Lesson 10: Finish the Prototype

In this lesson, students will learn how to implement a coin collection mechanic in a game. They will create a variable to store the player's score, detect when the player collects a coin using collision detection, and set up a score text to display the score during gameplay. Additionally, students will learn how to set up an end flag to trigger a change in the game when a certain condition is met. Through hands-on practice and group activities, students will gain a solid understanding of these game development concepts and be able to apply them in their own projects.

## **Objectives:**

- Students will learn how to create a variable to store the score in a game script.

- Students will understand how to detect when the player has collected a coin.

- Students will be able to set up a score text to display the score during gameplay.

- Students will learn how to set up an end flag to trigger a change in the game.

## **Materials:**

- Computer with game development software installed (e.g., Godot Engine)

- Project files for the game (e.g., Player.gd script, Coin scene)

- Projector or screen to display instructions and code examples

## **Bell-Ringer Activity:**

1. Display an image or video of a popular game that involves collecting coins (e.g., Super Mario Bros., Sonic the Hedgehog).

2. Ask students to share their thoughts on why collecting coins is important in a game.

3. Facilitate a brief discussion to explore the reasons behind collecting coins in games and how it adds to the gameplay experience.

## **Introduction:**

1. Explain to the students that in today's lesson, they will learn how to implement a coin collection mechanic in a game.

2. Emphasize the importance of keeping track of the player's score and triggering events when the player collects a coin.

3. Briefly introduce the concept of variables and how they can be used to store and manipulate data in a game.

## **Direct Instruction:**

1. Demonstrate how to create a variable to store the score in the Player.gd script:

```gdscript

var score = 0

```

- Explain that the variable "score" will be used to keep track of the player's score throughout the game.

- Mention that the initial value of the score is set to 0.

2. Show how to create an Area2D node and name it "Coin":

- Explain that the Area2D node will represent the coin in the game world.

- Mention that the name "Coin" will be used to refer to this node in the code.

3. Discuss how to detect when the player has collected the coin:

- Explain that the Area2D node representing the coin should have a collision shape attached to it.

- Demonstrate how to add code to the Player.gd script to detect collisions with the coin:

```gdscript

func \_on\_Player\_body\_entered(body):

if body.name == "Coin":

score += 1

body.queue\_free()

```

- Explain that the `\_on\_Player\_body\_entered` function is called automatically when the player's body enters the collision shape of another body.

- Mention that the `body` parameter represents the other body involved in the collision.

- Explain that the code checks if the name of the other body is "Coin" and if so, increments the score by 1 and removes the coin from the game.

4. Show how to set up a score text for the game:

- Explain that a score text will be displayed on the screen to show the player's current score.

- Demonstrate how to create a Label node and position it on the screen.

- Show how to add code to update the score text whenever the score changes:

```gdscript

func \_process(delta):

$ScoreLabel.text = "Score: " + str(score)

```

- Explain that the `\_process` function is called every frame to update the game logic.

- Mention that the code sets the text of the ScoreLabel node to the current score.

## **Guided Practice:**

1. Instruct the students to follow along and implement the code demonstrated in the Direct Instruction section.

2. Provide assistance and answer any questions as the students work on creating the variable, setting up the Coin node, detecting collisions, and updating the score text.

## **Independent Practice:**

1. Divide the students into pairs or small groups.

2. Assign each group the task of creating an end flag for the game.

3. Instruct the groups to discuss and plan how they will implement the end flag, considering factors such as the trigger condition and the desired change in the game.

4. Allow the groups to work independently to write the code for the end flag in the Player.gd script.

5. Encourage the groups to test their implementation and make any necessary adjustments.

## **Exit Ticket:**

1. Distribute an exit ticket to each student.

2. Ask the students to briefly explain the purpose of the score variable in a game and how it is used.

3. Collect the exit tickets to assess the students' understanding of the lesson objectives.

## **Closure:**

1. Recap the key points covered in the lesson:

- Creating a variable to store the score in a game script.

- Detecting when the player has collected a coin using collision detection.

- Setting up a score text to display the score during gameplay.

- Implementing an end flag to trigger a change in the game.

2. Emphasize the importance of keeping track of the player's progress and providing feedback through the score and other game mechanics.

3. Encourage the students to continue exploring game development and experimenting with different gameplay elements in their projects.