Blue Gravity Studios Programming Tech Task
Julio Ezequiel González Rojas

Play:
Get close to the counter to interact and press 'E'.

Buy items in buy mode by selecting the item containers. Sell them by accessing Sell Mode and selecting the outfits to sell. You can't sell an outfit you are currently wearing.

How to use the system:

The Project requires 2DAnimation, URP, TextMeshPro and Cinemachine to work properly.

Create new buyable ouftits via Right Click + Create->Product
- Outfit Info (Saves outfit info: Price, description, sprites and sprites libraries)
- Outfit Data Set (Saves OutfitInfo sets)

You'll need to create sprite libraries, which are common when creating a 2D game.

Once a new outfit is created and configured, you can add it to the MainOutfitDataSet, which is used by a ShopManager to allow the player to buy items.

The core of the system is in the 'Player' GameObject. The CharacterInventory manages gold and objects owned by the player and connects with the Shop Manager, which connects with the UI.

When interacting with the UI, it notifies the ShopManager when it presses buttons or add items to sell. It then extracts from them the info it requires to show to the player.

My thought process:

I rely heavily on the MVC pattern, separating it in managers (Controllers), which group logical components and connect them with the UI scripts (View), which update the view and notify the managers of any interaction.

I decompose the scripts in more manageable scripts that operate on smaller objects of the systems (i.e. InventoryUI has ItemsUI to display the inventory, or the CharacterInventory has an OutfitManager).

It gets very complex in bigger systems, so it needs to have careful planning to avoid confussion.

For a task, I start by enumerating the minimum classes necessary. Then I evaluate scalability and readability according to the task needs, and update accordingly.

Assessing my performance:

I wasn't able to write code with the quality I usually deliver due to time constraints. The repository specially suffered from this.

I also planned the system for a much bigger scope, which I shouldn't have done since it affected the code quality, which seems incomplete.

My file organization wasn't also that good. In particular, the sprite libraries and sprite naming were confusing.