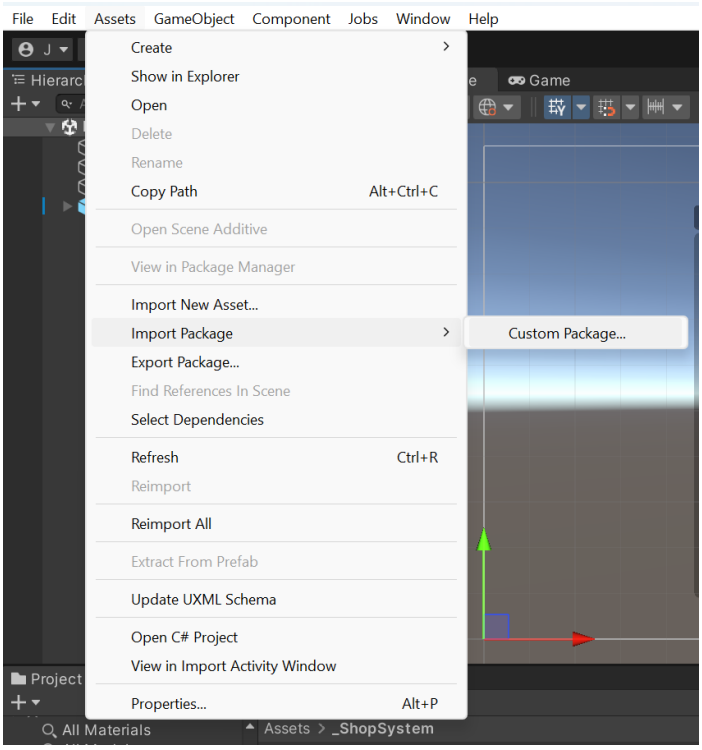
Shop System Instructions

This package is a shop system with an auto-populating UI that uses item information from custom scriptable objects. The package includes a simple scriptable object item and inventory creation system.

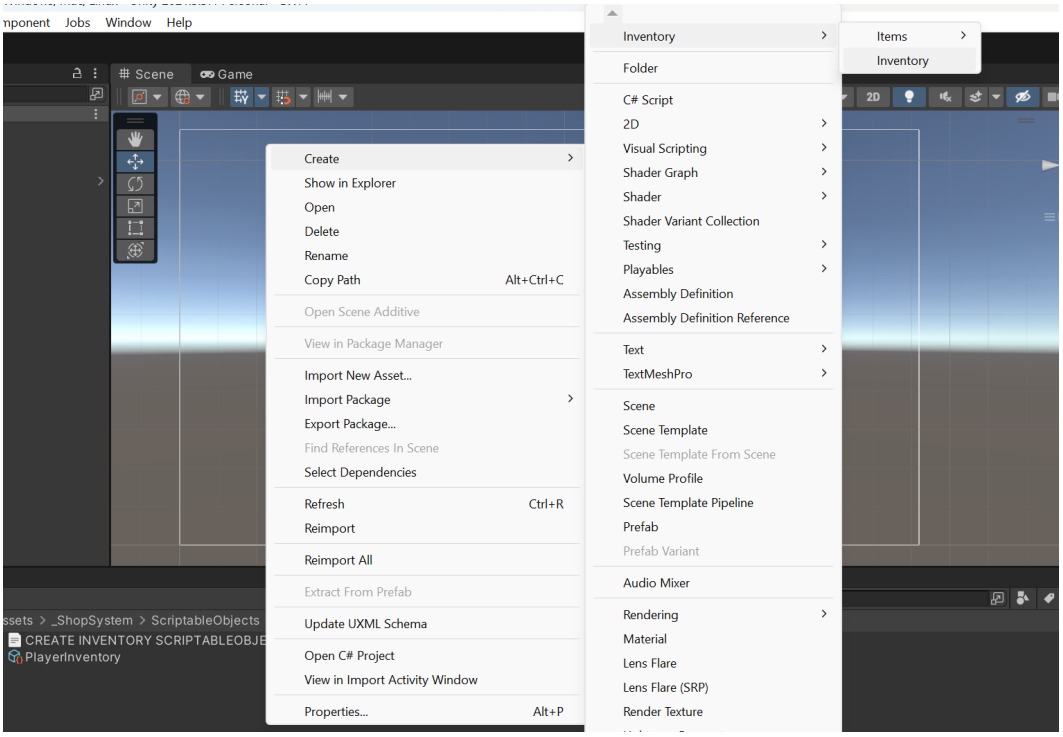
Verified Unity Version:

- Unity 2021.3.3f1

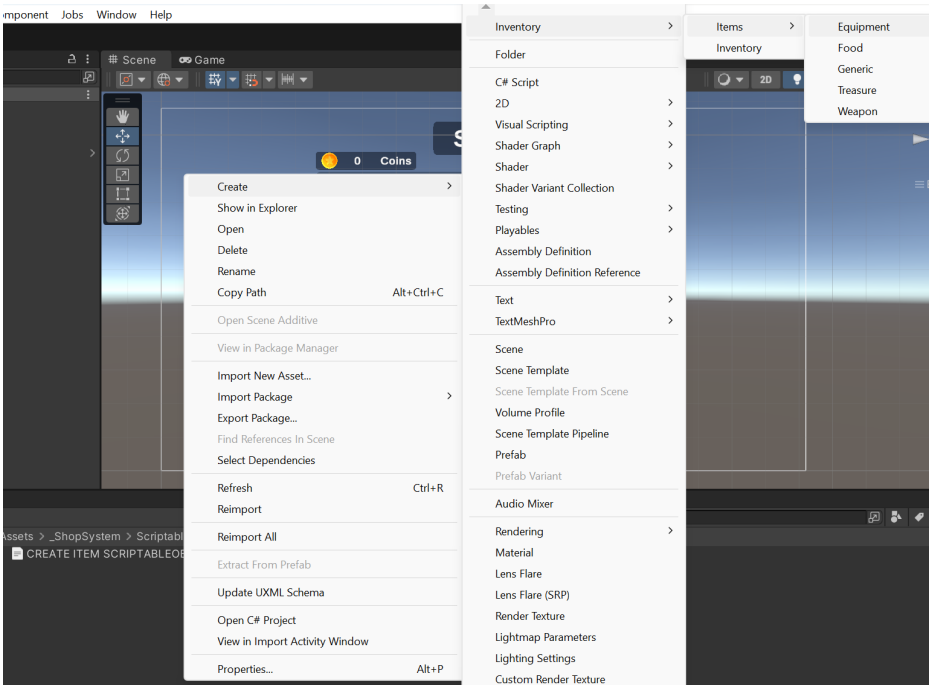
Implement the Shop System

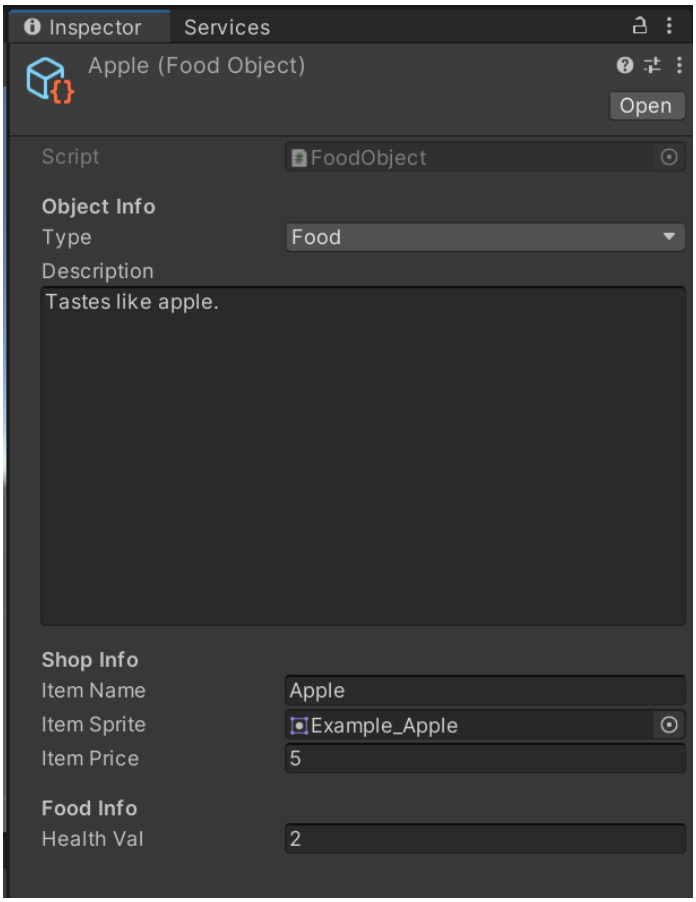
1. Import the package into Unity.
    a. Edit >> Import Package >> Custom Package… >> Navigate to package location on machine
    b. 

2. Create a player inventory scriptable object.
    a. To create an Inventory Object:
        i. Right click in the Project view in the _ShopSystem/ScriptableObjects/Inventory folder, or where you want to save project inventories
        ii. Create >> Inventory >> Inventory

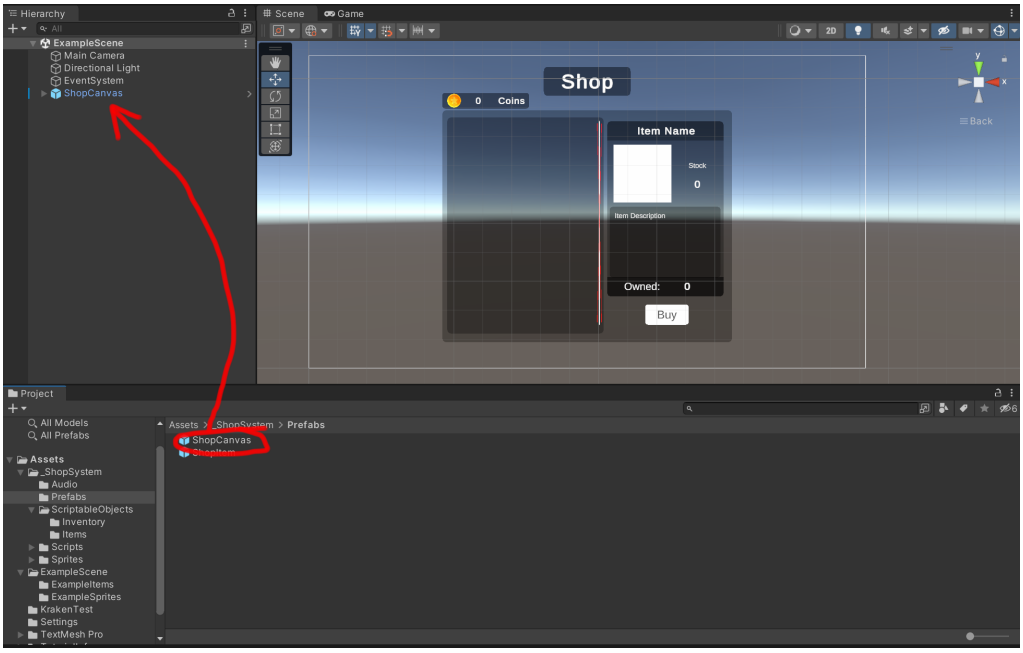
b. 

3. Create items for the project.
    a. To create an Item object:
        i. Right click in the Project view in the _ShopSystem/ScriptableObjects/Items folder, or where you want to save project items
        ii. Create >> Inventory >> Items >> Item Type

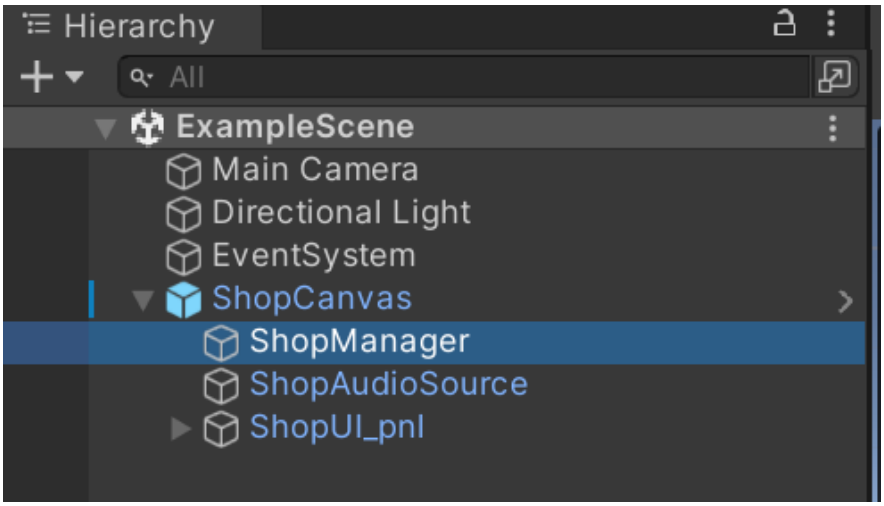
b. 

c. The package comes with 5 basic item types (Generic, Treasure, Food, Weapon, Equipment). The user can create their own item types by changing or adding types in the ItemObject script and creating new scriptable object scripts for each type.
    i. Fill out the Inspector fields accordingly.
    ii. 

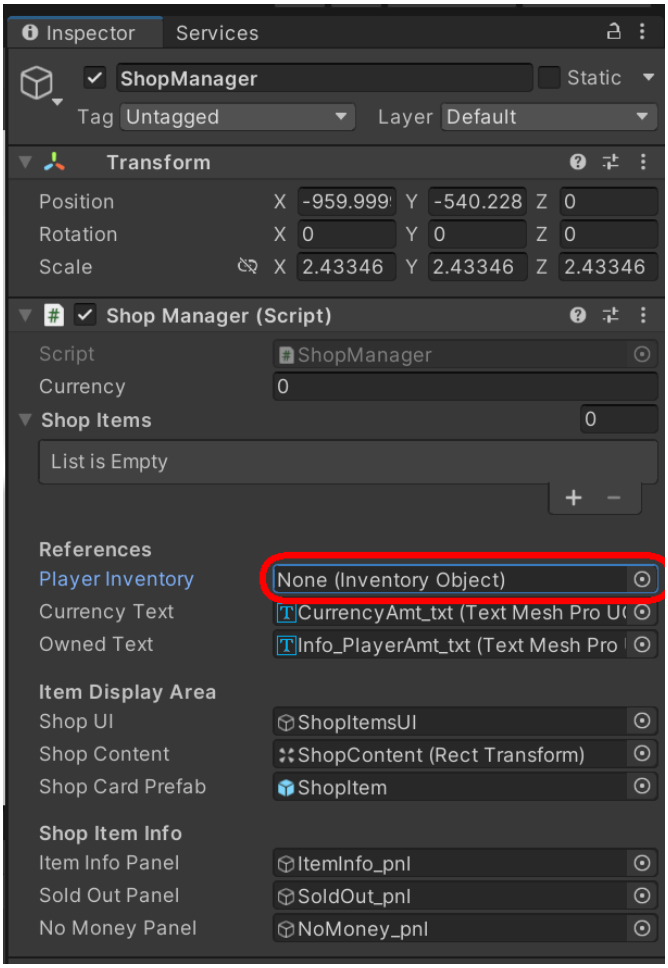
        1. Example: Food Object inspector fields for an Apple item
4. Drag and drop ShopCanvas prefab from the Prefabs folder into the scene hierarchy.

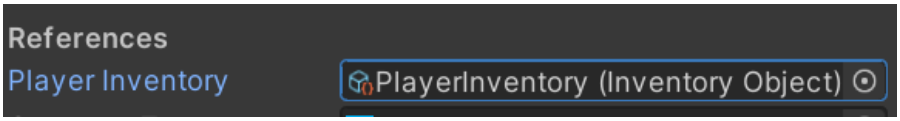
a. 

5. Open the ShopCanvas dropdown and locate the ShopManager GameObject.

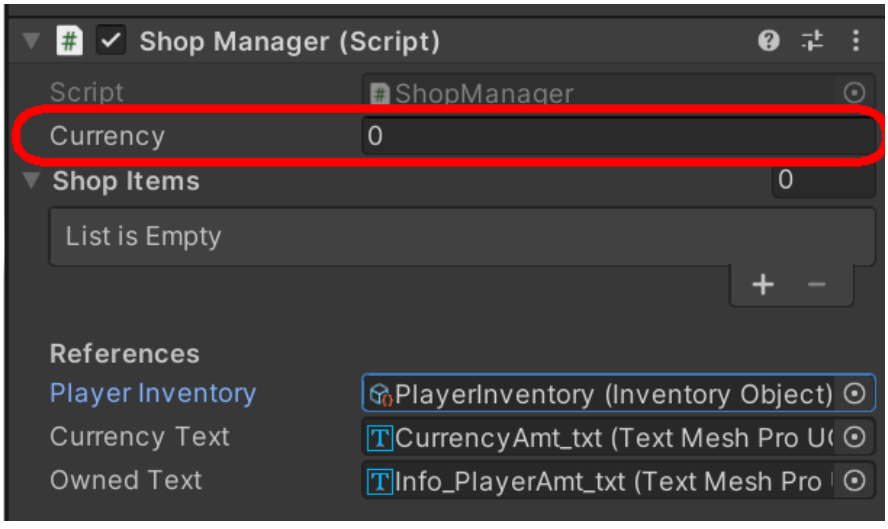
a. 

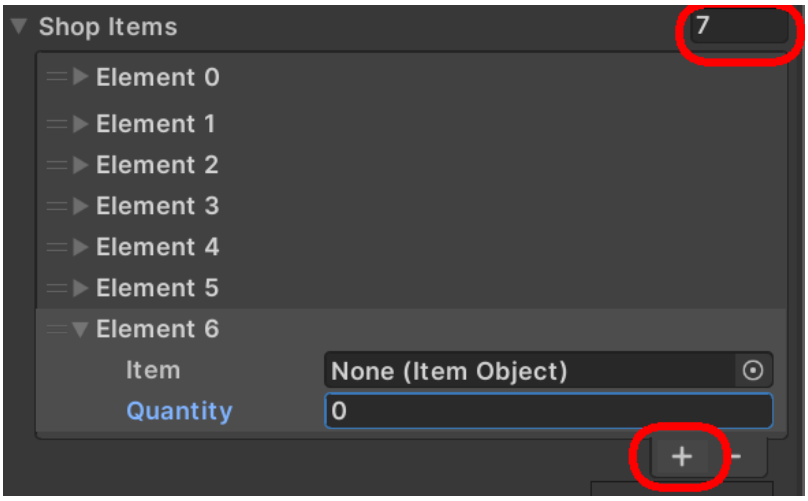
6. Drag and drop the created inventory scriptable object onto the “Player Inventory” reference in the Inspector

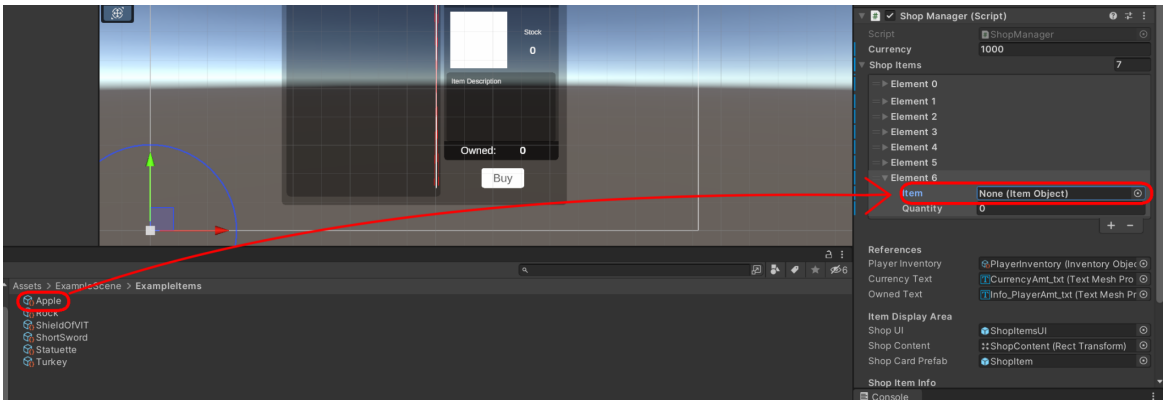
a. 

b. 

7. The user can set the player's currency amount in the ShopManager Inspector window, and/or add to it through their own external gameplay scripts.

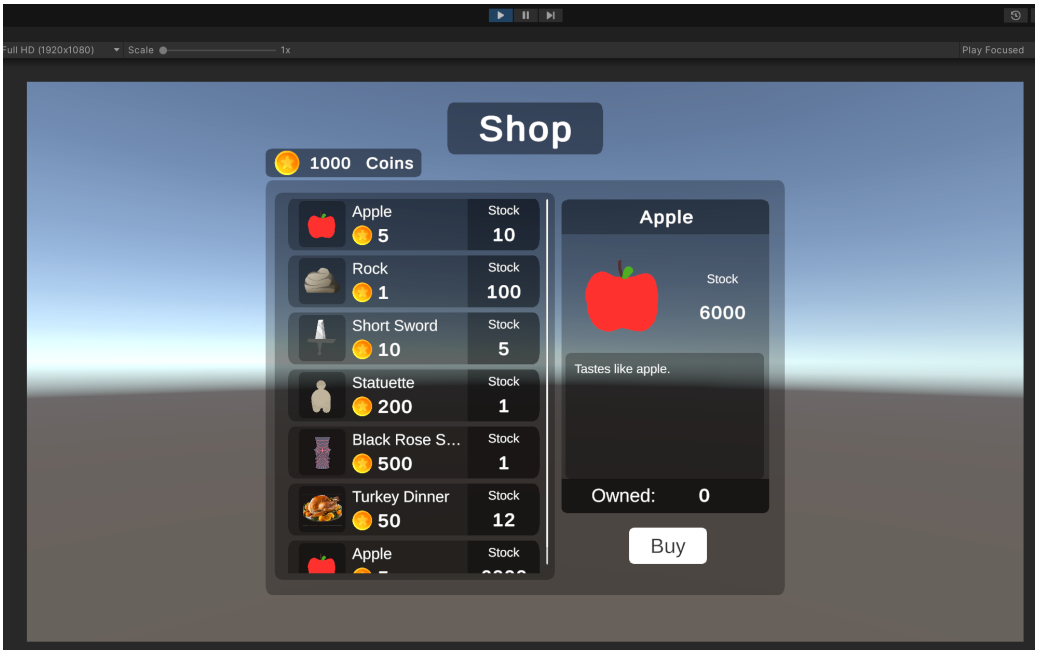
a. 

8. Click the plus (+) button under the ShopItems list dropdown to add a shop item to the list, or set the desired amount of shop items to the right of the list name
    a. 

9. After adding desired amount of shop items, drag and drop the Item scriptable object you want to sell and set the shop's stock amount from the Quantity field under the Item field
    a. 

    b. 

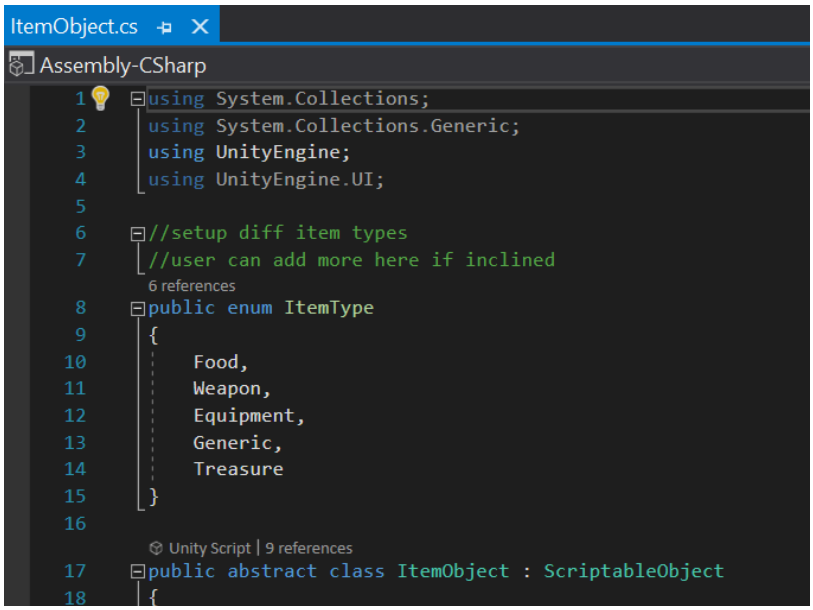
10. Press the Play button and see the shop items populate.

a. 

Guide to Creating New Item Types

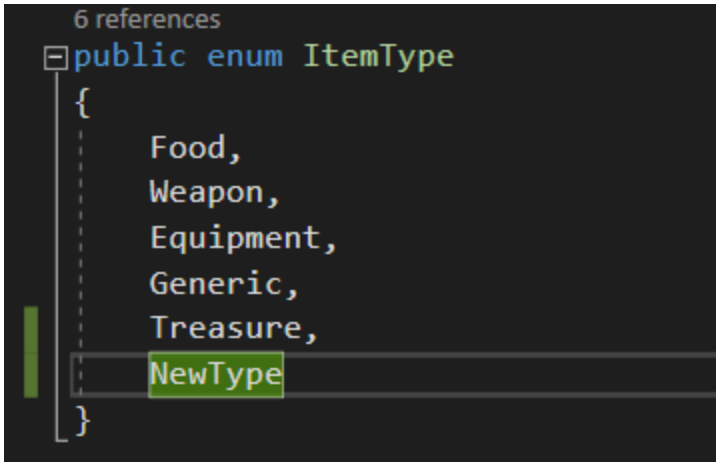
- **These instructions require a little knowledge of code**

All of the item types are stored in the “ItemObject” script which is the base for scriptable objects

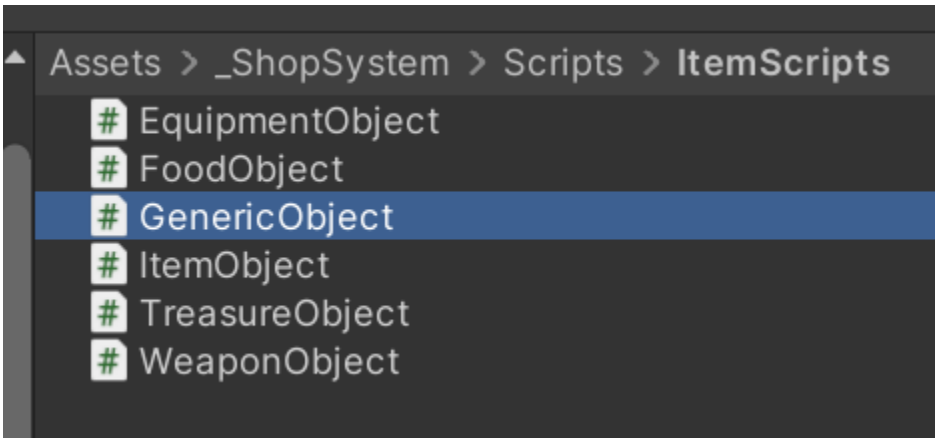


The package user can add more item types by putting a comma after the last item type and adding another item type name.

Example:



To create a new scriptable object script for this item type:

1. Create a new script with the same content as the GenericObject script found in _ShopSystem/Scripts/ItemScripts
    a. 

GenericObject Script

```
[CreateAssetMenu (fileName = "New Generic Object", menuName =
"Inventory/Items/Generic")]
public class GenericObject : ItemObject
{
   //generic item
   private void Awake()
   {
      //make sure correct type is set
      _type = ItemType.Generic;
   }
}
```

2. Change all instances of “Generic” to the name of the new ItemType

```
[CreateAssetMenu (fileName = "New NewType Object", menuName = "Inventory/Items/NewType")]
Unity Script | 0 references
public class NewTypeObject : ItemObject
{
    //generic item
    Unity Message | 0 references
    private void Awake()
    {
        //make sure correct type is set
        _type = ItemType.NewType;
    }
```

a. Make sure to change the “CreateAssetMenu” names as these are what show up in the Create >> Inventory >> Item menu

3. Add other variables to add statistics / data to the new item

Examples:

```
public class FoodObject : ItemObject
{
    //user can add more stats for food here
    [Header("Food Info")]
    public int healthVal;
```

```
Unity Script | 0 references
public class EquipmentObject : ItemObject
{
    //user can add more stats for equipment here
    [Header("Equipment Info")]
    public int defense;
```

Notes:

- The shop does not display the extra stats ItemObjects might have (healthVal, defense, damage, etc.) these values would be used by the package user’s game or displayed in the game inventory.

- To turn the ShopCanvas on and off during gameplay, the user can call the ShopManager script’s function ToggleShop() from any script with a reference to the ShopManager object.