Setup

ShopSystem:

After importing the asset, drag ShopSystemCanvas prefab (in Prefabs folder) and drop into the screen. It will come with 3 category and some products in it as default.

Note: By the way, you can unpack the prefab or leave it like that depending on your choice.

Adding Category

1. Create new instance of the CategoryTab prefab as a child to CategoryContainer

2. Create instance of ProductUIArea as a child of ProductUIAreaContainer. This will be the area in which corresponding category will display its products.

3. Click on the CategoryTab object you have added and you will see CategoryUI script and an empty slot named Product UI Area. Add instance of ProductUIArea you have created before to that slot.

Handling Purchasing

To handle purchasing, follow the steps:

1. Add a player game object (if you do not already have)
2. From imported asset, find a script named PlayerShopController and add it to the player.
3. In the player object's inspector, you will see an empty reference slot for ShopSystem, add ShopSystem reference to it.

PlayerShopController

This script implements IShopCustomer which provides all the necessary properties and methods to handle purchasing.

Fields:

_coinAmount – coin amount the player currently has

_shopSystem – a shop system the player is interacting with

Properties:

Inventory – your inventory to store the items, it doesn't matter how it stores items as long as you implement IInventory interface

Methods:

GetCoinAmount() - gets coin amount

SetCoinAmount(int newAmount) - sets coin amount to new amount

HandleBoughtProduct(Product product) - this method firstly decrease coinAmount by the price of the bought product, then call inventory.AddItem() method.

Events:

OnCoinChanged – fired whenever coin amount changes. Whenever you need to display coinAmount somewhere, you need to subscribe to this event.

For example: In the asset, you will see CoinInfoCanvas which just displays the coin, in that canvas there is a game object named CoinText and it has a CoinText script. That script subscribes to this event and updates ui when a change occur.

Displaying Coin Amount in the Screen:

1. Add an instance of CoinInfoCanvas prefab
2. In child named CoinText, there will be an empty slot for PlayerShopController
3. Drag a reference to your PlayerShopController and drop it to the empty slot in CoinText;

Where products are fetched from?

All the products are stored in the ProductContainer game object. There will be a prefab instance in the asset. Drag and drop it to the screen. It will come with default items.

ProductContainer object has a singleton class named ProductContainer (If you do not know about singletons, do not worry, you do not have to deal with that). In the start method, the shop system creates an instance of ProductFetcherFromContainerObject which implements IProductFetcher. Then by calling its Fetch method, it gets list of products.

In the future, if you want to fetch products from kind of json, database, etc. You just need to create a new class which implements IProductFetcher and change the ShopSystem to work with that class. For example:

```
public class JsonFetcher : IProdocutFetcher {
    public List<Product> Fetch() {
        // Fetch products from your json file;
    }
}
```