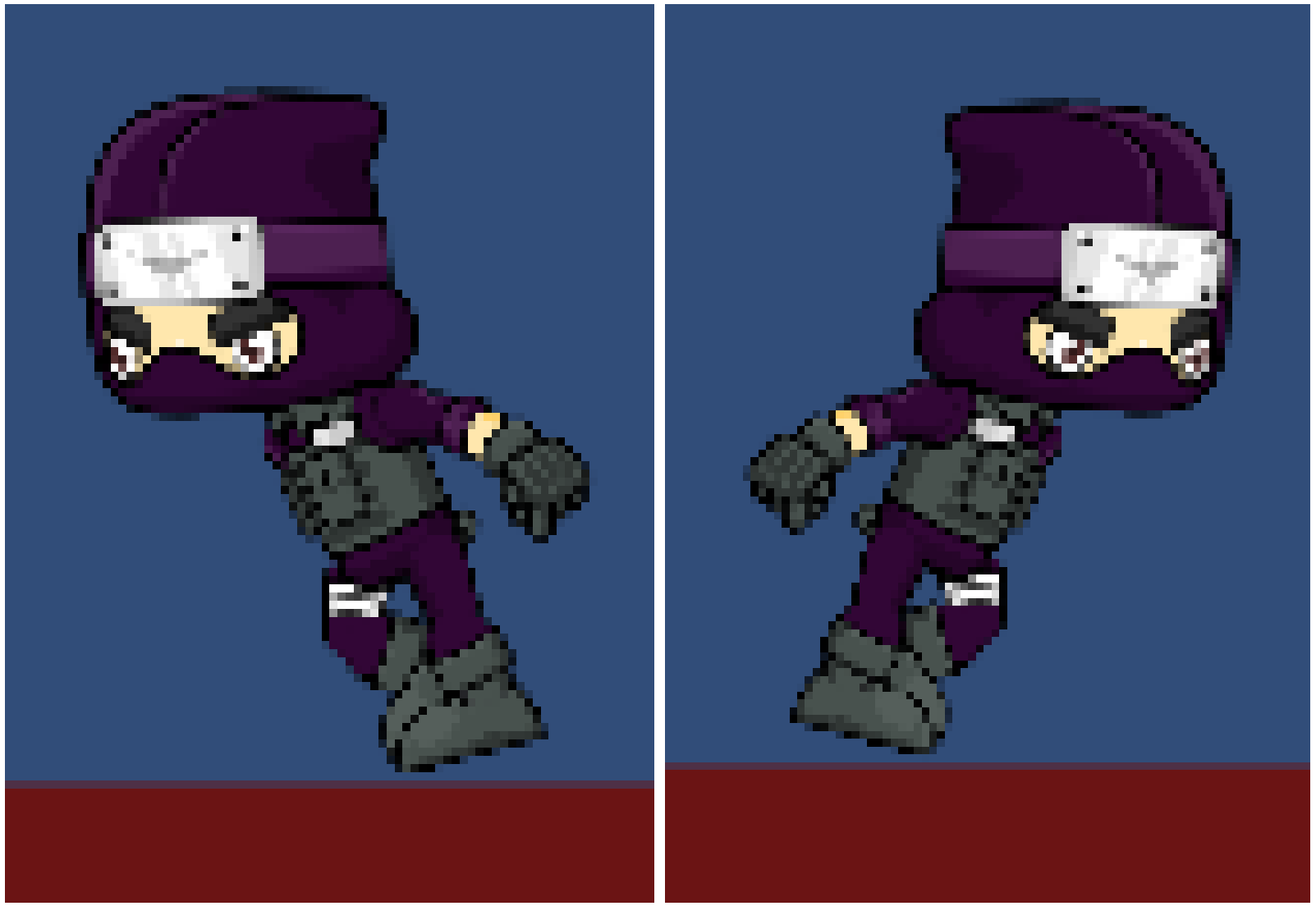


Flip

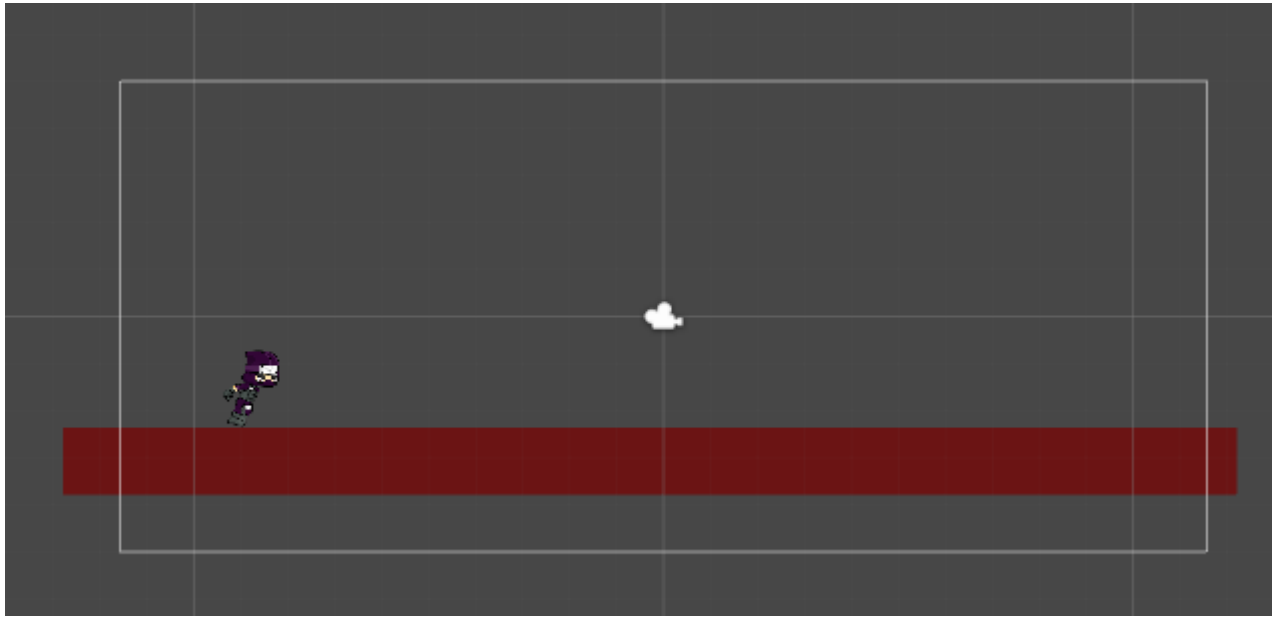
Prof. Me. Hélio Esperidião

Flip

- Permite inverter uma imagem como se observasse por um espelho.
- Tipicamente assets de personagens apontam para a direita.
  - Quando o personagem andar para a esquerda seria ideal espelhar o personagem e fazer com que sua imagem aponte para a esquerda.

Configuração

- Personagem:
  - Corpo Rígido
  - Caixa de Colisão
- Chão
  - Caixa de Colisão



Atributos

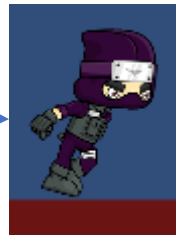
```
using UnityEngine;

public class Temp : MonoBehaviour {

    SpriteRenderer Renderer;

    string TagObjetoTocado;

    bool ApertouBotaoPular; // para saber se o botão de pular foi apertado
    float VelocidadePuloSimples; // varivael 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
    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
    // Referência ao componente Rigidbody2D do personagem
    Rigidbody2D CorpoRigidoPersonagem;
```

Inicializar o SpriteRenderer

```
void Start () {
    Renderer = GetComponent<SpriteRenderer>();

    string TagObjetoTocado;

    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
    // Referência ao componente Rigidbody2D do personagem
    Rigidbody2D CorpoRigidoPersonagem;

}
```

Update ()

```
void Update () {

        MovimentoHorizontalFlip ();
        MovimentoPuloUnico();
        //MovimentoHorizontal ();
        //MovimentoVertical ();
        //MovimentoPuloSimples();
    }
```

MovimentoHorizontalFlip ();

```
void MovimentoHorizontalFlip(){

    // Obtém a direção horizontal a partir do input do jogador (teclado ou controle)
    // Input.GetAxis("Horizontal") retorna um valor entre -1 e 1:
    // -1 para esquerda, 1 para direita, 0 se nenhuma tecla do eixo horizontal estiver sendo pressionada
    DirecaoHorizontal = Input.GetAxis ("Horizontal");

    // Calcula a velocidade no eixo X multiplicando a direção pela velocidade máxima
    VelocidadeX = VelocidadeHorizontalMaxima * DirecaoHorizontal;

    // Mantém a velocidade atual no eixo Y (para não interferir na gravidade ou pulo)
    VelocidadeY = CorpoRigidoPersonagem.velocity.y;

    // Cria um novo vetor de velocidade com os valores calculados
    VetorVelocidadePersonagem = new Vector2 (VelocidadeX, VelocidadeY);

    // Aplica o vetor de velocidade ao Rigidbody2D para mover o personagem
    CorpoRigidoPersonagem.velocity = VetorVelocidadePersonagem;
    if (DirecaoHorizontal < 0) {
        Renderer.flipX = true;
    } else if (DirecaoHorizontal > 0) {
        Renderer.flipX = false;
    }
}
```