Sim Clothes Shop Documentation

Introduction

The “Sim Clothes Shop” is a game that allows players to browse, purchase/sell clothes using in-game currency and then displaying and equipping them at player character. This document will look to explain the thought process during the project as well as how the system works and at the end a personal assessment from the overall result.

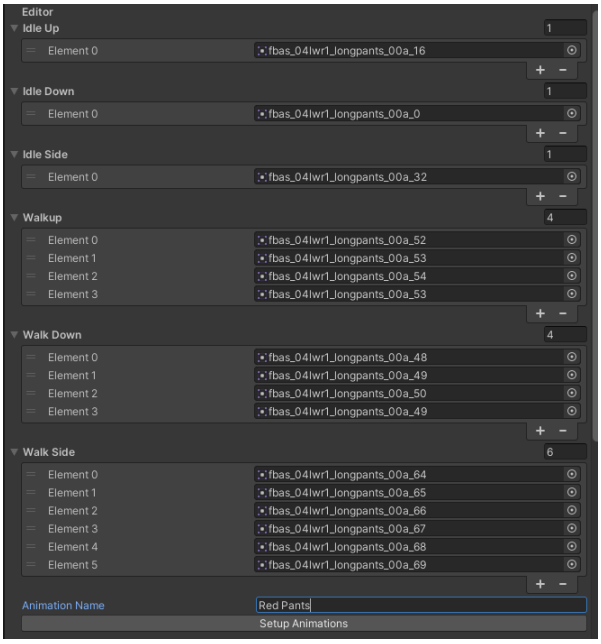
System Overview

For the basic top down functions, it’s used 3 scripts, PlayerMoveControllers, which is responsible for the simple movement using rigidbody 2D, PlayerAnimationController, which is responsible for handling all the player animations, and the PlayerInteractionController which is responsible for basic interactions with objects, on this demo is used only for the Shopkeeper npc, but any objects with the InteractableObject script and layer Interactable, can be interacted.

To manage all player data it uses the script PlayerData, which is set as a singleton so it can be easily accessed from other scripts. It holds values for the equipped items, items at inventory and the player money, which are all also saved and loaded at memory with the plugin ES3.

For handling the different equipped items at the player, it’s used a sprite renderer for each cloth type, and then attached a specific item animator controller that overrides from main player body animation and will play a specific animation based on the current at player. The script responsible for setting the animator controller for each cloth is the EquipItemsController, which checks if the player has any item equipped at the slot and sets the respective animator controller when there is.

In order to simplify the process for creating a animator and animations for each game item, the file AnimationSetupEditor, automize most of work, by just dragging the proper sprites at the Inspector and naming the item, we can create the item animator together with all proper animations in one click, they are then placed at Assets/2D/Animations/BodyItems.



After setting the animations a item can be added to the game through the ItemsData script, which holds a reference and the values for each item at the project and is also set as a Singleton for easy access.

For the inventory display and logics, it uses PlayerInventoryController, which loads a list of the player inventory items as a horizontal scroll and the Equipped Items both from PlayerData, items can be equipped when clicked from inventory scroll and unequipped when clicked from player item slots.

And lastly for the shop itself, the scripts responsible is the ShopKeeperController, items are displayed at a dynamic scroll list and separated based on the type of the item, they are then displayed either at Buy or Sell tab, ShopKeeperController will read from all the items at ItemsData and display then to be buyed, and from all inventory items at PlayerData to be selled, the sell value ratio can be configured by SellRatio variable, so that for example items loses 20% of values when selled.

Thought Process During Project

At the beginning of the project, it was written an overall workflow of all the architecture of the game, as well as an overview for how each system would work and how they would connect between each other. It was decided to use the new input system, since it allows for an easy and better maintainable setup of the controls.

One of the questions that I got thinking at the start, was how would be the best way to make the items display at the character together with animations, I considerer using 2 approaches, 1 would be to use a single animation for each movement, and for each item slot, I would toggle the objects at the animation for each frame, then later I would just replace the sprites by code respective to each item, the advantage of using this approach is that it was going to be a lot easier to add each item, since I would have to just drag the sprites to a field at inspector, but the downside is that was going to be a lot harder to initially setup the first animations and it would not be very organized to have all one object for each frame being toggle on/off, neither very optimal for expansion, so the other approach that I decided to go was just using a animator override controller, that would copy the animator at the base body of player, so it would have a respective animation for each of the player body animation, and to resolve the issue of having to create one animation for each at the player (total of 6) as well as the animator override, and then connecting each animation at the animator, I decided to wrote a code, so that I could just drag and drop the sprites in a field for each animation, choose a name for the item and automatize the rest of process by code, this way it got a lot more pratical.

Overall Assessment

In the end I believe I delivered a satisfactory result, I manage to make all the requirements of the task in a direct and simple to understand code, that can be easily expandable and tweaked, but as for things that I could have done better, I will say I could have made the interaction at shop more dynamic, maybe a dialog box for the shopkeeper, as well as adding more animations in general at UIs and shopkeeper. And another thing would be to use better suited assets, I noticed that the shoes animation was not playing properly, I checked the images but couldn’t find the respective animation for body moves. I also could

have spent a little more time adding more details to the environment. It was a nice experience working on this demo and I believe I got to learn more about a few things.