

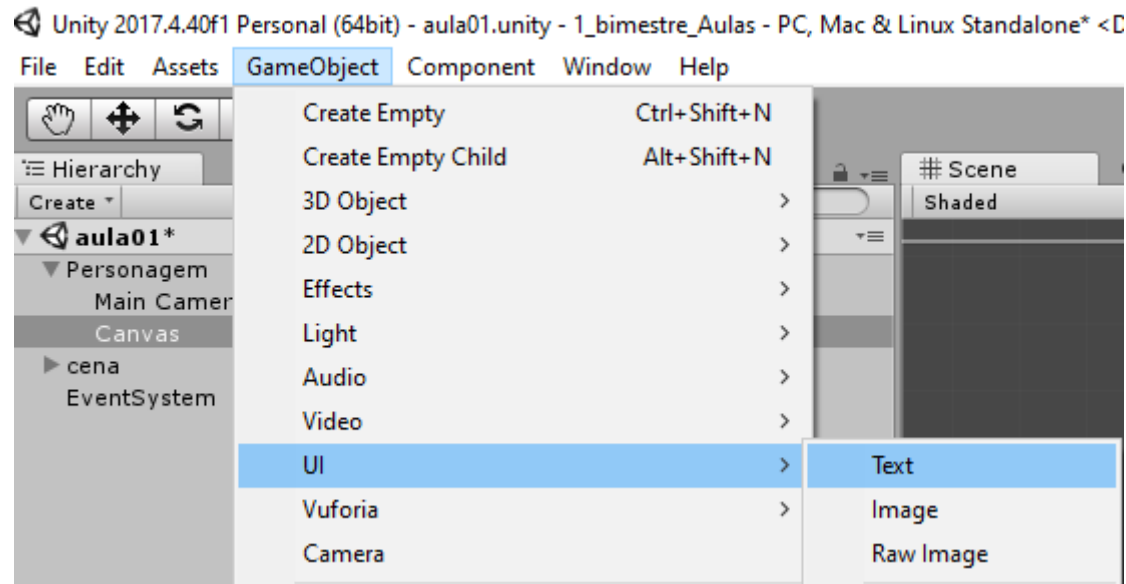


Prof. Me. Hélio Esperidião

Apresentação de textos em Unity - Componente “Text”

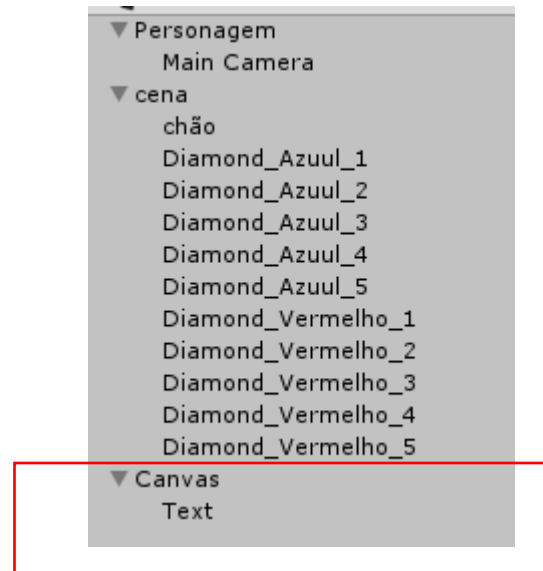
Adicionar um componente de texto

- GameObject >> UI >> Text



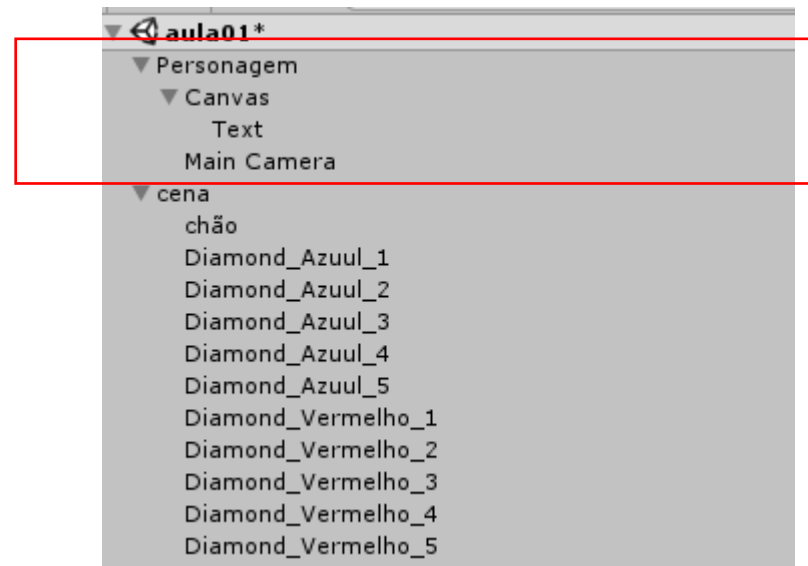
Canvas unity

- O Canvas do Unity é um componente gráfico que fornece um espaço de tela para que os elementos da interface do usuário (UI) sejam renderizados.
- Ele é usado para criar a interface do usuário do jogo, incluindo botões, textos, imagens e outros elementos interativos.



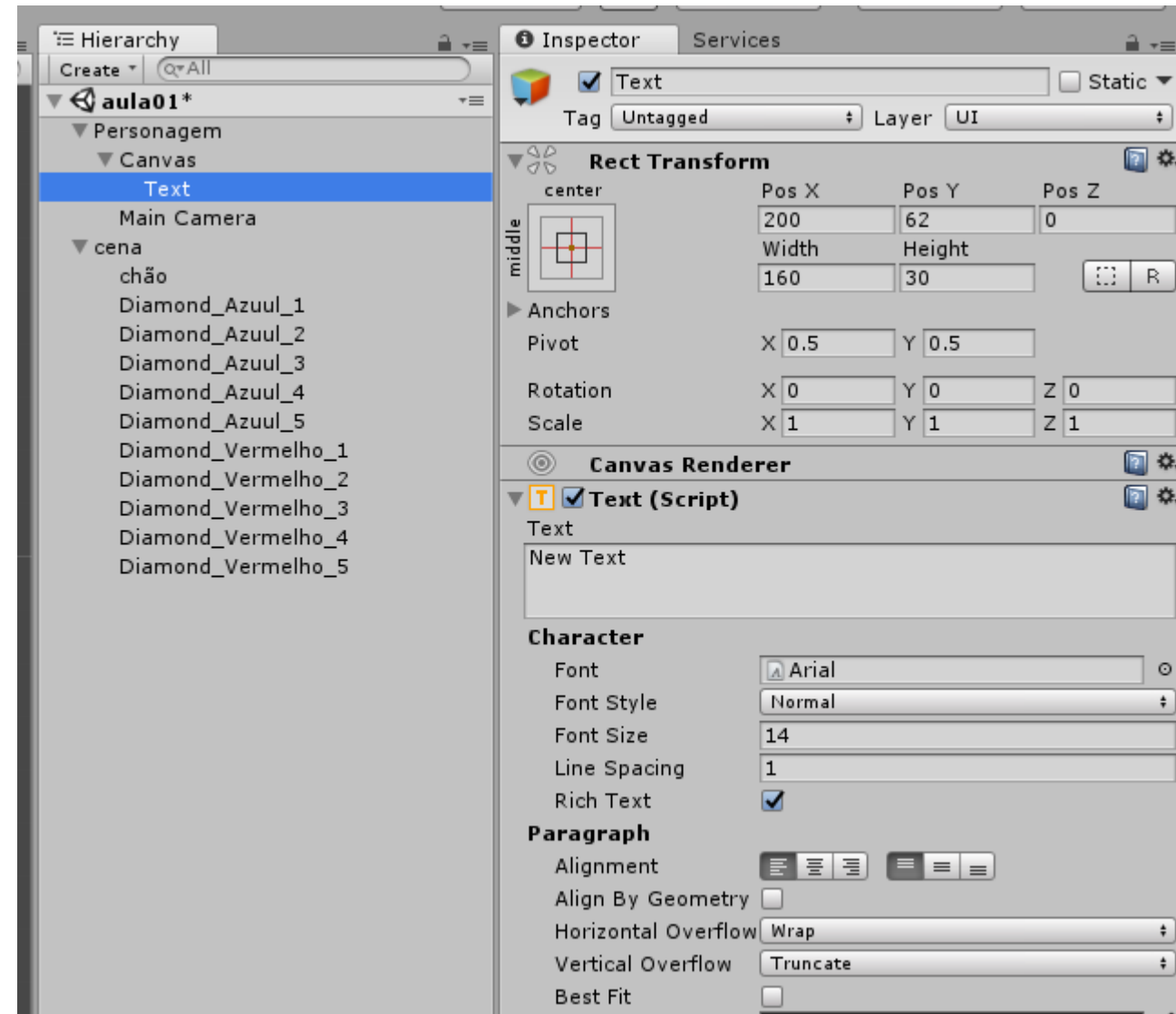
Mova o canvas para dentro do personagem

- Mover o canvas para dentro do personagem permite que o canvas se movimente junto com o personagem.



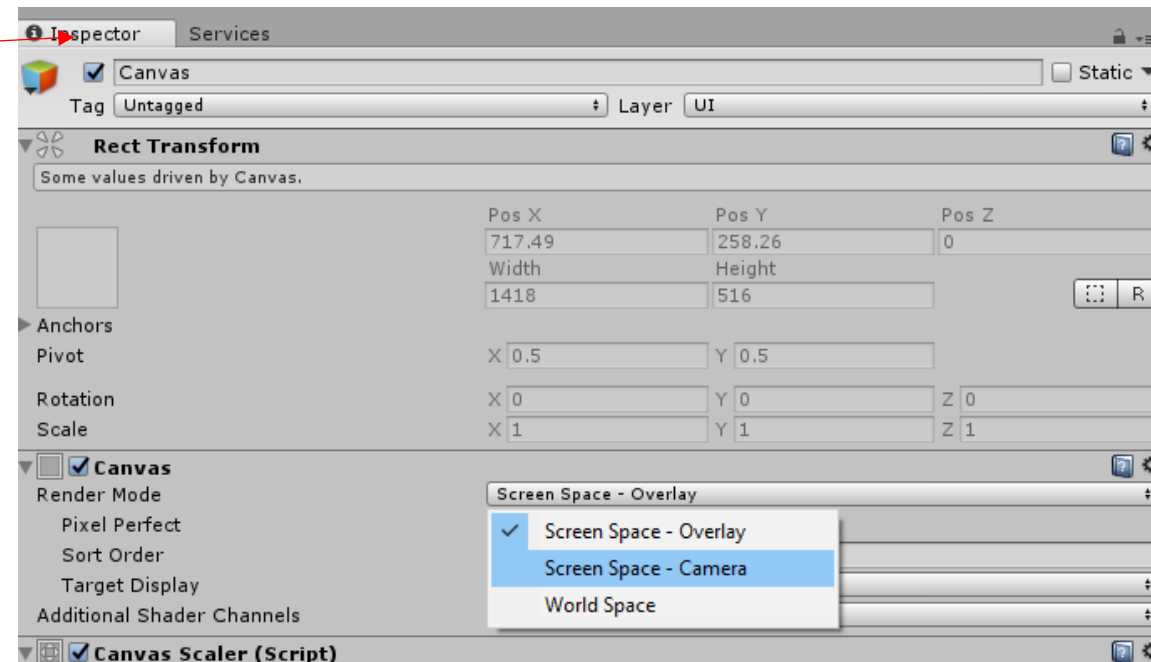
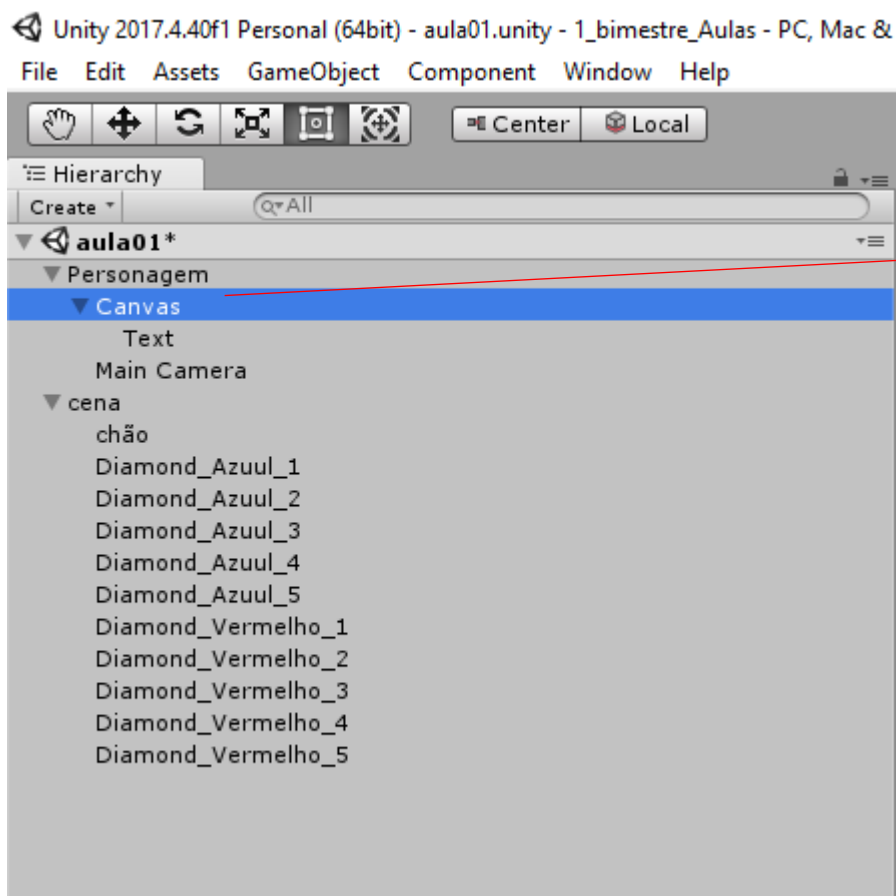
Propriedades de texto

- Selecione o componente de texto.
 - No inspector altere propriedades de texto como cor, **negrito** e *itálico*, alinhamento e etc.



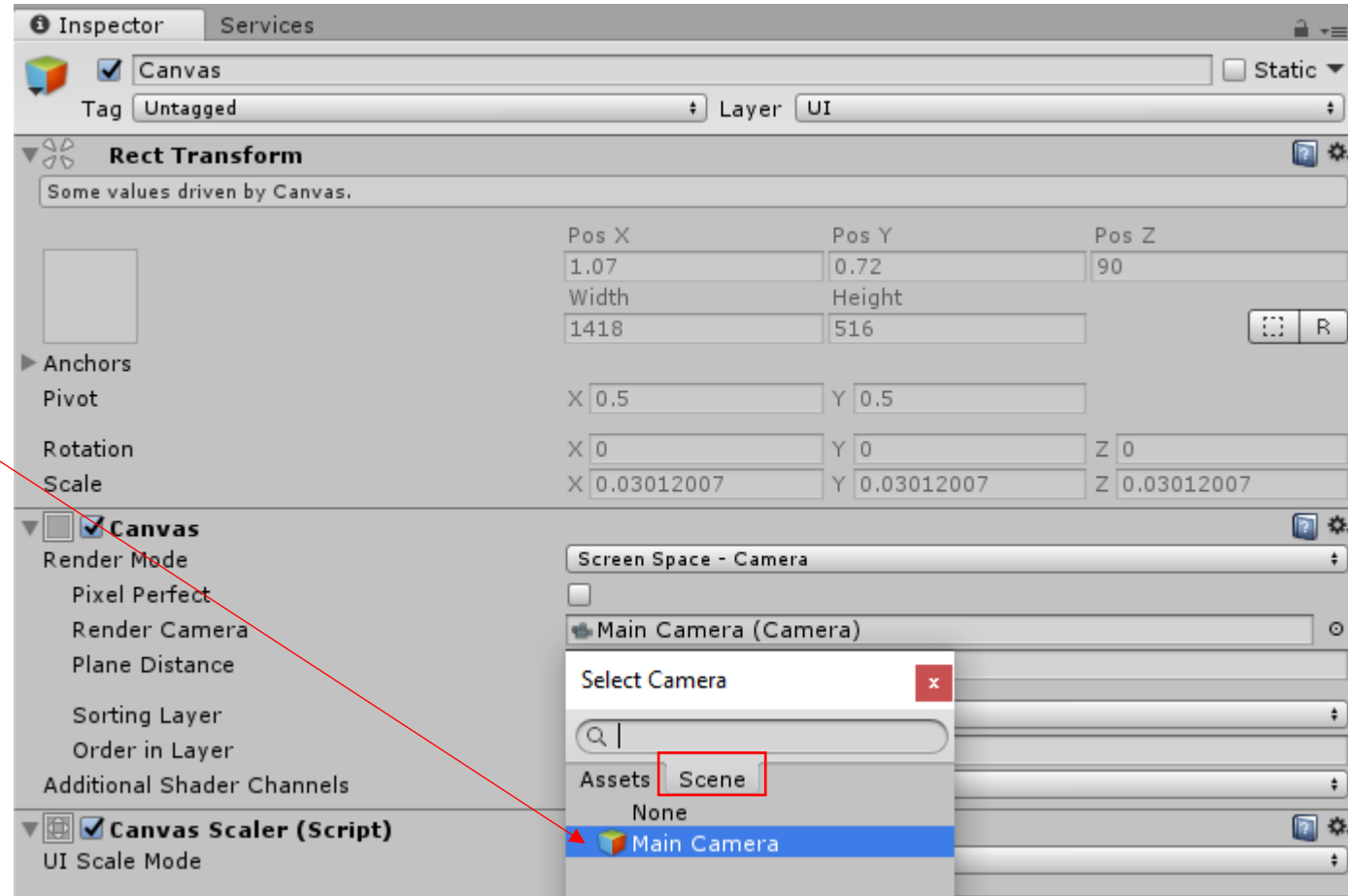
Altere a renderização de câmera

- Clique no canvas posicionado na hierarquia do projeto, no inspector mude a propriedade render mode para “Screen Space - Camera”



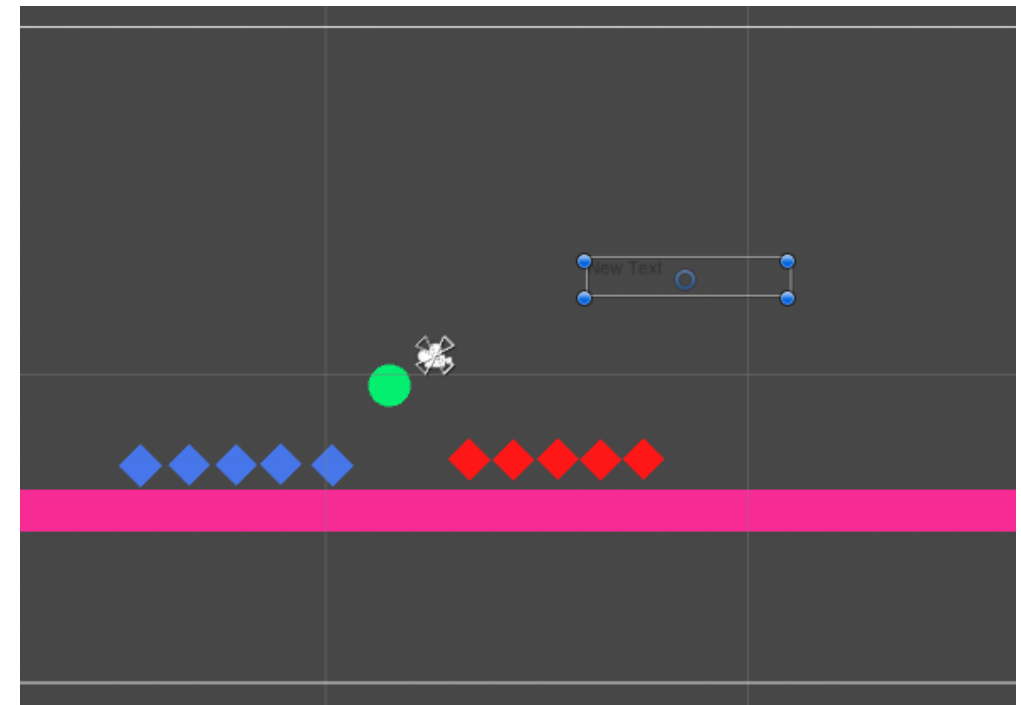
Render Camera

- Clique duas vezes na propriedade “Render Camera” e selecione “Main Camera” com dois cliques.



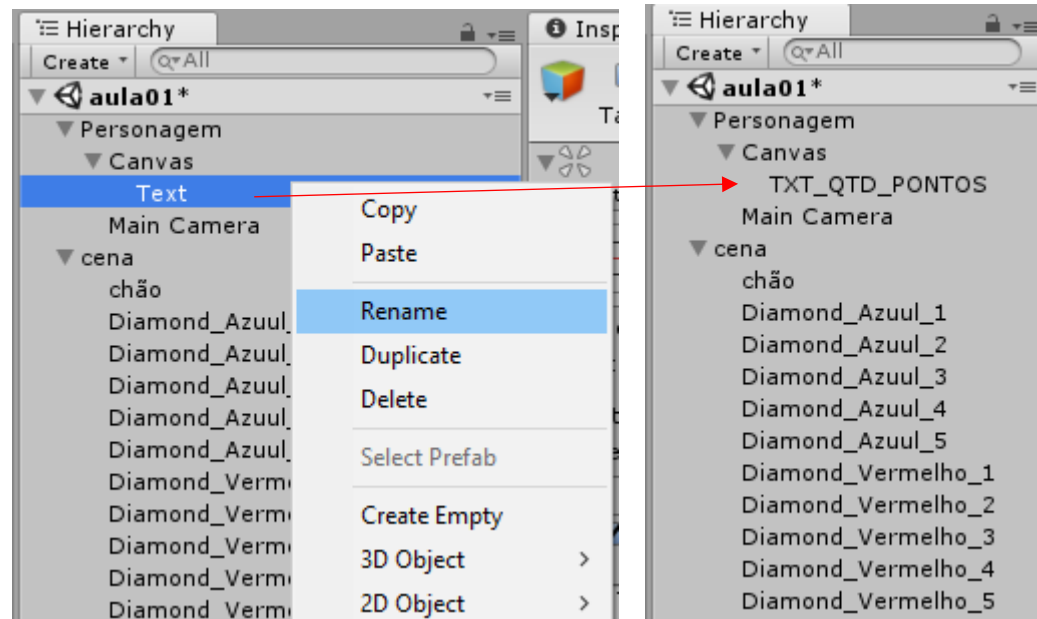
Configure os componentes

- Posicione livremente o componente de texto no cenário.
- Ao tocar em “pontos1” o personagem ganha 1 ponto.
Ao tocar em “pontos2” o personagem ganha 2 pontos.
- Configurações:
 - Diamante azul
 - Colisor
 - Tag: “pontos1”
 - Diamante vermelho
 - Colisor
 - Tag: “pontos2”



Altere o nome do componente

- Para identificar o componente de texto na programação é necessário modificar o seu nome.
- O exemplo modifica o nome de “Text” para “TXT_QTD_PONTOS”.



Biblioteca

```
using UnityEngine;  
//Biblioteca utilizada para cenários  
using UnityEngine.SceneManagement;  
//Biblioteca utilizada para trabalhar com o UI  
using UnityEngine.UI;
```

Atributos

```
public class tempBimestre2 : MonoBehaviour {
```

*// Observe que antes do tipo da variável há a palavra “public”, isso significa
// que esse atributo pode ser acessado de outros lugares
// e é possível visualizar e inicializar esse atributo na interface do inspector.*

```
public Text UITextPontos;
```

```
int QtdPontos;
```

```
string TagTriggerStay;  
string TagTriggerEnter;  
string TagTriggerExit;  
string TagObjetoStay;  
string TagObjetoExit;
```

```
int TotalPulos;
```

```
PhysicsMaterial2D MaterialSemAtrito;
```

```
int ContadorPulos;
```

```
SpriteRenderer Renderer;
```

```
string TagObjetoTocado;
```

```
bool PegouChave01;
```

```
bool ApertouBotaoPular; // para saber se o botão de
```

```
float VelocidadePuloSimples; // variavel que guarda
```

```
Collider2D Colisor2dPersonagem; //variável que arma
```

```
bool EstaTocandoAlgumColisor; //Variável que verific
```

```
// Variáveis para controlar a velocidade do personag
```

```
float VelocidadeX; // Velocidade no eixo X (horizont
```

```
float VelocidadeY; // Velocidade no eixo Y (vertical
```

```
float VelocidadeHorizontalMaxima; // Velocidade máxi
```

```
float DirecaoHorizontal; // Direção do movimento hor
```

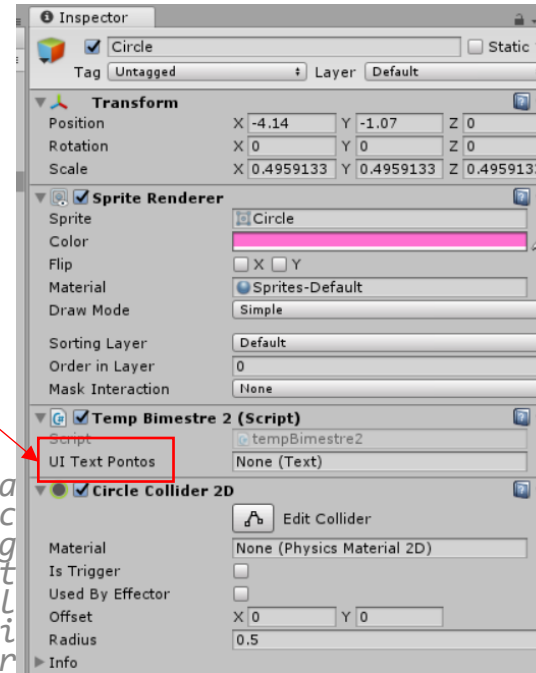
```
float VelocidadeVerticalMaxima; //Variável que armazena a velocidade Máxima Vertical do personagem
```

```
float DirecaoVertical; // Variável que armazena a direção vertical do personagem
```

```
Vector2 VetorVelocidadePersonagem; // Vetor que armazena a velocidade do personagem em X e Y
```

```
// Referência ao componente Rigidbody2D do personagem
```

```
Rigidbody2D CorpoRigidoPersonagem;
```



Personagem

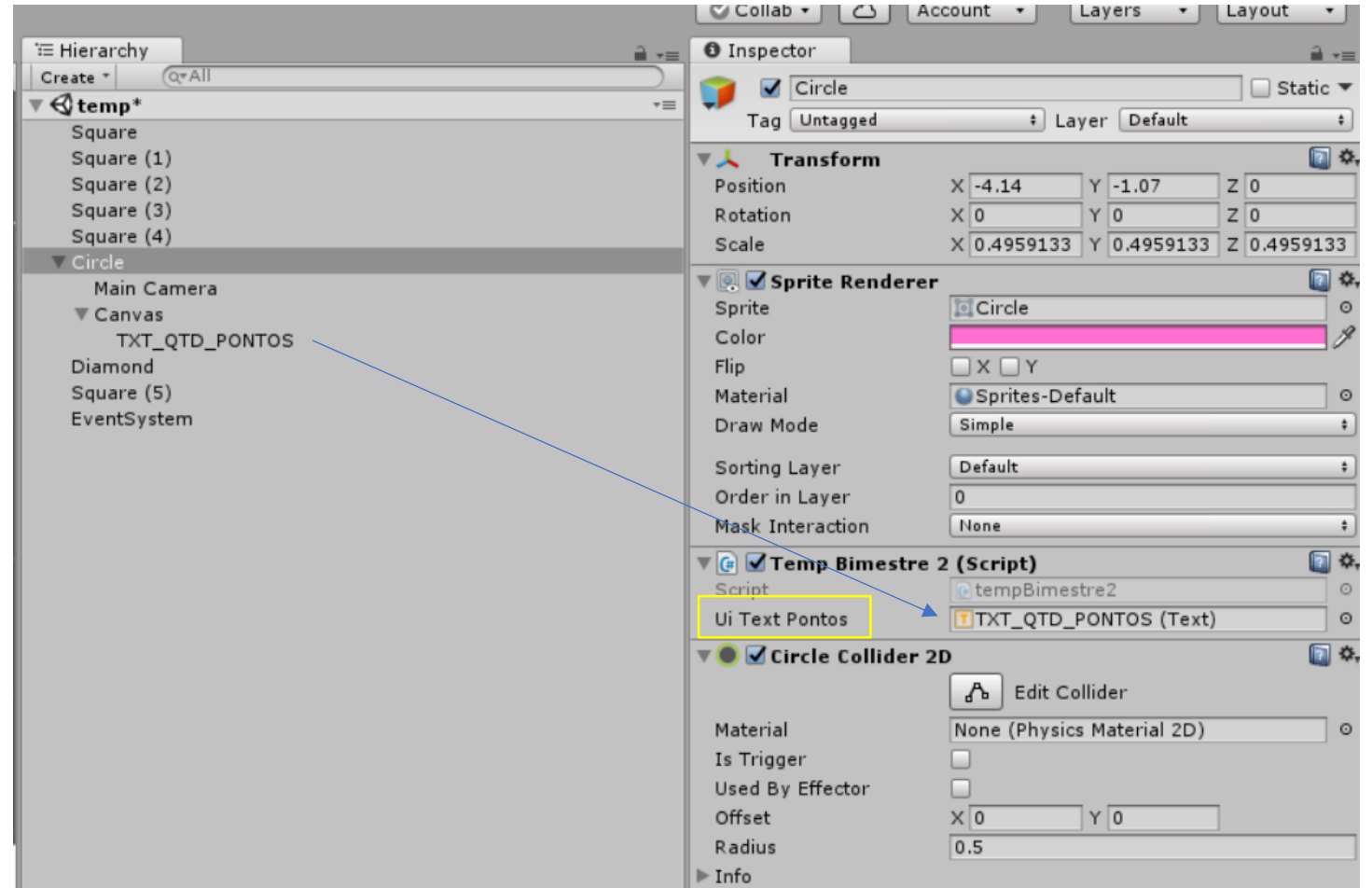
Quando um colisor

atingir no eixo X

(para direita, 0 para parado)

Configure

- A inicialização do atributo **UITextPontos** pode ser diferente quando público
- Para inicializar, arraste do componente “TXT_QTD_PONTOS” para dentro de **UITextPontos** no inspector



Método Start()

```
void Start () {  
    QtdPontos = 0;  
    TagTriggerStay="";  
    TagTriggerEnter="";  
    TagTriggerExit="";  
    TagObjetoStay="";  
    TagObjetoExit="";  
    MaterialSemAtrito = new PhysicsMaterial2D();  
    TotalPulos = 1;  
    ContadorPulos = 0;  
    Renderer = GetComponent<SpriteRenderer> ();  
    PegouChave01 = false;  
    TagObjetoTocado = "";  
    ApertouBotaoPular = false; // inicia a variavel com false;  
    EstaTocandoAlgunColisor = 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;  
}
```

OnCollisionEnter2D()

```
void OnCollisionEnter2D(Collision2D objetoTocado ){  
  
    TagObjetoTocado = objetoTocado.gameObject.tag;  
  
    if (TagObjetoTocado == "pontos1" ) {  
        QtdPontos=QtdPontos+1; //soma 1 em QtdPontos  
        //Converte o valor de QtdPontos para o tipo string(texto)  
        string textoPontos= QtdPontos.ToString();  
        //Armazena o texto com os pontos na propriedade  
        //Text de UITextPontos  
        UITextPontos.text = textoPontos;  
        //destrói pontos1  
        Destroy (objetoTocado.gameObject);  
    }  
  
    if (TagObjetoTocado == "pontos2" ) {  
        QtdPontos=QtdPontos+2; //soma 2 em QtdPontos  
        //Converte o valor de QtdPontos para o tipo string(texto)  
        string textoPontos= QtdPontos.ToString();  
        //Armazena o texto com os pontos na propriedade  
        //Text de UITextPontos  
        UITextPontos.text = textoPontos;  
        Destroy (objetoTocado.gameObject); //destrói pontos2  
    }  
  
}
```

Transforma um número em texto
Pois: **UITextPontos.text**
só é capaz de receber
variáveis
do tipo texto(string)

Text é uma propriedade de
UITextPontos que contém o texto
que será apresentado.