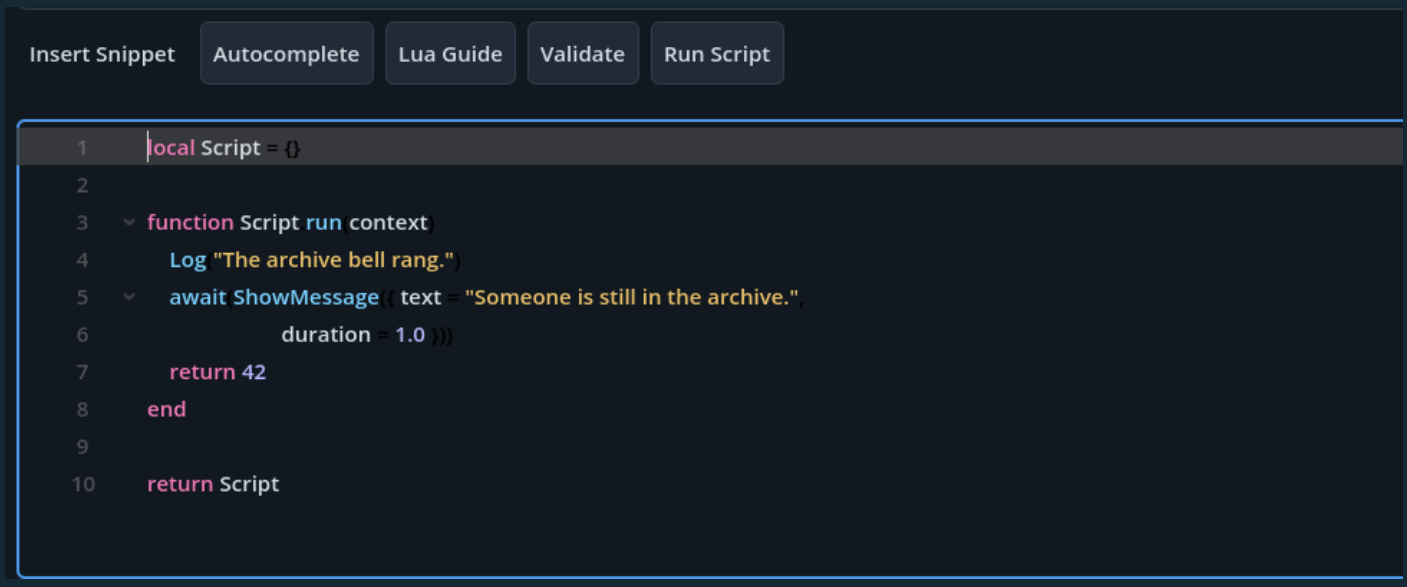
PCA 3D ADVENTURE STUDIO

Lua scripting

Write Lua for the adventure you’re making.



An illustrated programming tutorial

From your first line of Lua to GUI input, game events and save slots.

Lua 5.4 / Editor screenshots from PCA3D 0.86.3.4 on Linux

READER'S MAP

Choose where to begin

If this is your first script, start with the illustrated pages. If you already know Lua, jump straight to the PCA3D examples or the reference.

Follow the illustrated guide

Read a Lua record and find it in the editor

Write, run and repair your first script

Learn variables, decisions, lists and loops

Understand game variables, buttons and player input

Respond to events and expose a function as a visual node

Use time and files, then check your target platform

Look up a particular subject

All language lessons and reference chapters

Player input, timers, files, events and live game objects

Cross-language calls, visual nodes and native Godot controls

Choosing where to store a value and reading a return value

Event handlers, helper calls, action fields and property names

> The W-numbered pages are the illustrated guide. The programming chapters keep their original page numbers. Tap an entry above or use the PDF bookmarks.

START HERE

What happens when a script runs?

Suppose a player examines a bell. The game needs to know what to show the player and when to show it. A Lua script can describe that response in a few lines.



The player sees the sentence inside ShowMessage. The editor can also log a separate line for the author.

Who starts the script?

You can start it yourself with **Run Script** in the Lua Designer. In a game, a button, a hotspot or an event can start it. The same code can run from any of those places.

What does PCA3D look for?

```
function Script.run(context)
    Log("The archive bell rang.")
end
```

Read this as “When someone runs this record, write this line to the output.” **run** is the function name. **context** is information PCA3D sends along, such as which interaction started the script.

Where does the code live?

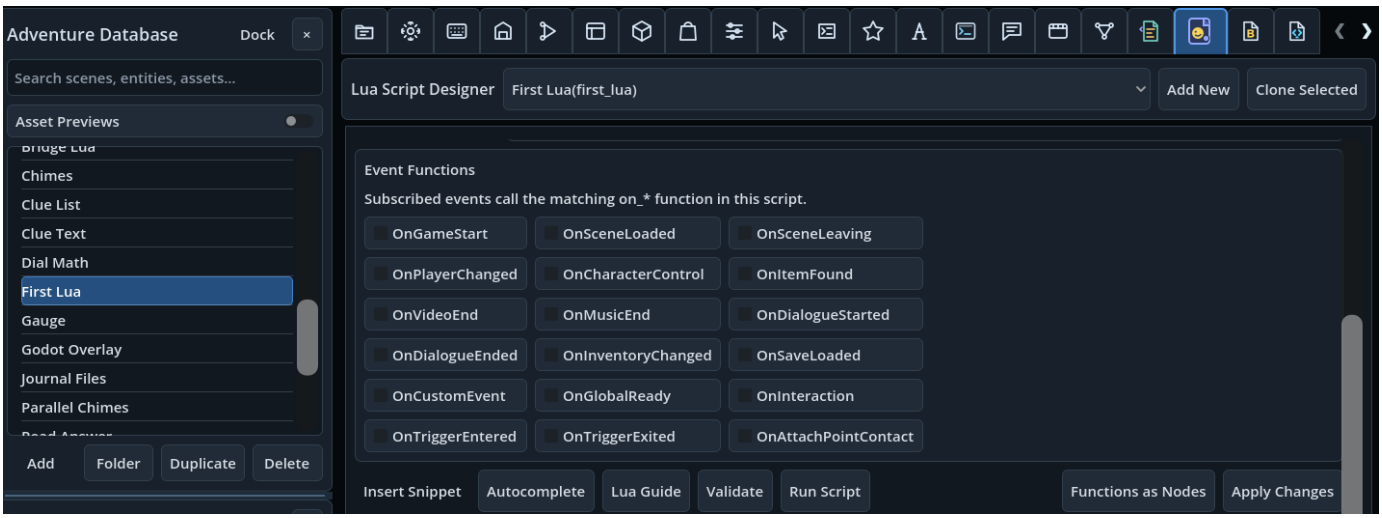
Open your game in PCA3D and select the Lua workspace. Each record has a name for you and an ID for the engine. The next two pages show the editor and a complete record, including the lines that go above and below the function.

> A box marked **fragment** shows part of a script. A **complete record** includes everything from the first local declaration through the final return.

EDITOR TOUR

Find the Lua Designer

A script record is a named piece of code in your game. The editor gives it a human-readable name and an ID that other parts of the game can use.



The script list is on the left. The selected workspace is Lua Script Designer. Add New sits next to the record picker.

01 Open the **Lua Script** workspace. Hover over the workspace icons to read their names. On a narrow screen, scroll the icon strip or select a Lua record in the Adventure Database.

02 In your game, select **Add New**. Enter **first_lua** for Script ID and **First Lua** for Name. If that ID is already in use, give the new record a different one.



The ID is the name you use in code. A space is fine in the display name. Use letters, numbers and underscores in the ID and start it with a letter or underscore.

Leave **Keep global instance alive** and the event boxes unchecked for now. The script should run when you ask it to. In the right-hand Inspector, **enabled** should be checked.

FIRST SCRIPT

Write a line the player can see

This is the complete first_lua record. Read it from top to bottom. To see it run in your game, enter it in the new record’s code box.

```
local Script = {}

function Script.run(context)
    Log("The archive bell rang.")
    await(ShowMessage({ text = "Someone is still in the archive.",
                        duration = 1.0 }))
    return 42
end

return Script
```

Read it from top to bottom

local Script = {}	Make an empty table. Think of it as a folder for the things this script can do.
function Script.run	Define the job PCA3D starts when you run the record. The parentheses receive information about what started it.
Log(...)	Write a line for the game’s author to see in the output.
ShowMessage(...)	Give the player a message. The braces hold settings, including the text and its duration in seconds.
return 42	Send the number 42 back to whoever called run. The player sees the message. A caller can use the number.
end / return Script	End the function, then hand the table of functions back to PCA3D.

> Keep the straight quotation marks when typing code. The two words in ShowMessage have capital S and M.

FIRST PLAYTEST

Run it. Then break it on purpose.

A deliberate typo is a good way to learn what the editor is trying to tell you.



The Lua toolbar. Validate checks the source, Apply Changes updates the record, and Run Script sends it to the game preview.

01 Select **Validate**. Look below the code for **Lua validation passed.** Select **Apply Changes** on the Lua toolbar, then **Save Project** in the main toolbar.

02 Select **Run Script**. If a preview is already running, PCA uses it. The player should see “Someone is still in the archive.” The output includes “The archive bell rang.”



The message in the running game preview. Replace the words inside the quotes, apply, and run again to see your own line.

What a missing parenthesis looks like

Go back to the script. Remove the closing parenthesis from **function Script.run(context)** and choose **Validate**.



The error points near Log because that is where Lua realizes the previous line never closed.

Put the parenthesis back. Validate again, apply, and save. If validation passes but the game does the wrong thing, check the record or button ID next. Correct spelling in Lua and the correct target in the editor are separate checks.

LANGUAGE BASICS

Read a decision like a sentence

The player types 02 into a box. Is that two turns of the dial, or just two characters? Lua can work with either meaning.

```
local limit = 3
local typed = "02"
local turns = tonumber(typed)
if turns == nil then
    Log("Type a number.")
elseif turns > limit then
    Log("Too many turns.")
else
    Log("The dial moves.")
end
```

This is a **fragment**. To try it, put it inside **function Script.run(context)**, before its **end**. Keep **return Script** at the bottom of the record.

typed = "02"	tonumber produces 2. The number fits within the limit, so you see “The dial moves.”
typed = "8"	Lua skips the first branch and takes the elseif branch. You see “Too many turns.”
typed = "bell"	tonumber produces nil, meaning there is no number here. The first branch handles it before Lua compares it to 3.

Three little marks worth learning

= gives a name a value. **==** asks whether two values match. **--** starts a comment, so you can leave yourself a note. For a limit you intend to keep fixed in Lua 5.4, write **local LIMIT <const> = 3**.

LANGUAGE BASICS

Give the game a list of clues

A table can hold a list in order. A loop lets you visit each entry without copying the same line over and over.

```
local clues = {"west", "bell", "moon"}
for place, clue in ipairs(clues) do
    Log(place .. ". " .. clue)
end
```

Paste this fragment inside **run**. The output is **1. west**, **2. bell**, then **3. moon**. Each trip around the loop gets the next place and its word. The two dots join pieces of text.

What changes if you use while?

```
local i = 1
while i <= #clues do
    Log(clues[i])
    i = i + 1
end
```

Put this below the first example so both can see **clues**. The # sign counts a list with no gaps. This loop checks its condition before each turn. The last line moves to the next clue. Without that line it would keep printing west.

Pick one clue instead

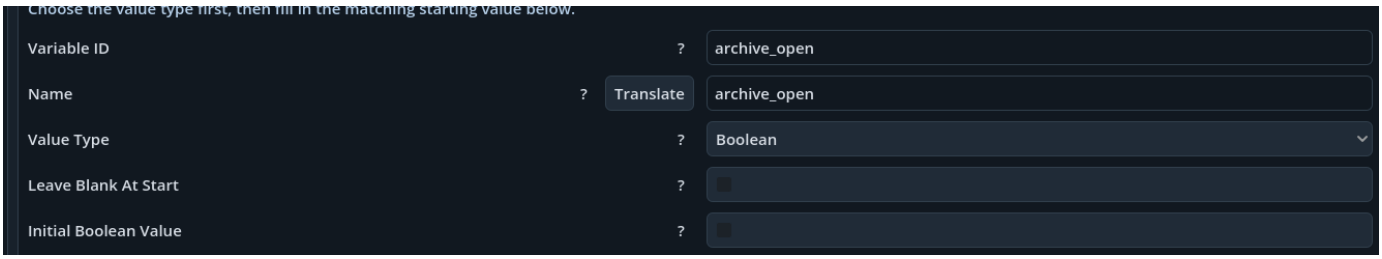
```
local pick = math.random(1, #clues)
Log("Today's clue is " .. clues[pick])
```

> Ordinary Lua lists start at 1. A Godot Array returned by a PCA helper such as Parallel starts at 0. The language chapter on arrays shows how to tell which kind you have.

GAME VARIABLES

Remember a yes or no

Has the player met the stationmaster? Has someone opened the archive? Each question has just two answers. A Boolean variable is a natural fit.



The inspector shows a Boolean named archive_open. The unchecked Initial Boolean Value means it starts false.

Set the starting value

In **Variable Designer**, give the variable an ID. Select **Boolean** for Value Type. Leave **Leave Blank At Start** unchecked when the game should know the answer from the first frame. The Initial Boolean Value checkbox then sets that answer.

Change it during play

```
await(SetVariable({ name = "archive_open", value = true }))
```

This fragment goes inside a function. Once the game runs this line, another script can ask about the same variable by ID.

Read it somewhere else

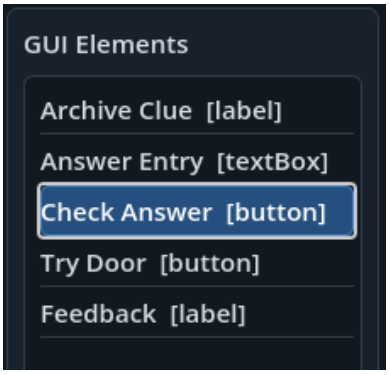
```
if IfVariable("archive_open", true) then
    Log("The bolt is open.")
end
```

> Keep IDs consistent in both places. A player can see the word “Archive” on a door even when the underlying variable has a completely different name.

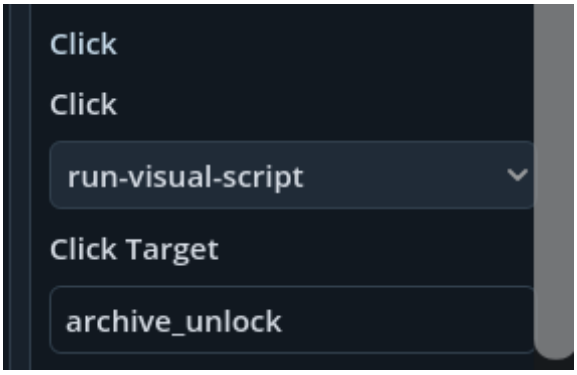
GUI EVENTS

How a button finds a Lua record

The word on a button tells the player what to expect. Its Click Target tells PCA3D which script should do the work.



The highlighted item is a button named Check Answer.



In this screenshot its Click Target is archive_unlock.

01 Select a button in **GUI Designer → GUI Elements**. In the right Inspector, open **Element Properties → Events**. On smaller screens, use the tab arrows to reach Events.

02 Find **Click**. In 0.86.3.4 the behavior named **run-visual-script** also starts Lua records. Enter your Lua record’s ID under **Click Target**.

03 To tell whether a click reaches the record, put a short Log call at the beginning of its run function. Apply the record, save your game and click the button in the game preview.

```
Log("The button reached this script.")
```

> If moving the pointer starts the script, inspect the Hover setting above Click. Those two events can have different behaviors.

TEXT AND INPUT

Read what the player typed

A player may enter “west”, “WEST” or a word surrounded by spaces. Decide which differences matter before comparing their answer.

Read the live text

```
local raw = GetGUIElementProperty(
    "archive_panel", "answer_entry", "text", "")
```

The first two arguments identify the GUI and its text box. The third requests its current text. The empty string at the end is a fallback if the lookup fails. Substitute the IDs used by your own GUI.

Clean it before comparing

```
local text = tostring(raw)
local trimmed = text:match("^%s*(.-)%s*$")
local answer = string.lower(trimmed)
```

The pattern removes spaces from the two ends. It keeps spaces inside the answer. Lowercasing then turns WeSt into west. The comparison now accepts both.

Give a helpful reply

```
local reply = "Look for wear on one of the dials."
if answer == "" then
    reply = "Type a direction first."
elseif answer == "west" then
    reply = "The lock releases."
end
SetGUIElementProperty("archive_panel", "feedback", "text", reply)
```

These three fragments belong inside the same function. The final line puts the chosen reply in a label. An empty answer gets its own message so the player knows what to do next.

READING GAME FEEDBACK

Find the first thing that went wrong

A button can respond, show the right message, and still leave a door closed. Check each piece in the order the game uses it.



The label shows a hint after the player enters east.



The same label shows a different reply for WeSt.

Nothing happens	Place a Log call at the start of run. If it never appears, check the Click Target and the record’s enabled setting.
Every entry looks blank	Check the GUI ID, the text-box ID and the property name text. Log the raw value inside square brackets to make extra spaces visible.
The label changes, but the door stays closed	Look at the condition the door reads. Changing label text does not write a Boolean. Check that the code sets the same variable ID that the door reads.
Progress disappears after loading	See where the value was stored. A local variable belongs to a script invocation or instance. Keep progress that belongs in a save slot in a PCA game variable.

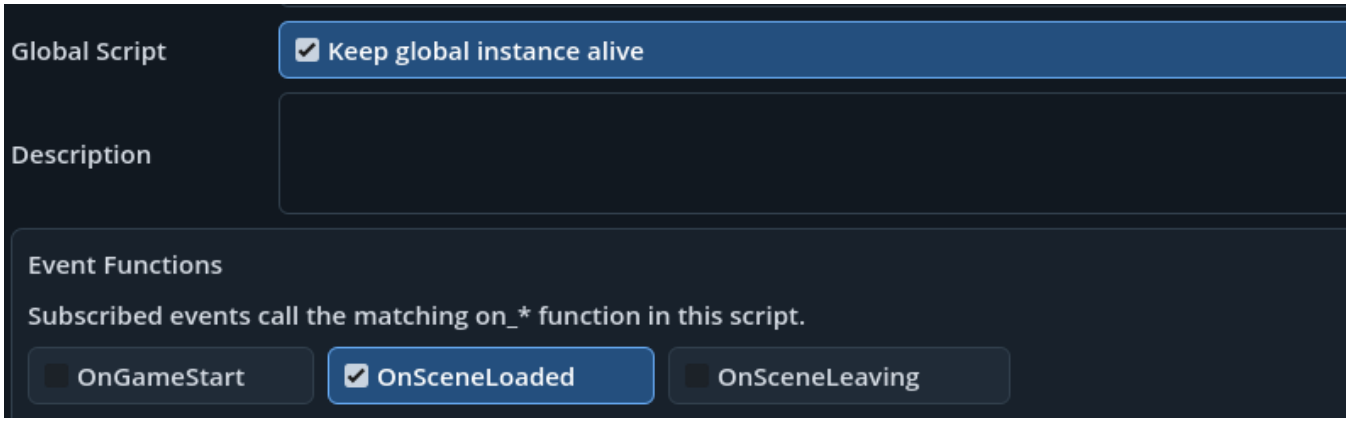
```
Log("Read [" .. tostring(raw) .. "]")
```

Place this fragment immediately after reading the text box. Seeing **Read [WeSt]** confirms the text reached Lua. Next, log the cleaned answer to check the comparison.

GAME EVENTS

Let entering a room run a script

You can ask PCA3D to call your code when something happens. Here, every arrival adds one to a running count and writes the room ID to the output.



In room_arrivals, Keep global instance alive retains the count for this play session. OnSceneLoaded subscribes the record to room arrivals.

```
local Script = {}
local arrivals = 0

function Script.on_scene_loaded(event)
    arrivals = arrivals + 1
    Log("Arrival " .. arrivals .. " in " .. tostring(event.sceneId))
end

function Script.run(context)
    return arrivals
end

return Script
```

The first arrival starts at zero and adds one. On entering the second room, the same script instance still holds one, so the next arrival becomes two. Checking an event and defining its matching function are both part of the hookup.

> In the capture, the room IDs were lab and roof. The output read **Arrival 1 in lab**, then **Arrival 2 in roof**. Use a PCA variable such as archive_open for progress a player needs after loading a game.

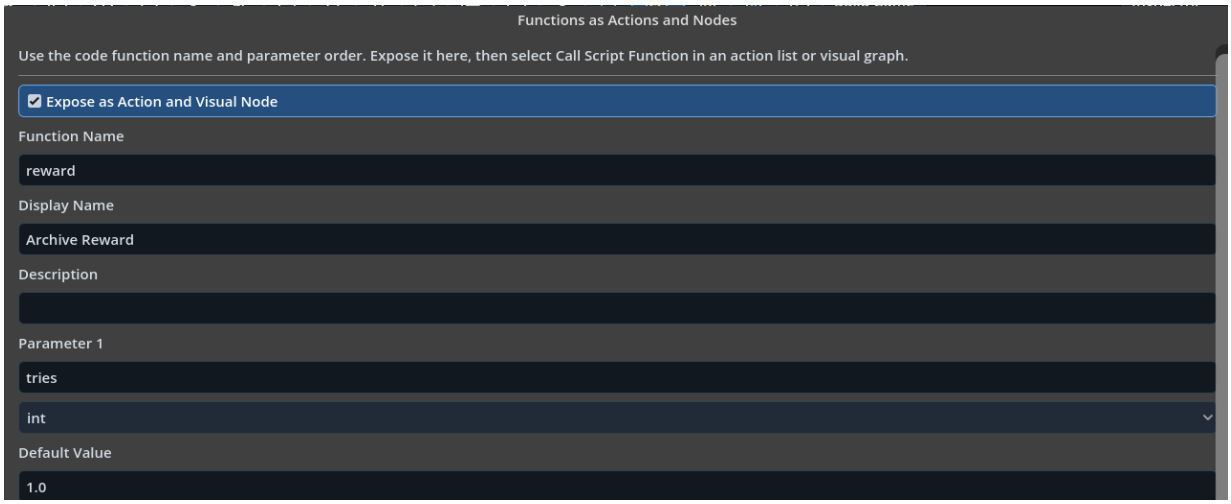
REUSABLE FUNCTIONS

Make a Lua function a visual node

Suppose a designer wants to award fewer points for extra attempts. You can write the rule once in Lua and give the designer two number fields.

```
function Script.reward(tries, maximum)
    return math.max(0, maximum - math.max(0, tries - 1))
end
```

This fragment belongs in a Lua record. The complete listing is on programming page 20. With three tries and ten possible points, the first try is free and two extra tries cost two points. The result is **8**. The outer math.max keeps the score above or equal to zero.



The source name must be reward. Archive Reward is the readable label. The first parameter is tries. Scroll down to add maximum as the second parameter.

01 Apply the Lua source first. Select **Functions as Nodes → Add Function**. Enter **reward** for Function Name and **Archive Reward** for Display Name. Enable **Expose as Action and Visual Node**.

02 Add **tries** and **maximum** in that order. Choose **int** for both, with defaults of **1** and **10**. Select **Use Definitions**, then **Apply Changes** and save.

03 In an action list or visual graph, add **Call Script Function**, pick Archive Reward, and enter 3 and 10. Choose a numeric Result Variable to keep the 8. The definition tells the editor what fields to draw. The Lua code supplies the result.

TIME, FILES AND SMALL EXPERIMENTS

Give a clue time to land

Two lines in a row can run so quickly that the player never sees the first result. A wait gives the game time to draw another frame.

```
Log("First chime")
await(Wait(0.5))
Log("Second chime")
```

Put this fragment inside run. The log gets two entries about half a second apart. **await** lets the current job pause while the game continues. Programming page 13 shows how to start two jobs together and wait for both.

Keep a written clue between sessions

```
local path = "user://journal/note.txt"
local previous = ReadTextFile(path, "")
local next_note = previous .. "The dial points west.\n"
local written = WriteTextFile(path, next_note)
Log("Wrote note: " .. tostring(written))
```

On the first run there is no old note, so ReadTextFile supplies empty text. On the second run it reads the first line before writing both lines back. Start at **user://** so PCA chooses the game’s writable location on each device.

Inspect the worked file example

The complete Lua record on programming page 14 shows what happens when it runs twice. It writes one dated journal entry per run and sets a JSON preference called **hints** to false. The function returns the stored preference. For this script, a return value of false is the expected result of a successful read.

> Choose a distinct filename for each purpose. WriteTextFile replaces the file contents. The code above preserves the old text by reading it first.

PLATFORM AND RELEASE NOTES

Use the guide on your version

The screenshots use the Linux editor in English. Follow the control names and record IDs on your platform. File-browser layouts, translations and panel positions can differ.

Windows, macOS, Linux	Use the same Lua records and project IDs. Open your game in PCA3D and select the Lua Designer. Use the same record and function names on all three platforms.
Touch and small screens	Tap the same controls. Scroll narrow toolbars and use the Inspector tab arrows. Test the text box with the device keyboard and use the Check button to submit.
Mobile and browser games	Android, iOS and Web have target-specific Lua extension entries in the 0.86.3.4 source. Use the matching build/export setup. Check the puzzle and save/load on the exported game.
Older PCA3D releases	Lua records first appeared in 0.39.0. Version 0.39.1 fixed nested table conversion across scripts. Functions as Nodes and Call Script Function arrived in 0.81.0. Use the Lua function directly on earlier Lua-capable releases.

What was checked

The screenshots and playtests used PCA3D 0.86.3.4 with Godot 4.7.2 on Linux. The tests included Lua basics, GUI click dispatch, the locked and open door, saving and restoring the flag, custom events, file writes, timed branches, calls between records and a native Godot overlay. Other operating systems and individual historical releases were not separately run.

Where to find a command for your build

Open **Lua Guide** in the Lua toolbar. **Insert Snippet** shows the templates shipped with your editor. The reference pages that follow describe the 0.86.3.4 catalog. For the language itself, see the Lua 5.4 Reference Manual.

NAVIGATION

Contents

Using this guide 3
Run your first Lua record 4
Names, comments and quoted text 5
Values, variables and constants 6
Operators and decisions 7
Loops that finish 8
Arrays, dictionaries and the bridge 9
Functions and reusable text 10
Math, randomness and vectors 11
Read input from an authored GUI 12
Timers, parallel actions and clocks 13
Edit text files and JSON 14
Events and the lifetime of a record 15
Send a custom event 16
Live records and the calling object 17
Talk, animate and stage a reveal 18
Cross-language calls and includes 19
Expose a Lua function as a visual node 20
Inventory, quests and save slots 21
Use Godot from a Lua record 22
Choose where to store a value 23
Read a return value 24
Debugging a Lua record 25
Event handlers 26
Helper functions and proxy methods 27
Gameplay action fields 31
Designer property paths 57

USING THIS GUIDE

How to read the code

Each lesson explains a piece of Lua or an editor control. Start with the subject you need for your game.

Complete records and fragments

A complete Lua record starts with **local Script = {}** and ends with **return Script**. You can read it from top to bottom. A fragment shows only the lines needed to explain a point. Its caption tells you where those lines belong.

Names that belong to your game

A word in quotes such as **"archive_panel"** can be an editor record ID. If you use the code with an existing GUI, substitute its ID and the ID of its text box. Built-in helper names such as **GetGUIElementProperty** stay exactly as written.

Three ways a script gives you an answer

Log(message)	Writes a line of output you can read while developing.
ShowMessage(...)	Displays a message to the player.
return value	Hands a value back to the function that called the script.

Where to look things up

The event-handler table starts on programming page 26, helper functions on page 27, gameplay actions on page 31 and record properties on page 57. Use the PDF bookmarks to go straight to one of them.

> The screenshots use PCA3D 0.86.3.4 on Linux. See the platform and release notes on page W16 for differences that matter when using another build.

LESSON 01

Run your first Lua record

Give the game a line of dialogue and give yourself a way to tell whether the script ran.

Open a test project. Select the Lua workspace, create a script and give it the unique ID first_lua. Leave Enabled on and Global Script off. Replace the entire source with the record below.

```
local Script = {}

function Script.run(context)
    Log("The archive bell rang.")
    await(ShowMessage({ text = "Someone is still in the archive.",
                        duration = 1.0 }))
    return 42
end

return Script
```

Choose Validate, then Apply Changes and save the project. Start Test Scene or Play. Use Run Script against the running preview, or attach the script to an interaction. Look for the log line and the on-screen message. The number 42 is available to a caller that captures the return value.

Read the four moving parts

Script is a local table. The function declaration puts run in that table. PCA calls it and supplies the caller context. The final return gives the whole table to PCA. Put other exported functions above that last line. A top-level return of 42 would hand PCA a number where it needs a module.

Each boxed Lua record in the lessons can replace an entire record unless labeled as a fragment. Use a fresh ID per lesson to keep earlier exercises available.

LESSON 02

Names, comments and quoted text

Write the clue exactly as it should appear, including a quotation mark or a second line.

```
local Script = {}

function Script.run(context)
    -- A single-line comment ends at the newline.
    local owner = "Ada"
    local clue = 'The label reads "west".'
    local note = [[First bell: enter.
    Second bell: wait.]]
    --[[
    An author note can span lines.
    ]]
    local result = owner .. ": " .. clue .. "\n" .. note
    Log(result)
    return result
end

return Script
```

Lua names are case-sensitive. A name can contain letters, digits and underscores, with a letter or underscore first. Scripts conventionally indent each block and put one statement on each line. Use end to close a function or conditional.

Single and double quotes both create strings. Pick the one that needs fewer escapes. Within a quoted string, \n makes a newline and \\ inserts one backslash. Long brackets preserve line breaks and backslashes. Use [=[and]=] if the text must contain]].

Try replacing the word west with east. Only the quoted clue should change.

LESSON 03

Values, variables and constants

Separate a number the player typed from the text that represents it.

```
local Script = {}

function Script.run(context)
    local MAX_TURNS <const> = 3
    local typed = "02"
    local turns = tonumber(typed)
    local allowed = turns ~= nil and turns <= MAX_TURNS
    local missing = nil
    return { raw = typed, turns = turns, allowed = allowed,
             missingType = type(missing) }
end

return Script
```

The result contains the string 02, the number 2, true and the type name nil. tonumber returns nil for a word such as bell. Short-circuit evaluation of and protects the numeric comparison. Use tostring for display and type to inspect a Lua value.

A local lives within its enclosing block. A declaration outside the module functions is shared by those functions for the lifetime of that instance. An assignment to an undeclared name writes into the Lua environment. Use local for temporary values and a PCA variable for progress that other records must see.

The Lua 5.4 <const> attribute prevents reassignment. A constant table can still have its fields changed. Only nil and false count as false in a conditional. The number 0 and an empty string both count as true.

LESSON 04

Operators and decisions

Use arithmetic to drive a puzzle, then choose exactly one response.

```
local Script = {}

function Script.run(context)
    local ticks = 17
    local turns = ticks // 6
    local remainder = ticks % 6
    local response
    if turns < 1 then
        response = "Wind the clock."
    elseif remainder == 0 and turns >= 2 then
        response = "The teeth line up."
    else
        response = "Advance " .. (6 - remainder) .. " tick(s)."
    end
    return response
end

return Script
```

The result is “Advance 1 tick(s).” Change ticks to 12 for the aligned branch. Integer floor division // counts completed turns. The % operator returns the remainder. The / operator returns a floating-point quotient. Use +, -, *, and ^ for addition, subtraction, multiplication and powers.

Use = to assign, == or ~= to test equality, and <, <=, > or >= to compare. Parentheses make precedence explicit. The .. operator joins strings. For branching, and and or return operands, and not returns a Boolean. Distinguish a missing setting from false with an explicit value == nil check.

LESSON 05

Loops that finish

Build a list of clock chimes and find the first one that opens the latch.

```
local Script = {}

function Script.run(context)
    local sounds = {}
    for bell = 1, 5, 2 do
        sounds[#sounds + 1] = bell
    end
    local i = 1
    while i <= #sounds do
        if sounds[i] == 3 then break end
        i = i + 1
    end
    local winding = 0
    repeat
        winding = winding + 1
    until winding >= 2
    return { position = i, turns = winding, bells = sounds }
end

return Script
```

The numeric for includes its endpoint when the step lands there, producing 1, 3, 5. The while checks before entering and stops at index 2. The repeat loop checks after the body, so it always runs at least once.

Use break to leave the nearest loop. To skip an item, put the work inside if eligible then … end. Choose a finite bound when scanning inventory. A tight loop waiting for a click prevents the runtime from processing that click. Use a GUI action or a signal for input.

LESSON 06

Arrays, dictionaries and the bridge

Use ordered slots for a combination and named fields for a clue.

```
local Script = {}

function Script.run(context)
    local slots = { "west", "bell", "moon" }
    table.insert(slots, "key")
    local removed = table.remove(slots, 2)
    local counts = {}
    for _, word in ipairs(slots) do
        counts[word] = (counts[word] or 0) + 1
    end
    local total = 0
    for word, count in pairs(counts) do
        total = total + count
    end
    return { first = slots[1], removed = removed, total = total }
end

return Script
```

Expect west, bell and 3. Ordinary Lua sequences start at 1. ipairs stops at the first missing index. Keep sequences dense when using #, table.insert or table.remove. pairs visits keys in an unspecified order. A table assignment shares the same table, so changing a field through one variable affects both.

PCA converts a table passed to a typed helper to its expected Godot Array or Dictionary. For an untyped return, a positive Lua sequence length selects an array. Otherwise the bridge selects a dictionary. Keep a list and its metadata in separate fields. Use Array{} to specify an empty Godot array. Godot arrays use index 0 for their first entry.

LESSON 07

Functions and reusable text

Normalize an answer once so a player's spacing and capitalization do not change the puzzle.

```
local Script = {}

local function tidy(text)
    return (text:match("^%s*(.-)%s*$"))
end

function Script.normalize(text)
    return string.lower(tidy(tostring(text or "")))
end

function Script.run(context)
    local answer = Script.normalize("  WeSt  ")
    local label = string.format("Dial %02d / %s", 3, answer)
    return label
end

return Script
```

Expected result: Dial 03 / west. The local function remains private. normalize is on the returned table, so another PCA record can call it by name. The ^ and $ pattern anchors mark the ends, %s matches whitespace, and the capture (.-) takes the shortest middle that fits.

Use text:sub(1, 4) for a slice, text:find(".", 1, true) for a literal dot, and text:gsub("bell", "chime") for replacement. gsub returns the new string and a replacement count. Strings are immutable. Lua string indexes count bytes. Use utf8.len for the number of UTF-8 code points and PCA's Lower or Upper for engine-side text handling.

LESSON 08

Math, randomness and vectors

Keep a gauge within its bounds and give a misplaced gear a measurable distance from its socket.

```
local Script = {}

function Script.run(context)
    local charge = math.max(0, math.min(100, 87 + 22))
    local angle = math.rad(90)
    local direction = math.floor(math.sin(angle) + 0.5)
    local choices = { "one chime", "two chimes", "silence" }
    local clue = choices[math.random(1, #choices)]
    local offset = Vector3(3, 0, 4)
    return { charge = charge, north = direction,
             distance = offset:length(), clue = clue }
end

return Script
```

Expect charge 100, north 1, distance 5 and one of the three strings. math.floor rounds down and math.ceil rounds up. math.abs gives magnitude. math.sin and math.cos take radians. math.random(lo, hi) includes both integers. Use a stored puzzle solution if a random choice must survive a reload.

The bridge also supplies Clamp, Lerp, Distance, RandomInt and Choose. Use a consistent random generator when reproducing a bug. math.randomseed controls Lua randomness. PCA's Randomize controls its engine-side generator.

LESSON 09

Read input from an authored GUI

Wire a text box and a button so the player can actually type the answer.

Create a GUI with ID archive_panel. Add a single-line text box answer_entry, a button check_answer and a label feedback. Use unique element IDs. Make the panel visible during play. Leave Clear on Enter and parser submission off for this button-driven exercise.

```
local Script = {}

function Script.run(context)
    local text = GetGUIElementProperty(
        "archive_panel", "answer_entry", "text", "")
    local answer = await(CallScript(
        "archive_words", "normalize", Array{tostring(text)}))
    local reply = "Try the direction on the label."
    if answer == "" then
        reply = "Type a direction first."
    elseif answer == "west" then
        reply = "The latch clicks."
    end
    SetGUIElementProperty("archive_panel", "feedback", "text", reply)
    return reply
end

return Script
```

Apply archive_words from the functions lesson. In the button's click action, choose the action that runs a script and select read_answer. If using Call Script Function, expose run with no authored parameters first. Apply the GUI, save and start Play. Type a space, WeSt, and another space. Click the button. The label should read “The latch clicks.”

If every click produces the empty-answer response, check the GUI ID and the live text-box element ID. The final empty string in the getter is the fallback for a missing property.

LESSON 10

Timers, parallel actions and clocks

Leave time for an animation or a sound while the game continues to process frames.

```
local Script = {}

function Script.run(context)
    local values = await(Parallel(
        function()
            await(Wait(0.05))
            return "short"
        end,
        function()
            await(Wait(0.10))
            return "long"
        end
    ))
    return tostring(values[0]) .. "/" .. tostring(values[1])
end

return Script
```

Expected result: short/long. Each branch is a function. Parallel starts the branches together and waits for all of them. The returned Godot array keeps the original branch order. Wait takes seconds. Use await(Wait(0.5)) between two sequential cues for a half-second pause.

A wait resumes the same invocation. A different button or scene event can run while it is waiting. Before changing a room-specific control afterward, verify CurrentScene() and the puzzle state. For a countdown that survives save/load, explicitly keep the remaining time or deadline in project state.

DateTimeString(true) returns a UTC timestamp for a journal. NowUnix() gives wall-clock seconds. For measuring short performance intervals use Godot's monotonic Time:get_ticks_msec(). Calendar changes can move wall-clock time backward.

LESSON 11

Edit text files and JSON

Keep a small journal and update a preference without erasing the other keys.

```
local Script = {}

function Script.run(context)
    local path = "user://archive_workshop/journal.txt"
    local old = tostring(ReadTextFile(path, ""))
    local line = DateTimeString(true) .. "  Inspected the dial.\n"
    if not WriteTextFile(path, old .. line) then
        return "Could not write journal."
    end
    local prefs = "user://archive_workshop/preferences.json"
    local data = ReadJSONFile(prefs, { hints = true })
    data.hints = false
    if not WriteJSONFile(prefs, data) then
        return "Could not write preferences."
    end
    return ReadJSONFile(prefs).hints
end

return Script
```

Run twice. The journal gains a second line and the preference remains false. WriteTextFile replaces the file and creates missing parent directories beneath user://. Reading before rewriting implements append for a small, single-writer journal. For large logs use a streaming file API. These sample paths are dedicated to the workshop.

ReadTextFile supplies its fallback when the file cannot be opened. ReadJSONFile uses its fallback for empty or unparseable JSON. If a file can contain valid JSON of the wrong shape, validate that it is a dictionary before changing fields. WriteJSONFile returns success or failure. Keep a save-slot action for a complete game snapshot.

Lua also opens standard libraries in this build. io.open uses operating-system paths. The PCA helpers understand Godot's user:// path on each export platform.

LESSON 12

Events and the lifetime of a record

Count scene arrivals during one game session and watch what changes when you turn on Global Script.

```
local Script = {}
local arrivals = 0

function Script.on_scene_loaded(event)
    arrivals = arrivals + 1
    Log("Arrival " .. arrivals .. " in " .. tostring(event.sceneId))
end

function Script.run(context)
    return arrivals
end

return Script
```

Enable the record, check OnSceneLoaded in the Lua Designer, check Global Script, apply, save and start a fresh play session. Move between two rooms. Expect the next arrival number each time. With Global Script off, each dispatch gets a fresh module and starts its own count.

The checkbox subscribes the record. The on_scene_loaded function receives that event. An ordinary explicit run uses run. An event without its matching handler can fall back to run. A global record can also implement on_global_ready for initialization. Changing source or rebuilding the runtime can recreate the module.

Use PCA variables for saveable progress. Local state in a global module lasts while that instance exists. It does not automatically become part of a saved game.

LESSON 13

Send a custom event

Connect two records without making the sender know which record handles a clue.

Sender · bell_sender

```
local Script = {}

function Script.run(context)
    await(EmitEvent("OnCustomEvent", {
        topic = "archive_bell", strikes = 3
    }))
    return true
end

return Script
```

Receiver · bell_listener

```
local Script = {}

function Script.on_custom_event(event)
    if tostring(event.topic) == "archive_bell" then
        Log("Archive bell rang " .. tostring(event.strikes) .. " times.")
    end
end

return Script
```

Check OnCustomEvent on bell_listener, then apply both records. Run bell_sender. The log should show three strikes. Subscribe the receiver only. Emitting the same event again from its own unguarded handler would cause repeated dispatch.

The dispatcher matches the emitted name. Use OnCustomEvent for this checkbox and carry the gameplay name in topic. The context also carries eventName. A custom field such as strikes belongs to your own protocol.

LESSON 14

Live records and the calling object

Look up a character by ID and change only the record that the interaction concerns.

```
local Script = {}

function Script.run(context)
    local id = tostring(ActivePlayer())
    if not Character:Has(id) then return "No active character" end
    local actor = Character:Get(id)
    local label = actor:Get("name", id)
    Log("Active character: " .. tostring(label))
    return tostring(label)
end

return Script
```

Character is a registry. Character:Get(id) returns a record proxy and actor:Get(path, fallback) reads a field. Character.ada is a shortcut for a fixed ID. Bracketed or method-based lookup accommodates IDs that contain a dash. player identifies the active character. this is the caller proxy when the interaction supplies an owner. A manual run may have no meaningful owner.

Fragments for an authored actor and room

```
Character.elias:Set("name", "Master Clockmaker")
await(AwaitCall(Character.elias, "Say", "Count the chimes."))
await(ActivateCamera({ cameraId = "wide", duration = 0.5 }))
await(ChangeScene({ sceneId = "archive", spawnId = "entrance" }))
```

Create the referenced records first. Use colon syntax for a synchronous Godot object method. Use AwaitCall for a yielding proxy method. The free-standing action wrappers take a Lua table. The registry field list at the back describes which changes can update a live instance and which affect its definition.

LESSON 15

Talk, animate and stage a reveal

Make a spoken clue finish before a camera moves and a cutscene takes over.

```
local Script = {}

function Script.run(context)
    await(Say({ speakerId = "elias", text = "Listen to the last bell.",
                waitForContinue = true, typewriter = true }))
    await(ActivateCamera({ cameraId = "wide", duration = 0.6,
                           wait = true }))
    await(PlaySound({ assetId = "chime" }))
    await(StartCutscene({ cutsceneId = "archive_reveal" }))
    return true
end

return Script
```

Create or substitute an NPC, a camera in the current room, an audio asset and a cutscene. Attach the record to the NPC's Talk interaction. The speech waits for advance. Camera blending finishes before the sound starts. Awaiting an action waits for that action's implementation. Audio playback may continue after the start action completes.

For branching conversation, replace the speech with await(StartDialogue({ dialogueId = "archive_intro" })). Author choices and exits in the Dialogue Designer. Where the voice line is localized, use the speech action's voiceKey or voiceAssetId with the character's configured portrait and lip-sync settings.

LESSON 16

Cross-language calls and includes

Let a single implementation of a puzzle rule serve Lua, GDScript and an action list.

Lua caller · bridge_lua

```
local Script = {}

function Script.run(context)
    local helper = IncludeScript("archive_words")
    helper:Set("last_room", "archive")
    local answer = await(AwaitCall(helper, "Call", "normalize",
                                   Array{"  EAST "}))
    return tostring(answer) .. "/" .. tostring(helper:Get("last_room"))
end

return Script
```

Expect east/archive. IncludeScript returns a call proxy. Set and Get use its shared included-variable store. The proxy does not copy source or give you a reference to the other module's private locals. Lua project records need no header files. A standalone Lua require uses the Lua package loader and its search paths. Use project IDs and IncludeScript or CallScript for records packaged by PCA.

GDScript caller · complete native record

```
func run() -> String:
    var value = await CallScript("archive_words", "normalize", [" WEST "])
    return str(value)
```

From Lua use await(CallScript(id, functionName, Array{arguments})). The shared router also supports PCA Script and BASIC records. Use unique record IDs across languages. RunVisualScript takes a scriptId for a saved graph. RunVisualGraph takes the graph data itself. Return plain numbers, strings, booleans and simple lists or dictionaries for portable results.

LESSON 17

Expose a Lua function as a visual node

Let an author set a puzzle's difficulty from the action editor.

```
local Script = {}

function Script.reward(tries, maximum)
    return math.max(0, maximum - math.max(0, tries - 1))
end

function Script.run(context)
    return Script.reward(3, 10)
end

return Script
```

1. Apply archive_score. Open Functions as Nodes and add a definition with the exact source name reward and the display label Archive Reward.

2. Add two int parameters in order: tries with default 1, then maximum with default 10. Enable Expose as Action and Visual Node. Choose Use Definitions, then Apply Changes and save.

3. Add Call Script Function to a visual graph or supported action list. Choose Archive Reward, enter 3 and 10, and select a numeric result variable. Trigger the action. The result should be 8. A first try should return 10.

Use Definitions accepts the metadata you entered. The Lua source provides the implementation. If the picker is empty, reopen the record and compare the saved function name, enabled state and parameter order.

LESSON 18

Inventory, quests and save slots

Build an optional reward for finding the key. Keep the inventory item definition, the visible pickup and the quest in separate records.

```
local Script = {}

function Script.run(context)
    if IfVariable("key_claimed", true) then return false end
    await(SetVariable({ name = "key_claimed", value = true }))
    await(AddInventory({ itemId = "key", count = 1 }))
    await(SetObjectVisible({ entityId = "key_pickup", visible = false }))
    await(StartQuest({ questId = "archive_search" }))
    await(AddScore({ amount = 5 }))
    return InventoryCount("key")
end

return Script
```

Create Boolean key_claimed with default false, inventory definition key, placed pickup key_pickup, and quest archive_search. Attach the script to the pickup's Take interaction. Test on a fresh save. Confirm one key and a five-point reward. Trigger it again and confirm that neither increases. If a hidden hotspot can still receive input, also disable it with SetObjectEnabled.

To require possession at the door, check HasInventory("key") before unlocking. To consume a single-use key, await RemoveInventory with itemId and count. Mark the door as unlocked separately so using the key once keeps the door open.

Save a checkpoint · fragment

```
await(SaveGameSlot({ slot = 2, saveName = "Before the archive" }))
```

Load that checkpoint with await(LoadGameSlot({ slot = 2 })). Slot is numeric. Save names are display text. Choose a disposable slot when practicing, since saving to an occupied slot can replace a prior checkpoint.

LESSON 19

Use Godot from a Lua record

Draw a temporary overlay using Godot controls. Keep the node alive until the timer finishes, then free it.

```
local Script = {}

function Script.run(context)
    local layer = ClassDB:instantiate("CanvasLayer")
    Runtime:add_child(layer)
    local swatch = ClassDB:instantiate("ColorRect")
    swatch.color = Color(0.08, 0.29, 0.34, 1.0)
    swatch.position = Vector2(24, 32)
    swatch.size = Vector2(270, 48)
    layer:add_child(swatch)
    local label = ClassDB:instantiate("Label")
    label.text = "Archive signal received"
    label.position = Vector2(12, 8)
    swatch:add_child(label)
    await(Wait(1.5))
    layer:queue_free()
    return true
end

return Script
```

Run from a game session. A teal rectangle with a label appears near the top left, then disappears. CanvasLayer draws its control children in screen space. The rectangle owns the label in the scene tree. Freeing the layer frees the temporary controls together. For production HUDs, set mouse_filter to ignore on decorative controls that would otherwise intercept clicks.

Godot values use constructors such as Vector2 and Color. Godot objects expose methods through Lua's colon syntax. For persistent native nodes, choose a parent whose lifetime matches the feature. A separately authored Godot drawing class can implement _draw and be called from Lua through a native PCA record when the overlay needs custom drawing.

LESSON 20

Choose where to store a value

How long should the game remember something? Answer that first. The right place to store it follows from that decision.

A value for one calculation

```
local price = 4
local quantity = 3
local total = price * quantity
```

Inside run, these names exist while that function is running. A later call can calculate a different total without carrying over an old purchase.

A value for the current session

A local declared above the functions belongs to the script instance. With **Keep global instance alive** selected, the same instance can count room arrivals. Programming page 15 shows that counter and the matching event handler.

A value other scripts and save slots need

```
await(SetVariable({ name = "met_keeper", value = true }))
if IfVariable("met_keeper", true) then
    Log("The keeper recognizes the player.")
end
```

This fragment assumes a Boolean game variable with ID **met_keeper** and a starting value of false. Another record can read the same flag later, even after the player changes rooms. Load a game slot to restore the progress captured when that slot was saved.

> A displayed name, a line of dialogue and a saved flag can all describe the same event. Decide which stored value tells the game what actually happened.

LESSON 21

Read a return value

A return value is an answer sent back to the code that asked a question. It can be a word, a number, true, false or another Lua value.

```
local Script = {}

local function can_pay(coins, price)
    return coins >= price
end

function Script.run(context)
    local allowed = can_pay(4, 3)
    Log(tostring(allowed))
    return allowed
end

return Script
```

Follow the call

Lua passes 4 as coins and 3 as price. The comparison succeeds, so can_pay returns true. The caller stores that answer in allowed. Log turns the value into readable output. The next line returns the same answer to whoever started run.

What happens at the boundary?

can_pay(3, 3)	Returns true. The >= comparison includes an exact match.
can_pay(2, 3)	Returns false. The function ran successfully and answered the question.
No explicit return	An ordinary function with no returned values yields nil when its caller assigns the result to a single local.

> For a function in another PCA record, CallScript accepts the record ID and function name. Programming page 19 shows how to capture that answer across records.

LESSON 22

Debugging a Lua record

Work from the first observable failure. A working script with a wrong record ID often looks like a button that does nothing.

The source fails validation

Check for a missing end, an unfinished string, a misplaced return or a copied smart quote. Keep top-level code limited to declarations. Try the first record unchanged and reintroduce one function at a time.

The record runs, but the interaction does nothing

Put Log("entered archive_unlock") on the first executable line of run. No log points to the event or button wiring or the record's enabled state. A log followed by no feedback points to the getter, GUI visibility, element ID or setter. Capture the Boolean returned by SetGUIElementProperty to distinguish lookup failure.

A number or collection has the wrong shape

Log(type(value)) and Log(tostring(value)) separately. Native Godot values may be userdata. Wrap an engine string in tostring before applying Lua patterns. Check whether you received a Lua table or a Godot Array before choosing index 1 or 0.

Catch an error in a pure helper

```
local Script = {}

function Script.run(context)
    local ok, result = pcall(function()
        local n = tonumber("bell")
        assert(n, "A numeric dial setting is required.")
        return n * 2
    end)
    if not ok then Log(tostring(result)) end
    return ok
end

return Script
```

Expect false and the diagnostic in the log. Reserve assertions for assumptions you control. A wrong puzzle answer is normal input and deserves an in-game hint.

REFERENCE A

Event handlers

Subscribe in the Lua Designer and put the corresponding function on the returned Script table. PCA supplies an event dictionary. OnGlobalReady runs specially for a global instance.

OnGameStart	`Script.on_game_start(event)`
OnSceneLoaded	`Script.on_scene_loaded(event)`
OnSceneLeaving	`Script.on_scene_leaving(event)`
OnPlayerChanged	`Script.on_player_changed(event)`
OnCharacterControl	`Script.on_character_control(event)`
OnItemFound	`Script.on_item_found(event)`
OnVideoEnd	`Script.on_video_end(event)`
OnMusicEnd	`Script.on_music_end(event)`
OnDialogueStarted	`Script.on_dialogue_started(event)`
OnDialogueEnded	`Script.on_dialogue_ended(event)`
OnInventoryChanged	`Script.on_inventory_changed(event)`
OnSaveLoaded	`Script.on_save_loaded(event)`
OnCustomEvent	`Script.on_custom_event(event)`
OnGlobalReady	`Script.on_global_ready(event)`
OnInteraction	`Script.on_interaction(event)`
OnTriggerEntered	`Script.on_trigger_entered(event)`
OnTriggerExited	`Script.on_trigger_exited(event)`
OnAttachPointContact	`Script.on_attach_point_contact(event)`

REFERENCE B

Helper functions and proxy methods

Signatures below show Lua argument order. Optional defaults use Lua spelling. Pass ordinary values to synchronous helpers. Wrap asynchronous global calls in await(...). Object methods take the receiver shown before the colon.

await(EndGame(message = "", ask_first = false))

await(QuitToMenu(message = "", ask_first = false))

await(CallScript(script_id, function_name = "run", arguments = Array{}, execution_context = {}))

IncludeScript(script_id)

GetIncludedVariable(script_id, variable_name, fallback = nil)

SetIncludedVariable(script_id, variable_name, value)

await(Action(action_type, properties = {}))

await(RunVisualGraph(graph, execution_context = {}))

await(EmitEvent(event_name, event_context = {}))

GetProjectProperty(property_path, default_value = nil)

SetProjectProperty(property_path, value)

GetSetting(setting_path, default_value = nil)

SetSetting(setting_path, value)

Log(message)

Warn(message)

ScriptError(message)

PCAAssert(condition, message = "PCA3D script assertion failed.")

Breakpoint()

TryCall(callable, arguments = Array{}, fallback = nil)

DoWhile(body, condition, safety_limit = 100000)

ForEach(collection_value, callback)

WhileLoop(condition, body, safety_limit = 100000)

ForLoop(count, body)

HasInventory(item_id, character_id = "player", amount = 1)

InventoryCount(item_id, character_id = "player")

CurrentScene()

ActivePlayer()

QuestState(quest_id)

ObjectiveState(quest_id, objective_id)

Helper functions and proxy methods / continued

HasVisited(scene_id, character_id = "player")

GetGUIProperty(gui_id, property_name, default_value = nil)

SetGUIProperty(gui_id, property_name, value)

GetGUIElementProperty(gui_id, element_id, property_name, default_value = nil)

SetGUIElementProperty(gui_id, element_id, property_name, value)

IfPlayerWasHere(scene_id)

IfPlayerWasNeverHere(scene_id)

IfVariable(variable_id, expected_value = true)

IfInventoryItemWasUsed(item_id)

IfCharacterMoving(character_id = "player")

EndVisualScript(end_parent = false)

End()

NowUnix()

DateTimeNow(utc = false)

DateTimeString(utc = false, use_space = true)

UnixToDateTime(unix_time, _utc = false)

RegexSearch(pattern, text, all_matches = false)

ReadTextFile(path, default_value = "")

WriteTextFile(path, text)

ReadJSONFile(path, default_value = nil)

WriteJSONFile(path, value, indent = "\t")

FileExists(path)

DirectoryExists(path)

StringLength(value)

StringSplit(value, delimiter = " ", allow_empty = true)

Lower(value)

Upper(value)

Trim(value)

Contains(value, search, case_sensitive = true)

StartsWith(value, prefix, case_sensitive = true)

EndsWith(value, suffix, case_sensitive = true)

Replace(value, search, replacement)

Split(value, delimiter = " ", allow_empty = true)

Join(values, delimiter = "")

Helper functions and proxy methods / continued

Clamp(value, minimum, maximum)

Lerp(from_value, to_value, weight)

RandomInteger(minimum, maximum)

Randomize(seed_value = -1)

RandomFloat(minimum = 0.0, maximum = 1.0)

RandomInt(minimum, maximum)

RandomNumber(minimum = 0.0, maximum = 1.0)

Choose(values, fallback = nil)

Shuffle(values)

Distance(first, second)

Format(template, values = {})

ToJSON(value, indent = "")

FromJSON(text_value, fallback = nil)

IsValid(value)

TypeOf(value)

registry:Get(record_id)

registry:Has(record_id)

registry:IDs()

record:Get(property_path, default_value = nil)

record:Set(property_path, value)

record:Exists()

record:ToDictionary()

record:AddInventory(inventory_id, amount = 1)

record:RemoveInventory(inventory_id, amount = 1)

record:AddDestinationPath(x, y = nil, z = nil, ignore_navigation_mesh_rules = false)

record:ClearDestinationPath()

record:WalkPath(speed = 0.0) · use AwaitCall for completion

record:WalkTo(position_value, speed = 0.0, ignore_navigation = false) · use AwaitCall for completion

record:Teleport(position_value)

record:ChangeRoom(room_id, position_value = nil, spawn_point_or_edge = "", facing = nil) · use AwaitCall for completion

record:FaceCharacter(target_id)

record:FaceCoordinates(position_value)

record:Say(text, portrait_id = "", voice_asset_id = "", voice_key = "") · use AwaitCall for completion

Helper functions and proxy methods / continued

record:PlayAnimation(animation_name)

record:StopAnimation()

record:SetVisible(visible_value)

record:SetEnabled(enabled_value)

included:Call(function_name = "run", arguments = Array{}, context = {}) · use AwaitCall for completion

included:Run(context = {}) · use AwaitCall for completion

included:Get(variable_name, fallback = nil)

included:Set(variable_name, value)

included:Emit(event_name, context = {}) · use AwaitCall for completion

Global:Get(script_id)

Global:IDs()

PCA:Get(collection_name)

PCA:Has(collection_name)

Lua-specific globals: await(Wait(seconds)), await(Parallel(functionA, functionB, ...)), and await(AwaitCall(target, methodName, ...)).

REFERENCE C

Gameplay action fields

All 206 callable actions, alphabetically. Use await(Name({ field = value })) inside a Lua module. For actions with no fields use await(Name({})). Field descriptions identify editor pickers, types and explicit defaults. Supply the fields your action needs.

ActivateCamera / Activate / Blend to Camera

cameraId — Camera [cameras]

duration — Blend Seconds (-1 = camera default) [float] · default -1.0

wait — Wait for Blend [bool] · default true

AddAssetToScene / Add Asset to Scene

assetId — Asset [assets]

id — New ID [string]

name — Name [string]

position — XYZ [vector3]

enabled — Enabled [bool]

visible — Visible [bool]

AddCharacterToScene / Add Character To Scene

characterId — Character [characters]

id — New ID [string]

name — Name [string]

position — XYZ [vector3]

enabled — Enabled [bool]

visible — Visible [bool]

animation — Animation (optional) [string]

AddCharacterWaypoint / Character Waypoints: Add Waypoint

characterId — Character [characters]

sceneId — Waypoint Room [scenes]

waypointId — Waypoint / Object (optional) [entities_for_scene]

position — XYZ when no waypoint [vector3]

ignoreNavigation — Ignore Navigation Mesh [bool] · default false

Gameplay action fields / continued

AddDialogueChoice / Add Dialogue Choice

dialogueId — Dialogue [dialogues]

nodeId — Choice Node ID [dialogue_choice_nodes]

choiceId — New Choice ID [string]

text — Text [string]

next — Next Node [dialogue_nodes]

enabled — Enabled [bool] · default true

repeatable — Always Allow Repeat [bool] · default false

voiceAssetId — Voice Audio [audio_assets]

voiceKey — Voice Key (optional) [string]

typewriter — Typewriter Effect [bool] · default false

typewriterSpeed — Typing Speed (characters/second) [float] · default 30.0

typewriterSound — Typing Sound [string] · default "default" · default, none, custom

typewriterSoundAssetId — Typing Sound Asset [audio_assets] · default ""

AddDialogueChoiceNode / Add Dialogue Choice Node

dialogueId — Dialogue [dialogues]

nodeId — New Node ID [string]

AddDialogueLine / Add Dialogue Line

dialogueId — Dialogue [dialogues]

nodeId — New Node ID [string]

speakerId — Speaker [characters]

text — Text [string]

next — Next Node [dialogue_nodes]

returnToPrevious — Return to previous node [bool] · default true

voiceAssetId — Voice Audio [audio_assets]

voiceKey — Voice Key (optional) [string]

typewriter — Typewriter Effect [bool] · default false

typewriterSpeed — Typing Speed (characters/second) [float] · default 30.0

typewriterSound — Typing Sound [string] · default "default" · default, none, custom

typewriterSoundAssetId — Typing Sound Asset [audio_assets] · default ""

AddInventory / Add Inventory

itemId — Item [inventory]

count — Count [int] · default 1

Gameplay action fields / continued

AddScore / Add Score

amount — Amount [int] · default 1

AlignAttachPoints / Align Attach Points

entityId — Object / Character [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

alignRotation — Align Rotation [bool] · default true

AttachObject / Attach Object

entityId — Object [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

AttachParticleEffect / Attach Particle Effect

entityId — Particle Effect [entities]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

offset — Attachment Offset [vector3] · default [0, 0, 0]

AttachPoints / Attach Points

entityId — Object / Character [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

alignRotation — Align Rotation [bool] · default true

suspendMovement — Suspend Movement [bool] · default true

AttachPointsTouching / Attach Points Touching

entityId — Object / Character [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

tolerance — Contact Tolerance [float] · default 0.0

resultVariable — Store Result in Variable [string]

Gameplay action fields / continued

CallScriptFunction / Call Script Function

functionKey — Function [string]

parameters — Parameters [variant] · default {}

resultVariable — Result Variable [string]

CameraOrbit / Camera Orbit

entityId — Target Object [entities]

speed — Degrees Per Second [float] · default 45

distance — Distance [float] · default 5

mode — Orbit Mode [string] · default "360" · 360, 180, 180_back

repetitions — Repeat Count [int] · default 1

clockwise — Clockwise [bool] · default false

height — Height [float] · default 2

CameraShake / Camera Shake

direction — Direction [string] · default "both" · both, horizontal, vertical

duration — Duration (seconds) [float] · default 0.5

power — Power [float] · default 0.25

CancelPreposition / Cancel Two-Object Command

No fields in the editor schema.

CarouselAddImage / Carousel: Add Image

guiId — GUI [guis]

elementId — Image Carousel [gui_elements]

assetId — Image Asset [image_assets]

CarouselClearImages / Carousel: Clear Images

guiId — GUI [guis]

elementId — Image Carousel [gui_elements]

CarouselNext / Carousel: Next Image

guiId — GUI [guis]

elementId — Carousel [gui_elements]

CarouselPrevious / Carousel: Previous Image

guiId — GUI [guis]

elementId — Carousel [gui_elements]

Gameplay action fields / continued

CarouselRemoveImage / Carousel: Remove Image

guiId — GUI [guis]

elementId — Image Carousel [gui_elements]

assetId — Image Asset [image_assets]

CenterGUIModel / Center GUI Model

guiId — GUI [guis]

elementId — 3D Model [gui_elements]

ChangeCamera / Change Camera

sceneId — Room [scenes]

cameraId — Camera View [cameras_for_scene]

ChangeCharacterDescription / Change Character Description

characterId — Character / Player [characters]

description — Description [string]

ChangeCharacterProperty / Change Character Property

characterId — Character [characters]

property — Property [character_properties]

value — To [variant]

ChangeGUIElementProperty / Change Gui Element Property

guiId — GUI [guis]

elementId — Element / Group [gui_elements]

property — Property [gui_element_properties]

value — To [variant]

ChangeGUIProperty / Change Gui Property

guiId — GUI [guis]

property — Property [gui_properties]

value — To [variant]

ChangeGameCursor / Change Game Cursor

cursorId — Cursor [cursors]

ChangeGameSetting / Change Game Setting

setting — Setting [game_settings]

value — To [variant]

Gameplay action fields / continued

ChangeInventoryDescription / Change Inventory Description

itemId — Inventory Item [inventory]

description — Description [string]

ChangeInventoryQuantity / Change Inventory Quantity

characterId — Player / Character [characters]

itemId — Inventory Item [inventory]

amount — Change By [int]

ChangeObjectDescription / Change Object Description

sceneId — Scene [scenes]

entityId — Object [entities_for_scene]

description — Description [string]

ChangeObjectProperty / Change Object Property

sceneId — Scene [scenes]

entityId — Object [entities_for_scene]

property — Property [object_properties]

value — To [variant]

ChangeProjectProperty / Change Project Property

property — Property [project_properties]

value — To [variant]

ChangeQuestObjectiveProperty / Quest: Change Objective Property

questId — Quest [quests]

stageId — Stage [quest_stages]

objectiveId — Objective [quest_objectives]

property — Objective Property [quest_objective_properties]

value — New Value [variant]

ChangeQuestProperty / Quest: Change Property

questId — Quest [quests]

property — Quest Property [quest_properties]

value — New Value [variant]

ChangeQuestStageProperty / Quest: Change Stage Property

questId — Quest [quests]

stageId — Stage [quest_stages]

property — Stage Property [quest_stage_properties]

Gameplay action fields / continued

value — New Value [variant]

ChangeRegionDescription / Change Region Description

sceneId — Scene [scenes]

regionId — Region [regions_for_scene]

description — Description [string]

ChangeScene / Change Scene

sceneId — Destination scene [scenes]

spawnId — Spawn (optional) [spawns_for_scene]

position — XYZ when no spawn [vector3]

ChangeSceneDescription / Change Scene Description

sceneId — Scene [scenes]

description — Description [string]

ChangeSubtitleLanguage / Change Subtitle Language

language — Language [string]

ChangeVariable / Change Variable

name — Variable [variables]

operation — Operation [string] · set, add, subtract, multiply, divide, append, toggle

value — Value [variant]

ChangeVerb / Change Current Verb Command

verbId — New Verb [verbs]

ChangeVoiceLanguage / Change Voice Language

language — Language [string]

CharacterTeleport / Character Instant Teleport To

characterId — Character [characters]

sceneId — Object Room [scenes]

targetId — Object (optional) [entities_for_scene]

position — XYZ [vector3]

CharacterWalkPath / Character Walk Through Waypoints

characterId — Character [characters]

sceneId — Default Waypoint Room [scenes]

waypointIds — Waypoint IDs / Positions [variant] · default []

speed — Speed [float]

Gameplay action fields / continued

ignoreNavigation — Ignore Navigation Mesh [bool] · default false

CharacterWalkTo / Character Walk To

characterId — Character [characters]

sceneId — Object Room [scenes]

targetId — Object (optional) [entities_for_scene]

position — XYZ [vector3]

animation — Walking Animation [animations_for_scene]

speed — Speed [float]

CharacterWalkWaypoints / Character Waypoints: Walk Queue

characterId — Character [characters]

speed — Speed [float]

ClearCharacterWaypoints / Character Waypoints: Clear Queue

characterId — Character [characters]

CloseInteractionCoin / Close Interaction Coin

No fields in the editor schema.

CompleteObjective / Quest: Complete Objective

questId — Quest [quests]

stageId — Stage [quest_stages]

objectiveId — Objective [quest_objectives]

CompleteQuest / Quest: Complete Quest

questId — Quest [quests]

CreatePrimitive / Create Primitive

primitive — Primitive [string] · box, sphere, cylinder, capsule, plane

id — ID [string]

position — XYZ [vector3]

CreateProcedural / Create Procedural

procedural — Procedural Type [string]

id — ID [string]

position — XYZ [vector3]

Gameplay action fields / continued

CreateVariable / Create Variable

name — Variable ID [string]

variableType — Type [string] · string, boolean, number, array

value — Initial Value [variant]

CycleInteractionVerb / Cycle Verb

direction — Direction [int] · default 1

DespawnObject / Despawn Scene Object

sceneId — Room [scenes]

entityId — Scene Object [entities_for_scene]

DetachObject / Detach Object

entityId — Object [entities]

DetachParticleEffect / Detach Particle Effect

entityId — Particle Effect [entities]

DisableGUI / Disable Gui

guiId — GUI [guis]

DisableGUIElement / Disable Gui Element

guiId — GUI [guis]

elementId — Element / Group [gui_elements]

EnableGUI / Enable Gui

guiId — GUI [guis]

EnableGUIElement / Enable Gui Element

guiId — GUI [guis]

elementId — Element / Group [gui_elements]

EndCutscene / Cutscene: End

restoreCamera — Restore Previous Camera [bool] · default true

EndDialogue / End Dialogue

No fields in the editor schema.

ExecutePreposition / Execute Two-Object Command

verbId — Verb [verbs]

sourceId — First Object (entity:id or inventory:id) [string]

targetId — Second Object (entity:id or inventory:id) [string]

Gameplay action fields / continued

FaceCharacter / Face Character

characterId — Character [characters]

targetCharacterId — Face Character [characters]

FaceCoordinates / Face Coordinates

characterId — Character [characters]

position — XYZ [vector3]

FaceObject / Face Object

characterId — Character [characters]

sceneId — Object Room [scenes]

targetId — Object [entities_for_scene]

FadeMusic / Music: Fade

volume — Target Volume [float] · default 0.0

duration — Duration (seconds) [float] · default 1.0

FadeScreen / Screen Fade

direction — Direction [string] · default "out" · out, in

color — Color [color] · default "#000000"

duration — Duration (seconds) [float] · default 0.5

FadeSound / Sound: Fade

channel — Channel / ID (empty = all) [string]

volume — Target Volume [float] · default 0.0

duration — Duration (seconds) [float] · default 1.0

FailObjective / Quest: Fail Objective

questId — Quest [quests]

stageId — Stage [quest_stages]

objectiveId — Objective [quest_objectives]

FailQuest / Quest: Fail Quest

questId — Quest [quests]

FollowCharacter / Follow Character

characterId — Character [characters]

targetCharacterId — Character to Follow [characters]

distance — Distance Behind [float]

speed — Speed [float]

Gameplay action fields / continued

GetAttachPoint / Get Attach Point

entityId — Object / Character [entities]

point — Attach Point [string]

resultVariable — Store Result in Variable [string]

GetAttachPointContacts / Get Attach Point Contacts

entityId — Object / Character [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

tolerance — Contact Tolerance [float] · default 0.0

resultVariable — Store Result in Variable [string]

GetControlState / Get Control State

resultVariable — Result Variable [variables]

HideGUI / Hide Gui

guiId — GUI [guis]

HideGUIElement / Hide Gui Element

guiId — GUI [guis]

elementId — Element / Group [gui_elements]

HidePlayerCharacter / Hide Player Character

hidden — Hidden [bool]

InteractNearest / Interact with Nearest Hotspot

verbId — Verb [verbs]

IsCurrentRoom / Is Current Room

sceneId — Scene [scenes]

Jump / Jump

characterId — Character [characters]

ListBoxAddItem / List Box: Add Item

guiId — GUI [guis]

elementId — List Box [gui_elements]

itemId — New Item ID [string]

text — Item Text [string]

iconAssetId — Optional Icon [image_assets]

Gameplay action fields / continued

disabled — Disabled [bool]

insertIndex — Insert Index (-1 = End) [int] · default -1

ListBoxClearItems / List Box: Clear Items

guiId — GUI [guis]

elementId — List Box [gui_elements]

ListBoxDeleteItem / List Box: Delete Item

guiId — GUI [guis]

elementId — List Box [gui_elements]

itemId — Item ID [string]

ListBoxModifyItem / List Box: Modify Item

guiId — GUI [guis]

elementId — List Box [gui_elements]

itemId — Item ID [string]

property — Property [string] · text, id, iconAssetId, disabled, selected

value — New Value [variant]

LoadGame / Load Game

No fields in the editor schema.

LoadGameSlot / Load Game Slot

slot — Slot [int]

MeltSnow / Melt Accumulated Snow

sceneId — Scene (empty = current) [scenes]

duration — Melt Duration (seconds) [float] · default 2.0

pace — Amount Per Second (used only when duration = 0) [float] · default 0.0

waitForCompletion — Wait Until Melting Finishes [bool] · default true

MoveNPCToRoom / Move NPC to a Different Room

characterId — Character [characters]

sceneId — Room [scenes]

spawnId — Spawn (optional) [spawns_for_scene]

position — XYZ when no spawn [vector3]

OpenInteractionCoin / Open Interaction Coin

entityId — Object / Character [entities]

Gameplay action fields / continued

PauseAnimation / Animation: Pause

sceneId — Room [scenes]

entityId — Character / Object [entities_for_scene]

PauseFMV / Pause FMV

No fields in the editor schema.

PauseMusic / Music: Pause

No fields in the editor schema.

PauseSound / Sound: Pause

channel — Channel / ID (empty = all) [string]

PauseVoice / Pause Voice

No fields in the editor schema.

PlayAnimation / Play Animation

sceneId — Room [scenes]

entityId — Character / Object [entities_for_scene]

clip — Animation [animations_for_scene]

PlayGUIModelAnimation / Play GUI Model Animation

guiId — GUI [guis]

elementId — 3D Model [gui_elements]

animation — Animation [string] · default ""

loop — Loop [bool] · default true

speed — Speed [float] · default 1.0

PlayGameplayRecording / Play Gameplay Recording

path — Recording Path [string] · default "user://walkthrough.json"

speed — Playback Speed [float] · default 1

PlayMusic / Play Music

assetId — Music Asset [audio_assets]

skippable — Skippable [bool]

PlaySound / Play Sound

assetId — Sound Asset [audio_assets]

channel — Channel / ID [string]

Gameplay action fields / continued

PlayVideo / Play Video

assetId — Video Asset [video_assets]

fullScreen — Full Screen [bool]

skippable — Skippable [bool]

muted — Muted [bool]

PlayVoice / Play Voice

voiceAssetId — Voice Audio [audio_assets]

voiceKey — Voice Key (optional) [string]

text — Text [string]

wait — Wait for Voice [bool] · default false

RemoveAttachPoint / Remove Attach Point

entityId — Object / Character [entities]

point — Attach Point [string]

RemoveDialogueChoice / Remove Dialogue Choice

dialogueId — Dialogue [dialogues]

nodeId — Choice Node ID [dialogue_choice_nodes]

choiceId — Choice ID [dialogue_choices]

RemoveDialogueNode / Remove Dialogue Node

dialogueId — Dialogue [dialogues]

nodeId — Dialogue Node ID [dialogue_nodes]

RemoveInventory / Remove Inventory

itemId — Item [inventory]

count — Count [int] · default 1

ResetDialogueHistory / Reset Dialogue History

dialogueId — Dialogue [dialogues]

nodeId — Dialogue Node ID [dialogue_nodes]

choiceId — Choice ID [dialogue_choices]

ResetObjective / Quest: Reset Objective

questId — Quest [quests]

stageId — Stage [quest_stages]

objectiveId — Objective [quest_objectives]

Gameplay action fields / continued

ResetQuest / Quest: Reset Quest

questId — Quest [quests]

ResumeAnimation / Animation: Resume

sceneId — Room [scenes]

entityId — Character / Object [entities_for_scene]

ResumeFMV / Resume FMV

mode — Playback Limit [string] · default "end" · end, point, interrupt

endTime — Play Until (seconds) [float] · default 0

scriptId — Completion Script [scripts_all] · default ""

ResumeMusic / Music: Resume

No fields in the editor schema.

ResumeSound / Sound: Resume

channel — Channel / ID (empty = all) [string]

ResumeVoice / Resume Voice

No fields in the editor schema.

RunVisualScript / Run Visual Script

scriptId — Script [scripts_all]

SaveGame / Save Game

No fields in the editor schema.

SaveGameSlot / Save Game Slot

slot — Slot [int]

saveName — Save Name (optional) [string]

SaveGameWidgetCommand / Save Game Widget: Command

guiId — GUI [guis]

elementId — Save Game Widget [gui_elements]

command — Command [string] · previous, next, up, down, refresh, select-first, select-previous, select-next, load-selected, save-selected, rename-selected, delete-selected, select-slot, load-slot, save-slot, rename-slot, delete-slot

slot — Explicit Slot (-1 = Selected) [int] · default -1

Say / Say

speakerId — Speaker [characters]

text — Text [string]

waitForContinue — Wait for Continue [bool] · default false

Gameplay action fields / continued

voiceAssetId — Voice Audio [audio_assets] · default ""

voiceKey — Voice Key (optional) [string] · default ""

typewriter — Typewriter Effect [bool] · default false

typewriterSpeed — Typing Speed (characters/second) [float] · default 30.0

typewriterSound — Typing Sound [string] · default "default" · default, none, custom

typewriterSoundAssetId — Typing Sound Asset [audio_assets] · default ""

ScreenTransition / Screen Transition

transitionType — Transition [string] · default "dissolve" · dissolve, fade, wipe, match-cut, whip-pan, iris-wipe, pixelation, glitch, comic-panel

duration — Duration (seconds) [float] · default 0.8

color — Color [color] · default "#000000"

direction — Direction [string] · default "left" · left, right, up, down

intensity — Intensity [float] · default 1.0

SeekFMV / Seek FMV

time — Time [float] · default 0

mode — Playback Limit [string] · default "end" · end, point, interrupt

endTime — Play Until (seconds) [float] · default 0

scriptId — Completion Script [scripts_all] · default ""

SeekMusic / Skip Ahead in Soundtrack

minutes — Minute [int]

seconds — Second [float]

SeekVideo / Seek Video

minutes — Minute [int]

seconds — Second [float]

SeekVoice / Seek Voice

seconds — Seconds [float] · default 0.0

Set2DCamera / Set 2D Camera

sceneId — Room (empty = current) [scenes]

position — Camera Position XY [vector2] · default [0.0, 0.0]

zoom — Camera Zoom XY [vector2] · default [1.0, 1.0]

rotation — Camera Rotation (degrees) [float] · default 0.0

Gameplay action fields / continued

Set2DEntityDepth / Set 2D Entity Depth

sceneId — Room (empty = current) [scenes]

entityId — 2D Character / Object [entities_for_scene]

depth — Drawing Depth [int] · default 100

Set2DLayerVisible / Show / Hide 2D Layer

sceneId — Room (empty = current) [scenes]

layerId — 2D Layer ID [string]

visible — Visible [bool] · default true

Set2DParallax / Set 2D Layer Parallax

sceneId — Room (empty = current) [scenes]

layerId — 2D Layer ID [string]

ratio — Parallax Ratio XY [vector2] · default [1.0, 1.0]

SetAttachPoint / Set Attach Point

entityId — Object / Character [entities]

point — Attach Point [string]

position — Position [vector3] · default [0, 0, 0]

rotation — Rotation (degrees) [vector3] · default [0, 0, 0]

bone — Bone Name (optional) [string]

space — Coordinate Space [string] · default "object" · object, model, sprite_uv

contactRadius — Contact Radius [float] · default 0.1

enabled — Enabled [bool] · default true

SetAutoInputSwitching / Automatic Input Switching

enabled — Enabled [bool] · default true

SetCameraBehavior / Set Advanced Camera Behavior

cameraId — Camera (empty = active) [cameras]

enabled — Enable Advanced Camera [bool] · default false

mode — Camera Behavior [string] · default "fixed" · fixed, follow, group, framed, path

targetId — Follow Target [entities] · default "player"

targetIds — Additional Target IDs (comma separated) [string] · default ""

lookAtId — Look At Target (optional) [entities] · default ""

targetOffset — Target Offset XYZ [vector3] · default [0.0, 1.3, 0.0]

damping — Follow Damping (seconds) [float] · default 0.2

blendDuration — Default Camera Blend (seconds) [float] · default 1.0

Gameplay action fields / continued

ease — Blend Curve [string] · default "sine" · linear, sine, cubic, quint

priority — Camera Priority [int] · default 0

autoFrame — Automatically Frame Group [bool] · default true

minDistance — Minimum Group Distance / View Height [float] · default 3.0

maxDistance — Maximum Group Distance / View Height [float] · default 18.0

framingPadding — Group Framing Padding [float] · default 1.3

deadZoneWidth — Framing Zone Width (0–1) [float] · default 0.4

deadZoneHeight — Framing Zone Height (0–1) [float] · default 0.3

pathPoints — Camera Path Points (XYZ array) [variant] · default []

respectBounds — Respect Room Camera Bounds [bool] · default true

SetCameraPriority / Set Camera Priority

cameraId — Camera [cameras]

priority — Camera Priority [int] · default 10

SetCameraTargets / Set Camera Targets

cameraId — Camera (empty = active) [cameras]

targetId — Follow Target [entities] · default "player"

targetIds — Additional Target IDs (comma separated) [string] · default ""

lookAtId — Look At Target (optional) [entities]

SetCharacterActionEnabled / Enable / Disable Character Action

characterId — Character [characters]

actionEvent — Mouse / Trigger Action [interaction_events]

enabled — Enabled [bool]

SetControlSetting / Set Control Setting

setting — Setting [string] · default "coinEnabled" · enabled, cursorAdaptation, cyclingEnabled, movementProfile, autoSwitchInput, inputDevice, autoSwitchMovement, mouseProfile, gamepadProfile, touchProfile, proximityInteraction, proximityRadius, showHints, controllerIcons, coinEnabled, dragThreshold, gamepadLookSpeed

value — Value [variant] · default true

SetDialogueChoiceEnabled / Enable / Disable Dialogue Choice

dialogueId — Dialogue [dialogues]

nodeId — Choice Node ID [dialogue_choice_nodes]

choiceId — Choice ID [dialogue_choices]

enabled — Enabled [bool] · default true

Gameplay action fields / continued

SetFlashlightMode / Enable / Disable Flashlight Mode

enabled — Flashlight Enabled [bool] · default true

strength — Light Strength [float] · default 1.0

radius — Light Radius [float] · default 180.0

shape — Light Shape [string] · circle, spherical, square, rectangle, triangle

SetGameLanguage / Set Game Language

language — Language Code [string]

persist — Save Player Preference [bool] · default true

SetGlow / Set Glow

entityId — Entity [entities]

enabled — Enabled [bool] · default true

variable — Variable [string]

SetInputDevice / Set Input Device

device — Input Device [string] · default "auto" · auto, mouse-keyboard, gamepad, touch

SetInteractionApproachEnabled / Enable / Disable Automatic Interaction Approach

enabled — Automatic Approach Enabled [bool] · default true

distance — Default Stop Distance [float] · default 0.35

faceTarget — Face Target After Approaching [bool] · default true

SetInteractionMode / Set Interaction Mode

mode — Interaction Mode [string] · default "context" · context, verb-single-click, verb-double-click, verb-direct, verb-direct-double-click, direct, cycling-cursor, verb-coin, verb-coin-first

SetLipSyncEnabled / Enable Lip Sync

characterId — Character [characters]

enabled — Enabled [bool] · default true

SetMotion / Set Motion

sceneId — Room [scenes]

entityId — Object [entities_for_scene]

enabled — Enabled [bool]

variable — State Variable (optional) [variables]

SetMouseClicksEnabled / Enable Mouse Clicks

enabled — Enabled [bool] · default true

Gameplay action fields / continued

SetMousePointerVisible / Show Mouse Pointer

visible — Visible [bool] · default true

SetMousePointingEnabled / Enable Mouse Pointing

enabled — Enabled [bool] · default true

SetMovementProfile / Set Movement Profile

profile — Movement Profile [string] · default "point-and-click" · legacy, point-and-click, direct, first-person, drag-to-move

SetObjectActionEnabled / Enable / Disable Object Action

sceneId — Room [scenes]

entityId — Object [entities_for_scene]

actionEvent — Mouse / Trigger Action [interaction_events]

enabled — Enabled [bool]

SetObjectEnabled / Enable / Disable Object

sceneId — Room [scenes]

entityId — Object [entities_for_scene]

enabled — Enabled [bool]

SetObjectInteractionApproach / Set Object Interaction Approach

sceneId — Object Room [scenes]

entityId — Object / Character [entities_for_scene]

mode — Approach Override [string] · inherit, enabled, disabled

distance — Stop Distance (0 = global) [float] · default 0.0

faceTarget — Face Target When Override Is Enabled [bool] · default true

SetObjectVisible / Hide / Unhide Object

sceneId — Room [scenes]

entityId — Object [entities_for_scene]

visible — Visible [bool]

SetPlayerInputEnabled / Enable / Disable Player Input

enabled — Enabled [bool] · default true

SetProximityInteractionEnabled / Proximity Interaction

enabled — Enabled [bool] · default true

Gameplay action fields / continued

SetQuestHidden / Quest: Hide / Unhide

questId — Quest [quests]

hidden — Hidden [bool] · default true

SetQuestStage / Quest: Set Stage

questId — Quest [quests]

stageId — Stage [quest_stages]

SetQuestTracked / Quest: Track / Untrack

questId — Quest [quests]

tracked — Tracked [bool] · default true

SetRetroStyle / Set Retro Rendering Style

style — Retro Rendering Style [string] · default "psx" · none, psx, n64, horror-fog

strength — Strength [float] · default 1.0

pixelSize — Retro Pixel Size [float] · default 3.0

SetRoomCameraBounds / Change Room Camera Bounds

sceneId — Room [scenes]

enabled — Bounds Enabled [bool]

minX — Left / Minimum X [float]

maxX — Right / Maximum X [float]

minZ — Top / Minimum Z [float]

maxZ — Bottom / Maximum Z [float]

SetRoomDimension / Set Room Dimension

sceneId — Room (empty = current) [scenes]

dimension — Room Dimension [string] · default "3d" · 3d, 2d, 2.5d

keep3DLayer — Keep 3D Layer [bool] · default true

SetRuntimeControlsEnabled / Set Runtime Controls Enabled

enabled — Enabled [bool] · default true

SetSaveSlotImage / Set Save Slot Image

assetId — Save Slot Image [image_assets]

SetSceneBrightness / Set Room Brightness

sceneId — Scene [scenes]

brightness — Brightness (0 = Dark, 1 = Normal) [float] · default 1.0

Gameplay action fields / continued

SetSceneTint / Set Room Tint

sceneId — Scene [scenes]

tintColor — Tint Color [color] · default "#ffffff"

strength — Tint Strength (0 = Off, 1 = Full) [float] · default 1.0

SetSnowAccumulation / Set Snow Accumulation

sceneId — Scene (empty = current) [scenes]

amount — Accumulation Amount (0 to 1) [float] · default 1.0

duration — Transition Duration (seconds, 0 = authored pace) [float] · default 0.0

pace — Amount Per Second (used only when duration = 0) [float] · default 0.0

waitForCompletion — Wait Until Accumulation Finishes [bool] · default true

SetTriggerEnabled / Enable / Disable Trigger

sceneId — Room [scenes]

entityId — Trigger [triggers_for_scene]

enabled — Enabled [bool]

SetVariable / Set Variable

name — Variable [variables]

value — Value [variant] · default true

SetVerbUsageEnabled / Enable / Disable Verb Command Usage

enabled — Enabled [bool]

SetVoiceSpeaker / Set Voice Speaker

characterId — Character [characters]

ShakeScreen / Shake Screen

direction — Direction [string] · default "both" · both, horizontal, vertical

duration — Duration (seconds) [float] · default 0.5

power — Power (pixels) [float] · default 8

ShowGUI / Show Gui

guiId — GUI [guis]

ShowGUIElement / Show Gui Element

guiId — GUI [guis]

elementId — Element / Group [gui_elements]

Gameplay action fields / continued

ShowMessage / Show Message

text — Text [string]

duration — Duration (seconds) [float] · default 1.8

blockInput — Block Player Input [bool] · default false

voiceAssetId — Voice Audio [audio_assets] · default ""

voiceKey — Voice Key (optional) [string] · default ""

typewriter — Typewriter Effect [bool] · default false

typewriterSpeed — Typing Speed (characters/second) [float] · default 30.0

typewriterSound — Typing Sound [string] · default "default" · default, none, custom

typewriterSoundAssetId — Typing Sound Asset [audio_assets] · default ""

SpawnObject / Spawn Scene Object

sceneId — Room [scenes]

entityId — Scene Object [entities_for_scene]

StartCutscene / Cutscene: Start

cutsceneId — Cutscene [cutscenes]

StartDialogue / Start Dialogue

dialogueId — Dialogue [dialogues]

StartFMV / Start FMV

assetId — Movie Asset ID [video_assets] · default ""

mode — Playback Limit [string] · default "end" · end, point, interrupt

endTime — Play Until (seconds) [float] · default 0

scriptId — Completion Script [scripts_all] · default ""

StartGameplayRecording / Start Gameplay Recording

path — Recording Path [string] · default "user://walkthrough.json"

StartQuest / Start Quest

questId — Quest [quests]

stageId — Initial Stage (optional) [quest_stages]

StopAnimation / Stop Animation

sceneId — Room [scenes]

entityId — Character / Object [entities_for_scene]

Gameplay action fields / continued

StopAttachPointWatch / Stop Attach Point Watch

watchId — Watch ID [string]

StopCameraOrbit / Stop Camera Orbit

No fields in the editor schema.

StopFMV / Stop FMV

No fields in the editor schema.

StopFollowCharacter / Stop Follow Character

characterId — Character [characters]

StopGameplayPlayback / Stop Gameplay Playback

No fields in the editor schema.

StopGameplayRecording / Stop Gameplay Recording

No fields in the editor schema.

StopMusic / Stop Music

No fields in the editor schema.

StopScreenShake / Stop Screen Shake

No fields in the editor schema.

StopSound / Stop Sound

channel — Channel / ID [string]

StopStartupSplashAudio / Stop Startup Splash Audio

No fields in the editor schema.

StopVideo / Stop Video

No fields in the editor schema.

StopVoice / Stop Voice

No fields in the editor schema.

StopZoomBlur / Stop Zoom Blur

No fields in the editor schema.

SwitchCharacter / Switch Character

characterId — Active Player [characters]

Gameplay action fields / continued

ToggleGlow / Toggle Glow

entityId — Entity [entities]

variable — Variable [string]

ToggleMotion / Toggle Motion

sceneId — Room [scenes]

entityId — Object [entities_for_scene]

variable — State Variable (optional) [variables]

ToggleRGBString / Toggle Rgb String

entityId — Entity [entities]

variable — Variable [string]

TransferInventory / Change Inventory from Player to Player

fromCharacterId — From Player [characters]

toCharacterId — To Player [characters]

itemId — Inventory Item [inventory]

count — Quantity [int]

UnlockHint / Unlock Hint Topic / Answer

topicId — Topic ID [string]

answerNumber — Answer Number [int] · default 1

WatchAttachPointContacts / Watch Attach Point Contacts

watchId — Watch ID [string]

entityId — Object / Character [entities]

point — Attach Point [string]

targetId — Target Object [entities]

targetPoint — Target Attach Point [string]

tolerance — Contact Tolerance [float] · default 0.0

swept — Detect Between Frames [bool] · default true

ZoomBlur / Zoom Blur

strength — Strength [float] · default 0.7

duration — Duration (seconds) [float] · default 0.6

centerX — Center X (0–1) [float] · default 0.5

centerY — Center Y (0–1) [float] · default 0.5

clearRadius — Clear Center Radius [float] · default 0.08

direction — Direction [string] · default "inward" · inward, outward

Gameplay action fields / continued

includeGUI — Include GUI in Effect [bool] · default false

wait — Wait Until Finished [bool] · default false

REFERENCE D

Designer property paths

These are the curated field paths from the 0.86.3.4 designer catalog. Resolved records can expose additional custom fields. Read with record:Get(path, fallback) and write supported fields with record:Set(path, value). A definition change may take effect when its instance is rebuilt.

Use Character, Objects, Scene, Inventory, GUI, Dialogue, Quest, Asset, Cursor, Verb, Fonts, Scripts, PCAScripts, LuaScripts, BasicScripts and VisualScript for their corresponding records. Camera and spawn records belong to scenes. Objectives belong to quest stages. Refer to a nested record through its parent or its dedicated gameplay action.

ASSETS

custom [dictionary] Project-defined asset properties.

files [array · read only] Files belonging to this asset. Managed by the Asset Designer.

folder [text] Asset database folder.

format [text] Asset file format.

id [text · read only] Stable asset ID.

importedAt [text · read only] Import timestamp.

kind [text · read only] Model, image, sprite sheet, image animation, audio, video, font, data, or other.

mainFile [text] Primary asset filename. Changing this does not import or reload a file; use the Asset Designer.

metadata [dictionary] PCA3D/Godot import metadata.

metadata.animations [array · read only] Discovered animation clip metadata. Refreshed by model import and clip discovery.

metadata.godotPath [text · read only] Resolved res:// asset path. Managed by asset import.

metadata.sourceFormat [text · read only] Original/imported source format.

name [text] Asset display/editor name.

BASIC SCRIPTS

custom [dictionary] Project-defined script properties.

description [text] BASIC Script description.

enabled [boolean] Whether this BASIC Script is enabled.

events [array] Declared BASIC-script event bindings.

folder [text] Editor organization folder.

global [boolean] Whether this BASIC script listens globally.

id [text · read only] Stable BASIC Script ID.

languageVersion [text · read only] PCA3D BASIC language version.

Designer property paths / continued

name [text] BASIC Script display/editor name.

source [text] PCA3D BASIC 1.0 source text. Edit source in the BASIC Designer; changes invalidate the BASIC runtime cache.

CAMERAS

custom [dictionary] Project-defined camera properties.

depthOfField [dictionary] Camera depth-of-field settings.

depthOfField.amount [number] Depth-of-field blur amount.

depthOfField.enabled [boolean] Enable camera depth of field.

depthOfField.farDistance [number] Far blur distance.

depthOfField.farEnabled [boolean] Enable far blur.

depthOfField.farTransition [number] Far blur transition distance.

depthOfField.nearDistance [number] Near blur distance.

depthOfField.nearEnabled [boolean] Enable near blur.

depthOfField.nearTransition [number] Near blur transition distance.

dynamicFollow [dictionary] Dynamic-camera follow settings.

dynamicFollow.axes [text] Axes followed by the camera.

dynamicFollow.boundsMargin [number] Room-bounds margin.

dynamicFollow.clampToRoomBounds [boolean] Clamp camera to room bounds.

dynamicFollow.deadZone [vector2] Follow dead-zone size.

dynamicFollow.enabled [boolean] Enable dynamic camera follow.

dynamicFollow.lookAhead [number] Movement look-ahead amount.

dynamicFollow.maxSpeed [number] Maximum follow speed.

dynamicFollow.mode [text] Dynamic follow mode.

dynamicFollow.overrideMode [text] Inherit or override room dynamic-camera behavior.

dynamicFollow.smoothing [number] Follow smoothing.

enabled [boolean] Whether this camera record is enabled.

far [number] Far clipping distance.

folder [text] Editor organization folder.

fov [number] Perspective vertical field of view.

id [text · read only] Stable camera ID within its room.

locked [boolean] Whether editor manipulation is locked.

name [text] Camera display/editor name.

near [number] Near clipping distance.

position [vector3] Authored camera position. This is not live-applied to the active runtime camera.

Designer property paths / continued

size [number] Orthographic camera size.

target [vector3] Authored camera look-at target. This is not live-applied to the active runtime camera.

type [text] Perspective or orthographic camera type.

visible [boolean] Editor visibility.

CHARACTERS

assetId [record] Model asset used by this character. Saved immediately; changing a loaded model is not guaranteed to rebuild the current actor.

character [dictionary] Movement, dialogue, player, model, and follow settings.

character.autoFitModel [boolean] Automatically fit unskinned models to modelHeight.

character.collisionRadius [number] Character navigation/collision radius.

character.defaultPortraitId [text] Default portrait ID.

character.defaultState [text] Default character animation state ID.

character.displayName [text] Name used for dialogue, character labels, and player switching. Saved on the shared Character definition; a loaded Object instance may require recreation before its visible label changes.

character.enabled [boolean] Whether character behavior is enabled.

character.follow [dictionary] Persistent follow defaults.

character.follow.distance [number] Desired distance behind the target.

character.follow.enabled [boolean] Whether persistent following is enabled.

character.follow.speed [number] Maximum following speed.

character.follow.targetId [record] Character to follow. This is a persistent default. Use follow statements for immediate runtime behavior.

character.gender [text] Normalized character gender value.

character.genderCustom [text] Custom gender label when gender is custom.

character.isPlayer [boolean] Whether this is the selected primary player definition.

character.lockRootMotion [boolean] Prevent imported animation root motion from moving the actor.

character.modelFacingOffset [number] Model yaw correction in radians.

character.modelHeight [number] Expected character model height.

character.modelOffset [vector3] Model offset relative to the actor.

character.modelScale [number] Manual model scale.

character.modelTint [color] Character model tint.

character.personalDialogueGuiId [record] Dialogue GUI used by this playable character.

character.personalHudGuiId [record] HUD GUI used by this playable character.

character.personalInventoryGuiId [record] Inventory GUI used by this playable character.

character.personalJournalGuiId [record] Journal GUI used by this playable character.

character.perspectiveScale [boolean] Use near/far perspective scaling.

Designer property paths / continued

character.playable [boolean] Whether players may control this character.

character.portraits [array] Authored portrait definitions.

character.role [text] Character role such as npc or player.

character.runSpeed [number] Character running speed in world units per second.

character.scaleFar [number] Far perspective scale percentage.

character.scaleNear [number] Near perspective scale percentage.

character.speechColor [color] Dialogue text color for this character.

character.states [array] Authored animation-state records.

character.switchEnabled [boolean] Whether this playable character appears in switching controls.

character.switchLabel [text] Label shown by character switching UI.

character.switchOrder [integer] Ordering in character switching UI.

character.transitions [array] Authored animation transitions.

character.turnSpeed [number] Character turning speed in degrees per second.

character.useCustomWalkSpeed [boolean] Use this character's authored walk speed.

character.walkSpeed [number] Character walking speed in world units per second.

custom [dictionary] Project-defined custom properties.

description [text] Character description.

enabled [boolean] Whether this character definition is enabled.

hotspot [dictionary] Interaction label, enabled state, and authored verb actions.

hotspot.collisionOffset [vector3] Interaction collision offset.

hotspot.collisionSize [vector3] Interaction collision size.

hotspot.defaultVerb [text] Context mode runs this verb for this object or character, regardless of the selected verb.

hotspot.enabled [boolean] Whether the hotspot accepts interactions.

hotspot.label [text] Fallback interaction label after character.displayName and object.name. This is not the object's primary display name.

hotspot.verbs [dictionary] Actions keyed by authored interaction verb.

id [text · read only] Stable character ID used by scripts and saved games. Rename with the editor so references can be updated.

inventory [array · read only · live] Current live inventory records for this character. Read-only live view; use give/remove inventory actions to change it.

opacity [number] Character opacity from 0 to 1.

pointerFeedback [dictionary] Hover outline, bloom, tint, and cursor feedback.

pointerFeedback.attentionMode [text] Outline, bloom, tint, or combined attention style.

pointerFeedback.bloomColor [color] Pointer bloom effect color.

Designer property paths / continued

pointerFeedback.bloomEnergy [number] Pointer bloom energy.

pointerFeedback.bloomStrength [number] Pointer bloom strength.

pointerFeedback.displayNameMode [text] Inherit, show, or hide hover-name behavior.

pointerFeedback.highlightOnHover [boolean] Enable pointer attention feedback.

pointerFeedback.hoverCursorId [record] Cursor used while hovering.

pointerFeedback.leftClickCursorId [record] Cursor used for left click.

pointerFeedback.outlineColor [color] Pointer outline effect color.

pointerFeedback.outlineThickness [number] Pointer outline thickness.

pointerFeedback.rightClickCursorId [record] Cursor used for right click.

pointerFeedback.tintColor [color] Pointer tint effect color.

pointerFeedback.tintStrength [number] Pointer tint blend strength.

position [vector3 · read only · live] Current live character position. Read-only live view; use move/walk statements or transform.position on a room object.

prepositionInteractions [array] Match a verb and second object. If no rule matches, the first object's ordinary verb action receives both objects in its script context.

room [record · read only · live] Current live room ID for this character. Read-only live view; use change room actions to move characters.

shadow [dictionary] Character shadow settings.

shadow.cast [boolean] Whether this object casts shadows.

shadow.contactEnabled [boolean] Enable contact shadow.

shadow.contactOpacity [number] Contact shadow opacity.

shadow.contactSize [number] Contact shadow size.

shadow.doubleSided [boolean] Render both sides for shadows.

shadow.optimization [text] Shadow optimization mode.

shadow.receive [boolean] Whether this object receives shadows.

shadow.transparent [boolean] Treat material as transparent for shadow setup.

synonyms [array] Alternative parser names for this character.

tags [array] Project-defined character tags.

transform [dictionary] Authored position, rotation, and scale when this record is a room instance.

transform.position [vector3 · live] Local position.

transform.rotation [vector3 · live] Euler rotation in editor-authored degrees.

transform.scale [vector3 · live] Local scale.

trigger [dictionary] Character enter/exit/click event volume.

trigger.offset [vector3] Trigger volume offset.

trigger.onClick [array] Actions run on left click.

Designer property paths / continued

trigger.onDoubleClick [array] Actions run on double click.

trigger.onDoubleRightClick [array] Actions run on double right click.

trigger.onEnter [array] Actions run when a character enters.

trigger.onExit [array] Actions run when a character exits.

trigger.onHover [array] Actions run when hover begins.

trigger.onHoverExit [array] Actions run when hover ends.

trigger.onRightClick [array] Actions run on right click.

trigger.shape [text] Trigger volume shape such as box, sphere, or capsule.

trigger.size [vector3] Trigger volume dimensions.

verbPrepositions [dictionary] Objects choose whether to use this preposition in their Verb Actions tab.

visible [boolean] Whether the character is visible.

CURSORS

color [color] Cursor tint color.

custom [dictionary] Project-defined cursor properties.

enabled [boolean] Whether this cursor is enabled.

hotspotX [integer] Cursor hotspot X coordinate.

hotspotY [integer] Cursor hotspot Y coordinate.

id [text · read only] Stable cursor ID.

imageAssetId [record] Cursor image asset.

name [text] Cursor display/editor name.

symbol [text] Fallback text/system cursor symbol.

verb [text] Legacy verb binding.

DIALOGUES

custom [dictionary] Project-defined dialogue properties.

id [text · read only] Stable dialogue ID.

name [text] Dialogue display/editor name.

nodes [dictionary · read only] Dialogue graph nodes keyed by ID. Edit dialogue graph structure in the Dialogue Editor.

start [text] Dialogue start-node ID.

ENTITIES

assetId [record] Model, sprite, or other source asset. Saved to the record; a loaded object may need to be recreated for asset changes.

attachmentOffset [vector3] Offset relative to the target point.

attachmentPoint [text] Optional named attach point on the target. Empty follows its origin. Offset is relative to the point; animation and bone movement are followed.

Designer property paths / continued

attachmentTargetId [text] Character or object followed by this particle effect.

boltJaggedness [number] Sideways displacement in scene units. Zero makes a straight beam. Start and end remain fixed.

boltLength [number] Distance from the emitter to the bolt tip in scene units, along Direction. Transform scale also affects the result.

boltSegments [number] Number of lightning segments, from 2 to 128. Fewer segments reduce rendering work.

boltWidth [number] Lightning thickness in scene units. The bright core is one third of this width.

color [text] Lightning uses Primary Color for the outer bolt, Emission Color for its bright core, and Opacity for both. Glow depends on the renderer and environment. Motion, amount and lifetime fields belong to other particle presets.

custom [dictionary] Project-defined custom properties.

description [text] Object description.

direction [vector3] Bolt direction relative to the emitter.

distortionAmount [number] Heat refraction distance in screen UV units. Start at 0.02; zero removes distortion.

distortionSpeed [number] Rising heat speed. Start at 1; zero freezes movement.

editorOnly [boolean] Editor visualization metadata. The runtime does not currently use this field to suppress object spawning. Use enabled/visible or scene authoring controls for runtime behavior.

emissionColor [text] Lightning uses Primary Color for the outer bolt, Emission Color for its bright core, and Opacity for both. Glow depends on the renderer and environment. Motion, amount and lifetime fields belong to other particle presets.

enabled [boolean · live] Whether this object can update and interact.

flashDuration [number] Seconds each flash stays visible. One Shot plays one flash; disable and enable Emit In Game to replay it.

flashInterval [number] Dark time in seconds between flashes. Zero produces a continuous bolt. Each new flash gets a new jagged shape.

folder [text] Editor organization folder.

generator [dictionary] Procedural generator settings.

glow [dictionary] Object glow settings.

glow.color [color] Glow color.

glow.enabled [boolean] Enable object glow.

glow.energy [number] Glow energy multiplier.

glow.startOn [boolean] Initial variable-controlled glow state.

glow.textureAssetId [record] Optional glow texture asset.

glow.useAlbedo [boolean] Use material albedo as glow color.

glow.variable [text] Variable used to control glow.

hotspot [dictionary] Interaction label, enabled state, and verb actions.

hotspot.collisionOffset [vector3] Interaction collision offset.

Designer property paths / continued

hotspot.collisionSize [vector3] Interaction collision size.

hotspot.defaultVerb [text] Context mode runs this verb for this object or character, regardless of the selected verb.

hotspot.enabled [boolean] Whether the hotspot accepts interactions.

hotspot.label [text] Fallback interaction label after character.displayName and object.name. This is not the object's primary display name.

hotspot.verbs [dictionary] Actions keyed by authored interaction verb.

id [text · read only] Stable room-entity ID. Rename with the editor so references can be updated.

imageAssetId [record · live] Optional image asset for particle sprites. Import a transparent PNG in Asset Designer, then select it here. Empty uses the preset artwork. Images keep their aspect ratio and are tinted by particle colors; use white colors to preserve the image. Heat Haze and lightning arcs keep their specialized materials; Lightning Ball uses the image for its halo.

intensity [number] Strength of a single-preset effect. Zero removes heat distortion; layer intensity overrides this value.

layers [array] Ordered effects. Heat Haze is heat-haze; each layer has preset and intensity. Local heat surfaces use heat alone.

light [dictionary] Light-specific settings.

locked [boolean] Whether editor manipulation is locked.

material [dictionary] Material color, roughness, metallic, and reflection settings.

material.color [color] Base material color.

material.metallic [number] Material metallic amount.

material.reflectionColor [color] Reflection tint color.

material.reflectionStrength [number] Reflection strength.

material.roughness [number] Material roughness.

motion [dictionary] Automatic or imported animation motion settings.

motion.animationName [text] Imported animation clip name.

motion.axis [vector3] Motion direction or rotation axis.

motion.enabled [boolean] Enable automatic object motion.

motion.space [text] Local or global motion space.

motion.speed [number] Motion speed.

motion.startOn [boolean] Initial variable-controlled motion state.

motion.useImportedAnimation [boolean] Drive motion with an imported clip.

motion.variable [text] Variable used to control motion.

name [text] Primary object display name. Character objects use character.displayName first. Hover precedence is character.displayName, then name, then hotspot.label, then the stable ID. Objects do not have generic displayName or displayColor properties.

oneShot [boolean] Play one flash instead of repeating.

Designer property paths / continued

opacity [number] Object opacity from 0 to 1.

particle [dictionary] Particle effect settings.

pointerFeedback [dictionary] Hover outline, bloom, tint, and cursor feedback. Hover text color is global at settings.objectDisplayName.textColor, not an object property.

pointerFeedback.attentionMode [text] Outline, bloom, tint, or combined attention style.

pointerFeedback.bloomColor [color] Pointer bloom effect color.

pointerFeedback.bloomEnergy [number] Pointer bloom energy.

pointerFeedback.bloomStrength [number] Pointer bloom strength.

pointerFeedback.displayNameMode [text] Inherit, show, or hide hover-name behavior.

pointerFeedback.highlightOnHover [boolean] Enable pointer attention feedback.

pointerFeedback.hoverCursorId [record] Cursor used while hovering.

pointerFeedback.leftClickCursorId [record] Cursor used for left click.

pointerFeedback.outlineColor [color] Pointer outline effect color.

pointerFeedback.outlineThickness [number] Pointer outline thickness.

pointerFeedback.rightClickCursorId [record] Cursor used for right click.

pointerFeedback.tintColor [color] Pointer tint effect color.

pointerFeedback.tintStrength [number] Pointer tint blend strength.

prepositionInteractions [array] Match a verb and second object. If no rule matches, the first object's ordinary verb action receives both objects in its script context.

preset [text] Use heat-haze for hot-air refraction. Explicit layers take priority over this legacy single preset.

primitive [dictionary] Primitive geometry settings.

rgbString [dictionary] RGB light-string settings.

ribbon [boolean] Flat camera-facing lightning strip instead of a volumetric bolt. Both follow attachment points. This is a beam, not a trail left behind a moving object.

shader [dictionary] Visual effect stack, including heat-haze.

shadow [dictionary] Shadow and contact-shadow settings.

shadow.cast [boolean] Whether this object casts shadows.

shadow.contactEnabled [boolean] Enable contact shadow.

shadow.contactOpacity [number] Contact shadow opacity.

shadow.contactSize [number] Contact shadow size.

shadow.doubleSided [boolean] Render both sides for shadows.

shadow.optimization [text] Shadow optimization mode.

shadow.receive [boolean] Whether this object receives shadows.

shadow.transparent [boolean] Treat material as transparent for shadow setup.

sprite [dictionary] Sprite dimensions and generation settings.

Designer property paths / continued

synonyms [array] Alternative parser names for this object.

tags [array] Project-defined object tags.

transform [dictionary] Position, rotation, and scale.

transform.position [vector3 · live] Local position.

transform.rotation [vector3 · live] Euler rotation in editor-authored degrees.

transform.scale [vector3 · live] Local scale.

trigger [dictionary] Shape, size, and event actions.

trigger.offset [vector3] Trigger volume offset.

trigger.onClick [array] Actions run on left click.

trigger.onDoubleClick [array] Actions run on double click.

trigger.onDoubleRightClick [array] Actions run on double right click.

trigger.onEnter [array] Actions run when a character enters.

trigger.onExit [array] Actions run when a character exits.

trigger.onHover [array] Actions run when hover begins.

trigger.onHoverExit [array] Actions run when hover ends.

trigger.onRightClick [array] Actions run on right click.

trigger.shape [text] Trigger volume shape such as box, sphere, or capsule.

trigger.size [vector3] Trigger volume dimensions.

type [text · read only] Entity kind used by PCA3D.

verbPrepositions [dictionary] Objects choose whether to use this preposition in their Verb Actions tab.

visible [boolean · live] Whether this object is visible.

FONTS

assetId [record] Font asset resource.

custom [dictionary] Project-defined font properties.

enabled [boolean] Whether this font is enabled.

fallback [text] Comma-separated fallback font-family names.

family [text] Font family label.

id [text · read only] Stable font ID.

name [text] Font display/editor name.

style [text] Font style.

weight [integer] Font weight.

Designer property paths / continued

GUIS

activationMode [text] Manual, hover, click, or another activation mode.

clickActivationAtMousePosition [boolean] Place click-activated GUI at the mouse.

clickActivationStopsPlayer [boolean] Stop player movement when click-activated.

closeTransition [dictionary] Transition used when this GUI closes.

closeTransition.avoidImmediateRepeat [boolean] Avoid choosing the same random transition twice.

closeTransition.color [color] Transition color.

closeTransition.direction [text] Transition direction.

closeTransition.duration [number] Transition duration in seconds.

closeTransition.intensity [number] Transition intensity.

closeTransition.origin [vector2] Normalized transition origin.

closeTransition.randomPool [array] Transition types available to random mode.

closeTransition.type [text] Cut, fade, wipe, dissolve, or another authored transition.

custom [dictionary] Project-defined GUI properties.

elements [array · read only] GUI element records. Edit individual elements in the GUI editor.

hideWhenMouseLeaves [boolean] Hide hover GUI when the pointer leaves.

hoverDelaySeconds [number] Delay before hover activation in seconds.

id [text · read only] Stable GUI ID.

modal [boolean] Whether this GUI blocks gameplay input.

name [text] GUI display/editor name.

openTransition [dictionary] Transition used when this GUI opens.

openTransition.avoidImmediateRepeat [boolean] Avoid choosing the same random transition twice.

openTransition.color [color] Transition color.

openTransition.direction [text] Transition direction.

openTransition.duration [number] Transition duration in seconds.

openTransition.intensity [number] Transition intensity.

openTransition.origin [vector2] Normalized transition origin.

openTransition.randomPool [array] Transition types available to random mode.

openTransition.type [text] Cut, fade, wipe, dissolve, or another authored transition.

visible [boolean] Authored initial GUI visibility. Use show/hide GUI statements for immediate runtime visibility.

windowFrame [dictionary] Window, title bar, and close-button appearance.

windowFrame.closeButtonBackground [color] Close-button background color.

windowFrame.closeButtonEnabled [boolean] Show a close button.

windowFrame.closeButtonText [text] Close-button text.

Designer property paths / continued

windowFrame.closeButtonTextColor [color] Close-button text color.

windowFrame.enabled [boolean] Enable authored window framing.

windowFrame.height [number] Window height.

windowFrame.opacity [number] Window opacity.

windowFrame.rootWindow [boolean] Treat this GUI as a root window.

windowFrame.titleBarBackground [color] Title-bar background color.

windowFrame.titleBarDraggable [boolean] Allow dragging by the title bar.

windowFrame.titleBarEnabled [boolean] Enable the title bar.

windowFrame.titleBarHeight [number] Title-bar height.

windowFrame.titleFontId [record] Window title font.

windowFrame.titleFontSize [integer] Window title font size.

windowFrame.titleText [text] Window title text.

windowFrame.titleTextColor [color] Window title text color.

windowFrame.width [number] Window width.

windowFrame.windowBackground [color] Window background color.

windowFrame.windowBackgroundEnabled [boolean] Enable window background.

windowFrame.windowBorderColor [color] Window border color.

windowFrame.windowBorderEnabled [boolean] Enable window border.

windowFrame.windowBorderRadius [number] Window border radius.

windowFrame.windowBorderWidth [number] Window border width.

windowFrame.x [number] Window X position.

windowFrame.y [number] Window Y position.

zIndex [integer] GUI draw ordering.

INVENTORY

acquireSceneRemoval [dictionary] Optional scene object removed when acquired.

acquireSceneRemoval.enabled [boolean] Remove a scene entity when this item is acquired.

acquireSceneRemoval.entityId [record] Object removed from the room.

acquireSceneRemoval.sceneId [record] Room containing the entity to remove.

canBeCarriedBy [array] Character IDs allowed to carry this item.

category [text] Inventory category.

consumeOnUse [boolean] Remove the item after a successful use.

custom [dictionary] Project-defined item properties.

description [text] Inventory item description.

enabled [boolean] Whether this inventory item is enabled.

Designer property paths / continued

icon [text] Legacy/direct icon path.

iconAnimated [boolean] Use animated inventory icon frames.

iconAssetId [record] Inventory icon asset.

iconFps [number] Animated icon frames per second.

iconLoop [boolean] Loop animated icon frames.

id [text · read only] Stable inventory item ID. Rename with the editor so references can be updated.

interactions [dictionary] Inventory interaction actions keyed by verb.

inventoryInteractions [array] Rules for using this item on inventory/world targets.

maxStack [integer] Maximum amount in one stack.

name [text] Inventory item display name.

prepositionInteractions [array] Match a verb and second object. If no rule matches, the first object's ordinary verb action receives both objects in its script context.

rarity [text] Game-defined rarity label.

sortOrder [integer] Inventory sorting order.

stackable [boolean] Whether item amounts stack.

tags [array] Item tags.

types [array] Item type/category tags.

value [number] Game-defined item value.

verbPrepositions [dictionary] Objects choose whether to use this preposition in their Verb Actions tab.

weight [number] Game-defined item weight.

LUA SCRIPTS

custom [dictionary] Project-defined script properties.

description [text] Lua Script description.

enabled [boolean] Whether this Lua Script is enabled.

events [array] Declared Lua-script event bindings.

folder [text] Editor organization folder.

global [boolean] Whether one persistent instance listens globally.

id [text · read only] Stable Lua Script ID.

languageVersion [text · read only] Lua language version.

name [text] Lua Script display/editor name.

source [text] Lua 5.4 source text. Edit source in the Lua Designer; changes invalidate the Lua runtime cache.

Designer property paths / continued

NATIVE SCRIPTS

custom [dictionary] Project-defined script properties.

description [text] Native script description.

enabled [boolean] Whether this native script is enabled.

events [array] Declared native-script event bindings.

folder [text] Editor organization folder.

global [boolean] Whether this script listens globally.

id [text · read only] Stable native script ID.

name [text] Native script display/editor name.

source [text] Native GDScript source text. Native code is unrestricted and should be edited in the Script Designer.

OBJECTIVES

custom [dictionary] Project-defined objective properties.

hidden [boolean] Hide this objective from normal journal views.

id [text · read only] Stable objective ID within its quest stage.

optional [boolean] Whether this objective is optional.

text [text] Text displayed for the objective.

textKey [text] Translation key for objective text.

PCA SCRIPTS

breakpoints [array] Authored breakpoint line numbers.

custom [dictionary] Project-defined script properties.

description [text] Script description.

enabled [boolean] Whether this script is enabled.

events [array] Stored event metadata.

folder [text] Editor organization folder.

id [text · read only] Stable PCA Script ID.

languageVersion [text · read only] PCA Script language version.

name [text] PCA Script display/editor name.

source [text · live] PCA Script source text. Changing source invalidates the compiled script cache.

QUESTS

autoTrack [boolean] Automatically track this quest.

category [text] Quest category.

completionActions [array] Actions run when the quest completes.

custom [dictionary] Project-defined quest properties.

Designer property paths / continued

description [text] Quest description.

descriptionKey [text] Translation key for the description.

failureActions [array] Actions run when the quest fails.

folder [text] Editor organization folder.

hidden [boolean] Hide this quest from normal journal views.

id [text · read only] Stable quest ID.

repeatable [boolean] Allow this quest to repeat.

stages [array · read only] Quest stage definitions. Edit nested objectives through the objective namespace or Quest Editor.

title [text] Quest title.

titleKey [text] Translation key for the title.

SCENES

cameraBounds [dictionary] Room camera movement limits.

cameraBounds.enabled [boolean] Enable camera movement bounds.

cameraBounds.maxX [number] Maximum camera X position.

cameraBounds.maxZ [number] Maximum camera Z position.

cameraBounds.minX [number] Minimum camera X position.

cameraBounds.minZ [number] Minimum camera Z position.

cameras [array · read only] Cameras nested in this room. Edit individual cameras through the cameras namespace.

custom [dictionary] Project-defined room properties.

defaultCameraId [record] Camera selected when the room loads.

description [text] Room description.

distortionAmount [number] Heat refraction distance in screen UV units. Start at 0.02; zero removes distortion.

distortionSpeed [number] Rising heat speed. Start at 1; zero freezes movement.

dynamicCamera [dictionary] Room dynamic-camera settings.

entities [array · read only] Objects placed in the room. Edit individual objects through the objects namespace.

environment [dictionary] Brightness, fog, glow, skybox, outline, and post-processing.

environment.brightness [number] Room environment brightness.

environment.fogColor [color] Fog color.

environment.fogDensity [number] Fog density.

environment.fogDepthBegin [number] Depth fog start distance.

environment.fogDepthEnd [number] Depth fog end distance.

environment.fogEnabled [boolean] Enable room fog.

environment.fogHeight [number] Height fog reference height.

Designer property paths / continued

environment.fogHeightDensity [number] Height fog density.

environment.fogMode [text] Depth, height, or volumetric fog mode.

environment.glowEnabled [boolean] Enable environment glow.

environment.glowIntensity [number] Environment glow intensity.

environment.glowStrength [number] Environment glow strength.

environment.postProcess [dictionary] Room post-processing effect settings.

environment.postProcessOverrideMode [text] Inherit, custom, or disabled post-processing.

environment.reflectionProbePosition [vector3] Room probe center in world coordinates.

environment.sceneOutline [dictionary] Full-scene outline settings.

environment.sceneOutline.edgesOnly [boolean] Outlines visible depth edges and keeps the original surface shading.

environment.shadowQuality [text] Room shadow-quality preset.

environment.skybox [dictionary] Room skybox settings.

environment.volumetricFogDensity [number] Volumetric fog density from 0 to 1.

environment.volumetricFogEnabled [boolean] Enable volumetric fog in Forward+.

environment.volumetricFogLength [number] Volumetric fog drawing distance.

folder [text] Editor organization folder.

hideAllGuisOnLoad [boolean] Hide authored GUIs when entering this room.

hidePlayerCharacter [boolean] Hide the player while this room is active.

id [text · read only] Stable room ID. Rename with the editor so references can be updated.

intensity [number] Strength of a single-preset effect. Zero removes heat distortion; layer intensity overrides this value.

layers [array] Ordered effects. Heat Haze is heat-haze; each layer has preset and intensity. Local heat surfaces use heat alone.

name [text] Display name for this room.

navigation [dictionary] Walkable areas and navigation-agent settings.

navigation.agentRadius [number] Default navigation-agent radius.

navigation.areas [array] Walkable navigation-area records.

navigation.blocked [array] Navigation blocker records.

navigation.bounds [dictionary] Primary navigation-area bounds.

navigation.cellHeight [number] Navigation map cell height.

navigation.cellSize [number] Navigation map cell size.

navigation.polygon [array] Primary navigation polygon points.

navigation.type [text] Plane or polygon navigation area type.

navigation.walkableY [number] Default walkable Y plane.

Designer property paths / continued

objectDisplayNameMode [text] Room override for object hover names.

preset [text] Use heat-haze for hot-air refraction. Explicit layers take priority over this legacy single preset.

regions [array] Navigation and gameplay regions.

screenOverlay [dictionary] Room screen-overlay settings.

screenOverlay.assetId [record] Image/video asset used by the overlay.

screenOverlay.enabled [boolean] Enable the screen overlay.

screenOverlay.overrideMode [text] Room/global overlay override mode.

screenOverlay.parallaxEnabled [boolean] Enable cursor/camera parallax.

screenOverlay.parallaxSmoothing [number] Overlay parallax smoothing.

screenOverlay.parallaxStrength [number] Overlay parallax strength.

screenOverlay.shaderMode [text] Overlay shader mode.

screenOverlay.transitionAffectsOverlay [boolean] Apply room transitions to the overlay.

spawns [array · read only] Spawn points nested in this room. Edit individual spawns through the spawns namespace.

transition [dictionary] Transition used when entering this room.

transition.avoidImmediateRepeat [boolean] Avoid choosing the same random transition twice.

transition.color [color] Transition color.

transition.direction [text] Transition direction.

transition.duration [number] Transition duration in seconds.

transition.intensity [number] Transition intensity.

transition.origin [vector2] Normalized transition origin.

transition.randomPool [array] Transition types available to random mode.

transition.type [text] Cut, fade, wipe, dissolve, or another authored transition.

viewport [dictionary] Room rendering and GUI design size overrides.

viewport.guiSizeMode [text] GUI design-size behavior.

viewport.height [integer] Room viewport height.

viewport.overlaySizeMode [text] Screen-overlay size behavior.

viewport.overrideMode [text] Inherit or override viewport sizing.

viewport.width [integer] Room viewport width.

SPAWNS

custom [dictionary] Project-defined spawn properties.

enabled [boolean] Whether this spawn is enabled.

facing [number] Spawn facing/yaw.

folder [text] Editor organization folder.

Designer property paths / continued

id [text · read only] Stable spawn ID within its room.

locked [boolean] Whether editor manipulation is locked.

name [text] Spawn display/editor name.

position [vector3] Spawn position.

visible [boolean] Editor visibility.

TRIGGERS

assetId [record] Model, sprite, or other source asset. Saved to the record; a loaded object may need to be recreated for asset changes.

attachmentOffset [vector3] Offset relative to the target point.

attachmentPoint [text] Optional named attach point on the target. Empty follows its origin. Offset is relative to the point; animation and bone movement are followed.

attachmentTargetId [text] Character or object followed by this particle effect.

boltJaggedness [number] Sideways displacement in scene units. Zero makes a straight beam. Start and end remain fixed.

boltLength [number] Distance from the emitter to the bolt tip in scene units, along Direction. Transform scale also affects the result.

boltSegments [number] Number of lightning segments, from 2 to 128. Fewer segments reduce rendering work.

boltWidth [number] Lightning thickness in scene units. The bright core is one third of this width.

color [text] Lightning uses Primary Color for the outer bolt, Emission Color for its bright core, and Opacity for both. Glow depends on the renderer and environment. Motion, amount and lifetime fields belong to other particle presets.

custom [dictionary] Project-defined custom properties.

description [text] Object description.

direction [vector3] Bolt direction relative to the emitter.

distortionAmount [number] Heat refraction distance in screen UV units. Start at 0.02; zero removes distortion.

distortionSpeed [number] Rising heat speed. Start at 1; zero freezes movement.

editorOnly [boolean] Editor visualization metadata. The runtime does not currently use this field to suppress object spawning. Use enabled/visible or scene authoring controls for runtime behavior.

emissionColor [text] Lightning uses Primary Color for the outer bolt, Emission Color for its bright core, and Opacity for both. Glow depends on the renderer and environment. Motion, amount and lifetime fields belong to other particle presets.

enabled [boolean · live] Whether this object can update and interact.

flashDuration [number] Seconds each flash stays visible. One Shot plays one flash; disable and enable Emit In Game to replay it.

flashInterval [number] Dark time in seconds between flashes. Zero produces a continuous bolt. Each new flash gets a new jagged shape.

folder [text] Editor organization folder.

Designer property paths / continued

generator [dictionary] Procedural generator settings.

glow [dictionary] Object glow settings.

glow.color [color] Glow color.

glow.enabled [boolean] Enable object glow.

glow.energy [number] Glow energy multiplier.

glow.startOn [boolean] Initial variable-controlled glow state.

glow.textureAssetId [record] Optional glow texture asset.

glow.useAlbedo [boolean] Use material albedo as glow color.

glow.variable [text] Variable used to control glow.

hotspot [dictionary] Interaction label, enabled state, and verb actions.

hotspot.collisionOffset [vector3] Interaction collision offset.

hotspot.collisionSize [vector3] Interaction collision size.

hotspot.defaultVerb [text] Context mode runs this verb for this object or character, regardless of the selected verb.

hotspot.enabled [boolean] Whether the hotspot accepts interactions.

hotspot.label [text] Fallback interaction label after character.displayName and object.name. This is not the object's primary display name.

hotspot.verbs [dictionary] Actions keyed by authored interaction verb.

id [text · read only] Stable room-entity ID. Rename with the editor so references can be updated.

imageAssetId [record · live] Optional image asset for particle sprites. Import a transparent PNG in Asset Designer, then select it here. Empty uses the preset artwork. Images keep their aspect ratio and are tinted by particle colors; use white colors to preserve the image. Heat Haze and lightning arcs keep their specialized materials; Lightning Ball uses the image for its halo.

intensity [number] Strength of a single-preset effect. Zero removes heat distortion; layer intensity overrides this value.

layers [array] Ordered effects. Heat Haze is heat-haze; each layer has preset and intensity. Local heat surfaces use heat alone.

light [dictionary] Light-specific settings.

locked [boolean] Whether editor manipulation is locked.

material [dictionary] Material color, roughness, metallic, and reflection settings.

material.color [color] Base material color.

material.metallic [number] Material metallic amount.

material.reflectionColor [color] Reflection tint color.

material.reflectionStrength [number] Reflection strength.

material.roughness [number] Material roughness.

motion [dictionary] Automatic or imported animation motion settings.

Designer property paths / continued

motion.animationName [text] Imported animation clip name.

motion.axis [vector3] Motion direction or rotation axis.

motion.enabled [boolean] Enable automatic object motion.

motion.space [text] Local or global motion space.

motion.speed [number] Motion speed.

motion.startOn [boolean] Initial variable-controlled motion state.

motion.useImportedAnimation [boolean] Drive motion with an imported clip.

motion.variable [text] Variable used to control motion.

name [text] Primary object display name. Character objects use character.displayName first. Hover precedence is character.displayName, then name, then hotspot.label, then the stable ID. Objects do not have generic displayName or displayColor properties.

oneShot [boolean] Play one flash instead of repeating.

opacity [number] Object opacity from 0 to 1.

particle [dictionary] Particle effect settings.

pointerFeedback [dictionary] Hover outline, bloom, tint, and cursor feedback. Hover text color is global at settings.objectDisplayName.textColor, not an object property.

pointerFeedback.attentionMode [text] Outline, bloom, tint, or combined attention style.

pointerFeedback.bloomColor [color] Pointer bloom effect color.

pointerFeedback.bloomEnergy [number] Pointer bloom energy.

pointerFeedback.bloomStrength [number] Pointer bloom strength.

pointerFeedback.displayNameMode [text] Inherit, show, or hide hover-name behavior.

pointerFeedback.highlightOnHover [boolean] Enable pointer attention feedback.

pointerFeedback.hoverCursorId [record] Cursor used while hovering.

pointerFeedback.leftClickCursorId [record] Cursor used for left click.

pointerFeedback.outlineColor [color] Pointer outline effect color.

pointerFeedback.outlineThickness [number] Pointer outline thickness.

pointerFeedback.rightClickCursorId [record] Cursor used for right click.

pointerFeedback.tintColor [color] Pointer tint effect color.

pointerFeedback.tintStrength [number] Pointer tint blend strength.

prepositionInteractions [array] Match a verb and second object. If no rule matches, the first object's ordinary verb action receives both objects in its script context.

preset [text] Use heat-haze for hot-air refraction. Explicit layers take priority over this legacy single preset.

primitive [dictionary] Primitive geometry settings.

rgbString [dictionary] RGB light-string settings.

Designer property paths / continued

ribbon [boolean] Flat camera-facing lightning strip instead of a volumetric bolt. Both follow attachment points. This is a beam, not a trail left behind a moving object.

shader [dictionary] Visual effect stack, including heat-haze.

shadow [dictionary] Shadow and contact-shadow settings.

shadow.cast [boolean] Whether this object casts shadows.

shadow.contactEnabled [boolean] Enable contact shadow.

shadow.contactOpacity [number] Contact shadow opacity.

shadow.contactSize [number] Contact shadow size.

shadow.doubleSided [boolean] Render both sides for shadows.

shadow.optimization [text] Shadow optimization mode.

shadow.receive [boolean] Whether this object receives shadows.

shadow.transparent [boolean] Treat material as transparent for shadow setup.

sprite [dictionary] Sprite dimensions and generation settings.

synonyms [array] Alternative parser names for this object.

tags [array] Project-defined object tags.

transform [dictionary] Position, rotation, and scale.

transform.position [vector3 · live] Local position.

transform.rotation [vector3 · live] Euler rotation in editor-authored degrees.

transform.scale [vector3 · live] Local scale.

trigger [dictionary] Shape, size, and event actions.

trigger.offset [vector3] Trigger volume offset.

trigger.onClick [array] Actions run on left click.

trigger.onDoubleClick [array] Actions run on double click.

trigger.onDoubleRightClick [array] Actions run on double right click.

trigger.onEnter [array] Actions run when a character enters.

trigger.onExit [array] Actions run when a character exits.

trigger.onHover [array] Actions run when hover begins.

trigger.onHoverExit [array] Actions run when hover ends.

trigger.onRightClick [array] Actions run on right click.

trigger.shape [text] Trigger volume shape such as box, sphere, or capsule.

trigger.size [vector3] Trigger volume dimensions.

type [text · read only] Trigger entity kind.

verbPrepositions [dictionary] Objects choose whether to use this preposition in their Verb Actions tab.

visible [boolean · live] Whether this object is visible.

Designer property paths / continued

VERBS

allowPreposition [boolean] Objects choose whether to use this preposition in their Verb Actions tab.

cursorId [record] Cursor used by this verb.

custom [dictionary] Project-defined verb properties.

enabled [boolean] Whether this verb is enabled.

folder [text] Editor organization folder.

id [text · read only] Stable lowercase verb ID.

name [text] Verb displayed to players.

preposition [text] Preposition

synonyms [array] Parser synonyms for this verb.

VISUAL SCRIPTS

custom [dictionary] Project-defined script properties.

edges [array · read only] Visual Script graph connections. Edit graph structure in the Visual Script Editor.

enabled [boolean] Whether this Visual Script is enabled.

id [text · read only] Stable Visual Script ID.

name [text] Visual Script display/editor name.

nodes [array · read only] Visual Script node records. Edit graph structure in the Visual Script Editor.