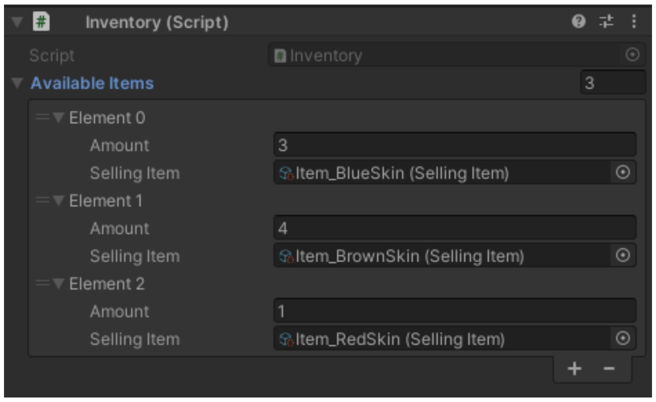
Game Document

In a prototype game, there's a shopkeeper that a player can approach and interact with. Shopkeeper allows the player both to buy or sell his/her outfits. All this is possible by using the **Inventory** system.

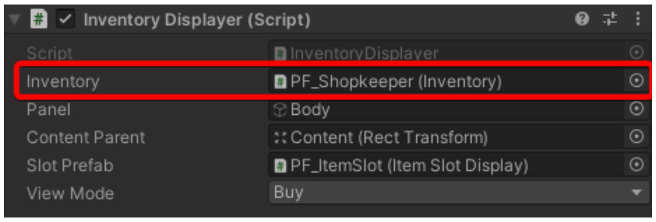
Inventory

Inventory system is a universal system that allows entities with **Inventory Component** to store different items in their storage. An example of how inventory looks like is on the right. It stores information about what Items an object has and how many of them.



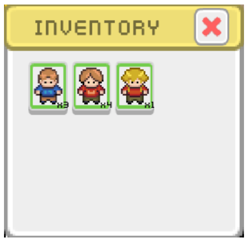
```
Inventory (Script)
Script              Inventory
Available Items                                   3
  Element 0
    Amount          3
    Selling Item    Item_BlueSkin (Selling Item)
  Element 1
    Amount          4
    Selling Item    Item_BrownSkin (Selling Item)
  Element 2
    Amount          1
    Selling Item    Item_RedSkin (Selling Item)
```

In order to visualize the items on UI we can use a prefab called **InventoryDisplayer.** Drop it in a scene and indicate in a field pointed in a red box to display the inventory of a specified object. But inventory won't show up on itself. It's up to us what opens the Inventory window. Whether it is a trigger, a button or an event.



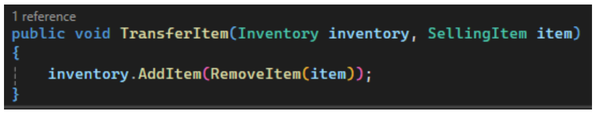
```
Inventory Displayer (Script)
Script              InventoryDisplayer
Inventory           PF_Shopkeeper (Inventory)
Panel               Body
Content Parent      Content (Rect Transform)
Slot Prefab         PF_ItemSlot (Item Slot Display)
View Mode           Buy
```

Inventory UI

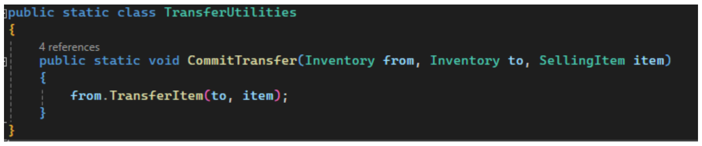


Inventory Transfer Feature

Transfer feature allows inventories to exchange items with each other. With the help of a static class called TransferUtility. Using this feature we can move items from Shopkeeper's inventory to Player's one or vice versa.

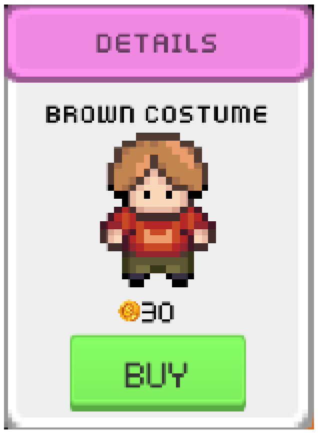


```
1 reference
public void TransferItem(Inventory inventory, SellingItem item)
{
    inventory.AddItem(RemoveItem(item));
}
```



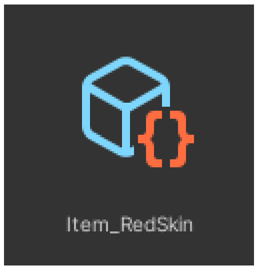
```
public static class TransferUtilities
{
    4 references
    public static void CommitTransfer(Inventory from, Inventory to, SellingItem item)
    {
        from.TransferItem(to, item);
    }
}
```

Details Pop Up



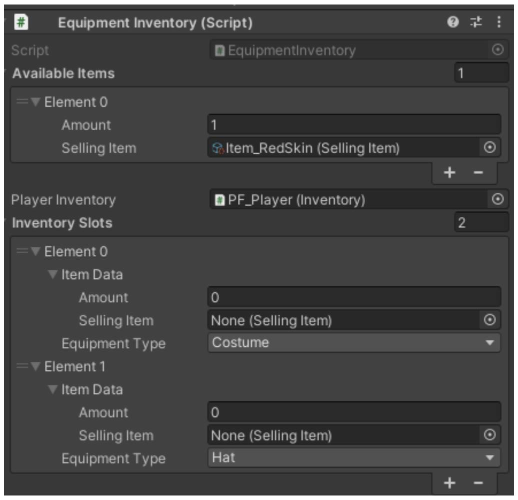
In order to sell or buy outfits from Shopkeeper we need something that acts as a bridge between the parties that ensures proper conditions are met. A simple example of a condition is having enough money. After clicking on an item in inventory **Details** window will pop up and display some information about an item e.g. Name, Visual and/or a price. Additionally, based on what action we want to commit with the item a certain button will be shown.This window allows us to transfer an item to another inventory. If we are buying it from someone, first it checks whether a player has enough money or not. If a player meets conditions transfer will occur, but if not, nothing will happen

Item Data



In order to store information about what name an item has, what icon it should display in inventory and what price it should have, we need a data container like scriptable object to hold all this information. It is called SellingItem. SellingItem scriptable objects are used by **Inventory** and **Details Pop Up.**

Player Equipment

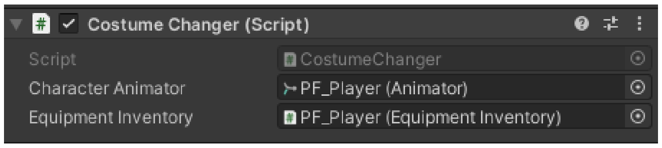


```
Equipment Inventory (Script)
Script                EquipmentInventory
Available Items                          1
  Element 0
    Amount            1
    Selling Item      Item_RedSkin (Selling Item)
Player Inventory      PF_Player (Inventory)
Inventory Slots                          2
  Element 0
    Item Data
      Amount          0
      Selling Item    None (Selling Item)
    Equipment Type    Costume
  Element 1
    Item Data
      Amount          0
      Selling Item    None (Selling Item)
    Equipment Type    Hat
```

A Player has 2 inventories: one to keep bought items in a bag and the other one that corresponds to equipped items. It is operated by **Equipment Inventory Component.** It behaves similarly to Inventory with an additional feature to allow only one type of equipment to be in inventory. This ensures that no same equipment will be worn by a player.

Costume Changer

Costume Changer is a component that listens to **Equipment Inventory** and changes the player's visual. In this prototype player's visuals are changed using an animator component. In order to swap visuals we need to assign a new **RuntimeAnimatorController.** Costume Changer uses controllers provided by SellingItems



```
Costume Changer (Script)
Script                CostumeChanger
Character Animator    PF_Player (Animator)
Equipment Inventory   PF_Player (Equipment Inventory)
```