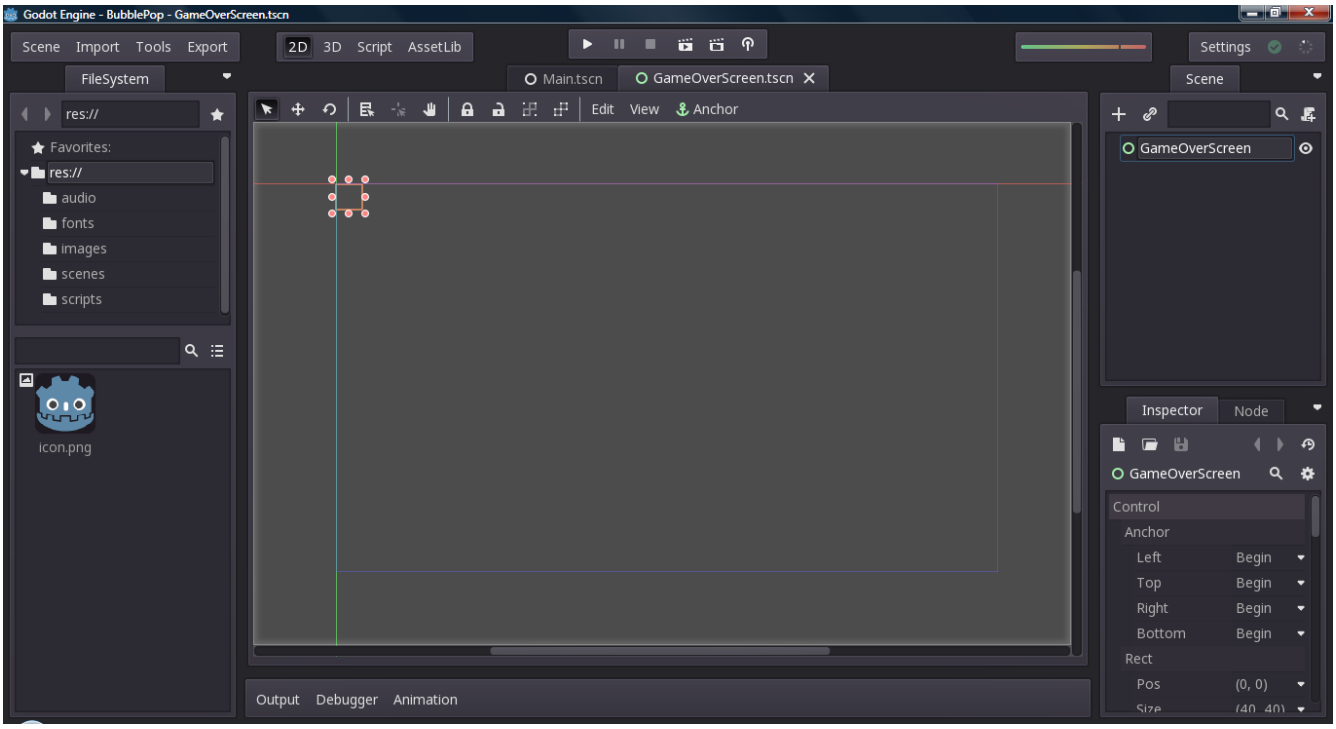
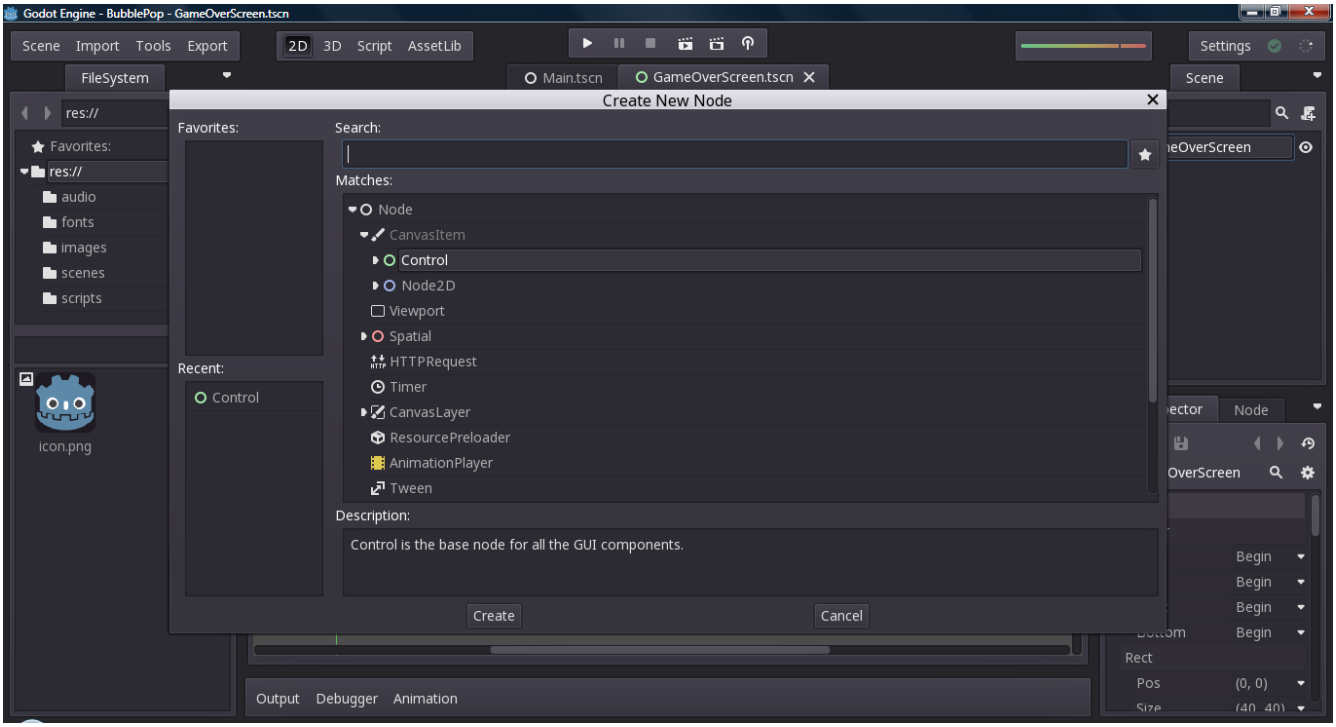
Godot 2D Game
Lesson 13: Game Over Screen

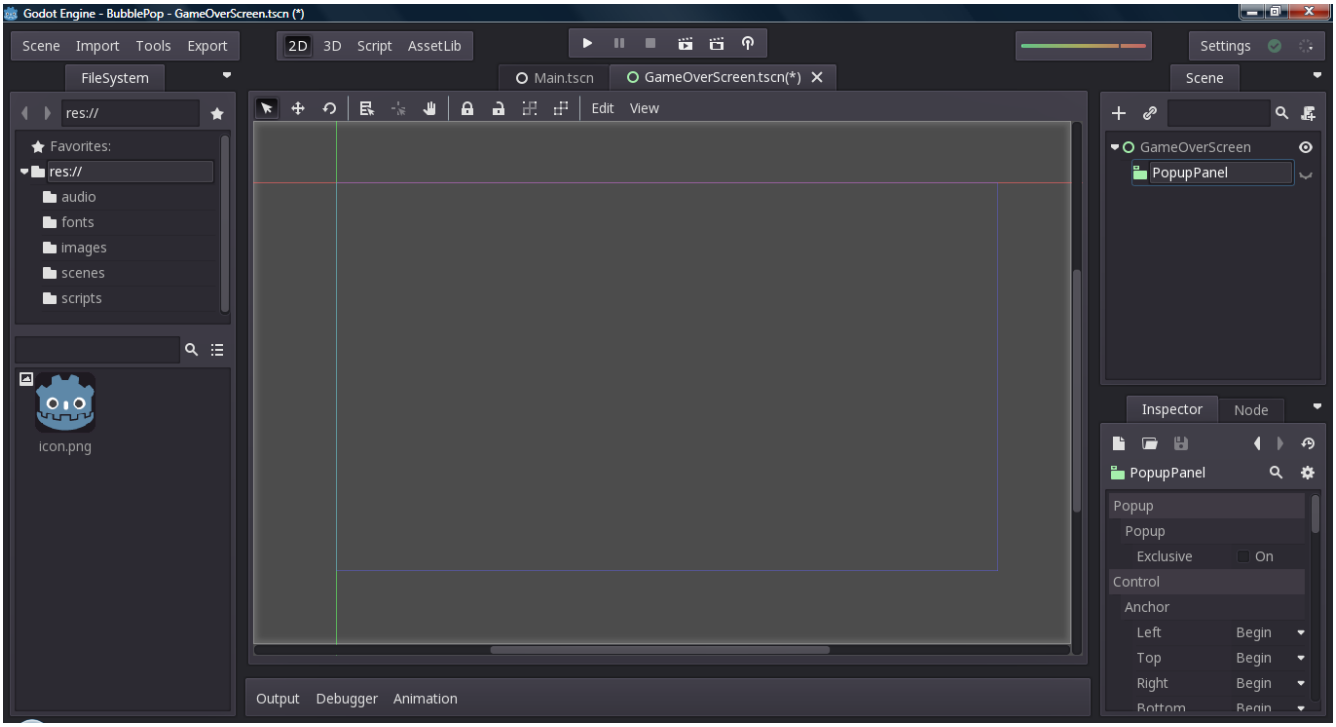
In this lesson, we will create a simple game over screen for our game. Let's start by creating a new scene with a Control node as the root. We will call the root node "GameOverScreen":



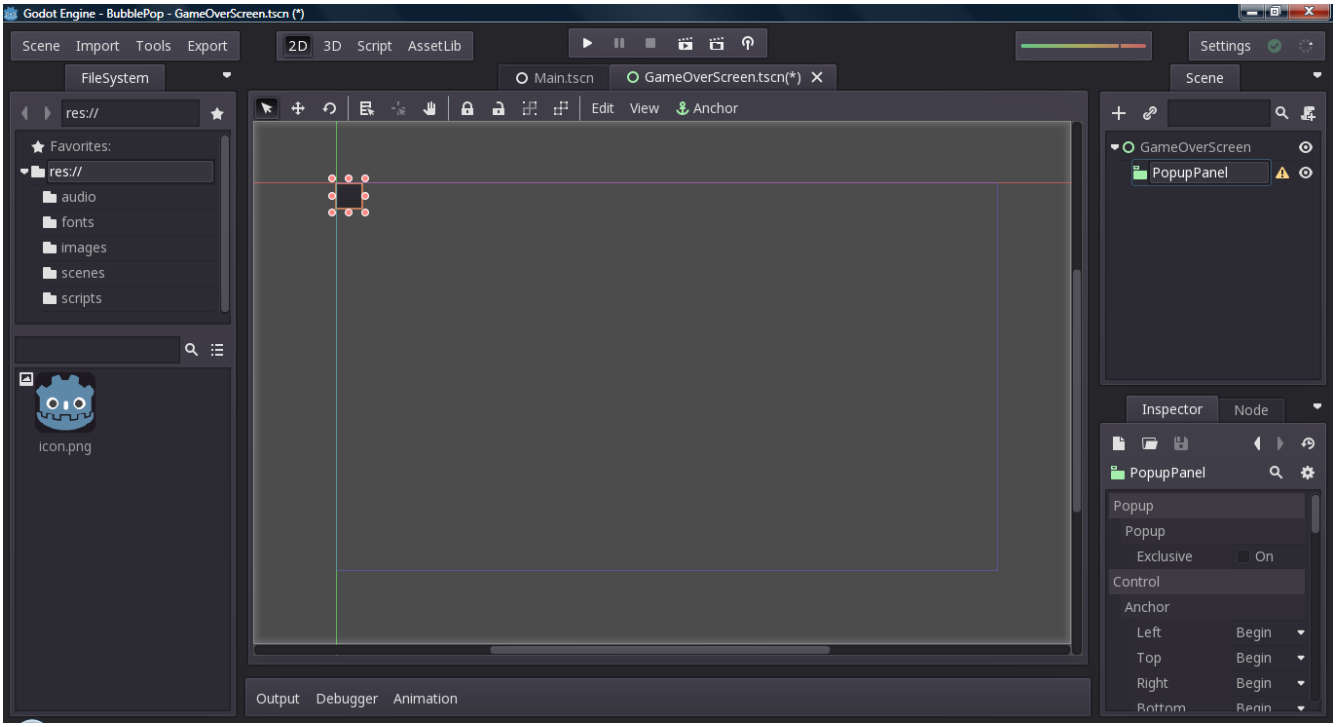
Save the new scene in your "scenes" folder. Then right-click your "GameOverScreen" node and choose "Add Child Node":



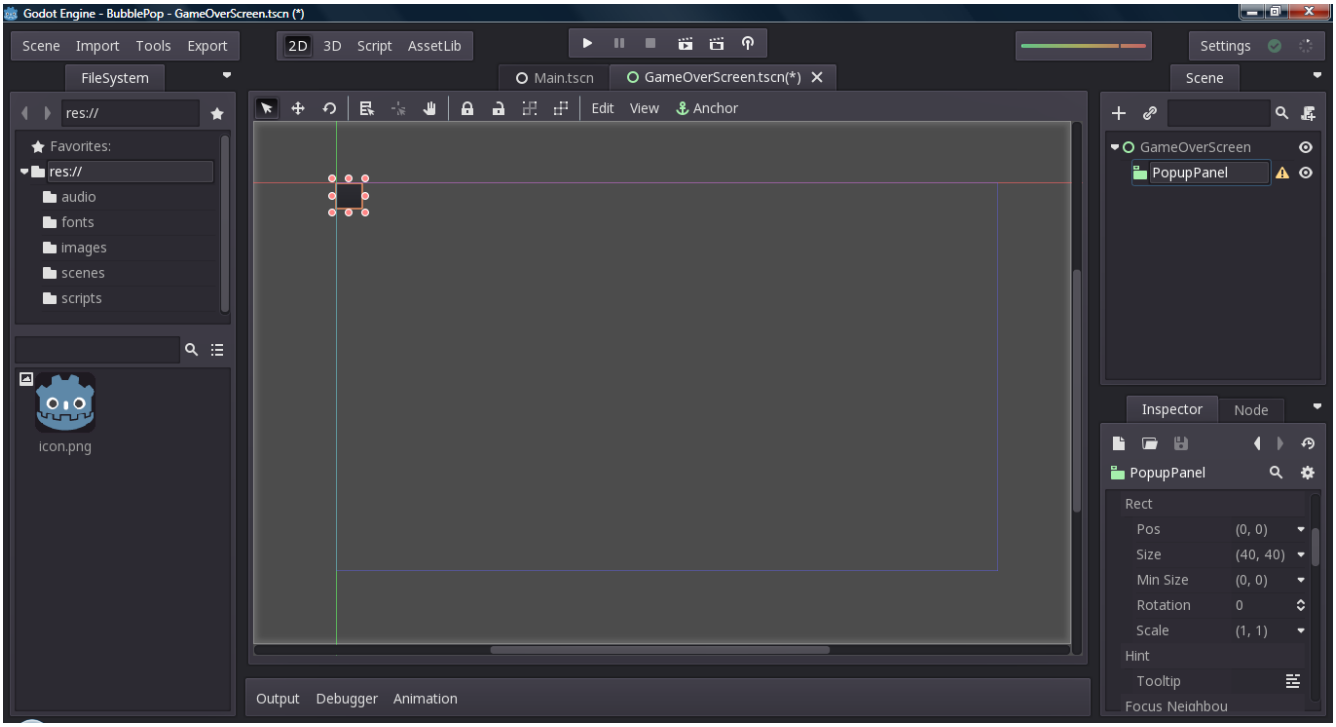
This time we are going to create a PopupPanel node:



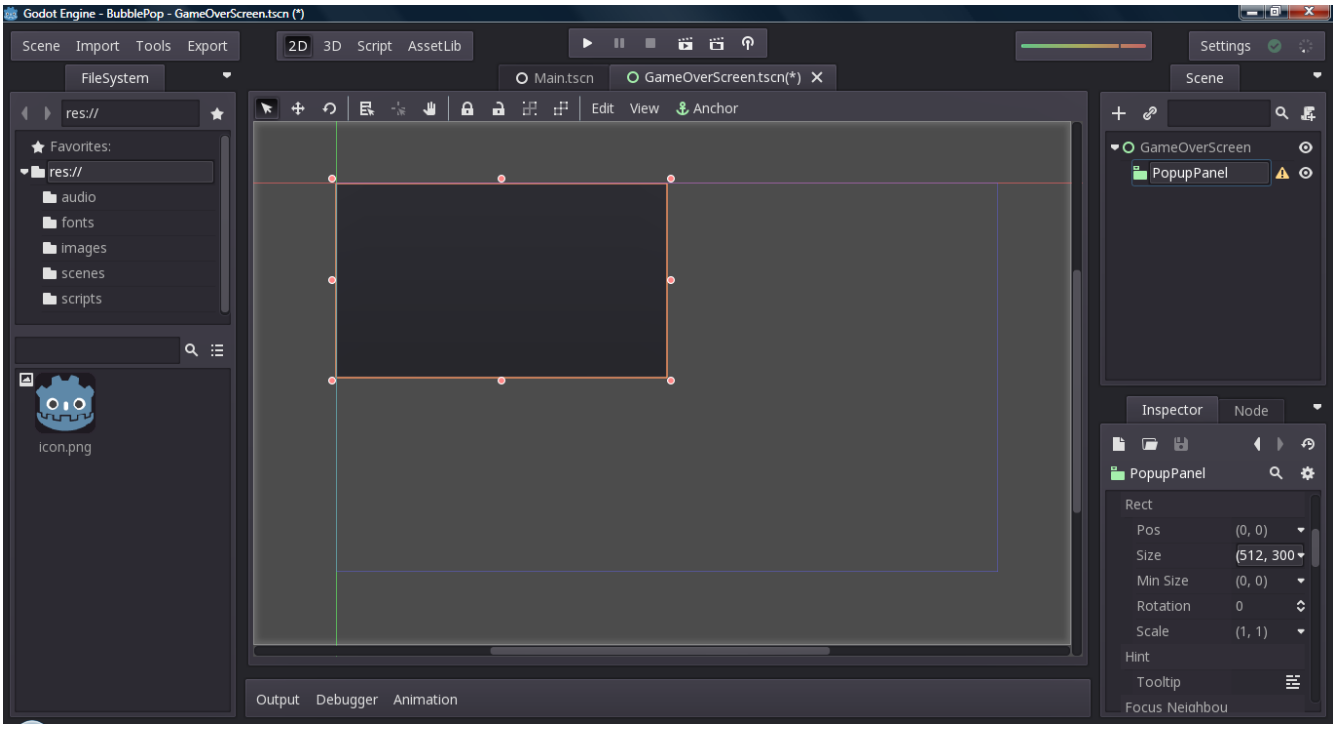
A PopupPanel is a type of node that works similar to a dialog box. However, it has no title bar or child widgets. By default, a PopupPanel is hidden. Click the closed eye icon beside the PopupPanel to show it:



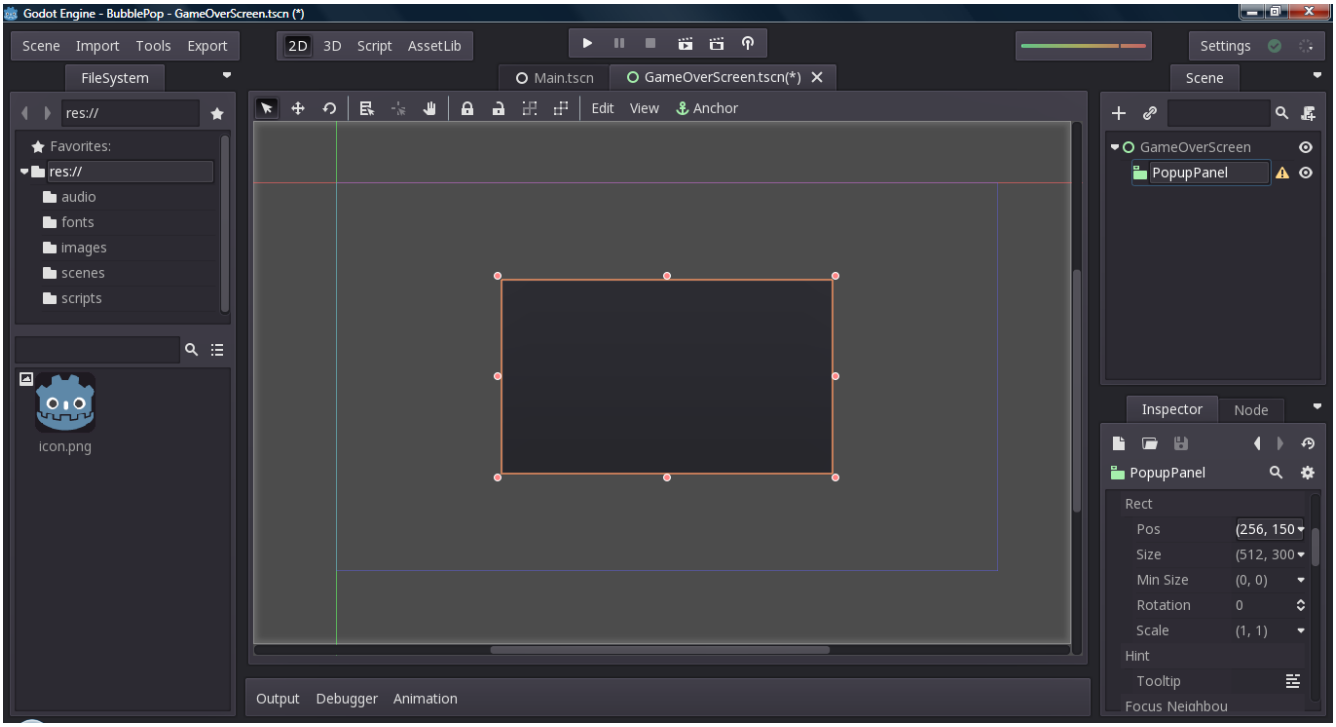
Notice that we now have an exclamation point beside the PopupNode. That is because when we run our game, the PopupPanel will be hidden by default even though we have it visible in the editor. I will explain how to display a hidden PopupPanel later in this lesson. For now, let's resize our PopupPanel and move it to the center. If you scroll the lower right pane, you will see a "Rect" section:



These values affect the position, size, minimum size, rotation, and scale of the selected Control node (keep in mind that all GUI elements inherit from Control). Let's set the size to (512, 300):

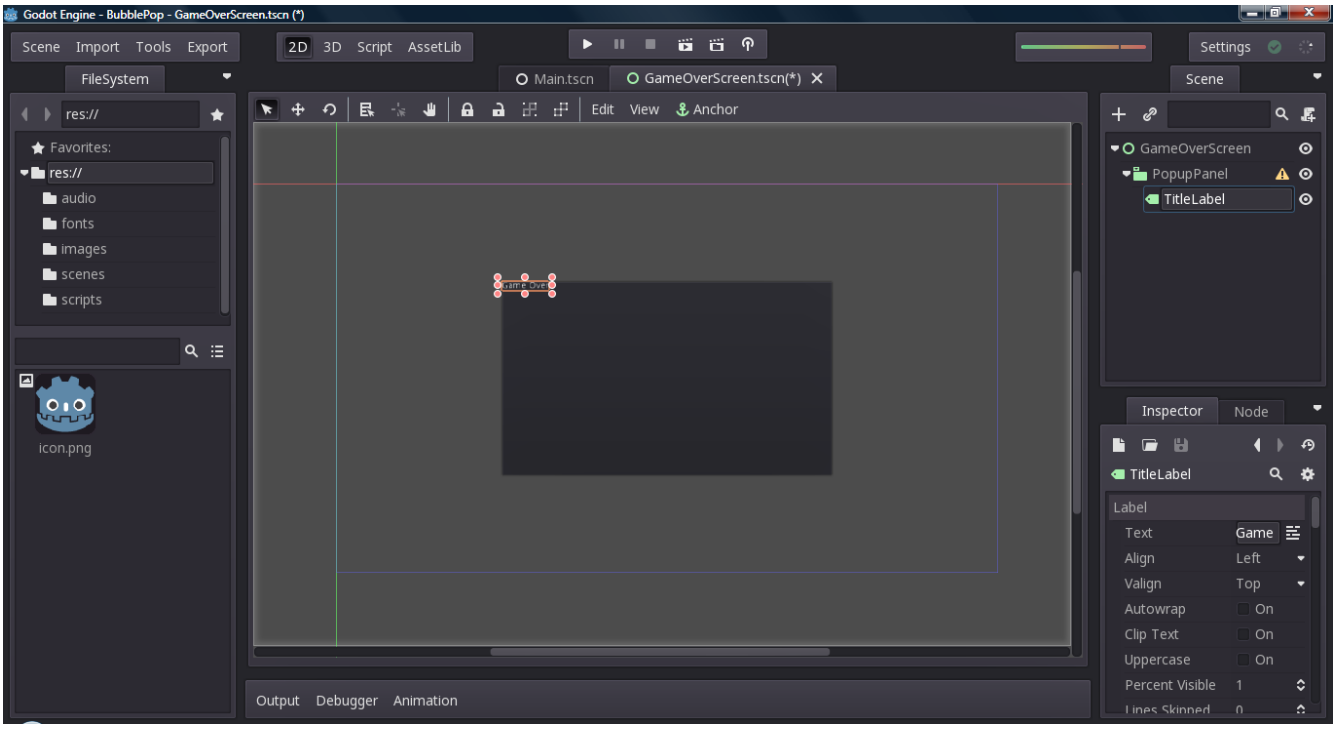


Now let's set the position to (512 – 512 / 2, 300 – 300 / 2):

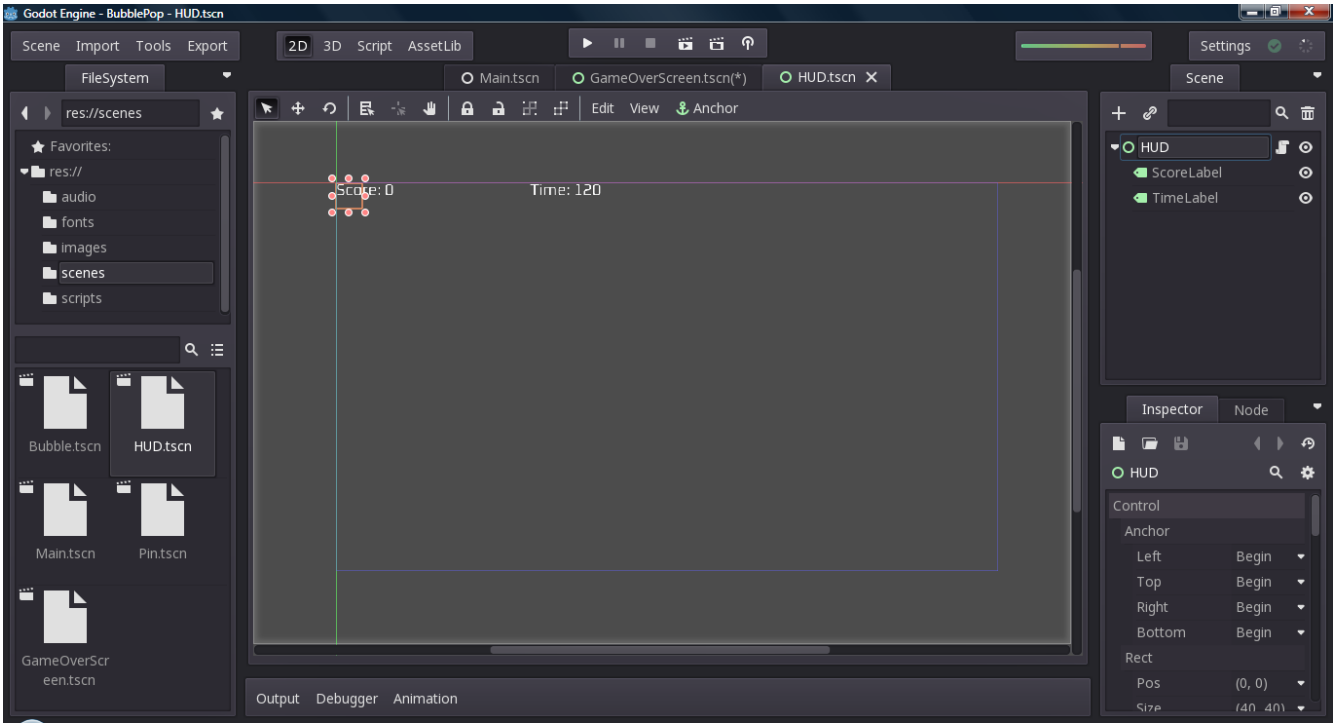


Notice that Godot allows you to enter a formula into a box that requires a numeric value. If we do this, Godot will automatically replace the formula with its value after solving it. It is a very convenient feature.

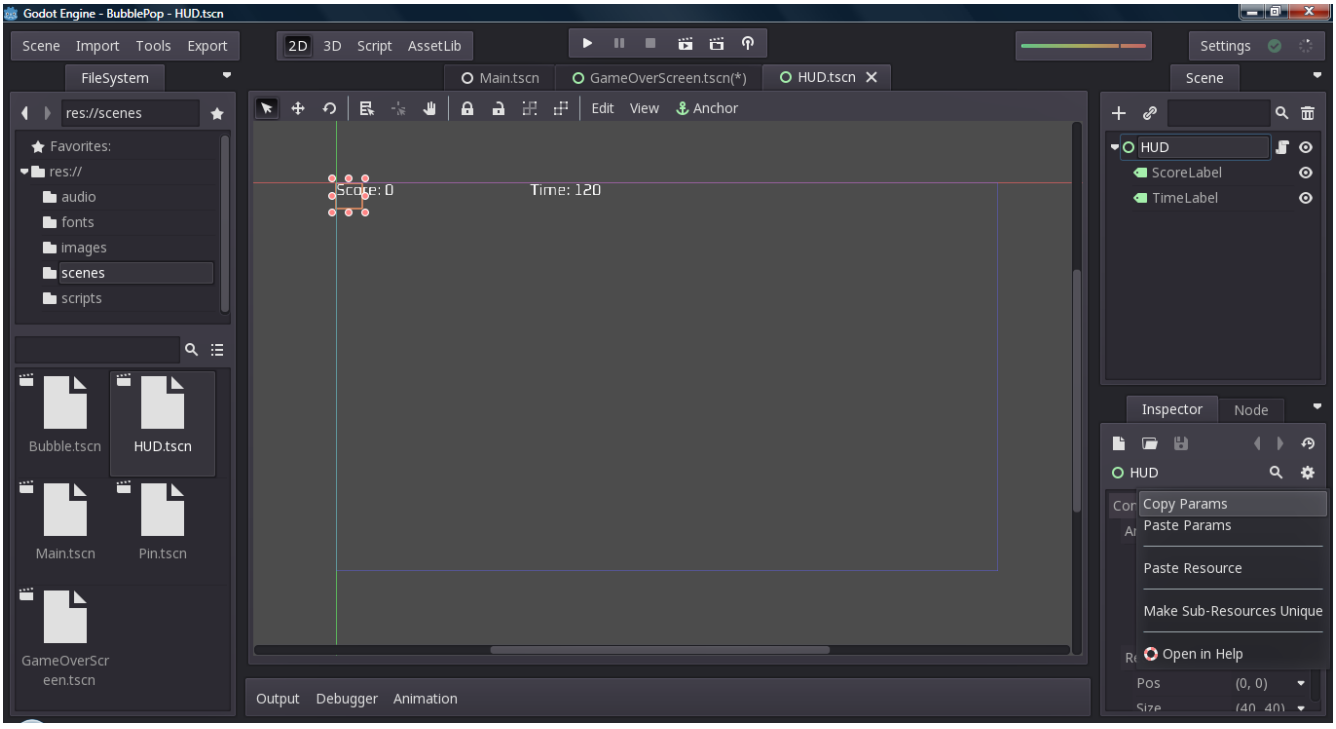
Next we will create a new label for our title:



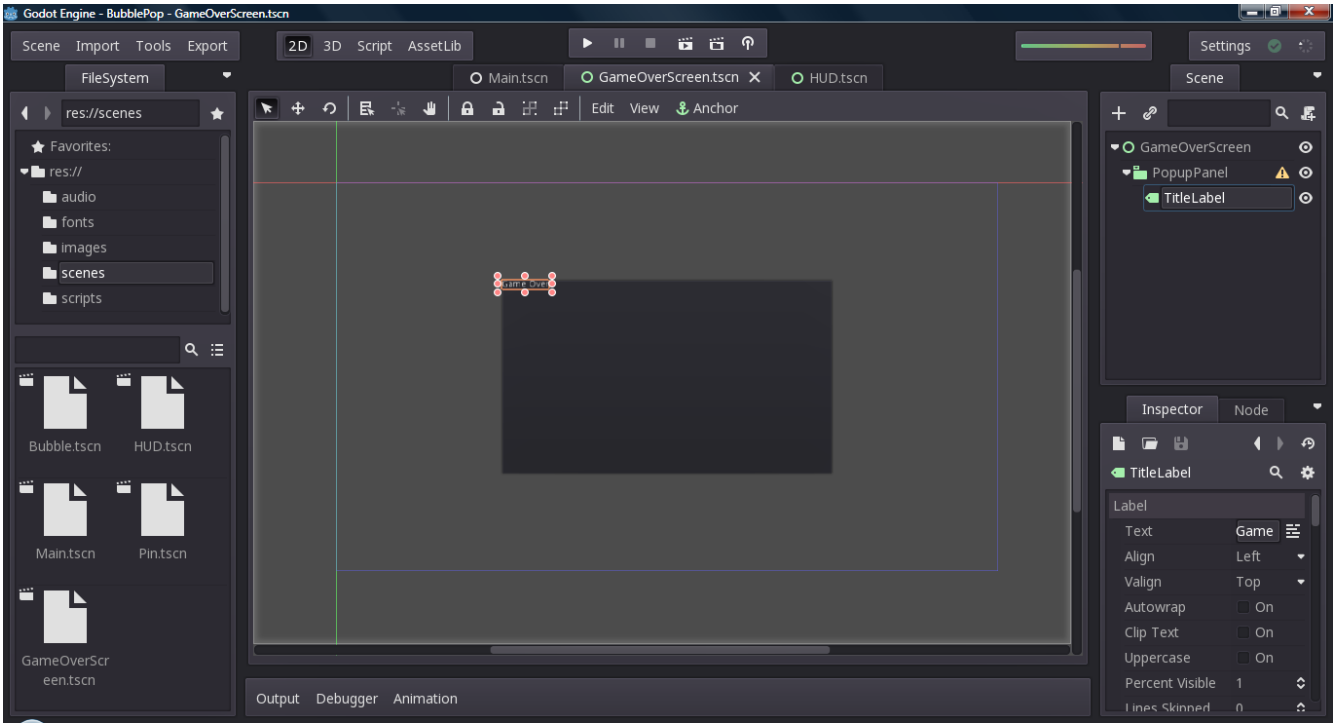
Uh oh… Looks like our text is tiny again. Why is that? This happened because we have not assigned our theme to the root node of this scene. Open your “HUD” scene:



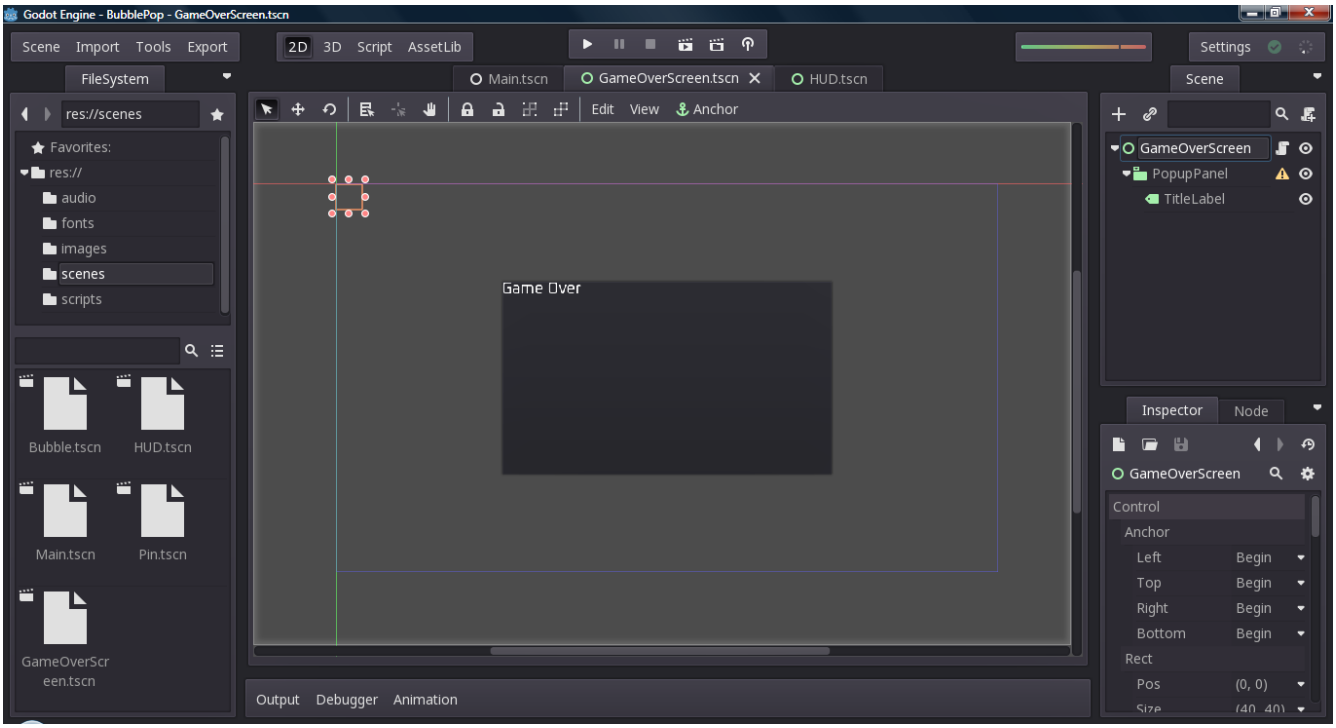
Now select the root node and click the gear icon in the lower right pane:



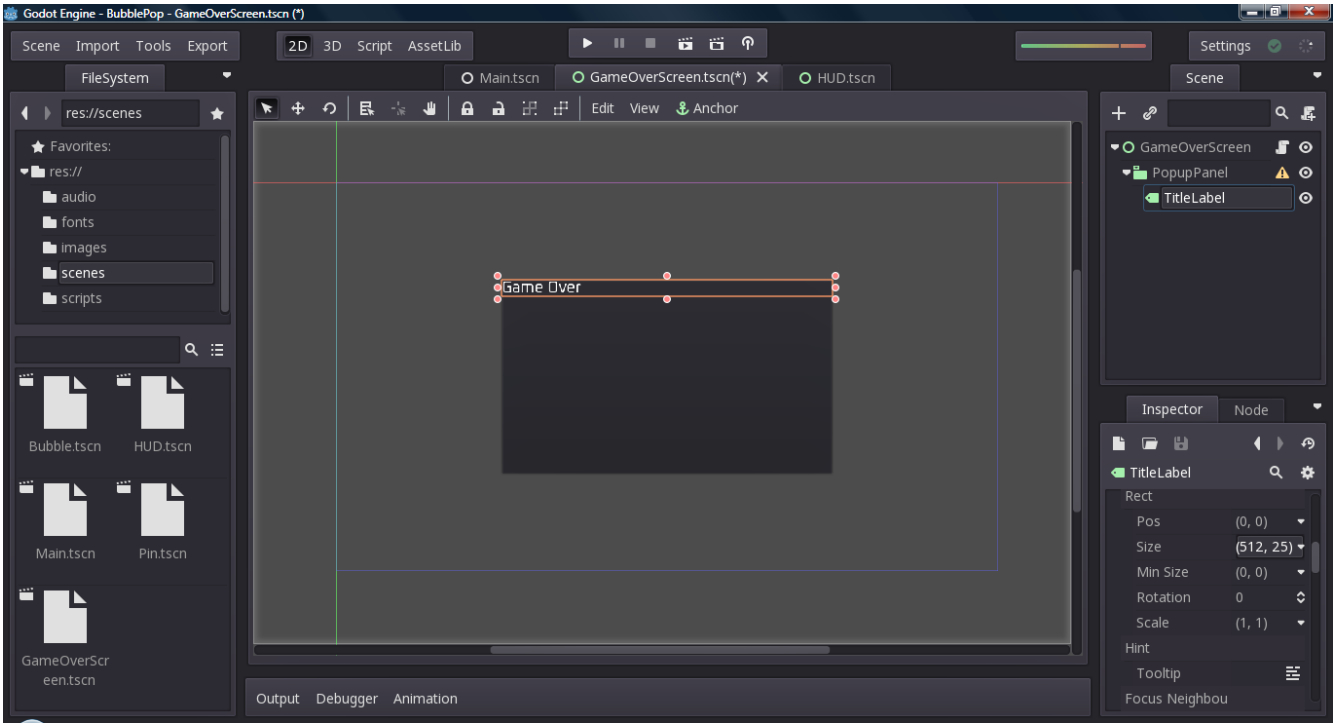
Choose “Copy Params”. This will copy all the properties for the selected node. Then switch back to your “GameOverScreen” scene:



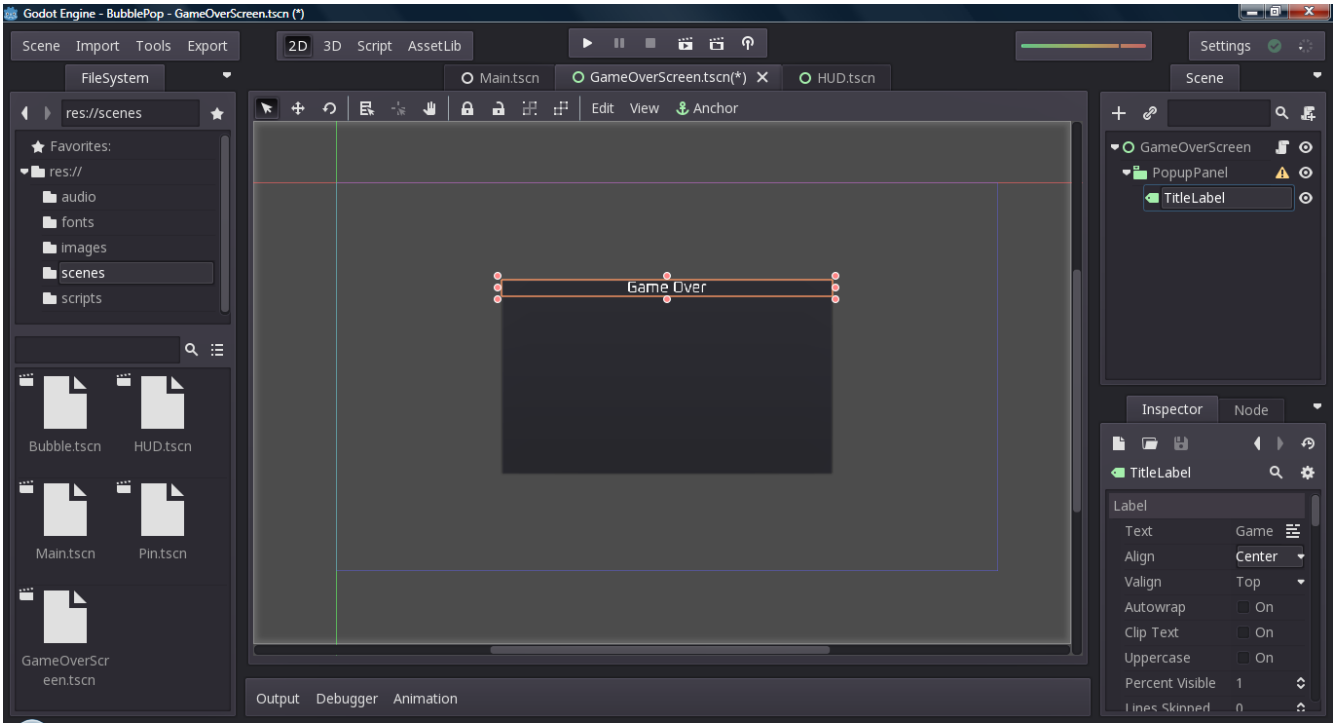
Select the root node, click the gear icon in the lower right pane, and choose "Paste Params":



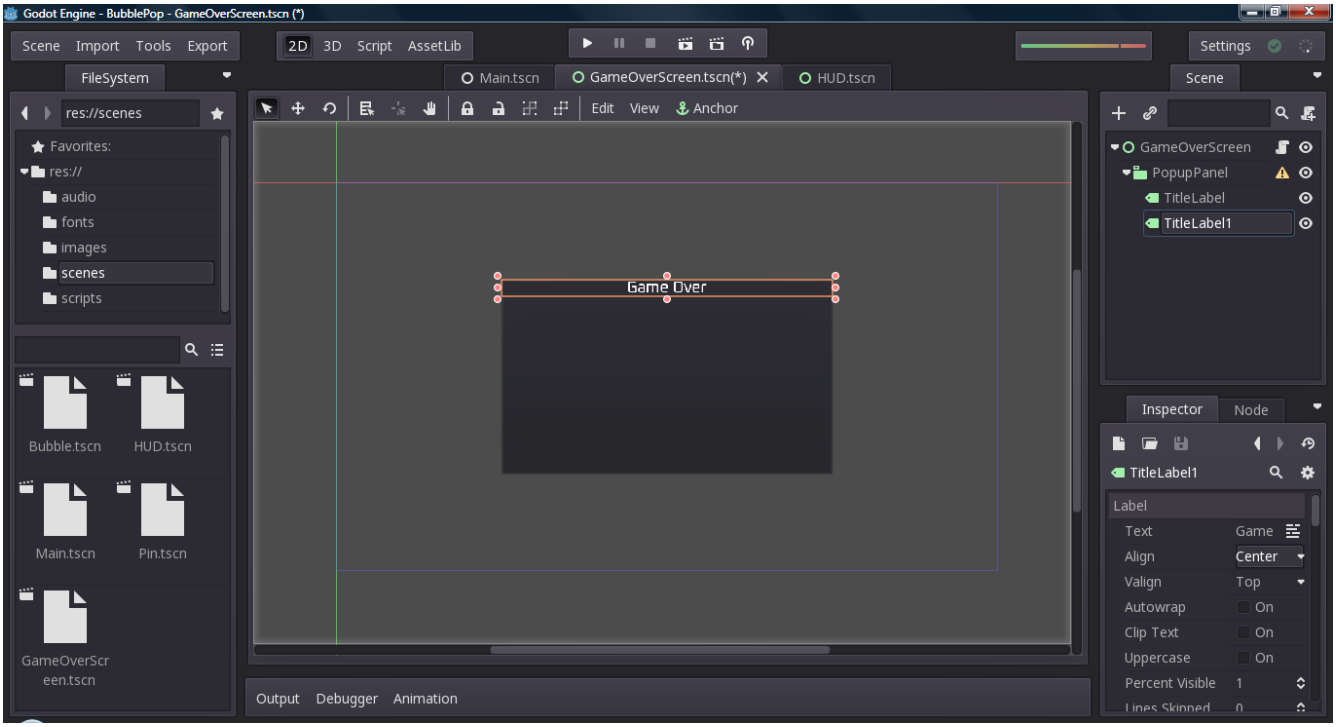
That's better. Now I will teach you a way to easily center the title. First, select your title label and set the width to 512:



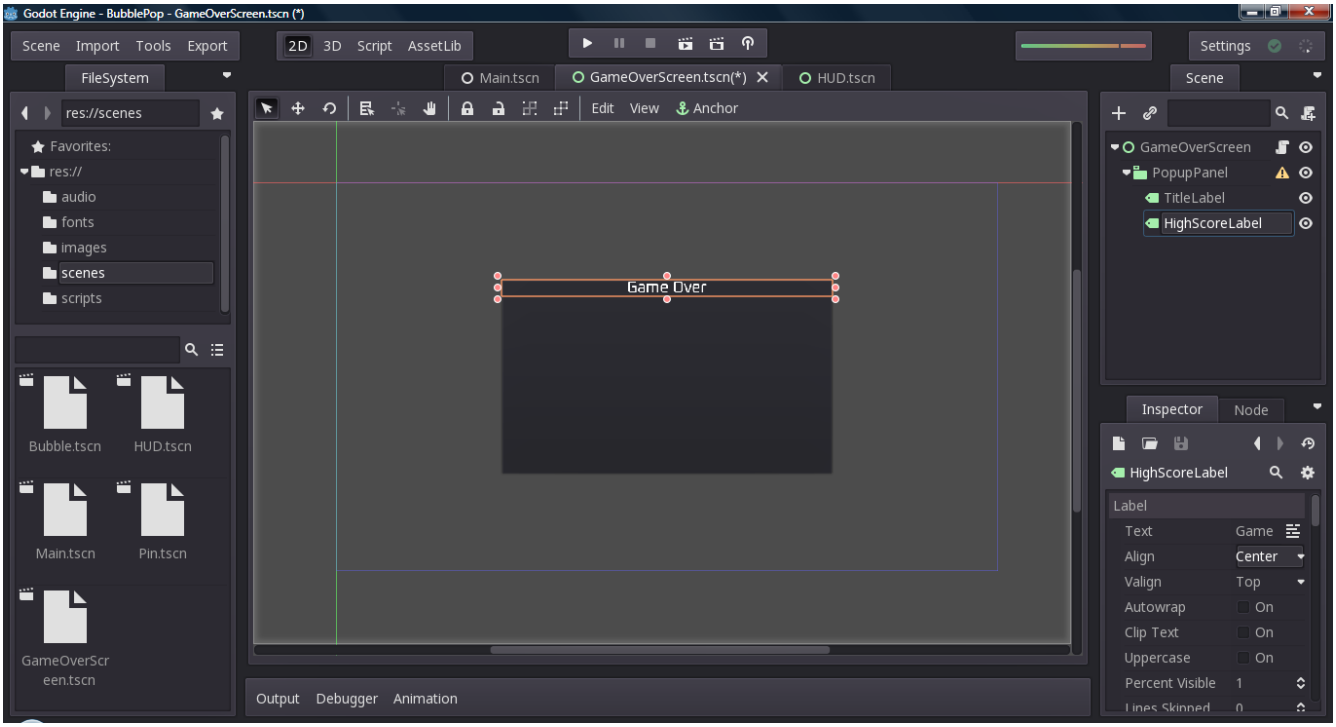
Now scroll up and set the “Align” property to “center”:



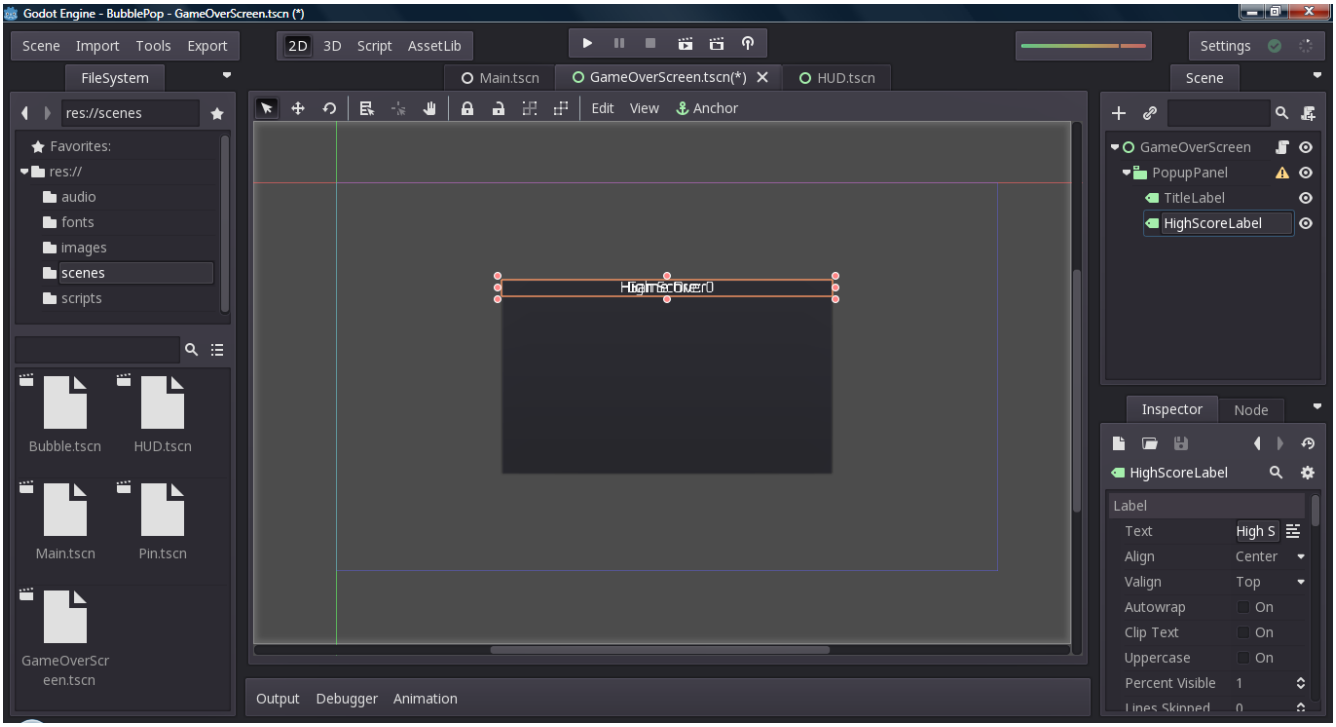
Now let’s duplicate our label:



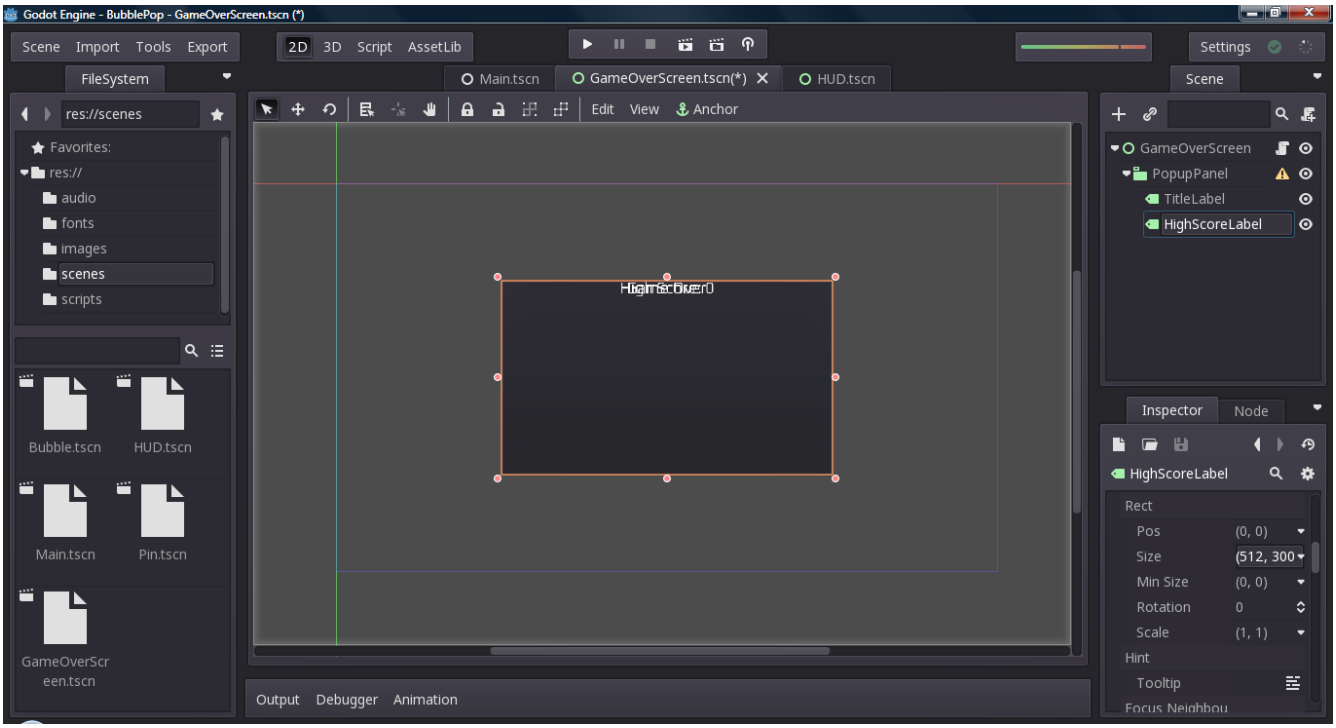
Change the name of the new label to “HighScoreLabel”:



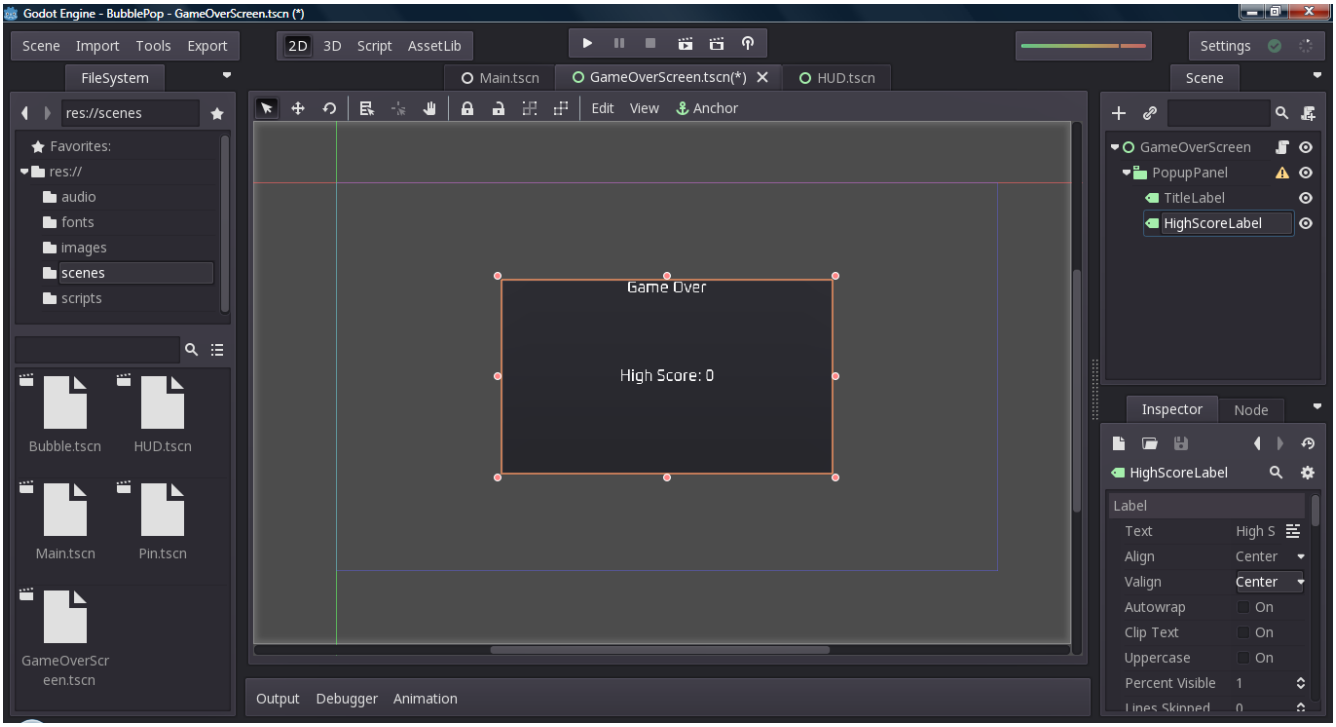
Now change the text to “High Score: 0”:



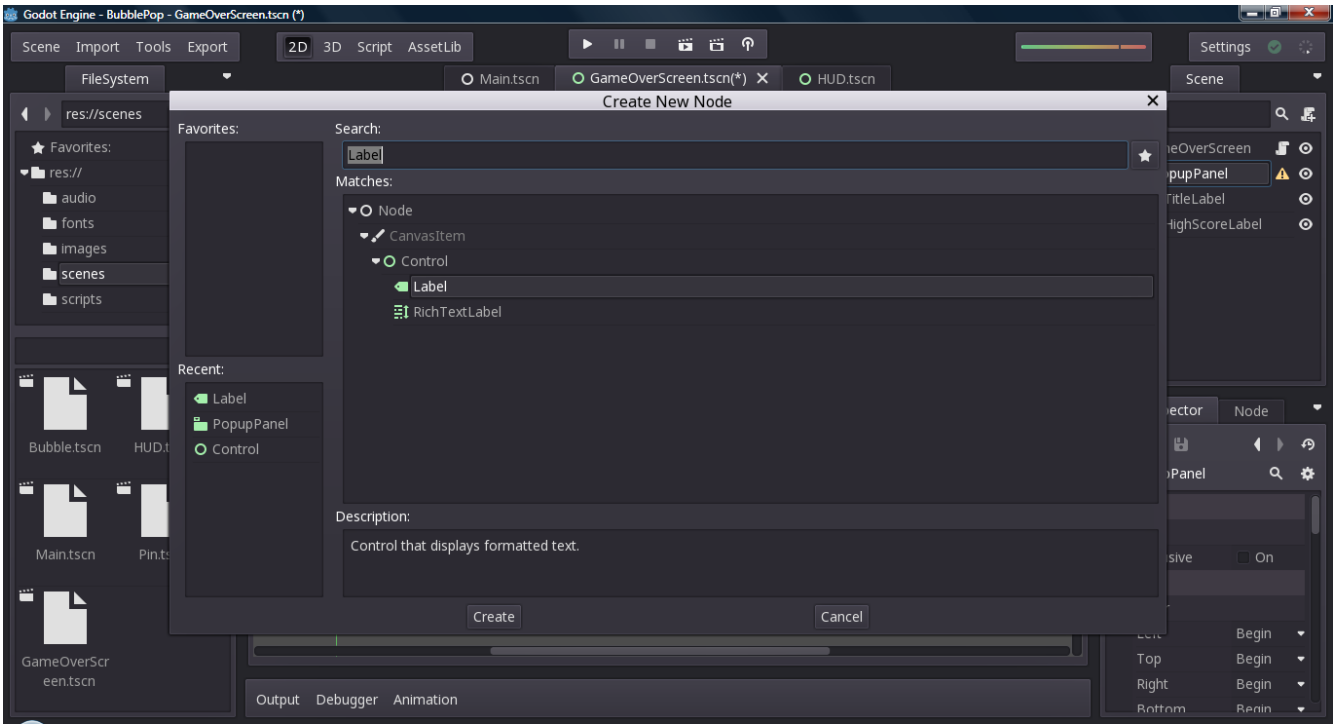
To fix the overlap, we are going to use a slightly different technique which will center the label vertically too. Set the height to 300:



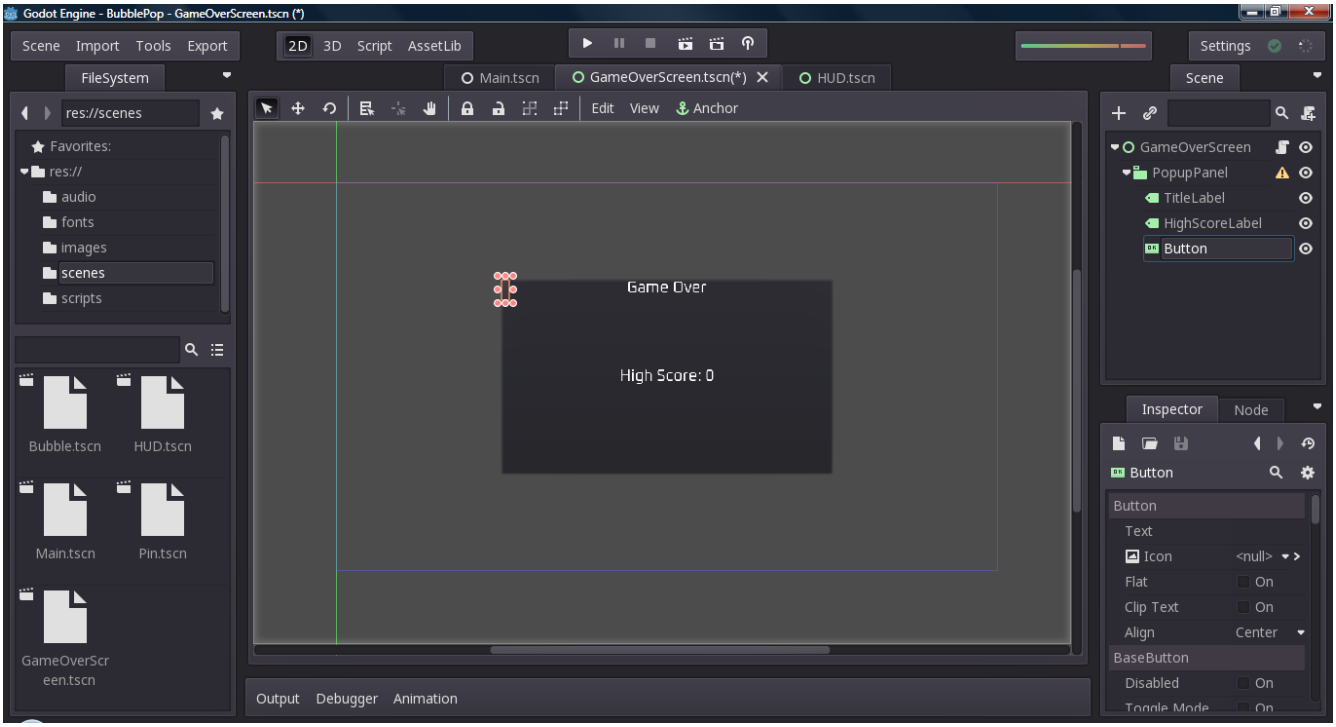
Now set the “Valign” property to “center”:



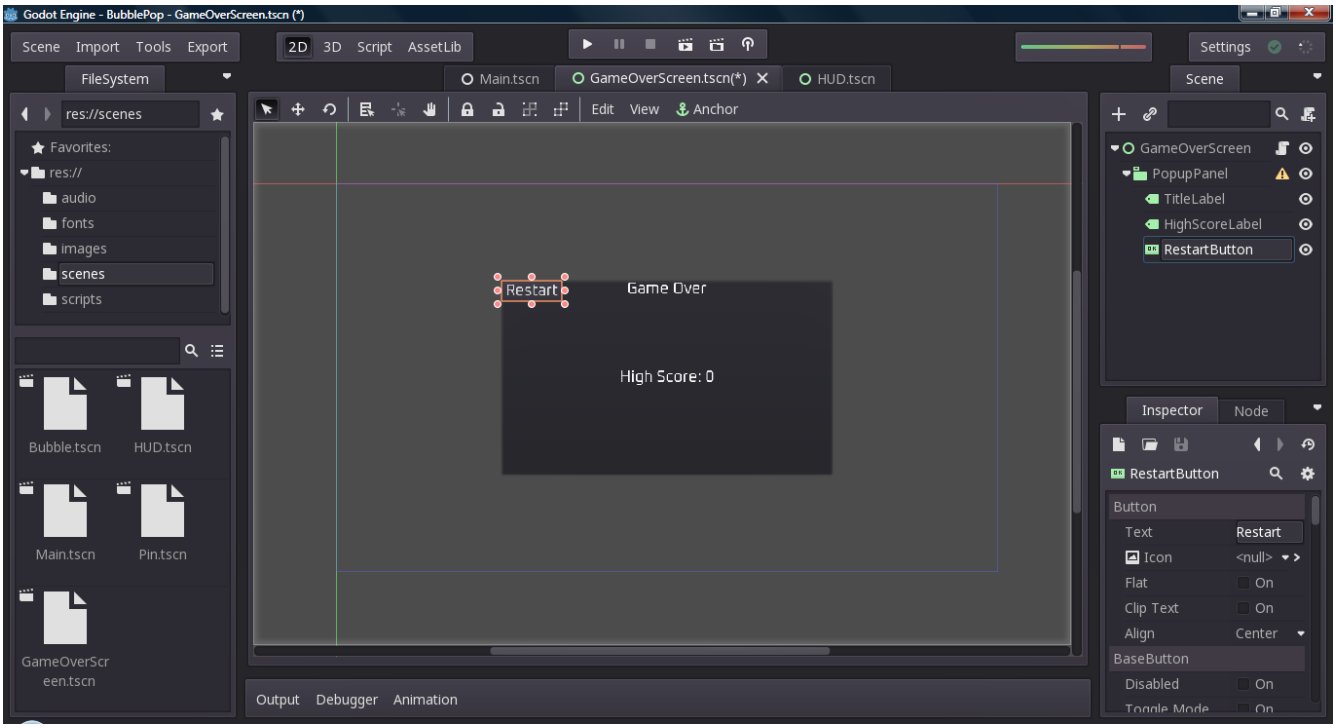
Now let’s create a button for our game over screen. Right-click your popup panel and choose “Add Child Node”:



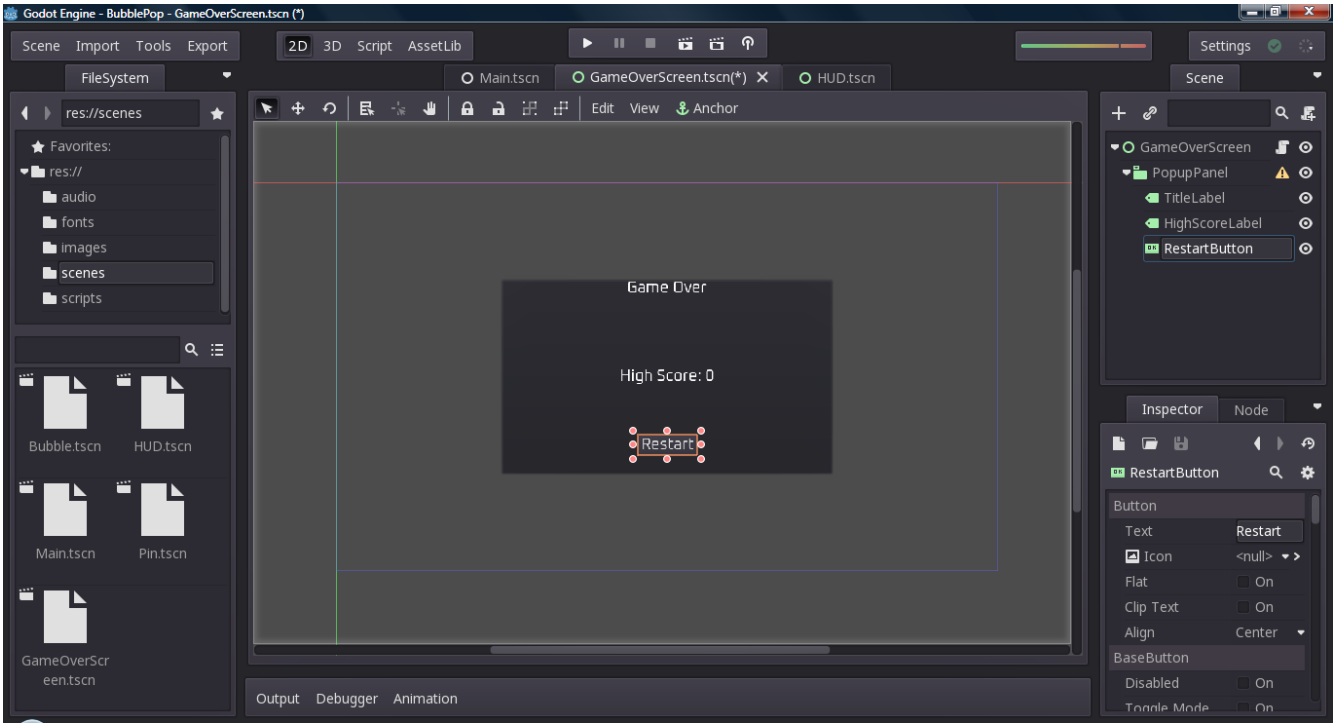
Now select Button:



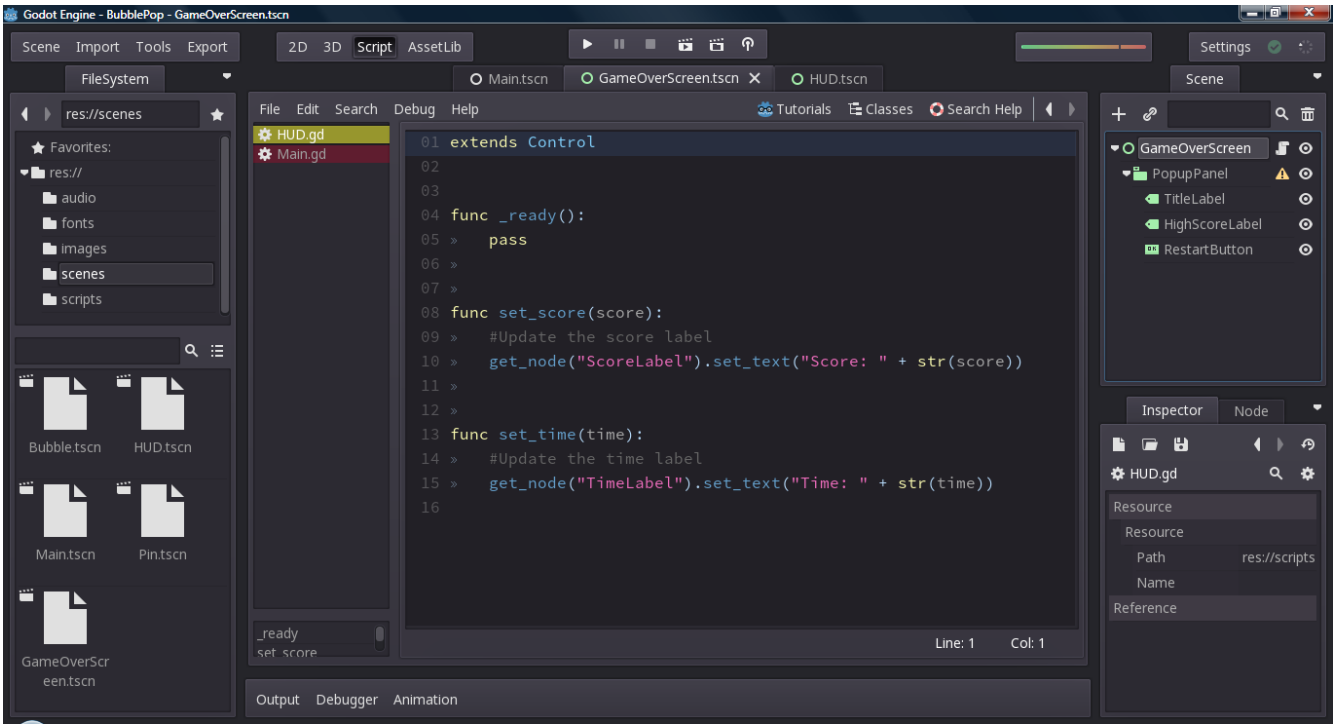
Change the name of your button to “RestartButton” and set its “Text” property to “Restart”:



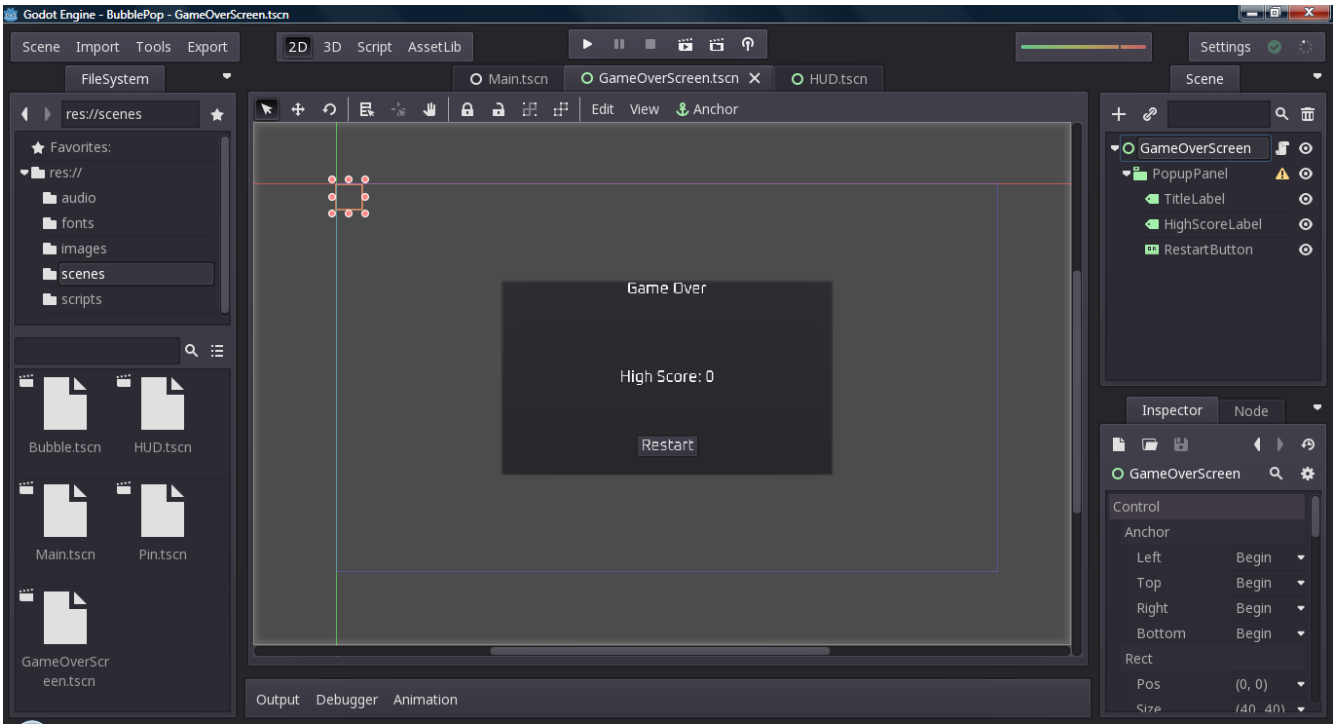
Now let’s drag our new button into place at the bottom of the PopupPanel:



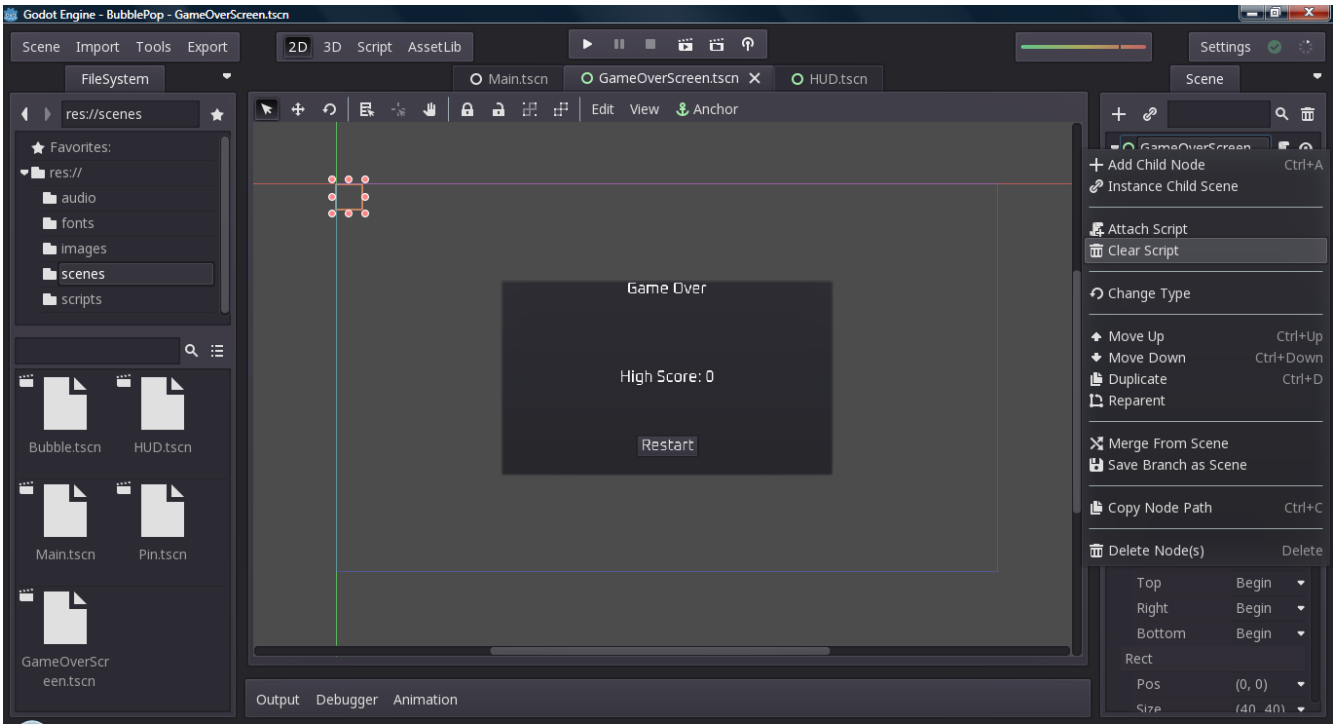
Now we are ready to write a script for our game over screen. Right-click your "GameOverScreen" node and choose "Attach Script":



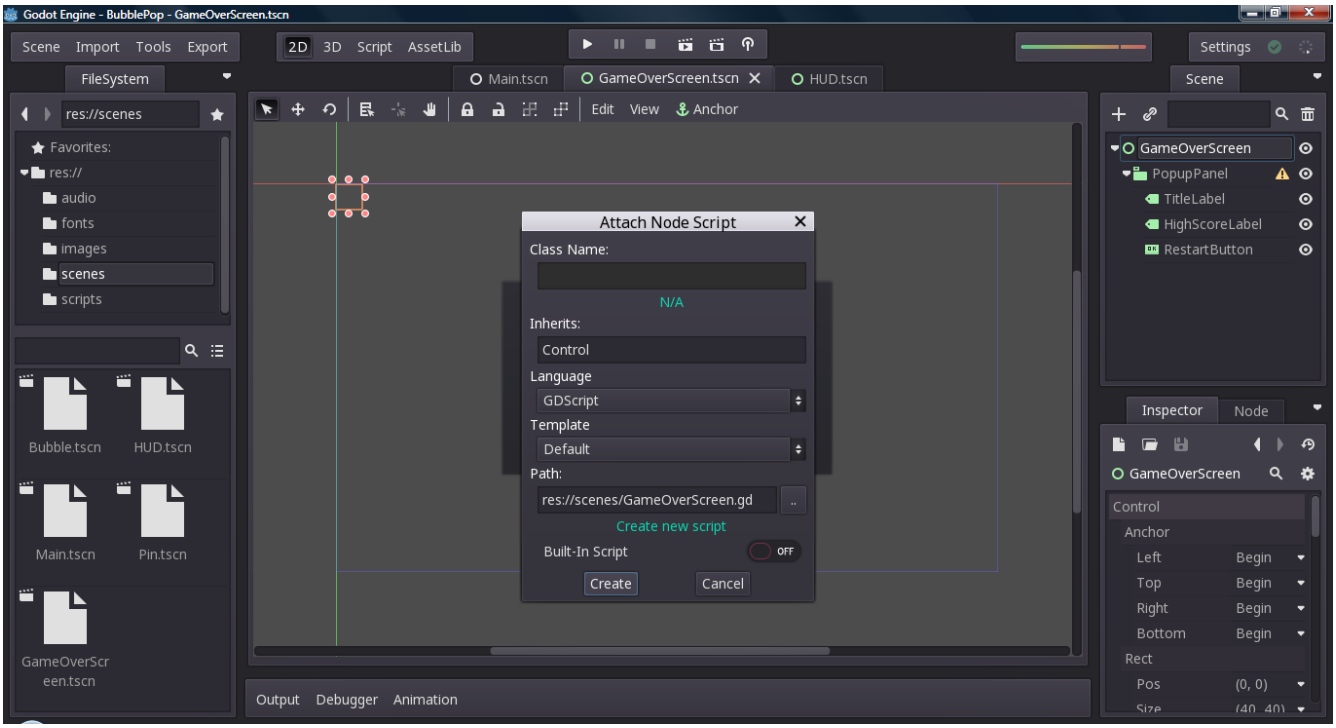
Huh? No script dialog opened and instead Godot opened the script for our HUD… What is going on? When we copied the properties of our HUD to our game over screen, we inadvertently copied the "Script" property that defines which script is attached to the node. To fix this, we will first select our "GameOverScreen" node:



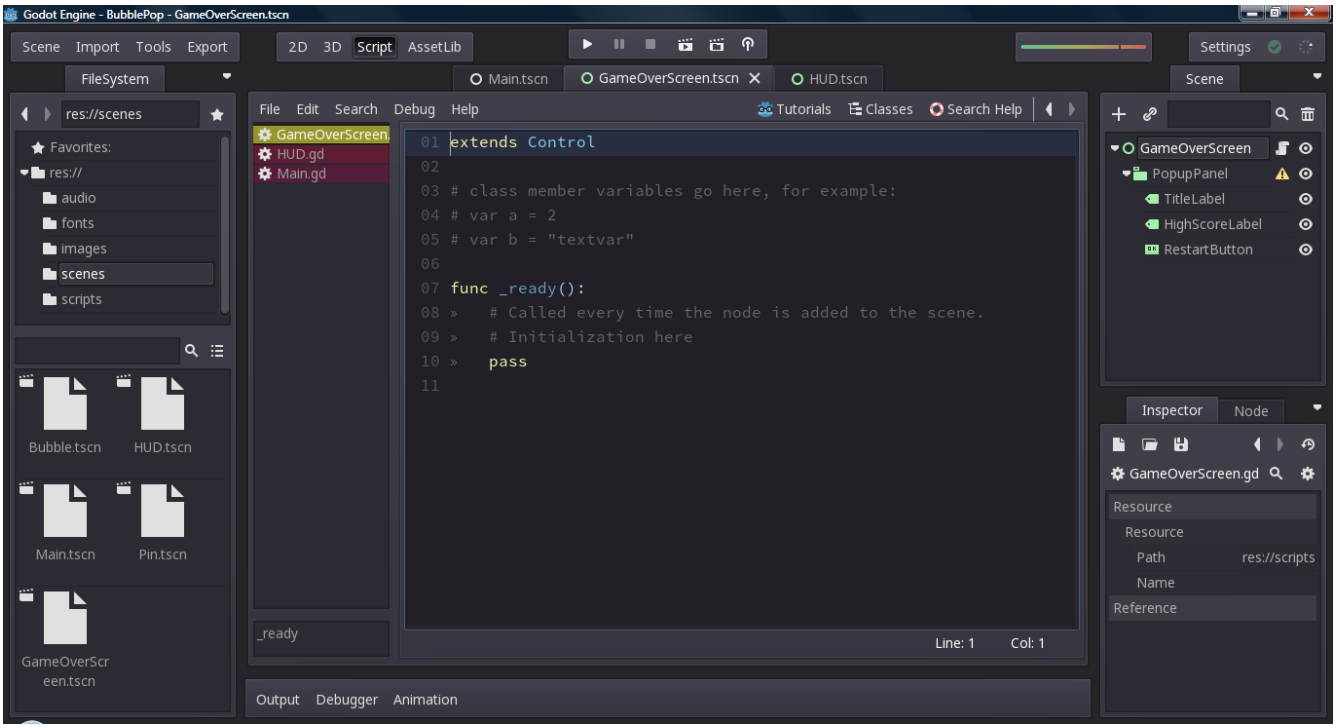
Right-click it:



Choose “Clear Script”. Don’t worry though. This will not delete the script for our HUD or affect the properties of our HUD in any way. We are simply detaching the HUD script from our game over screen. Now try attaching a new script again and it will work as expected:



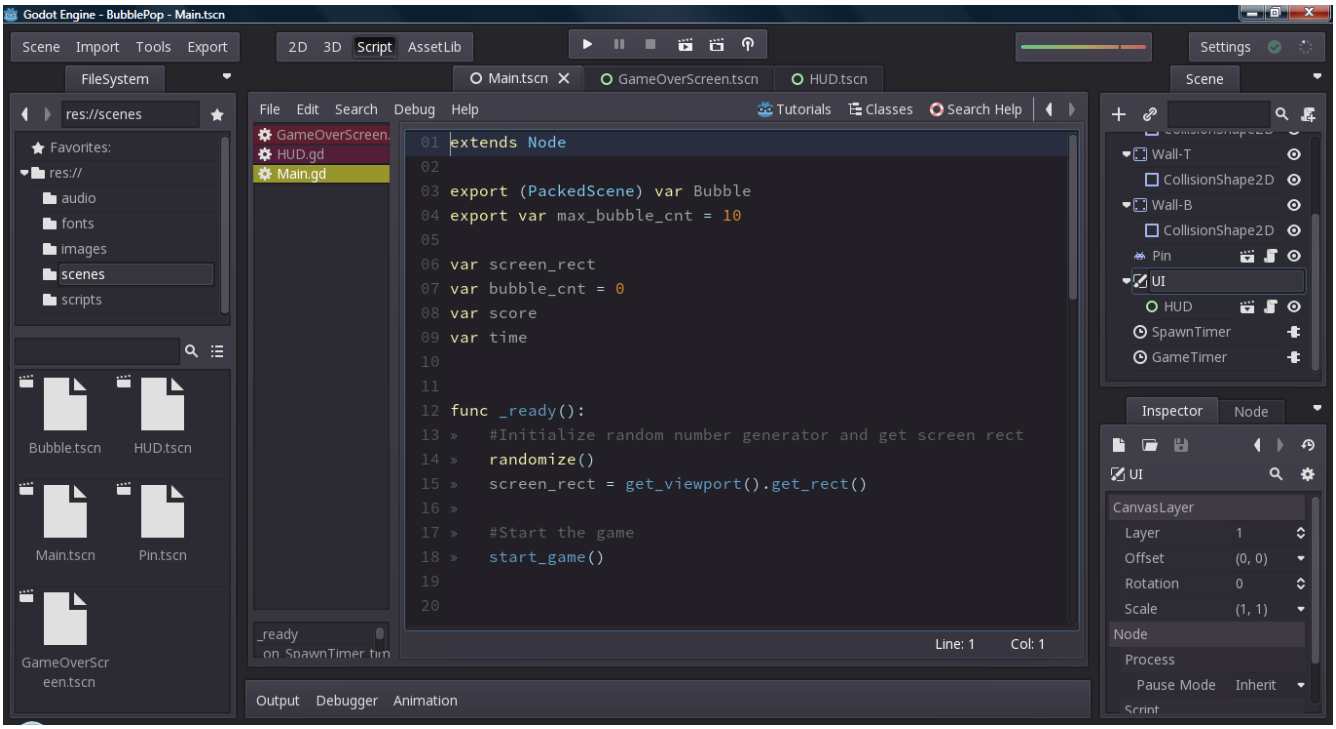
As usual, save the new script in your “scripts” folder:



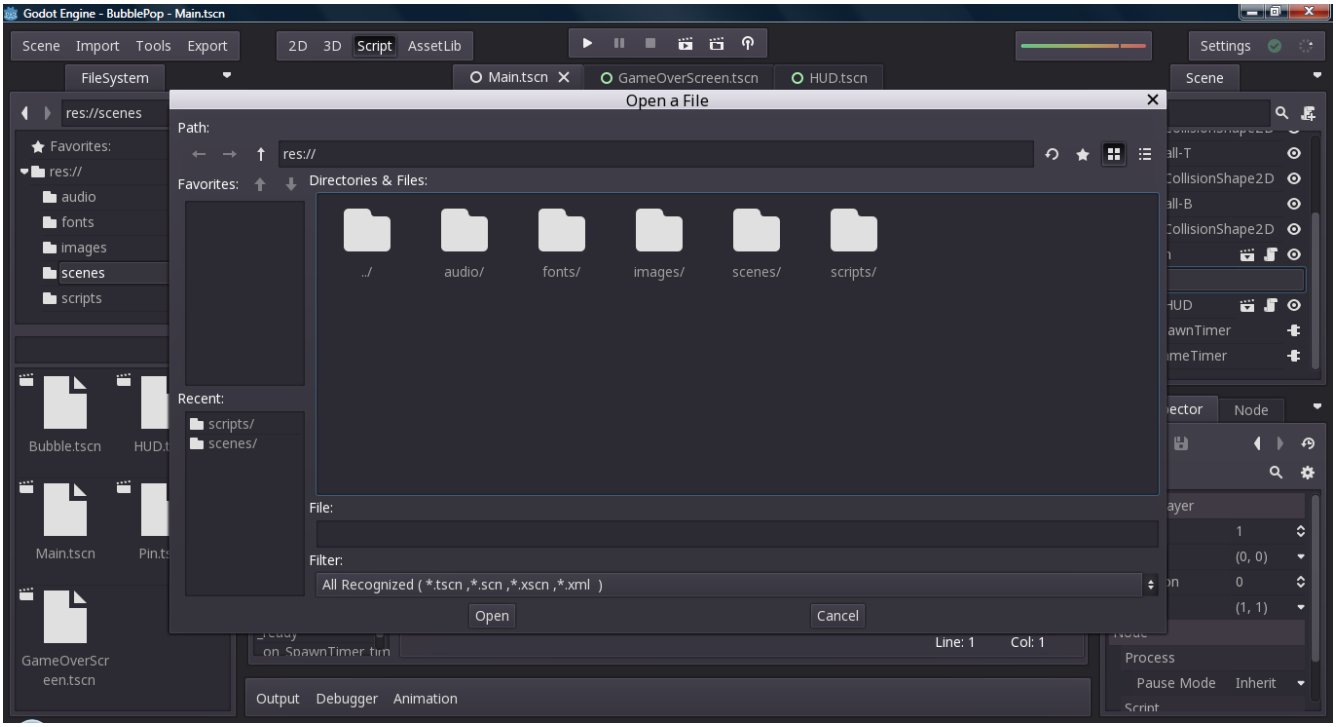
Now we can start writing the logic for our game over screen. Create a new function called “show_high_score”:

```
func show_high_score(score):
	#Display high score dialog
	get_node("PopupPanel").popup()
```

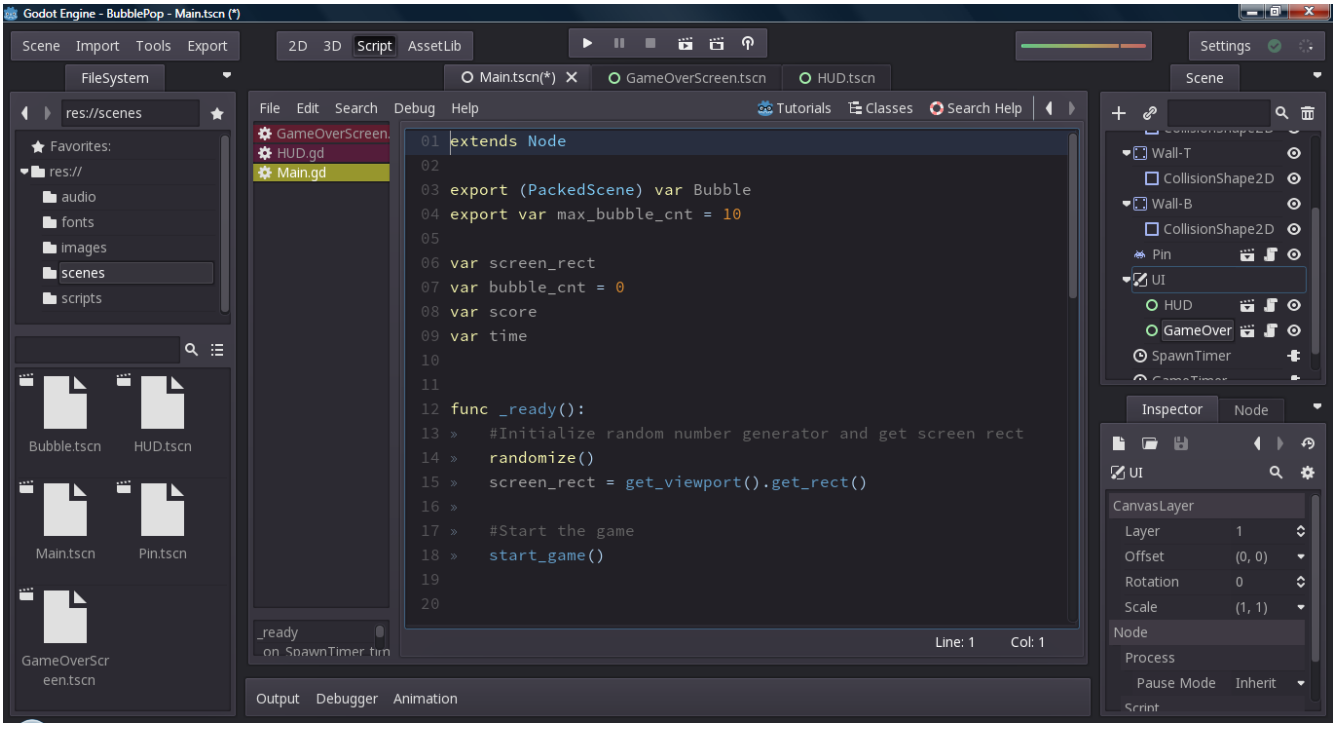
This will simply display our high score dialog when called. Now let's add our game over screen to the main scene. Open your "Main" scene:



Right-click the "UI" node and choose "Instance Child Scene":



Now select your “GameOverScreen” scene:



Now we need to write code that will stop the game when time has run out. Let’s create a new function called “stop_game”:

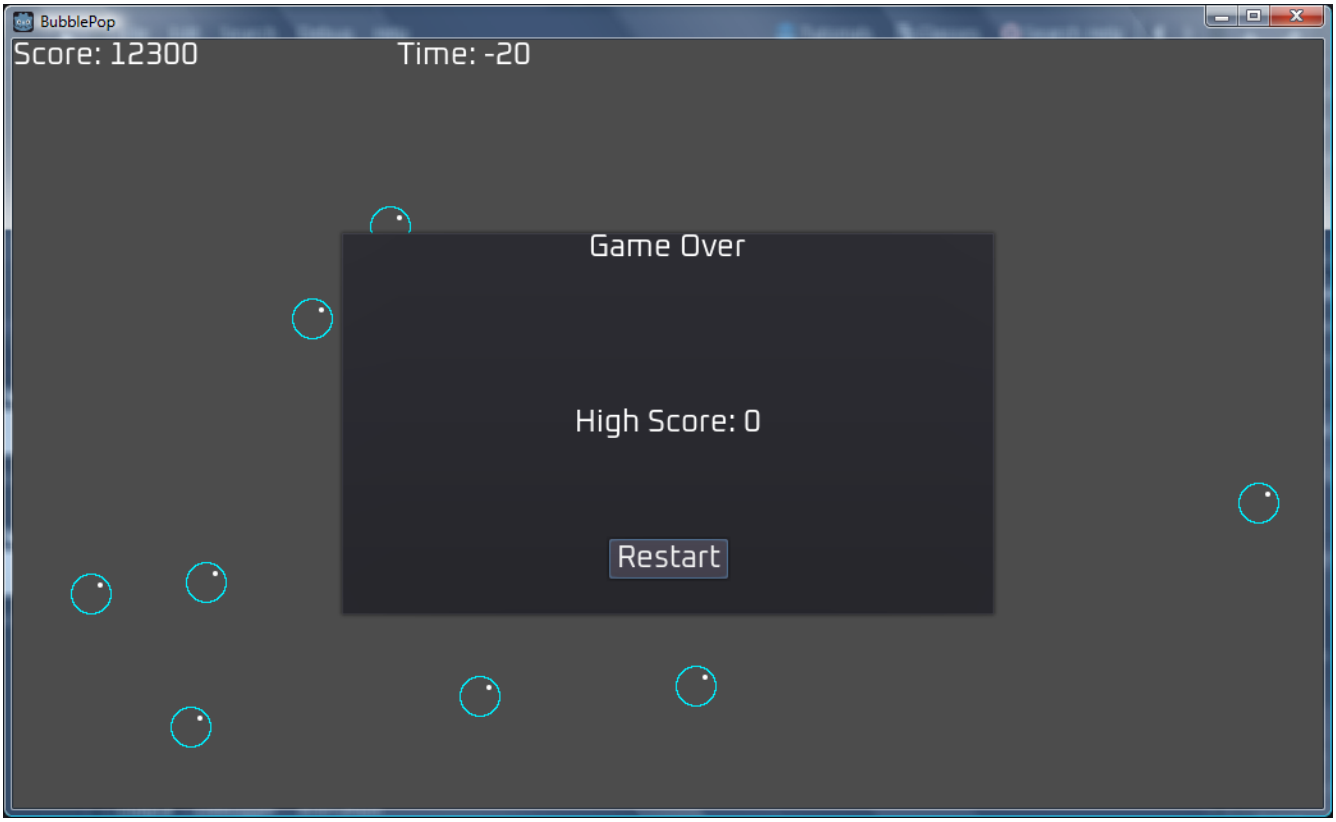
```
func stop_game():
        #Show high score dialog
        get_node("UI/GameOverScreen").show_high_score(score)
```

We will also need to modify our "_on_GameTimer_timeout" function:

```
func _on_GameTimer_timeout():
        #Update the timer variable
        time -= 1
        get_node("UI/HUD").set_time(time)

        #Are we out of time?
        if not time:
            stop_game()
```

Now let's try running our game and see what happens:



The high score dialog appears, but the game is still running in the background. And if we click outside the PopupPanel, it vanishes:

The reason why our game is still going, is because we never stopped our spawn and game timers. And the bubbles will need to be disposed of too. Let's start by stopping our timers:

```
func stop_game():
	#Stop all timers and show high score dialog
	get_node("SpawnTimer").stop()
	get_node("GameTimer").stop()
	get_node("UI/GameOverScreen").show_high_score(score)
```

Now if we run our game again, the timer will stop at 0 and no more bubbles will spawn. But how can we dispose of all the bubbles that are in play? We could create a function that loops through all the nodes and deletes each bubble it finds, but that would be kind of tedious and unnecessary. Godot actually provides something better that we can use. I will be teaching it in the next lesson.