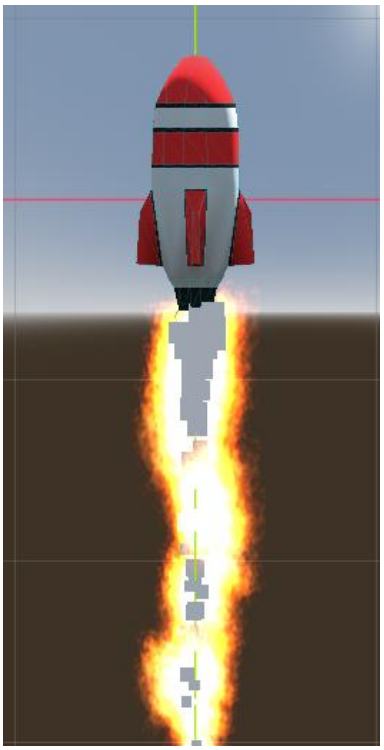
CODE NINJAS®

Bronze Belt Ninja Guide

Activity 12: Dropping Bombs Part 6

Particles



When an action occurs in a game, adding an effect can make it more visually interesting and inform the player what the nature of the action is. **Particles** are a useful, flexible way to implement these effects. Think of some examples of particles used in popular games: bubbles when a player is underwater, sparkles around a valuable item, and explosion particles around damaging items.

Particles can be used to create numerous effects and can be emitted in multiple dimensions within a scene. In 3D games within Godot, the **CPUParticles3D** node can be used to create particles. This specific kind of node uses the computers **CPU**, or Central Processing Unit, to render particles. There are usually many particles that need to be rendered at once to make an effect visible and visually attractive. Rendering so many particles can be computationally demanding; often it is helpful to use the GPU which is optimized to perform the exact kinds of computations that rendering particles requires. However, not all computers have a GPU strong enough to render particles, so using the CPU makes playing games that use particles more accessible for a variety of hardware.

Within the **CPUParticles3D** node, there are many settings that assist in determining the direction, lifetime, scale, look, and speed of particles when they emit. To function, each particle needs a **Mesh.** The **Mesh** determines the look that each particle will have when they spawn in.

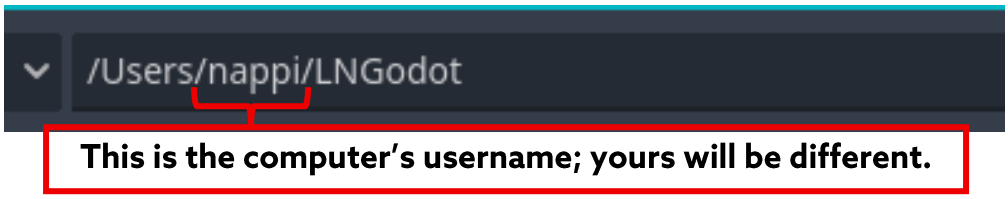
Activity 12: Dropping Bombs Part 6

In this project, you will expand on Dropping Bombs by adding particles to the Rocket and Bomb. Additionally, you will script these particles to emit at the appropriate time.

By the end of this activity, you will have explored how to create, edit, and script particles.

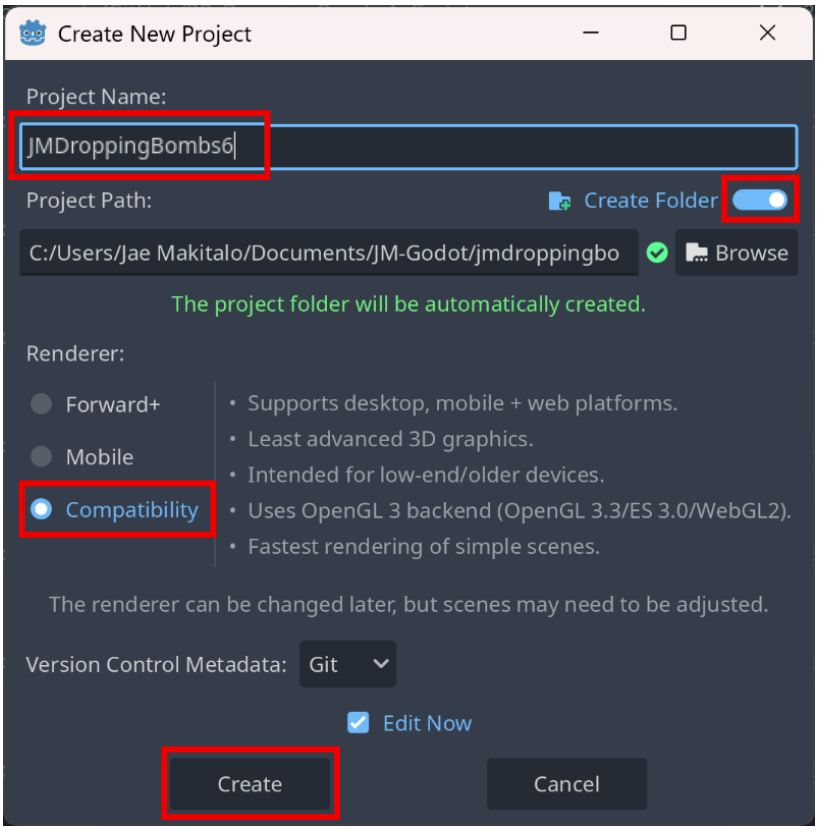
1 All projects will be stored in a path like:
/Users/[MyComputerUsername]/[MyInitials]Godot

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]DroppingBombs6.**

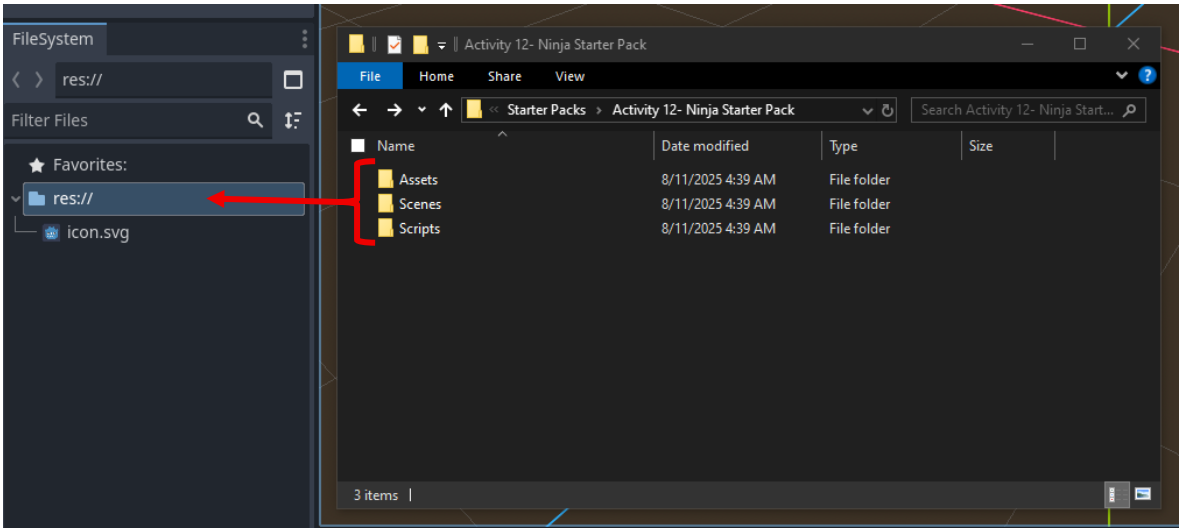
Check that **Create Folder** is turned on, and that the **Compatibility** mode for the renderer is being used. Then click **Create**.



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

Extract **BB Activity 12 – Ninja Starter Pack.zip** and select all folders inside. Drag them into the **res://** folder in **FileSystem**.

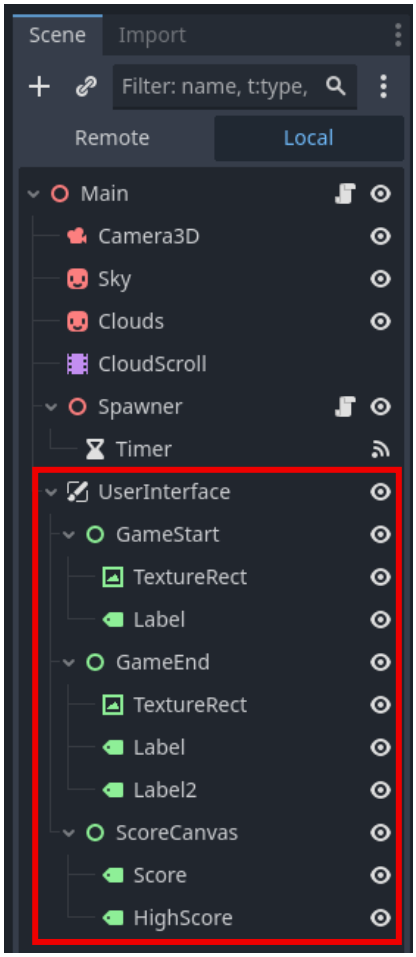
At the top center of the editor, check that **3D** is selected.



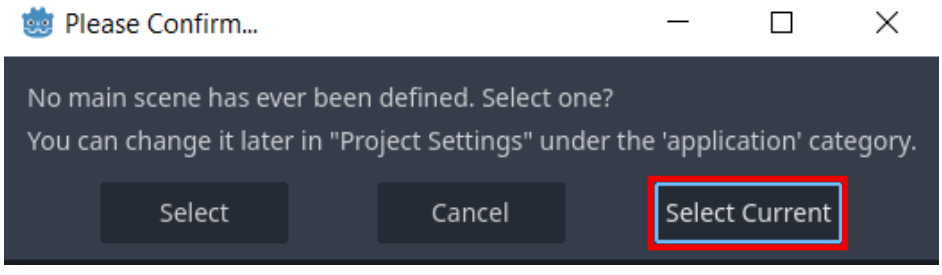
Reminder:

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 In **FileSystem**, navigate to the **Scenes** folder, and double click **main.tscn** to open it. Notice the main scene already has the score UI created from Part 5.

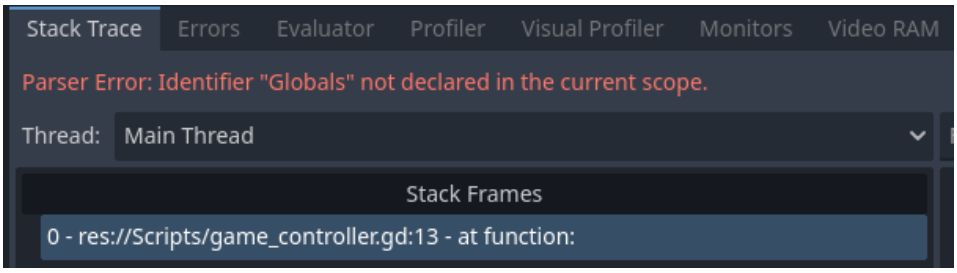


5 In the top right corner, click the **play** button to run the game. Click **Select Current** to define the main scene.



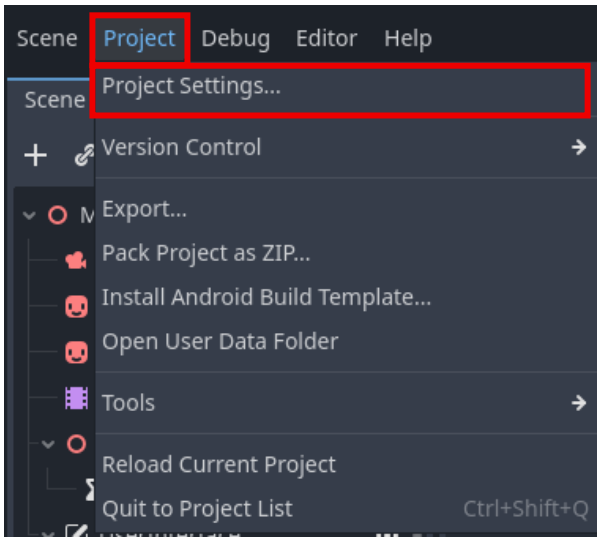
6 The game will not load on the first playtest and will crash immediately. There will be a parser error stating that **Globals** is not declared in the current scope.

Ignore this; it will get fixed in the next steps.

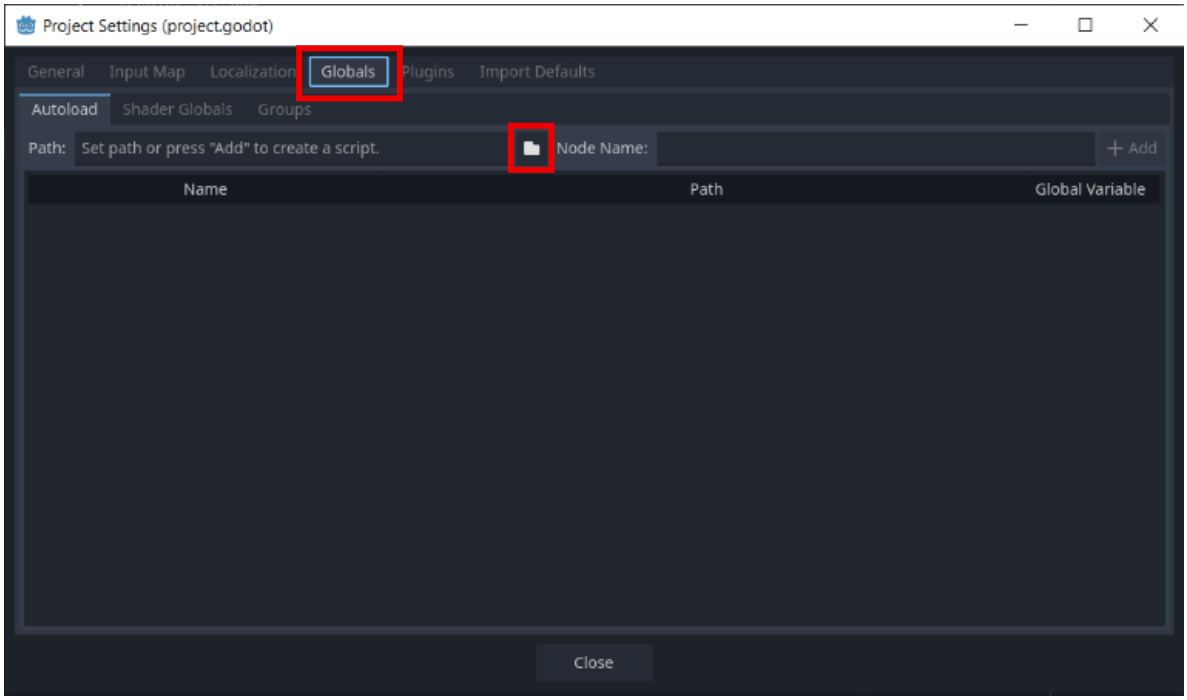


7 Add a Globals script to the project.

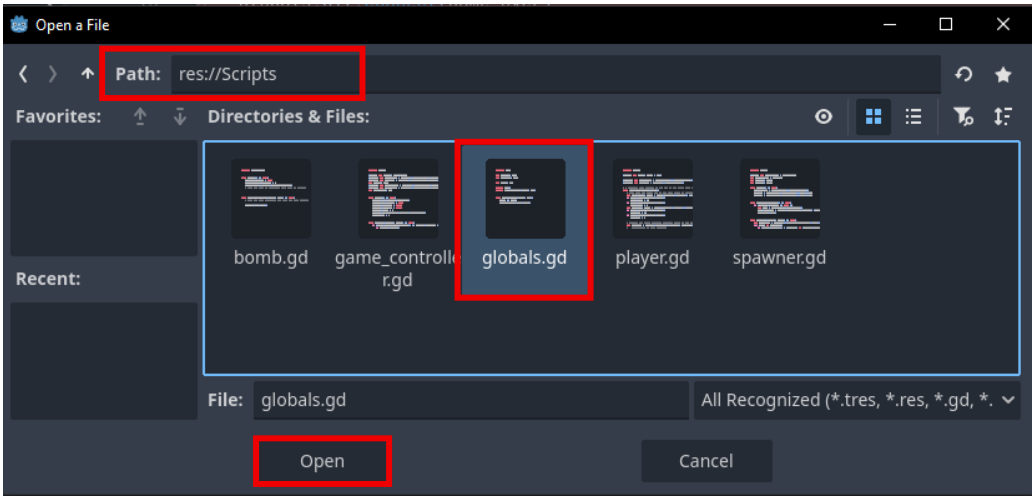
In the top left corner of the editor, select **Project**. Select **Project Settings** from the dropdown menu.



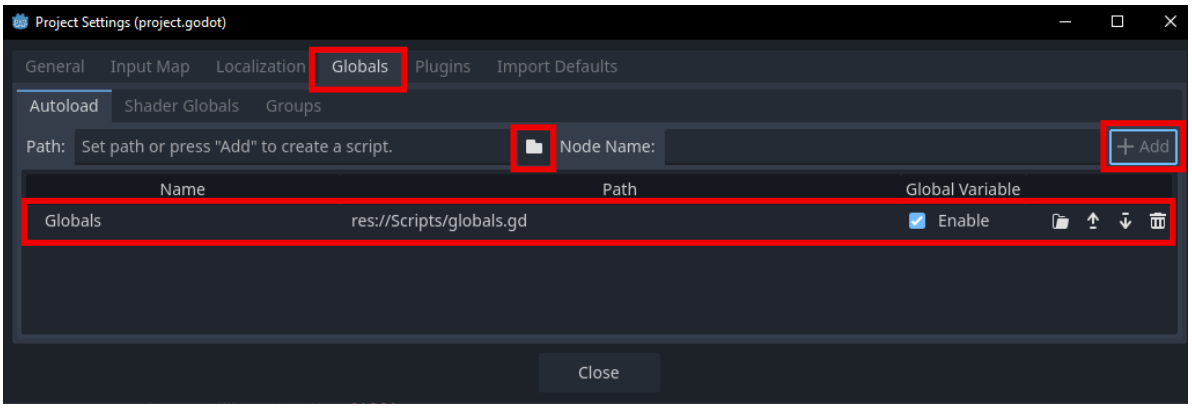
8 In the **Project Settings** window, select the **Globals** tab. Then, click the **folder icon.**



9 In the new popup window, open the **Scripts** folder. Select the **globals.gd** file and click **Open**.



10 Click **Add** and double check that the node has appeared in the list below. Close the **Project Settings** window.



11 Playtest the game. Now that the Globals script is set, the playtest window will open.

Notice that the game is the same as where **Dropping Bombs Part 5** left off.

Pause for **Sensei Stop #1**!

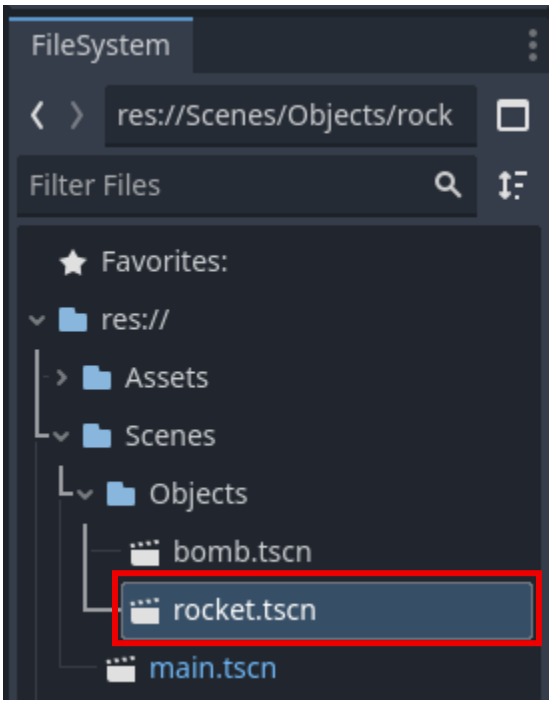
Check in with a Code Sensei before moving on. Make sure the **Globals** script, **Ninja Starter Pack,** and **main scene** are set up properly.

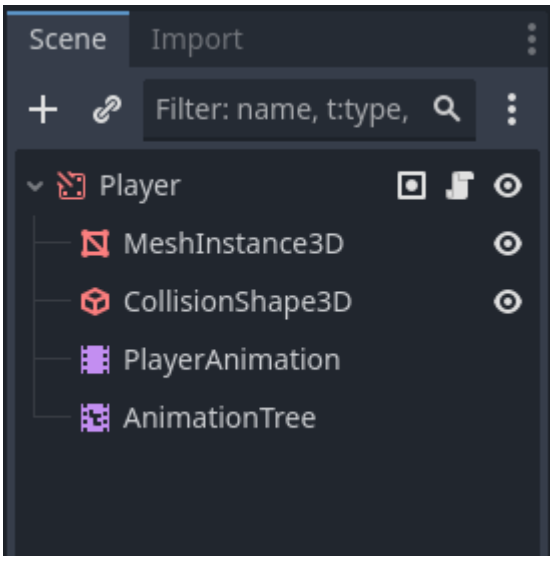
Reminder: Save your work!

12

Give the rocket smoke particle effects!

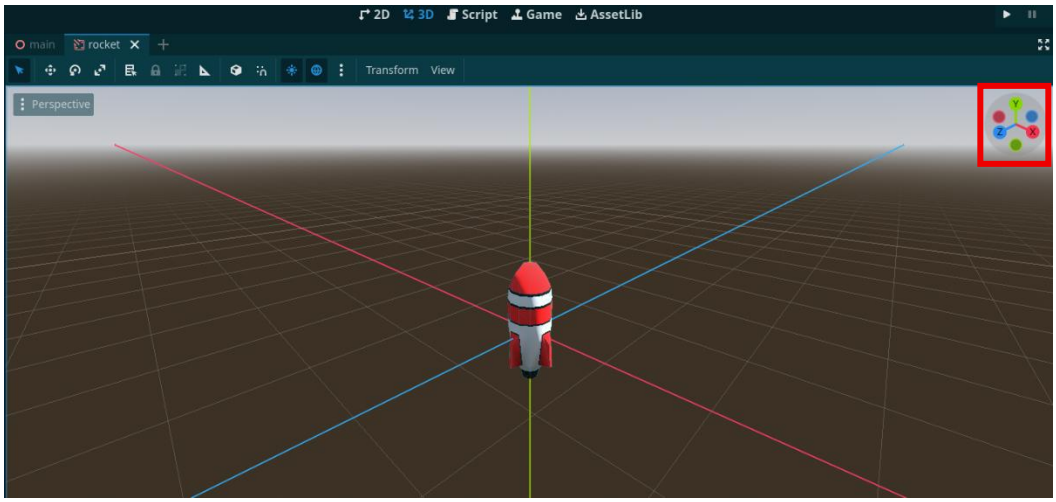
In **FileSystem**, open the **rocket.tscn** scene found under **Scenes > Objects.**

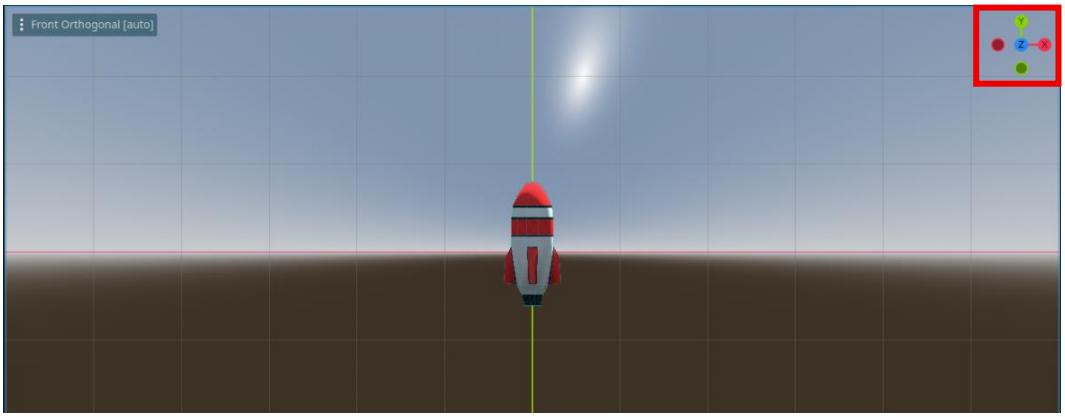




13 Select the **3D Workspace**.

Click the blue **Z** nodule on the orientation gizmo in the top right of the workspace to look down the Z axis.



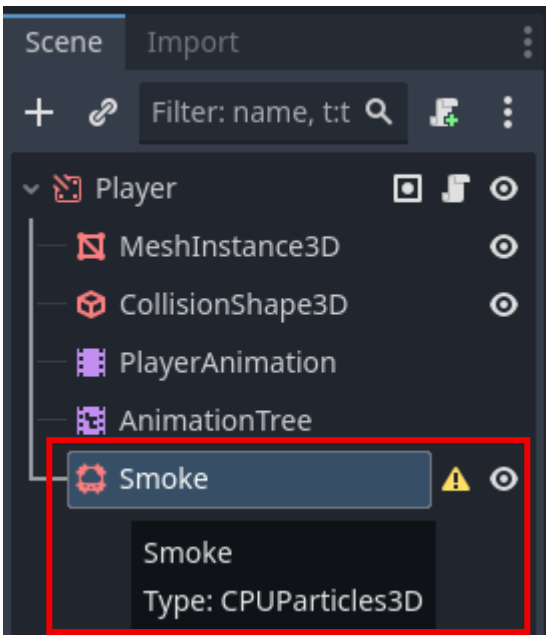


Pro Tip:

The particles will render in the 3D workspace without needing to playtest the game!

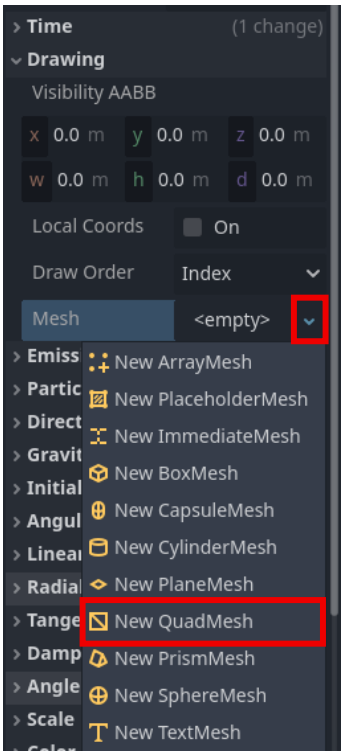
14 In **Scene**, add a new **CPUParticles3D** node as a child to the **Player** and rename it **Smoke**.

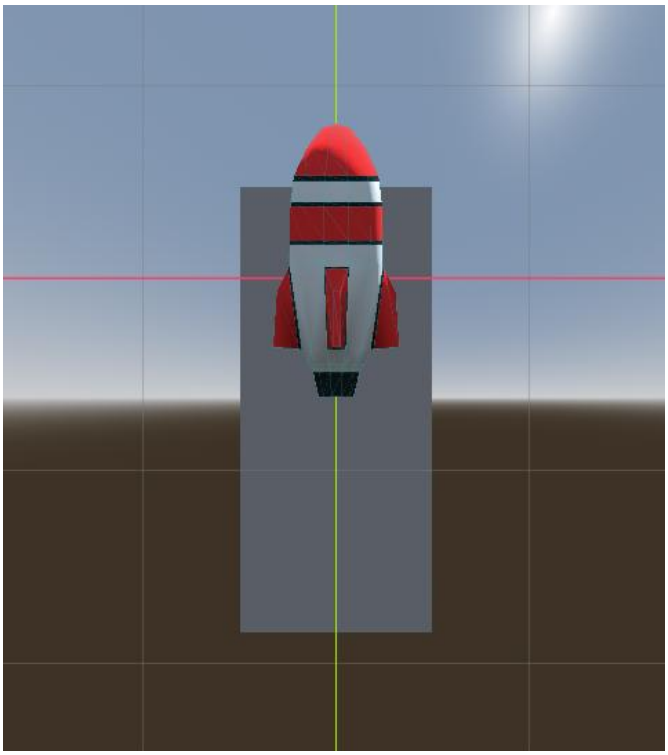
There will be an error icon next to the node because a mesh and a material have yet to be assigned. Don't worry; this will get fixed in the following steps.



15 In **Inspector**, under **Drawing**, click the drop-down menu to the right of **Mesh**. Select **New QuadMesh.**

The rocket should have many large grey squares dropping from it. Since they all have the same material, it looks like they are one long rectangle whose shape varies in length.





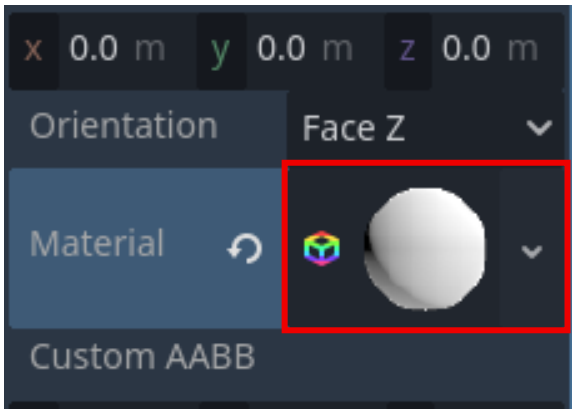
16 Click on the square icon next to **Mesh**.

Under **Mesh**, click the drop-down menu to the right of **Material**. Select **New StandardMaterial3D.**

17 The smoke particles should be darker and somewhat transparent.

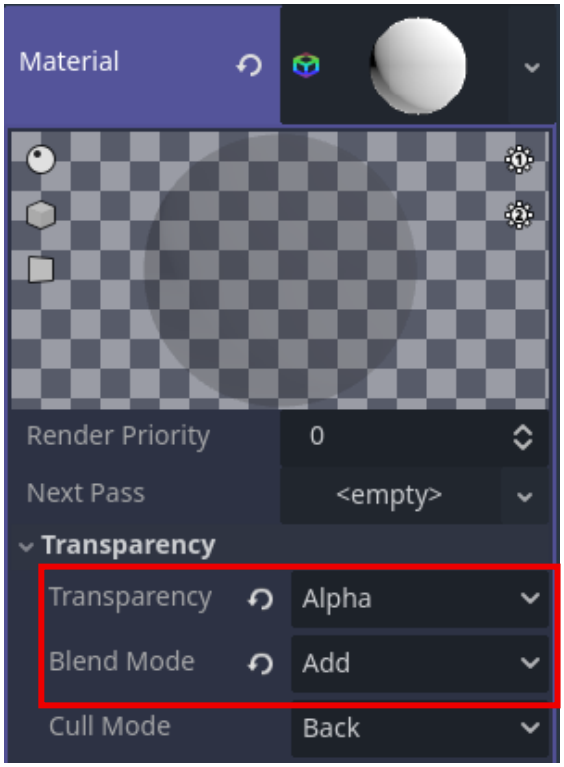
Click the **sphere** to open the material settings. Change the **Albedo** to a darker gray, and the **alpha** value to halfway.

Albedo is the shade and color of the texture. **Alpha** is the transparency amount.



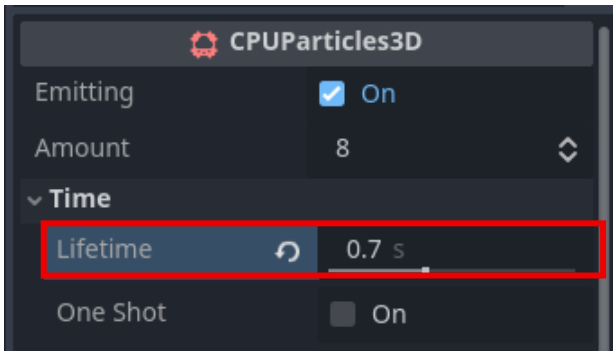
18 In the **Transparency** section, set **Transparency** to **Alpha** and **Blend Mode** to **Add**.

These settings enable **transparency** and add **transparency** based on the **alpha** value.



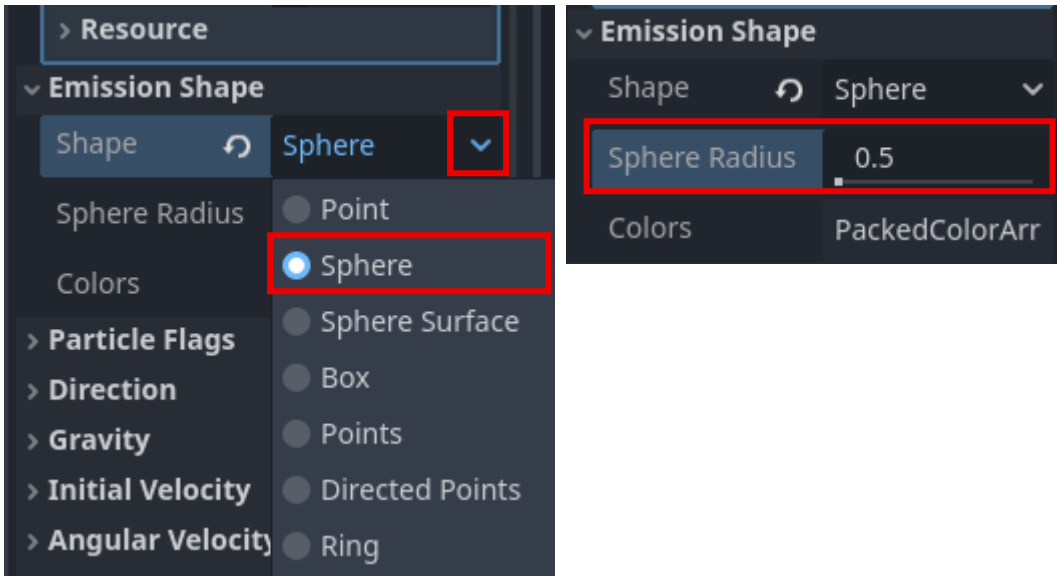
19 In **Inspector**, under **Time**, set the **Lifetime** to **0.7**.

Lifetime controls how long the particle will stay active in the game before it is destroyed.

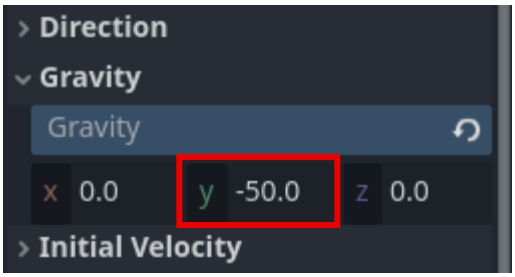


20 In **Inspector**, under **Emission Shape**, click on the dropdown arrow next to **Shape**. Select **Sphere** and change **Sphere Radius** to **0.5.**

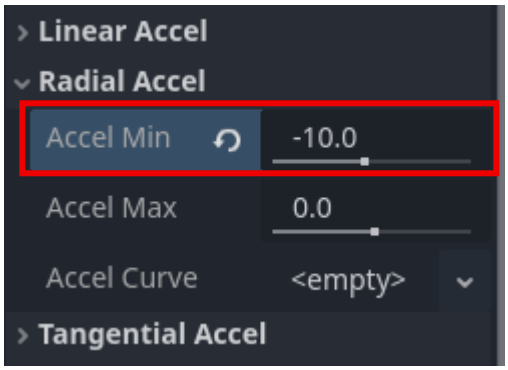
The **Emission Shape** determines what shape particles can spawn in. The sphere setting will spawn particles in a sphere with the given radius.



21 In **Inspector**, under **Gravity**, set the **y** to **-50**. This will cause the particles to fall with gravity.

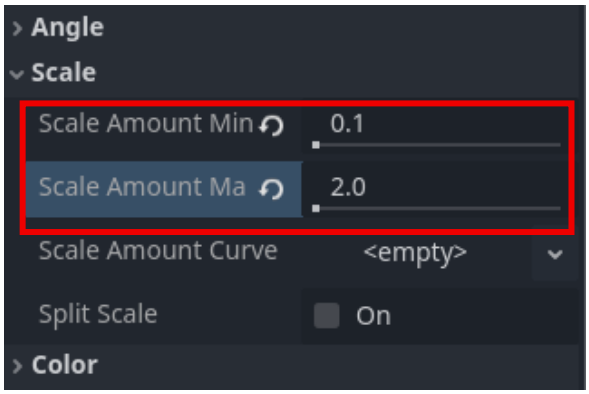


22 In **Inspector,** under **Radial Accel**, set the **Accel min** to **-10**. This will push the particles further away from the center of the sphere.

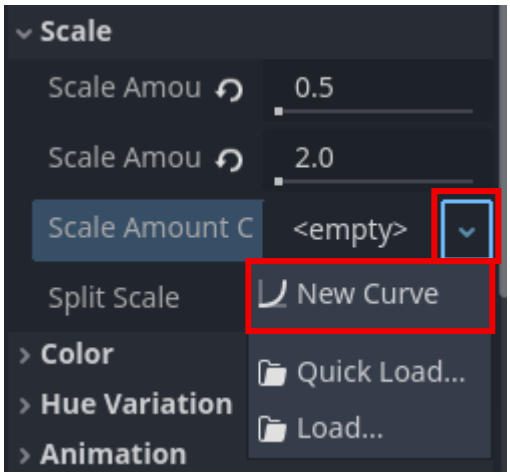


23 In **Inspector**, under **Scale**, set **Scale Amount min** to **0.1** and **Scale Amount max** to **2.0**.

Now, spawned particles will have a random size between **0.1** and **2.0** times their size.

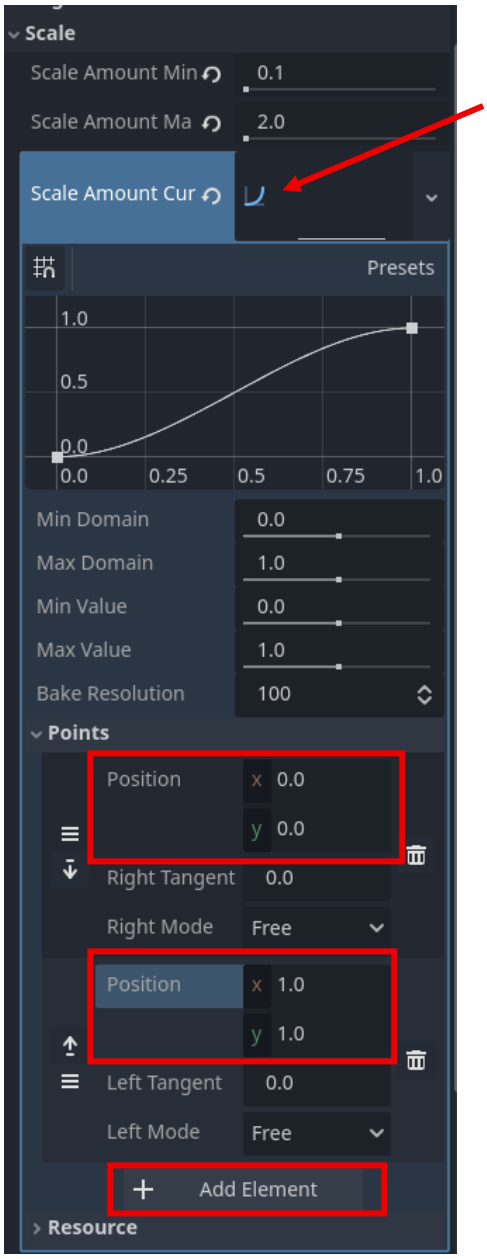


24 Click the dropdown next to **Scale Amount Curve** and select **New Curve**.

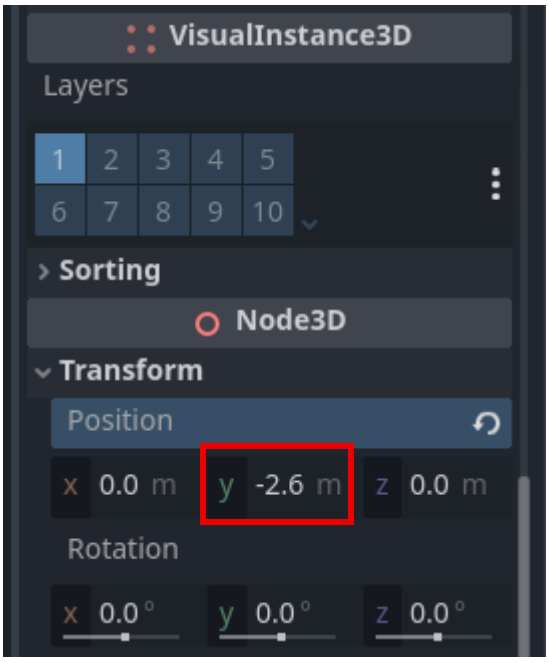


25 Click the image next to **Scale Amount Curve**. Under **Points**, click **Add Element** and set the **x** to **0** and the **y** to **0**.

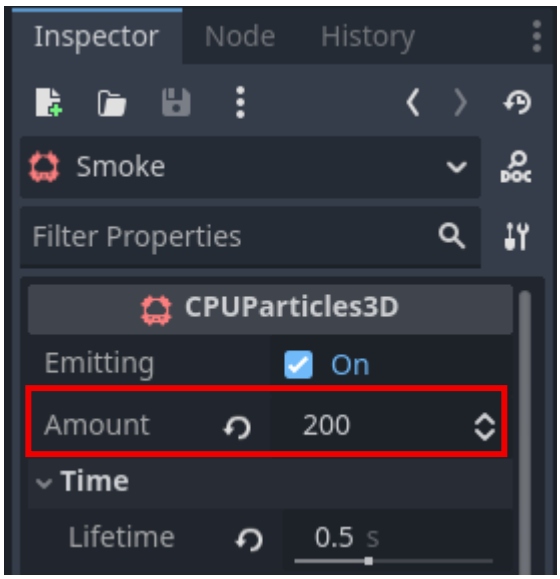
Click **Add Element** a second time and set the **x** to **1** and the **y** to **1**.
This curve will change the size of the particles over time, starting at **0%** of their size (because Y is 0) and increasing to **100%** of their size.



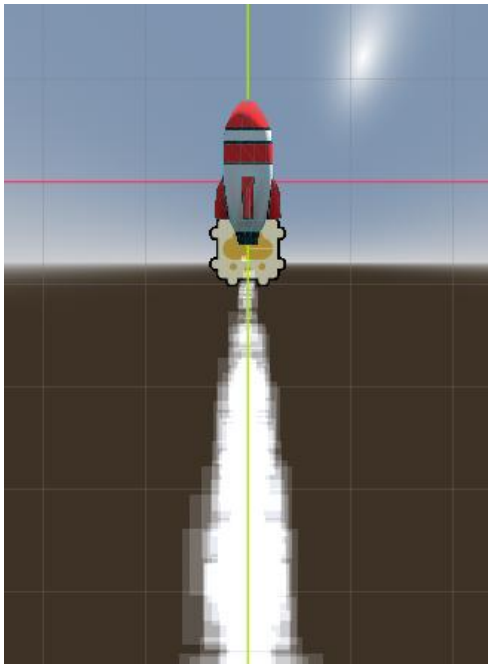
26 In **Inspector**, under **Transform**, set **Position y** to **-2.6.**



27 In **Inspector**, set **Amount** to **200**. If the performance is poor (the project or particles seem to lag), reduce the amount of particles.



28 View the particles as they render in the 3D workspace. Check that they resemble the image below.



Pause for **Sensei Stop #2**!

Check in with a Code Sensei before moving on. Make sure the **Smoke** particle effect is set up properly.

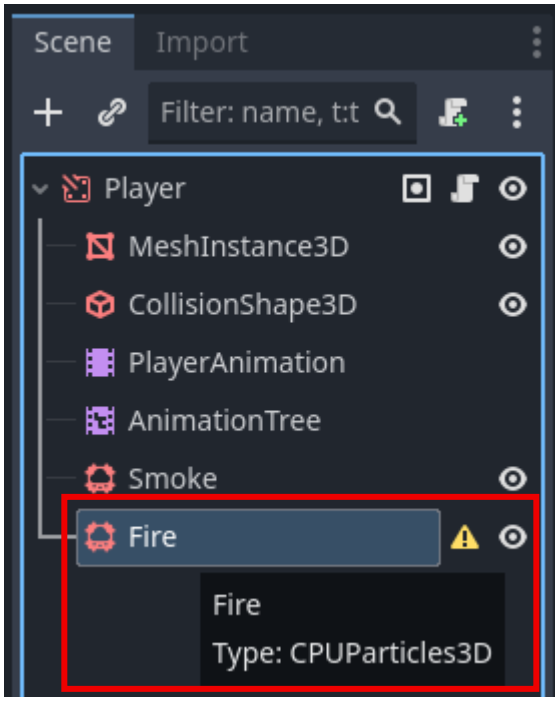
- ☐ *How did you change the color and transparency of the smoke?*
- ☐ *What settings did you change to affect the particle's size? How does the size change over time?*
- ☐ *What other settings did you change on the particle?*

Reminder: Save your work!

29

Give the rocket fire particle effects!

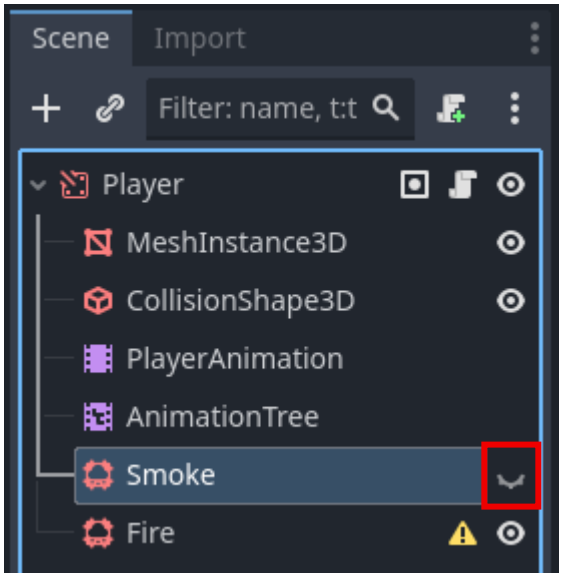
In **Scene**, add a new **CPUParticles3D** node as a child to the **Player**. Rename it **Fire**. There will be an error icon next to the node because a mesh has yet to be assigned. Don't worry; this will get fixed in the following steps.



30

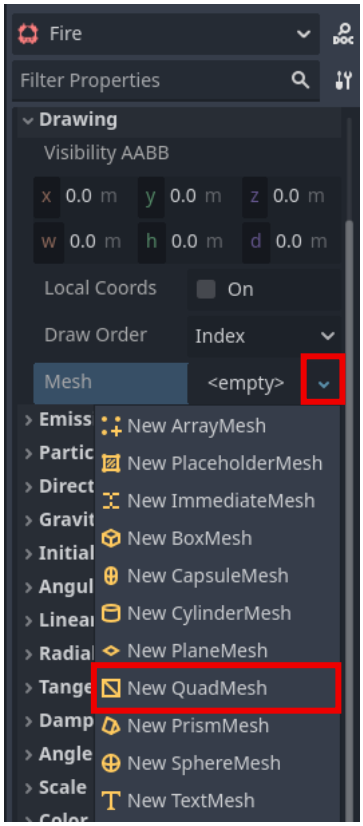
In **Scene**, click the open eye icon next to the **Smoke** node to make it invisible.

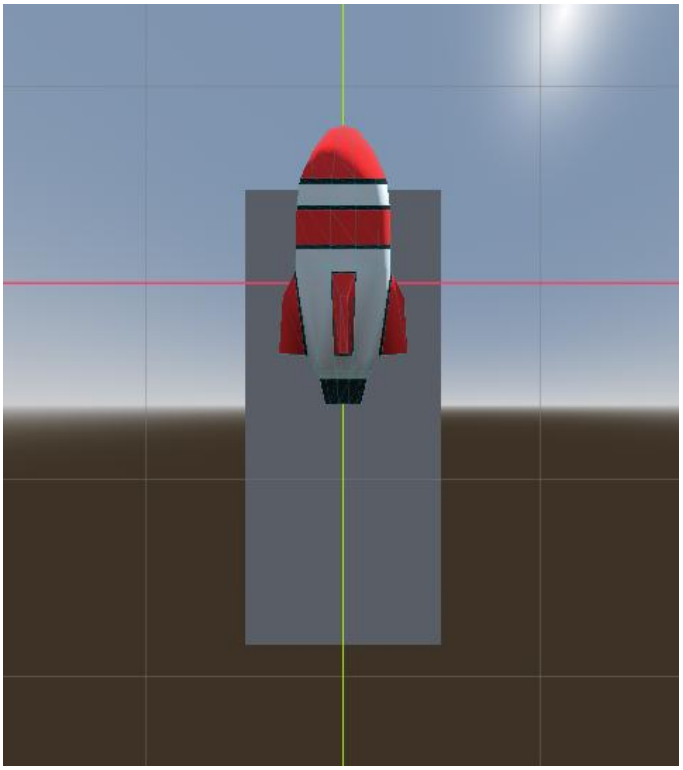
This will only render the fire particles, which will make them easier to edit.



31 In **Inspector**, under **Drawing**, click the dropdown to the right of **Mesh**. Select **New QuadMesh**.

The rocket should have many large grey squares dropping from it. Since they all have the same material, it looks like they are one long rectangle whose shape varies in length.



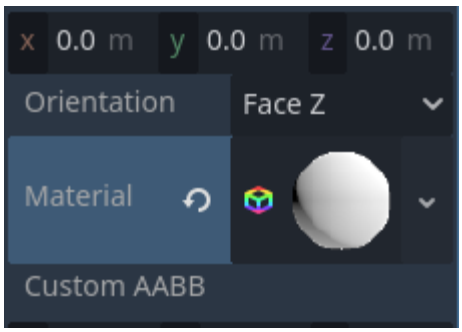


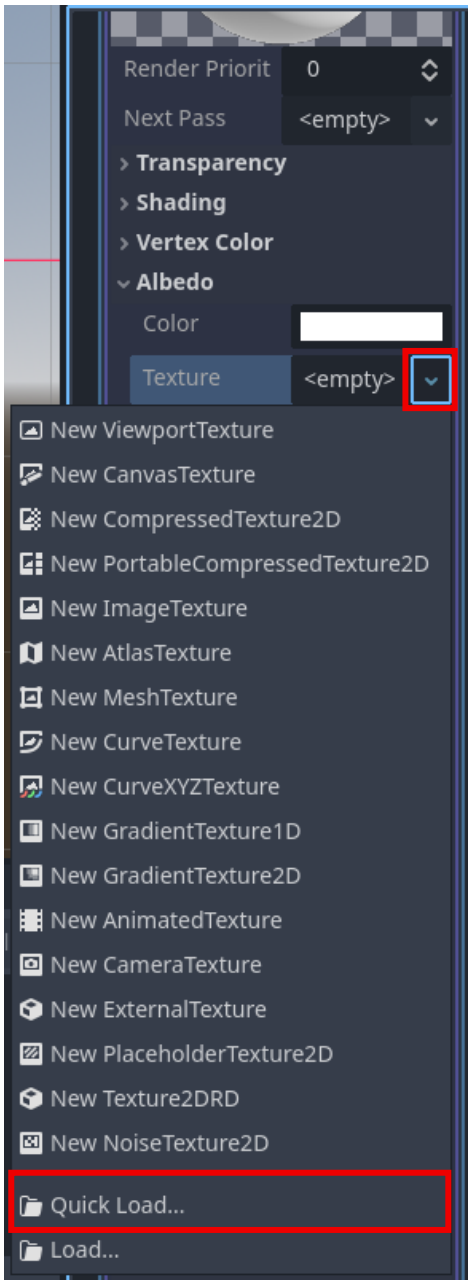
32 Click on the square icon next to **Mesh**.

Under **Mesh**, click the dropdown to the right of **Material**. Select **New StandardMaterial3D.**

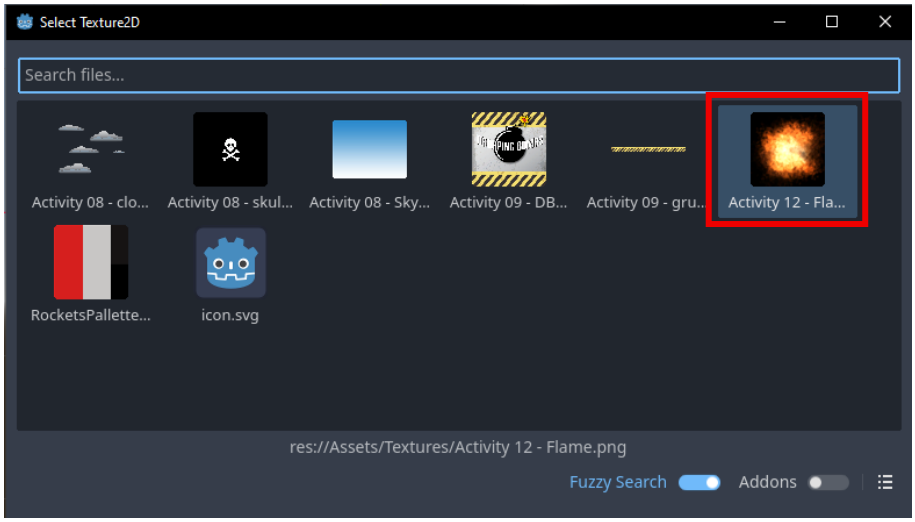
33 Click on the sphere to open **Material.**

Under **Material > Albedo**, click the dropdown next to **Texture**. Select **Quick Load.**

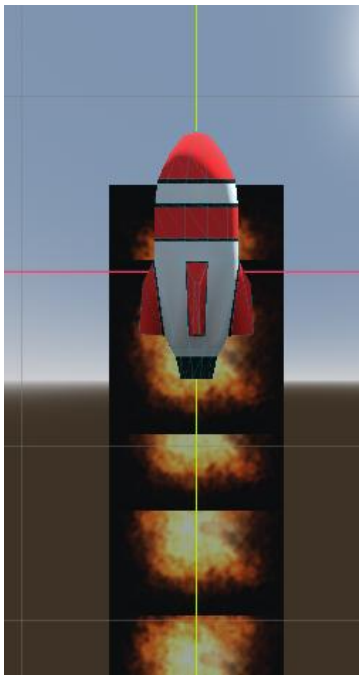




34 Select **BB Activity 12 – Flame.png** from the **Select Texture2D** window.

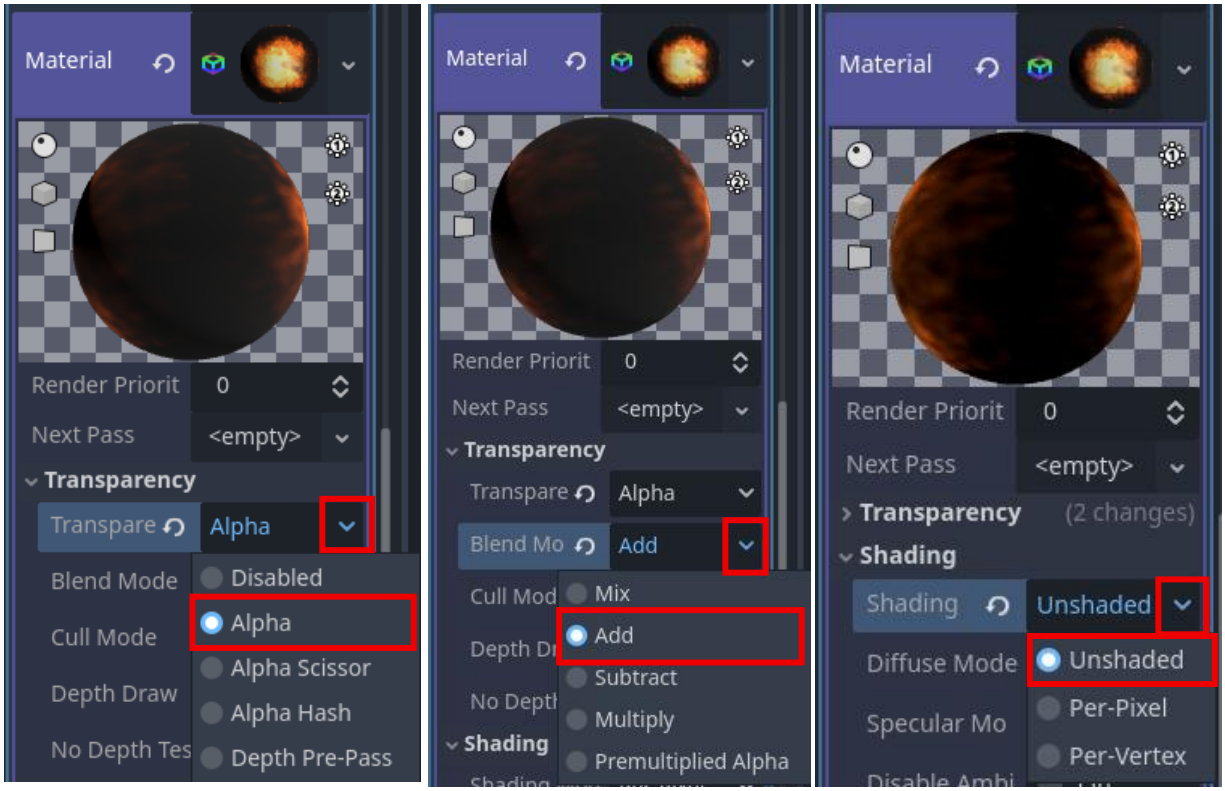


35 View the particles as they render in the 3D workspace. Check that they resemble the image below.

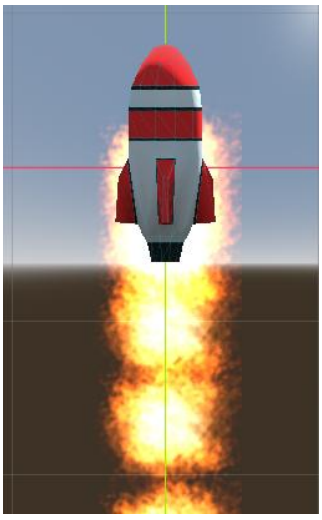


Currently, the particles still have a black background on the fire. This will be fixed in the following steps.

36 In **Material** under **Transparency**, set **Transparency** to **Alpha**. Then, set **Blend Mode** to **Add** and **Shading Mode** to **Unshaded**.



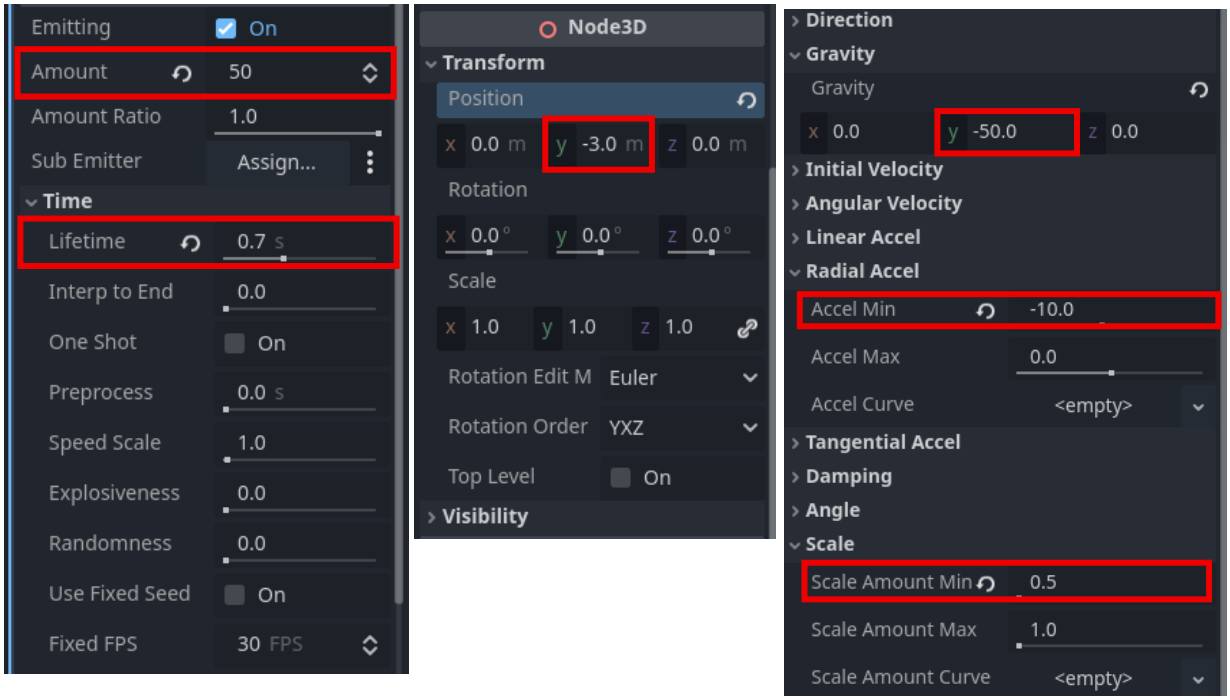
37 View the particles as they render in the 3D workspace. Check that they resemble the image below.



The fire particles should no longer have a visible background, are few, slow, and have no variation. Additionally, they emit from inside the rocket. This will get fixed in the following steps.

38 In **Inspector**, set the following properties of the **Fire** node to the specified values:

- **Amount** to **50**
- **Lifetime** to **0.7**
- **Position y** to **-3**
- **Gravity y** to **-50**
- **Radial accel min** to **-10**
- **Radial accel max** to **0**
- **Scale amount min** to **0.5**

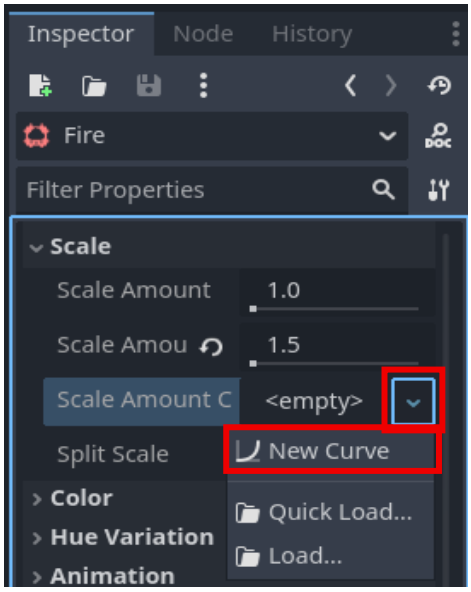


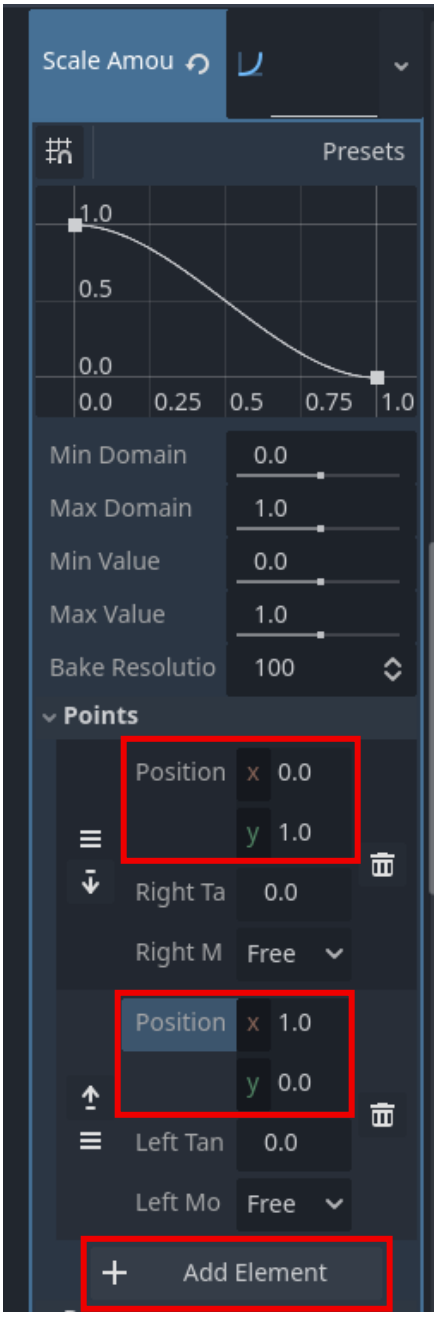
Refer to **steps 19-27** for additional guidance.

39 Click the dropdown next to **Scale Amount Curve** and select **New Curve**.

Click the image next to **Scale Amount Curve**. Under **Points**, click **Add Element** and set the **x** to **0** and the **y** to **1**.

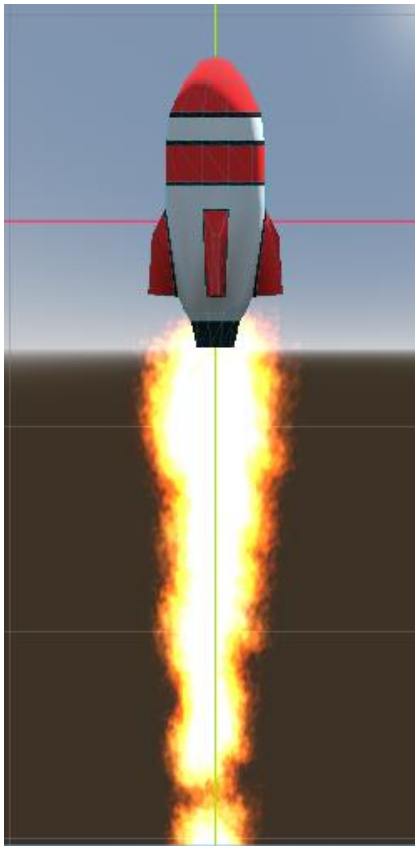
Click **Add Element** a second time, then set the **x** to **1** and the **y** to **0**.



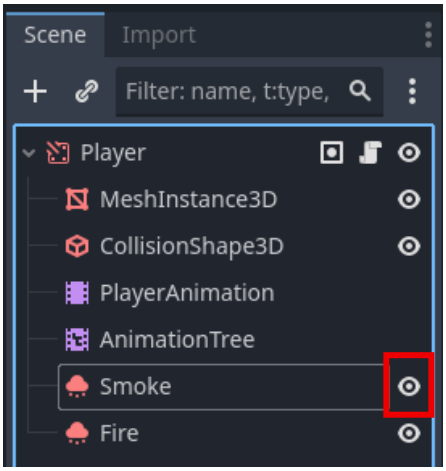


40 View the particles as they render in the 3D workspace. Check that they resemble the image below.

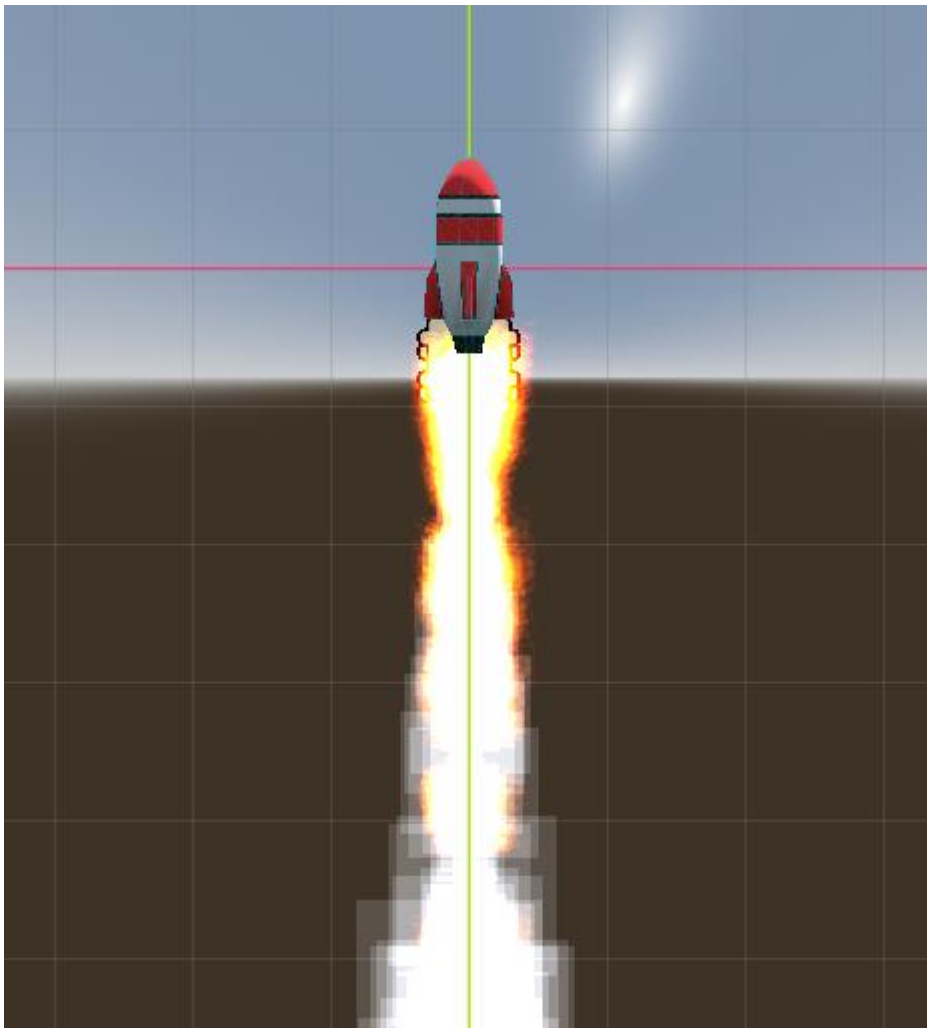
The particles are now much more interesting to look at. With the variation in size, the particles emulate the randomness of fire being emitted from a real rocket.



41 In **Scene**, click on the closed eye symbol next to the **Smoke** node to make the smoke particles visible again.



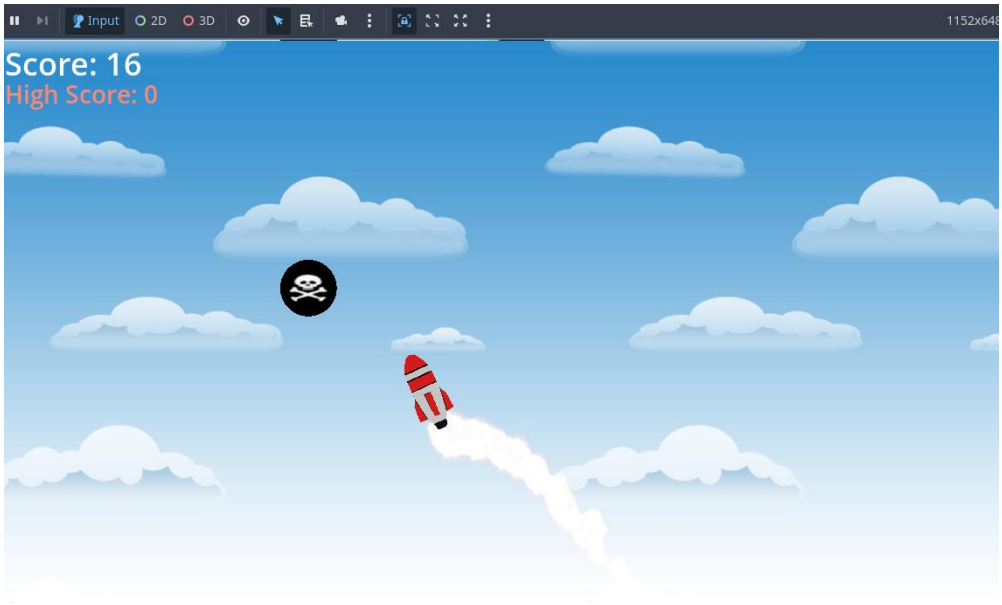
42 View the particles as they render in the 3D workspace. Check that they resemble the screenshot.



Both the smoke and fire particles should be visible and mixing as they fly downward.

43 Playtest the game.

Notice that the trail continues downward as the rocket moves, simulating the rocket flying and leaving its particles behind.



Pause for **Sensei Stop #3**!

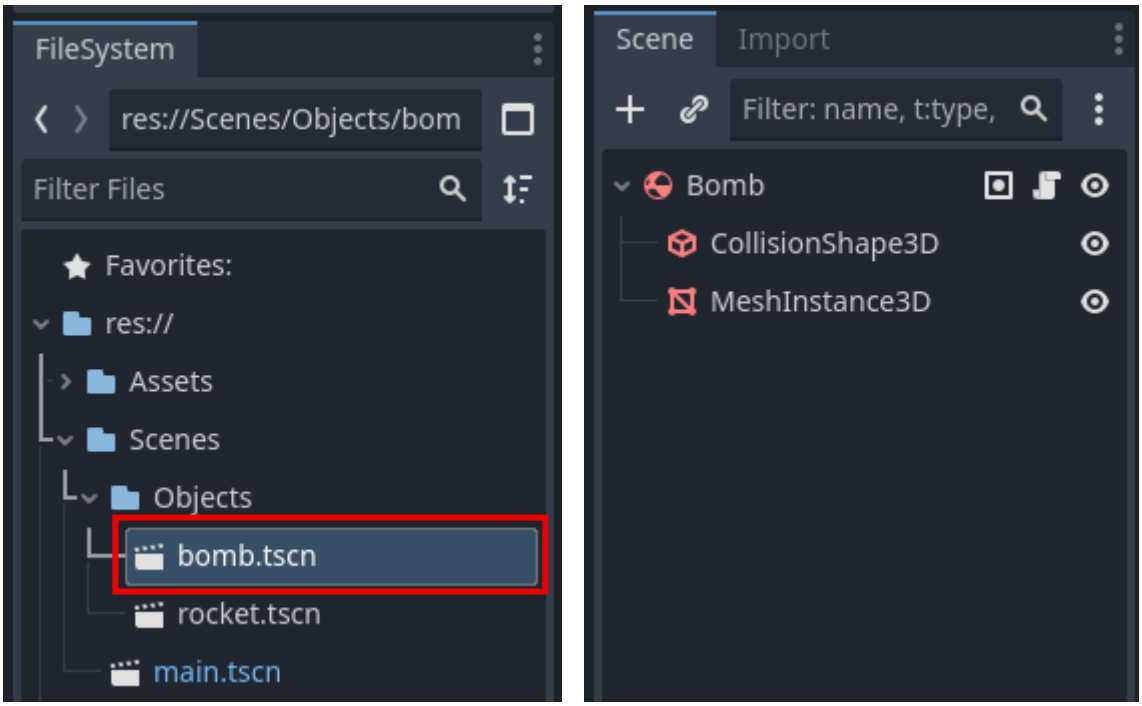
Check in with a Code Sensei before moving on. Make sure the **Fire** particle effect is set up properly.

- ☐ *How did adding a texture improve the look of the particle?*
- ☐ *What did changing the velocity of the particle change?*
- ☐ *What other differences were there between the fire and smoke particles. Why do you think those differences exist?*

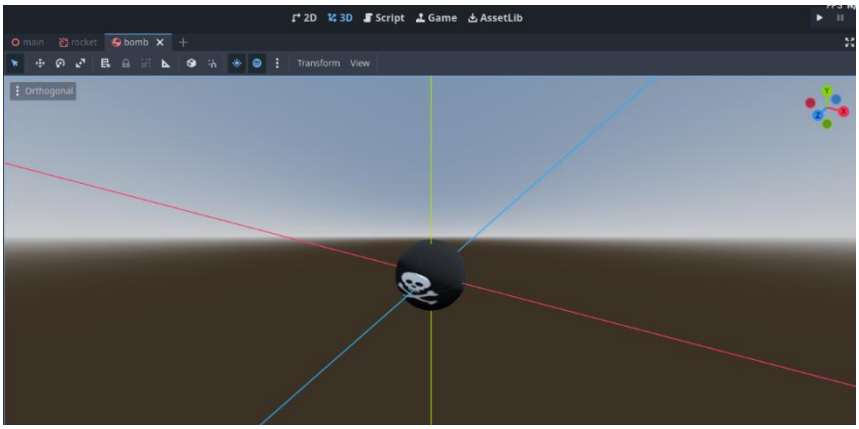
Reminder: Save your work!

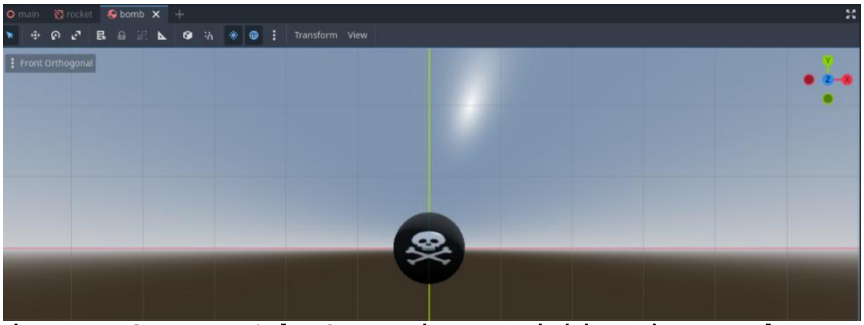
44 Give the bomb explosion particle effects!

In **FileSystem**, open the **Bomb Scene** found under **Scenes > Objects.**



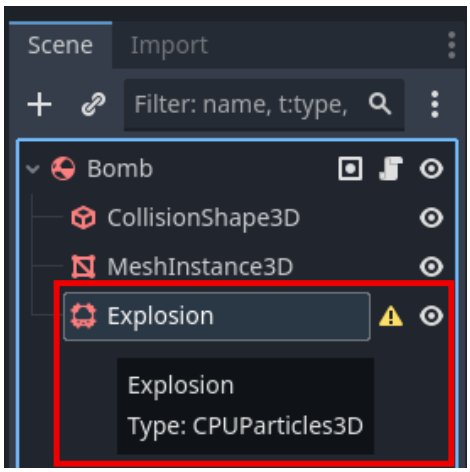
45 In the **3D Workspace**, use the blue **Z** nodule on the orientation gizmo in the top right of the workspace to look down the Z axis.



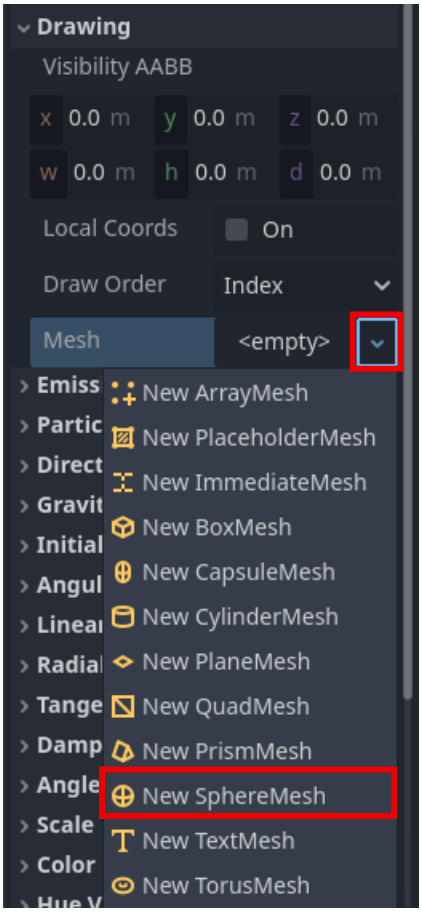


46 In **Scene**, add a new **CPUParticles3D** node as a child to the **Bomb**. Rename it **Explosion**.

There will be an error icon next to the node because a mesh and a material have yet to be assigned. Don't worry; this will get fixed in the following steps.

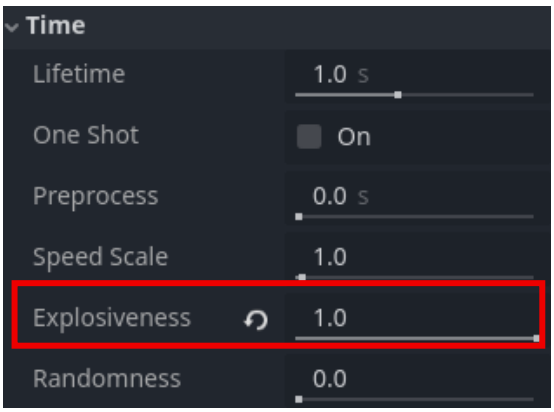


47 In the **Inspector** for **Explosion**, under **Drawing**, next to **Mesh**, click the drop-down arrow and select **New SphereMesh**.



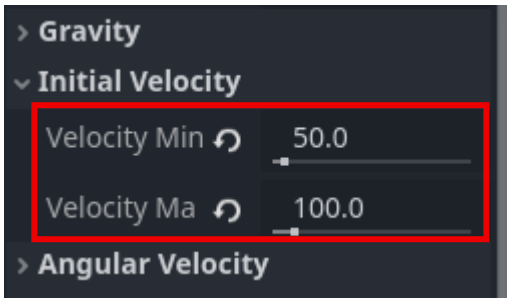
48 Add a **New StandardMaterial3D** to the mesh. Then, under Material, change the **Albedo** to a dark gray.

49 Under Time, set **Explosiveness** to **1**. **Explosiveness** determines the delay between particles being emitted.



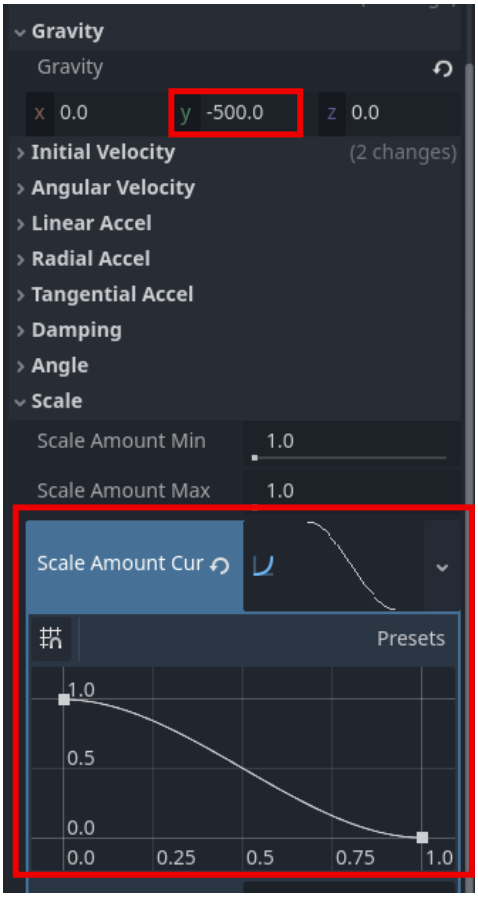
50 In **Inspector**, under **Direction**, set **Spread** to **180**. This sets the particles to emit outward from the bomb, instead of emitting them in a straight line.

51 Under **Initial Velocity**, Set **Velocity min** to **50** and **Velocity max** to **100.** This will give the particles a random velocity between 50 and 100, creating a more realistic explosion effect.



52 In **Inspector**, set the **Gravity y** to **-500.**

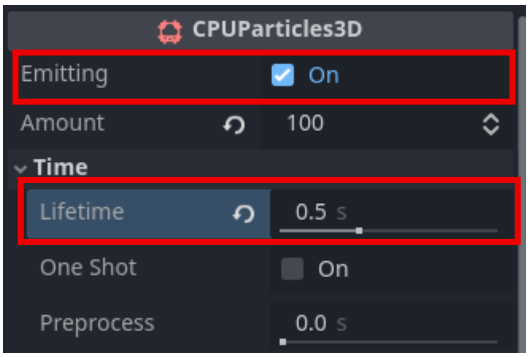
Then, create a **Scale Amount Curve** like the ones created for the **Smoke** and **Fire** particles.



Refer to **step 39** for additional guidance.

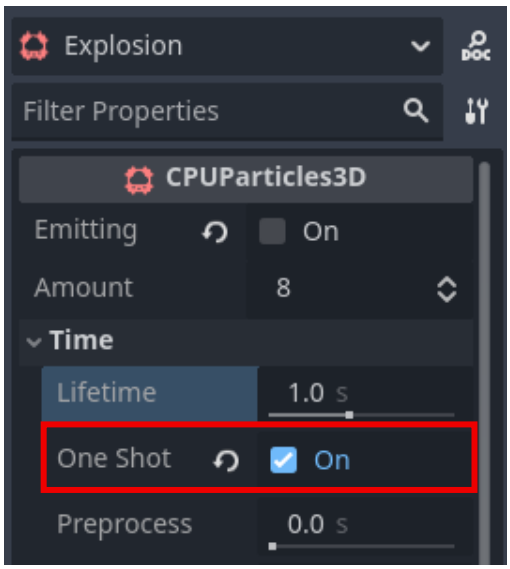
53 In **Inspector**, set the following properties of the **Explosion** node to the specified values:

- **Amount** to **100**
- **Lifetime** to **0.5**



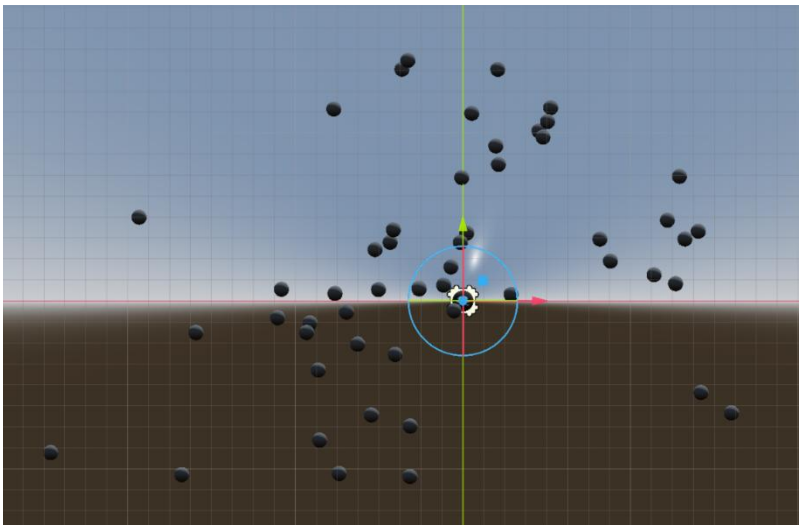
Refer to steps 19 and 27 for additional guidance.

54 In **Inspector**, under **Time**, toggle **One Shot** to **On**.



55 Keep the Inspector open and go to the 3D workspace. Toggling **Emitting** on will simulate the particles to explode.

Compare with the image below to make sure the explosion is going off correctly.



Pause for **Sensei Stop #4**!

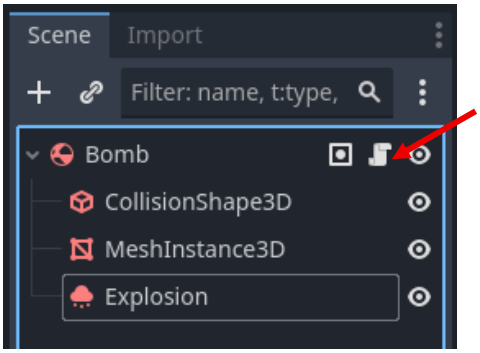
Check in with a Code sensei before moving on. Make sure the **Bomb Explosion** particle effect is set up properly.

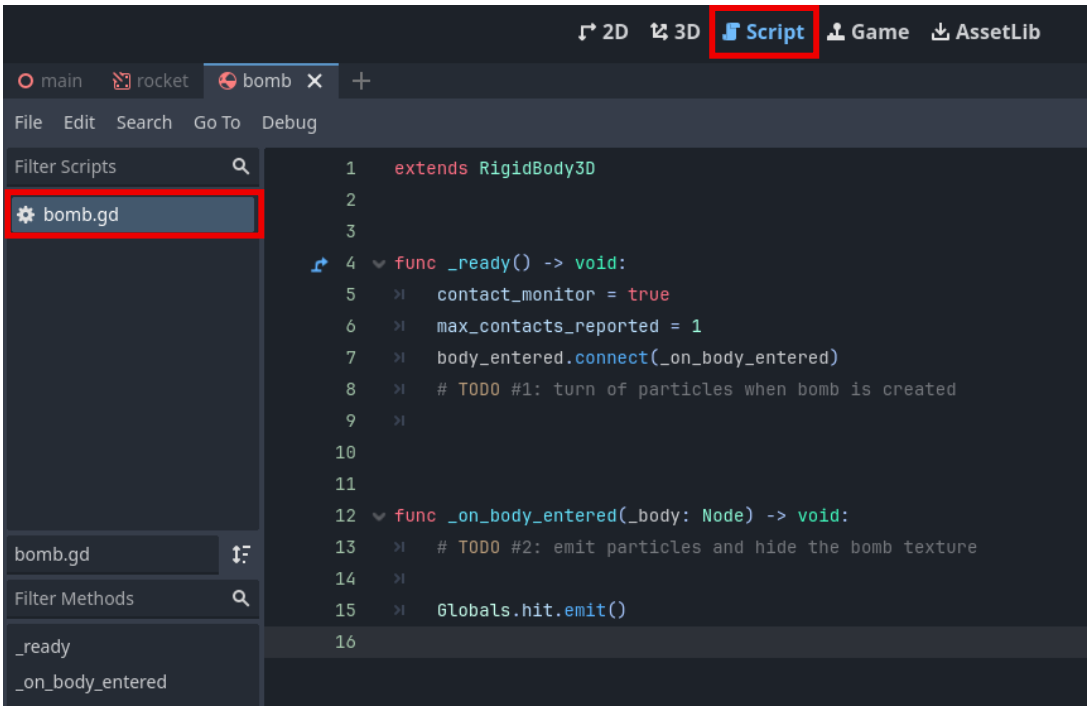
Reminder: Save your work!

56

Script the explosion particles to emit at the correct times!

In **Scene**, click the script icon next to **Bomb** to open the bomb.gd script.





57 Find **TODO 1** in the **bomb.gd** script.

On the next line, reference the **Explosion** child node using `get_node()`, then set its `emitting` property to `false`.

This will set the bomb to not emit particles when the game starts.

```
func _ready() -> void:
    contact_monitor = true
    max_contacts_reported = 1
    body_entered.connect(_on_body_entered)
    # TODO #1: turn of particles when bomb is created
    get_node("Explosion").emitting = false
```

58 Find **TODO 2**.

On the next line, reference the **Explosion** child node using `get_node()`, then set its `emitting` property to `true`.

On the next line, reference the **MeshInstance3D** child node using `get_node()`, then set its `visible` property to `false`.

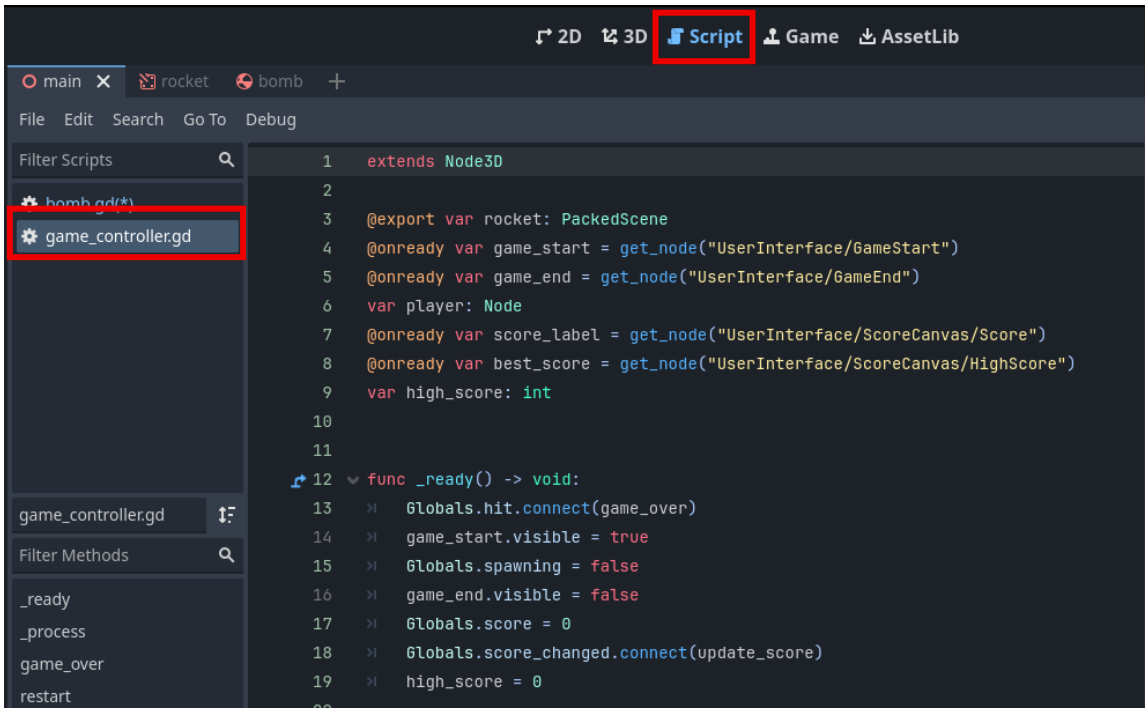
```
12
13 func _on_body_entered(_body: Node) -> void:
14     # TODO #2: emit particles and hide the bomb texture
15
16
17     Globals.hit.emit()
18
```

59 Check the code and update the script as needed.

```
func _on_body_entered(_body: Node) -> void:
    # TODO #2: emit particles and hide the bomb texture
    get_node("Explosion").emitting = true
    get_node("MeshInstance3D").visible = false
    Globals.hit.emit()
```

This will make the explosion particles emit and make the bomb invisible when the bomb detects a collision.

60 In **FileSystem**, under **Scripts**, open **game_controller.gd** in the **Script** workspace.



61 Find **TODO 3**. On the next line, add an `await` keyword. Then, chain together a call to the method `get_tree()`, a call to the method `.create_timer()` with a value of `1`, and a signal to `.timeout`.

Reference the timeout line in the `_process()` method for details

```
32
33  func game_over():
34      # TODO #4: destroy the player before timer so the game can't continue
35
36      # TODO #3: create timer so you can see the explosion
37
38      game_end.visible = true
```

62 Check the code and update the script as needed, then find **TODO 4**.

Move the line of code that will **remove the player from the scene** below the **TODO 4** comment.

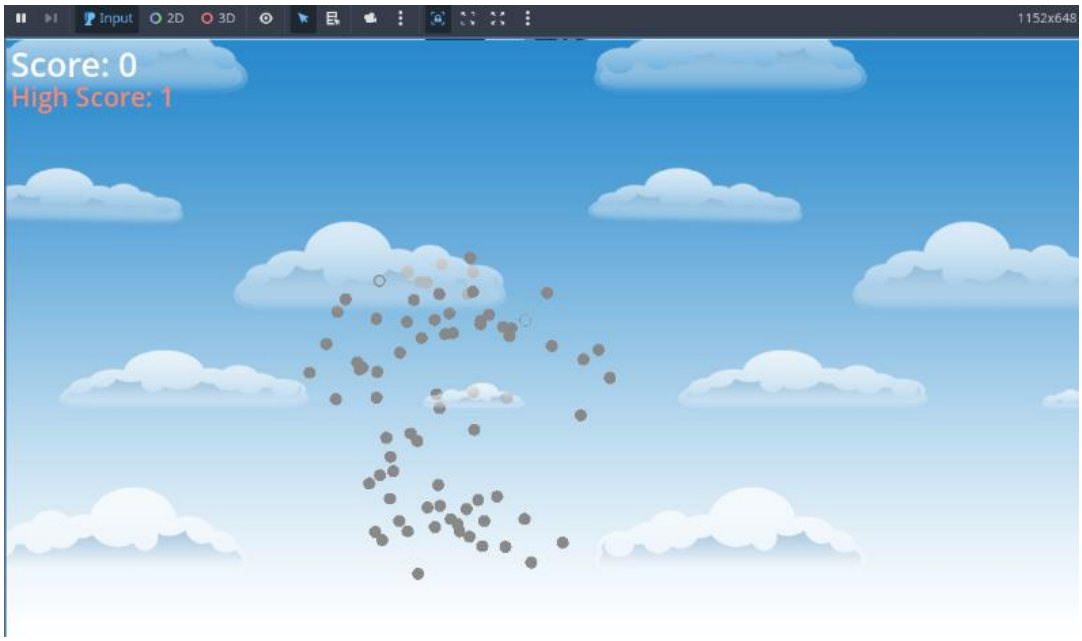
```
32
33  func game_over():
34      # TODO #4: destroy the player before timer so the game can't continue
35
36      # TODO #3: create timer so you can see the explosion
37      await get_tree().create_timer(1).timeout
38      game_end.visible = true
39      Globals.started = true
40      Globals.spawning = false
41      player.queue_free()
42      if Globals.score > high_score:
43          high_score = Globals.score
44          update_high_score(Globals.score)
45      Globals.score = 0
46      update_score(Globals.score)
47
```

63 Check the code and update the script as needed.

Notice the `player.queue_free()` has been moved from above the if-statement to below **TODO 4**.

```
32
33  func game_over():
34      # TODO #4: destroy the player before timer so the game can't continue
35      player.queue_free()
36      # TODO #3: create timer so you can see the explosion
37      await get_tree().create_timer(1).timeout
38      game_end.visible = true
39      Globals.started = true
40      Globals.spawning = false
41      if Globals.score > high_score:
42          high_score = Globals.score
43          update_high_score(Globals.score)
44      Globals.score = 0
45      update_score(Globals.score)
46
```

64 Playtest the game.



Notice that when a Bomb collides with the Rocket, the Bomb's explosion particles emit and then the game ends.

Pause for **Sensei Stop #5**!

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

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

- What did you learn about Particles?
- Which project in Bronze Belt did you most enjoy creating and why?
- What is something that could further improve the Dropping Bombs project?
- Which particle effect setting are you interesting in learning more about?

Reminder: Save your work!