Curso Técnico

de Programação de Jogos Digitais

Aula 16

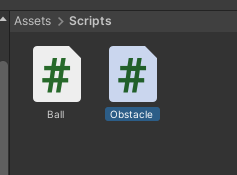
índice

[1. Lógica dos obstáculos 3](#_Toc169515519)

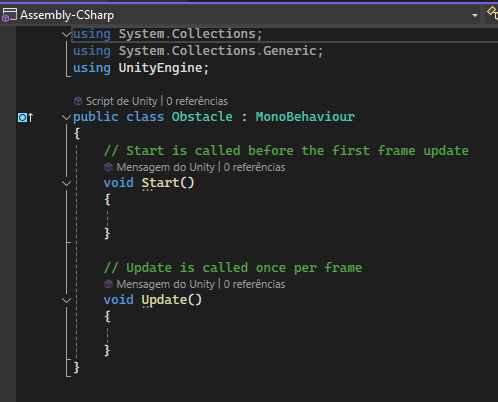
[2. Compreendendo todo o código 6](#_Toc169515520)

[3. Referencias 8](#_Toc169515521)

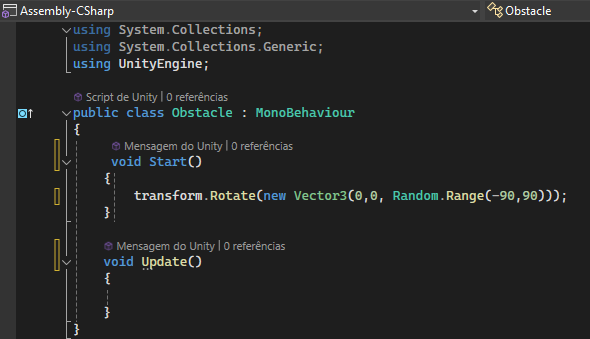
# Lógica dos obstáculos

Vamos para logica dos objetos restantes, para os objetos que caem queremos que eles fiquem rotacionando além de seguir em queda. Para gerar este efeito devemos criar um novo script para estes objetos.

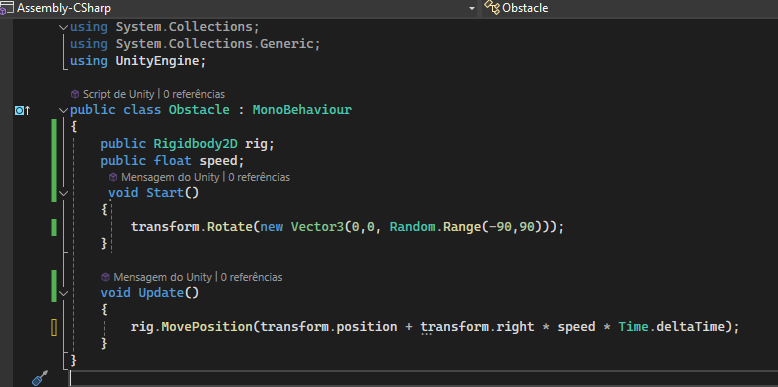
Abra o arquivo.



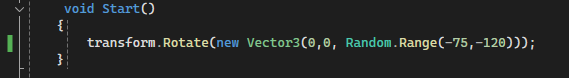
Vamos adicionar um modo randômico para a rotação dos objetos.



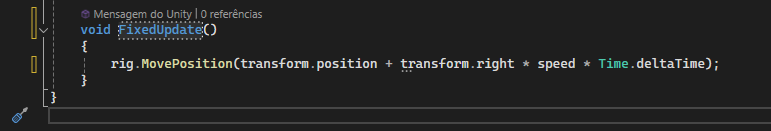
Agora vamos criar o efeito de gravidade (queda) usando o RigidBody e para isso devemos criar uma variável para o rigidbody e gerar o movimento de queda através do código abaixo. Alem disso devemos adicionar a velocidade através de uma variável.



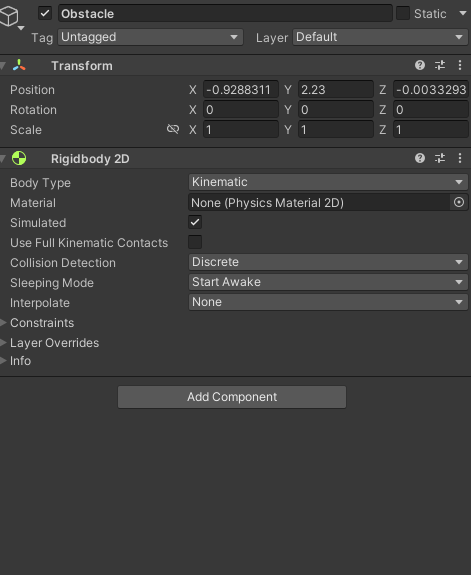
Faça o acerto quanto a posição de criação dos objeto conforme seu gosto



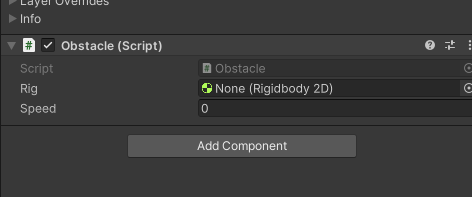
Quando trabalhamos com componentes de física sempre usamos o fixedupdate pois esta será a atualização de elemento com física.



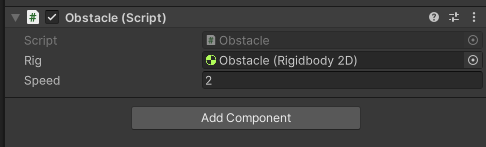
Salve e então adicione o script ao componente do obstacle



Arraste o script osbtacle para o componente do objeto obstacle.



Adicione o rigidbody ao Rig onde está None. Coloque também uma velocidade de queda, para iniciar vamos colocar 2 no Speed.



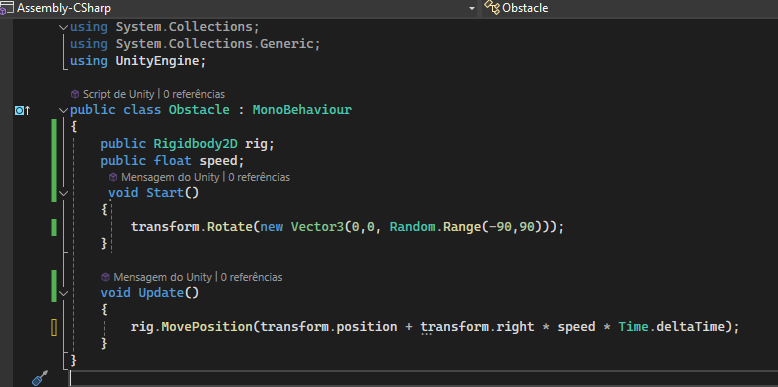
Faça novamente o teste de seu projeto.



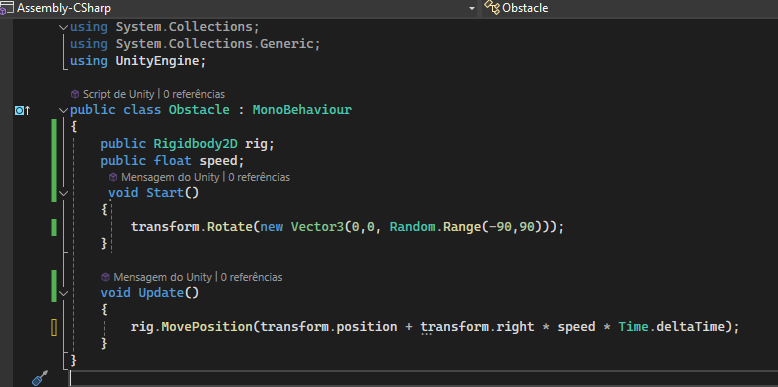
Todos esta modificações no código me deixaram com dúvidas. Melhor ver esse script em seus detalhes.

# Compreendendo todo o código

O código define uma classe chamada `Obstacle`, que é usada para controlar o comportamento de um obstáculo em um jogo 2D desenvolvido na Unity. Vamos analisar cada parte em detalhes:



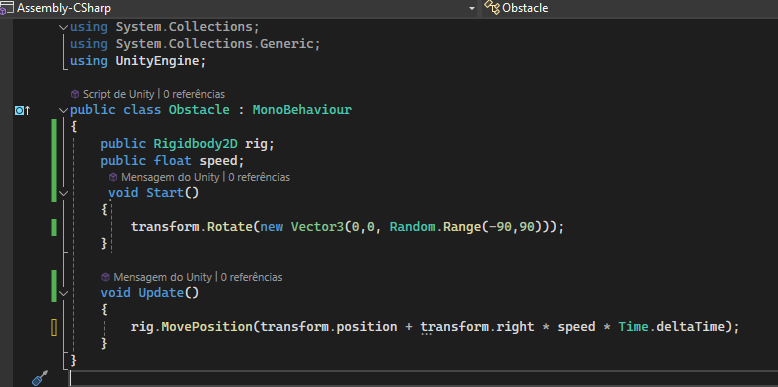
Essas diretivas `using` importam namespaces que contêm classes e funções necessárias para o script. `System.Collections` e `System.Collections.Generic` são usados para coleções de dados, enquanto `UnityEngine` é fundamental para qualquer script do Unity, pois contém as classes principais como `MonoBehaviour`, `GameObject`, `Transform`, etc.



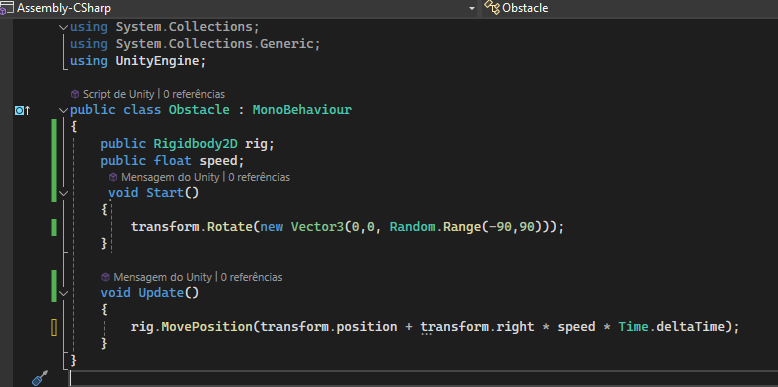
A classe `Obstacle` herda de `MonoBehaviour`, o que permite que ela seja anexada a objetos de jogo (game objects) na Unity. A classe tem duas variáveis públicas:

- `rig` do tipo `Rigidbody2D`, que controla a física do objeto 2D.

- `speed` do tipo `float`, que define a velocidade do objeto.



O método `Start` é chamado uma vez quando o script é ativado. Aqui, o objeto é rotacionado em torno de seu eixo Z por um ângulo aleatório entre -90 e 90 graus. Isso é feito para dar uma variação na orientação inicial do obstáculo, tornando o jogo mais dinâmico.



O método `FixedUpdate` é chamado em intervalos fixos, o que o torna ideal para manipulações relacionadas à física. Neste método, a posição do `Rigidbody2D` (`rig`) é atualizada. A nova posição é calculada movendo-se na direção direita do objeto (`transform.right`) com uma velocidade definida por `speed` e ajustada pelo tempo decorrido (`Time.deltaTime`) para garantir que o movimento seja independente da taxa de quadros.

Logo, em nosso código conseguimos ver os seguintes elementos:

- Namespaces Importados: As linhas `using` importam bibliotecas necessárias.

- Classe Obstacle: A classe herda de `MonoBehaviour`, permitindo que seja usada como um componente na Unity.

- Variáveis Públicas: `rig` para controle de física e `speed` para definir a velocidade do movimento.

- Método Start: Rotaciona o objeto inicialmente por um ângulo aleatório.

- Método FixedUpdate: Move o objeto em uma direção constante com base em sua velocidade.

Esse script permite que um objeto de jogo se mova constantemente e de forma controlada, com uma rotação inicial aleatória para maior variação.

Os códigos usados em nossas aulas estão disponíveis em nosso repositório no GitHub.



https://github.com/rildexter/pjd/tree/main/djm

# Referencias

1. Unity Technologies. (2020). **Unity User Manual.** Unity Technologies.
2. [C#, 2020] **Visual C# Developer Center**, Microsoft Docs.

Microsoft Corporation. (2020).

1. Geig, M. (2018). **Unity 2018 Game Development in 24 Hours**, Sams Teach Yourself. Sams Publishing.
2. Hocking, J. (2015). **Unity in Action: Multiplatform Game Development in C#.** Manning Publications.