Assignment 1: 2D Platformer Game Development - Godot Engine

Objective:

The objective of this assignment is to develop a 2D platformer game using the Godot Engine. The game should include a player character with animations for idle, movement, jump, and attack, as well as basic enemy AI. This assignment will help you explore the simple animation as well as finite state machines to implement fundamental game mechanics.

0. Environment:

0.1 TileMap:

- Create a simple AutoTiling & Physics Layer in a tilemap to set up the environment for the player.

0.2 Background:

- Use a TextureRect with a background image as the texture. Set the stretch mode to “Tile”

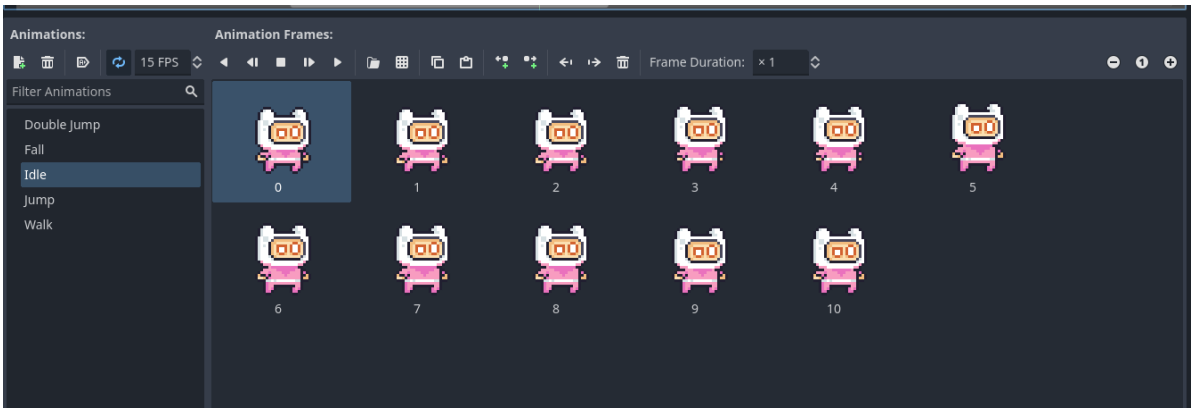
1. Player:

1.1 Movement:

- Add movement using the `move_and_slide` function.
- Implement horizontal movement (left and right) based on user input.
- Implement jumping mechanics using an upward velocity when the jump key is pressed.

1.2 Animation:

- Set up the AnimationSprite2D with all the animations(idle, run, jump, Fall).



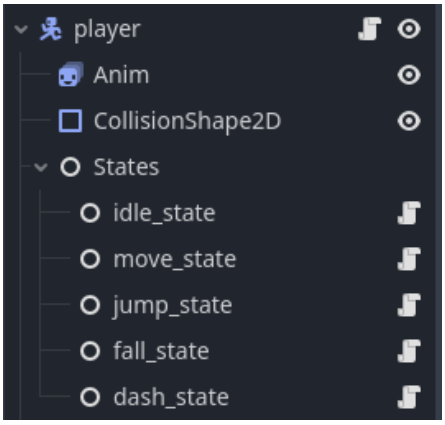
- Access the AnimatedSprite2D and play the corresponding animations in the code depending on the behavior.
- Flip the character depending on the direction. (Hint: look at the properties inside the AnimatedSprite2D, specifically Flip_h)

1.3 Camera:

- Add a Camera2D that follows the player and limits the camera within the level boundaries.
- Experiment with properties to make the camera movement smooth and fun.
- Suggestions: Play with values related to smoothing or drag.

1.4 State Machine (Optional):

- Create 4 class nodes (Idle, Move, Jump, Fall) and add them as nodes in the player.



- Add a function “change_state(new_state_name: String)” that takes in the name of state you want to switch to. This function has 2 main parts:
    - Keep track of the “current_state” (hint: use a variable outside the function to do so)
    - Call a “reset_node()” function in the state it changes into (Ex: If you change states to Idle, then it calls the “reset_node()” function inside the idle state node.)
- Add the corresponding functionality to each state machine (Ex: play the idle animation and do the idle “stuff” inside the idle state)
- Make sure each state has a way to “change_states” from one state to another. There generally should be a way to go full circle from all states to another.

2. Enemy AI:

2.1 Basic AI:

- Create a simple enemy that automatically walks in one direciton.
- The enemy can switch direction on 2 different conditions:
    - If it hits a wall.
    - If it is about to fall off a cliff
- Use an Area2D to detect If the player has “hit” the sides of the enemy. This should damage the player.
- Use an Area2D to detect if the player has “hit” the head of the enemy. This should kill the monster.

3. Extra:

3.1 Collectibles:

- Use an Area2D as the base of the node that allows the player to “collect” this object.
- Add an AnimatedSprite2D to Default for any image you want (Ex: Apple or Banana)
- Play the Animation “Collected”, then queue_free() the object. NOT when the object touches the player.

3.2 Game Over Overlay:

- Use a CanvasLayer, Label, Button and Panel to create a Game Over Overlay.
- The button should reset the level. (I encourage you to research on your own for this part, to find out how to reload a level.)

3.3 Audio:

- Use a AudioStreamPlayer2D to play sounds in different places that you choose. A good starting place is, the player, when he jumps or does damage to a enemy.
- Add sound to the collectibles when collected by the player.

Bonus:

- Implement a double jump for the player.
- Update the player's score when collected.
- Add health and damage mechanics for both the player and enemies.
- Create a basic UI to display the player's health and score.
- Design a simple level using TileMap with collisions.