Godot Game

Made by: Sebastijan Zindl - 211557

Project Download

Because some of the assets are to big I couldn't upload the file directly to iKnow. Here is the Github Link the gameplay of the game can be found on this Youtube Link

Controls

`Left, Right, Up, Down` Arrows for Player Movement

`A` or `D` for Camera Movement

Feature explanation

1. Player & Movement
   The movement of the player is done by creating a custom player scene that includes a `CameraController` that dynamically moves the camera with the player.

The main movement actions are provided by Godot, but the camera isn't. The player can turn the camera 90degrees using the A and D keys by using custom input mappings ('cam_left' and 'cam_right'). The direction is also interpolated by using the CameraController's basis as a base for the direction

```
if Input.is_action_pressed("cam_left"):
	$Camera_Controller.rotate_y(deg_to_rad(-90))

if Input.is_action_pressed("cam_right"):
	$Camera_Controller.rotate_y(deg_to_rad(90))

var direction = ($Camera_Controller.transform.basis * Vector3(input_dir.x, 0, input_dir.y)).normalized()
$Camera_Controller.position = lerp($Camera_Controller.position, position, 0.15)
```

2. Coins
   The coins are placed throught the level and they emit a signal when a collision is detected.

```
func _on_body_entered(body: Node3D) -> void:
        emit_signal("coin_collected")
        queue_free()
```

In the main level file upon creating we loop through all of the coins and add the function that increases the `coins_collected` and `score` variables as they are kept in the level file.

```
    for coin in $Coins.get_children():
        coin.connect("coin_collected", collect_coin)
```

3. Win/Lose Conditions

The game is won or lost based on two scenarios:

- If the player collects all of the coins in the given time he win's the game.
- If the player doesn't succeed with collecting all of the coins and the timer runs out, he loses.

The timer is setup on the `_ready` method of the level, when it runs out it calls the `_on_timer_timeout` function which ends the game.

```

    timer_node.wait_time = 120
    timer_node.one_shot = true
    timer_node.connect("timeout", _on_timer_timeout)
    timer_node.start()

    func _on_timer_timeout():
        if coins_collected < total_coins:
            game_over()
```

4. Win/Lose Screens
   The win and lose screens are custom control scenes with their own text and panels. Both of the screens have the same functionality, evidently in the future this can be a one panel where dynamically the text is changed.

They have their `process_mode` set to `Node.PROCESS_MODE_ALWAYS` so they work even if the main loop stops. Besides this they emit a signal when the retry button is pressed `retry_game` which we listen in the level file and restart the game if it happens. There is also a public function which I use to set the labels from when they need updating.

Upon running the game both of the screens are hidden and in their respective scenarios they are both called to be shown along with the function to update their labels.

Win/Lose Screen Script

```
func _ready() -> void:
    process_mode = Node.PROCESS_MODE_ALWAYS

$PanelContainer/VBoxContainer/HBoxContainer/RetryButton.connect("pressed",
_retry_pressed)

func set_labels(score: int, coins: int):
    $PanelContainer/VBoxContainer/FinalScoreLabel.text = "Total Score: " +
str(score)
    $PanelContainer/VBoxContainer/CoinsCollectedLabel.text = "Coins
Collected: " + str(coins)


func _retry_pressed():
    emit_signal("retry_game")
```

Main Script Logic

```
func _ready() -> void:
    win_screen.hide()
    lose_screen.hide()
    ...
    win_screen.connect("retry_game", _on_retry_pressed)
    lose_screen.connect("retry_game", _on_retry_pressed)

func game_win():
    timer_node.stop()
    get_tree().paused = true
    win_screen.show()
    win_screen.set_labels(score, coins_collected)

func game_over():
    get_tree().paused = true
    lose_screen.show()
    lose_screen.set_labels(score, coins_collected)

func _on_retry_pressed():
    get_tree().paused = false
    score = 0
    coins_collected = 0
    get_tree().reload_current_scene()
```

5. Score Text Updating
   The score text is updated whenever a coin is collected

```
func update_score_display():
    $ScoreLabel.text = "Score: " + str(score) + "\nCoins: " + str(coins_collected) + "/" + str(total_coins)
```