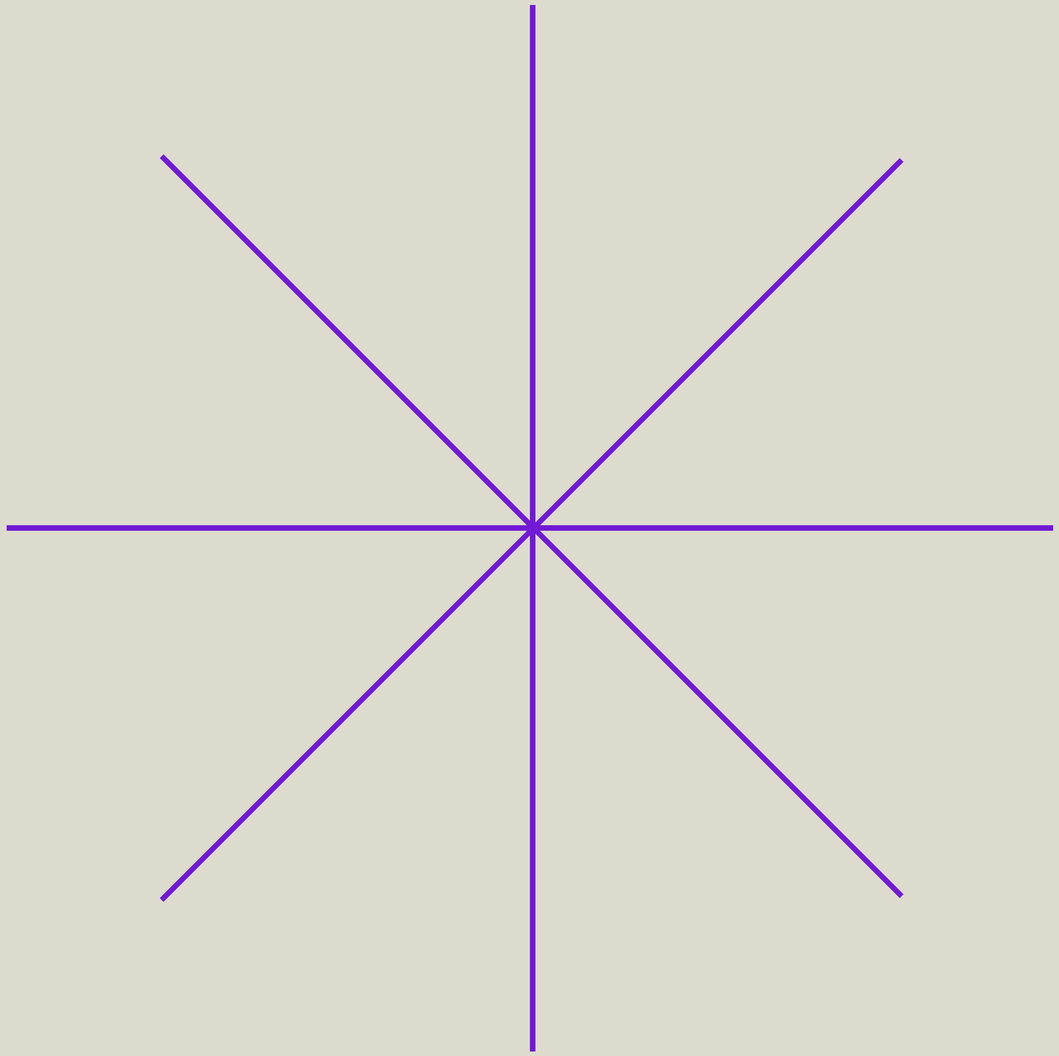
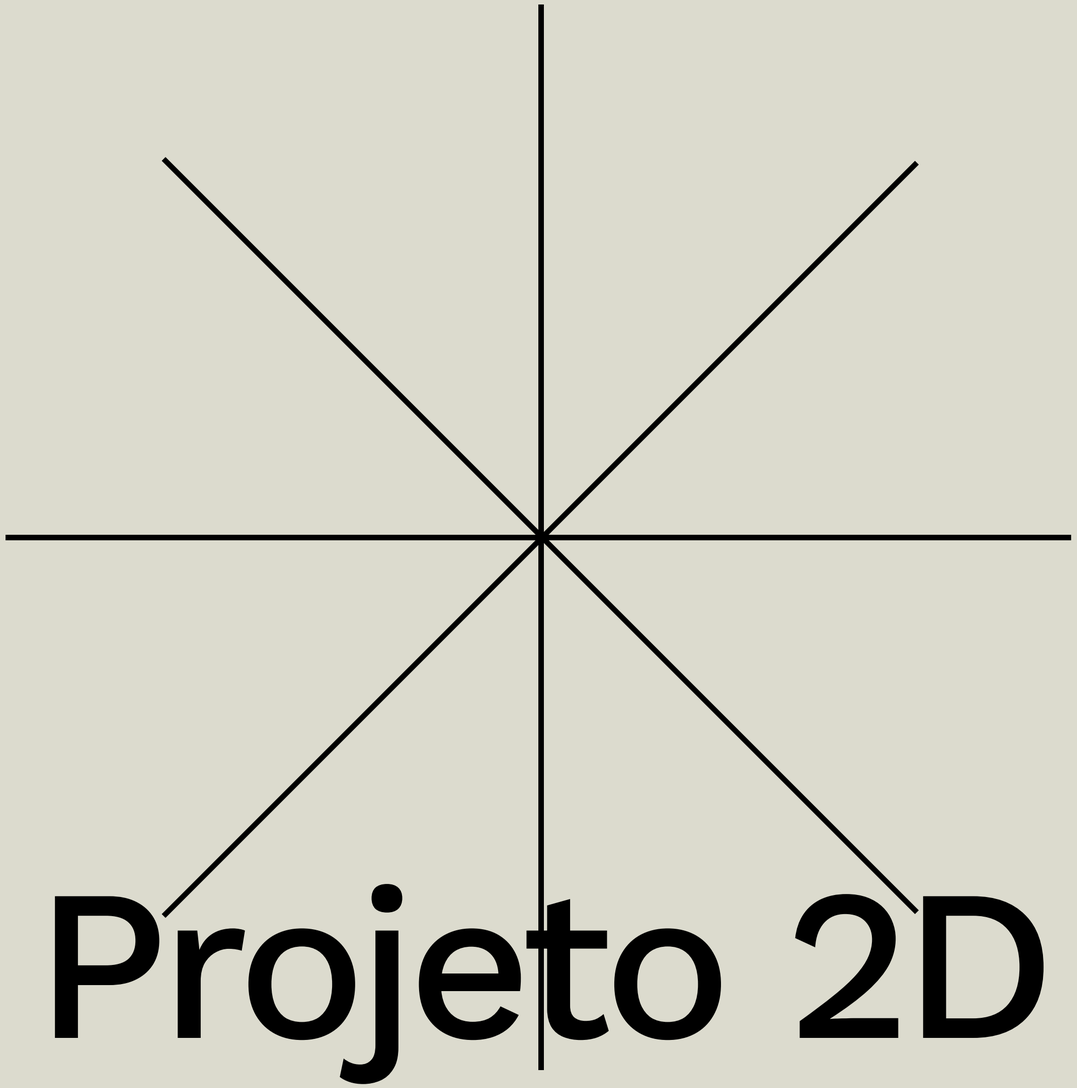
DESENVOLVIMENTO PARA ÂMBIENTES MÓVEIS



# Exercício 2D

DANIELA MONTEIRO

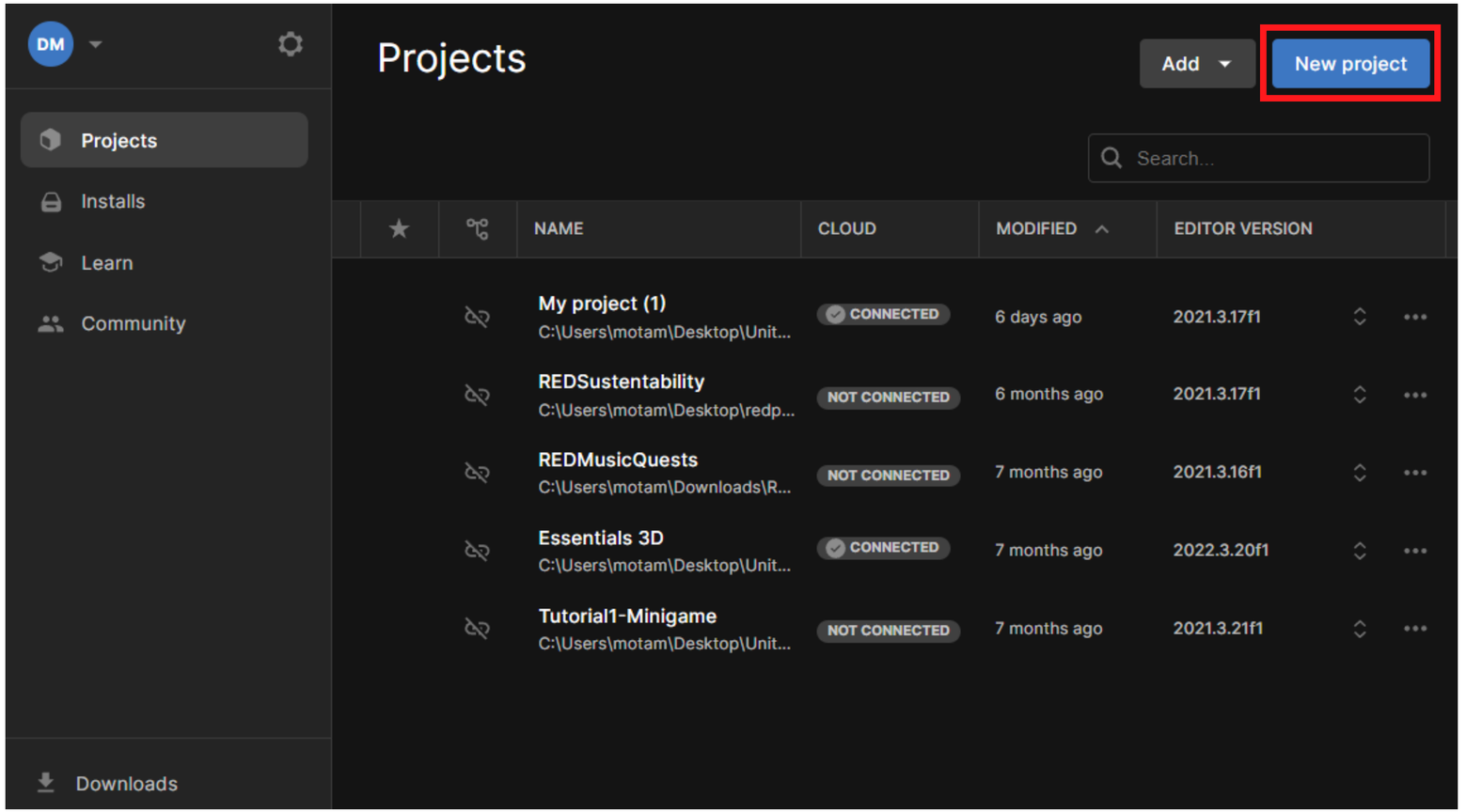
# ÍNDICE



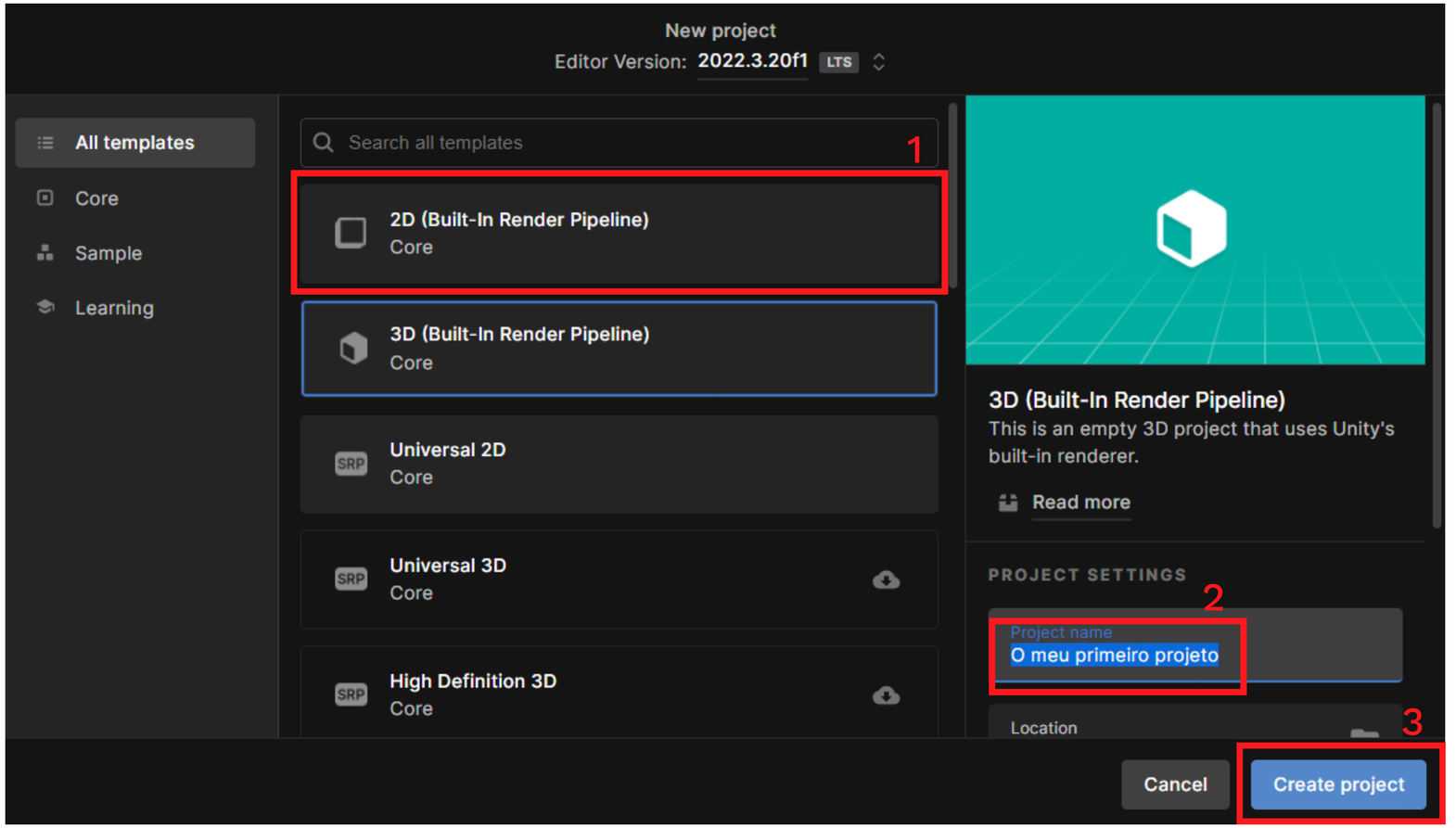
# 01
# Criar um novo Projeto 2D para Mobile

DANIELA MONTEIRO

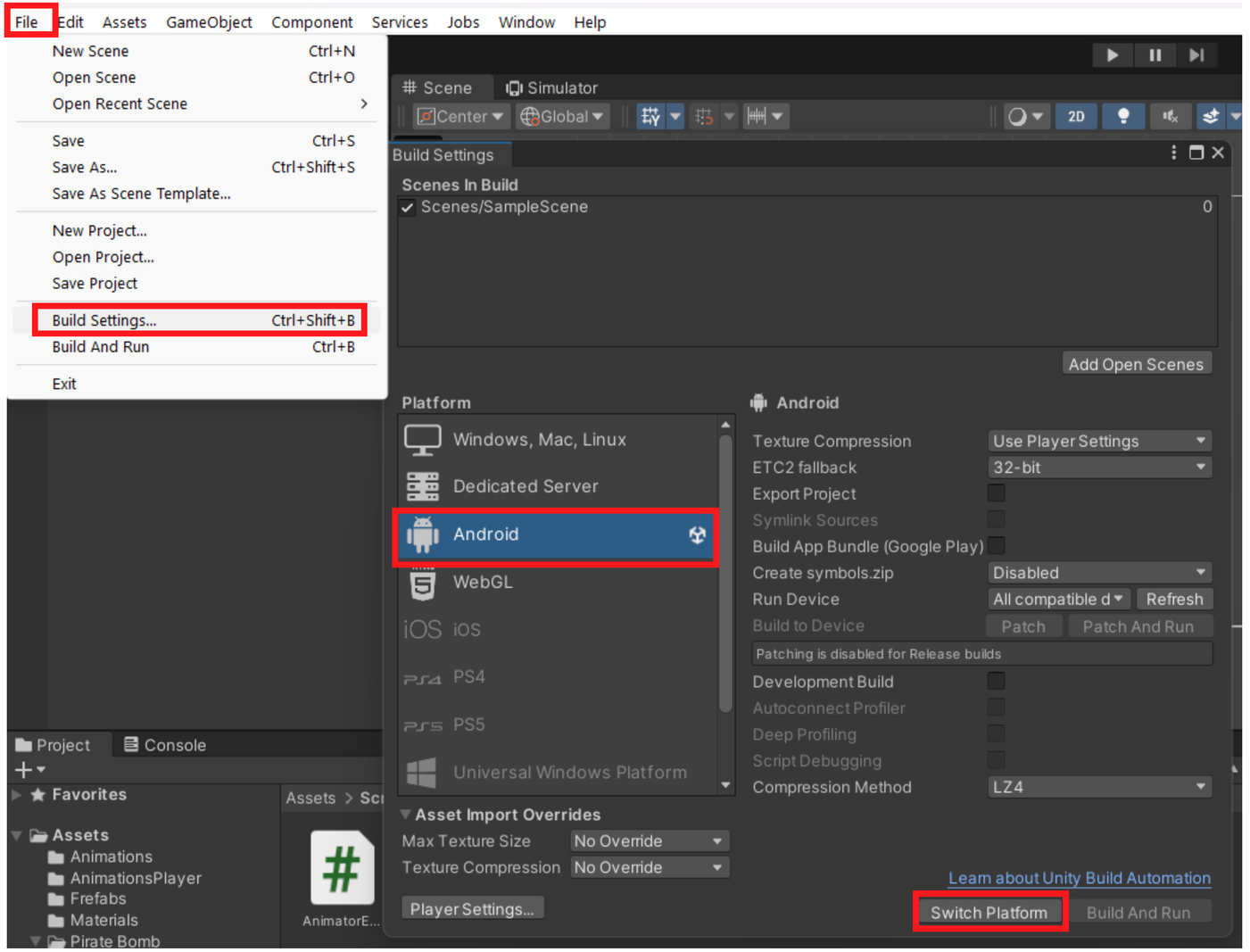
# CRIAR UM NOVO PROJETO



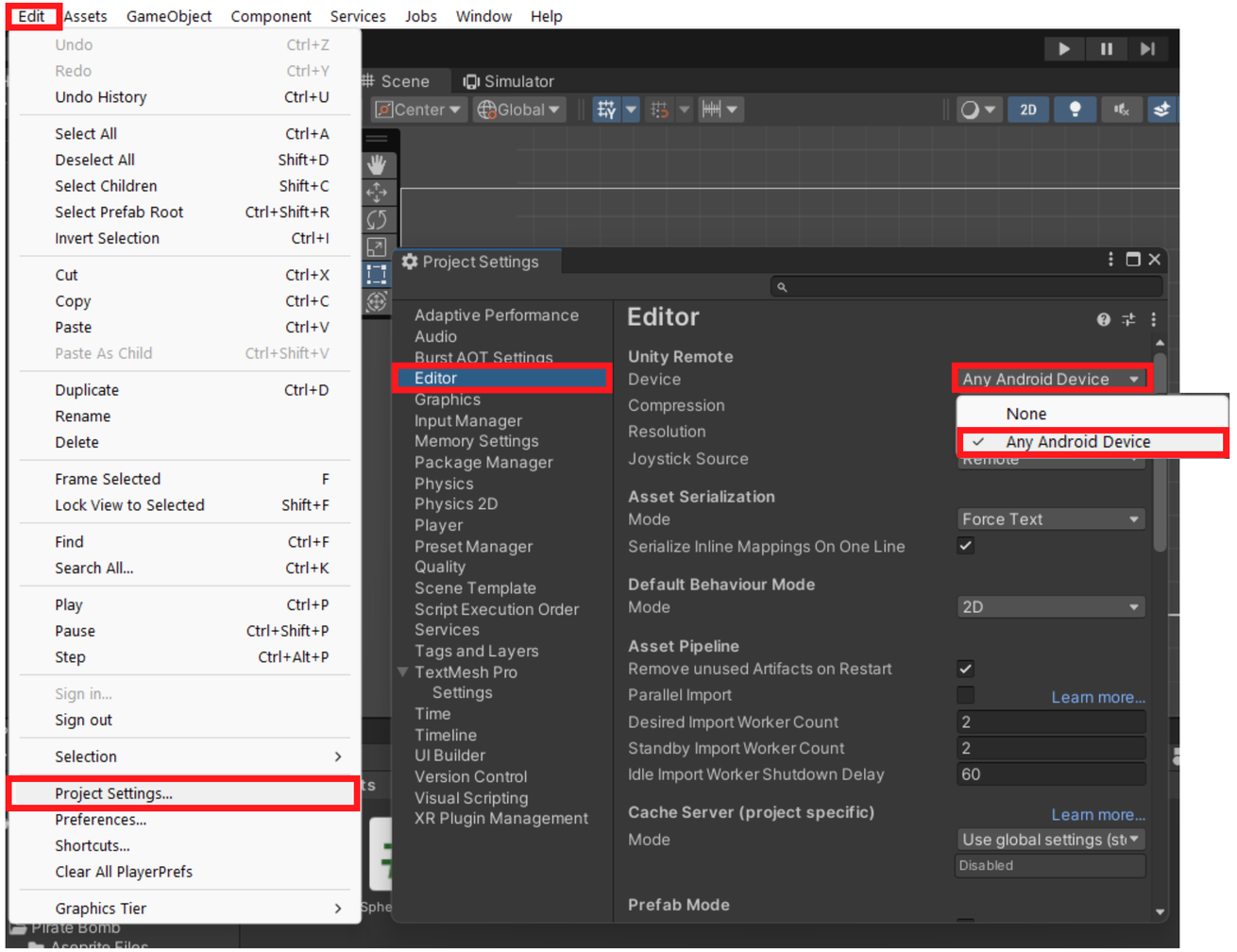
# CRIAR UM NOVO PROJETO

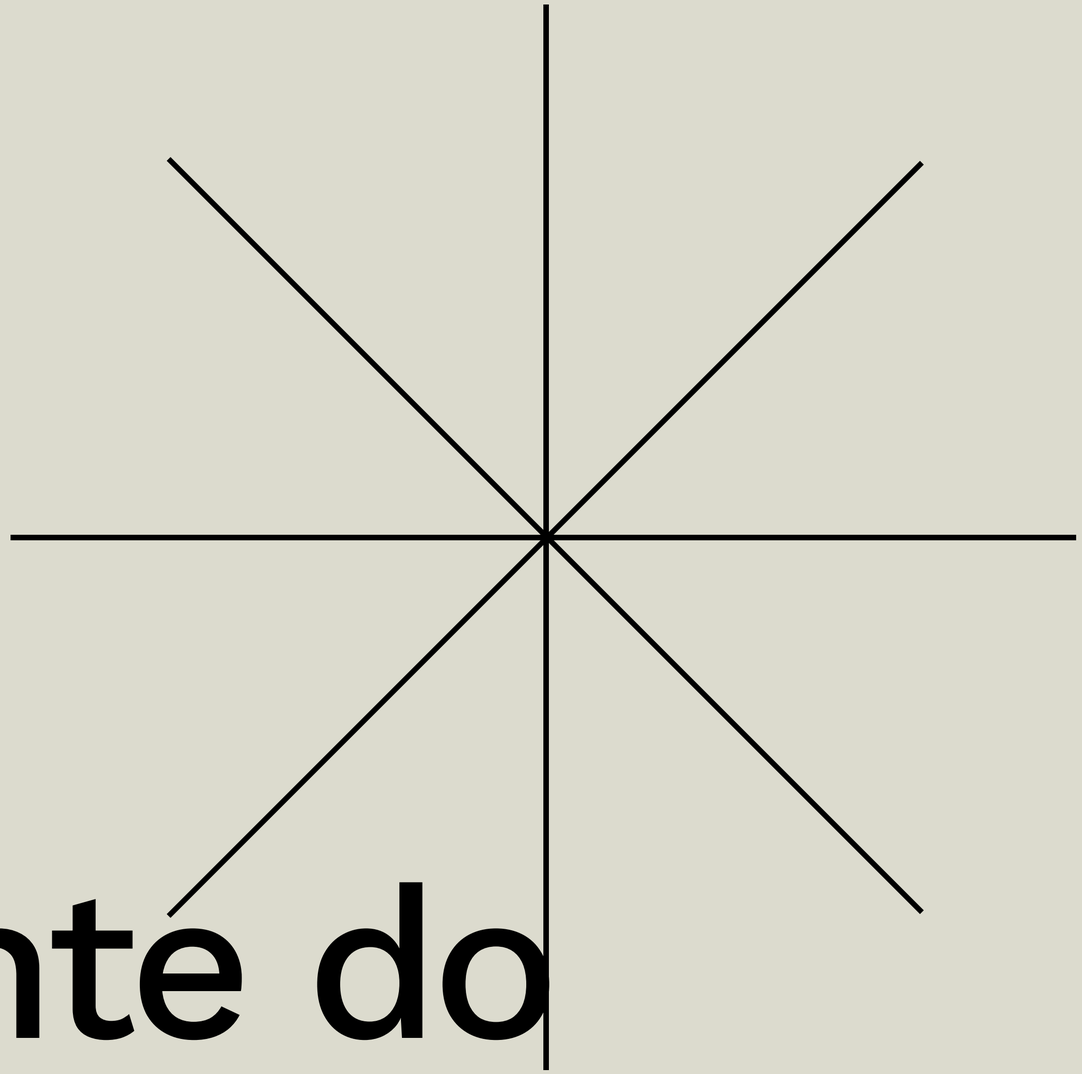


# MUDA DE PLATAFORMA



# MUDA AS PREFERÊNCIAS DO TEU PROJETO

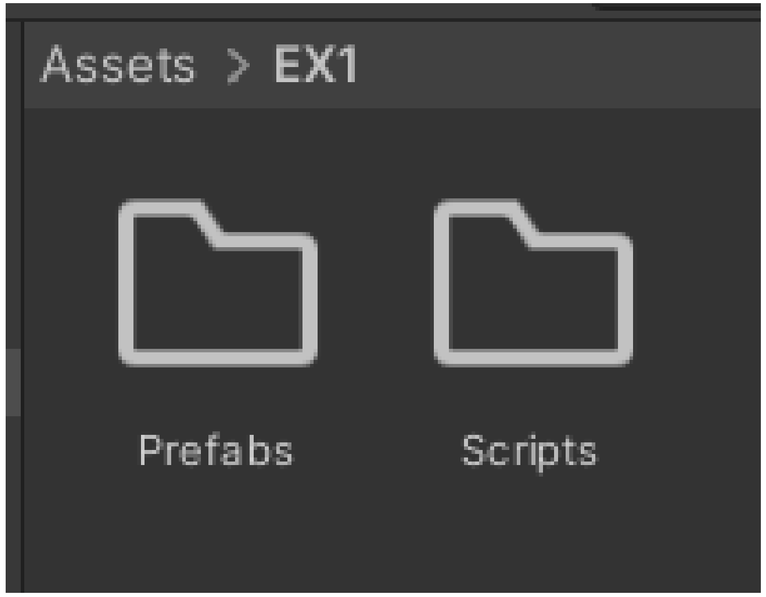




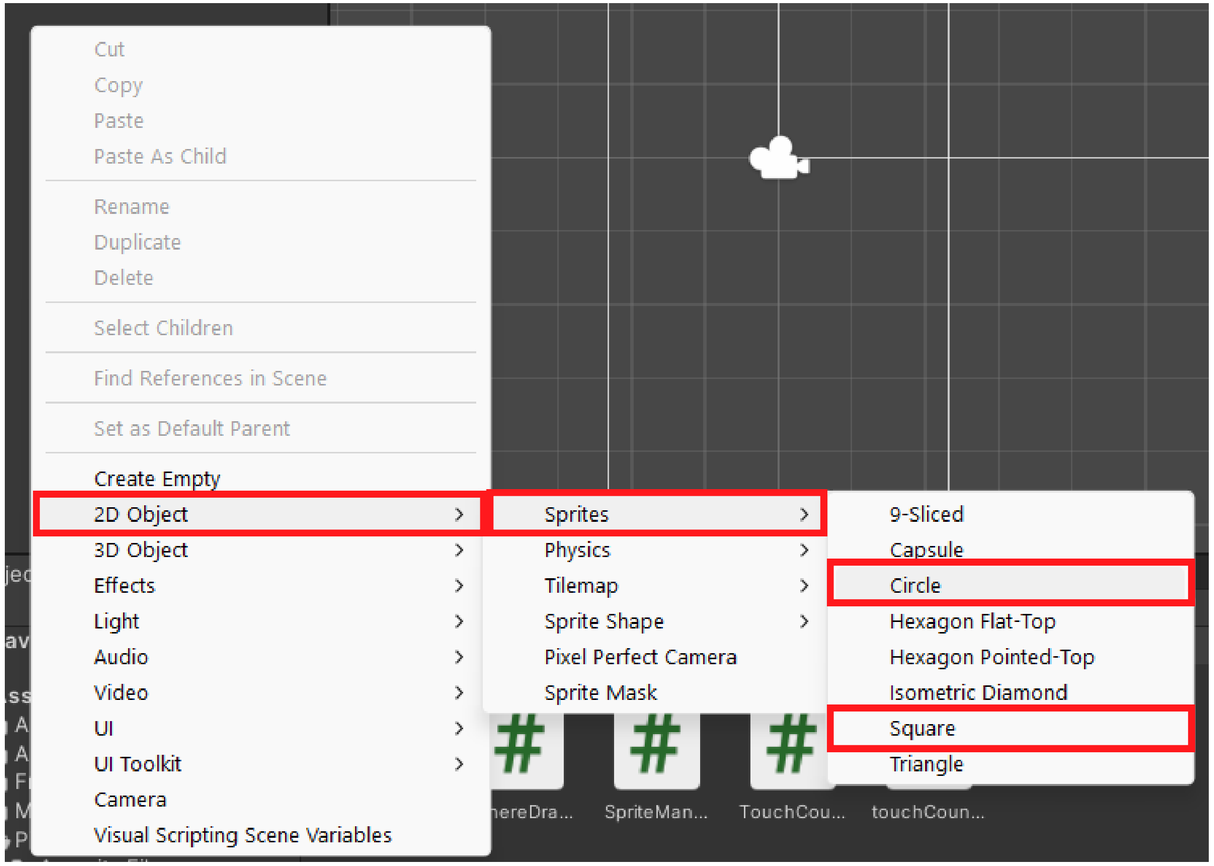
# 02
# Criar o Ambiente do Jogo

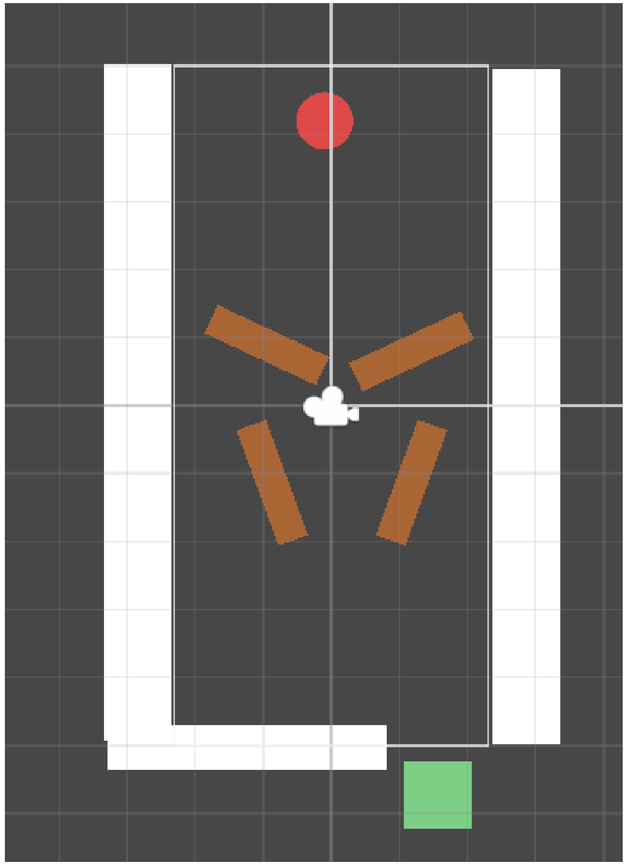
DANIELA MONTEIRO

# CRIA AS SEGUINTES PASTAS NO TEU PROJETO:

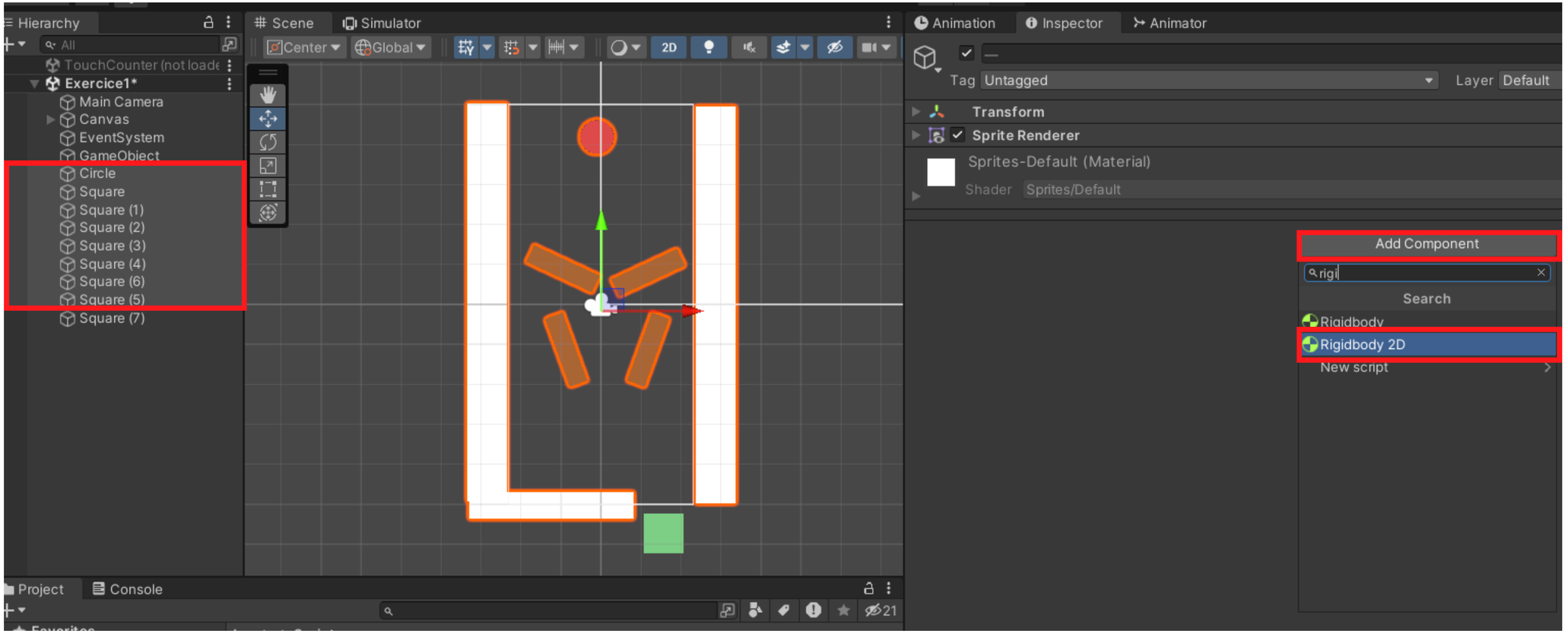


# ADICIONA UM CÍRCULO, QUATRO RETÂNGULOS E AS BARREIRAS DO JOGO:

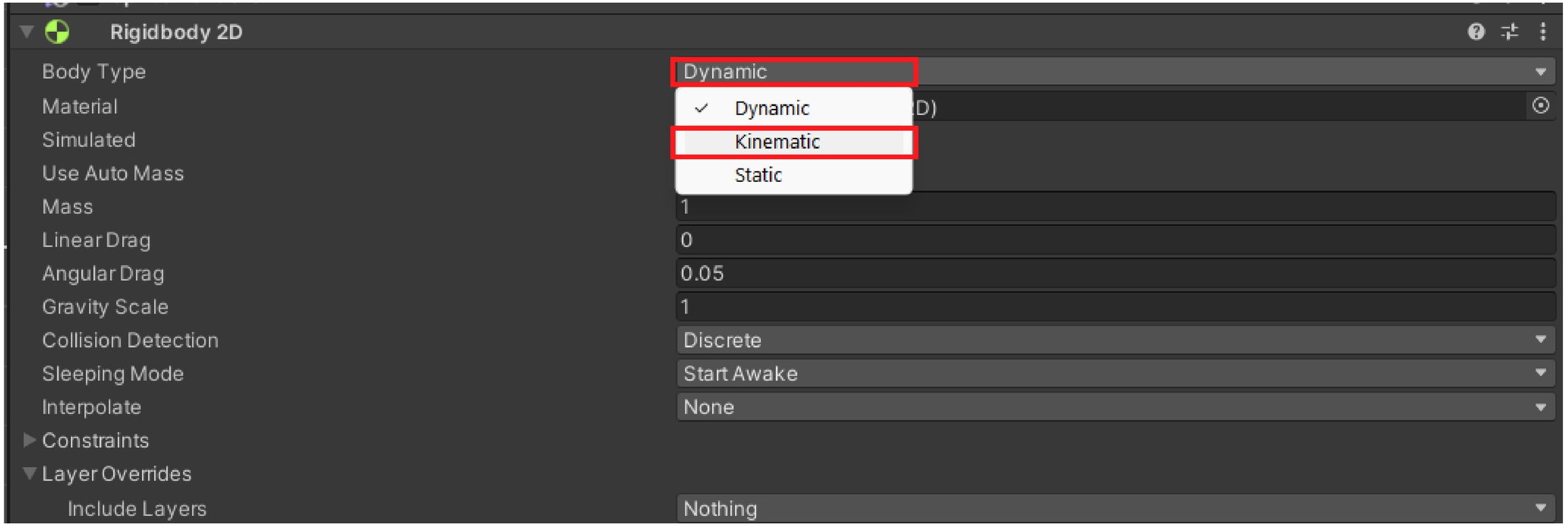


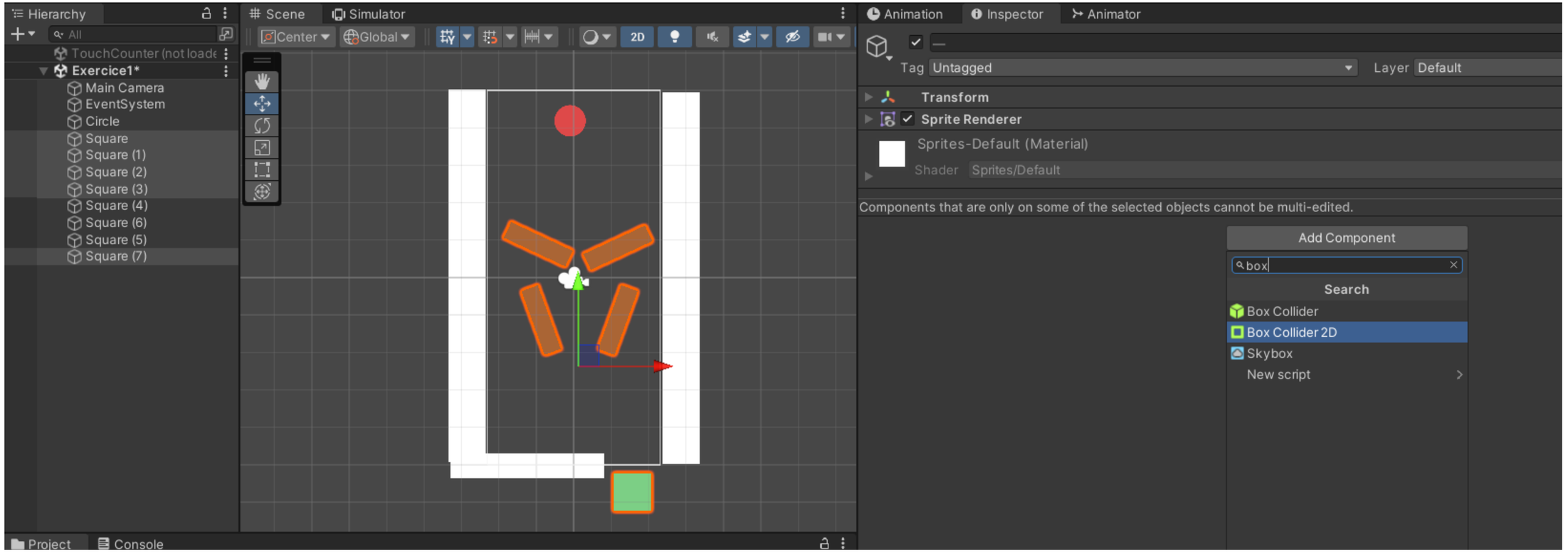
# ADICIONA O COMPONENTE RIGIDBODY2D A TODOS OS ELEMENTOS EXCEPTO O QUADRADO VERDE



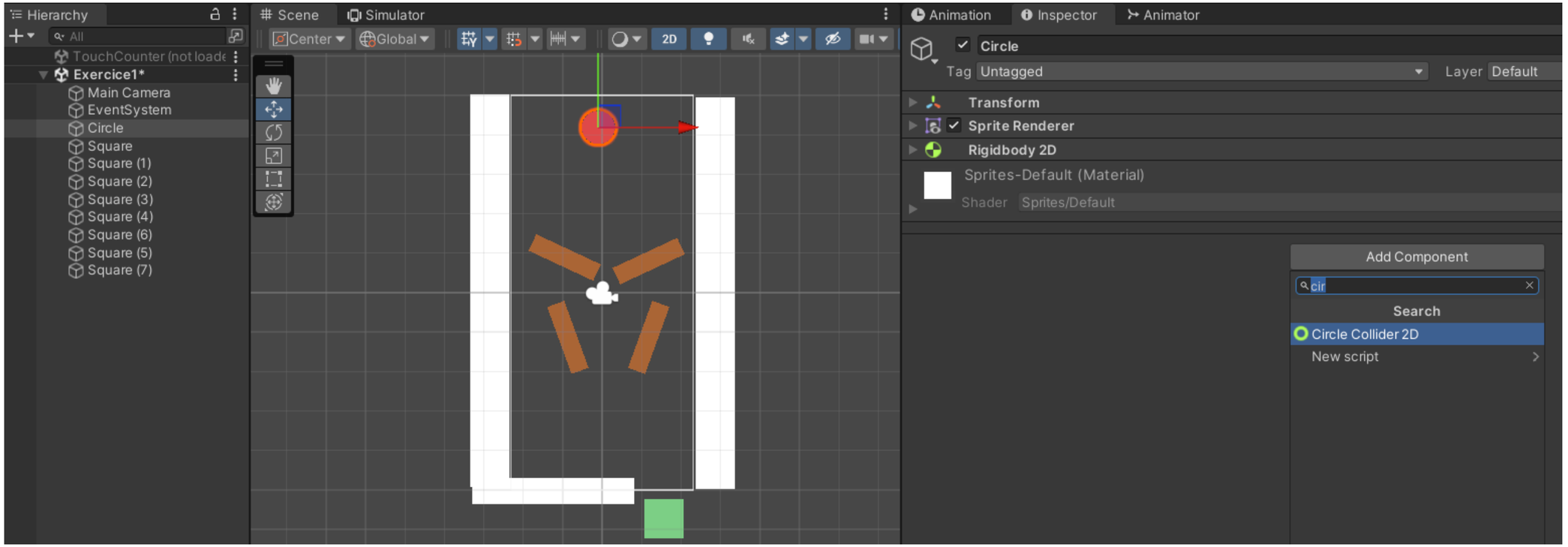
# MUDIFICA O BODY TYPE PARA ‘KINEMATIC’



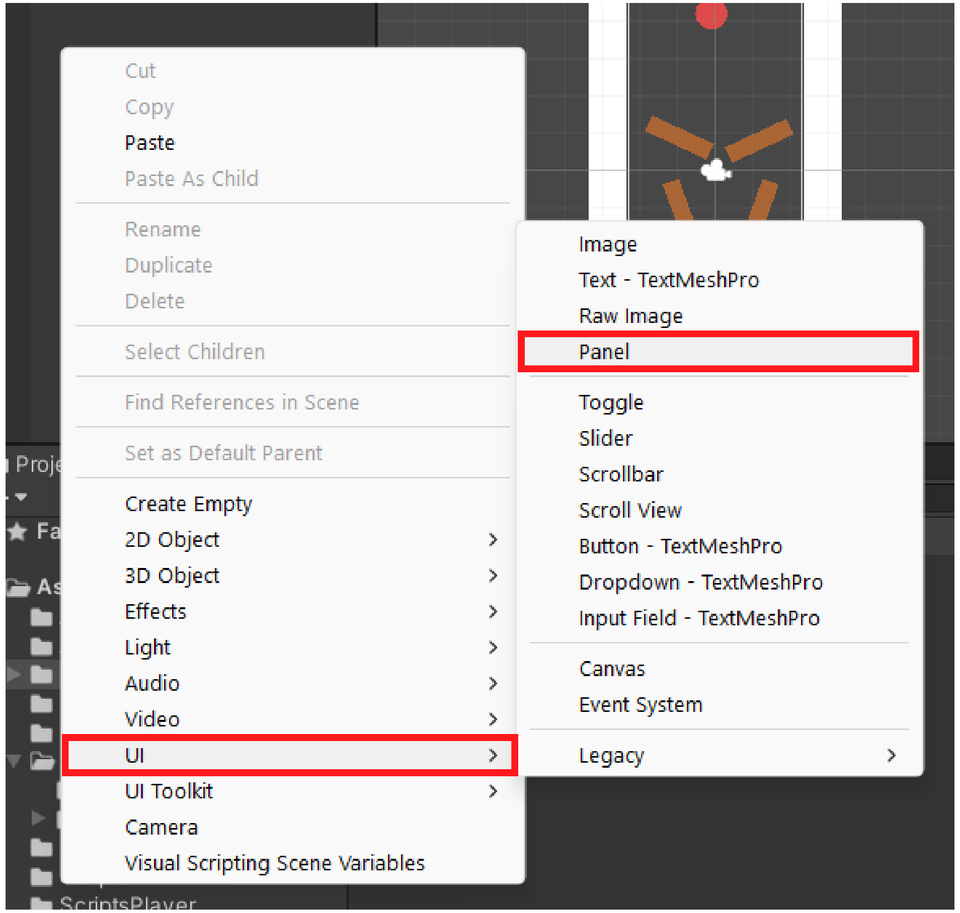
# ADICIONA O COMPONENTE ‘BOX COLLIDER 2D’ A TODOS OS ELEMENTOS EXCEPTO O CÍRCULO

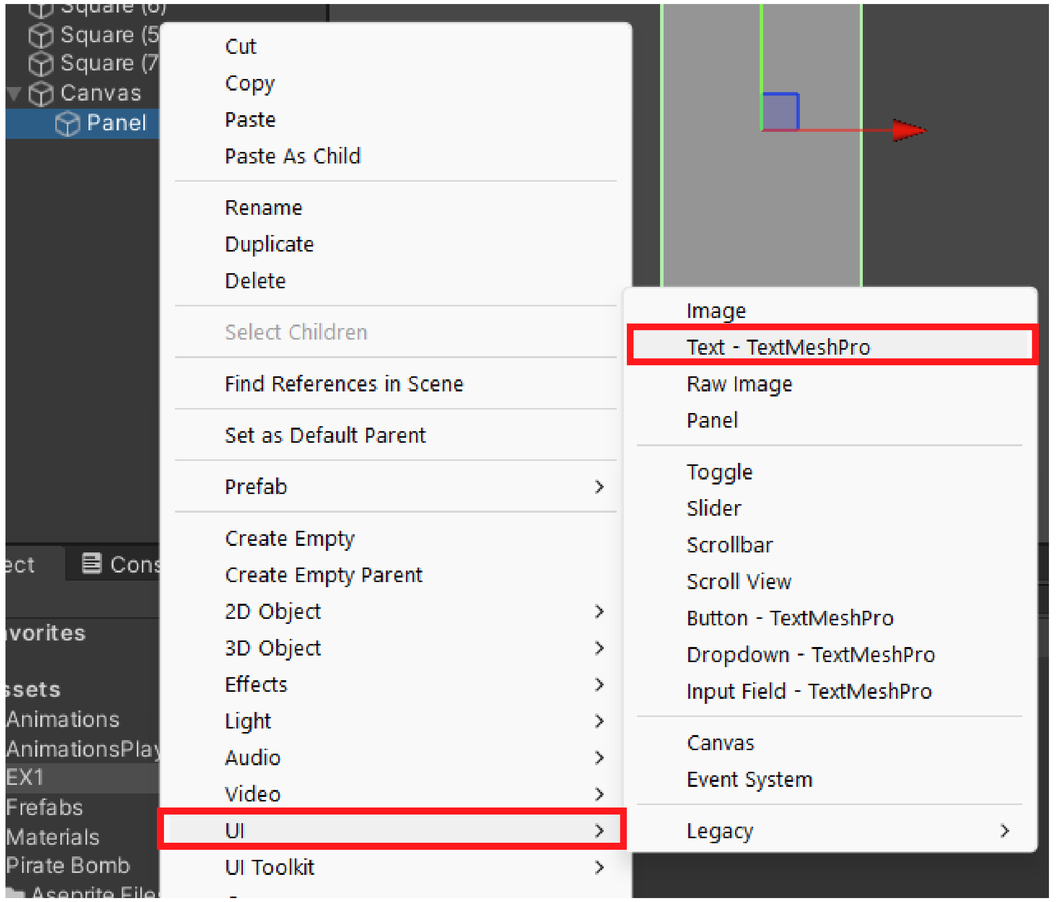


# ADICIONA O COMPONENTE ‘CIRCLE COLLIDER 2D’ AO CÍRCULO

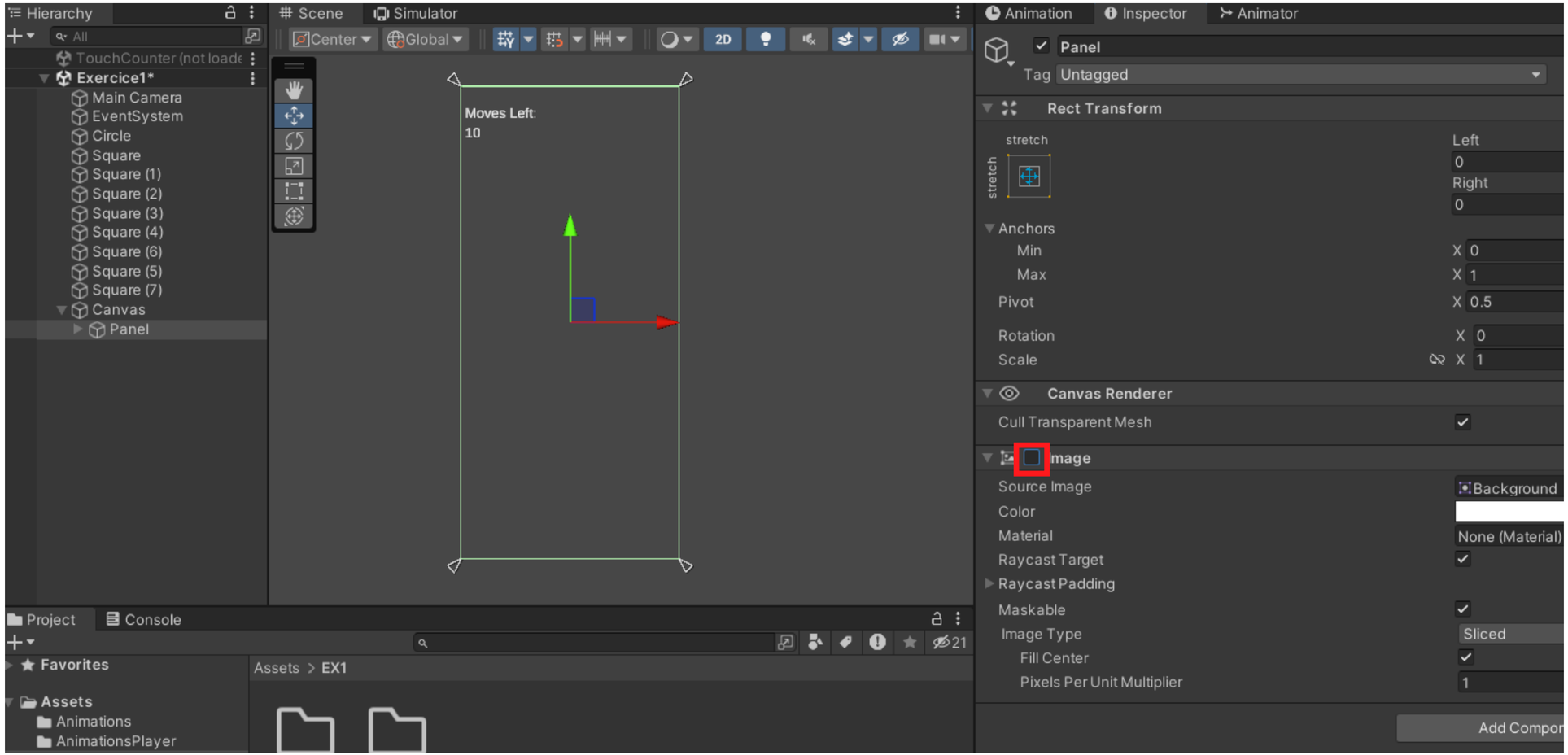


# ADICIONA UM PAINEL PARA O UI, COM DUAS CAIXAS DE TEXTO

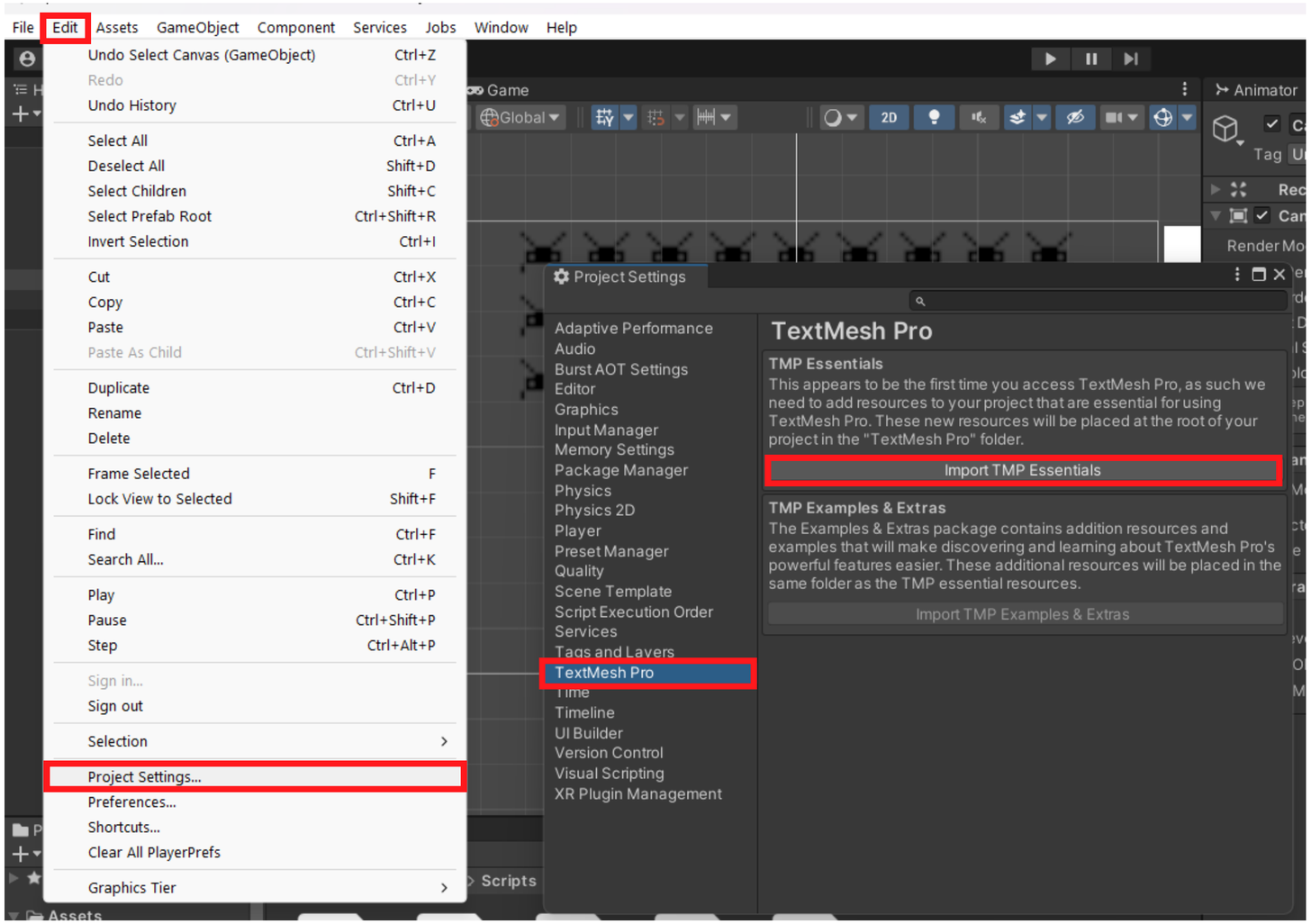


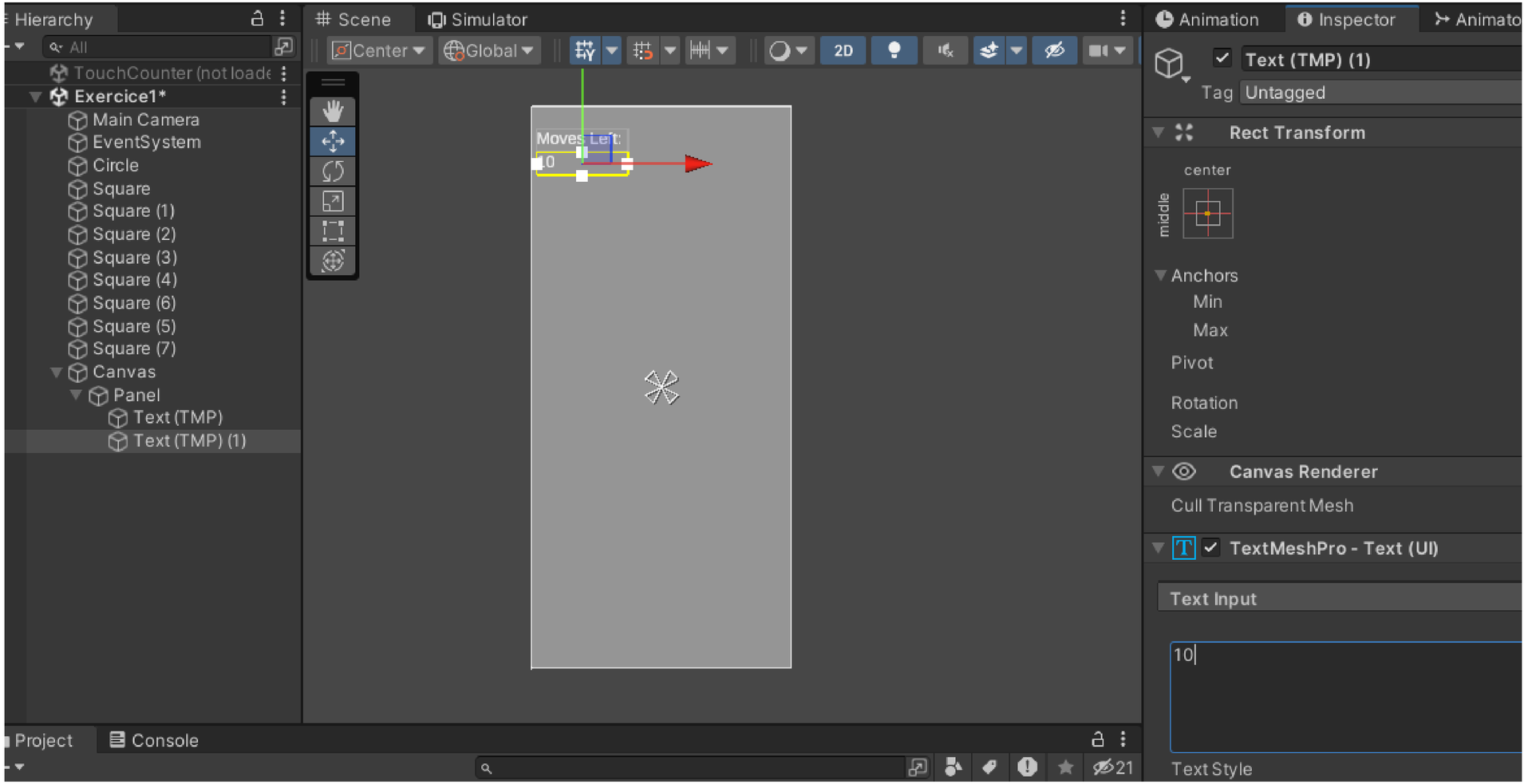
# DESATIVA O COMPONENTE DE IMAGEM DO PAINEL (OU REMOVE-O)



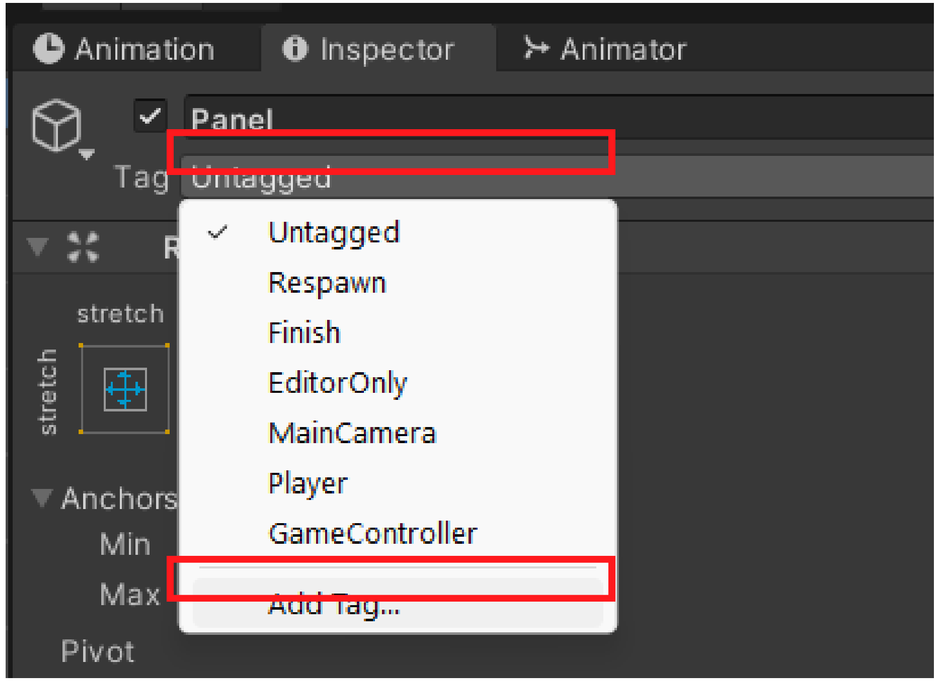
# IMPORTA OS RECURSOS PARA O TEXT MESH PRO

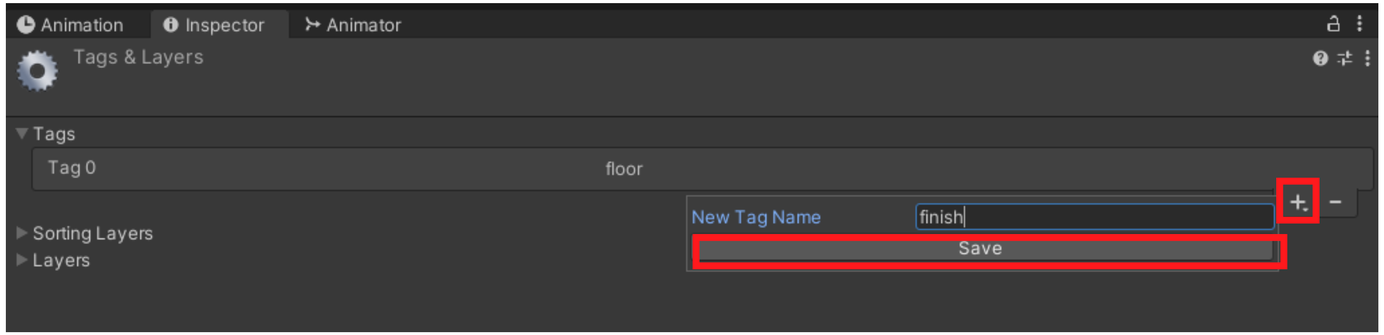


# MUDA O TEXTO PARA ‘MOVES LEFT:’ E ‘10’

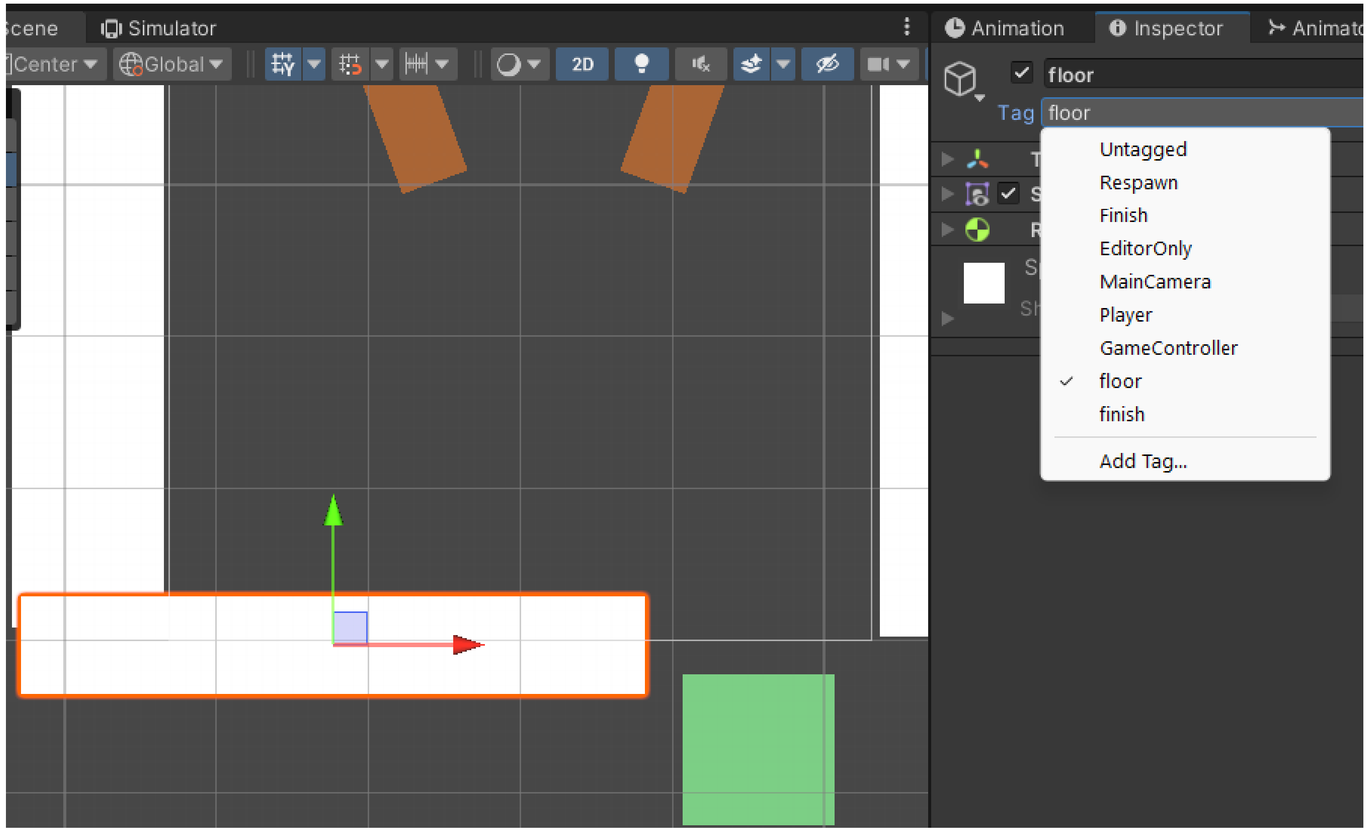


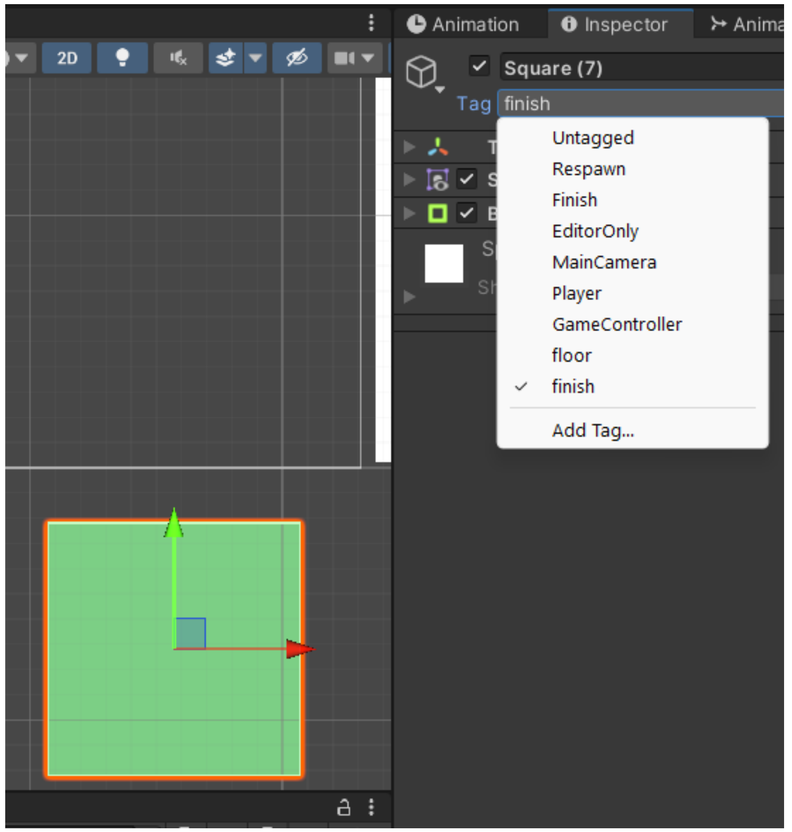
# ADICIONA AS TAGS: ‘FLOOR’ E ‘FINISH’

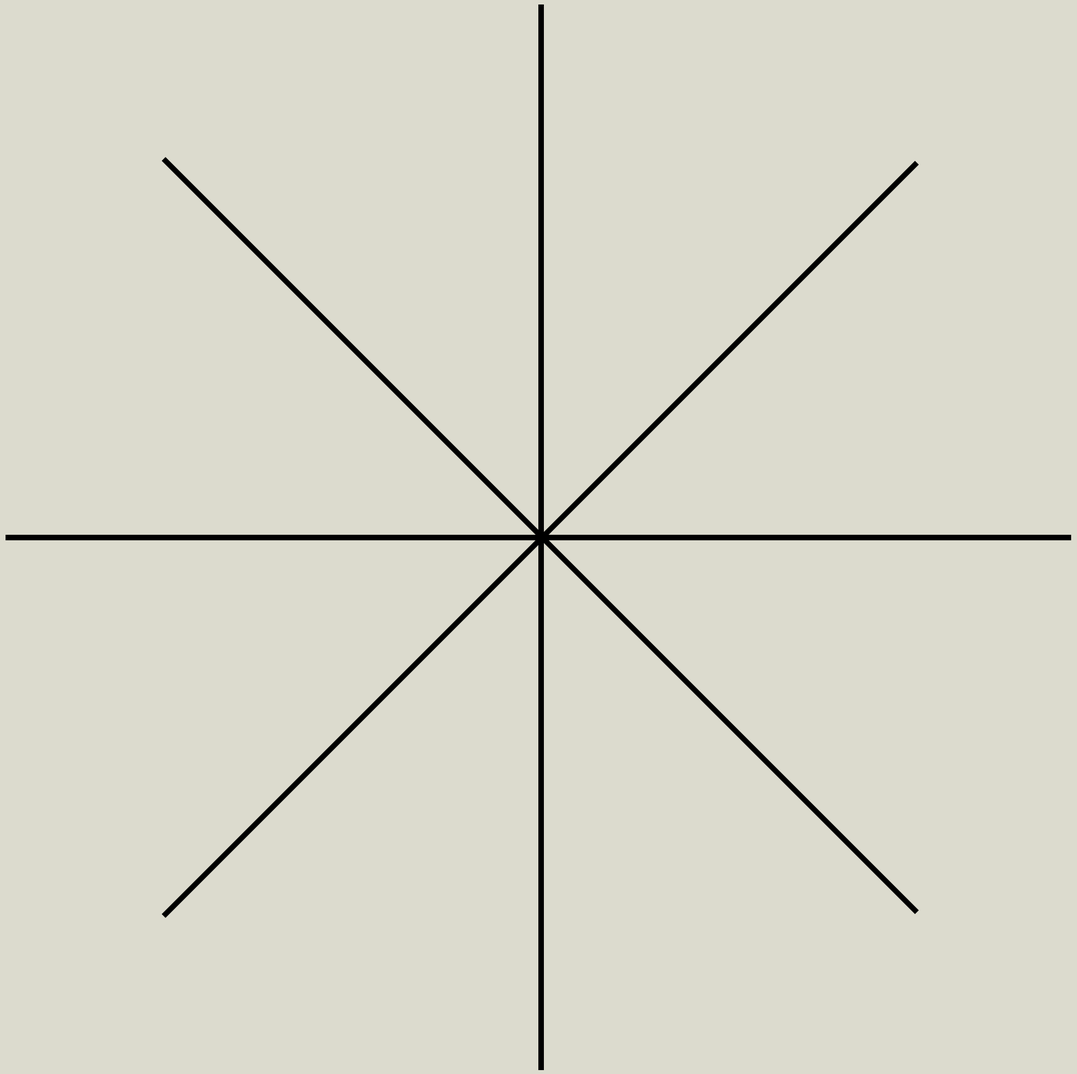


# APLICA AS TAGS À BARREIRA DO CHÃO E AO QUADRADO VERDE, RESPETIVAMENTE



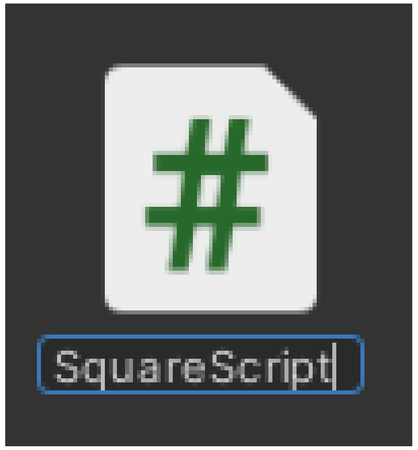



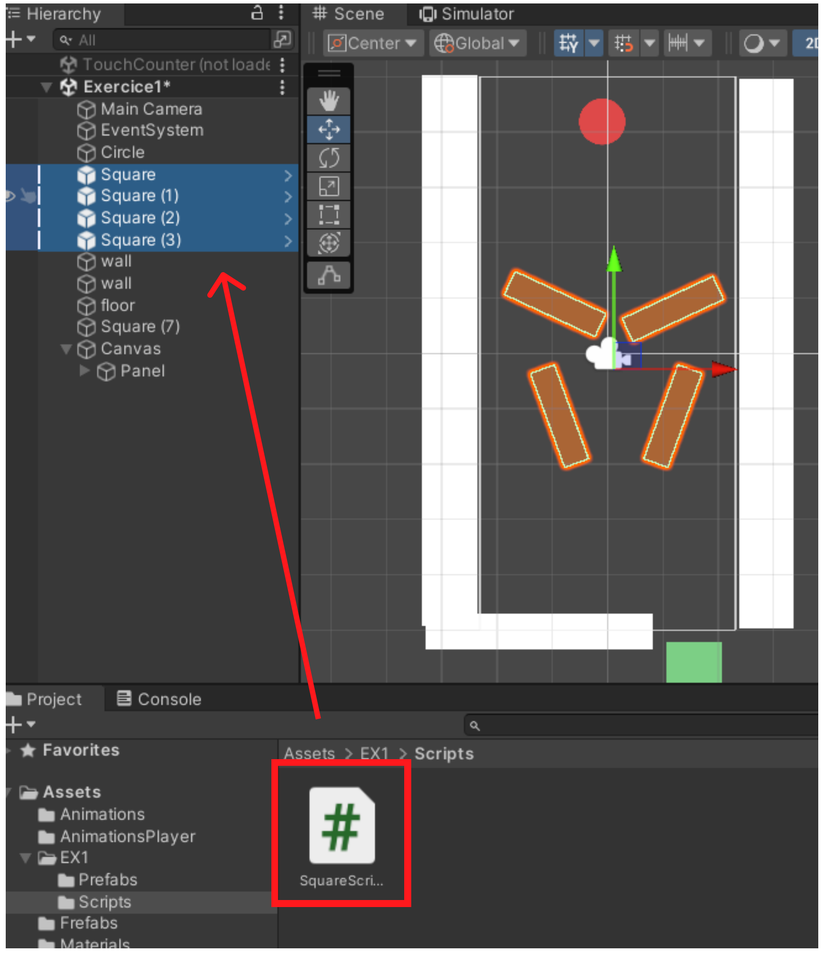


# 03
# Lista de Prefabs

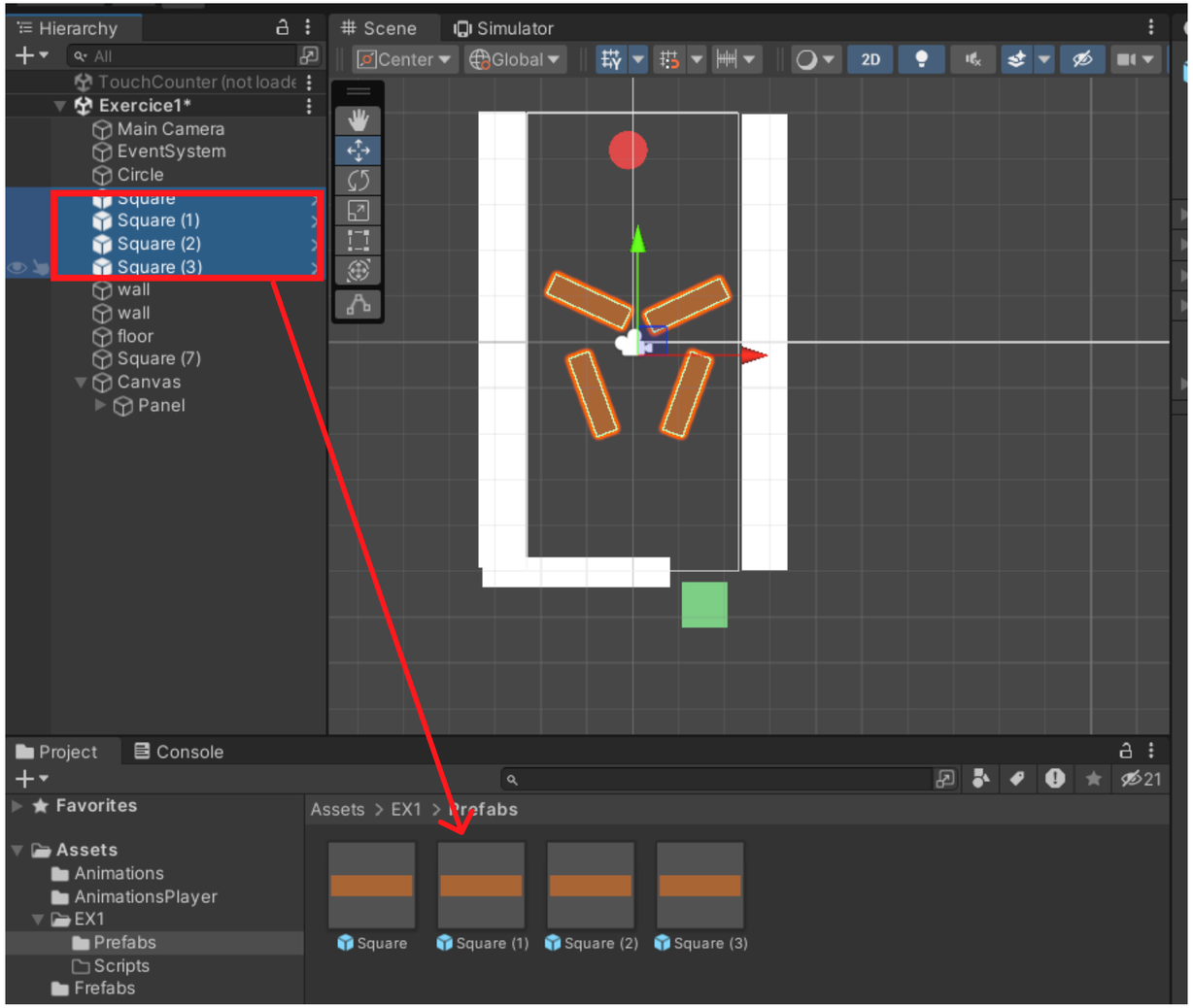
DANIELA MONTEIRO

# CRIA UM SCRIPT CHAMADO ‘SQUARESRIPT’ E APLICA-O A CADA UM DOS QUATRO RETÂNGULOS

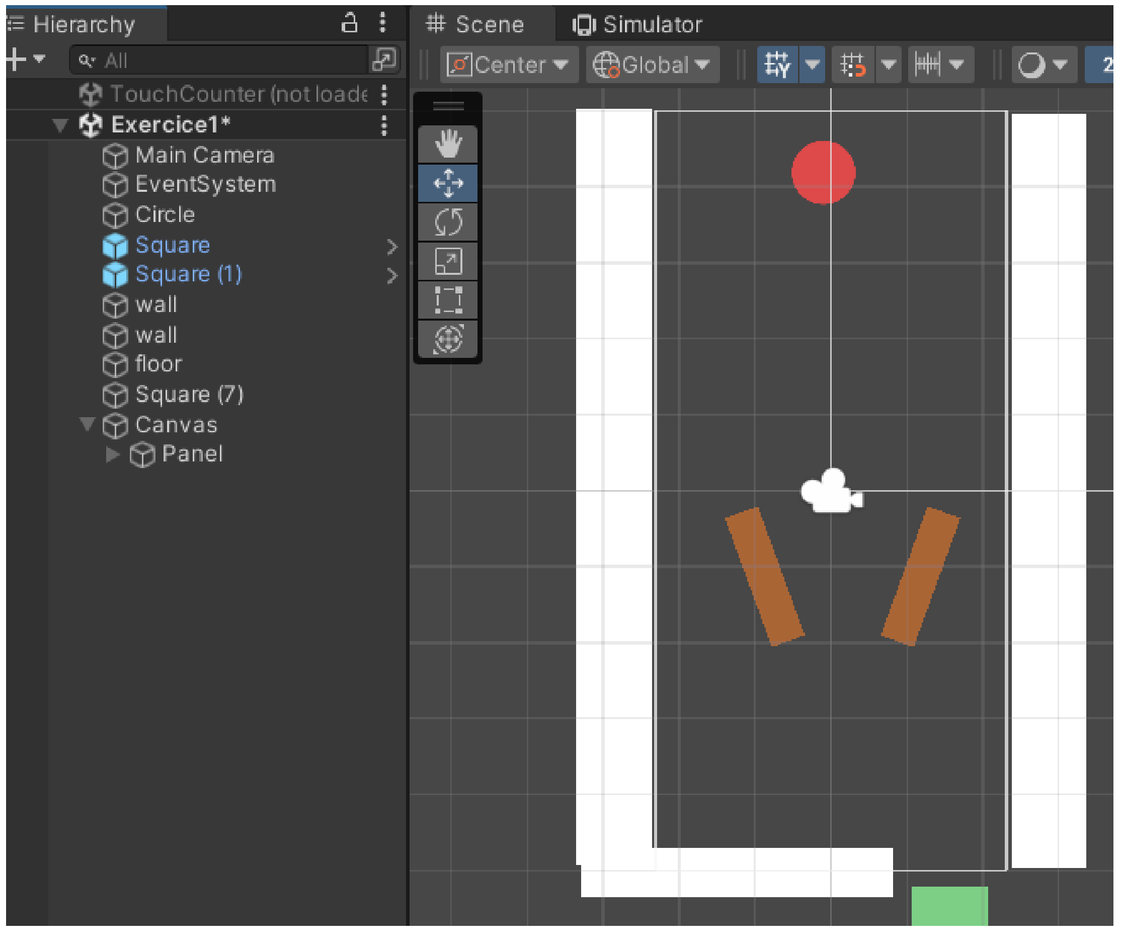


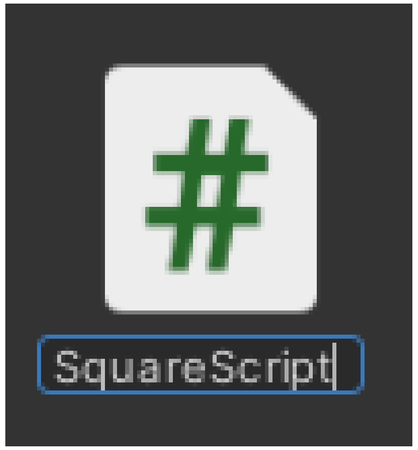
# ARRASTA OS RETÂNGULOS PARA A TUA PASTA ‘PREFABS’



# ELIMINA DOIS DOS RETÂNGULOS DA CENA

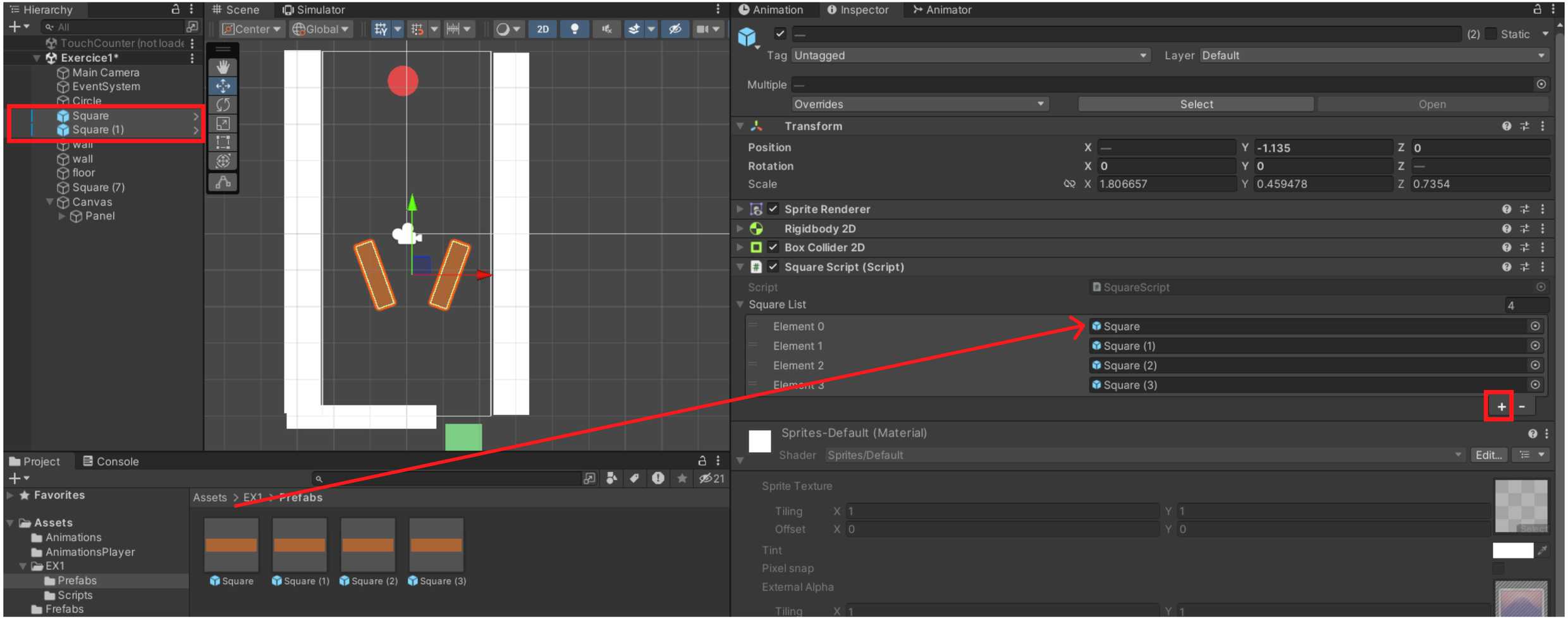


# ABRE O SCRIPT ‘SQUARESCRIPT’ E CRIA UMA LISTA DE OBJETOS




```
public class SquareScript : MonoBehaviour
{
    //list of objects
    [SerializeField] public List<GameObject> SquareList = new List<GameObject>();
```

# ADICIONA CADA UM DOS PREFABS À LISTA



# DETETA O TOQUE EM CADA UM DOS RETÂNGULOS

## !!!! NOTA: DEVES SEMPRE COMEÇAR A DETEÇÃO DE UM TOQUE COM ESTA CONDIÇÃO!

```
void Update()
{
    if (Input.touchCount > 0)
    {

    }
```

# DETETA O TOQUE EM CADA UM DOS RETÂNGULOS: VARIÁVEIS DO TOUCH E A SUA POSIÇÃO

```
if (Input.touchCount > 0)
{
    //saves the current touch
    Touch touch = Input.GetTouch(0);
    //saves the position of the touch in relation to the world
    Vector3 touchPos = Camera.main.ScreenToWorldPoint(touch.position);
```

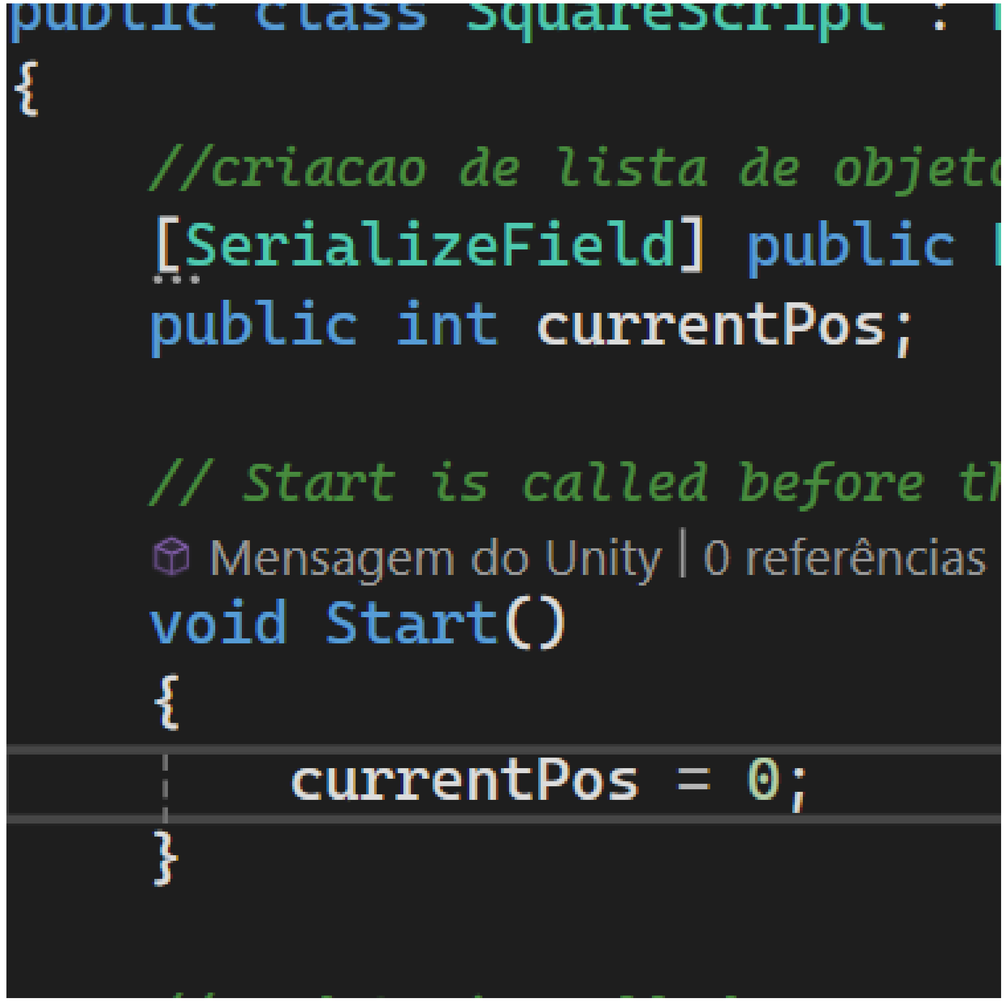
# DETETA O TOQUE EM CADA UM DOS RETÂNGULOS: VERIFICAR A POSIÇÃO DO TOQUE EM RELAÇÃO AO RETÂNGULO

```
if (Input.touchCount > 0)
{
    //saves the current touch
    Touch touch = Input.GetTouch(0);
    //saves the position of the touch in relation to the world
    Vector3 touchPos = Camera.main.ScreenToWorldPoint(touch.position);

    //a simple touch (as soon as it beggins)
    if (touch.phase == TouchPhase.Began)
    {
        if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
        {
            print("you are touching the square!");
        }
    }
}
```

# MUDA A ROTAÇÃO DO RETÂNGULO PARA A MESMA QUE O PRIMEIRO ELEMENTO DA LISTA DE QUADRADOS

```
if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
{
    print("you are touching the square!");
    this.gameObject.transform.rotation = SquareList[0].transform.rotation;
}
```

# GUARDA A POSIÇÃO NA LISTA (COMEÇA COM 0)

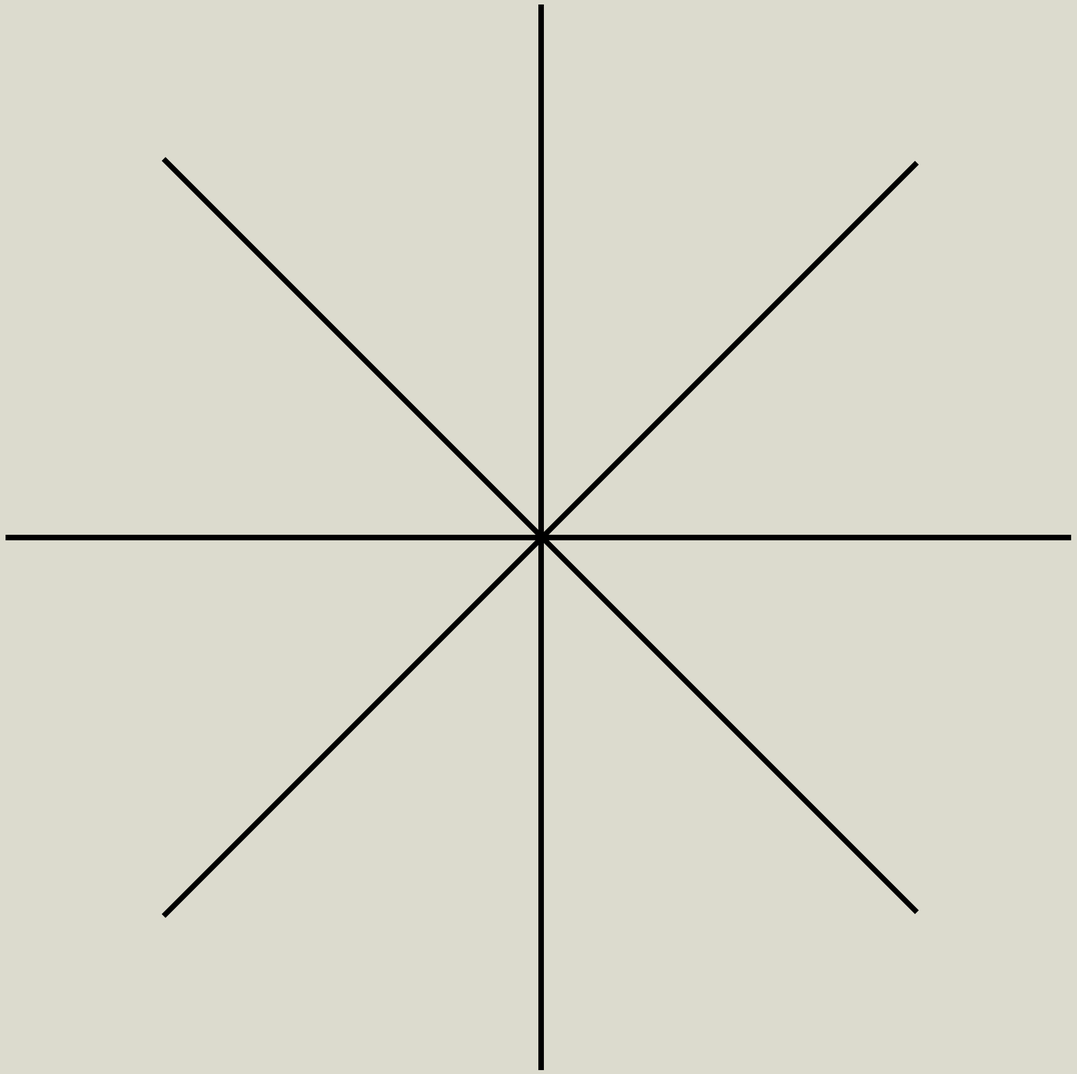


# MUDA A ROTAÇÃO DO RETÂNGULO PARA QUE PRECORRA TODA A LISTA

```
if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
{
    print("you are touching the square!");
    this.gameObject.transform.rotation = SquareList[currentPos].transform.rotation;
    currentPos++;
}
```

# QUANDO CHEGAR AO QUARTO ELEMENTO, VOLTA AO PRIMEIRO

```
if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
{
    print("you are touching the square!");
    if (currentPos > 3)
    {
        currentPos = 0;
    }
    this.gameObject.transform.rotation = SquareList[currentPos].transform.rotation;
    currentPos++;
}
```



# 04
# Drag and Drop

DANIELA MONTEIRO

# CRIA UMA VARIÁVEL QUE GUARDA SE O TOQUE FOI NO RETÂNGULO OU NÃO

```
public class SquareScript : MonoBehaviour
{
    //list of objects
    [SerializeField] public List<GameObject> SquareL
    public int currentPos;

    //saves whether or not the touch is on the ractangle
    public bool moveAllowed = false;
```

# ACIONA A VARIÁVEL COM O TOQUE

```
//a simple touch (as soon as it beggins)
if (touch.phase == TouchPhase.Began)
{
    if (GetComponent<Collider2D>() == Physics2D.
    {
        print("you are touching the square!");
        moveAllowed = true;
        if (currentPos > 3)
        {
```

# DETETA O MOVIMENTO

```
//a simple touch (as soon as it beggins)
if (touch.phase == TouchPhase.Began)
{
    if (GetComponent<Collider2D>() == Ph
}
//for drag
if (touch.phase == TouchPhase.Moved)
{

}
```

# ADICIONA A CONDIÇÃO COM A VARIÁVEL

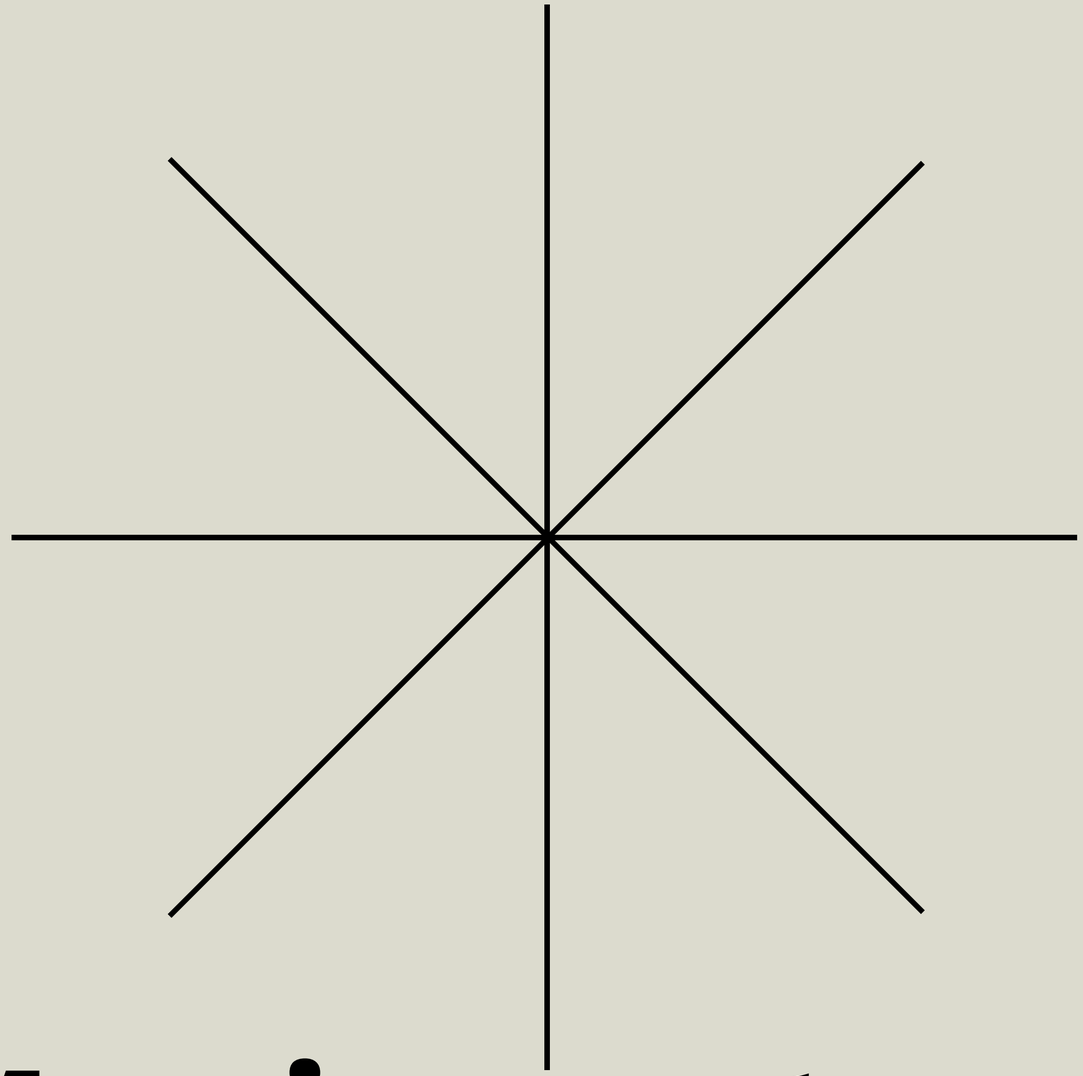
```
//for drag
if (touch.phase == TouchPhase.Moved && moveAllowed)
{

}
```

# MUDA A POSIÇÃO PARA ACOMPANHAR O TOQUE

```
//for drag
if (touch.phase == TouchPhase.Moved && moveAllowed)
{
    this.transform.position =
        Camera.main.ScreenToWorldPoint(new Vector3(touch.position.x, touch.position.y, 0));
}
```

# DESATIVA A VARIÁVEL APÓS O FIM DO TOQUE

```
//for drag
if (touch.phase == TouchPhase.Moved && moveAllowed)
{
    this.transform.position =
        Camera.main.ScreenToWorldPoint(new Vector3(tou
}
//for the drop
if (touch.phase == TouchPhase.Ended && moveAllowed)
{
    moveAllowed = false;
}
```



# 05
# Contador de Movimentos

DANIELA MONTEIRO

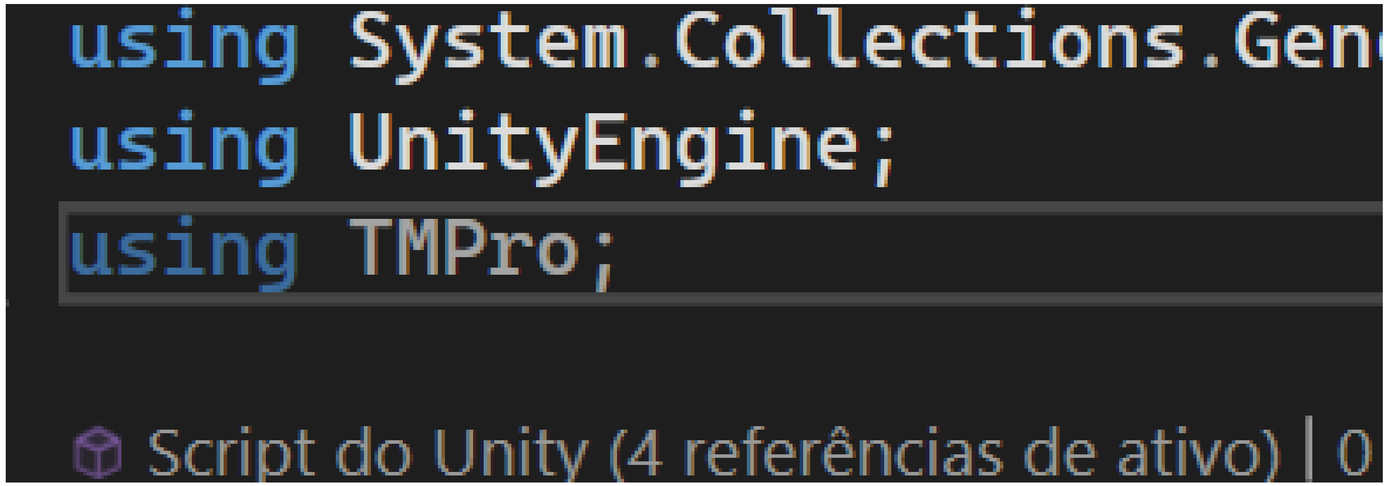
# ADICIONA UMA VARIÁVEL QUE GUARDE OS MOVIMENTOS FEITOS

```
public class SquareScript : MonoBehaviour
{
    //list of objects
    [SerializeField] public List<GameObject
    public int currentPos;

    public int movesLeft = 10;

    // Start is called before the first frame up
```

# DIMINUI A VARIÁVEL NO FINAL DE CADA TOQUE

```
//for the drop
if (touch.phase == TouchPhase.Ended && moveAllowed)
{
    movesLeft--;
    moveAllowed = false;
}
```

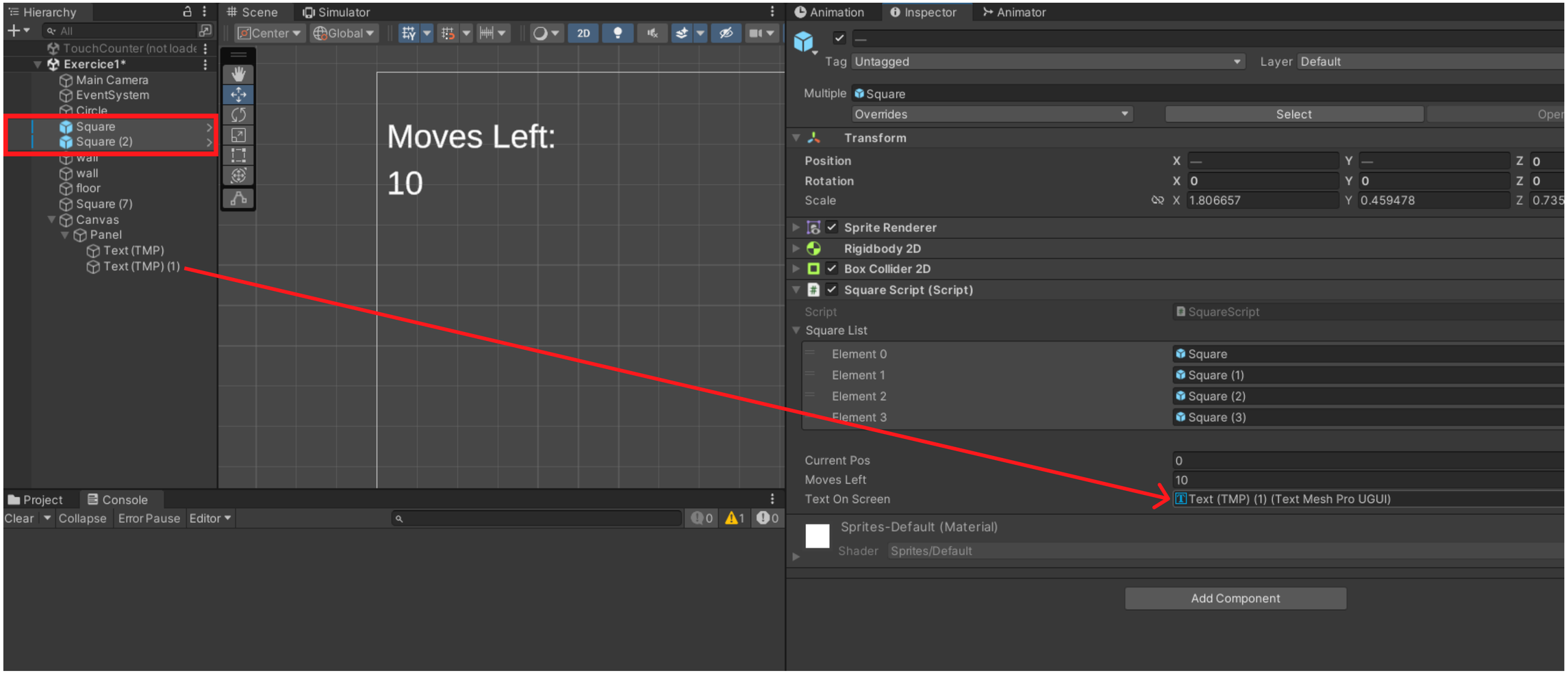
# IMPORTA A BIBLIOTECA TMPRO



# ADICIONA UMA VARIÁVEL PARA O TEXTO NO PAINEL

```
public int movesLeft = 10;
//for the text on screen to change
[SerializeField] public TextMeshProUGUI textOnScreen;

// Start is called before the first frame update
```

# ASSOCIA O TEXTO À VARIÁVEL A PARTIR DO EDITOR

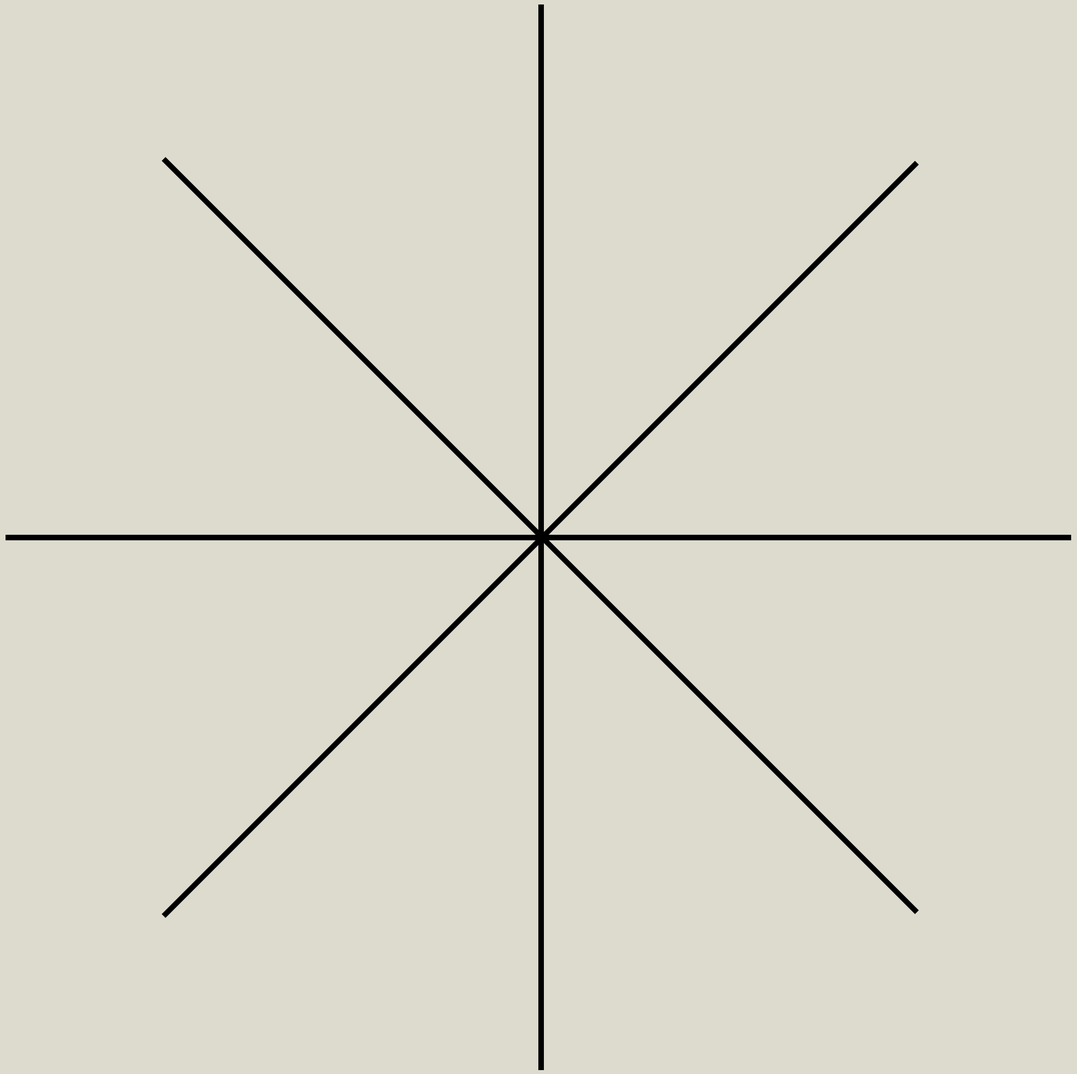


# MUDIFICA O TEXTO NO FINAL DE CADA TOQUE

```
//for the drop
if (touch.phase == TouchPhase.Ended && moveAllowed)
{
    movesLeft--;
    moveAllowed = false;
    textOnScreen.text = movesLeft.ToString();
}
```

# ADICIONA UMA CONDIÇÃO PARA A FASE “MOVED” PARA QUE OS MOVIMENTOS NÃO ULTRAPASSEM OS 10

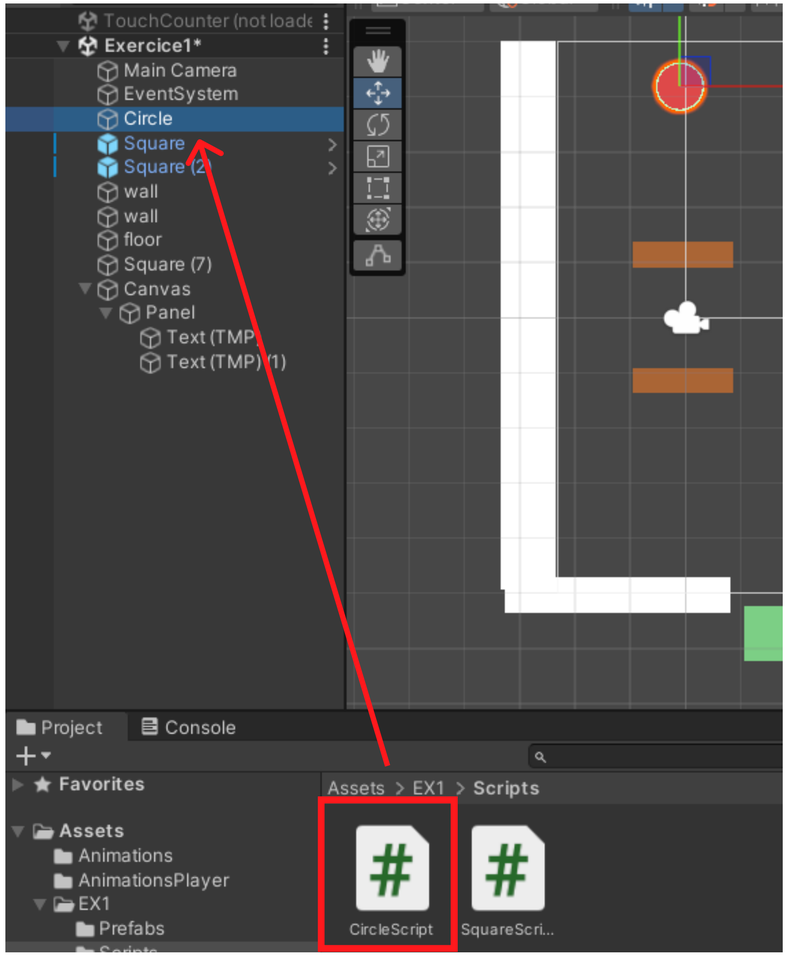
```
//for drag
if (touch.phase == TouchPhase.Moved && moveAllowed && movesLeft > 0)
{
    this.transform.position =
        Camera.main.ScreenToWorldPoint(new Vector3(touch.position.x, touch.position.y, 0));
}
//for the drop
```



# 06
# Fim de Jogo

DANIELA MONTEIRO

# CRIA UM SCRIPT CHAMADO ‘CIRCLESRIPT’ E APLICA-O AO CÍRCULO

# DETETA O TOQUE NO CÍRCULO, DA MESMA FORMA QUE FOI FEITA PARA OS RETÂNGULOS

```
Mensagem do Unity | 0 referências
void Update()
{
    if (Input.touchCount > 0)
    {
        //saves the current touch
        Touch touch = Input.GetTouch(0);
        //saves the position of the touch in relation to the world
        Vector3 touchPos = Camera.main.ScreenToWorldPoint(touch.position);

        //a simple touch (as soon as it beggins)
        if (touch.phase == TouchPhase.Began)
        {
            if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
            {
                print("you are touching the circle!");
            }
        }
    }
}
```

# FAZ COM QUE O CÍRCULO CAIA (MUDANDO O BODY TYPE DO SEU COMPONENTE RIGIDBODY2D PARA DYNAMIC)

```
if (GetComponent<Collider2D>() == Physics2D.OverlapPoint(touchPos))
{
    print("you are touching the circle!");
    GetComponent<Rigidbody2D>().bodyType = RigidbodyType2D.Dynamic;
}
```

# CRIA UM MÉTODO PARA O FIM DE JOGO

```
void Update()
{
    if (Input.touchCount > 0)...
}

0 referências
public void EndOfGame(string state)
{
    if (state == "win")
    {
        print("You Won!");
    }
    else
    {
        print("You Lost!");
    }
}
```

# DETETA A COLISÃO COM O QUADRADO VERDE (FINISH LINE)

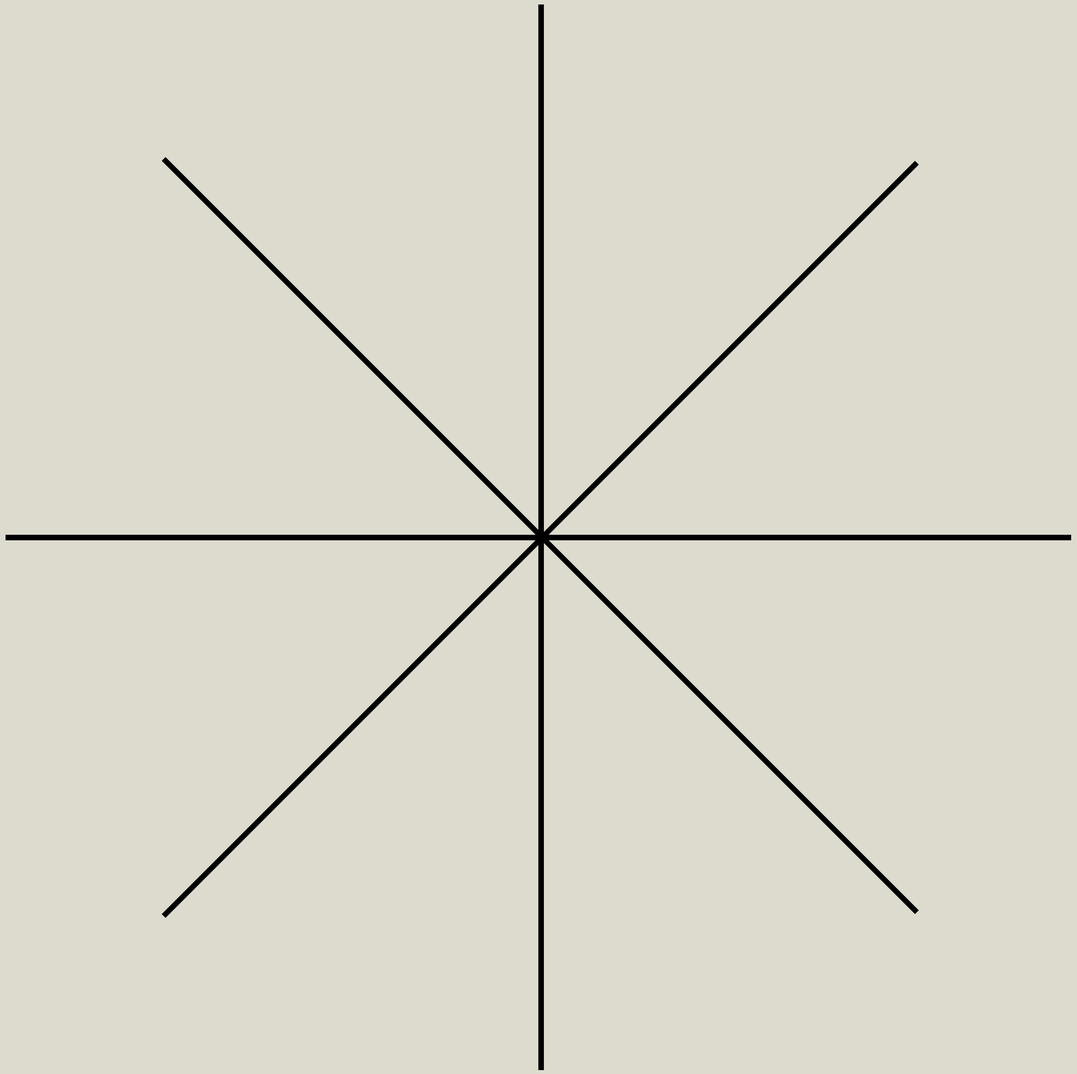
```
public void EndOfGame(string state)
{
    if (state == "win")...
    else...
    panelUI.SetActive(true);
    Time.timeScale = 0;
}


Mensagem do Unity | 0 referências
private void OnCollisionEnter2D(Collision2D collision)
{
    if (collision.gameObject.tag == "finish")
    {
        EndOfGame("win");
    }

}
```

# DETETA A COLISÃO COM O CHÃO

```
private void OnCollisionEnter2D(Collision2D collision)
{
    if (collision.gameObject.tag == "finish")
    {
        EndOfGame("win");
    }
    else if (collision.gameObject.tag == "floor")
    {
        EndOfGame("loose");
    }
}
```

# PAUSA O JOGO

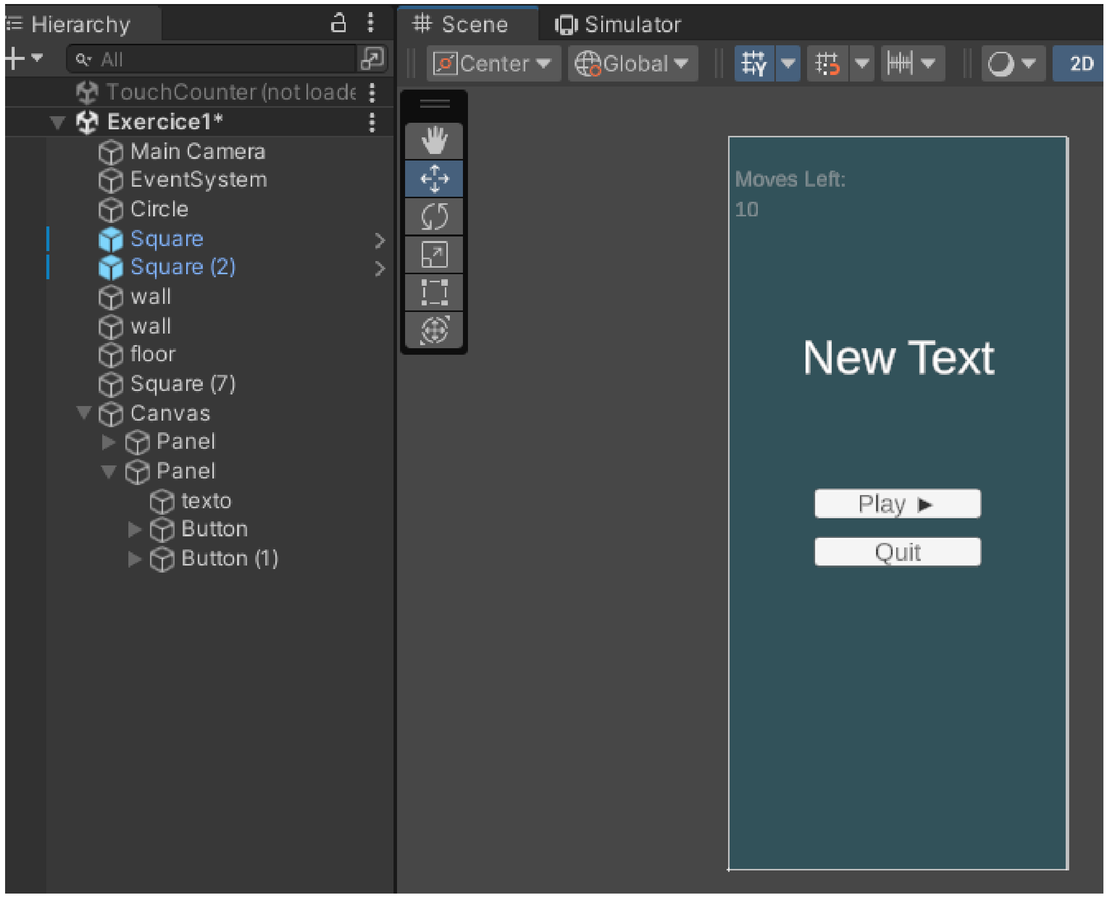
```
2 referências
public void EndOfGame(string state)
{
    if (state == "win")
    {
        print("You Won!");
    }
    else
    {
        print("You Lost!");
    }
    Time.timeScale = 0;
}
```



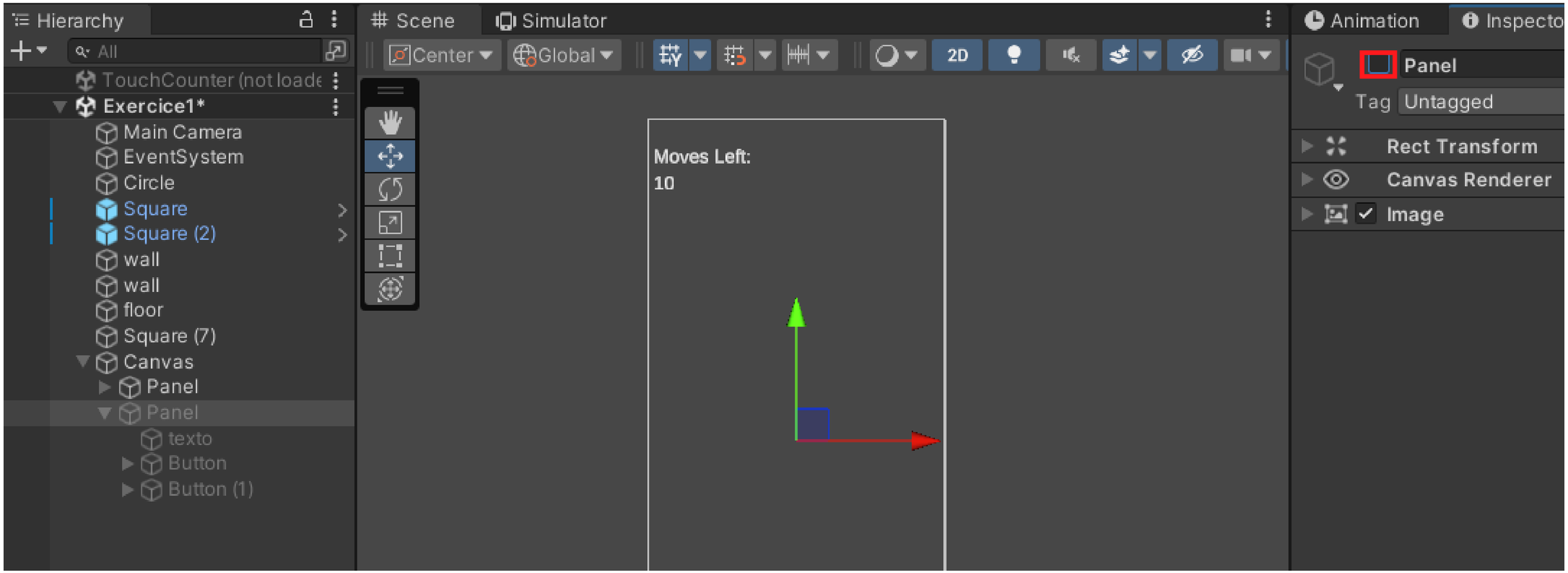
# 08
# Menu de Jogo

DANIELA MONTEIRO

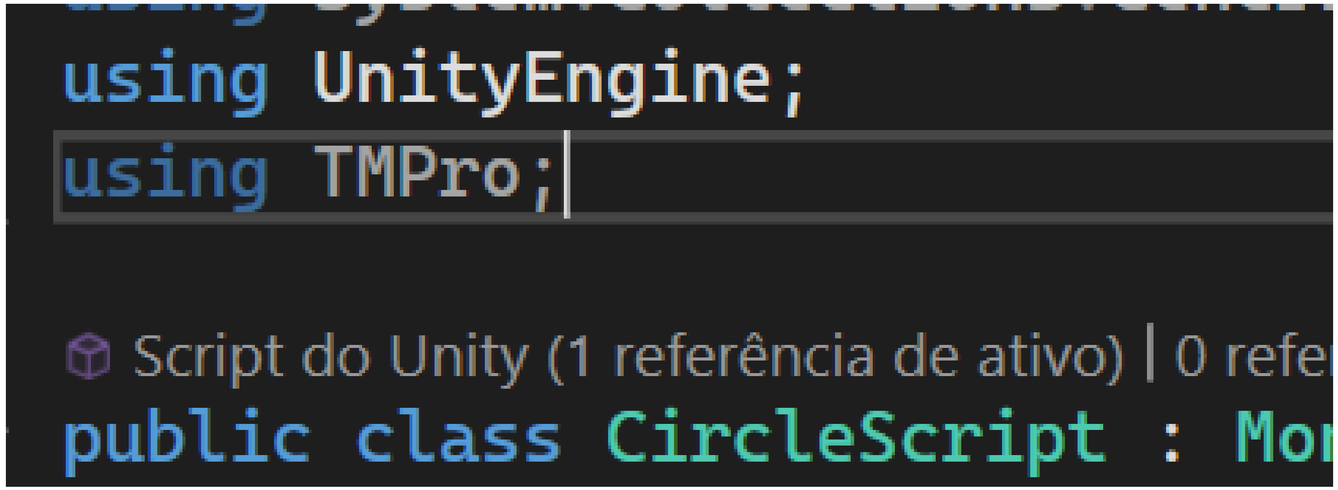
# ADICIONA UM PAINEL COM UM TEXTO E DOIS BOTÕES (JOGAR E SAIR)



# DESATIVA O NOVO PAINEL



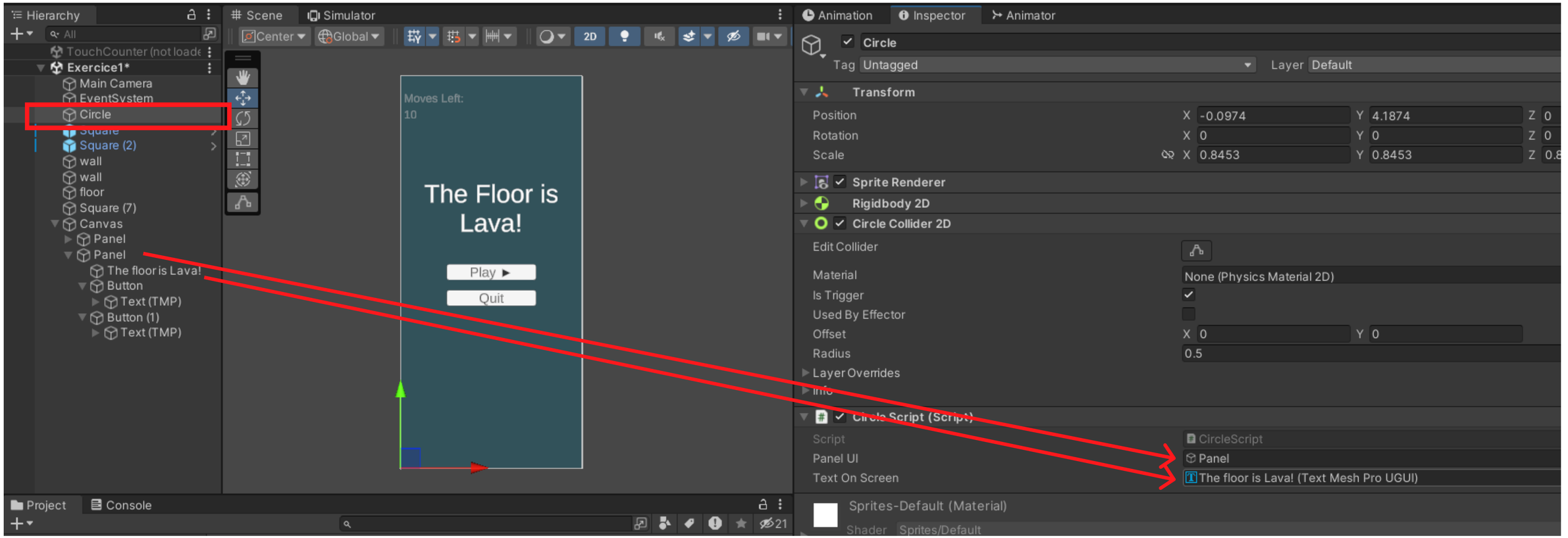
# IMPORTA A BIBLIOTECA ‘TMPRO’ NO SCRIPT ‘CIRCLESCRIPT’



# CRIA DUAS VARIÁVEIS PARA O PAINEL DO MENU E O TEXTO

```
public class CircleScript : MonoBehaviour
{
    [SerializeField] public GameObject panelUI;
    [SerializeField] public TextMeshProUGUI textOnScreen;

    // Start is called before the first frame update
    Mensagem do Unity | – referências
```

# ASSOSSIA AS VARIÁVEIS A CADA UM DOS ELEMENTOS EM CENA



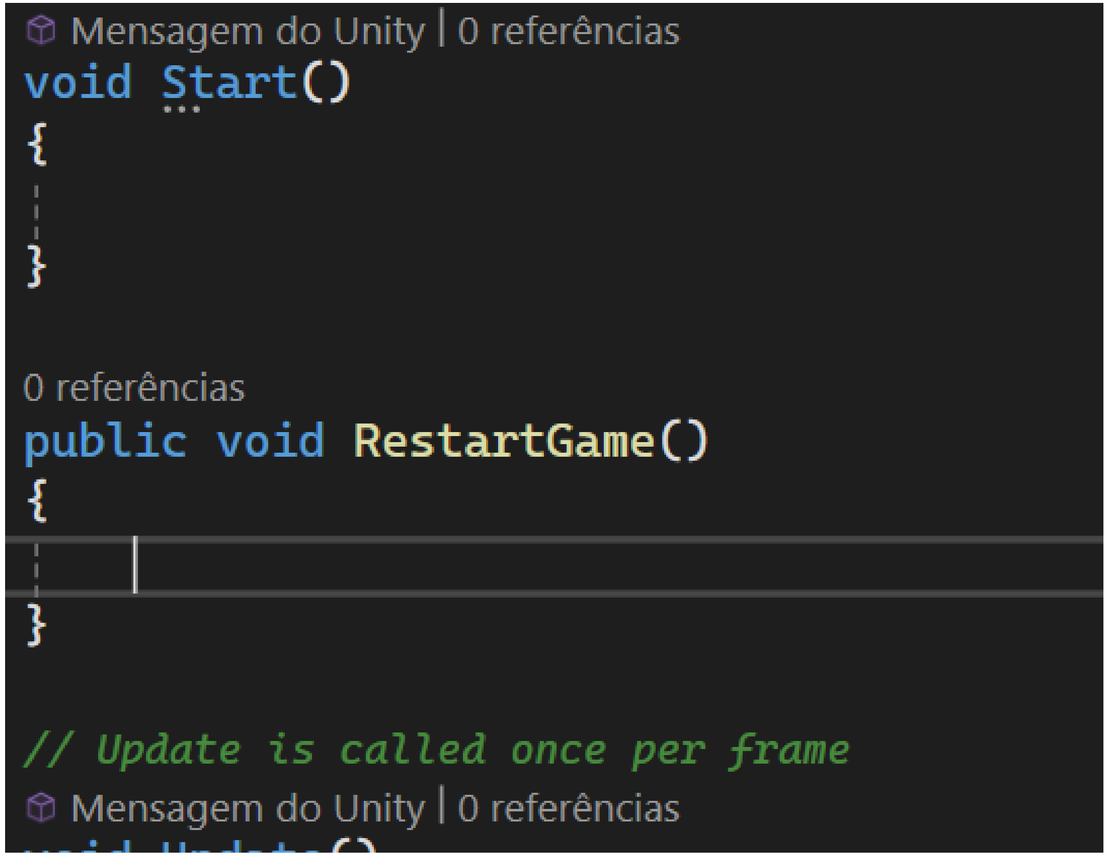
# ATIVA O PAINEL NO FIM DO JOGO

```
2 referências
public void EndOfGame(string state)
{
    if (state == "win")
    {
        print("You Won!");
    }
    else
    {
        print("You Lost!");
    }
    panelUI.SetActive(true);
    Time.timeScale = 0;
}
```

# MUDA O TEXTO NO PAINEL DE ACORDO COM O ESTADO (WIN/LOOSE)

```
2 referências
public void EndOfGame(string state)
{
    if (state == "win")
    {
        print("You Won!");
        textOnScreen.text = "You Won!";
    }
    else
    {
        print("You Lost!");
        textOnScreen.text = "You Lost!";
    }
    panelUI.SetActive(true);
    Time.timeScale = 0;
}
```

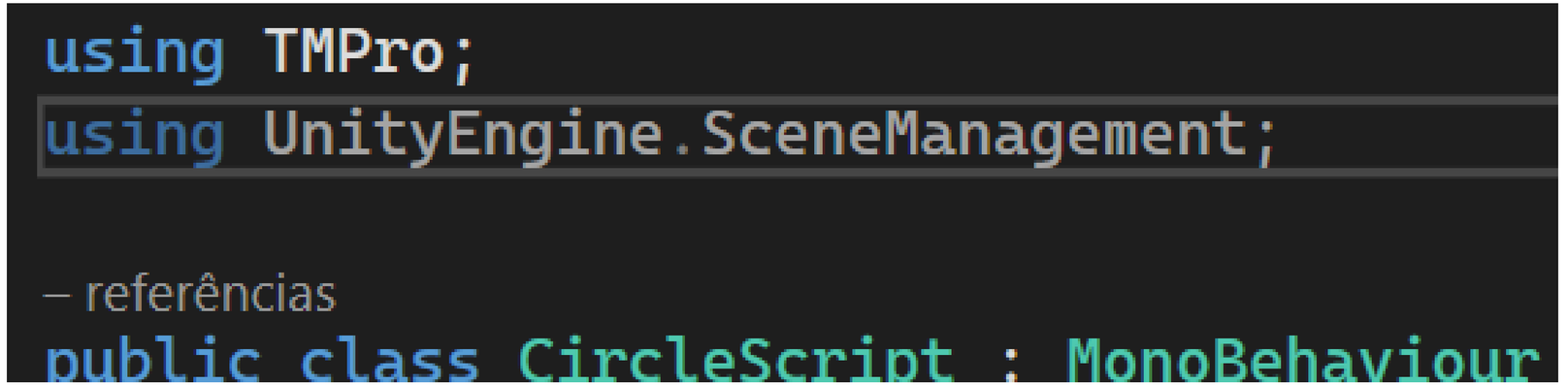
# CRIA UM MÉTODO PARA COMEÇAR A JOGAR OU RECOMEÇAR O JOGO



# RETOMA O TEMPO DO JOGO (UNPAUSE)

```
0 referências
public void RestartGame()
{
    Time.timeScale = 1;
}
```

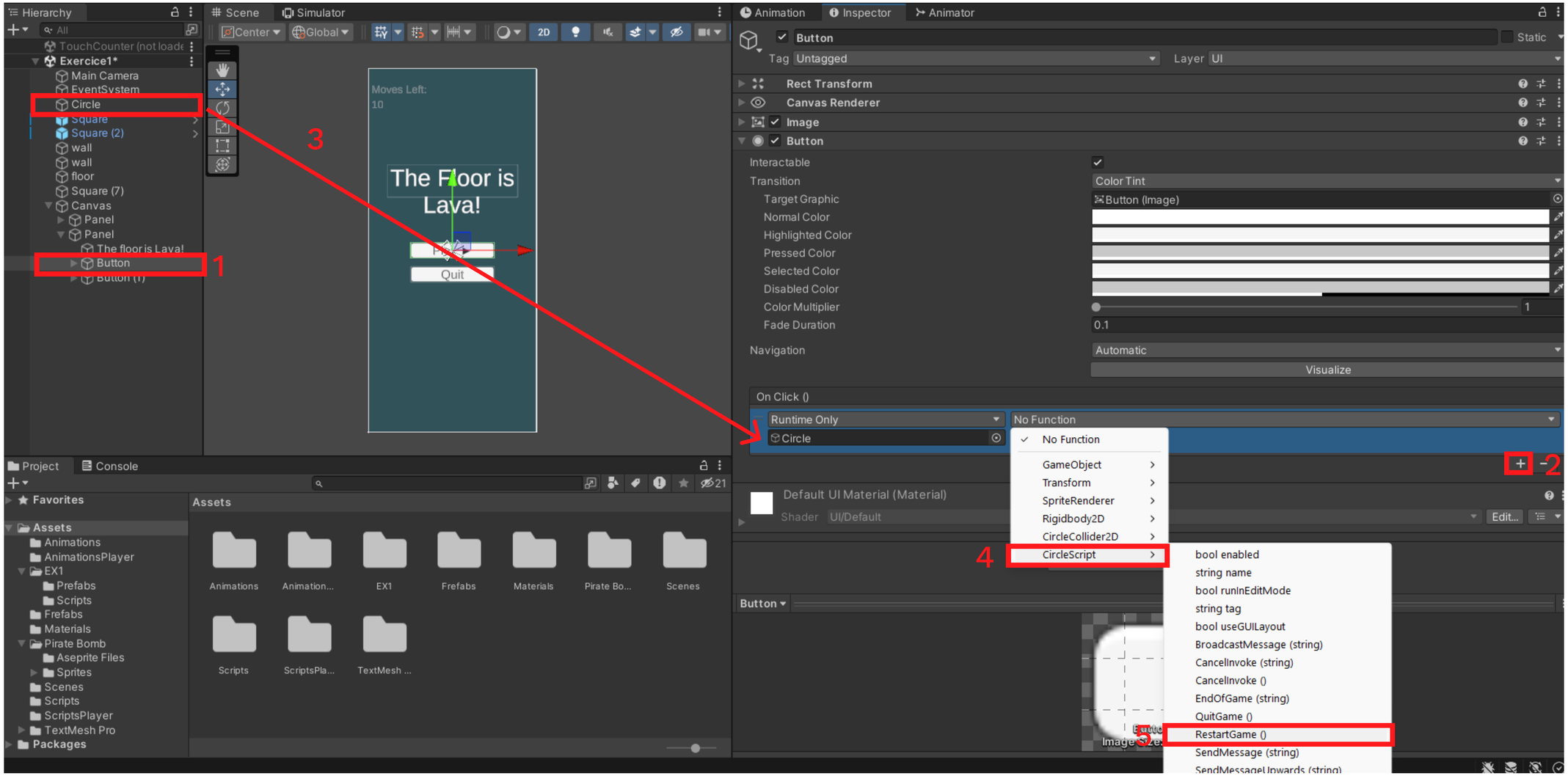
# IMPORTA A BIBLIOTECA DE GESTÃO DE CENAS



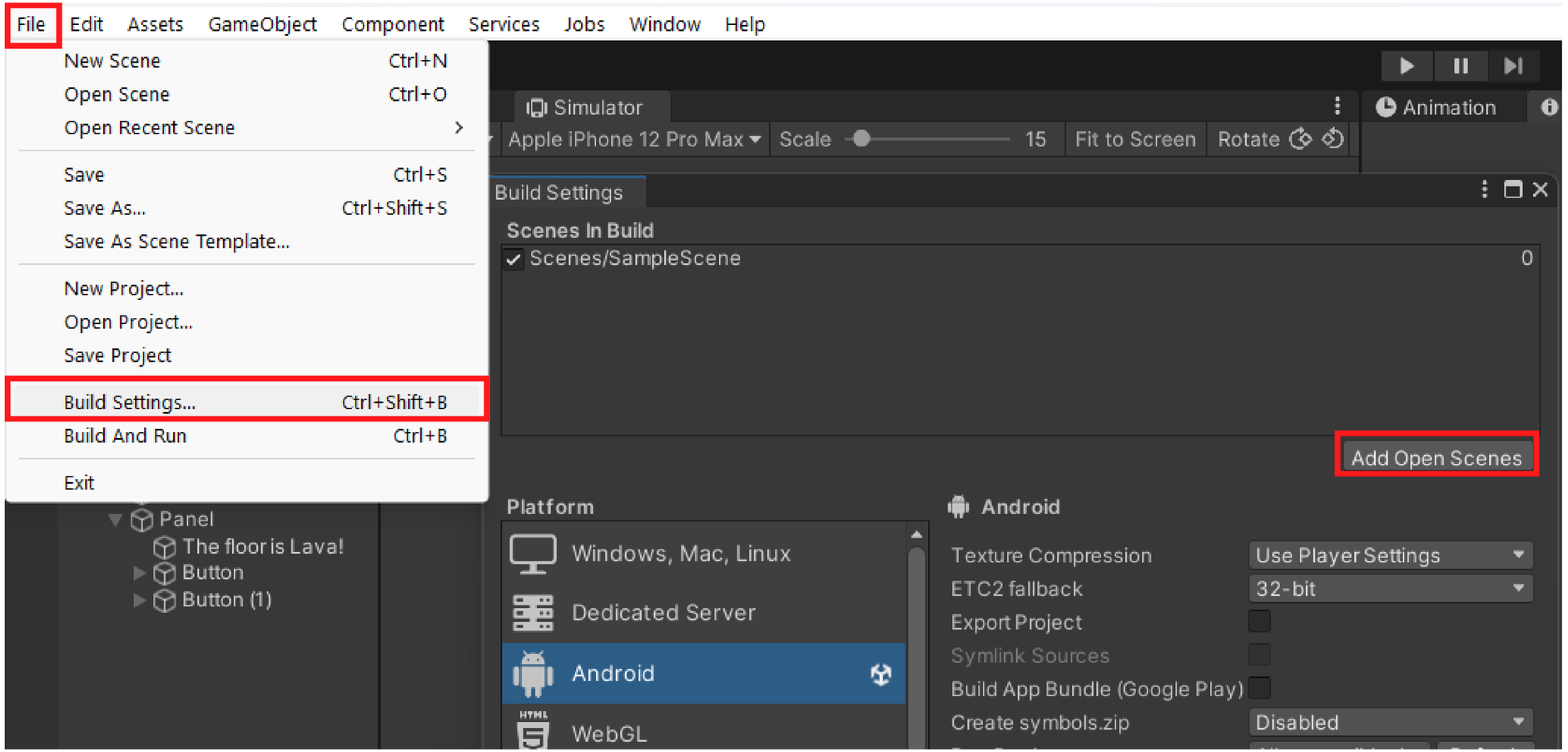
# REINICIA A CENA

```
0 referências
public void RestartGame()
{
    SceneManager.LoadScene(0);
    //alternative: name of the scene:
    SceneManager.LoadScene("SampleScene");
    Time.timeScale = 1;

}
```

# CONECTA O BOTAO DE ‘PLAY’ COM O MÉTODO



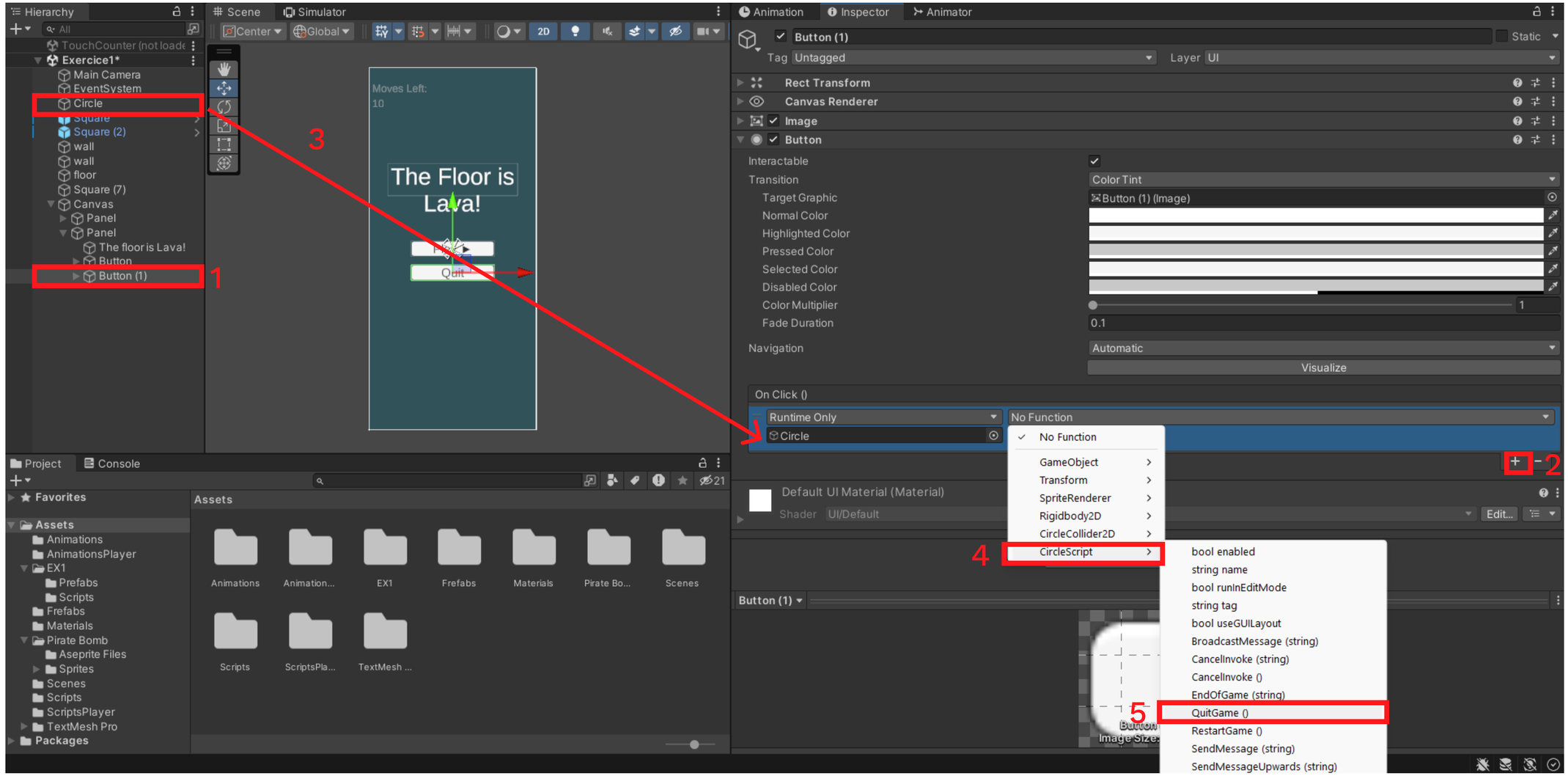
# ADICIONA A TUA CENA ÀS DEFINIÇÕES DO BUILD



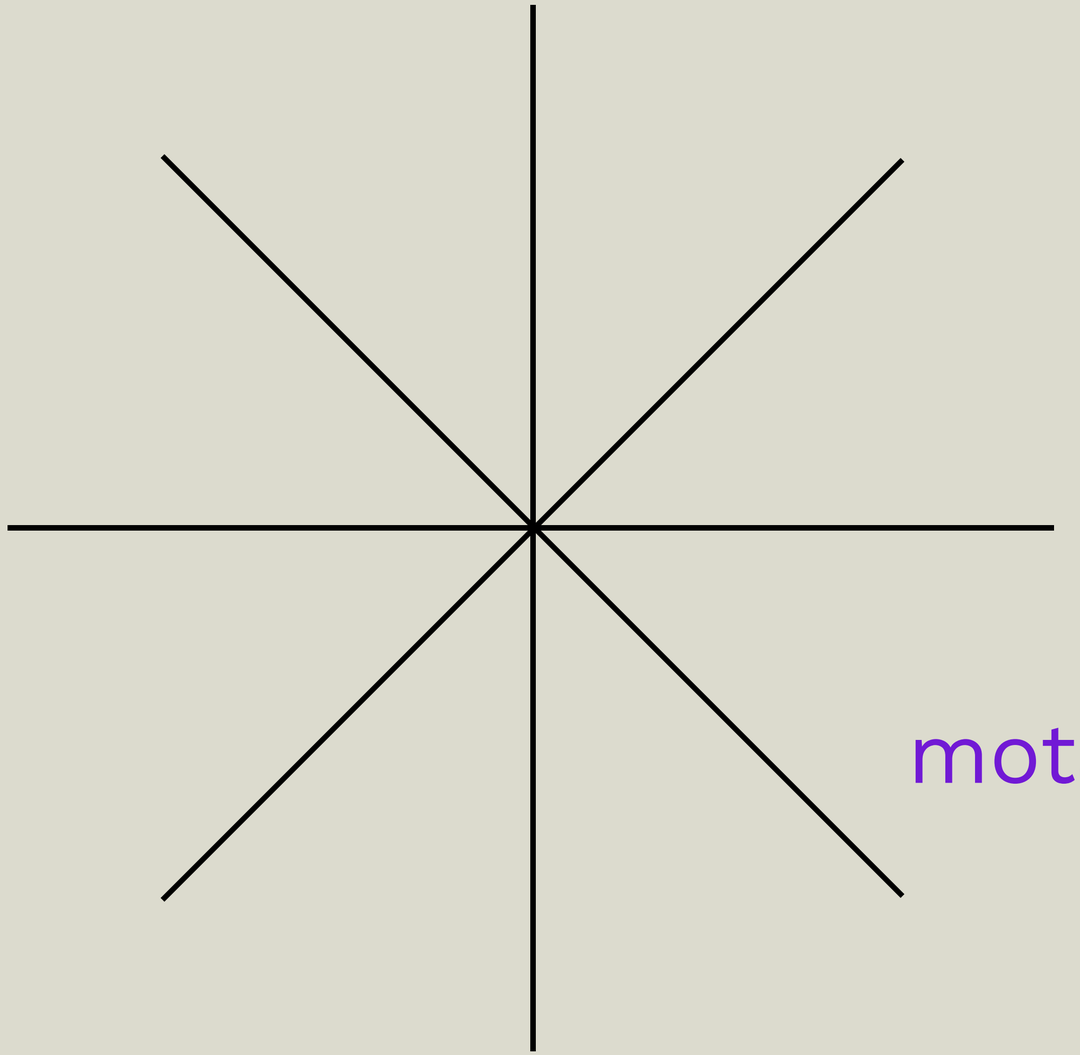
# CRIA UM MÉTODO PARA SAIR DO JOGO

```
0 referências
public void QuitGame()
{
    Application.Quit();
}
```

# CONECTA O BOTAO DE ‘QUIT’ COM O MÉTODO



# DESAFIO:
# FAZ O CÍRCULO CAIR AUTOMATICAMENTE APÓS OS MOVIMENTOS TERMINAREM!



# Obrigada!

Não te esqueças onde encontrar este ppt:

motamdaniela.github.io/dam