MODEL VISTA

INNOWORKS

ENGLISH DOCUMENTATION

Model Vista Create Unity Package

User documentation for the Model Vista Create Unity Package.

15 September 2026

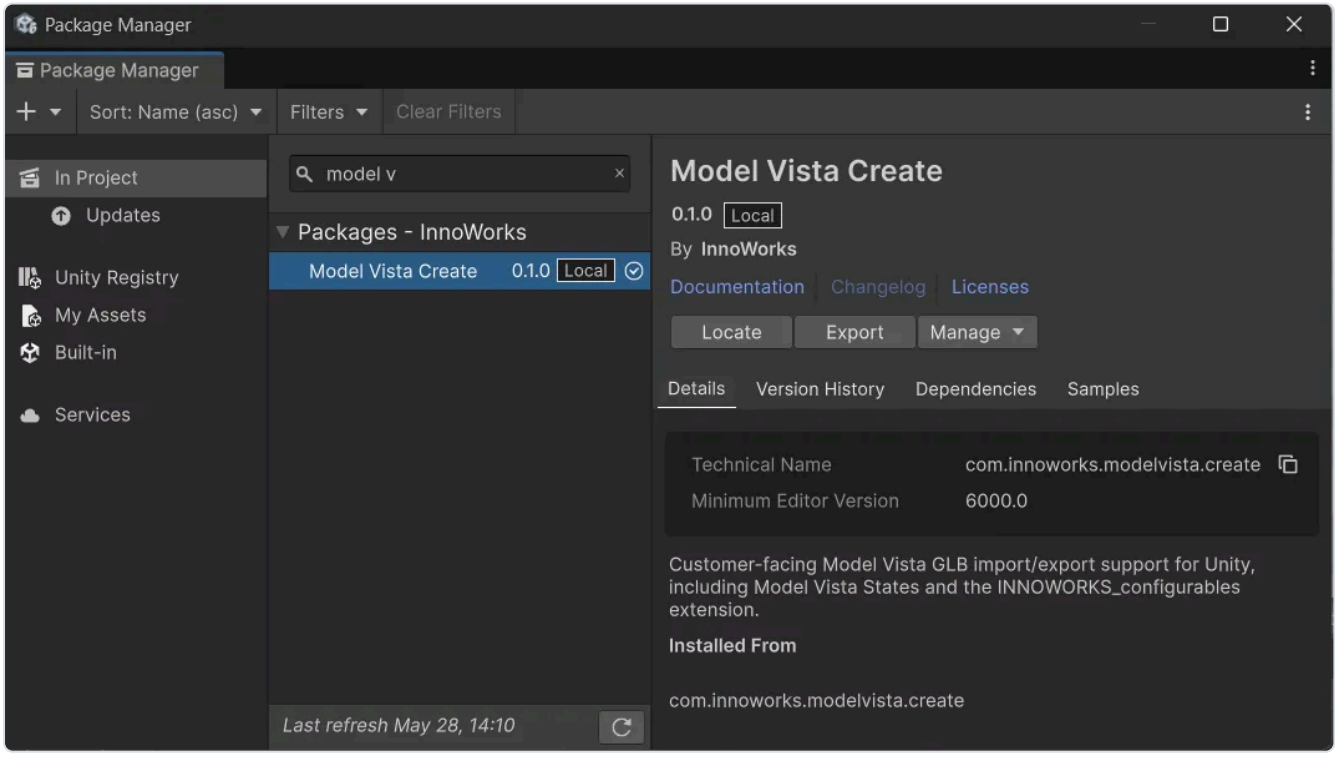
Contents

Unity package

The Model Vista Create Unity Package 0.1.0 adds Model Vista States authoring, GLB+ import/export, and supporting asset workflows to Unity 6 projects.

Quick start

1. Install **Model Vista Create** in the target Unity project.
2. Add or configure Model Vista States on a prefab or scene object.
3. Select one or more prefab assets or scene GameObjects.
4. Export using one of these entry points:
    - Select **Export GLB+** in the **Model Vista** Scene View overlay for the default Draco-compressed glTFast export.
    - Select **InnoWorks > Create > Export GLB+ via glTFast (Draco compression)**.
    - Select **InnoWorks > Create > Export GLB+ via glTFast** when uncompressed mesh data is required.
5. Choose a `.glb` path for one selection, or a destination folder for multiple selections.
6. Test the exported content with **InnoWorks > Create > Import GLB+ as Prefab**. Choose **Extract Images** or **Keep Unity Assets** when prompted about embedded textures.
7. Open **InnoWorks > State Manager** for a scene-wide or Prefab Stage view of Model Vista States.



Package Overview

The Model Vista Create package for Unity (`com.innoworks.modelvista.create`) provides the tools to import and export GLB+ files and author Model Vista States in your Unity projects.

It includes:

- glTFast Draco-compressed and uncompressed GLB+ export, plus an optional UnityGLTF backend.
- Single-file and folder-based prefab import with a choice between extracted image files and native Unity texture assets.
- A Scene View **Model Vista** overlay with **Export GLB+** and **Import GLB+** actions.
- Model Vista States runtime components, custom inspectors, and the **State Manager** window.
- Full `INNOWORKS_configurables` extension read/write support.
- Render-pipeline-aware material restoration for URP and HDRP.
- Reference-preserving texture conversion commands and editable-material copy tools.
- The optional **State Selector** sample for selecting Model Vista States from runtime UI.

Requirements

- Unity `6000.0` or newer.
- `com.unity.inputsystem` `1.14.0`.
- `com.unity.cloud.gltfast` `6.19.0`.
- `com.unity.cloud.draco` `5.4.3`.
- `com.unity.nuget.newtonsoft-json` `3.2.1`.

The package display name is **Model Vista Create** and the current package version is `0.1.0`.

Optional Integrations

- Install `org.khronos.unitygltf` only when the UnityGLTF export backend is required.
- URP and HDRP adapters compile when the corresponding render-pipeline package is installed.

Installation

Install The Package

1. Open **Window > Package Management > Package Manager**.
2. Select **Add package from disk**.
3. Select `com.innoworks.modelvista.create/package.json` from the supplied Model Vista package distribution.
4. Wait for Unity to resolve and compile the package and its declared dependencies.

The current package requires Unity 6000.0 or newer and declares Input System 1.14.0, glTFast 6.19.0, Draco 5.4.3, and Newtonsoft JSON 3.2.1 dependencies.

Verify The Installation

After compilation completes, confirm that Unity exposes:

- **InnoWorks > State Manager**;
- the GLB+ commands under **InnoWorks > Create**; and
- the **Model Vista** Scene View overlay.

Optional Packages

- To add the optional UnityGLTF exporter, select **InnoWorks > Create > Install UnityGLTF Package…** or install `org.khronos.unitygltf` through your approved package workflow.
- URP and HDRP material adapters become available automatically when the matching render-pipeline package is installed.

Model Vista States

Model Vista States are runtime components that expose configurable state sets in a scene or prefab. Each state set has a friendly configurable name, optional thumbnail, UI visibility, current option index, next/previous input actions, optional autoplay settings, and per-option trigger or hold input actions. Available state components:

Component	Use
`ActivationStates`	Switches sets of target GameObjects on and off.
`MaterialStates`	Switches materials on configured target renderers.
`TransformStates`	Applies local position and rotation options to a target transform.
`CameraStates`	Moves a target camera relative to a reference transform and can apply field of view or projection settings.
`AnimationStates`	Plays legacy Animation clips or a compatible glTF animation player by animation index.
`GroupStates`	Coordinates multiple Model Vista States containers by selecting target options together.
`ShaderStates`	Applies float, Boolean keyword, vector, color, or HDR color values globally or to target materials.
`LightStates`	Applies EV100 intensity, color filter, and color temperature options to target lights.

Use these components when the exported GLB should carry variant choices, presentation states, animation controls, shader values, or lighting choices into Model Vista and other compatible runtimes.

Shared Option Behavior

Each configurable can include:

- **Configurable Name:** Human-readable state set name shown in tools and runtime UI.
- **Thumbnail:** Sprite shown for the state set in editors or runtime browsers.
- **UI Visibility:** Visible or Hidden for UI systems and the State Manager visibility filter.
- **Current Option:** The currently active state index.
- **Previous / Next Option Actions:** Input actions that cycle through options in play mode.
- **Apply Current Option On Awake:** Applies the selected option when the component awakens.
- **Autoplay:** Optional sequential or random playback with a configurable interval. Each option can include:
- **Display Name:** Human-readable state name shown in tools and runtime UI.

- **Thumbnail:** Sprite shown in editors or runtime state browsers.
- **Transition Duration:** Blend time for state changes where blending is supported.
- **Ease Mode:** Ease In Out or Linear interpolation.
- **Trigger Action:** Input action that switches to the option.
- **Hold Action:** Input action that temporarily blends to the option while held, then returns to the previous option when released.
- **On Selected:** UnityEvent invoked when the option becomes active. Transform options include **Apply Position** and **Apply Rotation** toggles. Camera options include **Apply Position**, **Apply Rotation**, **Apply Field Of View**, and **Apply Projection** toggles.

Group Transition Timing

`GroupStates` provides two timing modes:

- **ParentGroupDuration** uses the selected Group option's transition duration and easing to drive all linked States together. This is the default.
- **LinkedStateDurations** starts all linked States together while preserving each linked option's own transition duration.

The timing mode is preserved by GLB+ export and import. Older files without the field use **ParentGroupDuration**.

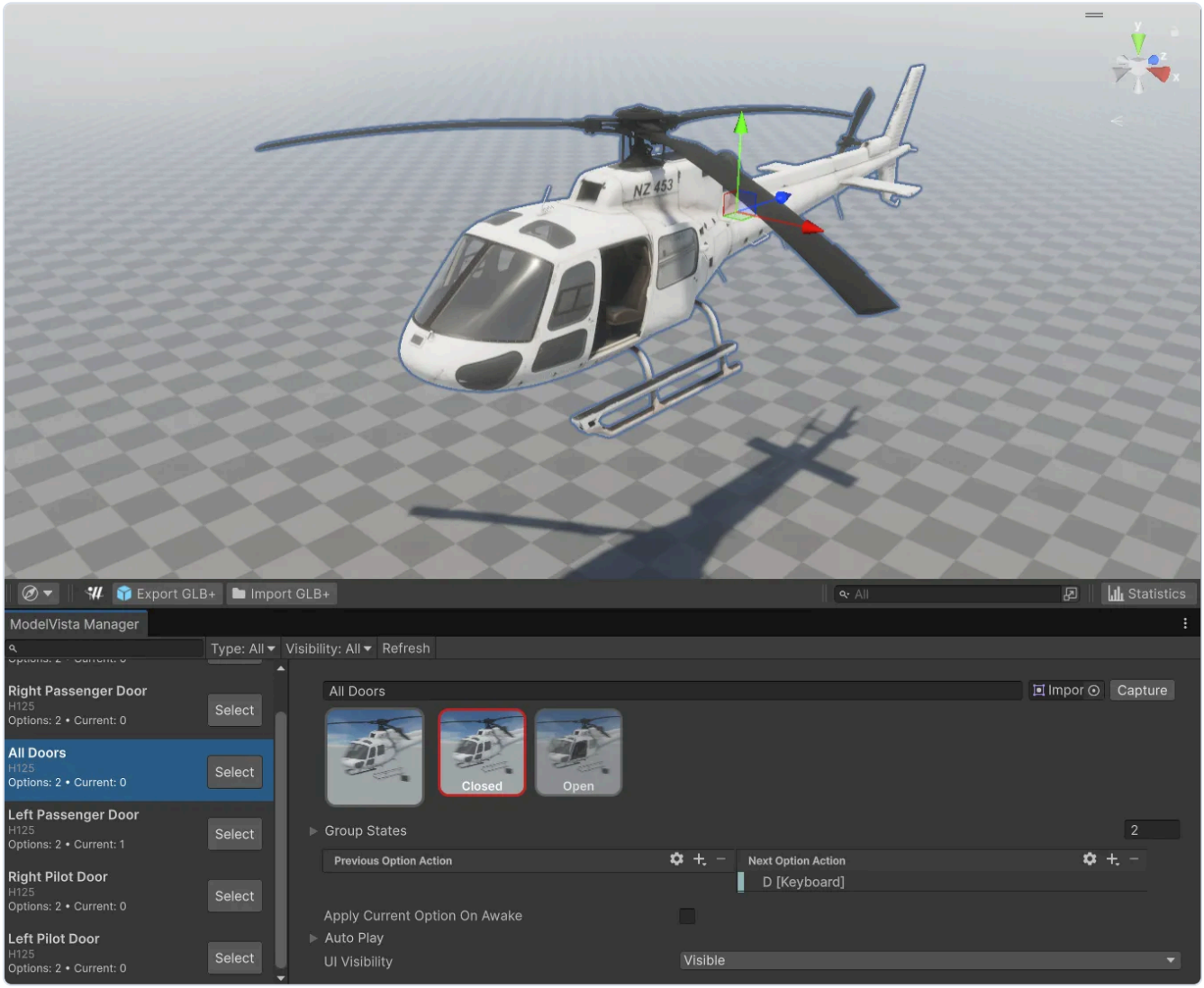
Model Vista Manager

Open **InnoWorks > State Manager** to inspect and author Model Vista States across the current scene or Prefab Stage.

The manager provides:

- Search by configurable name.
- Type filtering generated from the State component types found in the current context, including Shader and Light States when present.
- Visibility filtering for Visible and Hidden UI states.
- A selectable list of containers with option counts and missing-data status.
- A detail panel that uses the selected container’s current custom inspector for authoring.
- A selection action for locating the owning GameObject.

The window scans inactive objects in open scenes and the current Prefab Stage. Refresh the window after opening or closing scenes, changing Prefab Stage, or adding States through another inspector.



Editor Import And Export

Model Vista Create adds these Unity Editor commands under **InnoWorks > Create**:

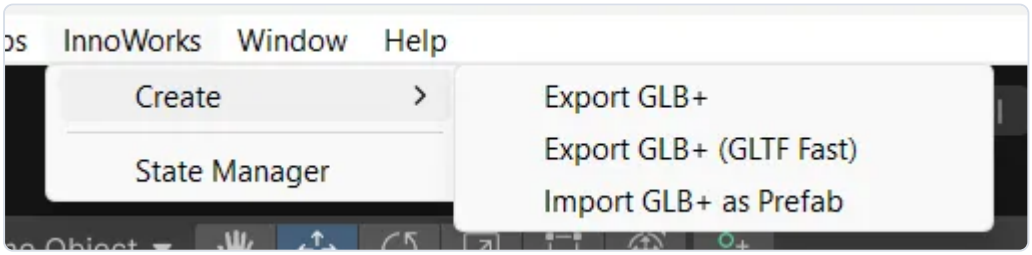
- **Export GLB+ via glTFast (Draco compression)** - the default export and the action used by the Scene View overlay.
- **Export GLB+ via glTFast** - exports uncompressed mesh data.
- **Export GLB+ via Unity GLTF** - optional and enabled when `org.khronos.unitygltf` is installed.
- **Import GLB+ as Prefab** - imports one or more selected or externally chosen GLB files.
- **Import GLB+ Folder as Prefabs** - imports the top-level `.glb` files in a selected folder.
- **Install UnityGLTF Package…** - starts installation of the optional UnityGLTF exporter.

Export A Model Vista GLB

1. Select one or more prefab assets or scene GameObjects.
2. Choose **InnoWorks > Create > Export GLB+ via glTFast (Draco compression)**, or use **Export GLB+** in the Scene View overlay.
3. For one selection, choose a `.glb` destination. For multiple selections, choose a destination folder; Unity writes one GLB per selected root.
4. Review the Unity Console for the export result and any compatibility warnings.

The default glTFast export uses Draco mesh compression with 20-bit position and texture-coordinate quantization. Use **Export GLB+ via glTFast** when uncompressed mesh data is required. UnityGLTF uses the same Model Vista extension payload format, but its mesh compression behavior is backend-specific.

Export writes a binary `.glb` and injects the `INNOWORKS_configurables` root extension. The extension can include States containers, Input Actions, autoplay data, metadata, thumbnails, lights, reflection probes, legacy animation data, material property records, material bindings, inactive GameObject records, and embedded texture payloads.



Import A Model Vista GLB As A Prefab

1. Optionally select one or more `.glb` assets in the Project window.
2. Choose **InnoWorks > Create > Import GLB+ as Prefab**.

3. If no Project assets were selected, choose an external `.glb` file.
4. Choose a prefab destination under `Assets/`, or a destination folder for multiple files.
5. At **GLB+ Texture Import**, choose:
    - **Extract Images** to persist compatible embedded PNG and JPEG 2D textures as image files.
    - **Keep Unity Assets** to keep imported textures as native `.asset` files.
    - **Cancel** to stop the import.
6. Let the importer create the prefab and supporting assets.

Use **Import GLB+ Folder as Prefabs** for batch import. It reads the top-level `.glb` files in the selected source folder and requires the destination folder to be inside `Assets`.

Import restores Model Vista States components, metadata, material assets, thumbnails, lights, reflection probes, animations, material bindings, and compatible exported shader properties and texture payloads. Supporting assets are generated below the import destination in `Materials`, `Textures`, `Thumbnails`, and `Animations` folders as required.

Raw HDR or floating-point textures, texture arrays, WebP/KTX2 resources, ambiguous images, and unsupported texture types remain native Unity assets even when **Extract Images** is selected. This is an Editor persistence choice and does not change runtime GLB loading.

Texture Compatibility

Model Vista preserves high-precision textures and texture arrays, including R16 textures, during GLB export and import.

Scene View Overlay

The package includes a **Model Vista** Scene View overlay for the most common Create workflows.

Use the overlay to:

- Export the selected prefab asset or scene GameObject with **Export GLB+**. This uses the default Draco-compressed glTFast export, not the optional UnityGLTF backend.
- Import a Model Vista GLB with **Import GLB+**. This opens the same prefab-import workflow and embedded-texture choice as **InnoWorks > Create > Import GLB+ as Prefab**.

The export button is enabled only when the current selection can be exported.

Material Restoration And Shader Support

Model Vista GLBs can store material property records, explicit renderer bindings, texture payloads, shader names, shader keywords, and source render-pipeline information. During editor prefab import, Create restores the most compatible material setup that is safe for the target project.

Automatic Material Policy

The default **Auto** policy applies this order:

1. Restore the original shader and properties when the shader exists and is compatible with the active render pipeline.
2. Convert known URP and HDRP Lit or Unlit materials when the source and target pipelines differ.
3. Keep the glTF material when the original shader is custom, unavailable, unsupported, or unsafe for the active pipeline.

Supported URP/HDRP conversion includes the main surface and blend settings, alpha clipping, culling, base color and texture, normal mapping, emission, metallic/smoothness/occlusion data, height approximation, and clear coat where the target shader supports them. Some texture channels are repacked during import to match the target pipeline.

Review the Unity Console after import for fallback decisions and properties that could not be reproduced exactly.

Make Materials Editable

Imported materials can be subassets of the generated prefab and should not be edited in place when you need independent project materials.

1. Select an imported prefab instance.
2. Choose **InnoWorks > Create > Materials > Make Editable Copies From Selection**.
3. Choose a destination folder. The default is `Assets/Materials_Editable`.
4. Edit the copied materials after Create reassigns them to the selected instance.

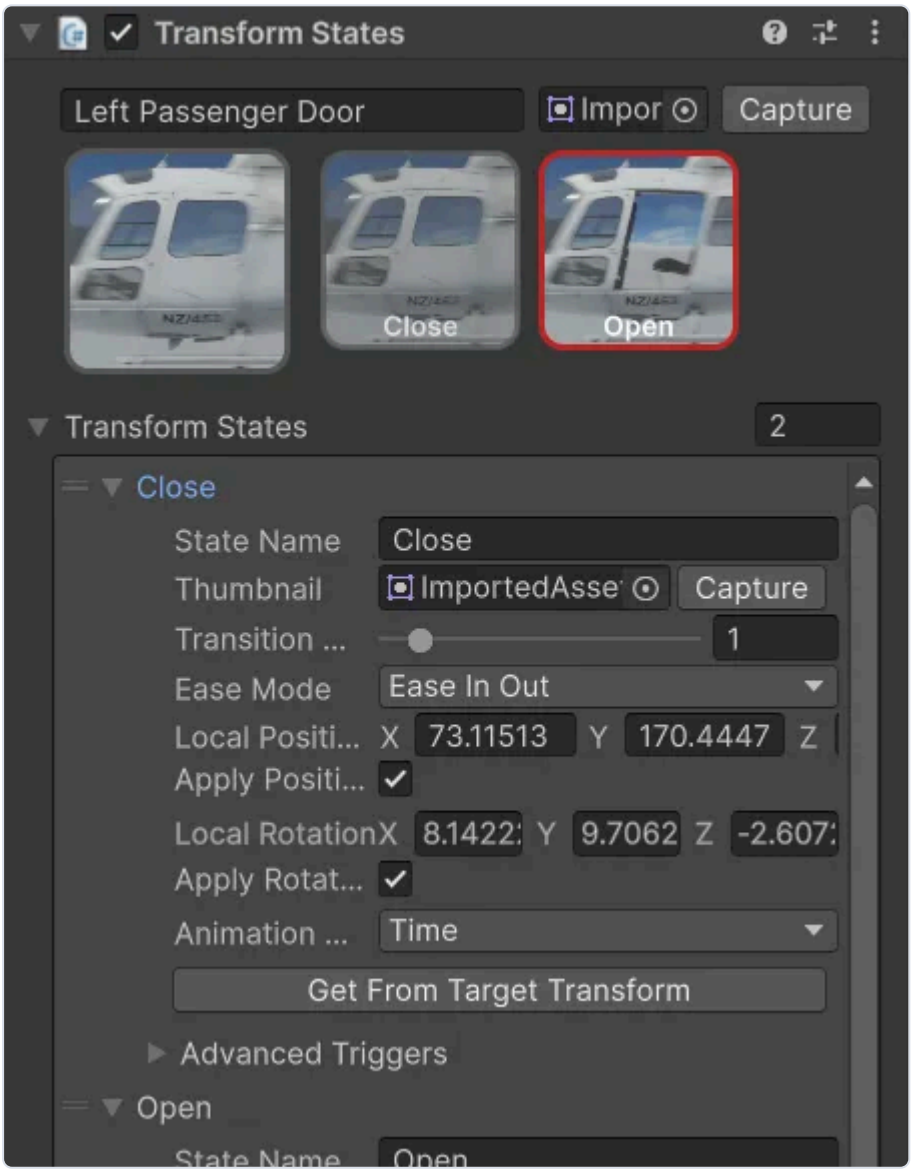
Metadata And Asset Thumbnails

Use `ModelVistaMetadata` to attach string key/value metadata to an exported object. Metadata is serialized into the `INNOWORKS_configurables` extension and restored on import.

Use `ModelVistaAssetThumbnail` to provide an asset-level thumbnail for lightweight browsing before a GLB is imported. Configurables and individual Model Vista options can also carry thumbnails.

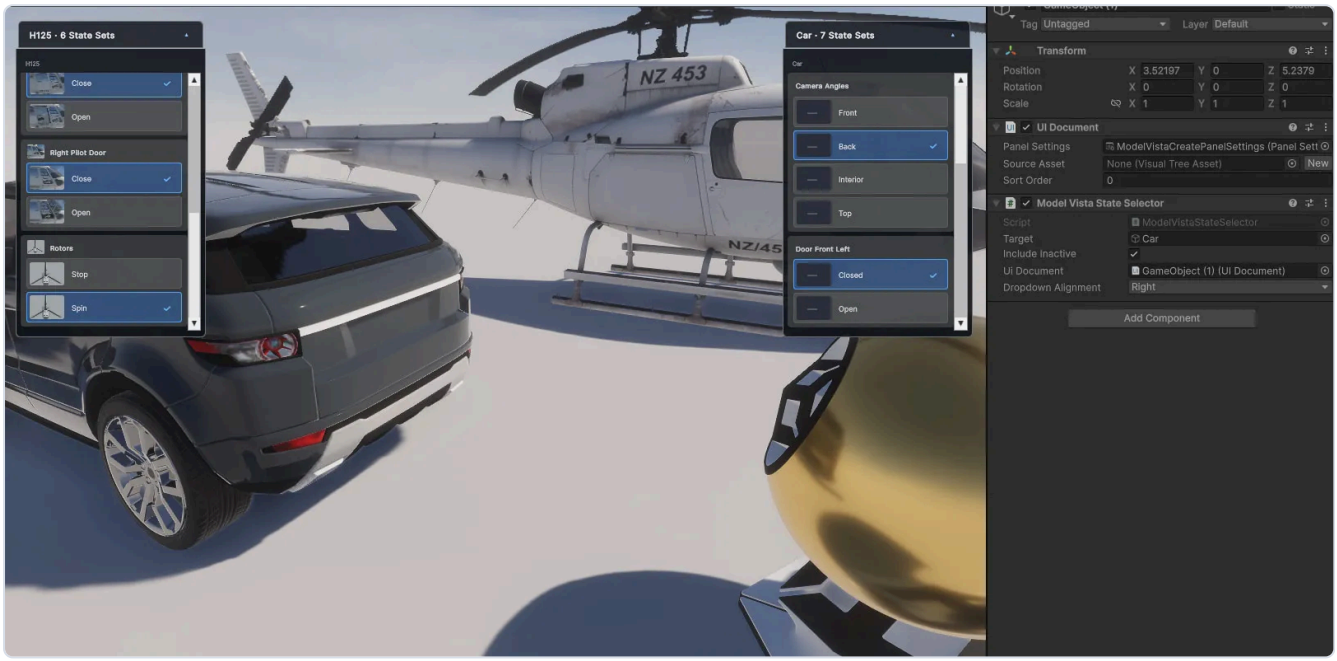
The custom inspectors include thumbnail capture helpers that capture the active Scene view and assign the result to the selected thumbnail field. Frame the Scene view deliberately before capture and review the assigned image before export.

Export embeds thumbnail data with the GLB+ content. Editor import restores generated thumbnail assets below the import destination's `Thumbnails` folder. Runtime browsers, import tools, and review applications can then show meaningful previews before or while users inspect the full model.



State Selector Sample

The optional **State Selector** sample adds a runtime UI for viewing Model Vista State Sets and selecting their options in Play mode. The selector uses the names and thumbnails configured on each State component, so you can test the same presentation data that is available to other Model Vista runtime interfaces.



Import The Sample

1. Open **Window > Package Management > Package Manager**.
2. Select **Model Vista Create**.
3. Open the **Samples** tab.
4. Select **Import** for the **State Selector** sample.

Configure A Selector

1. Add or select a GameObject with a `UI Document` component.
2. On `UI Document`, assign the sample's `ModelVistaCreatePanelSettings` asset to **Panel Settings**. The **Source Asset** can remain empty when you use the layout generated by the sample.
3. Add the `Model Vista State Selector` component.
4. Set **Target** to the root GameObject whose Model Vista State Sets you want to display.
5. Enable **Include Inactive** when State components on inactive objects in the target hierarchy must also be available.
6. Assign the hosting `UI Document` component to **UI Document**.

7. Set **Dropdown Alignment** to position the expanded State Set controls for the available screen space.
8. Enter Play mode and select an option to apply it to the target.

The selector header shows the target name and the number of available State Sets. Expand a State Set to view its options; a check mark identifies the currently selected option.

Control The Displayed Content

The runtime panel uses the Model Vista State data configured on the target:

- **Configurable Name** supplies the State Set heading.
- **Display Name** supplies each option label.
- **Thumbnail** supplies the optional State Set or option image.
- **UI Visibility** controls whether a State Set is available to runtime UI.

To show separate panels for multiple models, configure a `UI Document` and `Model Vista State Selector` for each target. Use **Dropdown Alignment** and the UI Document's **Sort Order** to keep the panels readable alongside other runtime UI.

Files, Assets, And Texture Conversion

GLB+ And Imported Assets

- Model Vista GLB+ files use the standard `.glb` extension.
- Model Vista data is stored in the `INNOWORKS_configurables` root extension.
- Create editor import saves prefab assets under `Assets/`.
- Generated support assets are placed in `Materials`, `Textures`, `Thumbnails`, and `Animations` folders below the chosen import destination as required.
- Asset thumbnails can come from `ModelVistaAssetThumbnail`, configurable thumbnail fields, or option thumbnail fields.

When importing GLB+ content, **Extract Images** writes compatible embedded PNG and JPEG 2D images as ordinary image files. **Keep Unity Assets** retains imported textures as `.asset` files. High-precision, array, WebP/KTX2, ambiguous, or unsupported resources remain `.asset` files even when image extraction is requested.

Convert Texture Assets

Create provides these Project-window asset commands:

- **Assets > INNOWORKS > Texture > Convert to JPG** for PNG images and standalone `Texture2D .asset` files.
- **Assets > INNOWORKS > Texture > Convert to PNG** for JPG/JPEG images and standalone GLB+ texture `.asset` files.

The conversion replaces the selected asset and changes its file extension while preserving its GUID and local file ID so normal serialized Unity references remain valid. String paths stored in scripts, JSON, or other text are not rewritten.

> **⚠Caution**
>
> Texture conversion is destructive. Recoverable backups are written under `Library/ModelVistaCreate/TextureFormatBackups`, but this folder is local project data and should not be treated as source control.

JPG cannot preserve alpha. When an eligible PNG contains alpha, Create asks whether to continue and discards the alpha channel only after confirmation. The JPG command also skips normal maps, linear/data textures, unsupported importer configurations, path collisions, and results that would not reduce file size.

PNG conversion preserves alpha and uses 8-bit output, but the resulting file may be larger than the source JPG or serialized texture asset.

Troubleshooting

Export Command Is Disabled

- Select a prefab asset or scene GameObject.
- Confirm the selected object is not missing scripts required by its Model Vista States.
- Use the main menu command if the Scene View overlay is hidden.

Import Or Export Menu Names Look Different

- The default main-menu command is **InnoWorks > Create > Export GLB+ via glTFast (Draco compression)**.
- Use **Export GLB+ via glTFast** for uncompressed mesh data.
- **Export GLB+ via Unity GLTF** is optional. Install `org.khronos.unitygltf` or choose **Install UnityGLTF Package…** when that backend is required.
- The Scene View **Export GLB+** action always uses the default Draco-compressed glTFast path.

Imported GLB Has No Model Vista States

- Confirm the GLB was exported by Model Vista Create or another exporter that writes `INNOWORKS_configurables`.
- For editor prefab import, check the Unity Console for extension parsing or component restore warnings.

Embedded Textures Were Kept As Unity Assets

- **Extract Images** extracts compatible embedded PNG and JPEG 2D images only.
- HDR/float textures, texture arrays, WebP/KTX2 resources, ambiguous images, and unsupported texture types intentionally remain `.asset` files.
- Choose **Keep Unity Assets** when ordinary image files are not required.

Materials Look Different After Import

- Confirm the target Unity project contains compatible shaders.
- Confirm the active render pipeline matches the exported material or supports a known URP/HDRP conversion.

- Review the Unity Console for exact-restore, cross-pipeline conversion, and glTF fallback decisions.
- Use **InnoWorks > Create > Materials > Make Editable Copies From Selection** before editing imported material subassets.

Texture Conversion Was Skipped

- Confirm the command supports the selected source format.
- JPG conversion skips textures that contain unsupported importer settings, represent normal/linear/data content, collide with an existing path, or would not become smaller.
- JPG cannot retain alpha. Confirm **Convert Anyway** only when losing alpha is acceptable.
- Check `Library/ModelVistaCreate/TextureFormatBackups` when a local recovery copy is needed.