Tech Challenge: Clothes Shop

Heitor Oliveira

Introduction

Hello. I’m Heitor, and this is what I’ve manage to build of the Clothes Shop challenge. Unfortunately I couldn’t finish the scene and etc, I’ve paid more attention in code and the “back-end” (if we could call that) aspects of the task, following and MVP architecture.
There’s alto a project in the git repository, where I keep track of my projects through issues, in a simple Kanban board. I’ve also followed some conventions in my commits (mainly prefixes), which can be found here:

https://www.conventionalcommits.org/en/v1.0.0/

A build for windows can be found inside the Release folder.

Architecture

This project follow a simple MVP architecture, which can be found here:

https://unity.com/how-to/build-modular-codebase-mvc-and-mvp-programming-patterns

The idea is to architecture the code in two pillars: Core (from core gameplay) and Management. Usually, there’s some other, like an App namespace that takes care of scene management, I didn’t got there.

Inside each of these namespaces, the code is separated in a Model (data), View (display and UI) and Presenter (a controller middle ground between them), as the MVP pattern follows.

Core

Core comes from core gameplay, and contains the movement of the character, and where it would contain the interactions with the world and the main gameplay loop. In this case, it was developed a State Machine to handle player movement. It inherits from generic State Machine logic, so it could be used for other state machines if needed.

In the beginning of the challenge, I was trying to use UnityEvents for passing functions through Inspector, decoupling code and lowering dependency.

Core has some Input logic as well, and I’m using the new Unity’s Input System.

Management

Management, in this context, refers to things you need to take care in the game that are not part in the core game loop, for example managing items, and buying and selling things. This is where the MVP comes in hand, where I can have Models for Inventory and Shop that only holds data, Presenter who takes care of the logic of the application and a View that display information and handle player interaction with UI. There are a lot of sanity checks to improve stability and could be improves using unit tests.

Gameplay

Right now, you can use WASD for walking around with the Player Character, and the I key for open Inventory, where you can click on items to Equip/Unequip them.

You can use E key to open the shop. You can’t keep Inventory and Shop open in the same time. In shop, you can buy and sell items.

Conclusion

Thank you for the opportunity. I hope you like the logic and architecture I’ve implemented, and interface would be easier to implement in top of that with separated logic. I would love any feedback.