Shop prototype

For the interview game prototype, I needed to implement several systems to make the game work as intended. These systems included character movement, inventory management, a shop system, interaction with a shopkeeper, and currency management.

Character movement

One of the critical aspects of the game was the character movement system. I decided to use Unity's new input system because of its improved performance and flexible input handling.

The new input system provides a more customizable and efficient approach to input processing and allows for more complex input handling of multiple input devices. Moreover, the new input system provides consistent input handling across different platforms, making it easier to develop and test cross-platform games.

To handle the sprite animation of the character, I used the State Pattern. The character has different states, which depend on the direction in which the character is facing. By implementing the State Pattern, I was able to handle the character's sprite animation seamlessly.

Currency

For the currency management system, I used the Model-View-Presenter (MVP) architecture pattern. I created three components: CurrencyView, CurrencyModel, and CurrencyPresenter.

The CurrencyView component represents the view of the currency, and the CurrencyModel component holds the amount of gold. The CurrencyPresenter component handles displaying and updating the gold amount. The MVP architecture pattern allowed me to keep the code modular, making it easier to manage and maintain the currency system.

The presenter component is created in the UIManager and handles all the logic regarding the currency. To ensure that the amount of gold is saved between sessions, I used PlayerPrefs to store the gold amount.

Shop

For the shop system, I used scriptable objects to create shop items. In this prototype, I used colors as the items that could be bought and added to the inventory. When an item is bought, it can be sold back at the same price. All the shop items are loaded at runtime, making it easy to manage the shop system.

Inventory

After an item is purchased, it will be unlocked in the inventory, ready to be equipped. The inventory system is persisted by parsing the list of unlocked items to a JSON-formatted string and saving it in PlayerPrefs to keep the items between play sessions.

In summary, by implementing these systems, I was able to create a fully functioning prototype game. The use of Unity's new input system, State Pattern, MVP architecture pattern, and scriptable objects allowed me to create a flexible, efficient, and modular codebase that was easy to manage and maintain.