# Lesson 3 Quick Reference Guide

## Collectible Coins & Score System

Duration: 40 minutes

Godot Version: 4.5

Prerequisites: Completed Lessons 1 & 2

## 🎯 What We're Building Today

By the end of this lesson, your game will have:

- ✨ Collectible coins placed throughout the level
- 📊 A score counter that updates when coins are collected
- 🔄 **Your first reusable scene** (Coin.tscn)
- 📡 Working signals for event-driven programming

New Concept: Making one scene (Coin) that can be used many times!

## 📚 Key Vocabulary

Term	**Meaning**	**Example**
Scene Instancing	Creating multiple copies from one template	One Coin.tscn → 10 coins in level

Signal | Event notification system (like a doorbell) | Coin emits "I was collected!" signal

queue_free() | Delete this node safely | Coin disappears when collected

String Concatenation | Combining text | "Score: " + "5" = "Score: 5"

Emit | Send out a signal | Broadcasting an announcement

## 🏗️ Project Structure After This Lesson

```
Year9_Platformer/
├── level_1.tscn        ← Main gameplay scene
├── level_2.tscn        ← Victory screen scene
├── player.gd          ← Player movement script
├── coin.tscn          ← NEW! Reusable coin scene
└── coin.gd            ← NEW! Coin behavior script
```

New Files: coin.tscn and coin.gd

## 🪙 PART 1: Create the Coin Scene (5 minutes)

### Step 1: Create New Scene

1. Click **Scene menu** → **New Scene** (top left)

2. Choose **Other Node**

3. Search for: `Area2D`

4. Click **Create**

5. **Rename** the root node: Right-click Area2D → Rename → `Coin`

Why Area2D? It detects when other objects touch it (like the player)!

### Step 2: Add Collision Shape

1. Select the **Coin** node

2. Click the **+** button (Add Child Node)

3. Search: `CollisionShape2D`

4. Click **Create**

5. In the **Inspector** (right side):

   - Find **Shape** property

   - Click **[empty]** → **New CircleShape2D**

   - Click the **CircleShape2D** to expand it

   - Set **Radius:** `16`

Visual check: You should see a circle outline in the viewport!

### Step 3: Add Visual Appearance

1. Select the **Coin** node again

2. Add child: **ColorRect**

3. In the Inspector, set:

  - **Size:** `32 x 32`

  - **Position:** `-16, -16` (centers it on the collision)

  - **Color:** Yellow/Gold (click the color box)

Tip: You can also use a Sprite2D with a coin image if you have one!

### Step 4: Save the Coin Scene

1. Click **Scene menu** → **Save Scene**

2. Name it: `coin.tscn`

3. Click **Save**

✅ **Checkpoint:** You should see `coin.tscn` in your FileSystem panel (bottom left)!

## 💻 PART 2: Add Coin Script with Signal (8 minutes)

### Step 5: Attach Script to Coin

1. Select the **Coin** node (root of coin.tscn)

2. Click the **script icon** (📄) in the toolbar

3. Path should be: `res://coin.gd`

4. Click **Create**

### Step 6: Write the Coin Script

DELETE ALL the template code and replace with this:

```gdscript
extends Area2D

# === SIGNAL DEFINITION ===
# This signal announces "A coin was collected!"
# Other scripts can listen for this event
signal coin_collected

# === INITIALIZATION ===
func _ready():
    # Connect our collision detection to our response function
    # When something enters our area, call _on_body_entered
    body_entered.connect(_on_body_entered)

# === COLLISION DETECTION ===
# This function runs when ANY body touches the coin
func _on_body_entered(body):
    # Check if the body that touched us is named "Player"
    if body.name == "Player":
        # Announce that this coin was collected
        coin_collected.emit()
        # Delete this coin from the game
        queue_free()
```

Save: Press **Ctrl+S** (Windows) or **Cmd+S** (Mac)

### 🧠 Understanding the Code

Line	**What It Does**
`signal coin_collected`	Creates a signal other scripts can listen to
`body_entered.connect()`	When something touches coin, call our function
`if body.name == "Player":`	Only respond if the Player touched us
`coin_collected.emit()`	Send the signal (ring the doorbell!)
`queue_free()`	Safely delete this coin

Think of it like: The coin rings a doorbell (emit signal) before disappearing!

## 🎮 PART 3: Place Coins in Level (5 minutes)

### Step 7: Switch to Level Scene

1. Click the **level_1** tab at the top of the editor

2. Make sure you can see your platform and player

### Step 8: Instance Your First Coin

1. **Drag** `coin.tscn` from the FileSystem panel into the viewport

2. **Position** it above the platform using the move tool

3. Click the viewport to deselect

Important: You just created an **instance**, not a copy!

### Step 9: Add More Coins

Quick Method:

1. Select the coin you just placed

2. Press **Ctrl+D** (Windows) or **Cmd+D** (Mac) to duplicate

3. Move the new coin to a different position

4. Repeat 5-8 times to create a trail of coins

Suggested layout: Place coins in a line across the platform, or create a pattern!

### Step 10: Save Level

Press **Ctrl+S** / **Cmd+S**

✅ **Checkpoint:** You should see multiple yellow/gold coins in your level!

## 📊 PART 4: Create Score System (8 minutes)

### Step 11: Add Script to Level

1. Click on the **Node2D** root node (level_1)

2. Click **Attach Script** icon (📄)

3. Path: `res://level_1.gd`

4. Click **Create**

### Step 12: Write Level Script

Replace ALL template code with:

```gdscript
extends Node2D

# === SCORE TRACKING ===
# This variable stores the player's current score
var score = 0
```

```
# === INITIALIZATION ===
func _ready():
    # Find ALL coin instances in this level
    var coins = get_tree().get_nodes_in_group("coins")

    # Connect to each coin's signal
    for coin in coins:
        coin.coin_collected.connect(_on_coin_collected)

# === COIN COLLECTION RESPONSE ===
# This function runs whenever ANY coin emits its signal
func _on_coin_collected():
    # Increase score by 1
    score += 1

    # Update the UI display
    $ScoreLabel.text = "Score: " + str(score)

    # Print to console for debugging
    print("Score: " + str(score))
```

Save: Ctrl+S / Cmd+S

### 🧠 Understanding the Score Code

Line	**What It Does**
`var score = 0`	Creates a variable to track points
`get_nodes_in_group("coins")`	Find all coins (we'll add them to group next)
`coin.coin_collected.connect()`	Listen for each coin's signal
`score += 1`	Add 1 to score (same as `score = score + 1`)
`str(score)`	Convert number to text for display
`$ScoreLabel.text`	Update the UI label

### Step 13: Add Coins to Group

We need to tell the coins they're in the "coins" group:

1. In level_1.tscn, select **any coin instance**
2. Look at the **Node panel** (next to Inspector)
3. Click **Groups** tab
4. Type: `coins`
5. Click **Add**
6. **Repeat for EVERY coin** in your level

Shortcut: Hold **Shift** and click all coins → Groups tab → Add all at once!

## 🖥️ PART 5: Add Score Display (5 minutes)

### Step 14: Create Score Label

1. Select the **Node2D** root of level_1

2. Add child node: **Label**

3. Rename it: `ScoreLabel`

4. In the Inspector, set:

  - **Text:** `Score: 0`

  - **Position:** `20, 20` (top-left of screen)

  - **Theme Overrides → Font Size:** `32`

Visual check: You should see "Score: 0" in the top-left corner!

### Step 15: Save Everything

Press **Ctrl+S** / **Cmd+S**

## 🧪 PART 6: Test Your Game! (7 minutes)

### Step 16: Run and Test

1. Press **F5** to run the game

2. Move the player to collect coins

3. **Watch for:**

  - ✅ Coins disappear when touched

  - ✅ Score increases in top-left

  - ✅ Console shows score updates

4. Press **F8** to stop

### 🐛 Common Problems & Solutions

Problem	**Cause**	**Solution**
Coins don't disappear	Script not attached	Check coin has 📄 icon
Score doesn't update	Coins not in group	Add each coin to "coins" group
"Node not found: ScoreLabel"	Label named wrong	Check exact spelling: `ScoreLabel`
Can walk through coins	No CollisionShape2D	Add collision shape to coin
Score updates but coins stay	Missing queue_free()	Check coin.gd has `queue_free()`
Nothing happens	Signal not emitted	Check `coin_collected.emit()` line

## ✅ Success Checklist

Before finishing, verify:

- [ ] coin.tscn exists in FileSystem panel

- [ ] Coin has Area2D, CollisionShape2D, and ColorRect

- [ ] coin.gd script attached to Coin (📄 icon visible)

- [ ] At least 5 coins placed in level_1

- [ ] All coins added to "coins" group

- [ ] ScoreLabel created and positioned

- [ ] level_1.gd script attached to root Node2D

- [ ] Coins disappear when collected

- [ ] Score increases from 0 → 1 → 2 → etc.

- [ ] All files saved

## 🎮 Controls Reference

Key	**Action**
← →	Move player
Spacebar	Jump
F5	Run game
F8	Stop game
Ctrl+S / Cmd+S	Save
Ctrl+D / Cmd+D	Duplicate selected node

## 🚀 Challenge Extensions

### Challenge 1: Count Total Coins

Show "Score: 3/10" instead of just "Score: 3"

```gdscript
# In level_1.gd, add to _ready():
var total_coins = coins.size()
$ScoreLabel.text = "Score: 0/" + str(total_coins)

# Update _on_coin_collected():
$ScoreLabel.text = "Score: " + str(score) + "/" + str(total_coins)
```

### Challenge 2: Different Coin Values

Make some coins worth 5 points!

In coin.gd:

```gdscript
# Add at top
var coin_value = 1  # Can change in Inspector per coin!

# Change emit line to:
coin_collected.emit(coin_value)
```

In level_1.gd:

```gdscript
# Update function to accept value:
func _on_coin_collected(value):
	score += value
```

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

Then select some coins and change their `coin_value` to 5 in Inspector!

### Challenge 3: Coin Collection Sound

Add satisfying audio feedback!

1. Find a coin sound effect (freesound.org)
2. In coin.tscn, add **AudioStreamPlayer** child node
3. Drag sound file into **Stream** property
4. In coin.gd, add before queue_free():

```gdscript
$AudioStreamPlayer.play()
await $AudioStreamPlayer.finished
queue_free()
```

## 🤔 Reflection Questions

Answer in your exercise book:

1. What's the difference between **duplicating a node** and **instancing a scene**?
2. Why do we use signals instead of directly calling functions?

3. What happens if you change coin.tscn's color now? (Try it!)

4. Where is the score variable stored? Why did we choose that location?

5. What other things could you collect in a game besides coins?

## 🎓 What You Learned Today

Concepts:

- Scene instancing (reusable templates)

- Signal system (event-driven programming)

- String concatenation (combining text)

- Groups (organizing nodes)

- UI display (showing game state)

Skills:

- Creating separate scene files

- Connecting signals in code

- Managing game state (score variable)

- Debugging with print statements

## 🔮 Next Lesson Preview: Enemies & Basic AI

Coming in Lesson 4:

- Create patrolling enemies

- Learn about RayCast2D for edge detection

- Set up collision layers properly

- Add danger to your game!

- Player loses when touching enemy

Think about: What games have memorable enemies? What makes them fun or frustrating?

## 📋 Quick Code Reference

### Complete coin.gd

```gdscript
extends Area2D

signal coin_collected

func _ready():
	body_entered.connect(_on_body_entered)

func _on_body_entered(body):
	if body.name == "Player":
		coin_collected.emit()
		queue_free()
```

### Complete level_1.gd (Score System)

```gdscript
```

```
extends Node2D

var score = 0

func _ready():
    var coins = get_tree().get_nodes_in_group("coins")
    for coin in coins:
        coin.coin_collected.connect(_on_coin_collected)

func _on_coin_collected():
    score += 1
    $ScoreLabel.text = "Score: " + str(score)
    print("Score: " + str(score))
```

Great work! You now have a proper collectible system! 🎉

Save your project and test it thoroughly before next lesson.