Little Sim World Interview Project

The System

Movimentation:

There are two ways of moving the character, the first is by pressing the desired direction on the keyboard, this will give the character a new location to move with Unity's "MovePosition" method. The direction is combined with the speed and time to get the target location.

The other way to move is by pressing the right mouse button at any point on the ground. The first thing that happens when the mouse is clicked is gather the position based on the screen and change it to world position, after that, the player receives a target position to calculate the path. The path is calculated with the "a* algorithm". This part of the code take some time to optimize, so the code in the project is unoptimized and follows the base introduced by Sebastian Lague in the "A* Pathfinding" (Algorithm Explanation). To get the right path, the method is used in a grid of nodes, each node has information about position, possibility of walking on it, energy cost based on movement, distance from target, neighbour nodes and xy coordinates in the grid. The path node that leads to the target is added to the character private path queue, for each node in the path, the target will move at its position with the default speed.

All moveable objects that may need a path, like Npcs, inherit from the same script, this way we can have multiple path movement based characters.

Movimentation can occurs to any xy directions, every time a character moves he will receive the movement base direction (up, down, left or right) so he can update everything that needs a direction type variable to work, like the x sprite scale, camera "lookahead" and the player animator.

Sorting:

Sorting is made by setting a base number to all objects, then subtracting from this number the y position of the base sort position.

The player, and most of the objects that have more than one sprite object in its hierarchy are sorted based on their position plus the offset in the script, objects that aren't static keep being sorted in the "LateUpdate" method because they need new sorting every time they can possibly move. Static objects are sorted once at start.

If the object doesn't have more than one sprite object, it will be sorted with the y position of it's render's bounds.

Player:

The player can be settled as a copy just to be a player representation. The player can also have its movement and input control turned off, for moments where the player needs to wait for something. He can also execute “Unity Events” when reaching a near position from the selected object, for this was used the path from movimentation and compare with the desired interacted object, so player keep chasing the desired object to interact if it moves, after he achieves the desired object he receives the events delegate and execute all the events.

When the player moves it changes it’s animation based on the moved direction, he also changes the x scale of it’s sprites so they match the direction it is looking.

Player’s body is made of different parts, he can change the sprite of each part to change his clothes, these parts are also used to animate the character, any change made beside those inside the animation will persist, so the character can be easily animated and customized.

The animation for the player is made with a blend tree between xy values getted from the input xy, in this project i made only two animations due time limit, but the way is setted you can have unlimited animations, one for each direction you want.

NPCs:

The NPCs can do most of the things the player can, since they inherit the same movement script, the only difference is that npcs receive a random target position to move every a random couple of seconds.

NPCs can also have multiple dialogues setted for when the player interacts with them.

Dialogue System:

For the dialogue system a “Dialogue” type variable was created, it stores an array of sentences, a events delegate and a sprite to indicate the npc who is talking.

When the dialogue starts, it gives to the “GameController” script itself as reference so any npc can access it and start the dialogue. When the player invokes the dialogue from the npc the dialogue system receives the npcs dialogue list and starts showing it, every time a dialogue ends it is removed from the npc dialogue list, unless it is a generic dialogue for the npc to randomly repeat.

To show the dialogue, a string is created and it receives character by character from the sentence, to animate it, this is done in a coroutine that waits the desired time before adding another character to the dialogue text. A coroutine is also used for the dialogue box to fade out when there are no more sentences to display.

Instead of waiting for the dialogue to fully be displayed the player can skip it by clicking on the dialogue box, to go to the next sentence the dialogue box will always wait for the player to confirm the change.

In the corner of the box, there is a display of the character who's talking, his hat will always change to the hat selected in each sentence, this happens so there can be a dialogue with multiple characters at once but receiving the dialogues queue from just one.

Money System:

To show the player's current money a simple UI was created that has all the information from the player's money change. Every time the money value is changed a text indicating the amount of the change is shown and the current money text is slowly updated until it is the same value of the current money. When money change is positive the indicator will have a color and another if it is negative, beside that i will show the signs "+" or "-" depending on the change value.

To have a fun way to get money, a money bag was created for the player to catch, or you can use the key "M" to give money to the player.

Mouse Cursor:

The mouse cursor represents all the feedback needed for the player to interact with the game, every time the cursor is over a object i will change if the object is interactable, when clicking on a object the cursor changes for a millisecond and then turn back into the default cursor to indicate that a interaction was initiated.

There are four cursors, one for interactable objects, other for buttons, other for the default cursor and other to indicate a confirmation in action.

Store and Clothes:

All clothes and most of the game sprites, beside tiled textures, were made for this project, the character is based on a model from another self project and he's a man with a tomato head.

The store can be accessed from talking to a npc behind a counter inside it, after a quick dialogue the store opens with a simple animation and shows some clothes hanger with a cloth of a specific body part from player, clicking in the hanged clothes change the store section, each section can have it own catalog.

The selected cloth is showed with its name, body part and price, if the player has the amount of money necessary to buy it and clicks in the buy button, the cloth will be setted to bought, then the buy button becomes a equip button, if equipped the cloth will be equipped in the player representation on the store and the actual player outside. If the player decides to unequip the cloth, in the same place where he bought it he will be able to sell it or unequip it, selling will give back the cost of the cloth to the player.

Camera:

The camera follows the target and displays a little forward in the direction he is facing so the player can look more ahead of your path.

There are bounds for the camera movement, so it stays in the desired position, those bounds are also used to verify the new position for the player to move in, if the position is out of bounds the movement is canceled.

Game Controller:

The game controller is a singleton used for gatter most references in a scene and setting the default settings for sorting. It is also used to get the mouse input and verify with a ray the collisions with UI buttons.

Most references are automatically setted at start and the singleton creates an instance persistent and public for every other script to access it, which makes acquiring references a lot easier.

Other Comments:

There was added a shader for grass based on Code Monkey's shader tutorial, but with some tweaks.

There are some simple codes used to quickly add some effects and animations, like auto rotate or auto move, others are used for delivering a quick delegate event in animation or trigger.

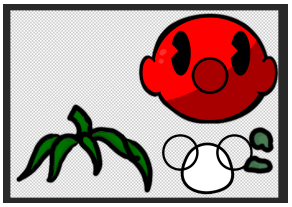
Some extensions were created just to exemplificate their use, even though they work, they weren't really necessary, like some other parts like try and catch.

The resources folder was used to quickly and easily access files.

Attachments:

 - Base for character design from my other project.

- My thinking process

  - Character creation process with mouse and basic forms

My Thought Process

My Pipeline:

For the most part I usually do things a little differently, but for this project I went for the quick way to start and have something.

First i planned everything needed and wrote it down, after that i drew it to have a better concept.

I did the most relevant and time consuming things with “simple shapes”, then focused on little things to make everything a little prettier.

When something was finished, before going to the next I started testing it, then I fixed every bug and just after that I went to another task.

Once everything was finished, I started to make a review. If something could be improved and wouldn't consum much time I would make some changes and even add something more that I haven't thinked.

My Decisions:

At first I always search for other methods besides the ones I usually do to see if there is a better one, or in some cases just a quick one.

With the list of things to do i had the thing i would like to do, by the time i stick just to what's more relevant and won‘t affect the required things, like audio that i decided not include.

My Opinion On My Perform

I wouldn’t know for sure if I did well or not, but I did my best to accomplish all the requirements in the test description and deliver a project that I would be proud to say it’s mine. I understand that a thing or another could be improved, but i never knew a project where it couldn't, i don’t like to think just in the outcome, i maybe started this project for a reason, but while i was doing it, it felt like the only reason i needed was to see the forms taking shape and that little world making every time more sense.

- Thank you for the opportunity, I was thrilled to be able to do it, hope you enjoy it.