

Desenvolvimento com Motores de Jogos II

Aula 05 - Jogo Polygonal Rescue - Parte 3 - HUD e Pontuação do Jogo

Apresentação

Olá pessoal, tudo bem com vocês? Hoje estamos iniciando a Aula 5 da nossa disciplina. Mas antes, vamos lembrar o conteúdo visto na aula anterior! Finalizamos uma versão do jogo *Polygonal Rescue*, na qual o personagem já consegue resgatar os polígonos perdidos no labirinto. Para isso, usamos um mecanismo de coleta de item em que, ao contato, o personagem remove os objetos marcados como coletável. Na aula de hoje, continuaremos com uma difícil tarefa, SQN! Criaremos um simples HUD para o nosso jogo e exibiremos a pontuação do jogador nele. Então, inicialmente precisamos aprender a contabilizar os pontos (polígonos resgatados) do jogador e armazená-los em uma variável, para posteriormente serem exibidos na tela. Vamos lá?

Objetivos

Ao final desta aula, você deverá ser capaz de:

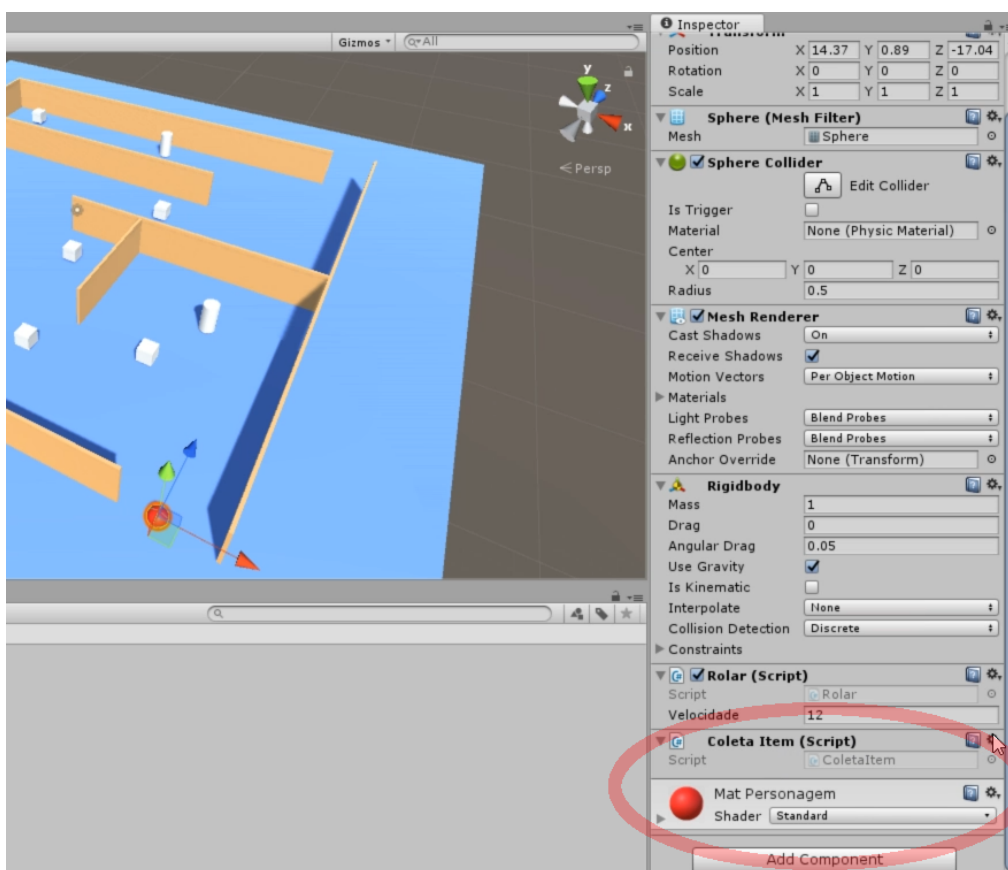
- Criar um HUD (Heads-up display), componente em jogos digitais que exibe informações importantes sobre o jogo;
- Contabilizar e exibir a pontuação do jogador durante o jogo.

Contabilização dos Pontos

No *Polygonal Rescue*, os pontos do jogador correspondem à quantidade de polígonos (atualmente Cubos e Cilindros) que ele resgatou no labirinto.

Criaremos um mecanismo para armazenar os pontos atuais do jogador e iremos incrementá-los cada vez que o jogador resgatar mais um polígono. Para isso, clique no personagem e localize no Inspector o **script ColetaItem**, que está associado ao personagem como um componente. Veja a **Figura 1**.

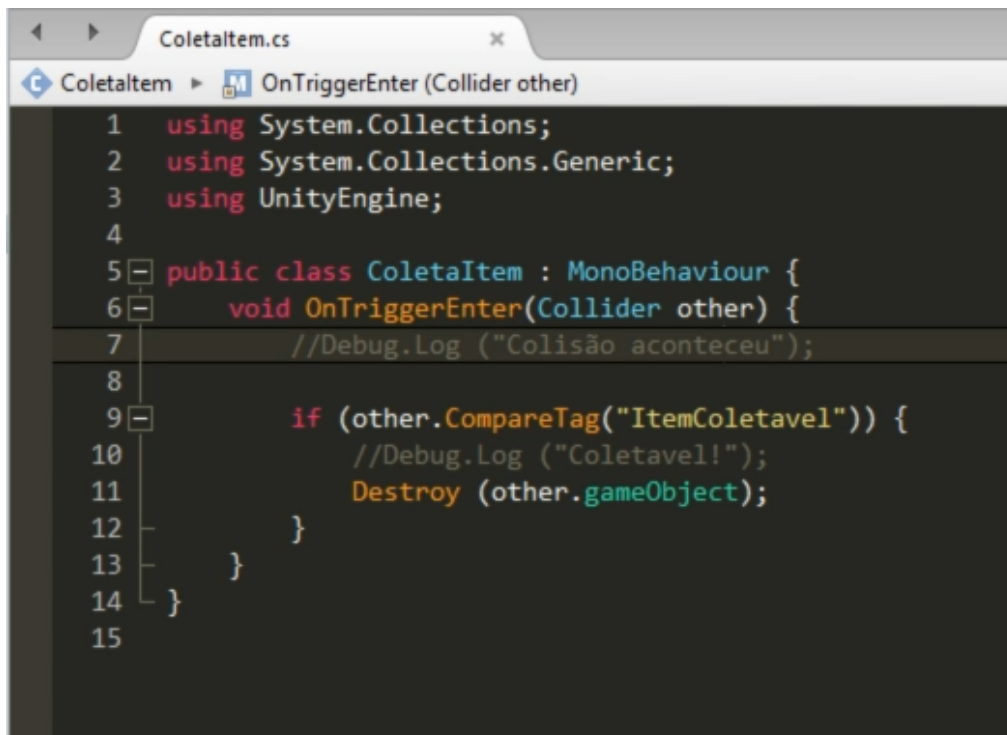
Figura 01 - Script ColetaItem associado ao personagem.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Depois de localizar o script, escolha a opção para editá-lo no MonoDevelop. Atualmente, o **script ColetaItem** deve estar com o código-fonte, assim como mostra a **Figura 2**.

Figura 02 - Código atual do script ColetaItem.



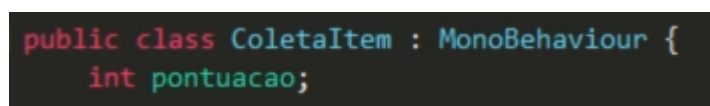
```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class ColetaItem : MonoBehaviour {
6     void OnTriggerEnter(Collider other) {
7         //Debug.Log ("Colisão aconteceu");
8
9         if (other.CompareTag("ItemColetavel")) {
10             //Debug.Log ("Coletavel!");
11             Destroy (other.gameObject);
12         }
13     }
14 }
15
```

Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Fazendo uma breve revisão, repare que o script tem apenas um método chamado `OnTriggerEnter`, o qual recebe um `Collider` como parâmetro. Esse método é chamado quando o personagem entra em contato com um objeto qualquer que tem um colisor marcado como `Is Trigger`, que tem sua referência passada para o método na variável **other**. Dentro do método, uma verificação é realizada usando o `other.CompareTag("ItemColetavel")` a fim de assegurar que o objeto é um elemento marcado para ser coletado, ou seja, um polígono a se resgatar. Se for verdade, o seu `GameObject` é removido da cena.

Pela praticidade, inicialmente usaremos esse mesmo script para armazenar a pontuação atual do jogador, pois nele existe o método que resgata os polígonos e modificará essa pontuação em breve. Para isso, crie uma variável do tipo **int** chamada "pontuacao" nessa classe, como mostra a **Figura 3**.

Figura 03 - Variável criada para armazenar a pontuação do jogador.



```
public class ColetaItem : MonoBehaviour {
    int pontuacao;
}
```

Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Em seguida, crie o método **Start()** para inicializar o valor dessa variável em zero, como na **Figura 4**.

Atenção!

Lembre-se que o método `Start()` é executado apenas no início do jogo e somente uma vez para cada objeto ao qual o script está associado.

Figura 04 - Método `Start` do script `ColetaItem` iniciando o valor da pontuação em zero.

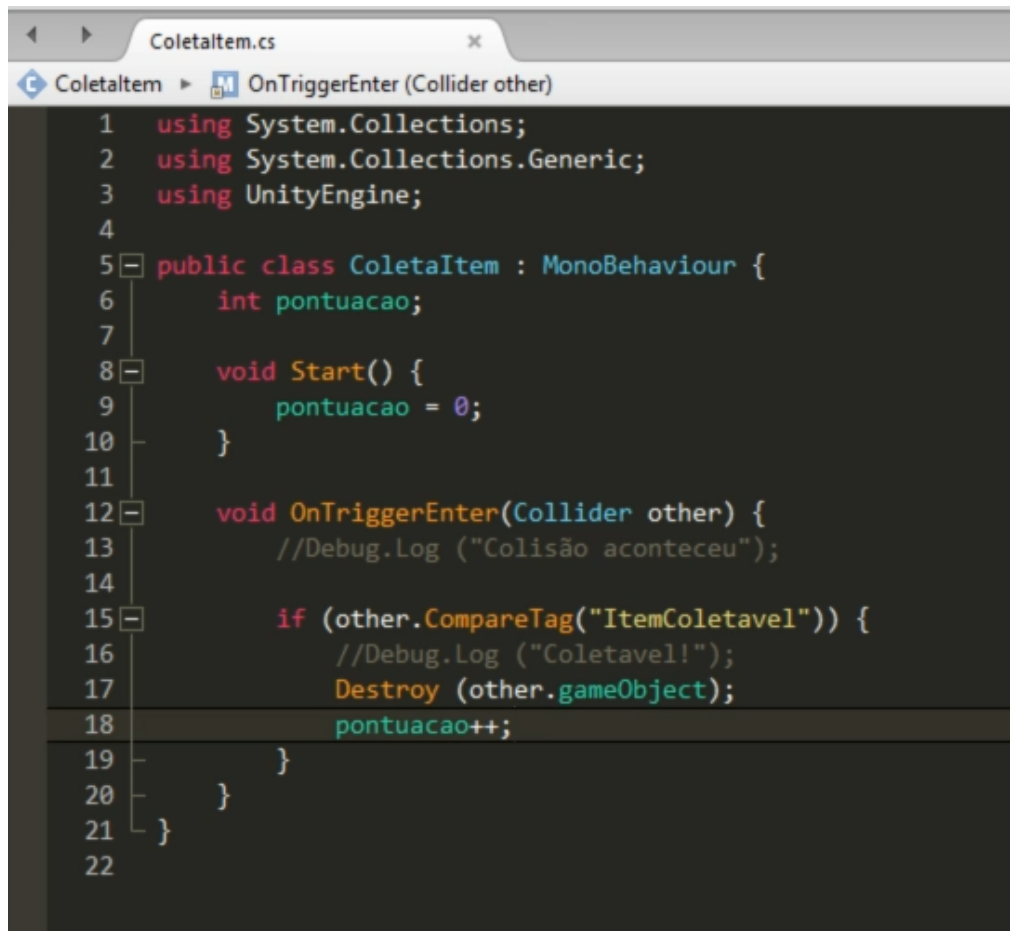
```
public class ColetaItem : MonoBehaviour {  
    int pontuacao;  
  
    void Start() {  
        pontuacao = 0;  
    }  
}
```

Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Agora, todas as vezes que resgatarmos um polígono (no método `OnTriggerEnter`, nesse mesmo script), Após o comando **`Destroy(other.gameObject);`** incrementaremos a variável **`pontuacao`** criada em 1 utilizando o comando **`pontuacao++`**.

Sendo assim, modifique o método `OnTriggerEnter` para adicionar essa funcionalidade, deixando o código do **script `ColetaItem`**. Veja a **Figura 5**.

Figura 05 - Script ColetaItem acumulando a pontuação do jogador.



```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class ColetaItem : MonoBehaviour {
6     int pontuacao;
7
8     void Start() {
9         pontuacao = 0;
10    }
11
12    void OnTriggerEnter(Collider other) {
13        //Debug.Log ("Colisão aconteceu");
14
15        if (other.CompareTag("ItemColetavel")) {
16            //Debug.Log ("Coletavel!");
17            Destroy (other.gameObject);
18            pontuacao++;
19        }
20    }
21 }
22
```

Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Muito simples, não é? Mas calma, até agora só criamos uma forma de cada polígono resgatado adicionando 1 ao valor da variável **pontuacao**. Apesar do funcionamento desse código, mesmo executando o jogo não veremos em lugar algum a pontuação do jogador, pois ainda não criamos os elementos na interface do jogo para exibir o código nem adicionamos os comandos de exibição desse valor nessa interface.

Criação do HUD

HUD, do inglês, heads-up display, ou “tela de alerta”, é um componente em jogos digitais que exibe informações importantes sobre o jogo na tela, em frente aos objetos do jogo. Essas informações podem ser de vários tipos, como a vida do jogador, energia, dinheiro, pontos, inventário, etc.

O Unity tem diversos componentes para criação de **HUDs** nos nossos jogos. No Unity, esses elementos fazem parte de um grupo chamado de **UI** (do inglês User Interface).

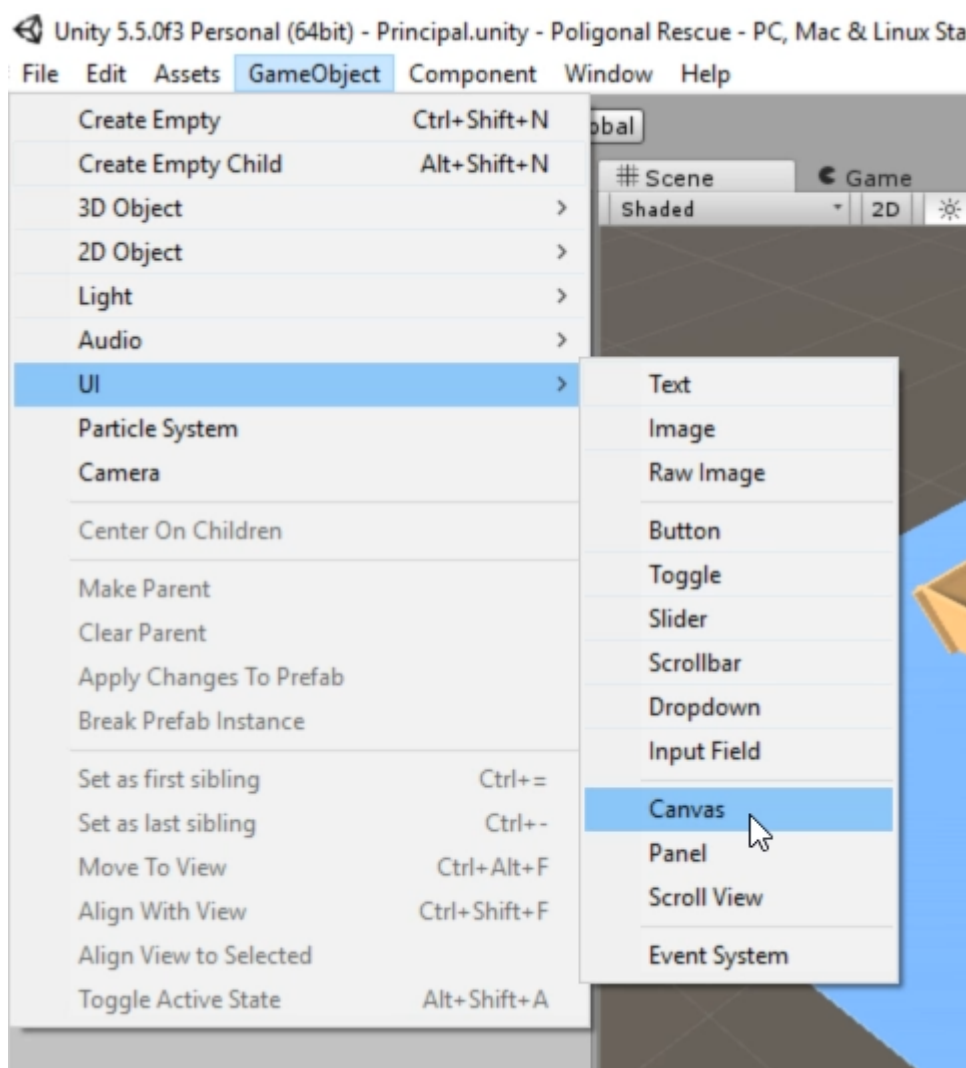
Atenção!

Para criar elementos **UI** e exibi-los em seu jogo, o Unity recomenda que inicialmente você crie um GameObject do tipo **Canvas** e, dentro desse **Canvas**, adicione diversos GameObjects filhos de outros tipos para representar textos, imagens, etc.

O Canvas é um GameObject que pertence ao grupo UI especial e representa a tela do seu dispositivo, ocupando-a totalmente, na configuração padrão, e permitindo serem adicionados outros GameObjects do grupo UI. Nas próximas aulas, estudaremos os objetos do grupo **UI** com mais detalhes. Neste momento, nos preocuparemos em criar o necessário para exibir nossa pontuação. Precisamos, então, criar um **Canvas** que ocupará toda a tela e, dentro dele, um Text que exibirá a pontuação no canto superior esquerdo do **Canvas** e, conseqüentemente, da tela do dispositivo no qual você está executando o jogo.

Para criar o **Canvas**, vá ao menu GameObject -> UI -> Canvas, como exibe a **Figura 6**.

Figura 06 - Escolhendo a opção de criação de um novo Canvas.

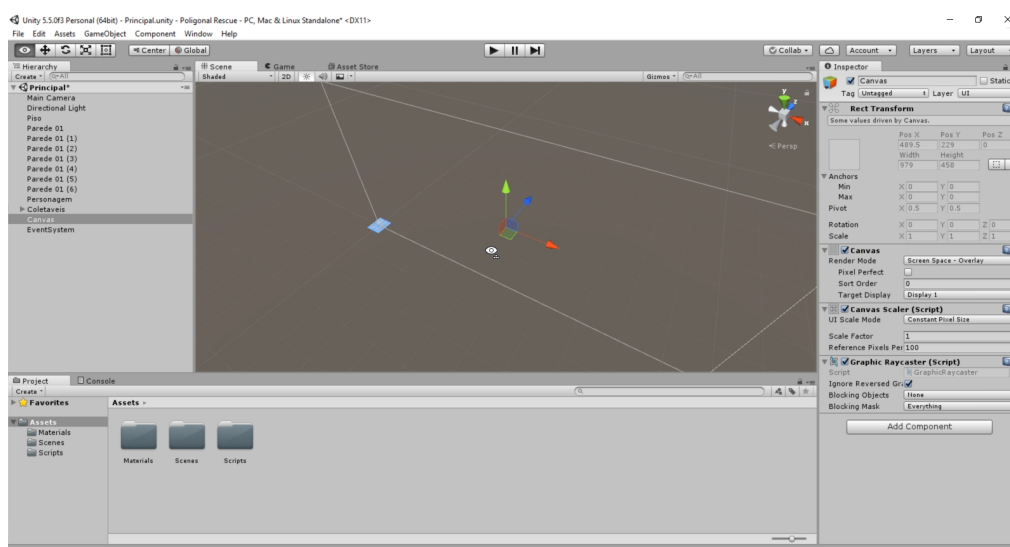


Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Logo após a sua criação, o **Canvas** estará adicionado no seu Hierarchy e, também, na janela Scene. Ele aparecerá com o nome “Canvas” no Hierarchy e na cena ele será representado como um grande objeto plano nos eixos X e Y, ocupando um espaço enorme.

Não se preocupe, pois o **Canvas**, devido à configuração padrão, ajusta-se automaticamente à sua tela, portanto as dimensões atuais dele podem ser mantidas. Veja, na **Figura 7**, o **Canvas** criado na sua cena e o modo como o labirinto fica pequeno em relação a ela (é este pequeno objeto azul).

Figura 07 - Canvas criado na cena.

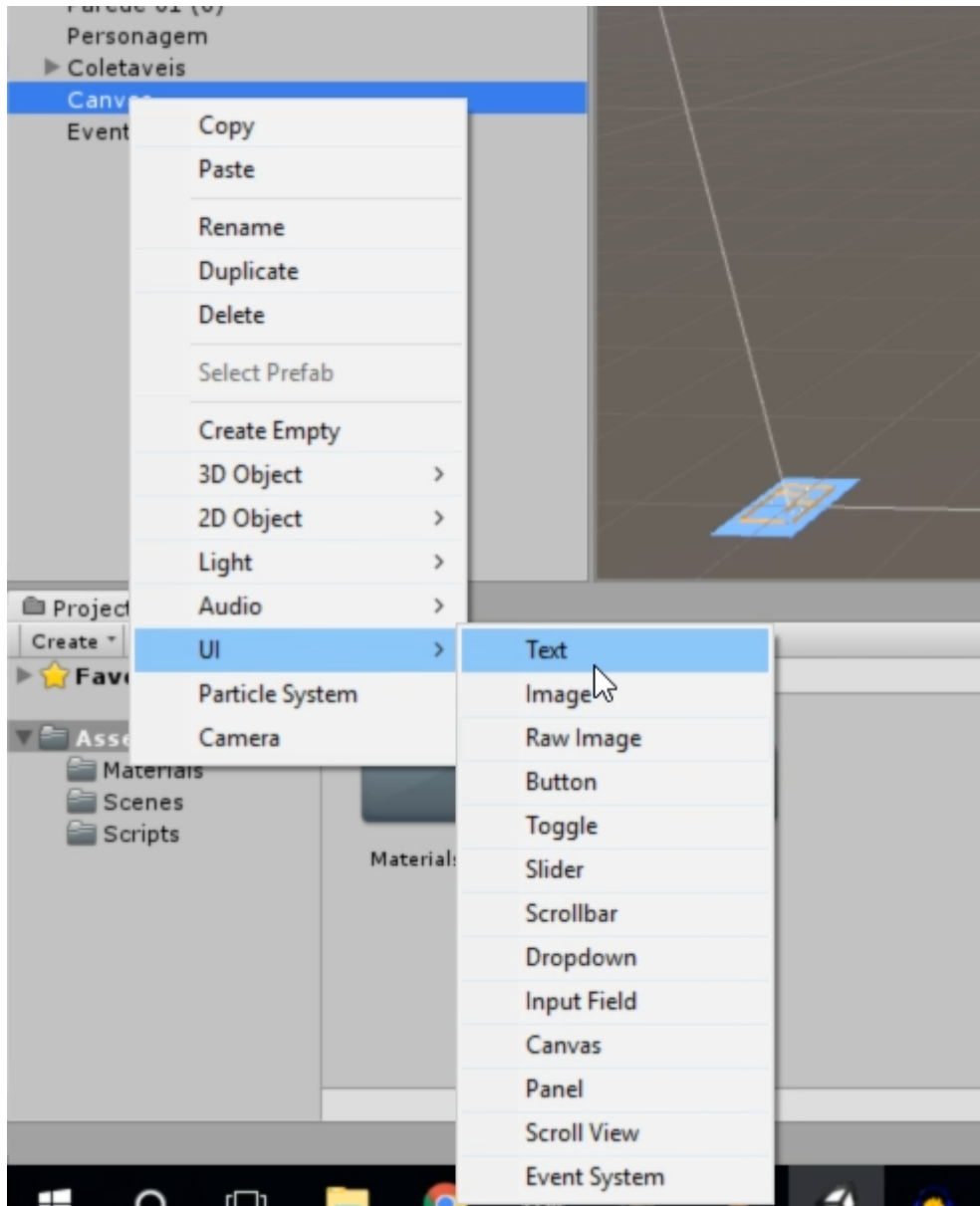


Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Para manipular o **Canvas**, a melhor maneira é ter uma visão frontal dele. Assim, mova sua visão da cena para deixá-lo ocupando uma área razoável da sua tela.

Agora, adicionaremos um elemento do tipo Text dentro do **Canvas**, a fim de exibir a nossa pontuação. Para isso, clique com o botão direito do mouse no Canvas que está criado no Hierarchy e escolha a opção UI -> Text (**Figura 8**).

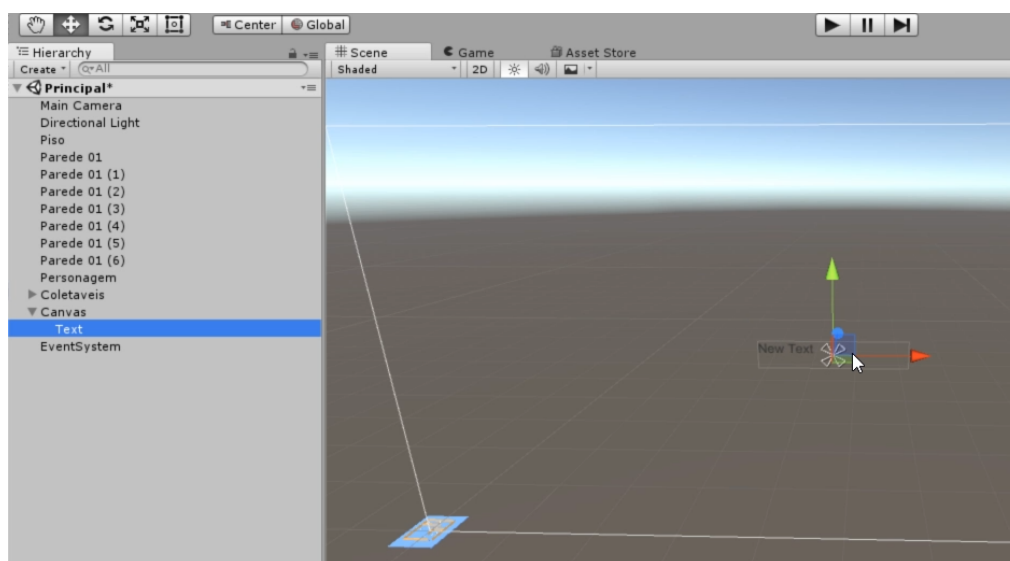
Figura 08 - Criação de um Text dentro do Canvas para exibir a pontuação.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Na **Figura 9**, repare que, depois da criação do elemento Text como filho do **Canvas**, esse elemento é exibido no seu centro com o texto padrão “New Text” em cinza.

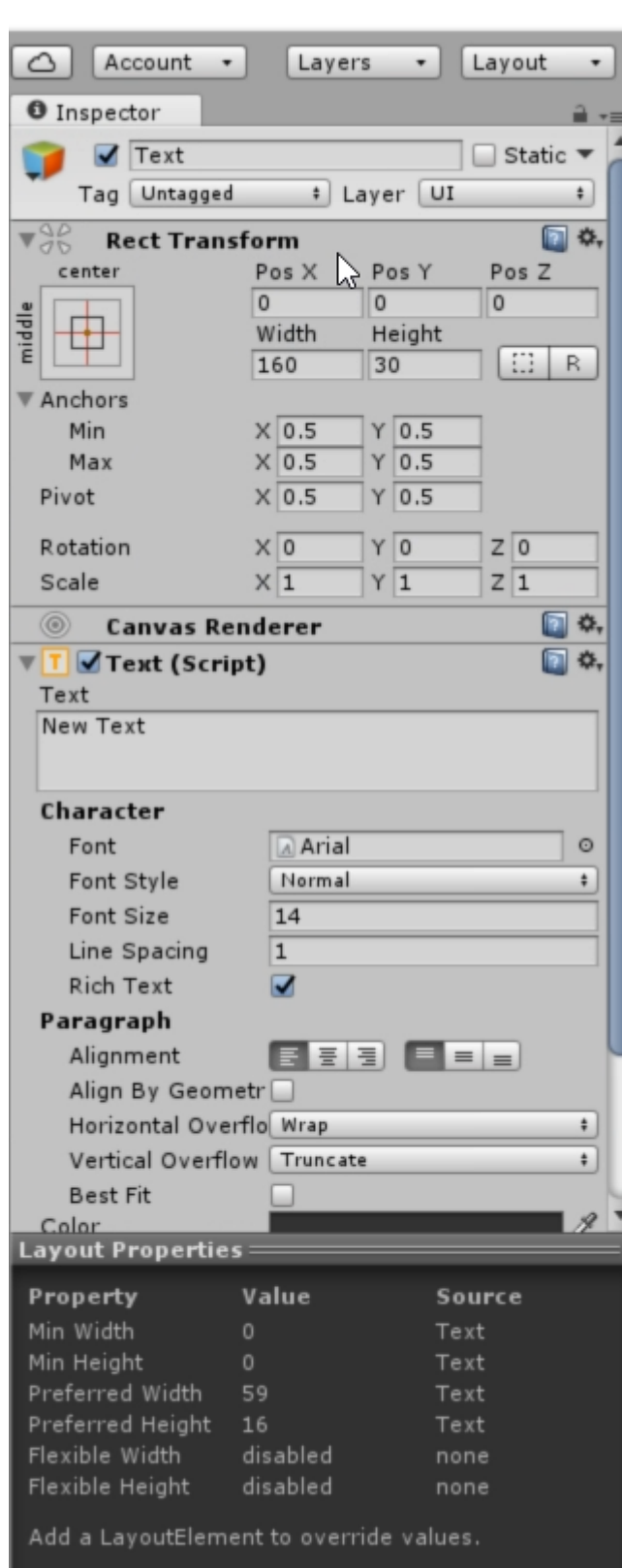
Figura 09 - Novo elemento Text, com seus valores padrão, criado e visto na cena.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Com o novo Text selecionado, vá ao Inspector para ver suas propriedades. As propriedades padrão do Text estão exibidas parcialmente na **Figura 10**.

Figura 10 - Propriedades padrão de um elemento UI do tipo Text.



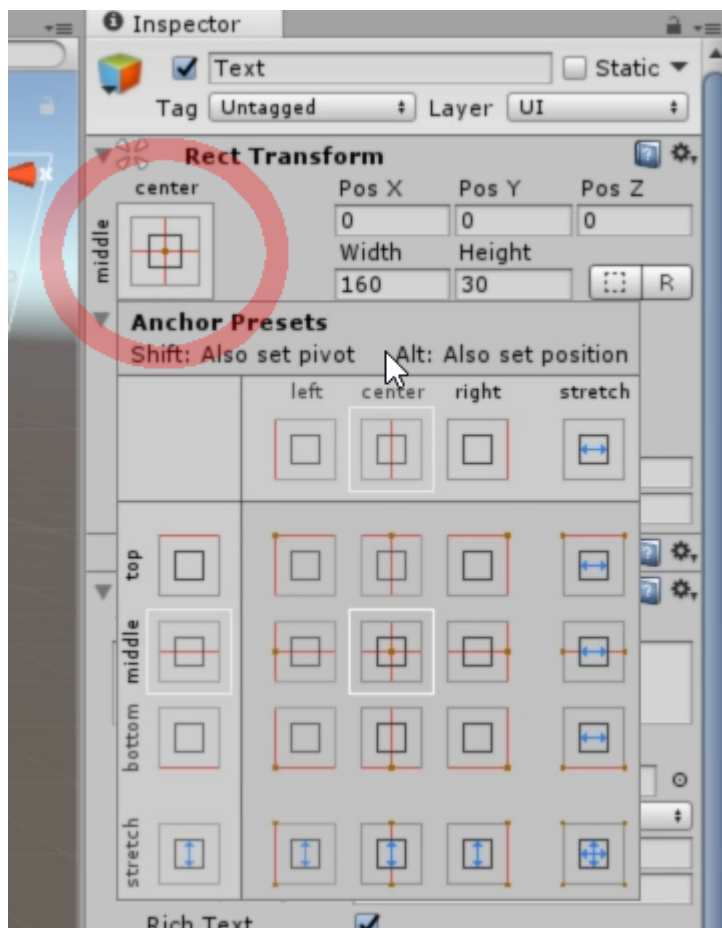
Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Observe, inicialmente, a seção Rect Transform. Ela mantém informações sobre a dimensão, rotação, alinhamento e ancoragem do Text.

O **Canvas**, conforme vimos, varia de tamanho de acordo com a tela do dispositivo que você está usando para executar nosso jogo. Então, os elementos dentro do **Canvas** (como o nosso Text) podem ser configurados para se manterem ancorados em uma das bordas da tela. Isso é muito útil, pois, se a tela for relativamente larga ou relativamente alta, não existe o risco das informações dentro do nosso **Canvas** serem perdidas e não serem vistas.

No nosso caso, queremos que o texto fique ancorado no canto superior esquerdo da tela. Assim, clique no ícone relativo às pré-configurações de âncora (Anchor Presets), o qual está dentro do RectTransform e pode ser observado detalhadamente na **Figura 11**.

Figura 11 - Ícone Anchor Presets clicado e exibindo suas opções.

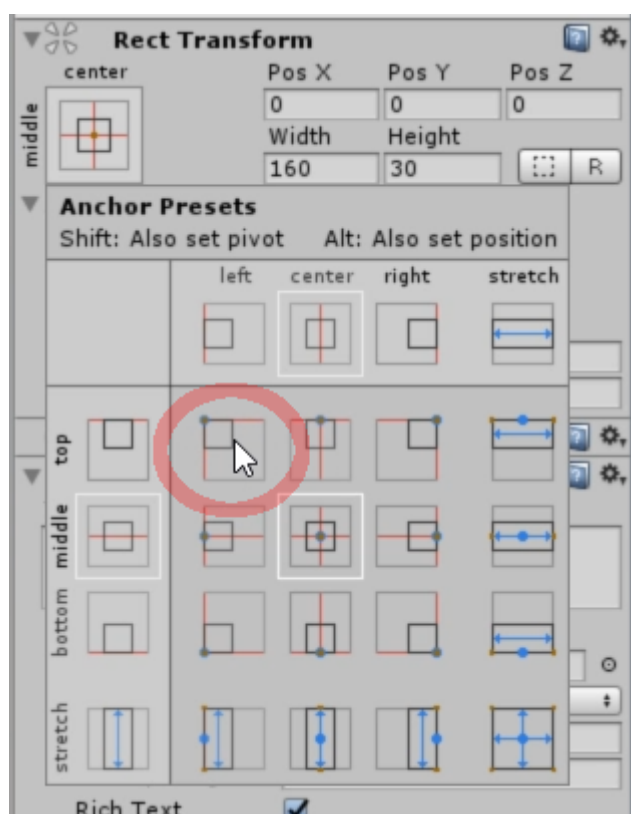


Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Repare existir, nas opções mostradas, a informação que, se você mantiver pressionado **Shift**, também será setado o pivô e, se mantiver pressionado **Alt**, também será setada a posição. Isso significa que tanto a posição como o pivô (usado também como centro de rotação e ancoragem) irão para um dos cantos escolhidos por você dentre os exibidos na tela.

Em outras aulas, aprenderemos em mais detalhes como usar o Rect Transform. Por ora, mantenha a tecla **Shift** e, também, a tecla **Alt** pressionadas, enquanto clica na opção que representa o canto superior esquerdo da tela, detalhado na **Figura 12**.

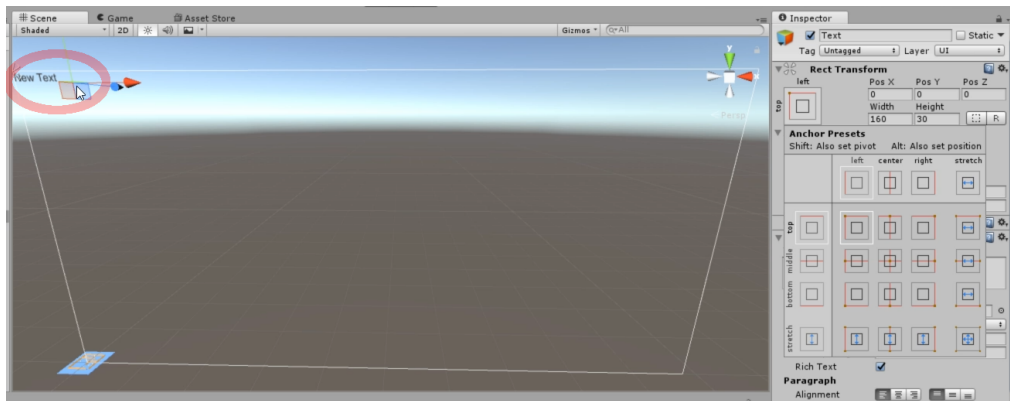
Figura 12 - Escolhendo a ancoragem e o posicionamento no canto superior esquerdo da tela para o texto da pontuação (com as teclas Shift e Alt pressionadas).



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Repare na **Figura 13** que, após selecionar essa opção, o texto será movido para o canto superior esquerdo da tela, assim como o seu pivô (representado por um ícone em formato de alvo). Assim, embora a tela tenha uma largura menor, o texto não deixará de ser exibido, pois ele continuará ancorado nesse canto.

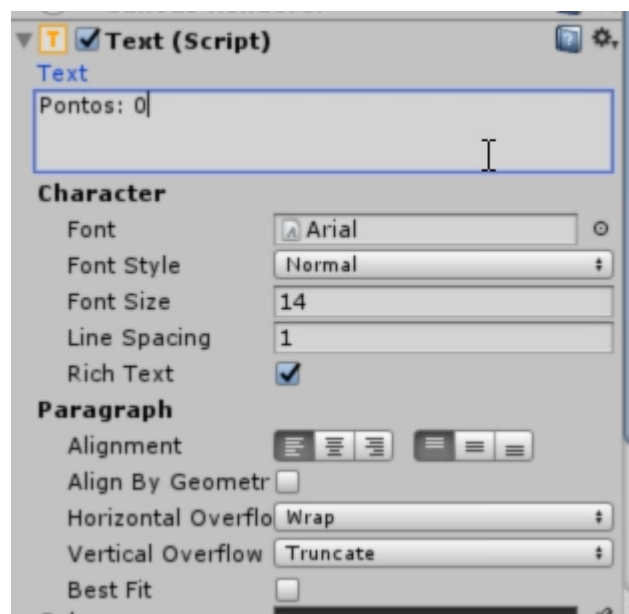
Figura 13 - Texto ancorado e movido para o canto superior esquerdo do Canvas.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

O nosso objeto do tipo Text está com o texto “New Text” devido à configuração padrão. No entanto, podemos mudá-lo no editor do Unity apenas selecionando o objeto Text e localizando a seção “Text (Script)” no Inspector. Dentro dela, existe uma propriedade chamada Text, na qual você pode digitar o texto padrão. Troque o texto para “Pontos: 0”, como mostra a **Figura 14**.

Figura 14 - Texto padrão do elemento Text modificado.



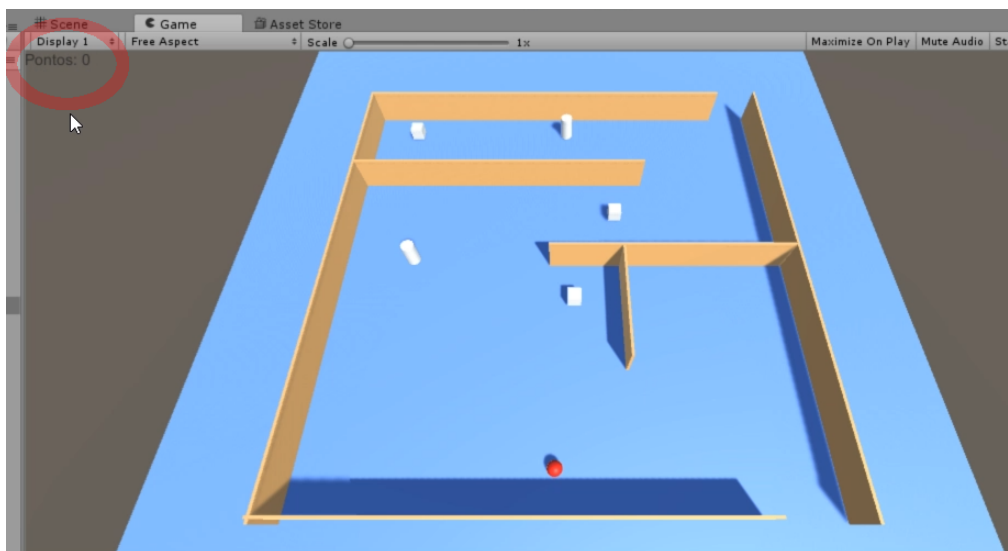
Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Nessa seção, você pode fazer várias outras mudanças no texto, tais como alterar sua fonte, mudar o seu tamanho, cor, alinhamentos, etc. Mas, por enquanto, manteremos essas propriedades como estão, para iniciarmos o jogo e verificarmos a

exibição da pontuação.

Veja, na **Figura 15**, que o texto “Pontos: 0” ficou, como havíamos planejado, no canto superior esquerdo da tela.

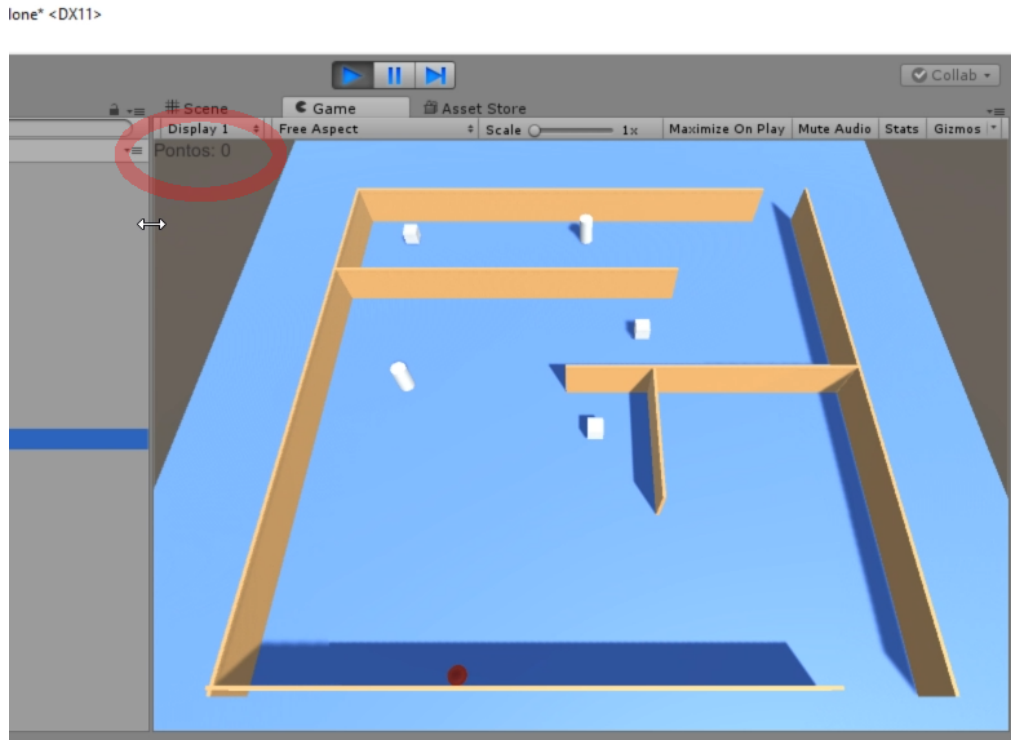
Figura 15 - Texto que exibirá a pontuação no canto superior esquerdo da tela.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Testaremos a ancoragem diminuindo a largura da janela “Game” enquanto o jogo está rodando e, então, veremos o que acontece. Para isso, inicialmente diminua a largura da janela até que ela fique aproximadamente com um aspecto quadrado, como na **Figura 16**.

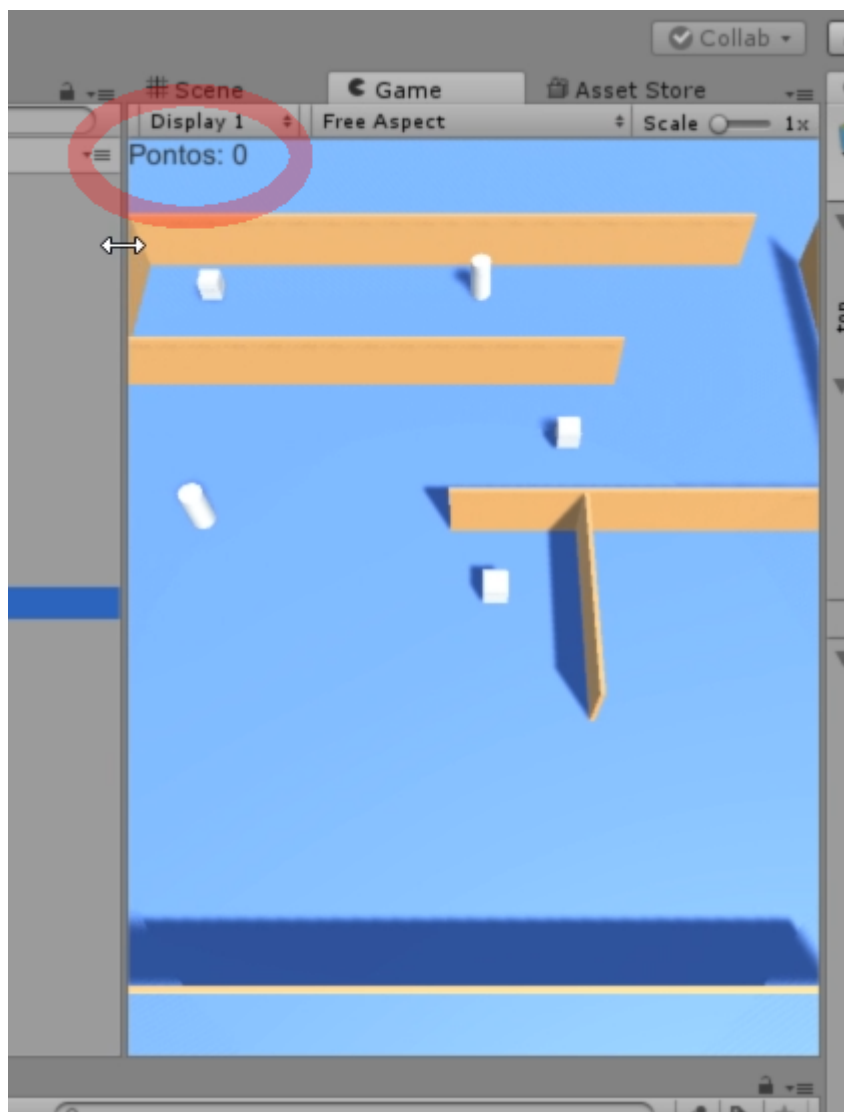
Figura 16 - Ancoragem do texto da pontuação em uma janela com um aspecto quadrado.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Repare que o texto “acompanha” a dimensão da tela, mantendo-se no canto superior esquerdo. Então, diminua mais ainda a largura da janela Game para ela ficar bem estreita em relação à altura, similar à tela de um smartphone. Veja o resultado na **Figura 17**.

Figura 17 - Ancoragem do texto da pontuação em uma janela similar à de um smartphone.



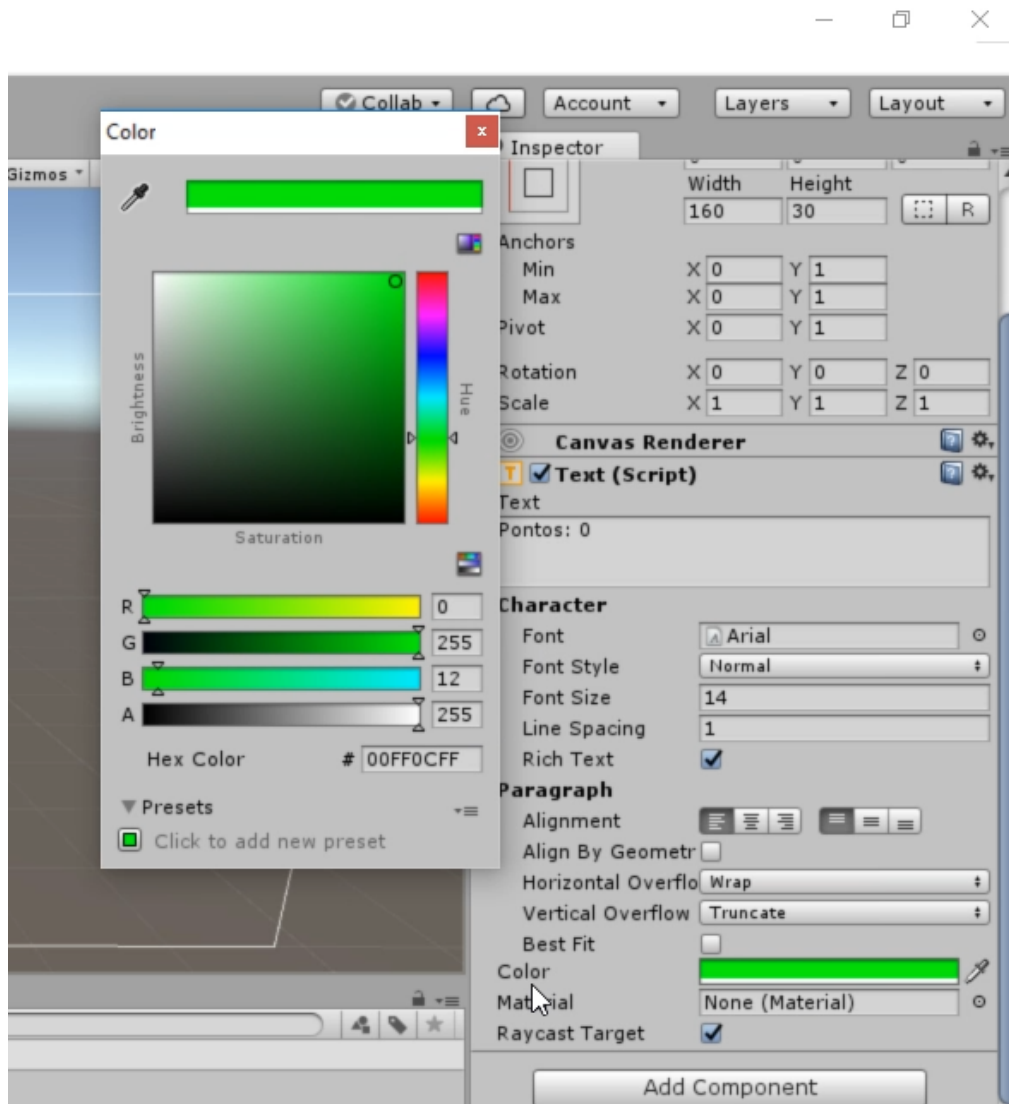
Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Mesmo com a tela bem estreita, o texto continua no canto que configuramos. Ótimo! Exatamente como deveria ocorrer. Agora, volte o tamanho da janela Game para que ela ocupe um espaço maior da tela, como estava originalmente, a fim de podermos trabalhar com mais tranquilidade.

Não estamos trabalhando o acabamento visual do nosso jogo detalhadamente, uma vez que ele servirá mais para aprendermos as primitivas do Unity. Porém, não custa fazermos pequenos ajustes, não é mesmo?

Mudaremos, então, a cor do nosso texto para verde. Clique no texto no Hierarchy, vá ao Inspectore troque, na seção Paragraph, a propriedade Color para uma cor verde, conforme a **Figura 18**.

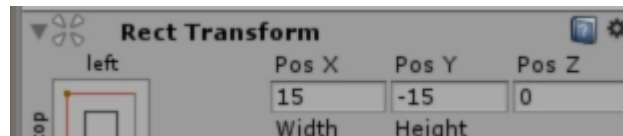
Figura 18 - Mudança de cor do Text do HUD.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Ainda no Inspector do Text, localize a seção Rect Transform e modifique as propriedades Pos X = 15 e Pos Y = -15, como na **Figura 19**. Isso fará com que o texto não fique tão encostado na borda da tela, dando um espaço de 15 pontos da lateral e de -15 da parte superior (esse número é negativo, pois queremos movê-lo 15 pontos para baixo).

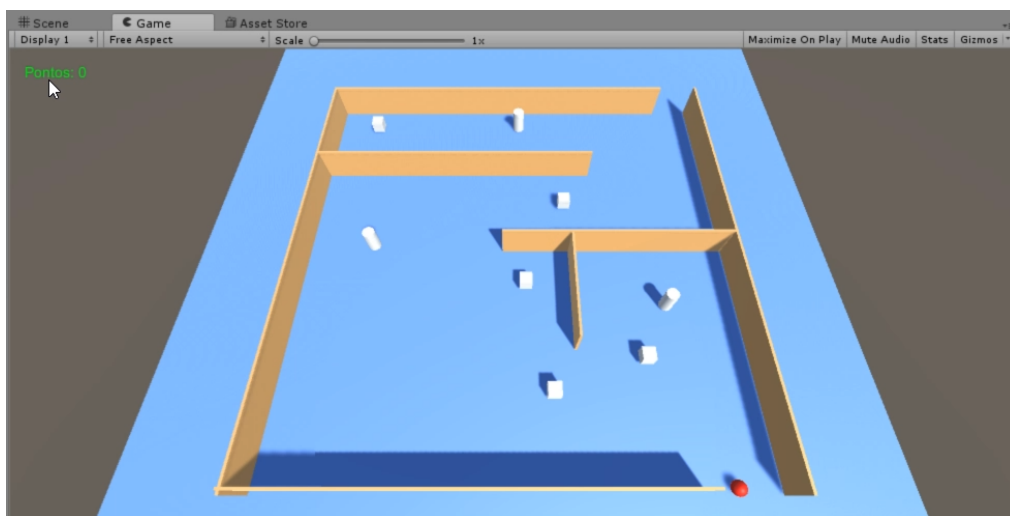
Figura 19 - Mudança do Pos X e Pos Y do Rect Transform do Text.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Inicie o jogo novamente e veja como ficou o resultado. Na **Figura 20**, você pode verificar o resultado final dessas mudanças.

Figura 20 - Texto da pontuação em cor verde e com uma margem de 15 pontos para o canto esquerdo e superior da tela.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

A criação de **HUDs** (usando o **Canvas** e outros objetos filhos) bem elaborados exige bastante prática, mas o Unity oferece muitos recursos para te ajudar. Nas próximas aulas, veremos mais sobre essas funcionalidades. Nesse momento, aprenderemos a mudar o valor dessa pontuação através de scripts, para ela refletir a pontuação real que está em uma variável no script Coletaltem do personagem.

Modificando o Texto da Pontuação

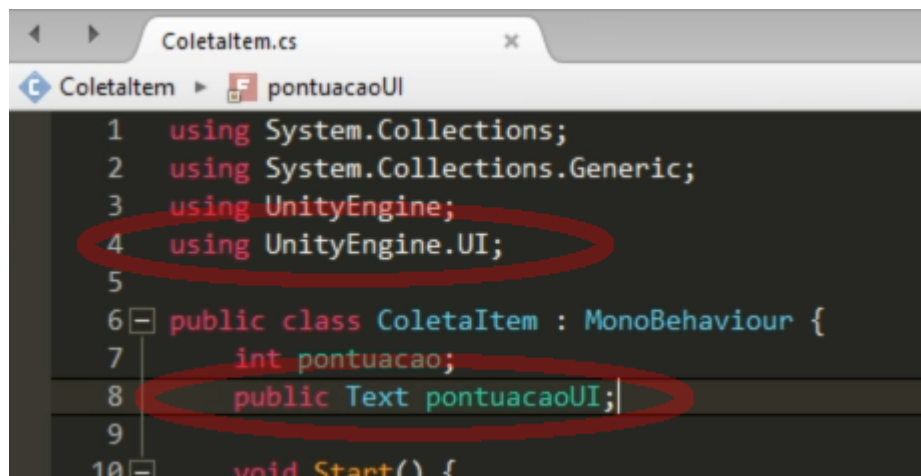
Estamos próximos de exibir nossa pontuação real no nosso **HUD** usando o Text criado. Se você iniciar o jogo agora, verá que ele resgata os polígonos e saberá ser incrementada, a cada vez que isso ocorre, a variável pontuacao do script Coletaltem

em 1. Precisamos, agora, modificar também o texto “Pontos: 0” para exibir a **pontuação** correta quando um novo polígono for resgatado.

Primeiramente, edite o script ColetaItem no MonoDevelop. Precisamos criar uma nova variável para guardar uma referência ao Text que está exibindo a pontuação do jogador. Essa variável será utilizada para manipular esse texto através do script, permitindo modificar o texto com facilidade.

A variável que precisamos criar é do tipo Text, o mesmo do objeto colocado no **Canvas** para exibir a pontuação. Os scripts padrão do Unity não vêm configurados para permitir a criação de variáveis do tipo Text, porém é muito fácil adicionar essa possibilidade. No início do script, logo após os comandos “using”, adicione mais um assim: **using UnityEngine.UI;**. Isso permitirá a você criar variáveis de tipos existentes no namespace **UI**, ou seja, podemos criar variáveis do tipo Text. Crie, então, uma nova variável pública, do tipo Text e chamada de pontuacaoUI. Essa variável será utilizada para armazenar uma referência ao Text criado na interface e, assim, permitirá que ele seja modificado no script. A **Figura 21** destaca as duas novas linhas adicionadas no script.

Figura 21 - Adicionado o namespace UnityEngine.UI e a variável pontuacaoUI.



```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.UI;
5
6  public class ColetaItem : MonoBehaviour {
7      int pontuacao;
8      public Text pontuacaoUI;
9
10 void Start() {
```

Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Agora, dispondo da variável que terá a referência do elemento com o texto da pontuação, vamos alterar os scripts para modificar esse texto.

Atenção!

Precisamos fazer a mudança do texto todas as vezes em que a pontuação mudar, então são em dois lugares: no método **Start**, quando ele é iniciado em zero, e no método **OnTriggerEnter**, quando um novo polígono é resgatado (coletado).

No método **Start**, logo depois de alterar o valor da variável **pontuacao**, adicione o comando:

```
1 pontuacaoUI.text = "Pontos: " + pontuacao.ToString();
```

Isso fará com que o texto na interface mude para "Pontos: XX", onde XX é o número atual de pontos que está na variável **pontuacao**. Repare que convertemos **pontuacao** para string com o método `ToString()`. Isso faz com que ela possa ser operada com outras strings com o operador `+`.

Então adicione o mesmo comando no método **OnTriggerEnter**, logo após o incremento da variável **pontuacao**. Assim, toda vez que o valor da variável **pontuacao** mudar o texto também mudará. A **Figura 22** mostra as duas linhas adicionadas no script e seu código final.

Figura 22 - Modificando o texto da pontuação com o valor real dos pontos acumulados.

```
ColetaItem.cs
OnTriggerEnter (Collider other)

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.UI;
5
6  public class ColetaItem : MonoBehaviour {
7      int pontuacao;
8      public Text pontuacaoUI;
9
10     void Start() {
11         pontuacao = 0;
12         pontuacaoUI.text = "Pontos: " + pontuacao.ToString ();
13     }
14
15     void OnTriggerEnter(Collider other) {
16         //Debug.Log ("Colisão aconteceu");
17
18         if (other.CompareTag("ItemColetavel")) {
19             //Debug.Log ("Coletavel!");
20             Destroy (other.gameObject);
21             pontuacao++;
22             pontuacaoUI.text = "Pontos: " + pontuacao.ToString ();
23         }
24     }
25 }
26
```

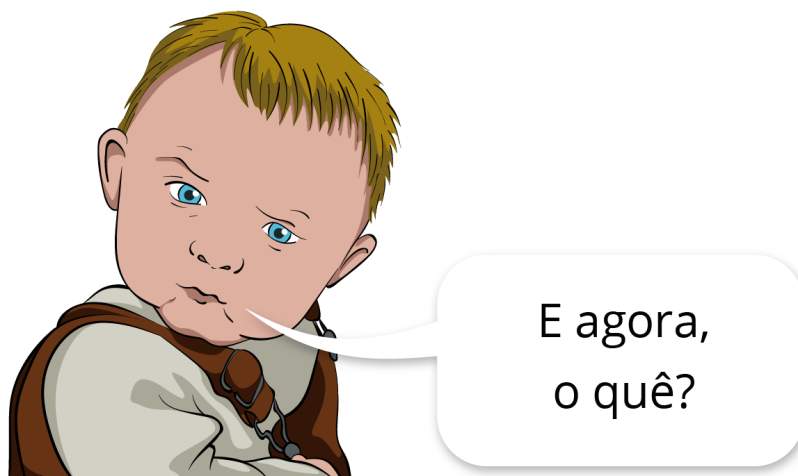
Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Congratulations, Game designer! Está praticamente tudo pronto!



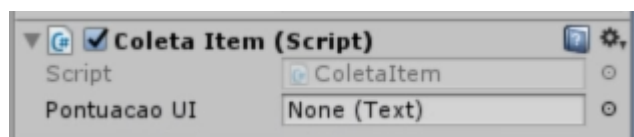
Antes de comemorar, lembre-se que falta um pequeno detalhe: a variável `pontuacaoUI` criada para armazenar uma referência ao objeto `Text` com o texto da pontuação dentro do canvas não está ainda configurada para ter essa referência.

Somente criamos a variável, falta associá-la ao elemento e isso é feito no editor do Unity.



Volte ao Unity, selecione o personagem e repare no componente com o script ColetaItem, nele existe uma propriedade chamada Pontuacao UI que está com “None (Text)” como valor. Essa é a variável criada por nós e “None (Text)” significa que ela está sem referência. Veja em detalhe na **Figura 23**.

Figura 23 - Variável pontuacaoUI sem referência.

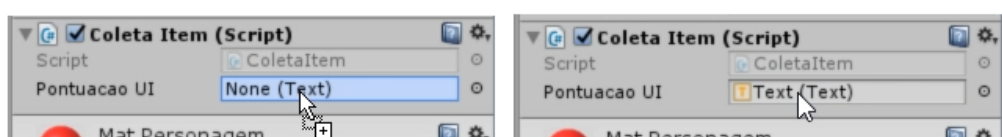


Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Para corrigir é bastante simples, mantenha o personagem selecionado para que você possa continuar visualizando o seu Inspector. Clique e arraste o Text filho do **Canvas** (o que tem o nosso texto de pontuação) para cima do valor “None (Text)” da propriedade Pontuacao UI no script ColetaItem.

A **Figura 24** mostra a mudança do valor “None (Text)” para “Text (Text)” indicando que atribuímos a referência com sucesso.

Figura 24 - Adicionando a referência do texto de pontuação à variável pontuacaoUI.

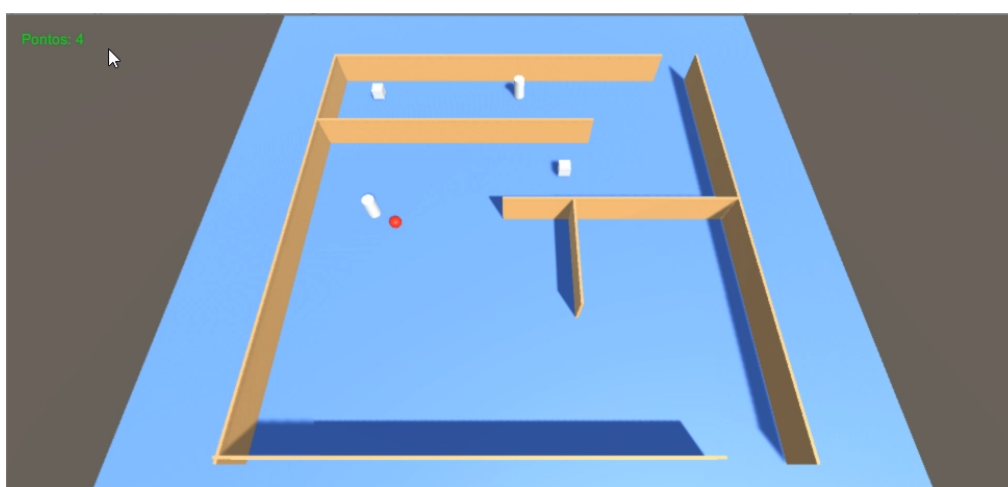


Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Agora está tudo ok! A variável `pontuacaoUI` está corretamente referenciando o `GameObject` do tipo `Text` na interface e as mudanças do seu texto no script afetarão diretamente o que está sendo exibido na tela. Vamos testar iniciando o jogo e resgatando alguns polígonos.

Veja na **Figura 25** que o jogo agora está exibindo corretamente a pontuação no canto superior esquerdo da tela e ela se modifica assim que mais um polígono é resgatado.

Figura 25 - Pontuação mudando corretamente cada vez que um novo polígono é resgatado.



Fonte: Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 10 de março de 2017.

Isso aí, muito bem! O nosso jogo agora tem o recurso de acumular e exibir a pontuação do personagem corretamente, exibindo o total de polígonos resgatados do labirinto.

Esta aula chegou ao fim e espero que tenham gostado dos aprendizados de hoje e do importante passo que demos em relação ao desenvolvimento do nosso jogo. Lembre-se que qualquer dúvida pode ser esclarecida nos encontros presenciais ou no Moodle. Na próxima aula, aprenderemos a criar objetos reutilizáveis no Unity usando Prefabs. Até lá!

Resumo

Nesta aula aprendemos a criar o HUD do nosso jogo *Polygonal Rescue* utilizando o Canvas e os elementos de interface do usuário (UI) necessários para exibir a pontuação atual do jogador em cada fase. Criamos um script que atualiza a pontuação do jogador em uma variável quando ele resgata um novo polígono. Esse mesmo script que muda o valor do elemento do Canvas, também exibe a pontuação do jogador na tela, em tempo real, quando os polígonos são resgatados no andamento da partida. Pela observação dos aspectos estudados, somos levados a acreditar que demos um importante passo no desenvolvimento do jogo.

Leitura Complementar

A criação de interfaces para HUDs em jogos no unity e o uso dos elementos de UI para isso é bastante extenso, então recomendo que você leia a documentação do Unity para mais informações. Seguem alguns links importantes.

- <https://unity3d.com/pt/learn/tutorials/topics/user-interface-ui>
- <https://docs.unity3d.com/Manual/UISystem.html>

Autoavaliação

1. O que você entende por um GameObject ser filho de outro?
2. Como o elemento Canvas se adequa à tela do dispositivo que o jogo está executando?
3. Como a âncora afeta elementos dentro do Canvas em um HUD de um jogo?

Referências

UNITY TECHNOLOGIES. 2016 (C). Unity 3D Online Tutorials [online]. Disponível em: <https://unity3d.com/pt/learn/tutorials> [Acessado em 16 de novembro de 2016].

UNITY TECHNOLOGIES. 2016 (C). Unity Manual - UI [online]. Disponível em: <https://docs.unity3d.com/Manual/UISystem.html> [Acessado em 16 de novembro de 2016].

K. Aava Rani. 2014. Learning Unity Physics. Packt Publishing

STOY, C. 2006. Game object component system. In Game Programming Gems 6, Charles River Media, M. Dickheiser, Ed., Páginas 393 a 403.

MARQUES, Paulo; PEDROSO, Hernâni - C# 2.0 . Lisboa: FCA, 2005. ISBN 978-972-722 208-8

UNITY TECHNOLOGIES. 2016 (C). Unity 3D Manual [online]. Disponível em: <https://docs.unity3d.com/Manual/index.html> [Acessado em 16 de novembro de 2016].