonagem = GetComponent<Collider2D> ();
        // Define a escala da gravidade para o Rigidbody2D (1 é o valor padrão)
        CorpoRigidoPersonagem.gravityScale = 3.0f;
        // Congela a rotação do Rigidbody2D para evitar que o personagem gire ao colidir com algo
        CorpoRigidoPersonagem.freezeRotation = true;
        // Inicializa as variáveis de velocidade
        VelocidadeX = 0f; // Começa parado no eixo X
        VelocidadeY = 0f; // Começa parado no eixo Y
        VelocidadeHorizontalMaxima = 5.0f; // Define a velocidade máxima no eixo X
        DirecaoHorizontal = 0f; // Começa sem direção definida
        VelocidadeVerticalMaxima = 5.0f; // Define a velocidade maxima no eixo y
        DirecaoVertical = 0f;              // Define a direção do movimento na vertical
        // Cria um vetor de velocidade inicial para o personagem
        VetorVelocidadePersonagem = new Vector2 (VelocidadeX, VelocidadeY);
        // Aplica o vetor de velocidade ao Rigidbody2D para mover o personagem
        CorpoRigidoPersonagem.velocity = VetorVelocidadePersonagem;
    }
```

```
    void MovimentoPuloMultiploComReducaoVelocidade() {
        // Verifica se o botão de pulo (configurado como "Jump") foi pressionado neste frame
        ApertouBotaoPular = Input.GetButtonDown("Jump");
        // Condições para executar o pulo:
        // 1. Botão de pulo pressionado
        // 2. Número de pulos atual menor que o total permitido
        if (ApertouBotaoPular == true && ContadorPulos < TotalPulos)        {
            // Incrementa o contador de pulos realizados
            ContadorPulos = ContadorPulos + 1;
            // Preserva a velocidade horizontal atual do personagem
            VelocidadeX = CorpoRigidoPersonagem.velocity.x;
            // Define a força vertical do pulo com redução progressiva:
            // - 1º pulo: 100% da velocidade normal
            // - 2º pulo: 80% da velocidade normal
            // - 3º pulo: 70% da velocidade normal
            if (ContadorPulos == 1) {
                VelocidadeY = VelocidadePuloSimples;
            }
            if (ContadorPulos == 2) {
                VelocidadeY = VelocidadePuloSimples * 0.8f;
            }
            if (ContadorPulos == 3) {
                VelocidadeY = VelocidadePuloSimples * 0.7f;
            }
            // Cria vetor de velocidade combinando:
            // - X: velocidade horizontal preservada
            // - Y: nova velocidade vertical calculada
            VetorVelocidadePersonagem = new Vector2(VelocidadeX, VelocidadeY);
            // Aplica o vetor de velocidade ao Rigidbody2D para executar o movimento
            CorpoRigidoPersonagem.velocity = VetorVelocidadePersonagem;
        }
    }
```

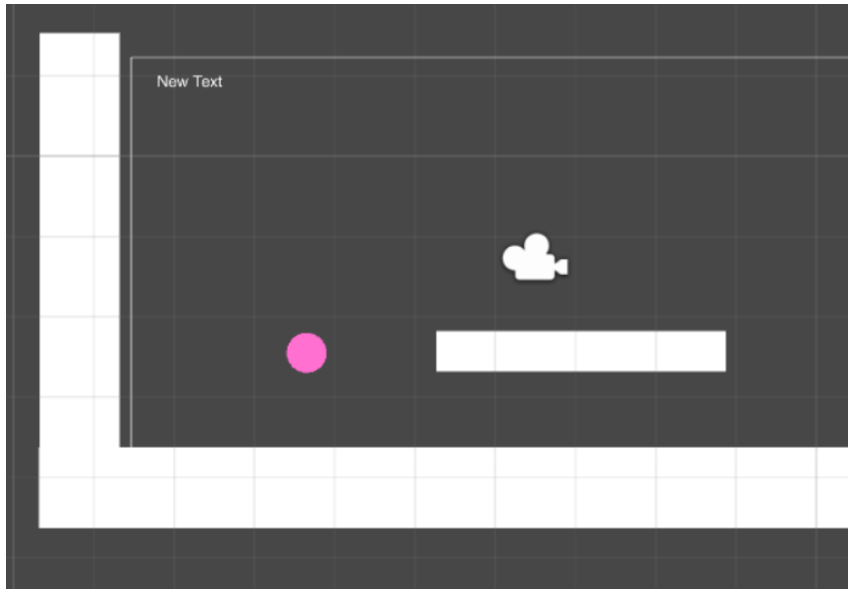
MovimentoHorizontalComReducaoAoPular()

```
void MovimentoHorizontalFlipComReducaoAoPular()
{
    // Obtém a direção horizontal do input (-1 = esquerda, 1 = direta, 0 = neutro)
    DirecaoHorizontal = Input.GetAxis("Horizontal");
    // Calcula a velocidade base multiplicando a direção pela velocidade máxima
    VelocidadeX = VelocidadeHorizontalMaxima * DirecaoHorizontal;
    // Aplica redução progressiva da velocidade horizontal conforme quantidade de pulos:
    // - 1 pulo: 80% da velocidade normal
    // - 2 pulos: 60% da velocidade normal
    // - 3 pulos: 40% da velocidade normal
    if (ContadorPulos == 1) {
        VelocidadeX = (VelocidadeHorizontalMaxima * DirecaoHorizontal) * 0.8f;
    }
    if (ContadorPulos == 2) {
        VelocidadeX = (VelocidadeHorizontalMaxima * DirecaoHorizontal) * 0.6f;
    }
    if (ContadorPulos == 3) {
        VelocidadeX = (VelocidadeHorizontalMaxima * DirecaoHorizontal) * 0.4f;
    }
    // Preserva a velocidade atual no eixo Y para manter a física de pulo/queda
    VelocidadeY = CorpoRigidoPersonagem.velocity.y;
    // Cria um novo vetor de velocidade combinando os componentes X (calculado) e Y (preservado)
    VetorVelocidadePersonagem = new Vector2(VelocidadeX, VelocidadeY);
    // Aplica o vetor de velocidade ao Rigidbody2D para efetivar o movimento
    CorpoRigidoPersonagem.velocity = VetorVelocidadePersonagem;
    // Controla a inversão do sprite (flip) baseado na direção do movimento:
    // - Para esquerda (valores negativos): ativa flipX
    // - Para direita (valores positivos): desativa flipX
    if (DirecaoHorizontal < 0) {
        Rerender.flipX = true;
    } else if (DirecaoHorizontal > 0) {
        Rerender.flipX = false;
    }
}
```

O que o código faz?

Como fazer um trampolim!

- Configure a plataforma com a tag “trampolim” e com um colisor.




```
void OnCollisionEnter2D(Collision2D objetoTocado ){
   TagObjetoTocado = objetoTocado.gameObject.tag;
   if (TagObjetoTocado == "trampolim") {
       CorpoRigidoPersonagem.velocity = new Vector2 (0, 15);

   }
}
```