Curso de Pós-Graduação em Cloud Computing e Mobile
Disciplina DM 117 – Introdução a Desenvolvimento de Jogos com Unity
Professor: Phyllipe Lima

Aula 3

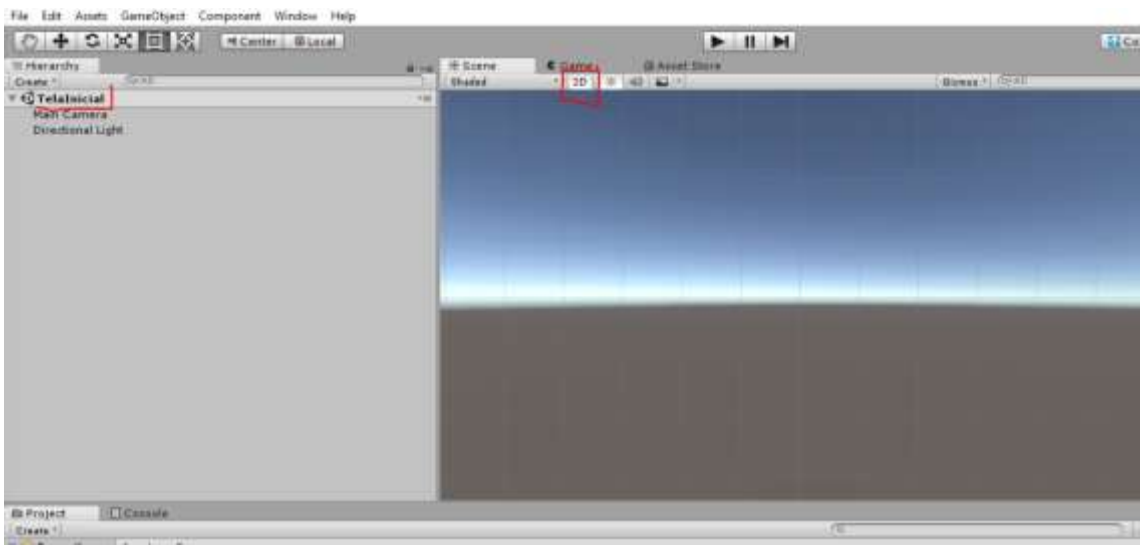
Ao final desse relatório você terá:

- Aprendido a usar elementos de UI

1 – Criando uma tela inicial com o título do jogo

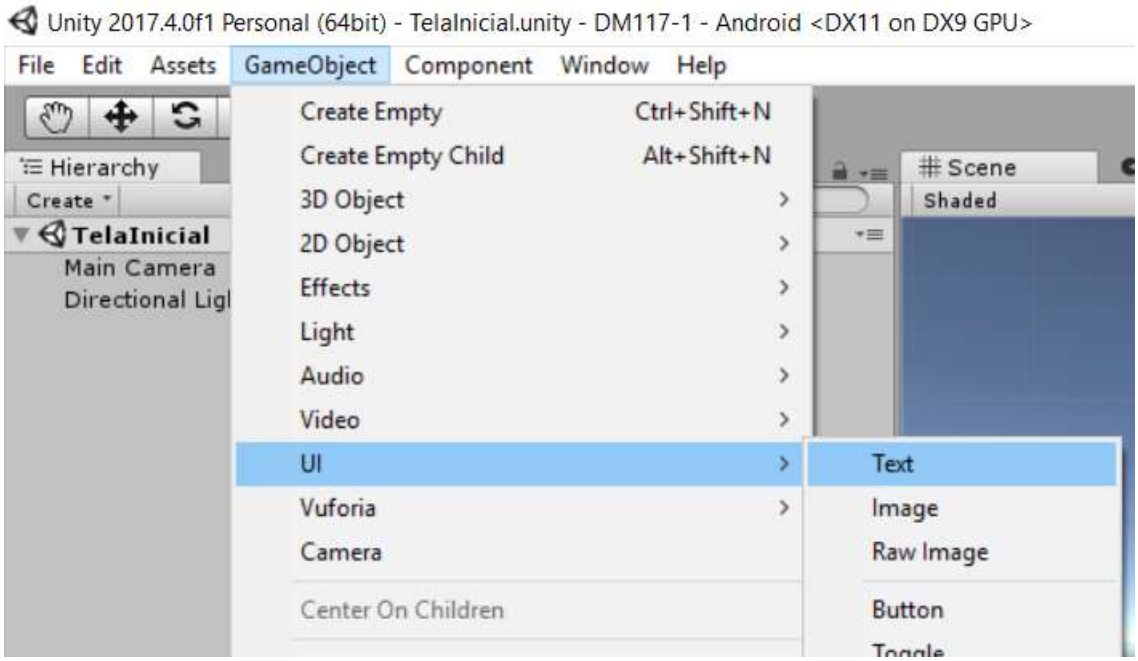
Vamos começar este relatório criando uma tela com o título de jogo. Comece criando uma Scene. **File -> New Scene.** Salve na pasta Scenes e nomeie para TelaInicial.

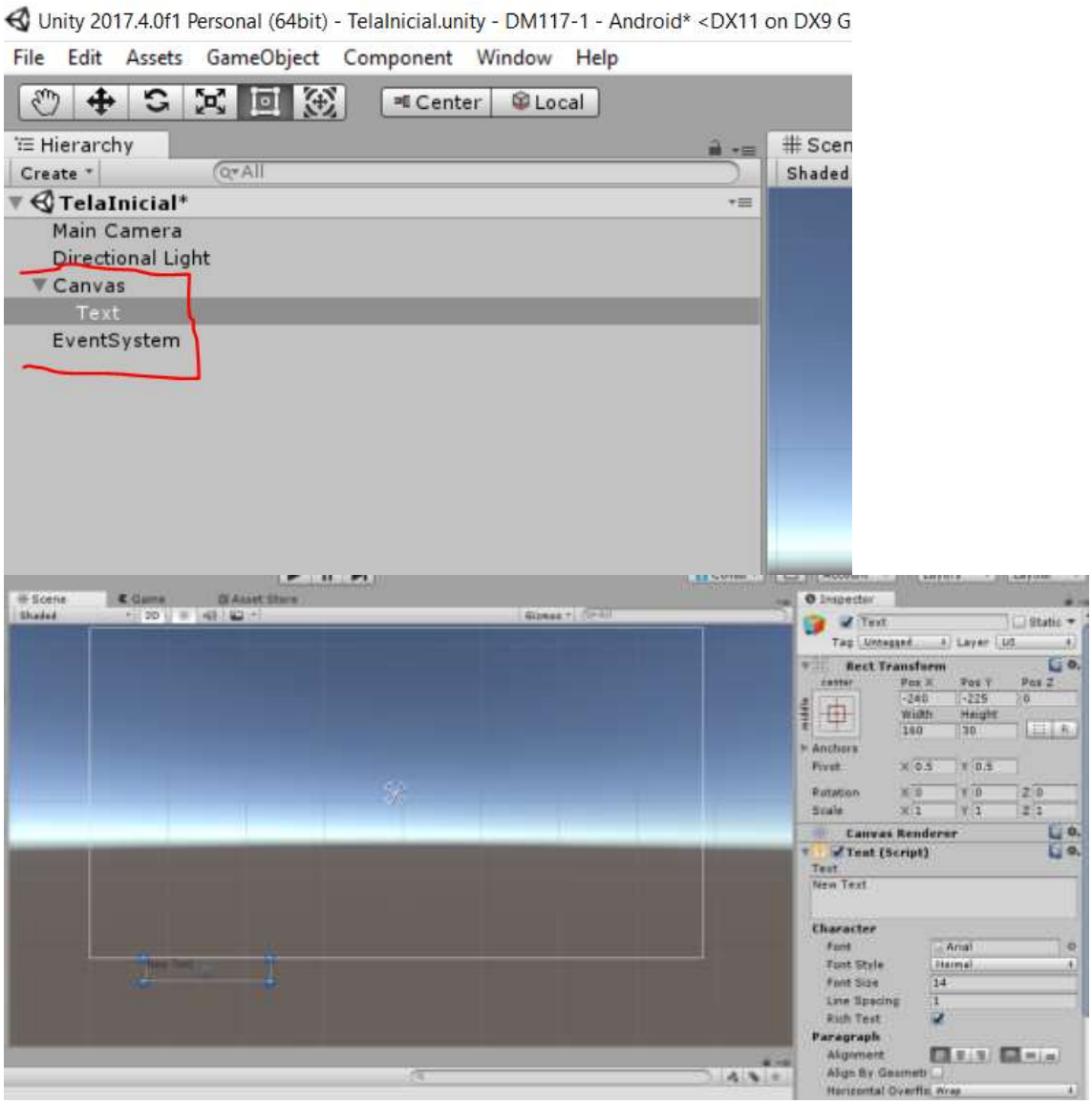
Muitas vezes ao lidar com elementos UI, é mais interessante fazermos isso com a tela em modo 2D. Vá na aba Scene e na barra horizontal clique em 2D.



Observe como fica a sua Scene. Note que não existe mais o gizmo com as coordenadas. Agora não é mais necessário pois na horizontal deslocamos em X e na vertical deslocamos em Y. A câmera se encontra no modo ortográfico, onde antes estava no modo de perspectiva.

O primeiro elemento UI que criaremos é o Text. **GameObject -> UI -> Text**
Isso irá criar três elementos, visualizados na aba Hierarchy.


Unity 2017.4.0f1 Personal (64bit) - TelaInicial.unity - DM117-1 - Android <DX11 on DX9 GPU>
File Edit Assets GameObject Component Window Help
Create Empty Ctrl+Shift+N
Create Empty Child Alt+Shift+N
3D Object
2D Object
Effects
Light
Audio
Video
UI
Vuforia
Camera
Center On Children
Text
Image
Raw Image
Button
Hierarchy
Create
TelaInicial
Main Camera
Scene
Shaded


Unity 2017.4.0f1 Personal (64bit) - TelaInicial.unity - DM117-1 - Android* <DX11 on DX9 G
File Edit Assets GameObject Component Window Help
Center
Local
Hierarchy
Create
All
TelaInicial*
Main Camera
Directional Light
Canvas
Text
EventSystem
Shaded
Inspector
Rect Transform
Canvas Renderer
Text (Script)
New Text
Character
Paragraph

Vamos analisar esses GOs:

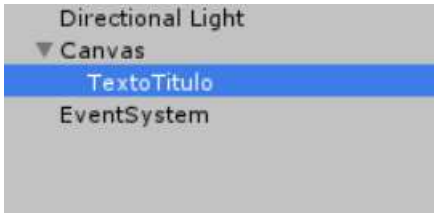
Canvas: Essa palavra se traduz para “tela”, e é usada no mesmo contexto da tela de um pintor. É neste Canvas que todos os nossos elementos de UI ficarão. Se tentar criar um UI em uma Scene sem o Canvas, o próprio Unity criará um (igual aconteceu agora). Na Scene o Canvas é representado pelo retângulo branco que irá alterar seu tamanho dependendo da aba Game.

Esse GO contém um componente chamado Canvas, que é responsável por renderizar a imagem. O componente Canvas Scaler é responsável pelo escalonamento da arte dependendo da resolução do dispositivo. E o componente Graphic Raycaster determinar se algum GO no Canvas foi atingido.

Text: Esse é de fato um GO do tipo UI. Ele representa um texto e possui as propriedades de posicionamento, cor, fonte, tamanho e etc.

EventSystem: Esse GO permite usuários enviar eventos a GOs no jogo vindo de uma série de tipos diferentes de entrada como mouse, teclado, controle e touch. Esse GO tem propriedades que permitem especificar como o usuário irá interagir com a UI. Assim como o Canvas, se tentar criar algum elemento UI sem EventSystem o próprio Unity o fará.

Se você não estiver vendo o GO Text na sua Scene de um clique duplo na aba Hierarchy para que ele fique em foco. Para melhorar a organização, altere o nome desse GO para TextoTitulo.

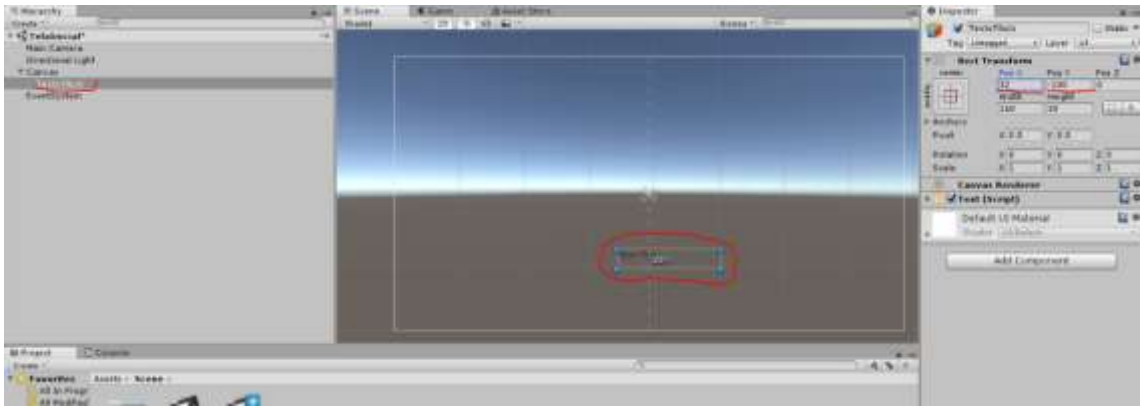


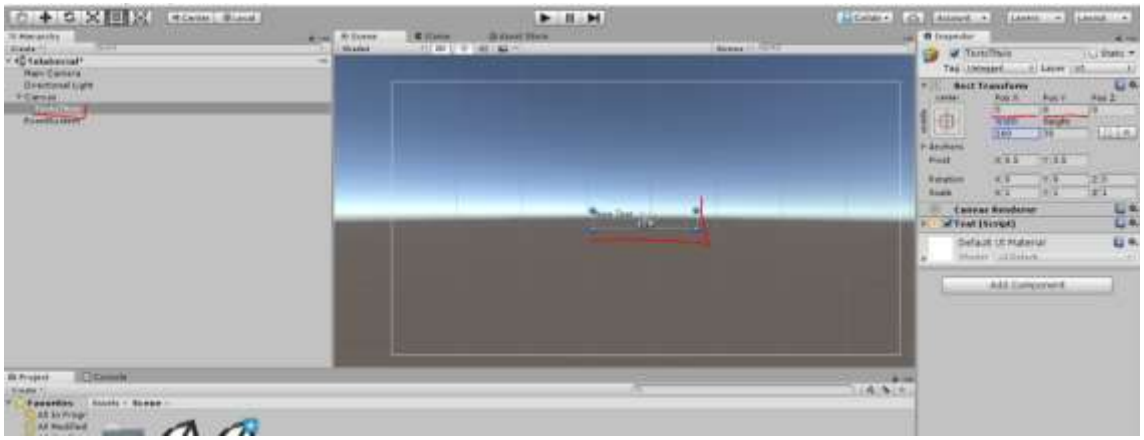
Como esse GO Text não está dentro do Canvas na Scene (retângulo branco), ele não será exibido dentro do jogo. Sendo um GO podemos movimentá-lo, porém para elementos UI não temos o componente Transform e sim o Rect Transform.

Rect Transform (RT): Enquanto Transform representa um ponto no mundo do jogo, o Rect Transform representa um retângulo, onde o UI irá residir. Se o elemento UI for filho (no nosso caso o TextoTitulo é filho do Canvas), o pai também obrigatoriamente possui uma RT. A RT do filho define como o filho estará posicionado em relação ao pai.

Para começarmos a entender a RT, mude PosX e PosY para 0 (do TextoTitulo). Isso irá centralizar o objeto (no caso o TextoTitulo) em volta da âncora dele no pai (confuso?). De um duplo clique na Hierarchy para colocar o objeto em foco na Scene.

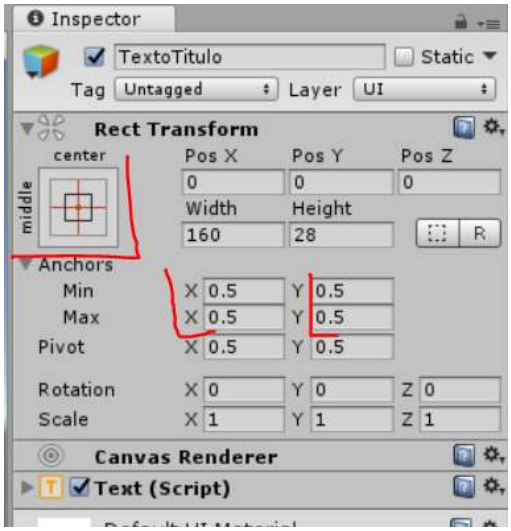
Na primeira figura mostrado abaixo, originalmente o PosX e PosY valiam 32 e -100. Na segunda figura foi alterado para 0 e 0, observe o resultado.





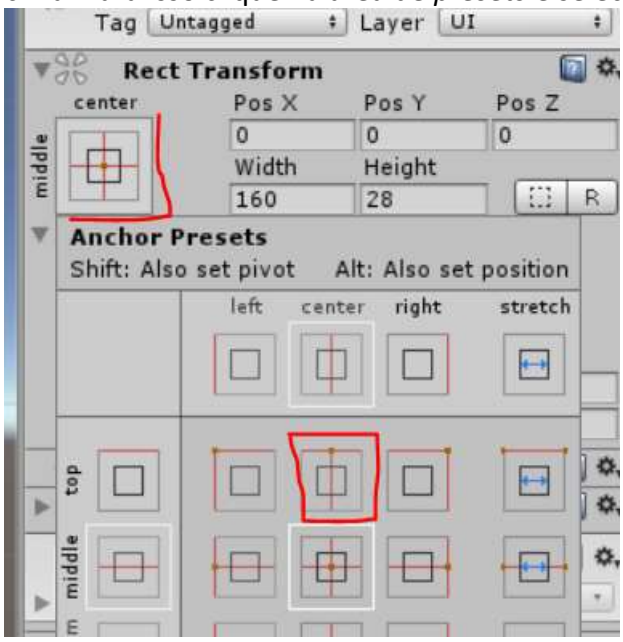
O objeto agora foi para o centro do Canvas. Mas quem definiu isso? As âncoras. Elas podem ser identificadas pelo desenho branco parecendo um X.

Âncoras: Permitem que o UI elemento fique preso a um canto ou parte do Canvas. Caso o Canvas mude de tamanho os pedaços da UI ficará em um lugar apropriado de acordo com a âncora. Você pode determinar os valores manualmente ou utilizar alguns valores predefinidos.

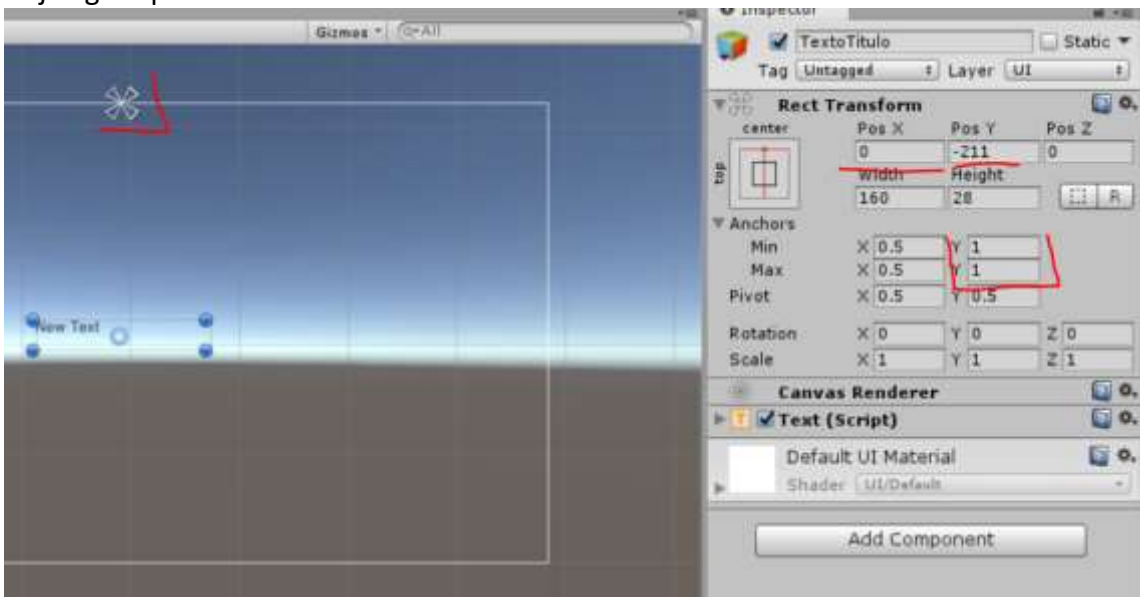


Nesse exemplo temos que a ancora do TextoTitulo está no centro do Canvas, ou seja, todo o movimento do TextoTitulo é em relação ao centro do Canvas. Vamos utilizar posições predefinidas de âncoras pois é melhor de trabalhar.

Para nosso TextoTítulo, vamos querer que a ancora fique na parte superior da tela, pois caso o Canvas se modifique queremos esse texto preso (ancorado) na parte de cima. Para isso clique na área de *presets* e selecione conforme a figura:



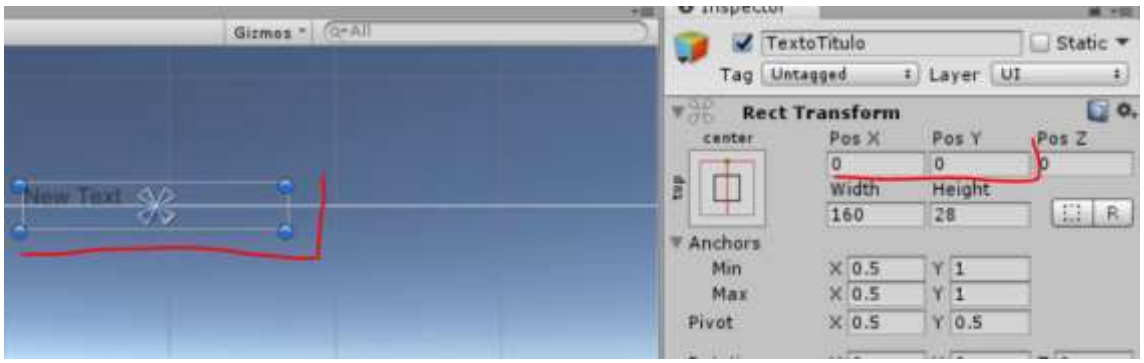
Veja agora para onde foi a ancora e observe também os valores de Pos X e Pos Y



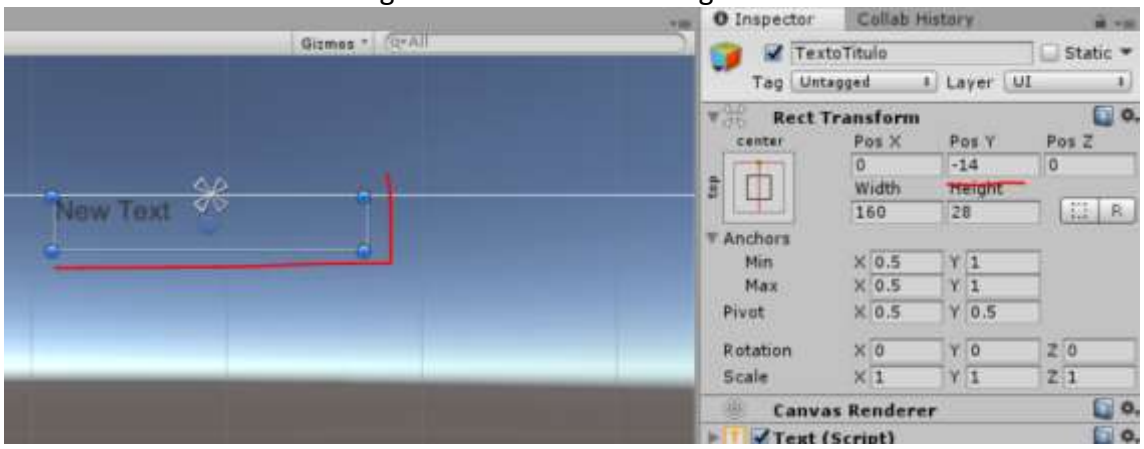
Vamos analisar o resultado. Primeiro vemos que a âncora foi para cima, e os valores Min e Max de Y alterados para 1. Nós poderíamos ter alterado diretamente esses valores de Y que a âncora teria se deslocado da mesma forma. Trabalhe da forma que mais lhe agrade.

Veja também o valor de PosY. Agora vale -211 (neste caso, talvez na sua tela esteja diferente). Isto quer dizer que a posição Y está 211 unidades abaixo da âncora. Isso faz sentido, pois agora a âncora mudou de lugar, mas o TextoTítulo não.

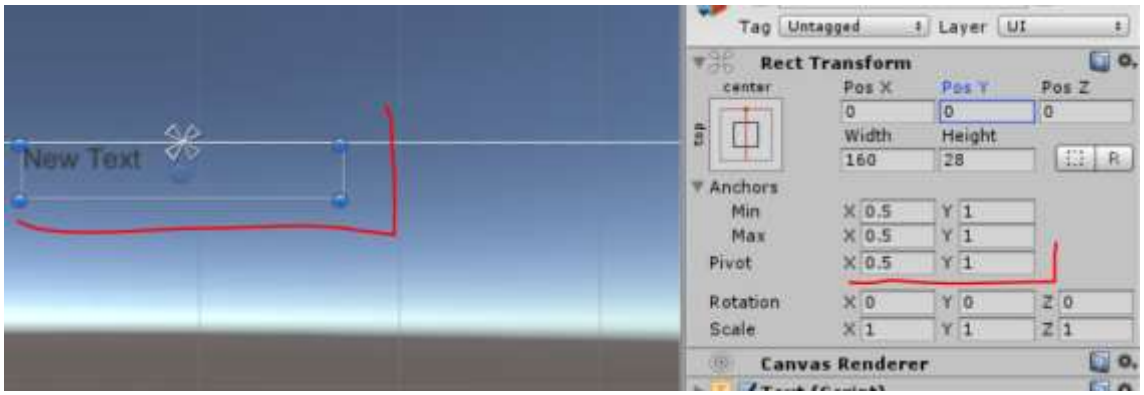
Coloque agora 0 na PosY



Agora o TextoTitulo se encontra centralizado em torno da âncora. Se modificarmos o PosY de forma manual conseguiremos coloca-lo no lugar correto. Observe

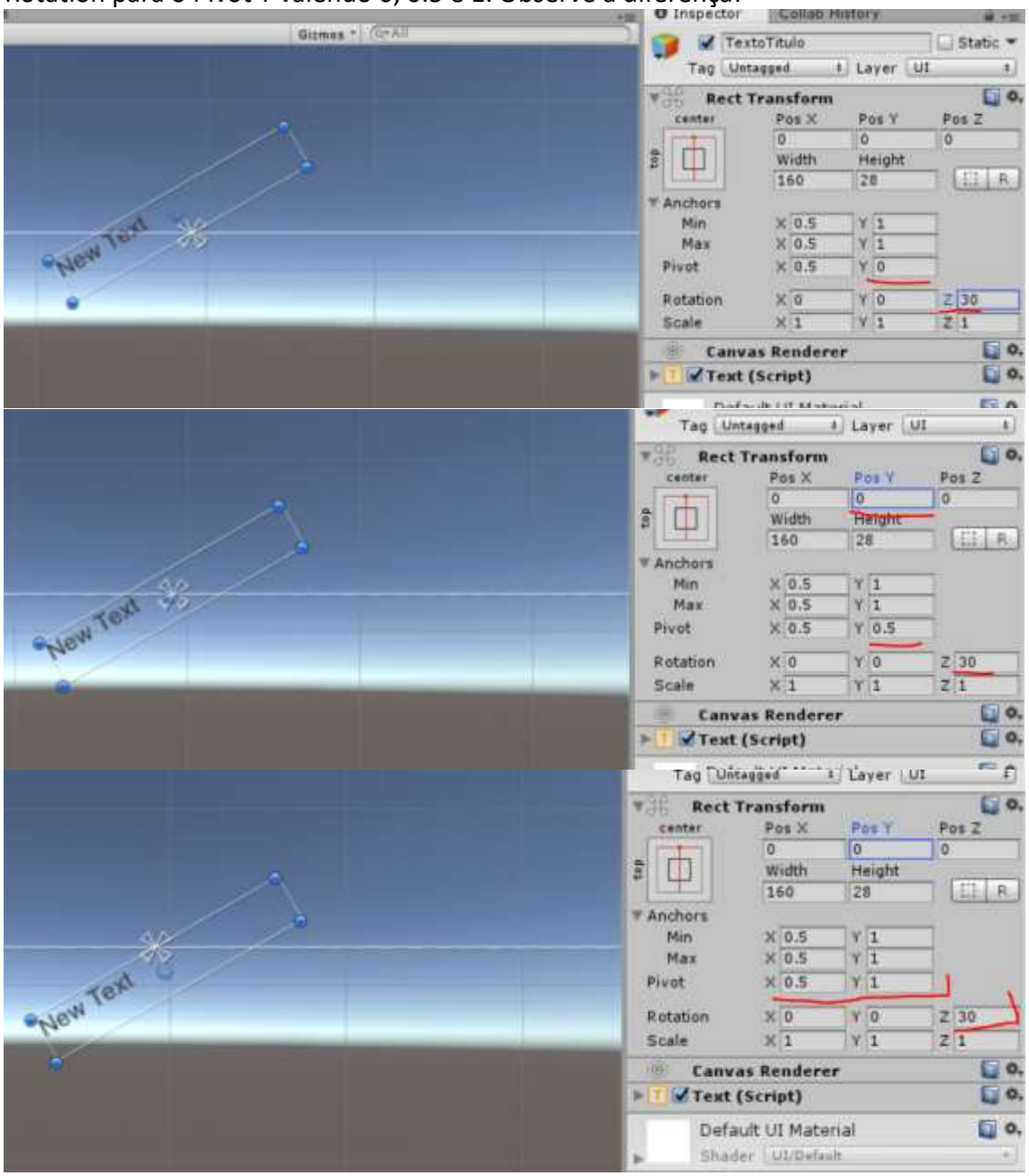


Mas essa abordagem só funciona bem se o nosso jogo for utilizado nessa única resolução (talvez no seu computador você deverá colocar outro valor). Deve ter alguma forma que faça isso automaticamente. Para isso usaremos o *pivot*.



Na figura acima coloquei o Pivot Y = 1. Antes de fecharmos a âncora quero deixar como exercício você experimentar o que acontece conforme você muda os valores Min e Max. É a única forma de você realmente compreender como ela funciona.

Pivot: Diferentemente das âncoras que fazem a relação do nosso UI com o Canvas, o Pivot faz a relação da posição em relação ao nosso próprio UI. Basicamente estamos dizendo onde está o centro do nosso UI. Para entender melhor, mude o valor da Rotation para o Pivot Y valendo 0, 0.5 e 1. Observe a diferença:

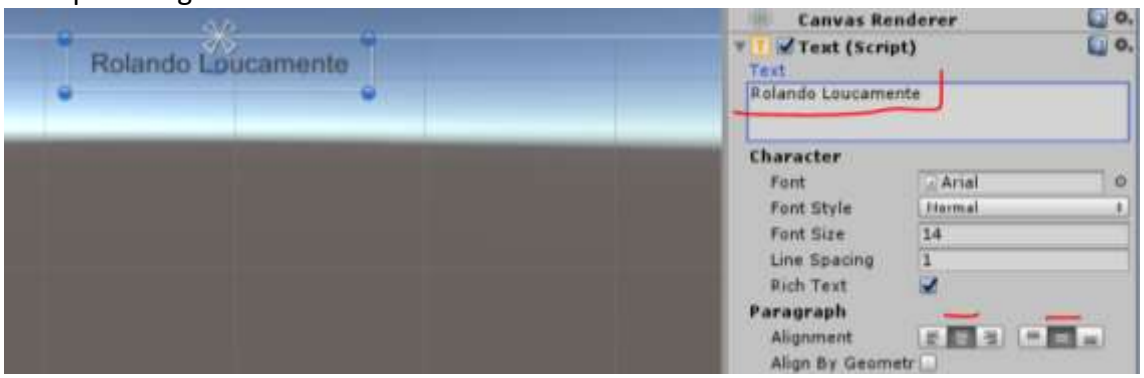


Observe que conforme o pivot foi modificado, a figura girou no eixo Z em relação ao pivot. Brinque com mais valores e observe o resultado.

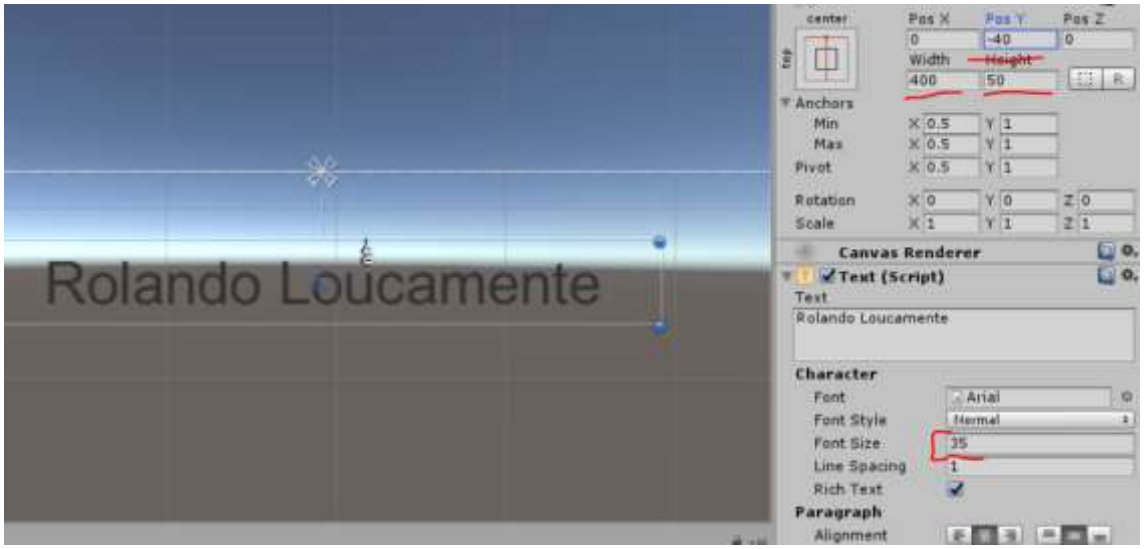
2 – Acertando o Título

Com o nosso TextoTitulo posicionado de forma correta, vamos trabalhar no conteúdo propriamente dito. Na aba Inspector, vá até o componente Text. E mude o conteúdo para Rolando Loucamente (fique à vontade para não usar esse nome horrível).

Coloque o Alignment centralizado na vertical e horizontal.



Mude o tamanho da fonte para 35. Com esse tamanho o texto não caberá no tamanho da Rect Transform, assim altere a largura para 400 e altura para 50. E dê um offset no PosY de -40.

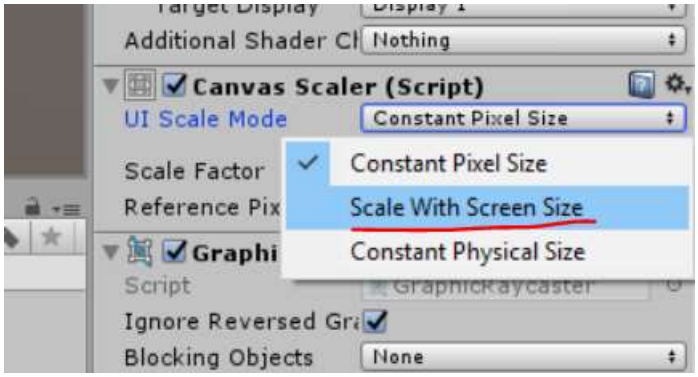


Note que esse offset foi feito *hardcoded* pois foi um ajuste fino. Vá até a aba a Game e observe como fica.

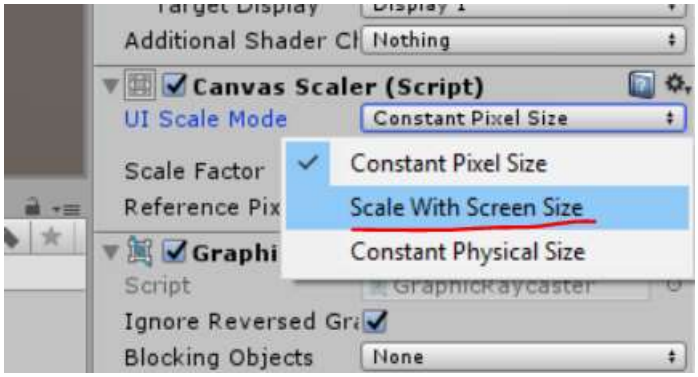
Agora maximize a aba (botão direito -> maximizar).

Não parece muito legal. Para um HUD pode até ficar bacana. Mas como é o nosso título, desejamos que pareça maior. Vamos começar o processor de adaptar nossa UI para resoluções diferentes.

Vamos começar pelo componente Canvas Scaler, do nosso Canvas. Mude o modo para Scale With Screen Size

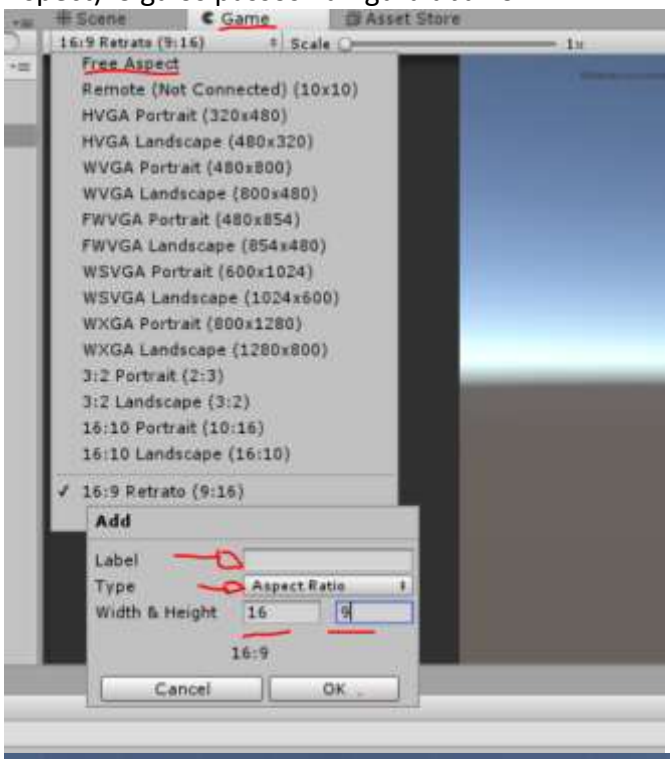


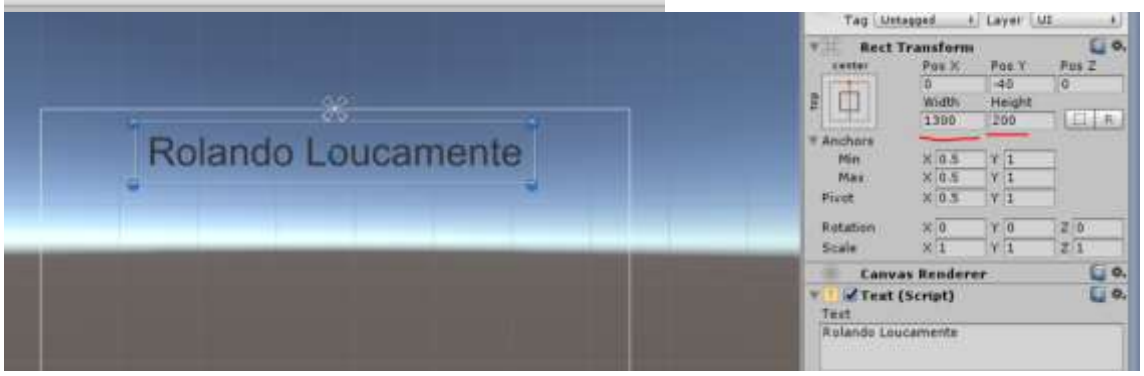
A propriedade principal é a Reference Resolution. Vamos definir essa referência, e o Unity cuidará de adaptar. Como estou pensando em um jogo no modo paisagem, vou colocar x = 1920 y = 1080. Altere também o Match para Height (isso faz com que o ajuste ocorra quando alterarmos a altura da tela). Mude a fonte para 130, a largura do RT para 1000 e altura para 350.



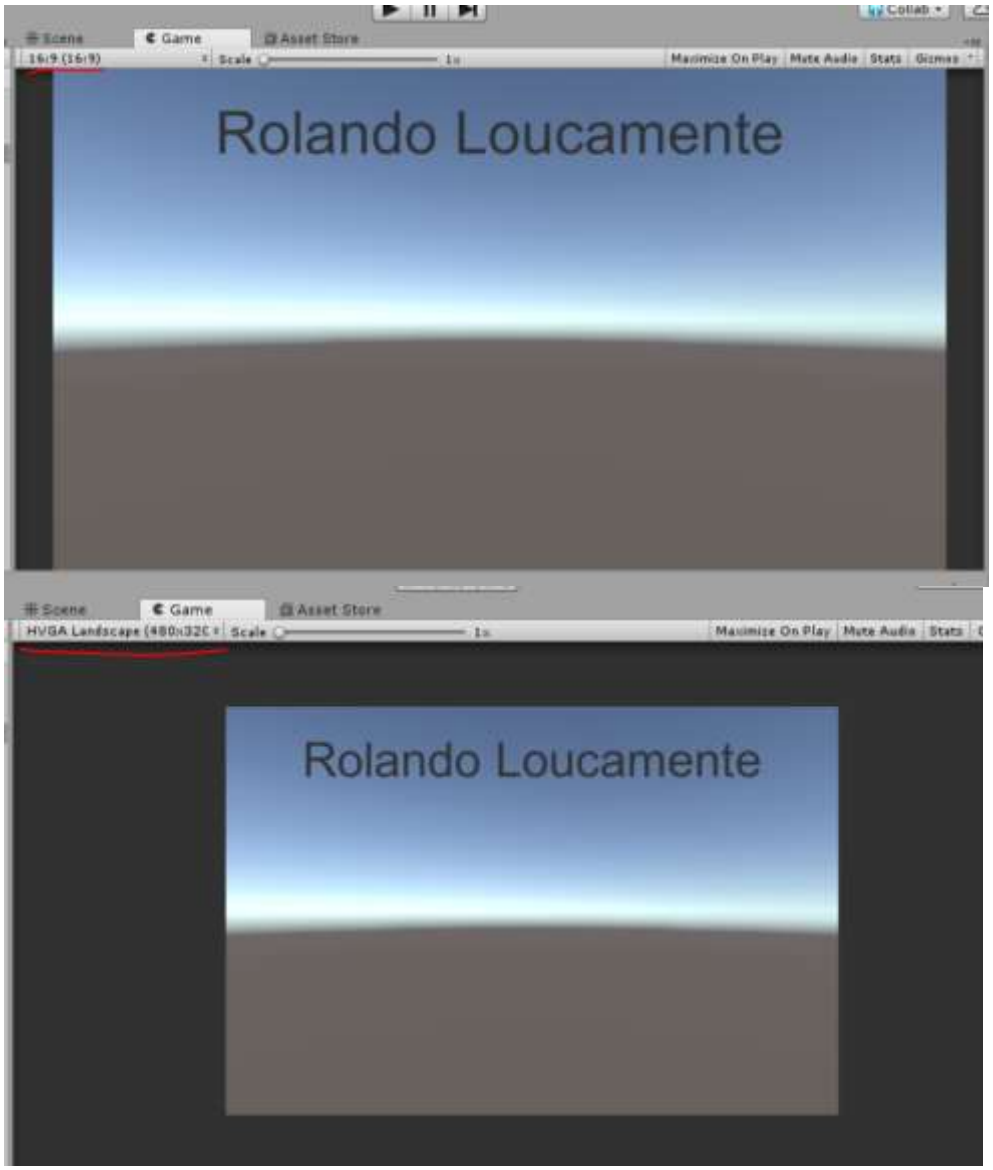
Vamos deixar a aba game já em uma resolução 16:9 no modo paisagem.

Vá na aba Games, clique no menu *dropdown* logo abaixo (deve estar escrito Free Aspect). Siga os passos na figura abaixo:





Observe na Figura abaixo o resultado em duas resoluções diferentes

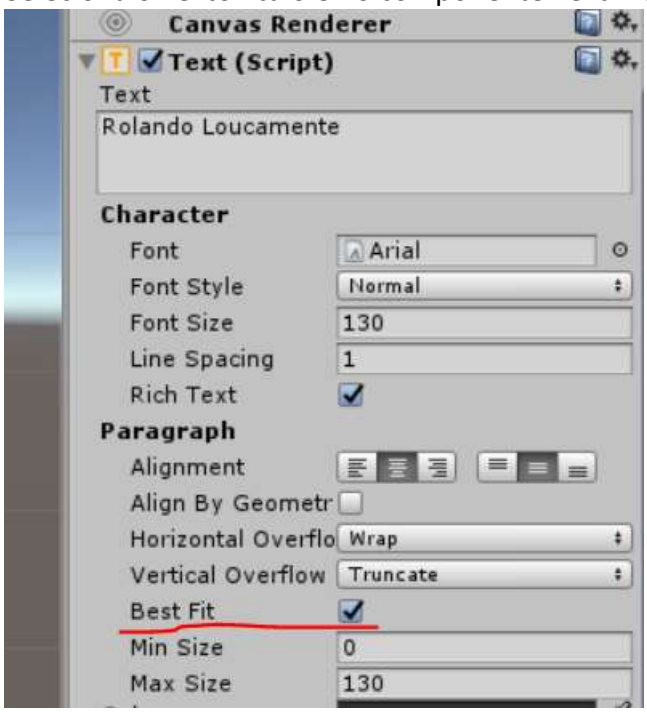


O título se adaptou bem. Mas e se mudarmos a resolução de aspecto?

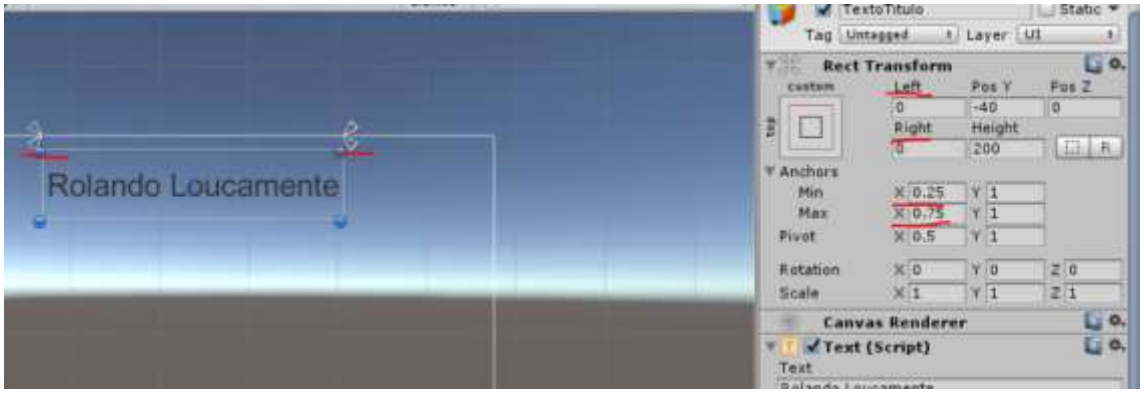
3 – Alterando a resolução de aspecto

Coloque a aba Games na em 16:10 portrait.

Eita que zica! Vamos arrumar isso.
Seleciona o TextoTitulo e no componente Text mude para Best Fit.



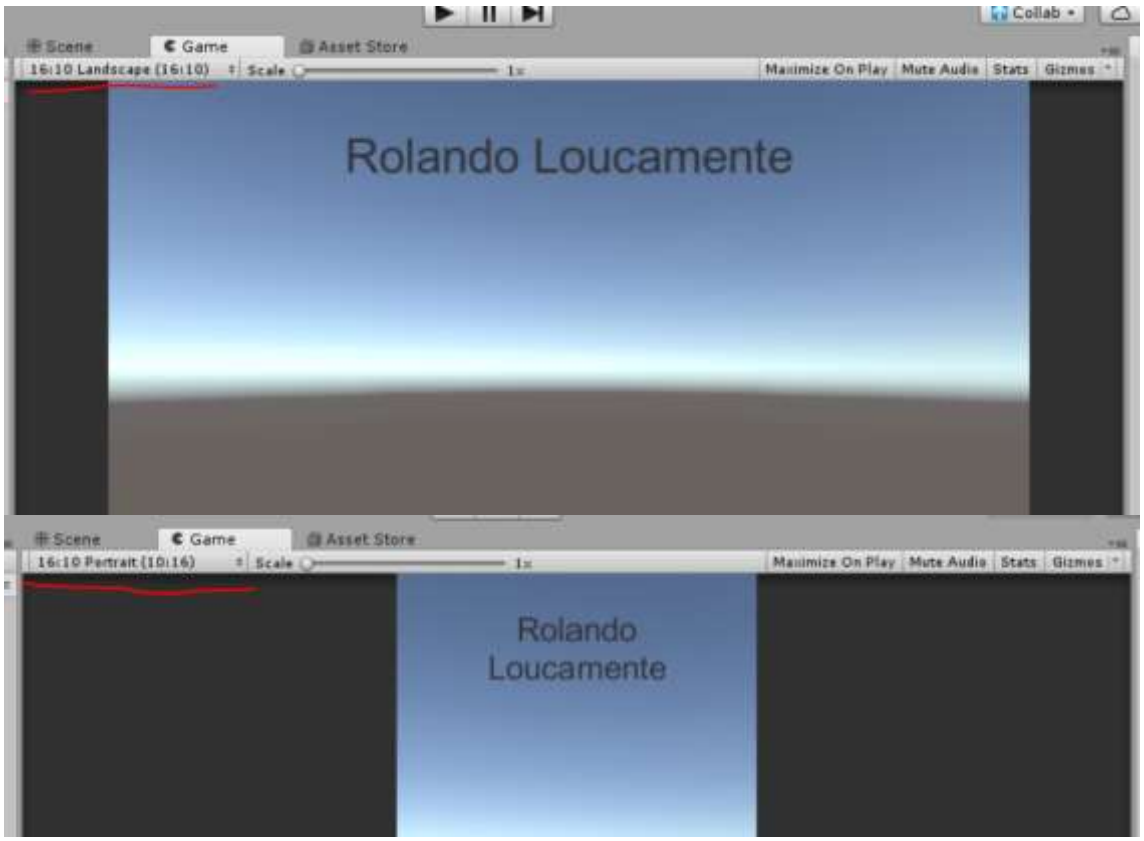
Para que isso tenha o efeito desejado, altere o RT do TextoTitulo. Coloque X min 0.25 e X max 0.75. Observe



Algumas coisas mudaram, veja a âncora. Ela se dividiu no eixo horizontal. O eixo é normalizado entre 0 e 1. Colocando o ponto mínimo com 0.25 e max com 0.75 temos o resultado acima. Repare que isso alterou uma nomenclatura do RT. Agora tem-se Left e Right. Agora você define o quanto a UI pode esticar para a esquerda e direita da âncora. Coloque como teste -300 nos dois e veja o que ocorre.

O que estamos dizendo agora é que a área ocupada por esta UI está -300 unidades de distância da âncora em 25% e 75% da tela. Nós queremos travar a UI dentro de 25% e 75%, então coloque 0 nesses valores.

Vá na aba games o coloque a resolução 16:10 e depois 10:16 e veja o resultado.

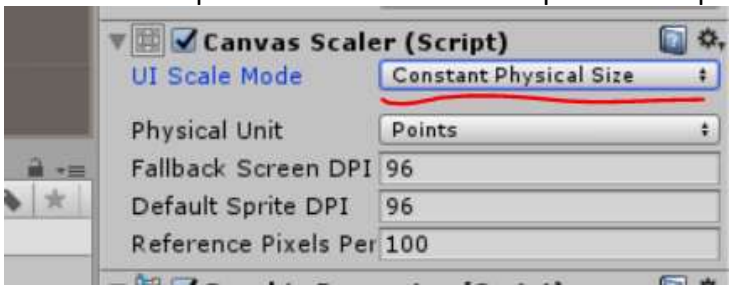


Vamos colocar alguns botões para interagirmos com nossa UI

4 – Usando botões

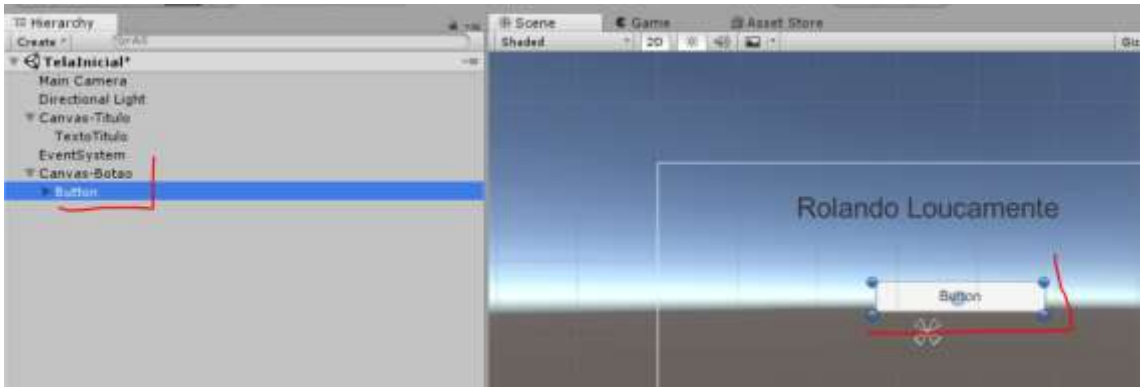
A primeira função será sairmos da tela inicial e começar o nosso jogo. Mas existe uma diferença com relação ao Canvas. No caso anterior queríamos que o elemento UI se adaptasse ao tamanho da tela. No caso do botão é um pouco diferente, pois o nosso dedo não muda de tamanho (vai que o seu muda né?). Então vamos criar um outro canvas para os botões.

Mas antes, renomeio o atual para Canvas-Titulo. Em seguida crie outro canvas
GameObject -> UI -> Canvas
Mude o nome para Canvas-Botao. E coloque o modo para Constant-Physical Size



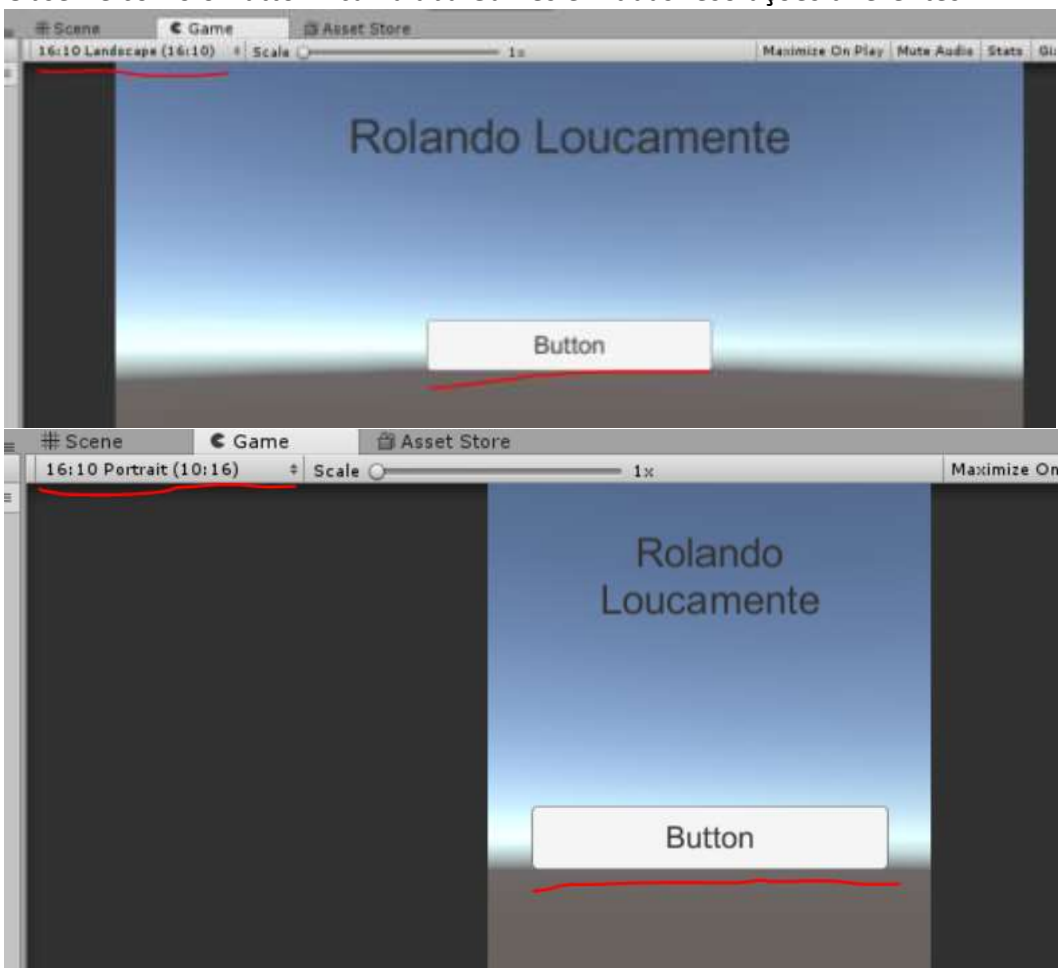
Esse modo faz com que o Unity tente adaptar o canvas de forma a todos os elementos UI terem o mesmo tamanho físico. O que faz sentido para botões.

Mantenha esse Canvas selecionado na Hierarchy e crie um botão.
GameObject -> UI -> Button



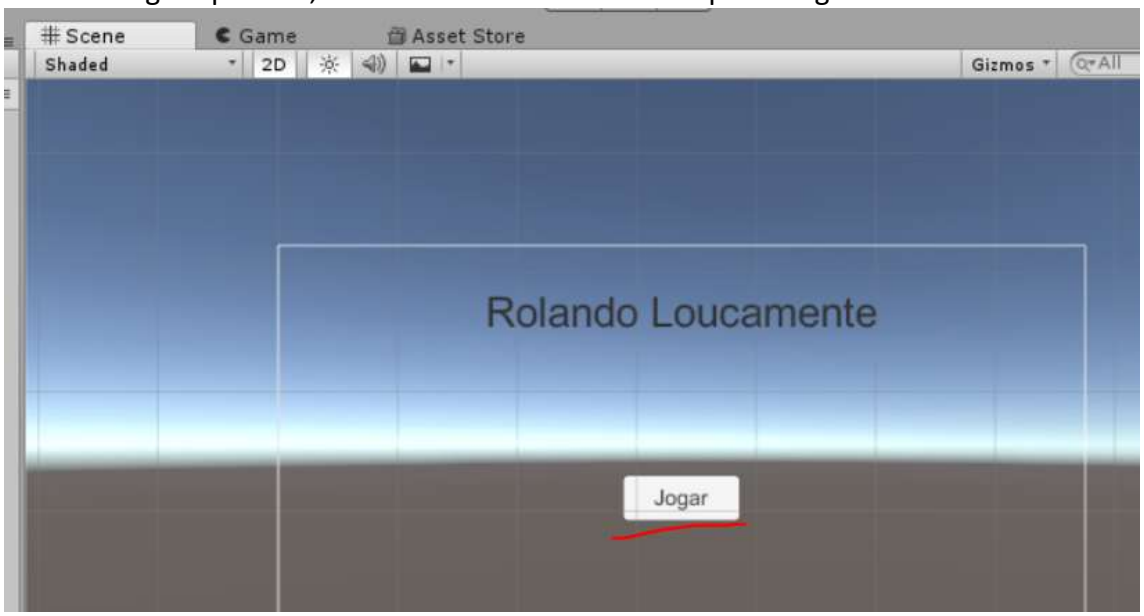
Observe o elemento UI Button. Se você o expandir verá que tem um Text como filho. Faz sentido não é mesmo? Observe os valores do RT e tente mais uma vez entender como funciona.

Observe como o Button fica na aba Games em duas resoluções diferentes.



O tamanho se mantém. Mas está muito grande. Vamos mudar isso.

Mude a largura para 75, e altere o conteúdo do Texto para “Jogar”

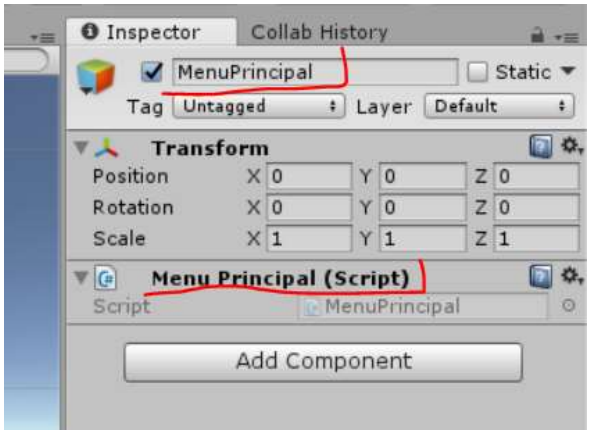


Vamos agora adicionar a funcionalidade deste botão. Para isso crie um script chamado MenuPrincipal.

```
6   public class MenuPrincipal : MonoBehaviour {
7
8       public void CarregaScene(string nomeScene) {
9           SceneManager.LoadScene(nomeScene);
10      }
11  }
12
```

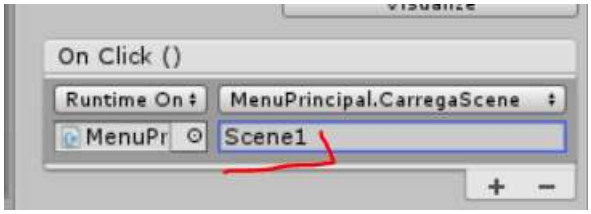
Observe que não iremos usar os métodos Start e Update, então pode remove-los. Adicione o método CarregaScene, que será invocado pelo clique do botão.

Sabemos que scripts no Unity são componentes, e componentes precisam estar atrelado a um GO. Portanto crie um GO chamado MenuPrincipal e adicione esse script como componente.

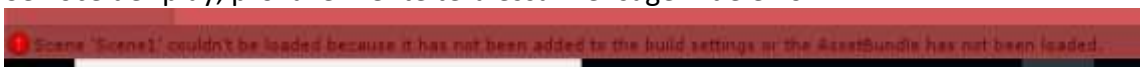


Selecione o Button na aba Hierarchy, expanda a propriedade On Click() e clique no +. Arraste o GO MenuPrincipal e procure o método CarregaScene. Veja a Figura abaixo:

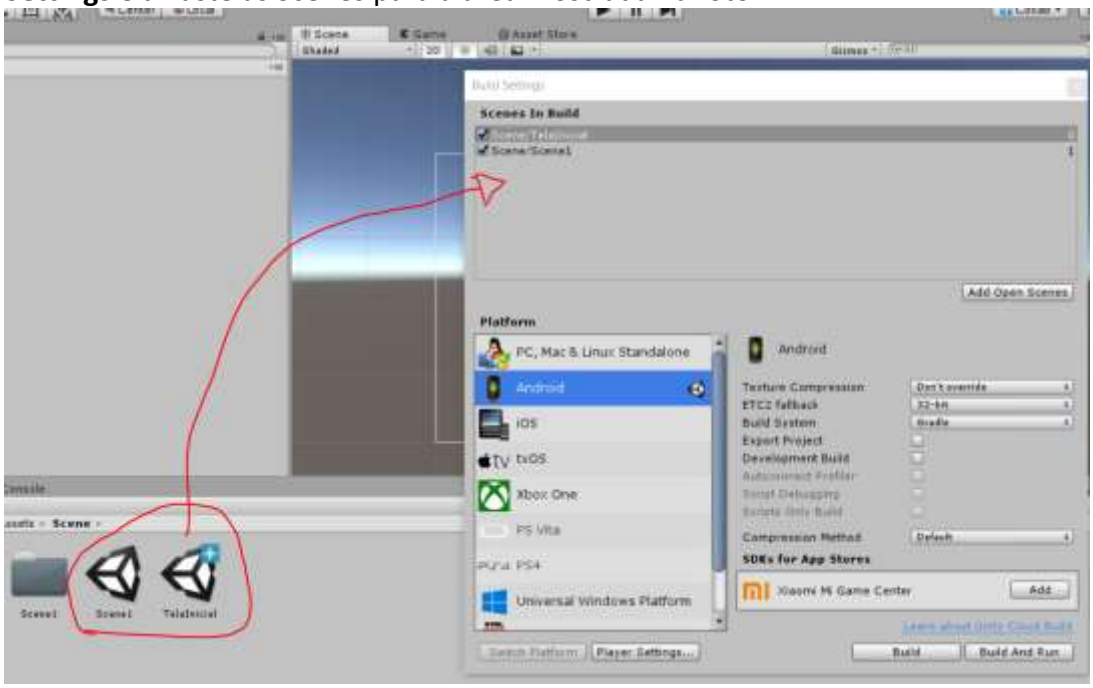
No campo escreva o nome da Scene que deseja carregar, no meu caso é Scene1 (talvez você tenha colocado outro nome).



Se você der play, provavelmente terá essa mensagem de erro.



O que acontece é que as Scenes ainda não estão no build path. Vá em **File -> Build Settings** e arraste as Scenes para a área mostrada na foto.



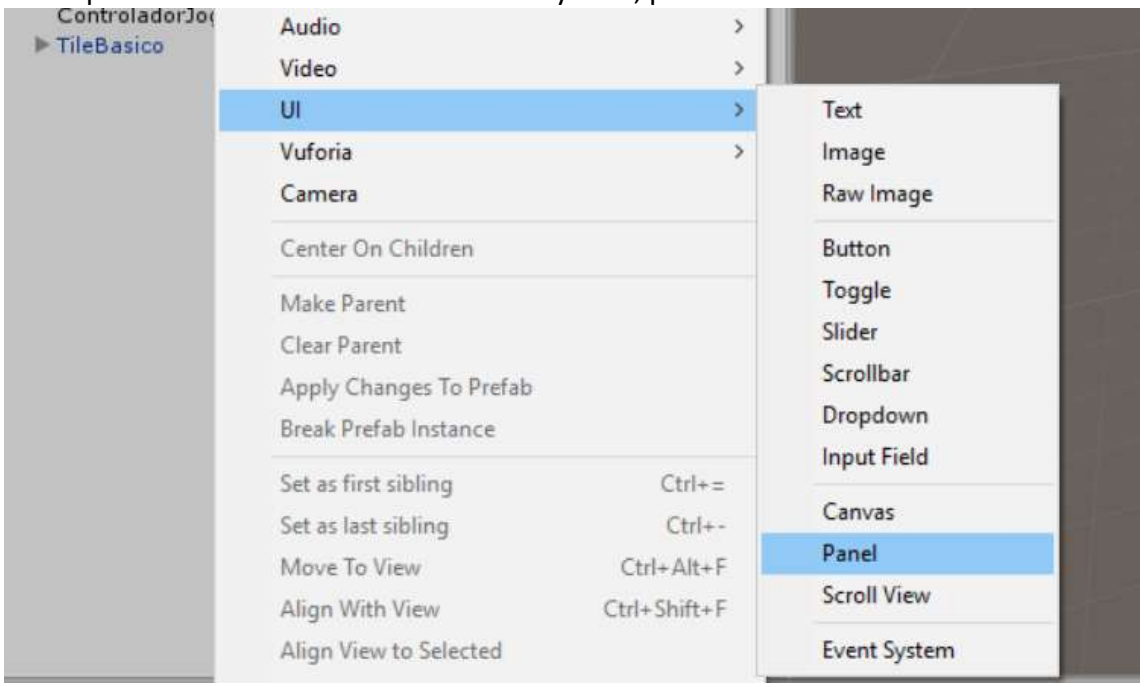
Você alterar a ordem de build, então coloque o MenuPrincipal primeiro. Agora aperte play e divirta-se 😊. Mas e se você quer ir no banheiro e deseja pausa o jogo?

5 – Adicionando um menu com Pausa

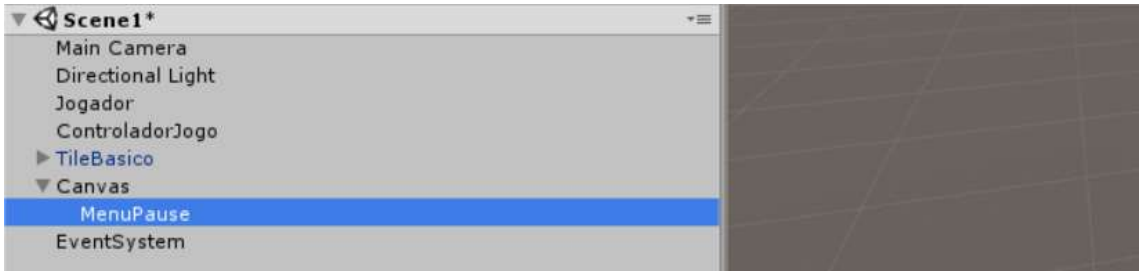
Volte para a Scene1, clicando com o botão direito no arquivo Scene1 dentro da pasta Scenes.

Primeiro passo é escurecer um pouco a tela do jogo assim que entrar em modo pausado. Usando o GO Panel fica bem fácil. Ele se comportará como uma imagem que irá cobrir a tela. Vá em **GameObject -> UI -> Panel.**

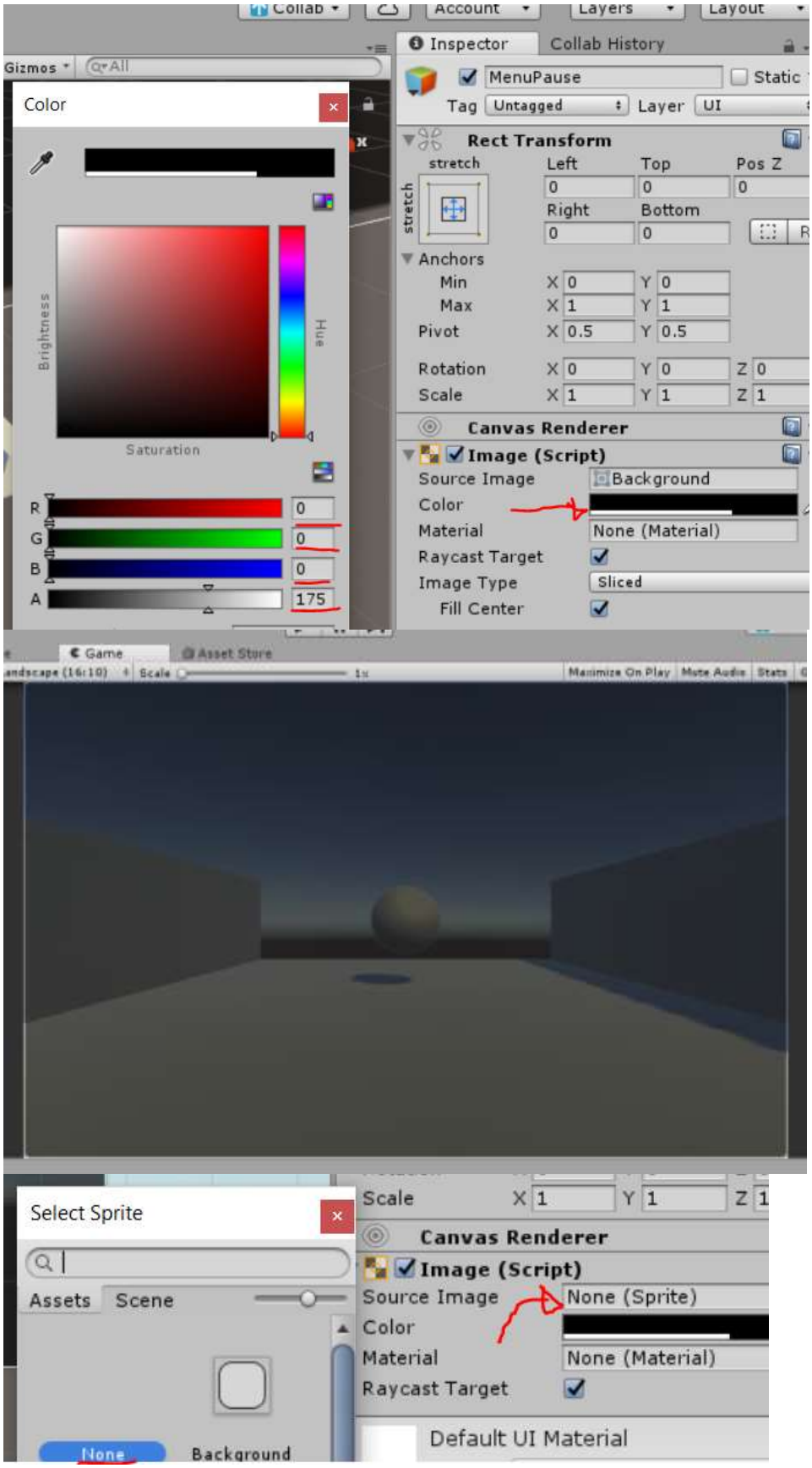
Note que isso irá criar o Canvas e o EventSystem, pois nessa Scene não tem.



Renomeie para MenuPause



Mude a cor para preto com uma transparência de 175. Isso é feito no componente Image. Depois de alterado a cor, na aba Game vai perceber uma borda branca. Para mudar isso, altere a imagem desse componente. Atualmente se encontra em Background, coloque para None.

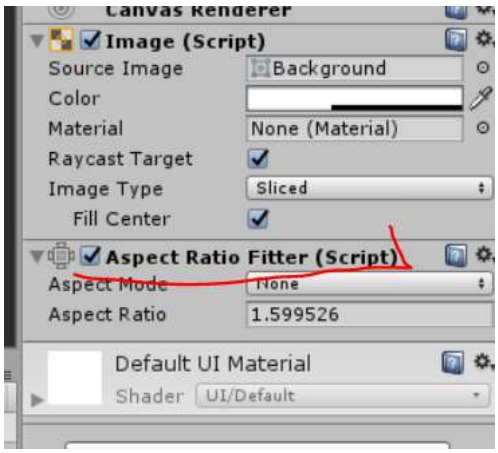


Resultado abaixo, veja que não há mais a borda branca.

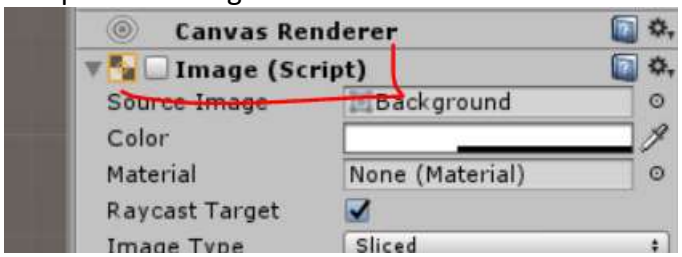
Vamos começar a popular essa tela. Colocaremos um texto informado que o jogo está pausado e mais três botões. Resumir, Reiniciar ou Retornar ao Menu Principal.

Esse Panel atual é responsável pelo sombreamento do jogo. Vamos criar outro Panel para ser o Menu. Queremos que o novo Panel seja filho do primeiro Panel (*Pause Menu*).

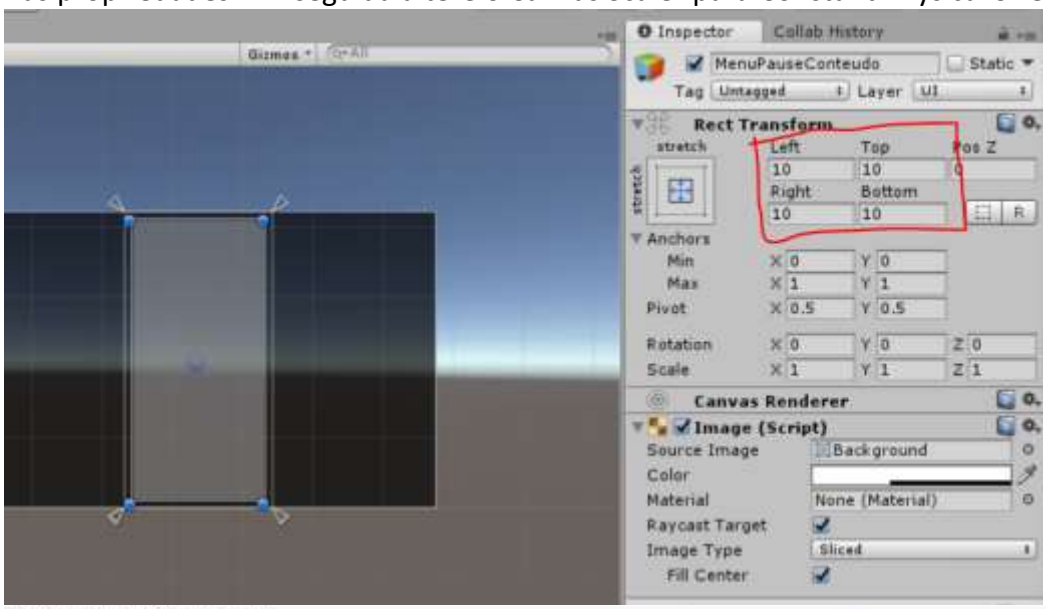
Esse novo Panel não deverá ocupar toda a tela. Para isso vamos usar o componente Aspect Ratio Fitter. Na aba Inspector, com o Panel selecionado, clique em Add Component e procure o Aspect Ratio Fitter.

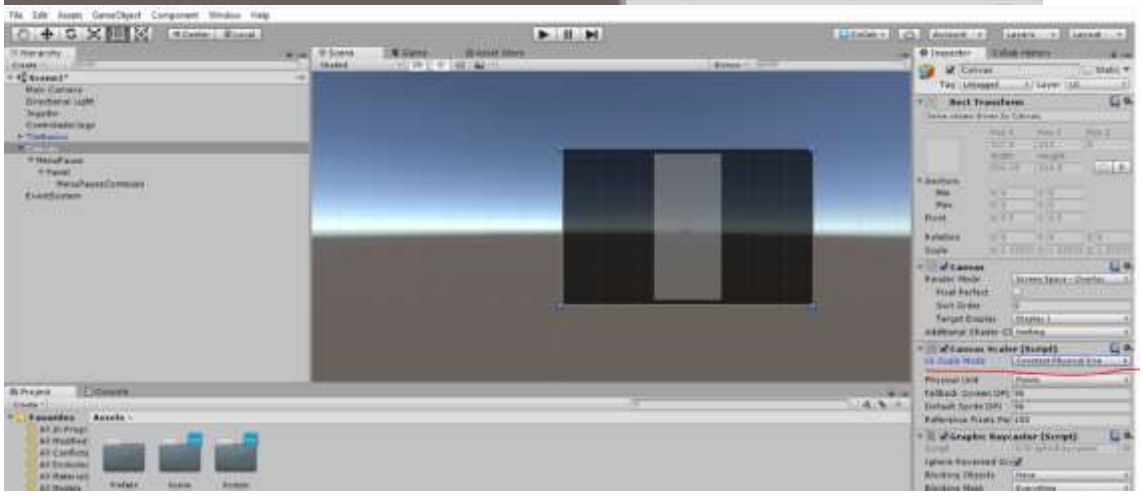


Mude o Aspect Mode para *Fit in Parent*. E coloque 0.5 no Aspect Ratio. Isso quer dizer que será 2x mais alto do que largo. Largura/Altura. ½ = 0.5. Brinque com os valores para você entender o funcionamento. Note que o Panel está encostado nas bordas superiores. Vamos agora deixar esse Panel invisível clicando no checkbox do componente Image e vamos criar outro Panel filho desse.



Crie outro Panel. Renomeie como MenuPauseConteudo e coloque o RT com valor 10 nas propriedades. Em seguida altere o Canvas Scaler para Constant Physical Size.

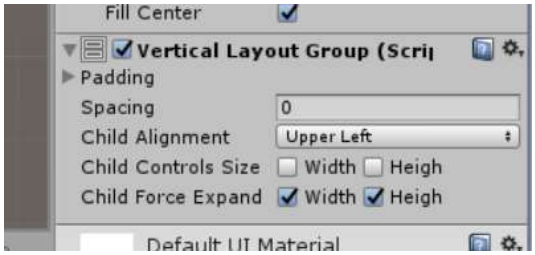




Para adicionar os botões vamos usar outro recurso do Unity, o Layout Groups. Esse recurso permite organizar melhor alguns componentes UIs. Os componentes podem ser organizar em grids, horizontal e vertical. No nosso menu vamos usar Vertical.

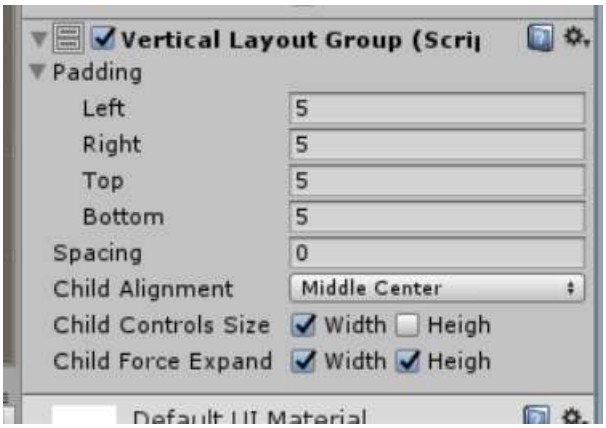
6 – Usando Layout Groups.

Selecione o Panel MenuPauseConteudo e na aba Inspector clique em Add Component. Busque o Vertical Layout Group e o adicione.

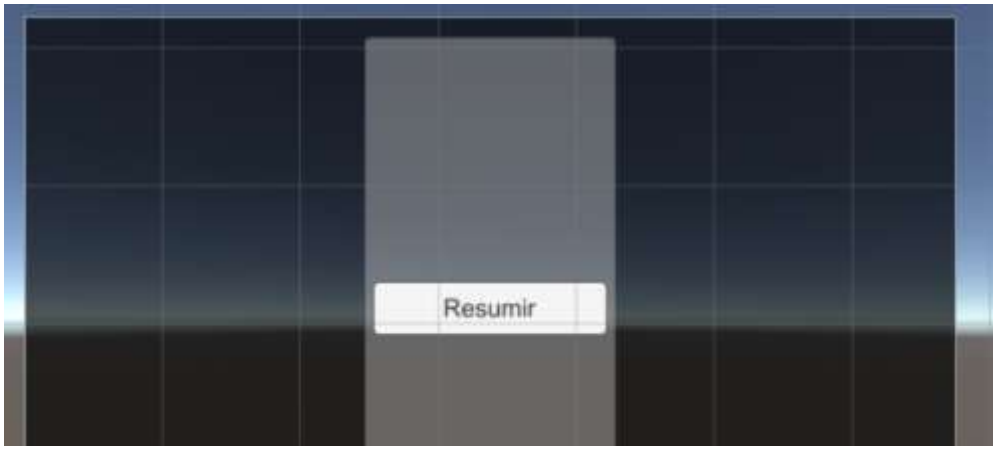


Crie um Button dentro do MenuPauseConteudo. Mude o componente Text para Resumir. Você irá perceber que o Botão está meio fora de centro, talvez preso no canto esquerdo.

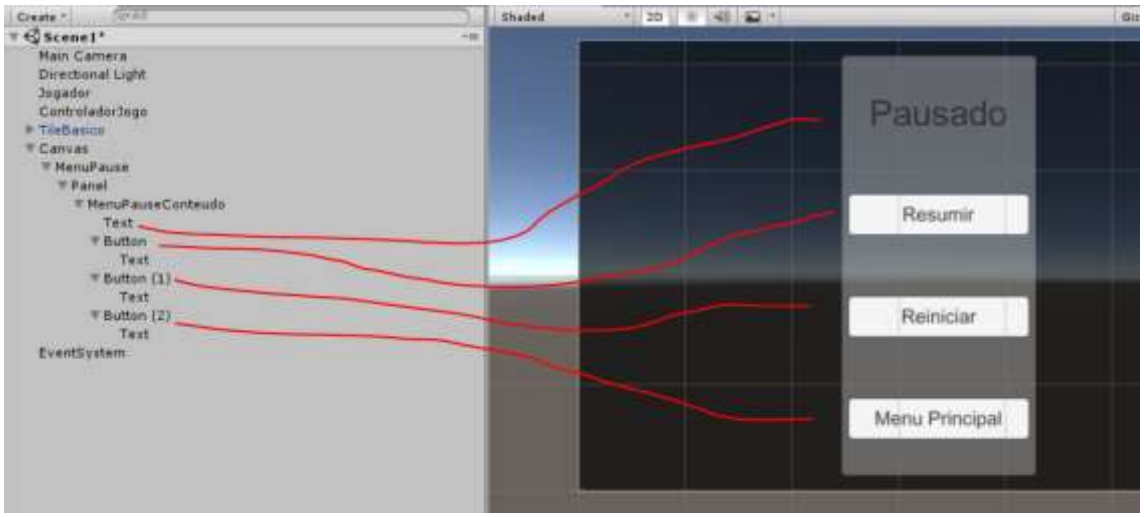
Clique no MenuPauseConteudo e no componente Vertical Layout Group mude o Child Alignment para Middle Center. Marque a opção Width no Child Control Size e coloque nos 4 campos do padding.



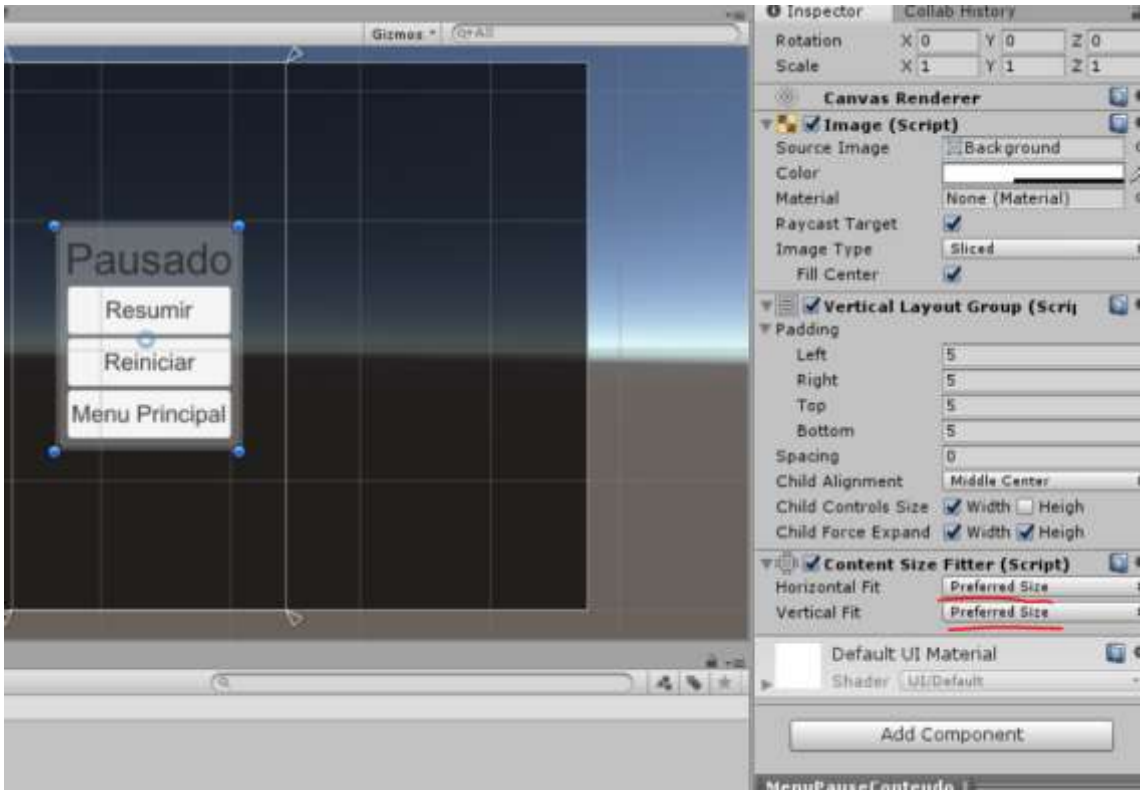
Veja o resultado.



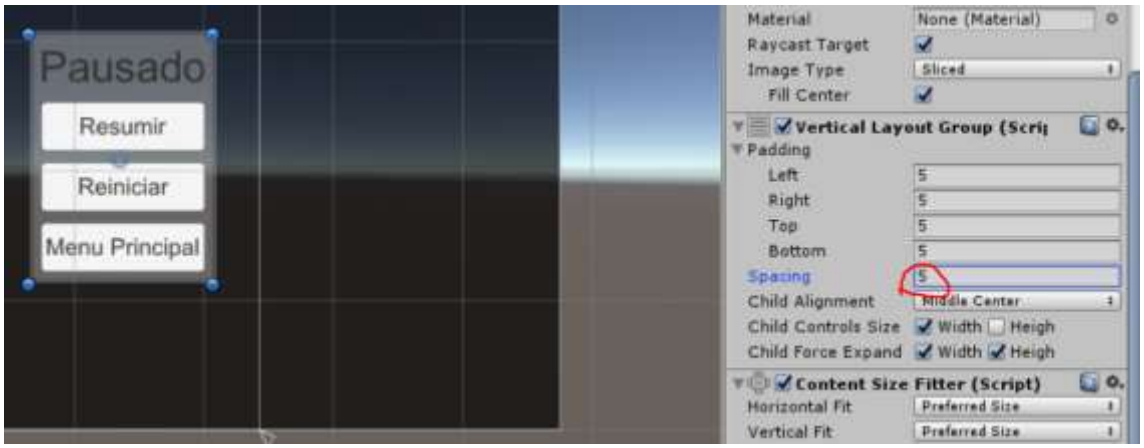
Agora crie duas cópias desse botão (Ctrl D) e mude o texto para Reiniciar e Menu Principal. Adicione também um Text no Panel, e mude o texto para Pausado. A ordem que os elementos UI são dispostos no Layout é baseada na Hierarchy. Então coloque o Text no topo.



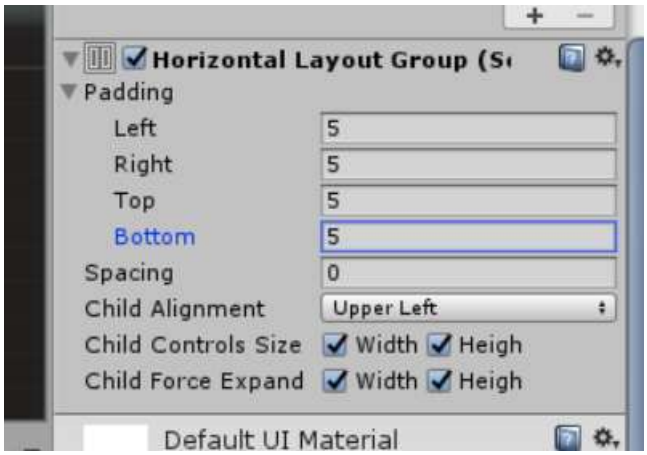
Vamos condensar um pouco esse menu. Com o MenuPauseConteudo, adicione um componente chamado Content Size Fitter. Depois altere Horizontal Fit e Vertical Fit para Preferred Size.



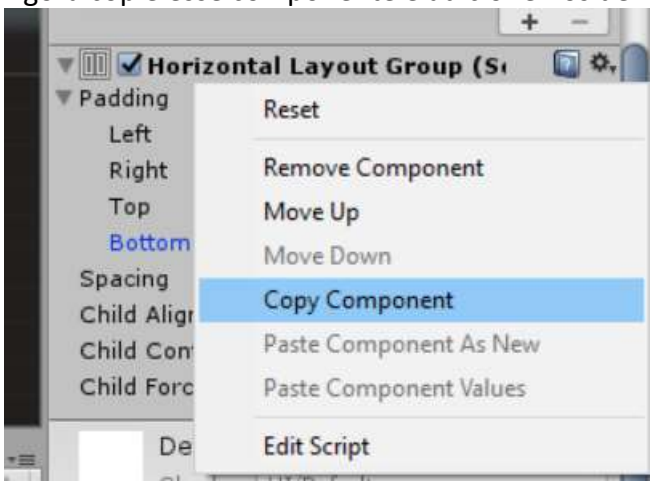
Isso irá agrupar os elementos UI. Não precisa ficar tão apertado assim também. Vá no componente Vertical Layout Group e mude o Spacing para 5.



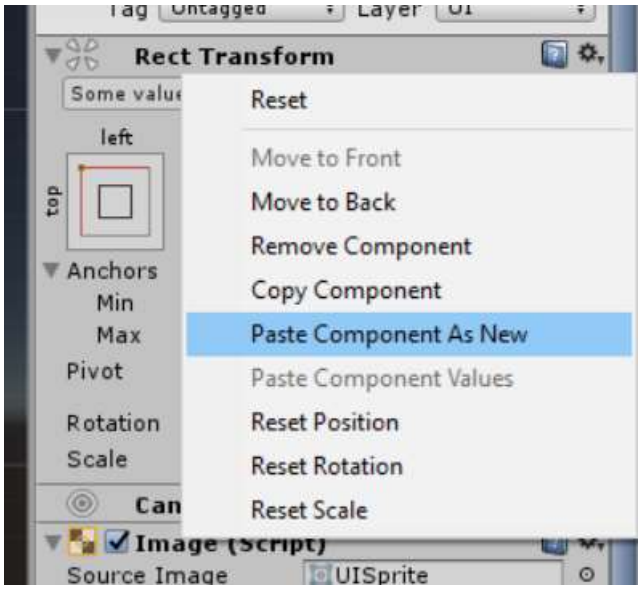
Para ter certeza que estes botões se adaptem a qualquer tamanho, selecione um botão e adicione um Horizontal Layout Group. Adicione um padding 5 e no Child Controls Size marque Width e Height.



Agora copie esse componente e adicione nos demais botões.



Para copiar, clique na engrenagem do componente e clique em Copy Component (figura acima). Para colar, vá no componente RT e clique em Paste Component as New (figura abaixo).



Vamos agora trabalhar em um script para dar a funcionalidade a esses botões. Na pasta Script crie um chamado MenuPauseComp.

```
6   public class MenuPause : MonoBehaviour {
7
8       public static bool pausado;
9
10      [Tooltip("Referencia para o GO. Usado para ligar/desligar")]
11      public GameObject menuPause;
12
13      /// <summary>
14      /// Metodo para reiniciar a Scene, reiniciando o jogo
15      /// </summary>
16      public void Restart() {
17          SceneManager.LoadScene(SceneManager.GetActiveScene().name);
18      }
19
20      /// <summary>
21      /// Metodo para pausar o jogo
22      /// </summary>
23      /// <param name="isPausado">Parametro booleano que indicara se o jogo será pausado</param>
24      public void SetMenuPause(bool isPausado) {
25          pausado = isPausado;
26
```

Na linha 8 temos uma variável static, pois ela pertence a classe e não a instância, ou seja, esta compartilhada. Na linha 11 é a nossa referencia para o GO MenuPause.

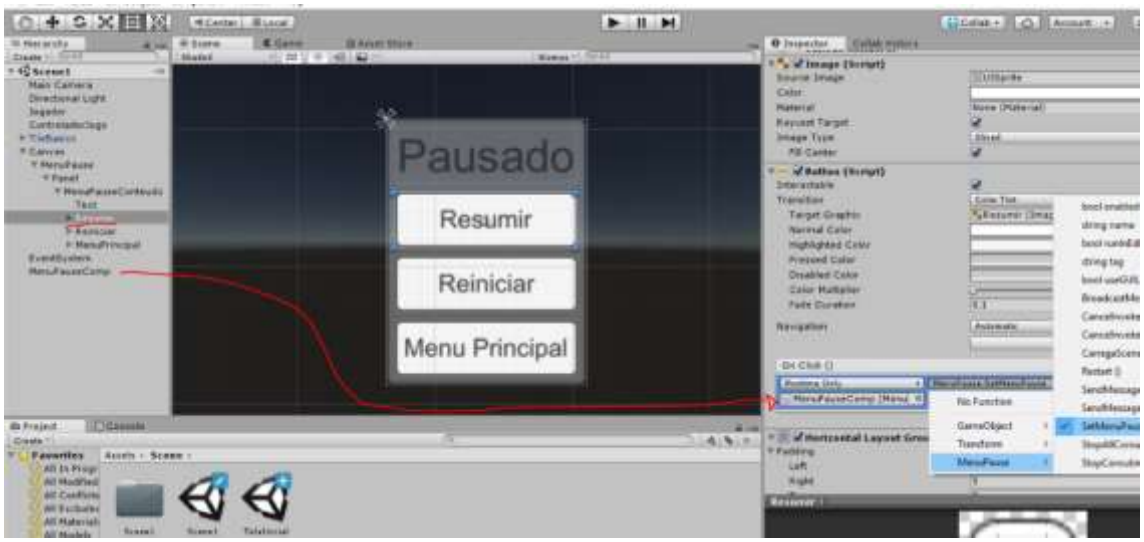
Na linha 16 o método para reiniciar o jogo, a linha 17 não é novidade para nós.
E na linha 24 é um método para pausar o jogo. A primeira ação é atribuir o valor novo para a variável estática. Se o parâmetro recebido for true então pausaremos o jogo colocando 0 na timeScale (parando o tempo). Caso contrário colocamos 1, fazendo o tempo fluir normalmente.

Como variáveis estáticas não resetam com novas instancias no Unity, no método Start colocamos pausado como false para garantir que o jogo por padrão não está pausado.
E temos o método CarregaScene, já conhecido, que ira carregar Scenes baseada no nome. Será usada para ir para o MenuPrincipal que no nosso caso é a Tela Inicial.

```
32      /// <summary>
33      /// Metodo para carregar uma Scene
34      /// </summary>
35      /// <param name="nomeScene">nome da scene que sera carregada</param>
36      public void CarregaScene(string nomeScene) {
37          SceneManager.LoadScene(nomeScene);
38      }
39
40      // Use this for initialization
41      void Start () {
42          pausado = false;
43      }
```

Voltando para o Unity, crie um GO chamado MenuPauseComp e adicione o script MenuPause como componente. E para a variável MenuPause coloque o Panel MenuPause, veja a Figura abaixo.

Agora podemos configurar os botões. Primeiramente renomeie os GOs dos botões para ficar mais claro (atualmente está Button, Button (1) e Button (2)). E agora com o Resumir selecionado, vá no On Click e pressione +. Adicione o GO MenuPauseComp e busque o método MenuPauseComp.SetPauseMenu.



Repita para os outros botões. Mas use o método apropriado. Para o botão Reiniciar escolha o Restart(). Para o botão MenuPrincipal escolha o CarregaScene e passe o nome da sua tela inicial, no meu caso é TelaInicial.

Dê play e veja o que ocorre.

7 – Para de enrolar e pausa logo o jogo

Se você realmente jogou, viu que estamos com alguns problemas. O botão Menu Principal funciona e o Reiniciar também. Se você pressionar Resumir o jogo prossegue normalmente, mas não conseguimos pausar novamente. Além disso o jogo não fica pausado no menu Pausado. Para isso vamos até o Script que controla o jogador, JogadorComportamento.

Observe as modificações.

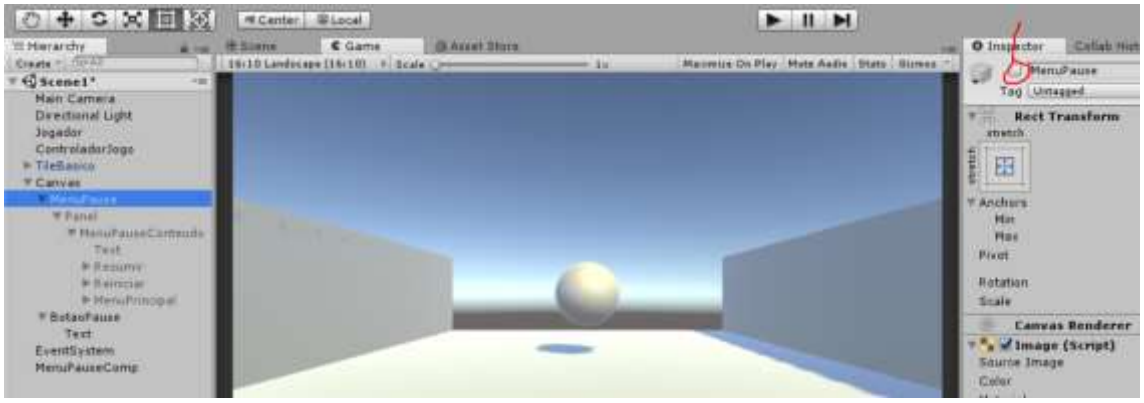
```
52      void Update () {
53
54          //Se o jogo esta pausado nao faca nada
55          if (MenuPause.pausado)
56              return;
57
```

Logo no início do método Update verificamos se o jogo está pausado. Se estiver não fazemos nada.

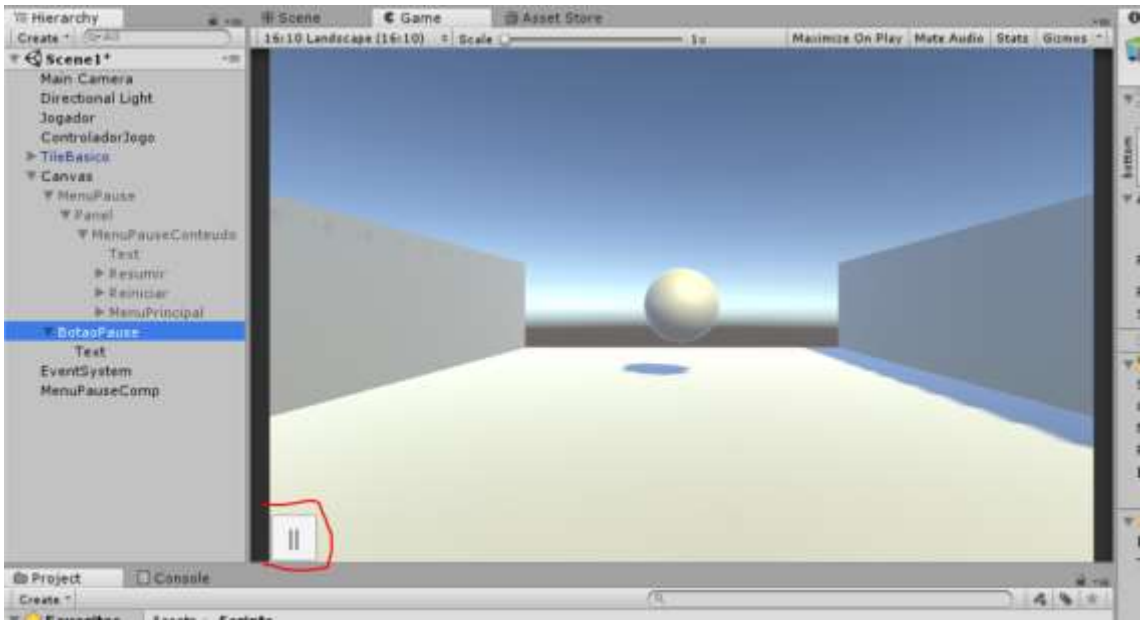
```
92
93          var forcaMovimento = new Vector3(velocidadeHorizontal,0,velocidadeRolamento);
94
95          //Time.delta nos retorna o tempo gasto no frame anterior
96          //algo em torno de 1/60fps
97          //Usamos esse valor para garantir que a nossa bola se desloque com a mesma velocidade
98          //nao importa o hardware.
99          forcaMovimento *= (Time.deltaTime * 60);
100
101         //Aplicar uma força para que a bola se desloque
102         rb.AddForce(forcaMovimento);
103     }
```

E no fim do método Update ao invés de aplicarmos a força diretamente na bola, nós primeiro multiplicamos pelo Time.deltaTime. Esse valor nos retorna o tempo gasto para se executar o Update anterior. Assim mantemos a velocidade do jogo *frame independent.*

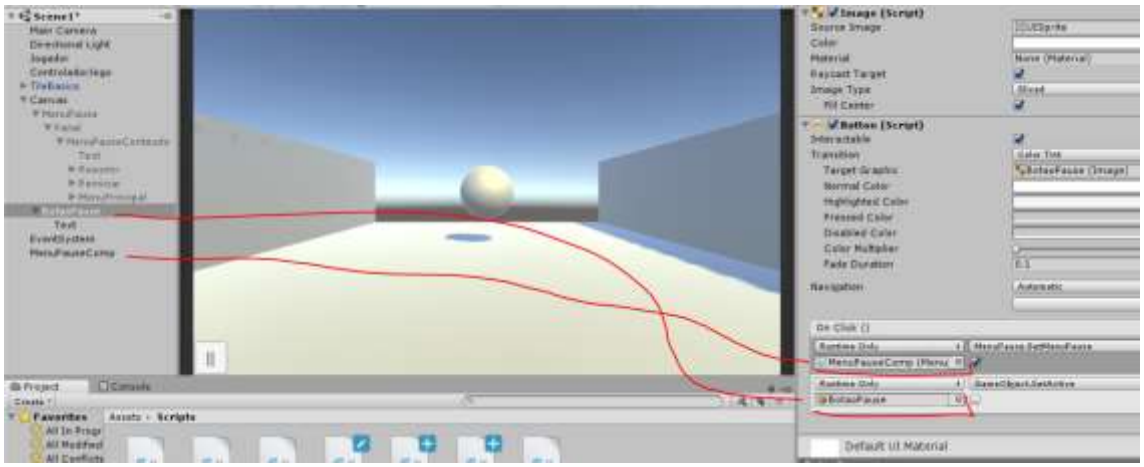
O jogo deve começar fora do modo pausado, para isso vamos desabilitar o Panel MenuPause. Selecione na aba Hierarchy e na aba Inspector desmarque o checkbox ao lado do nome, bem no começo. Observe que ele fica com o nome cinza na Hierarchy.



Agora precisamos de ativa-lo durante o jogo. Vamos criar um botão para isso. Com o Canvas selecionado crie um UI->Button e chame de BotaoPause. Altere as âncoras e pivot para que o botão fique no canto esquerdo, coloque uma largura de 30 e mude o texto para ||.

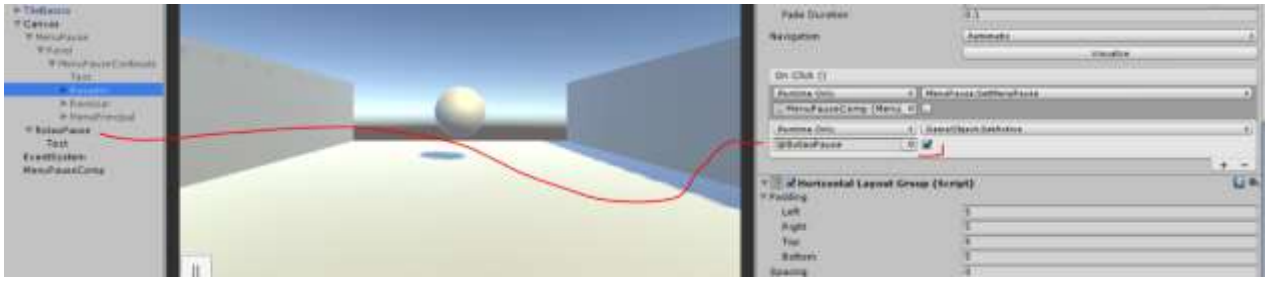


Ainda neste botão vá no On Click () e adicione um comportamento com o GO MenuPauseComp. Use o método SetPauseMenu e marque o checkbox, pois queremos passar True para de fato pausar o jogo. E enquanto o jogo está pausado queremos remover o botão de pause. Para clique novamente no + e arraste o próprio BotaoPause. Busque a função GameObject->SetActive e deixe desmarcado.



O que estamos fazendo é no de pressionar o botão pause, pausamos o jogo e logo em seguida desativamos o botão pause.

Mas agora precisamos que este botão apareça novamente depois do jogo reiniciado. Assim, vá no botão Reiniciar e adicione a segunda funcionalidade. Mas desta vez deixe o checkbox marcado, pois queremos que o botão pause seja reativado.



Agora aperte Play e jogo. Depois aperte o botão pause e tente Reiniciar o jogo.....

Oops o jogo continua pausado. Lembre-se, sobre as variáveis estáticas. Precisamos chamar o método SetPauseMenu(false) para sempre começar com o jogo fora do modo pause.

```
40      // Use this for initialization
41      void Start () {
42          pausado = false;
43          SetMenuPause(false);
44      }
45  }
```

Divirta-se!