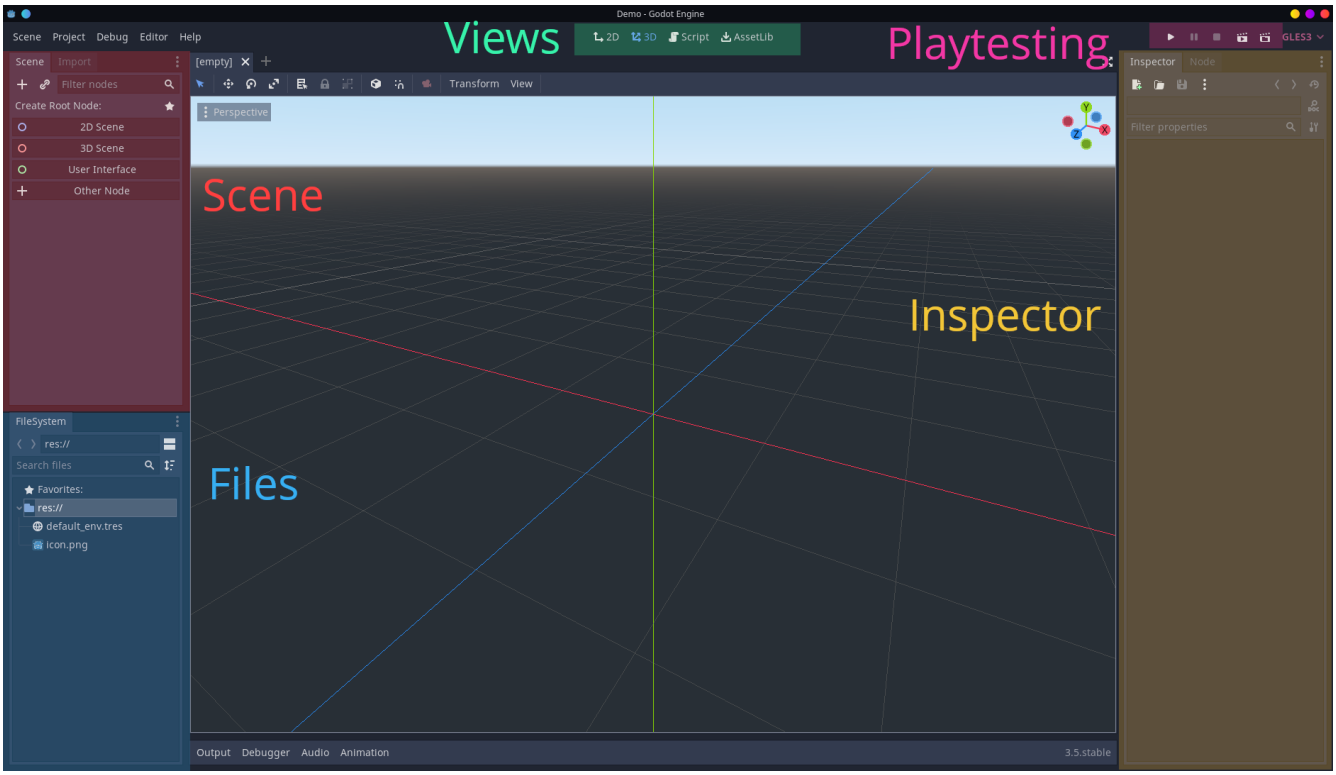
Getting Started

Let’s start with a quick tour of what you’ll mainly work with.



Scene
Every object in your game (player character, projectiles, items, props, levels etc.) is a scene. In the scene panel, you can add nodes by pressing on the + icon. Nodes give the scene additional attributes (e.g. collision). You can also add other scenes into the current scene as a node (e.g. adding the player scene into a level scene).

File System
This is where all the files used in the game are located. This is the same folder you chose when creating the project at the start. For example, if you saved the project in C:/godot/project, then res://icon.png can be found at C:/godot/project/icon.png.

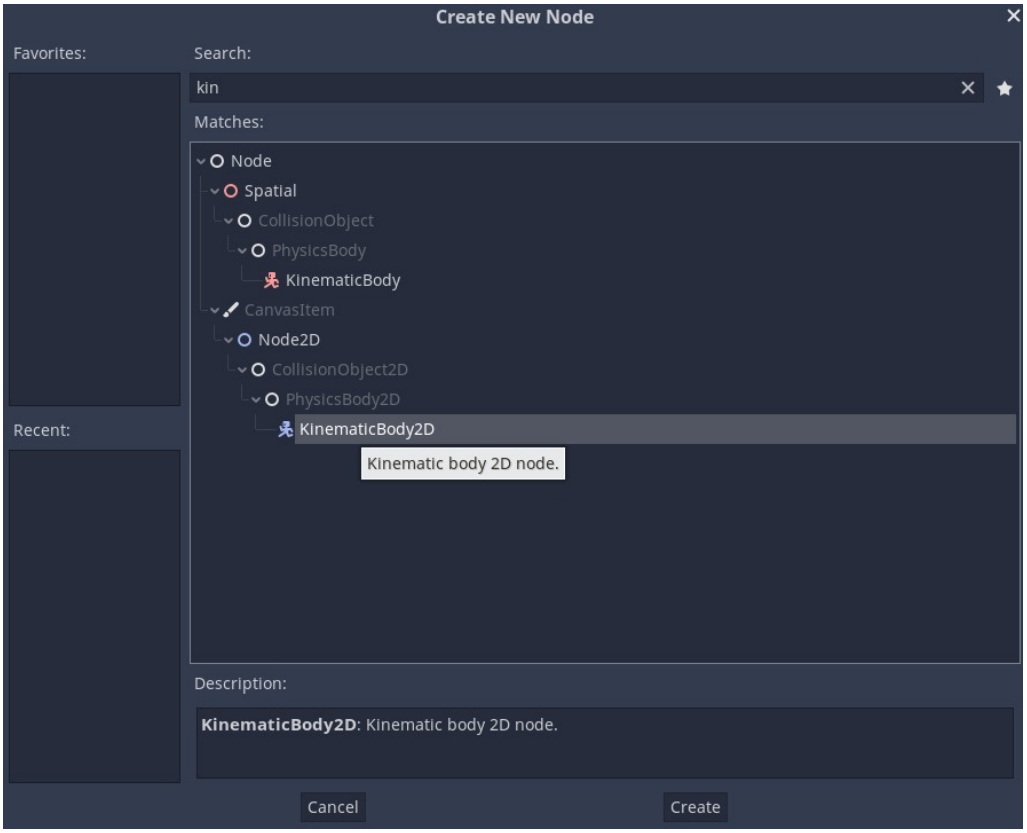
Views
This is used to change what is shown on the main panel (which is the 3D view in the screenshot above). For example, clicking on “Script” will bring you to the built-in text editor for you to write code.

Playtesting
These are self-explanatory. They open up a new window where your game can be played.

Inspector
This is where specific properties of a node (e.g. transform) can be modified.

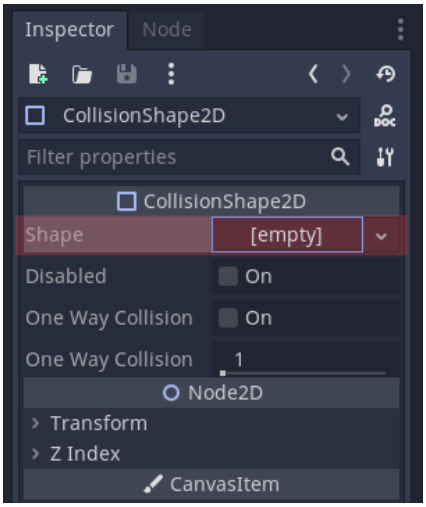
Making an Actor

In the scene panel, select “Other Node”. Use the search bar to find KinematicBody2D.



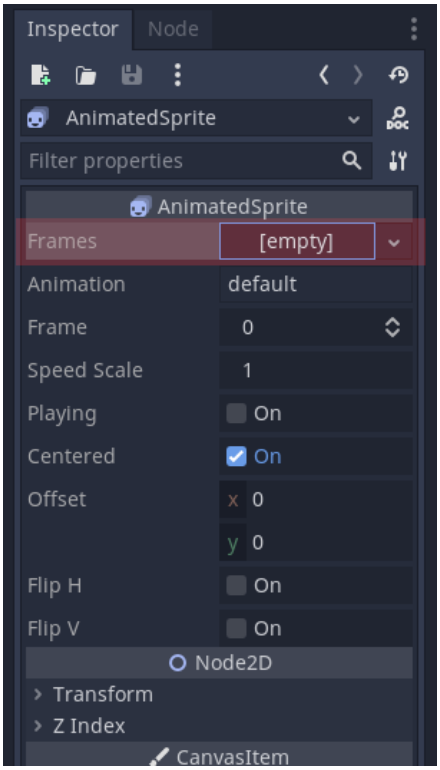
You will see a warning sign that says the KinematicBody2D requires another node to give it a shape. Click on the + icon in the scene panel, then search for CollisionShape2D.

There will be another warning sign on CollisionShape2D. Click on CollisionShape2D in the scene panel, then go to the inspector panel. Currently, the Shape property is [empty], so click on that and choose one of the options listed in the dropdown menu. Now, your kinematic body has a shape.

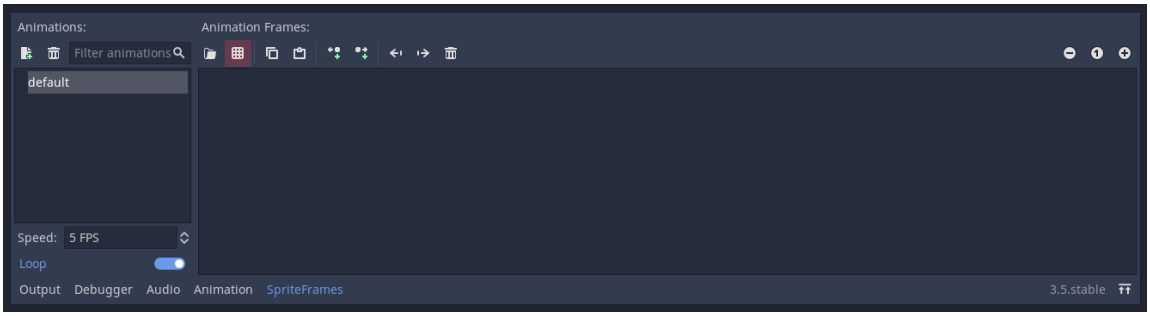


Animation

Add an AnimatedSprite node to the KinematicBody2D. Over at the inspector panel, the Frames property is set to [empty], so click on that and select New SpriteFrames. After selecting, click it again to open up a new panel at the bottom.

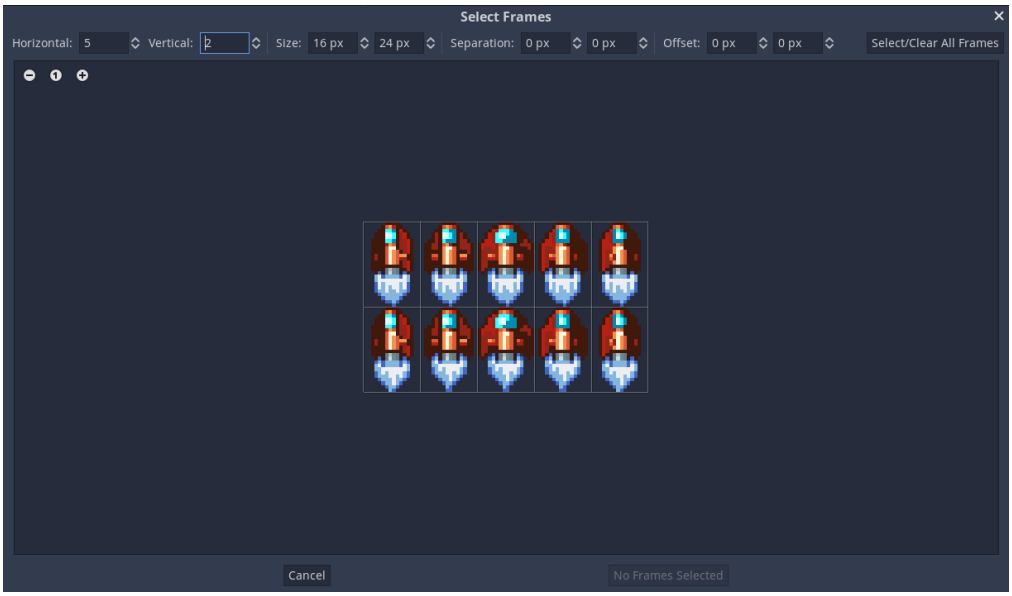


Select the second icon from the left to add animation frames from a spritesheet. This will bring up a new window.

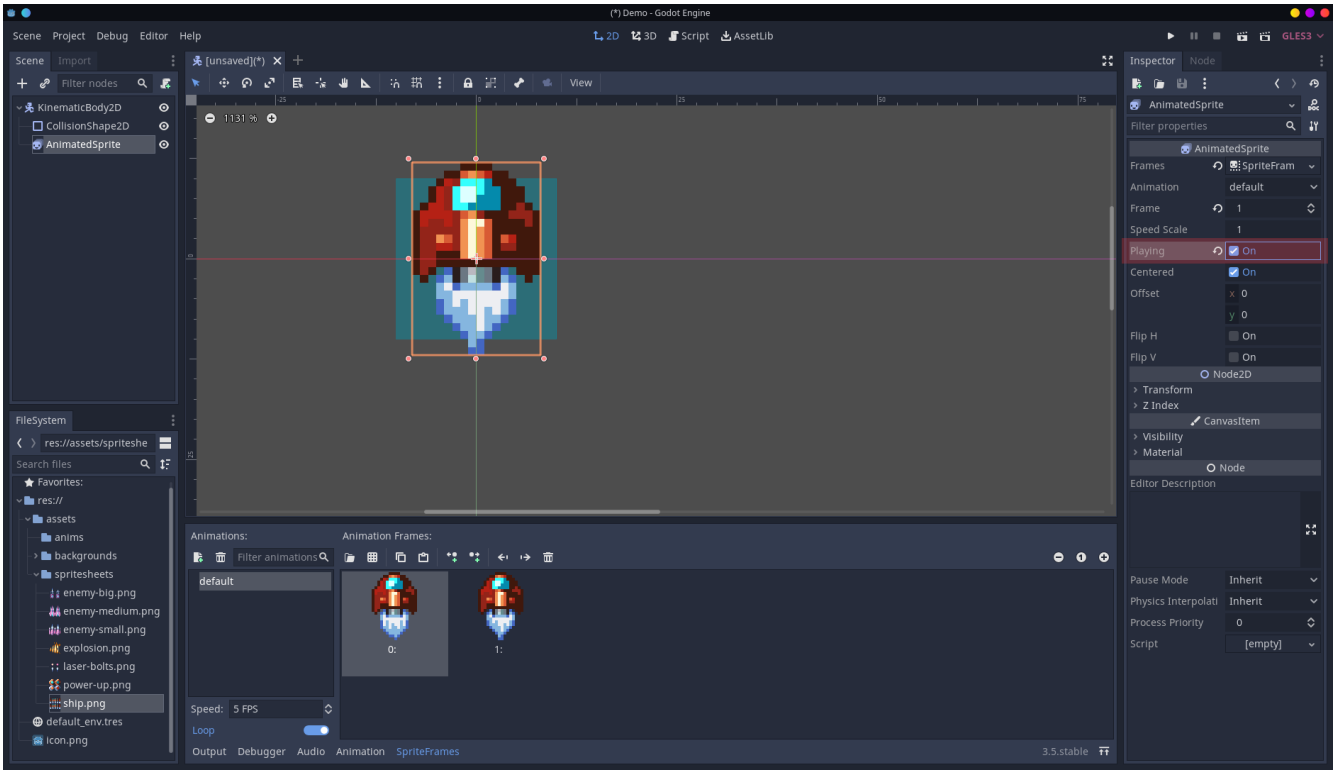


Godot works well with spritesheets that follow a regular grid. By changing the values at the top, Godot can properly extract the individual sprites from the spritesheet. For spritesheets that use an irregular structure in order to conserve space, you will have to use another software to extract the sprites and import them as standalone images.

You can use the + and – buttons on the top left to zoom in and out respectively.



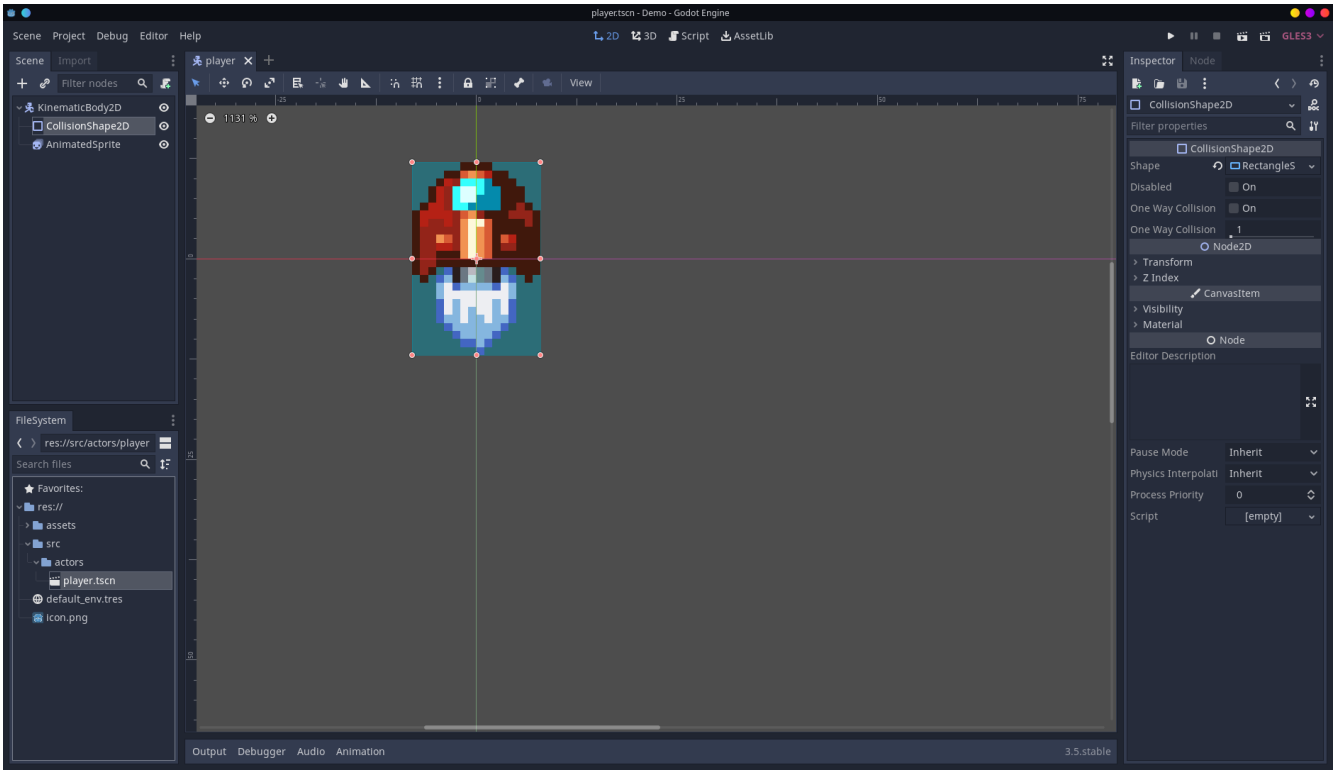
Once you have selected the frames you want to use for the animation, make sure to go back to the inspector and enable the Playing property for the AnimatedSprite.



Resize the CollisionShape2D by dragging around the four corners such that it matches the sprite.

You can add a script by right-clicking on KinematicBody2D and selecting Attach Script. Code logic will be explained in comments within the code files, instead of here.

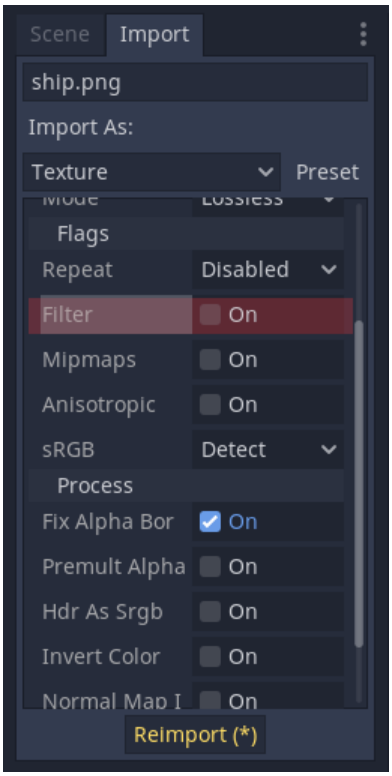
Now, save the current scene. You may want to create a dedicated folder for actors.



Removing the Filter

By default, Godot imports assets with an anti-aliasing filter enabled, which makes pixel art look muddy. It may also cause sprites that fit exactly in its cell in the spritesheet to “leak over” into the next cell.

To fix this, locate the asset of interest in the file system, then click on the Import tab on the scene panel. Uncheck the Filter option, then click on Reimport at the bottom. Any sprites from this particular asset will be automatically updated, but remember to reimport the other affected assets.



Project Settings

You can access the project settings by clicking on Project on the top left, then selecting Project Settings.

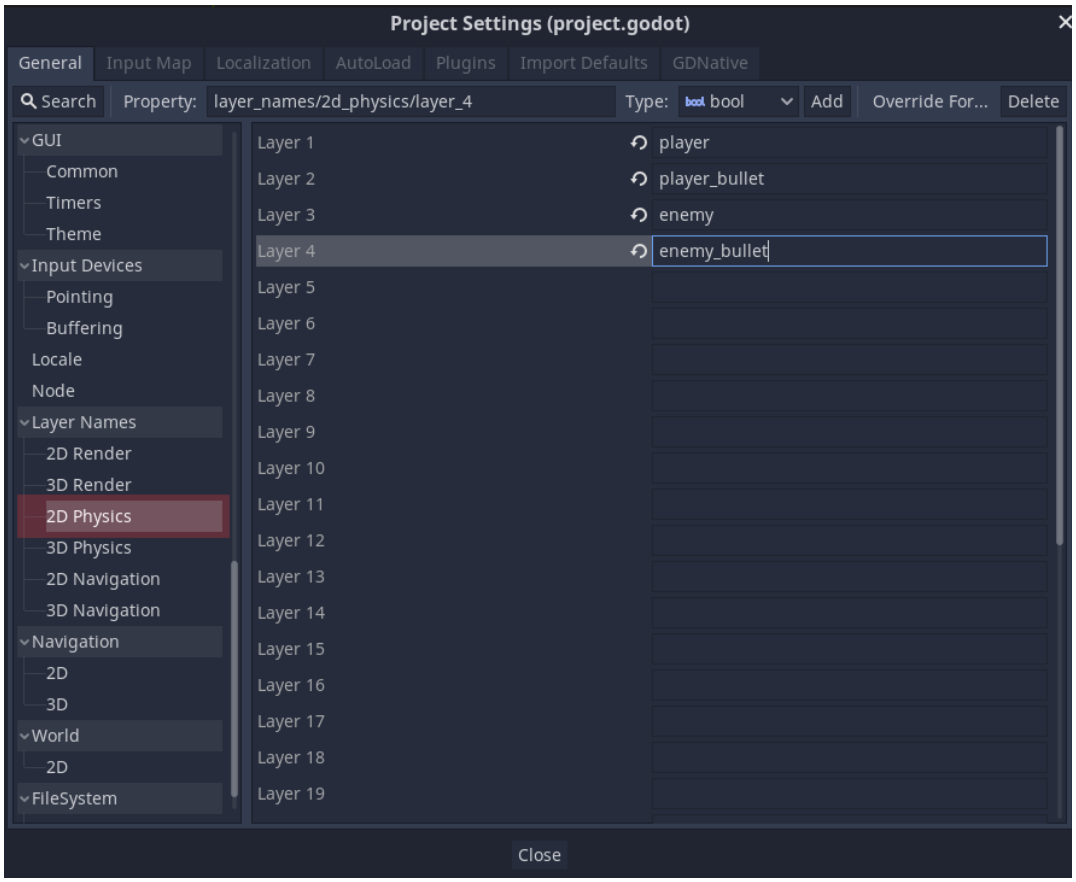
Input Mapping

This is accessed from the Input Map tab. You can add inputs by typing into the Action field at the top. Then, you'll see your new inputs when you scroll down.

Pressing the + icon allows you to select which key/button you want to map the input to.

Physics Layers

Returning to the General tab, scroll down to Layer Names and select 2D Physics. Here, you can name the layers.



Physics layers allow Godot to more efficiently check for collision, since it only has to check for collision between specified layers. This can be configured through the inspector.

Layer refers to which layer(s) the collider belongs to, while Mask refers to the layer(s) that should be checked for collision. In the example below, the actor belongs to the player layer (1), and checks for collision on objects belonging to the enemy (3) or enemy bullet (4) layers.


Inspector
Node
KinematicBody2D
Filter properties
KinematicBody2D
Collision
Motion
Moving Platform
CollisionObject2D
Collision
Layer
1 2 3 4 9 10 11 12
5 6 7 8 13 14 15 16
Mask
1 2 3 4 9 10 11 12
5 6 7 8 13 14 15 16
Input
Node2D
Transform
Z Index
CanvasItem
Visibility