DESKTOP TUTORIAL / PCA 3D 0.86.3.4

PCA 3D Adventure Studio

Your first adventure, from an empty room to Clockmaker



Interface guide • Two complete projects • Exact reconstruction recipes

Windows • macOS • Linux

November 2026

How to use this book

Use the contents and PDF bookmarks to jump to a lesson. Parts One and Two are the best starting point if you have not made a game in PCA 3D before. The reconstruction tables are reference material, not extra steps to memorize.

The practice archive is attached to this PDF and also supplied separately. It contains the full original records, an asset-and-character starter and the opening helper scripts. A viewer that cannot open PDF attachments can use the separate ZIP.

Choose your route

If you want to...	Start here
Learn the editor from scratch	Before you begin, then Chapters 1-15. Keep the editor beside the book and try one control at a time.
Build a first playable adventure	Chapters 16-21. Make The Last Signal with simple shapes before importing character models.
Recreate the furnished demo	Chapters 22-33. Work from the Clockmaker asset/character starter or a separate template copy.
Find an exact value	R1-R8. These tables contain the placements, events, GUI rectangles, camera settings and timeline.
Read the full opening code	R10 and the attached Scripts folder. Long source lines wrap for print; the files preserve exact line breaks.
Prepare a player download	Chapter 34, after a fresh complete playthrough.

Keep three copies

Keep the supplied reference unchanged, save your own working project and make a checkpoint before a large experiment. The built player game belongs in a fourth folder. This simple habit makes it much easier to recover from a wrong deletion or compare a setting.

Recipe coordinates are rounded to six decimal places on the page for readability. The attached source records retain the full original precision. A missing reference in the empty Clockmaker starter is expected until you build the record that the retained script calls.

Contents

How to use this book 2

Contents 3

Before you begin 6

1. Find your way around 8

2. Records, saving and testing 11

3. Project, settings and controls 12

4. Build and navigate a room 16

5. Navigation, spawns and cameras 17

6. Characters, models and animation 20

7. Assets, previews and imported files 23

8. Events, verbs and inventory rules 25

9. GUI Designer: layout and appearance 28

10. GUI behavior and preview 31

11. Visual scripts and text scripts 32

12. Dialogue, messages and sound 35

13. Cutscenes and camera behavior 37

14. Lighting, regions and screen effects 40

15. The remaining designers 42

16. Start The Last Signal 47

17. Build the lab and beacon deck 49

18. Pick up and use the power cell 50

19. Align the receiver and finish 52

20. Build the HUD, inventory and ending 54

21. Finish and test The Last Signal 56

22. Prepare the Clockmaker project 60

23. Make the three indoor rooms 62

24. Set up Ivo and Elias 63

25. The key, notebook and locked door 66

26. Find the parts and repair the clock 67

27. Build Elias's conversation 70

28. Make the winding puzzle 72

29. Rebuild the Clockmaker screens 73

30. Build the camera set 74

31. Rebuild the directed indoor opening 84

32. Rebuild the snowy exterior 86

33. Play through Clockmaker 88

34. Build a playable release 93

35. Fix common problems 95

36. A practical editing routine 96

R1. Primitive room placements 98

R2. Primitive actions and GUI rectangles 105

R3. Clockmaker indoor placements 111

R4. Clockmaker puzzle and room events 124

R5. Clockmaker GUI rectangles 130

R6. Clockmaker indoor timeline 138

R7. Exterior placement reference 146

R8. Asset manifest and scripts 170

R9. Glossary and further reading 175

R10. Directed-opening helper sources 177

Before you begin

This book takes you from an empty room to two complete little adventures. The first is The Last Signal, the Primitive Prototype included with PCA 3D. It uses plain shapes so you can learn the editor without worrying about model imports. The second is The Clockmaker - A Minute Borrowed. It adds furnished rooms, two animated characters, a conversation, an inventory puzzle, voice recordings and a directed opening.

Work through the Primitive lessons in order. Once that game works, the Clockmaker lessons will feel familiar: the art changes, but the way you build the game stays much the same. Keep the finished template open in a separate saved copy when you need to compare a setting. Give your own working copies different filenames.

What this book was checked against

These lessons cover PCA 3D Adventure Studio 0.86.3.4 on Windows, macOS and Linux. The screenshots were captured on Linux with Godot 4.7.2, using the Clean Dark editor theme so small controls are easier to read. Your colors, panel widths and file dialogs may look different; the record names and editing workflow are the same.

The captures use the Compatibility renderer. It is useful for editing on modest graphics hardware, but it does not show every Forward+ effect. The lighting chapter explains that difference. A missing glow or depth-of-field effect in a screenshot is not a promise that every renderer supports it.

The lessons use the supplied demo assets. No separate art purchase is needed to follow them. Paths beginning with res:// point inside the PCA 3D installation or Godot project. A .pca3d file stores your game records; imported files still need to be available when you reopen, move or build that game.

Set up a sensible working folder

1. Extract the package for your operating system into a folder you can write to. Do not run it from inside the archive viewer.
2. Make a separate folder for your own games. For example, create PCA Projects in Documents, with Last Signal and Clockmaker Practice beneath it.
3. If your package includes a standalone Studio application, launch it using the instructions below for your operating system.
4. If you have the Complete source package, open its project.godot in Godot 4.7.2 for your operating system. Let the initial import finish, then run the main project. The compiled runners in runner_bins run exported games; they are not the Studio editor.
5. At the PCA 3D welcome screen, choose New Game for the Primitive lessons. Use Start from Template Game when a lesson asks for a reference or the Clockmaker asset starter.

Do not save your practice project over data/prototype_project.pca3d or data/clockmaker_project.pca3d. Those are the supplied examples. Save As creates a working copy without altering them.

Platform	Starting a supplied standalone Studio application
Windows	Open the extracted folder in File Explorer and run the Studio .exe. Keep its companion files together.
macOS	Extract the archive in Finder and open the supplied Studio .app. Keep the application bundle intact.
Linux	Open the extracted folder in your file manager. If necessary, enable permission to run the Studio executable in its Properties, then launch it.

The table applies when a standalone Studio application is included. For a Complete source package, use the project.godot route on any of the three systems.

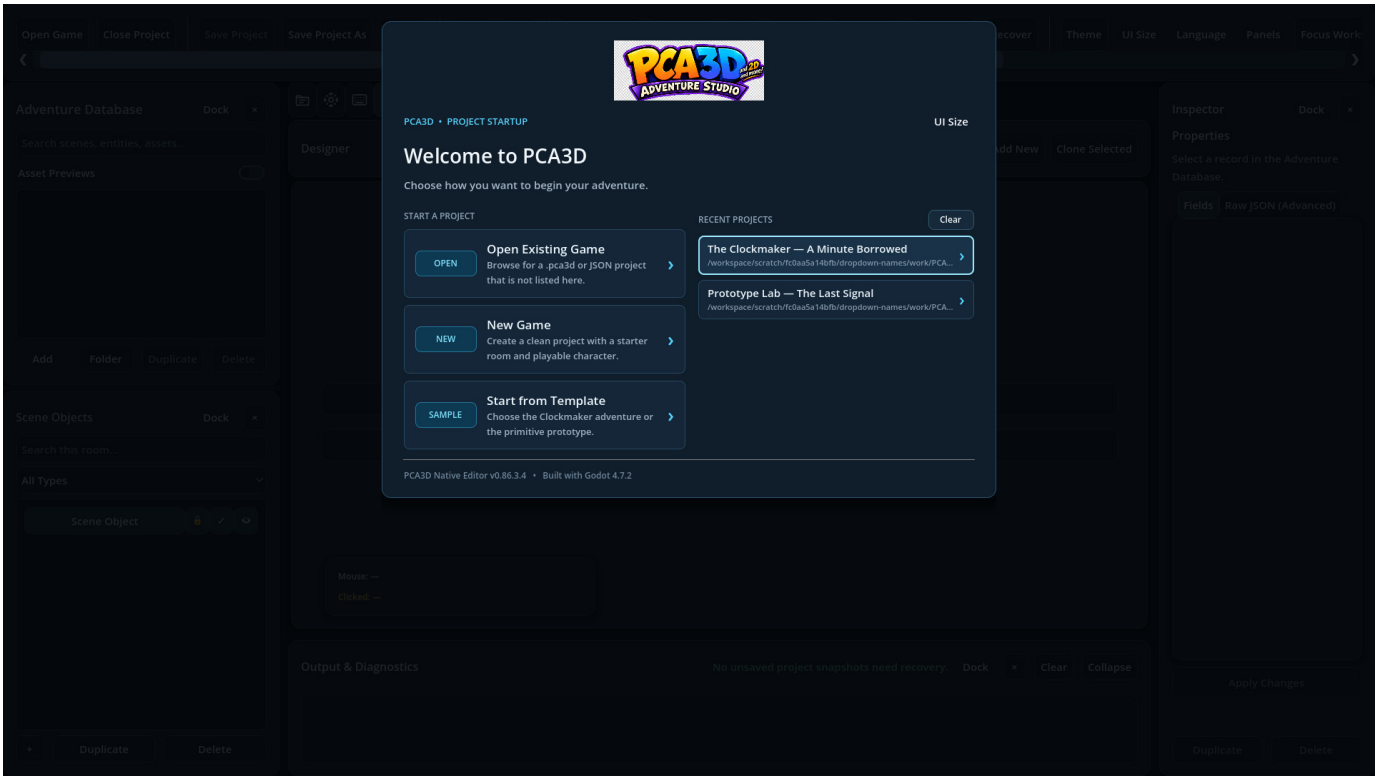

Figure 1. The welcome screen. New Game starts an empty project; Start from Template Game makes a copy of a supplied demo.

Reading the recipes

A name is the label a person reads. An ID is the reference the game stores. A camera may appear as Room view(wide): Room view is its name and wide is its ID. Set IDs early and leave them alone once other records refer to them. The parentheses in a dropdown are display text; do not type Room view(wide) into a field that asks only for an ID.

Numbers such as [2.5, 0.9, -2.4] mean X, Y and Z, in that order. X runs sideways, Y is height and Z is depth. Positions and dimensions in the room use world units. GUI rectangles use pixels at the project's design resolution. Angles in the placement tables are degrees.

Action recipes are listed from top to bottom in the order they run. Indented True and False branches belong inside the preceding If action. An action called Add Inventory in the interface is stored as AddInventory. Both forms appear in this book so you can recognize the typed editor and its raw data.

> Check your work. At the end of a lesson, stop and try its short check before adding more. A working plain room is a much better starting point than a beautiful room with three different broken puzzles.

1. Find your way around

The editor has four main areas. The Adventure Database holds records for the whole game. Scene Objects lists the contents of the current room. The middle workspace is where you build or preview the selected record. The Inspector edits its properties. Output & Diagnostics sits along the bottom and reports what the editor and game are doing.

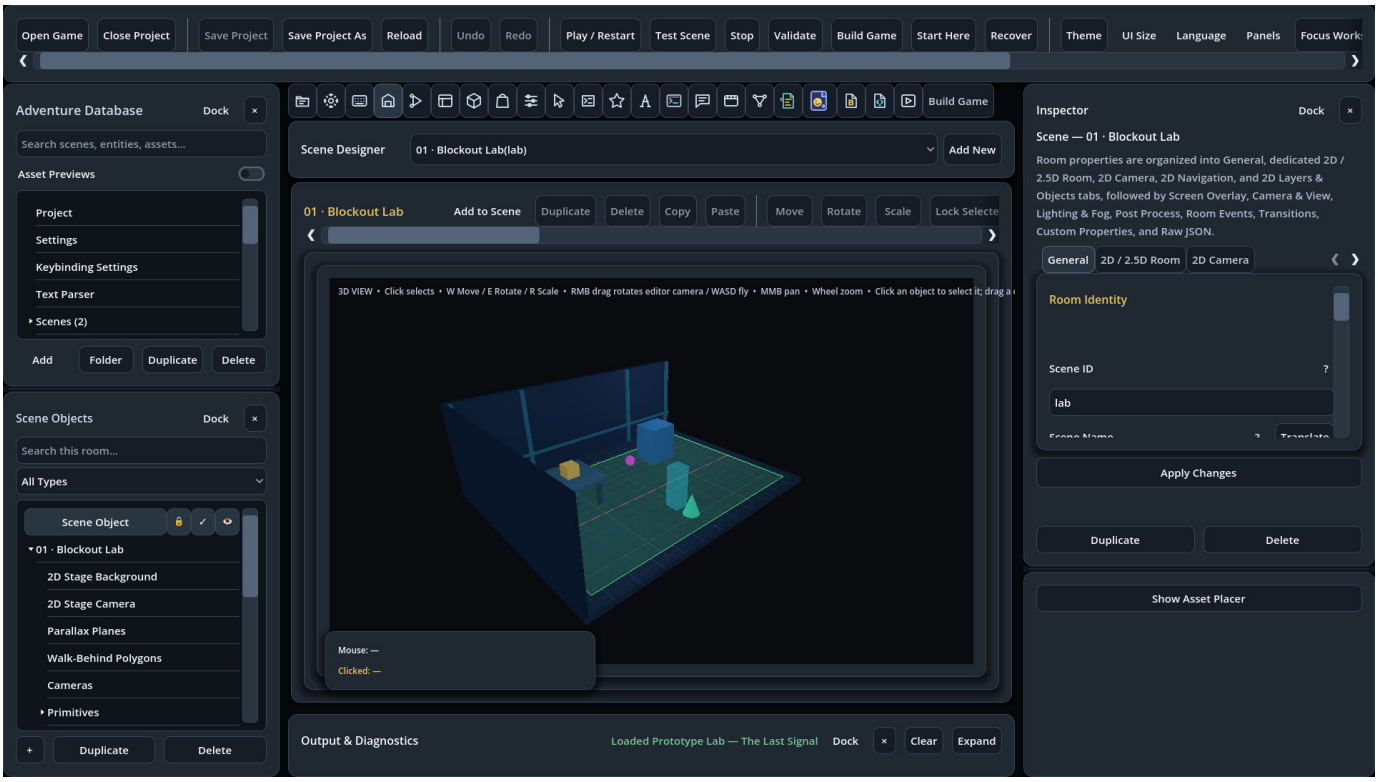

Figure 2. A supplied Primitive room in the editor. Database and room contents are on the left, the active designer is in the middle and the Inspector is on the right.

The top toolbar

Control	What you use it for
Open Game / Close Project	Open a saved .pca3d game or return to the welcome screen.
Save Project / Save Project As	Save your working file, or make a separate copy with a new filename.
Reload	Read the saved file again. Unsaved work can be lost, so read the confirmation.
Undo / Redo	Step backward or forward through editing changes.
Play / Restart	Start the game from its configured beginning with current editor data.
Test Scene	Start in the room you are editing. Useful for a quick walking or lighting check.
Stop	End the running preview. It does not delete your authored records.
Validate	Find missing references, invalid records and other project problems.
Build Game	Create a distributable game or an editable Godot runtime project.
Start Here	Return to the editor's built-in starting guidance.
Recover	Look for snapshots from an interrupted editing session.
Theme / UI Size / Language	Change the editor's appearance, scale and interface language.
Panels / Focus Workspace	Arrange panels, or temporarily make more room for the central designer.

The toolbar can be wider than your window. Its horizontal bar and end arrows reveal the rest. The same applies to designer toolbars and Inspector tabs. If a tab seems to be missing, scroll the tab strip before assuming the feature is unavailable.

The Adventure Database

Expand a category to see its records. Selecting a record opens its designer or property form. The search box finds records without making you open every folder. Asset Previews changes how assets are presented; it does not import or delete them.

Use Add for a new record, Folder to organize a category, Duplicate for a copy and Delete to remove the selection. A duplicate receives a new ID. Read deletion prompts carefully: deleting a reusable character definition affects its room placements, while deleting one placed character affects that room.

Scene Objects

This list belongs to the current room. It includes cameras, placed characters, scenery, spawns, regions, navigation information and other scene items. Search this room or choose a type filter when the scene becomes crowded. A character in this list is a placement of a reusable character from the main database.

The lock, enabled and visibility controls do different jobs. Lock protects editing selection or movement. Enabled controls whether an item participates. Visible controls its drawing. An invisible trigger may still be enabled and useful. A hidden object is not automatically a deleted object.

The workspace and Inspector

Workspace icons switch between the designers. Hover an icon for its tooltip rather than memorizing every symbol immediately. The record dropdown at the top of a designer switches to another record of the same kind. Add New and Clone Selected are shortcuts into the same project data.

The Inspector title tells you what you are editing. Check it before typing. Scene, Entity, Character and GUI Element are different kinds of records, even when their names are similar. Many Inspectors have tabs such as General, Object Events, Verb Actions and Visual Effects. Use the small tab-strip arrows to reach the tabs to the right.

Arrange the window once

1. Drag a panel boundary until the Inspector is wide enough to read its labels.
2. Use Dock on a panel title bar to move or float it. On a small screen, hide a panel from Panels when you need more canvas space.
3. Use Focus Workspace for a larger designer, then toggle it again to restore the working layout.
4. Save the workspace layout from Panels after arranging it. Reset Workspace Layout brings the normal arrangement back if a panel goes astray.

Your editor layout is a personal preference. It is not the layout of the game you are building.

2. Records, saving and testing

A small map of the project

Record	Example	What it contains
Scene	Blockout Lab(lab)	A room, its cameras, navigation, placed objects and room events.
Character	Tester(tester)	Reusable appearance, movement, animation and speech settings.
Asset	Oil Icon(oil_icon)	An imported image, model, sound, font or other source file.
Inventory item	Power cell(cell)	The carried item, its icon and its use-on rules.
Variable	powered	A value the game remembers, such as whether a relay has power.
GUI	Inventory(inventory)	A screen or overlay containing controls.
Dialogue	Elias(elias)	A conversation graph with lines, choices and actions.
Cutscene	Opening(opening)	Timed camera, character, sound and gameplay events.
Script / visual script	send_signal	Reusable instructions that an event can call.
Quest	The last signal(signal)	Journal stages and objectives.

The physical power cell in a room, the carried Power cell item and its icon asset are three records. Picking it up runs actions that connect them. This separation lets you show one pickup in the room, a different picture in Inventory and different text in the journal.

Apply is not Save

Some Inspectors keep a draft until you click Apply Changes. That writes the form into the working project in memory. Save Project then writes the project to disk. If you switch records with an unapplied draft, the editor asks whether to apply it or discard it.

GUI element fields update live. They do not have a general Apply button for every edit. Script editors, definition dialogs and track editors have their own commit buttons. Follow the button beside the work you are doing, then save the project normally.

Use Ctrl+S to save, Ctrl+Shift+S for Save As, Ctrl+Z to undo and Ctrl+Y or Ctrl+Shift+Z to redo. Save before a major experiment. A separate checkpoint copy is easier to understand than a file called final_final_really_final.pca3d. We have all met that file.

On macOS, the project commands also accept Command: Cmd+S saves, Cmd+Shift+S opens Save As, Cmd+Z undoes and Cmd+Shift+Z redoes. Mouse-selection gestures are described separately in each designer.

Test from the right place

Test Scene starts quickly, but it can skip the beginning of the story. If an earlier room normally sets a variable or gives an item, the later room may not yet be in a playable state. That is expected. Test a local change with Test Scene, then use Play / Restart for a full check from the start.

Restarting initializes the game again. Picking up an item during a preview does not add that item to the editor's starting inventory. Runtime changes and authored defaults are separate. A saved game can also restore old puzzle values; use a fresh game when testing new initial conditions.

Output and validation

Expand Output & Diagnostics when a preview does not behave as expected. Read the first relevant error, not just the last one: one missing ID can cause several later messages. Validate before a build and after renaming or deleting records.

Warnings deserve a look, but a warning is not automatically a crash. A script name-shadowing warning and a missing model asset are very different problems. Keep the actual message and record path when reporting a fault.

> Check your work. Create a temporary variable, apply it and save. Change its value, apply, undo and redo. Reopen the project once. You should be able to explain which value lives in the form, the working project and the saved file.

3. Project, settings and controls

Project sets the identity and starting conditions of the game. Settings controls shared gameplay and presentation behavior. Neither is the operating-system preferences window, and neither is a single room's Inspector.

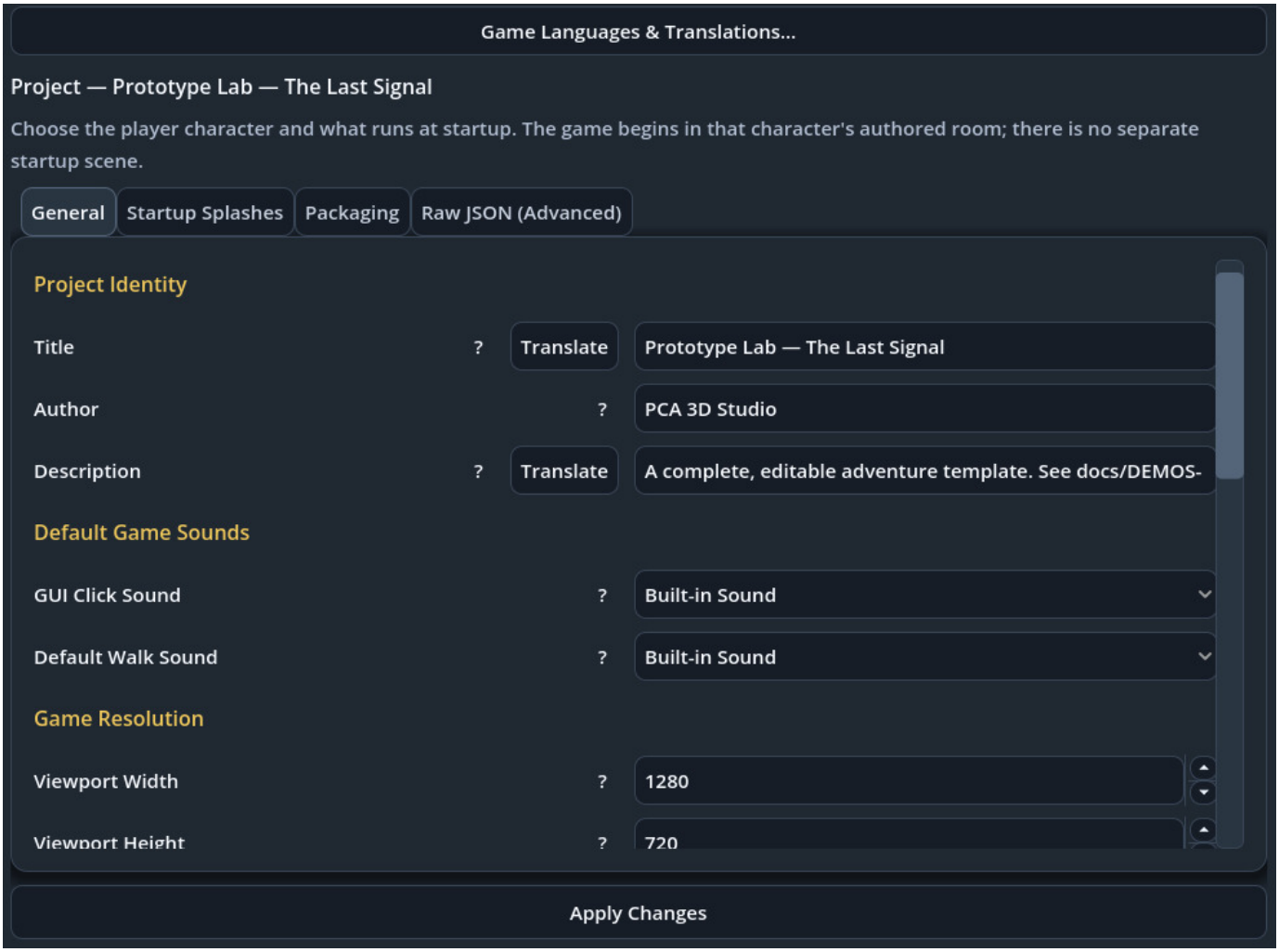

Figure 3. The Project form. Start here for the title, controlled character, startup action and design resolution.

Project

Set the game title, version, starting character, startup action and resolution here. Choose the character by its assigned name and ID. The game begins in that character's authored room; this version has no separate Startup Scene picker. A selected playable character with no room placement starts in the first room. To appear as a usable player choice, a reusable character must be configured as playable; merely importing a model does not create a player character.

Startup Action can enter normal player control, run a startup cutscene or run a startup script. The Clockmaker demo instead begins through its exterior room's First Enter event. Do not add a second startup call to the same sequence or the introduction may run twice.

The demo GUIs use a 1280 by 720 design canvas. Keep that resolution while copying their pixel measurements. Changing the design resolution halfway through the GUI lessons makes identical numbers look different.

Startup splash screens belong here too. Choose the image, hold time, fades and whether it can be skipped. An opening cutscene is a separate record; do not try to turn a splash image into a room event.

Settings

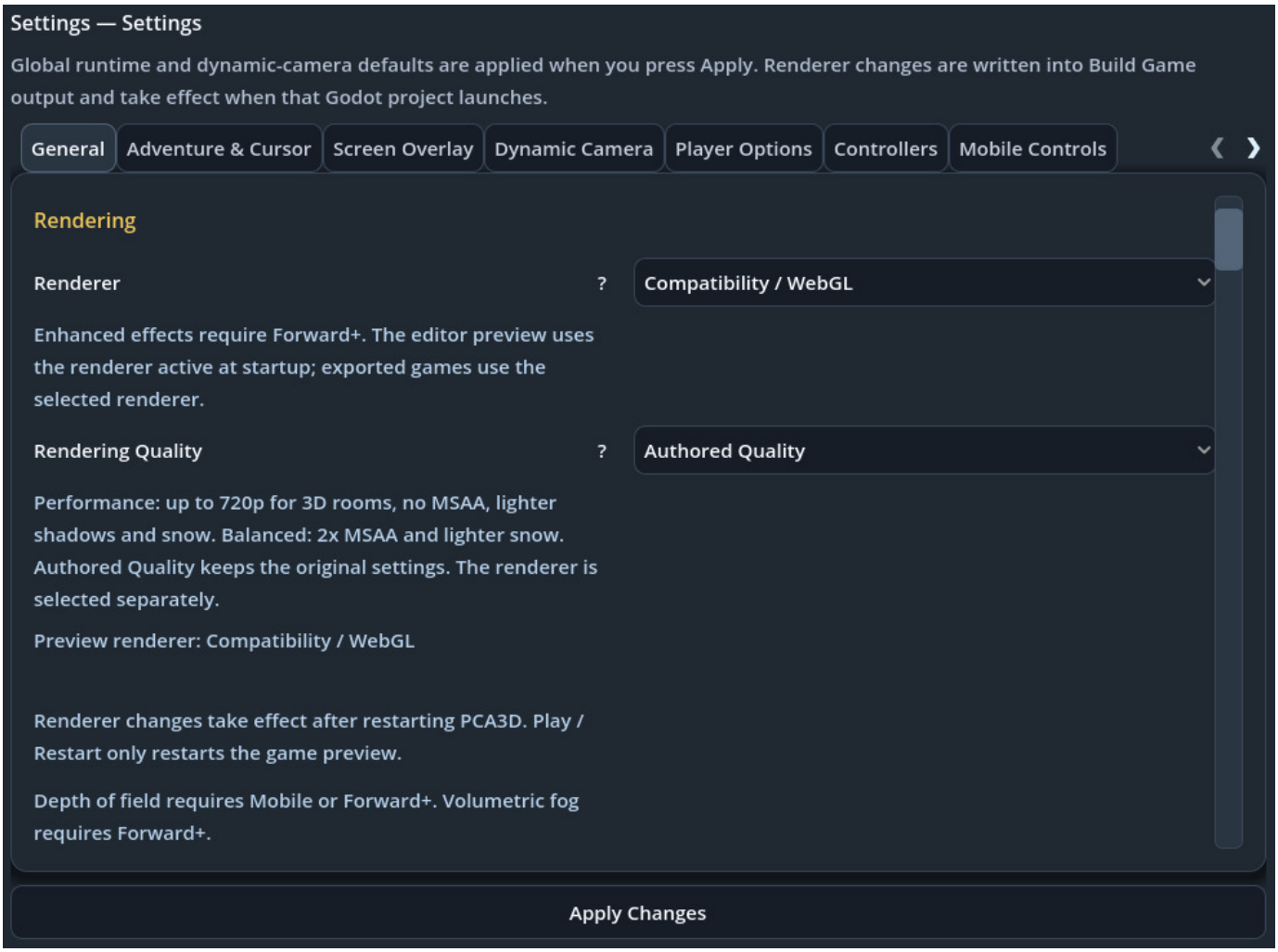

Figure 4. Shared game settings. The tab strip groups a large number of options; the form can also scroll vertically.

Use Settings for renderer choice, ambient light defaults, shadows, antialiasing, movement defaults, interaction rules, sound options, save-system behavior and the default HUD, Inventory and Journal GUIs. A room may override some shared presentation settings.

For these tutorials, keep the supplied click-to-interact controls until the puzzle works. You can change the control scheme afterward without rebuilding the puzzle actions. Optional Runtime Control Profiles add point-and-click, direct movement, first person and drag-to-move profiles. Automatic device switching, cursor adaptation, proximity interaction and the verb coin have separate options beneath the master switch.

The interaction mode determines how a click chooses a verb. Context-sensitive interaction uses the hotspot's default verb. A verb bar lets the player choose Look, Use, Talk or Take first. Cycling changes the active cursor/verb. A verb coin shows a choice near the clicked target. The actions attached to the target remain the same; the mode changes how the player reaches them.

Keybindings

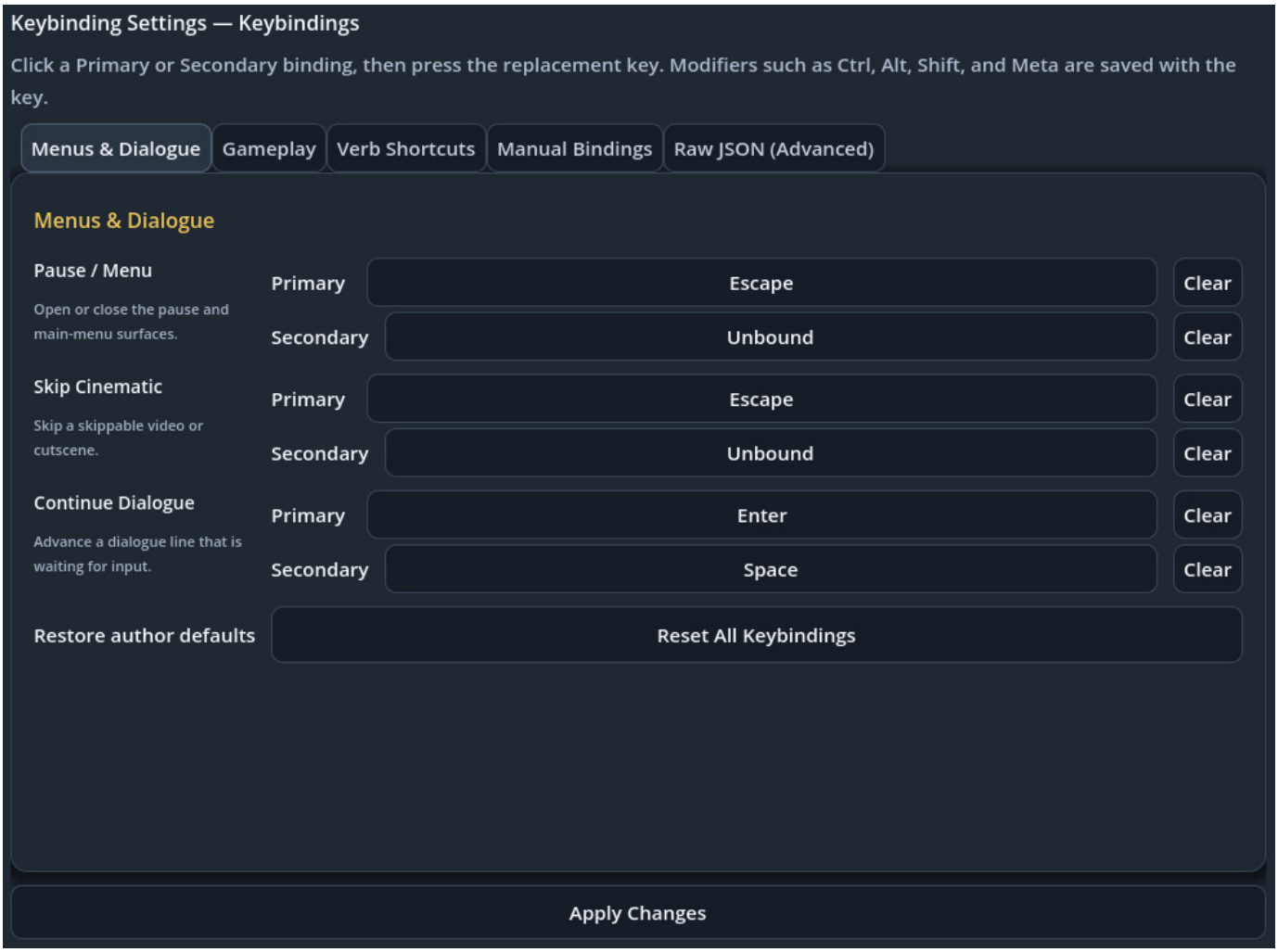

Figure 5. Keybinding settings keep game controls together. Editor shortcuts and game bindings are different sets of controls.

Check the bindings for movement, inventory, journal, menu and continuing dialogue. A text field taking input should not also trigger an inventory shortcut. Test keyboard controls while the running game has focus, and test text entry with a text box focused.

Change controls from an action

Use Set Runtime Controls Enabled to turn the optional profile system on or off. Set Control Setting adjusts an individual profile option. Set Interaction Mode changes the interaction style. These actions are available in ordinary action lists, visual graphs and text scripts.

For example, a tutorial prompt could turn cursor adaptation off, demonstrate the mouse controls, then turn it back on. Keep the choice optional for the player when the game does not depend on one particular device.

4. Build and navigate a room

Scene Designer is the main workbench for rooms. The editor camera lets you inspect the room; the room's game camera determines what the player sees. Moving one does not automatically move the other.

Choose a room type

A new room starts unassigned. Select its room record, choose 3D, 2D Stage or 2.5D Stage and apply. Make that decision before building the room. A room's dimension family is locked after assignment to protect incompatible content. For the two projects in this book, choose 3D.

A 3D room uses actual geometry. A 2D Stage keeps a perpendicular stage camera and flat composition. A 2.5D Stage combines a background with depth-aware actors and navigation. The grid is an editing aid; it is not a wall around the world.

Move the editor view

Input	Result
Right mouse drag	Rotate the editor view.
Right mouse with W/A/S/D/Q/E	Fly through the room; hold Shift to move faster.
Middle mouse drag	Pan the view.
Mouse wheel	Zoom.
F / Frame Selected	Bring the selected item into view.
Frame Scene	Fit the room.
Top / Plan View	Work on the floor plan from above.

Click a visible object or its row in Scene Objects to select it. Use W for Move, E for Rotate and R for Scale. Red is X, green is Y and blue is Z. Drag a handle deliberately; simply clicking a model is a selection, not a request to move it.

Use the Inspector for exact transforms. In the placement tables, Position is the object's actual room position. Visual Adjustments offset its appearance relative to that anchor. A model can need a visual correction without moving its interaction point or its navigation obstacle.

Add, duplicate and organize

Use Add to Scene for a primitive, imported asset, character, camera, spawn, light, trigger or other room item. Choose the required type, name it, set its ID and apply. Duplicate repeated scenery, then change the new position and ID. Use folders for Architecture, Props, Characters and Effects.

Most walls and furniture should have Hotspot disabled. Otherwise they become targets just because the player clicks them. Enable a hotspot only when the object needs Look, Use, Take or Talk behavior.

Three different boxes

The yellow selection outline shows the bounds of what is selected. It helps editing and is not a direct statement of where walking stops. The event volume detects entry, exit or overlap. A solid collider or navigation blocker prevents movement. Changing an event volume does not resize a solid model's collision.

For a door that stops a character too far away, inspect its collision shape, local center, scale and approach Stop Distance. Use a custom box collider when the automatic bounds include handles, trim or unused model space. Keep it narrow enough to fit the door but thick enough to block passage. Disabling Blocks Other Characters removes the obstruction entirely; that is why the player can then walk through it.

> Check your work. Add one box, put its center at [0, 0.5, 0] and give it size [1, 1, 1]. Its bottom should meet a floor at Y=0. Rotate the editor view, then return to the game camera. The object should not have moved.

5. Navigation, spawns and cameras

Where the player may walk

Navigation is the allowed floor area. A visible floor mesh does not create it automatically, and a large background image does not enlarge it. Use Edit Navigation to work on the walkable shape from above. Drag vertices, double-click an edge to insert a point and use the Navigation menu to reset or convert older rectangular bounds.

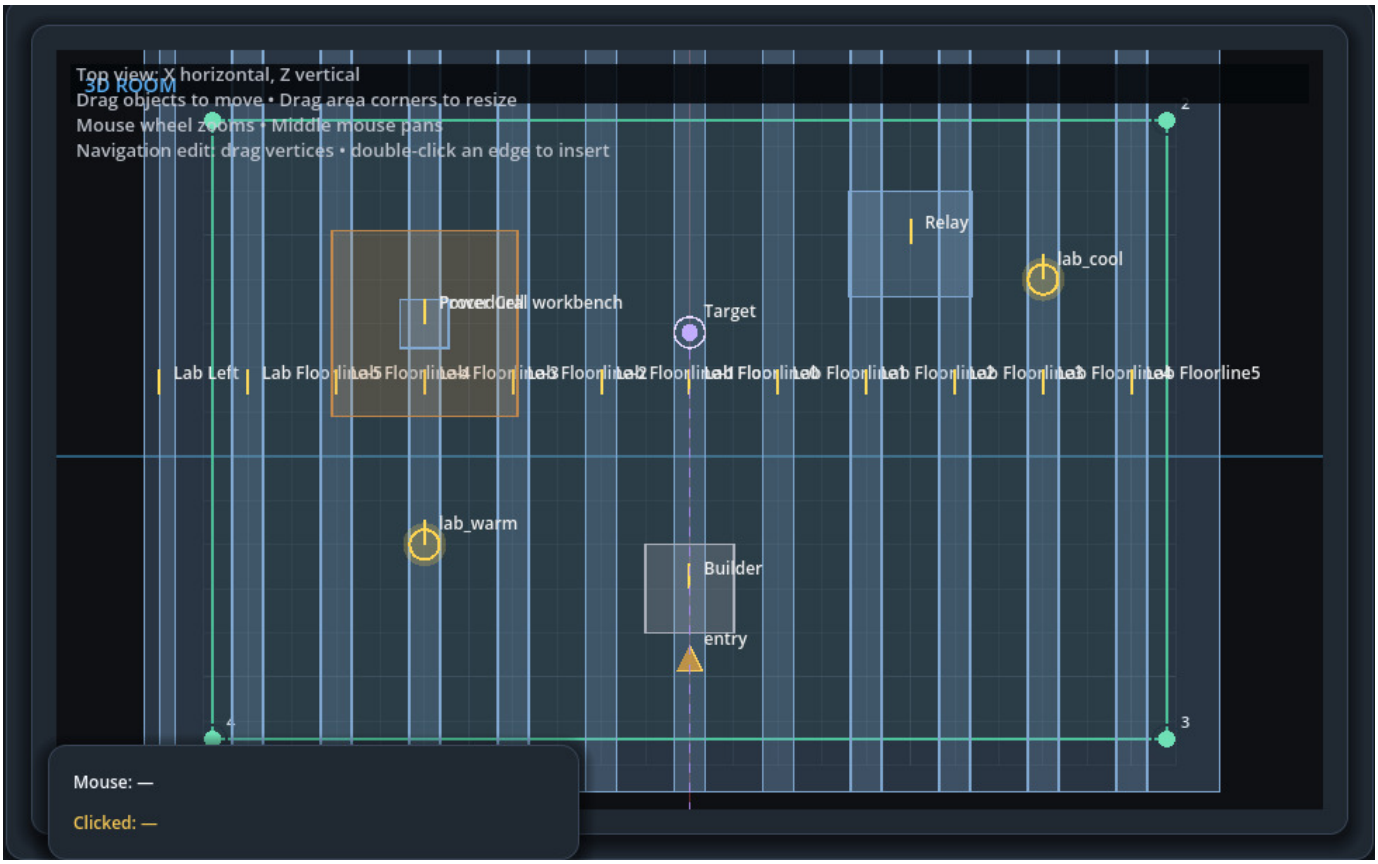

Figure 6. Top view with navigation editing. The walkable area and blockers describe travel, independently of the decorative room meshes.

Keep the polygon simple while learning. Add blockers for solid furniture, then test routes around both sides. Agent Radius reserves clearance around obstacles; it should roughly match the character's walking footprint. Large radii can make a narrow gap impassable even when the mesh looks open.

For a 2.5D room that stops walking before the edge of a large backdrop, expand the actual navigation bounds or polygon, check the room's stage/world dimensions, and check any camera bounds. Also look for blockers and the walkable Y plane. Enlarging only the editor grid does not solve a restricted navigation area.

Spawns and exits

A spawn is a named arrival point. Give it a position on the navigation surface and a facing direction. Use Change Scene with a destination scene and spawn ID. Each room owns its own coordinates, so two different rooms can both have an entry spawn at [0, 0, 2.3].

Place an arrival point clear of the destination door's blocker. If it sits inside a solid object, the player may appear trapped or be pushed away. A room entry transition belongs to the destination room; use a short fade while learning.

Author the game camera

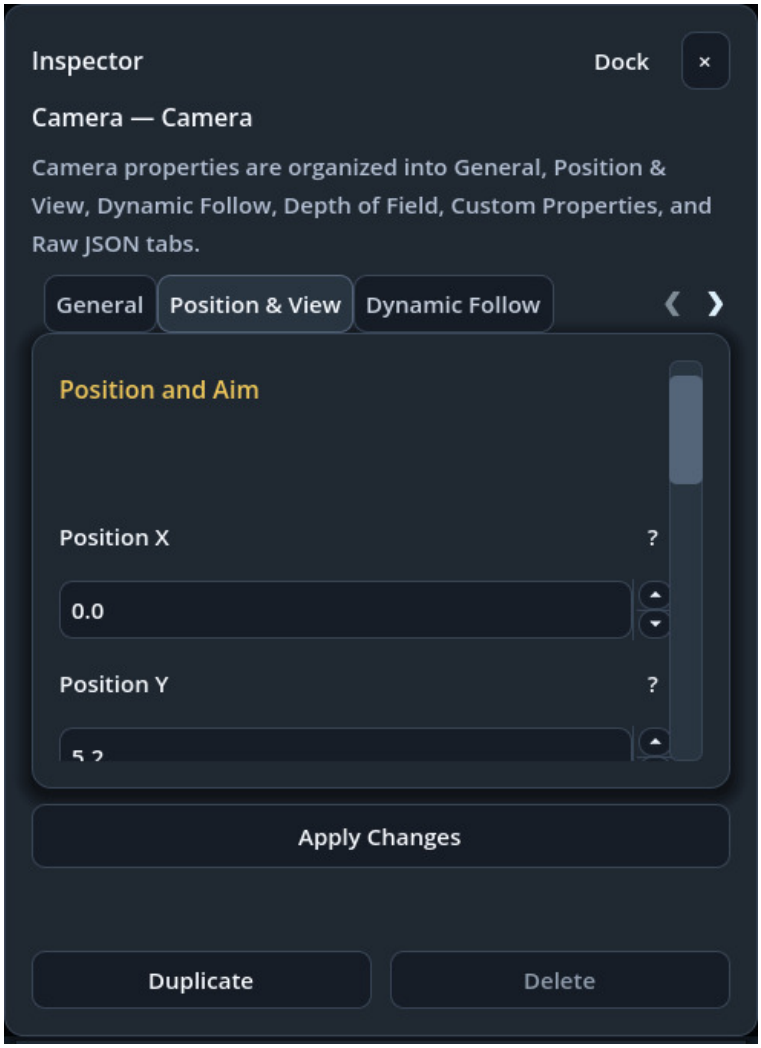

Figure 7. A room camera selected in the Inspector. Position and look target define the shot; the editor view remains free to move.

Select a camera under Cameras. Set Position, Look Target, field of view and clipping planes in Position & View. A perspective camera has depth and perspective; an orthographic camera uses view height instead of perspective distance to control framing.

Use Copy View to Camera when you have already found a useful editor angle. Then check the shot in Test Scene. Set the room's Default Camera to this camera. A camera can exist in a room without being the active camera.

Advanced tracking is covered during Clockmaker. For your first room, a fixed camera is easier to judge. Do not make camera movement and player movement your first two debugging jobs at the same time.

6. Characters, models and animation

A reusable Character defines who somebody is. A placed character says where that person is in one room. The model asset is the file that supplies the body. Animation states say which clip to play for a job such as idle or walk.

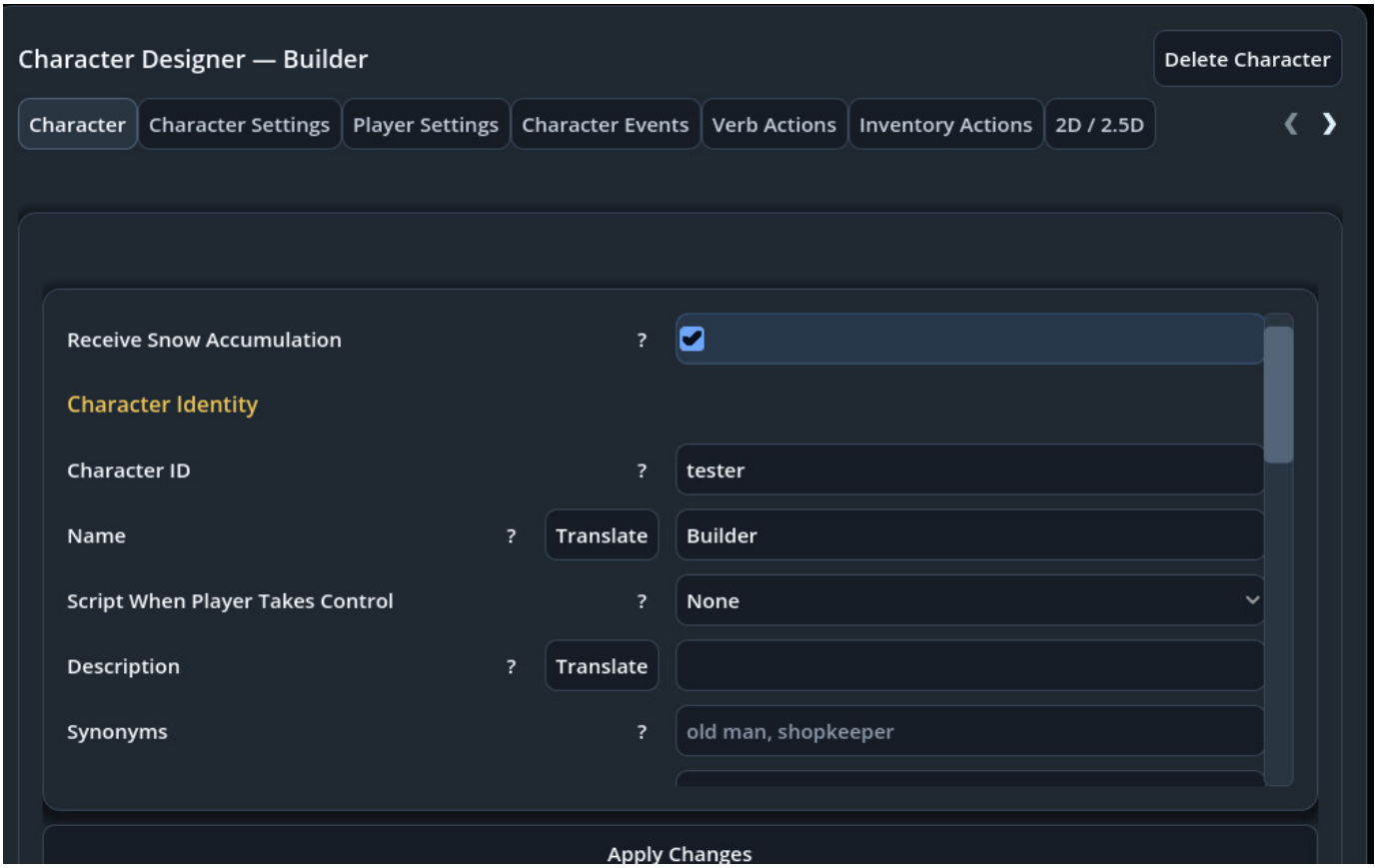

Figure 8. The Primitive game's reusable Tester character. A capsule is enough to test movement and interaction before importing a model.

Make a reusable character

1. Add a Character in the Adventure Database. Enter its name and stable ID.
2. Choose its model asset, or use the stock character presentation for a blockout.
3. Set walking speed, collision radius, speech color and facing correction.
4. Enable the relevant playable/player options if the character can be controlled.
5. Apply Changes. Place that Character in the room with Add to Scene, then set its transform.
6. In Project, choose it as the starting character. Place it in the room where the adventure should begin.

If the game shows only a polygon or capsule, first check Model Asset on the reusable Character. Adding animation states does not assign a body. Also check that the room instance refers to the intended character definition and that its asset is enabled, visible and available.

Scale and orientation

Static models can be fitted to a desired height. Skinned models have a skeleton and should keep their rig transforms. Use Raw Model Scale and Model Offset to correct a skinned model deliberately. A giant bind-pose box is not a good reason to repeatedly auto-fit an animated character.

If a character walks backward, check Model Facing Offset before rotating all of its animation clips. If its feet slide while the body moves, compare Walk Speed with the clip speed and check root-motion settings. Lock Root Motion keeps the animation from translating the actor a second time.

Animation States

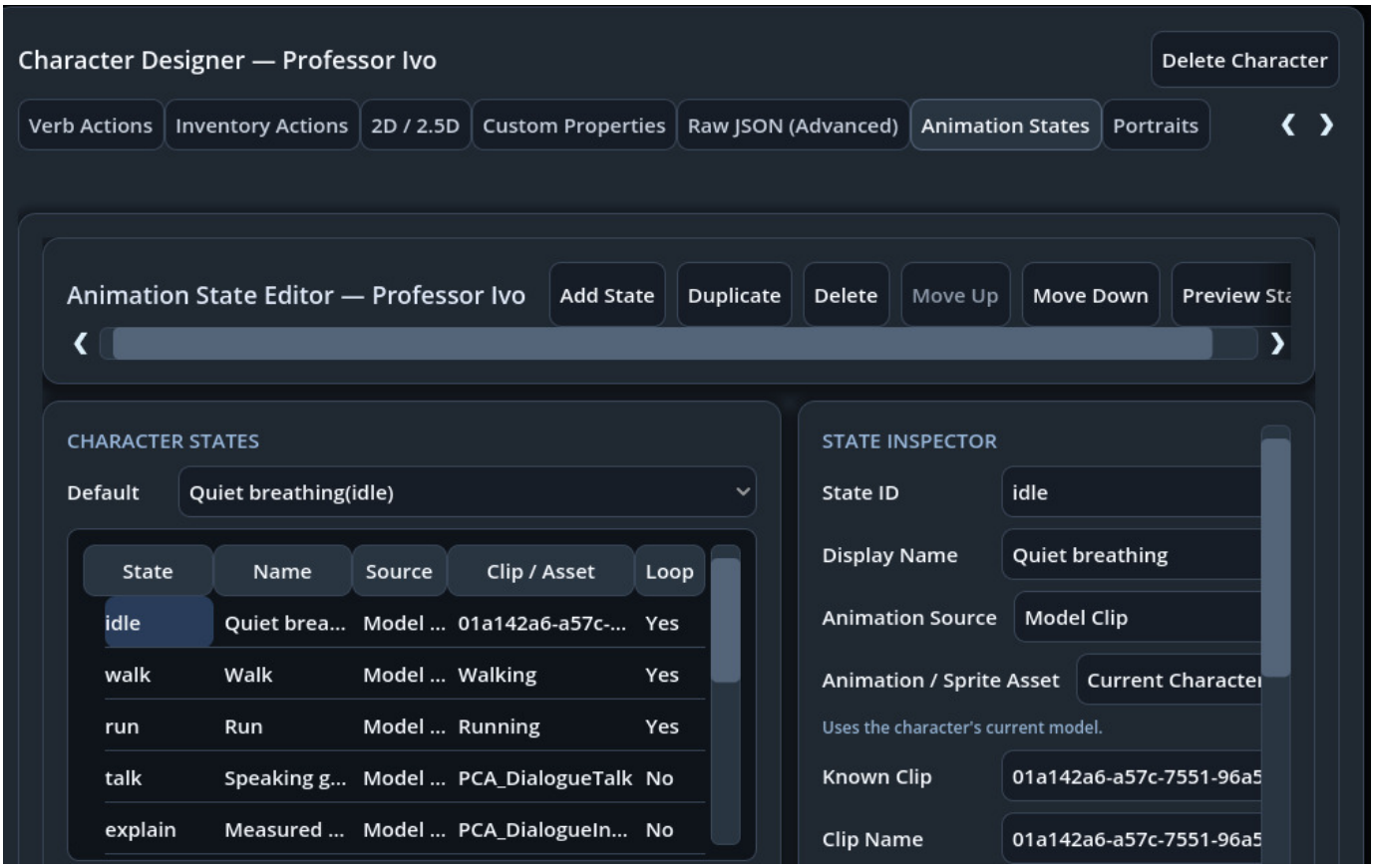

Figure 9. Animation states name the role of a clip. The actual imported clip name can be much less friendly than idle or walk.

Add an idle state and a walk state, select a clip from the assigned model, set looping and speed, and choose the default state. Preview a state to check it. Then test the character walking through a room; a clip preview alone cannot show navigation speed or collision.

Imported clips must fit the model's skeleton and animation paths. Selecting a clip from an unrelated character is not automatic retargeting. Use the supplied Clockmaker rigs unchanged during these lessons; customize them after the example works.

Portraits and lip sync

Portraits can use a still image, frames or supported media. They are separate from the body model. In Lip Sync, choose Scene Character, Portrait or Both and configure each mapping. An enabled setting cannot invent mouth controls that the artwork or model does not contain.

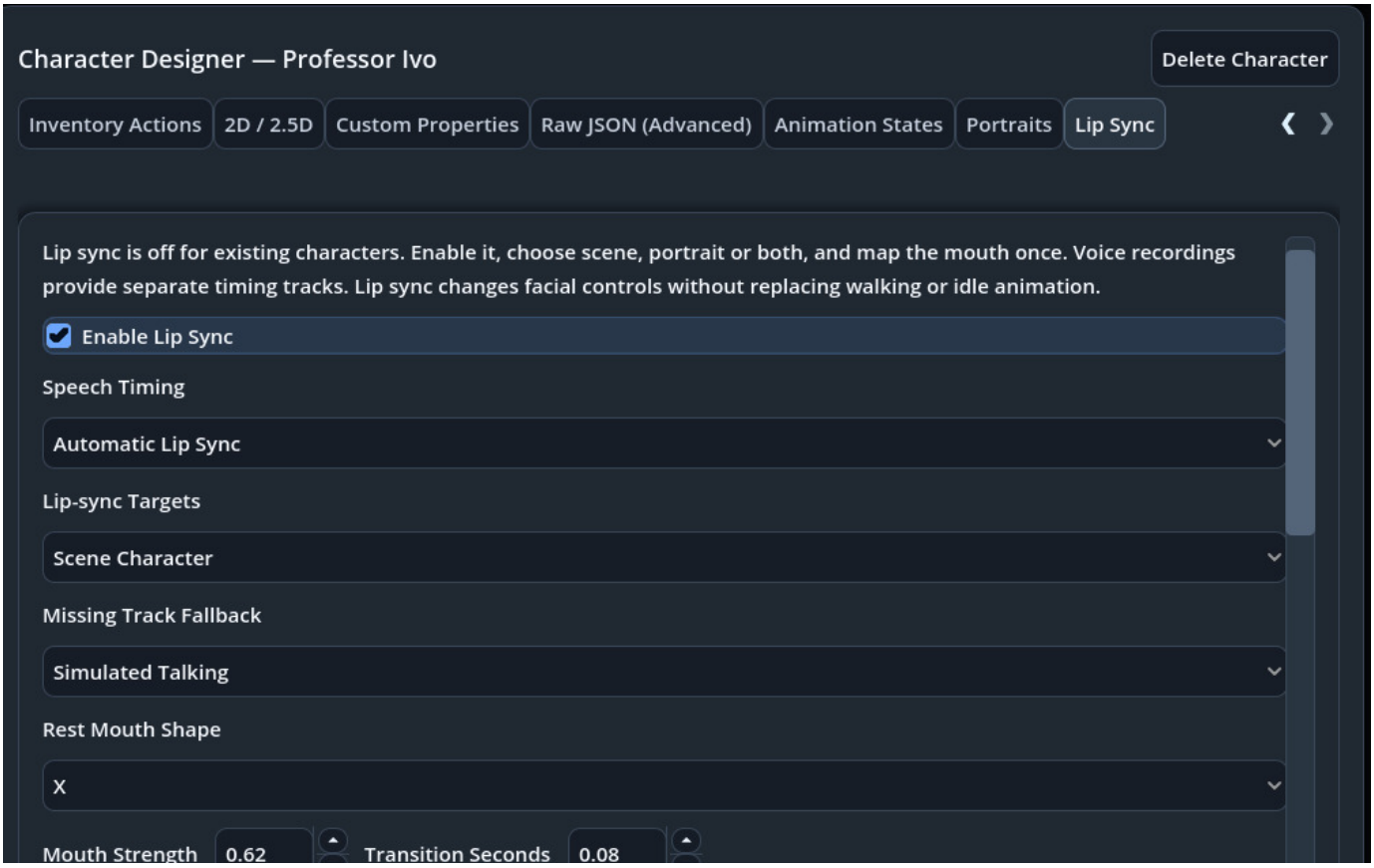

Figure 10. The lip-sync page keeps the character's mouth mapping separate from the timing stored on a voice recording.

For a first lip-sync test, import one short voice recording, enable the character, configure its mouth or jaw mapping, generate or import the track on that Audio asset, review it, and Save Track. Apply the character settings and save the project. Then assign that recording to an actual spoken line and play it.

Basic Mouth Movement follows volume and suits simple open/closed mouths. Prepared cues distinguish mouth shapes more accurately. Rhubarb-compatible tracks use X for rest and A-H for other mouth poses. Portraits and the scene character can follow the same recording while using different artwork.

7. Assets, previews and imported files

An Asset record points to a source file and stores information PCA 3D needs to use it. The asset ID is what a model, GUI, inventory item or sound action refers to. The filename and record name can be different.

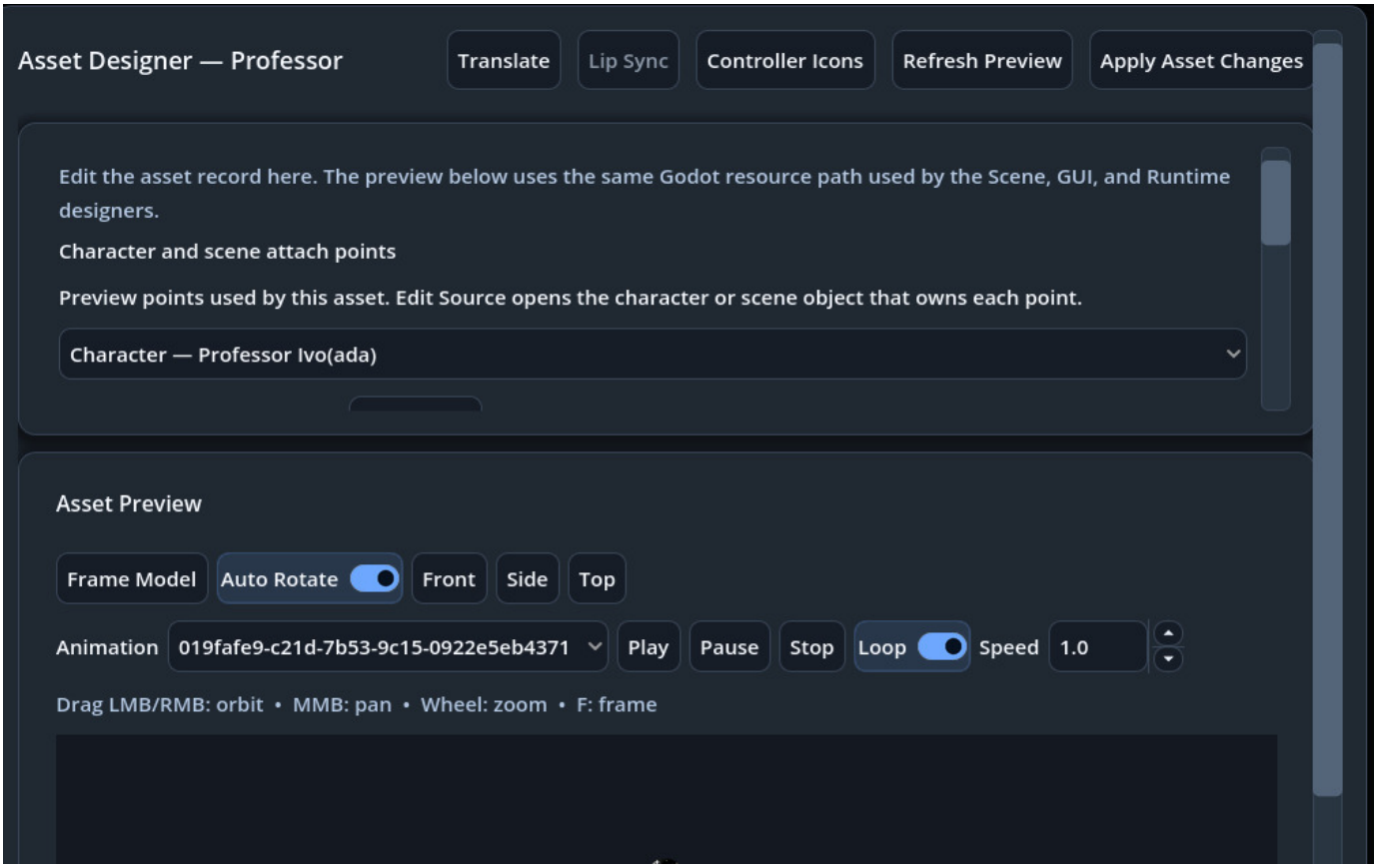

Figure 11. An imported Clockmaker model in Asset Designer. Inspect the actual file and available animation information before assigning it to a character.

Import and check one asset

1. Add or import the appropriate Asset type.
2. Select the source file and give the record a useful name and ID.
3. Review the preview. A model should have the expected orientation and textures; an image should have the expected transparency; an audio file should play.
4. Apply and save. Assign the asset to the record that uses it.
5. Test it in a room or GUI at its intended size. A correct thumbnail does not guarantee a useful scale in the game.

Keep related files together. A model that depends on external textures needs those files as well. Bundled demo assets use res:// paths inside the installation. For your own art, keep an organized project asset folder and verify a copied or exported project on another machine.

Asset Placer

The Asset Placer is a gallery for putting already imported content into a room. Select an asset, place it, then inspect the created room entity. Importing once and placing it several times avoids unnecessary duplicate Asset records.

Adjust the placed object's transform rather than editing a shared source model just to move one copy. For an animated asset, preview its clips and use a state or Play Animation action when the game needs motion.

Images, sound and fonts

Inventory icons and GUI state images should be ordinary Image assets. Choose readable silhouettes with enough padding. Sound effects and music are Audio assets; give voice files one clear line each when practical so they are easy to replace and translate.

Fonts can come from the system or imported files. An imported font is more predictable in a shipped game because another computer may not have your desktop font installed. Test long translated labels before settling on button widths.

Krita and specialist formats

Krita work belongs in the asset workflow. In Asset Designer, choose Image for a flattened drawing, Sprite Sheet for a painted sheet, or Animation for a paint timeline. Import the .kra file. Image imports use the document's saved merged image; Animation bakes frames from supported paint layers. For Animation, choose Krita Clip, check FPS and preview, then Apply Changes and save. Reimport when the original drawing changes.

Krita does not need to be installed to use these imports or run the game. Complex animated transforms, masks and specialized layer modes can differ; export a PNG sequence from Krita when a document uses unsupported features. Check the resulting frame order and alignment before assigning the animation to an actor or GUI.

Controller Icons in Asset Designer opens the bundled controller, keyboard and mouse artwork library. Search for an input, select an icon and use Use Selected Icon, then apply it as an ordinary Image asset. It can appear on a GUI button or a control-help screen. Adaptive prompts can select the appropriate device family when the relevant control-profile hints are enabled.

PCA 3D also includes specialist model and sprite import paths. Use their preview and conversion options, then check the produced asset on the target platform. For this book, stick to the bundled GLB, SVG/PNG and Audio assets so an importer problem cannot interrupt the puzzle lessons.

8. Events, verbs and inventory rules

An action is one instruction: show a message, change a variable, give an item, move a character or switch rooms. An action list runs instructions in sequence. A visual graph can run many of the same instructions with visible connections. A script can call them by name.

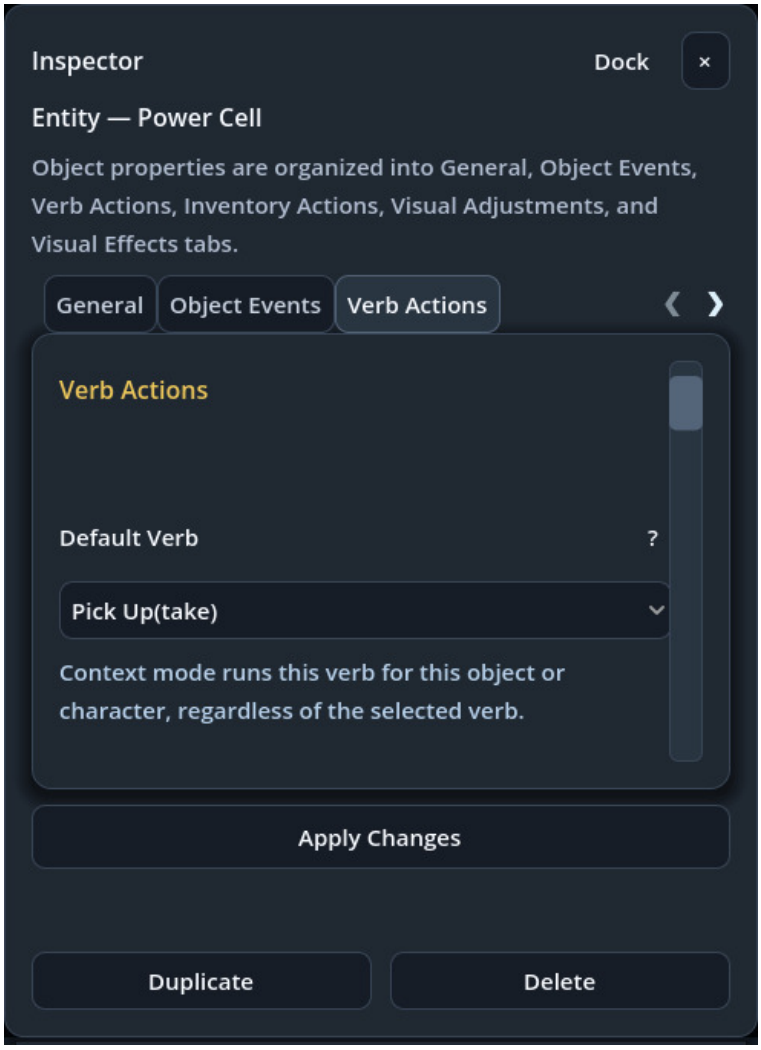

Figure 12. The selected power cell's Verb Actions. Its Take behavior belongs to the physical pickup in the room.

Choose the right event

Object Events handles events such as clicks, entry or collision where supported. Verb Actions handles Look, Use, Talk and Take. Inventory Actions handles using a carried item on a target. A scene's First Enter or Enter events belong to the room, not to a random scenery object.

Avoid putting the same effect in both an object's general click event and its Use action unless you intend both to run. When a click produces two messages or awards points twice, check for duplicate wiring before changing the script.

Make an action list

1. Select the owner: room, object, character, inventory rule or GUI element.
2. Open the relevant Events, Verb Actions or Inventory Actions page.
3. Select the event or verb and add an action.
4. Choose its action type, then fill in its typed fields. Select referenced records from their Name(id) dropdowns.
5. Add the next instruction and use the reorder controls to put the list in the required order.
6. Apply the owning Inspector, save and test that exact event.

If a nested value is not exposed by a typed control, the validated Raw JSON view represents the same record. Use the complete recipe in the appendix rather than guessing a field name. Save a checkpoint before replacing a large nested value.

State makes a puzzle reliable

A boolean is either true or false. Use a flag such as key_taken to remember that a reward was collected. Test the flag before giving the item or points. Without that guard, clicking a drawer ten times may give ten keys.

A number can hold a score or progress counter. A string can hold the current objective. Update the objective after a successful action so the journal follows the player's progress. GUI text such as {objective} and {score} reads those values during play.

Inventory is a separate system

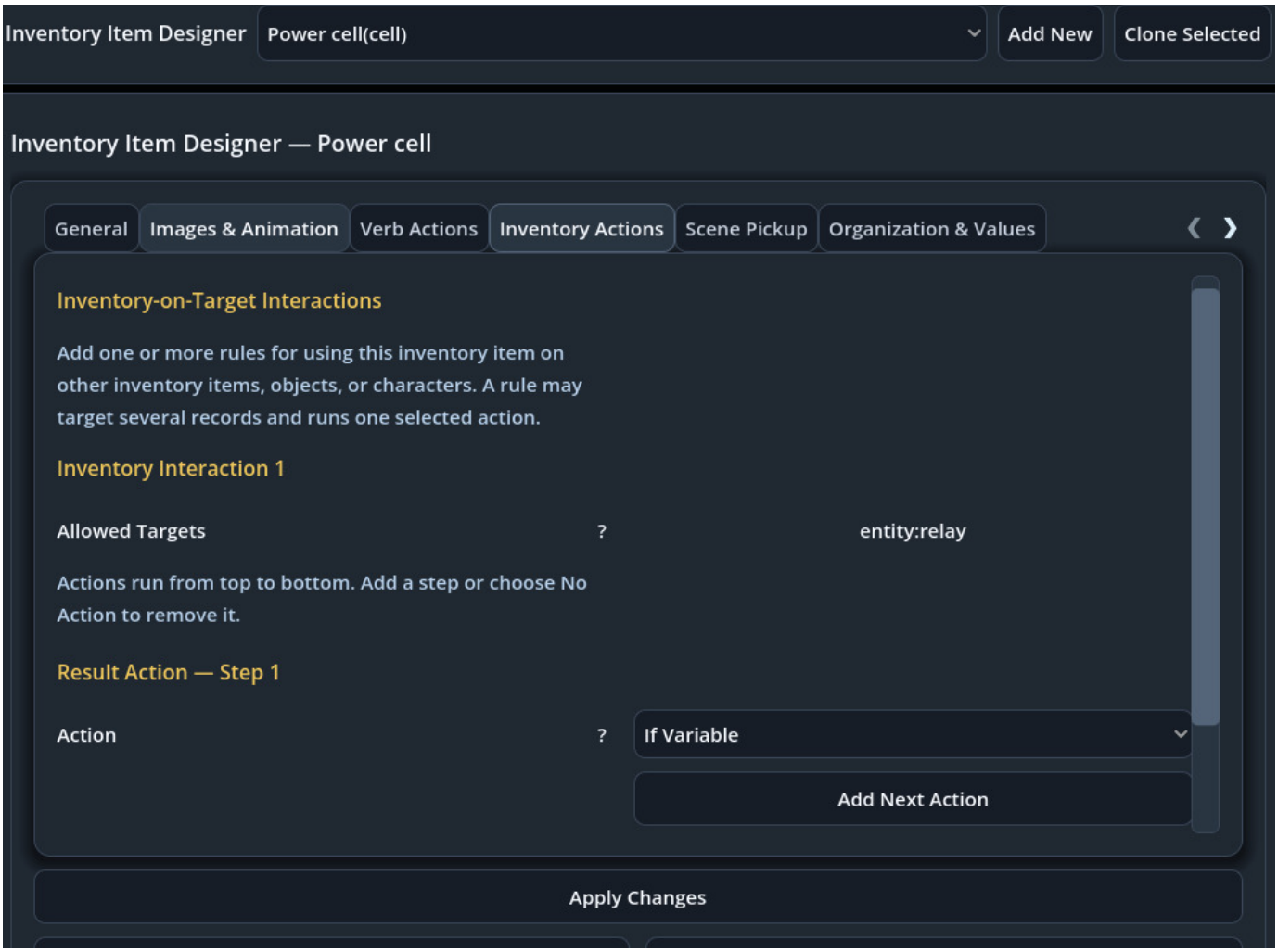

Figure 13. The carried Power cell item. Its rule targets the relay and runs when the player uses this item on that target.

Create an Inventory item definition for what can be carried. Give it a name and icon. Add Inventory puts it in the player's inventory; Remove Inventory takes it out. A physical pickup can hide itself after adding the carried item.

For a use-on puzzle, make a rule on the inventory definition, choose the target object and write the successful actions there. Enable the Use preposition when you want a phrase such as Use key on door. The verb's preposition text and the item's use-on rule do different jobs: one presents the sentence, the other implements the result.

9. GUI Designer: layout and appearance

A GUI is a named collection of elements. The HUD may stay visible during ordinary play. Inventory, Journal, a dial puzzle and an ending panel usually begin hidden and appear only when requested.

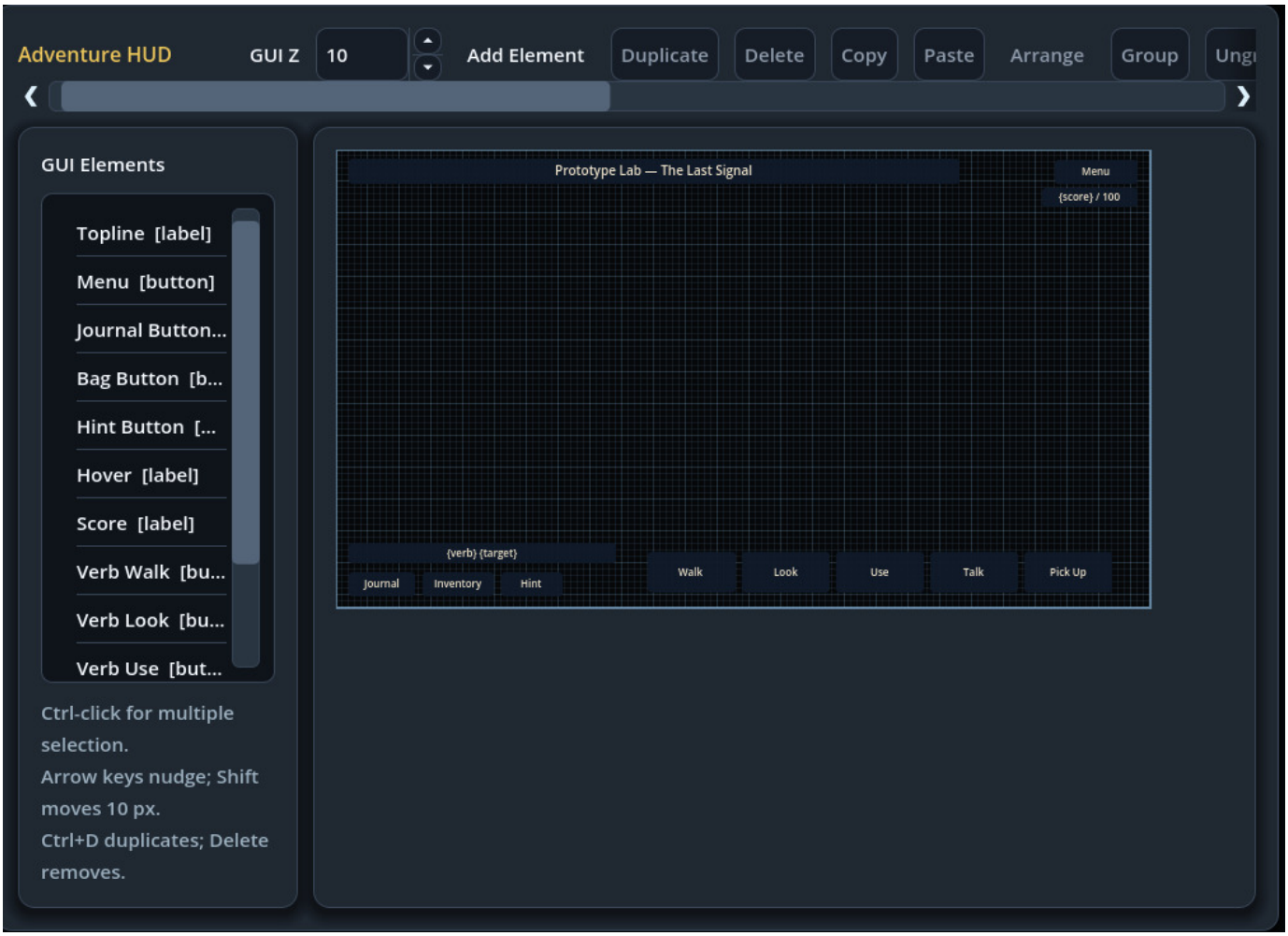

Figure 14. The Primitive HUD at reduced canvas zoom. Zoom changes the editor view, not the game's design resolution.

Create and position elements

1. Add a GUI record and set its ID, initial visibility, modal state and Z order.
2. Add an element of the required type. Select it in the element list or on the canvas.
3. Set its anchor and pixel rectangle in General. Use top-left anchors for the exact demo recipes.
4. Set text or content, then appearance. Dragging and resizing are useful for rough layout; typed numbers are better for matching a recipe.
5. Use Ctrl-click for multiple selection. Align, arrange and matching-size tools tidy a row of controls.

6. Group related elements when they should move together. Check the whole group after resizing it.

The canvas grid and Snap are editing aids. GUI Z controls draw order. An invisible panel can still block input if its element or GUI is configured to do so, so check the active controls when a game click appears to vanish.

Know the element tabs

Tab or control	Purpose
General	ID, type, anchor, size, position, visibility and enabled state.
Text & Content	Labels, font choices, alignment and the displayed content.
Type Settings	Options specific to a slider, inventory grid, model viewer or other control.
Events	The actions assigned to hover and mouse interactions.
Visual Style	Colors, border settings, opacity, reflections and other appearance values.
Images	Normal/background, hover and pressed imagery, including widget subparts.
GUI Window / Activation / Transitions	Whole-GUI presentation, when it appears and how it opens or closes.
Element Shader	Effects applied to the selected element rather than the entire game screen.

Images and corners

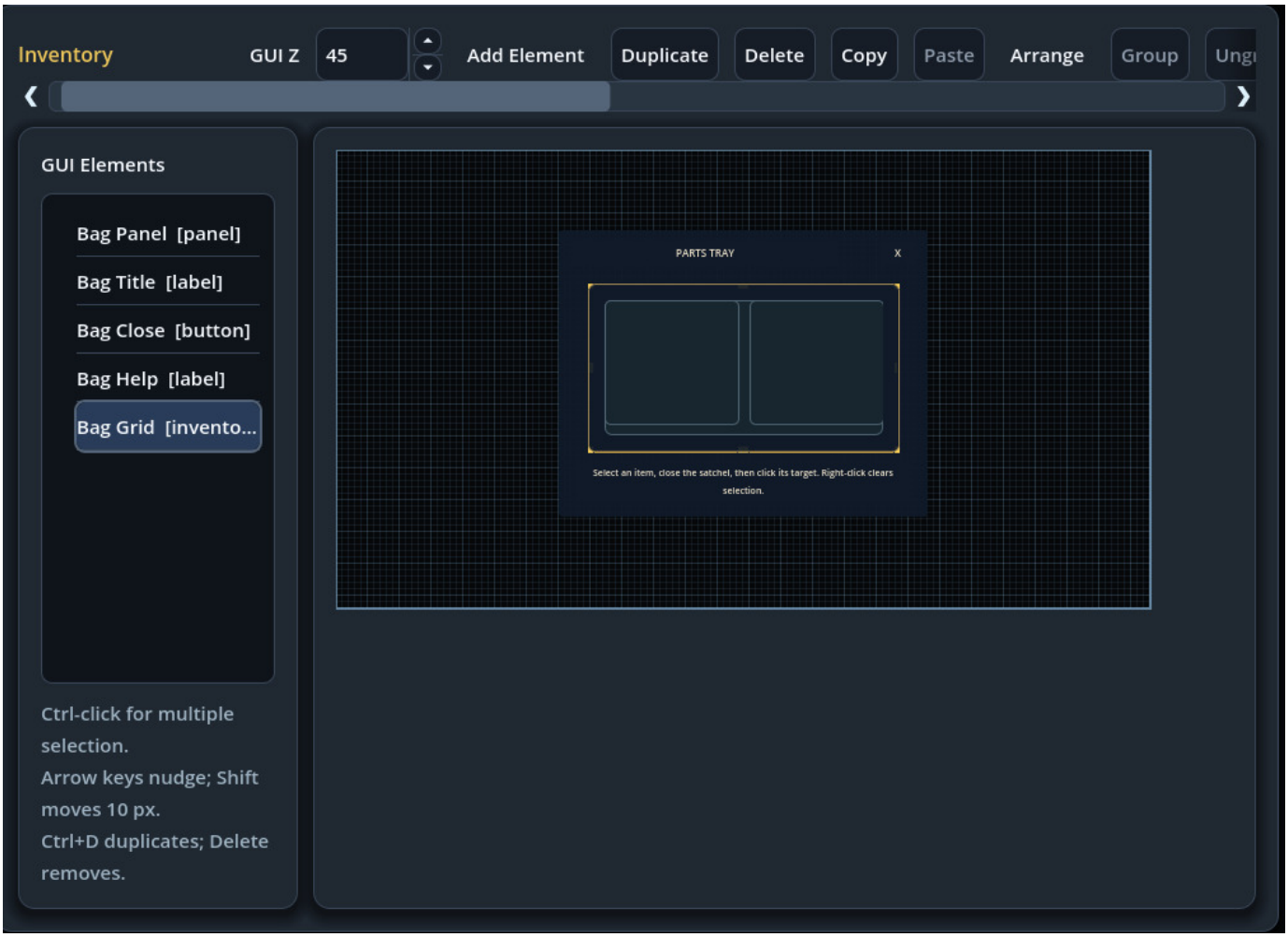

Figure 15. Images for an inventory grid. A compound widget has several separately styled parts.

The normal background is the resting image. Hover is shown while the pointer is over the element. Pressed is temporary: after release it returns to the appropriate hover or normal image. Test both releasing over the button and dragging outside before release.

Each corner can have its own radius. A value of -1 inherits the overall radius; 0 is square. Inventory and Save Game widgets have an outer frame, an inner grid or list, and item/row or screenshot parts. Set each part independently. Changing the outer radius should not be your way of rounding every slot.

For a slider, the dragged handle can use an image. Keep the track and handle visually distinct so the player can see both the available range and current value. For a 3D Model Viewer, choose the model, background image, animation and camera view. Center the model first, then enable user orbit, rotation and zoom if the viewer is meant to be interactive.

Inventory and save widgets

Columns and rows control capacity. Slot width and height control the size of each item area. Gap and padding control spacing. Do not enlarge a slot accidentally while trying to add a second row. Use the independent frame, grid, slot and navigation images for a consistent skin.

A Save Game widget displays slots and optional screenshots. The screenshot area, row area, background and navigation parts can need different normal/hover/pressed images. Treat the Images page as a list of parts to style, rather than one image pasted across the entire control.

10. GUI behavior and preview

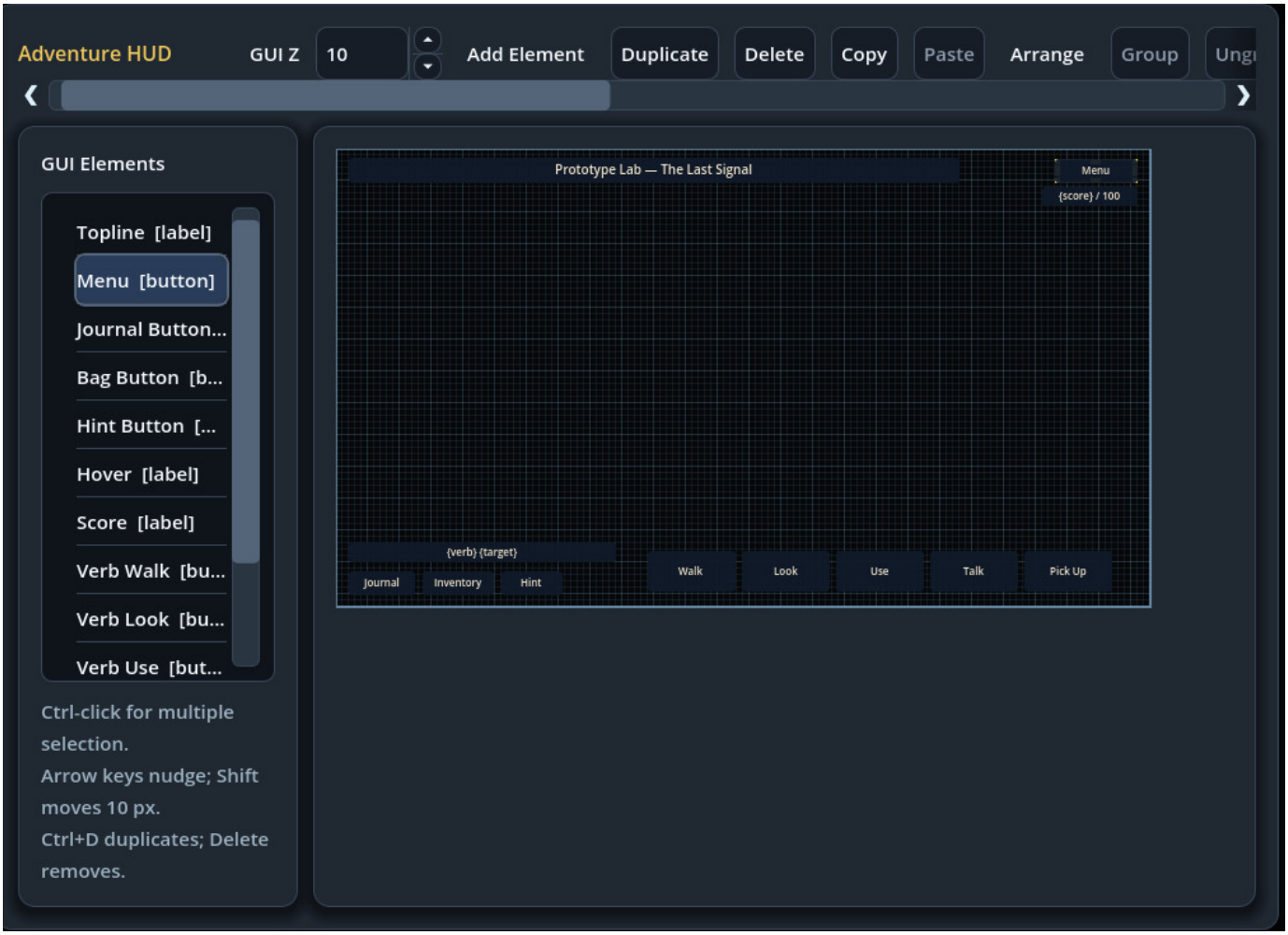

Figure 16. The HUD Menu element's Events page. Use Play / Restart to test the actions assigned here.

Add an event without losing the layout

Select the element, open Events and choose the relevant event. Add an action from the shared action list. A button can show a GUI, call a script, run a visual script, set a verb or perform another supported action. Keep long puzzle logic in a named script or graph so the GUI stays easy to inspect.

The supplied demos also use short legacy Click Action strings such as script:menu or gui:inventory:toggle. These are valid shortcuts. For new work, the Events action editor is clearer when a click needs more than one instruction. Do not enter both forms for the same effect unless you intend the combined behavior.

What Preview does

Turn on Preview in GUI Designer to try hover images, pressed images, sliders, check boxes, inventory styling and model-viewer controls. Preview is for presentation and control feel. It does not execute your authored game events or scripts.

That means a Menu button can show its pressed image in GUI Preview without opening the real menu. To test the real Menu action, run the game. This distinction keeps a harmless layout preview from changing scenes, granting items or saving a game.

A useful test routine

1. In the designer, check the layout at 100 percent and at the actual design resolution.
2. Turn Preview on. Hover, press, release, drag out and return. The pressed image must not remain latched after release.
3. Turn Preview off before moving or resizing elements.
4. Use Play / Restart. Test the assigned actions and whether a modal window blocks the room behind it.
5. Close and reopen Inventory and Journal. Check both their opening and closing transitions.
6. Resize the running window and check anchors, long labels and text contrast.

Choose the default system GUIs in Settings. A GUI called inventory does not become the Inventory screen solely because of its ID. The system mapping tells the game which screen to toggle.

11. Visual scripts and text scripts

Visual Script Designer

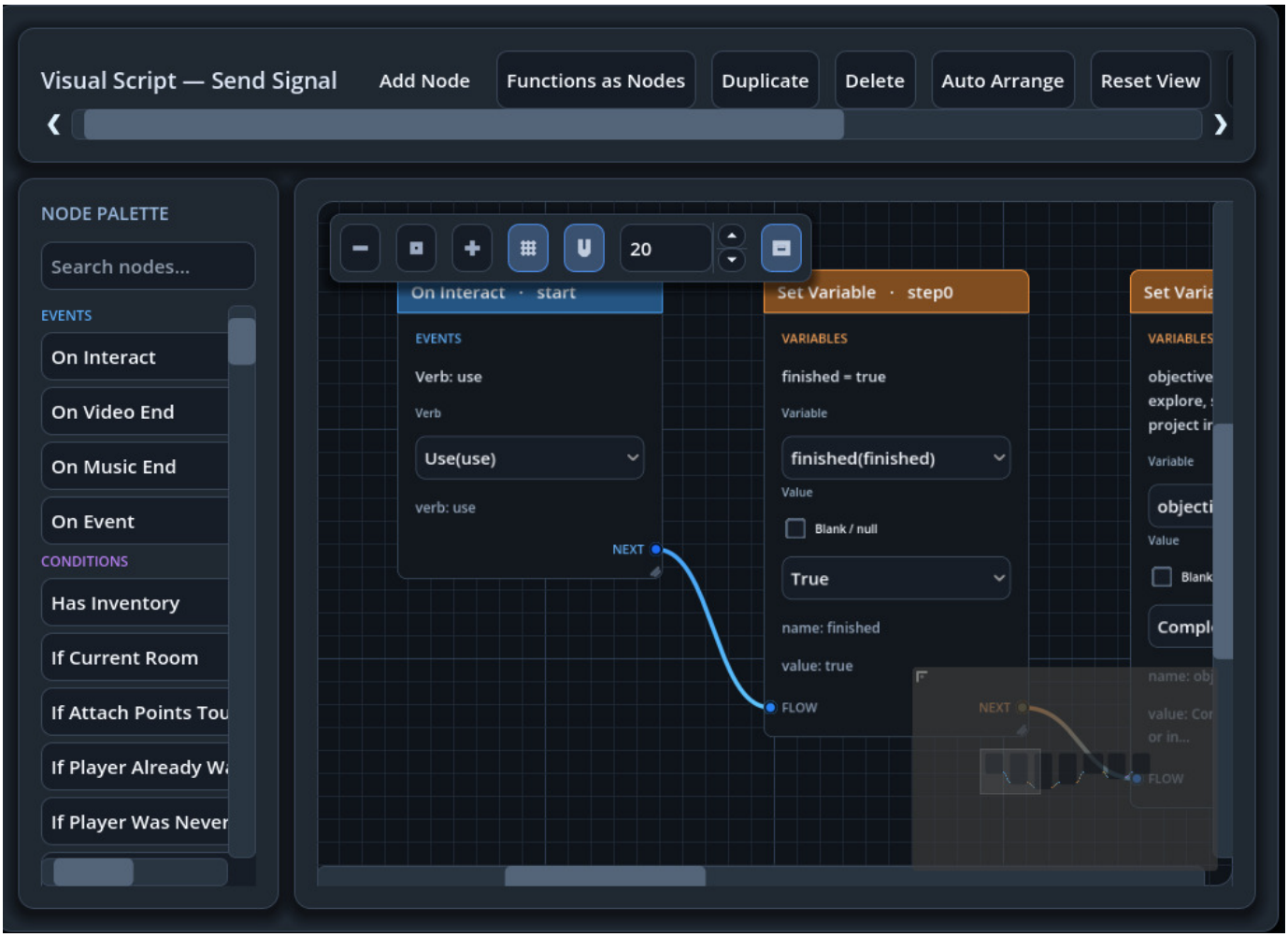

Figure 17. The send_signal graph. Its connections make the completion sequence explicit.

Add a Visual Script record, select an entry node and connect actions in the order they should run. Search the Node Palette by action name. Select a node to edit its parameters. Duplicate and Delete work on selected nodes; Auto Arrange improves the layout, while Reset changes the view rather than rewriting the puzzle.

Flow ports answer what happens next. Value expressions supply data. A row of unconnected action nodes does nothing just because they sit next to each other. Start with one entry and one action, run it, then extend the chain.

If a graph is called by an event, use the entry expected by that graph. The demo completion graphs start at an Interact entry and are invoked through Run Visual Script. Their first action must be connected to the entry's outgoing flow.

The script editors

PCA Script, Lua, BASIC and GDScript are separate script categories. Each record has an ID and source. Choose the language you intend to use when adding the record. The same ID can exist in different language tables, so select the language as well when a function picker asks for it.

The editors provide source entry, validation or checking tools, record selection and run/test controls. PCA Script also provides adventure-oriented navigation, autocomplete, references and debugging tools. Keep project IDs consistent with the names used in source.

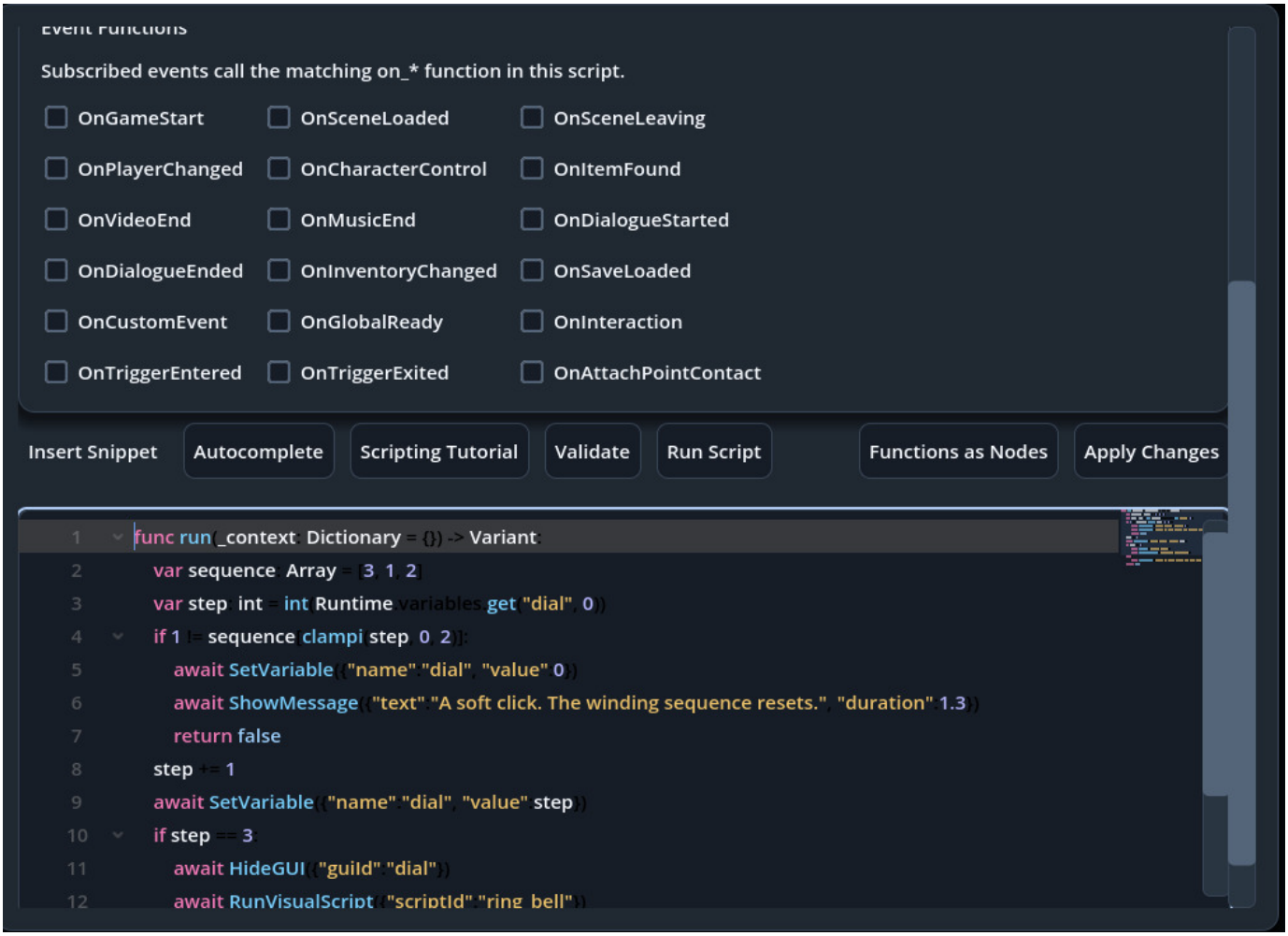

Figure 18. One of the Clockmaker dial scripts. The small script is easier to read than three duplicated puzzle graphs.

The GDScript examples in this book use PCA 3D's normal script base and a run entry point. Helpers such as ShowMessage and SetVariable call the shared action system. await waits for an action that can take time. Use a dictionary of named properties instead of relying on a remembered positional argument order.

Functions as Nodes and Use Definitions

1. Write the actual function in the script's language.
2. Open Functions as Nodes in that script editor and add a function definition.
3. Enter its exact source name, a friendly display name and parameters in source order. Give each parameter a type and a default.
4. Leave Expose as Action and Visual Node checked.
5. Click Use Definitions to accept that definition list. Then Apply Changes in the script editor and save the project.

6. In an action list or visual graph, add Call Script Function. Select the script, language and function. Its named parameter fields should now appear.

Use Definitions is a commit button for the definitions you have just edited. It does not create a graph in the background or run the function. If it appears to do nothing, check that you added a definition, accepted it, applied the script and selected Call Script Function in the destination editor.

Definitions describe code; they do not rewrite a function's signature. If you rename or reorder parameters, update the source and definition together. Result Variable is optional and receives the function's return value.

What this means in a script

this refers to the object or record whose event called the script. A shared door script can read this.id rather than containing one particular door ID. GDScript self still means the script instance. It is not a replacement for the game object's this.

A startup action may have no calling object. Check this before reading it in code that can also run without an owner. In visual/action target fields, choose this when the action should affect its caller. A scene change does not turn an existing caller into a different object.

```
func describe(amount, label):
    var owner_name = this.name if this != null else "No caller"
    return {"owner": owner_name, "value": amount, "label": label}
```

Expose describe with amount as int, default 7, and label as string, default Hello. Call it from an object's event, store the result if needed, and inspect the output. This is a useful first function because it proves both parameters and caller identity before it changes the game.

12. Dialogue, messages and sound

Dialogue Designer builds conversations as a graph. A Line displays speech, a Choice offers responses, an Action changes the game and an End closes the conversation. Connect outputs to destinations, then set the intended first node as Start.

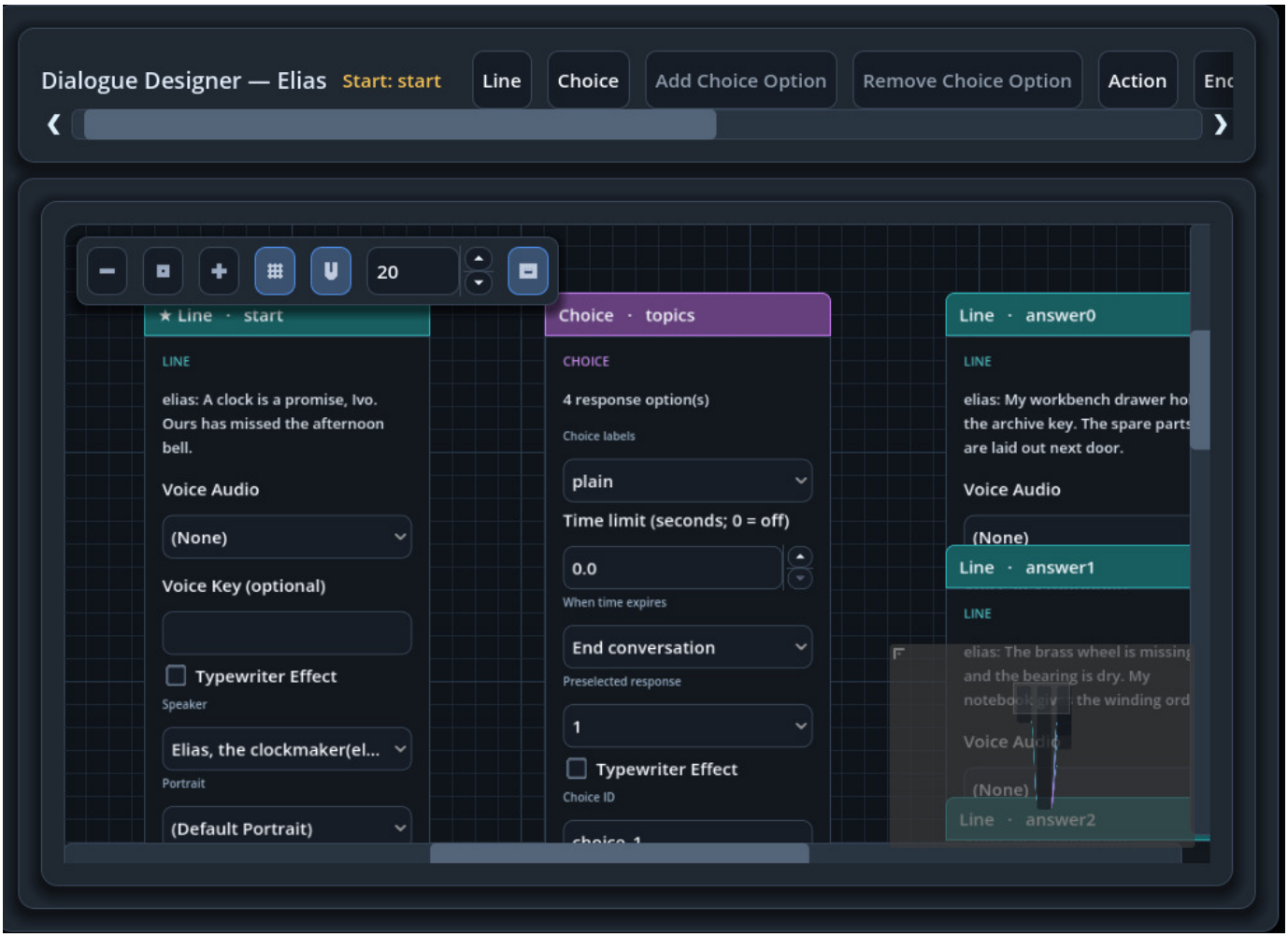

Figure 19. The Elias conversation. Its answers return to a common topic list.

Write a small conversation first

1. Add a Dialogue and a Line node. Choose its speaker and enter the line.
2. Add a Choice with two responses. Connect the Line to the Choice.
3. Add a reply Line for each response and an End node for leaving.
4. Connect the appropriate response outputs. Use Return to Previous on an answer that should return to the choice it came from, or wire an explicit return.
5. Set Start, use Auto Arrange and Test Dialogue.

A condition on a response controls whether it is offered. It does not run the response's actions by itself. Keep conditions simple until both branches work. After deleting a node, validate the graph so no destination points at a missing node.

Speech, narration and ordinary messages

Use a spoken line when a character is speaking. Use Show Message for a notification or unvoiced observation. Pick the right duration or continuation behavior for the amount of text. A long paragraph disappearing in one second is a settings problem, not a reading challenge for the player.

The Typewriter option reveals text gradually. Choose its speed and a sound mode: none, the built-in blip or a chosen Audio asset. The same fields are available in supported message actions and visual scripting. The first Continue can reveal the rest; the next advances. Test that behavior with both mouse and keyboard.

Voice and subtitle timing are related but separate. A typed subtitle should not force the mouth to follow letters while a recording supplies the actual spoken timing. Test voice, lip sync and skipping together.

Sound channels

Use music for sustained background audio, sound effects for short events and the voice path for dialogue recordings. Give sounds meaningful IDs. Set volume before adding several layers; a clock tick, footsteps, wind and music can easily compete with speech.

The Clockmaker opening uses prepared voice recordings and helper scripts. The later reconstruction lessons show where they are called. Do not test every new sound at full level and only mix the whole scene at the end.

13. Cutscenes and camera behavior

A cutscene schedules actions on a timeline. Its tracks organize Camera, Character, Dialogue, Gameplay, Effects and Audio. Track names help you read the sequence; the action stored in an event is what actually runs.

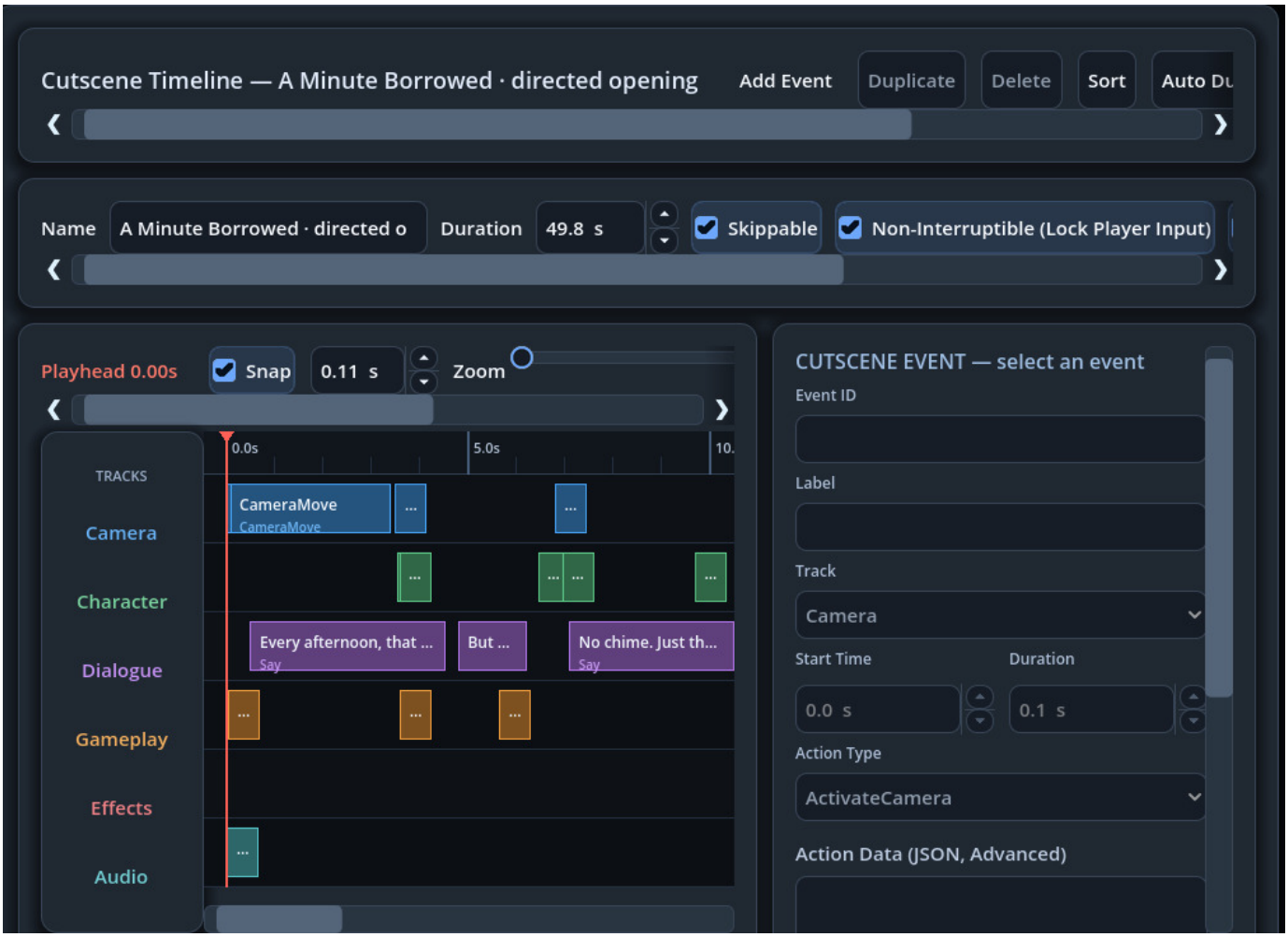

Figure 20. Clockmaker's directed opening. Several tracks run alongside each other rather than one long blocking action list.

Add and test an event

1. Add a Cutscene and set its duration and input/HUD options.
2. Add Event, choose the track and action, then enter its start time and parameters.
3. Set the event's visible duration so the timeline remains readable. An action's own duration or wait field can have a separate meaning.
4. Drag the event to adjust timing, or type exact times in the event Inspector.
5. Test Cutscene from a valid room. Review the result, then adjust one timing issue at a time.

Timeline events often need wait false so speech, movement and a camera blend can overlap. In a sequential action list, wait true is useful when the next instruction depends on the first finishing. Copying one into the other without checking Wait can change the result.

Make a skip path. It must leave characters, camera, GUI visibility and puzzle variables in a usable state. Test a natural finish, an early skip and a skip in the middle of speech. Restore Camera returns to the prior shot when enabled.

Optional advanced cameras

Select a camera, open Advanced Camera, enable it and choose a behavior. Apply Changes, then Test Scene. The ordinary thumbnail shows the authored composition; live tracking needs the running game.

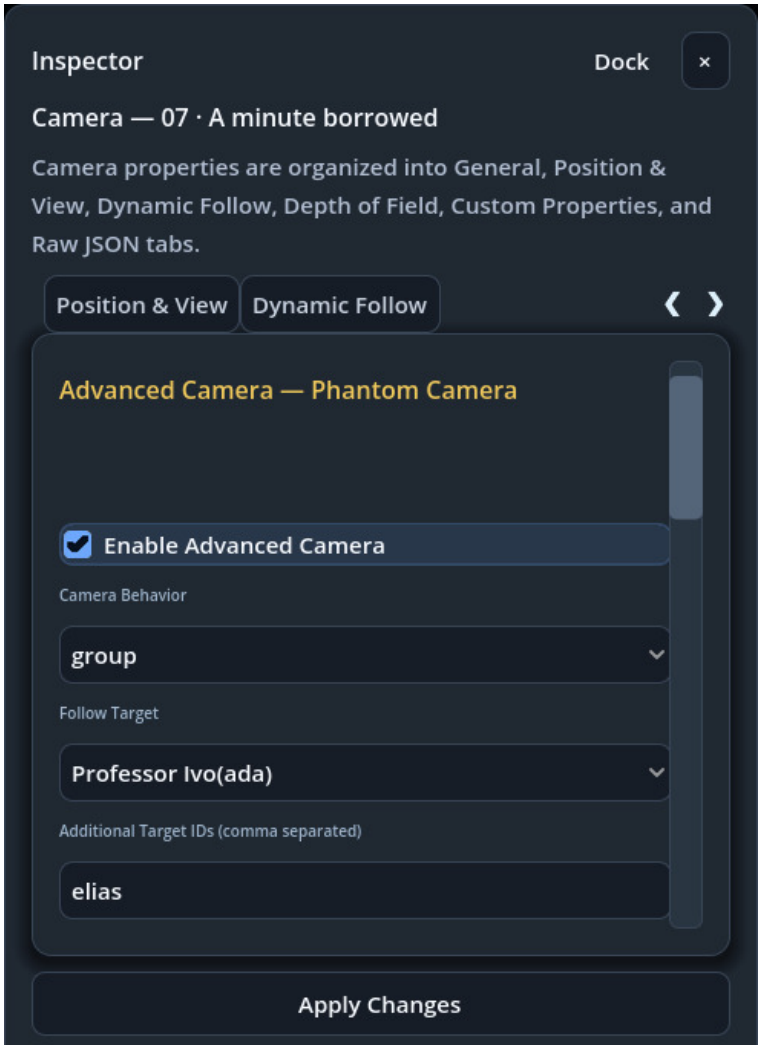

Figure 21. Advanced camera settings in the Workshop. The camera still needs an authored starting shot.

Behavior	Use it when	What to check
fixed	You want a composed shot and controlled blends.	Position, look target and blend time.
follow	One actor should stay in view.	Target, offset above the feet and damping.
group	Two or more actors should share the shot.	Targets, automatic framing, padding and min/max distance.
framed	The camera should wait until an actor reaches a screen boundary.	Screen-zone width and height.

Behavior	Use it when	What to check
path	The camera should follow a target along a defined route.	Two or more path points and a valid moving target.

player is a useful target alias because it follows the controlled character. A target offset near [0, 1.3, 0] aims above the floor. Lower damping reacts faster; zero removes smoothing. Room camera bounds can restrict a shot, so disable Respect Room Camera Bounds for a deliberately unrestricted cinematic angle.

A path camera finds the point on its route nearest the target. It is not a timed rail ride by itself. For a slow establishing move, blend between authored fixed cameras with Activate Camera. Clockmaker's exterior does exactly that.

Activate Camera selects or blends to a shot. Set Camera Behavior changes tracking settings. Set Camera Targets changes who it follows. Set Camera Priority chooses the highest-priority enabled advanced shot. Priority does not create a physical trigger zone; a region or trigger still needs to run the priority action.

14. Lighting, regions and screen effects

Start with a readable room and one useful light. Add atmosphere after the puzzle can be played. Dark corners, glare and fog can hide a missing object as effectively as a broken script.

Lighting and material response

A directional light supplies a broad direction, a point light shines around one position and a spotlight has a cone. Materials control color, roughness, metallic response and emission. Emission makes a surface appear bright; it is not automatically a real light shining on its neighbors.

Reflections need something to reflect. A glossy material may show the environment or a light without showing a moving character. For a character's mirror image, use the supported mirror/planar reflection setup and orient the mirror surface correctly; roughness and metallic sliders alone do not create another rendered view. Test with an obvious object in front of it and with the renderer you intend to ship.

Regions and triggers

A trigger detects an event volume. A region describes an area and can also carry interactions or a visual treatment. Give the area a name and simple bounds, then test entry and exit before adding a shader. For clicks or verbs, enable the relevant interaction and add its actions just as you would for another target.

Area effects such as tint, glow, blur or a shader need a clear boundary and sensible intensity. A mirror region also needs the corresponding reflection configuration. Do not expect a material word such as metallic to manufacture a geometrically correct mirror in every view.

Screen shaders versus object effects

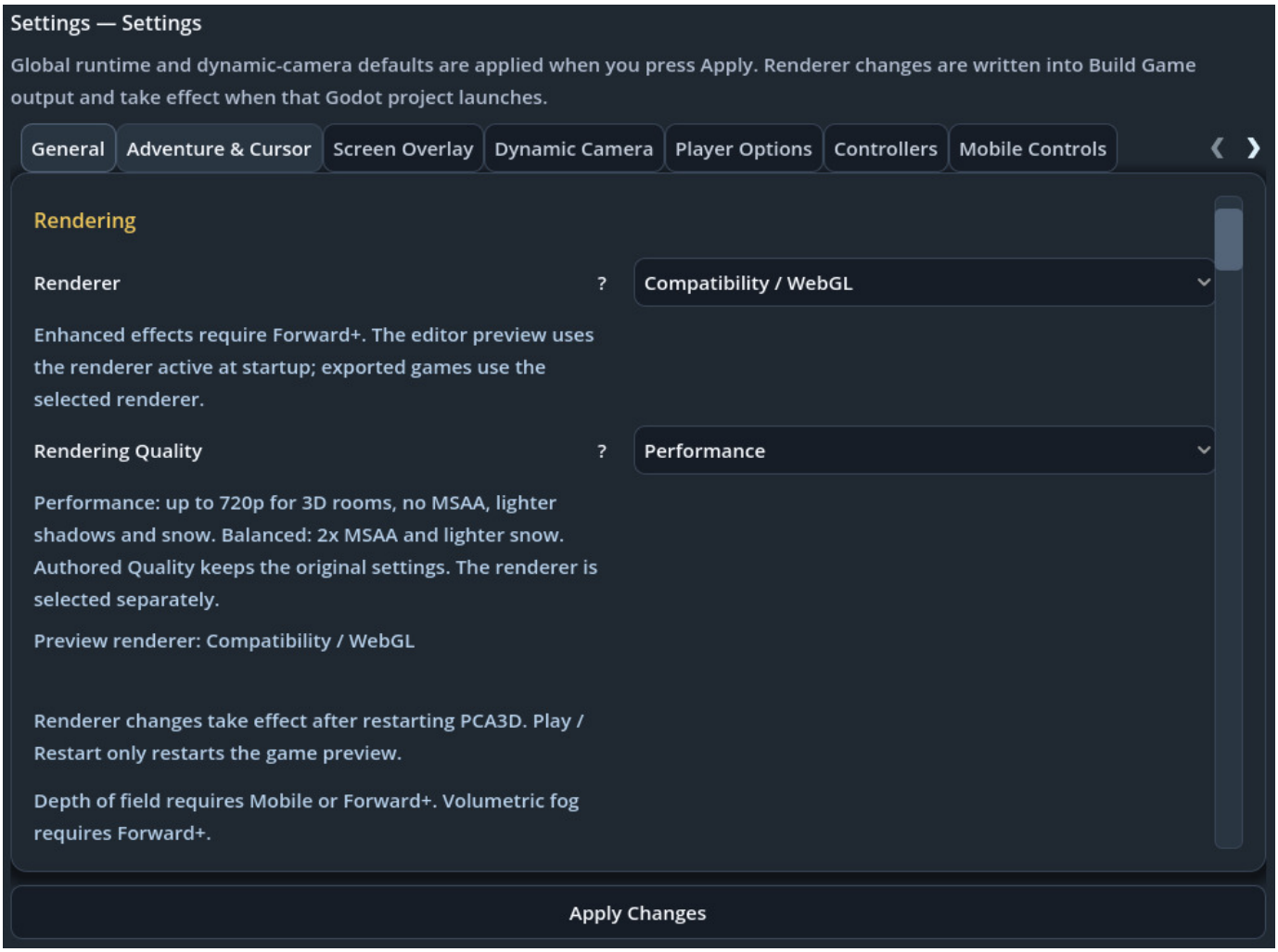

Figure 22. Screen shader settings. These affect the rendered scene rather than only one selected model.

PSX, N64 and Horror Fog belong in the screen shader stack when you want a whole-screen look. Set an effect, test it in the game, then add another. Strong pixelation, color reduction, blur and fog all compete with text and fine details.

An object shader changes the selected object. An element shader changes a GUI element. A whole-scene silhouette can use edge-only outlining when you want contours rather than a filled silhouette. Keep these scopes straight when an effect appears in the wrong place.

Zoom blur suits a short emphasis: a sudden discovery, a hit or a transition. Use a brief rise and fall instead of leaving a heavy blur over ordinary exploration. Check whether GUI text is included in the chosen effect path.

Renderer and performance

Forward+ offers the richer 3D effect path but costs more than Compatibility. Mobile and Compatibility have different effect support. Use the preview renderer label to see what is actually running. A project option and the editor's current startup renderer may require a restart to match.

For a modest laptop, reduce shadows, expensive reflections, depth of field, particle count and resolution before simplifying puzzle content. Compare a fresh Primitive game with Clockmaker to separate a rendering load problem from a general startup problem. Keep a record of renderer, graphics device and the scene that fails.

15. The remaining designers

These tools are worth knowing before you need them. You do not have to master them all to finish the first game.

Variables and quests

Variable Designer gives each value an ID, type and starting value. Use boolean for flags, number for counters and string for text. Choose the type before wiring conditions. A string containing false is not the same kind of value as the boolean false.

Quest records hold journal stages and objectives. Start Quest makes a quest active; Complete Quest marks it finished. A separate objective string in the demo HUD is a convenient short prompt, not a substitute for the Quest record.

Verbs, cursors and fonts

Verb records define actions such as Look and Use, including whether a preposition is allowed and its text. A Cursor record controls the pointer presentation. Fonts supply text appearance. These are reusable game records, so changing one can affect several screens or interactions.

Name verbs for the player, keep their IDs stable for scripts and ensure the selected interaction mode can reach the verbs you author. A beautifully styled Talk cursor does not create a Talk action on an NPC.

Text Parser

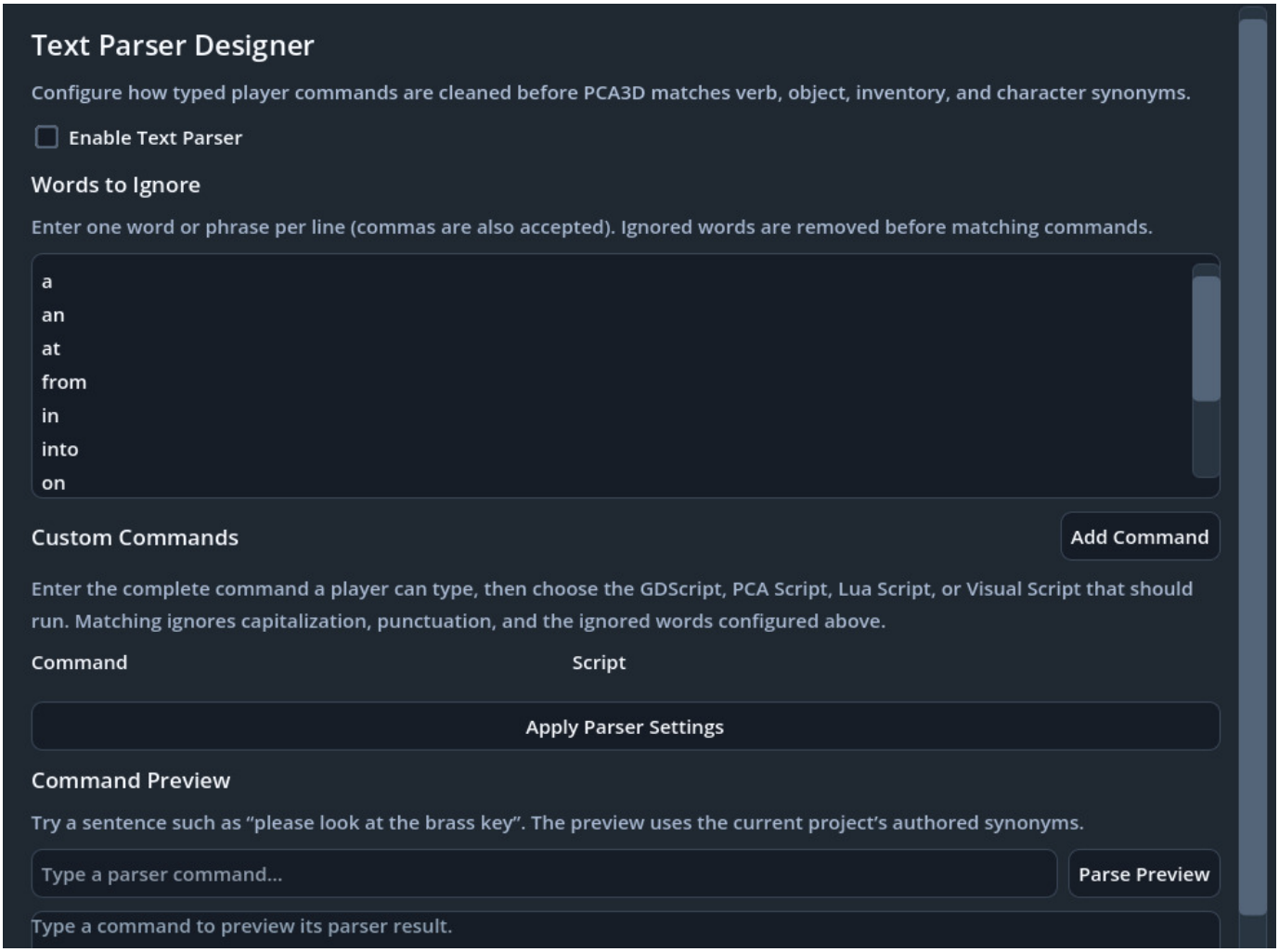

Figure 23. Text Parser settings. Its vocabulary connects typed commands to the same game records and actions used elsewhere.

The parser is an alternative way for the player to express intent. Set the vocabulary, synonyms and response behavior, then test a few simple commands. Give important objects useful synonyms. Test a recognized command, a missing noun and an unknown word so the player gets a useful response rather than silence.

Keep a point-and-click game working while adding parser support. It is easier to debug a command that reaches a known Use action than an entirely new puzzle written only through the parser.

Translation editor

The translation editor gathers localizable text and asset substitutions. Export a CSV for a language, let translators work on that copy, then reimport into the matching language. Stable keys identify the text. Translators should change translated cells, not the keys used to match records.

1. Open Project and click Game Languages & Translations.
2. Add the language code and name, for example it and Italian. Leave the default game language as your original language unless you intend to change it.

3. Choose the intended CSV language and click Export CSV. Include voice and asset mappings only if those substitutions are part of this translator's work.
4. Give each collaborator a clearly named copy and an agreed set of lines or language. Preserve the exported header row and matching keys.
5. Back in the same language, click Import CSV and review the import preview. Resolve conflicts deliberately rather than overwriting existing translations without checking.
6. Save Translations, then save the project. Run the game in that language and inspect actual dialogue and GUI screens.

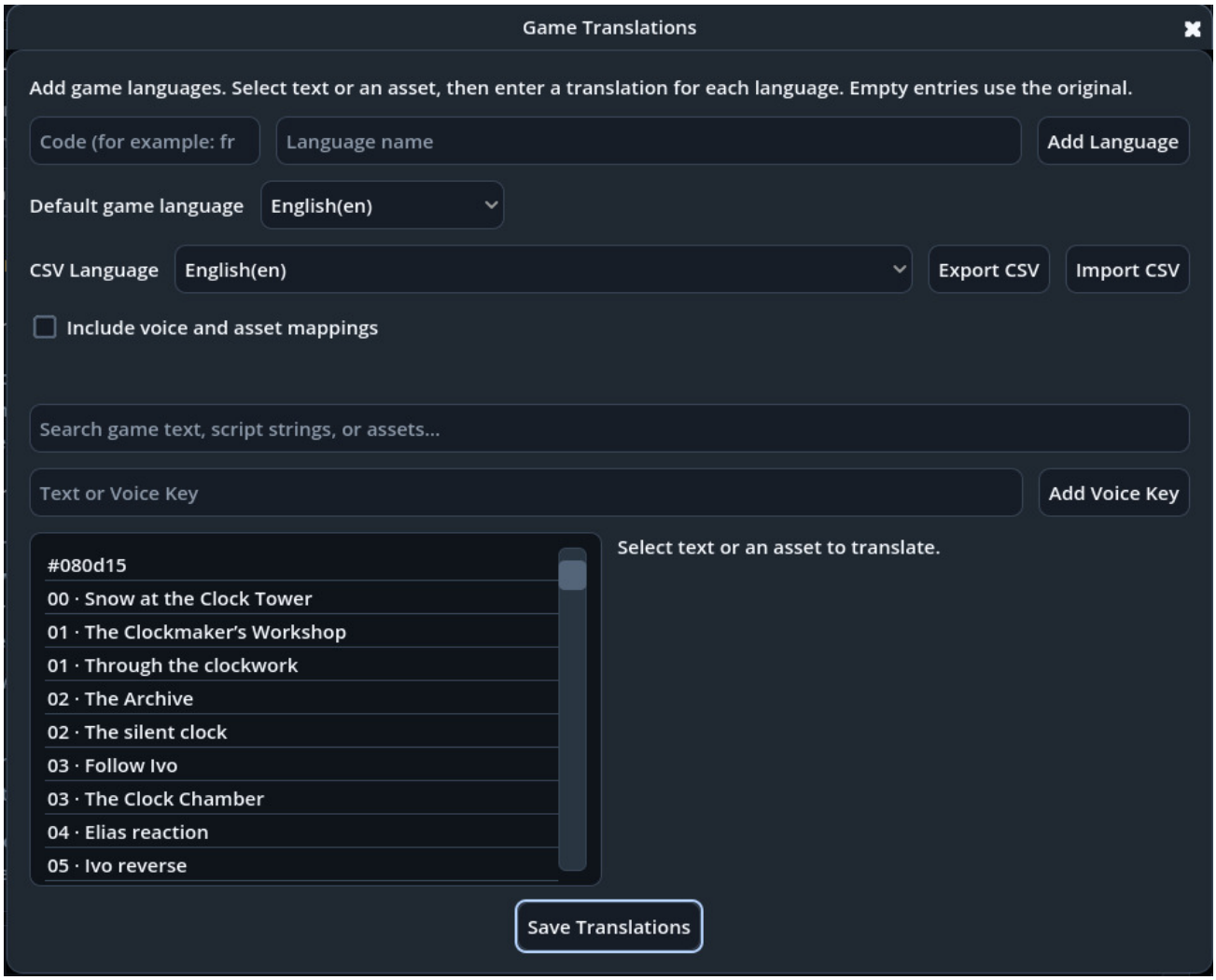

Figure 24. The translation window. Keep one language's work separate from another and review the import result before saving.

Use UTF-8 CSV and preserve quoting around commas, quotes and line breaks. Keep placeholders such as {score}, @PlayerName or other game substitutions intact. Give collaborators separate files or agreed sections so one import does not silently replace another person's work. Save a project checkpoint before importing.

A translated voice recording needs its own asset and lip-sync timing. Translating subtitles does not turn the English recording into another language. Check longer labels, choice lists and button widths in the running game.

Notes, help and attach points

Use the Notes workspace for project reminders, puzzle plans and things you still need to test. Keep a clear distinction between a note and an action: writing open the archive in Notes does not change the game's locked-door logic. Start Here provides built-in guidance, while small question-mark controls beside fields explain local settings.

Attach Points opens a placement editor for named sockets on an asset, character or object. Choose a point name, position it and bind it to a skeleton bone when it should move with a hand or head. The runtime can align or attach another object's point to it. Contact Radius and contact tolerance describe how close two live points must be; they are not the character's walking collider. Clockmaker uses right_hand_grip on both actors for the handshake.

Project JSON, validation and build

Project JSON is the complete document. It is useful for exact data, inspection and a carefully prepared copy operation. It is also easier to damage than a typed form. Work on a saved copy, keep valid JSON and validate immediately after a large replacement.

The validation window lists issues and their locations. Fix missing IDs and assets before building. Preview with errors can help inspect an incomplete room, but it is not evidence that a release is healthy.

Build Game packages the playable runtime. It is the final step of this book, after a fresh playthrough of both puzzles.

PART TWO

Your first adventure: The Last Signal

16. Start The Last Signal

The game is deliberately small. Pick up an amber power cell, use it on a blue relay, go to the beacon deck, align the receiver and transmit. You will learn the complete chain: visible object, carried item, remembered state, room change and ending.

Keep a reference copy

1. From the welcome screen, choose Start from Template Game.
2. Choose Primitive Prototype. Give the copy the title Last Signal Reference and save it in your practice folder.
3. Run it once. Solve the game, then stop. Look at its two rooms, inventory item, variables and completion graph.
4. Close that copy. Choose New Game, name it My Last Signal and save it as my-last-signal-01.pca3d.

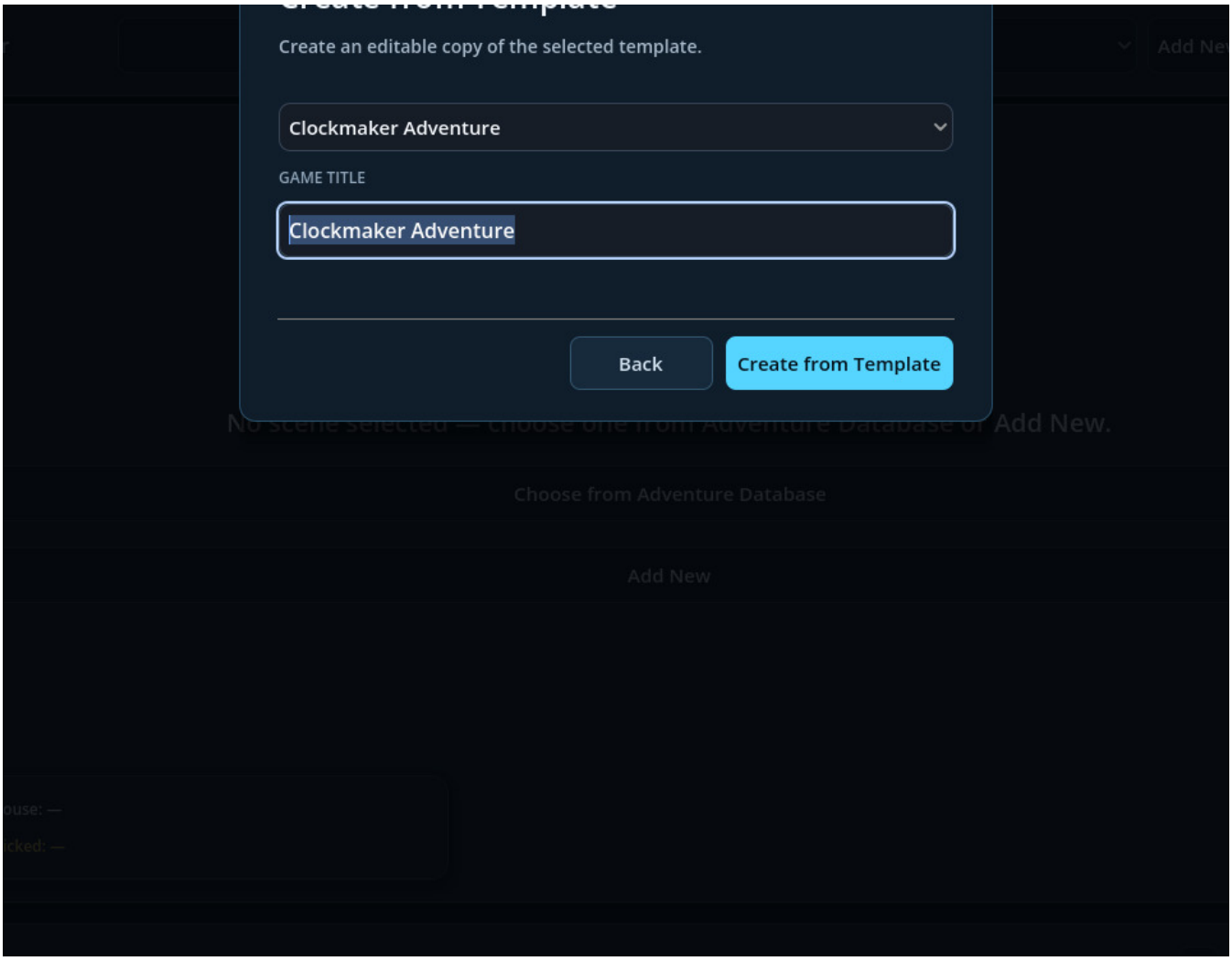

Figure 25. The template picker. Make a reference copy before building your own version.

The following lessons build the game in the blank project. The reference is there to compare settings and to supply the five bundled icons/sound records. It is not a replacement for building the puzzle.

Set up the project

Set the design resolution to 1280 by 720. Your controlled character's placement in the lab determines the starting room. Keep the supplied verbs walk, look, use, talk and take enabled. Use context interaction initially so each hotspot's default verb works with one click. Later, the HUD verb buttons let you try explicit verbs.

Rename Room 1 to 01 - Blockout Lab and set its ID to lab before making scene references. Assign it to 3D and apply. The starter player is a reusable Character and a room placement. To match the demo IDs, create a new reusable Character named Tester with ID tester, mark it playable, place it in lab, select it in Project, and remove the unused starter player definition after checking the deletion prompt.

Give Tester a walking speed of 2.8, running speed 4.5 and collision radius 0.3. Use the stock capsule presentation. In the room, put the character near [0, 0, 2.3]. Add a spawn with ID entry at the same position and facing 180 degrees.

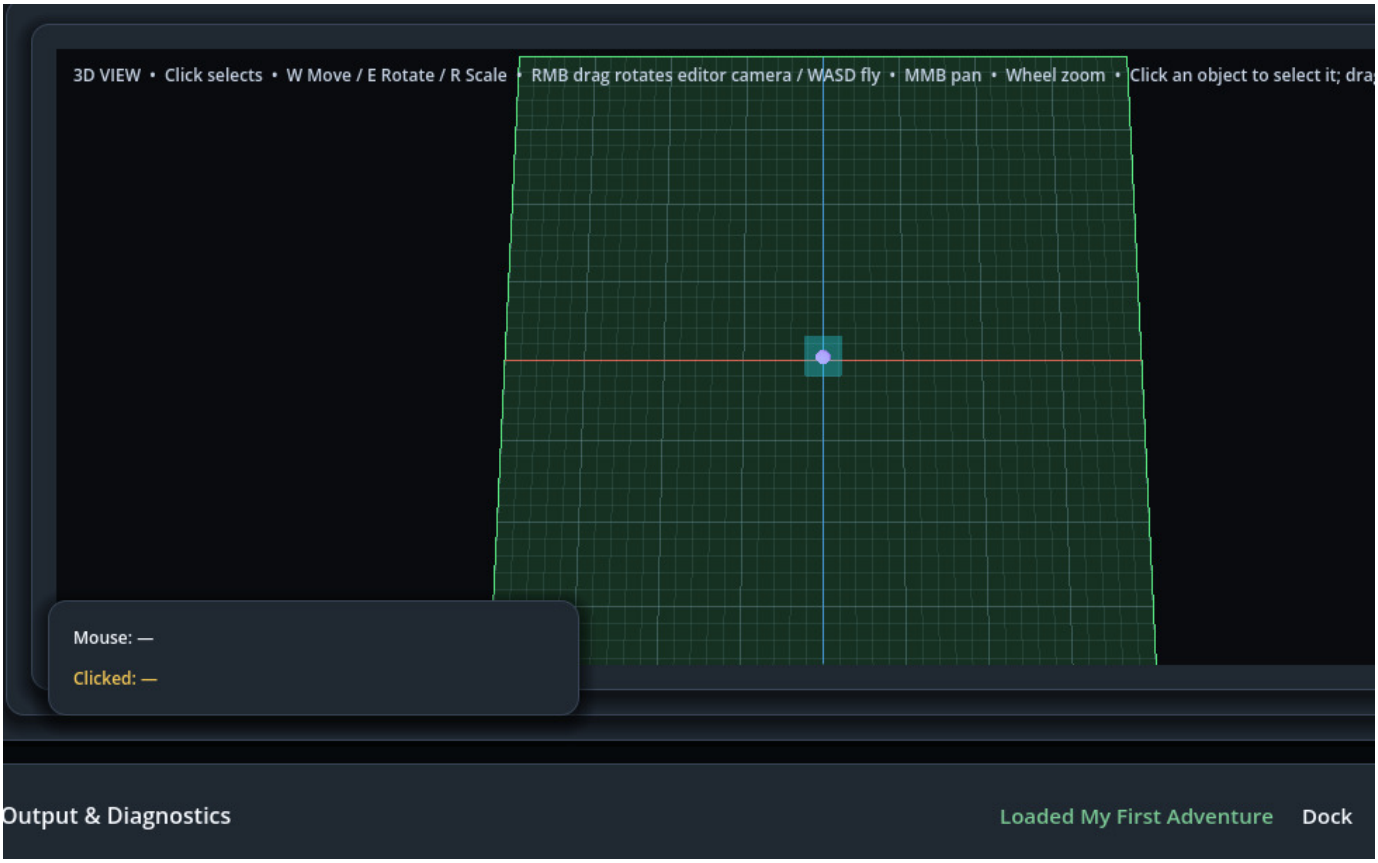

Figure 26. A new room begins with a starter character and navigation information. The floor grid is not a floor mesh.

Create the state variables

Add these in the Adventure Database, apply each record and save. Use the exact IDs because the later recipes refer to them.

ID	Type	Starting value
score	number	0
objective	string	Collect the amber power cell from the supply bench.
cell_taken	boolean	false
powered	boolean	false
aligned	boolean	false
finished	boolean	false

Create Quest signal, named The last signal. Its description is Power the relay, align the receiver and transmit. Enable automatic tracking. Add stage work, title To do, with objective finish containing the same instruction. The room's First Enter actions will start it.

> Check your work. Stop here and reopen your saved project. You should have one 3D room, one reusable Tester, one placed Tester, an entry spawn, six variables and the signal quest. No imported character model is needed.

17. Build the lab and beacon deck

Make a floor and three walls

Use Add to Scene to create a box primitive. Name it Lab Floor, set ID lab_floor, size [12, 0.24, 9] and position [0, -0.12, -0.7]. Its top is at Y=0. Set color #24364a and roughness 0.83. Disable its hotspot.

Add the back and side walls from the placement table in the reference section. Primitive dimensions describe the full box, not half-extents. Set scale to [1, 1, 1] while entering those sizes. Use the supplied table for the ceiling beams, panels and decorative blocks as well.

Do not make every wall a navigation blocker while learning. The authored navigation area already keeps the player inside the room. Add blockers for objects in the walking area where they are needed.

Set navigation and the fixed camera

Set walkable Y to 0 and the agent radius to 0.3. Use bounds X -5.4 to 5.4 and Z -3.8 to 3.2. If the current editor is showing a polygon, make the equivalent rectangle. Keep the initial blocker list empty in this simple demo.

Create or rename the camera to Room view with ID wide. Set position [0, 7.4, 12.7], target [0, 1, -1.4], FOV 46, near clip 0.05 and far clip 100. Make wide the room's default camera. Test the room before adding puzzle logic.

Add the workbench and props

The bench uses procedural geometry, and the power cell and relay are simple primitives. Their exact transforms are in the Primitive placement tables. Create the interactive objects with these IDs now: power_cell and relay. Leave their actions empty until the next lesson.

For power_cell, use an amber box at [-3, 1.5, -1.5], size [0.55, 0.6, 0.55]. For relay, use a blue box at [2.5, 0.9, -2.4], size [1.4, 1.8, 1.2]. Enable both hotspots. The cell's default verb is Take; the relay's is Use.

Make the second room

Create a second 3D Scene, ID roof, name 02 - Beacon Deck. It uses the same floor dimensions, navigation and wide camera as the lab. You may duplicate the room as a starting point, but remove its lab-only objects and give copied scenery unique IDs. Do not leave a second power cell or relay accidentally hidden in the copy.

Add dish at [2, 1.2, -1.5], size [1.8, 0.22, 1.2]. Add transmit at [0, 0.65, 0.6], size [0.8, 1.3, 0.8]. Add back_to_lab at [-4, 1, -2], size [1, 2, 0.35]. Their default verb is Use. Set an entry spawn clear of the return door.

The reference dish has motion enabled so it visibly searches until aligned. Reproduce its Motion values from the full record recipe, or leave it still for the first functional test. The later Set Motion action stops the searching effect.

> Check your work. Test lab, then roof. Click several open floor positions near all four bounds. The capsule should stay on the floor and reach the intended puzzle props. If it cannot, fix navigation now.

18. Pick up and use the power cell

Create the carried item

Add an Inventory item with ID cell and name Power cell. Use Oil Icon(oil_icon) for the icon, matching the supplied demo. It is a placeholder icon; the physical pickup remains the amber cube. Leave stacking disabled and maximum count at 1.

Import or copy the bundled key_icon, wheel_icon, oil_icon, prototype_icon and chime Asset records into your practice game. The asset manifest gives their exact source locations. Importing these files creates asset records; it does not create the inventory item or room props automatically.

To copy a prepared record precisely, save your project, add an Asset of the matching kind, open its Raw JSON view and replace that one Asset record with the corresponding object from the complete recipe. Apply it and validate. Do not paste an entire project into one Asset field. Alternatively, import the source file listed in the manifest and set the same ID. The prepared-copy route keeps metadata that is useful for models and recorded voices.

Give the pickup its Take action

Select power_cell in the lab, open Verb Actions and choose Take. Add this guarded sequence:

1. If Variable - name = cell_taken; equals = true

True

1. Show Message - text = The cell is already in your parts tray.; duration = 3.2

False

1. Set Variable - name = cell_taken; value = true

2. Add Inventory - itemId = cell

3. Set Variable - name = score; value = 20

4. Set Variable - name = objective; value = Use the power cell on the blue relay.

5. Set Object Visible - entityId = power_cell; visible = false

Use Set Variable for the score values here, not Add Score. The demo sets milestone scores to 20, 45, 70 and 100. Those values should not accumulate on repeat clicks.

Its Look action is Show Message: A cube with a job: power the relay. The default Take verb makes a context click pick it up; the explicit Look verb can still inspect it.

Add the cell-on-relay rule

Open Inventory item cell, then Inventory Actions. Add a rule with ID cell_on_relay. Choose the target relay from the current project; the stored qualified target is entity:relay. Add the sequence below.

cell > cell_on_relay

Targets: entity:relay

1. If Variable - name = powered; equals = true

True

1. Show Message - text = The relay is powered.; duration = 3.2

False

1. Set Variable - name = powered; value = true

2. Remove Inventory - itemId = cell

3. Set Variable - name = score; value = 45

4. Set Variable - name = objective; value = Use the relay to reach the beacon deck.

5. Show Message - text = The relay comes online. Click it to reach the beacon deck.; duration = 3.2

If you want the player to form a use-on sentence, allow the Use preposition on the item and set the Use verb's preposition text to on. Test selecting the cell in Inventory and then clicking the relay. The rule belongs to the carried item; a plain Use click on the empty relay is a different event.

Give the relay its own Use action

If powered is true, Change Scene to roof at spawn entry. Otherwise, show The relay needs a power cell. This means using the item powers the relay, then using the relay takes the player to the deck. That two-step behavior is intentional.

Check your work. Pick up the cell twice, use it on the relay, then use the relay again. You should receive only one cell and only one 20-point pickup reward. After powering the relay, the cell leaves Inventory, score reads 45 and the next relay click changes rooms.

19. Align the receiver and finish

The receiver dish

Select dish in roof and open its Use actions. Test aligned first. If it is already true, show that the receiver is already aligned. Otherwise set aligned true, set score to 70, set the objective to press the green transmit switch, stop the dish's motion and show the lock-on message.

1. If Variable - name = aligned; equals = true

True

1. Show Message - text = The receiver is aligned.; duration = 3.2

False

1. Set Variable - name = aligned; value = true

2. Set Variable - name = score; value = 70

3. Set Variable - name = objective; value = Press the green transmit switch to send the signal.

4. Set Motion - entityId = dish; enabled = false

5. Show Message - text = The dish locks onto the relay.; duration = 3.2

The return door

On back_to_lab, add Change Scene to lab, spawn entry. Give the hotspot a useful label. If the destination room or spawn does not appear in the picker, check that it was applied and that you are editing the correct project.

Create send_signal

Add a Visual Script with ID send_signal and name Send Signal. Add an Interact entry set to use. Add these action nodes and connect their flow in order:

Order	Node	Important value
1	Set Variable	finished = true
2	Set Variable	objective = Complete - explore, save, or inspect this project in the editor.
3	Set Variable	score = 100
4	Complete Quest	signal
5	Play Sound	chime

Order	Node	Important value
6	Show GUI	ending

The ending GUI is built in the next lesson. Until it exists, validation correctly reports that missing reference. Save the graph, then select transmit. Its Use action checks aligned: True runs send_signal; False shows Align the receiver dish first.

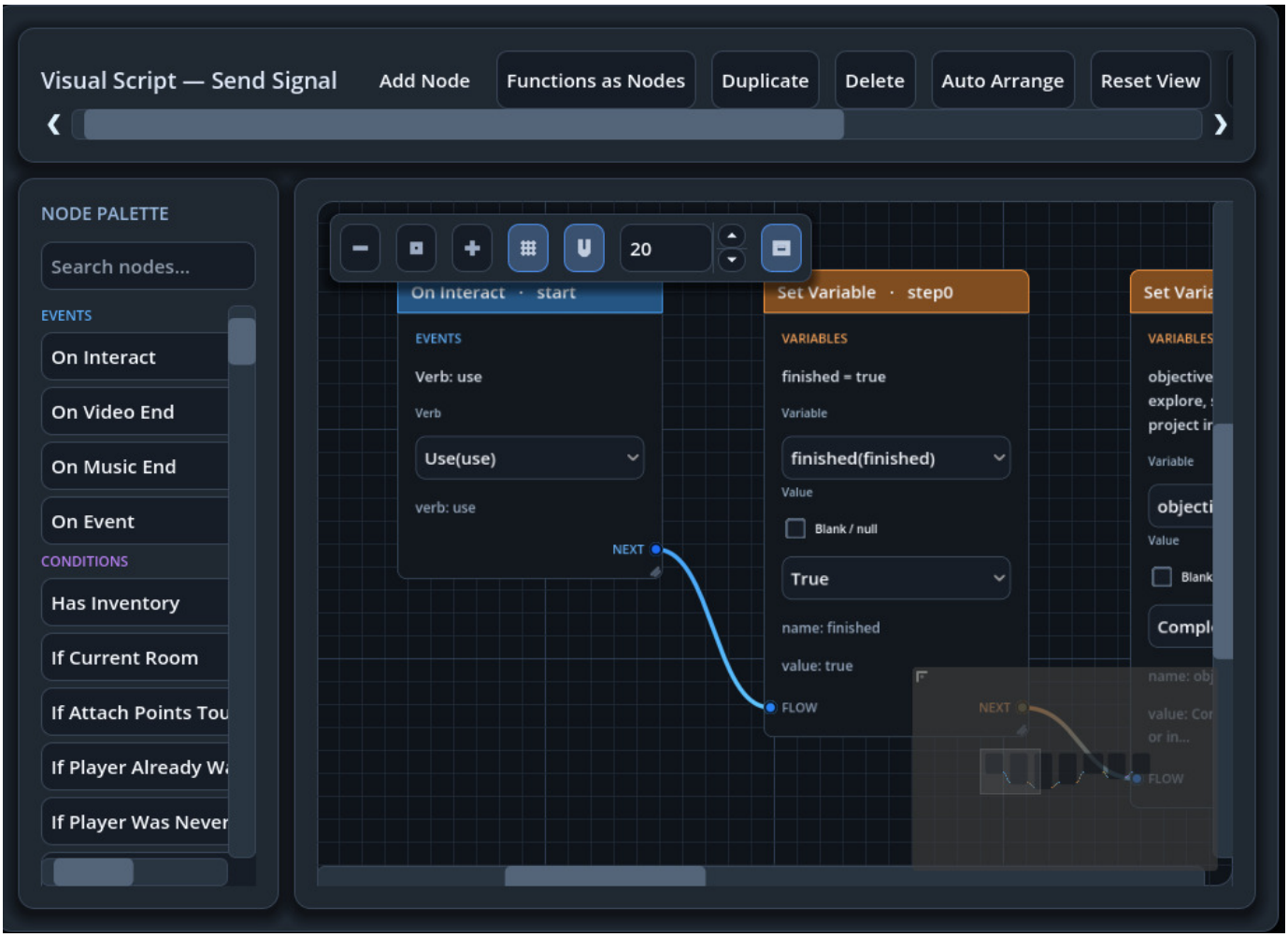

Figure 27. The finished completion chain. One connected sequence is enough for the ending.

First Enter actions

Select the lab room itself. In its First Enter event, Start Quest signal, show the short restore-the-beacon instruction, then show the controls/introduction message. First Enter runs once for that room in a fresh game; ordinary Enter can run on every visit. Do not reset puzzle flags in every Enter event.

> Check your work. Try transmitting before alignment. You should get the reminder and no ending. Align, transmit and check finished, score 100 and the completed quest. Return to the lab and back; the puzzle should remember its state.

20. Build the HUD, inventory and ending

The demo uses four GUIs. Build the HUD first, then Inventory, Journal and the ending. All rectangles in the reference tables use the same 1280 by 720 design resolution.

A shared style

Use background #101a29ed, text #f3e7cf, border #a58b55, border width 1 and radius 5 for the basic panels and buttons. Copy the exact per-element differences from the GUI tables. Keep the HUD visible initially with Z index 10. Keep Inventory, Journal and ending hidden initially.

The HUD

Add a title label, Menu, Journal, Inventory and Hint buttons. Add Look, Use, Talk, Take and Walk controls and a hover label showing {verb} {target}. Add the score label {score} / 100. The reference table gives every rectangle and its Click Action.

For a button that runs a script, choose Call Script in Events or use the supplied legacy script:ID shortcut. For a GUI toggle, use Toggle GUI with the appropriate ID. For a verb button, use Set Verb. Use only one route per button during this lesson.

Inventory

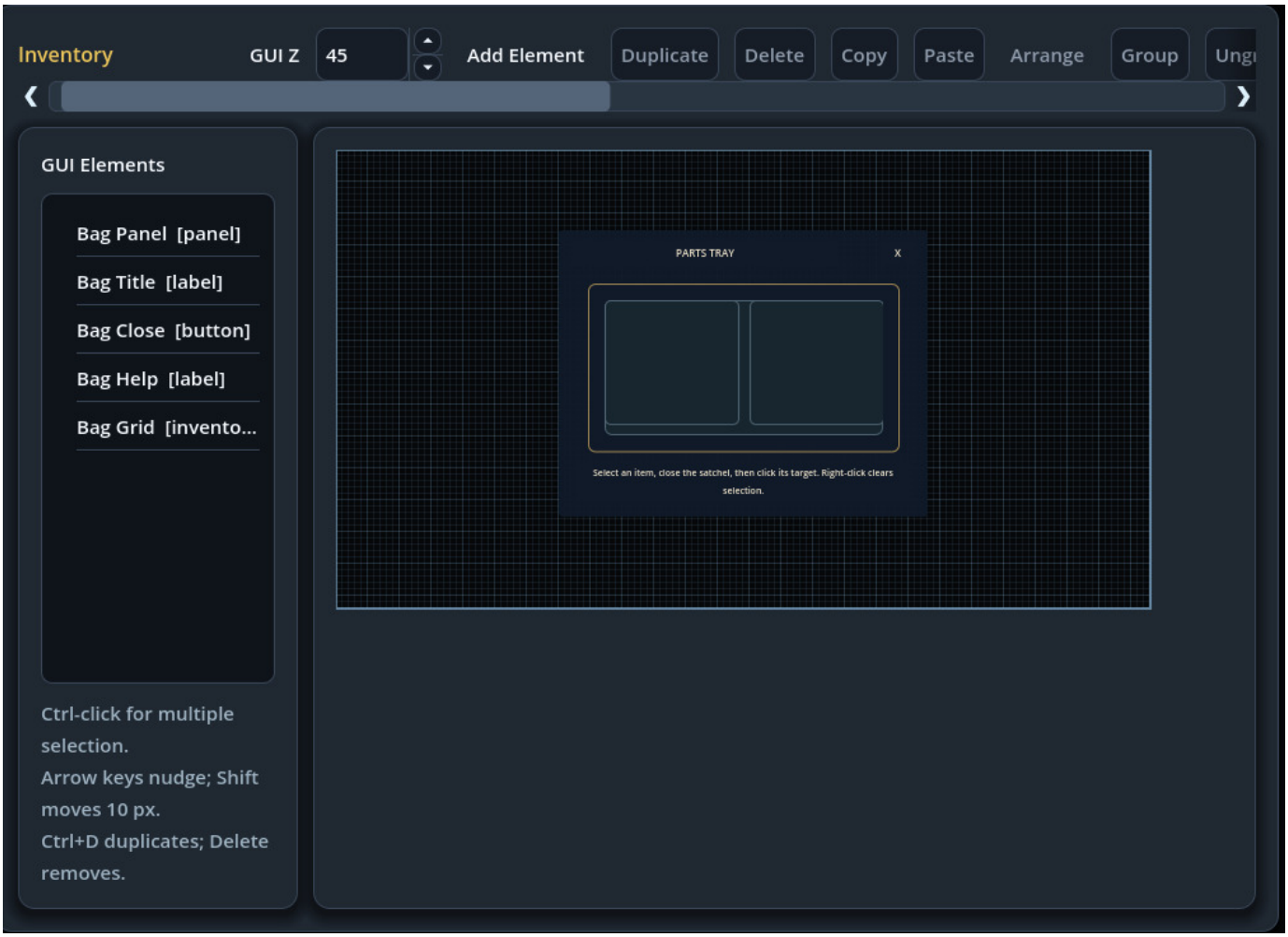

Figure 28. The Primitive Inventory screen. A panel, title, close button, help text and inventory grid are separate elements.

Add the outer panel, PARTS TRAY title, X button, help label and inventoryGrid. Set the grid's columns, rows, slot dimensions and gap from the exact element recipe. A grid renders the carried item definitions; it does not need one hand-built image per possible item.

Wire X to hide inventory. Set Settings > System GUIs > Inventory Screen to inventory. The HUD's Inventory button can now toggle the intended screen.

Journal and ending

Journal uses the title FIELD NOTES, the objective string and a journal list tied to the quest system. Choose journal as the system Journal screen. The ending panel uses SIGNAL RESTORED, a short completion message and a Keep exploring button that hides ending.

Use the GUI tables to match the remaining positions and content. Test each GUI in Preview for appearance, then in the game for events. A modal inventory should block room clicks behind it until closed.

Add the two scripts

Create GDScript menu and hint. The complete short sources are in the script listings. menu opens the pause menu. hint reads the progress flags and chooses one helpful message. Neither script needs to be copied onto every object; the HUD calls each by ID.

menu

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.system_menu.show_pause()
    return true
```

hint

```
func run(_context: Dictionary = {}) -> Variant:
    var hint: String = "Press the green transmit switch."
    if not Runtime.variables.get("cell_taken",false): hint = "Pick up the amber cube on the workbench."
    elif not Runtime.variables.get("powered",false): hint = "Use the power cell from Inventory on the
        blue relay."
    elif not Runtime.variables.get("aligned",false): hint = "Use the blue relay to change rooms, then
        click the receiver dish."
    await ShowMessage({"text":hint,"duration":4.0})
    return true
```

Check your work. Start a fresh game. Open and close Inventory and Journal, use Hint at each puzzle stage, and finish the game. The ending should appear once the receiver is aligned and the switch is used.

21. Finish and test The Last Signal

Use this playthrough before adding your own art. It tests both the happy path and the guards that stop repeated rewards.

Test	Expected result
Click the unpowered relay	A message asks for a power cell; no room change.
Take the amber cell	The pickup hides; Inventory gains one Power cell; score becomes 20.
Use cell on relay	Inventory loses the cell; powered becomes true; score becomes 45.
Use relay	The game enters roof at entry.
Use transmit before dish	A reminder appears; no completion.
Use dish	It stops searching; aligned becomes true; score becomes 70.
Use dish again	An already-aligned message; no extra score.
Use transmit	Quest completes, chime plays, ending appears, score becomes 100.

Test	Expected result
Keep exploring and change rooms	Progress remains intact.
Play / Restart fresh	The cell is back, flags are false and score begins at 0.

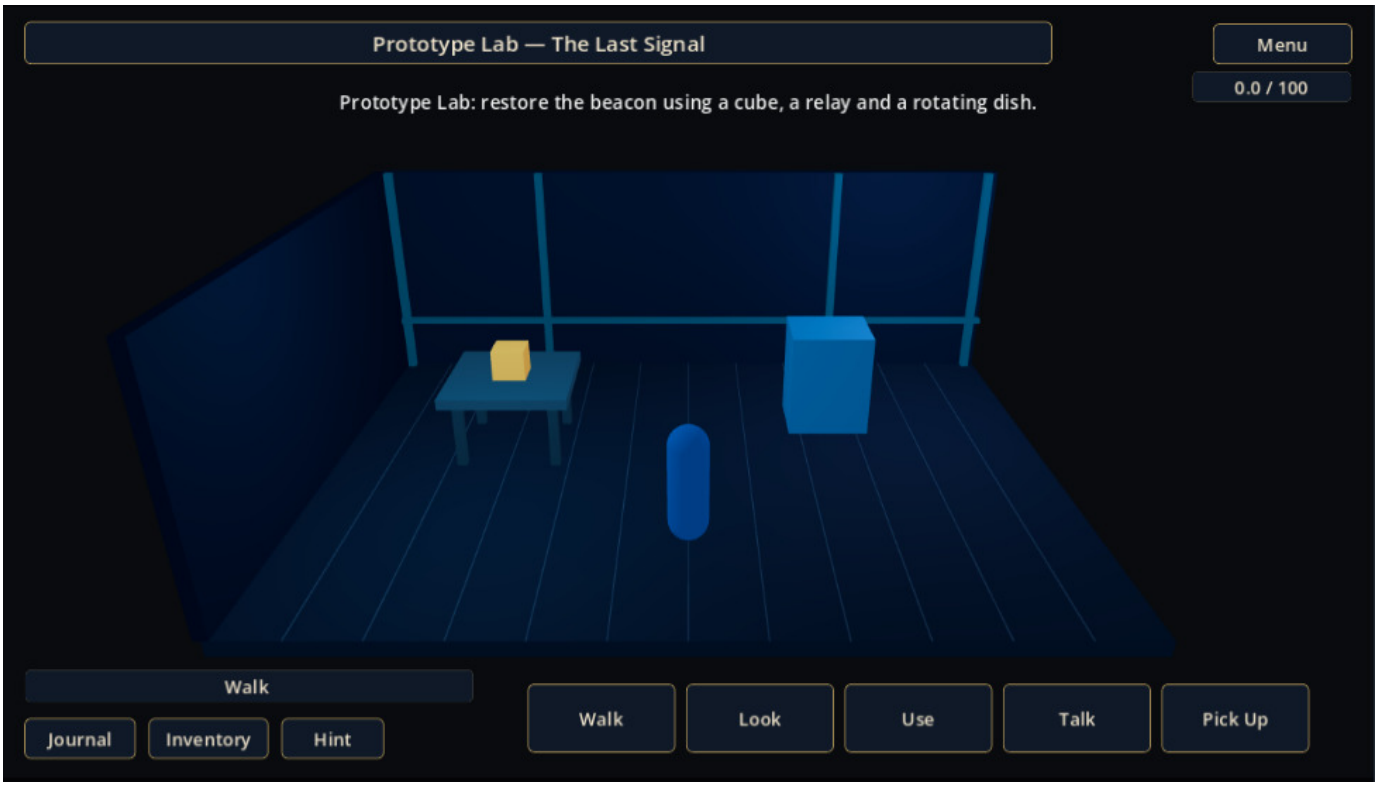

Figure 29. The Last Signal running in the embedded game preview.

Match the supplied demo

The placement and GUI reference sections list the full scene dressing and exact values. Add those finishing details after the functional test passes. This keeps a decorative mistake from looking like a puzzle mistake.

Save a checkpoint called my-last-signal-complete.pca3d. Then make one small change of your own: move the relay, change the title or replace the cell's icon. Test it again. You now have a complete adventure loop, not just a room with clickable objects.

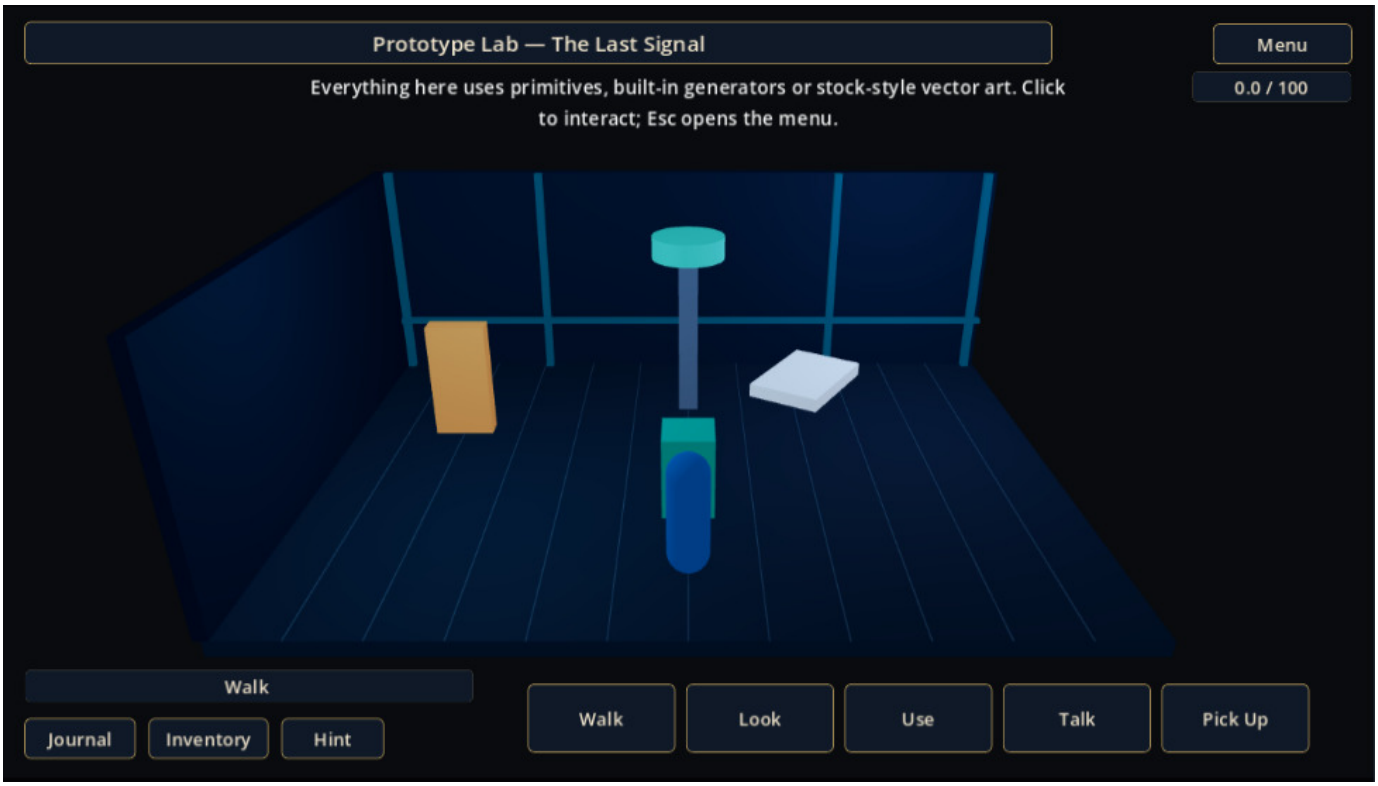

Figure 30. The second room. The same variables and inventory state continue across the scene change.



Figure 31. The Primitive completion screen, shown by the last node in send_signal.

PART THREE

Rebuild The Clockmaker

22. Prepare the Clockmaker project

Clockmaker uses the same basic puzzle structure as The Last Signal. A key opens a new room, spare parts repair a clock and three dial turns finish the game. The difference is presentation: furnished rooms, two animated characters, a voiced opening and cameras that follow the actors.

Use the supplied art without rebuilding the rigs

Create a new game from Clockmaker Adventure and save it as clockmaker-reference.pca3d. Run it once and solve the puzzle. Then Save As clockmaker-practice.pca3d. This gives you the prepared model imports, animation clips, voice metadata and helper scripts while keeping a clean reference.

For the reconstruction, work in the practice copy. Rebuild the three indoor rooms and their puzzle wiring in the order below. You can clear a room's scenery and interactions, then recreate them from the placement and action recipes. Keep the prepared reusable characters and assets. Save a checkpoint before deleting anything. The supplied winter exterior is rebuilt later, after the indoor game works.

If you prefer a completely empty Clockmaker document, the companion practice archive includes an asset-and-character starter. It keeps the original prepared assets, character rigs and scripts, but leaves the room-building and puzzle records for these lessons. Use it with the same PCA 3D installation so the bundled res:// paths resolve.

Learn the two character IDs now

Visible person	Reusable Character ID	Model Asset ID
Professor Ivo	ada	professor
Elias, the clockmaker	elias	ada

These are the actual supplied demo IDs. The model asset called ada is Elias's body; the character called ada is Professor Ivo. Use the name shown beside the ID to avoid picking the wrong record. You can choose tidier IDs in your own game, but changing these mid-tutorial would break the supplied scripts and recordings.



Figure 32. The Workshop in Scene Designer. Most furnishings are ordinary imported assets placed into a primitive room shell.

Plan the puzzle

The workbench gives an archive key. The notebook says to fit the brass wheel, oil the bearing and wind III, I, II. The key opens the archive. The owl holds the wheel, and the oil can supplies bearing oil. In the clock chamber, use the wheel and oil on the master clock, then turn its dial.

Keep that plan beside you. A puzzle becomes much easier to edit when every flag has a clear meaning and every item has one obvious job.

ID	Type	Starting value
score	number	0
objective	string	Ask Elias about the stopped clock. Search the workbench.
key_taken	boolean	false
archive_open	boolean	false
read_note	boolean	false
wheel_taken	boolean	false
oil_taken	boolean	false

ID	Type	Starting value
wheel_fitted	boolean	false
oiled	boolean	false
finished	boolean	false
dial	number	0
intro_skip_all	boolean	false

Create Quest repair, title A minute borrowed. Its work stage has objective finish: Find the parts, repair the clock and ring the afternoon bell. Enable automatic tracking. Keep the original renderer/settings and system GUI records while reconstructing the room logic.

23. Make the three indoor rooms

Create workshop, archive and chamber as 3D rooms. Use the names 01 - The Clockmaker's Workshop, 02 - The Archive and 03 - The Clock Chamber. While building, use Test Scene in workshop so the winter exterior does not replay on every small test. The ordinary full game's first room comes from Ivo's authored placement, rather than a separate startup-scene setting.

Build the common shell

The rooms share a 12-unit-wide floor, back wall, side walls and warm materials. Follow the indoor placement tables for the architecture. The floor is centered at [0, -0.12, -0.7], size [12, 0.24, 9]. Its top is at Y=0.

Use navigation X -5.4 to 5.4, Z -3.8 to 3.2, walkable Y 0 and agent radius 0.3. Add entry at [0, 0, 2.3], facing 180. Create wide at [0, 7.4, 12.7], targeting [0, 1, -1.4], FOV 46. Make it the default gameplay camera.

Furnish the Workshop

Place the work table, bookcases, clocks, lanterns, rug, notebook, doors and window from the manifest. Set the exact transforms in the Inspector after rough placement. Keep scenery hotspots off. Enable only the intended interactive objects.

The workbench is bench at [-2.8, 0.58994, -2.3], scale [1.45, 1.45, 1.45], asset work_table. The notebook is notebook at [-2.5, 1.349782, -2.1], scale [0.28, 0.28, 0.28], asset open_book. The archive door is archive_door at [4.6, 1.454727, -4.9], scale [1.45, 1.45, 1.45], asset door.

Add the bench navigation blocker: X -4.4 to -1.3 and Z -3.1 to -1.5. Check that Ivo can still approach the drawer from the walking side. The blocker's job is movement, not deciding whether Use is allowed.

Furnish Archive and Chamber

In archive, place owl at [-1.5, 1.605234, -1.4], scale 0.32. Put oil_can at [-2.2, 1.5795904, -1.4], size [0.35, 0.55, 0.35]. Add archive_return and tower_door using the door asset and the reference transforms.

In chamber, put master_clock at [0, 1.987614, -3.8], scale 2, asset grandfather_clock. Add chamber_return at [-4.8, 1.454727, -4], scale 1.45. Use the remaining scenery recipes for the complete demo appearance.

Wire the ordinary exits

archive_return uses Change Scene workshop, spawn entry. tower_door uses Change Scene chamber, spawn entry. chamber_return uses Change Scene archive, spawn entry. The locked archive_door gets its conditional behavior in the key lesson.

> Check your work. Test each room locally. Check camera framing, floor height, foot contact and walking clearances. Then travel archive to chamber and back. A door model's visual bounds should not force Ivo to stop halfway across the room.

24. Set up Ivo and Elias

The prepared character records already contain the correct model references and animation mappings. Inspect them now so you know what you are preserving.

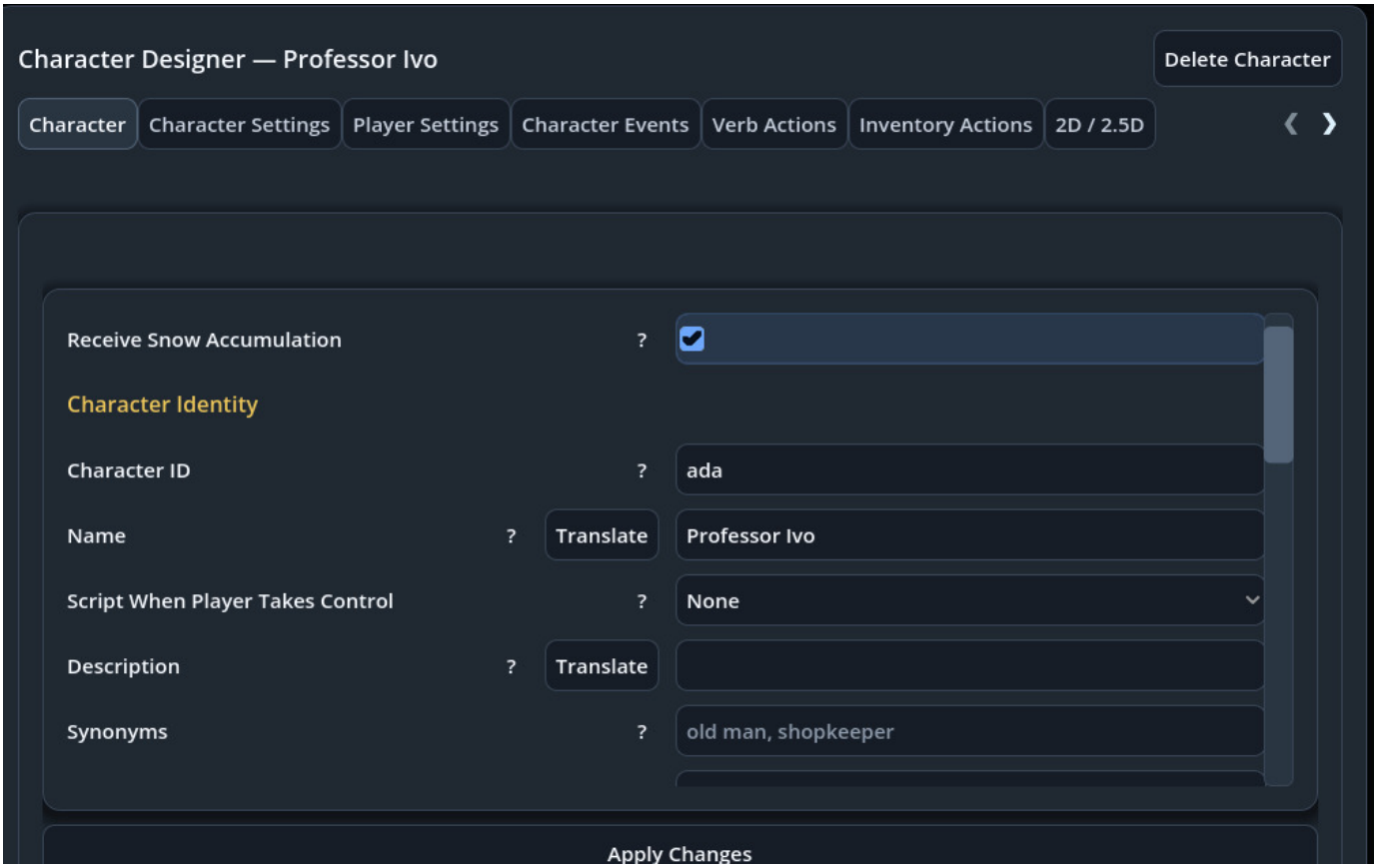

Figure 33. Professor Ivo's reusable definition. His Character ID is ada; his model asset is professor.

Character settings

Both use model scale 1, model height 1.8, collision radius 0.3, walking speed 2.8 and running speed 4.5. Lock Root Motion is on. Facing offset is 0. The footstep asset is wood_step with stride 0.8. Ivo is playable and switchable; Elias is an NPC.

Use the animation-state table to check idle, walk, run, talk, explain, consider, listen, nod, look_around, open_hands, think and handshake. A display label can read Quiet breathing while the imported idle clip has an opaque ID. Select the actual clip listed by the model, not the friendly state label.

State ID	Clip	Speed	Loop
idle	01a142a6-a57c-7551-96a5-f1229d38cb51	1.0	true
walk	Walking	1.0	true
run	Running	1.0	true
talk	PCA_DialogueTalk	1.0	false
explain	PCA_DialogueIndicate	1.0	false
consider	PCA_DialogueConsider	1.0	false

State ID	Clip	Speed	Loop
listen	PCA_DialogueListen	1.0	true
nod	PCA_DialogueNod	1.0	false
look_around	PCA_LookAround	1.0	false
open_hands	PCA_DialogueOpen	1.0	false
think	PCA_Think	1.08	false
handshake	PCA_Handshake	1.0	false

Place them in the Workshop

Place Elias using reusable Character elias. He stands at [2.1, 0, -1.6]. Keep the NPC's body asset inherited from the definition rather than assigning an unrelated model on its room instance.

Ivo uses reusable Character ada. In the complete demo his authored placement is outside the tower, and the opening brings him indoors. Test Scene can place the active player at the Workshop's entry spawn for local testing, so you do not need a second permanent Ivo placement in the Workshop. In the asset starter, the selected unplaced Ivo can start in its first room until you author his outdoor placement later.

Set Elias's Talk verb to Start Dialogue elias. Give him a useful hotspot label. The dialogue's speakerId elias refers to his reusable character identity, not his model file.

Check speech and gestures separately

First preview idle and walk. Then play a talk gesture. Finally test a voiced line and inspect the mouth. If the body moves but the face does not, check the Lip Sync mapping and the recording's prepared track. If the face moves but the body stays idle, check the animation state action; mouth timing does not automatically select every gesture.

The opening alternates gestures so the characters do not repeat one exaggerated movement for every sentence. Use listening, small nods and pauses as well as speaking. A conversation needs someone to listen, too.

25. The key, notebook and locked door

Make the three carried items

Create archive_key, brass_wheel and bearing_oil. Their names and icons are in the table. Keep stacking off and enable the Use preposition so they can be applied to a room target.

ID	Name	Icon
archive_key	Archive key	key_icon
brass_wheel	Brass wheel	wheel_icon
bearing_oil	Bearing oil	oil_icon

The workbench drawer

On bench, default verb Use. Its Use action checks key_taken. If true, show that the drawer is empty. Otherwise set key_taken true, Add Inventory archive_key, Add Score 10, show the discovery message and update objective to try the key on the archive door.

The exact action text and branch order are in the puzzle recipe reference. Use Add Score here: Clockmaker awards separate rewards. The guard prevents the same drawer reward from accumulating.

The repair notebook

On notebook, Use sets read_note true and shows the repair instructions. Include all three facts: replace the missing brass wheel, oil the bearing and wind III - I - II. Let the player reread the notebook. A clue does not need to disappear after the first look.

The archive door

On archive_door, Use checks archive_open. If true, change to archive at entry. Otherwise show the locked-door message. Do not make a plain Use click unlock it without the key.

Open the Inventory definition archive_key. Add a target rule for entity:archive_door. If archive_open is true, change to archive. Otherwise set archive_open true, Add Score 10, show the lock-turning message and change to archive.

The supplied demo keeps the key after unlocking. Do not add Remove Inventory archive_key if you want to match it exactly. In your own game, consuming or keeping a key is a design choice; in a reconstruction, follow the intended behavior.

archive_key > archive_key_on_archive_door

Targets: entity:archive_door

1. If Variable - name = archive_open; equals = true

True

1. Change Scene - sceneId = archive; spawnId = entry

False

1. Set Variable - name = archive_open; value = true

2. Add Score - amount = 10

3. Show Message - text = The lock turns. The archive is open.; duration = 3.2

4. Change Scene - sceneId = archive; spawnId = entry

Check your work. Start fresh in workshop. Try the locked door first. Get the key, open the door and return. The second visit should not require another key or award another 10 points.

26. Find the parts and repair the clock

The owl

Give owl a Use interaction. Show the short owl message, then test wheel_taken. If the wheel was already collected, show the empty result. Otherwise set wheel_taken true, give brass_wheel, add 20 points and update the objective.

Keep the owl itself visible. It is a prop that held an item, not the item that should vanish into Inventory.

The oil can

Give oil_can a Take interaction. Guard it with oil_taken, then set that flag, give bearing_oil, add 10 points and show the pickup message. The original can remains visible in the demo, with the guard preventing another reward. If you later choose to hide it, that is a deliberate variation rather than a required step.

Fit the wheel

On Inventory item brass_wheel, add a target rule for entity:master_clock. If wheel_fitted is true, show the already-fitted message. Otherwise set wheel_fitted true, remove brass_wheel, add 20 points and explain that the bearing needs oil.

Oil the bearing

On bearing_oil, target entity:master_clock. First test wheel_fitted. If false, ask the player to fit the wheel. If true, test oiled. If already true, show the repeat message. Otherwise set oiled true, add 10 points and explain that the clock is ready for winding.

The demo keeps the oil item after use. The nested checks stop repeated points. A visible item remaining in Inventory is not itself proof that the rule failed.

Open the clock dial

On master_clock, Use checks wheel_fitted, then oiled. If both are true, Show GUI dial. Otherwise give the message for the missing step. Look can describe the stopped clock without opening the puzzle.

owl > use

1. Show Message - text = The owl is a paperweight. Beneath it: a brass wheel, carefully wrapped.; duration = 3.2

2. If Variable - name = wheel_taken; equals = true

True

1. Show Message - text = You already have the wheel.; duration = 3.2

False

1. Set Variable - name = wheel_taken; value = true

2. Add Inventory - itemId = brass_wheel

3. Add Score - amount = 20

4. Set Variable - name = objective; value = Collect the bearing oil on the archive floor, then enter the clock chamber.

owl > look

1. Show Message - text = The owl guards the one part Elias could not find.; duration = 3.2

oil_can > take

1. If Variable - name = oil_taken; equals = true

True

1. Show Message - text = You have the oil.; duration = 3.2

False

1. Set Variable - name = oil_taken; value = true

2. Add Inventory - itemId = bearing_oil

3. Add Score - amount = 10

4. Show Message - text = The label reads: "One drop. We are fixing a clock, not frying it."; duration = 3.2

oil_can > look

1. Show Message - text = A small tin of clean bearing oil.; duration = 3.2

master_clock > use

1. If Variable - name = wheel_fitted; equals = true

True

1. If Variable - name = oiled; equals = true

True

1. Show GUI - guiId = dial

False

1. Show Message - text = The wheel fits, but the bearing is dry. Use the oil here.; duration = 3.2

False

1. Show Message - text = A wheel is missing from the mechanism. Select the brass wheel from your inventory and use it here.; duration = 3.2

master_clock > look

1. Show Message - text = Elias's master clock. Its open mechanism awaits a wheel and one drop of oil.; duration = 3.2

brass_wheel > brass_wheel_on_master_clock

Targets: entity:master_clock

1. If Variable - name = wheel_fitted; equals = true

True

1. Show Message - text = The wheel is already fitted.; duration = 3.2

False

1. Set Variable - name = wheel_fitted; value = true

2. Remove Inventory - itemId = brass_wheel

3. Add Score - amount = 20

4. Show Message - text = The teeth mesh perfectly. Now the bearing needs oil.; duration = 3.2

bearing_oil > bearing_oil_on_master_clock

Targets: entity:master_clock

1. If Variable - name = wheel_fitted; equals = true

True

1. If Variable - name = oiled; equals = true

True

1. Show Message - text = One drop was enough.; duration = 3.2

False

1. Set Variable - name = oiled; value = true

2. Add Score - amount = 10

3. Show Message - text = The bearing turns freely. Use the clock to wind it.; duration = 3.2

False

1. Show Message - text = Fit the wheel before oiling its bearing.; duration = 3.2

Check your work. Try oil before fitting the wheel. Try the clock with only the wheel fitted. Fit both parts, then open the dial. Test every failure message once; these messages are how the player learns what the game expects.

27. Build Elias's conversation

The reusable conversation is short and useful. It gives the repair plan and a reason to care about the bell. It is separate from the voiced opening timeline, even though the same characters appear in both.

Create the graph

Add Dialogue elias. Add start as a Line, topics as a Choice, answer0, answer1 and answer2 as Lines, and end as End. Set start as the start node. Use speaker elias on the spoken lines.

start (line)

Setting	Value
speakerId	elias
text	A clock is a promise, Ivo. Ours has missed the afternoon bell.
next	topics

topics (choice)

Setting	Value
options	[{"text": "Where should I begin?", "next": "answer0"}, {"text": "What is wrong with the clock?", "next": "answer1"}, {"text": "Why does the bell matter?", "next": "answer2"}, {"text": "I will get back to work.", "next": "end"}]

end (end)

answer0 (line)

Setting	Value
speakerId	elias
text	My workbench drawer holds the archive key. The spare parts are laid out next door.
next	topics
returnToPrevious	true

answer1 (line)

Setting	Value
speakerId	elias
text	The brass wheel is missing, and the bearing is dry. My notebook gives the winding order.
next	topics
returnToPrevious	true

answer2 (line)

Setting	Value
speakerId	elias
text	My wife used to say the afternoon bell meant it was time to come home. I like to keep that promise.
next	topics
returnToPrevious	true

Connect start to topics. Connect each response to its answer or end. The answer lines return to topics. You can enable Return to Previous so an answer returns to the choice that called it; the supplied records also contain explicit topics destinations. Keep one clear intended return path and test it.

Run it from the NPC

Select Elias's Workshop placement, open Verb Actions and set Talk to Start Dialogue elias. Test the room and click Talk on him. Then Test Dialogue directly from the graph. The first proves the NPC's event wiring; the second isolates the conversation itself.

Add a condition after the plain version works

For practice, duplicate the working file. Add a response such as I found the wheel, conditioned on wheel_taken true. Give Elias a short reply and return to topics. Test before and after finding the wheel. This is how a conversation can acknowledge progress without making a separate dialogue for every stage.

You do not need this extra response to match the demo. Keep your reconstruction checkpoint before making the variation.

28. Make the winding puzzle

The dial is a GUI with three buttons: I, II and III. A number variable, dial, stores how many correct turns have been made. It starts at 0. The required sequence is [3, 1, 2]. A wrong turn resets the counter.

Build the dial GUI

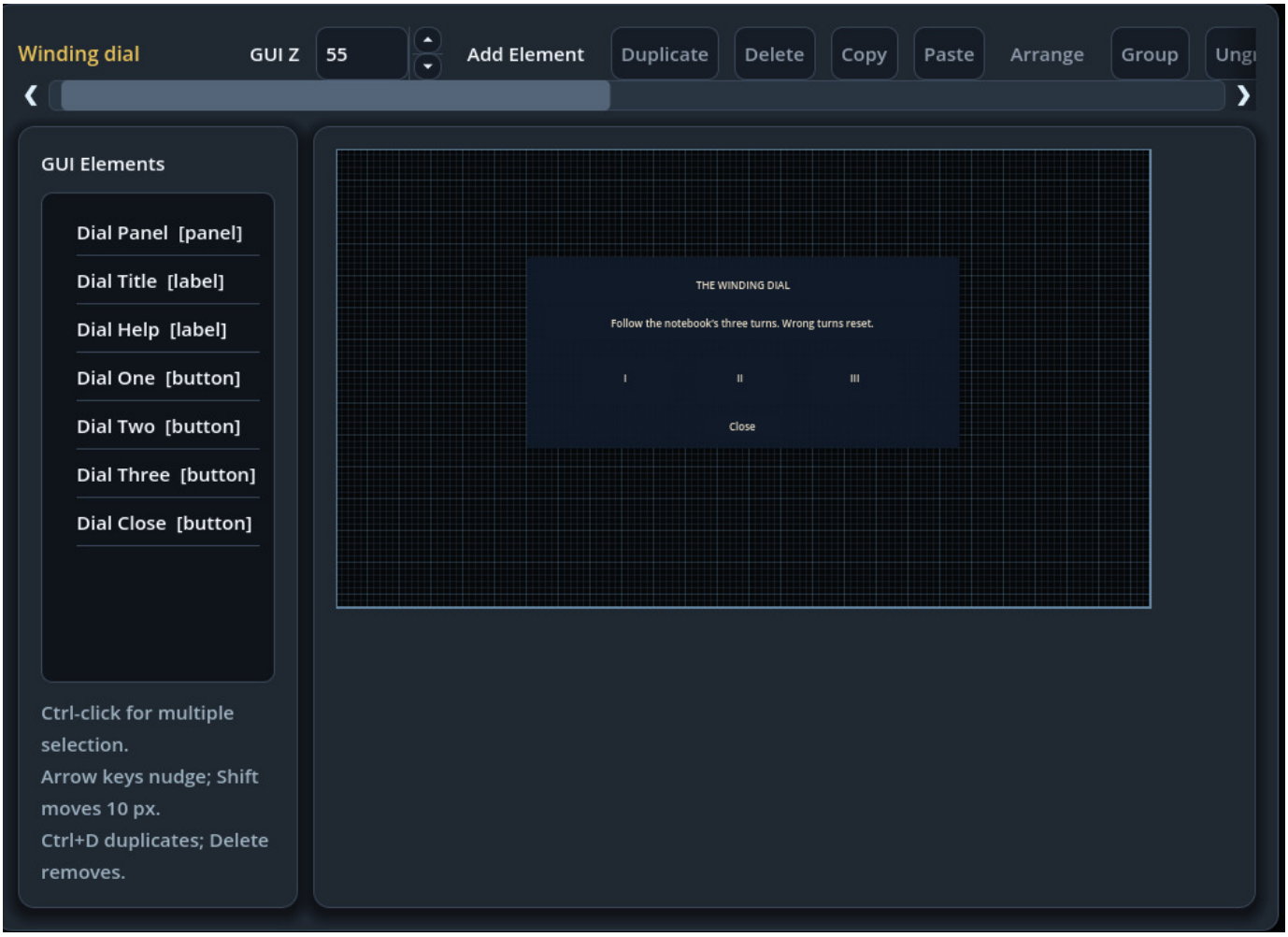

Figure 34. The dial GUI in the designer. The buttons call three small scripts; the dial variable remembers progress.

Create GUI dial, hidden initially. Add its panel, title, help text, three numeral buttons and Close button from the GUI recipe. The numeral buttons call turn_one, turn_two and turn_three. Close hides dial; it does not complete the puzzle.

Create the three scripts

Use the following complete source for turn_one. The other two scripts are identical except for the tested number: use 2 in turn_two and 3 in turn_three. Change only that comparison, not the required sequence.

turn_one

```
func run(_context: Dictionary = {}) -> Variant:
    var sequence: Array = [3, 1, 2]
    var step: int = int(Runtime.variables.get("dial", 0))
    if 1 != sequence[clampi(step, 0, 2)]:
        await SetVariable({"name":"dial", "value":0})
        await ShowMessage({"text":"A soft click. The winding sequence resets.", "duration":1.3})
        return false
    step += 1
    await SetVariable({"name":"dial", "value":step})
    if step == 3:
        await HideGUI({"guiId":"dial"})
        await RunVisualScript({"scriptId":"ring_bell"})
    else:
        await ShowMessage({"text":"A clean click. The next turn is ready.","duration":0.8})
    return true
```

At dial 0, the expected number is 3. At dial 1 it is 1. At dial 2 it is 2. A correct turn increments the counter. At 3, the script hides the dial and runs ring_bell. The clamp prevents an invalid array index if a completed counter is read again.

Build ring_bell

Make a Visual Script with an Interact/use entry. Connect Set Variable finished true, Set Variable objective to the completion text, Set Variable score 100, Complete Quest repair, Play Sound chime and Show GUI ending. This is the same pattern as send_signal in the Primitive game.

> Check your work. Turn I first. The sequence should reset. Then turn III, I, II. The dial should close, the quest should complete and the ending should appear. Restart and try III, II: it must reset rather than accepting two out-of-order correct numbers.

29. Rebuild the Clockmaker screens

Clockmaker's GUI system extends the Primitive version. It has HUD, Inventory, Journal, ending and dial, plus opening_title, cinematic_dialogue and opening_controls for the introduction.

The everyday screens

Use the exact GUI rectangles and content in the reference tables. Keep the shared 1280 by 720 design canvas. The HUD pairs verb images with labels; the click action belongs on the intended interactive element. Avoid overlapping two active buttons that both run the same verb change.

Inventory uses the supplied satchel background, a grid and navigation controls. Its visible slot count is 8. Set the grid's slot size, spacing and title independently. Journal shows the current objective and quest list. The ending screen can be closed so the player can keep exploring.

The introduction screens

opening_title presents the title. cinematic_dialogue contains a label reading {_dialogue_speaker}: {_dialogue_text}. Those runtime values are supplied by the directed speech sequence; they are not ordinary puzzle variables you need to update after every line.

opening_controls contains Skip intro, calling intro_skip. Its visibility is controlled by the introduction scripts. Keep it available while the cutscene locks normal player input.

Style without tying every corner together

Set the inventory's outer frame, inner grid and slots separately. Do the same for any save-game widget you add to a menu. Put normal, hover and pressed images in the Images page for the correct part. Keep the screenshot region distinct from the row background.

The supplied GUI transitions can make Inventory and Journal open or close with visible effects. Use cut while debugging events, then restore the intended transition. A long close effect can make an otherwise correct hide action look delayed.

> Check your work. In GUI Preview, test pointer images and control values. In the game, test toggles, scripts, modal input and the dial. Confirm that Skip intro is accessible during the opening but does not cover the ordinary HUD afterward.

30. Build the camera set

The Workshop contains a wide gameplay view and eleven directed shots. They include clock and notebook details, Ivo and Elias reverses, a moving entrance shot, a pair shot and a handshake shot. Reuse the exact camera records in the camera reference while learning; then change one composition at a time.

Start with the wide view

Keep wide as the room default. Confirm ordinary play is readable from it. Then create the directed cameras under Cameras and enter Position & View values. The names include numbers to make the intended shot order easier to scan; their IDs are what the timeline uses.

00 · Snow at the Clock Tower

Snowy tower establishing shot (exterior_wide)

Setting	Value
id	exterior_wide
name	Snowy tower establishing shot
type	perspective
position	[12, 7.0, 29]
target	[0, 5.8, 0]
fov	46
near	0.1
far	250
dynamicFollow	{"overrideMode": "disabled"}
depthOfField	{"enabled": true, "amount": 0.12, "nearEnabled": true, "nearDistance": 3.0, "nearTransition": 3.0, "farEnabled": true, "farDistance": 43.0, "farTransition": 25.0}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 0, "respectBounds": false}

Tower · slow establishing approach (exterior_tower)

Setting	Value
id	exterior_tower
name	Tower · slow establishing approach
type	perspective
position	[2.8, 5.5, 19]
target	[0, 5.9, 0]
fov	43
near	0.1
far	250
dynamicFollow	{"overrideMode": "disabled"}
depthOfField	{"enabled": true, "amount": 0.12, "nearEnabled": true, "nearDistance": 3.0, "nearTransition": 3.0, "farEnabled": true, "farDistance": 43.0, "farTransition": 25.0}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 12.2, "respectBounds": false, "ease": "sine"}

01 · The Clockmaker's Workshop

Room view (wide)

Setting	Value
id	wide
name	Room view
type	perspective
position	[0, 7.4, 12.7]
target	[0, 1.0, -1.4]
fov	46
near	0.05
far	100

01 · Through the clockwork (intro_establish)

Setting	Value
id	intro_establish
name	01 · Through the clockwork
type	perspective
position	[3.6, 2.6, 6.2]
target	[0, 1, -2]
fov	49
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 0.75, "ease": "sine", "respectBounds": false}

02 · The silent clock (intro_clock)

Setting	Value
id	intro_clock
name	02 · The silent clock
type	perspective
position	[-0.1, 2.25, -1.8]
target	[-1.5, 2.05, -4.8]
fov	36
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 0.75, "ease": "sine", "respectBounds": false}

03 · Follow Ivo (intro_follow)

Setting	Value
id	intro_follow
name	03 · Follow Ivo
type	perspective
position	[2.4, 1.9, 5.8]
target	[0, 1.2, 2.3]
fov	43
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "path", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "ada", "lookAtId": "ada", "targetOffset": [0, 1.25, 0], "damping": 0.25, "pathPoints": [[2.4, 1.9, 5.8], [2.1, 1.85, 4.3], [1.8, 1.8, 2.8]]}

04 · Elias reaction (intro_elias)

Setting	Value
id	intro_elias
name	04 · Elias reaction
type	perspective
position	[-0.6, 1.8, 2.2]
target	[0.8, 1.2, -0.8]
fov	41
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "follow", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "elias", "targetOffset": [0, 1.3, 0], "damping": 0.25}

05 · Ivo reverse (intro_ivo)

Setting	Value
id	intro_ivo
name	05 · Ivo reverse
type	perspective
position	[1.7, 1.65, -0.45]
target	[-0.4, 1.28, 0.3]
fov	38
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "follow", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "ada", "targetOffset": [0, 1.3, 0], "damping": 0.25}

06 · The repair notebook (intro_notebook)

Setting	Value
id	intro_notebook
name	06 · The repair notebook
type	perspective
position	[-1.2, 2.2, -0.1]
target	[-2.6, 1.45, -2.3]
fov	34
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 0.75, "ease": "sine", "respectBounds": false}

07 · A minute borrowed (intro_pair)

Setting	Value
id	intro_pair
name	07 · A minute borrowed
type	perspective
position	[2.7, 1.9, 2.3]
target	[0.1, 1.15, -1.5]
fov	48
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "group", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "ada", "targetIds": "elias", "targetOffset": [0, 1.15, 0], "autoFrame": true, "minDistance": 3.3, "maxDistance": 6.5, "framingPadding": 1.22, "damping": 0.3}

08 · Into the archive (intro_door)

Setting	Value
id	intro_door
name	08 · Into the archive
type	perspective
position	[1.7, 1.7, -0.6]
target	[4.6, 1.7, -4.8]
fov	44
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "fixed", "blendDuration": 0.75, "ease": "sine", "respectBounds": false}

09 · Elias remembers (intro_elias_detail)

Setting	Value
id	intro_elias_detail
name	09 · Elias remembers
type	perspective
position	[0, 1.63, -1.2]
target	[0.8, 1.44, -2.6]
fov	35
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "follow", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "elias", "targetOffset": [0, 1.3, 0], "damping": 0.25}

10 · Ivo makes a plan (intro_ivo_detail)

Setting	Value
id	intro_ivo_detail
name	10 · Ivo makes a plan
type	perspective
position	[0.15, 1.68, -3.5]
target	[-0.4, 1.23, -1.6]
fov	42
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "follow", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "ada", "targetOffset": [0, 1.3, 0], "damping": 0.25}

11 · A silent agreement (intro_handshake)

Setting	Value
id	intro_handshake
name	11 · A silent agreement
type	perspective
position	[1.8, 1.6, 0.2]
target	[0.2, 1.03, -2.1]
fov	39
near	0.05
far	80
dynamicFollow	{"overrideMode": "disabled"}
advancedCamera	{"enabled": true, "mode": "group", "blendDuration": 0.75, "ease": "sine", "respectBounds": false, "targetId": "ada", "targetIds": "elias", "targetOffset": [0, 1.0, 0], "autoFrame": true, "minDistance": 3.3, "maxDistance": 6.5, "framingPadding": 1.22, "damping": 0.3}

02 · The Archive

Room view (wide)

Setting	Value
id	wide
name	Room view
type	perspective
position	[0, 7.4, 12.7]
target	[0, 1.0, -1.4]
fov	46
near	0.05
far	100

03 · The Clock Chamber

Room view (wide)

Setting	Value
id	wide
name	Room view
type	perspective
position	[0, 7.4, 12.7]
target	[0, 1.0, -1.4]
fov	46
near	0.05
far	100

Enable tracking deliberately

Open Advanced Camera for the follow and group shots. Enable it, choose the behavior and set the target. For a two-person shot, include both ada and elias. Use the supplied target offsets and framing limits. Test while the actors move; a still preview can hide an incorrect target.

intro_follow uses a three-point route and a look target during Ivo's entrance. The speaking and reaction shots follow the relevant actor with damping. The pair shot keeps both people in frame. The handshake shot needs enough room to show their hands and faces.

Blend between shots

Use Activate Camera with the camera ID and blend seconds. Zero cuts; a positive value blends. In the opening timeline use Wait false so camera motion can overlap speech and character movement. In a sequential test script, Wait true makes it easier to inspect one finished blend before the next.

Keep the shot direction coherent. If the actors swap left and right between every cut, the conversation becomes harder to follow. Use a wide shot to reestablish the room before making a large change of angle.

Check your work. Activate each camera in the Workshop and move the actors. Confirm the camera follows the intended person, does not clip through the wall and returns to wide after the cutscene. Check group framing with the actors both close together and farther apart.

31. Rebuild the directed indoor opening

The opening is 49.8 seconds long and contains 72 scheduled events. It is more than a camera pan: the actors move, look at each other, speak, gesture, pause and finally shake hands. Build it in passes so you can tell which part caused a problem.

Pass 1: settings and camera cuts

Create Cutscene opening. Set duration 49.8, Skippable on, Restore Camera on, Lock Player Input on, Hide HUD on and Hide Mouse Pointer on. Add only the camera events from the timeline reference first. Test the shots without speech, then check the ending returns to ordinary play.

Pass 2: positions and movement

Add the character placement, facing, movement and animation events. Ivo and Elias must arrive where the close shots expect them. Test the sequence from a fresh Workshop entry rather than moving them manually and starting halfway through the story.

The helper intro_prepare initializes the introduction state. intro_face_pair keeps their orientation sensible. The tracking and orbit helpers support directed shots where an ordinary fixed camera is not enough. The script reference explains each helper and gives the original sources in the companion files.

Pass 3: speech and audio

Add voice events and their subtitle/speaker values in the order listed. Use the prepared voice assets and lip-sync metadata from the starter. Keep events nonblocking where the timeline schedules the next action independently. Mix soundscape, voice and footsteps so the words remain clear.

Test one line with its camera and gesture before adding all eleven. If a line finishes late, check the actual recording length and the next scheduled event. Increasing only the subtitle's duration does not stretch the voice recording.

Pass 4: gestures and the handshake

Use explain, consider, listen, nod and open_hands where the schedule requests them. The final handshake helper coordinates both actors and their attach points. Preserve its prepared rig/hand settings while recreating the sequence. Two independent handshake clips started at arbitrary positions will not make the hands meet.

intro_handshake also restores the actors when the sequence is skipped or completed. Do not remove its cleanup simply because the natural finish looks correct once.

Pass 5: skipping and the handoff

intro_skip records a skip request. The handoff helpers restore the gameplay camera, visibility, input and GUI state and move into the playable scene. Test the opening normally, skip near the beginning, and skip during a voiced line. The game must reach the same usable puzzle state in all three cases.

Script ID	Job
menu	Opens the built-in pause/menu screen.
turn_one	Checks numeral I against the winding sequence and advances or resets dial.
turn_two	Checks numeral II against the winding sequence and advances or resets dial.
turn_three	Checks numeral III against the winding sequence and advances or resets dial.
hint	Chooses a hint from the current puzzle flags, then shows it.
intro_prepare	Prepares the directed indoor opening, including initial actor and presentation state.
intro_tracking	Runs the original tracking helper during the opening; stops when its generation/cutscene is no longer active.
intro_skip	Records the request to skip the introduction and asks the active cutscene to stop.
intro_foreground_on	Shows the prepared cinematic foreground overlay.
intro_foreground_off	Hides the cinematic foreground overlay.
intro_reverse_on	Shows reverse-angle architecture needed behind the conversation shots.
intro_reverse_off	Hides the temporary reverse-angle architecture.
exterior_sequence	Runs wind and the exterior cutscene, fades to the Workshop and activates its startup events.
interior_sequence	Starts the indoor opening or takes the skip/handoff path when the introduction was skipped.
intro_handoff	Makes natural completion and skipping leave the same playable state; stops voice/sound, releases attachments, restores actors, HUD and wide camera.
intro_dialogue_orbit	Supports the slow directed orbit during conversation.
intro_face_pair	Keeps Ivo and Elias facing each other during the relevant opening phase.
intro_score_fade	Fades the opening score/soundscape instead of cutting it abruptly.
intro_voice_mix	Sets the opening voice mix for intelligible dialogue.
intro_handshake	Moves the actors to nearby marks, starts compatible gestures, tests hand attach-point contact, briefly links the hands, then detaches safely.

The full event-by-event schedule appears in the timeline reference. Each row supplies time, track, visible duration and the complete action properties. It is intentionally separate from the lesson so you can keep the practical steps readable while still matching the original precisely.

> Check your work. After the opening, Ivo should be controllable, Elias should respond to Talk, the HUD should be visible and no voice or camera helper should continue taking over ordinary play.

32. Rebuild the snowy exterior

The exterior is the largest scene: 206 placed entities, including repeated architecture and effects. Build the large shapes first. Small trim, snow caps and repeated details should come after the camera composition works.

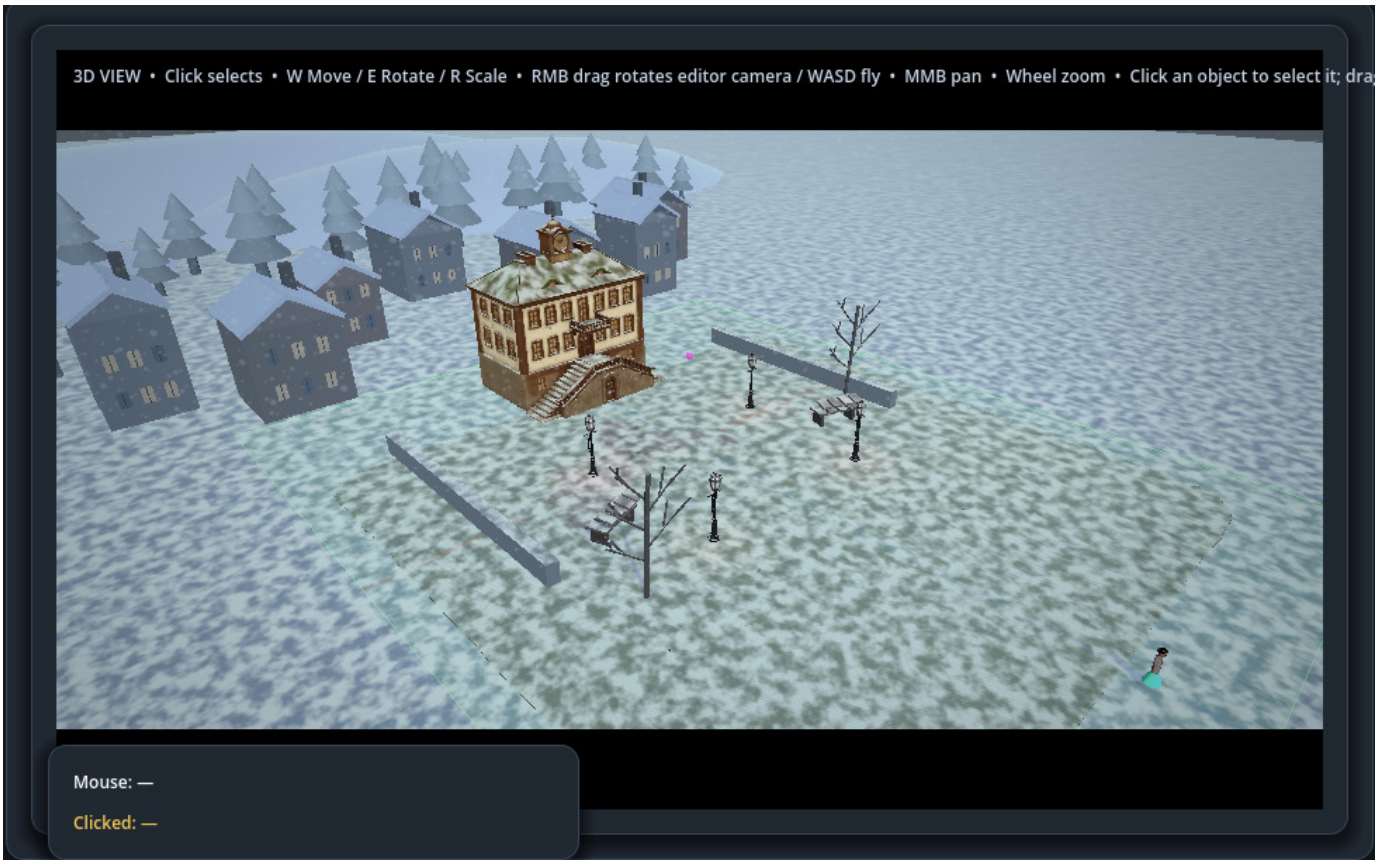

Figure 35. The clock-tower exterior in Scene Designer. Repeated details are easier to manage in folders than as an unstructured list.

Main pieces

Place town_hall using clock_tower_exterior at [0, 6.18, -12], scale [6.3, 6.3, 6.3]. Place cobbled_courtyard using courtyard_ground at [0, -0.1, 7], scale [18, 3, 18]. Add the courtyard lamps and main architectural pieces from the exterior tables.

Use navigation X -20 to 20 and Z -20 to 30, walkable Y 0. Put the arrival point clear of the building. The opening controls the camera and actor movement, so the outdoor area still needs valid floor/navigation data even though it is mostly an establishing scene.

Repeated details and snow

Use duplicate for repeated trim, windows, lamps and snow-covered shapes. Enter the exact transforms in the exterior reference, organized by folder. The companion recipe files contain the complete entity values, including effect parameters, so you do not have to infer particle settings from a screenshot.

Add the snow emitter and check its direction in the game. Snow should fall through the view, not race upward because of a rotated emitter. Check receiving surfaces separately from particles; a falling snow effect and a snow accumulation material are different parts of the scene.

Add winter wind and the broken-clock tick at restrained levels. Keep the start readable even on the smaller laptop screen you intend to support.

The two exterior cameras

Create exterior_wide and exterior_tower. exterior_wide begins at [12, 7, 29], targeting [0, 5.8, 0], FOV 46, near 0.1 and far 250. Copy the second shot and the remaining camera settings from the camera reference.

The exterior_opening cutscene is 13 seconds long. Its central camera action blends toward exterior_tower over 12.2 seconds. This is a timed establishing move between authored cameras, not a path camera left waiting for a moving target.

Setting	Value
id	exterior_opening
name	Snow at the Clock Tower · exterior opening
duration	13.0
skippable	true
restoreCamera	false
lockPlayerInput	true
hideHUD	true
hideMousePointer	true

0s - Wide snowy courtyard

ID exterior_camera; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = exterior_wide; duration = 0; wait = false

0.4s - Slow push toward the clock tower

ID exterior_push; track camera; timeline duration 12.2s.

1. Activate / Blend to Camera - cameraId = exterior_tower; duration = 12.2; wait = false

0.6s - exterior_title_on

ID exterior_title_on; track gui; timeline duration 0.01s.

1. Show GUI - guiId = opening_title

7.6s - exterior_title_off

ID exterior_title_off; track gui; timeline duration 0.01s.

1. Hide GUI - guiId = opening_title

Start the complete sequence

Set Project's Player Character to Professor Ivo(ada) and make sure his authored starting placement is in clocktower_exterior at [0, 0, 26]. The indoor sequence brings him into the Workshop; do not create a second competing startup placement there. Restore the two startup splashes, each with hold 2.8 seconds and fade 0.65. Check the exterior's scene events against the recipe: its sequence leads into the Workshop opening and then ordinary play.

> Check your work. Use Play / Restart from the project start. Check splash order, snow direction, outdoor sound, the tower approach, the indoor opening and the final player handoff. Test skipping as well as waiting through the whole sequence.

33. Play through Clockmaker

Save a checkpoint before this test. Use a fresh start rather than loading an old save with partial puzzle state.

Step	What should happen
Watch or skip the introduction	The game reaches a controllable Ivo in the Workshop.
Talk to Elias	The topic conversation works and each answer returns correctly.
Use the workbench	The key appears in Inventory; the reward happens once.
Read the notebook	The wheel, oil and III-I-II clue are clear.
Try the archive door without the key	It stays locked and explains the problem.
Use key on archive door	The archive opens and the scene changes.
Use owl	The brass wheel is awarded once.

Step	What should happen
Take oil	Bearing oil is awarded once.
Use oil on clock before the wheel	A message asks for the wheel first.
Use wheel, then oil on master clock	Repair flags are set and the dial becomes available.
Turn a wrong numeral	The winding sequence resets.
Turn III, I, II	The quest completes and the ending appears with score 100.
Return to earlier rooms	The collected parts and open door keep their state.



Figure 36. The Workshop in the running game preview.



Figure 37. The Archive. The wheel and oil are separate inventory rewards with independent flags.



Figure 38. The Clock Chamber. The master clock checks the fitted wheel and oil before opening the dial.

Compare against the reference

Open the saved reference and compare one category at a time: room layout, character settings, variables, item rules, conversation, GUIs, cameras and opening timeline. Use the full record recipe if a nested parameter differs.

If a result differs, isolate it. A wrong camera target will not be fixed by moving the inventory button. A missing wheel_fitted flag will not be fixed by changing the clock's material. Keep the record that owns the behavior in view while you work.

Save your complete version as clockmaker-practice-complete.pca3d. You can now replace the story, artwork and puzzle with your own while keeping a working structure to learn from.

PART FOUR

Finish, share and keep a useful reference

34. Build a playable release

An editor project is your working source. A built game is what the player runs. Keep them in separate folders. Do not give players your only editable copy and call it a release.

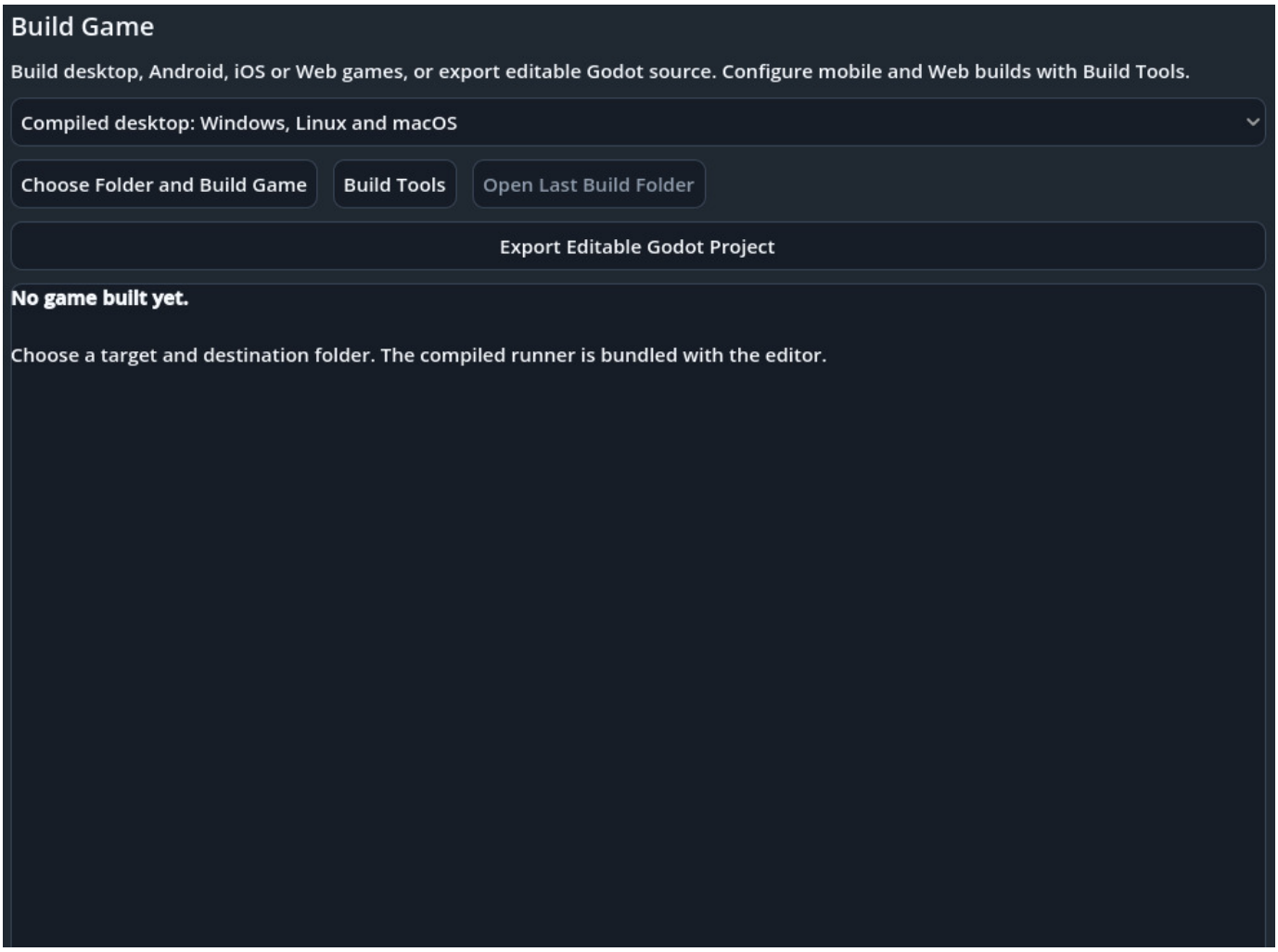

Figure 39. Build Game. Choose a compiled target for a ready-to-run desktop game, or editable source for the Godot export pipeline.

Before building

1. Save the project and run Validate. Fix missing asset, scene, character, GUI and script references.
2. Play from the real beginning. Do not rely only on Test Scene.
3. Check a fresh game, an ordinary save/load, an inventory action, a room change, a conversation and the ending.
4. Check the renderer and resolution on the machine you will use for the final test.
5. Keep the complete source project and original assets as a separate backup.

Choose a desktop target

Players will use...	Select in Build Game	Launch the finished game with...
Windows	Compiled desktop: Windows	The generated .exe, with its companion files.
macOS	Compiled desktop: macOS	The generated .app bundle.
Linux	Compiled desktop: Linux	The generated Linux executable; enable its run permission if needed.
All three desktop systems	Compiled desktop: Windows, Linux and macOS	The appropriate output for each operating system.

Choose the player's target system, which may differ from the system you use for editing. Test each output on its intended operating system before distributing it.

Build and test the game

1. Open Build Game.
2. Choose the target from the table above.
3. Click Choose Folder and Build Game. Choose a new output parent folder you can write to.
4. Wait for the report. Use Open Last Build Folder to inspect the result.
5. Keep the executable and all generated companion data together. Do not zip only the executable if the build produced a separate data directory or pack.
6. Extract a test copy of the result into another folder on the target system and launch it outside Studio. Keep a macOS .app bundle intact. On Linux, enable the executable's run permission if needed.

The complete package includes compiled desktop runners. Editable Godot Source is a different build mode and produces a project for further development or platform export. Mobile and Web build choices use Build Tools and their required SDKs; iOS builds need a Mac or an appropriate configured build worker.

Package and check

Use a single top-level folder with a clear game name and version, then zip that folder. Include a short Read Me with launch instructions and controls, plus required credits. Avoid shipping temporary captures, old builds or caches as part of the player's game.

Check the ZIP by extracting it. Test sound, imported textures, fonts and both puzzle endings. A smaller ZIP can come from compression or removal of unused build material; size alone does not prove that features disappeared. Judge a release by its contents and a fresh playthrough.

35. Fix common problems

Start with the symptom you can reproduce, then check the record that owns the behavior.

Symptom	First checks
I changed a field, but the game ignores it.	Apply the owning form, save, then restart the preview. Check that the Inspector title is the intended record.
The character is a capsule or polygon.	Assign the model on the reusable Character; verify the placed character reference, visibility and asset path.
An animation plays but the body is wrong.	Check the model asset and compatible skeleton/clip mapping. Animation states do not assign a model.
The character cannot reach a door.	Check navigation polygon, blockers, Agent Radius, solid collider size and approach Stop Distance. Event volume size is separate.
The scene ends at a small square despite a huge backdrop.	Expand the navigation/stage bounds and inspect camera bounds. The visible grid is not the only limit.
Use does nothing.	Check hotspot enabled, default verb, interaction mode and the correct Verb Actions owner.
Using an item does nothing.	Check the inventory target rule, exact target ID, required flag and use-on/preposition behavior.
Points can be earned repeatedly.	Put the flag test before the reward and set the flag on the first success.
A GUI looks right but its button does nothing in Preview.	GUI Preview tests visuals and controls. Run the game to execute authored events.
A pressed image stays on.	Check normal/hover/pressed assignments, release behavior and any custom event changing the image permanently.
One corner edit changes several widget parts.	Use the specific frame/grid/slot/screenshot radius fields rather than the outer element's shared default.
A camera exists but is never seen.	Set the room default or activate it by ID; enable advanced behavior when required.
A follow camera does not move in its thumbnail.	Tracking runs in Test Scene; the thumbnail shows the authored pose.
A mirror shows a light but no character.	Glossy materials are not a full mirror view. Check the supported reflection setup, surface orientation and renderer.
Lip sync works on the body but not the portrait.	Enable Both and configure the portrait's independent mapping. Check the actual played recording's track.
Clockmaker is slow or crashes but Primitive does not.	Compare renderer, GPU, shadows, particles, reflections and resolution. Capture the first relevant error and the failing scene.

Symptom	First checks
Use Definitions seems to do nothing.	Add a real function definition, accept it, apply the script and use Call Script Function at the destination.
An imported translation changes the wrong language.	Check selected language, stable keys, CSV encoding and import summary; restore the checkpoint if needed.

Change one thing and retest. If you change collision, camera, scale and navigation together, a working result does not tell you which one mattered.

36. A practical editing routine

Begin with the action the player should be able to perform. Build the smallest room that lets you test it. Add its variable and item rules, then test the unsuccessful case as well as the successful one. Dress the room after that.

For each new puzzle, write down the starting state, required item or condition, success actions and repeat behavior. For each new cutscene, write down how it begins, how it finishes and what skipping must restore. These notes prevent more mistakes than another evening spent adjusting wall trim.

Keep IDs short and stable, names clear and folders ordinary. Architecture, Props, Characters, Effects and Cameras are perfectly good folder names. Let the game be mysterious; the project file should be easy to read.

The reference sections that follow contain the exact demo data behind the lessons. Use them when matching the supplied games, or as examples of how a complete small adventure is organized.

REFERENCE

Exact settings and reconstruction recipes

R1. Primitive room placements

These tables include every placed entity in the two supplied rooms. Position, rotation and scale are separate values. For a primitive, Size is its full local size before scale. For a procedural object, use its Type. Set hotspots only where listed.

P means Position, R means Rotation in degrees and S means Scale. The printed placement tables convert stored room rotations to the degrees used by the transform controls. Complete raw room records store transform.rotation in radians; copy those raw values unchanged instead of converting them a second time. Attach-point action rotations use their own documented degree values.

The compact tables are for placement. The companion recipe files and the PDF's attached source recipes preserve the complete records, including materials, motion, lights and event data. Use those values in Raw JSON when a typed field is not exposed.

01 · Blockout Lab (lab)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -5.4, "maxX": 5.4, "minZ": -3.8, "maxZ": 3.2}, "blocked": []}
Spawns	[{"id": "entry", "position": [0, 0, 2.3], "facing": 180}]
Default camera	wide
Background	#121a28
Environment	{"brightness": 1, "fogEnabled": false, "glowEnabled": false, "skybox": {"enabled": false}}
Transition	{"type": "fade", "duration": 0.35, "color": "#0c1420", "showName": false}

Architecture

Entity / source	Transform	Size, appearance and behavior
Lab Floor lab_floor primitive: box	P [0, -0.12, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.24, 9] material: {"color": "#24364a", "roughness": 0.83}
Lab Back lab_back primitive: box	P [0, 2, -5.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.25] material: {"color": "#20334b", "roughness": 0.83}
Lab Left lab_left primitive: box	P [-6, 2, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.2, 4, 9] material: {"color": "#1c2b41", "roughness": 0.83}
Lab Beam-5.8 lab_beam-5.8 primitive: box	P [-5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Lab Beam-2.9 lab_beam-2.9 primitive: box	P [-2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Lab Beam2.9 lab_beam2.9 primitive: box	P [2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Lab Beam5.8 lab_beam5.8 primitive: box	P [5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Lab Rail lab_rail primitive: box	P [0, 1.0, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.09, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Lab Floorline-5 lab_floorline-5 primitive: box	P [-5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline-4 lab_floorline-4 primitive: box	P [-4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline-3 lab_floorline-3 primitive: box	P [-3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline-2 lab_floorline-2 primitive: box	P [-2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline-1 lab_floorline-1 primitive: box	P [-1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}

Entity / source	Transform	Size, appearance and behavior
Lab Floorline0 lab_floorline0 primitive: box	P [0, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline1 lab_floorline1 primitive: box	P [1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline2 lab_floorline2 primitive: box	P [2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline3 lab_floorline3 primitive: box	P [3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline4 lab_floorline4 primitive: box	P [4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Lab Floorline5 lab_floorline5 primitive: box	P [5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Power Cell power_cell primitive: box	P [-3, 1.5, -1.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.55, 0.6, 0.55] material: {"color": "#ffc35a", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Amber power cell", "defaultVerb": "take", "collisionSize": [1, 1, 1]}
Relay relay primitive: box	P [2.5, 0.9, -2.4] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [1.4, 1.8, 1.2] material: {"color": "#3987b5", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Blue relay", "defaultVerb": "use", "collisionSize": [1, 1, 1]}

Room contents

Entity / source	Transform	Size, appearance and behavior
lab_warm lab_warm light: light	P [-3, 4, 1] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#86d8ff", "energy": 1.7, "range": 12, "shadows": false}
lab_cool lab_cool light: light	P [4, 4, -2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#a6caff", "energy": 1.3, "range": 10, "shadows": false}
Procedural workbench proto_bench procedural: procedural	P [-3, 0, -1.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	generator: square-table material: {"color": "#3c6475"}

Characters

Entity / source	Transform	Size, appearance and behavior
Builder tester character:	P [0, 0, 1.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	Character definition: tester

02 · Beacon Deck (roof)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -5.4, "maxX": 5.4, "minZ": -3.8, "maxZ": 3.2}, "blocked": []}
Spawns	[{"id": "entry", "position": [0, 0, 2.3], "facing": 180}]
Default camera	wide
Background	#121a28
Environment	{"brightness": 1, "fogEnabled": false, "glowEnabled": false, "skybox": {"enabled": false}}
Transition	{"type": "fade", "duration": 0.35, "color": "#0c1420", "showName": false}

Architecture

Entity / source	Transform	Size, appearance and behavior
Roof Floor roof_floor primitive: box	P [0, -0.12, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.24, 9] material: {"color": "#24364a", "roughness": 0.83}
Roof Back roof_back primitive: box	P [0, 2, -5.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.25] material: {"color": "#20334b", "roughness": 0.83}
Roof Left roof_left primitive: box	P [-6, 2, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.2, 4, 9] material: {"color": "#1c2b41", "roughness": 0.83}
Roof Beam-5.8 roof_beam-5.8 primitive: box	P [-5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Roof Beam-2.9 roof_beam-2.9 primitive: box	P [-2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Roof Beam2.9 roof_beam2.9 primitive: box	P [2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Roof Beam5.8 roof_beam5.8 primitive: box	P [5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Roof Rail roof_rail primitive: box	P [0, 1.0, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.09, 0.18] material: {"color": "#2f7285", "roughness": 0.83}
Roof Floorline-5 roof_floorline-5 primitive: box	P [-5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline-4 roof_floorline-4 primitive: box	P [-4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline-3 roof_floorline-3 primitive: box	P [-3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline-2 roof_floorline-2 primitive: box	P [-2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline-1 roof_floorline-1 primitive: box	P [-1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}

Entity / source	Transform	Size, appearance and behavior
Roof Floorline0 roof_floorline0 primitive: box	P [0, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline1 roof_floorline1 primitive: box	P [1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline2 roof_floorline2 primitive: box	P [2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline3 roof_floorline3 primitive: box	P [3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline4 roof_floorline4 primitive: box	P [4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Roof Floorline5 roof_floorline5 primitive: box	P [5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#3b586b", "roughness": 0.83}
Mast mast primitive: box	P [0, 1.4, -3] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.35, 2.8, 0.35] material: {"color": "#607a91", "roughness": 0.83}
Beacon beacon primitive: cylinder	P [0, 3.1, -3] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [1.3, 0.4, 1.3] material: {"color": "#49d3be", "roughness": 0.83}
Dish dish primitive: box	P [2, 1.2, -1.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [1.8, 0.22, 1.2] material: {"color": "#d8e9ed", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} motion: {"enabled": true, "startOn": true, "axis": "y", "speed": 30} Hotspot: {"enabled": true, "label": "Receiver dish", "defaultVerb": "use", "collisionSize": [1, 1, 1]}
Transmit transmit primitive: box	P [0, 0.65, 0.6] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.8, 1.3, 0.8] material: {"color": "#2da995", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Transmit switch", "defaultVerb": "use", "collisionSize": [1, 1, 1]}

Entity / source	Transform	Size, appearance and behavior
Back To Lab back_to_lab primitive: box	P [-4, 1, -2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [1, 2, 0.35] material: {"color": "#e8a55b", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Back to lab", "defaultVerb": "use", "collisionSize": [1, 1, 1]}

Room contents

Entity / source	Transform	Size, appearance and behavior
roof_warm roof_warm light: light	P [-3, 4, 1] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#86d8ff", "energy": 1.7, "range": 12, "shadows": false}
roof_cool roof_cool light: light	P [4, 4, -2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#a6caff", "energy": 1.3, "range": 10, "shadows": false}

R2. Primitive actions and GUI rectangles

The following action recipes are taken from the supplied project. They are shown in execution order. True/False sections belong inside their If action. GUI rectangles are X, Y, Width, Height in design pixels.

01 · Blockout Lab (lab)

Room event: firstEnter

1. Start Quest - questId = signal

2. Show Message - text = Prototype Lab: restore the beacon using a cube, a relay and a rotating dish.; duration = 3.2

3. Show Message - text = Everything here uses primitives, built-in generators or stock-style vector art. Click to interact; Esc opens the menu.; duration = 3.2

power_cell > take

1. If Variable - name = cell_taken; equals = true

True

1. Show Message - text = The cell is already in your parts tray.; duration = 3.2

False

1. Set Variable - name = cell_taken; value = true

2. Add Inventory - itemId = cell

3. Set Variable - name = score; value = 20

4. Set Variable - name = objective; value = Use the power cell on the blue relay.

5. Set Object Visible - entityId = power_cell; visible = false

power_cell > look

1. Show Message - text = A cube with a job: power the relay.; duration = 3.2

relay > use

1. If Variable - name = powered; equals = true

True

1. Change Scene - sceneId = roof; spawnId = entry

False

1. Show Message - text = The relay needs the amber power cell. Select it in Inventory and use it here.; duration = 3.2

relay > look

1. Show Message - text = A plain box stands in for a finished machine.; duration = 3.2

02 · Beacon Deck (roof)

dish > use

1. If Variable - name = aligned; equals = true

True

1. Show Message - text = The receiver is aligned.; duration = 3.2

False

1. Set Variable - name = aligned; value = true

2. Set Variable - name = score; value = 70

3. Set Variable - name = objective; value = Press the green transmit switch to send the signal.

4. Set Motion - entityId = dish; enabled = false

5. Show Message - text = The dish locks onto the relay.; duration = 3.2

dish > look

1. Show Message - text = A cylinder plus continuous rotation makes a receiver.; duration = 3.2

transmit > use

1. If Variable - name = aligned; equals = true

True

1. Run Visual Script / Script - scriptId = send_signal

False

1. Show Message - text = Align the receiver dish first.; duration = 3.2

transmit > look

1. Show Message - text = A box is all you need to test the final interaction.; duration = 3.2

back_to_lab > use

1. Change Scene - sceneId = lab; spawnId = entry

back_to_lab > look

1. Show Message - text = Back to lab; duration = 3.2

Carried inventory items

Power cell (cell)

Setting	Value
id	cell
name	Power cell
description	An amber cube: an inventory item with one purpose.
iconAssetId	oil_icon
stackable	false
maxStack	1

Inventory verb: look

1. Show Message - text = An amber cube: an inventory item with one purpose.; duration = 3.2

cell > cell_on_relay

Targets: entity:relay

1. If Variable - name = powered; equals = true

True

1. Show Message - text = The relay is powered.; duration = 3.2

False

1. Set Variable - name = powered; value = true

2. Remove Inventory - itemId = cell

3. Set Variable - name = score; value = 45

4. Set Variable - name = objective; value = Use the relay to reach the beacon deck.

5. Show Message - text = The relay comes online. Click it to reach the beacon deck.; duration = 3.2

Shared baseline where present: background #101a29ed; text #f3e7cf; border #a58b55; width 1; radius 5. Per-element overrides below take precedence. Text and click action strings are literal project values.

Adventure HUD (hud)

Setting	Value
id	hud
name	Adventure HUD
visible	true
zIndex	10

Element	X, Y, W, H	Content / overrides
topline label	20, 14, 960, 40	Prototype Lab - The Last Signal fontSize: 20
menu button	1130, 16, 130, 38	Menu Click: script:menu
journal_button button	20, 664, 104, 38	Journal Click: gui:journal:toggle
bag_button button	136, 664, 112, 38	Inventory Click: gui:inventory:toggle
hint_button button	260, 664, 96, 38	Hint Click: script:hint
hover label	20, 618, 420, 32	{verb} {target} borderOpacity: 0
score label	1110, 60, 150, 30	{score} / 100 borderOpacity: 0

Element	X, Y, W, H	Content / overrides
verb_walk button	490, 632, 138, 64	Walk Click: verb:walk
verb_look button	638, 632, 138, 64	Look Click: verb:look
verb_use button	786, 632, 138, 64	Use Click: verb:use
verb_talk button	934, 632, 138, 64	Talk Click: verb:talk
verb_take button	1082, 632, 138, 64	Pick Up Click: verb:take

Inventory (inventory)

Setting	Value
id	inventory
name	Inventory
visible	false
zIndex	45

Element	X, Y, W, H	Content / overrides
bag_panel panel	350, 125, 580, 450	(no text)
bag_title label	370, 142, 420, 40	PARTS TRAY
bag_close button	855, 143, 55, 36	X Click: gui:inventory:hide
bag_help label	375, 490, 530, 62	Select an item, close the satchel, then click its target. Right-click clears selection. fontSize: 15

Element	X, Y, W, H	Content / overrides
bag_grid inventoryGrid	395, 210, 490, 265	fontSize: 14 columns: 4 rows: 2 slotWidth: 106 slotHeight: 98 gridGapX: 8 gridGapY: 8 showHeader: false scrollMode: none showNames: true showCount: true gridIconPlacement: top-center iconSize: 44 slotPadding: 4

Journal (journal)

Setting	Value
id	journal
name	Journal
visible	false
zIndex	46

Element	X, Y, W, H	Content / overrides
journal_panel panel	260, 110, 760, 485	(no text)
journal_title label	280, 125, 600, 42	FIELD NOTES
journal_close button	940, 130, 54, 36	X Click: gui:journal:hide
journal_objective label	285, 175, 710, 65	{objective} textAlignment: left-center borderOpacity: 0
journal_list journalList	285, 250, 710, 305	(no text)

Completion (ending)

Setting	Value
id	ending
name	Completion
visible	false
zIndex	70

Element	X, Y, W, H	Content / overrides
end_panel panel	230, 160, 820, 330	(no text)
end_title label	260, 185, 760, 60	SIGNAL RESTORED fontSize: 28
end_text label	280, 260, 720, 130	A complete adventure, built from simple shapes. Open the template to inspect the switches, inventory rules, journal and ending.
end_close button	490, 420, 300, 42	Keep exploring Click: gui:ending:hide

R3. Clockmaker indoor placements

Build the room shell first, then major furniture, interactive props, characters, lights and effects. The table includes scenery as well as puzzle targets so the finished rooms can match the demo.

01 · The Clockmaker's Workshop (workshop)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -5.4, "maxX": 5.4, "minZ": -3.8, "maxZ": 3.2}, "blocked": [{"id": "bench_block", "enabled": true, "minX": -4.4, "maxX": -1.3, "minZ": -3.1, "maxZ": -1.5}]}
Spawns	[{"id": "entry", "position": [0, 0, 2.3], "facing": 180}]
Default camera	wide
Background	#121a28
Environment	{"brightness": 1, "fogEnabled": false, "glowEnabled": false, "skybox": {"enabled": false}, "cinematicMode": "inherit"}
Transition	{"type": "fade", "duration": 0.35, "color": "#0c1420", "showName": false}

Architecture

Entity / source	Transform	Size, appearance and behavior
Workshop Floor workshop_floor primitive: box	P [0, -0.12, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.24, 9] material: {"color": "#4f3d2f", "roughness": 0.83}
Workshop Back workshop_back primitive: box	P [0, 2, -5.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.25] material: {"color": "#61513f", "roughness": 0.83}
Workshop Left workshop_left primitive: box	P [-6, 2, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.2, 4, 9] material: {"color": "#4d4036", "roughness": 0.83}
Workshop Beam-5.8 workshop_beam-5.8 primitive: box	P [-5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Workshop Beam-2.9 workshop_beam-2.9 primitive: box	P [-2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Workshop Beam2.9 workshop_beam2.9 primitive: box	P [2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Workshop Beam5.8 workshop_beam5.8 primitive: box	P [5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Workshop Rail workshop_rail primitive: box	P [0, 1.0, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.09, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Workshop Floorline-5 workshop_floorline-5 primitive: box	P [-5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline-4 workshop_floorline-4 primitive: box	P [-4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline-3 workshop_floorline-3 primitive: box	P [-3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline-2 workshop_floorline-2 primitive: box	P [-2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline-1 workshop_floorline-1 primitive: box	P [-1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}

Entity / source	Transform	Size, appearance and behavior
Workshop Floorline0 workshop_floorline0 primitive: box	P [0, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline1 workshop_floorline1 primitive: box	P [1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline2 workshop_floorline2 primitive: box	P [2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline3 workshop_floorline3 primitive: box	P [3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline4 workshop_floorline4 primitive: box	P [4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Workshop Floorline5 workshop_floorline5 primitive: box	P [5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}

Room contents

Entity / source	Transform	Size, appearance and behavior
workshop_warm workshop_warm light: light	P [-2.8, 3.3, 1.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#ffd5a3", "intensity": 1.15, "distance": 11, "castShadow": true, "shadowBias": 0.05, "normalBias": 0.6, "shadowOpacity": 0.72}
workshop_cool workshop_cool light: light	P [1.2, 3.1, -3.6] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#9ebcfa", "intensity": 0.7, "distance": 9, "castShadow": false}
Uneven clock ticking workshop_clock_tick ambient-sound: ambient-sound	P [-1.5, 1.45, -4.8] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	ambientAudio: {"assetId": "broken_clock_tick", "loop": true, "volume": 0.3, "radius": 24}

Characters

Entity / source	Transform	Size, appearance and behavior
Elias, the clockmaker elias character: ada	P [2.1, 0, -1.6] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} attachPoints: [{"name": "right_hand_grip", "bone": "mixamorig_RightHand", "position": [-0.008721, 0.068404, -0.005185], "rotation": [0, 0, 0], "contactRadius": 0.018, "enabled": true, "space": "model"}] Hotspot: {"enabled": true, "label": "Elias", "defaultVerb": "talk", "collisionSize": [1, 1, 1]} Character definition: elias

Set dressing

Entity / source	Transform	Size, appearance and behavior
Bench bench asset: work_table	P [-2.8, 0.58994, -2.3] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Workbench drawer", "defaultVerb": "use", "collisionSize": [2, 0.9, 1.1]}
Desk Side desk_side asset: desk	P [3.8, 0.694843, -0.8] R degrees [0.0, 0.0, 0.0] S [1.1, 1.1, 1.1]	No extra override.
Mantel Clock mantel_clock asset: grandfather_clock	P [-1.5, 1.44102, -4.8] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	No extra override.
Bookcase Left bookcase_left asset: bookcase	P [-4.2, 1.314152, -4.7] R degrees [0.0, 90.0, 0.0] S [1.65, 1.65, 1.65]	No extra override.
Window Workshop window_workshop asset: window	P [0, 2.5, -5] R degrees [0.0, 0.0, 0.0] S [0.95, 0.95, 0.95]	No extra override.
Rug Workshop rug_workshop asset: rug	P [0, 0.036272, 0] R degrees [0.0, 0.0, 0.0] S [2.2, 2.2, 2.2]	No extra override.
Lantern Bench lantern_bench asset: lantern	P [-3.7, 1.463173, -2.5] R degrees [0.0, 0.0, 0.0] S [0.28, 0.28, 0.28]	No extra override.
Notebook notebook asset: open_book	P [-2.5, 1.349782, -2.1] R degrees [0.0, 0.0, 0.0] S [0.28, 0.28, 0.28]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Repair notebook", "defaultVerb": "use", "collisionSize": [1, 1, 1]}
Archive Door archive_door asset: door	P [4.6, 1.454727, -4.9] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Archive door", "defaultVerb": "use", "collisionSize": [1.1, 2, 0.35]}

Cinematic atmosphere

Entity / source	Transform	Size, appearance and behavior
Window light - drifting dust window_dust particle: particle	P [0.3, 2, -3.5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	particle: {"preset": "snow", "presetVersion": 2, "enabled": true, "amount": 65, "lifetime": 10.0, "preprocess": 4.0, "oneShot": false, "localCoords": false, "drawStyle": "quad", "size": 0.018, "velocityMin": 0.015, "velocityMax": 0.06, "spread": 100, "gravity": [0, -0.005, 0], "direction": [0, -1, 0], "areaExtents": [1.6, 1.3, 1.2], "color": "#edd4a9", "emissionColor": "#e4cc9a", "opacity": 0.28, "imageAssetId": "window_dust_image", "randomness": 0.6, "explosiveness": 0.0, "turbulenceEnabled": false}

Cinematic reverse backdrop

Entity / source	Transform	Size, appearance and behavior
Intro Reverse Wall intro_reverse_wall primitive: box	P [0, 2, 3.75] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.12] material: {"color": "#4d4036", "roughness": 0.9} visible: false
Intro Reverse Trim intro_reverse_trim primitive: box	P [-2.9, 2, 3.65] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.12, 4, 0.15] material: {"color": "#9c815d", "roughness": 0.9} visible: false
Intro Reverse Rail intro_reverse_rail primitive: box	P [0, 1, 3.65] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.08, 0.15] material: {"color": "#9c815d", "roughness": 0.9} visible: false

02 · The Archive (archive)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -5.4, "maxX": 5.4, "minZ": -3.8, "maxZ": 3.2}, "blocked": []}
Spawns	[{"id": "entry", "position": [0, 0, 2.3], "facing": 180}]
Default camera	wide
Background	#121a28
Environment	{"brightness": 1, "fogEnabled": false, "glowEnabled": false, "skybox": {"enabled": false}}
Transition	{"type": "fade", "duration": 0.35, "color": "#0c1420", "showName": false}

Architecture

Entity / source	Transform	Size, appearance and behavior
Archive Floor archive_floor primitive: box	P [0, -0.12, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.24, 9] material: {"color": "#4f3d2f", "roughness": 0.83}
Archive Back archive_back primitive: box	P [0, 2, -5.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.25] material: {"color": "#61513f", "roughness": 0.83}
Archive Left archive_left primitive: box	P [-6, 2, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.2, 4, 9] material: {"color": "#4d4036", "roughness": 0.83}
Archive Beam-5.8 archive_beam-5.8 primitive: box	P [-5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Archive Beam-2.9 archive_beam-2.9 primitive: box	P [-2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Archive Beam2.9 archive_beam2.9 primitive: box	P [2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Archive Beam5.8 archive_beam5.8 primitive: box	P [5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Archive Rail archive_rail primitive: box	P [0, 1.0, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.09, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Archive Floorline-5 archive_floorline-5 primitive: box	P [-5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline-4 archive_floorline-4 primitive: box	P [-4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline-3 archive_floorline-3 primitive: box	P [-3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline-2 archive_floorline-2 primitive: box	P [-2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline-1 archive_floorline-1 primitive: box	P [-1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}

Entity / source	Transform	Size, appearance and behavior
Archive Floorline0 archive_floorline0 primitive: box	P [0, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline1 archive_floorline1 primitive: box	P [1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline2 archive_floorline2 primitive: box	P [2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline3 archive_floorline3 primitive: box	P [3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline4 archive_floorline4 primitive: box	P [4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Archive Floorline5 archive_floorline5 primitive: box	P [5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Oil Can oil_can primitive: cylinder	P [-2.2, 1.57959, -1.4] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.35, 0.55, 0.35] material: {"color": "#c8a149", "roughness": 0.83} pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Bearing oil", "defaultVerb": "take", "collisionSize": [1, 1, 1]}

Room contents

Entity / source	Transform	Size, appearance and behavior
archive_warm archive_warm light: light	P [-3, 4, 1] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#ffcf8b", "energy": 1.7, "range": 12, "shadows": false}
archive_cool archive_cool light: light	P [4, 4, -2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#a6caff", "energy": 1.3, "range": 10, "shadows": false}

Set dressing

Entity / source	Transform	Size, appearance and behavior
Archive Shelves archive_shelves asset: bookcase	P [-3.3, 1.592912, -4.5] R degrees [0.0, 90.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.
Archive Shelves2 archive_shelves2 asset: bookcase	P [0, 1.592912, -4.5] R degrees [0.0, 90.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.
Reading Table reading_table asset: work_table	P [-2.5, 0.650968, -1.4] R degrees [0.0, 0.0, 0.0] S [1.6, 1.6, 1.6]	No extra override.
Typewriter typewriter asset: typewriter	P [-3, 1.460811, -1.5] R degrees [0.0, 0.0, 0.0] S [0.42, 0.42, 0.42]	No extra override.
Brass Lamp brass_lamp asset: lantern	P [3.7, 0.403001, -3] R degrees [0.0, 0.0, 0.0] S [0.4, 0.4, 0.4]	No extra override.
Owl owl asset: owl	P [-1.5, 1.605234, -1.4] R degrees [0.0, 0.0, 0.0] S [0.32, 0.32, 0.32]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Brass owl", "defaultVerb": "use", "collisionSize": [1, 1, 1]}
Archive Return archive_return asset: door	P [-5, 1.454727, -4.5] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Workshop", "defaultVerb": "use", "collisionSize": [1.1, 2, 0.35]}
Tower Door tower_door asset: door	P [5, 1.454727, -4.5] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Clock chamber", "defaultVerb": "use", "collisionSize": [1.1, 2, 0.35]}

03 · The Clock Chamber (chamber)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -5.4, "maxX": 5.4, "minZ": -3.8, "maxZ": 3.2}, "blocked": []}
Spawns	[{"id": "entry", "position": [0, 0, 2.3], "facing": 180}]
Default camera	wide
Background	#121a28
Environment	{"brightness": 1, "fogEnabled": false, "glowEnabled": false, "skybox": {"enabled": false}}
Transition	{"type": "fade", "duration": 0.35, "color": "#0c1420", "showName": false}

Architecture

Entity / source	Transform	Size, appearance and behavior
Chamber Floor chamber_floor primitive: box	P [0, -0.12, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.24, 9] material: {"color": "#4f3d2f", "roughness": 0.83}
Chamber Back chamber_back primitive: box	P [0, 2, -5.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 4, 0.25] material: {"color": "#61513f", "roughness": 0.83}
Chamber Left chamber_left primitive: box	P [-6, 2, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.2, 4, 9] material: {"color": "#4d4036", "roughness": 0.83}
Chamber Beam-5.8 chamber_beam-5.8 primitive: box	P [-5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Chamber Beam-2.9 chamber_beam-2.9 primitive: box	P [-2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Chamber Beam2.9 chamber_beam2.9 primitive: box	P [2.9, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Chamber Beam5.8 chamber_beam5.8 primitive: box	P [5.8, 2, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.14, 4, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Chamber Rail chamber_rail primitive: box	P [0, 1.0, -5] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [12, 0.09, 0.18] material: {"color": "#a08b67", "roughness": 0.83}
Chamber Floorline-5 chamber_floorline-5 primitive: box	P [-5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline-4 chamber_floorline-4 primitive: box	P [-4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline-3 chamber_floorline-3 primitive: box	P [-3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline-2 chamber_floorline-2 primitive: box	P [-2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline-1 chamber_floorline-1 primitive: box	P [-1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}

Entity / source	Transform	Size, appearance and behavior
Chamber Floorline0 chamber_floorline0 primitive: box	P [0, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline1 chamber_floorline1 primitive: box	P [1, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline2 chamber_floorline2 primitive: box	P [2, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline3 chamber_floorline3 primitive: box	P [3, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline4 chamber_floorline4 primitive: box	P [4, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}
Chamber Floorline5 chamber_floorline5 primitive: box	P [5, 0.008, -0.7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.018, 0.012, 9] material: {"color": "#30251f", "roughness": 0.83}

Room contents

Entity / source	Transform	Size, appearance and behavior
chamber_warm chamber_warm light: light	P [-3, 4, 1] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#ffcf8b", "energy": 1.7, "range": 12, "shadows": false}
chamber_cool chamber_cool light: light	P [4, 4, -2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"type": "omni", "color": "#a6caff", "energy": 1.3, "range": 10, "shadows": false}

Set dressing

Entity / source	Transform	Size, appearance and behavior
Master Clock master_clock asset: grandfather_clock	P [0, 1.987614, -3.8] R degrees [0.0, 0.0, 0.0] S [2.0, 2.0, 2.0]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Master clock", "defaultVerb": "use", "collisionSize": [0.8, 2.0, 0.6]}
Chamber Window chamber_window asset: window	P [-3.2, 2.6, -5] R degrees [0.0, 0.0, 0.0] S [1.1, 1.1, 1.1]	No extra override.
Chamber Desk chamber_desk asset: desk	P [3.5, 0.821178, -3.8] R degrees [0.0, 0.0, 0.0] S [1.3, 1.3, 1.3]	No extra override.
Chamber Book chamber_book asset: open_book	P [3.5, 1.84327, -3.5] R degrees [0.0, 0.0, 0.0] S [0.32, 0.32, 0.32]	No extra override.
Chamber Lantern chamber_lantern asset: lantern	P [2.75, 2.05306, -3.7] R degrees [0.0, 0.0, 0.0] S [0.4, 0.4, 0.4]	No extra override.
Chamber Return chamber_return asset: door	P [-4.8, 1.454727, -4] R degrees [0.0, 0.0, 0.0] S [1.45, 1.45, 1.45]	pointerFeedback: {"highlightOnHover": true, "attentionMode": "outline", "outlineColor": "#f4cc78", "outlineThickness": 0.025} Hotspot: {"enabled": true, "label": "Back to archive", "defaultVerb": "use", "collisionSize": [1.1, 2, 0.35]}

R4. Clockmaker puzzle and room events

These are the complete interactive room and inventory recipes. A flag guard is part of the puzzle, not optional cleanup. Keep a reference copy open if you need to compare a nested branch in the action editor.

00 · Snow at the Clock Tower (clocktower_exterior)

Room event: firstEnter

1. Run Visual Script / Script - scriptId = exterior_sequence

01 · The Clockmaker's Workshop (workshop)

Room event: firstEnter

1. Start Quest - questId = repair

2. Run Visual Script / Script - scriptId = interior_sequence

elias > talk

1. Start Dialogue - dialogueId = elias

elias > look

1. Show Message - text = The clockmaker has been waiting for an apprentice with a steady hand.; duration = 3.2

bench > use

1. If Variable - name = key_taken; equals = true

True

1. Show Message - text = The drawer is empty now.; duration = 3.2

False

1. Set Variable - name = key_taken; value = true

2. Add Inventory - itemId = archive_key

3. Add Score - amount = 10

4. Show Message - text = An iron key. The tag reads: ARCHIVE.; duration = 3.2

5. Set Variable - name = objective; value = Use the iron key on the archive door.

bench > look

1. Show Message - text = A drawer with an iron key inside.; duration = 3.2

notebook > use

1. Set Variable - name = read_note; value = true

2. Show Message - text = ELIAS'S NOTE: Fit the brass wheel. Oil the bearing. Wind the dial in order: III - I - II.; duration = 3.2

notebook > look

1. Show Message - text = The repair order is written in a firm, careful hand.; duration = 3.2

archive_door > use

1. If Variable - name = archive_open; equals = true

True

1. Change Scene - sceneId = archive; spawnId = entry

False

1. Show Message - text = Locked. Elias keeps a spare key in his workbench drawer.; duration = 3.2

archive_door > look

1. Show Message - text = An iron lock guards the archive.; duration = 3.2

02 · The Archive (archive)

Room event: firstEnter

1. Show Message - text = Elias has laid out the missing parts. The brass owl watches the reading table.; duration = 3.2

owl > use

1. Show Message - text = The owl is a paperweight. Beneath it: a brass wheel, carefully wrapped.; duration = 3.2

2. If Variable - name = wheel_taken; equals = true

True

1. Show Message - text = You already have the wheel.; duration = 3.2

False

1. Set Variable - name = wheel_taken; value = true

2. Add Inventory - itemId = brass_wheel

3. Add Score - amount = 20

4. Set Variable - name = objective; value = Collect the bearing oil on the archive floor, then enter the clock chamber.

owl > look

1. Show Message - text = The owl guards the one part Elias could not find.; duration = 3.2

oil_can > take

1. If Variable - name = oil_taken; equals = true

True

1. Show Message - text = You have the oil.; duration = 3.2

False

1. Set Variable - name = oil_taken; value = true

2. Add Inventory - itemId = bearing_oil

3. Add Score - amount = 10

4. Show Message - text = The label reads: "One drop. We are fixing a clock, not frying it."; duration = 3.2

oil_can > look

1. Show Message - text = A small tin of clean bearing oil.; duration = 3.2

archive_return > use

1. Change Scene - sceneId = workshop; spawnId = entry

archive_return > look

1. Show Message - text = Workshop; duration = 3.2

tower_door > use

1. Change Scene - sceneId = chamber; spawnId = entry

tower_door > look

1. Show Message - text = Clock chamber; duration = 3.2

03 · The Clock Chamber (chamber)

Room event: firstEnter

1. Set Variable - name = objective; value = Fit the wheel and oil the master clock. Read the workshop notebook for the winding order.

master_clock > use

1. If Variable - name = wheel_fitted; equals = true

True

1. If Variable - name = oiled; equals = true

True

1. Show GUI - guiId = dial

False

1. Show Message - text = The wheel fits, but the bearing is dry. Use the oil here.; duration = 3.2

False

1. Show Message - text = A wheel is missing from the mechanism. Select the brass wheel from your inventory and use it here.; duration = 3.2

master_clock > look

1. Show Message - text = Elias's master clock. Its open mechanism awaits a wheel and one drop of oil.; duration = 3.2

chamber_return > use

1. Change Scene - sceneId = archive; spawnId = entry

chamber_return > look

1. Show Message - text = Back to archive; duration = 3.2

Carried inventory items

Archive key (archive_key)

Setting	Value
id	archive_key
name	Archive key
description	The key to Elias's archive.
iconAssetId	key_icon
stackable	false
maxStack	1
verbPrepositions	{"use": true}

Inventory verb: look

1. Show Message - text = The key to Elias's archive.; duration = 3.2

archive_key > archive_key_on_archive_door

Targets: entity:archive_door

1. If Variable - name = archive_open; equals = true

True

1. Change Scene - sceneId = archive; spawnId = entry

False

1. Set Variable - name = archive_open; value = true

2. Add Score - amount = 10

3. Show Message - text = The lock turns. The archive is open.; duration = 3.2

4. Change Scene - sceneId = archive; spawnId = entry

Brass wheel (brass_wheel)

Setting	Value
id	brass_wheel
name	Brass wheel
description	The missing escapement wheel.
iconAssetId	wheel_icon
stackable	false
maxStack	1
verbPrepositions	{"use": true}

Inventory verb: look

1. Show Message - text = The missing escapement wheel.; duration = 3.2

brass_wheel > brass_wheel_on_master_clock

Targets: entity:master_clock

1. If Variable - name = wheel_fitted; equals = true

True

1. Show Message - text = The wheel is already fitted.; duration = 3.2

False

1. Set Variable - name = wheel_fitted; value = true

2. Remove Inventory - itemId = brass_wheel

3. Add Score - amount = 20

4. Show Message - text = The teeth mesh perfectly. Now the bearing needs oil.; duration = 3.2

Bearing oil (bearing_oil)

Setting	Value
id	bearing_oil
name	Bearing oil
description	One drop is enough.
iconAssetId	oil_icon
stackable	false
maxStack	1
verbPrepositions	{"use": true}

Inventory verb: look

1. Show Message - text = One drop is enough.; duration = 3.2

bearing_oil > bearing_oil_on_master_clock

Targets: entity:master_clock

1. If Variable - name = wheel_fitted; equals = true

True

1. If Variable - name = oiled; equals = true

True

1. Show Message - text = One drop was enough.; duration = 3.2

False

1. Set Variable - name = oiled; value = true

2. Add Score - amount = 10

3. Show Message - text = The bearing turns freely. Use the clock to wind it.; duration = 3.2

False

1. Show Message - text = Fit the wheel before oiling its bearing.; duration = 3.2

R5. Clockmaker GUI rectangles

All rectangles use a 1280 by 720 design canvas. Most elements use top-left anchors. The tables list the actual anchor if it differs. Detailed widget settings follow each GUI where needed.

Shared baseline where present: background #101a29ed; text #f3e7cf; border #a58b55; width 1; radius 5. Per-element overrides below take precedence. Text and click action strings are literal project values.

Adventure HUD (hud)

Setting	Value
id	hud
name	Adventure HUD
visible	true
zIndex	10

Element	X, Y, W, H	Content / overrides
topline label	20, 14, 960, 40	The Clockmaker - A Minute Borrowed fontSize: 20
menu button	1130, 16, 130, 38	Menu Click: script:menu
journal_button button	20, 664, 104, 38	Journal Click: gui:journal:toggle
bag_button button	136, 664, 112, 38	Inventory Click: gui:inventory:toggle
hint_button button	260, 664, 96, 38	Hint Click: script:hint
hover label	20, 618, 420, 32	{verb} {target} borderOpacity: 0
score label	1110, 60, 150, 30	{score} / 100 borderOpacity: 0
verb_walk image	490, 618, 126, 64	Click: verb:walk background: #17202be8 borderColor: #8a7855 assetId: gui_walk contentFit: contain

Element	X, Y, W, H	Content / overrides
verb_look image	638, 618, 126, 64	Click: verb:look background: #17202be8 borderColor: #8a7855 assetId: gui_look contentFit: contain
verb_use image	786, 618, 126, 64	Click: verb:use background: #17202be8 borderColor: #8a7855 assetId: gui_use contentFit: contain
verb_talk image	934, 618, 126, 64	Click: verb:talk background: #17202be8 borderColor: #8a7855 assetId: gui_talk contentFit: contain
verb_take image	1082, 618, 126, 64	Click: verb:take background: #17202be8 borderColor: #8a7855 assetId: gui_take contentFit: contain
verb_walk_label label	490, 684, 126, 25	Walk Click: verb:walk background: #101822d9 borderColor: #8a7855 borderWidth: 0 contentFit: contain
verb_look_label label	638, 684, 126, 25	Look Click: verb:look background: #101822d9 borderColor: #8a7855 borderWidth: 0 contentFit: contain
verb_use_label label	786, 684, 126, 25	Use Click: verb:use background: #101822d9 borderColor: #8a7855 borderWidth: 0 contentFit: contain
verb_talk_label label	934, 684, 126, 25	Talk Click: verb:talk background: #101822d9 borderColor: #8a7855 borderWidth: 0 contentFit: contain

Element	X, Y, W, H	Content / overrides
verb_take_label label	1082, 684, 126, 25	Pick Up Click: verb:take background: #101822d9 borderColor: #8a7855 borderWidth: 0 contentFit: contain

Inventory (inventory)

Setting	Value
id	inventory
name	Inventory
visible	false
zIndex	45

Element	X, Y, W, H	Content / overrides
bag_panel panel	350, 125, 580, 450	background: #101820f5 borderColor: #8f7c58 borderWidth: 2 borderOpacity: 1
bag_title label	370, 142, 420, 40	SATCHEL background: #00000000 borderOpacity: 0
bag_close button	855, 143, 55, 36	X Click: gui:inventory:hide
bag_help label	375, 490, 530, 62	Select an item, close the satchel, then click its target. Right-click clears selection. background: #00000000 fontSize: 15 borderOpacity: 0

Element	X, Y, W, H	Content / overrides
bag_grid inventoryGrid	392, 219, 396, 234	background: #00000000 borderWidth: 0 fontSize: 14 columns: 4 rows: 2 slotWidth: 94 slotHeight: 110 gridGapX: 6 gridGapY: 7 showHeader: false scrollMode: none showNames: true showCount: true gridIconPlacement: top-center iconSize: 42 slotPadding: 8 padding: 0 borderOpacity: 0 gridBackground: #00000000 gridBorderWidth: 0 slotBackground: #293038ec slotBorderWidth: 1 slotBorderOpacity: 1 slotHoverBackground: #c8ab6544 slotSelectedBackground: #d7bb7755 slotClickBackground: #ffffff22 slotBorderColor: #8f7c58
bag_up button	801, 218, 91, 112	▲ Click: inventory:up:bag_grid background: #222a31ee borderColor: #8f7c58 fontSize: 27 borderOpacity: 1 tooltip: Previous items
bag_down button	801, 335, 91, 112	▼ Click: inventory:down:bag_grid background: #222a31ee borderColor: #8f7c58 fontSize: 27 borderOpacity: 1 tooltip: Next items

Journal (journal)

Setting	Value
id	journal
name	Journal
visible	false
zIndex	46

Element	X, Y, W, H	Content / overrides
journal_panel panel	260, 110, 760, 485	(no text)
journal_title label	280, 125, 600, 42	FIELD NOTES
journal_close button	940, 130, 54, 36	X Click: gui:journal:hide
journal_objective label	285, 175, 710, 65	{objective} textAlignment: left-center borderOpacity: 0
journal_list journalList	285, 250, 710, 305	(no text)

Completion (ending)

Setting	Value
id	ending
name	Completion
visible	false
zIndex	70

Element	X, Y, W, H	Content / overrides
end_panel panel	230, 160, 820, 330	(no text)
end_title label	260, 185, 760, 60	A MINUTE WELL SPENT fontSize: 28

Element	X, Y, W, H	Content / overrides
end_text label	280, 260, 720, 130	The village clock is running again. Elias has his afternoon back-and Ivo has earned his place at the workbench.
end_close button	490, 420, 300, 42	Keep exploring Click: gui:ending:hide

Winding dial (dial)

Setting	Value
id	dial
name	Winding dial
visible	false
zIndex	55

Element	X, Y, W, H	Content / overrides
dial_panel panel	300, 170, 680, 300	(no text)
dial_title label	320, 190, 640, 50	THE WINDING DIAL
dial_help label	330, 250, 620, 50	Follow the notebook's three turns. Wrong turns reset.
dial_one button	385, 325, 140, 70	I Click: script:turn_one
dial_two button	565, 325, 140, 70	II Click: script:turn_two
dial_three button	745, 325, 140, 70	III Click: script:turn_three
dial_close button	580, 420, 120, 32	Close Click: gui:dial:hide

Opening · title card (opening_title)

Setting	Value
openTransition	{"type": "fade", "duration": 0.9}
closeTransition	{"type": "fade", "duration": 0.6}
id	opening_title
name	Opening · title card
visible	false

Element	X, Y, W, H	Content / overrides
title label	280, 478, 720, 59	THE CLOCKMAKER background: #080d1555 color: #f6ead6 fontSize: 42 borderWidth: 0
subtitle label	280, 537, 720, 35	A MINUTE BORROWED background: #080d1555 color: #f6ead6 fontSize: 19 borderWidth: 0

Cinematic dialogue · subtitles (cinematic_dialogue)

Setting	Value
id	cinematic_dialogue
name	Cinematic dialogue · subtitles
visible	false

Element	X, Y, W, H	Content / overrides
line label	100, 590, 1080, 64	{_dialogue_speaker}: {_dialogue_text} background: #0a0f18ce color: #f6ead6 fontSize: 23 borderWidth: 0 groupId: @talkdialogue radius: 6

Opening · skip controls (opening_controls)

Setting	Value
id	opening_controls
name	Opening · skip controls
visible	false

Element	X, Y, W, H	Content / overrides
skip button	1090, 18, 166, 44	Skip intro Click: script:intro_skip background: #101a29b0 color: #f6ead6 borderColor: #9f8b64 radius: 6

R6. Clockmaker indoor timeline

The opening uses 49.8 seconds and 72 events. Copy time and duration separately from the action's own fields. For example, a timeline event lasting 0.01 seconds can launch a camera blend lasting several seconds.

Setting	Value
id	opening
name	A Minute Borrowed · directed opening
duration	49.8
skippable	true
restoreCamera	true
lockPlayerInput	true
hideHUD	true
hideMousePointer	true

0s - intro_prepare

ID flow_000; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_prepare

0s - Camera · clock

ID flow_001; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_clock; wait = false; duration = 0

0.02s - PlaySound

ID flow_002; track audio; timeline duration 0.01s.

1. Play Sound - assetId = opening_soundscape; channel = opening

0.04s - intro_voice_mix

ID flow_003; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_voice_mix

0.08s - CameraMove

ID flow_004; track camera; timeline duration 3.3s.

1. Camera Move - position = [-0.4, 2.15, -2.45]; target = [-1.5, 2.05, -4.8]; duration = 3.3; fov = 35

0.5s - Every afternoon, that bell calls the town home.

ID flow_063; track dialogue; timeline duration 4.0173s.

1. Say - speakerId = elias; text = Every afternoon, that bell calls the town home.; timeout = 4017.3; voiceAssetId = voice_elias_01

3.5s - Camera · follow

ID flow_005; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_follow; wait = false; duration = 0.75

3.55s - WalkTo

ID flow_006; track character; timeline duration 0.01s.

1. Walk To - entityId = ada; position = [-0.4, 0, 0.2]; speed = 0.75

3.58s - WalkTo

ID flow_007; track character; timeline duration 0.01s.

1. Walk To - entityId = elias; position = [0.8, 0, -0.8]; speed = 0.8

3.6s - intro_tracking

ID flow_008; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_tracking

4.8s - But not today.

ID flow_064; track dialogue; timeline duration 1.3893s.

1. Say - speakerId = ada; text = But not today.; timeout = 1389.3; voiceAssetId = voice_ivo_02

5.65s - intro_face_pair

ID flow_009; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_face_pair

6.45s - SetCharacterState

ID flow_010; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

6.45s - SetCharacterState

ID flow_011; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

6.8s - Camera · elias

ID flow_012; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_elias; wait = false; duration = 0.75

6.95s - SetCharacterState

ID flow_014; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = explain

7.1s - No chime. Just that uneven ticking.

ID flow_065; track dialogue; timeline duration 3.38s.

1. Say - speakerId = elias; text = No chime. Just that uneven ticking.; timeout = 3380.0; voiceAssetId = voice_elias_03

9.7s - SetCharacterState

ID flow_015; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

10.62s - SetCharacterState

ID flow_016; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = open_hands

10.8s - Hear it catch? Something inside has slipped.

ID flow_066; track dialogue; timeline duration 3.462s.

1. Say - speakerId = elias; text = Hear it catch? Something inside has slipped.; timeout = 3462.0; voiceAssetId = voice_elias_04

13.97s - SetCharacterState

ID flow_017; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

14.3s - intro_reverse_on

ID flow_018; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_on

14.35s - Camera · ivo

ID flow_019; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_ivo; wait = false; duration = 0.75

14.48s - SetCharacterState

ID flow_020; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = nod

14.6s - Then let's find it.

ID flow_067; track dialogue; timeline duration 1.25s.

1. Say - speakerId = ada; text = Then let's find it.; timeout = 1250.0; voiceAssetId = voice_ivo_05

15.85s - SetCharacterState

ID flow_021; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

15.95s - Camera · notebook

ID flow_022; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_notebook; wait = false; duration = 0.75

15.95s - intro_reverse_off

ID flow_023; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_off

16s - CameraMove

ID flow_024; track camera; timeline duration 4.15s.

1. Camera Move - position = [-1.55, 1.95, -0.65]; target = [-2.6, 1.45, -2.3]; duration = 4.15; fov = 33

16s - WalkTo

ID flow_025; track character; timeline duration 0.01s.

1. Walk To - entityId = ada; position = [-0.4, 0, -1.6]; speed = 0.75

16s - WalkTo

ID flow_026; track character; timeline duration 0.01s.

1. Walk To - entityId = elias; position = [0.8, 0, -2.6]; speed = 0.8

16.15s - My notebook's on the bench. The archive key is in the drawer.

ID flow_068; track dialogue; timeline duration 4.45s.

1. Say - speakerId = elias; text = My notebook's on the bench. The archive key is in the drawer.; timeout = 4450.0; voiceAssetId = voice_elias_06

18.8s - intro_face_pair

ID flow_027; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_face_pair

19.55s - SetCharacterState

ID flow_028; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

19.55s - SetCharacterState

ID flow_029; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

20.4s - Camera · pair

ID flow_030; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_pair; wait = false; duration = 0.75

20.95s - SetCharacterState

ID flow_032; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = explain

21.1s - We'll need the brass wheel from the archive-and a little bearing oil.

ID flow_069; track dialogue; timeline duration 4.5267s.

1. Say - speakerId = elias; text = We'll need the brass wheel from the archive-and a little bearing oil.; timeout = 4526.7; voiceAssetId = voice_elias_07

23.7s - SetCharacterState

ID flow_033; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

23.8s - SetCharacterState

ID flow_034; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = nod

25.15s - SetCharacterState

ID flow_035; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

25.65s - Camera · clock

ID flow_036; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_clock; wait = false; duration = 0.75

25.68s - CameraMove

ID flow_037; track camera; timeline duration 0.7s.

1. Camera Move - position = [-0.25, 2.2, -2]; target = [-1.5, 2.05, -4.8]; duration = 0.7; fov = 35

26.4s - intro_reverse_on

ID flow_038; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_on

26.43s - Camera · ivo detail

ID flow_039; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_ivo_detail; wait = false; duration = 0.75

26.45s - SetCharacterState

ID flow_040; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = open_hands

26.6s - The wheel, the oil, the key. One piece at a time.

ID flow_070; track dialogue; timeline duration 3.63s.

1. Say - speakerId = ada; text = The wheel, the oil, the key. One piece at a time.; timeout = 3630.0; voiceAssetId = voice_ivo_08

29.8s - SetCharacterState

ID flow_041; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

30.3s - intro_reverse_off

ID flow_042; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_off

30.33s - Camera · elias detail

ID flow_043; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_elias_detail; wait = false; duration = 0.75

30.4s - SetCharacterState

ID flow_045; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = nod

30.55s - I've wound it every morning for thirty years. I'd hate to leave it silent.

ID flow_071; track dialogue; timeline duration 5.05s.

1. Say - speakerId = elias; text = I've wound it every morning for thirty years. I'd hate to leave it silent.; timeout = 5050.0; voiceAssetId = voice_elias_09

31.75s - SetCharacterState

ID flow_046; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

35.8s - intro_reverse_on

ID flow_047; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_on

35.82s - Camera · ivo detail

ID flow_048; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_ivo_detail; wait = false; duration = 0.75

35.85s - SetCharacterState

ID flow_049; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = explain

36s - Then we'd better give the town its afternoon back.

ID flow_072; track dialogue; timeline duration 2.34s.

1. Say - speakerId = ada; text = Then we'd better give the town its afternoon back.; timeout = 2340.0; voiceAssetId = voice_ivo_10

38.45s - intro_reverse_off

ID flow_051; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_reverse_off

38.48s - Camera · elias detail

ID flow_052; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_elias_detail; wait = false; duration = 0.75

38.5s - SetCharacterState

ID flow_053; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = nod

38.6s - SetCharacterState

ID flow_050; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = listen

38.65s - Thank you, Ivo.

ID flow_073; track dialogue; timeline duration 1.7187s.

1. Say - speakerId = elias; text = Thank you, Ivo.; timeout = 1718.7; voiceAssetId = voice_elias_11

39.85s - SetCharacterState

ID flow_054; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = listen

40.6s - Camera · handshake

ID flow_055; track camera; timeline duration 0.01s.

1. Activate / Blend to Camera - cameraId = intro_handshake; wait = false; duration = 0.75

40.6s - Reach, clasp and release with hand attach points

ID handshake_0788; track character; timeline duration 4.95s.

1. Run Visual Script / Script - scriptId = intro_handshake

45.85s - WalkTo

ID flow_056; track character; timeline duration 0.01s.

1. Walk To - entityId = elias; position = [1.2, 0, -1.2]; speed = 0.75

45.88s - WalkTo

ID flow_057; track character; timeline duration 0.01s.

1. Walk To - entityId = ada; position = [-0.4, 0, 0.2]; speed = 0.8

45.9s - CameraMove

ID flow_058; track camera; timeline duration 3.8s.

1. Activate / Blend to Camera - cameraId = wide; duration = 3.8; wait = false

46.5s - intro_score_fade

ID flow_059; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_score_fade

48.35s - intro_face_pair

ID flow_060; track gameplay; timeline duration 0.01s.

1. Run Visual Script / Script - scriptId = intro_face_pair

49.3s - SetCharacterState

ID flow_061; track character; timeline duration 0.01s.

1. Set Character State - entityId = ada; stateId = idle

49.3s - SetCharacterState

ID flow_062; track character; timeline duration 0.01s.

1. Set Character State - entityId = elias; stateId = idle

R7. Exterior placement reference

The exterior contains repeated detail. Its folders keep related pieces together. Use the placement tables for the large pass and the attached complete recipes for the exact material, particle, accumulation and audio settings.

00 · Snow at the Clock Tower (clocktower_exterior)

Setting	Value
Navigation	{"type": "plane", "walkableY": 0, "agentRadius": 0.3, "bounds": {"minX": -20, "maxX": 20, "minZ": -20, "maxZ": 30}, "blocked": []}
Spawns	[{"id": "entry", "position": [0, 0, 26], "facing": 180}]
Default camera	exterior_wide
Background	#172335
Environment	{"brightness": 1, "fogEnabled": true, "glowEnabled": false, "skybox": {"enabled": true, "source": "procedural", "topColor": "#33485e", "horizonColor": "#65798d", "groundBottomColor": "#3b4a57", "groundHorizonColor": "#65798d", "energy": 0.55, "useSkyAmbient": false}, "postProcessOverrideMode": "custom", "postProcess": {"preset": "snow", "intensity": 0.8, "snowDensity": 0.34, "snowSpeed": 1.3, "snowAngle": 9, "snowScale": 0.82, "snowColor": "#dce7f1ff", "snowAccumulation": 0.52, "snowAccumulationSeconds": 26, "snowAccumulationPatchiness": 0.62, "snowAccumulationNoiseScale": 1.25}, "fogMode": "depth", "fogColor": "#728391", "fogDepthBegin": 16.0, "fogDepthEnd": 78.0, "fogDepthCurve": 1.15, "cinematicMode": "custom", "cinematic": {"enabled": true, "exposure": -0.16, "contrast": 1.04, "saturation": 0.78, "warmth": -0.05, "vignette": 0.23, "grain": 0.012, "letterbox": 0.105}, "fogAerialPerspective": 0.12, "shadowQuality": "low"}
Transition	{"type": "fade", "duration": 0.55, "color": "#080d15", "showName": false}

Winter courtyard

Entity / source	Transform	Size, appearance and behavior
Town Hall town_hall asset: clock_tower_exterior	P [0, 6.18, -12] R degrees [0.0, 0.0, 0.0] S [6.3, 6.3, 6.3]	No extra override.
Cobbled Courtyard cobbled_courtyard asset: courtyard_ground	P [0, -0.1, 7] R degrees [0.0, 0.0, 0.0] S [18, 3, 18]	No extra override.
Lamp Post 1 lamp_post_1 asset: courtyard_lamp	P [-5.8, 1.95, 8] R degrees [0.0, 0.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.
Lamp Post 2 lamp_post_2 asset: courtyard_lamp	P [5.8, 1.95, 8] R degrees [0.0, 0.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.
Lamp Post 3 lamp_post_3 asset: courtyard_lamp	P [-6.5, 1.95, -1] R degrees [0.0, 0.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.
Lamp Post 4 lamp_post_4 asset: courtyard_lamp	P [6.5, 1.95, -1] R degrees [0.0, 0.0, 0.0] S [2.0, 2.0, 2.0]	No extra override.

Room contents

Entity / source	Transform	Size, appearance and behavior
Distant snow ground distant_snow_ground primitive: box	P [0, -0.38, 0] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [440, 0.2, 440] material: {"color": "#6c7c8d", "roughness": 1.0} lowDetailGeometry: true receiveSnow: true shadow: {"cast": false}
Warm lamp 1 lamp_glow_1 light: light	P [-5.8, 3.5, 8.05] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#ffca83", "intensity": 0.85, "distance": 5.5, "castShadow": false}
Warm lamp 2 lamp_glow_2 light: light	P [5.8, 3.5, 8.05] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#ffca83", "intensity": 0.85, "distance": 5.5, "castShadow": false}
Warm lamp 3 lamp_glow_3 light: light	P [-6.5, 3.5, -0.95] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#ffca83", "intensity": 0.85, "distance": 5.5, "castShadow": false}
Warm lamp 4 lamp_glow_4 light: light	P [6.5, 3.5, -0.95] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#ffca83", "intensity": 0.85, "distance": 5.5, "castShadow": false}
Winter blue key winter_key light: light	P [-7, 15, 15] R degrees [-42.0, -25.0, 0.0] S [1, 1, 1]	light: {"kind": "directional", "color": "#c3d4eb", "intensity": 0.52, "distance": 42, "castShadow": true, "shadowDistance": 45, "shadowBias": 0.04, "normalBias": 0.8, "angularDistance": 0, "shadowOpacity": 0.64, "cascadeMode": "two"}
Winter fill winter_fill light: light	P [9, 10, 7] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	light: {"kind": "point", "color": "#9db9df", "intensity": 0.38, "distance": 30, "castShadow": false}

Characters

Entity / source	Transform	Size, appearance and behavior
Professor Ivo ada character: professor	P [0, 0, 26] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	attachPoints: [{"name": "right_hand_grip", "bone": "RightHand", "position": [-0.007982, 0.06379, 0.006534], "rotation": [0, 0, 0], "contactRadius": 0.018, "enabled": true, "space": "model"}] receiveSnow: false Character definition: ada

Winter town backdrop

Entity / source	Transform	Size, appearance and behavior
Town House 0 town_house_0 primitive: box	P [-15, 2.8, -24] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [6, 6, 6] material: {"color": "#656969", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 0 -1 town_roof_0_-1 primitive: box	P [-16.5, 6.85, -24] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.9, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 0 1 town_roof_0_1 primitive: box	P [-13.5, 6.85, -24] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.9, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 0 town_chimney_0 primitive: box	P [-13.5, 7.7, -25.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 -1 0 town_window_0_-1_0 primitive: box	P [-16.56, 1.6, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 0 -1 0 town_window_bar_0_-1_0 primitive: box	P [-16.56, 1.6, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 -1 1 town_window_0_-1_1 primitive: box	P [-16.56, 4.1, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 0 -1 1 town_window_bar_0_-1_1 primitive: box	P [-16.56, 4.1, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 0 0 town_window_0_0_0 primitive: box	P [-15.0, 1.6, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window Bar 0 0 0 town_window_bar_0_0_0 primitive: box	P [-15.0, 1.6, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 0 1 town_window_0_0_1 primitive: box	P [-15.0, 4.1, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 0 0 1 town_window_bar_0_0_1 primitive: box	P [-15.0, 4.1, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 1 0 town_window_0_1_0 primitive: box	P [-13.44, 1.6, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 0 1 0 town_window_bar_0_1_0 primitive: box	P [-13.44, 1.6, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 0 1 1 town_window_0_1_1 primitive: box	P [-13.44, 4.1, -20.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 0 1 1 town_window_bar_0_1_1 primitive: box	P [-13.44, 4.1, -20.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 1 town_house_1 primitive: box	P [-23, 3.3, -30] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [5, 7, 7] material: {"color": "#697277", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 1 -1 town_roof_1_-1 primitive: box	P [-24.25, 7.85, -30] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.25, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Roof 1 1 town_roof_1_1 primitive: box	P [-21.75, 7.85, -30] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.25, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 1 town_chimney_1 primitive: box	P [-21.75, 8.7, -31.4] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 -1 0 town_window_1_-1_0 primitive: box	P [-24.3, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 1 -1 0 town_window_bar_1_-1_0 primitive: box	P [-24.3, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 -1 1 town_window_1_-1_1 primitive: box	P [-24.3, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 1 -1 1 town_window_bar_1_-1_1 primitive: box	P [-24.3, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 0 0 town_window_1_0_0 primitive: box	P [-23.0, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 1 0 0 town_window_bar_1_0_0 primitive: box	P [-23.0, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 0 1 town_window_1_0_1 primitive: box	P [-23.0, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window Bar 1 0 1 town_window_bar_1_0_1 primitive: box	P [-23.0, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 1 0 town_window_1_1_0 primitive: box	P [-21.7, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 1 1 0 town_window_bar_1_1_0 primitive: box	P [-21.7, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 1 1 1 town_window_1_1_1 primitive: box	P [-21.7, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 1 1 1 town_window_bar_1_1_1 primitive: box	P [-21.7, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 2 town_house_2 primitive: box	P [-8, 2.3, -34] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [5, 5, 6] material: {"color": "#6a6868", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 2 -1 town_roof_2_-1 primitive: box	P [-9.25, 5.85, -34] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.25, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 2 1 town_roof_2_1 primitive: box	P [-6.75, 5.85, -34] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.25, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 2 town_chimney_2 primitive: box	P [-6.75, 6.7, -35.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 2 -1 0 town_window_2_-1_0 primitive: box	P [-9.3, 1.6, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 2 -1 0 town_window_bar_2_-1_0 primitive: box	P [-9.3, 1.6, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 2 -1 1 town_window_2_-1_1 primitive: box	P [-9.3, 4.1, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 2 -1 1 town_window_bar_2_-1_1 primitive: box	P [-9.3, 4.1, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 2 0 0 town_window_2_0_0 primitive: box	P [-8.0, 1.6, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 2 0 0 town_window_bar_2_0_0 primitive: box	P [-8.0, 1.6, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 2 0 1 town_window_2_0_1 primitive: box	P [-8.0, 4.1, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 2 0 1 town_window_bar_2_0_1 primitive: box	P [-8.0, 4.1, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 2 1 0 town_window_2_1_0 primitive: box	P [-6.7, 1.6, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window Bar 2 1 0 town_window_bar_2_1_0 primitive: box	P [-6.7, 1.6, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 2 1 1 town_window_2_1_1 primitive: box	P [-6.7, 4.1, -30.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 2 1 1 town_window_bar_2_1_1 primitive: box	P [-6.7, 4.1, -30.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 3 town_house_3 primitive: box	P [10, 2.8, -30] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [6, 6, 7] material: {"color": "#656969", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 3 -1 town_roof_3_-1 primitive: box	P [8.5, 6.85, -30] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.9, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 3 1 town_roof_3_1 primitive: box	P [11.5, 6.85, -30] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.9, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 3 town_chimney_3 primitive: box	P [11.5, 7.7, -31.4] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 3 -1 0 town_window_3_-1_0 primitive: box	P [8.44, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 3 -1 0 town_window_bar_3_-1_0 primitive: box	P [8.44, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 3 -1 1 town_window_3_-1_1 primitive: box	P [8.44, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 3 -1 1 town_window_bar_3_-1_1 primitive: box	P [8.44, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 3 0 0 town_window_3_0_0 primitive: box	P [10.0, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 3 0 0 town_window_bar_3_0_0 primitive: box	P [10.0, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 3 0 1 town_window_3_0_1 primitive: box	P [10.0, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 3 0 1 town_window_bar_3_0_1 primitive: box	P [10.0, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 3 1 0 town_window_3_1_0 primitive: box	P [11.56, 1.6, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 3 1 0 town_window_bar_3_1_0 primitive: box	P [11.56, 1.6, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 3 1 1 town_window_3_1_1 primitive: box	P [11.56, 4.1, -26.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window Bar 3 1 1 town_window_bar_3_1_1 primitive: box	P [11.56, 4.1, -26.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 4 town_house_4 primitive: box	P [19, 3.3, -28] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [5, 7, 6] material: {"color": "#697277", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 4 -1 town_roof_4_-1 primitive: box	P [17.75, 7.85, -28] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.25, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 4 1 town_roof_4_1 primitive: box	P [20.25, 7.85, -28] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.25, 0.3, 7] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 4 town_chimney_4 primitive: box	P [20.25, 8.7, -29.2] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 4 -1 0 town_window_4_-1_0 primitive: box	P [17.7, 1.6, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 -1 0 town_window_bar_4_-1_0 primitive: box	P [17.7, 1.6, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 4 -1 1 town_window_4_-1_1 primitive: box	P [17.7, 4.1, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 -1 1 town_window_bar_4_-1_1 primitive: box	P [17.7, 4.1, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 4 0 0 town_window_4_0_0 primitive: box	P [19.0, 1.6, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 0 0 town_window_bar_4_0_0 primitive: box	P [19.0, 1.6, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 4 0 1 town_window_4_0_1 primitive: box	P [19.0, 4.1, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 0 1 town_window_bar_4_0_1 primitive: box	P [19.0, 4.1, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 4 1 0 town_window_4_1_0 primitive: box	P [20.3, 1.6, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 1 0 town_window_bar_4_1_0 primitive: box	P [20.3, 1.6, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 4 1 1 town_window_4_1_1 primitive: box	P [20.3, 4.1, -24.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 4 1 1 town_window_bar_4_1_1 primitive: box	P [20.3, 4.1, -24.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 5 town_house_5 primitive: box	P [27, 2.8, -37] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [7, 6, 8] material: {"color": "#6a6868", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Roof 5 -1 town_roof_5_-1 primitive: box	P [25.25, 6.85, -37] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [4.55, 0.3, 9] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 5 1 town_roof_5_1 primitive: box	P [28.75, 6.85, -37] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [4.55, 0.3, 9] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 5 town_chimney_5 primitive: box	P [28.75, 7.7, -38.6] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 5 -1 0 town_window_5_-1_0 primitive: box	P [25.18, 1.6, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 -1 0 town_window_bar_5_-1_0 primitive: box	P [25.18, 1.6, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 5 -1 1 town_window_5_-1_1 primitive: box	P [25.18, 4.1, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 -1 1 town_window_bar_5_-1_1 primitive: box	P [25.18, 4.1, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 5 0 0 town_window_5_0_0 primitive: box	P [27.0, 1.6, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 0 0 town_window_bar_5_0_0 primitive: box	P [27.0, 1.6, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 5 0 1 town_window_5_0_1 primitive: box	P [27.0, 4.1, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 0 1 town_window_bar_5_0_1 primitive: box	P [27.0, 4.1, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 5 1 0 town_window_5_1_0 primitive: box	P [28.82, 1.6, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 1 0 town_window_bar_5_1_0 primitive: box	P [28.82, 1.6, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 5 1 1 town_window_5_1_1 primitive: box	P [28.82, 4.1, -32.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 5 1 1 town_window_bar_5_1_1 primitive: box	P [28.82, 4.1, -32.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 6 town_house_6 primitive: box	P [-30, 3.8, -43] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [7, 8, 7] material: {"color": "#656969", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 6 -1 town_roof_6_-1 primitive: box	P [-31.75, 8.85, -43] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [4.55, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 6 1 town_roof_6_1 primitive: box	P [-28.25, 8.85, -43] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [4.55, 0.3, 8] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Chimney 6 town_chimney_6 primitive: box	P [-28.25, 9.7, -44.4] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 6 -1 0 town_window_6_-1_0 primitive: box	P [-31.82, 1.6, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 -1 0 town_window_bar_6_-1_0 primitive: box	P [-31.82, 1.6, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 6 -1 1 town_window_6_-1_1 primitive: box	P [-31.82, 4.1, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 -1 1 town_window_bar_6_-1_1 primitive: box	P [-31.82, 4.1, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 6 0 0 town_window_6_0_0 primitive: box	P [-30.0, 1.6, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 0 0 town_window_bar_6_0_0 primitive: box	P [-30.0, 1.6, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 6 0 1 town_window_6_0_1 primitive: box	P [-30.0, 4.1, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 0 1 town_window_bar_6_0_1 primitive: box	P [-30.0, 4.1, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 6 1 0 town_window_6_1_0 primitive: box	P [-28.18, 1.6, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 1 0 town_window_bar_6_1_0 primitive: box	P [-28.18, 1.6, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 6 1 1 town_window_6_1_1 primitive: box	P [-28.18, 4.1, -39.47] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 6 1 1 town_window_bar_6_1_1 primitive: box	P [-28.18, 4.1, -39.42] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town House 7 town_house_7 primitive: box	P [3, 2.8, -43] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [6, 6, 8] material: {"color": "#697277", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 7 -1 town_roof_7_-1 primitive: box	P [1.5, 6.85, -43] R degrees [0.0, 0.0, 32.0] S [1, 1, 1]	size: [3.9, 0.3, 9] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Roof 7 1 town_roof_7_1 primitive: box	P [4.5, 6.85, -43] R degrees [0.0, 0.0, -32.0] S [1, 1, 1]	size: [3.9, 0.3, 9] material: {"color": "#99a9b8", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Chimney 7 town_chimney_7 primitive: box	P [4.5, 7.7, -44.6] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 2, 0.7] material: {"color": "#52575b", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 7 -1 0 town_window_7_-1_0 primitive: box	P [1.44, 1.6, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window Bar 7 -1 0 town_window_bar_7_-1_0 primitive: box	P [1.44, 1.6, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 7 -1 1 town_window_7_-1_1 primitive: box	P [1.44, 4.1, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 7 -1 1 town_window_bar_7_-1_1 primitive: box	P [1.44, 4.1, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 7 0 0 town_window_7_0_0 primitive: box	P [3.0, 1.6, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 7 0 0 town_window_bar_7_0_0 primitive: box	P [3.0, 1.6, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 7 0 1 town_window_7_0_1 primitive: box	P [3.0, 4.1, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 7 0 1 town_window_bar_7_0_1 primitive: box	P [3.0, 4.1, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window 7 1 0 town_window_7_1_0 primitive: box	P [4.56, 1.6, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#a3967f", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 7 1 0 town_window_bar_7_1_0 primitive: box	P [4.56, 1.6, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Town Window 7 1 1 town_window_7_1_1 primitive: box	P [4.56, 4.1, -38.97] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.65, 1.05, 0.06] material: {"color": "#576470", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Town Window Bar 7 1 1 town_window_bar_7_1_1 primitive: box	P [4.56, 4.1, -38.92] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.06, 1.08, 0.05] material: {"color": "#424d55", "roughness": 0.92} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 0 forest_pine_0 procedural: procedural	P [-52.0, -0.2, -68.384077] R degrees [0.0, 86.5777, 0.0] S [1.393316, 1.671979, 1.393316]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 1 forest_pine_1 procedural: procedural	P [-47.6, -0.2, -68.806006] R degrees [0.0, 171.4452, 0.0] S [1.203556, 1.444268, 1.203556]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 2 forest_pine_2 procedural: procedural	P [-43.2, -0.2, -64.042105] R degrees [0.0, 52.3697, 0.0] S [1.6502, 1.98024, 1.6502]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 3 forest_pine_3 procedural: procedural	P [-38.8, -0.2, -65.351711] R degrees [0.0, 306.4028, 0.0] S [1.550558, 1.860669, 1.550558]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 4 forest_pine_4 procedural: procedural	P [-34.4, -0.2, -54.510322] R degrees [0.0, 62.1647, 0.0] S [1.961458, 2.35375, 1.961458]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 5 forest_pine_5 procedural: procedural	P [-30.0, -0.2, -68.640395] R degrees [0.0, 300.8497, 0.0] S [1.556583, 1.8679, 1.556583]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 6 forest_pine_6 procedural: procedural	P [-25.6, -0.2, -57.291732] R degrees [0.0, 31.3856, 0.0] S [1.389334, 1.667201, 1.389334]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Forest Pine 7 forest_pine_7 procedural: procedural	P [-21.2, -0.2, -68.016945] R degrees [0.0, 73.7915, 0.0] S [1.580571, 1.896686, 1.580571]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 8 forest_pine_8 procedural: procedural	P [-16.8, -0.2, -57.450383] R degrees [0.0, 72.0475, 0.0] S [1.172854, 1.407425, 1.172854]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 9 forest_pine_9 procedural: procedural	P [-12.4, -0.2, -64.081875] R degrees [0.0, 253.5339, 0.0] S [1.493877, 1.792652, 1.493877]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 10 forest_pine_10 procedural: procedural	P [-8.0, -0.2, -55.373379] R degrees [0.0, 353.5503, 0.0] S [1.855538, 2.226646, 1.855538]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 11 forest_pine_11 procedural: procedural	P [-3.6, -0.2, -67.314575] R degrees [0.0, 305.908, 0.0] S [1.76882, 2.122584, 1.76882]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 12 forest_pine_12 procedural: procedural	P [0.8, -0.2, -55.708634] R degrees [0.0, 28.5592, 0.0] S [1.515115, 1.818138, 1.515115]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 13 forest_pine_13 procedural: procedural	P [5.2, -0.2, -68.068229] R degrees [0.0, 38.3316, 0.0] S [1.584176, 1.901012, 1.584176]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 14 forest_pine_14 procedural: procedural	P [9.6, -0.2, -62.005649] R degrees [0.0, 198.7394, 0.0] S [1.799158, 2.158989, 1.799158]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 15 forest_pine_15 procedural: procedural	P [14.0, -0.2, -56.186641] R degrees [0.0, 20.3017, 0.0] S [1.943168, 2.331801, 1.943168]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Forest Pine 16 forest_pine_16 procedural: procedural	P [18.4, -0.2, -69.175901] R degrees [0.0, 270.3809, 0.0] S [1.976617, 2.371941, 1.976617]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 17 forest_pine_17 procedural: procedural	P [22.8, -0.2, -71.025604] R degrees [0.0, 78.8973, 0.0] S [1.577872, 1.893446, 1.577872]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 18 forest_pine_18 procedural: procedural	P [27.2, -0.2, -62.5012] R degrees [0.0, 276.6639, 0.0] S [1.764335, 2.117201, 1.764335]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 19 forest_pine_19 procedural: procedural	P [31.6, -0.2, -60.917904] R degrees [0.0, 12.6349, 0.0] S [1.952518, 2.343022, 1.952518]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 20 forest_pine_20 procedural: procedural	P [36.0, -0.2, -62.924622] R degrees [0.0, 128.8986, 0.0] S [1.135512, 1.362614, 1.135512]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 21 forest_pine_21 procedural: procedural	P [40.4, -0.2, -71.807409] R degrees [0.0, 354.8258, 0.0] S [1.206108, 1.44733, 1.206108]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 22 forest_pine_22 procedural: procedural	P [44.8, -0.2, -70.292131] R degrees [0.0, 20.1442, 0.0] S [1.719446, 2.063335, 1.719446]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 23 forest_pine_23 procedural: procedural	P [49.2, -0.2, -62.487305] R degrees [0.0, 34.269, 0.0] S [1.72276, 2.067312, 1.72276]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Forest Pine 24 forest_pine_24 procedural: procedural	P [53.6, -0.2, -59.368231] R degrees [0.0, 303.6621, 0.0] S [1.182642, 1.419171, 1.182642]	generator: pine-tree material: {"color": "#73878d", "roughness": 0.96} lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
Courtyard Bench 0 courtyard_bench_0 procedural: procedural	P [-9, 0, 4] R degrees [0.0, 20.0, 0.0] S [1.1, 1.1, 1.1]	generator: park-bench material: {"color": "#5c5146", "roughness": 0.96} lowDetailGeometry: true
Courtyard Wall 0 courtyard_wall_0 primitive: box	P [-13.5, 0.6, -3] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 1.2, 17] material: {"color": "#707779", "roughness": 0.92} lowDetailGeometry: true
Courtyard Bench 1 courtyard_bench_1 procedural: procedural	P [9, 0, 4] R degrees [0.0, -20.0, 0.0] S [1.1, 1.1, 1.1]	generator: park-bench material: {"color": "#5c5146", "roughness": 0.96} lowDetailGeometry: true
Courtyard Wall 1 courtyard_wall_1 primitive: box	P [13.5, 0.6, -3] R degrees [0.0, 0.0, 0.0] S [1, 1, 1]	size: [0.7, 1.2, 17] material: {"color": "#707779", "roughness": 0.92} lowDetailGeometry: true
Distant snow ridge 1 distant_ridge_0 primitive: sphere	P [-65, -10, -75] R degrees [0.0, 0.0, 0.0] S [65.0, 18.0, 32.5]	material: {"color": "#a6b4c0", "roughness": 1.0} size: [2, 2, 2] lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Distant snow ridge 2 distant_ridge_1 primitive: sphere	P [35, -12, -85] R degrees [0.0, 0.0, 0.0] S [60.0, 20.0, 32.5]	material: {"color": "#a6b4c0", "roughness": 1.0} size: [2, 2, 2] lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
Distant snow ridge 3 distant_ridge_2 primitive: sphere	P [5, -13, -110] R degrees [0.0, 0.0, 0.0] S [75.0, 22.5, 30.0]	material: {"color": "#a6b4c0", "roughness": 1.0} size: [2, 2, 2] lowDetailGeometry: true receiveSnow: false shadow: {"cast": false}
winter tree 0 trunk winter_tree_0_trunk primitive: cylinder	P [-10.94, 3.2, 9.1] R degrees [1.7896, 0.0, -1.0742] S [1, 1, 1]	size: [0.23, 6.404249, 0.23] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 0 winter_tree_0_0 primitive: cylinder	P [-11.875, 2.625, 8.825] R degrees [-8.4693, 0.0, 51.9112] S [1, 1, 1]	size: [0.1, 2.376447, 0.1] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 0 twig0 winter_tree_0_0_twig0 primitive: cylinder	P [-12.2425, 3.1725, 8.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 0 twig1 winter_tree_0_0_twig1 primitive: cylinder	P [-12.7595, 3.6365, 8.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
winter tree 0 1 winter_tree_0_1 primitive: cylinder	P [-10.155, 3.325, 9.2] R degrees [10.5302, 0.0, -47.6368] S [1, 1, 1]	size: [0.088, 2.188744, 0.088] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 1 twig0 winter_tree_0_1_twig0 primitive: cylinder	P [-9.8005, 3.8725, 8.995] R degrees [-22.2906, 0.0, -30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 1 twig1 winter_tree_0_1_twig1 primitive: cylinder	P [-9.3667, 4.3365, 9.623] R degrees [28.0834, 0.0, -22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 2 winter_tree_0_2 primitive: cylinder	P [-11.715, 4.225, 8.825] R degrees [-9.4273, 0.0, 46.5378] S [1, 1, 1]	size: [0.076, 2.136797, 0.076] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 2 twig0 winter_tree_0_2_twig0 primitive: cylinder	P [-12.0665, 4.7725, 8.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 2 twig1 winter_tree_0_2_twig1 primitive: cylinder	P [-12.4811, 5.2365, 8.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 3 winter_tree_0_3 primitive: cylinder	P [-10.315, 5.025, 9.2] R degrees [11.7235, 0.0, -41.2139] S [1, 1, 1]	size: [0.064, 1.968604, 0.064] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 3 twig0 winter_tree_0_3_twig0 primitive: cylinder	P [-9.9765, 5.5725, 8.995] R degrees [-22.2906, 0.0, -30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 3 twig1 winter_tree_0_3_twig1 primitive: cylinder	P [-9.6451, 6.0365, 9.623] R degrees [28.0834, 0.0, -22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 0 4 winter_tree_0_4 primitive: cylinder	P [-11.555, 5.825, 8.825] R degrees [-10.4994, 0.0, 39.8444] S [1, 1, 1]	size: [0.052, 1.920703, 0.052] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 0 4 twig0 winter_tree_0_4_twig0 primitive: cylinder	P [-11.8905, 6.3725, 8.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
winter tree 0 4 twig1 winter_tree_0_4_twig1 primitive: cylinder	P [-12.2027, 6.8365, 8.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 trunk winter_tree_1_trunk primitive: cylinder	P [13.06, 3.2, 2.1] R degrees [1.7896, 0.0, -1.0742] S [1, 1, 1]	size: [0.23, 6.404249, 0.23] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 0 winter_tree_1_0 primitive: cylinder	P [12.125, 2.625, 1.825] R degrees [-8.4693, 0.0, 51.9112] S [1, 1, 1]	size: [0.1, 2.376447, 0.1] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 0 twig0 winter_tree_1_0_twig0 primitive: cylinder	P [11.7575, 3.1725, 1.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 0 twig1 winter_tree_1_0_twig1 primitive: cylinder	P [11.2405, 3.6365, 1.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 1 winter_tree_1_1 primitive: cylinder	P [13.845, 3.325, 2.2] R degrees [10.5302, 0.0, -47.6368] S [1, 1, 1]	size: [0.088, 2.188744, 0.088] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 1 twig0 winter_tree_1_1_twig0 primitive: cylinder	P [14.1995, 3.8725, 1.995] R degrees [-22.2906, 0.0, -30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 1 twig1 winter_tree_1_1_twig1 primitive: cylinder	P [14.6333, 4.3365, 2.623] R degrees [28.0834, 0.0, -22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 2 winter_tree_1_2 primitive: cylinder	P [12.285, 4.225, 1.825] R degrees [-9.4273, 0.0, 46.5378] S [1, 1, 1]	size: [0.076, 2.136797, 0.076] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 2 twig0 winter_tree_1_2_twig0 primitive: cylinder	P [11.9335, 4.7725, 1.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 2 twig1 winter_tree_1_2_twig1 primitive: cylinder	P [11.5189, 5.2365, 1.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}

Entity / source	Transform	Size, appearance and behavior
winter tree 1 3 winter_tree_1_3 primitive: cylinder	P [13.685, 5.025, 2.2] R degrees [11.7235, 0.0, -41.2139] S [1, 1, 1]	size: [0.064, 1.968604, 0.064] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 3 twig0 winter_tree_1_3_twig0 primitive: cylinder	P [14.0235, 5.5725, 1.995] R degrees [-22.2906, 0.0, -30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 3 twig1 winter_tree_1_3_twig1 primitive: cylinder	P [14.3549, 6.0365, 2.623] R degrees [28.0834, 0.0, -22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 4 winter_tree_1_4 primitive: cylinder	P [12.445, 5.825, 1.825] R degrees [-10.4994, 0.0, 39.8444] S [1, 1, 1]	size: [0.052, 1.920703, 0.052] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true
winter tree 1 4 twig0 winter_tree_1_4_twig0 primitive: cylinder	P [12.1095, 6.3725, 1.5825] R degrees [-22.2906, 0.0, 30.0686] S [1, 1, 1]	size: [0.033, 1.186381, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}
winter tree 1 4 twig1 winter_tree_1_4_twig1 primitive: cylinder	P [11.7973, 6.8365, 1.9705] R degrees [28.0834, 0.0, 22.8337] S [1, 1, 1]	size: [0.033, 1.168332, 0.033] material: {"color": "#514c47", "roughness": 1.0} lowDetailGeometry: true shadow: {"cast": false}

R8. Asset manifest and scripts

The manifest lists the actual bundled source locations. Model, image and audio names are not interchangeable: choose the correct asset kind when recreating a record. The full project recipes retain prepared metadata such as voice tracks.

Primitive assets

Asset	Kind	Source path
Key Icon key_icon	image svg	res://assets/prototype/key.svg
Wheel Icon wheel_icon	image svg	res://assets/prototype/wheel.svg
Oil Icon oil_icon	image svg	res://assets/prototype/oil.svg
Prototype Icon prototype_icon	image svg	res://assets/prototype/prototype_icon.svg
Chime chime	audio wav	res://assets/prototype/chime.wav

Clockmaker assets

Asset	Kind	Source path
Grandfather Clock grandfather_clock	model glb	res://assets/clockmaker/grandfather_clock.glb
Work Table work_table	model glb	res://assets/clockmaker/work_table.glb
Bookcase bookcase	model glb	res://assets/clockmaker/bookcase.glb
Desk desk	model glb	res://assets/clockmaker/desk.glb
Lantern lantern	model glb	res://assets/clockmaker/lantern.glb
Open Book open_book	model glb	res://assets/clockmaker/open_book.glb
Owl owl	model glb	res://assets/clockmaker/owl.glb
Door door	model glb	res://assets/clockmaker/door.glb
Window window	model glb	res://assets/clockmaker/window.glb
Rug rug	model glb	res://assets/clockmaker/rug.glb
Typewriter typewriter	model glb	res://assets/clockmaker/typewriter.glb
Ada ada	model glb	res://assets/clockmaker/ada.glb
Professor professor	model glb	res://assets/clockmaker/professor.glb
Brand Icon brand_icon	image png	res://assets/branding/pca3d_app_icon.png
Brand Logo brand_logo	image png	res://assets/branding/pca3d_compact_logo.png
Brand Splash brand_splash	image png	res://assets/branding/pca3d_splash_logo.png
Walk · brass engraving gui_walk	image webp	res://assets/clockmaker/interface/walk.webp

Asset	Kind	Source path
Look · brass engraving gui_look	image webp	res://assets/clockmaker/interface/look.webp
Use · brass engraving gui_use	image webp	res://assets/clockmaker/interface/use.webp
Take · brass engraving gui_take	image webp	res://assets/clockmaker/interface/take.webp
Gui Inventory gui_inventory	image png	res://assets/demo_gui/gui_inventory_grid.png
Key Icon key_icon	image svg	res://assets/prototype/key.svg
Wheel Icon wheel_icon	image svg	res://assets/prototype/wheel.svg
Oil Icon oil_icon	image svg	res://assets/prototype/oil.svg
Prototype Icon prototype_icon	image svg	res://assets/prototype/prototype_icon.svg
Chime chime	audio wav	res://assets/prototype/chime.wav
Talk · brass engraving gui_talk	image webp	res://assets/clockmaker/interface/talk.webp
Clockwork foreground - transparent parallax cinematic_foreground	image png	res://assets/images/clockmaker_cinematic_foreground.png
A minute borrowed - opening soundscape opening_soundscape	audio ogg	res://assets/clockmaker/opening_soundscape.ogg
Soft dust mote window_dust_image	image svg	res://assets/clockmaker/dust_mote.svg
Clock tower exterior clock_tower_exterior	model glb	res://assets/clockmaker/exterior/clock_tower.glb
Courtyard ground courtyard_ground	model glb	res://assets/clockmaker/exterior/courtyard_ground.glb
Courtyard lamp post courtyard_lamp	model glb	res://assets/clockmaker/exterior/lamp_post.glb
Winter wind winter_wind	audio mp3	res://assets/audio/clockmaker/snowstorm.mp3

Asset	Kind	Source path
Uneven clock ticking broken_clock_tick	audio mp3	res://assets/audio/clockmaker/broken_clock.mp3
Wood footsteps original recording wood_steps_original	audio mp3	res://assets/audio/clockmaker/wooden_footsteps_original.mp3
Wooden floor single footstep wood_step	audio wav	res://assets/audio/clockmaker/wood_step.wav
Created with PCA 3D clockmaker_created	image webp	res://assets/clockmaker/interface/splash_created.webp
A DiPerla Game Productions clockmaker_productions	image webp	res://assets/clockmaker/interface/splash_diperla.webp
Elias - 01 - Every afternoon, that bell calls the town home. voice_elias_01	audio ogg	res://assets/clockmaker/voices/elias_01.ogg
Ivo - 02 - But not today. voice_ivo_02	audio ogg	res://assets/clockmaker/voices/ivo_02.ogg
Elias - 03 - No chime. Just that uneven ticking. voice_elias_03	audio ogg	res://assets/clockmaker/voices/elias_03.ogg
Elias - 04 - Hear it catch? Something inside has slipped. voice_elias_04	audio ogg	res://assets/clockmaker/voices/elias_04.ogg
Ivo - 05 - Then let's find it. voice_ivo_05	audio ogg	res://assets/clockmaker/voices/ivo_05.ogg
Elias - 06 - My notebook's on the bench. The archive key is in the drawer. voice_elias_06	audio ogg	res://assets/clockmaker/voices/elias_06.ogg
Elias - 07 - We'll need the brass wheel from the archive-and a little bearing oil. voice_elias_07	audio ogg	res://assets/clockmaker/voices/elias_07.ogg
Ivo - 08 - The wheel, the oil, the key. One piece at a time. voice_ivo_08	audio ogg	res://assets/clockmaker/voices/ivo_08.ogg
Elias - 09 - I've wound it every morning for thirty years. I'd hate to leave it silent. voice_elias_09	audio ogg	res://assets/clockmaker/voices/elias_09.ogg

Asset	Kind	Source path
Ivo - 10 - Then we'd better give the town its afternoon back. voice_ivo_10	audio ogg	res://assets/clockmaker/voices/ivo_10.ogg
Elias - 11 - Thank you, Ivo. voice_elias_11	audio ogg	res://assets/clockmaker/voices/elias_11.ogg

Script sources and complete records

The practice archive includes the original short puzzle scripts and the longer directed-opening helpers. The PDF also carries a recipe ZIP as an attachment for readers who want everything alongside the book. Some mobile PDF viewers do not expose attachments; use the separate practice download in that case.

The recipes are source records, not a second engine installation. They use the same bundled asset paths as the supplied demos. Open them with the matching PCA 3D package. Keep that source package separate from your built player release.

R9. Glossary and further reading

Term	Meaning here
Action	One instruction run by the shared gameplay system.
Action list	An ordered sequence attached to an event or rule.
Anchor	The reference point used to position a GUI element.
Asset	An imported source file and its project record.
Blocker	An area that prevents navigation.
Character definition	Reusable identity, model, movement, speech and animation settings.
Character placement	One instance of that character in a room.
Collider	A shape used for physical obstruction or contact.
Context	Information about the event that called an action or script.
Default verb	The action a context click chooses for a hotspot.
Event volume	An area used to detect entry, exit or overlap.
Hotspot	An enabled interaction target in the room.
ID	A stable reference stored by other records and scripts.
Navigation	The allowed walking surface and its obstacles.
Record	A saved project object such as a scene, item, GUI or script.
Spawn	A named arrival position and facing in a room.
State flag	A boolean remembering whether something has happened.
this	The record whose event called the script.
Track	A timeline row used to organize scheduled events.
Variable	A named value the game stores and changes.

Manuals worth reading

The manuals and tutorials below are useful companion reading. This book follows a familiar teaching order: begin with a working project, explain the main editors, build one interaction and test it before adding complexity. PCA 3D's controls and recipes were checked against its own source and demos; the other engines use different terminology and workflows.

- PowerQuest, Getting Started: https://powerquest.powerhoof.com/getting_started.html
- Adventure Creator, Introduction: https://adventurecreator.org/tutorials/introduction
- Adventure Creator, official manual: https://adventurecreator.org/files/Manual.pdf
- Adventure Game Studio, Getting Started: https://adventuregamestudio.github.io/ags-manual/acintro1.html
- RPG in a Box, official documentation: https://docs.rpginabox.com/
- RPG in a Box, developer's introductory tutorial: https://www.youtube.com/watch?v=PEqCujPa2HI

Credits and version note

PCA 3D Adventure Studio and its bundled demos are the software covered by this book. Screenshots show the actual editor and its supplied content. Advanced cameras use the bundled Phantom Camera integration by Marcus Skov / Ramokz. Refer to the installation's third-party notices for the complete asset and component credits.

Component used or described	Developer / credit
Phantom Camera integration	Marcus Skov / Ramokz
PSX shader, N64 filter and fog source	Francisco Cidade
Krita Wizard parser/compositor	Aditya Nandan
Controller Icons mapping	Ricardo Subtil
Controller, keyboard and mouse artwork	Nicolae (XELU) Berbece / Those Awesome Guys; additional artwork by TacticalLaptopBag and DataPlusProgram
Typewriter Dialogue Blips	yetiancai
Easy Zoom Blur visual reference	Vefects; PCA 3D uses its own implementation, not the restricted asset pack

This edition follows 0.86.3.4. Earlier copies of user projects retain their own authored records; updating the editor does not automatically replace an old opening cutscene with the current Clockmaker template. Make a fresh template copy when you want to compare the current example.

R10. Directed-opening helper sources

These are the supplied GDScript helpers used by the Clockmaker introduction. Keep their IDs and prepared character/asset data unchanged while matching the demo. Long lines continue at an extra indentation for print; the attached .gd files preserve exact source line breaks.

The small dial, menu and hint scripts are included too. Use the normal GDScript editor, apply each source and save. Some old demo action records say RunVisualScript even when their ID names one of these text scripts; the bundled runtime resolves that shared script call. Preserve the supplied recipe when matching the example.

menu

Opens the built-in pause/menu screen.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.system_menu.show_pause()
    return true
```

turn_one

Checks numeral I against the winding sequence and advances or resets dial.

```
func run(_context: Dictionary = {}) -> Variant:
    var sequence: Array = [3, 1, 2]
    var step: int = int(Runtime.variables.get("dial", 0))
    if 1 != sequence[clampi(step, 0, 2)]:
        await SetVariable({"name":"dial", "value":0})
        await ShowMessage({"text":"A soft click. The winding sequence resets.", "duration":1.3})
        return false
    step += 1
    await SetVariable({"name":"dial", "value":step})
    if step == 3:
        await HideGUI({"guiId":"dial"})
        await RunVisualScript({"scriptId":"ring_bell"})
    else:
        await ShowMessage({"text":"A clean click. The next turn is ready.","duration":0.8})
    return true
```

turn_two

Checks numeral II against the winding sequence and advances or resets dial.

```
func run(_context: Dictionary = {}) -> Variant:
    var sequence: Array = [3, 1, 2]
    var step: int = int(Runtime.variables.get("dial", 0))
    if 2 != sequence[clampi(step, 0, 2)]:
        await SetVariable({"name":"dial", "value":0})
        await ShowMessage({"text":"A soft click. The winding sequence resets.", "duration":1.3})
        return false
    step += 1
    await SetVariable({"name":"dial", "value":step})
    if step == 3:
        await HideGUI({"guiId":"dial"})
        await RunVisualScript({"scriptId":"ring_bell"})
    else:
        await ShowMessage({"text":"A clean click. The next turn is ready.","duration":0.8})
    return true
```

turn_three

Checks numeral III against the winding sequence and advances or resets dial.

```
func run(_context: Dictionary = {}) -> Variant:
    var sequence: Array = [3, 1, 2]
    var step: int = int(Runtime.variables.get("dial", 0))
    if 3 != sequence[clampi(step, 0, 2)]:
        await SetVariable({"name":"dial", "value":0})
        await ShowMessage({"text":"A soft click. The winding sequence resets.", "duration":1.3})
        return false
    step += 1
    await SetVariable({"name":"dial", "value":step})
    if step == 3:
        await HideGUI({"guiId":"dial"})
        await RunVisualScript({"scriptId":"ring_bell"})
    else:
        await ShowMessage({"text":"A clean click. The next turn is ready.","duration":0.8})
    return true
```

hint

Chooses a hint from the current puzzle flags, then shows it.

```
func run(_context: Dictionary = {}) -> Variant:
    var v: Dictionary = Runtime.variables
    var hint: String = "Wind the clock III, I, II. A wrong turn resets; you cannot get stuck."
    if not v.get("key_taken",false): hint = "Open the workbench drawer in the workshop."
    elif not v.get("archive_open",false): hint = "Select the archive key in Inventory, close Inventory,
        then click the archive door."
    elif not v.get("wheel_taken",false): hint = "Use the brass owl on the archive reading table."
    elif not v.get("oil_taken",false): hint = "Pick up the small brass oil tin on the archive floor."
    elif not v.get("wheel_fitted",false): hint = "Use the brass wheel from Inventory on the master clock
        in the chamber."
    elif not v.get("oiled",false): hint = "Use the bearing oil on the master clock."
    await ShowMessage({"text":hint,"duration":4.5})
    return true
```

intro_prepare

Prepares the directed indoor opening, including initial actor and presentation state.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.change_game_setting("cinematic.letterbox", 0.105)
    Runtime.project_data.settings.screenOverlay.enabled = false
    Runtime._set_screen_overlay_visible(false)
    Runtime.show_gui("opening_controls")
    Runtime.teleport_entity("ada", Vector3(0,0,1.4))
    Runtime.teleport_entity("elias", Vector3(2.1,0,-1.6))
    for actor_id: String in ["ada", "elias"]:
        Runtime.entity_nodes[actor_id].animation_state_records["walk"]["speed"] = 0.82
        Runtime.entity_defs[actor_id]["character"]["walkSoundStride"] = 0.45
    Runtime.set_character_state("ada", "listen")
    Runtime.set_character_state("elias", "listen")
    Runtime.face_position("ada", Vector3(0,0,-2))
    Runtime.face_character("elias", "ada")
    await Runtime.cutscene_runner.cutscene_finished
    await Runtime.run_script("intro_handoff")
    return true
```

intro_tracking

Runs the original tracking helper during the opening; stops when its generation/cutscene is no longer active.

```
func run(_context: Dictionary = {}) -> Variant:
    # A true tracking shot: follow the live actor, not a second guessed walk timer.
    var generation: int = Runtime.runtime_generation
    var actor: AdventureActor = Runtime.entity_nodes.get("ada") as AdventureActor
    while is_instance_valid(actor) and Runtime.runtime_generation == generation and
        Runtime.cutscene_runner.running and Runtime.active_camera_id == "intro_follow":
        var subject: Vector3 = actor.global_position
        Runtime.set_camera_pose(subject + Vector3(2.4,1.9,3.5), subject + Vector3(0,1.2,0),43)
        await Runtime.get_tree().process_frame
    return true
```

intro_skip

Records the request to skip the introduction and asks the active cutscene to stop.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.variables["intro_skip_all"] = true
    Runtime.cutscene_runner.skip()
    return true
```

intro_foreground_on

Shows the prepared cinematic foreground overlay.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime._set_screen_overlay_visible(true)
    return true
```

intro_foreground_off

Hides the cinematic foreground overlay.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime._set_screen_overlay_visible(false)
    return true
```

intro_reverse_on

Shows reverse-angle architecture needed behind the conversation shots.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.set_entity_enabled("intro_reverse_wall", true)
    Runtime.set_entity_visible("intro_reverse_wall", true)
    Runtime.set_entity_enabled("intro_reverse_trim", true)
    Runtime.set_entity_visible("intro_reverse_trim", true)
    Runtime.set_entity_enabled("intro_reverse_rail", true)
    Runtime.set_entity_visible("intro_reverse_rail", true)
    return true
```

intro_reverse_off

Hides the temporary reverse-angle architecture.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.set_entity_enabled("intro_reverse_wall", false)
    Runtime.set_entity_visible("intro_reverse_wall", false)
```

```
    Runtime.set_entity_enabled("intro_reverse_trim", false)
    Runtime.set_entity_visible("intro_reverse_trim", false)
    Runtime.set_entity_enabled("intro_reverse_rail", false)
    Runtime.set_entity_visible("intro_reverse_rail", false)
    return true
```

exterior_sequence

Runs wind and the exterior cutscene, fades to the Workshop and activates its startup events.

```
func run(_context: Dictionary = {}) -> Variant:
    Runtime.variables["intro_skip_all"] = false
    Runtime.change_game_setting("cinematic.letterbox", 0.105)
    Runtime.project_data.settings.screenOverlay.enabled = false
    Runtime._set_screen_overlay_visible(false)
    Runtime.set_gameplay_hud_visible(false)
    Runtime.entity_nodes.ada.visible = false
    Runtime.show_gui("opening_controls")
    Runtime.play_sound("winter_wind", "exterior_wind")
    var wind: AudioStreamPlayer = Runtime.media_manager.sound_players.get("exterior_wind")
    if is_instance_valid(wind): wind.volume_db = -8.0
    await Runtime.start_cutscene("exterior_opening")
    if is_instance_valid(wind):
        var fade: Tween = Runtime.create_tween()
        fade.tween_property(wind, "volume_db", -40.0, 0.5)
    Runtime.hide_gui("opening_title")
    Runtime.hide_gui("opening_controls")
    await Runtime.fade_screen(true, "#080d15", 0.55)
    Runtime.stop_sound("exterior_wind")
    # Load behind the fade, then reveal before first-enter dialogue starts.
    Runtime.playable_character_states["ada"]["sceneId"] = "workshop"
    Runtime.playable_character_states["ada"]["position"] = [0,0,2.3]
    await Runtime.load_scene("workshop", false, "intro_establish", true)
    await Runtime.fade_screen(false, "#080d15", 0.55)
    await Runtime._activate_pending_startup_room(Runtime.runtime_generation)
    return true
```

intro_handoff

Makes natural completion and skipping leave the same playable state; stops voice/sound, releases attachments, restores actors, HUD and wide camera.

```
func run(_context: Dictionary = {}) -> Variant:
    # Normal finish and Skip share exactly the same playable end state.
    await DetachObject({"entityId":"ada"})
    Runtime.set_meta("clockmaker_handshake_phase", "finished")
    Runtime.stop_sound("opening")
    Runtime.stop_voice()
    Runtime.set_entity_visible("intro_reverse_wall", false)
    Runtime.set_entity_visible("intro_reverse_trim", false)
    Runtime.set_entity_visible("intro_reverse_rail", false)

    Runtime.hide_gui("opening_controls")
    Runtime.hide_gui("opening_title")
    Runtime.hide_gui("cinematic_dialogue")
    Runtime.message_session_serial += 1
    Runtime.project_data.settings.screenOverlay.enabled = true
    Runtime._apply_screen_overlay()
    Runtime._set_screen_overlay_visible(true)
    Runtime.change_game_setting("cinematic.letterbox", 0.0)
    for actor_id: String in ["ada", "elias"]:
        var actor: AdventureActor = Runtime.entity_nodes.get(actor_id) as AdventureActor
        if actor != null:
            actor.stop_moving()
            actor.animation_state_records["walk"]["speed"] = 1.0
            Runtime.entity_defs[actor_id]["character"]["walkSoundStride"] = 0.8
```

```
            actor.global_position = Vector3(-0.4,0,0.2) if actor_id == "ada" else Vector3(1.2,0,-1.2)
            actor.play_state("idle")
    Runtime.face_character("ada", "elias")
    Runtime.face_character("elias", "ada")
    Runtime.set_gameplay_hud_visible(true)
    Runtime.change_camera("wide")
    # Display guidance without holding the room-entry action queue.
    Runtime.show_message("Click objects to interact. Use the verb bar to Look or Pick Up. Esc opens
        saves and options.", 3.2, false)
    return true
```

interior_sequence

Starts the indoor opening or takes the skip/handoff path when the introduction was skipped.

```
func run(_context: Dictionary = {}) -> Variant:
    if bool(Runtime.variables.get("intro_skip_all", false)):
        await Runtime.run_script("intro_handoff")
    else:
        await Runtime.start_cutscene("opening")
    return true
```

intro_dialogue_orbit

Supports the slow directed orbit during conversation.

```
func run(_context: Dictionary = {}) -> Variant:
    var generation: int = Runtime.runtime_generation
    var actor: Node3D = Runtime.entity_nodes.get("elias")
    if actor == null: return false
    var center: Vector3 = actor.global_position + Vector3(0,1.2,0)
    var offset: Vector3 = Vector3(-1.4,0.60,3.0)
    var seconds: float = 0.0
    while seconds < 7.4 and Runtime.runtime_generation == generation and Runtime.cutscene_runner.running
        and Runtime.active_camera_id == "intro_elias":
        var fraction: float = clampf(seconds/7.4,0,1)
        var eased: float = fraction*fraction*(3.0-2.0*fraction)
        Runtime.set_camera_pose(center + offset.rotated(Vector3.UP,deg_to_rad(55.0)*eased),center,41.0)
        await Runtime.get_tree().process_frame
        seconds += Runtime.get_process_delta_time()
    return true
```

intro_face_pair

Keeps Ivo and Elias facing each other during the relevant opening phase.

```
func run(_context: Dictionary = {}) -> Variant:
    var generation: int = Runtime.runtime_generation
    var a: AdventureActor = Runtime.entity_nodes.get("ada") as AdventureActor
    var b: AdventureActor = Runtime.entity_nodes.get("elias") as AdventureActor
    if a == null or b == null: return false
    var a_start: float = a.rotation.y
    var b_start: float = b.rotation.y
    var a_end: float = a._local_yaw_for_world_direction(b.global_position-a.global_position)
    var b_end: float = b._local_yaw_for_world_direction(a.global_position-b.global_position)
    var seconds: float = 0.0
    while seconds < 0.75 and Runtime.runtime_generation == generation and
        Runtime.cutscene_runner.running:
        var t: float = clampf(seconds/0.75,0,1)
        t = t*t*(3.0-2.0*t)
        a.rotation.y = lerp_angle(a_start,a_end,t)
        b.rotation.y = lerp_angle(b_start,b_end,t)
        await Runtime.get_tree().process_frame
        seconds += Runtime.get_process_delta_time()
    if Runtime.runtime_generation == generation and Runtime.cutscene_runner.running:
```

```
        a.rotation.y = a_end
        b.rotation.y = b_end
    return true
```

intro_score_fade

Fades the opening score/soundscape instead of cutting it abruptly.

```
func run(_context: Dictionary = {}) -> Variant:
    var score_player: AudioStreamPlayer = Runtime.media_manager.sound_players.get("opening")
    if is_instance_valid(score_player):
        var fade: Tween = Runtime.create_tween()
        fade.tween_property(score_player,"volume_db",-40.0,3.0)
    return true
```

intro_voice_mix

Sets the opening voice mix for intelligible dialogue.

```
func run(_context: Dictionary = {}) -> Variant:
    var score_player: AudioStreamPlayer = Runtime.media_manager.sound_players.get("opening")
    if is_instance_valid(score_player):
        score_player.volume_db = -16.0
    return true
```

intro_handshake

Moves the actors to nearby marks, starts compatible gestures, tests hand attach-point contact, briefly links the hands, then detaches safely.

```
func active(generation: int) -> bool:
    return Runtime.runtime_generation == generation and Runtime.cutscene_runner.running and
        Runtime.cutscene_runner.active_cutscene_id == "opening" and not
        Runtime.cutscene_runner.skip_requested

func run(_context: Dictionary = {}) -> Variant:
    var generation: int = Runtime.runtime_generation
    var ivo: AdventureActor = Runtime.entity_nodes.get("ada") as AdventureActor
    var elias: AdventureActor = Runtime.entity_nodes.get("elias") as AdventureActor
    if ivo == null or elias == null: return false
    var direction: Vector3 = (elias.global_position-ivo.global_position).normalized()
    var center: Vector3 = (ivo.global_position+elias.global_position)*0.5
    var ivo_mark: Vector3 = center-direction*0.3275
    var elias_mark: Vector3 = center+direction*0.3275
    Runtime.set_meta("clockmaker_handshake_phase", "approach")
    Runtime.move_entity_to("ada", ivo_mark, 0.85, 0.005)
    Runtime.move_entity_to("elias", elias_mark, 0.85, 0.005)
    var seconds: float = 0.0
    while seconds < 1.05 and active(generation):
        await Runtime.get_tree().process_frame
        seconds += Runtime.get_process_delta_time()
    if not active(generation): return false
    ivo.stop_moving()
    elias.stop_moving()
    var ivo_start: Vector3 = ivo.global_position
    var elias_start: Vector3 = elias.global_position
    # Settle the final navigation tolerance over several frames, never snap bodies
    # into a hand socket from across the room.
    if ivo_start.distance_to(ivo_mark) > 0.15 or elias_start.distance_to(elias_mark) > 0.15:
        Runtime.log_message("Handshake approach could not reach its marks.")
        return false
    seconds = 0.0
    while seconds < 0.20 and active(generation):
        var fraction: float = smoothstep(0.0, 0.20, seconds)
```

```
        ivo.global_position = ivo_start.lerp(ivo_mark, fraction)
        elias.global_position = elias_start.lerp(elias_mark, fraction)
        await Runtime.get_tree().process_frame
        seconds += Runtime.get_process_delta_time()
    if not active(generation): return false
    ivo.global_position = ivo_mark
    elias.global_position = elias_mark
    Runtime.face_character("ada", "elias")
    Runtime.face_character("elias", "ada")
    Runtime.set_character_state("ada", "handshake")
    Runtime.set_character_state("elias", "handshake")
    Runtime.set_meta("clockmaker_handshake_phase", "reach")
    var linked: bool = false
    var released: bool = false
    seconds = 0.0
    while seconds < 3.65 and active(generation):
        if seconds >= 0.95 and not linked and not released:
            var touching: bool = await AttachPointsTouching({"entityId":"ada",
                "point":"right_hand_grip", "targetId":"elias", "targetPoint":"right_hand_grip",
                "tolerance":0.012})
            if touching:
                linked = await AttachPoints({"entityId":"ada", "point":"right_hand_grip",
                    "targetId":"elias", "targetPoint":"right_hand_grip", "alignRotation":false,
                    "suspendMovement":true})
                Runtime.set_meta("clockmaker_handshake_phase", "clasp")
            elif seconds > 1.15:
                released = true
                Runtime.log_message("Handshake contact missed; keeping the authored animation without
                    snapping.")
        if seconds >= 2.5 and not released:
            await DetachObject({"entityId":"ada"})
            released = true
            Runtime.set_meta("clockmaker_handshake_phase", "release")
        await Runtime.get_tree().process_frame

        seconds += Runtime.get_process_delta_time()
    if Runtime.runtime_generation == generation:
        await DetachObject({"entityId":"ada"})
        Runtime.set_meta("clockmaker_handshake_phase", "finished")
    return linked
```