Tutorial Two: Dragging an Item

Game Programming Project

By Mariana Neiva Santos Silva

What you’ll learn

In this Tutorial you will learn how to drag an item in a 2D game in Unity.

Programs used

UNITY

Game Engine

VISUAL STUDIO

Code Editor

ADOBE STOCK

Stock images

What you should already know:

1 A basic understanding of **Unity;**

2 Basic understanding of **C#**

2 Have followed tutorial 1

Let’s beginning!

Steps

timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

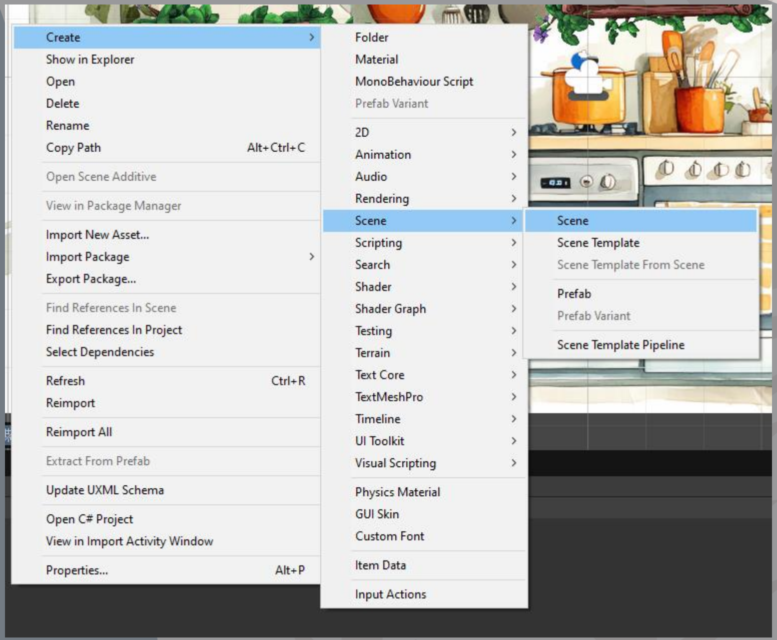
CONNECTING SCRIPT TO THE ITEMS

TESTING

Step 1: Setting up the project.

THE SCENE:

- In your scene folder create a new scene.
- I will name mine Bakery, but feel free to name it whatever you want, (e.g. Level1).
  - Project > Scene Folder > Right Click > Create > Scene > Scene



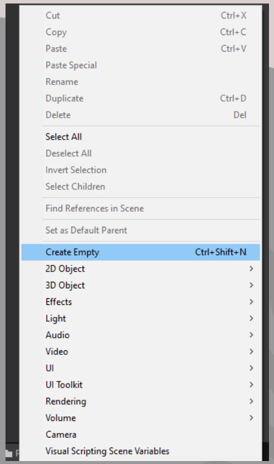
Step 1: Setting up the project.

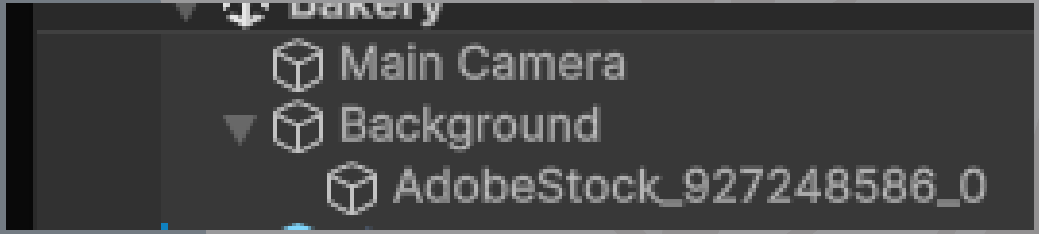
BACKGROUND

- In the Hierarchy of this new scene, go we are going to create an Empty Object and name it background.
- Inside this Empty Object I will add everything to do with my background.

ADDING A PICTURE

- Go to your Art folder and drag the background image you have chosen.
- Then resize it to your desired size.
- Don't forget to add it to the Empty object we created in the hierarchy.





timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

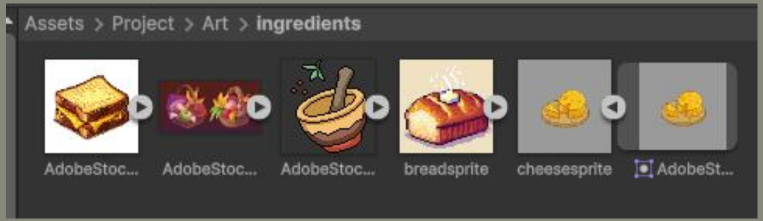
CONNECTING SCRIPT TO THE ITEMS

TESTING

Step 2: creating the items:

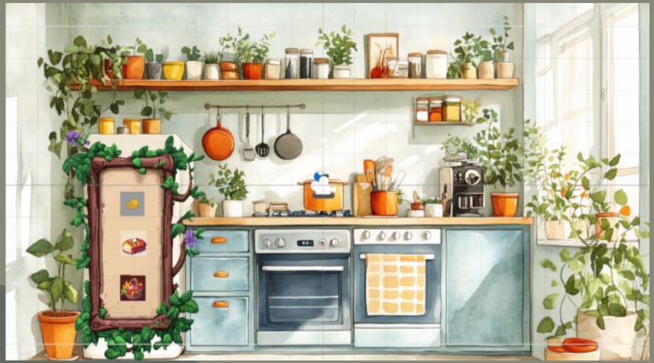
ORGANIZATION

- I created a folder for my item sprites in the art folder.
- Don’t forget to name them!



SPRITES

- Drag the sprites to the scene and resize them.
- Position them as you would like.
- This is how I placed mine:



timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

CONNECTING SCRIPT TO THE ITEMS

TESTING

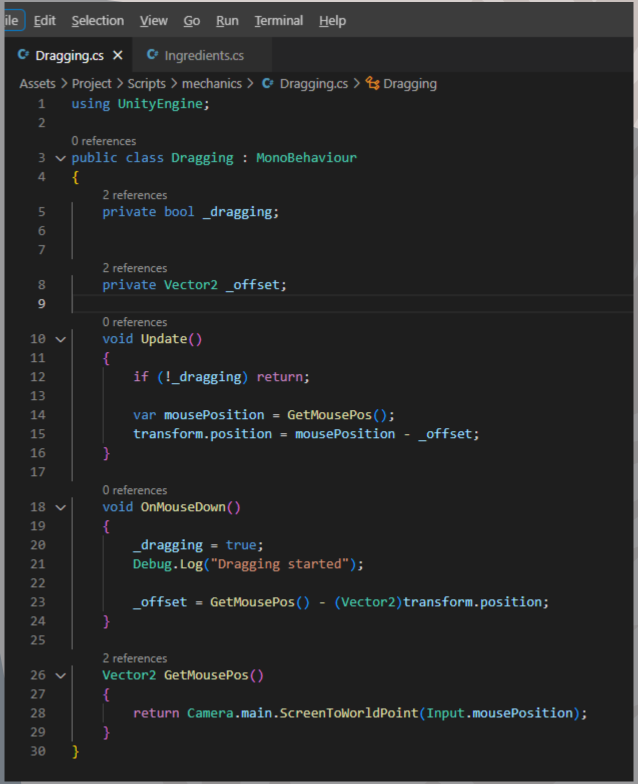
Step 3: The Script

CREATE A SCRIPT

- Create a script on the script folder and name it whatever you want for this tutorial I will name it "Dragging".

THE SCRIPT

- This script will allow you to drag an object around in unity. It won't allow you to do anything else. We will cover that in the next tutorials.


ile Edit Selection View Go Run Terminal Help
Dragging.cs
Ingredients.cs
Assets > Project > Scripts > mechanics > Dragging.cs > Dragging

```csharp
using UnityEngine;

0 references
public class Dragging : MonoBehaviour
{
    2 references
    private bool _dragging;


    2 references
    private Vector2 _offset;

    0 references
    void Update()
    {
        if (!_dragging) return;

        var mousePosition = GetMousePos();
        transform.position = mousePosition - _offset;
    }

    0 references
    void OnMouseDown()
    {
        _dragging = true;
        Debug.Log("Dragging started");

        _offset = GetMousePos() - (Vector2)transform.position;
    }

    2 references
    Vector2 GetMousePos()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }
}
```

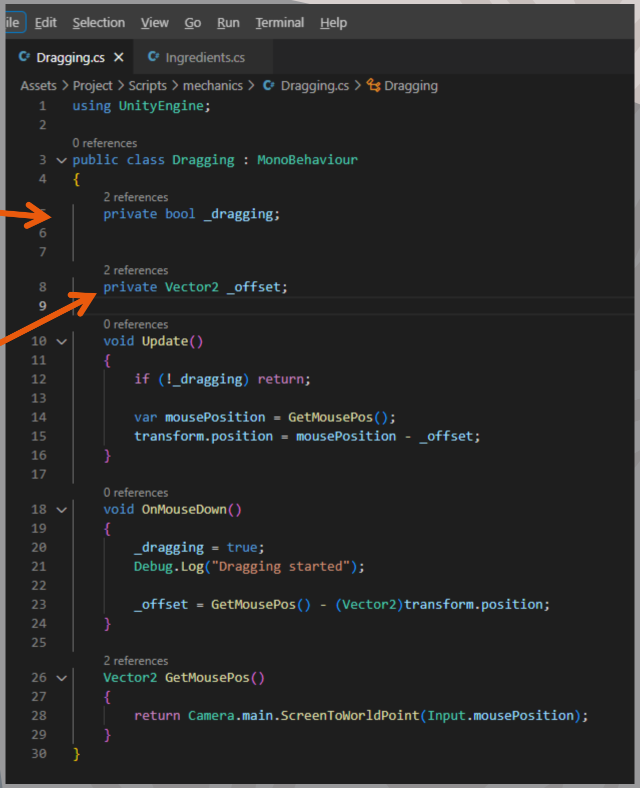
Step 3: Understanding the script

BOOLEANS

- Firstly, we start by setting a private Boolean called "_dragging".
- This will allow us to see control if dragging is either true or false.
- Since a Boolean (or a bool) is a type of data is either true or false.

VECTOR 2

- We will create another more private variables called "_offset"
- The "_offset" will allows us to offset the item so when we drag it, it won't be completely under the mouse.

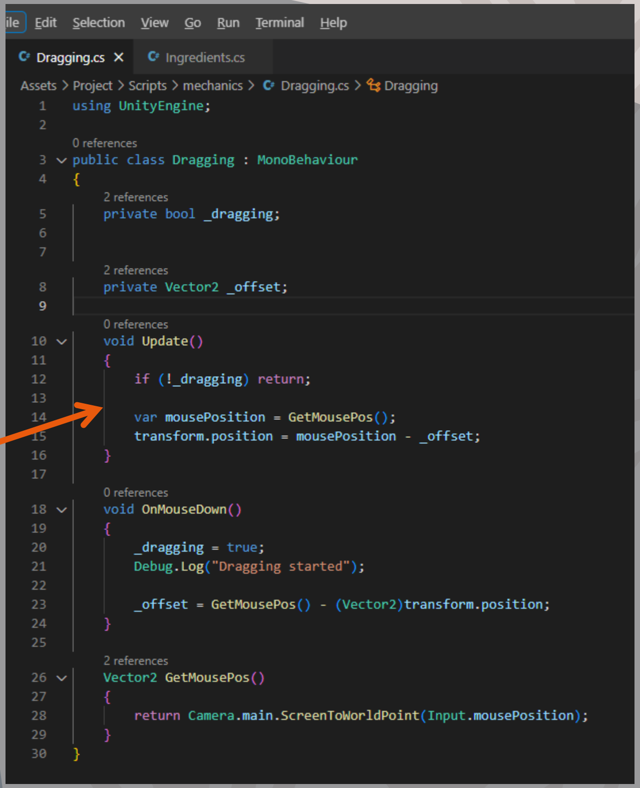

ile Edit Selection View Go Run Terminal Help
Dragging.cs
Ingredients.cs
Assets > Project > Scripts > mechanics > Dragging.cs > Dragging

```
using UnityEngine;

0 references
public class Dragging : MonoBehaviour
{
    2 references
    private bool _dragging;


    2 references
    private Vector2 _offset;

    0 references
    void Update()
    {
        if (!_dragging) return;

        var mousePosition = GetMousePos();
        transform.position = mousePosition - _offset;
    }

    0 references
    void OnMouseDown()
    {
        _dragging = true;
        Debug.Log("Dragging started");

        _offset = GetMousePos() - (Vector2)transform.position;
    }

    2 references
    Vector2 GetMousePos()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }
}
```

Step 3: Understanding the script

VOID UPDATE()

- ***`if (!_dragging) return;`***
- This line essentially means that If dragging is false the rest of the script is skipped.
- This is to ensure that dragging only happens when the player is actually dragging an item.
- ***`var mousePosition = GetMousePos();`***
- Here we create a variable "***mousePosition***" that gets the position of the mouse in the scene by using method GetMousePos() we will create later in the script.
- ***`transform.position = mousePosition - _offset;`***
- This one is simple! We are saying that "***transform.position***" is equal to the "***mousePosition***" variable we created earlier minus the "***_offset***" value.



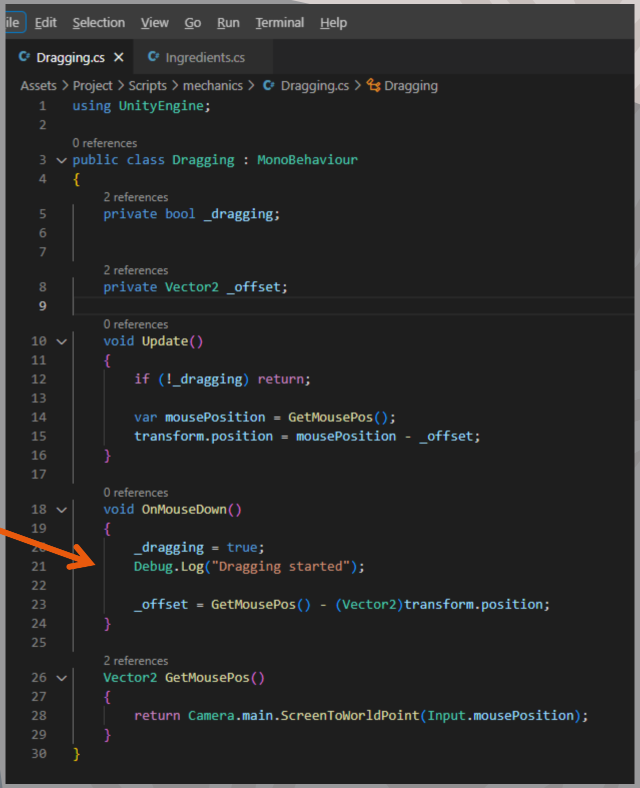
ile Edit Selection View Go Run Terminal Help
Dragging.cs
Ingredients.cs
Assets > Project > Scripts > mechanics > Dragging.cs > Dragging

```
using UnityEngine;

0 references
public class Dragging : MonoBehaviour
{
    2 references
    private bool _dragging;


    2 references
    private Vector2 _offset;

    0 references
    void Update()
    {
        if (!_dragging) return;

        var mousePosition = GetMousePos();
        transform.position = mousePosition - _offset;
    }

    0 references
    void OnMouseDown()
    {
        _dragging = true;
        Debug.Log("Dragging started");

        _offset = GetMousePos() - (Vector2)transform.position;
    }

    2 references
    Vector2 GetMousePos()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }
}
```

Step 3: Understanding the script

VOID ONMOUSEDOWN()

- This method will only happen if the player presses down on their mouse in an item that has this script, since we call the method OnMouseDown() from unity.
- ***`_dragging = true;`***
- This tells unity that "***`_dragging`*** " is true and that the player is dragging.
- ***`Debug.Log("Dragging started");`***
- This is just a debug log or message that will show up in the console if the player drags the item.
- ***`_offset = GetMousePos() - (Vector2)transform.position;`***
- Now we will calculate the "***`_offset`*** " by using the method "***`GetMousePos()`***" again and subtracting it from "***`Vector2)transform.position`***" which is the 2D world space position of the transform.


File Edit Selection View Go Run Terminal Help
Dragging.cs
Ingredients.cs
Assets > Project > Scripts > mechanics > Dragging.cs > Dragging

```
using UnityEngine;

0 references
public class Dragging : MonoBehaviour
{
    2 references
    private bool _dragging;


    2 references
    private Vector2 _offset;

    0 references
    void Update()
    {
        if (!_dragging) return;

        var mousePosition = GetMousePos();
        transform.position = mousePosition - _offset;
    }

    0 references
    void OnMouseDown()
    {
        _dragging = true;
        Debug.Log("Dragging started");

        _offset = GetMousePos() - (Vector2)transform.position;
    }

    2 references
    Vector2 GetMousePos()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }
}
```

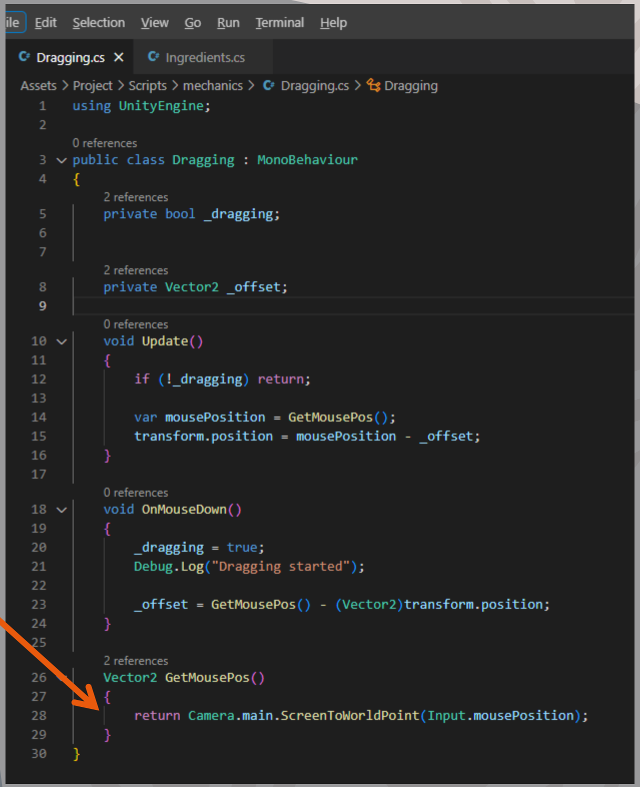
IMPORTANT

Why did we add the line “***transform.position = mousePosition - _offset;***” in Void Update() if we set the value in Void OnMouseDown()?
We did this because Void Update() updates every frame!
Void OnMouseDown() is only used when the player clicks down on their mouse.
When “***_dragging***” is true we want the mouse position with the offset to be updated every frame not just once.

Step 3: Understanding the script

VECTOR 2 GETMOUSEPOS()

- Vector 2 ensures that our coordinates will only be in 2 axes since we are creating a 2D game.
- The method GetMousePos() gets the position of the mouse and guarantees that it's updated when moved.
- How does it do this?
- ***`return`***
- This means that the method can be used elsewhere in the code. It sends it back – returns it – to where it was called in the script.
- ***`Camera.main.ScreenToWorldPoint`***
- This converts the players screen into coordinates.
- ***`(Input.mousePosition);`***
- Finally, like the name says this gets the player's mouse input. In other words, this tells the method where the mouse.
- Essentially this gets the players mouse position and converts it in to coordinates in the x and y axis, for example (13,24).


ile
Edit
Selection
View
Go
Run
Terminal
Help
Dragging.cs
Ingredients.cs
Assets > Project > Scripts > mechanics > Dragging.cs > Dragging

```
using UnityEngine;

0 references
public class Dragging : MonoBehaviour
{
    2 references
    private bool _dragging;


    2 references
    private Vector2 _offset;

    0 references
    void Update()
    {
        if (!_dragging) return;

        var mousePosition = GetMousePos();
        transform.position = mousePosition - _offset;
    }

    0 references
    void OnMouseDown()
    {
        _dragging = true;
        Debug.Log("Dragging started");

        _offset = GetMousePos() - (Vector2)transform.position;
    }

    2 references
    Vector2 GetMousePos()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }
}
```

timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

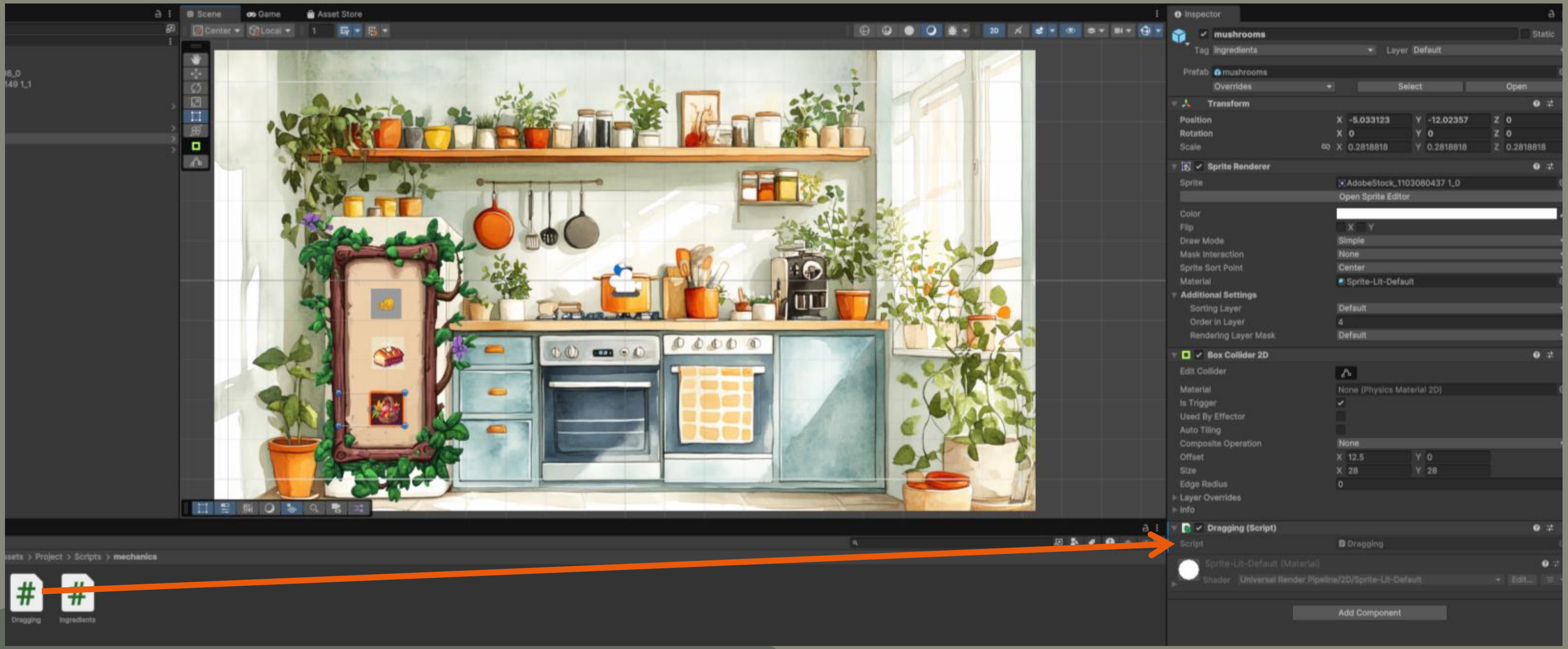
CONNECTING SCRIPT TO THE ITEMS

TESTING

Step 4: Connecting the script.

ITEMS

- This is the easy part.
- Go to your scene and pick the object/objects you want to be able to drag.
- With the item open drag the script into the inspector.



```
mushrooms                                   Static
Tag Ingredients            Layer Default
Prefab mushrooms
Overrides        Select        Open
Transform
Position    X -5.033123   Y -12.02357   Z 0
Rotation    X 0           Y 0           Z 0
Scale       X 0.2818818   Y 0.2818818   Z 0.2818818
Sprite Renderer
Sprite                  AdobeStock_1103080437 1_0
                        Open Sprite Editor
Color
Flip                    X  Y
Draw Mode               Simple
Mask Interaction        None
Sprite Sort Point       Center
Material                Sprite-Lit-Default
Additional Settings
  Sorting Layer         Default
  Order in Layer        4
  Rendering Layer Mask  Default
Box Collider 2D
Edit Collider
Material                None (Physics Material 2D)
Is Trigger              ✓
Used By Effector
Auto Tiling
Composite Operation     None
Offset                  X 12.5   Y 0
Size                    X 28     Y 28
Edge Radius             0
Layer Overrides
Info
Dragging (Script)
Script                  Dragging
Sprite-Lit-Default (Material)
Shader  Universal Render Pipeline/2D/Sprite-Lit-Default   Edit...
```

timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

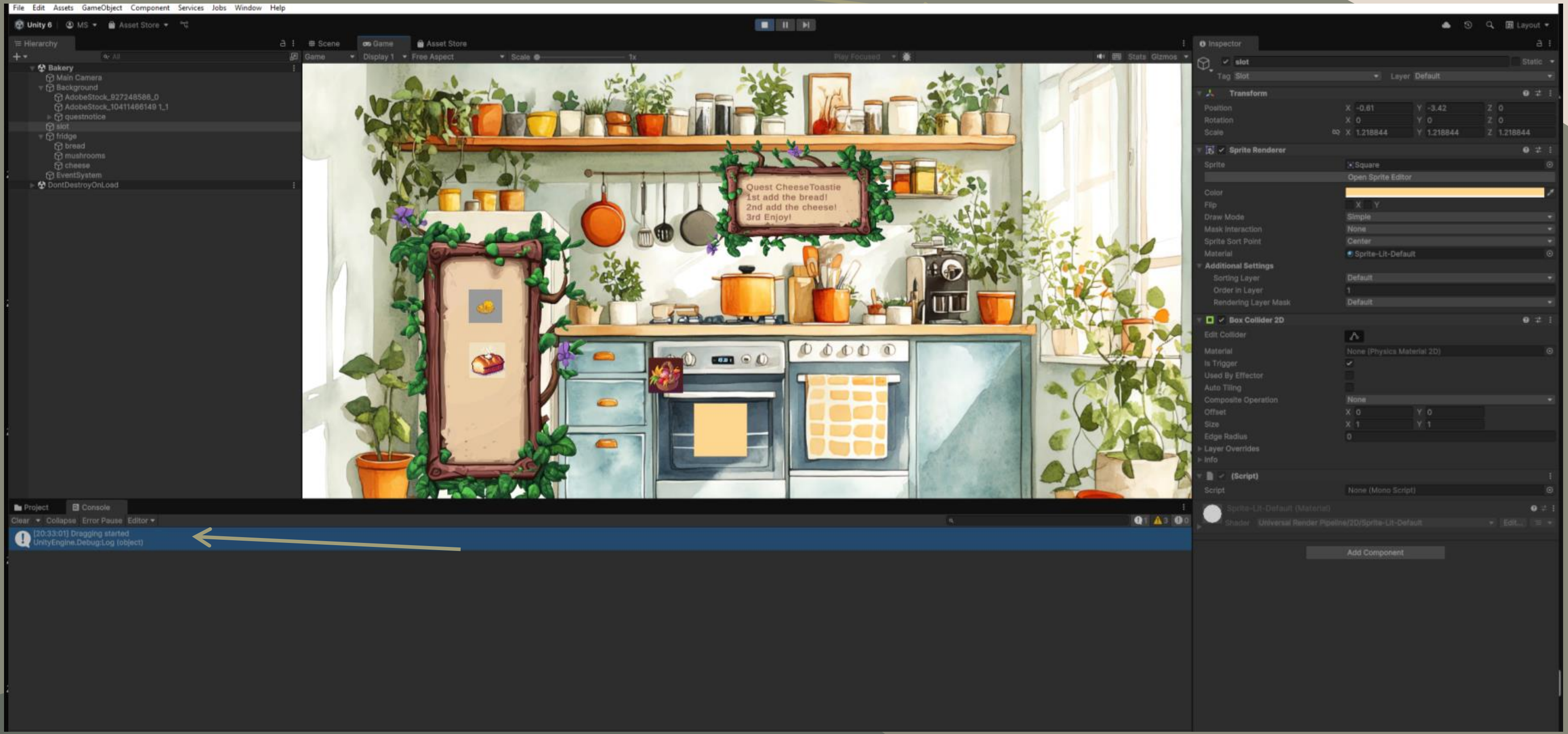
CONNECTING SCRIPT TO THE ITEMS

TESTING

Step 5: Testing.

PLAYING OUR GAME:

- Now let's test our work!
- If you click on an item with the script, it will follow your mouse.
- Remember that when you drag it a log will appear in the console it will read "Dragging started"



Quest CheeseToastie
1st add the bread!
2nd add the cheese!
3rd Enjoy!

```
[20:33:01] Dragging started
UnityEngine.Debug:Log (object)
```

timeline

SETTING UP THE SCENE

CREATING THE ITEMS

CODING THE SCRIPT

CONNECTING SCRIPT TO THE ITEMS

TESTING

Congratulations!

You now can drag an item in unity!

Thank you