INTRODUCTION

Hi, thanks for buying this asset, here I will explain you everything you should know about this asset and how to use it easily. This does not require a lot of knowledge, so don't worry if you are not familiar with Unity. You will need TextMeshPro for this, it is very easy to install, probably, Unity will tell you to install it when you import this asset, if not, go to the Unity asset store and download it.

HOW TO SET UP THE SHOP

To set up the shop you need to follow some steps:

1 - Create your "Item card" (There is an example prefab in the prefabs folder of this), this card will be used for every item, and there will generate as many cards as you want for the shop, the video should clarify this. This item card by default need some objects:

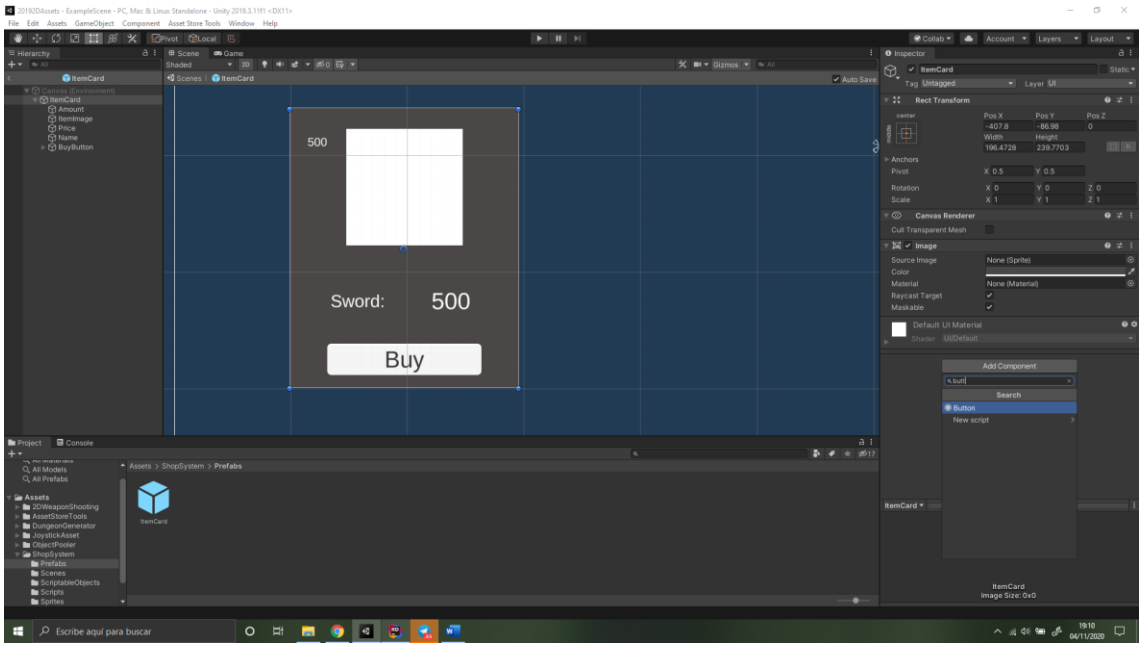
- Amount (TextMeshProUGUI), a text to display the quantity of ítems of one type available.

- ItemImage (Image), an image for the item.

- Name (TextMeshProUGUI), a text to display the name of the item.

- Price (TextMeshProUGUI), a text to display the price of the item

- BuyButton (Button that uses TMPro), a button to buy the item, if you want to just click the card to buy the item, just add a button component to the body of the card. (As shown in the next image)



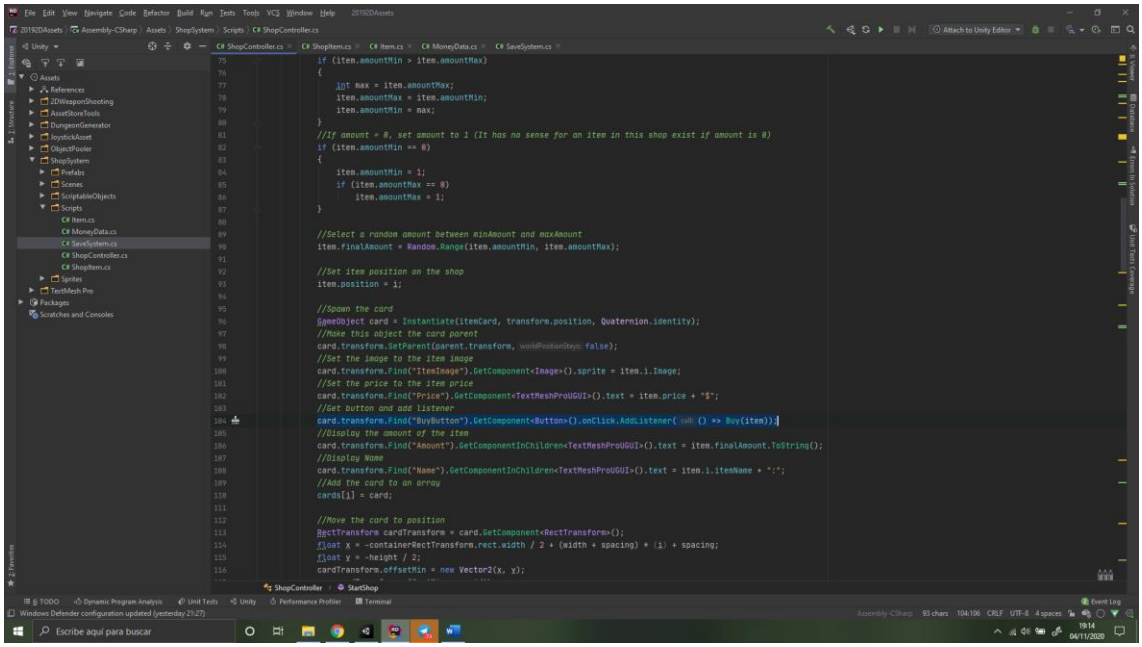
*If you add the button to the card body you will have to change 2 things in the code. Open the "ShopController" script and go to lines 104 and 212, you will see this:

```
card.transform.Find("BuyButton").GetComponent<Button>().onClick.AddLis
tener(() => Buy(item));
```

Change it to something like this:

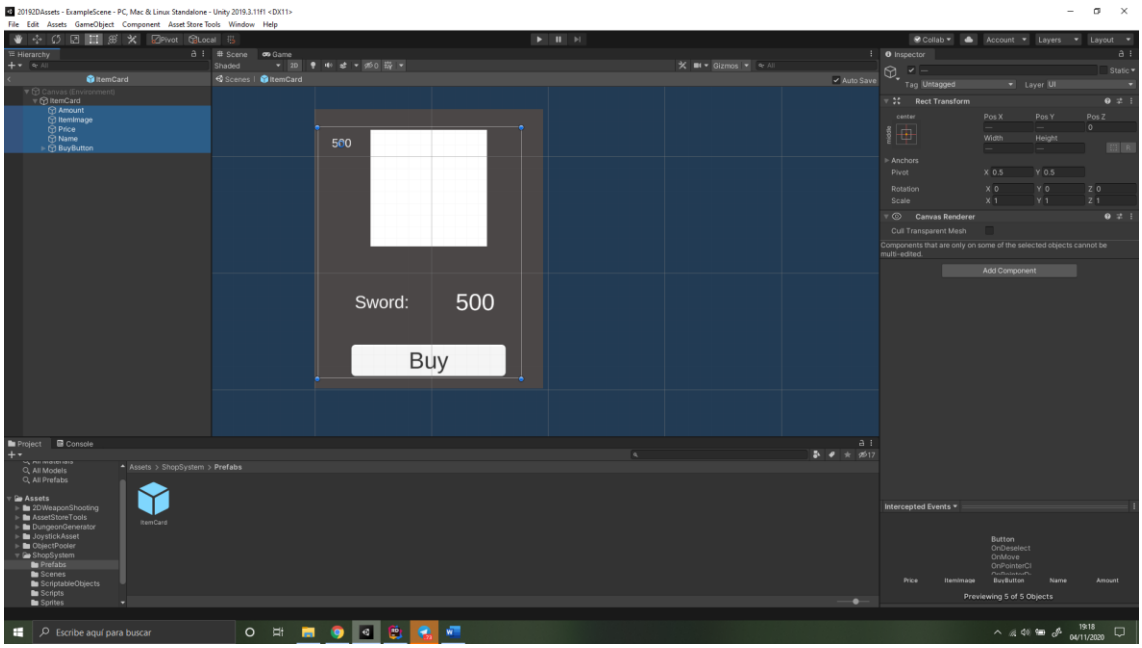
```
Card.GetComponent<Button>().onClick.AddListener(() => Buy(item));
```

Here is an image in case you can't find the line:



As you can see in this image, there are some yellow words, those words are the names that the Item card objects (previously mentioned) need (Amount, ItemImage, Name…). If you have other names on your objects, just change the names in this lines, and the lines between 208 and 216. (If you don't do this, the script will not work).

The objects are shown in the next image.



You can make your Item Card as you want, this mean you can add or delete any texto r image, but if you do that, just make sure you also do this:

-If you add an object (for example a text called “Rarity”) :

First, open the “ShopItem” script, and add the property that you want if it does not exist, to add a “rarity” property, just decide what type of variable should be (for example an int → 0 for common, 1 for rare, 2 for super rare, 3 for epic, and 4 for legendary), and then add that line:

public int rarity;

(Each time you change something in this script all Scriptable objects created based on this script will change, I will explain this better later).

Then on the “ShopController” script add this line of code after lines 108 and 216, do it before the “cards[i] = card” (it’s important):

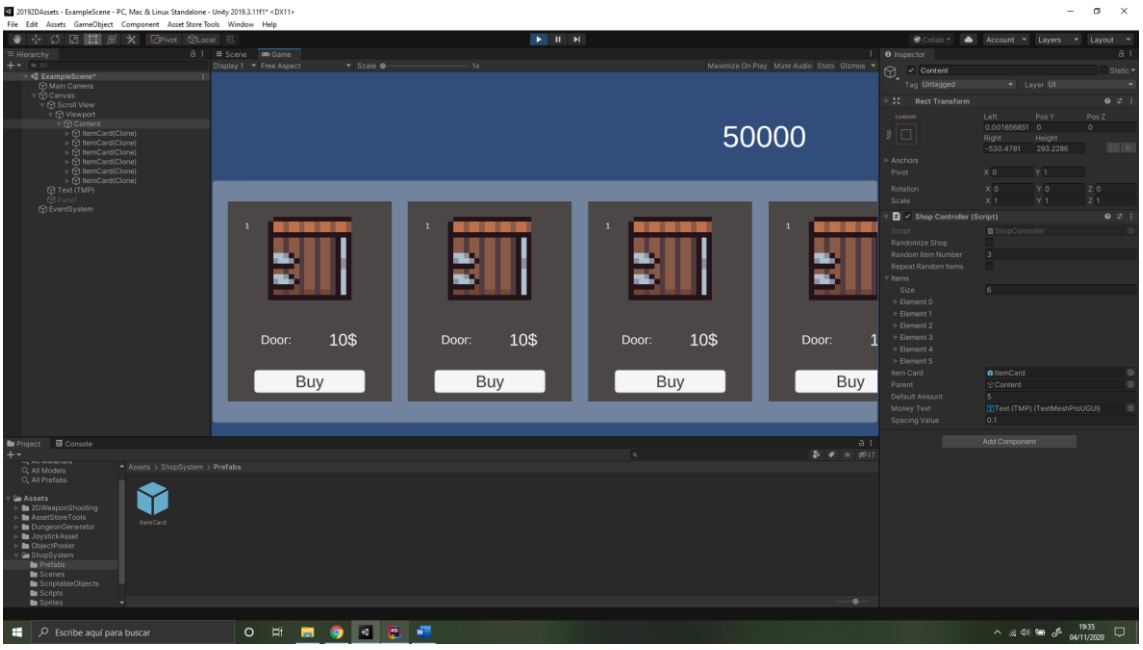
card.transform.Find(“Rarity”).GetComponent<TextMeshProUGUI>().text = item.i.rarity;

(You must have previously created the text on the Item Card)

You will change the size of the card in the future to ensure that it fits your shop, so don’t worry if it ends being big the first time you generate the shop.

Save it as a prefab.

2 - Create the UI “Container” for the shop, this will be the “window” where the cards will be, look this image to understand properly:

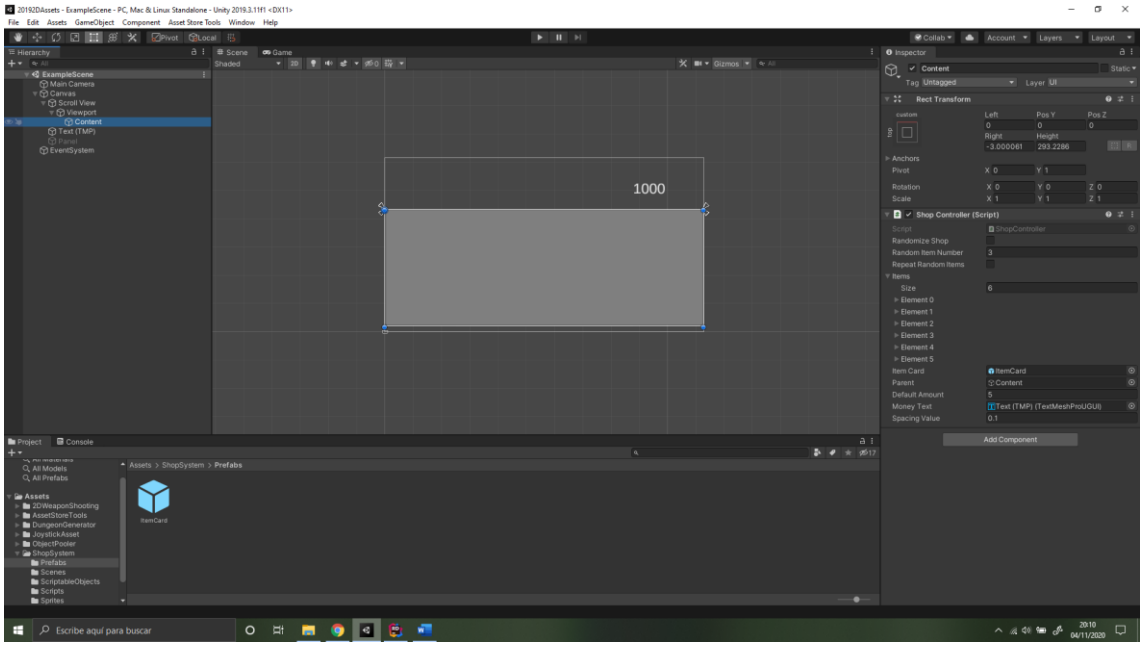


The grey part on the background is the “Container”, you can use almost everything (Panel, image, etc), but I recommend using a Scroll view. There will be a tutorial on the package and also a link to a video on youtube:

https://www.youtube.com/watch?v=QRKff9HxSdw

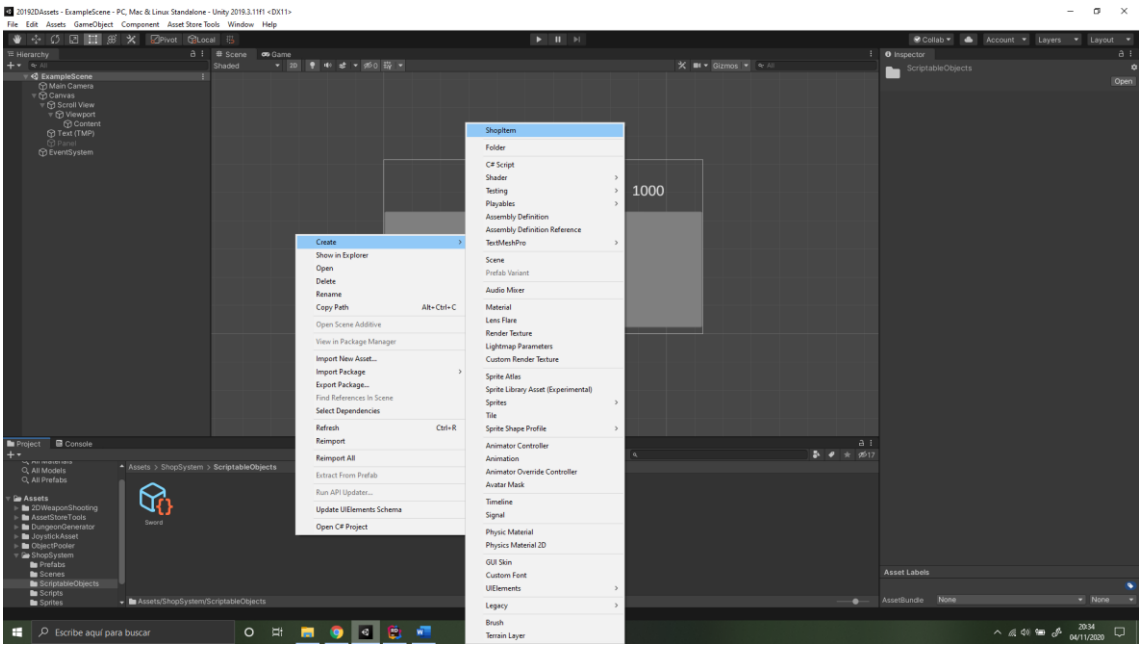
By default it is an horizontal shop, to make it vertical, visit this website: https://forum.unity.com/threads/solved-trying-to-create-a-dynamic-scroll-view.313422/

To explain it quickly, create a scrollview on the editor, on the “Scroll view” object, untick vertical (unless you’ve done what’s on that website), go to the “content” child (If you dislike the scrollbars, just delete the children called scrollbars), and here, paste the “ShopController” script (you can paste this script wherever you want, is better explained in the scripts part), fill the properties input fields(Explained in the next part), and you are done.



I’ve also created a text for the money, that you will see in one input field, delete on the code if you don’t like it. It is on lines 20, 29, 30, 253, 254.

3 – Create “Shop Items”. If you right click inside any folder of the “Project tab” and then click on create, at the top you will see “Shop Item”, this is a Scriptable object, and there are some examples on the scriptable objects folder.



Click on that, and it will create a “Shop Item”, fill the properties of the Shop Item (You will need to create one shop item for every item you want on your shop).

Each time you modify the “Shop Item” script, go and check the scriptable objects, they will have changed.

SCRIPTS AND PROPERTIES

Save System: This script saves whatever information you want into files, you can write and read that, the example I’m using is with money, if you want to change how this is saved, feel free to use any way you like. If you like this, there is a tutorial here:

https://www.youtube.com/watch?v=XOjd_qU2Ido

Static functions used here are:

- SaveSystem.SaveMoney(int money) → It stores a new value for the money.
- SaveSystem.LoadMoney() → which returns a “MoneyData” which has a public property called “money”, just use SaveSystem.LoadMoney().money to get the stored money value.

MoneyData: A script needed to save information into files, it’s explained in the tutorial on the link above.

ShopController: This script manages all the shop, and here are all the important properties and functions.

- RandomizeShop: if you click this, the shop will generate random items from the ones that you have previously selected (You will see this on the Items property).
- RandomItemNumber: max number of objects you want your random shop to have. It always has the max number of items, if you want to change this, just go to line 144 and you wil find this:

```
for(int i = 0; i < randomItemNumber; i++)
```

Change it to something like this:

```
for(int i = 0; i < Random.Range(1, randomItemNumber); i++)
```

- RepeatRandomItems: If you click this items from the pool of items (See later), can be repeated.
- Items: Array of objects type Item, just place the items in the order that you like and write the values as you like (Price, amount, etc), the order of this will not change in the shop if the shop is not random. This items require a “Shop Item” previously created for every Item.
- ItemCard: Just drag the card you created earlier there (This card must be a Prefab).
- Parent: If you don’t want the script to be on the content object, drag it wherever you want but, drag the place you want the cards to be into this field.
- DefaultAmount: If you forget to write properly the amount on any object, the object will take this default value.
- MoneyText: The text that displays the money you currently have, this is optional.
- Spacing value: the more the value is, the larger the distance between cards in the shop.

Inside the script, in the Start() method, there is written “SaveSystem.SaveMoney(50000)”, this is just for testing purposes, delete if you want, manage your money wherever you want with the functions explained in SaveSystem script.

There are three main functions here:

StartShop(): Normal start of the shop. Explained step by step on the code

RandomizeShop(): Same as StartShop() but random, notice that most of the changes you make to one function, you may change it in the other

Buy(): Used whenever you click the button Buy, call it from other scripts if you want.

There is another function, the Sell() function, you have to code this function, because it depends on how are you going to sell your items (from your inventory, clicking again, etc), the thing that you have to do, is, if you sell a new Item, add the item to the list of items:

(Give the item the information it needs first)

newItem.finalAmount = 1;

newItem.amountMin = ?;

newItem.amountMax = ?;

newItem.price = ?;

newItem.postion = items.Count;

newItem.randomSelected = false;

(Now add the item)

Items.Add(newItem);

If the item already exist, just increase the final amount of the item:

(You need to get the item by it's position)

```
foreach(Item i in items){
        if(i.i.itemName == item.i.itemName)
                i.finalAmount++;
}
```

And at the end just add your money back:

```
SaveSystem.SaveMoney(SaveSystem.LoadMoney().money + item.price);
```

ShopItem: This script has the properties that each "Shop Item" will have, change it if you want to add or delete things.

Item: This class is used to display the information of an item easier in the shop, does not need to be modified, if you want to add properties, go to the "Shop Item" script.

CONCLUSION

If you have any question, ask me here:

Youtube: (write a comment in any video)
https://www.youtube.com/channel/UCko8fDha1u0eplpEr4JrQPQ?view_as=subscriber

Twitter: @EpicidadSoftware

Telegram: @TebionDL

Gmail: purreren@gmail.com

Hope this has been useful.