Introduction

Out of hundreds of people who were invited on Upwork, i was very happy to found out that i was eligible for a test task to become a part of Blue Gravity Studios. After i did my research about the studio and a brief discussion, i was given my task, and set out to work on it immediately , which i have managed to complete in around 72-74 hours.

Examining The Tasks

Once i was given the task, i examined it carefully.

I noticed that the tasks heavily depended on character interaction, so i deemed cruical for my success to write a proper base class for character entities.

Getting Started

Until now, my experience with 2D games were limited, so the first thing i did was to learn the process to make a proper 2D game in Unity which took me around 2-3 hours. I was ready to make the project.

My Development Style

Before i dig deeper into the project, i think it's better that i explain how i make develop games.

1) Developer Friendly Code

First and foremost, i am a firm believer that the best code is one that doesn't require a lot of setup. I tried my best to write code that took care of itself, instead of me having to specify every single variable's value in Unity Editor.

The biggest example of this is the fact that many of the gameobjects in my project will create itself, with minimal interaction from the developer. Keeping everything clean and under control.

Attach "BaseCharacter" component to a GameObject? When you hit play, you will have a fully functional character spawned in ready to be interacted with. That's all it takes.

2) Dynamic, Flexible Code

Second, i value code that can fit many scenarios and can be extended, with minimal copy-pasting required. I wrote the character classes with this in mind. While i wanted to do this approach for the entire project, time constraints made it a waste of time to develop an elegant solution for each part of the game

3) Clean, Understandable Code

Third, i absolutely try my best to write clean code, following naming conventions, leaving comments/summaries, utilizing abstraction and many more. It's important to me that when someone reads my code, they are able to make sense of it.

How The Game Works

The game in technical side, primarily works by the close communication of the player and UI entities.

The game has been written from SCRATCH, and i can guarantee there is simply no case where i lifted code from anywhere, or used third party libraries. The project was written in 1827 lines.

Like i mentioned, most of the GameObjects create themselves at start, including the UI and characters.

UI

UIManager will load the game UI resource and spawn it, this resource contains every single UI made and activates/enables them as needed. It also communicates with various entities. Each UI component has its own script attached for its purpose.

(UI Manager -> Shopping UI -> Shop Salesman)

(UI Manager -> Inventory UI -> Player Inventory)

Character

There are two types of characters, **BaseCharacter and CharacterPlayer**. Any GameObject which has any of these scripts attached will automatically spawn a character. CharacterPlayer inherits from BaseCharacter.

The characters are very modular, and every single line of code related to them is not jam-packed into one class.

CharacterAnimEvents – Extra animation event functions

CharacterBase – Controls direction/animation updates, physics, responsible for some animation events

CharacterAppearance – Appearance module that controls outfit

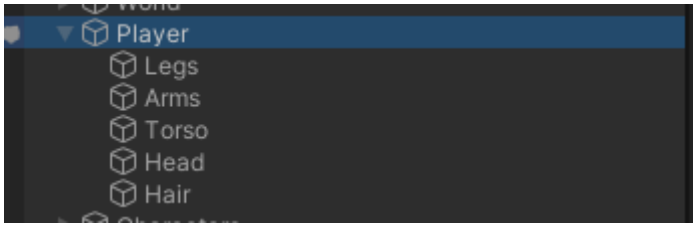
PlayerController – If the character is the player, this lets them move

Inventory – A generic inventory class that can be used for all things

Character Appearance

The characters consist of 5 layers (legs, torso, arms, head, hair) and thus compromise of 5 sprites to be fully customizable part by part.

The player character, before being created at play mode



The player character, after spawning

Character textures are stored in Resources as Sprite Atlas'es and follow a very specific naming convention

variant0_lft_0 – The character is using the first style, is facing left, and is at animation frame 0.

Items

Items are ScriptableObjects that have a buying, selling price, name, description and a category.

Speech

The speech system mimicks those that belong to popular indie games like Undertale, a system where the character speaks letter by letter and makes a sound for each letter.

Conclusion

I hope this document explains my train of thought and process while making this game nicely. I thank you once again for this opportunity.

All things considered (that i never had to make a game in 3 days and had no proper 2D experience) i think i have done well and i am personally proud of my work.