Game Art

Lesson 1 Quick Reference Guide

From Image to Game: Creating Your First Game Character

Duration: 40 minutes
Software: Adobe Photoshop CC 2024 + Godot 4.5
Prerequisites: None - Complete beginners welcome!

🎮 What We're Building Today

Photoshop (20 min):

- Find character images online
- Remove backgrounds manually
- Create a simple sprite sheet
- Export as transparent PNG

Godot (20 min):

- Import your sprite into your platformer
- Replace ColorRect with YOUR art
- Make your character appear in the game!

Challenge:

- Flip character to face movement direction

📚 Key Vocabulary

- **Sprite:** A 2D image/character used in games
- **Transparent Background:** See-through areas (checkerboard pattern)
- **PNG:** Image format that supports transparency (perfect for games!)
- **Layer:** Like transparent sheets stacked on top of each other
- **Selection:** Highlighted area you can modify or delete
- **Magic Wand:** Tool that selects similar colors
- **Sprite Sheet:** Multiple character poses in one image file

---

🖼️ Part 1: Finding Your Character Images (5 min)

Your Mission: Find 3-5 Character Images

What to look for:

- Same character in different poses/angles
- Character facing: **right**, **left**, **front** (or back)
- Ideally with simple backgrounds (easier to remove!)
- Clear, visible character (not too small)

Best Free Sources:

Website	What You'll Find	Best For
spriters-resource.com	Game sprites (already transparent!)	Easy mode
cleanpng.com	PNG images, many transparent	Quick start
pngwing.com	Huge PNG library	Variety
pixabay.com	Free photos/illustrations	Original art
Google Images	Everything!	Use filter: Tools → Type → PNG

Search Terms to Try:

- "character sprite facing right"
- "pixel art character walking"
- "game character front view"
- "platformer hero sprite"
- "[favorite game] character sprite"

Download 3 Images Minimum:

✅ Character facing RIGHT

✅ Character facing LEFT (or you can flip the right one!)

✅ Character facing FRONT

```

### **Save Location:**

Create this folder first:
```
Desktop/
└── GameArt_Year9/
    ├── reference_images/   ← Save downloads here
    ├── working_files/   ← Photoshop files (.psd)
    └── exported_sprites/   ← Final PNGs for Godot
```

** 💡 Pro Tip:** If you find sprites that are ALREADY transparent (checkerboard background in preview), you're in luck! You can skip the background removal steps!

---

## 🎨 **Part 2: Photoshop - Create Your Sprite Sheet (15 min)**

### **Step 1: Open Photoshop & Create New Document (2 min)**

**Launch Photoshop CC 2024**

**File → New** (or `Ctrl+N` / `Cmd+N`)

**Settings:**
```

| Setting | Value | Why? |

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

| **Width** | **192 pixels** | Fits 3 sprites of 64px each |

| **Height** | **64 pixels** | Standard sprite height |

| **Resolution** | **72 pixels/inch** | Screen graphics standard |

| **Color Mode** | **RGB Color** | Games use RGB |

| **Background Contents** | **Transparent** | ⚠️ CRITICAL! No white background! |

Click Create

You should see: Gray and white checkerboard pattern = Transparency! ✓

Step 2: Import Your First Character Image (2 min)

Method 1: Drag & Drop (Easiest)

1. Open File Explorer/Finder

2. Find your first downloaded image

3. **Drag it onto the Photoshop canvas**

4. Press **Enter** to place it

Method 2: File Menu

1. **File → Place Embedded**

2. Navigate to `reference_images/` folder

3. Select image → **Place**

4. Press **Enter**

You should now see:

- Your image on the canvas

- New layer in **Layers panel** (right side)

Step 3: Remove Background Manually (8 min)

Even if your image has a background, we'll remove it!

Resize Your Image First (If Needed)

Is your image too big or too small?

1. Press `Ctrl+T` / `Cmd+T` (Free Transform)
2. **Hold Shift** + drag corner to resize
3. Make it about **50-60 pixels tall**
4. Press **Enter**

Method A: Magic Wand Tool (Best for solid backgrounds)

Perfect for: Images with white/solid color backgrounds

1. **Select the layer** with your character
2. Press **W** for **Magic Wand Tool**

Top toolbar settings:

```
Tolerance: 32    ← How similar colors it selects
Contiguous: ✓    ← Only connected areas
Sample All Layers: ☐ ← Off
```

3. **Click the background** (the part you want to DELETE)
4. You'll see "marching ants" selection
5. Press **Delete** key

Did it work? You should see checkerboard where background was! ✓

If some background remains:
- Click other background areas while holding **Shift** (adds to selection)
- Then press **Delete**

If it deleted too much:
- Press `Ctrl+Z` / `Cmd+Z` to undo
- Lower **Tolerance** to 20
- Try again

Method B: Eraser Tool (For detailed cleanup)

Use this after Magic Wand to clean up edges!

1. Press **E** for **Eraser Tool**

2. **Brush settings (top toolbar):**

  - Size: 10-20 pixels (use `[` `]` keys to change size)

  - Hardness: 100% (for sharp edges)

3. **Zoom in:** `Ctrl+Plus` / `Cmd+Plus`

4. **Erase remaining background pixels** around character

5. **Zoom out:** `Ctrl+Minus` / `Cmd+Minus`

** 💡 Pro Tip:**

- Hold **Spacebar** and drag to pan around the image

- Make eraser smaller for detail work near edges

- Press `X` to switch between eraser and brush (undo mistakes)

Method C: Quick Selection Tool (Smart selection)

Good for: Complex backgrounds

1. Press **W** (click and hold) → Select **Quick Selection Tool**

2. **Click and drag** over your CHARACTER (not background!)

3. Tool automatically finds edges

4. If it selects too much: Hold **Alt/Option** and click to subtract

5. **Select → Inverse** (or `Ctrl+Shift+I` / `Cmd+Shift+I`)

6. Now background is selected instead!

7. Press **Delete**

Checking Your Work

Is the background REALLY gone?

1. Click the **New Layer** icon (📄) at bottom of Layers panel
2. Drag this new layer BELOW your character layer
3. Fill it with a bright color:
   - **Edit → Fill → Color**
   - Choose bright green or pink
   - Click **OK**

Can you see green/pink ONLY where background should be? ✓ Perfect!

Still see white/gray around edges? Use Eraser tool to clean up!

When finished, delete the colored layer (just for testing)

Step 4: Create Sprite Sheet with Multiple Poses (5 min)

Now let's add your other character images!

Import Second Image

1. **File → Place Embedded**
2. Select your **facing left** image

3. Press **Enter**

4. Press **V** for Move Tool

5. **Drag it to the RIGHT** of your first character

6. Leave about **5-10 pixels gap** between them

Remove Background from Second Image

1. Repeat Magic Wand + Eraser process

2. Make sure both sprites are same height!

To match heights:

1. Press `Ctrl+T` / `Cmd+T` (Transform)

2. Look at **H:** (height) in top toolbar

3. Make both images the same height (e.g., both 60px)

4. Press **Enter**

Add Third Image (Optional)

If you have time, add a third pose!

Your canvas should look like:

```
[Character    [Character    [Character
 facing       facing       facing
 right]       left]        front]
```

Step 5: Export Your Sprite Sheet (3 min)

Time to save for Godot!

Hide or Delete Reference Layers

- If you have reference/guide layers, hide them (👁️ icon) or delete them

Export as PNG

File → Export → Export As...

Critical Settings:

Setting	Value
Format	**PNG**
Transparency	**✓ Checked**
Smaller File	✓ Checked
Convert to sRGB	✓ Checked

Save As:

```
Desktop/GameArt_Year9/exported_sprites/player_sprite.png
```

Click Export

Save Your Working File

File → Save As...

- Format: **Photoshop (.psd)**

- Location: `Desktop/GameArt_Year9/working_files/player_sprite.psd`

Why both?

- **PNG:** For Godot (final, compressed)

- **PSD:** Keep editing layers later!

✅ **Photoshop Checklist**

```
☐ 3 character images downloaded

☐ Photoshop document created (192×64, transparent)

☐ Backgrounds removed from all images

☐ Sprites arranged in a row with spacing

☐ PNG exported to exported_sprites folder

☐ PSD saved to working_files folder

☐ PNG preview shows checkerboard (transparent!)
```

🎮 **Part 3: Import to Godot (15 min)**

Now let's put YOUR art into YOUR game!

Step 1: Copy Sprite to Godot Project (2 min)

Create folder structure in your Godot project:

```
Desktop/Year9_Platformer/
  └── assets/
    └── sprites/
      └── player_sprite.png ← Copy your PNG here!
```

How to:

1. Open your `Year9_Platformer` folder in File Explorer/Finder
2. Create folders: `assets/` then `sprites/` inside it
3. **Copy** `player_sprite.png` from `GameArt_Year9/exported_sprites/`
4. **Paste** into `Year9_Platformer/assets/sprites/`

Step 2: Open Your Godot Project (1 min)

1. Launch **Godot 4.5**
2. Open **Year9_Platformer**
3. Open **level_1.tscn** scene

Step 3: Configure Import Settings (3 min)

CRITICAL for pixel art to look sharp!

1. Look at **FileSystem** panel (bottom-left)
2. Navigate to: `res://assets/sprites/`
3. **Click on** `player_sprite.png`

Import Panel (top-right):

Change these settings:

Setting	Change To	Why?
Compress → Mode	**Lossless**	Keeps pixel art crisp
Filter	**Nearest**	No blur on pixels!
Repeat	**Disabled**	Prevents edge issues

Click Reimport button at bottom

Preview your sprite - should be crisp and clear! ✓

Step 4: Add Sprite2D to Player (4 min)

Time to replace that ColorRect with your art!

Add Sprite2D Node

1. **Select Player node** in Scene tree (left panel)
2. Click **+** (Add Child Node)
3. Search: **Sprite2D**
4. Click **Create**

Load Your Sprite

1. **Select the Sprite2D node** you just created
2. **Inspector panel** (right side) → Find **Texture** property
3. **Drag** `player_sprite.png` from FileSystem panel → Drop on **Texture** field

You should see your sprite on the character! 🎉

Step 5: Position and Scale Sprite (2 min)

Your sprite might be in the wrong spot or wrong size!

If You Have Multiple Sprites in One Image:

Inspector → Animation:

```
Hframes: 3    ← If you have 3 sprites in a row
Vframes: 1    ← Only 1 row
Frame: 0      ← Shows first sprite (facing right)
```

Adjust Size (If Needed):

Inspector → Transform → Scale:

- Try: 1, 1 (normal size)
- Too big? Try: 0.5, 0.5 (half size)
- Too small? Try: 2, 2 (double size)

Center the Sprite:

1. With Sprite2D selected, look at viewport
2. Make sure sprite is **centered on the Player position**
3. If not, adjust **Transform → Position** in Inspector

Step 6: Update Collision Shape (2 min)

Your collision box needs to match your new sprite!

1. **Select CollisionShape2D** (child of Player)
2. Look at the **orange rectangle** in the viewport
3. **Drag the orange handles** to match your sprite size
4. Make collision **slightly smaller** than sprite (better gameplay!)

💡 **Good collision size:**

- Width: About 80% of sprite width
- Height: Full sprite height
- Should fit inside the character silhouette

Step 7: Delete Old ColorRect (1 min)

You don't need the colored square anymore!

1. Find **ColorRect** node (child of Player)
2. **Right-click → Delete**
3. Click **OK**

Step 8: Test Your Game! (2 min)

Press F5 to run!

You Should See:

✅ YOUR custom character on screen (not a colored box!)
✅ Character moves left/right with arrow keys
✅ Character falls and lands on platform
✅ Character can reach the goal box

If Something's Wrong:

Problem	Solution
Can't see sprite	Check Sprite2D → Texture is filled
Sprite is HUGE	Lower Scale in Transform (try 0.5)
Sprite looks blurry	Check Import settings: Filter = Nearest
Character falls through floor	Check CollisionShape2D has a shape
Sprite is upside down	Inspector → Transform → Rotation: 0°
Multiple sprites showing	Set correct Hframes and Frame number

---

🏆 **Challenge: Flip Character to Face Movement Direction**

Make your character face the direction they're walking!

Open player.gd Script

1. Select **Player** node
2. Click **script icon** (📄)

Add This Code

Find the _physics_process function. Add this code **before** move_and_slide():

gdscript

```
func _physics_process(delta):
    # --- APPLY GRAVITY ---
    if not is_on_floor():
        velocity += get_gravity() * delta
```

```
    # --- JUMP CODE (DISABLED FOR LESSON 1) ---
    # if Input.is_action_just_pressed("ui_accept") and is_on_floor():
    #     velocity.y = JUMP_VELOCITY

    # --- GET PLAYER INPUT ---
    var direction = Input.get_axis("ui_left", "ui_right")

    # --- MOVE THE PLAYER ---
    if direction:
        velocity.x = direction * SPEED
    else:
        velocity.x = move_toward(velocity.x, 0, SPEED)

    # === CHALLENGE: FLIP SPRITE === (ADD THIS!)
    if direction > 0:
        $Sprite2D.flip_h = false  # Facing right (normal)
    elif direction < 0:
        $Sprite2D.flip_h = true   # Facing left (flipped)

    # --- EXECUTE THE MOVEMENT ---
    move_and_slide()
```

Understanding the Code:

gdscript

```
if direction > 0:
```

Meaning: If moving right (positive direction)...

gdscript

```
$Sprite2D.flip_h = false
```

Meaning: Don't flip (show normal sprite)

gdscript

elif direction < 0:

Meaning: Else if moving left (negative direction)...

gdscript

$Sprite2D.flip_h = true

Meaning: Flip horizontally (mirror the sprite!)

Test the Challenge:

Press F5

✅ Press **Right Arrow** → Character faces right
✅ Press **Left Arrow** → Character faces left (flipped!)
✅ Character smoothly changes direction

Success! Your character now faces where they're going! 🎉

Challenge Extension: Use Different Sprite for Each Direction

If you have separate left/right sprites in your sheet:

gdscript

```
# In _physics_process, before move_and_slide():
if direction > 0:
    $Sprite2D.frame = 0     # First sprite (facing right)
    $Sprite2D.flip_h = false
elif direction < 0:
    $Sprite2D.frame = 1     # Second sprite (facing left)
    $Sprite2D.flip_h = false  # Don't flip if you have a proper left sprite!
```

This uses different frames instead of flipping!

📖 **Complete Code Reference**

player.gd with sprite flipping:

gdscript

```
extends CharacterBody2D

# === MOVEMENT CONSTANTS ===
const SPEED = 300.0
const JUMP_VELOCITY = -400.0

# === MAIN PHYSICS FUNCTION ===
func _physics_process(delta):
	# --- APPLY GRAVITY ---
	if not is_on_floor():
		velocity += get_gravity() * delta

	# --- JUMP CODE (DISABLED FOR LESSON 1) ---
	# if Input.is_action_just_pressed("ui_accept") and is_on_floor():
	#     velocity.y = JUMP_VELOCITY

	# --- GET PLAYER INPUT ---
	var direction = Input.get_axis("ui_left", "ui_right")

	# --- MOVE THE PLAYER ---
	if direction:
		velocity.x = direction * SPEED
	else:
		velocity.x = move_toward(velocity.x, 0, SPEED)

	# --- SPRITE FLIPPING ---
```

```
        if direction > 0:
                $Sprite2D.flip_h = false
        elif direction < 0:
                $Sprite2D.flip_h = true

        # --- EXECUTE THE MOVEMENT ---
        move_and_slide()

# === GOAL DETECTION SETUP ===
func _ready():
        var goal_box = get_parent().get_node("GoalBox")
        goal_box.body_entered.connect(_on_goal_box_body_entered)

# === GOAL COLLISION RESPONSE ===
func _on_goal_box_body_entered(body):
        if body == self:
                print("Level Complete! Great job!")
                get_tree().change_scene_to_file("res://level_2.tscn")
```

🎨 **Photoshop Tools Quick Reference**

Tool	Shortcut	Use For
Move Tool	V	Moving layers/sprites
Magic Wand	W	Selecting similar colors

| **Quick Selection** | W (hold, select) | Smart selection |

| **Eraser** | E | Removing pixels |

| **Zoom In** | Ctrl/Cmd + Plus | See details |

| **Zoom Out** | Ctrl/Cmd + Minus | See full image |

| **Pan** | Spacebar + Drag | Move around canvas |

| **Transform** | Ctrl/Cmd + T | Resize/rotate |

| **Undo** | Ctrl/Cmd + Z | Fix mistakes |

🎮 ***Godot Controls***

| Key | Action |

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

| **F5** | Run game |

| **F8** | Stop game |

| **Ctrl+S / Cmd+S** | Save scene |

| **← →** | Move character |

✅ ***Final Success Checklist***

Photoshop:

```

☐ Character images found and downloaded

☐ Backgrounds removed manually

☐ Sprites arranged in a sheet
```

☐ PNG exported with transparency

☐ PSD working file saved

```

**Godot:**

```

☐ PNG copied to assets/sprites/ folder

☐ Import settings configured (Filter: Nearest)

☐ Sprite2D added to Player

☐ Texture loaded and visible

☐ Collision shape matches sprite

☐ Old ColorRect deleted

☐ Game runs (F5) with your custom sprite

☐ Character appears correctly

```

**Challenge:**

```

☐ Code added to flip sprite

☐ Character faces right when moving right

☐ Character faces left when moving left

☐ Code saved (Ctrl+S)

🔧 Common Problems & Solutions

Problem	Cause	Solution
White background in Godot	Didn't remove background	Go back to Photoshop, delete background

Problem	Cause	Solution
Sprite looks blurry	Wrong import filter	Import settings: Filter = Nearest
Can't see sprite	Wrong layer order	Make sure Sprite2D is child of Player
Sprite too big/small	Wrong scale	Adjust Transform → Scale in Inspector
Character walks backwards	Wrong flip direction	Change < 0 to > 0 (or vice versa)
Error: "Invalid get index 'flip_h'"	Wrong node path	Check: is it $Sprite2D or different name?
Collision doesn't match	Wrong collision size	Resize CollisionShape2D handles

---

🌟 What You Learned Today

Photoshop Skills:

- Finding game assets online
- Creating new documents with transparency
- Removing backgrounds manually (Magic Wand, Eraser)
- Organizing sprites in a sheet
- Exporting PNGs for games

Godot Skills:

- Importing external assets
- Configuring import settings for pixel art
- Adding Sprite2D nodes
- Connecting sprites to game objects
- Basic GDScript for sprite manipulation

Game Development Concepts:

- Asset pipeline (creation → export → import)
- Sprite management

- Character representation
- Visual feedback (facing direction)

🔮 **Next Lesson Preview**

Lesson 2: Animated Sprites - Walking & Idle

We'll create:

- Walk cycle animation (3-4 frames)
- Idle pose animation
- Smooth transitions between animations
- AnimatedSprite2D in Godot
- Animation code

Think about: What games have smooth, satisfying character animations?

📝 **Homework (Optional)**

1. Find 5 more character poses (jumping, attacking, crouching)
2. Practice removing backgrounds on different images
3. Create enemy sprite sheet using same technique
4. Sketch your own original character design!

🎉 **Congratulations! You've created your first game character from scratch and put it in your game!**

End of Lesson 1 Quick Reference Guide