Side Quest 1: Teleportation System 🌀

Duration: 30 minutes
Godot Version: 4.5
Prerequisites: Completed Lessons 1 & 2 (Movement + Jumping)

🎯 What We're Building

A **reusable teleporter system** where:

- Player touches a **TeleportBox** (Entry Point)
- Screen **fades to black**
- Player **instantly moves** to destination
- Screen **fades back in**
- Uses a **standalone script** you can reuse anywhere!

Result: Touch glowing box → fade out → appear at new location → fade in!

📚 Key Vocabulary

Term	Meaning
Modulate	Controls color and transparency (alpha channel)
Tween	Smooth animation between values (like fade effects)
Exported Variable	Variable you can change in Inspector per instance
Standalone Script	Self-contained code that works independently

---

📐 Project Structure

New Files:

Year9_Platformer/

├── teleporter.tscn ← NEW! Reusable teleport box scene

├── teleporter.gd ← NEW! Standalone teleport script

└── level_1.tscn ← Add teleporters here

PART 1: Create Teleporter Scene (7 min)

Step 1: Create New Scene

1. **Scene** menu → **New Scene**
2. Select **Other Node**
3. Search: Area2D
4. Click **Create**
5. **Rename** root node: Teleporter

Why Area2D? Detects when player enters, but doesn't block movement!

Step 2: Add Collision Shape

1. Select **Teleporter** node
2. Click **+** (Add Child Node)
3. Add: CollisionShape2D
4. In **Inspector** → **Shape**: **New RectangleShape2D**
5. Set **Size**: 64 x 64

Step 3: Add Visual Appearance

1. Select **Teleporter** node
2. Add child: ColorRect
3. **Inspector** settings:
    - **Size**: 64 x 64
    - **Position**: -32, -32 (centers it)
    - **Color**: Cyan/Aqua blue (0, 255, 255) or purple (128, 0, 255)

Visual Tip: Choose a color that stands out from your platforms!

Step 4: Add Destination Marker

1. Select **Teleporter** node

2. Add child: Marker2D
3. Rename it: Destination

What's Marker2D? An invisible point that marks where the player will appear!

Step 5: Save the Scene

Scene → **Save Scene** → Name: teleporter.tscn

✅ **Checkpoint:** You should see teleporter.tscn in FileSystem panel!

PART 2: Write Standalone Teleporter Script (10 min)

Step 6: Attach Script

1. Select **Teleporter** root node
2. Click **script icon** (📄)
3. Path: res://teleporter.gd
4. Click **Create**

Step 7: Write the Teleporter Script

DELETE ALL template code and replace with:

gdscript

```
extends Area2D


# === CONFIGURATION ===
# EXPORTED: You can change this in the Inspector for each teleporter!
# This is where the player will teleport to
@export var destination_position: Vector2 = Vector2(500, 300)


# How fast the fade animation plays (seconds)
@export var fade_duration: float = 0.5
```

```
# Is this teleporter currently active? (prevents double-teleport)
var is_active: bool = true

# === INITIALIZATION ===
func _ready():
    # Connect the collision detection signal
    # When a body enters this Area2D, call our function
    body_entered.connect(_on_body_entered)

    print("Teleporter ready at: ", global_position)
    print("Will teleport player to: ", destination_position)

# === COLLISION DETECTION ===
# This runs when ANY physics body touches the teleporter
func _on_body_entered(body):
    # Check if it's the player AND teleporter is active
    if "Player" in body.name and is_active:
        print("Player entered teleporter!")

        # Prevent multiple teleports while fading
        is_active = false

        # Start the teleportation sequence
        _teleport_player(body)

# === TELEPORTATION SEQUENCE ===
# Handles the fade out → move → fade in effect
func _teleport_player(player):
```

```
print("Starting teleportation sequence...")

# --- STEP 1: FADE OUT ---
# Create a smooth animation (Tween) to fade player to invisible
var fade_out = create_tween()
# Animate the player's modulate.a (alpha/transparency) from 1.0 to 0.0
fade_out.tween_property(player, "modulate:a", 0.0, fade_duration)

# Wait for fade out to complete
await fade_out.finished
print("Fade out complete!")

# --- STEP 2: TELEPORT (INSTANT MOVE) ---
# Move player to the destination position
player.global_position = destination_position
print("Player teleported to: ", destination_position)

# --- STEP 3: FADE IN ---
# Create another Tween to fade player back to visible
var fade_in = create_tween()
# Animate from invisible (0.0) back to fully visible (1.0)
fade_in.tween_property(player, "modulate:a", 1.0, fade_duration)

# Wait for fade in to complete
await fade_in.finished
print("Fade in complete! Teleportation finished.")

# Re-enable teleporter after a short delay (prevents accidental re-teleport)
```

```
        await get_tree().create_timer(0.5).timeout
        is_active = true
        print("Teleporter reactivated!")
```

Save: Ctrl+S / Cmd+S

🧠 *Understanding the Code*

Section | **What It Does**

----------|----------

`@export var destination_position` | Lets you set teleport target in Inspector per instance

`is_active` | Prevents player from teleporting multiple times during fade

`create_tween()` | Creates smooth animation for fading

`modulate:a` | Controls transparency: 1.0 = visible, 0.0 = invisible

`await fade_out.finished` | Pauses code until animation completes

`global_position` | Moves player to exact world coordinates

`create_timer(0.5).timeout` | Waits 0.5 seconds before reactivating

Think of it like:

1. 🌫️ Fade player to invisible
2. ⚡ Instantly move them (they can't see it happening!)
3. ✨ Fade back to visible at new location

PART 3: Add Teleporters to Your Level (8 min)

Step 8: Open Level Scene

1. Click **level_1** tab at top

2. Make sure you can see your platform and player

Step 9: Instance First Teleporter (Entry Point)

1. **Drag** `teleporter.tscn` from FileSystem into viewport

2. Position it on your platform (near the start)

3. **Rename** this instance: `Teleporter_A`

Step 10: Set Destination Position

1. Select **Teleporter_A**

2. Look at **Inspector** panel

3. Find **Destination Position** property

4. Decide where you want player to appear

5. Set coordinates (example: `800, 200`)

How to find good coordinates:

- Look at your level in the viewport

- Check the position values as you move your mouse

- OR place the **Destination Marker** visually first!

Step 11: Add Visual Destination Marker (Optional)

This helps you see where players will land:

1. In **level_1**, add child to root: `Marker2D`

2. Rename: `TeleportDestination_A`

3. **Position** it where you want players to appear

4. Copy its **Position** values (Inspector → Transform → Position)

5. Select **Teleporter_A** → Paste those values into **Destination Position**

Visual Check: Place a coin or enemy near the destination to test placement!

Step 12: Test Single Teleporter

Save: Ctrl+Shift+S (save all)

Press F5:

- Walk to teleporter

- Player should fade out

- Player appears at destination coordinates

- Player fades back in

- Check Output panel for debug messages

Step 13: Add More Teleporters (Optional)

Want multiple teleport points?

1. **Duplicate** teleporter: Select Teleporter_A → Ctrl+D
2. Rename: `Teleporter_B`
3. Move to new position
4. Change **Destination Position** in Inspector
5. Repeat!

Each teleporter is independent! Change destination per instance.

✅ Success Checklist

```
```

☐ teleporter.tscn created and saved

☐ Teleporter has Area2D, CollisionShape2D, ColorRect

☐ teleporter.gd script attached (📄 icon visible)

☐ Script has @export var destination_position

☐ At least 1 teleporter placed in level_1

☐ Destination Position set in Inspector

☐ Player fades out when touching teleporter

☐ Player moves to destination

☐ Player fades back in

☐ Debug messages appear in Output panel

☐ Teleporter reactivates after use

☐ No errors in Output

🎮 Controls Reference

Key	Action
← →	Move player
Spacebar	Jump
F5	Run game
F8	Stop game
Ctrl+S	Save current file
Ctrl+Shift+S	Save all files

🐛 Troubleshooting

Problem	Cause	Solution
Player doesn't teleport	Script not attached	Check 📄 icon on Teleporter node
Teleports to (0,0)	Default destination	Set Destination Position in Inspector
No fade effect	Tween not working	Check Godot version is 4.5+
Player stays invisible	Fade in didn't complete	Check for errors in Output panel
Teleports immediately back	is_active not working	Check timer at end of script
"Player" not detected	Name mismatch	Check player node name has "Player" in it
Teleports multiple times	No cooldown	Ensure is_active = false line exists

---

🏆 Challenge Extensions

Challenge 1: Change Teleporter Color When Used

Make it glow or change color after teleporting:

gdscript

 # Add to _teleport_player function, after player.global_position line:

 # Change teleporter color to show it was used

$ColorRect.color = Color.GRAY

Challenge 2: Two-Way Teleportation

Create a return teleporter at the destination:

1. Place second teleporter at destination point
2. Set its **Destination Position** back to first teleporter's location
3. Now you can go back and forth!

Challenge 3: Teleporter Sound Effect

Add audio feedback:

1. Add child to Teleporter: AudioStreamPlayer
2. Find a "whoosh" or "warp" sound effect
3. Add to script before fade out:

gdscript

 # Add after is_active = false line:

if has_node("AudioStreamPlayer"):

    $AudioStreamPlayer.play()

Challenge 4: Particle Effect

Add a visual effect at teleport points:

1. Add child to Teleporter: GPUParticles2D

2. Configure particles (Amount: 20, Lifetime: 1.0)
3. Add to script:

gdscript

```
# Add before fade_out line:
if has_node("GPUParticles2D"):
    $GPUParticles2D.emitting = true
```

Challenge 5: Adjustable Fade Speed

Make different teleporters fade at different speeds:

- Already built in! Just change **Fade Duration** in Inspector
- Try 0.3 for fast, 1.0 for slow, dramatic teleports

Challenge 6: Teleporter Requires Coin/Key

Lock teleporter until player collects something:

gdscript

```
# Add at top with other exports:
@export var requires_coins: int = 5


# Change in _on_body_entered:
func _on_body_entered(body):
    if "Player" in body.name and is_active:
        # Check if player has enough coins (assuming you have score system)
        var level = get_parent()
        if level.score >= requires_coins:
            print("Player has enough coins! Teleporting...")
            is_active = false
            _teleport_player(body)
        else:
```

```
print("Need ", requires_coins, " coins to use teleporter!")
```

📖 What You Learned

Concepts:

- Exported variables (configurable per instance)
- Tweens for smooth animations
- Async/await for sequenced actions
- Modulate alpha for transparency
- Standalone, reusable scripts

Skills:

- Creating configurable scene templates
- Implementing visual effects
- Using Inspector to customize instances
- Sequencing multiple actions
- Debugging with print statements

Game Design:

- Teleportation as a level design tool
- Visual feedback for player actions
- Preventing exploit behaviors (cooldowns)

🤔 Reflection Questions

Answer in your exercise book:

1. Why do we need is_active to prevent multiple teleports?
2. What does @export do that regular var doesn't?
3. How could you use teleporters to create puzzle levels?
4. What's the difference between position and global_position?
5. Name a game that uses teleportation well. How does it feel?

📝 Complete Code Reference

teleporter.gd (Full Script)

gdscript

```
extends Area2D

# === CONFIGURATION ===
# EXPORTED: You can change this in the Inspector for each teleporter!
# This is where the player will teleport to
@export var destination_position: Vector2 = Vector2(500, 300)

# How fast the fade animation plays (seconds)
@export var fade_duration: float = 0.5

# Is this teleporter currently active? (prevents double-teleport)
var is_active: bool = true

# === INITIALIZATION ===
func _ready():
    # Connect the collision detection signal
    # When a body enters this Area2D, call our function
    body_entered.connect(_on_body_entered)

    print("Teleporter ready at: ", global_position)
    print("Will teleport player to: ", destination_position)

# === COLLISION DETECTION ===
# This runs when ANY physics body touches the teleporter
func _on_body_entered(body):
```

```
        # Check if it's the player AND teleporter is active
        if "Player" in body.name and is_active:
                print("Player entered teleporter!")

                # Prevent multiple teleports while fading
                is_active = false

                # Start the teleportation sequence
                _teleport_player(body)

# === TELEPORTATION SEQUENCE ===
# Handles the fade out → move → fade in effect
func _teleport_player(player):
        print("Starting teleportation sequence...")

        # --- STEP 1: FADE OUT ---
        # Create a smooth animation (Tween) to fade player to invisible
        var fade_out = create_tween()
        # Animate the player's modulate.a (alpha/transparency) from 1.0 to 0.0
        fade_out.tween_property(player, "modulate:a", 0.0, fade_duration)

        # Wait for fade out to complete
        await fade_out.finished
        print("Fade out complete!")

        # --- STEP 2: TELEPORT (INSTANT MOVE) ---
        # Move player to the destination position
        player.global_position = destination_position
```

```
    print("Player teleported to: ", destination_position)

    # --- STEP 3: FADE IN ---
    # Create another Tween to fade player back to visible
    var fade_in = create_tween()
    # Animate from invisible (0.0) back to fully visible (1.0)
    fade_in.tween_property(player, "modulate:a", 1.0, fade_duration)

    # Wait for fade in to complete
    await fade_in.finished
    print("Fade in complete! Teleportation finished.")

    # Re-enable teleporter after a short delay (prevents accidental re-teleport)
    await get_tree().create_timer(0.5).timeout
    is_active = true
    print("Teleporter reactivated!")
```

🎓 **Teacher Notes**

Common Student Mistakes:

1. Forgetting to set Destination Position → player teleports to (0,0)
2. Not saving scene after editing script
3. Confusion between position vs global_position
4. Placing destination inside solid platforms (player gets stuck)

Differentiation:

- **Support:** Provide pre-set destination coordinates
- **Extension:** Add multiple linked teleporters creating a network
- **Advanced:** Create teleporter unlock system requiring keys

Time Management:

- Scene creation: 7 minutes
- Script writing: 10 minutes
- Implementation/testing: 8 minutes
- Challenge activities: 5 minutes

Assessment Opportunities:

- Understanding of exported variables
- Proper use of await/async
- Debugging skills (reading Output panel)
- Creative level design with teleporters

🎯 Next Steps

Ideas for Expanding:

- Create **danger zones** that teleport you back to start
- Make **puzzle levels** requiring specific teleport sequences
- Add **visual indicators** showing teleport destination
- Create **teleport mazes** for advanced gameplay

Congratulations! You've created a professional teleportation system! 🌟

This standalone script can be used in ANY of your levels - just drag in teleporter.tscn, set the destination, and you're done!

Save your project and experiment with creative teleporter placements!

Retry