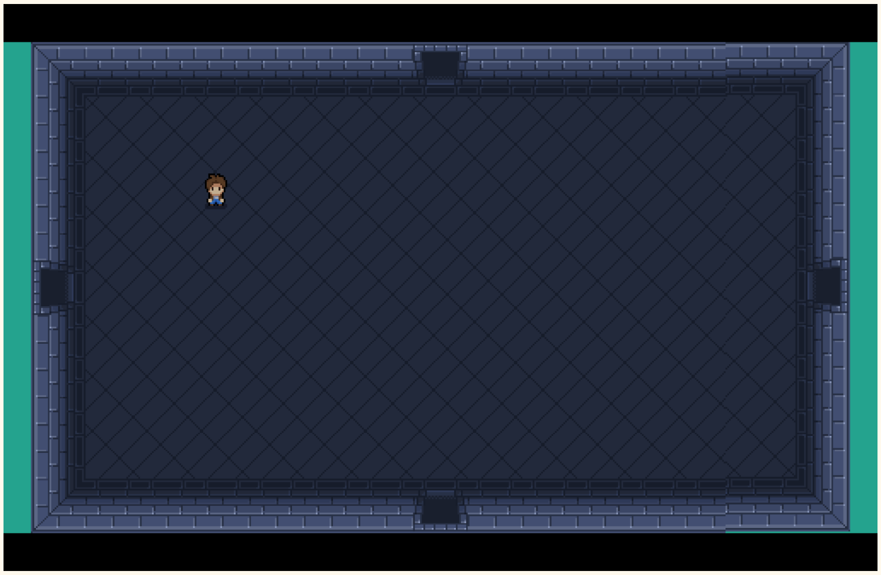
GODOT 4

Using maximized ‘borderless fullscreen’ windows (and scaling our game to fit)

By Felix



Introduction

Games in *borderless fullscreen* windows use the full screen, but maintain the ability for users to quickly switch between the game window and other programs without delay. This is *unlike regular fullscreen windows*, where switching incurs a delay.

Setting up *borderless fullscreen* is separate from making a 2D game scale its content to fit the window, but I will describe both.

This is useful if you have pixel art graphics or a high enough resolution that can scale to various sizes. Otherwise, if you scale low-res textures to higher-res, they will look pixelated.

Create *borderless fullscreen* window

This involves a small amount of code. We can use Godot 4's built-in *DisplayServer* singleton class (a class that can only have one instance). *DisplayServer* is pre-configured and globally available from GDScript.

NOTE: *DisplayServer* is new in Godot 4. In Godot 3, this functionality was accessible through the *OS* class. Many online guides still refer to *OS* for this use case.

1. Create a script for your main scene (which is probably a *Node* or *Node2D*)

2. Create a function maximize the screen

```
# Resize window to fit the screen.

func maximize_window(_unused_arg):
    # This is the primary screen as decided by the OS
    var screen_idx: int = DisplayServer.get_primary_screen()

    # Get screen size and set window size to it
    var screen_size = DisplayServer.screen_get_size(screen_idx)
    DisplayServer.window_set_size(screen_size)

    # Center window on screen (this is relevant for multi-screen setups)
    var screen_pos = DisplayServer.screen_get_position(screen_idx)
    DisplayServer.window_set_position(screen_pos)
```

3. **Create a _ready function and call maximize_window from it**

```
# Called when the node enters the scene tree for the first time.
func _ready():
    # Fix the Mac bug
    var os_name = OS.get_name()
    if os_name == "macOS":
        # Set window to borderless here, and move to foreground to make it appear
        DisplayServer.window_set_flag(DisplayServer.WINDOW_FLAG_BORDERLESS, true)
        DisplayServer.window_move_to_foreground()

    # Change window into a borderless fullscreen window initially
    maximize_window(get_viewport().size)
    # And everytime the viewport (window) changes
    get_viewport().size_changed.connect(maximize_window)
```

Scaling to window

2D games can scale to fit the window using the ‘Viewport’ Stretch Mode setting.

1. **Decide on your game’s native rendering resolution.**

   This is the resolution that you use to design your game within the Godot GUI.

2. ***Project Settings* -> *Display* -> *Window* -> *Size* -> enter your native render resolution in *Viewport Width* and *Viewport Height***

3. ***Project Settings* -> *Display* -> *Window* -> *Stretch* -> *Mode* -> select *viewport***

4. **(Optional) For scaling pixel art games without blur:**

   > ***Project Settings* -> *Rendering* -> *Textures* -> *Default Texture Filter* -> select *Nearest***