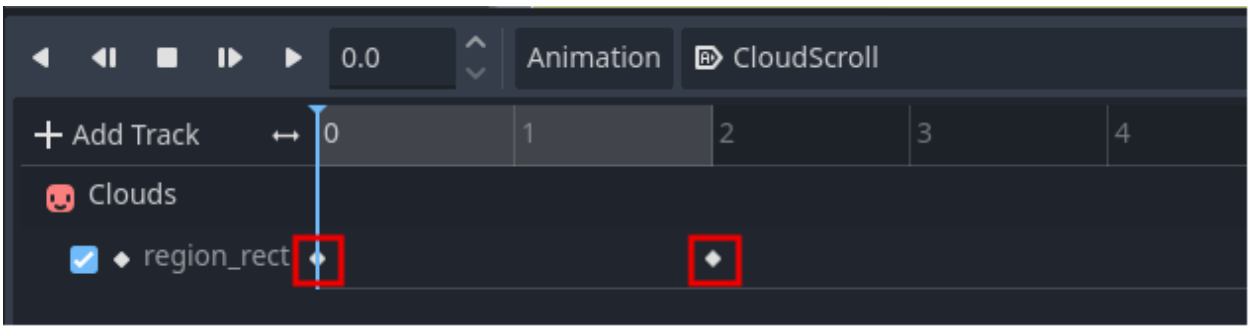
CODE NINJAS®

Bronze Belt Ninja Guide

Activity 08: Dropping Bombs Part 2

Animations

An easy way to improve the user experience and make a game more immersive is to animate its characters. Godot offers a host of nodes to facilitate the addition of animations. In this project, the **AnimationPlayer** and **AnimationTree** nodes will be used. Why are there two different animation nodes, and how do they work together?

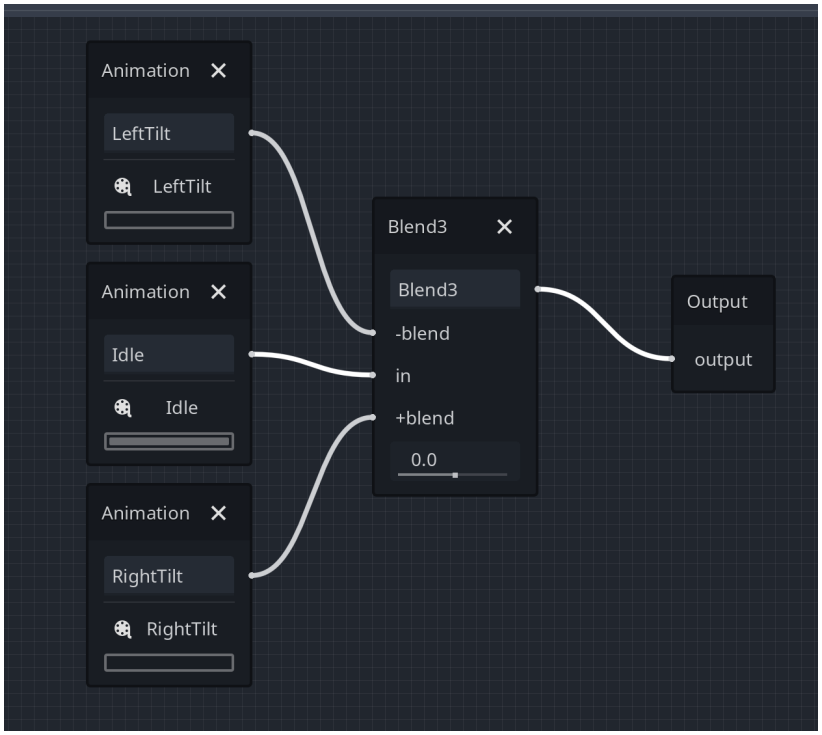


An **AnimationPlayer** node is used to create separate animations targeting one kind of node. Each **AnimationPlayer** node can store several different named animations. Within each of those animations, key frames, or **keys,** can be created to define different "extremes" of positions that the model is supposed to reach during its animated movements.

For example, in this activity, the background will show clouds scrolling vertically across the screen. This is achieved with 2 **key** frames: the first **key** shows the clouds in the middle of the screen as its default position, and the second **key** shows the clouds placed down on the screen after 2 seconds. Godot calculates the positions the model needs to be set to in between those two **keys** and will attempt to blend them smoothly. It is this blended transition that produces the animation, such as the clouds moving down across the sky. One or more **keys** can be placed along the animation track in Godot's **Animation Bottom Panel**, the graphical editor used to plot **keys**. Animations within the **AnimationPlayer** node have several other settings, including their duration, whether they loop, and if they autoplay when the node loads in.

An **AnimationTree** node connects to an **AnimationPlayer** node and a target character node. The **AnimationTree** connects the animations stored in the **AnimationPlayer** node. Succinctly, it controls the playback of previously created animations. The **AnimationTree** also has its own editor within Godot's **Animation Bottom Panel** which will always include an **Output** node. The **Output** node only takes one input, so multiple animation nodes must be combined for them to be played together smoothly.

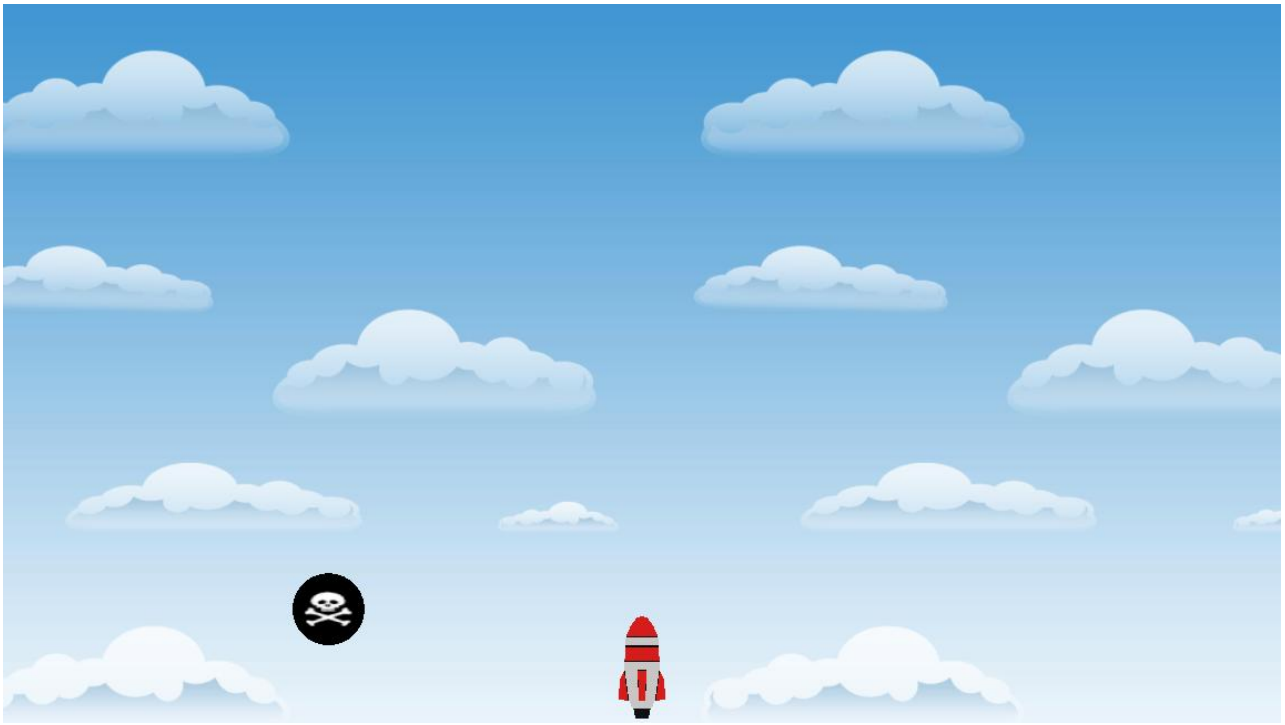
To achieve this, Godot offers several different **Blend** nodes. Each **Blend** node can accept one or more inputs from animation nodes and exports an animation output. Each **Blend** node uses a sliding scale variable that can be manipulated with code to determine which animation node has a stronger influence over the produced animation. This produced animation can then be fed into other **Blend** nodes, or directly into the **Output** node to create an animated character!



Activity 08: Dropping Bombs Part 2

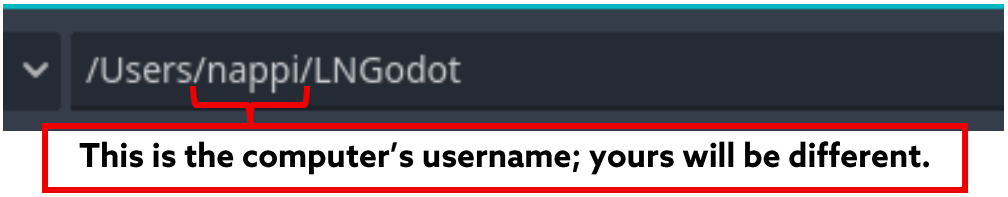
In this project, you will expand on Dropping Bombs by adding textures, meshes, and animations to the Rocket and Bomb nodes. Additionally, you will animate the Rocket to tilt in the direction it flies in as it flies through a new animated sky.

By the end of this activity, you will have explored how to create, connect, and blend animations.



1 All projects will be stored in a path like:
/Users/[MyComputerUsername]/[MyInitials]Godot

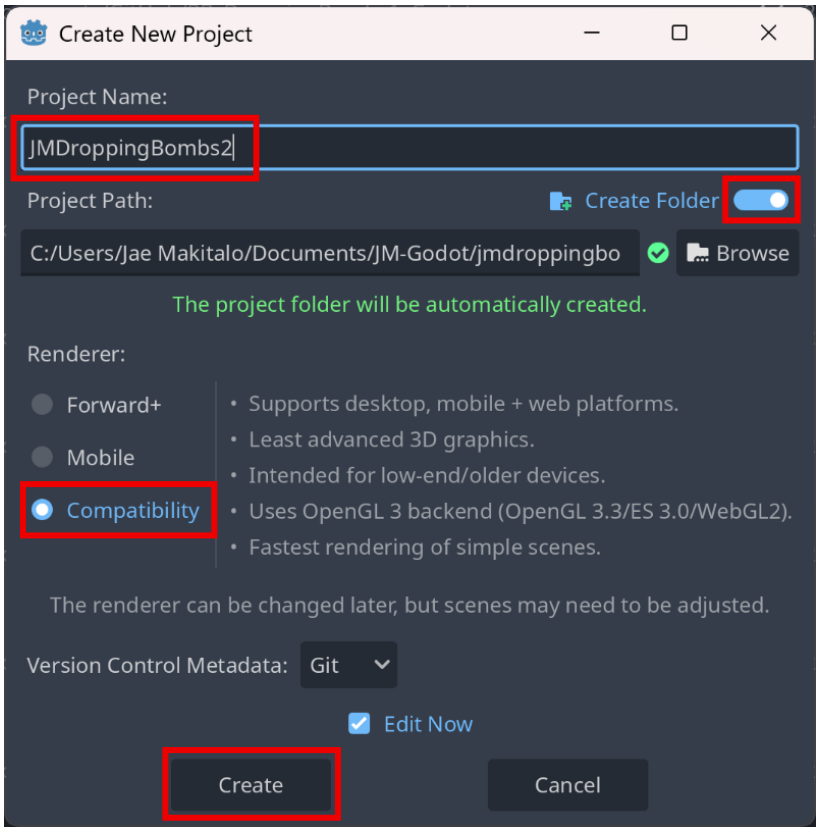
Don't worry if your path looks slightly different from the image shown! All computers have their own username.



2 After opening Godot, in the top left corner select **+ Create.**

A **Create New Project** window will pop up. Name the project **[MyInitials]DroppingBombs2.**

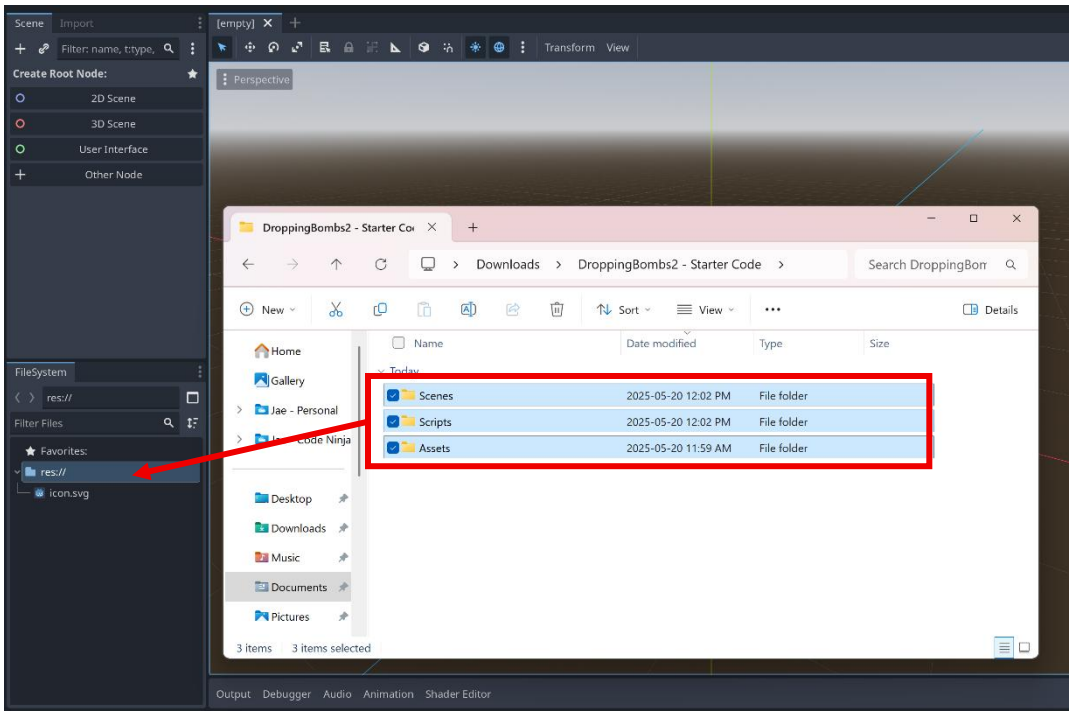
Check that **Create Folder** is turned on, and that the **Compatability** mode for the renderer is being used. Then click **Create**.



3 Don't create the **main scene** and **Main root** node just yet!

Extract **BB Activity 08 – Ninja Starter Pack.zip** and select all folders inside. Drag them into the **res://** folder in **FileSystem**.

At the top center of the editor, check that **3D** is selected.



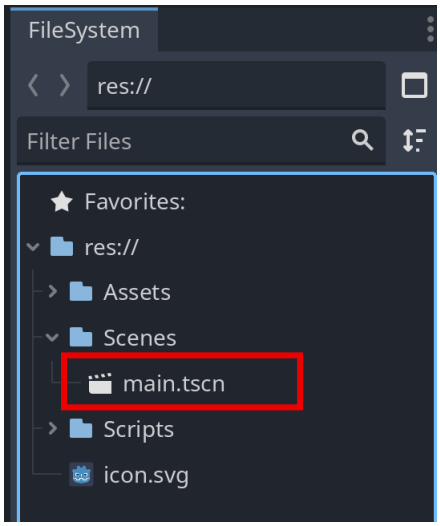
Reminder:

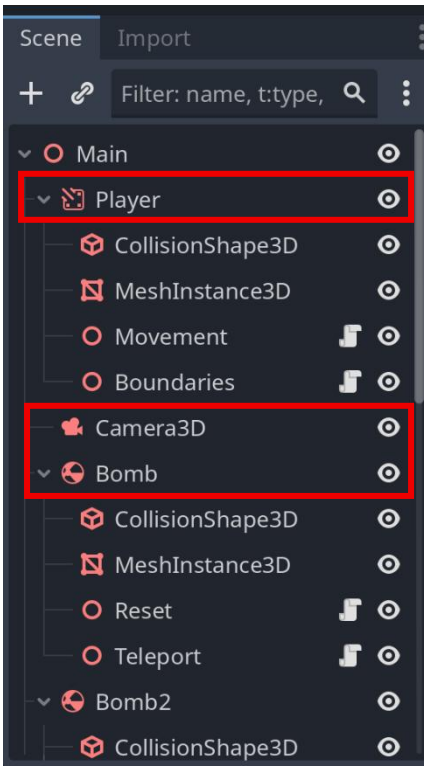
Double-click on the folder icon from **Downloads**. Then **right-click** on the zip file and select **Extract All.** Press **CTRL + J** on the keyboard to reopen the **File Explorer**.

4 An error like this may appear in the debug window – this can be ignored for now.

```
Godot Engine v4.4.stable.official (c) 2007-present Juan Linietsky, Ariel Manzur & Godot Contributors.
--- Debug adapter server started on port 6006 ---
--- GDScript language server started on port 6005 ---
● ERROR: Couldn't open MTL file 'res://Assets/Models/Rocket.mtl', it may not exist or not be readable.
```

5 In **FileSystem**, navigate to **main.tscn** and **open it** by **double-clicking** on it. Notice the main scene already has the player, camera and bomb nodes created from part 1.



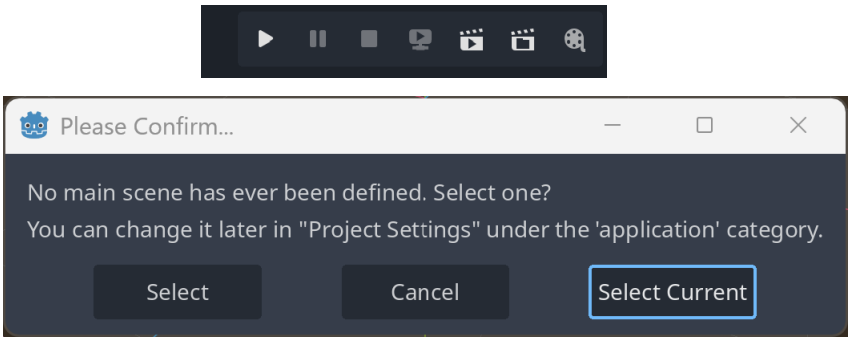


Reminder:

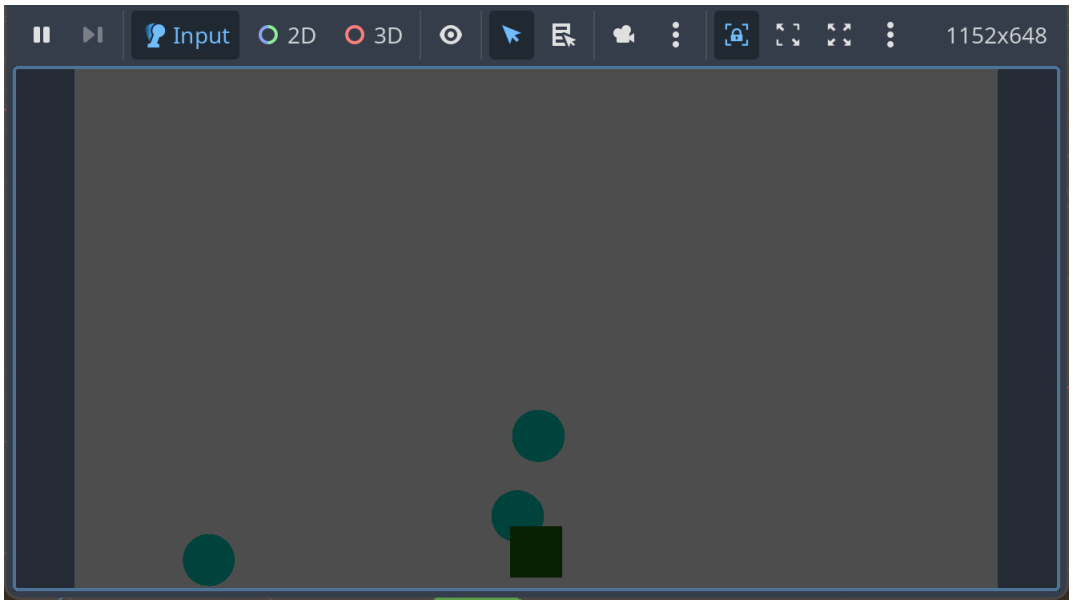
Click the arrows next to the folders to open them.

6 In the top right corner, click the **play** button to run the game.

Click **Select Current** to define the main scene.



7 The game window will now open. Notice that the game is the same as where **Dropping Bombs Part 1** left off.



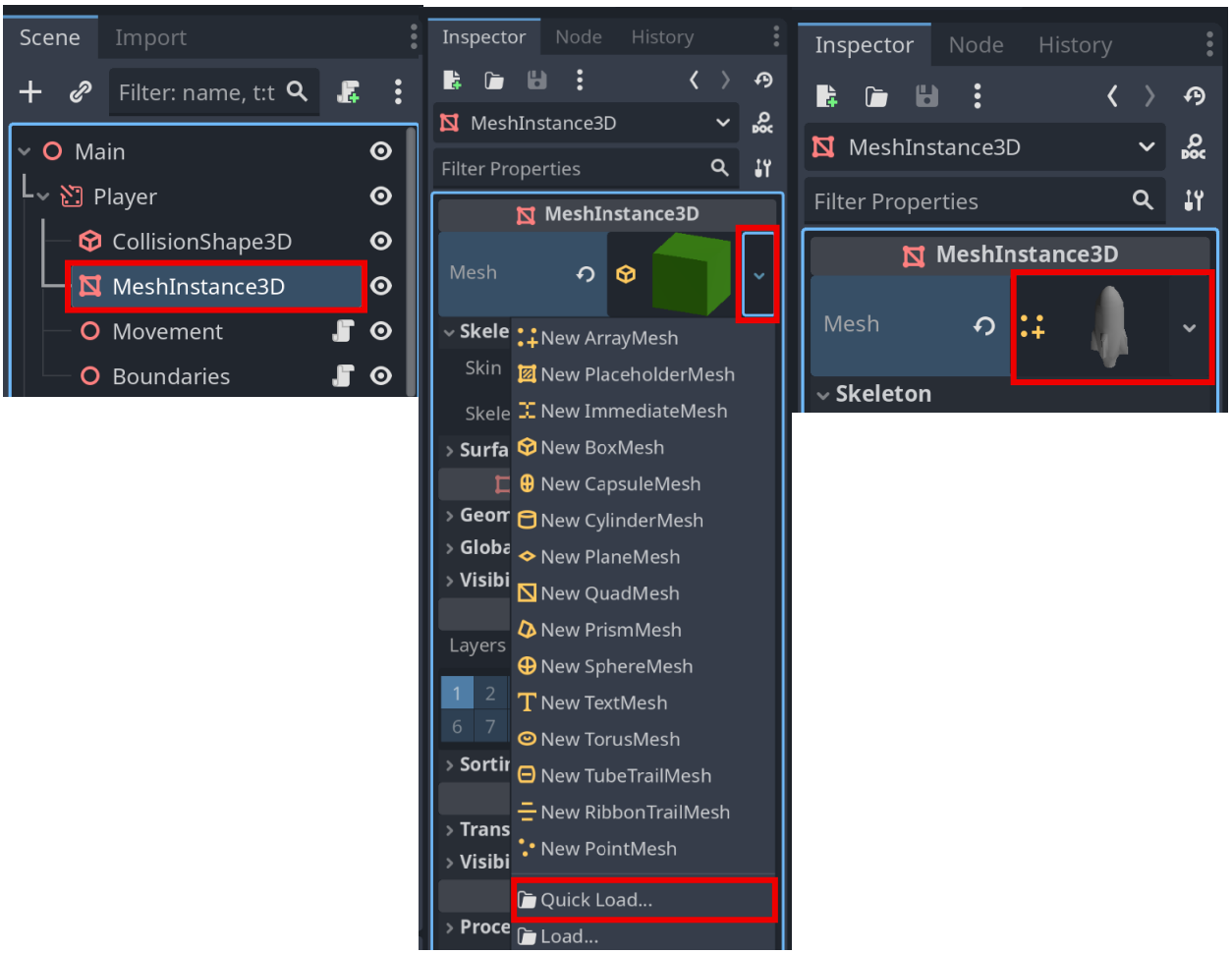
Pause for **Sensei Stop #1**!

Check with a Code Sensei before moving on. Make sure the **Starter Pack** and **main scene** were set up properly.

Reminder: Save your work!

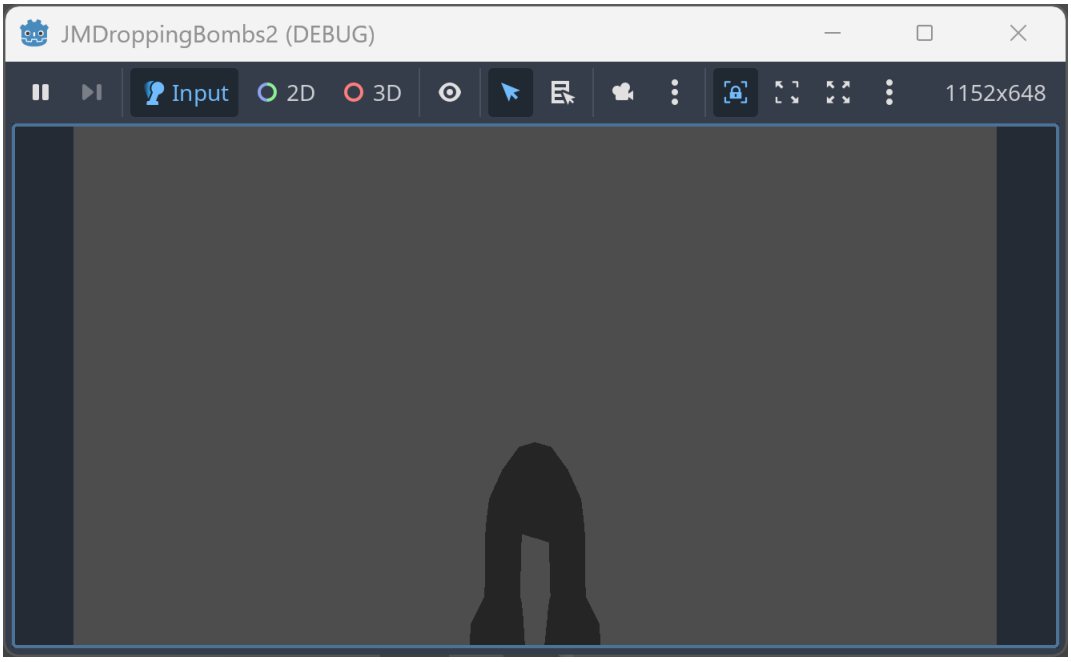
8 Find the **Player** node in **Scene** and open its **MeshInstance3D** child node in the **Inspector**.

The **Mesh** property is currently set to a plain cube. Select **quick load** from the drop-down menu next to the cube, then load the rocket mesh instead.



9 Playtest the game. Notice how the cube is replaced with the rocket mesh model instead!

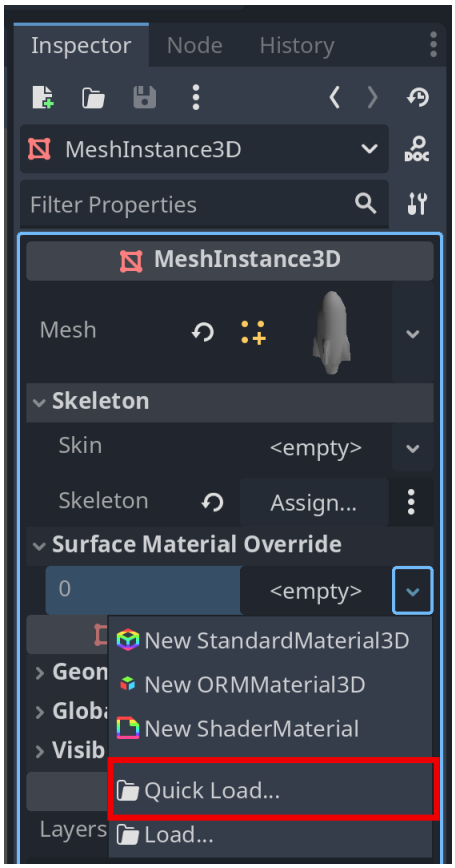
How do the bombs and the rocket interact?



10 In the **Inspector** of the **MeshInstance3D**, open the **Surface Material Override** menu.

Click the arrow to the right of **<empty>** and select **Quick Load**.

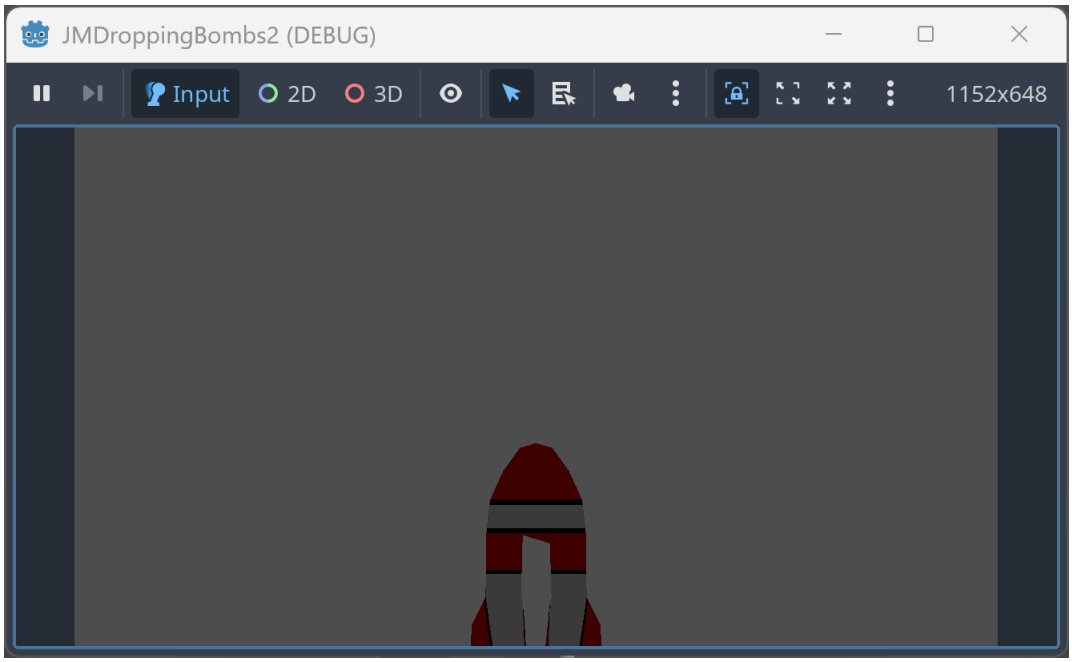
In the **Select Resource** window, select **RocketMAT.tres** to add the surface material to the Rocket.



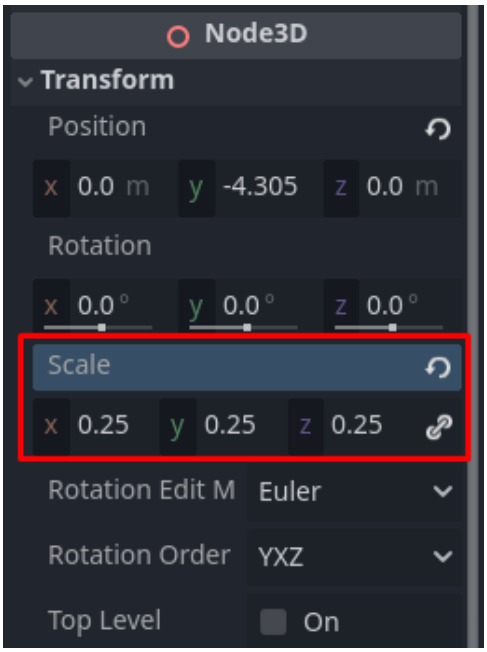
11 Playtest the game.

Notice that the rocket now has the colored material that was selected in the previous step.

Currently, the rocket is too big for the screen. This will get fixed in the next few steps.



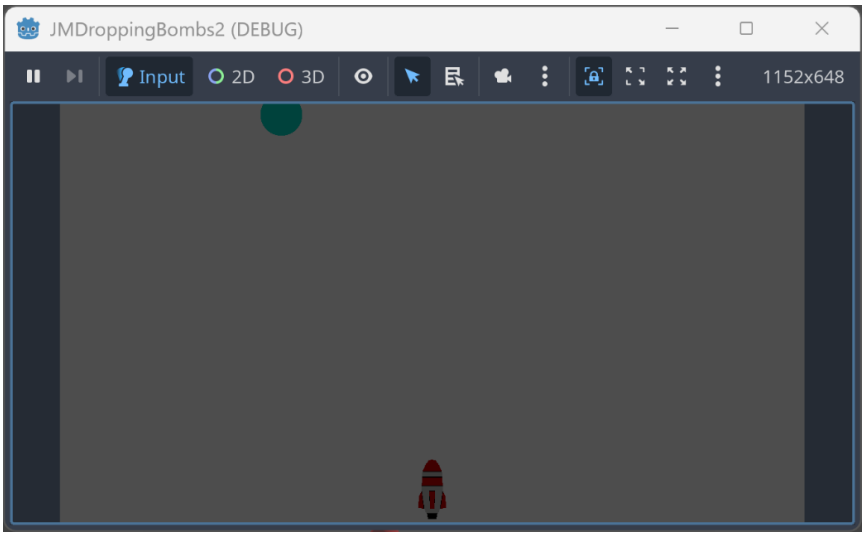
12 In **Scene**, select the **Player.** In the **Inspector,** update the Player's **Scale**. Set **x, y,** and **z** to **0.25**.



13 Playtest the game. Notice how the rocket's size has decreased, and the entire model is visible.

Tinker with the Player's **Scale** in the **Inspector.** What happens if a value larger than 1 is used? What happens if a negative value, less than 1, is used?

Change the **Scale** values back to **0.25** before continuing.

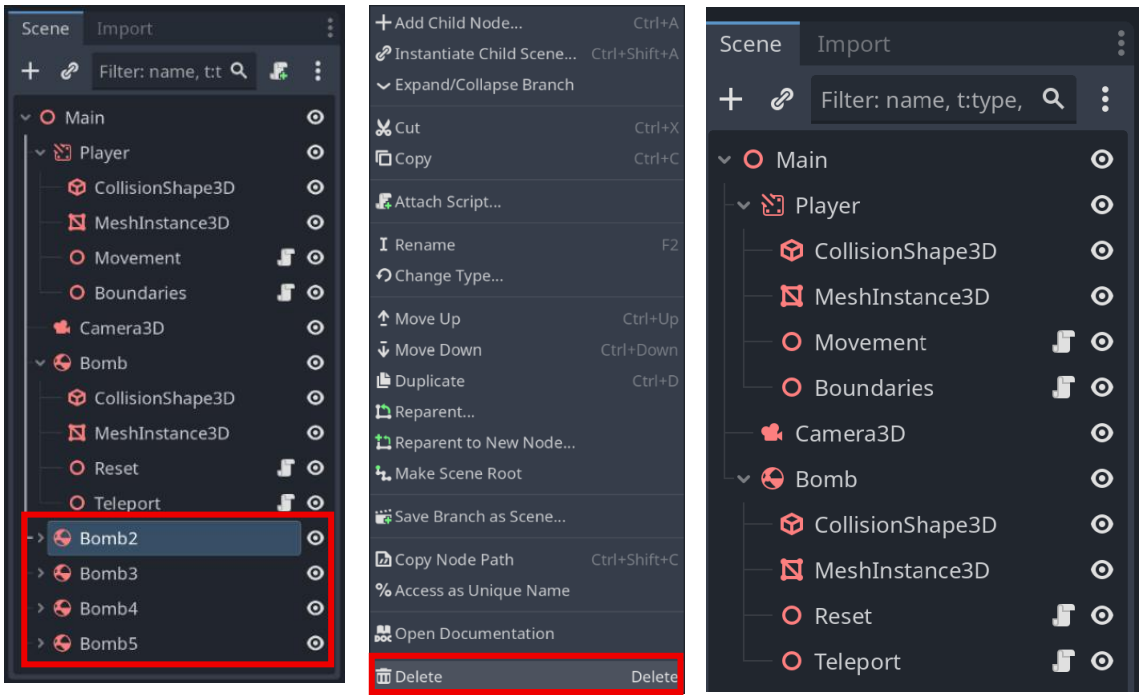


14

Update the Bomb's material!

To change every bomb individually would be time-consuming. Instead, delete all the copies and only make changes to the original. Copies of the new bomb will be added in a later step.
In **Scene**, delete the **Bomb2, Bomb3, Bomb4,** and **Bomb5** nodes so that only the original **Bomb** remains in the Scene menu.

Right-click on each node, then select **Delete** at the bottom of the pop-up menu to remove it.

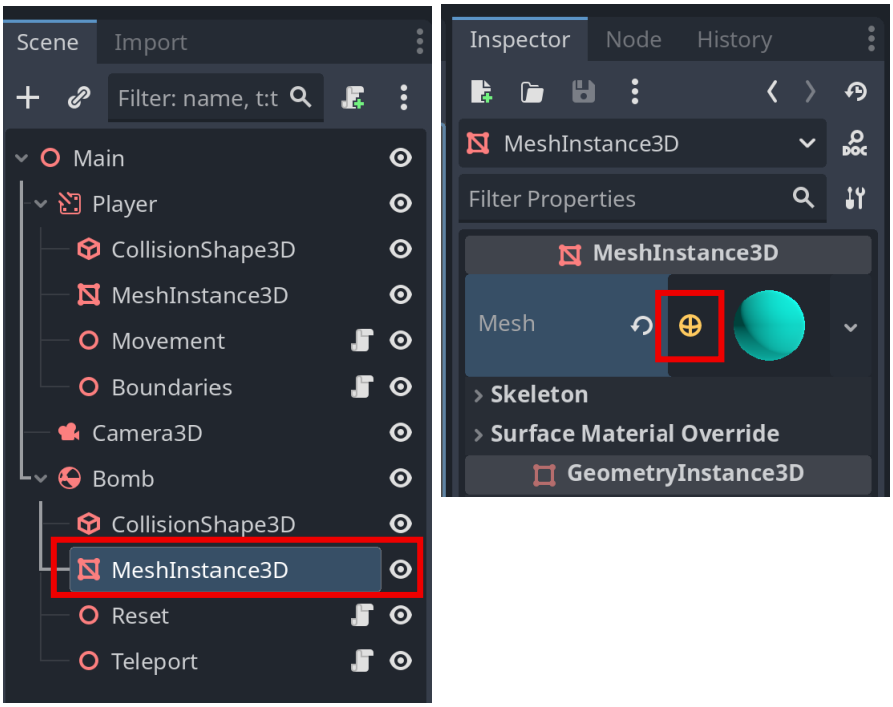


Pro Tip:

Click on a node and press the **Delete** button on the keyboard to get rid of it!

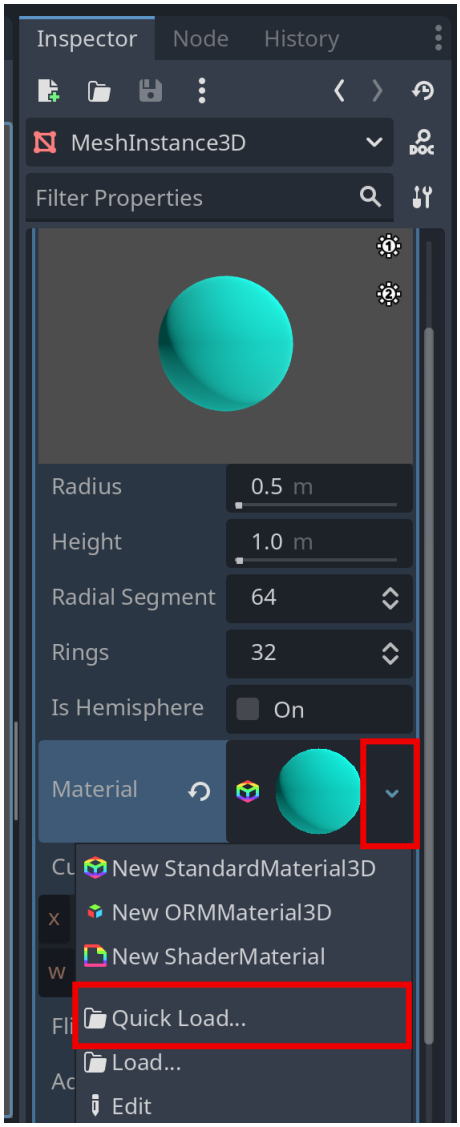
15 In **Scene**, select the Bomb's **MeshInstance3D** child node**.**

In the **Inspector,** click on the yellow icon next to the preview of the teal sphere mesh.



16 Click on the dropdown arrow to the right of **Material**. From the dropdown menu, select **Quick Load**.

Select **BombMAT.tres** to add the bomb texture to the **Bomb**.

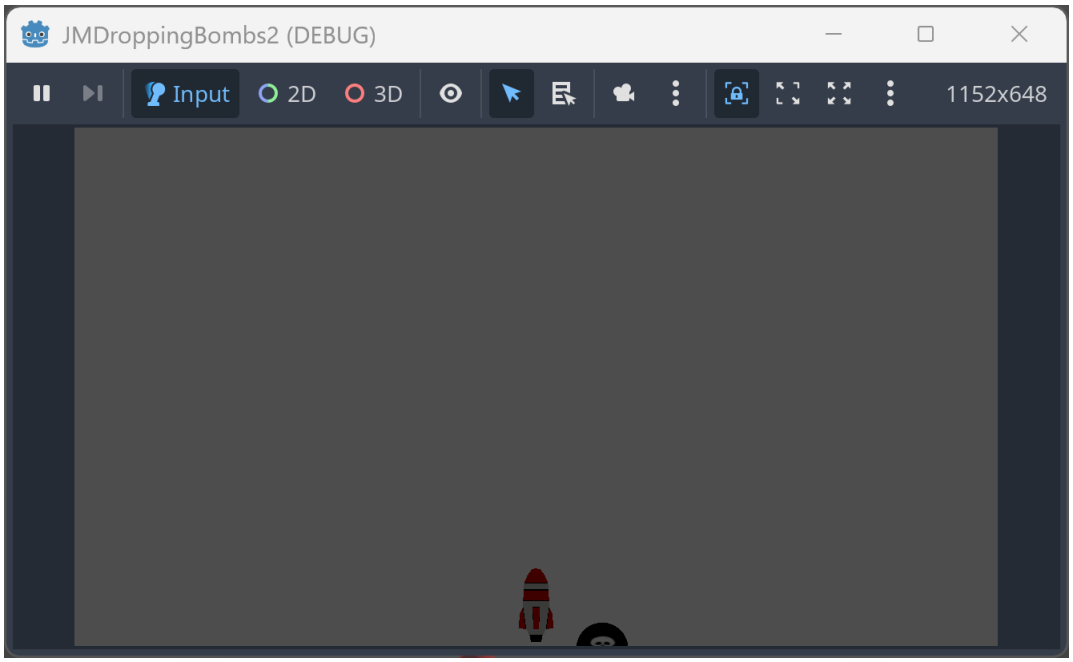


17 In **Scene**, select the **Bomb**. In **Inspector**, update the **rotation.** Set **y** to **180**.

18 Playtest the game!

The bomb should now have the texture applied and the skull face should be visible.

What happens if the **rotation** value are changed in the **Bomb** node?



Pause for **Sensei Stop #2**!

Check with a Code Sensei before moving on. Make sure the **Player** and **Bomb** materials are set up correctly.

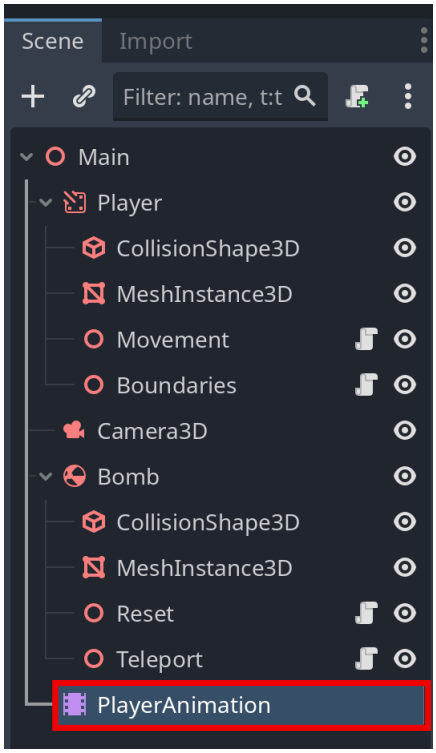
Reminder: Save your work!

19

Add animations to the **Rocket**.

In **Scene**, add an **AnimationPlayer** node as a child to the **Main root** node.

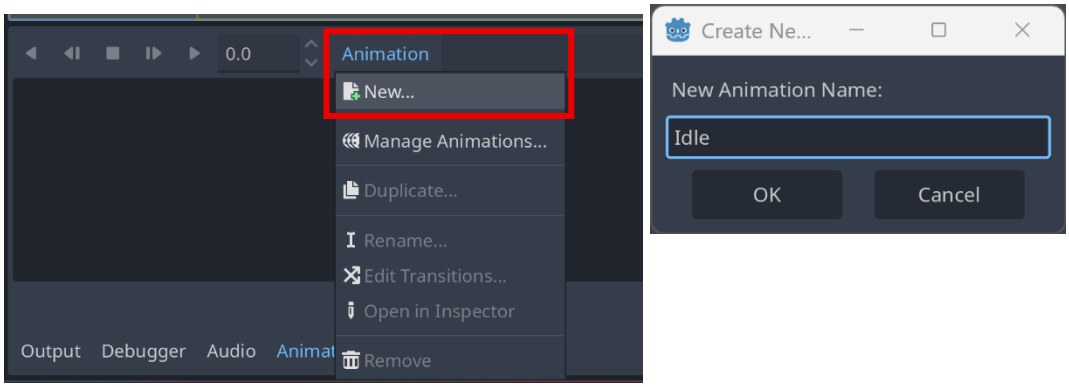
Rename the new node to **PlayerAnimation**.



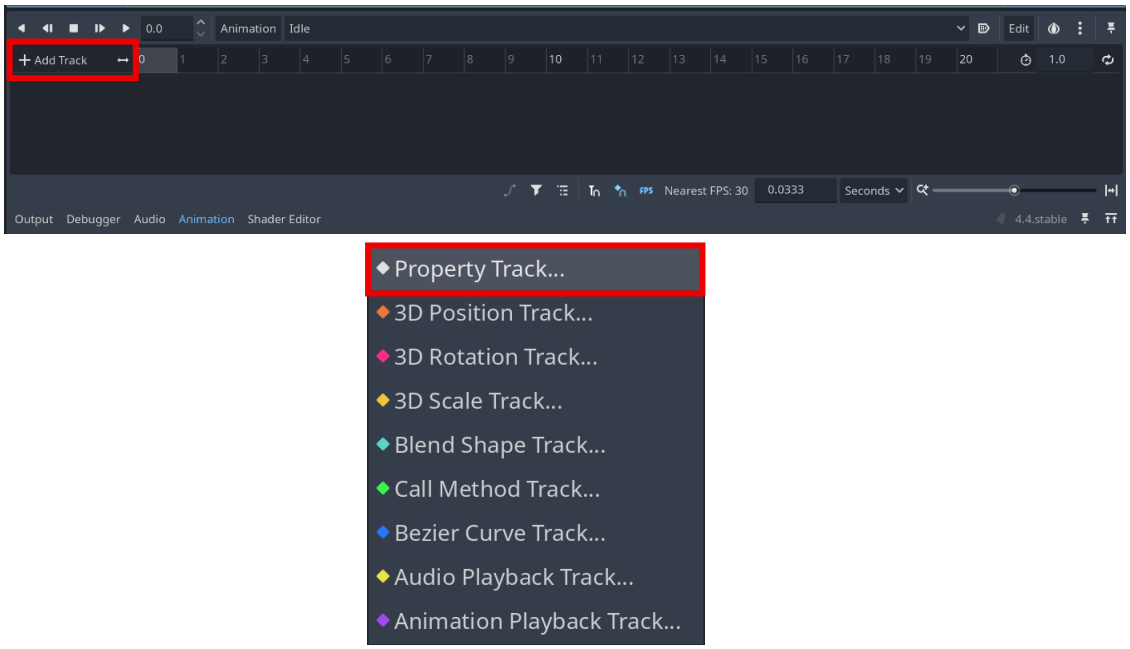
20 In **Scene,** select the **PlayerAnimation** node.

Find the animation panel at the bottom of the screen. Click on **Animation** to add a **New** animation.

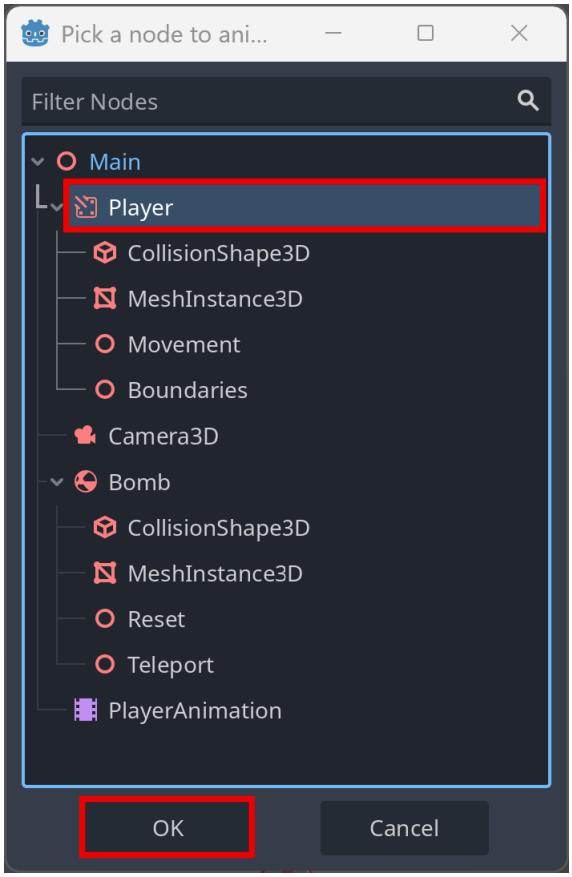
Name the new animation **Idle**. Click **Ok**.



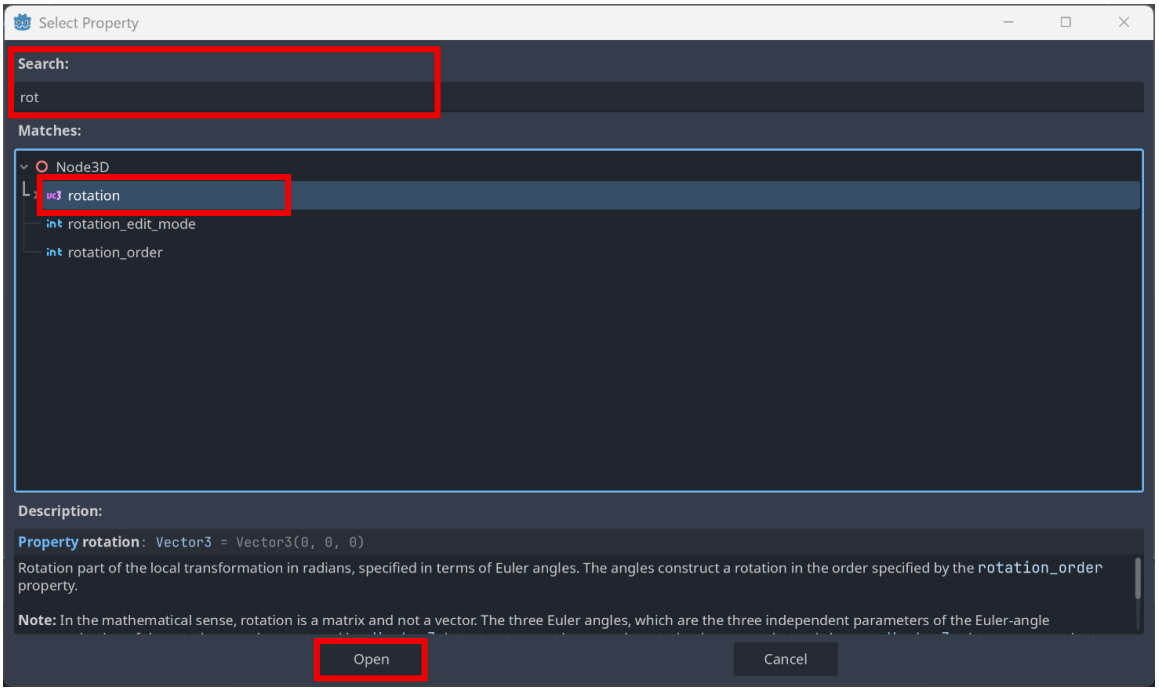
21 In the **Animation Bottom Panel**, click **Add Track**, then select **Property Track**.



22 Select **Player** and click **OK**.

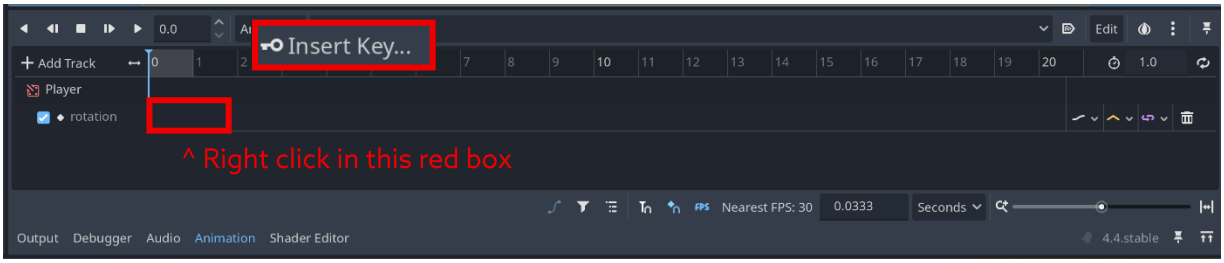


23 In the search bar, type **rotation**. Select the **rotation** node and click **Open.**

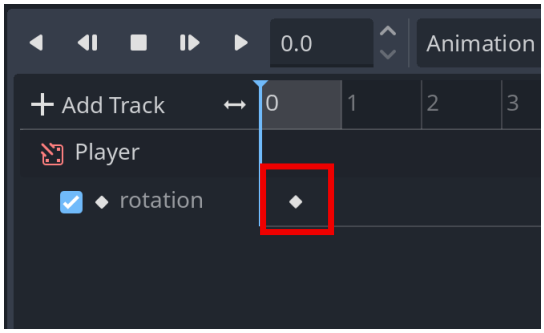


24 To the right of **rotation**, right click on the track.

From the dropdown menu, select **Insert Key**. This action will create a small, white diamond in the track.

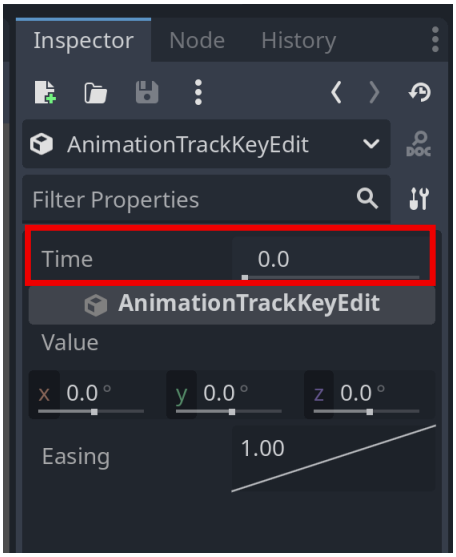


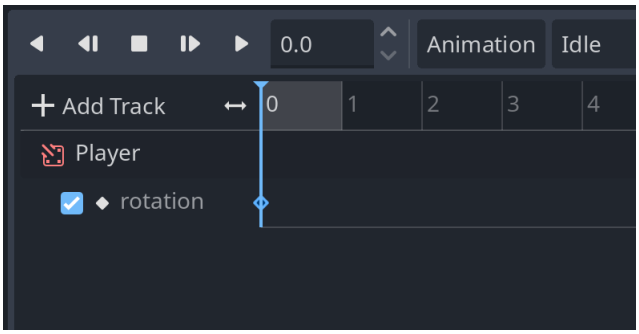
25 Click on the white diamond on the track to edit it in the **Inspector**.



26 In **Inspector**, set the **Time** property to **0**.

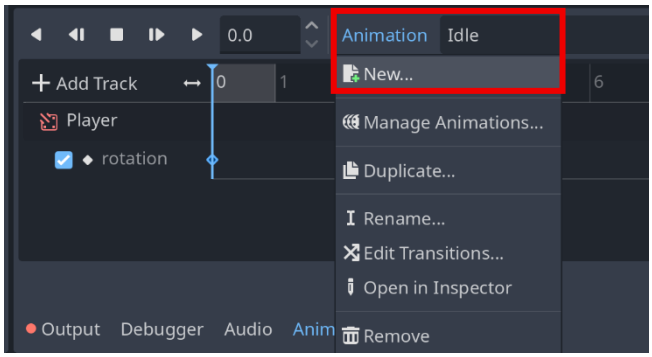
This will move the diamond to the very start of the track in the **Animation Bottom Panel**.

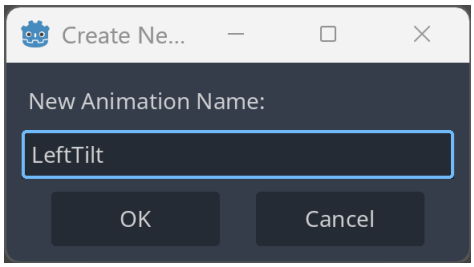




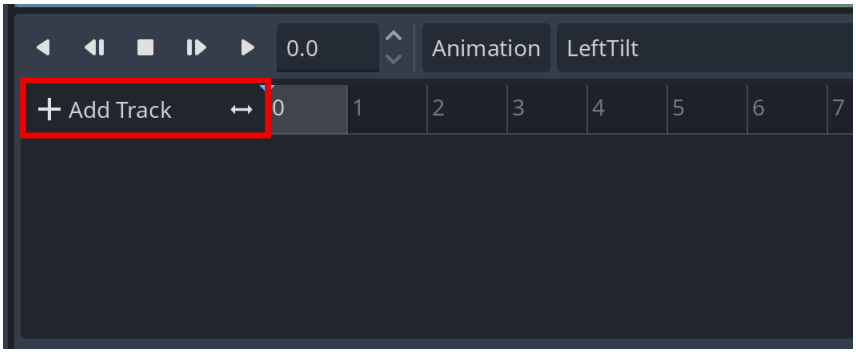
27 Create another animation by clicking **Animation** in the **Animation Bottom Panel**, then selecting **New**.

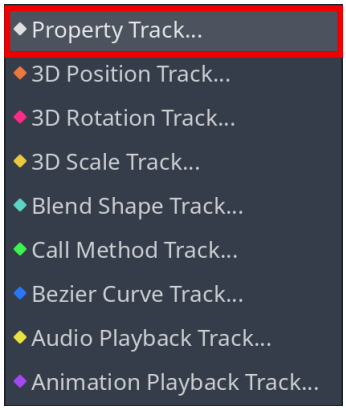
Name the new animation **LeftTilt** and select **OK**.



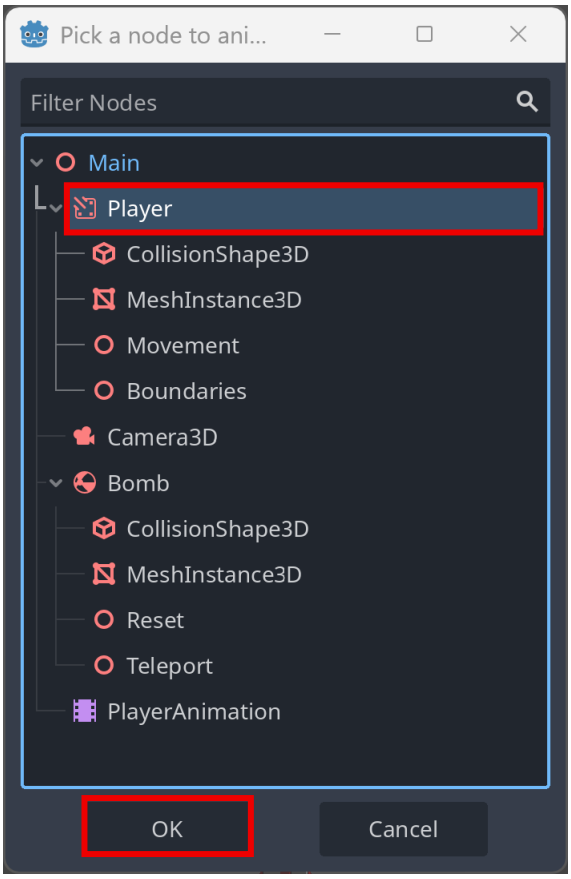


28 Click on **Add Track**, then select **Property Track**.

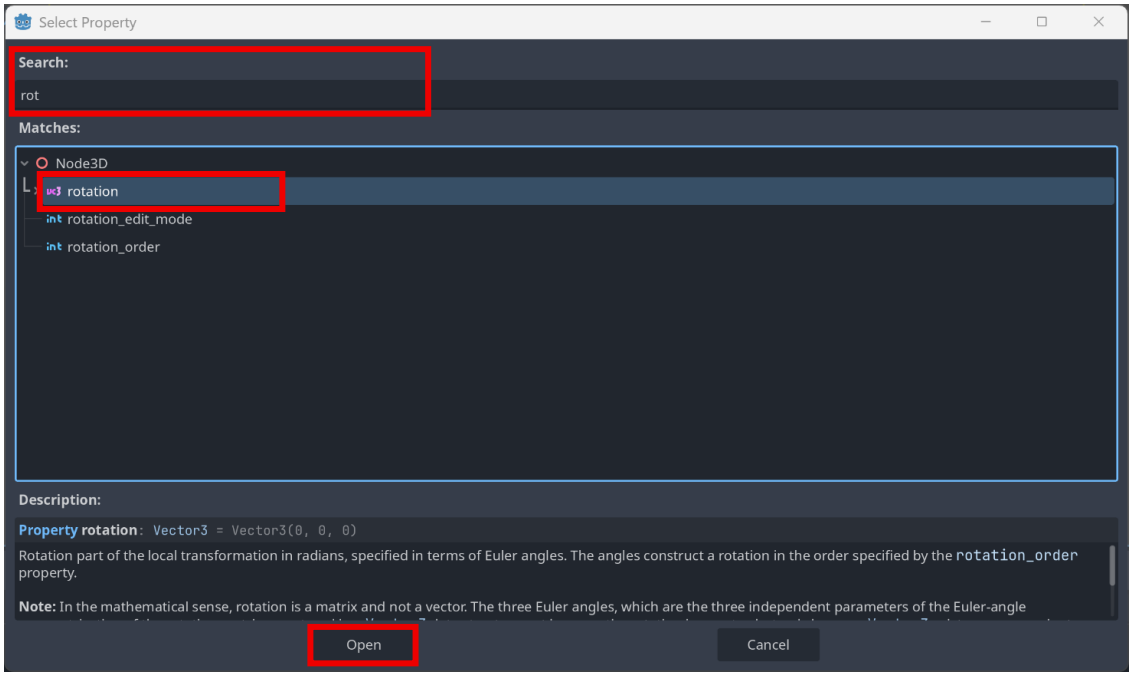




29 Select **Player**, then click **OK.**



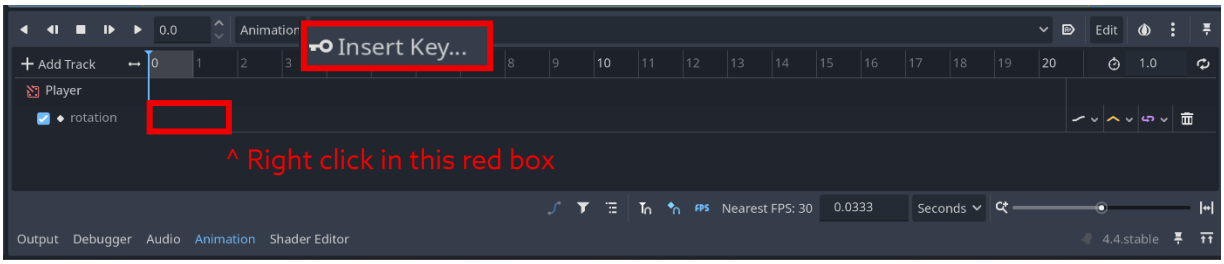
30 Search for and select the **rotation** node, then click **Open.**

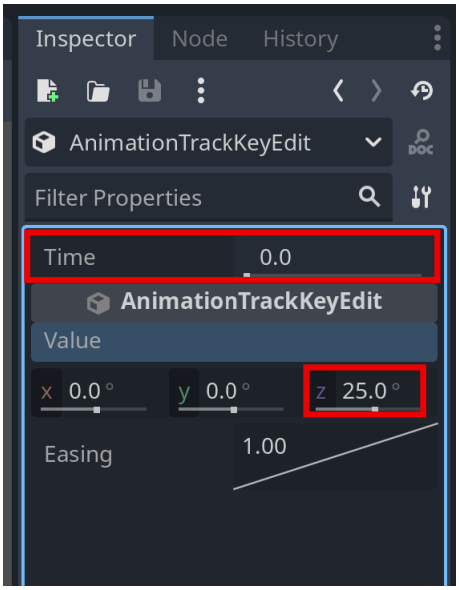


31 Right click on the track to the right of **rotation** and select **Insert Key** to create a small, white diamond in the track.

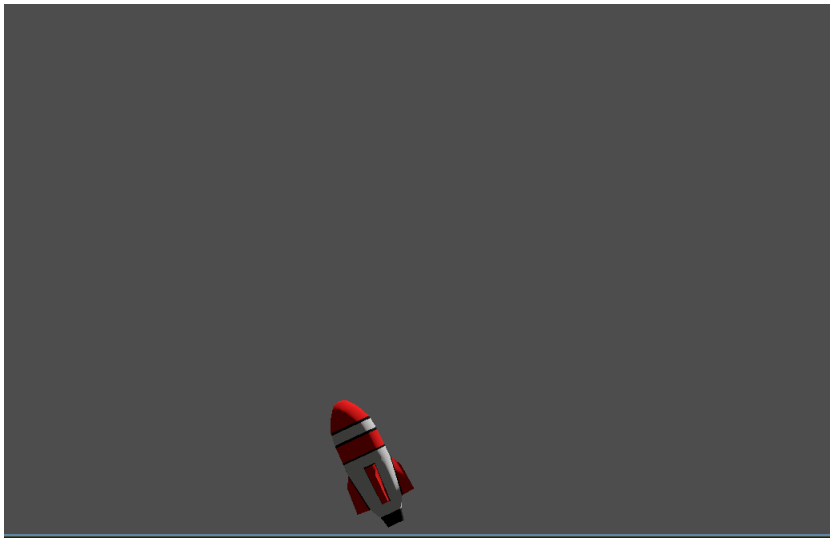
Click on the white diamond on the track to edit it in the **Inspector**.
In **Inspector**, set the **Time** property to **0**, and set the **z** value to **25**.

Refer to **Steps 22-24** for additional guidance.



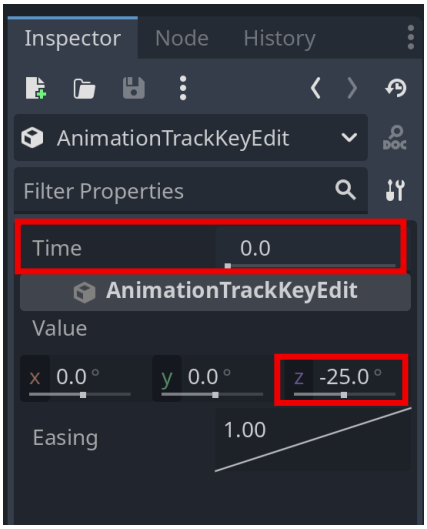


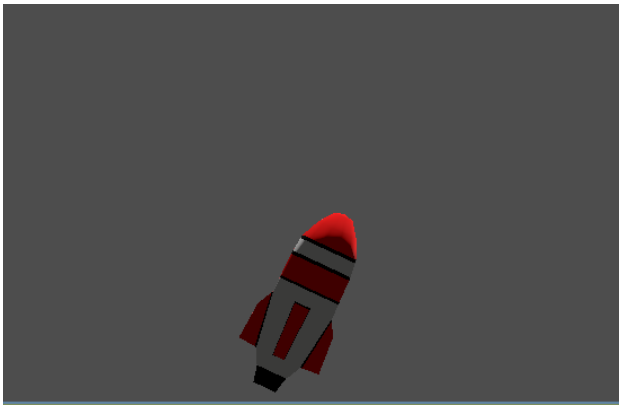
32 In the **Game Window**, the rocket should now be tilting left.



33 Create a new animation in the same way that the 2 previous animations were created in **Steps 25-30.**

Name this new animation **RightTilt.** In **Inspector,** set **Time** to **0** and **z** to **-25**.





Pro Tip:

Press **CTRL + Z** on the keyboard to undo your last action!

Pause for **Sensei Stop #3**!

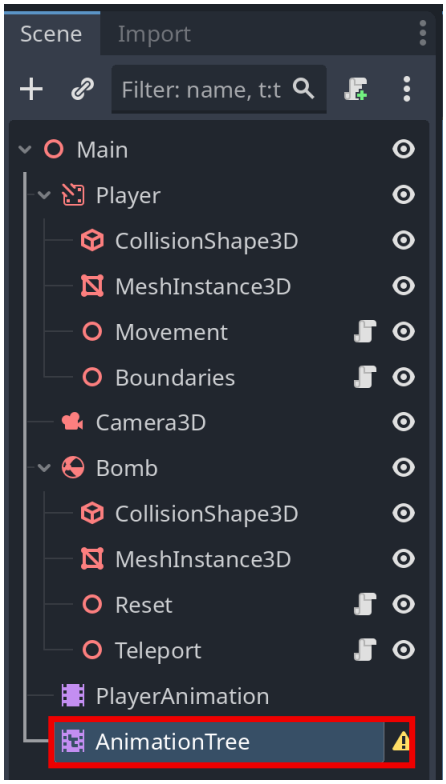
Check with a Code Sensei before moving on. Make sure the **Idle**, **LeftTilt**, and **RightTilt** animations were set up correctly.

Reminder: Save your work!

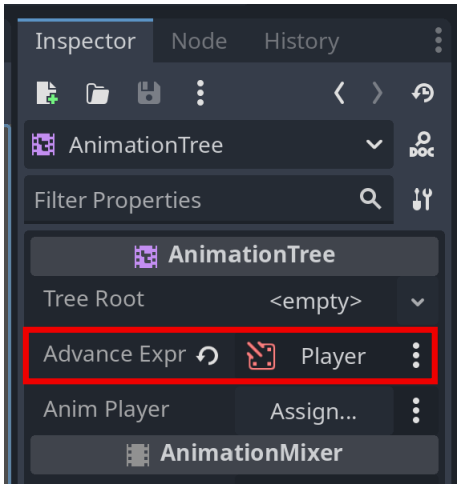
34

Connect the three animations to the **Rocket**.

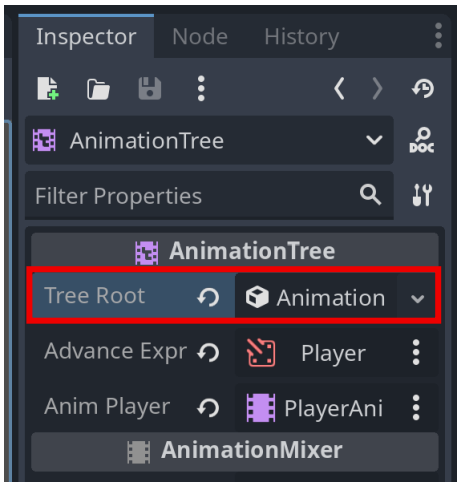
In **Scene**, add **AnimationTree** as a child node to the **Main root**.



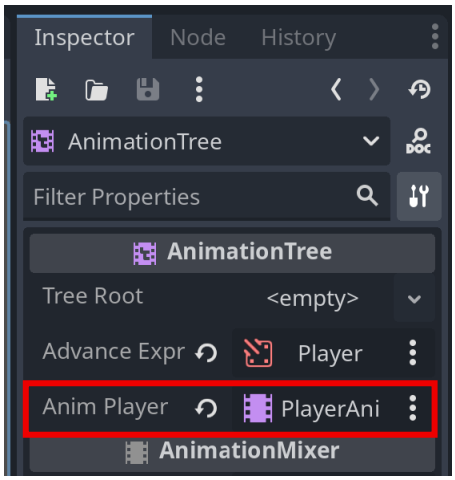
35 In the **Inspector** for **AnimationTree**, click the space to the right of **Advance Expression Base Node**. Select **Player** and click **OK**.



36 In **Inspector**, click <empty> next to **Tree Root**. Select **New AnimationNodeBlendTree**.



37 Underneath, click **Assign** to the right of **Anim Player**. Select **PlayerAnimation** and click **OK**.



38 Open the **AnimationTree** menu in the **Animation Bottom Panel**.

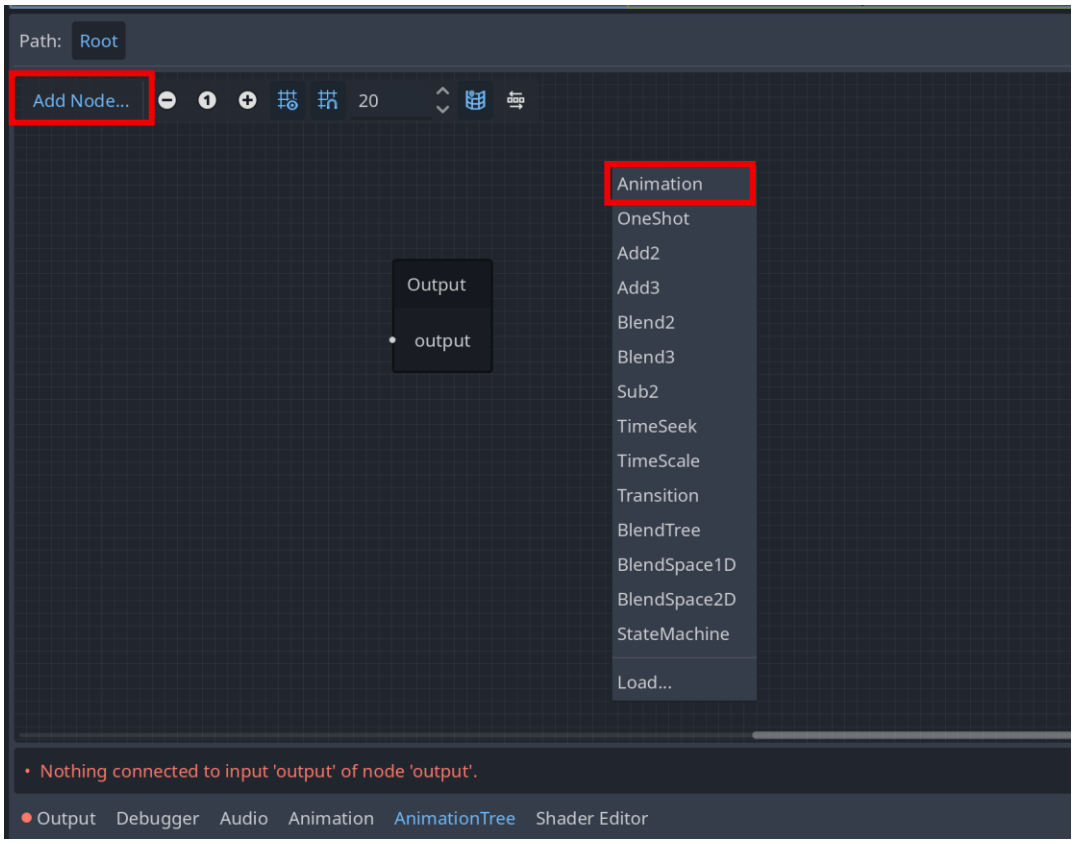
Ignore the warning for now. The warning is because the output animation node has yet to be set up. This will get fixed after following the next few steps.

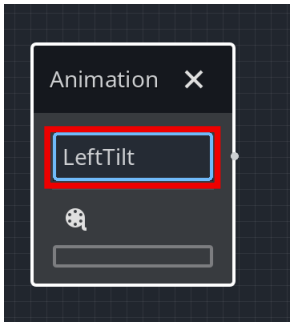
Pro Tip:

The window size can be adjusted, if necessary.

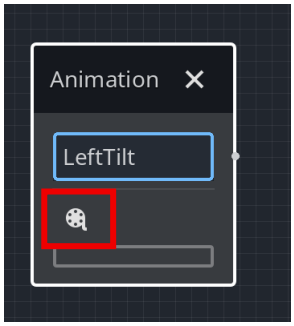
39 In the **AnimationTree** window, click **Add Node**. Select **Animation** from the dropdown menu.

Click on "**Animation**" in the new node and rename it **LeftTilt**.





40 In the **LeftTilt** node, click the film reel icon. Select **LeftTilt** from the dropdown menu.



Pro Tip:

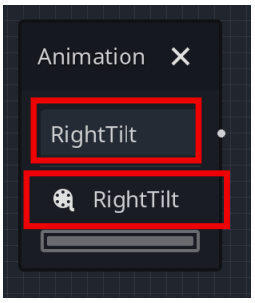
The animation nodes can be rearranged by dragging and dropping them so they do not overlap.

41 In the **AnimationTree** window, click **Add Node**. Select **Animation** from the dropdown menu.

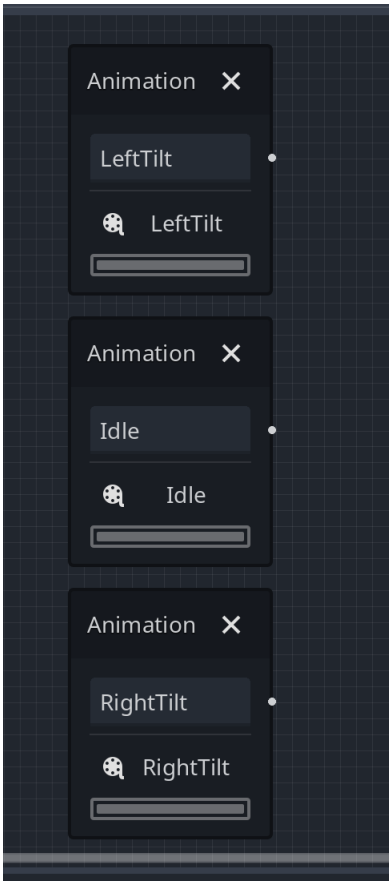
Rename the node to **Idle.**

Click the Film Reel icon. Select **Idle** from the dropdown menu.

42 Create a third animation for **RightTilt**. Refer back to **Steps 39-41** for additional guidance.



43 Drag and drop the nodes in the **AnimationTree** window so that they are organized like the image below. Make sure to match the order as well – **LeftTilt**, **Idle**, then **RightTilt**.

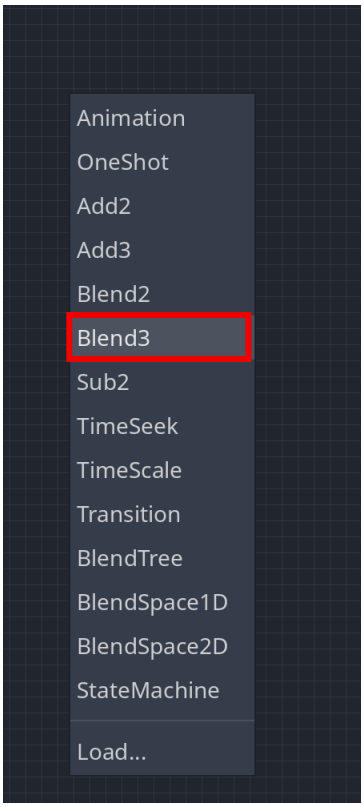


Pro Tip:

The window size can be adjusted, if necessary.

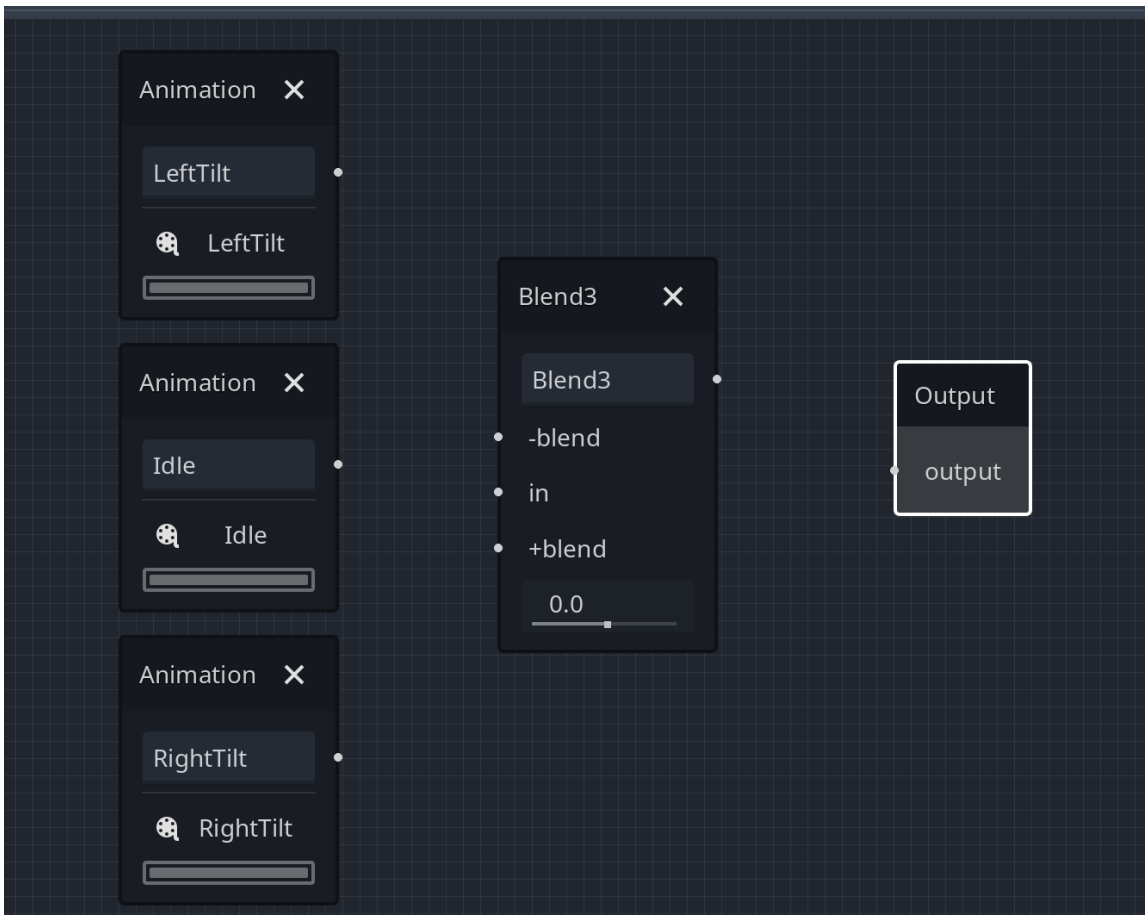
44 In the **Animation Tree** window, click **Add Node**.

Select **Blend3** from the dropdown menu.

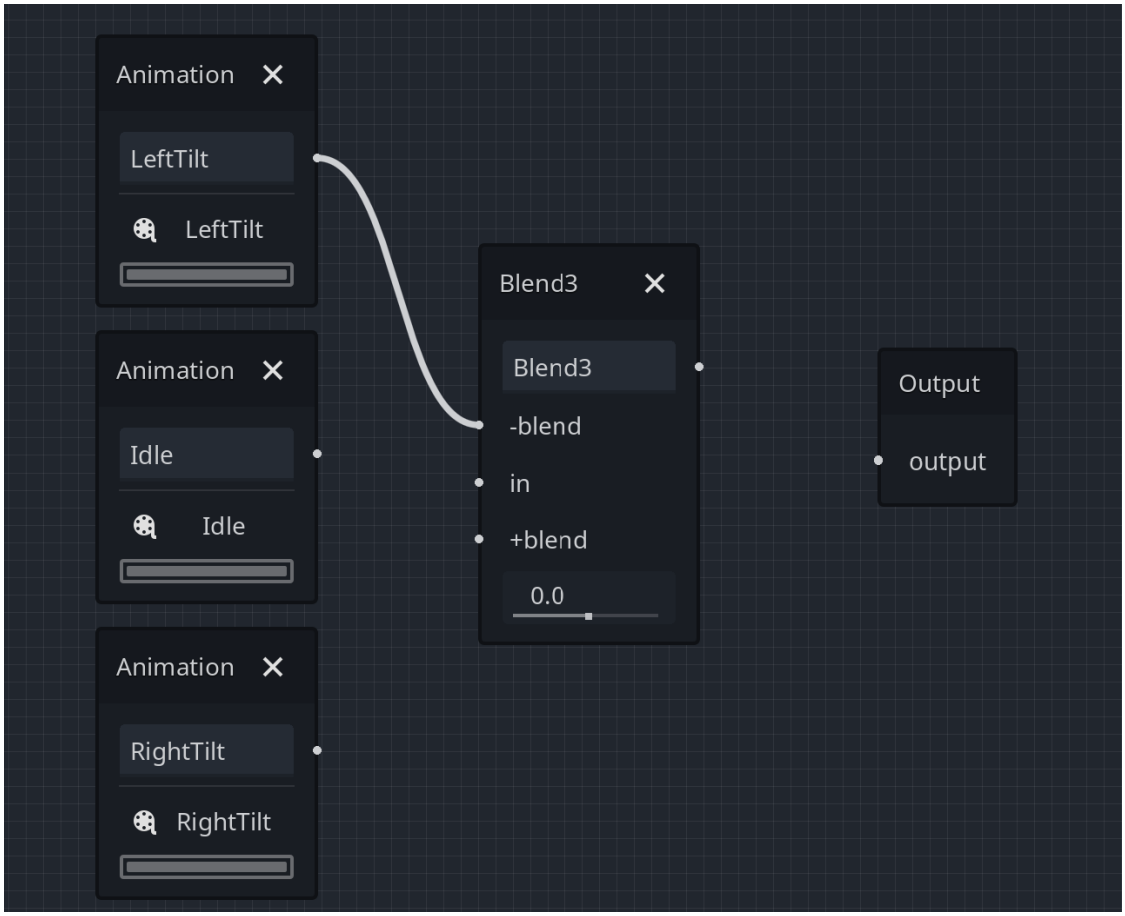


45 Drag and drop the nodes in the **AnimationTree** window so that they are organized like the image below.

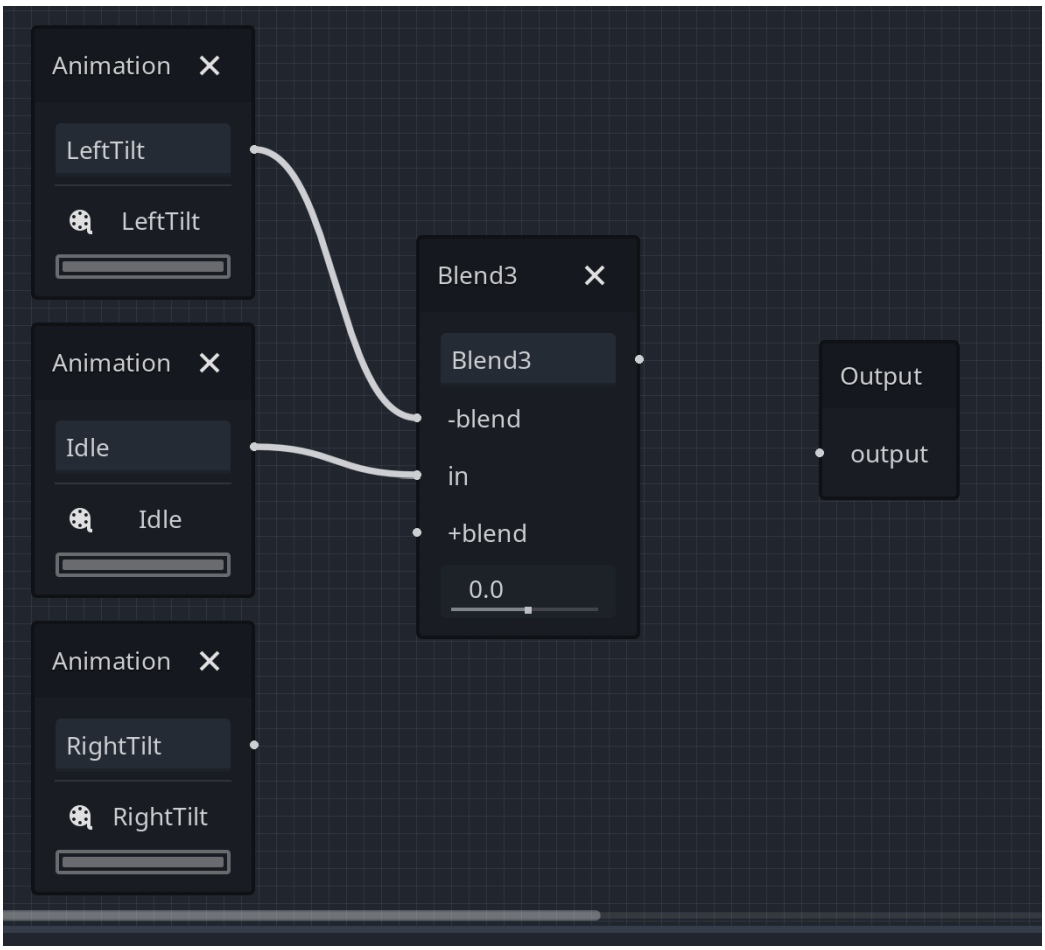
This represents the order in which Godot will use these animation nodes. The three **Input** Animation nodes will be combined into one animation, using the **Blend3**. The **Output** node will display the combined animation in the game.



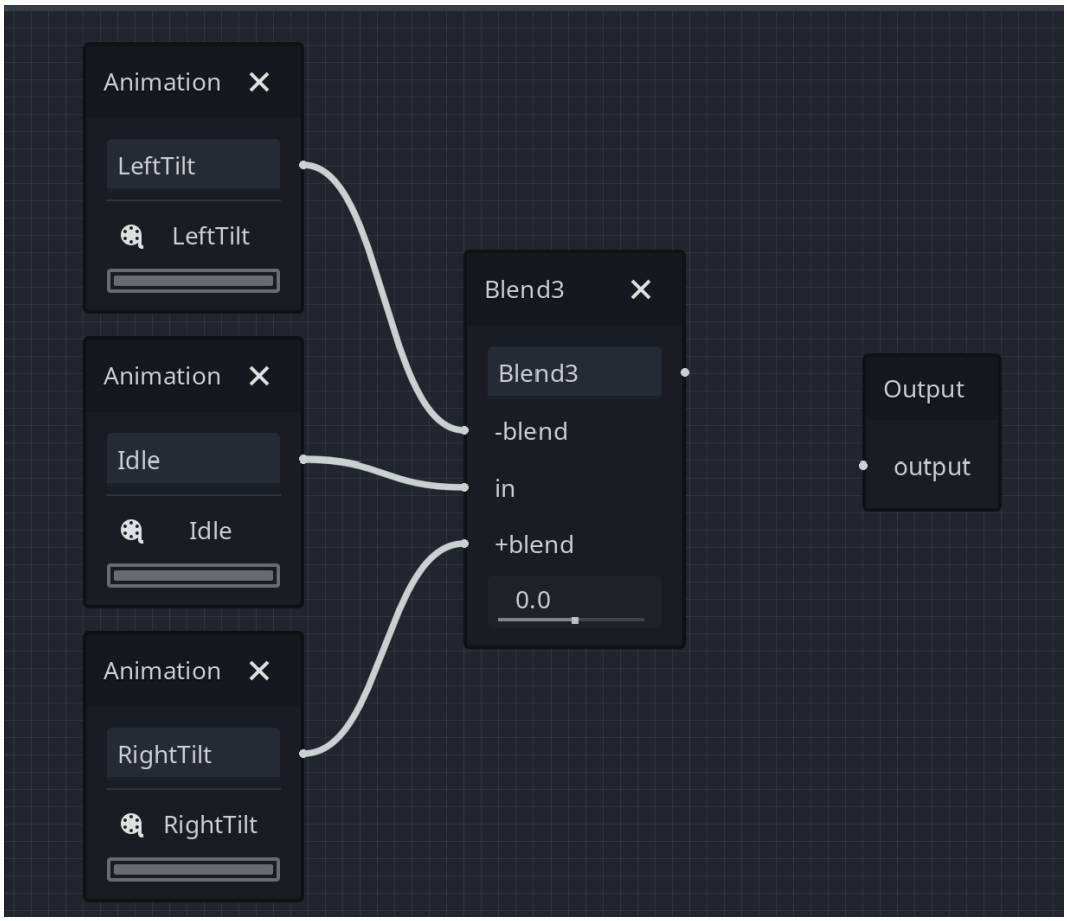
46 Attach the **LeftTilt** node to the top input of **Blend3** by dragging from the output dot of **LeftTilt** to the input dot labelled **-blend**.



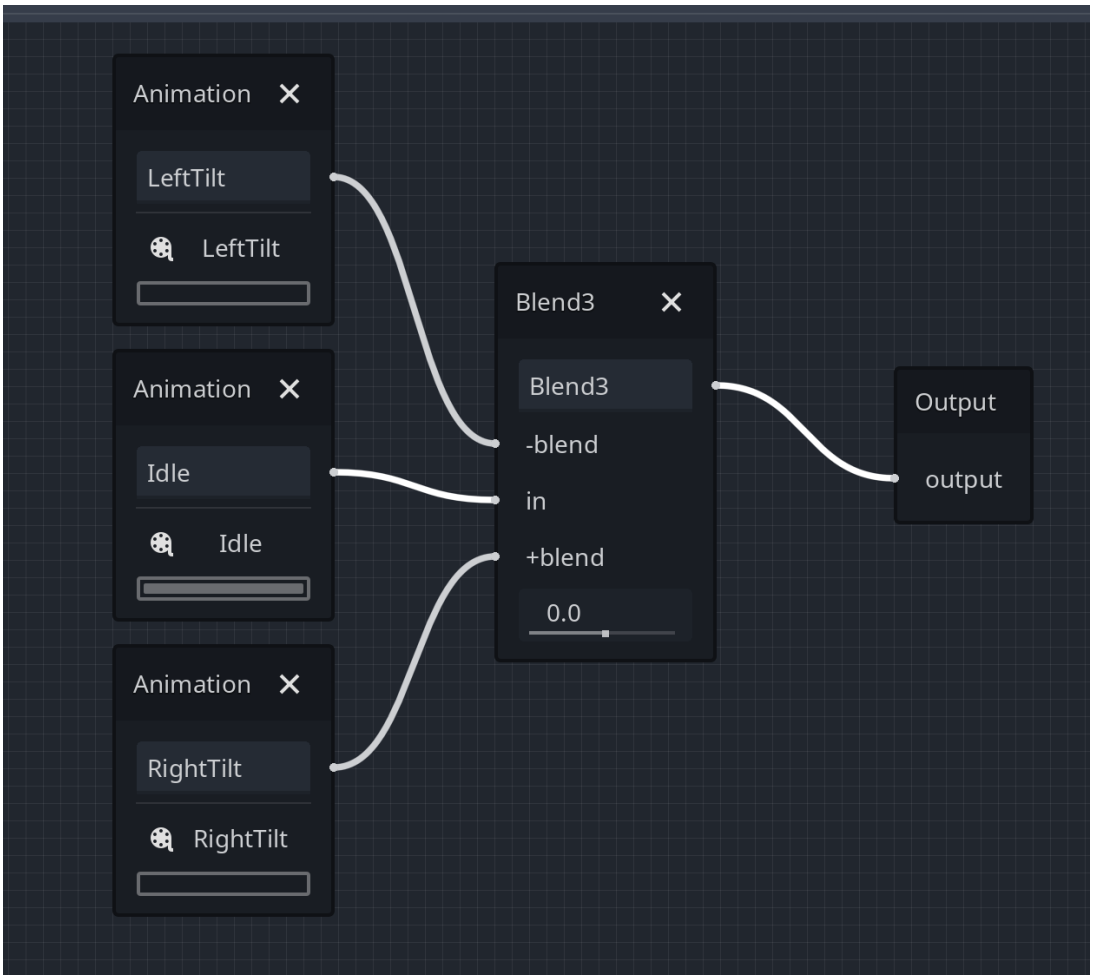
47 Attach the **Idle** node to the middle input of **Blend3** by dragging from the output dot of **Idle** to the input dot labelled **in**.



48 Attach the **RightTilt** node to the bottom input of **Blend3** by dragging from the output dot of **RightTilt** to the input dot labelled **+blend**.

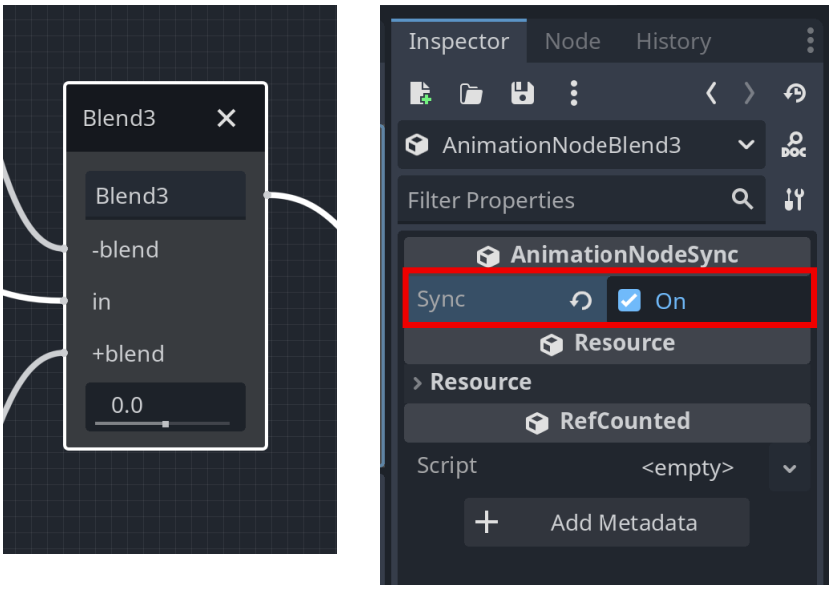


49 Attach the output of the **Blend3** node to the input of the **Output** node by dragging from the output dot labelled **Blend3** into the input dot labelled **output**.

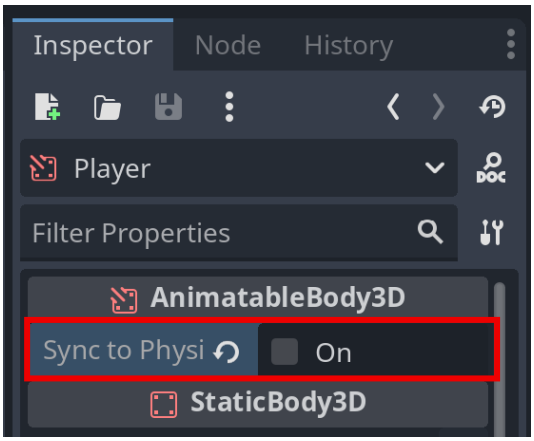


50 Click on the **Blend3** node to open it in the **Inspector**.

Make sure the **Sync** property is **Toggled On**.



51 In **Scene**, select the **Player** node. In Player's **Inspector**, find the **Sync to Physics** property at the top. Make sure it is **Toggled Off.**



52

In **FileSystem**, find **movement.gd** in the **Scripts** folder. Double click to open the script in the editor.

Notice the code that's already present in the movement script. This script sets whether the rocket's movement is vertical or horizontal, then calculates the speed at which the rocket moves.

Find the **TODO STEP 52** comment in the script. On the next line, use the `@onready` keyword to declare a new **AnimTree** variable, of type **AnimationTree**.

Assign the variable the value `: $"../../AnimationTree"`.
This will allow the script to reference the **AnimationTree** object and connect its animation. `"../../"` refers to the node path, where the **AnimationTree** is in the Scene tree.

```
13  # --------------------
14  # TODO STEP 52
15  # Add a reference to the AnimationTree node.
16  # --------------------
17  @onready var AnimTree: AnimationTree = $"../../AnimationTree"
```

53

Scroll to the bottom of the script to find the **TODO STEP 53** comment. On the next line, set the `blend_amount` parameter of the `AnimTree` variable to `horizontal`.

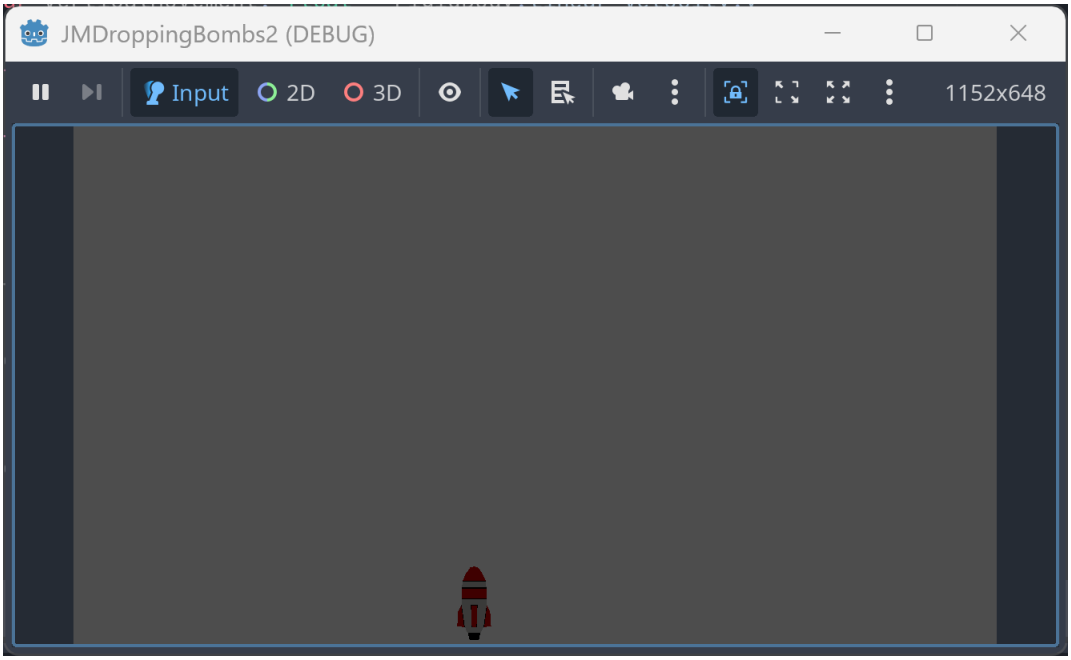
This will set the `blend_amount` to match the horizontal speed of the rocket. This allows the rocket to tilt in the direction the rocket moves.

```
71      # --------------------
72      # TODO STEP 53
73      # Set the direction the rocket will tilt in.
74      # --------------------
75      AnimTree["parameters/Blend3/blend_amount"] = horizontal
```

54 Playtest the game.

Notice the rocket's new animation; it should tilt left and right as it moves, then return to an upright position when still.

What might happen if the node connections in the **AnimationTree** panel were reversed?



Pause for **Sensei Stop #4**!

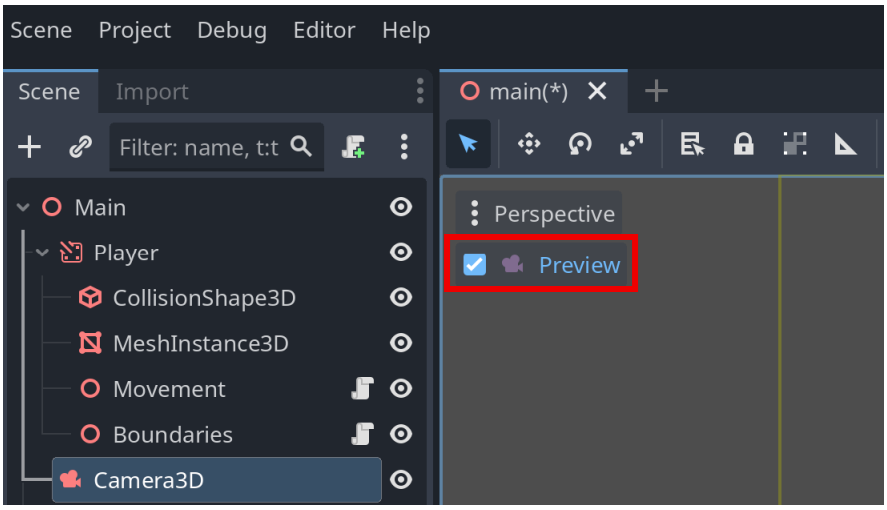
Check with a Code Sensei before moving on. Make sure all the animation nodes have been created and connected properly. Great work so far!

Reminder: Save your work!

55 Select the **3D** workspace button at the top of the editor.

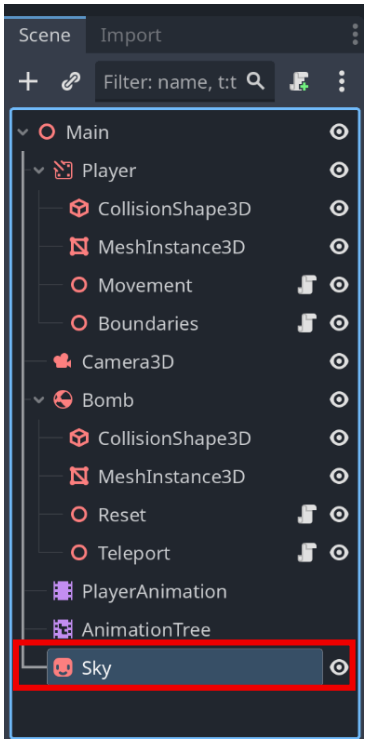
In **Scene**, select the **Camera3D** node.

In the top left of the **Viewport,** ensure that **Preview** is **Toggled On**.



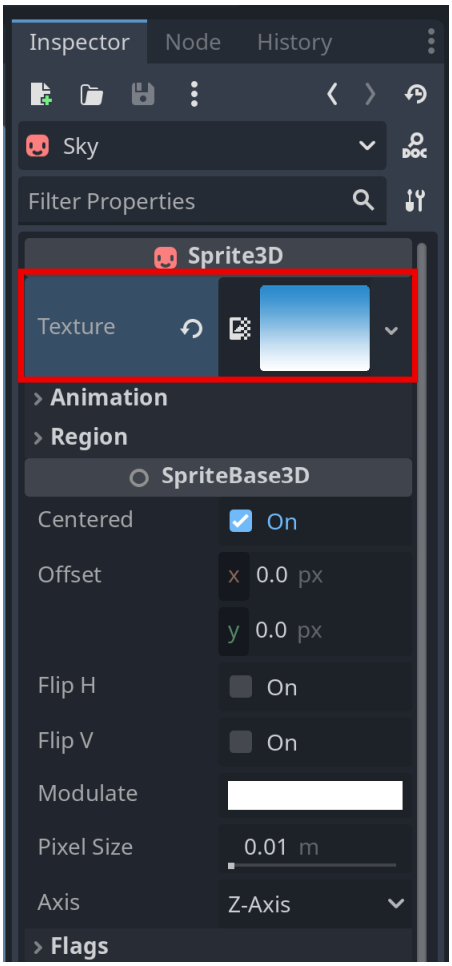
56 In **Scene**, create a new **sprite3D** node as a child to the **Main root**.

Rename the node to **Sky**.



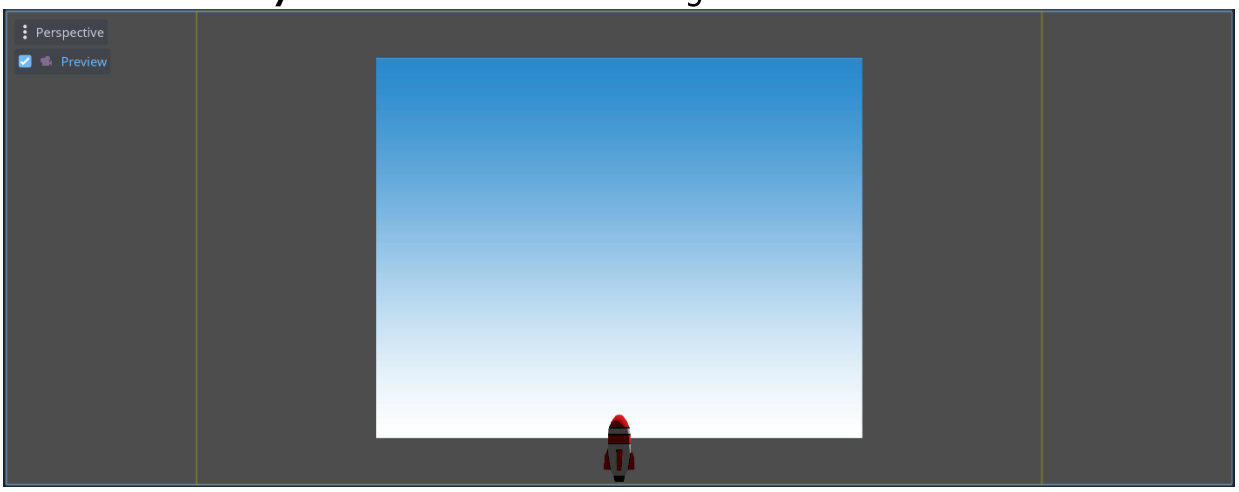
57 In **Inspector**, locate the **Texture** property and click **<empty>**.

Select **Quick Load** from the dropdown menu, then choose the **Activity 8 – Sky Gradient.png** texture.



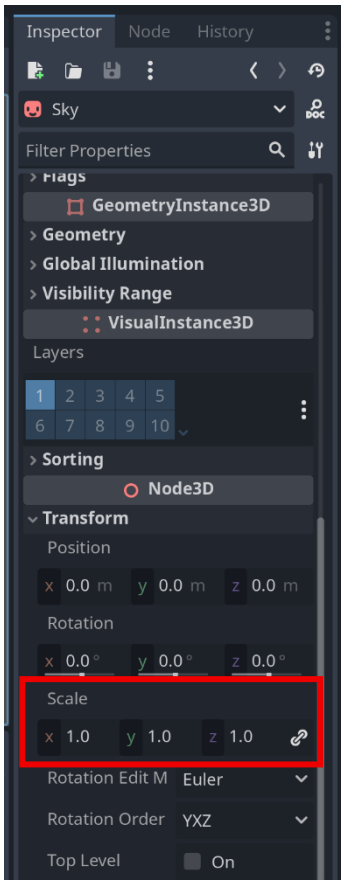
58 Playtest the game.

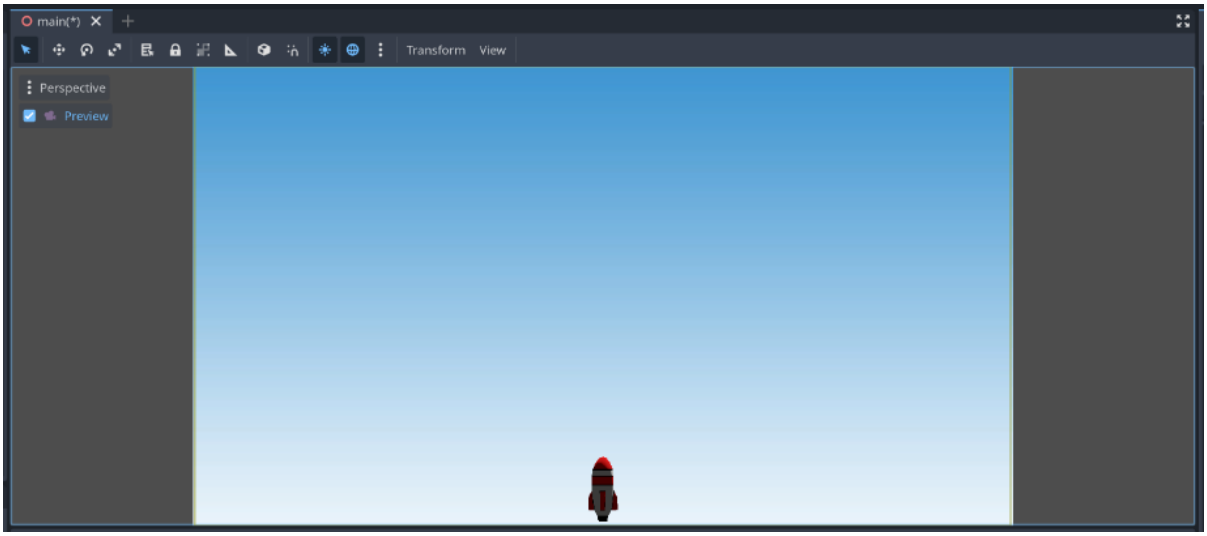
Notice that the **Sky** does *not* cover the whole game window.



59 In **Inspector**, in the **Transform** section, change the **Scale** to **1.75**. Changing **x, y,** or **z** will update all the values.

Notice that this resizes the background instantly.

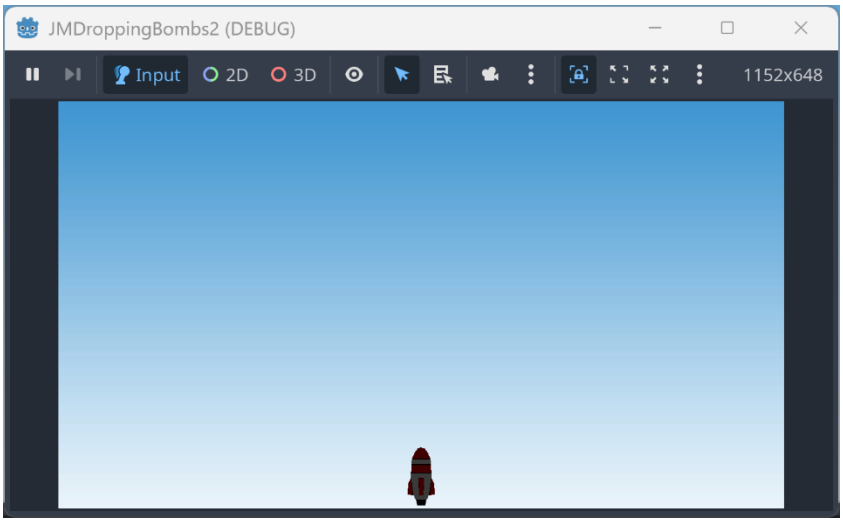




60 Playtest the game again.

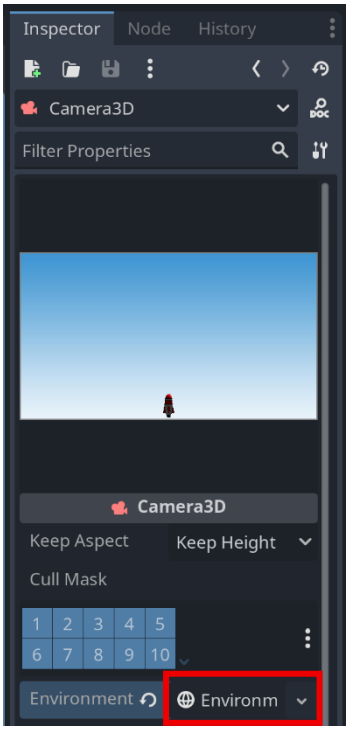
Now, the **Sky** covers the whole screen!

What happens if the **Scale** is set to a value less than 1?

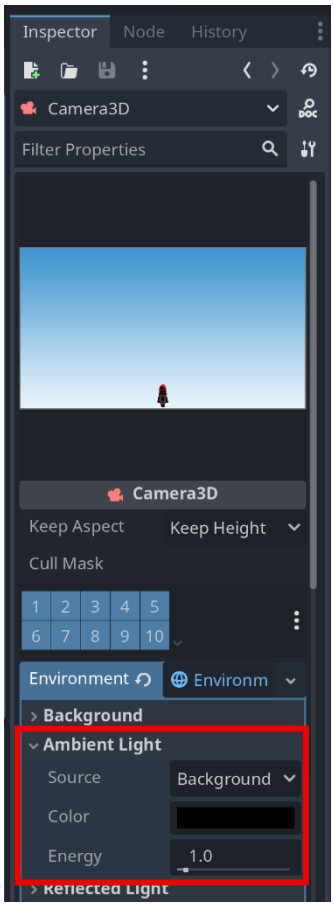


61 In **Scene**, select the **Camera3D** node.

In **Inspector**, locate the **Environment** property and click the **Globe Symbol**.

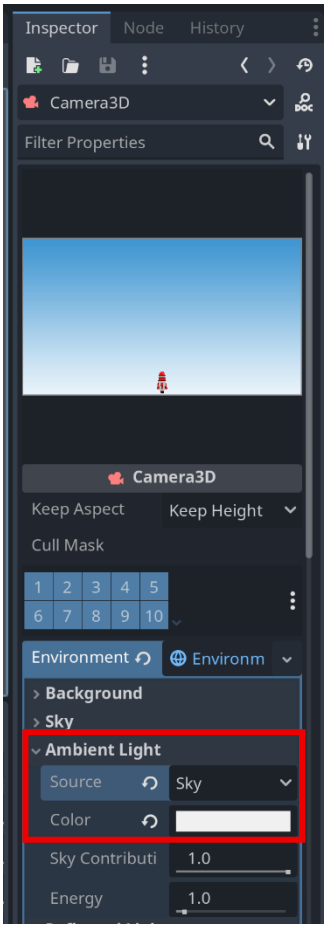


62 Select **Ambient Light** from the dropdown menu.



63 Locate the **Source** property and set it to **Sky** from the dropdown menu.

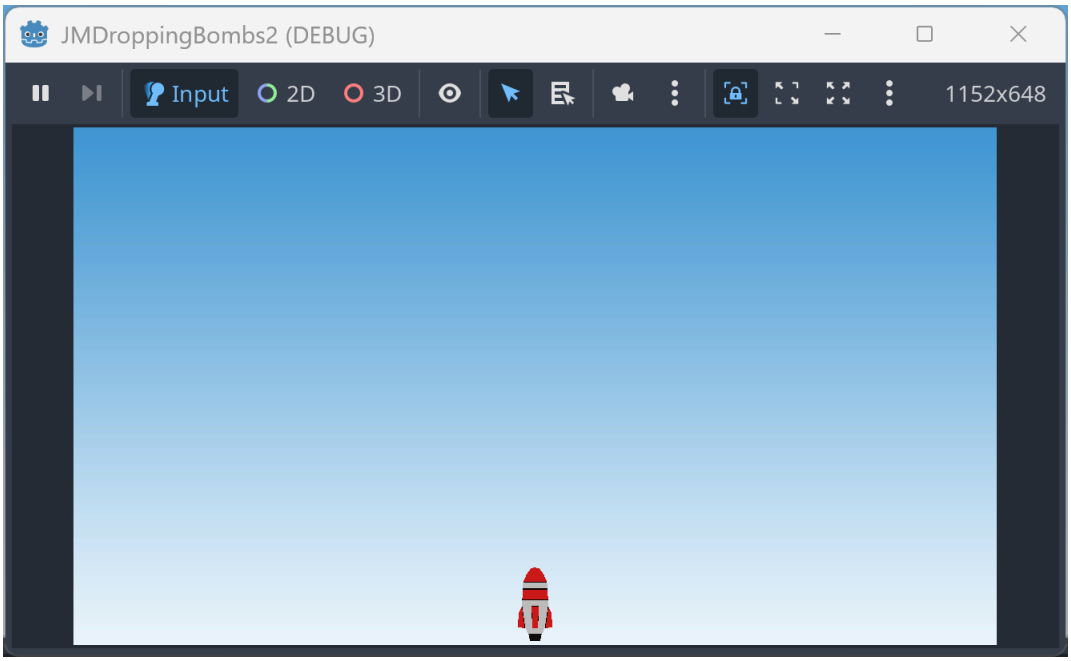
Underneath, open the **Color** property. Type in **White** or **FFFFFF** as the **Hex** property.



64 Playtest the game.

The rocket should now be fully lit up and colorful.

What happens if the **Hex** color gets changed under **Ambient Light?**



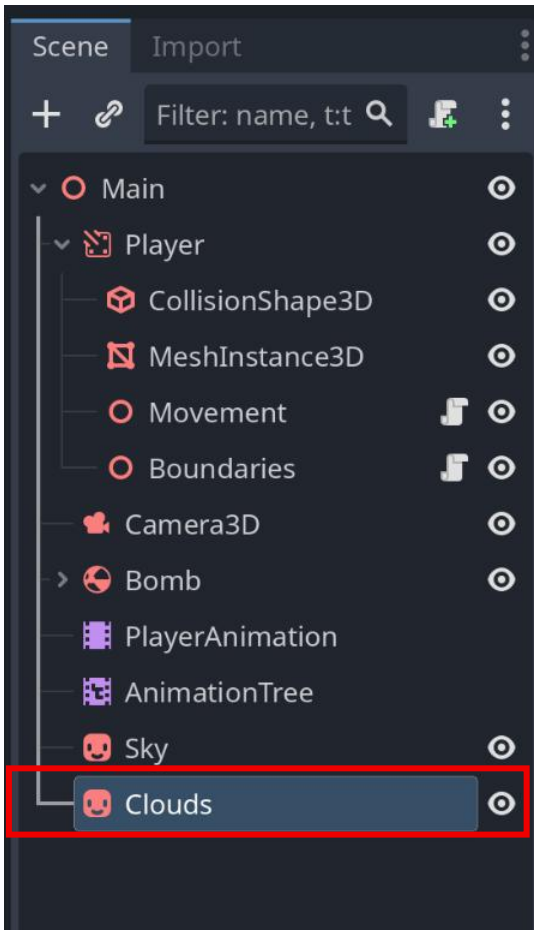
Pause for **Sensei Stop #5**!

Check with a Code Sensei before moving on. Make sure the sky background is set up and covers the whole screen. After this, the final animations will be added for this part of the project!

Reminder: Save your work!

65 Add clouds to the background scene!

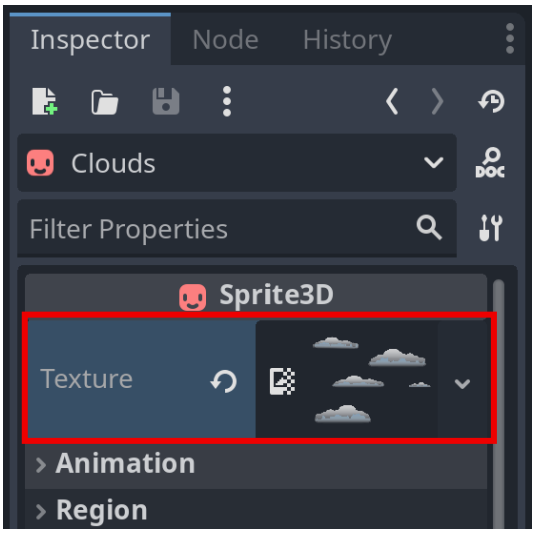
In **Scene**, create a new **Sprite3D** node as a child to **Main root**. Rename it **Clouds**.

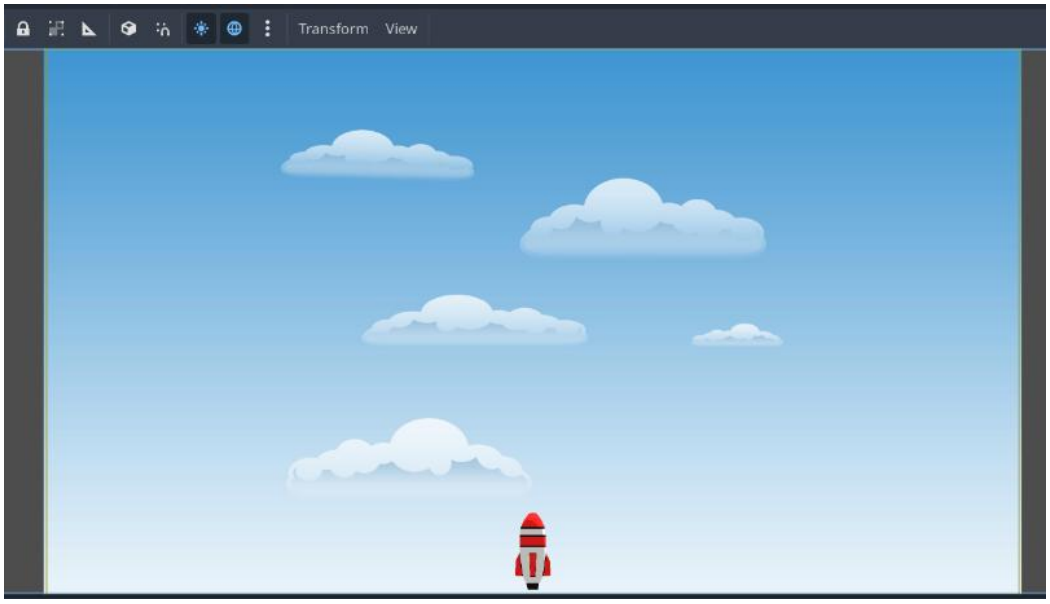


66 In the **Inspector** of the **Clouds** node, locate the **Texture** property and click **<empty>.**

Select **Quick Load** from the dropdown menu.

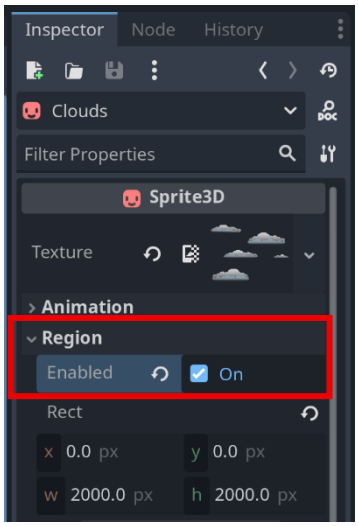
Select the **BB Activity 08 – clouds.png** texture from the window.

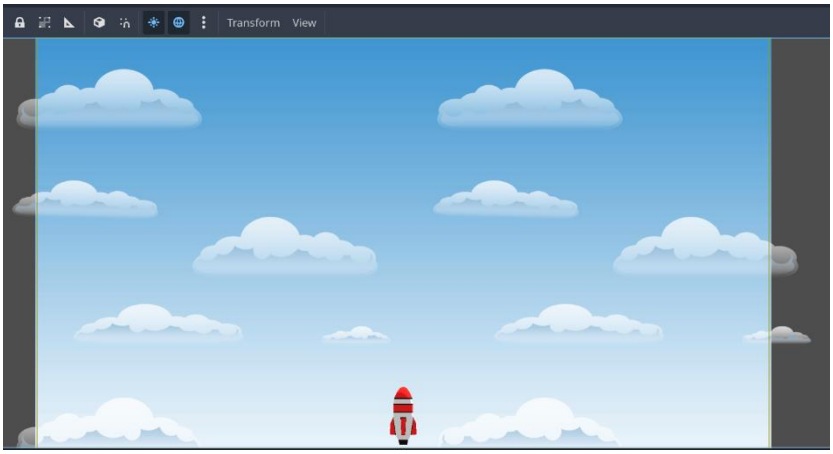




67 In **Inspector**, open the **Region** menu. Make sure the **Enabled** property is **Toggled On**.

Under **Region**, set the **w** and **h** properties to **2000.** This sets the size of the rectangular region where the clouds will be visible.



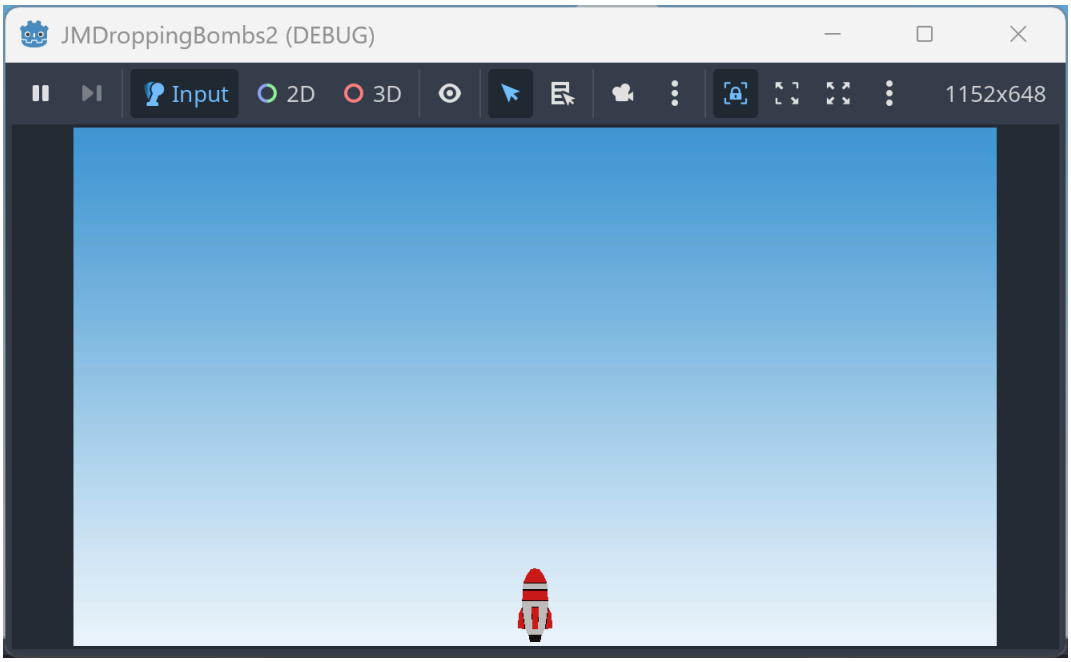


Reminder:

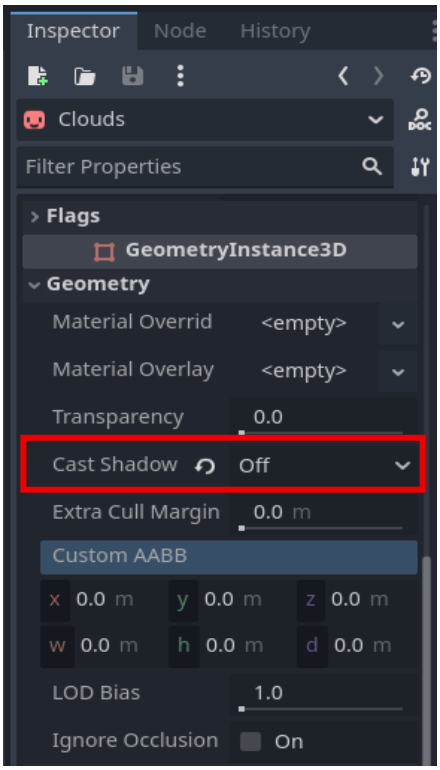
Don't forget to manually set both **w** and **h** values!

68 Playtest the game.

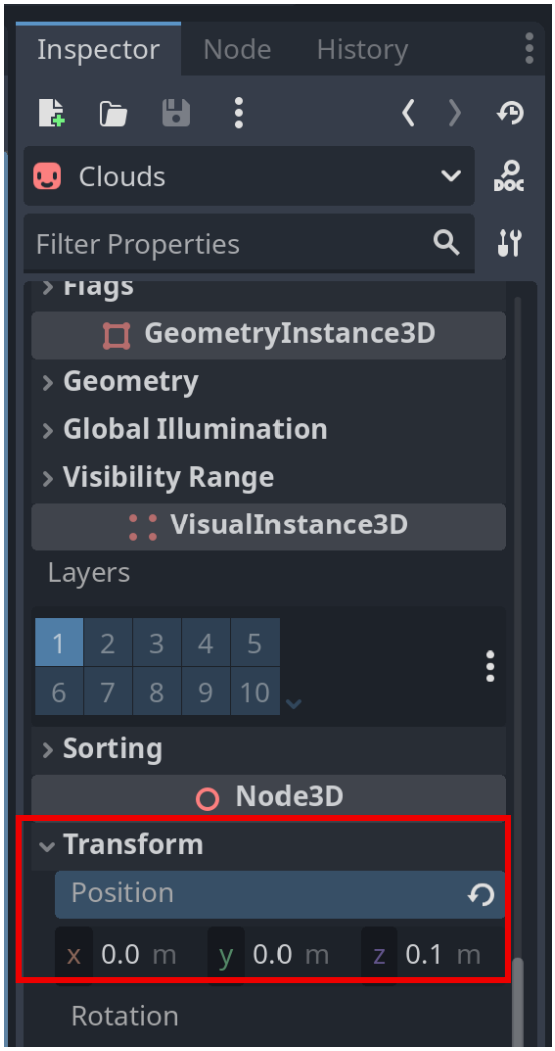
Notice that the **Clouds** are not visible! This will be fixed in the next steps.



69 In **Inspector**, locate and open the **Geometry** menu. Find the **Cast Shadow** property and set it to **Off**.



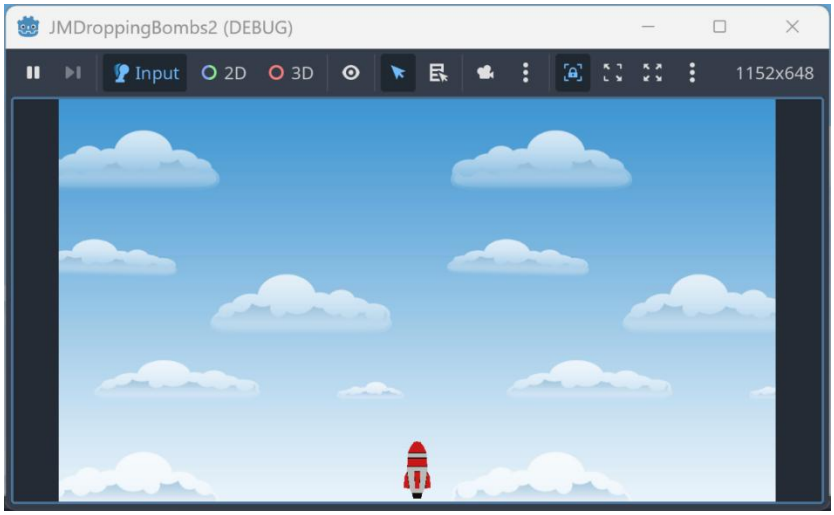
70 In **Inspector**, locate the **Transform** menu. Set the **z Position** to **0.1** to display the clouds in front of the sky when playing the game.



71 Playtest the game.

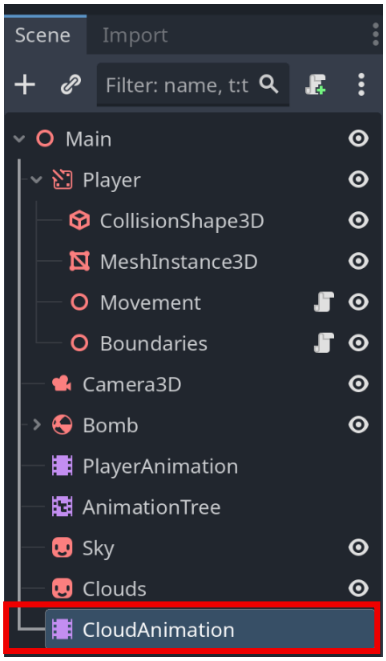
The clouds should now be visible in front of the **Sky**. Notice that changing the **z** value sets the order that sprites appear in.

What happens if the **z** value is set to a negative value?



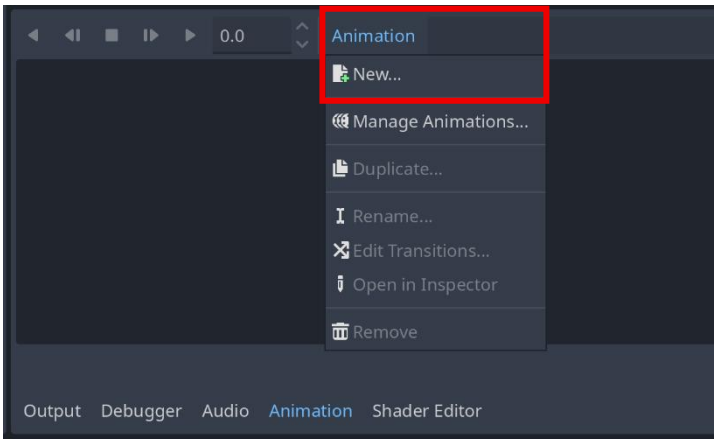
72 Create a vertical scrolling animation for the clouds.

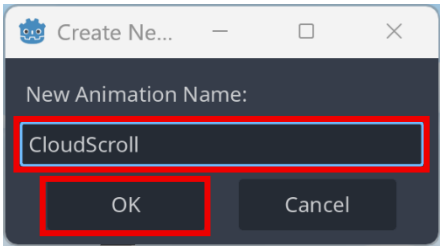
In **Scene**, create an **AnimationPlayer** node as a child to **Main root.** Rename it **CloudAnimation**.



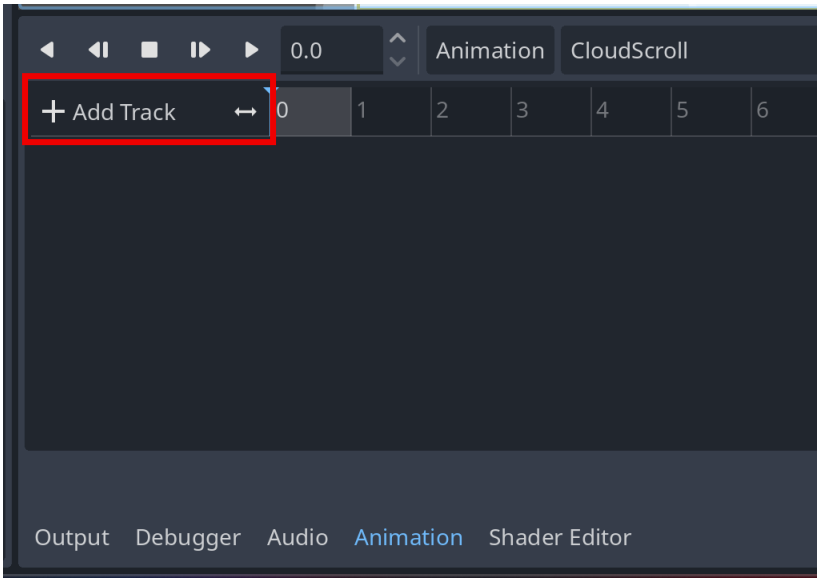
73 In the **Animation Bottom Panel,** click **Animation** and select **New** from the dropdown menu.

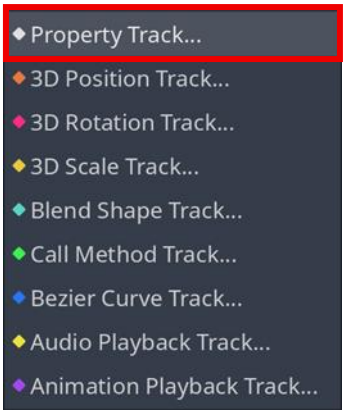
Name the new animation **CloudScroll** and click **OK**.



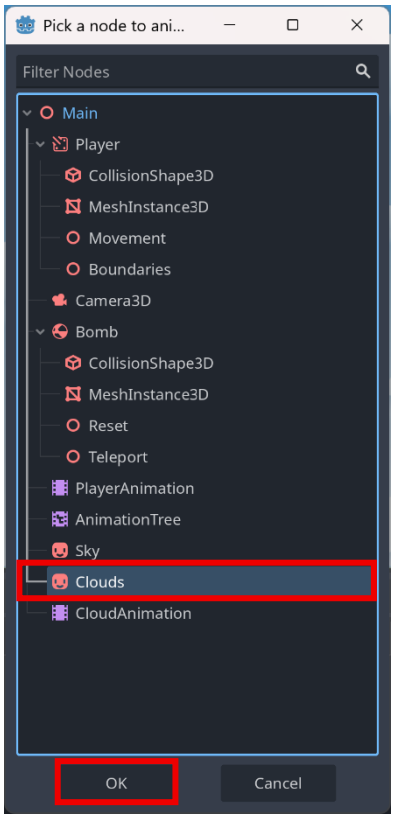


74 Click **Add Track**, then select **Property Track** from the dropdown menu.

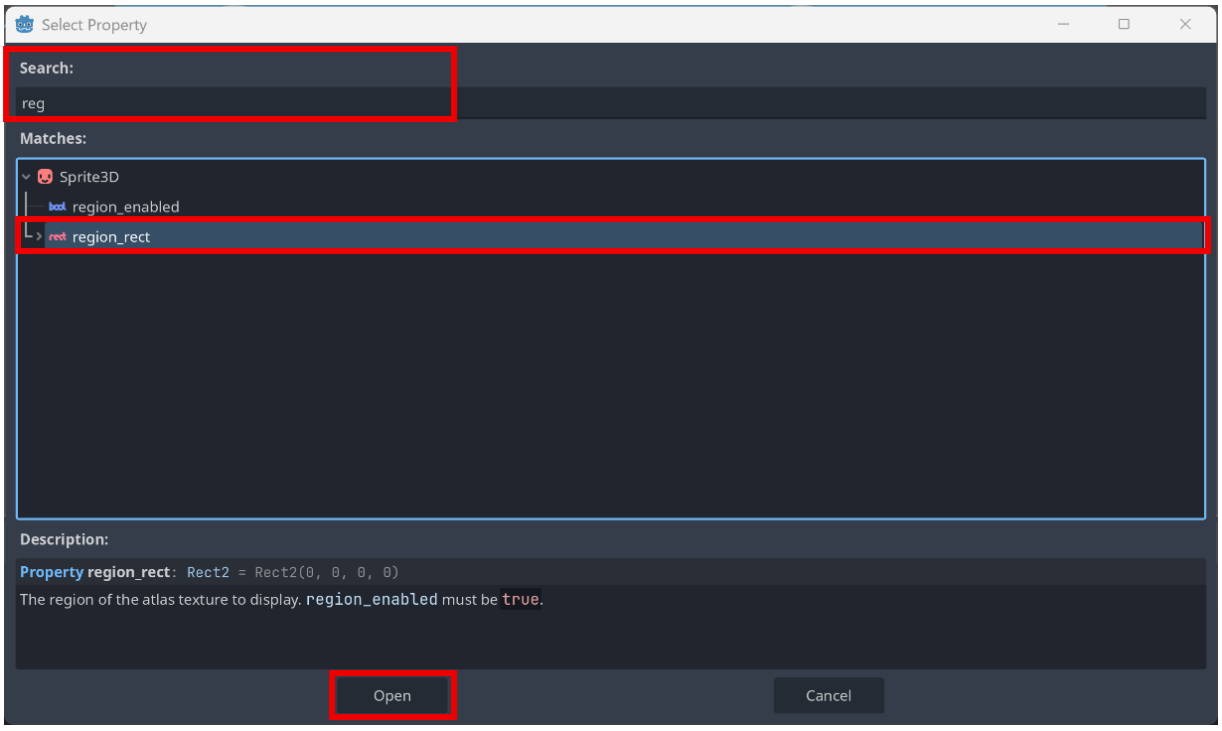




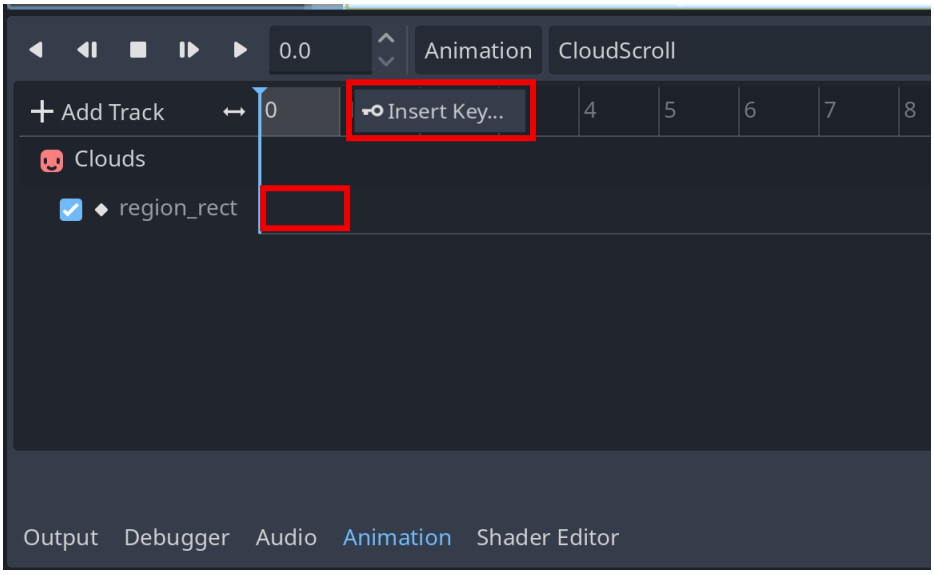
75 Select **Clouds** and click **OK**.



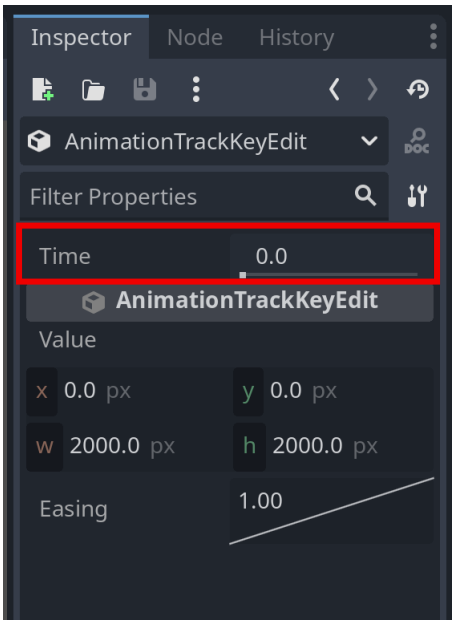
76 In the search bar, type in **region_rect**. Select it from the results and click **Open**.



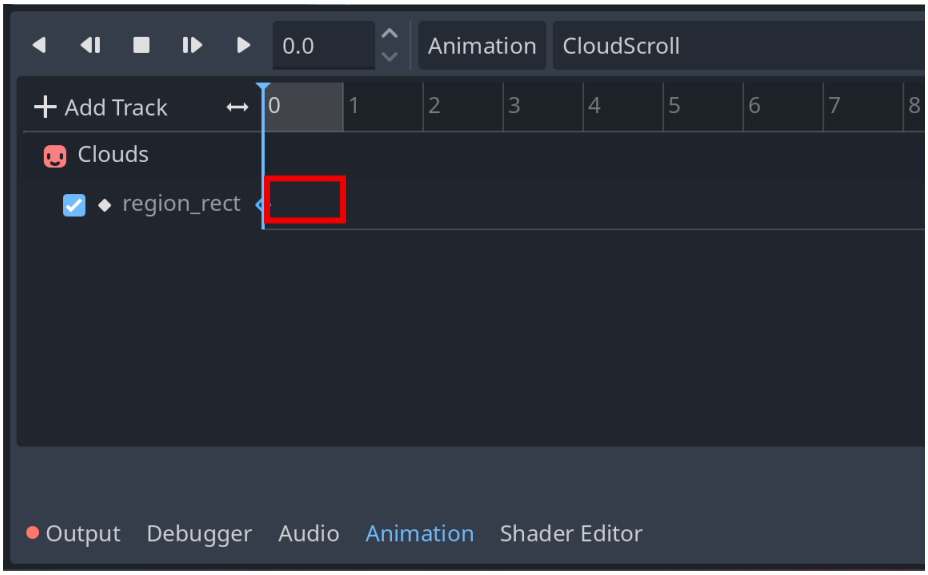
77 Right click in the space next to **region_rect** and select **Insert Key**.



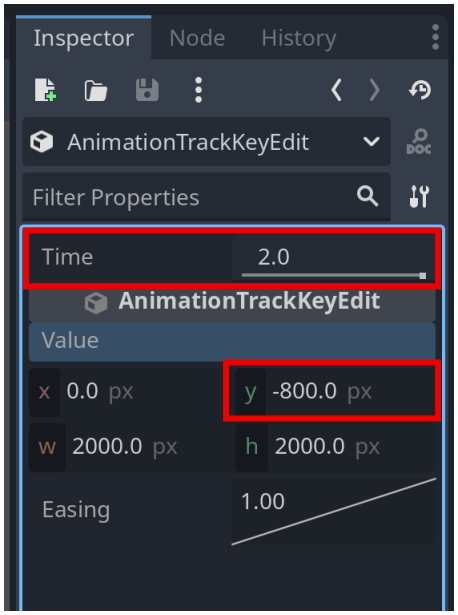
78 In **Inspector,** locate the **Time** property and set it to **0.**



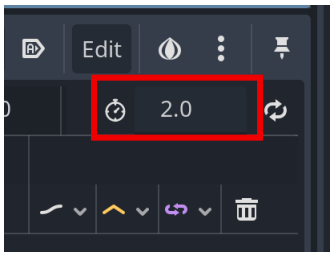
79 Right click in the area next to **region_rect** and select **Insert Key**.



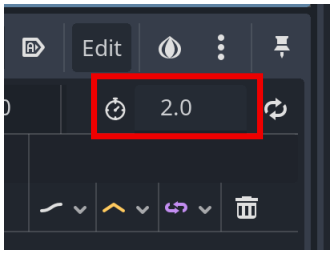
80 In **Inspector**, set **Time** to **2** and **y** to **-800**.



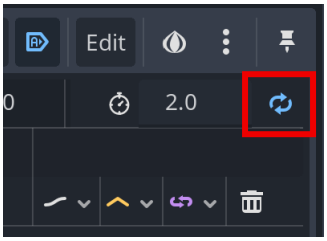
81 On the right side of the **Animation Panel**, set the **Animation Length** to **2**.



82 In **Animation Panel**, set **Autoplay on Load** to be **Toggled On** by clicking the icon. A blue icon means that Autoplay on Load is on.



83 In **Animation Panel**, set **Animation Looping** to be **Toggled On** by clicking the repeat icon . A blue icon means that looping is turned on.

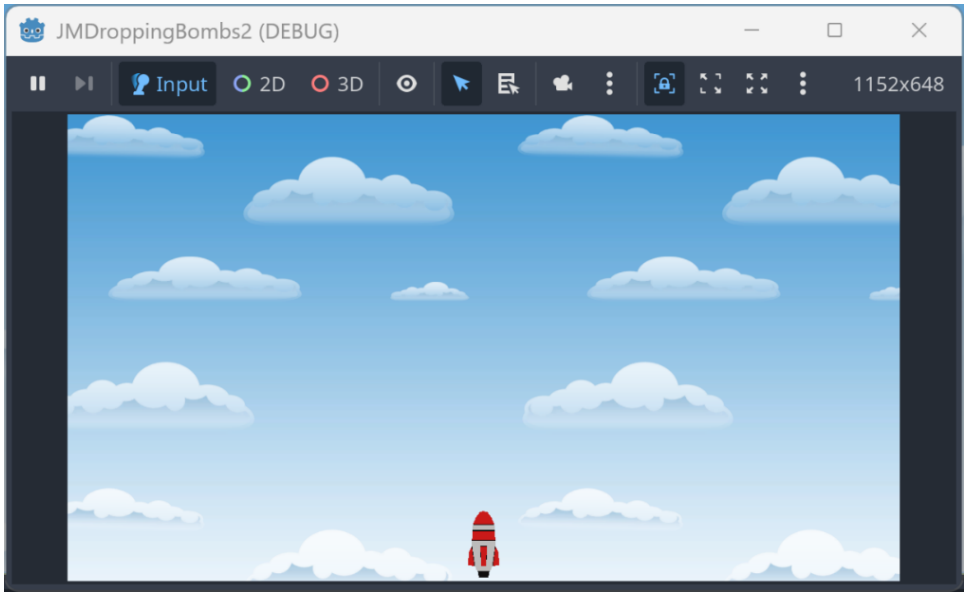


84 Playtest the game.

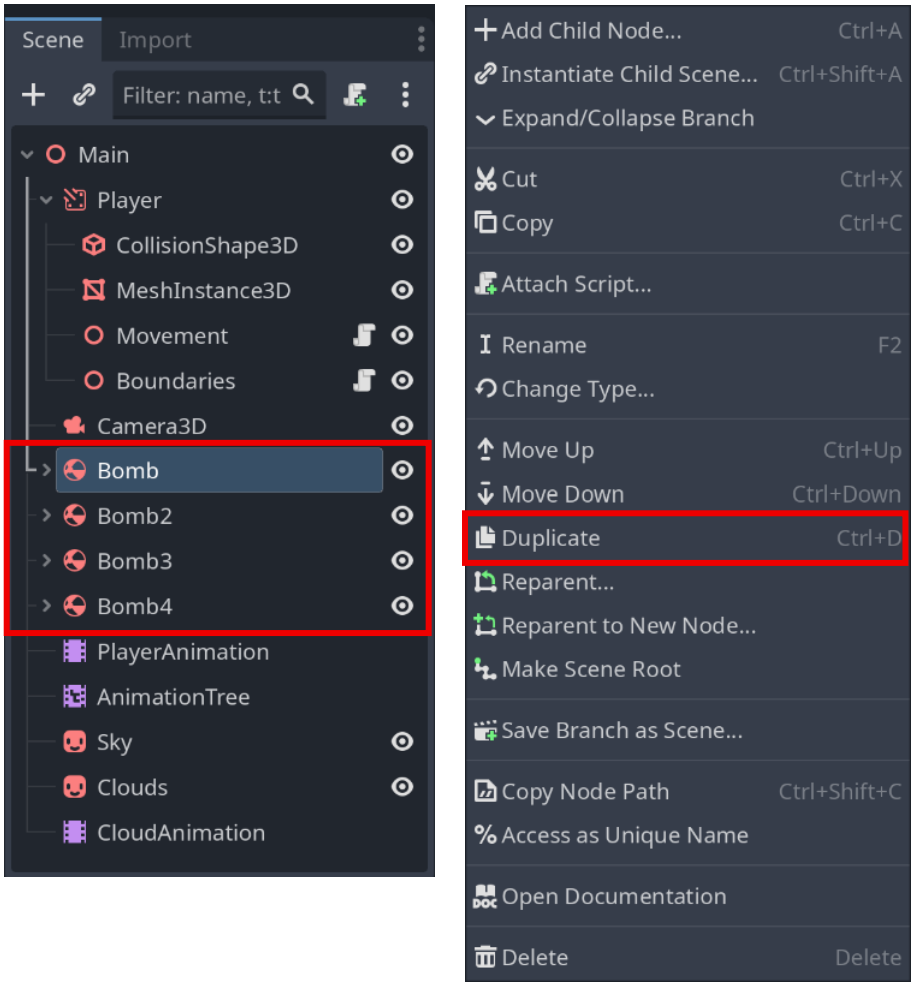
The clouds should now move vertically across the sky in a loop.

The game is almost done, but right now it's too easy with only one bomb.

How could the game be made more challenging?



85 In **Scene**, right click the **Bomb** node and select **Duplicate** from the dropdown menu to make copies of the node. Make **at least 3** copies.



Pro Tip:

Press **CTRL + D** on the keyboard while a node is selected to quickly duplicate it!

Pause for **Sensei Stop #6**!

Congratulations on creating your first game with animations in Godot! Great job!

Before submitting, check in with a Code Sensei to make sure the animations and sprites work then reflect on the following:

- What did you learn about animations? Sprites?
- What did you enjoy most when creating this project?
- What was something you found difficult and why?

Reminder: Save your work!

Congratulations on completing **BB Activity 08: Dropping Bombs Part 2** and in Godot – **You Rock**! You are now ready to save this project and submit it.

Continue your exploration with Godot by opening the **BB Activity 09: Dropping Bombs Part 3** Ninja Guide.