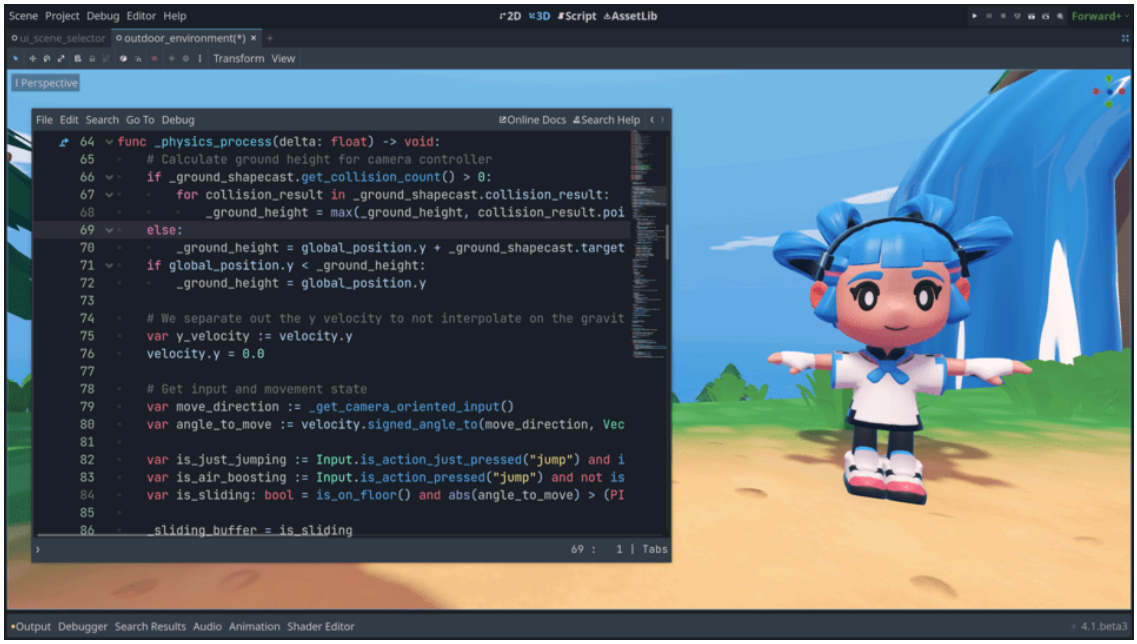
Title options:

- Godot 4.1 is out. Meet the bug buster that packs a punch.
- Godot 4.1 is here, smoother, more reliable, and with plenty of new features.

Four months after the release of Godot 4.0 in March 2023, we are happy to bring you Godot 4.1, an update that follows our pledge with a focus on stability, performance, and polish.

As always, this version comes with welcome new features, like the improved AI avoidance and the ability to detach script editors and put them on other monitors.



Still, we took great care to prioritize the bugs you encountered in Godot 4.0. This update fixes over 1,000 of them and should feel more reliable overall. We will keep this effort going throughout the year to make Godot one of the most stable general game engines out there.

If you wish to get straight into action, you can download Godot 4.1 now.

This release was made possible by our dedicated core development team and the help of over 300 contributors! We warmly thank you all for your work and dedication.

For an exhaustive list of all the bug fixes and improvements, head on over to our interactive changelog.

Once again, you can get the main highlights in this video.

https://youtu.be/PAtG_fHhIx8

Giving back

As a community effort, Godot relies on individual contributors to improve. In recent years, user and company donations allowed us to also hire a number of core contributors to work full-time on the engine and bring you these fast-paced releases.

Our monthly expenses remain higher than monthly donations, and we still depend on large one-time company donations to fund development. Currently, we need a lot more

monthly donations to keep up the pace with Godot 4 updates, not to mention the need to hire more maintainers to review every contribution.

Besides financial support, you can also give back by: writing detailed bug reports, contributing to the code base, writing documentation, writing guides (for the docs or on your own website), and supporting others on the various community platforms by answering questions and providing helpful tips.

Last but not least, making games with Godot and crediting the engine goes a long way to help make it more popular and convince more people to contribute and improve Godot for everyone. Remember, we are all in this together and Godot requires community support in every area to thrive.

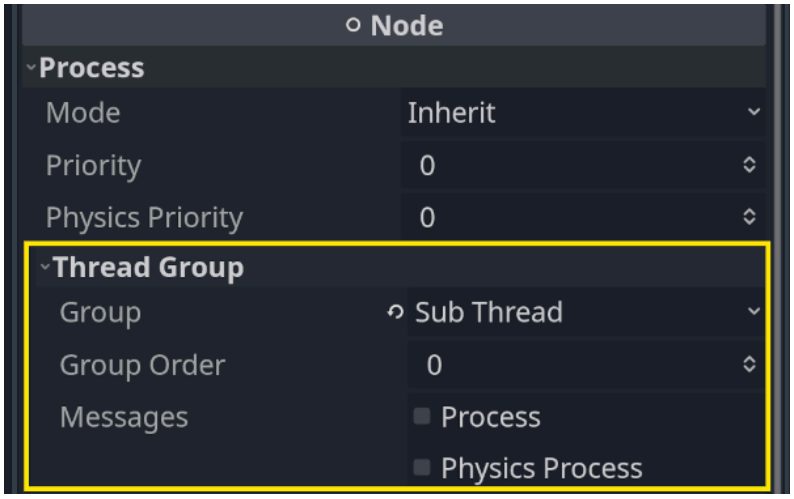
Without further ado, here's a breakdown of the main changes and new features.

Performance

Godot games are built as a tree of nodes, which are the engine's base building block for game entities. Adding and removing nodes are operations the engine needs to do extremely often, so they need to be as fast as possible.

In 4.1, we changed the algorithm to use a fast hash map to make adding and removing child nodes multiple times faster. Uncommon node operations are slightly slower as a result, and the memory footprint of the base Node class is 10% higher, but this is a small and necessary trade-off for a big benefit to all Godot users, especially to more complex projects with a lot of node manipulation.

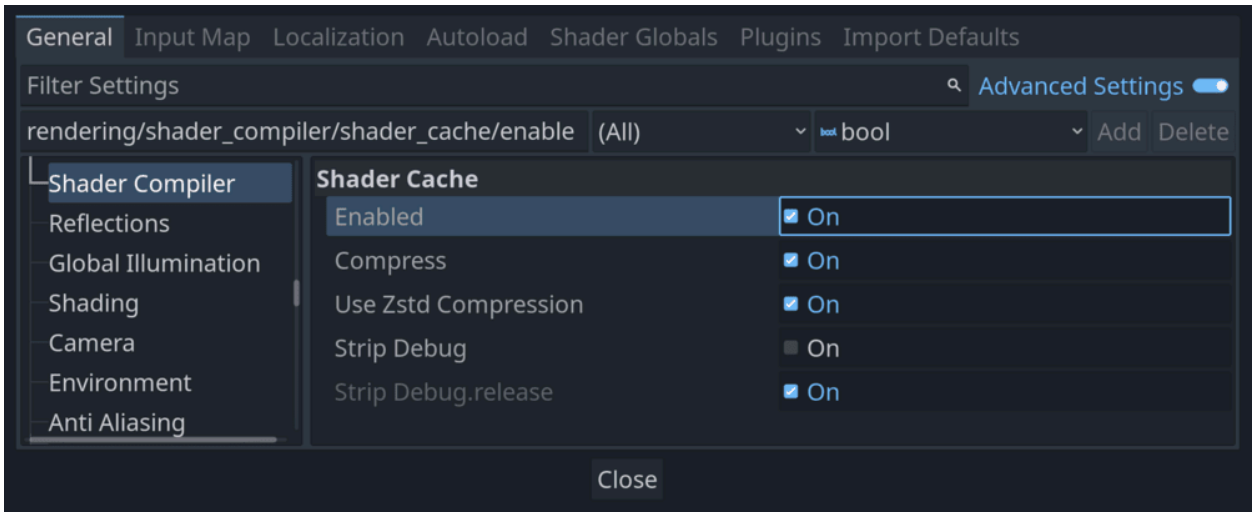
This version also introduces experimental multithreading for your scenes, a feature kickstarted by Juan Linietsky. New node properties give you full control over how nodes get processed, sequentially or in parallel.



As mentioned, the feature is currently marked experimental as it needs extensive testing to find edge cases. We do not recommend using it in production yet. But it sets the foundation for making the most of modern hardware, with simple controls.

Also on the multithreading front, Pedro J. Estébanez worked hard on fixing a large amount of multi-threaded loading issues and limitations.

On the rendering side, the Vulkan renderer got a shader cache to help prevent stalls due to shaders compiling at runtime. After the first compilation, the shader will be pulled directly from the cache. This cache can be toggled off in the project settings and is on by default in Godot 4.1.

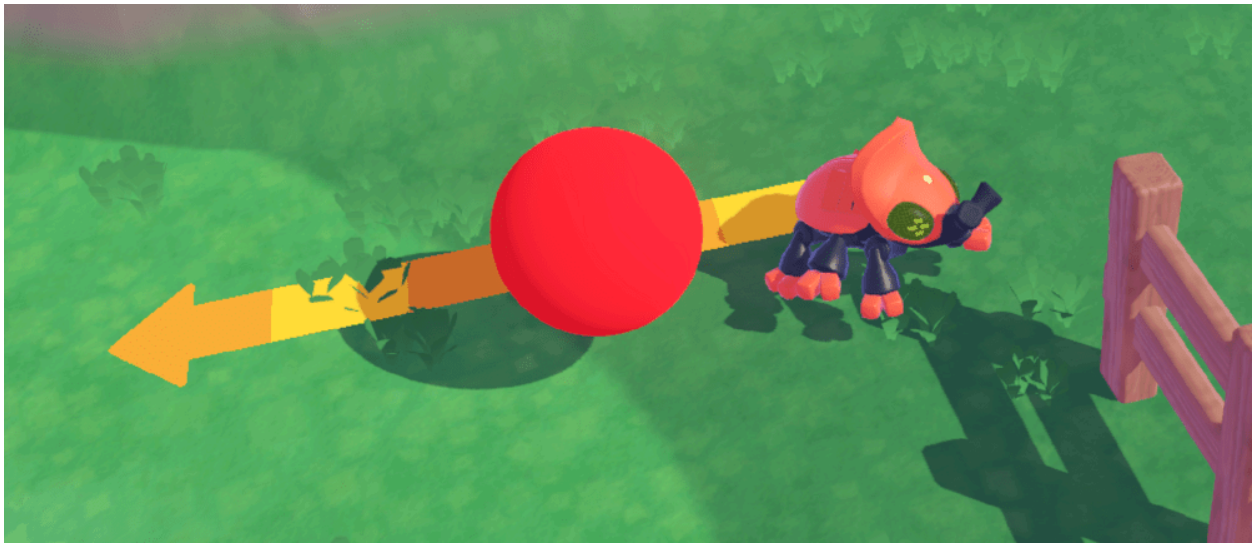


This improvement doesn't address all possible causes of stalls, but it's a first step in the right direction by Alexander Streng. We'll keep working on it in upcoming releases.

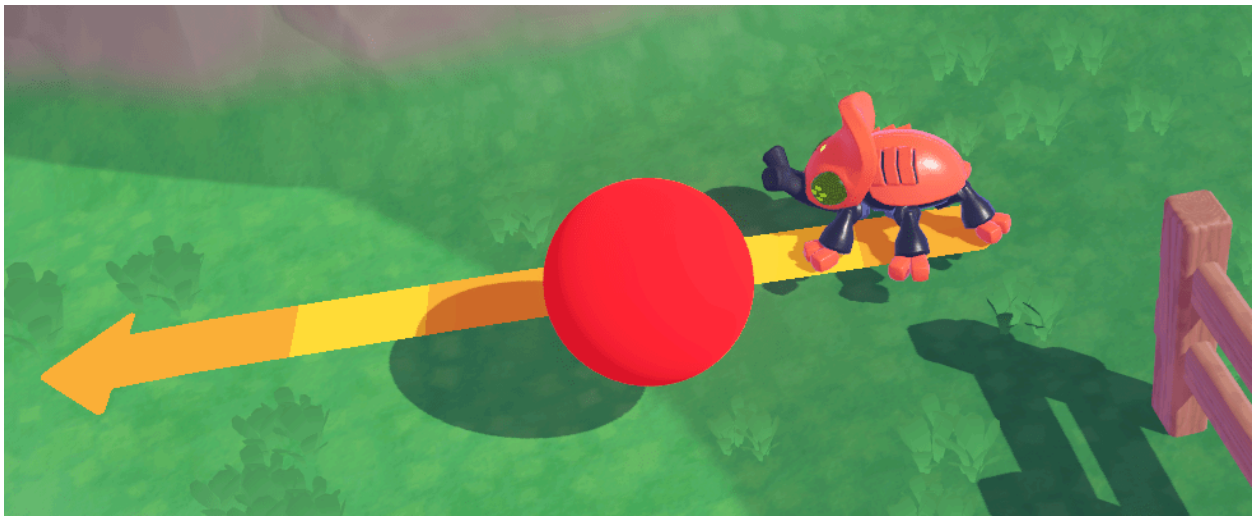
More work will follow on the performance front throughout the year.

Core

When importing models into Godot, there was often the problem that they ended up facing backward.



Juan Linietsky, Tokage, and Aaron Franke addressed the issue by swapping the front and back camera directions in the editor. Also, the `look_at()` function now has an argument to use the model space as the reference for looking forward instead of the camera's minus z-axis. These changes also help fix a long-standing bug with path following.



This update also brings frame delta smoothing to Godot 4, by lawnjelly. This can significantly improve the fluidity of motion and give smoother gameplay. This option is enabled by default, though please note that it only works when VSync is also enabled.

Scripting

GDScript

Until now, in GDScript, you needed to use a resource or a Singleton to share data between multiple instances of the same script.

Thanks to George Marques, you can now create and use static variables instead. Static variables store data on the class instead of each instance, so they're shared between every instance of the class.

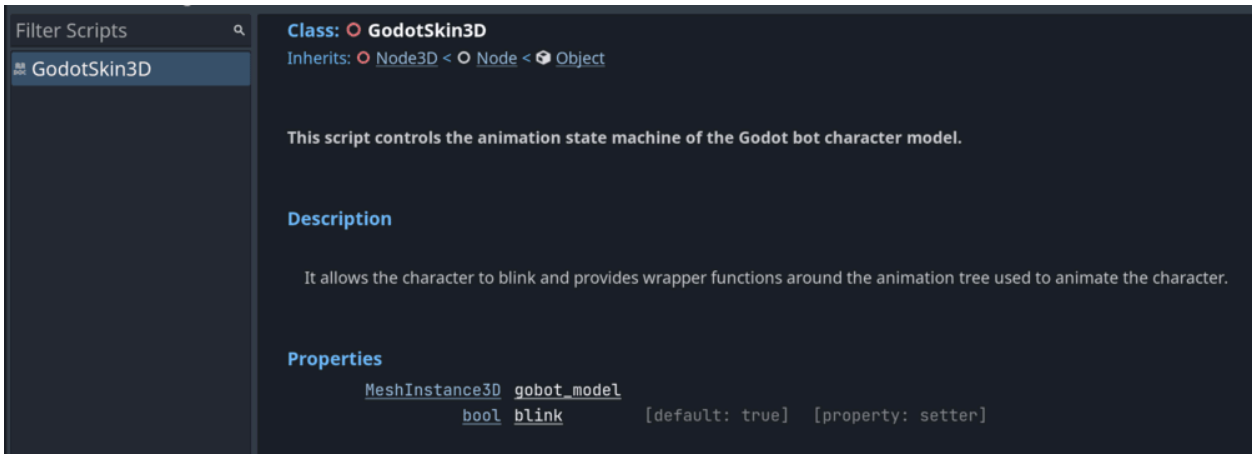
To make a variable static, add the static keyword in front of a variable defined at the top of your script.

```gdscript
static var active_instance_count := 0

func _enter_tree():
    active_instance_count += 1

func _exit_tree():
    active_instance_count -= 1
```

A great feature of GDScript added in Godot 4 is the automatic generation of documentation pages for your named classes.



This version includes a rework of the system by ocean that now treats enumerations as types, making the generated documentation more precise. You can also now use inline docstrings instead of having to always place them above a variable or a function's definition.

C#

The focus in this release was on bringing feature parity between C# and GDScript.

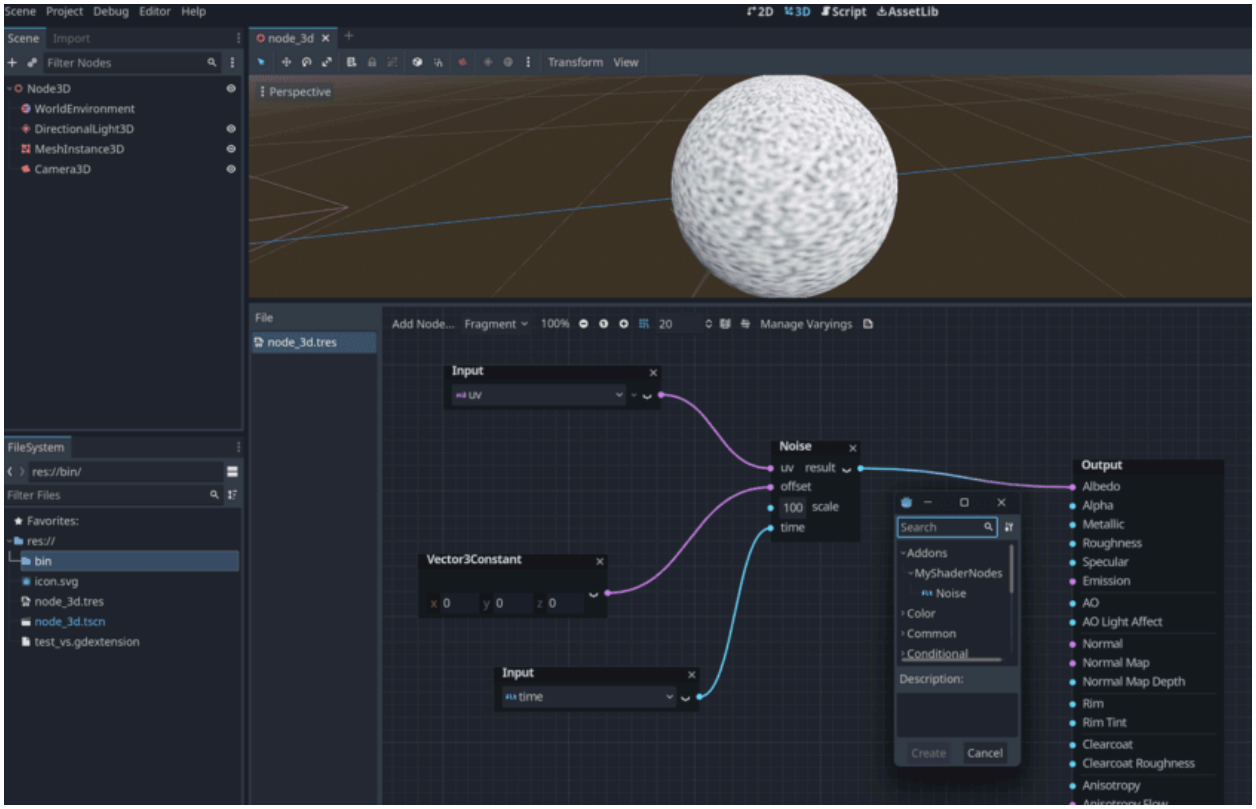
When using GDScript you can define a new node type to use in the editor by adding a global class name to your script.

From Godot 4.1 this is also possible in C# by adding the GlobalClass attribute to your file, thanks to Raul Santos and Will Nations. You can also use the Icon attribute to give your global class a unique icon.

GDExtension

Godot comes with a unique technology to use low-level languages like C++ as a game scripting language, without having to recompile the engine.

While the technology is still in beta for this release, GDExtension is now even closer to GDScript and C# in terms of scripting capabilities. You can now implement new visual shader nodes and create editor plugins with GDExtension.

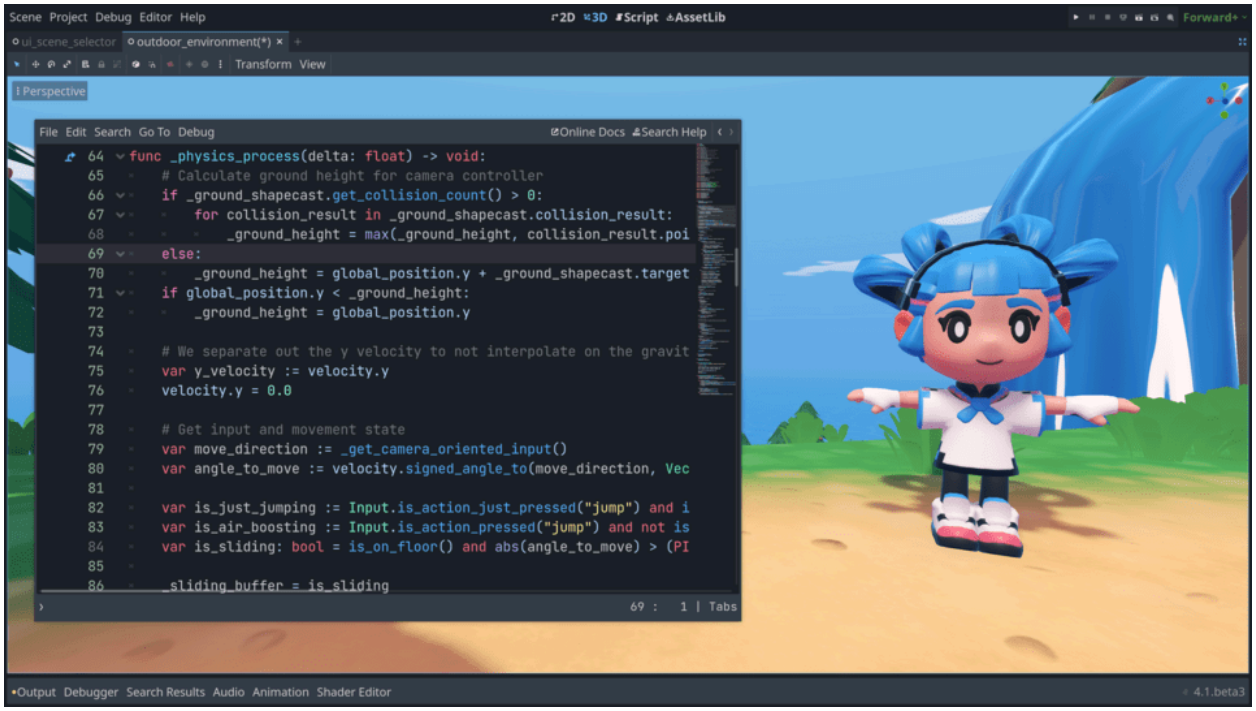


The team also implemented a backward compatibility system to help ensure that code written for Godot 4.1 keeps working even if the API changes in future releases. Finally, a lot of work was done on the architecture once again to make the GDExtension API extensible in the future.

All the above resulted from the teamwork of David Snopek, Juan Linietsky, RedworkDE, Yuri Rubinsky, and Yuri Sizov.

Editor

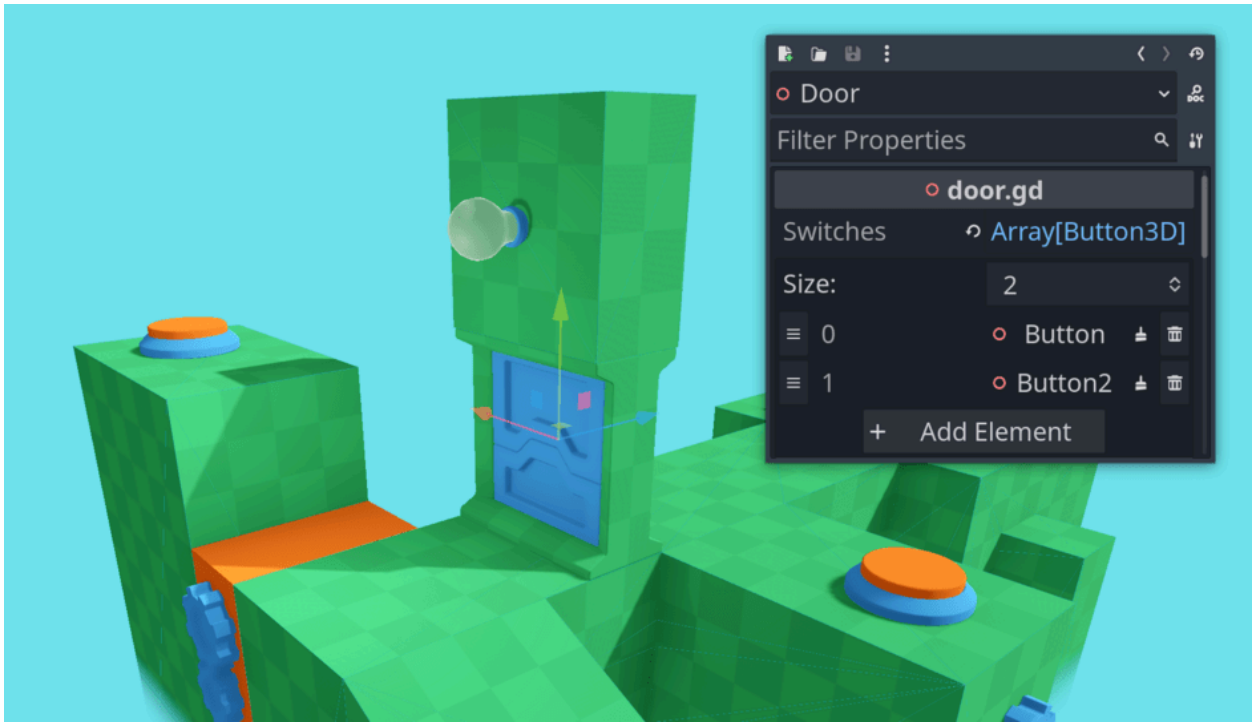
Godot 4 has support for multiple windows which we use for the project and editor settings and various pop-ups. Starting with this version, you can now also detach docs into floating windows and detach script editors, including the shader editor, and place them on a separate monitor.



In addition to that, the editor will now keep track of your window layout so that when you close and reopen the editor, you should often find yourself exactly where you left off.

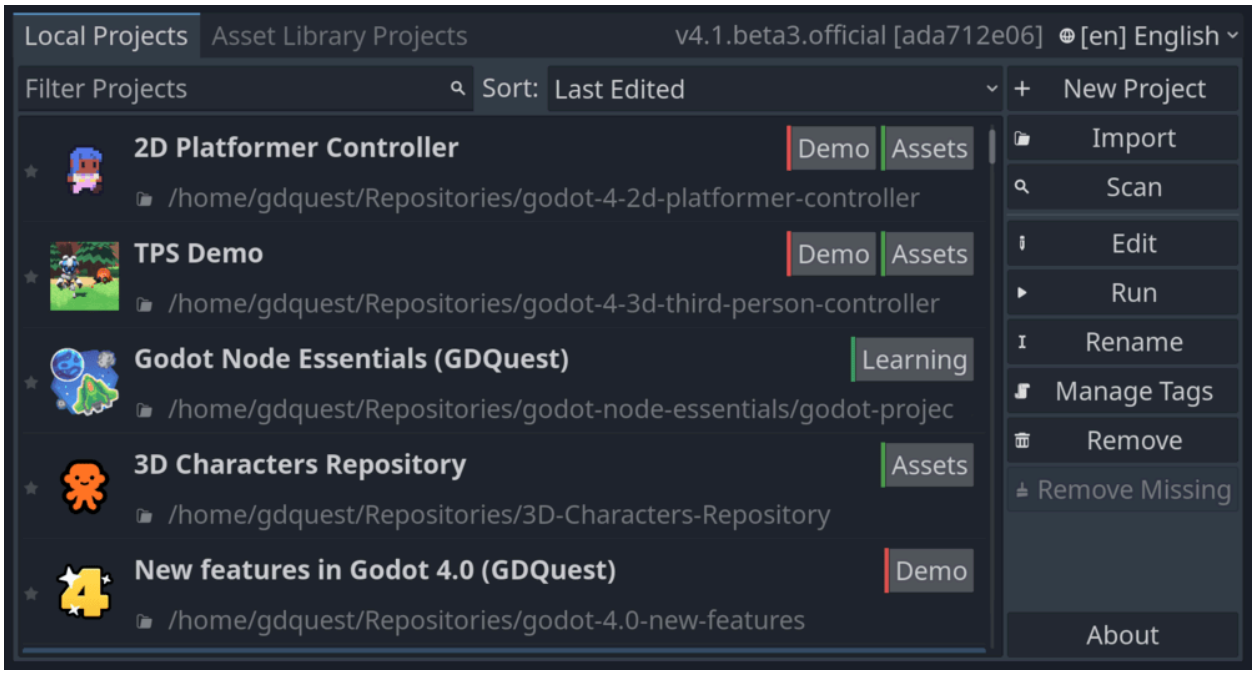
You can thank trollodel, Hendrik Brucker, and Tomasz Chabora for this.

Godot 4.0 introduced the option to define and export typed arrays, and to export individual nodes to the inspector, but it was not possible to combine the two. From Godot 4.1 onwards, you can export arrays of nodes to the inspector, which is great to link game objects together.



This work was done by Tomasz and [Timothe Bonhoure]

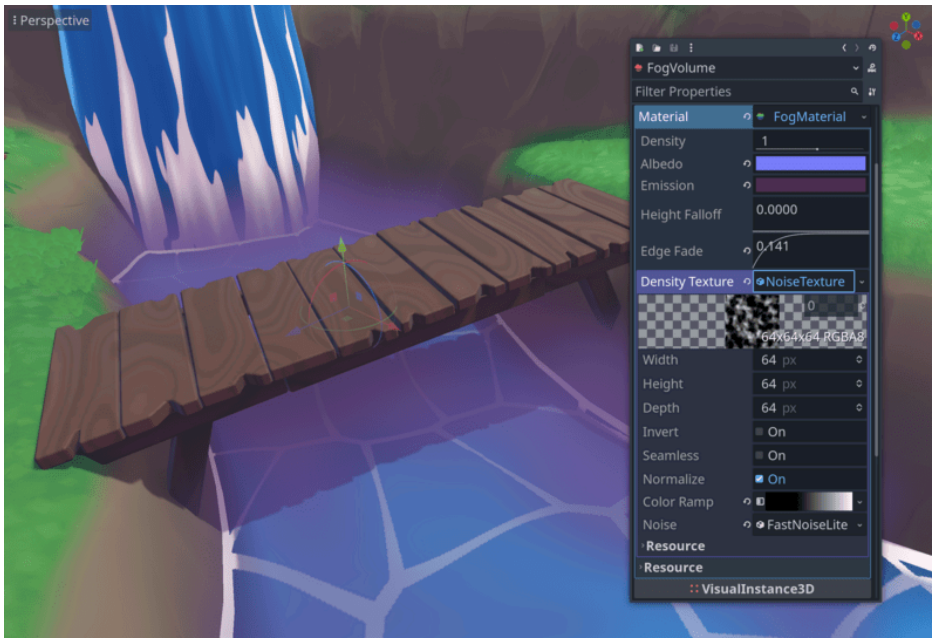
The project manager now allows you to assign tags to individual projects and filter projects by tags. It makes it much easier to search through dozens, if not hundreds of Godot projects.



Rendering

Particle turbulence got reworked in this version based on artists' feedback to offer greater creative control. Turbulence is used extensively to create these rich sprawling effects seen in many modern games. Big thanks to KdotJPG and Raffaele Picca for this contribution.

Using the new 3D noise textures you can control the density of volumetric fog easily, and make it thinner in certain areas.

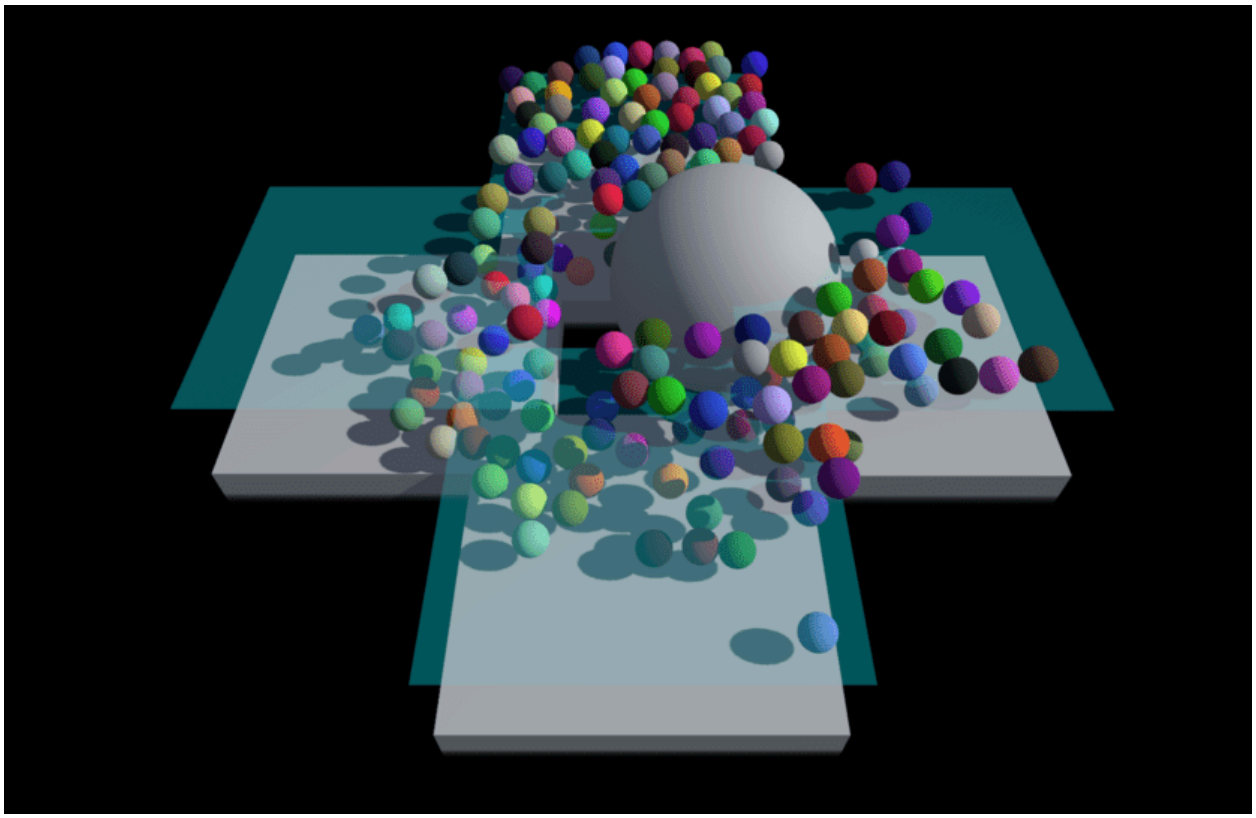


Work by Lasuch and Clay John.

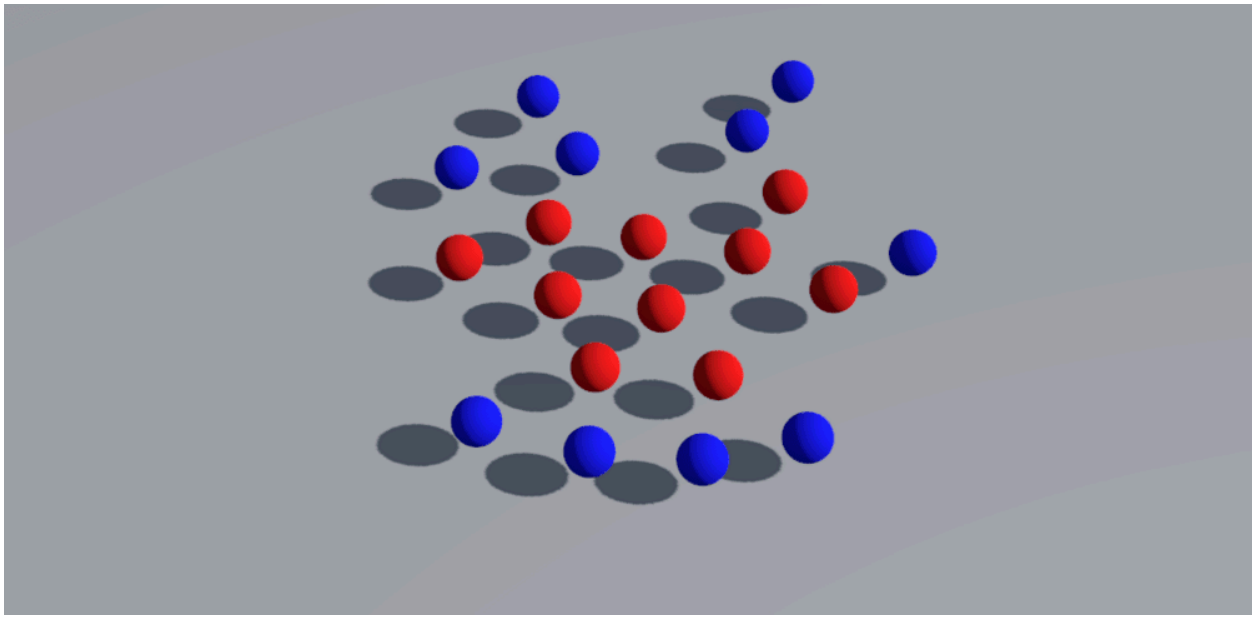
Navigation

Godot 4.0 introduced real-time avoidance for AI navigation but it was limited to a single plane.

This release includes completely rewritten avoidance algorithms by smix8 to give you much better behaviors and greater control. Avoidance can now happen in 2D or 3D, allowing flying agents to move over those walking on the ground.



On top of that, you can now use layers to control which agents avoid which and assign priorities to have some agents push others away.



Check out the updated documentation to learn more about how the improved navigation works.

On to the next release!

We already started work on Godot 4.2, which will come out in four months. We are dedicated to keeping up the pace and sticking to this reliable release cycle, so you never have to wait long for the next batch of improvements and new features.

Until then, enjoy Godot 4.1!