

TOOL

ERAGANX

LOADING SCREEN BUILDER PRO

Design, Customize, Animate, Load, Play.



ERAGANX / UNITY PACKAGE GUIDE

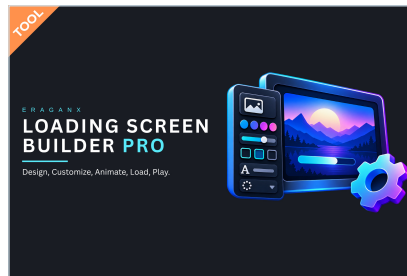
The complete guide from the beginning

Design, scene integration and runtime C# usage.
Step by step, using your actual Builder screens.

- ENGLISH EDITION
- SCREENSHOT GUIDE
- C# EXAMPLES

START HERE

How to use this guide



This manual explains the supplied Builder screenshots and current package source. Follow the first scene exercise, then use the detailed workspace walkthroughs and C# examples you need. Screenshot values are examples; incomplete fields are called out explicitly.

Foundation	3 - 6
Library & First Load	7 - 9
Visual Designer	10 - 15
Post-Loading & Devices	16 - 20
Content & Backgrounds	21 - 25
Behavior, Transitions & Audio	26 - 30
Scene Integration	31 - 35
Preview & Validation	36 - 37
Runtime C# Examples	38 - 45
API Quick Reference	46 - 48
Troubleshooting	49 - 50
Completion	51
Sources & Version Scope	52 - 54

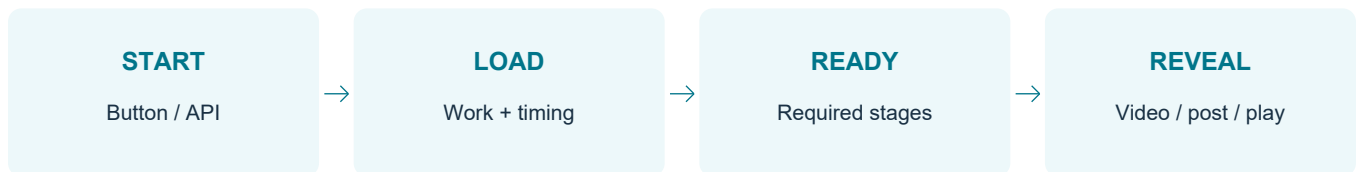
Suggested reading path New users: install, try the gallery and finish your first scene load. Existing users: go to the workspace you are editing. Runtime scripts and API references follow the visual walkthroughs.

The package namespace is LoadingScreenBuildPro. This revision uses 28 supplied UI screenshots and the supplied cover artwork.

01 / FOUNDATION

How the loading flow works

A saved loading design defines the visuals and the rules for showing them. Your game starts a request; the package covers the current scene, runs the work, waits for required readiness, and reveals gameplay.



Piece	What it controls
LoadingLayout	Loading layers, local theme, behavior, content, media, motion and responsive profiles.
Scene connection	Which design is used for a destination. Unconnected scenes can use the project default.
Request / handle	One execution, its progress and status, controls, and final result.
PostLoadingLayout	Separate ready-screen visuals and optional countdown after required work completes.

Progress, readiness and completion are different

The scene operation can be prepared for activation while the destination is still not ready for play. Required Game Ready checkpoints can keep the overlay visible after activation. A post screen or remaining video can then keep the flow active even when progress is 100%.

Start explicitly A saved connection selects a design. It does not start a load. Use the Builder's no-code trigger or `LoadingScreenBuilder.LoadScene(...)`. Arbitrary `SceneManager` calls do not automatically pass through this package.

Check You should be able to identify the destination, its design, its start action, and the event that makes gameplay ready.

01 / FOUNDATION

01. Install and open the Builder

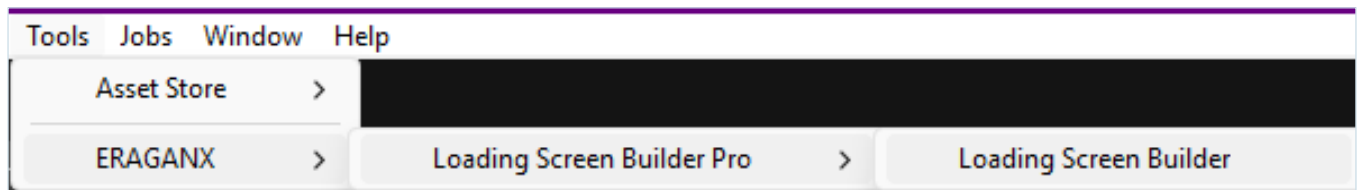


Figure 01 / The exact Tools menu path for opening the authoring window.

- 1 Import the supplied .unitypackage using **Assets > Import Package > Custom Package**, or copy the complete package folder and its .meta files into your project.
- 2 Keep the folder at **Assets/ERAGANX/Loading Screen Builder Pro/**. The current editor uses this path for package assets.
- 3 Wait for Unity to finish importing and compiling. Resolve red Console compilation errors before opening the tool.
- 4 Ensure uGUI and TextMeshPro are available. Import **TMP Essential Resources** through Unity's TextMeshPro setup if they are not already in the project.
- 5 Choose **Tools > ERAGANX > Loading Screen Builder Pro > Loading Screen Builder**, as shown above.

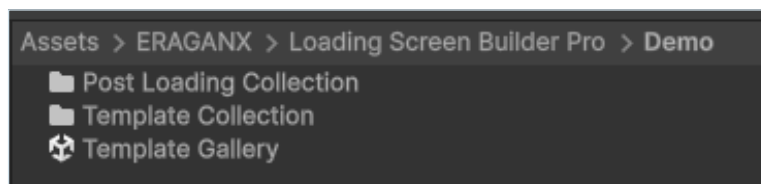


Figure 02 / Demo contains the loading collection, post-loading collection and Template Gallery scene.

Use	Requirements
Authoring and default runtime UI	UI Toolkit editor; uGUI and TextMeshPro runtime. Source target: Unity 6000.3.13f1.
Optional Input System adapter	Enabled when the Input System package is available. The core has no mandatory DOTween, Addressables or Parallax Pro dependency.
Custom assembly definition	Reference LoadingScreenBuildPro.Runtime in your own runtime asmdef.

Check The Builder opens, Library shows available loading assets, and a selected design can render text. Opening the window does not generate demos or start playback.

01 / FOUNDATION

02. Understand the asset commands

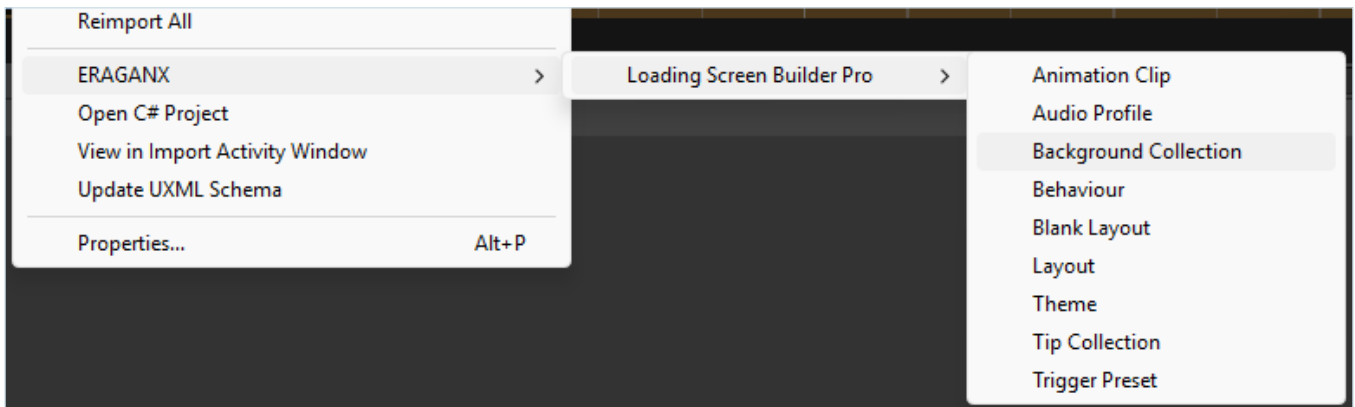


Figure 03 / Project context menu: ERAGANX > Loading Screen Builder Pro. The expanded menu shows reusable asset types.

Right-click in the Project window to reach the shown asset commands. You can also begin with **Library > + New layout**, **Blank layout** or **Quick Setup**. Use the Library workflow for a normal first design.

Command	Creates / stores
Layout / Blank Layout	A loading design with a starting composition or a blank canvas.
Theme / Behaviour	Reusable defaults for appearance or loading policy.
Tip / Background Collection	Reusable content and background settings.
Audio Profile / Animation Clip	Reusable sound-track settings or property-animation data.
Trigger Preset	Reusable start-action configuration.

One design, one main file New LoadingLayout assets embed their local theme, behavior, tips, backgrounds, audio, motion, profiles and drafts. You do not need a separate settings asset for every edit. Fonts, sprites, clips, prefabs and the project registry remain separate.

Screenshot context The context menu is shown over the Template Gallery scene. The underlying scene view and warning strip are not steps you need to reproduce.

01 / FOUNDATION

03. Try the Template Gallery

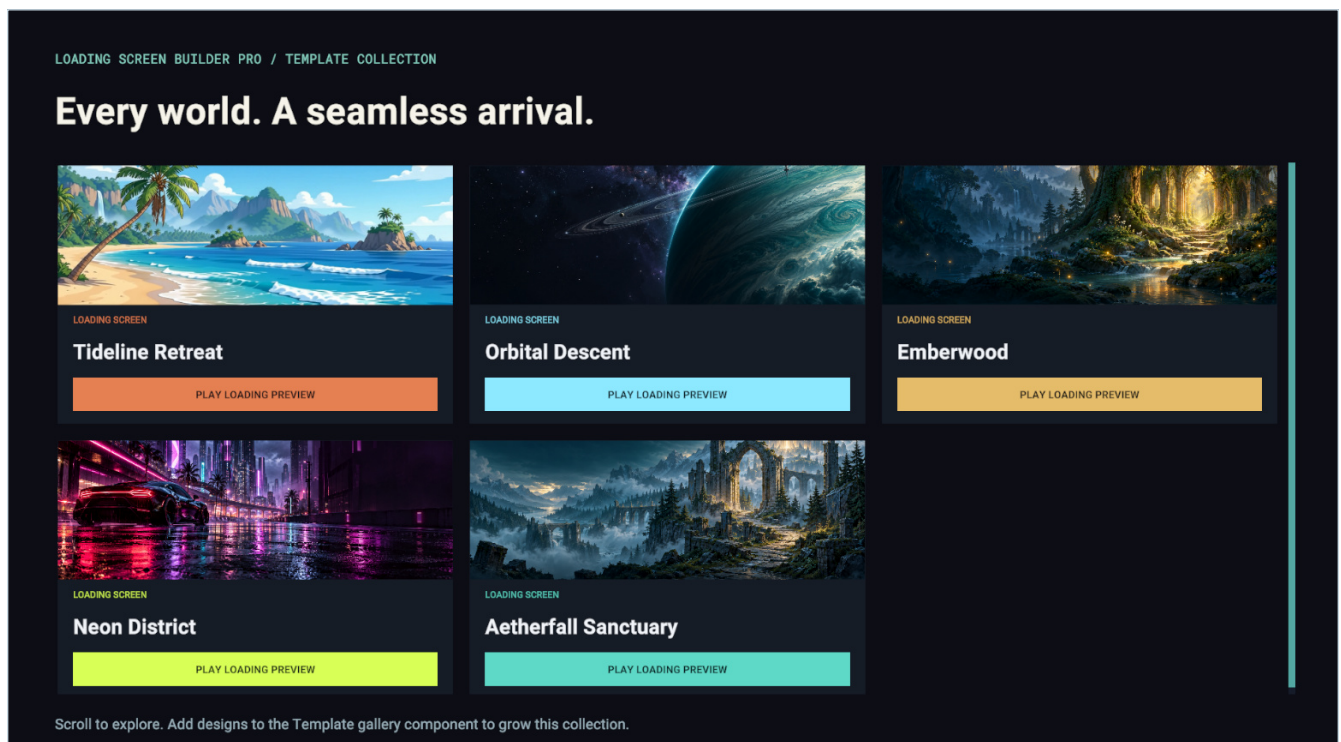


Figure 04 / The supplied gallery presents loading designs as preview cards. Scroll to see all available entries.

- 1 In the Project window open **Assets/ERAGANX/Loading Screen Builder Pro/Demo/Template Gallery.unity**.
- 2 Enter Play Mode, then select a **PLAY LOADING PREVIEW** button. The gallery's default loading preview duration is eight seconds and does not change scenes.
- 3 Try Timeline Retreat, Orbital Descent, Emberwood, Neon District and Aetherfall Sanctuary. Scroll for the post-loading examples, including Aurora Ready and Momentum Ready.
- 4 Exit Play Mode. Open a design in Library and use **Save as > Independent copy** before adapting a supplied template.

Design	What to examine
Tideline / Orbital	Video and poster fallback / radial progress with mission text.
Emberwood / Neon	Independent forest layers / segmented progress and neon artwork.
Aurora / Momentum Ready	Separate post-loading screens with optional audible or silent countdown.

Check A preview returns to the gallery after its complete flow. Add the demo scene to the active build scene list only if you want to include the gallery in a player build.

04. The Library workspace

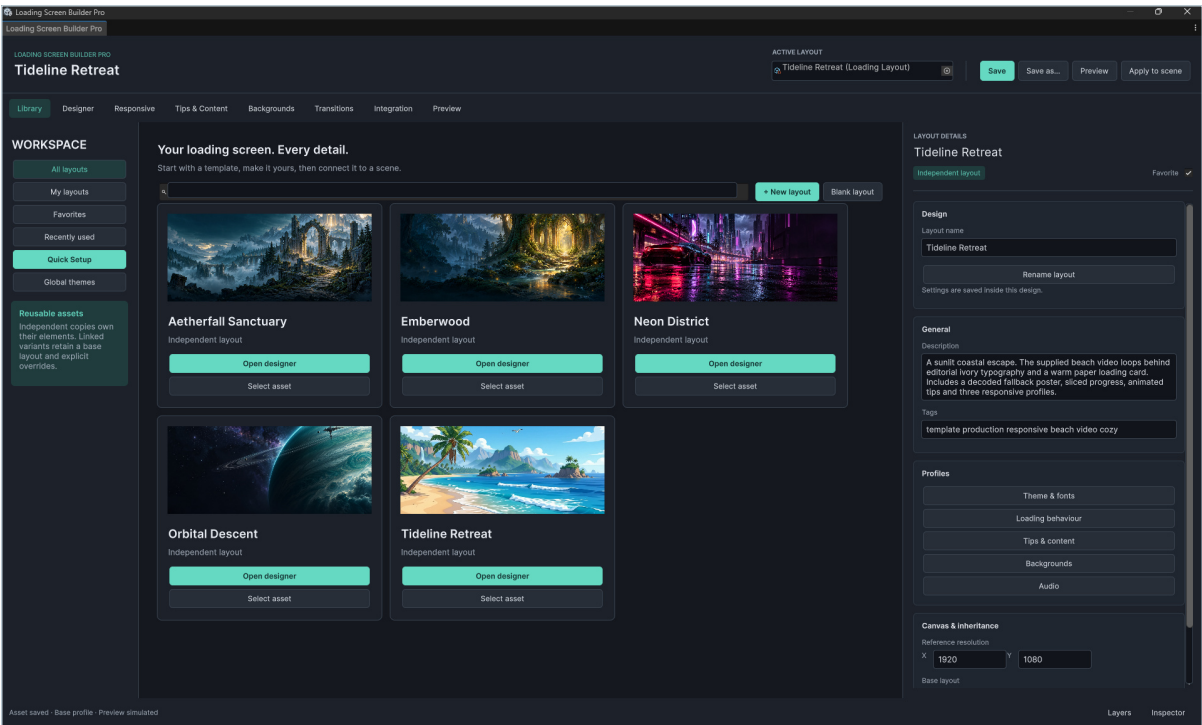


Figure 05 / Library: filters on the left, layout cards in the center, and details for the active layout on the right.

Area shown	How to use it
All layouts / My layouts / Favorites / Recently used	Choose a library view. Use the search field to narrow the card list by name or tags.
Open designer	Select the card's design and open its visual document.
Select asset	Select the saved LoadingLayout in Unity's Project window/Inspector.
+ New layout / Blank layout	Create a new design; Blank is useful when building the composition yourself.
Quick Setup / Global themes	Use the guided template/behavior/review steps or open the active design's theme controls.
Active Layout / Save / Save as...	Choose the document, save edits, or create a copy/linked variant.

Read the selected design The screenshot's active layout is Tideline Retreat. The card grid can show other designs at the same time; the right-hand inspector follows the active selection.

Library details, copies and ownership

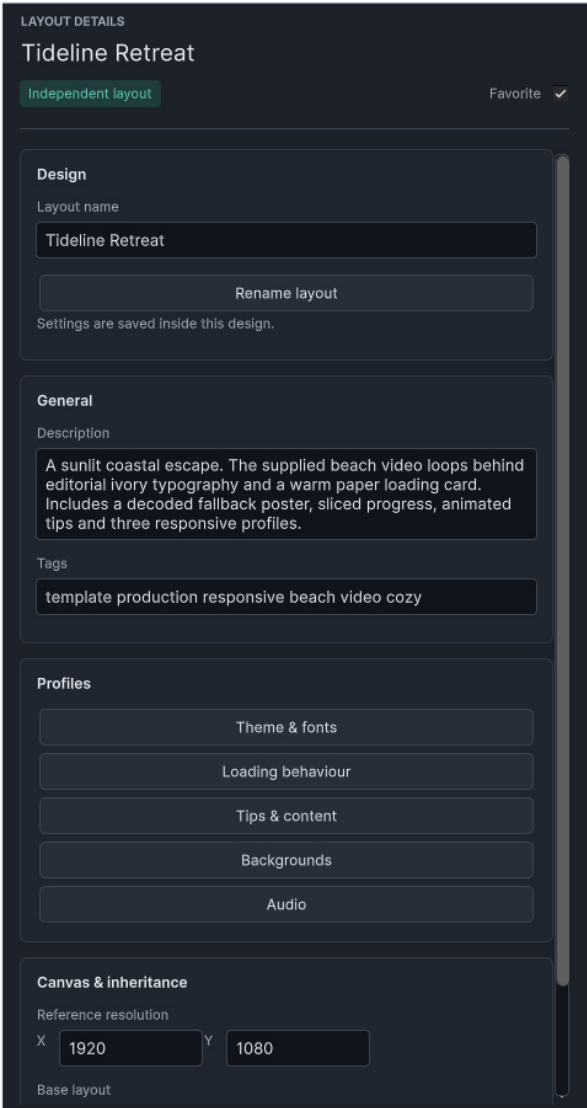


Figure 06 / Close-up of the selected layout's details.

Start with a copy

Choose **Save as... > Independent copy** when you want your own design. Choose **Linked variant** when you deliberately want inheritance from a base layout.

Control	Purpose
Favorite	Include this asset in Favorites.
Layout name / Rename layout	Rename while preserving its GUID and references.
Description / Tags	Describe and search your design.
Profiles	Open theme/fonts, behavior, tips, backgrounds or audio.
Canvas & inheritance	Set reference resolution and a base layout.

In a linked variant, base values remain inherited until an explicit override is stored. Resetting an override returns to inheritance. Independent copies own their editable composition.

Save and delete Save persists the design. Scene objects still need a scene save. Delete selected layout reports dependencies and asks for confirmation. Backups are stored under the project-relative work/design-backups/<timestamp>/ folder; preserve both asset and .meta when restoring.

Check Rename your copy, save it, switch to another layout and return. Your copy should retain its own name and edits.

02 / LIBRARY & FIRST LOAD

Guided exercise: your first scene load

Complete this exercise before adding custom stages, video continuation or a post-loading countdown. Use a menu scene with a UI Button and a saved destination scene.

- 1 Open a supplied design in Library and choose **Save as > Independent copy**. Rename your copy.
- 2 Open **Transitions > Mode & activation**. Choose **Both**, **Scene Policy = Single**, **Progress Source = Operation**, and **Minimum Time = 2**. Leave optional Game Ready off for this exercise.
- 3 Open **Integration > Scene setup**. Assign your destination Scene asset, then choose **Save scene connection**. Ensure the destination is enabled in Unity's active Build Profile scene list.
- 4 Open your menu scene. Select the UI Button in the Hierarchy.
- 5 Open **Integration > No-code trigger**. Choose **Kind = Button**, confirm the destination, choose **Use selected scene object**, then **Connect loading action**.
- 6 Save the design and your menu scene. Enter Play Mode and press the Button.
- 7 Verify that the overlay covers the old scene, displays progress, respects the minimum time and reveals the destination.

If this happens	Check
No request starts	Button/EventSystem input, enabled/configured LoadingTrigger, and the selected scene object.
Destination fails	Its saved name/path and enabled runtime build-scene entry.
Wrong visual design	Explicit override, scene connection, then project default.
Overlay stays up	Unintended required stage/Game Ready or a player Continue policy.

Alternative start action After the destination connection is saved, your own method can call `LoadingScreenBuilder.LoadScene("YourSceneName")`. Use that actual scene name, a full asset path or an integer build index.

Check A connection and a start action must both exist. A connection on its own does not load the scene.

03 / VISUAL DESIGNER

05. The complete Designer layout

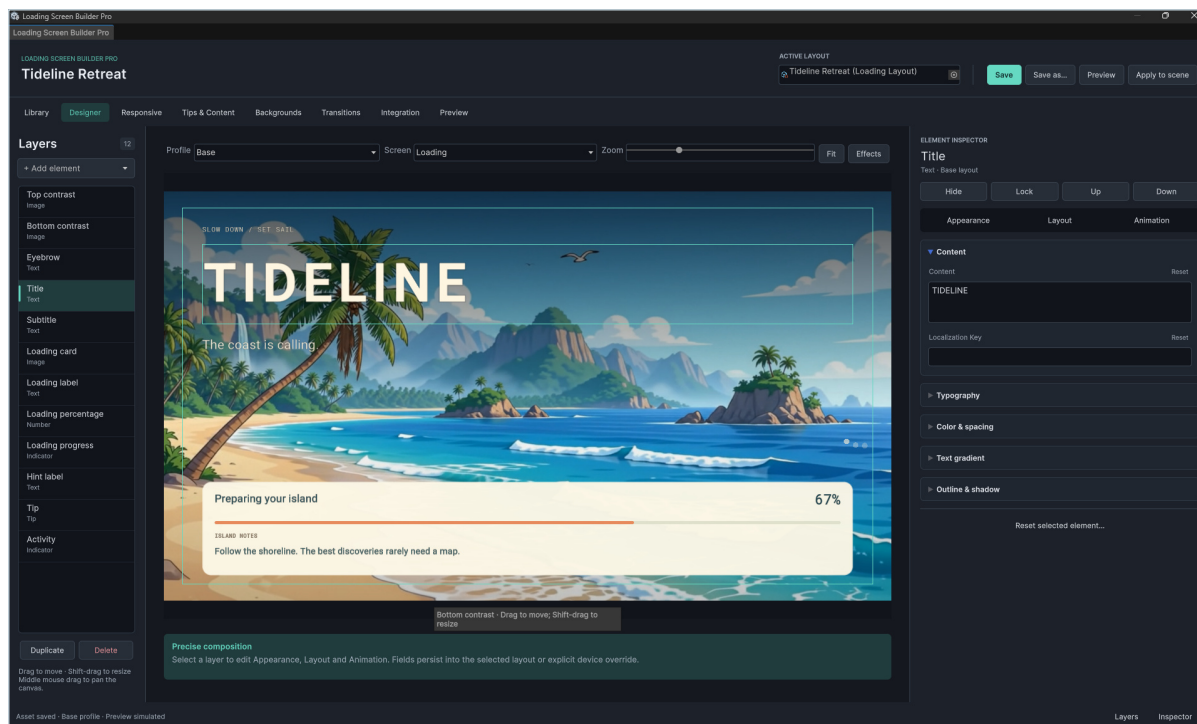


Figure 07 / Designer with the Title layer selected in Tideline Retreat.

Area	What to do
Layers list	Select a layer. Add element creates content; Duplicate/Delete act on the selection.
Canvas	Drag an unlocked element to move it. Hold Shift before pointer-down and drag to resize.
Profile	Choose Base or a device override before editing.
Screen	Choose Loading or the separate Post-Loading visual document.
Zoom / Fit / Effects	Inspect the viewport, recenter it, or browse indicator styles.
Element Inspector	Use Appearance, Layout and Animation for the selected layer.
Hide / Lock / Up / Down	Control visibility, editing lock and drawing order.

Middle mouse drag pans the complete preview. This changes your viewport, not the background or element data. Ctrl+S saves while the Builder handles the shortcut; Ctrl+D and Delete apply when the Designer canvas has focus.

Check Select Title, make one small position change, then Undo. The selected layer should redraw and the change should be reversible.

03 / VISUAL DESIGNER

06. Edit text content

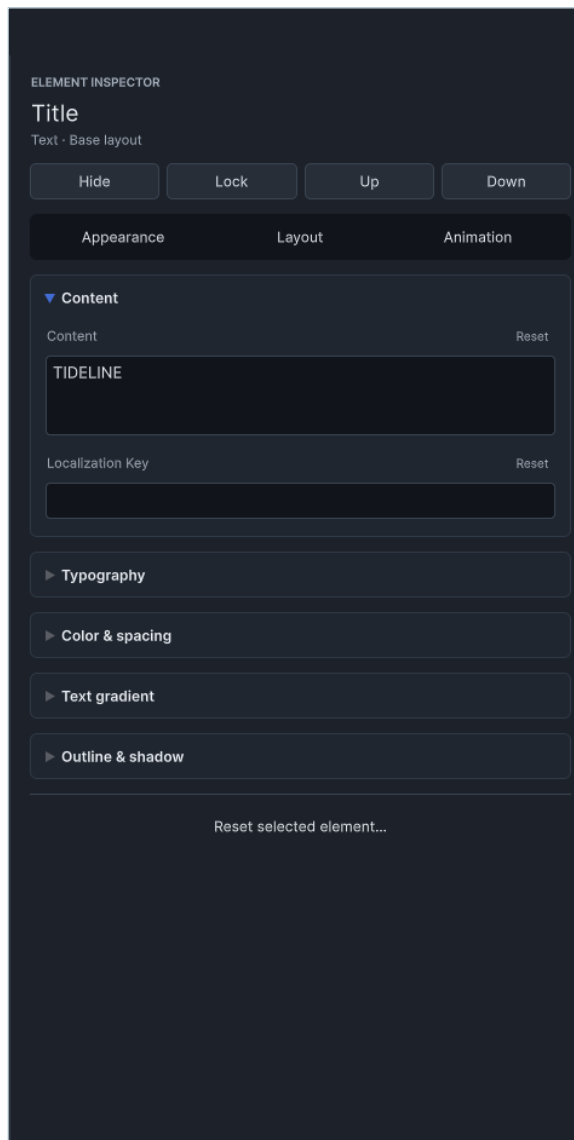


Figure 08 / Appearance > Content for the Title text layer.

Follow the selected layer

The inspector identifies **Title**, type **Text**, in the **Base layout**. Its controls edit that layer, not every text object in the design.

- 1 Open **Appearance > Content**.
- 2 Replace **TIDELINE** with your own title in Content.
- 3 Leave Localization Key empty for direct text, or supply the key your request's localization callback will resolve.
- 4 Save and review the title in the canvas.

Control	Meaning
Content	Authored copy and fallback when localization is unavailable.
Localization Key	Optional lookup key, not an automatic translation.
Reset beside a field	Reset that field/override; inspect its scope before using it.
Reset selected element...	A broader reset action for this selection.

Separate text roles Use Text for a title/status, Number for percentage/fraction/stage formatting, and Tip for the currently selected collection entry. A Countdown layer reads the post-screen countdown.

Check Your title changes while the subtitle and hint remain unchanged. If another item changes, verify the selected layer and the active Screen/Profile.

03 / VISUAL DESIGNER

Typography, color and text gradients

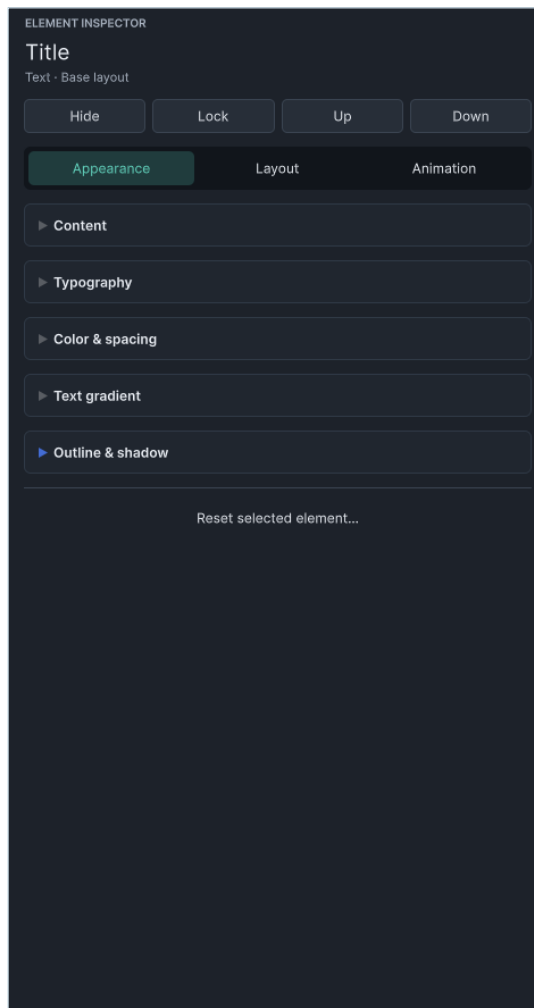


Figure 09 / The Appearance foldouts. Expand the group you need.

Foldout shown	Settings to inspect
Typography	Theme/local TMP font, style, size, auto-size range, alignment, wrapping and overflow.
Color & spacing	Theme/local color plus letter and line spacing.
Text gradient	Enable a horizontal, vertical or four-corner blend.
Outline & shadow	Contrast styling for text over artwork.

Theme or local font

Use Theme Font keeps the element on its design default. Assigning a local font disables that choice for this element. Use a suitable TMP fallback chain for additional characters.

Keep text readable

Set bounds for the longest intended title or hint. Use wrapping and a sensible auto-size minimum; a very small fitted title still needs a layout change.

Gradient mode	Colors used
Horizontal / Vertical	Top Left to Top Right / Bottom Left to Top Left.
Four Corners	All four colors. Gradient values multiply the base color/tint; white preserves them directly.

Check Review a longer title and Reduced Motion. Static SDF template fonts cover the supplied English copy; assign appropriate fonts/fallbacks for your languages.

03 / VISUAL DESIGNER

07. Layout, anchors and size

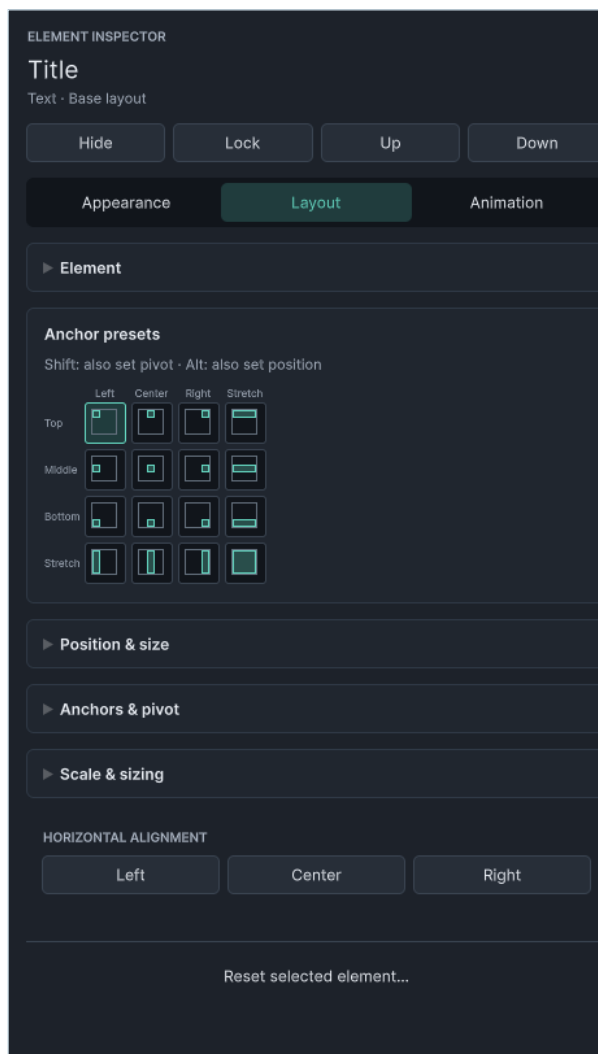


Figure 10 / Layout tab: anchor presets, rectangle groups and horizontal alignment.

Choose an anchor

Fixed anchors position an element relative to one point. Stretched anchors describe spacing between edges. The pivot is the point used for positioning and scaling.

- 1 Choose an anchor preset. A normal click preserves the current visible rectangle.
- 2 Hold **Shift while clicking** to set the pivot too; hold **Alt while clicking** to set position too.
- 3 Expand Position & size and enter precise values.
- 4 Use Left/Center/Right for horizontal alignment, then inspect a phone profile.

Foldout	Use
Element	Element-level layout settings.
Position & size	Position/dimensions or stretched-edge margins.
Anchors & pivot	Advanced normalized anchors and pivot.
Scale & sizing	Scale, pixel/parent-percent units, min/max and aspect lock.

Fixed versus stretched Fixed axes show Pos X/Width or Pos Y/Height. Stretched axes show Left/Right or Top/Bottom. Keep important text/buttons inside the safe area; full-screen decorative artwork can extend behind it.

Check A bottom-anchored progress row should remain near the bottom when the aspect changes, with readable inset spacing.

03 / VISUAL DESIGNER

08. Animate and combine effects

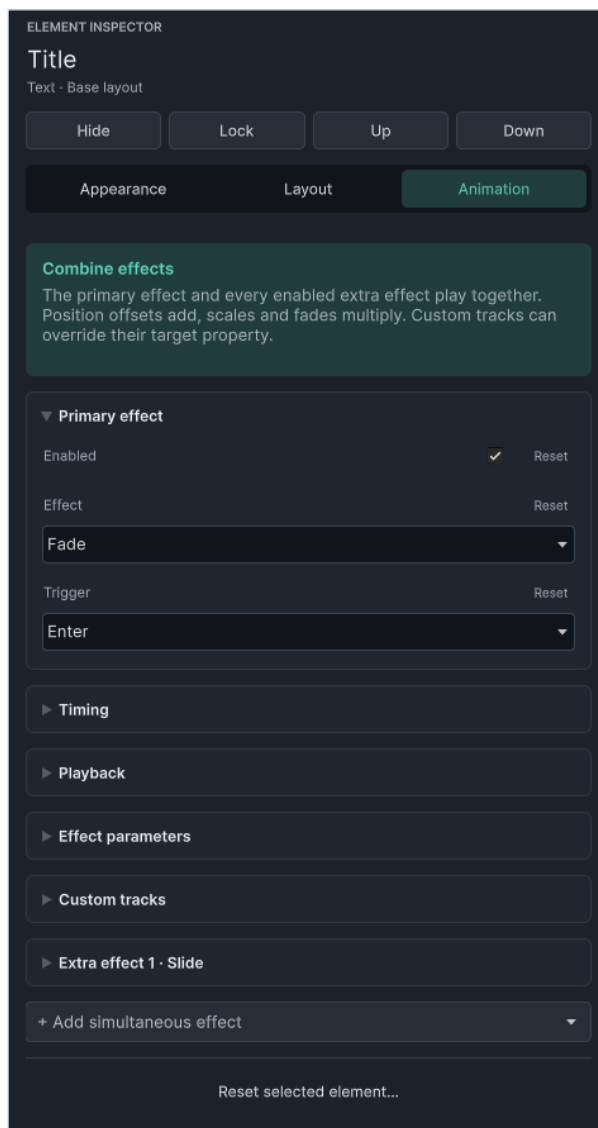


Figure 11 / The Title combines an Enter Fade with an extra Slide effect.

Reproduce this combination

- 1 Open **Animation > Primary effect** and enable it.
- 2 Choose **Effect = Fade, Trigger = Enter**.
- 3 Expand Timing to choose duration/delay/speed/easing.
- 4 Use **Add simultaneous effect > Slide**. Give it the intended trigger and timing.
- 5 Preview, then test Reduced Motion.

Group	Meaning
Playback	Loop, repeat, reverse, ping-pong and unscaled-time options.
Effect parameters	Effect-specific distance, typewriter speed/pauses or character scope.
Custom tracks	Curves targeting opacity, position, scale, rotation or RGB properties.

Position offsets and rotation add. Scale and opacity multiply. Explicit custom tracks can override their target property. Each effect keeps its own timing.

Triggers Enter, Loading, Ready and Exit are distinct moments. Use Enter/Ready for post-screen effects: a post screen does not emit Loading. Typewriter, Wave and Color Sweep can be combined with other effects.

Check Disable the extra Slide without deleting the Title. The Fade should still run.

03 / VISUAL DESIGNER

Indicators, numbers and reusable themes

Open **Effects** from Designer. **Add as new indicator** adds a separate layer; **Replace selected style** changes the selected Indicator. A spinner and progress bar can coexist without replacing each other.

Indicator family	Styles and main controls
Measured progress	Linear, Radial, Segmented, Logo Reveal, Wave Fill and Steps. Choose direction/origin, geometry, count/gaps and suitable sprites.
Activity	Spinner, Dots, Wave Dots, Orbit, Pulse, Bounce, Shimmer and Sprite Sequence. Choose speed, radius/amplitude, spacing, frames/FPS as applicable.
Text / custom	Text Only uses text/number binding. Custom uses a RectTransform prefab implementing <code>ILoadingIndicator</code> .

Format a separate Number layer

Setting	How to use it
Percentage / Fraction / Stage	Show 0-100, a configured fraction total, or stage/status copy.
Decimals / Rounding	Choose zero to two decimals and Floor/Round/Ceiling.
Prefix / Suffix / Show Stage	Add units or stage text. Clear the default % suffix when it does not suit a fraction.
Unknown Text	A useful message when progress is unavailable. Final full values are reserved for readiness.

Themes and sprite quality

Open **Global themes** or Theme & fonts for design defaults: colors, TMP font/body size and Reduced Motion. Local overrides remain explicit. For a resizable Linear bar, define nine-slice borders on its fill/track sprites. A suitable source size and filtering are still needed.

Check The indicator style changes appearance; Progress Source supplies its value. Decorative shimmer or manual progress never completes unfinished work.

04 / POST-LOADING & DEVICES

09. Switch to Post-Loading

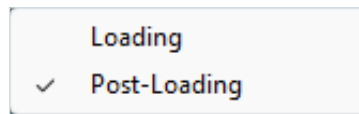


Figure 12 / Designer's Screen menu selects one of two visual documents.

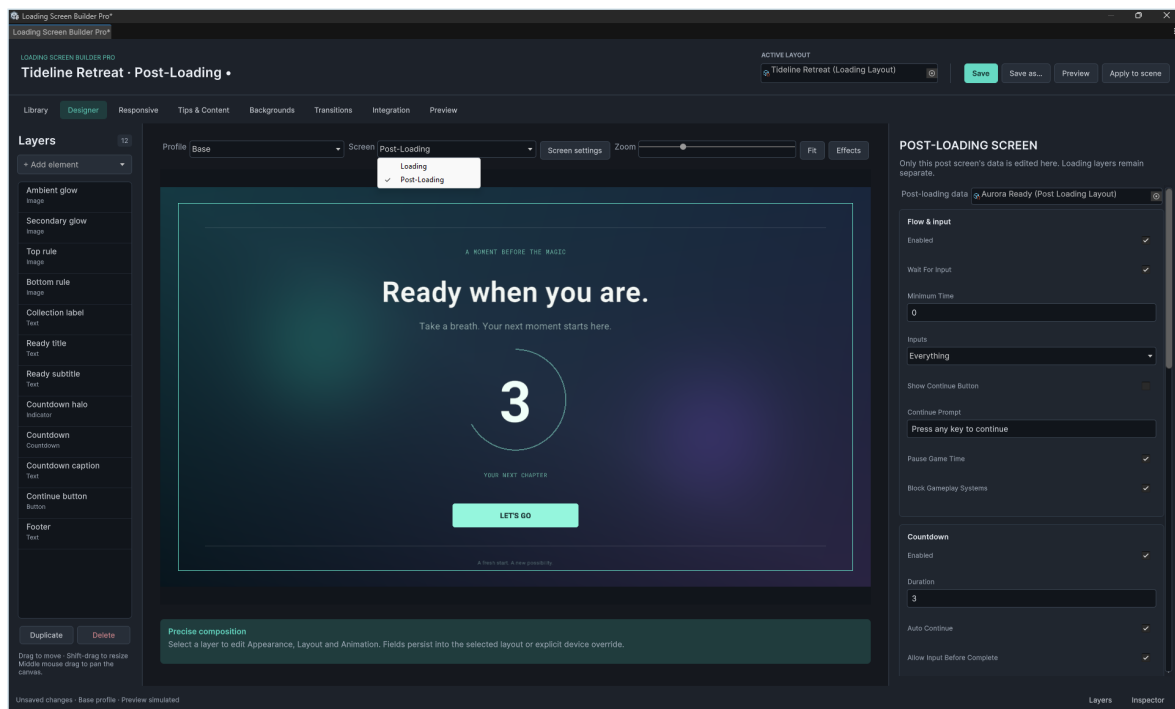


Figure 13 / Post-Loading selected, with Aurora Ready assigned. Its layers are separate from the loading layers.

- 1 Open a loading design, then change **Designer > Screen** from Loading to **Post-Loading**.
- 2 Choose/create post-loading data in Screen settings. Assign a **PostLoadingLayout**, such as Aurora Ready.
- 3 Edit the ready title, countdown and Continue Button layers in this document.
- 4 Switch back to Loading to confirm that its original composition remains separate. Save your edits.

Data ownership New post data can be a subasset in its owner LoadingLayout file. It is still a distinct PostLoadingLayout type. Loading Library/pickers accept LoadingLayout; post pickers accept only PostLoadingLayout.

Check The document title includes Post-Loading and the layer list changes to ready-screen elements. Profile, selection, zoom and pan are maintained separately for each screen.

04 / POST-LOADING & DEVICES

10. Configure the post-loading flow

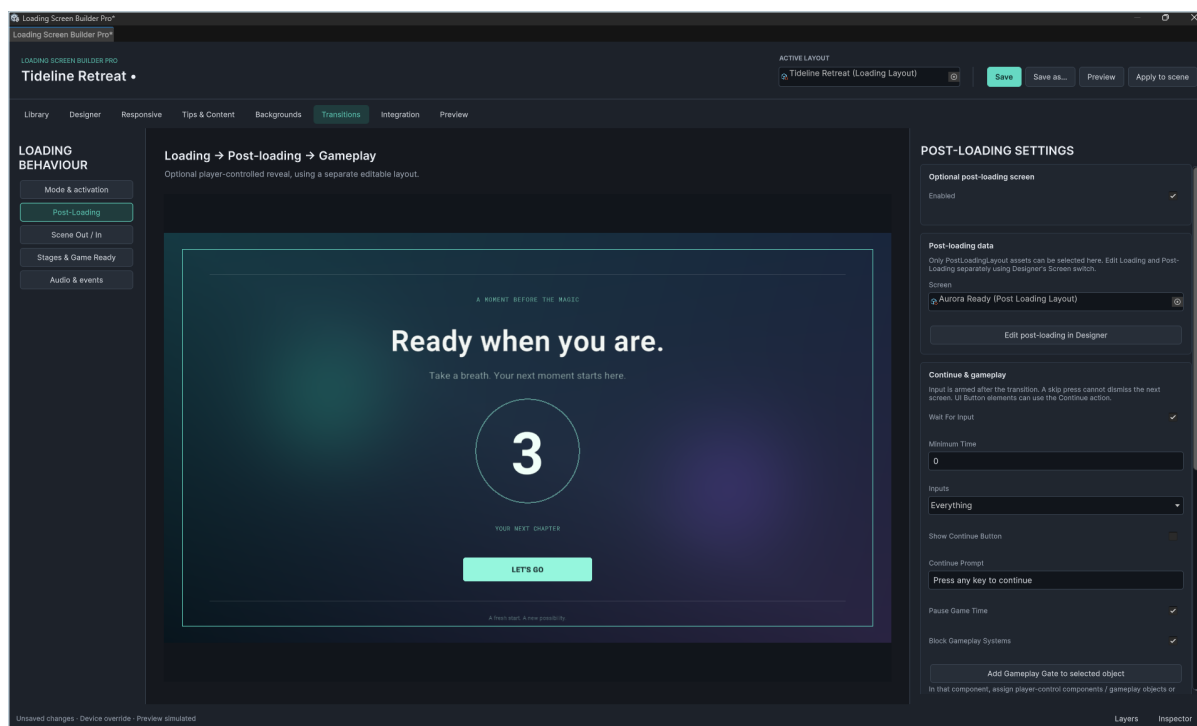


Figure 14 / Transitions > Post-Loading: enable the optional screen, assign post data and choose how gameplay is held.

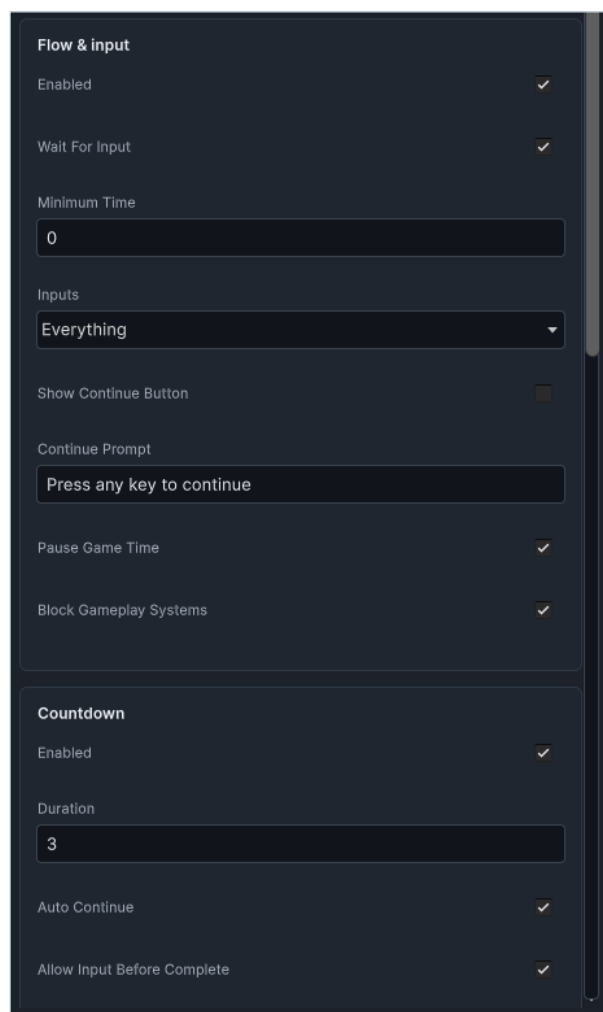
Control shown	Meaning
Enabled / Screen	Turn on the flow and assign a PostLoadingLayout. Edit post-loading in Designer opens its visuals.
Wait For Input / Minimum Time	Choose a player wait and the minimum post-screen duration.
Inputs	Filter keyboard, mouse, touch and gamepad; Button Only uses UI/API actions.
Show Continue Button / Continue Prompt	Generate an action, or use your own Button layer with Continue.
Pause Game Time	Capture and hold time scale, then restore the original value.
Block Gameplay Systems	Hold only assigned LoadingGameplayGate targets. Initialization required for readiness must keep running.

Scroll for Loading-to-Post and Post-to-Gameplay transition settings. In Integration a destination's Post control can use design policy, disable it or supply a custom override. LoadScene/CreateSceneRequest resolve those destination overrides; the current no-code LoadingTrigger uses design-level post settings.

Check The ready screen appears only after required loading/readiness. Continue does not bypass unfinished work.

04 / POST-LOADING & DEVICES

11. Countdown and Continue rules



Use the shown settings

In this screenshot **Wait For Input** is on and the screen has an authored Continue button. **Allow Input Before Complete** is checked, so the countdown is not a mandatory lock.

- 1 Enable Countdown and choose Duration.
- 2 Turn **Auto Continue** on to leave after completion, subject to Minimum Time.
- 3 Turn **Allow Input Before Complete** off when the player must wait for zero.
- 4 Select a Countdown layer and style its text/appearance.

Other countdown controls	Use
Format / Completed Text	Use {seconds} or {time}; choose the zero-time copy, such as GO!.
Effect / Intensity	None, Pop, Fade, Slide or Pulse.
Play Sounds / clips / level	Opt into tick and completion sounds; these controls may require scrolling.

Figure 15 / The shown countdown is enabled, lasts 3 seconds, auto-continues and allows early input.

Timer ownership The countdown begins after entry and runs unscaled. Minimum Time remains enforced. A locked countdown rejects early Continue. With Auto Continue off and Wait For Input on, the screen waits for input after zero.

Sound Countdown audio is separate from ordinary tracks. It respects post-profile/session Master/SFX/mute. Aurora Ready demonstrates optional Soft Tick/Ready Chime; Momentum Ready is a silent example.

12. Responsive device profiles

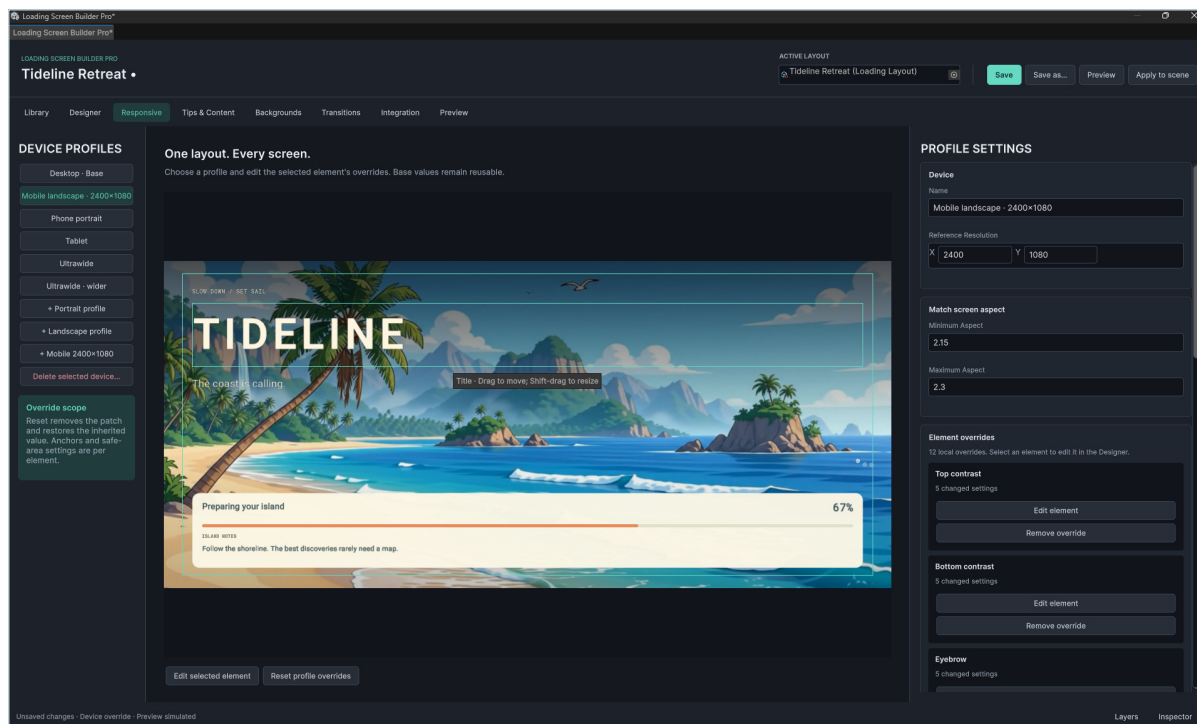


Figure 16 / Responsive: Mobile landscape 2400 x 1080 is selected, with a 2.15-2.30 aspect range.

- 1 Choose **Desktop - Base** and finish the reusable base composition first.
- 2 Select an existing profile or use **+ Portrait profile**, **+ Landscape profile**, or **+ Mobile 2400x1080**.
- 3 Set a descriptive Name, Reference Resolution and aspect range. Aspect is width divided by height; 2400 / 1080 is about 2.22.
- 4 Select an element and choose **Edit element**, or return to Designer with that profile selected.
- 5 Adjust only the values that need to differ on that device family. Save and compare with Base.

Control	Effect
Remove override	Remove an element's profile patch and restore inherited values.
Reset profile overrides	Return that profile's patches to inherited/base settings.
Delete selected device...	Remove the selected profile/overrides after confirmation. Base cannot be deleted.

Runtime selection A profile is chosen when a request starts. Orientation changes during that request do not select another profile. Check safe areas and intended player resolutions.

Device overrides - read the inspector

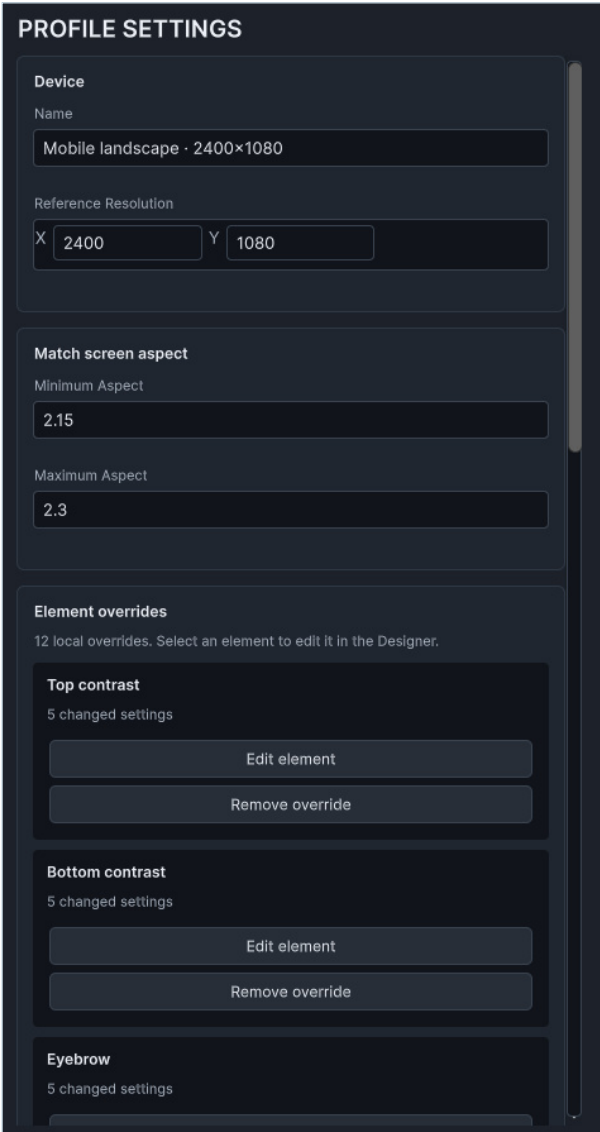


Figure 17 / Close-up: name, reference resolution, aspect range and per-element override entries.

What the screenshot means

Shown value	Interpretation
2400 x 1080	Authoring reference space for this profile, not a guarantee of physical pixels on every device.
Minimum 2.15 / Maximum 2.3	Match this landscape aspect band.
12 local overrides	Twelve elements have profile-specific patches.
Top contrast: 5 changed settings	This element has five overridden fields, not five extra layers.
Edit element	Open its selected-profile values in Designer.
Remove override	Discard this element's profile patch.

The left list can contain additional portrait, tablet and ultrawide profiles. Set deliberate ranges and avoid ambiguous overlap when defining them manually.

Check Change a title in the phone profile, return to Base and verify the base is retained. Preview a narrow phone and a wide screen, not just the reference canvas.

05 / CONTENT & BACKGROUNDS

13. Tips and content collections

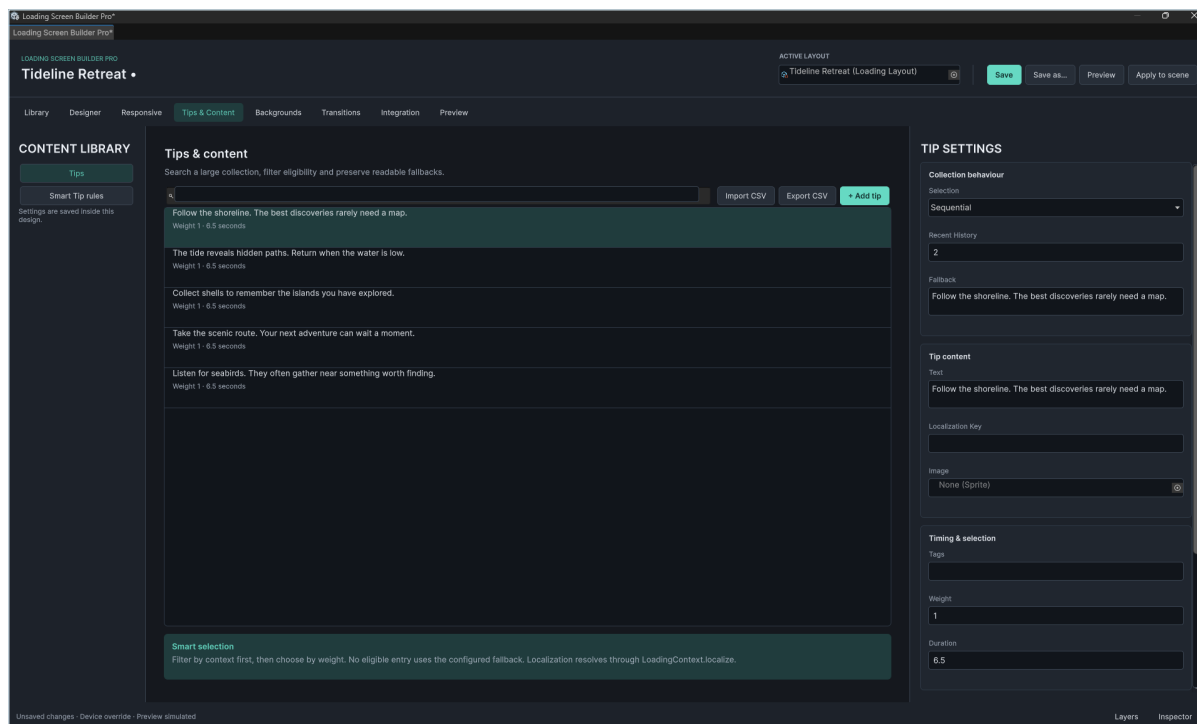


Figure 18 / Tips & Content: searchable entries, CSV controls, collection behavior and selected tip settings.

- 1 Choose **+ Add tip**, or select an existing entry in the central list.
- 2 Edit Text and optional Localization Key/Image in the right inspector.
- 3 Set Weight and Duration for the entry. Use tags for authoring/search.
- 4 Choose collection Selection and Recent History, then author a readable Fallback.
- 5 Add a Tip layer in Designer to display the selected collection content.

Screenshot setting	Meaning
Selection = Sequential	Eligible tips are selected in sequence; Random and Weighted Random are alternatives.
Recent History = 2	Avoid recent entries while other eligible choices exist.
Fallback	Copy used when the collection has no eligible tip.
Weight / Duration	Relative choice weight / display interval. Weight zero excludes an entry.

Check Start a longer preview and verify tips remain readable. Eligibility filtering happens before random/weighted selection.

05 / CONTENT & BACKGROUNDS

14. Smart Tip rules

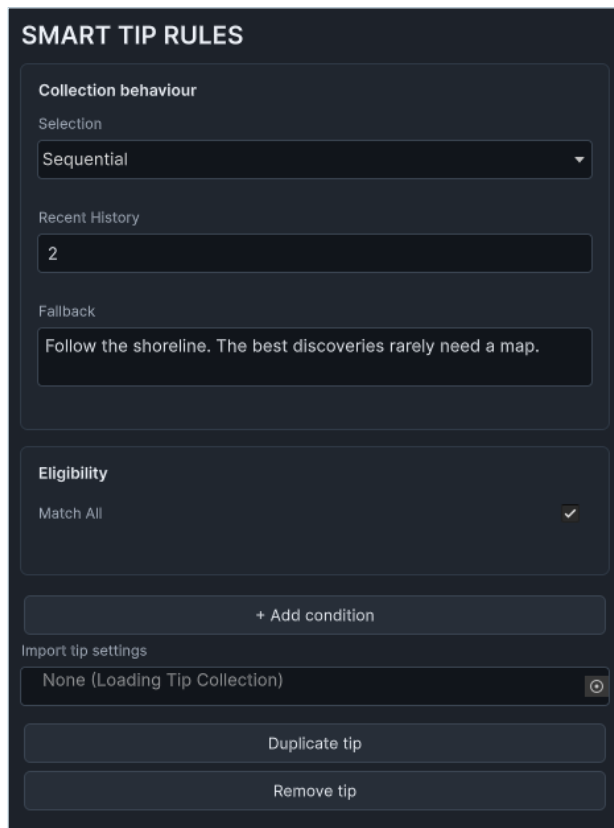


Figure 19 / Smart Tip rules inspector: Match All is enabled; no condition has been added yet.

Add a condition

- 1 Select the tip, then choose **Smart Tip rules**.
- 2 Press **+ Add condition**. Choose Scene Tag, Player Flag or Input.
- 3 Set the comparison and value. For Player Flag, set its key too.
- 4 Choose Match All for an AND rule, or disable it for OR behavior.
- 5 Set Missing Matches deliberately and provide collection fallback copy.

The screenshot shows an empty rule list. Checking Match All alone does not add a filter. Duplicate tip makes another entry with a new ID; Remove tip removes the selected one.

Example rule	Request context it can match
Scene Tag Contains forest	<code>request.context.sceneTags.Add("forest")</code>
Player Flag chapter Equals 2	<code>request.context.playerFlags["chapter"] = "2"</code>
Input Equals Gamepad	<code>request.context.input = "Gamepad"</code>

Rules Equals uses exact values. Scene tags are joined into a string, so Contains is usually useful for one tag. Missing context follows Missing Matches; it is not inferred automatically from your player.

Check Test the same tip with matching and missing context, and verify the collection fallback remains useful.

05 / CONTENT & BACKGROUNDS

CSV exchange and localization

Use **Import CSV** and **Export CSV** from the Tips toolbar. Import validates entries before offering Replace or Append. Keep a design copy before replacing a large collection.

CSV format

```
id,text,localizationKey,tags,weight,duration
"forest-01","Follow the trail.", "tips.forest.01", "forest",1,6
"forest-02","Watch the canopy.", "tips.forest.02", "forest",2,8
```

Rule	Why it matters
text is required	The CSV must contain a text column.
Unique IDs	Missing IDs are generated. Duplicate IDs are rejected; Append cannot reuse existing IDs.
Numeric values	Weight is nonnegative, duration positive. Use invariant decimal notation.
Quoted fields	Commas, escaped quotes and line breaks can be quoted.
Exchange scope	The six columns above do not include images or eligibility-rule trees.

Supply a localization lookup

The package reads localization keys through `LoadingContext.localize`. Connect it to your own string source; no localization package is mandatory. Return null for a missing key to retain authored fallback copy.

Snippet inside your existing method

```
var request = LoadingScreenBuilder.CreateSceneRequest("Level01");
request.context.localize = key =>
    key == "tips.forest.01" ? "Stay on the forest trail." : null;
LoadingScreenBuilder.Show(request);
```

Check Verify font coverage, actual translation length and wrapping. A localization key does not create a translation by itself.

05 / CONTENT & BACKGROUNDS

15. Background types and layers

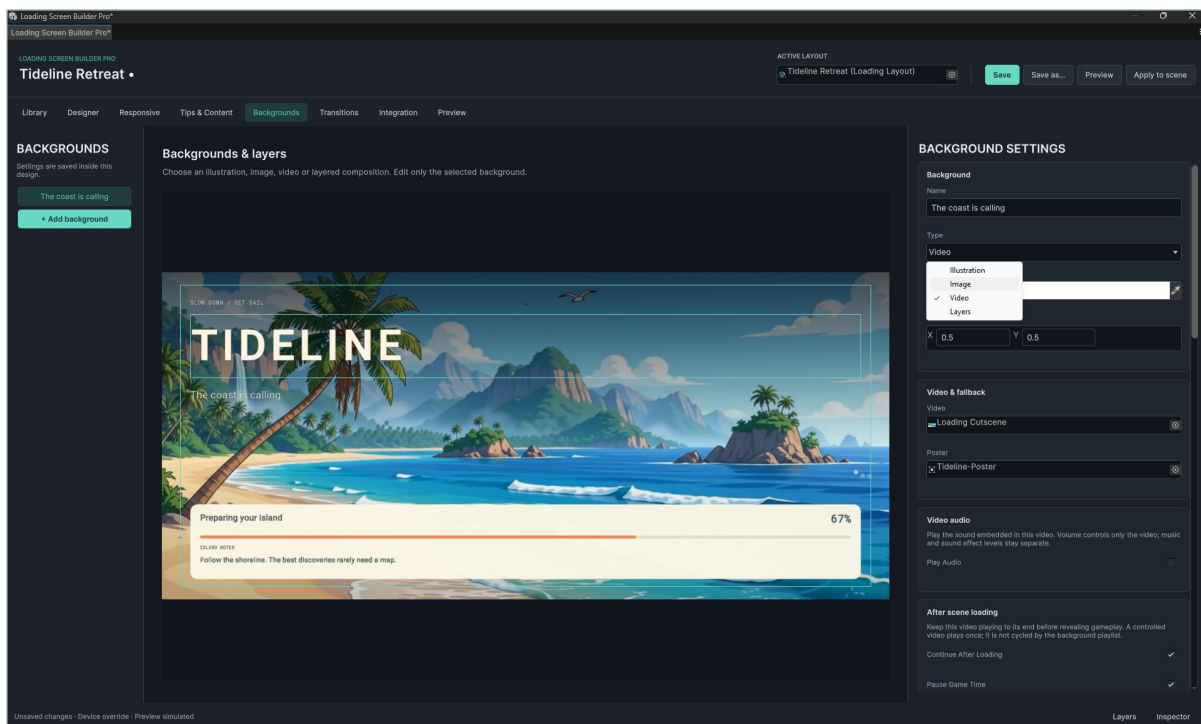


Figure 20 / Backgrounds: the selected entry is Video. The menu also offers Illustration, Image and Layers.

Type / field	How to use it
Illustration	Use the built-in illustrated background mode.
Image	Assign a sprite. Images preserve proportions and crop to cover; review portrait framing.
Video	Assign a clip and poster; configure sound and optional after-loading playback.
Layers	Add independent image layers with visibility, tint, offset, scale, depth and motion.
Tint / Focal Point	Multiply media color and control the framing target when the viewport crops.
Multiple entries	Use playlist selection and per-entry duration. A controlled after-loading video locks its own flow.

For layered art, add each prepared sprite separately and leave overscan so movement does not reveal image edges. Emberwood is an example. The package does not automatically separate a flat image into semantic depth layers.

Check Changing the selected background should affect that entry. Test its framing in portrait and ultrawide profiles.

05 / CONTENT & BACKGROUNDS

Video, poster, audio and Skip

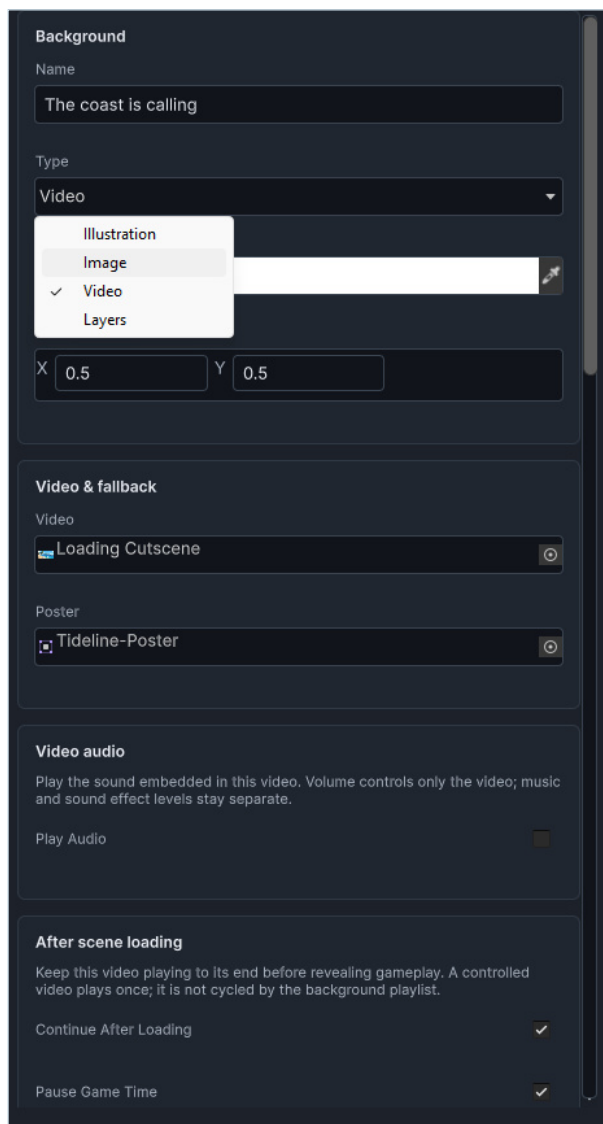


Figure 21 / Video settings shown: clip, poster, embedded audio and Continue After Loading.

Follow the shown setup

- 1 Assign **Video** and **Poster** in Video & fallback.
- 2 Enable **Play Audio** to use embedded sound. The screenshot has it off, so this entry is silent.
- 3 Use **Continue After Loading** when the video should finish before gameplay. The screenshot has it on.
- 4 Scroll for Skip controls, which loading elements to hide, natural-end/skip routes and exit transitions.

A controlled video plays once and is not cycled by the playlist. On End and On Skip can independently route to Post-Loading or Gameplay.

Play Audio controls the video only. Its level multiplies session Master and respects mute; ordinary Music/Ambience/SFX tracks remain separate.

Fallback and skip Provide a poster for preparation, missing/error/stalled video and Reduced Motion. An early allowed Skip never bypasses real loading or required readiness. Check CanSkipVideo before offering the API action.

Check Test natural end and Skip, with the intended post policy. Keep Unity foreground for Editor playback checks, then verify your actual target player/codec.

06 / BEHAVIOR, TRANSITIONS & AUDIO

16. Loading mode and activation

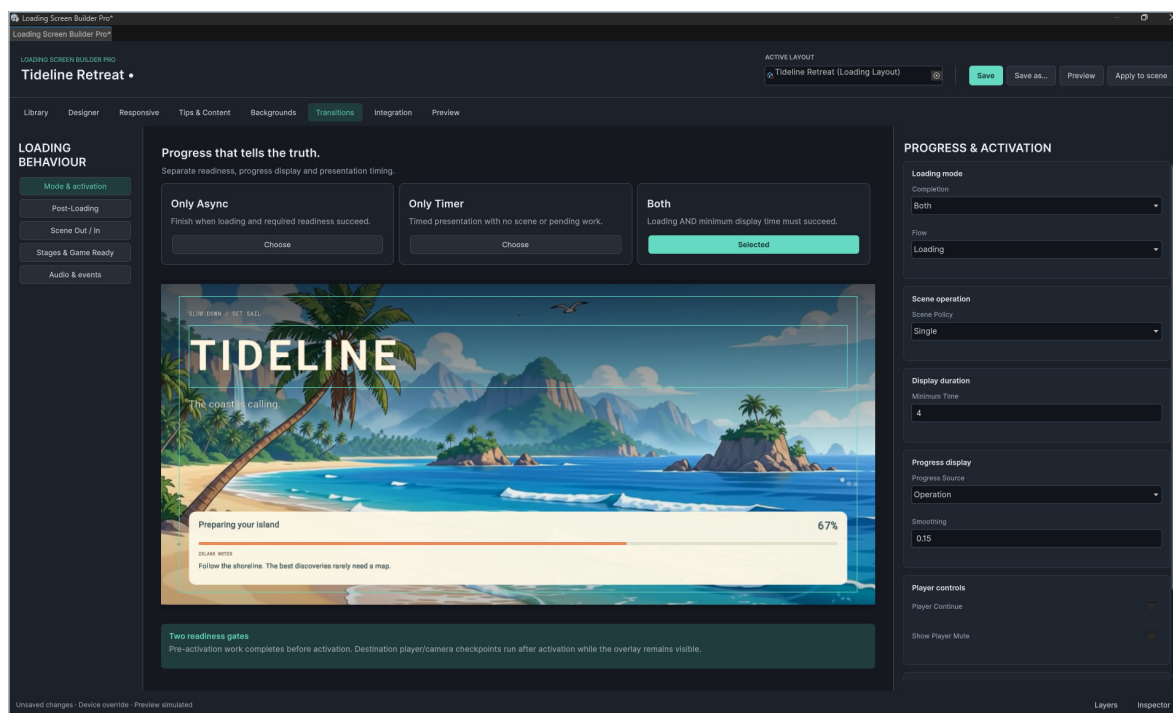


Figure 22 / Both is selected. The right inspector shows Single, Minimum Time 4, Operation progress and Smoothing 0.15.

Mode / source	Behavior
Only Async	Complete actual work and required readiness without a timer minimum.
Only Timer	Timed current-scene presentation with Scene Policy None; owns no real work/required stages.
Both	Actual work and the minimum display interval must finish.
Operation / Timer progress	Display measured work or elapsed duration. This choice does not replace completion gates.
Manual / Indeterminate	Display API-reported progress or an unknown/activity state.
Scene Policy	Single replaces; Additive loads alongside and activates the destination; None is for no-scene/custom work.

The screenshot's four seconds is an example. A one-second scene with Both waits for the minimum interval; a five-second scene still waits for actual work. Smoothing eases visual changes. Transition Only is a separate no-scene presentation flow.

Continue controls Player Continue is a pre-activation player gate with optional hold duration/timeout. Post-loading Continue is later, after readiness. Neither completes unfinished work.

17. Scene Out / In transitions

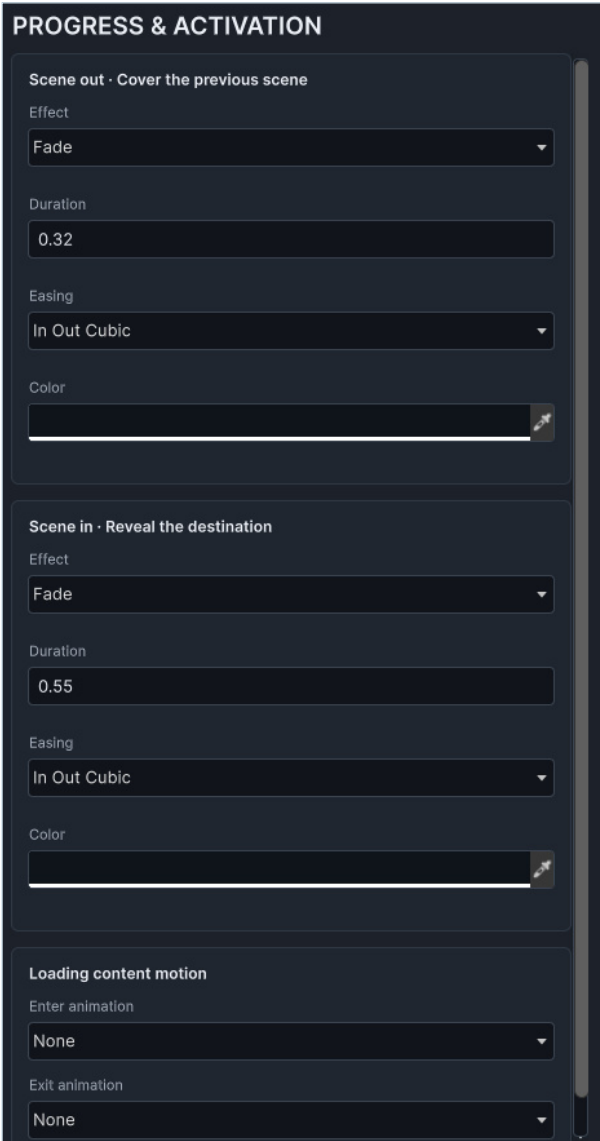


Figure 23 / Scene Out and Scene In are configured separately; the example uses Fade for each.

Field	Purpose
Scene out	Cover the previous scene before showing loading content.
Scene in	Reveal the ready destination.
Effect	None, Fade, directional Slide, Iris or Wipe.
Duration	Transition length; shown as 0.32 and 0.55 seconds.
Easing	How the transition interpolates over time.
Color	The cover/transition color.
Loading content motion	Optional loader Enter/Exit animation clips.

Scene transitions are distinct from a layer's Enter/Exit motion. Post/video flow transitions have their own Fade, Slide, Zoom, Cross Dissolve, Custom or None choices.

Custom flow transition Custom post/video transitions expose opacity, offset X/Y and scale curves. Keep essential prompts readable throughout entry/exit and test very fast loads.

Check Change Scene Out without changing Scene In, then run a real request. Preview is simulated and cannot prove native activation timing.

18. Stages & Game Ready

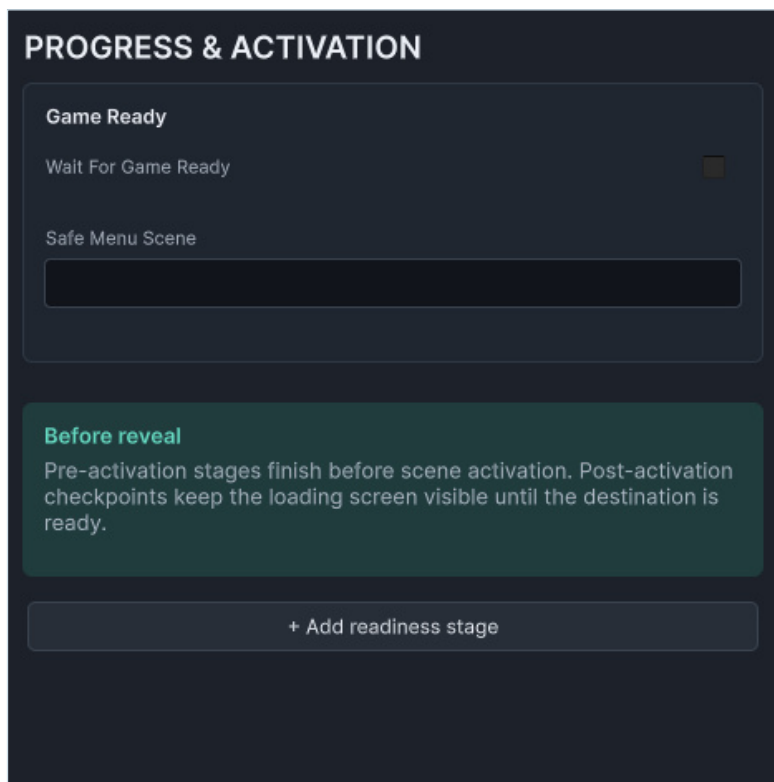


Figure 24 / Readiness inspector: Wait For Game Ready is off, Safe Menu Scene is empty, and no custom stage has been added.

- 1 Enable Game Ready only when destination initialization must finish before reveal.
- 2 Choose **+ Add readiness stage**. Set a stable ID, readable Label, Phase, Source, Weight, Required, Timeout and dependencies.
- 3 Use **Post Activation** when the stage depends on destination objects, such as player/camera initialization.
- 4 For Signal stages, send the exact checkpoint ID after initialization. For Task stages, supply the matching request.tasks entry.
- 5 Set a suitable timeout and deliberate recovery/menu action.

Two phases Pre Activation finishes before the destination exists. Do not wait there for one of its scene components. Post Activation keeps the overlay visible after activation until required work completes.

Check The screenshot's unchecked box and empty stage list mean no custom destination-ready requirement has been authored. Do not copy an unnecessary game.ready wait into a scene that will never signal it.

06 / BEHAVIOR, TRANSITIONS & AUDIO

19. Audio mix and preview

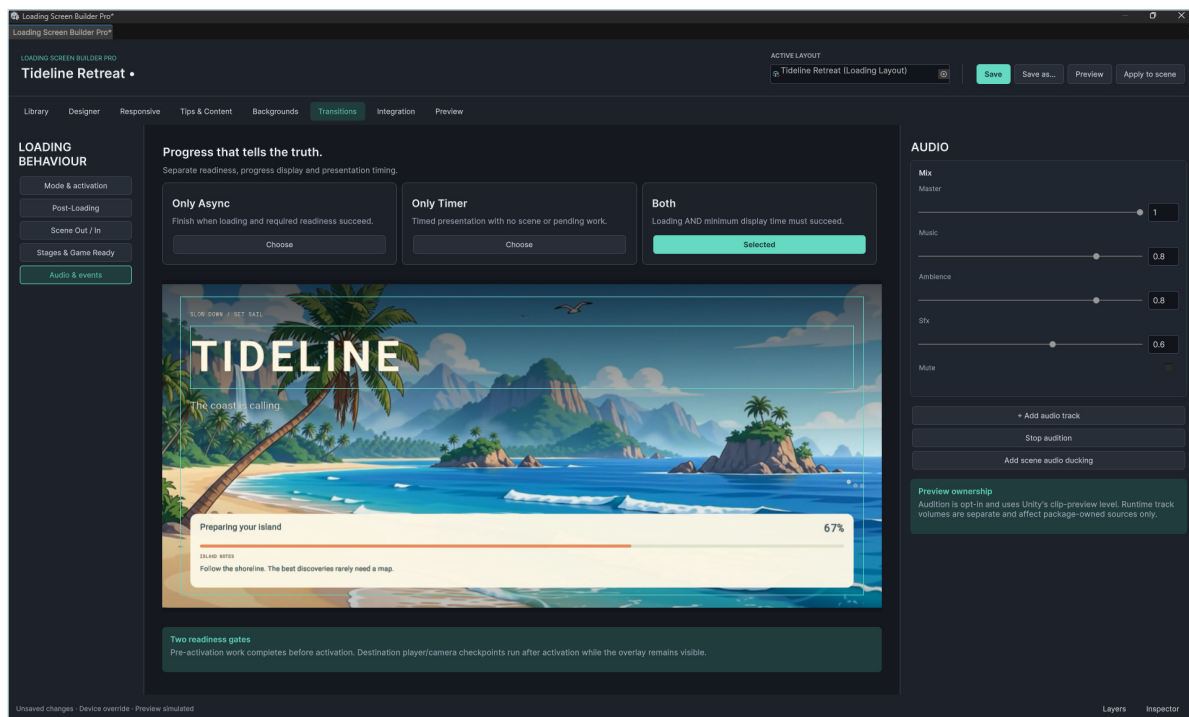


Figure 25 / Transitions > Audio & events: design channel levels, Mute, track creation, audition and scene ducking.

Shown control	Meaning
Master = 1	Design master multiplier.
Music / Ambience = 0.8; Sfx = 0.6	Authored channel multipliers, not mandatory values.
Mute	Mute the design's audio mix.
+ Add audio track	Create a track and assign a clip, channel and trigger.
Stop audition	Stop opt-in Unity Editor clip preview.
Add scene audio ducking	Add LoadingAudioDuckTargets to a selected scene audio-manager object; assign sources you own.

A normal loading track, video audio and countdown sounds have different settings. Configure each intentionally. Session Master/Music/Ambience/Sfx/mute preferences multiply authored playback settings without rewriting the design.

Ownership Ducking captures/restores assigned source volumes and yields if other code changes them. Use a persistent manager for audio that must span Single scene replacement. Editor audition level is separate from the runtime mix.

Check Test the complete request with mute on/off. Verify your existing game audio restores when loading ends.

06 / BEHAVIOR, TRANSITIONS & AUDIO

Audio track fields - screenshot walkthrough



Figure 26 / A newly added track named Ambience, with no clip assigned yet.

Shown field	How to interpret it
Name: Ambience	An authoring label only.
Clip: None	No clip can play until you assign one.
Channel: Music	This track currently uses Music, despite its name. Choose Ambience if that is the intended routing.
Mixer: None	Optional Audio Mixer Group output.
Volume 0.7 / Mute off	Per-track level and mute state.
Loop on / Pitch 1	Repeat the clip at normal pitch.
Fade In 0.5 / Out 0.75	Transition the track's volume in/out.
Crossfade 0.5	Authored crossfade interval.
Trigger: Enter	Start at Enter; Loading, Ready and Exit are alternatives.

Finish the setup Assign a clip, choose the correct channel, audition it, then run the request. The screenshot is a configuration example, not proof that a clip is playing.

Check Changing the track name does not change its channel. Listen at the authored/runtime levels as well as in Editor audition.

07 / SCENE INTEGRATION

20. Connect a destination scene

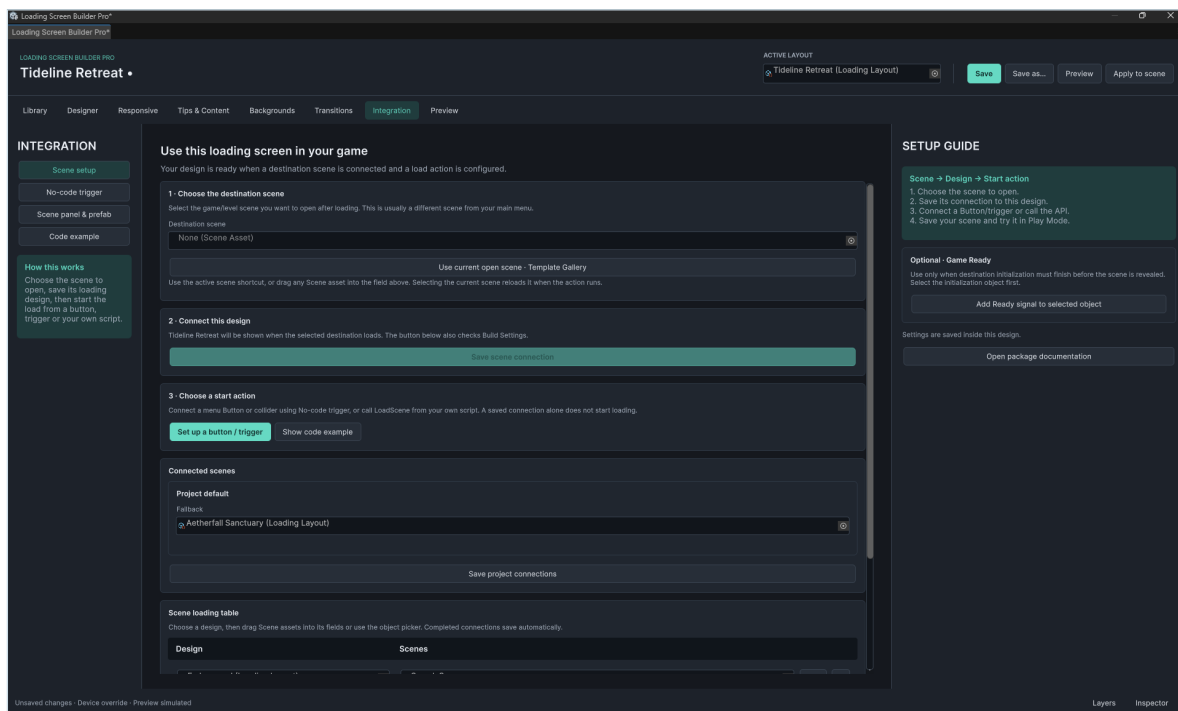


Figure 27 / Integration > Scene setup: destination, saved connection, start action, project default and the scene table.

- 1 Assign the saved **Destination scene** asset. **Use current open scene** selects the active saved scene; loading it reloads that scene.
- 2 Choose **Save scene connection**. In the screenshot this is disabled because Destination scene is None.
- 3 Enable the destination in the active Build Profile scene list. Scene asset references do not by themselves make it build-loadable.
- 4 Choose a start action: a Button/trigger or your script.
- 5 Save your scene and test in Play Mode.

Project default	Purpose
Fallback	Design used for an unconnected destination. The screenshot has Aetherfall Sanctuary.
Save project connections	Persist changes to the shared Resources registry.

Design choice An explicit API layout override wins; otherwise use the saved destination design, then project fallback. The registry loads from Resources/LoadingScreenBuildPro/SceneRegistry.

Check Choosing a destination enables the connection action. Selecting the current gallery scene is not a shortcut to your game's level unless you intend to reload the gallery.

21. Maintain the scene loading table

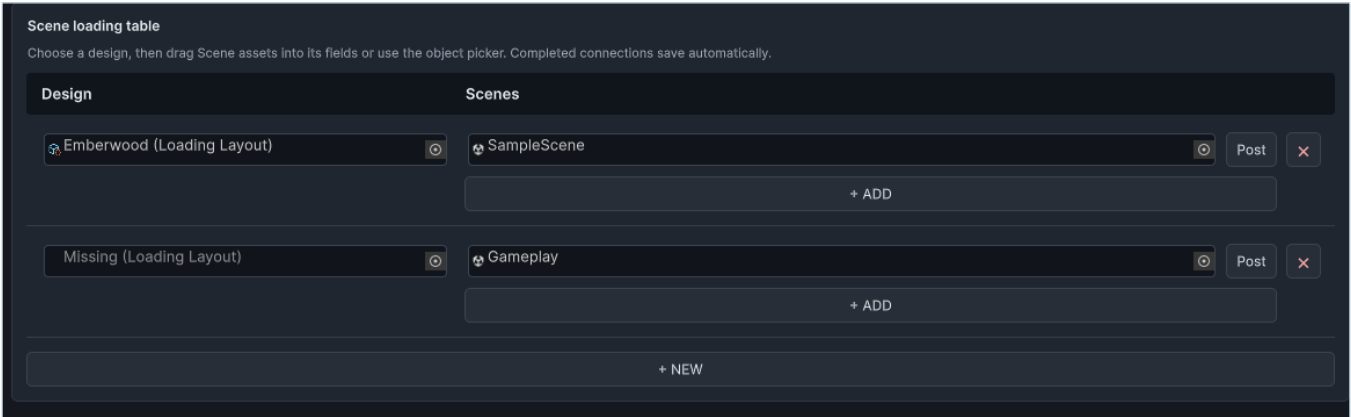


Figure 28 / One design per row, with multiple destination fields. This image also shows a broken Missing layout reference.

Control / row	What to do
Design field	Choose the LoadingLayout for that row's destinations.
Scene field / + ADD	Assign a Scene asset; add more scenes sharing this design.
+ NEW	Start another design row.
Post	Use Design, Disabled or Custom for this destination's post-loading settings.
x	Remove only that scene connection. Removing the final scene removes the row.
Missing (Loading Layout)	Repair by choosing a valid design, or remove the obsolete Gameplay row. This is not a valid finished setup.

Completed fields save automatically. Incomplete rows are temporary editor drafts. A scene has one design connection: assigning it to another row moves it, and matching design rows merge.

Per-scene Post overrides LoadScene/CreateSceneRequest include these policies. Direct Show requests must carry request.postLoading explicitly. In the current source, LoadingTrigger constructs a direct design-level request and does not apply the row's Post override.

Check Repair the Missing reference before testing Gameplay. Then load SampleScene through the API and confirm it resolves to Emberwood as shown.

22. Connect a no-code trigger

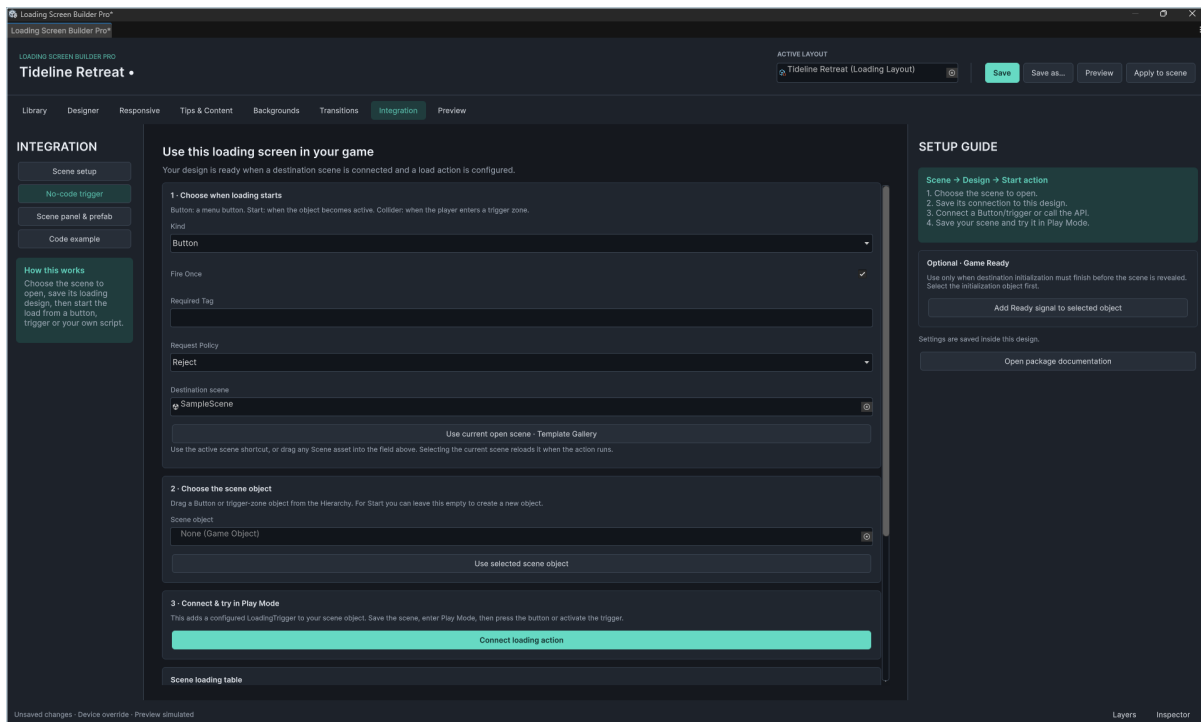


Figure 29 / Button trigger setup. Destination is SampleScene, but the Scene object target has not been assigned yet.

- 1 Choose **Kind = Button, Start or Collider**.
- 2 Set Fire Once, optional Required Tag and Request Policy. Reject refuses a request while another is active; Queue schedules it afterward.
- 3 Confirm Destination scene. Drag a Button or trigger-zone object into **Scene object**, or choose **Use selected scene object**.
- 4 Choose **Connect loading action** to add/configure LoadingTrigger, then save the scene.
- 5 Enter Play Mode and use the actual Button/trigger.

Kind	Needed setup
Button	A UI Button and functioning EventSystem/input.
Start	Starts from the configured component's Start method. Avoid a scene endlessly reloading itself.
Collider	A valid 3D/2D trigger physics arrangement and matching tag if used.

Check The screenshot's None Scene object is unfinished for a Button. Assign it before expecting a click to load. ResetTrigger() can clear Fire Once when your game needs another attempt.

23. Export Canvas or prefab output

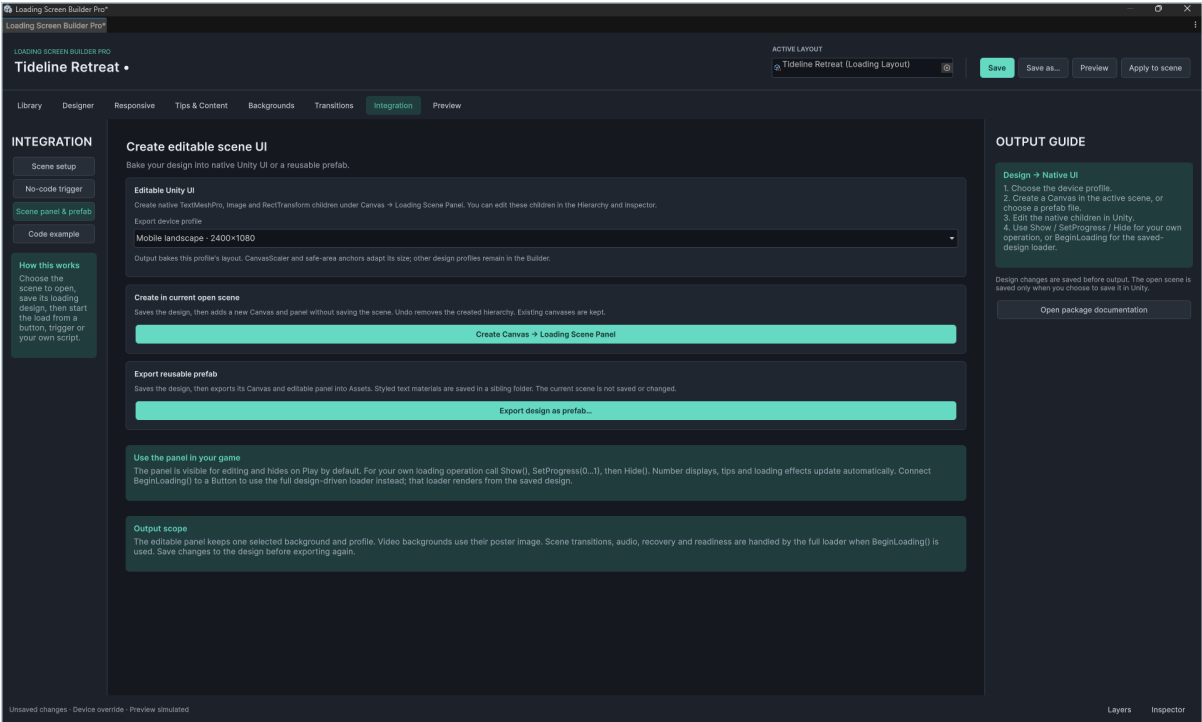


Figure 30 / Integration > Scene panel & prefab: choose a profile, create scene UI or export a reusable prefab.

- 1
- Choose the **Export device profile** to bake. The screenshot uses Mobile landscape 2400x1080.
- 2
- Choose **Create Canvas > Loading Scene Panel** to add editable native UI to the active scene, or **Export design as prefab...** to choose an Assets output path.
- 3
- Inspect the generated TextMeshPro, Image and RectTransform children. Save the scene/prefab yourself after edits.
- 4
- For your own operation call Show / SetProgress / Hide. For the full saved-design loader connect BeginLoading().

Baked output	Scope
One profile / first background	Other profile data remains in the Builder. Video uses a poster.
Native hierarchy edits	Do not write back to Builder data. Re-export regenerates output from the saved design.
BeginLoading	Hides the panel and starts the full loader from saved design data, not native-only hierarchy changes.

Check The panel is visible in Edit Mode and hides on Play by default. Export saves design data; it does not silently save your open scene.

07 / SCENE INTEGRATION

24. Use the built-in code example

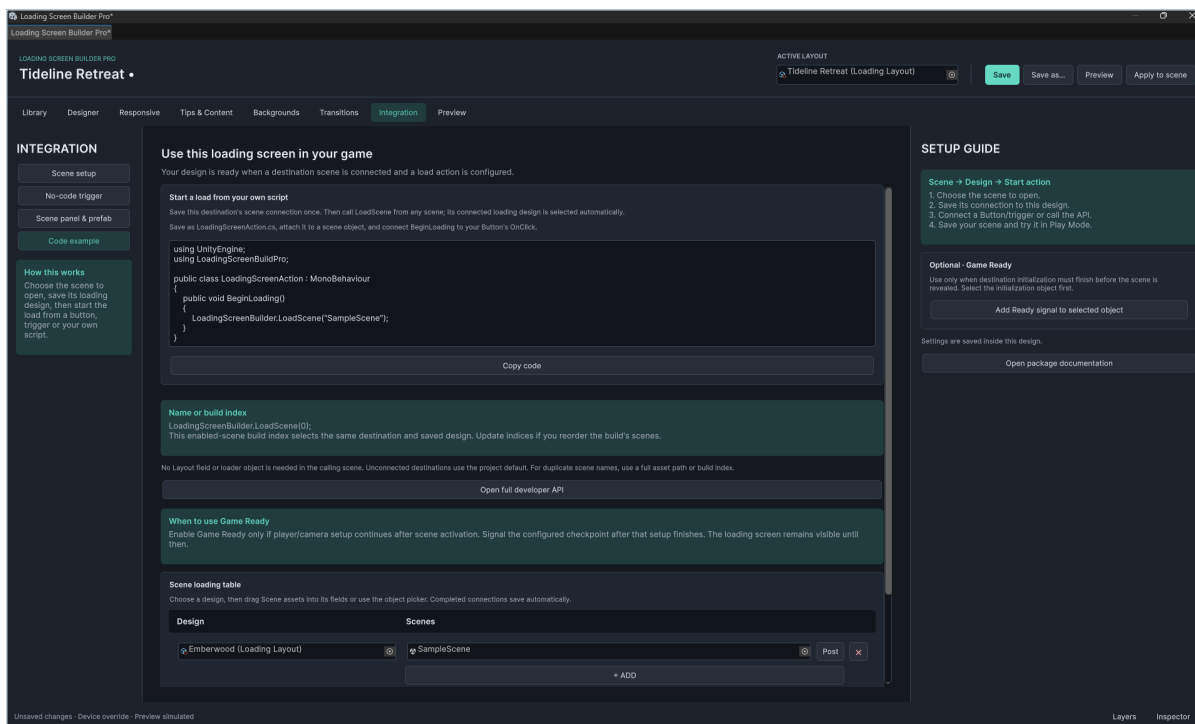


Figure 31 / Integration > Code example: copy a destination-specific script and use the existing scene connection.

- 1 Choose the destination and save its connection first.
- 2 Choose **Code example**, then **Copy code**.
- 3 Create the script using the displayed class/file name, attach it to a scene object and connect the Button's **OnClick** to **BeginLoading()**.
- 4 Use a unique scene name, full asset path or integer build index. Enable the destination in the active build scene list.

The essential call

```
using LoadingScreenBuilderPro;
// Inside your Button method:
LoadingScreenBuilder.LoadScene("SampleScene");
// An alternative integer index: LoadingScreenBuilder.LoadScene(0);
```

No extra loader object The service creates its persistent loader automatically. A calling scene does not need a Layout field when it uses the saved registry. Indices change if you reorder enabled build scenes; a numeric string is a scene name.

Check Use this API path when the destination's per-scene Post override matters. Do not start both the copied script and a separate LoadingTrigger from the same Button.

08 / PREVIEW & VALIDATION

25. Preview without loading a scene

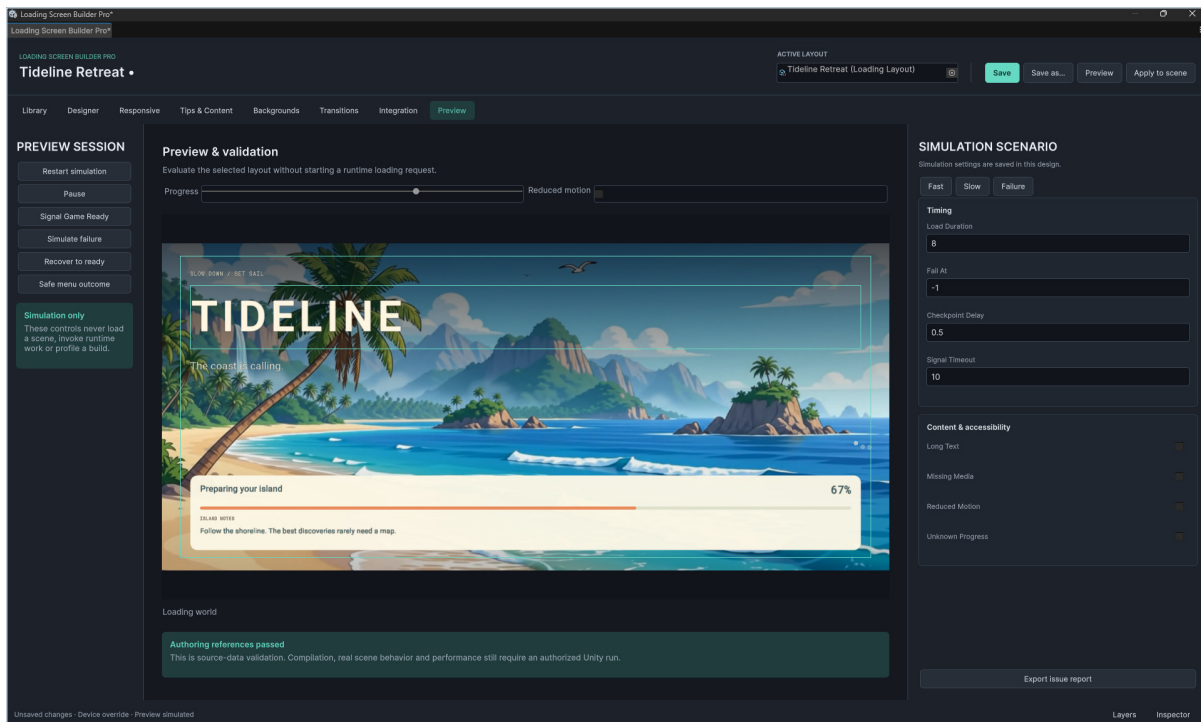


Figure 32 / Preview uses simulated progress and scenario settings. The visible 67% is not a measured scene load.

Control shown	Use
Restart simulation / Pause	Replay or pause simulated time.
Progress / Reduced motion	Inspect a particular visual state or accessibility fallback.
Signal Game Ready	Simulate the ready visual state.
Simulate failure / Recover to ready	Inspect failure/recovery presentation.
Safe menu outcome	Simulate the outcome without actually loading a menu scene.
Authoring references passed	Source-data checks found no issue; this does not run a scene or profile a build.

- 1 Replay Fast and Slow presets to inspect entry/exit and hint readability.
- 2 Try Failure, Unknown Progress and Missing Media.
- 3 Return to Designer after the stress test; Preview's long-text stress does not rewrite your title/hint content.

Check No real scene operation should start from these controls. Use an actual API/trigger request afterward for integration verification.

Scenario fields and validation reports

SIMULATION SCENARIO

Simulation settings are saved in this design.

Fast

Slow

Failure

Timing

Load Duration

8

Fail At

-1

Checkpoint Delay

0.5

Signal Timeout

10

Content & accessibility

Long Text

Missing Media

Reduced Motion

Unknown Progress

Export issue report

Field	Meaning
Load Duration = 8	Simulated work duration.
Fail At = -1	This sample does not schedule a time-based failure.
Checkpoint Delay = 0.5	Simulated ready-checkpoint delay after load.
Signal Timeout = 10	Simulated signal wait limit.
Long Text	Stress text bounds/wrapping.
Missing Media	Inspect authored fallback appearance.
Reduced Motion	Inspect a static/lower-motion state.
Unknown Progress	Simulate an unavailable fraction.
Export issue report	Write a source-data authoring report.

The Fast/Slow/Failure buttons provide starting scenarios. Your real destination load time is independent of this simulated duration.

Figure 33 / The scenario inspector is saved in this design; shown timing values are examples.

Release verification A clean issue report is useful, but it cannot prove native video decoding, scene activation, physical-device layout or performance. Test those in an appropriate player build.

09 / RUNTIME C# EXAMPLES

API example 1: a menu Button

Save this complete script as GuideLoadButton.cs. Attach it to a menu object, set Destination and connect your Button's OnClick to BeginLoading. Use a saved destination connection; the explicit override may stay empty.

GuideLoadButton.cs

```
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuideLoadButton : MonoBehaviour
{
    [SerializeField] private string destination = "SampleScene";
    [SerializeField] private LoadingLayout optionalOverride;

    public void BeginLoading()
    {
        if (LoadingScreenBuilder.IsLoading) return;
        var handle = LoadingScreenBuilder.LoadScene(
            destination, optionalOverride);
        handle.Failed += OnFailed;
    }

    private void OnFailed>LoadingRequestHandle handle)
    {
        if (handle.State != LoadingState.Cancelled)
            Debug.LogException(handle.Error);
    }
}
```

Calling rules

Use Unity's main thread in Play Mode. If your own script belongs to a custom asmdef, reference **LoadingScreenBuildPro.Runtime**. The namespace is **LoadingScreenBuildPro**, including that exact spelling.

Handle ownership Single scene replacement can destroy ordinary scene callback owners. Use a persistent observer for UI/state that must survive activation. The next example subscribes to the global lifecycle from such an observer.

Check The Button begins one request and selects the design mapped to its destination. There is no HTTP endpoint or API key.

09 / RUNTIME C# EXAMPLES

API example 2: observe lifecycle events

Save as GuideLoadingObserver.cs and put one instance on a root GameObject in your startup scene. This complete example persists its observer and pairs subscription/unsubscription. Use one instance to avoid duplicate logs.

GuideLoadingObserver.cs

```
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuideLoadingObserver : MonoBehaviour
{
    private void Awake() => DontDestroyOnLoad(gameObject);
    private void OnEnable()
        => LoadingScreenBuilder.Lifecycle += OnLifecycle;
    private void OnDisable()
        => LoadingScreenBuilder.Lifecycle -= OnLifecycle;

    private void OnLifecycle(string milestone,
        LoadingRequestHandle handle)
    {
        if (milestone == "OnProgress") return;
        string progress = handle.Progress.HasValue
            ? $"{handle.Progress.Value * 100f:0}% " : "unknown";
        Debug.Log($"{milestone}: {handle.Scene}, {progress}");
    }
}
```

Observe	Meaning
handle.Changed / Completed / Failed	Per-request updates and final outcomes. Inspect State: cancellation also uses the Failed notification.
Active / global properties	Describe only the current active request; nullable values become null when idle.
OnGameReady / OnComplete	Required readiness finished / the entire presentation and gameplay hold have exited.

Check Keep a returned handle for a final result instead of reading Active after completion. Progress is nullable and can be unknown.

09 / RUNTIME C# EXAMPLES

API example 3: timer and transition only

Save as GuidePresentationActions.cs, assign a LoadingLayout and connect two separate UI Buttons to the public methods. The busy guard keeps these demonstrations from starting together.

GuidePresentationActions.cs

```
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuidePresentationActions : MonoBehaviour
{
    [SerializeField] private LoadingLayout layout;

    public void PlayTimer()
    {
        if (!layout || LoadingScreenBuilder.IsLoading) return;
        LoadingScreenBuilder.RunTimer(3f, layout);
    }

    public void PlayTransition()
    {
        if (!layout || LoadingScreenBuilder.IsLoading) return;
        LoadingScreenBuilder.TransitionOnly(layout);
    }
}
```

API	Behavior
RunTimer(seconds, layout)	Use an explicit timed presentation without a native scene operation.
RunTimer(layout)	Use that layout's authored duration.
TransitionOnly(layout)	Play Scene Out/In in the current scene.
Omitted layout	Use the registry project default where the overload allows it.

Configured optional flow Timer APIs remove real operation/stage requirements. Other optional authored presentation policy can still matter, so use a simple design when you want a simple three-second overlay.

Check The current scene stays loaded. To represent real custom work, use Async/Both with Scene Policy None and an operation instead.

09 / RUNTIME C# EXAMPLES

API example 4: signal Game Ready

In the design, configure a required **Post Activation > Signal** checkpoint with ID **game.ready**. Save the following as `GuideDestinationReady.cs` and attach it to the destination initializer. This example waits two frames; replace that wait with your actual initialization.

GuideDestinationReady.cs

```
using System.Collections;
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuideDestinationReady : MonoBehaviour
{
    private IEnumerator Start()
    {
        string requestId = LoadingScreenBuilder.Active?.Id;
        yield return null;
        yield return null;
        // Player/camera/data initialization must be finished here.
        if (!string.IsNullOrEmpty(requestId))
            LoadingScreenBuilder.SignalReady(
                requestId, "game.ready");
    }
}
```

No-code equivalent

Add **LoadingReadySignal** to the initializer. Set Checkpoint ID. Signal On Start / Wait Frames is suitable only if those frames represent readiness. Otherwise turn automatic signaling off and call `Signal()` after your work succeeds.

Scope and timing The request-ID overload rejects stale callbacks from another request. Post-activation checkpoints are armed before activation. Keep the initializer outside held `GameplayGate` targets and use unscaled timing if game time is paused.

Check Withhold the signal deliberately and inspect timeout recovery, then restore the signal. A progress report of 1 cannot replace this checkpoint.

API example 5: custom work and progress

Save as GuideCustomWork.cs, assign Layout and call Begin from a Button. This complete two-second demonstration stays in the current scene. Keep its host alive until cleanup; replace simulated elapsed work with your actual operation.

GuideCustomWork.cs

```
using System.Collections;
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuideCustomWork : MonoBehaviour
{
    [SerializeField] private LoadingLayout layout;
    public void Begin() => StartCoroutine(Work());
    private IEnumerator Work()
    {
        if (!layout || LoadingScreenBuilder.IsLoading) yield break;
        var policy = ScriptableObject.CreateInstance<LoadingBehaviourProfile>();
        policy.completion = CompletionMode.AsyncOnly;
        policy.scenePolicy = ScenePolicy.None;
        policy.progressSource = ProgressSource.Manual;
        var operation = new ManualLoadingOperation();
        var handle = LoadingScreenBuilder.Show(new LoadingRequest
        {
            layout = layout, behaviour = policy, operation = operation
        });
        float elapsed = 0f;
        while (!handle.IsDone && elapsed < 2f)
        {
            elapsed += Time.unscaledDeltaTime;
            operation.Report(elapsed / 2f);
            LoadingScreenBuilder.ReportProgress(elapsed / 2f, handle);
            yield return null;
        }
        if (!handle.IsDone) operation.Complete();
        while (!handle.IsDone) yield return null;
        Destroy(policy);
    }
}
```

Actual completion ReportProgress controls a Manual display; Complete finishes the underlying operation. Required gates still apply. A temporary behavior avoids editing shared saved settings. Custom operation Begin must clean up work it created if it throws.

API example 6: stage task and context

Create a required **Task** stage with ID **prepare.data** in the saved design. Choose its phase, dependencies, weight and timeout. Save the script below as `GuideStageTask.cs` and connect `Begin` to a `Button`. The delay is illustrative token-aware work.

`GuideStageTask.cs`

```
using System.Threading.Tasks;
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuideStageTask : MonoBehaviour
{
    [SerializeField] private string destination = "SampleScene";

    public void Begin()
    {
        if (LoadingScreenBuilder.IsLoading) return;
        var request = LoadingScreenBuilder.CreateSceneRequest(destination);
        request.policy = RequestPolicy.Queue;
        request.context.sceneTags.Add("forest");
        request.context.playerFlags["chapter"] = "2";
        request.context.input = "Gamepad";
        request.tasks["prepare.data"] = new TaskLoadingStage(
            token => Task.Delay(500, token));
        LoadingScreenBuilder.Show(request);
    }
}
```

Match the authored contract

The dictionary key must equal the stage ID. A progress callback can report meaningful task progress; otherwise it can remain unknown while running. Retry requires both stage permission and a safely retryable task. Cancellation is cooperative through its token.

Threads and Unity Do not access Unity objects from background work. Marshal scene/UI/readiness changes to the main thread. For task work without a scene, use Scene Policy None with Async/Both; Timer Only does not own tasks.

Check A missing required task should produce a useful error. With the matching task supplied, the stage finishes and context selects the intended tips.

API example 7: choose a post screen per call

Save as GuidePostScreen.cs, assign a PostLoadingLayout such as Aurora Ready and connect Begin to a Button. This explicitly overrides the destination's resolved post policy for this request.

GuidePostScreen.cs

```
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuidePostScreen : MonoBehaviour
{
    [SerializeField] private string destination = "SampleScene";
    [SerializeField] private PostLoadingLayout readyScreen;

    public void Begin()
    {
        if (!readyScreen || LoadingScreenBuilder.IsLoading) return;
        var request = LoadingScreenBuilder.CreateSceneRequest(destination);
        request.postLoading = new PostLoadingSettings
        {
            enabled = true, screen = readyScreen,
            waitForInput = true,
            inputs = LoadingInputKind.Keyboard | LoadingInputKind.Gamepad,
            pauseGameTime = true, blockGameplaySystems = true
        };
        LoadingScreenBuilder.Show(request);
    }

    public void Continue() => LoadingScreenBuilder.Continue();
}
```

Override	Meaning
request.postLoading = null	Inherit the resolved design policy.
new PostLoadingSettings { enabled = false }	Disable the post screen for this request.
screen	Must be PostLoadingLayout; ordinary LoadingLayout is rejected.

Hold ownership Assign LoadingGameplayGate targets for the systems you want held. Pause Game Time alone does not infer your gameplay input lock. Keep required initialization outside those targets.

09 / RUNTIME C# EXAMPLES

API example 8: drive an exported panel

Save as GuidePanelActions.cs and assign the exported LoadingScenePanel component. Wire its methods to your own operation's progress and completion events. These calls update native output without starting a scene request.

GuidePanelActions.cs

```
using UnityEngine;
using LoadingScreenBuildPro;

public sealed class GuidePanelActions : MonoBehaviour
{
    [SerializeField] private LoadingScenePanel panel;

    public void Open()
    {
        if (panel) LoadingScreenBuilder.ShowPanel(panel);
    }
    public void Report(float value)
    {
        if (panel) LoadingScreenBuilder.SetPanelProgress(
            panel, value, "Preparing...");
    }
    public void Unknown()
    {
        if (panel) LoadingScreenBuilder.SetPanelIndeterminate(panel);
    }
    public void Finish()
    {
        if (!panel) return;
        LoadingScreenBuilder.SetPanelProgress(panel, 1f, "Ready", true);
        LoadingScreenBuilder.HidePanel(panel);
    }
}
```

Full-loader alternative BeginPanelLoading(panel) / panel.BeginLoading() hides this panel and starts its saved-design loader. Native hierarchy customizations are used in manual panel mode, not when the full loader renders saved design data.

Check The panel responds to your reported value and hides when your own operation completes. No destination should load from ShowPanel alone.

10 / API QUICK REFERENCE

Input, audio and custom renderer hooks

Hook	Use
DevicePressed / DeviceHeld	Func<LoadingInputKind, bool> callbacks for filtered devices. Return a fresh press versus held state separately.
ContinuePressed / ContinueHeld	Legacy Func<bool> callbacks. When assigned, they take precedence; clear owned callbacks to use filtered hooks.
AcquireGameplayBlock	Return an IDisposable scope that blocks your gameplay actions and restores only its own state.
ConfigureEventSystem	Configure the overlay's EventSystem for your project.
PresentationFactory	Provide an ILoadingPresentation for a custom renderer.
ILoadingFlowPresentation	Optional video hiding/playback and post/gameplay transitions.
IPostLoadingCountdownPresentation	Optional countdown rendering integration.
ILoadingIndicator	Update a custom prefab graphic using progress, stage, ready, elapsed time and reduced motion.

Session audio controls

Snippet

```

LoadingScreenBuilder.MasterVolume = 0.8f;
LoadingScreenBuilder.MusicVolume = 0.5f;
LoadingScreenBuilder.AmbienceVolume = 0.7f;
LoadingScreenBuilder.SfxVolume = 0.6f;
LoadingScreenBuilder.Muted = true;

```

Session levels are clamped to 0-1 and multiply authored playback. They do not rewrite the saved design or unrelated mixer/AudioListener settings. Set integration hooks before starting requests and clear only hooks your component owns when it is destroyed.

Custom indicator contract UpdateIndicator(float? progress, string stage, bool ready, float elapsed, bool reducedMotion). A null value means unknown work. Custom prefab scripts are not executed in the Designer canvas.

10 / API QUICK REFERENCE

Start methods and active-request controls

Method	Use
LoadScene(string/int, layout, behaviour)	Start a mapped destination load; optional explicit layout/behavior overrides.
CreateSceneRequest(string/int, layout, behaviour)	Create an unstarted request to configure, then pass it to Show.
Show>LoadingRequest)	Start the configured scene/custom operation. Reject/Queue controls busy behavior.
RunTimer(seconds, layout, behaviour)	No-scene timed presentation; another overload uses authored duration.
TransitionOnly(layout, behaviour)	No-scene Scene Out/In flow.
Continue(handle = null)	Satisfy the applicable player gate, subject to policy.
SkipVideo(handle = null)	Skip only when allowed; no bypass of real work/readiness.
Cancel(handle = null)	Request cancellation. Native in-flight work must settle.
ReportProgress(value, handle = null)	0-1 visual progress for Manual source. Rejects NaN/infinity; clamps finite values.
SetStatus(text, handle = null)	Override {stage}/stage-formatted copy; null restores it.
SignalReady(checkpointId)	Signal an armed checkpoint on Active.
SignalReady(requestId, checkpointId)	Use request scope for delayed readiness callbacks.
RetryCheckpoint / ReturnToMenu / DismissFailure	Request supported checkpoint recovery, configured safe menu, or failure dismissal.

Control result An omitted handle targets Active. A true result means an instruction was accepted; configured gates determine its effect. Manual progress does not complete an operation. Native Single activation cannot be undone by Cancel.

Request settings Configure before Show. Effective settings freeze at execution start; keep queued transient configuration alive until then. Use a temporary behavior copy for per-call changes instead of editing a shared saved design.

10 / API QUICK REFERENCE

Status, milestones and final results

Property	Meaning
Active / IsLoading / State	Active handle, busy flag and nullable lifecycle state.
Progress / ProgressPercent	Nullable fractions 0-1 / values 0-100. Null means idle or unknown.
DestinationScene / CurrentLayout	Canonical destination and selected source design.
Stage / Status / Error	Lifecycle description, status override and active error.
IsReady	Required readiness is complete; optional presentation can remain.
IsPostLoading / GameplayHeld / CanSkipVideo	Ready-screen state, optional destination hold and Skip availability.
Registry	Automatically loaded scene registry or your session override.

Lifecycle event names

Phase	Milestones
Start and measured work	OnRequest, OnEnter, OnLoadStarted, OnProgress, OnLoadReady
Destination	OnActivating, OnActivated, OnReadinessStarted, OnGameReady
Optional flow	OnVideoPlayback, OnPostLoading, OnPostLoadingContinue
Finish	OnExit, OnComplete, OnFailed

OnActivated is only emitted by operations that activate a destination. Final full progress follows required readiness. Keep the handle for Scene/Layout/Context/State/Progress/Error after completion; Active becomes null when idle.

Failure versus cancellation Blank or ambiguous scene arguments / invalid indices can throw before start. Request execution failures are recorded on the handle. Cancellation uses its Failed notification path too, so inspect State before reporting an execution failure. Recovery waits for native work to settle.

11 / TROUBLESHOOTING

Troubleshooting: setup and appearance

Problem	What to check or fix
Tools menu missing	Resolve compilation errors; preserve the package folder and Editor assembly.
Text missing / pink	Import TMP Essential Resources and check font/material/shader references and glyph coverage.
Theme change has no effect	Inspect Use Theme Font / Use Theme Color and explicit local overrides.
Gradient looks too dark	The gradient multiplies base tint; start with white if you want direct gradient colors.
Bar end caps stretch	Set correct nine-slice sprite borders and use suitable source size/filtering.
Phone composition clips	Inspect aspect range, active profile, safe-area choice, anchors and long-text bounds.
Rotating device keeps old arrangement	A request selects its profile at start. Stabilize orientation during that flow.
Custom prefab is static in Designer	Its scripts do not run in the authoring canvas; test the runtime loader.
Missing design in scene table	Choose a valid LoadingLayout or remove the obsolete mapping. The supplied screenshot contains one such row.
Wrong design at runtime	Check explicit API override, saved destination connection and project fallback.
Exported panel differs from full loader	Native edits stay in the baked hierarchy. BeginLoading renders saved Builder data.

Preserve your work

Copy an affected design before broad edits. Restore a disk backup with its matching .meta file so references retain the original GUID. Native scene/prefab changes require a scene/prefab save in addition to Builder Save.

Check Reduce the reproduction to one saved scene connection, one start action and a known supplied design. Add media, overrides and stages back one group at a time.

11 / TROUBLESHOOTING

Troubleshooting: work, input and sound

Problem	What to check or fix
Nothing loads from a Button	Assign Scene object in no-code setup, or wire the script method. A connection alone does not start loading.
Scene unavailable / wrong index	Save it and enable it in the active build list. Use path/index for duplicate names.
Stuck progress or Game Ready timeout	Check stage ID, phase, dependencies, matching task/signal and timeout. Do not await destination objects before activation.
Initializer never completes	Keep it outside held GameplayGate targets and avoid scaled waits while game time is paused.
Manual progress is 100%, flow remains	Complete the real operation or required checkpoint. Progress reporting is presentation only.
Continue / Skip ignored	Inspect the active gate, input filter, countdown lock and fresh Pressed versus Held callbacks.
Busy request immediately fails	Reject is default. Choose Queue deliberately or prevent duplicate start actions.
Video stays on poster	Check clip/decoder, Reduced Motion, prepare timeout and foreground/target-player playback.
Video or countdown silent	Enable their own audio options and assign/import clips. Check session Master/SFX as applicable and mute.
Track named Ambience uses Music level	Name does not route audio. Change its Channel to Ambience.
Per-scene Post override ignored	Use LoadScene/CreateSceneRequest. The current LoadingTrigger uses its design's post settings.
Cancel does not restore old scene	Single activation is irreversible. Use deliberate safe-menu recovery for the reached phase.

Diagnose the actual request Keep the handle and inspect State/Error and lifecycle milestones. Editor simulation cannot establish native loading or video behavior.

12 / COMPLETION

Final guided checklist and glossary

- 1 Save your design copy, post data, registry connections and changed scenes/prefabs.
- 2 Verify the destination and safe-menu scene are enabled in the active build list.
- 3 Run each real start action, including repeated input and busy-request handling.
- 4 Verify readiness success and a deliberate timeout/recovery path.
- 5 Test video natural-end/Skip, Continue and countdown routes; confirm time/input/gameplay targets restore.
- 6 Review fonts, translations, safe areas, portrait/wide profiles and Reduced Motion.
- 7 Test ordinary tracks, video audio, countdown sound and mute in the target player.
- 8 Measure activation/initialization, memory and frame time on representative hardware.

Term	Meaning
Prepared / Activation	Operation at its activation boundary / making the destination live.
Game Ready / Complete	Required readiness finished / the entire flow exited and released its hold.
Profile / Override	Aspect-matched reference space / a local change to inherited/base data.
Checkpoint	A signal/task stage with stable ID, phase and completion contract.
Panel	Baked native UI driven manually or used to start the saved-design loader.

Report a reproducible issue Include package revision, Unity version, target, minimal scene/design setup, expected/actual behavior and Console output. A clean Editor preview is useful evidence but does not replace player verification.

13 / SOURCES & VERSION SCOPE

Sources and how to maintain this guide

This guide follows the supplied **2D_Parallax_Pro_Full_Guide_English - Sample.pdf**: a branded cover, numbered exercises, screenshot walkthroughs, enlarged inspector panels, field tables, full C# examples, API reference and completion checks. Product instructions come from Loading Screen Builder Pro, not from the parallax package.

Source	Use in this edition
28 supplied screenshots	The actual Builder UI, menu paths, demo folders, gallery and settings. Enlarged figures are view crops of these screenshots.
Loading Screen- Cover.png	The supplied cover artwork, preserved without recreating its branding.
LoadingScreenBuilder.cs	Facade methods, events, status and integration hooks.
Runtime/Core and Runtime/Data	Requests, stage/readiness contracts, post screens, countdown, progress and settings.
Editor views / authoring sources	Current field labels, scene connections, output and asset commands.
GettingStarted.md / DeveloperAPI.md	Companion references checked against current source.

Version scope

Documentation edition: **20 October 2026**. Source project: **Unity 6000.3.13f1**. This identifies the inspected environment, not a broader minimum-version/platform certification. Screenshot values are examples: None, unchecked fields and Missing references can represent incomplete setup, as explained in their walkthroughs.

Portable authoring files

The repository's Documentation/FullGuide folder retains original screenshot copies, their manifest, guide_content.py, the PDF generator, generated Markdown and complete CodeExamples scripts. The eight scripts are also in **Guide_Code_Examples.zip** beside the package PDF. Copy only the examples you need into Assets. Rebuild, render and inspect changed pages before distributing a revision.

Verification The guide is checked against source and rendered for visual QA. This documentation revision does not change runtime code or claim a new player/device test. Source-image captions describe what is visible, not commands to reproduce incidental unsaved or broken states.

13 / SOURCES & VERSION SCOPE

Source image index

Every supplied screenshot is used. Figure numbers identify both full views and enlarged crops; the original image copies are retained beside the guide source.

Supplied filename	Figure (PDF page)
1.png	01 (p4)
Screenshot 2026-10-20 181017.png	02 (p4)
Screenshot 2026-10-20 180930.png	03 (p5)
Scene.png	04 (p6)
Screenshot 2026-10-20 180420.png	05 (p7), 06 (p8)
Screenshot 2026-10-20 180444.png	07 (p10)
Screenshot 2026-10-20 180547.png	08 (p11)
Screenshot 2026-10-20 180600.png	09 (p12)
Screenshot 2026-10-20 180618.png	10 (p13)
Screenshot 2026-10-20 180637.png	11 (p14)
Screenshot 2026-10-20 180503.png	12 (p16)
Screenshot 2026-10-20 180508.png	13 (p16), 15 (p18)
Screenshot 2026-10-20 180734.png	14 (p17)
Screenshot 2026-10-20 180659.png	16 (p19), 17 (p20)

Cover artwork Loading Screen- Cover.png is used on the cover and reading-path page. Screenshot crops are views of the original files; no UI values have been retouched.

13 / SOURCES & VERSION SCOPE

Source image index / continued

Every supplied screenshot is used. Figure numbers identify both full views and enlarged crops; the original image copies are retained beside the guide source.

Supplied filename	Figure (PDF page)
Screenshot 2026-10-20 180708.png	18 (p21)
Screenshot 2026-10-20 180712.png	19 (p22)
Screenshot 2026-10-20 180722.png	20 (p24), 21 (p25)
Screenshot 2026-10-20 180731.png	22 (p26)
Screenshot 2026-10-20 180739.png	23 (p27)
Screenshot 2026-10-20 180742.png	24 (p28)
Screenshot 2026-10-20 180747.png	25 (p29)
Screenshot 2026-10-20 180828.png	26 (p30)
Screenshot 2026-10-20 180836.png	27 (p31)
Screenshot 2026-10-20 180857.png	28 (p32)
Screenshot 2026-10-20 180840.png	29 (p33)
Screenshot 2026-10-20 180843.png	30 (p34)
Screenshot 2026-10-20 180846.png	31 (p35)
Screenshot 2026-10-20 180908.png	32 (p36), 33 (p37)

Cover artwork Loading Screen- Cover.png is used on the cover and reading-path page. Screenshot crops are views of the original files; no UI values have been retouched.