Godot Sandbox

https://github.com/libriscv/godot-sandbox

Who are we?

fwsGonzo: Alf-André Walla, PhD student at University of Oslo, Norway

fire: V-Sekai & Godot Engine contributor (https://github.com/fire - Open for work)

dragos: Dragos Daian (https://github.com/Ughuuu)

What does a scripting sandbox do? Part 1 of 2

Modding with User Generated Content (UGC)

Build once, run everywhere.

Can support many different programming languages.

Same performance on all platforms.
No surprises on mobile and console.

What does a scripting sandbox do? Part 2 of 2

Prohibits access to the host but allows normal game code to execute

Execution timeouts to stop scripts from looping infinitely.

Resource and memory limits.

Restrictions on classes and objects.

Why does Godot Sandbox use Riscv?

RISC-V (Pronounced risk-five) is a modern open standard instruction set (ISA)

Riscv64-linux is a linux based 64-bit operating system target

We execute the riscv64-linux target on desktop, mobile and web platforms

Godot Sandbox is low latency

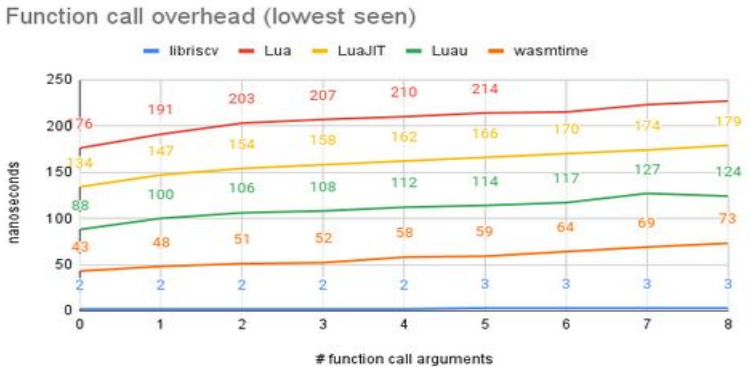
Most script functions are small, and reducing general overhead will make both the editor and run-time feel snappier!

Rule of thumb: Sandbox likely to finish a small function before other solutions would have fully started executing

https://libriscv.no/docs/performance/latency

Function call latency

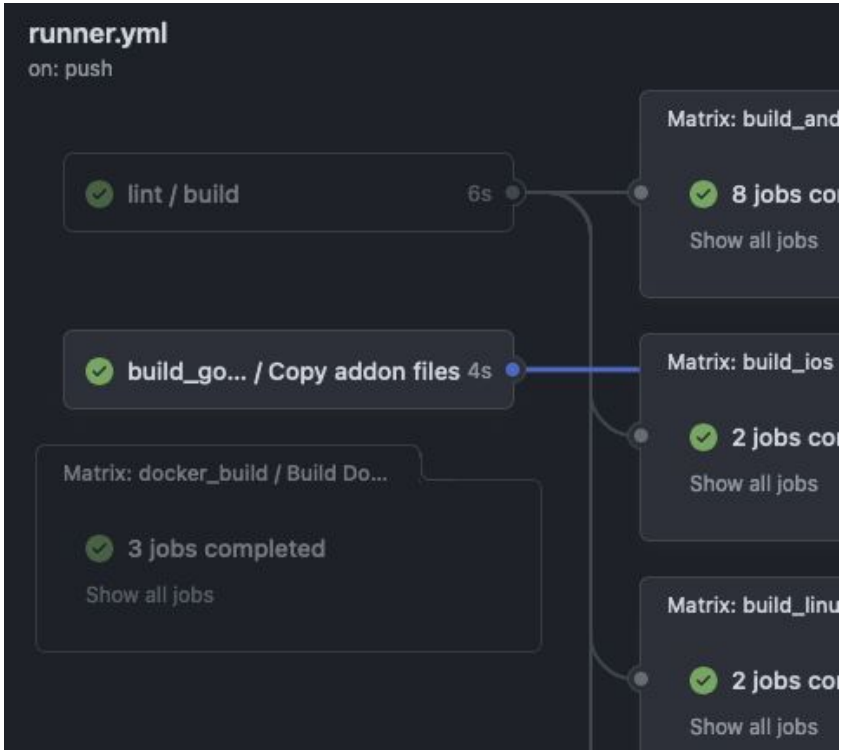
When the game engine wants to enter our script, a fixed cost has to be paid to enter and then leave the sandbox again. Sandboxes provide safety, and requires some setup in order to activate itself, and then afterwards some work to wind down. By call, we mean a function call into the VM, so a VM function call. It's just like a regular function call, but it happens inside our safe sandbox.



Portable on all of Godot Engine platforms

Uses automatic Github Actions to build for desktop, mobile and web. Single and double precision builds included.

Uses the GDExtension API.

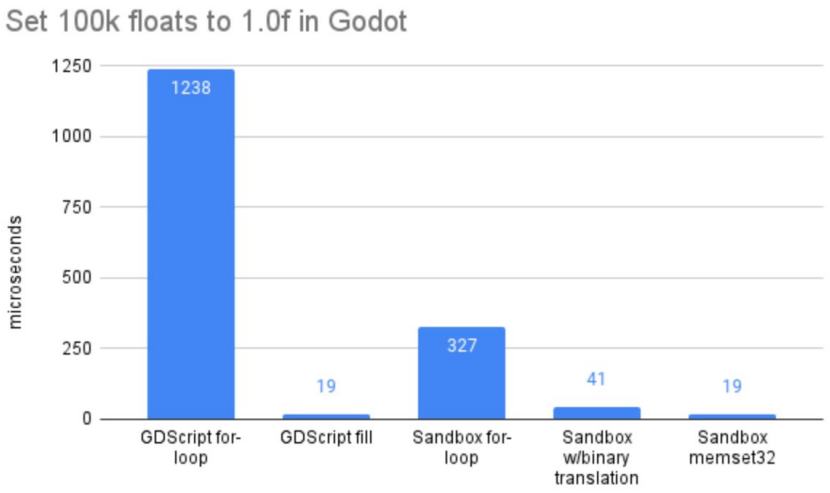


Godot Sandbox performance: Part 1 of 2

GDScript is a dynamic, interpreted language used for Godot Engine scripting. GDScript is generally fast due to Godot native functions.

Eg. Array.fill(1).

Godot Sandbox also has access, as well as a fast interpreter and native performance memory operations.



We can see that the Godot Sandbox using libriscv is 4x faster than GDScript at processing floats in this benchmark. However, when a Godot engine helper function is used (`fill(1.0)` in this instance), GDScript can have native performance.

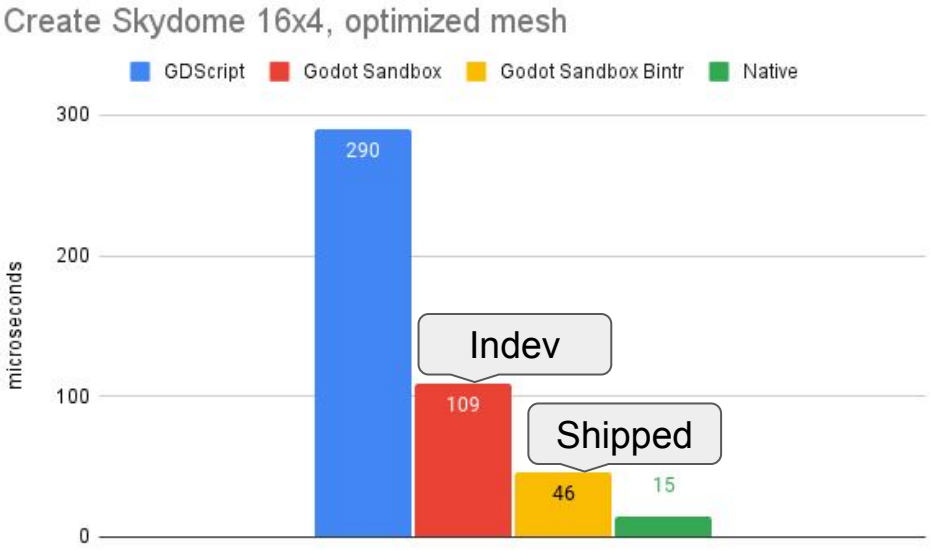
Godot Sandbox performance: Part 2 of 2

Godot Sandbox uses libriscv, one of the fastest interpreters.

Godot Sandbox utilizes Godot when possible, but will also be able to do heavy computations on its own.

Has a high-performance mode for final builds that works on all platforms.

https://libriscv.no/docs/performance/



Godot Sandbox is 2.6x faster than GDScript in the default interpreter mode, in this particular benchmark.

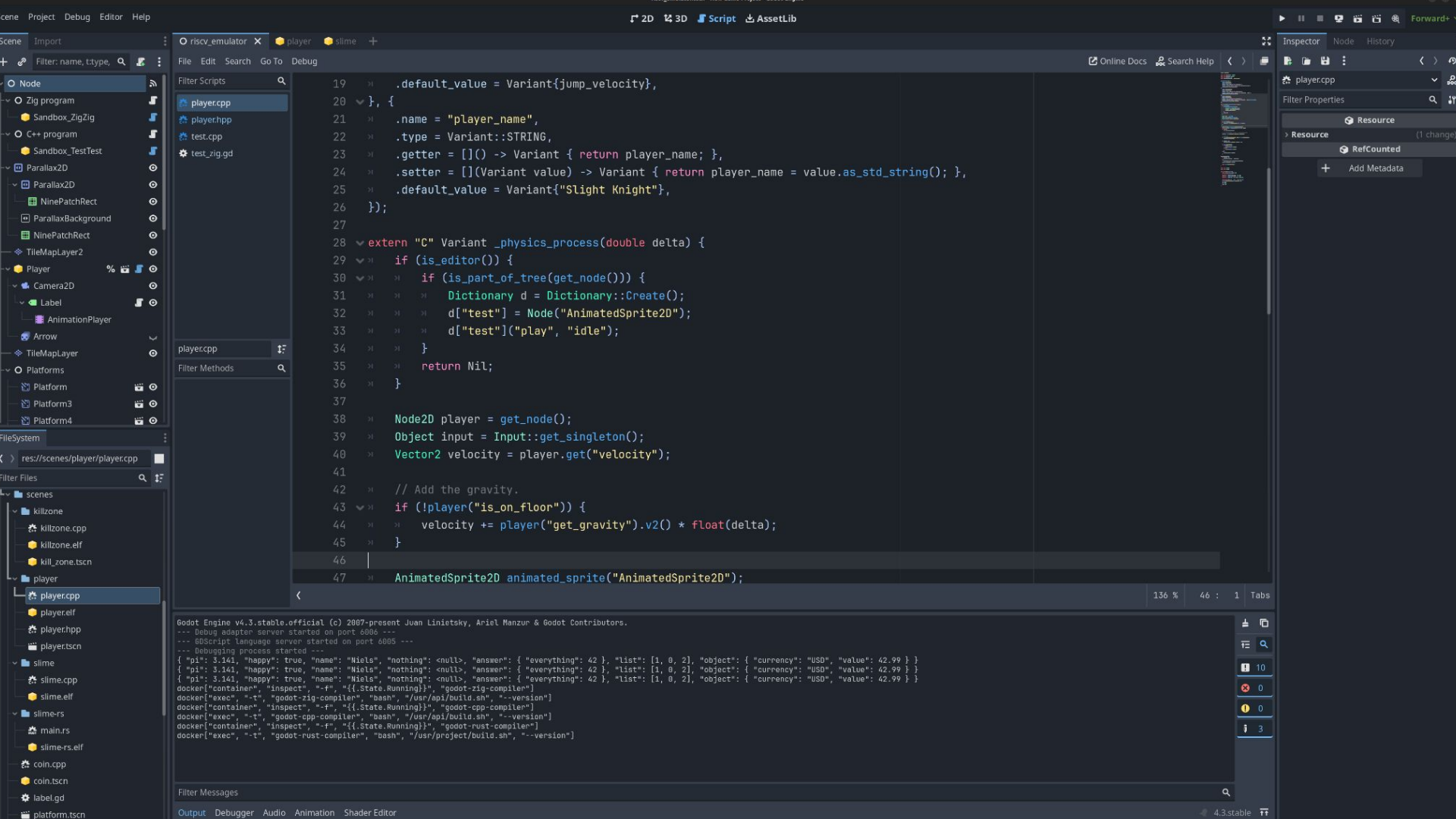
What can you do with godot-sandbox?

Features system languages like Rust, C++, Zig in Godot Engine.

```
#include "api.hpp"
// We can store data in the script, just like a regular C++ program
static int coins = 0;

extern "C" void reset_game() {
	coins = 0;
}

static void add_coin(const Node& player) {
	coins ++;
	// In our demo project we can access the coin label from the player
	// using a node path: Player -> get_parent() -> Texts -> CoinLabel
	Node coinlabel = player.get_node("../Texts/CoinLabel");
	coinlabel.set("text", "You have collected "
		+ std::to_string(coins) + ((coins == 1) ? " coin" : " coins"));
}
```


Project Debug Editor Help
2D 3D Script AssetLib
Forward+
Scene Import
Node
Zig program
Sandbox_ZigZig
C++ program
Sandbox_TestTest
Parallax2D
Parallax2D
NinePatchRect
ParallaxBackground
NinePatchRect
TileMapLayer2
Player
Camera2D
Label
AnimationPlayer
Arrow
TileMapLayer
Platforms
Platform
Platform3
Platform4
FileSystem
res://scenes/player/player.cpp
Filter Files
scenes
killzone
killzone.cpp
killzone.elf
kill_zone.tscn
player
player.cpp
player.elf
player.hpp
player.tscn
slime
slime.cpp
slime.elf
slime-rs
main.rs
slime-rs.elf
coin.cpp
coin.tscn
label.gd
platform.tscn
riscv_emulator
player
slime
File Edit Search Go To Debug
Online Docs
Search Help
Filter Scripts
player.cpp
player.hpp
test.cpp
test_zig.gd
player.cpp
Filter Methods
.default_value = Variant{jump_velocity},
}, {
.name = "player_name",
.type = Variant::STRING,
.getter = []() -> Variant { return player_name; },
.setter = [](Variant value) -> Variant { return player_name = value.as_std_string(); },
.default_value = Variant{"Slight Knight"},
});
extern "C" Variant _physics_process(double delta) {
if (is_editor()) {
if (is_part_of_tree(get_node())) {
Dictionary d = Dictionary::Create();
d["test"] = Node("AnimatedSprite2D");
d["test"]("play", "idle");
}
return Nil;
}
Node2D player = get_node();
Object input = Input::get_singleton();
Vector2 velocity = player.get("velocity");
// Add the gravity.
if (!player("is_on_floor")) {
velocity += player("get_gravity").v2() * float(delta);
}
AnimatedSprite2D animated_sprite("AnimatedSprite2D");
136 %
46 : 1
Tabs
Godot Engine v4.3.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 ---
--- Debugging process started ---
{ "pi": 3.141, "happy": true, "name": "Niels", "nothing": <null>, "answer": { "everything": 42 }, "list": [1, 0, 2], "object": { "currency": "USD", "value": 42.99 } }
{ "pi": 3.141, "happy": true, "name": "Niels", "nothing": <null>, "answer": { "everything": 42 }, "list": [1, 0, 2], "object": { "currency": "USD", "value": 42.99 } }
{ "pi": 3.141, "happy": true, "name": "Niels", "nothing": <null>, "answer": { "everything": 42 }, "list": [1, 0, 2], "object": { "currency": "USD", "value": 42.99 } }
docker["container", "inspect", "-f", "{{.State.Running}}", "godot-zig-compiler"]
docker["exec", "-t", "godot-zig-compiler", "bash", "/usr/api/build.sh", "--version"]
docker["container", "inspect", "-f", "{{.State.Running}}", "godot-cpp-compiler"]
docker["exec", "-t", "godot-cpp-compiler", "bash", "/usr/api/build.sh", "--version"]
docker["container", "inspect", "-f", "{{.State.Running}}", "godot-rust-compiler"]
docker["exec", "-t", "godot-rust-compiler", "bash", "/usr/project/build.sh", "--version"]
10
0
0
3
Filter Messages
Output Debugger Audio Animation Shader Editor
4.3.stable
Inspector Node History
player.cpp
Filter Properties
Resource
Resource (1 change)
RefCounted
Add Metadata

Convert code to runnable binaries

You can write C++/Rust/Zig directly in the editor.

Every time you save, the code is compiled to a binary that can be run by the Sandbox.

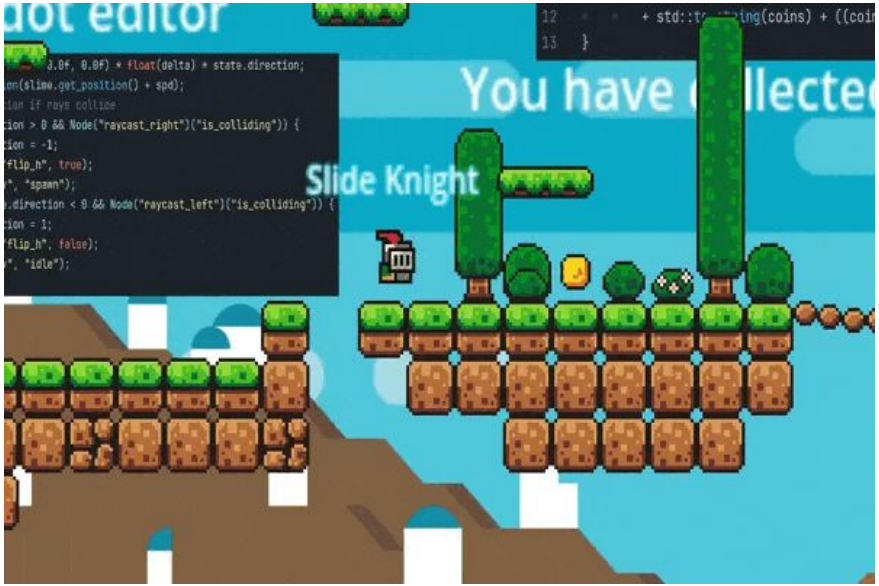
This binary is then set on the Sandbox and executed at runtime.

Godot Sandbox Demo

Based on a Brackeys tutorial, however, all the GDScripts are now C++ or Rust!

https://gonzerelli.itch.io/demo

https://github.com/libriscv/godot-sandbox-demo



Extra Slides

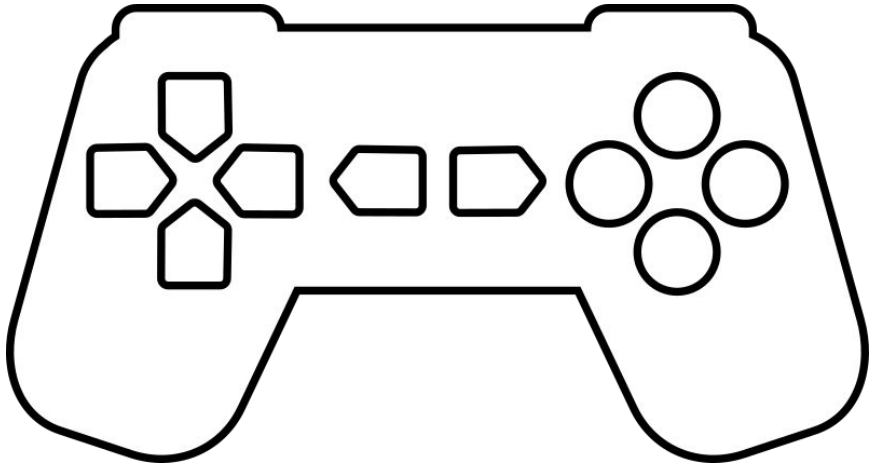
This page is left empty.

About a balloon game

fwsGonzo – Bloons TDS 6 is one of our favorite games, and when we bought it on a console we believed that it would be the latest version and compatible with the savegame we have online that is shared among the existing platforms.

Instead, the console has its own accounts, its own save games and cannot play with other platforms. The game is also outdated.

An official blog post states that *"console will generally lag behind the main game as updates take longer"*. No updates yet.



GodotCon 2024

12th & 13th of October 2024 - Berlin, Germany

Our event aims to give the Godot Engine community a platform to meet in person and exchange knowledge, form business contacts, and showcase their achievements. GodotCon is organized by the Godot Foundation and made possible by a very dedicated volunteer team.

https://conference.godotengine.org/2024/