



THE COMPLETE GUIDE · VERSION 1.11.0

Venom Bridge Link

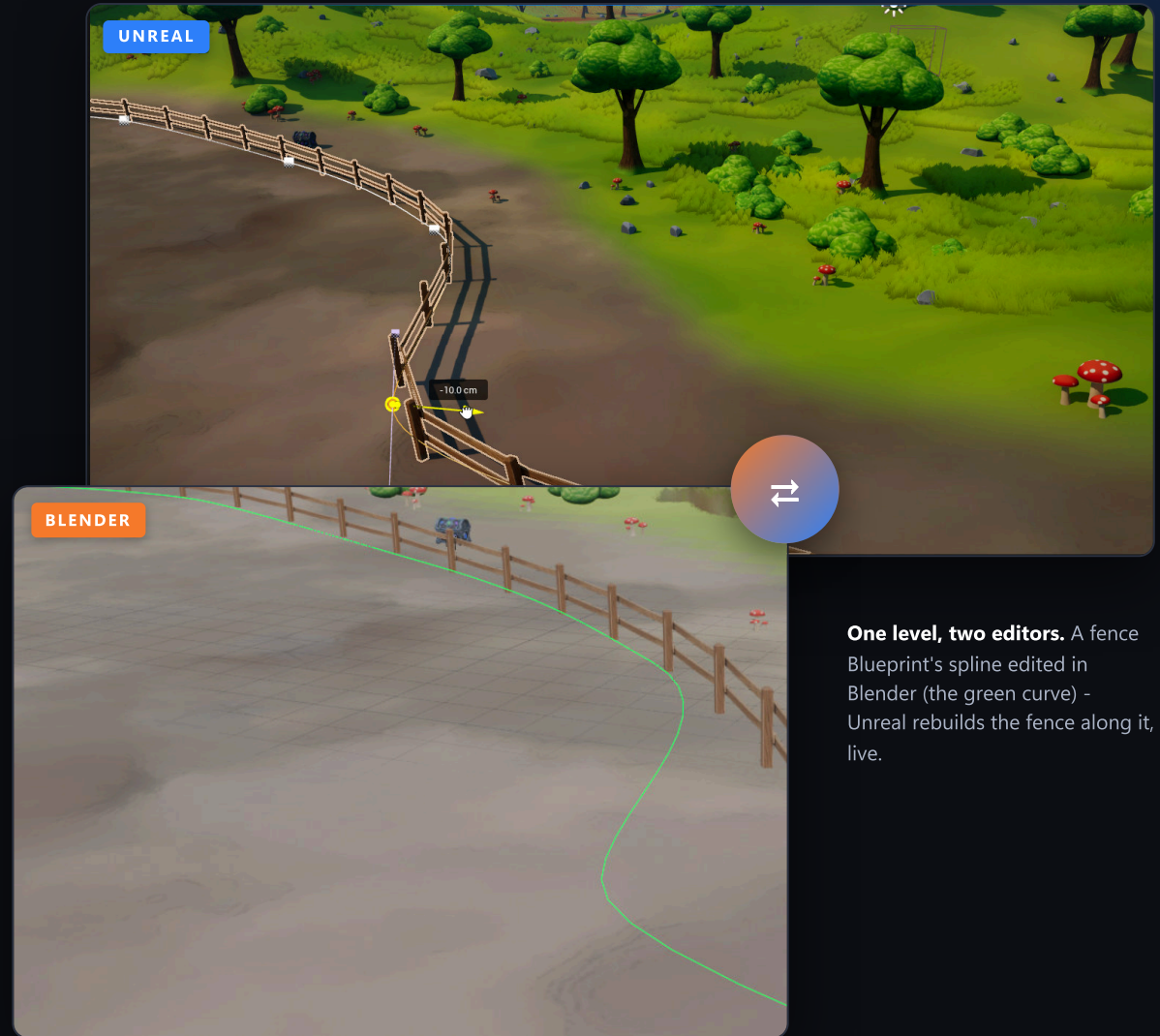
Your Unreal level, live in Blender. Model, light, paint, sculpt and scatter with Blender's tools - Unreal updates while you drag, and every change made in Unreal comes straight back.

Unreal Engine 5.8

Blender 5.2

Windows

Two-way · real time



One level, two editors. A fence Blueprint's spline edited in Blender (the green curve) - Unreal rebuilds the fence along it, live.

Contents

Start with **Quick start** if you just want to connect. The tutorials each cover one feature from start to finish; the last part is about using the link well: use cases, best practices, safety and troubleshooting.

OVERVIEW

Why VenomBridgeLink	3
Features at a glance	4
Feature tour (video)	5
What syncs	6
How it works	7

GETTING STARTED

Install	8
Licence & activation	9
Quick start	10
The panel	11
The panel (2)	12
Both apps as one	13

TUTORIALS

1 · Set dressing	14
2 · View Sync, lights, cameras	15
3 · Materials & textures	16
4 · Exact Materials	17
4 · Exact Materials (2)	18
5 · Meshes & object data	19
6 · Geometry Nodes scatter	20
7 · Foliage	21
8 · Curves & text	22
9 · Blueprint splines	23
10 · Landscape sculpting & grass	24
11 · Landscape paint layers	25
12 · Rigs & animation	26
13 · Levels	27
14 · Working offline	28
14 · Offline cards	29

IN PRACTICE

Use cases	30
Use cases (2)	31
Best practices	32
Safety & performance	33
Troubleshooting	34
Limits & compatibility	35

SCREENSHOTS

BLENDER

UNREAL

Every screenshot is real: the add-on and the plugin running live, side by side.

DIRECTION

BLENDER → UNREAL

UNREAL → BLENDER

BOTH WAYS

What travels which way.

KEYS AND LABELS

Keys look like **Shift** + **A** ; buttons and menus in the panel are in **bold**, e.g. **Send to Unreal**.

BOXES

TIP

GOOD TO KNOW

LIMITS

One level. Two apps. No export loop.

Unreal decides what exists; Blender becomes a live editor for it.

Open a level in Unreal, click **Connect** in Blender, and the level appears in Blender at its exact transforms. From then on, whatever you change in either app shows up in the other one **while you work**: actors, lights, cameras, materials, painted textures, terrain - even the viewport camera.

The export loop

1. Model in Blender, export an FBX
2. Import into Unreal, fix scale, rotation and pivot
3. Reassign materials, re-import textures
4. Place it, look at it in the level...
5. ...back to Blender, and again for every tweak

With VenomBridgeLink

- The **real level** in Blender: meshes, lights, cameras, Blueprints, foliage, landscape
- Move, model, paint, sculpt - Unreal follows **while you drag**
- Change it in Unreal - Blender follows
- Every change is **one Ctrl+Z** in Unreal
- Unreal closed? Keep working, then **choose who wins**

WHO IT'S FOR

Environment and level artists, technical artists, indie teams and solo developers who build in Unreal but prefer Blender's modelling, sculpting, painting and Geometry Nodes.

~1 ms

for a material value change to reach Unreal:
no re-import, no shader compile

~8 ms

to re-import a texture you are painting,
while you keep painting

0.001 cm

transform accuracy, verified on rotated,
non-uniformly scaled actors and single
vertices

386 tests

Exact Materials matched 1:1 against Cycles:
325 node-image tests + 61 mesh tests,
Nanite included

~1.1 ms

typical Unreal editor cost per frame on a
270-actor level

25+

objects deleted at once never vanish in
Unreal without your OK (mass-delete
guard)

Everything it does, at a glance

Twelve areas, each with its own tutorial in this guide.

01

Live level mirror

Static meshes, lights, cameras and Blueprints. Move, duplicate, rename, re-folder or delete in either app; the other follows while you drag.

→ page 14

02

Both apps as one

Show in Unreal / Show in Blender jump to the selection in the other app; Unreal's toolbar shows the link; View Sync keeps the cameras together.

→ page 13

03

Materials & textures

Full PBR material instances with their textures. Values reach Unreal in ~1 ms; painted textures are re-imported in ~8 ms.

→ page 16

04

Exact Materials

A Blender node tree becomes one Unreal shader running Cycles' own maths - verified 1:1, on Nanite meshes too.

→ page 17

05

Meshes & object data

Meshes update in place; Unreal's meshes get a safe copy or Edit Original with a backup. UVs, vertex colours, weight masks, tags.

→ page 19

06

Geometry Nodes scatter

Scatter in Blender, get Instanced Static Mesh components in Unreal - one per mesh, Nanite when the mesh is, live with every slider.

→ page 20

07

Foliage editing

Unreal foliage and ISM / HISM instances as points: move, duplicate and delete them in Edit mode. Foliage stays on the landscape.

→ page 21

08

Curves & text

Curves, text, surfaces and metaballs become live static meshes. Text Render actors come over as editable text objects.

→ page 22

09

Blueprint splines

Fences, roads and rails: edit a Blueprint's spline points in Blender and its construction script rebuilds it in Unreal.

→ page 23

10

Landscapes

Exact heights. Sculpt in Blender, paint the real paint layers with their textures showing, and see Unreal's grass.

→ page 24

11

Rigs & animation

An armature with skinned meshes becomes a Skeletal Mesh actor; poses and actions play right in the Unreal editor.

→ page 26

12

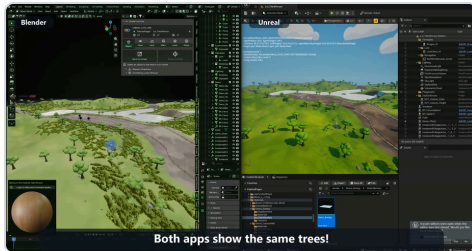
Levels & offline work

Browse every level and send objects into any of them. Close Unreal, keep working, and choose who wins when it's back.

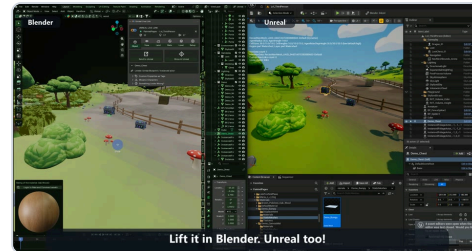
→ page 28

Feature tour: the demo video

The demo video [LiveLink_Demo.mp4](#) (9 min 10 s) in twelve chapters - Blender on the left, Unreal on the right. A still from each, and the time it starts.



- 01 Say Hello** 0:07
Connect: the same level appears in both apps, and you choose which side wins.



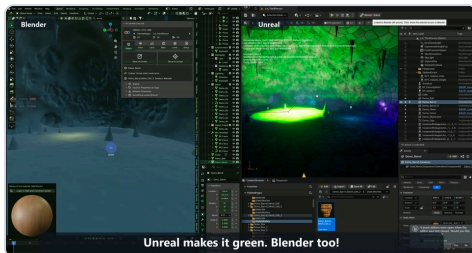
- 02 Move Things** 0:35
Select, lift, spin and scale in Blender - Unreal follows while you drag. And back again.



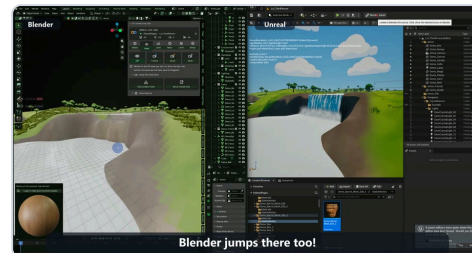
- 03 Add New Things** 1:11
New boxes, copies, names, barrels added in Unreal and tags travel both ways.



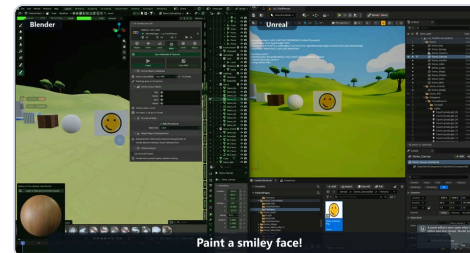
- 04 Build and Bend** 1:51
Type 3D letters, stretch meshes, scatter mushrooms and bend a fence Blueprint's spline.



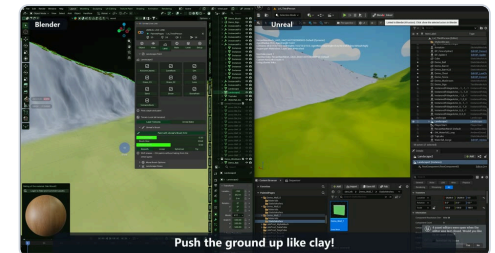
- 05 Lights and Camera** 2:47
Add a lamp, change its colour on either side, and drop in a camera.



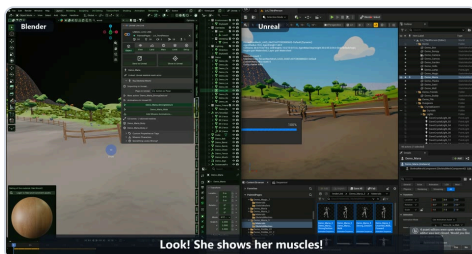
- 06 Look Together** 3:20
View Sync: Follow, Drive, Both, matching zoom - and one-off jumps to the other view.



- 07 Colours and Paint** 3:55
Live material values, texture painting, Exact procedural materials, bakes and bumps.



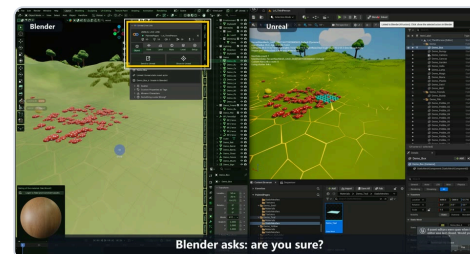
- 08 Hills and Trees** 5:13
Sculpt the landscape, paint its layers, and pick, copy or remove a single tree.



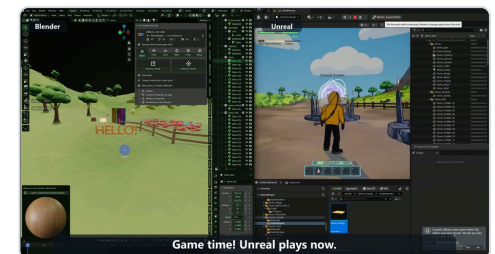
- 09 A New Friend** 6:11
Import a Mixamo character: her animations and poses play in Unreal.



- 10 On and Off** 6:41
Pick a level by name, switch selection sync, mirror only some things, hide the top-bar button.



- 11 Oops, All Safe** 7:22
Undo on either side, the mass-delete guard, and choosing who wins after working offline.



- 12 Play and Tidy** 8:23
Unreal plays (PIE) and the link waits; your Blender edit lands when play stops.

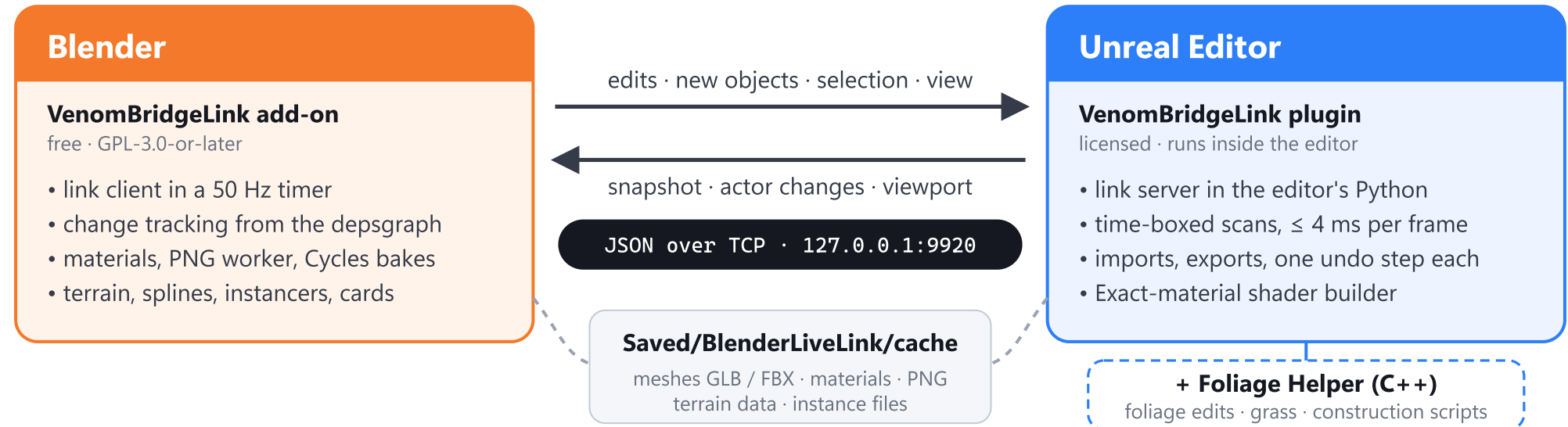
What syncs - and in which direction

Anything inside the collection **Unreal: <Level>** is live.

YOU WORK WITH...	UNREAL → BLENDER	BLENDER → UNREAL
Static meshes	an object (a box first, then the real mesh and textures)	an actor; the mesh is imported to <code>/Game/BlenderLink/<Name>/</code>
Lights & cameras	light / camera objects with matching settings	Point, Spot, Rect, Directional lights and Cine Cameras: colour, unit-correct intensity, radius, angles, size, shadows, focal length
Move · rotate · scale	updates while you drag	updates while you drag
Duplicate · rename · folders	object copy · new name · matching child collection	a real Unreal duplicate (overrides and every setting kept) · actor label · Outliner folder
Select · delete	selects · removes the object	selects · removes the actor (one Ctrl+Z in Unreal brings it back)
Blueprint actors	parts that move as one object; spline components as editable curves	move the actor; edit its spline points → the Blueprint rebuilds along them
Curves, text, surfaces, metaballs	-	a static mesh of what you see (bevel, extrude, modifiers), live while you edit - typing included
Text Render actors	text objects: same text, size, alignment, colour, facing	edit text, size, alignment or colour → the Text Render, live
Foliage, ISM / HISM	one instancer per component, each instance a point	move, duplicate or delete instances in Edit mode
Landscape	terrain mesh with exact heights, layer textures, paint layers as images	sculpt → heights · paint → paint layers (one Ctrl+Z each)
Landscape grass	one read-only instancer per grass variety	- (paint or sculpt; Unreal regrows it)
Materials	baked base colour, roughness, metal, normal, emissive; overrides as object-linked slots	full PBR instance + textures, or Exact node-tree shaders; slot assignments and overrides
Mesh data	all UV channels, vertex colours, actor tags → custom properties	UV maps, vertex colours, weight-paint masks, custom properties → actor tags
Rigs	-	armature + skinned meshes → Skeletal Mesh actor, morph targets, animation
Viewport	Blender's view follows (View Sync)	Unreal's viewport follows (View Sync)

How it works

Two parts that talk over a local socket: a free Blender add-on and a licensed Unreal plugin, plus an optional C++ helper for the things editor Python can't do.



The connection

Newline-separated JSON over TCP on `127.0.0.1:9920` - it never leaves your PC. A second editor uses 9921-9925 (set the port in Blender). The add-on and the plugin belong together: use the same version on both sides.

Light on both apps

Unreal is checked in time-boxed slices (≤ 4 ms of scanning per frame, selected actors at 45 Hz for live drags). Blender only re-reads what its dependency graph reports as changed. PNG encoding, bakes and Unreal's textured exports run in the background or while you are in the other app.

Files

Meshes travel as GLB or FBX, materials as a GLB or a `.gltf` + images, through the project's `Saved/BlenderLiveLink/cache/` folder, which both apps can always see.

Coordinates

Unreal: left-handed, Z-up, centimetres. Blender: right-handed, Z-up, metres.

$p = (x, -y, z) / 100$, $R = S \cdot R \cdot S$ with $S = \text{diag}(1, -1, 1)$; lights and cameras add the axis fix. Exact to 0.001 cm, both ways.

YOU DRAG IN BLENDER

The object moves

the depsgraph reports the change

NEXT TICK (50 HZ)

The add-on sends it

one small JSON message, only what changed

NEXT EDITOR FRAME

Unreal applies it

as one undo step, the actor moves

AND BACK

Unreal's own edits

are found by its time-boxed scan and sent the same way

Install in five minutes

A plugin folder for the Unreal project, an extension for Blender.

Unreal - once per project

- 1 Copy the `VenomBridgeLink` folder into `<YourProject>/Plugins/` (create `Plugins` if needed). Copy `VenomBridgeLinkFoliage` there too - the **Foliage Helper**.
- 2 Start the editor. The plugin switches on the four engine plugins it needs (Python Editor Script Plugin, Editor Scripting Utilities, glTF Exporter, Interchange). If the editor asks to restart, restart it.
- 3 The link server starts by itself with the editor. Menu: **Tools** → **VenomBridgeLink** → Start / Stop / Print Status / License Status.
- 4 Activate it once on this PC (next page). Upgrading from *Unreal Live Link*? Delete the old `BlenderLiveLink` folders first - your activation carries over.

```
<YourProject>/
  Plugins/
    VenomBridgeLink/      ← the link
    VenomBridgeLinkFoliage/ ← the Foliage Helper (C++)
  Saved/BlenderLiveLink/cache/ ← files in transit
```

Blender - once

- 1 **Edit** → **Preferences** → **Get Extensions** → ▾ (top right) → **Install from Disk...** and pick `venom_bridge_link-1.11.0.zip`.
- 2 In the 3D View press **N** and open the **VenomBridgeLink** tab - or click the **Unreal** button at the right of Blender's top bar. Upgrading? Uninstall *Unreal Live Link* first - your linked scenes keep working.

Requirements

Unreal Engine	5.8, Windows (the plugin's compiled core is built for Windows)
Blender	5.2
Foliage Helper	<code>VenomBridgeLinkFoliage</code> , built for UE 5.8 Win64 (its Source rebuilds it for others)
Network	none for the link (127.0.0.1); store keys check in online every 3 days, the free trial starts online
Same version	always use the add-on and the plugin from the same release

THE FOLIAGE HELPER

Everything works without it except **foliage editing**, **landscape grass** and **Blueprints rebuilding** along splines edited in Blender (Python can't rerun construction scripts). Blender tells you once when a feature needs it.

CHECK THAT IT RUNS

In Unreal, **Tools** → **VenomBridgeLink** → **Print Status** writes the server's state to the Output Log. In Blender the panel says *Waiting for Unreal on 127.0.0.1:9920* until the editor is up, then connects by itself.

TIP

Install the plugin per project, the add-on once: one Blender works with every project that has the plugin.

Licence & activation

The Blender add-on is free (GPL-3.0-or-later). The Unreal plugin is the paid part: activate it once per PC.

Activate from Blender

- 1 Connect to Unreal from Blender.
- 2 Open **Setup** → **License** - or click **Enter License Key...** on the red card.
- 3 Paste your key → **OK**. The card turns to **Activated**; the level starts mirroring.
- 4 No key yet? Click **Start Free 1-Day Trial** on the same card: everything works for 24 hours, once per PC, with the time left on the card.

BEFORE ACTIVATION

Blender connects, but the panel shows *Unreal plugin not activated* and nothing is mirrored. The key goes to the Unreal plugin, which keeps the activation - per Windows user, sealed to the PC.

From Unreal alone

```
import ue_blender_link as L; L.activate("KEY"), L.deactivate(),
L.license_status() - also Tools → VenomBridgeLink → License Status.
```

Buy a key · get help

Keys: payhip.com/b/bzcvZ - or **Setup** → **License** → **Get a License**. Help, updates and contact: VenomBridgeLink.com.



KEY	HOW IT WORKS
Store key bought	Activated online on up to 2 PCs . Re-checked every 3 days in the background; keeps working 30 days offline . Deactivate This PC gives the seat back - do it before you change PCs.
Friend key from the author	Made for one PC : send the author that PC's machine code (shown under <i>This PC</i> , with a copy button). Works offline, may have an end date.
Free trial no key	24 hours of every feature, once per PC (the licence server remembers the machine code). Needs the internet to start; when it ends the card asks for a key, and a key entered during the trial takes over at once.

- Copying a project, the plugin or the activation file to another PC does not carry the activation.
- Store keys send only the key and an instance name (computer name + machine code) to the licence server, the trial only that name - never project data.
- *reached the activation limit*: deactivate a PC you no longer use. *could not be reached for 30 days*: connect that PC to the internet once.

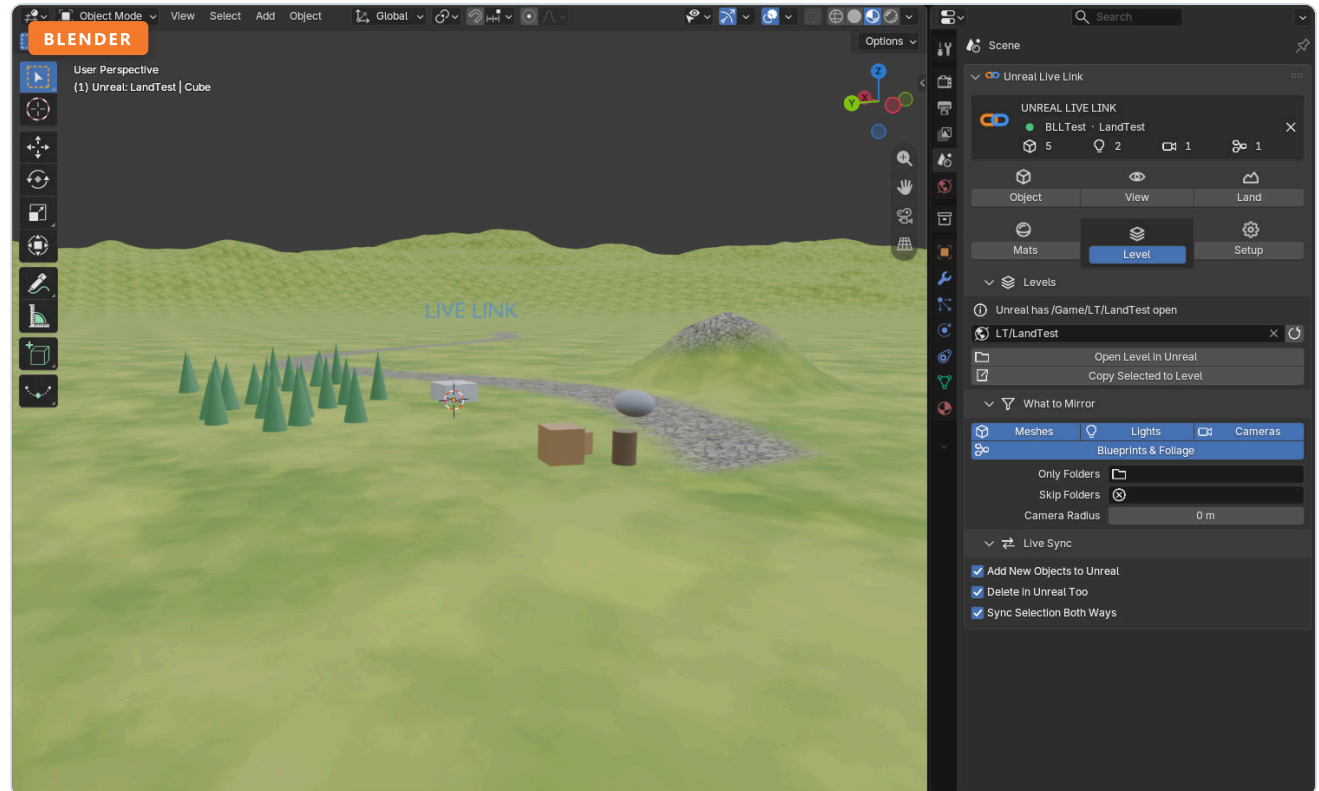
Quick start: your first live session

Five minutes from an open level to editing it in Blender.

- 1 Open your level in Unreal. The link server starts with the editor.
- 2 In Blender: **N** → **VenomBridgeLink** → **Connect to Unreal**.
- 3 The level arrives in the collection **Unreal: <Level>**; Unreal's Outliner folders become child collections. Meshes show as boxes first, then the real mesh and textures.
- 4 Grab something (**G**) and move it: the actor moves in Unreal **while you drag**.
- 5 **Shift** + **A** adds straight into Unreal (the level collection is active). To send an object you already have, select it → **Object** tab → **Send to Unreal**, or drag it into the collection. **Show in Unreal** jumps there.
- 6 Save the level in Unreal as usual. Saving the .blend while linked makes the next connect faster.

TIP

The **Unreal** button in the top bar opens the same panel from any workspace, and turns red when something needs you.



A connected session: the level **LandTest** in Blender, the link panel on the right. Counts on the card: meshes, lights, cameras, Blueprints, rigs.

The level collection

Everything in **Unreal: LandTest** is live; Outliner folders are child collections.

The status card

Project · level, linked counts, what is loading or sending - and every choice card.

Six tabs

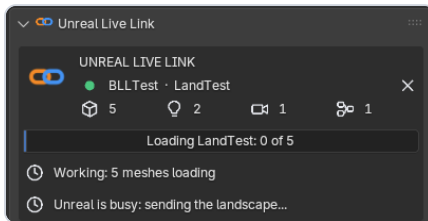
Object, View, Land, Mats, Level, Setup - one per kind of work; the next two pages.

The panel, tab by tab

Everything is in one panel: the VenomBridgeLink tab of the 3D View's sidebar (N). Under the status card, six tabs - Object, View and Land here, the other three on the next page.

Status card

Project · level, what is linked (meshes, lights, cameras, Blueprints, rigs), what is still loading or sending, what Unreal is busy with, PIE pauses and problems (with a dismiss button). ⊗ disconnects. Choice cards (which side wins, another level, missing actors) appear right under it.



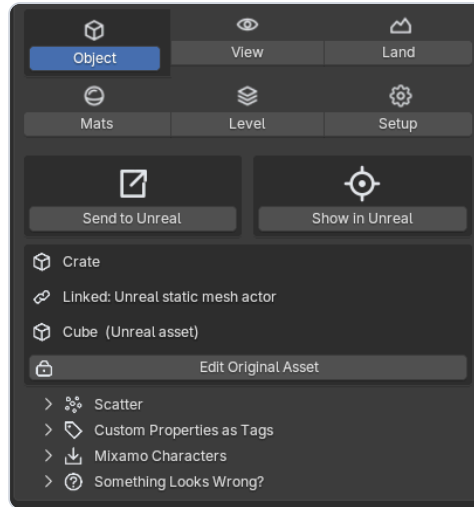
Loading a level: progress, and what Unreal is busy with.



The **Unreal** button in the top bar: the same panel as a popover.

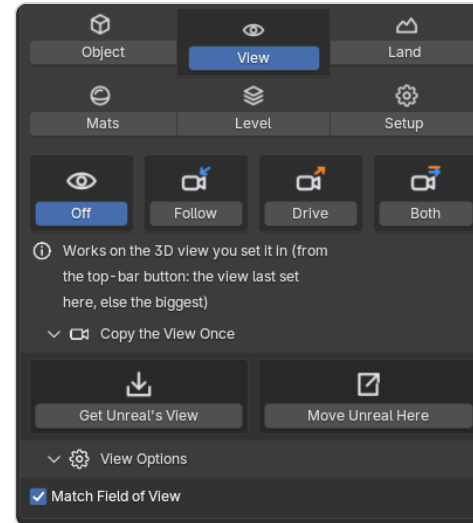
Six tabs

One for each kind of work. Each tab is a picture over its name: two rows of three in Blender's sidebar, one row when the panel is wide, pictures only when it is very narrow. Sections you need less often start folded - click a heading to open it.



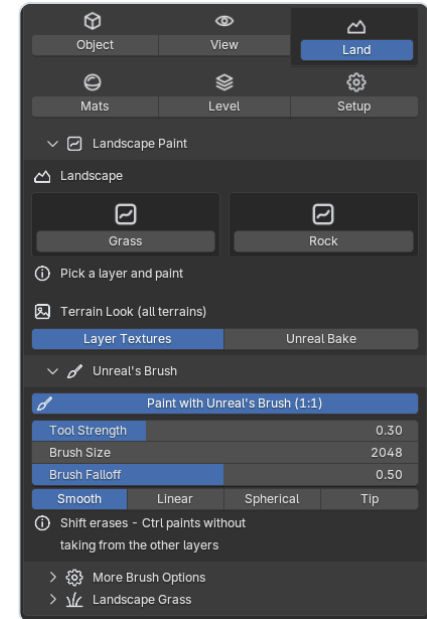
Object tab

- **Send to Unreal** and **Show in Unreal** (select, frame, bring Unreal to the front).
- The active object's card: its actor and asset (Unreal asset / made in Blender / edited copy), **Edit Original Asset**, Send / Retry when it isn't linked.
- Sections: **Rig**, **Scatter**, **Custom Properties as Tags**, **Mixamo Characters**, **Something Looks Wrong?** (Resync from Unreal).



View tab

- The View Sync mode: **Off**, **Follow**, **Drive**, **Both** - for the 3D view you set it in.
- **Copy the View Once**: **Get Unreal's View** / **Move Unreal Here**.
- **View Options**: **Match Field of View** keeps both lenses the same.

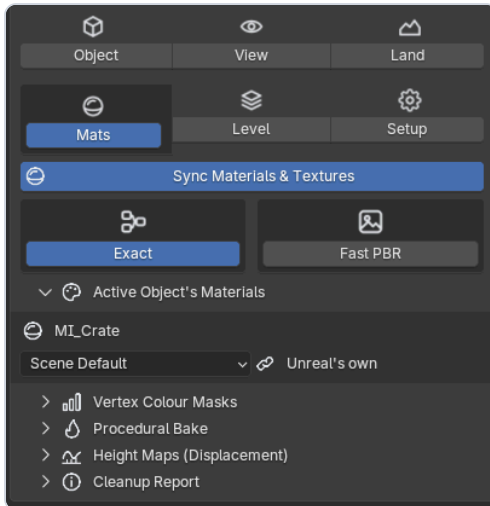


Land tab

- **Landscape Paint**: a card per paint layer - one click selects the terrain and starts painting - and the terrain's look.
- **Unreal's Brush**: paint with Unreal's own brush, 1:1 - strength, size, falloff; **More Brush Options**.
- **Landscape Grass**: mirror it or not, the **Grass Limit**, the terrain's grass.

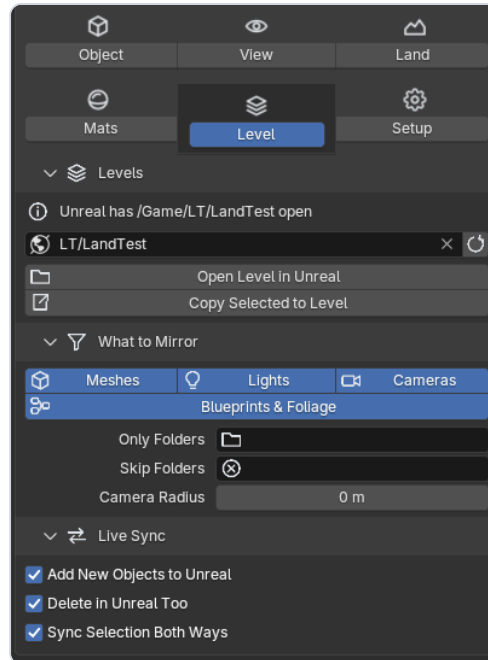
The panel, continued

Mats, Level and Setup. A scene saved on 1.9's old Link tab opens on Object.



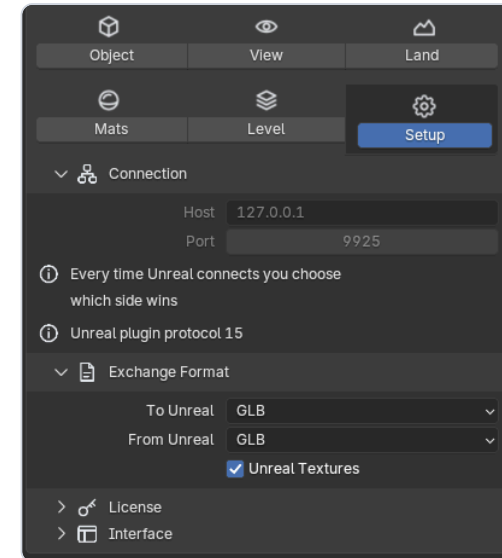
Mats tab

- **Sync Materials & Textures** on / off; **Exact** or **Fast PBR** material mode.
- The active object's slots and their state in Unreal (in Unreal / sending / Unreal's own), each with its own mode: Scene Default / Exact / Fast PBR.
- **Vertex Colour Masks**, **Procedural Bake** (noise, ramps and mixes baked with Cycles; needs UVs), **Height Maps**, **Cleanup Report**.



Level tab

- **Levels**: search the project's levels, **Open Level in Unreal**, **Copy Selected to Level**.
- **What to Mirror**: meshes / lights / cameras / Blueprints & foliage, **Only Folders** / **Skip Folders**, **Camera Radius** around Unreal's camera.
- **Live Sync**: add new objects, delete in Unreal too, sync the selection both ways.

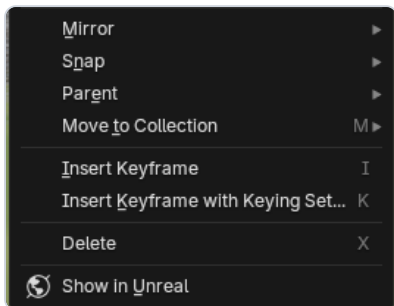
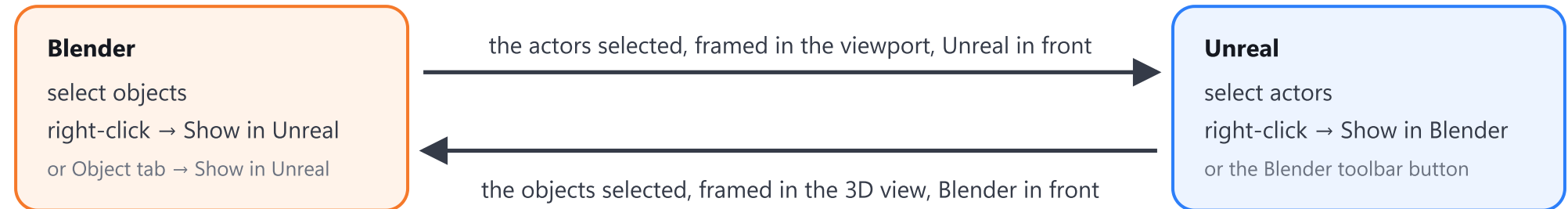


Setup tab

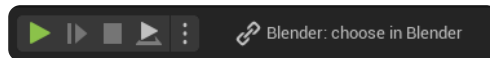
- **Connection**: host / port.
- **Exchange Format**: To Unreal GLB or FBX (both give the same full PBR materials); From Unreal GLB (recommended - Unreal's FBX exporter writes no textures); **Unreal Textures**.
- **License** (page 9) and **Interface** (the top-bar button).

Both apps as one

Jump to the selection in the other app, see the link's state in both, and undo on either side safely.



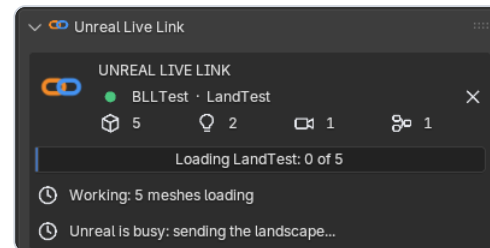
Blender: right-click → **Show in Unreal**.



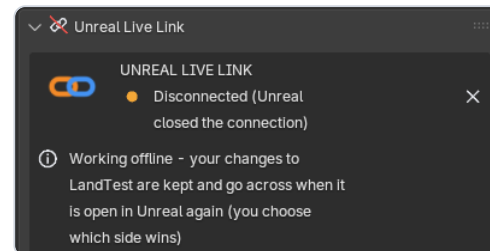
Unreal's toolbar while Blender waits for an answer.

Unreal's toolbar

- **Blender: linked** - click to show the selected actors in Blender.
- **choose in Blender** - Blender asks which side wins or which level: click to go there.
- **not connected, paused (Play), off** at a glance.



Loading a level: a progress bar, and what Unreal is busy with meanwhile.



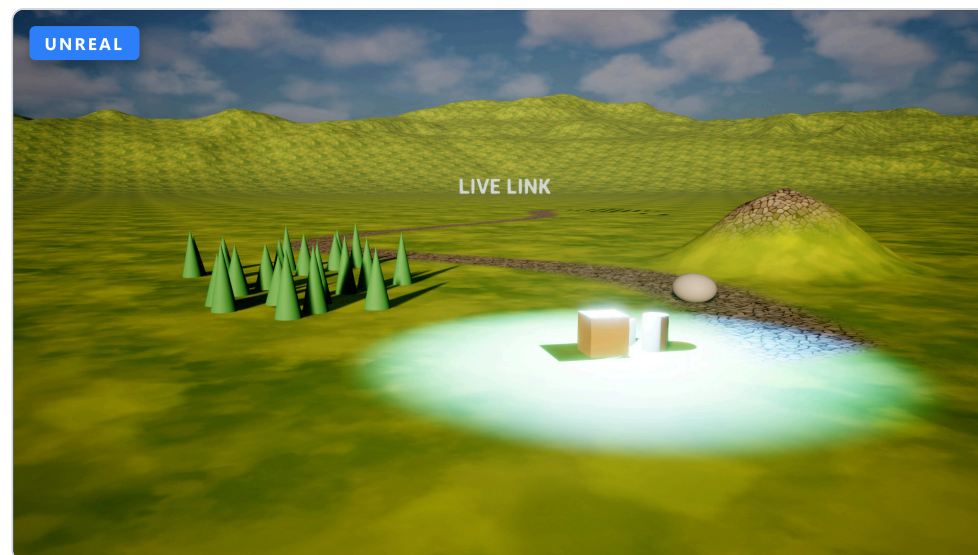
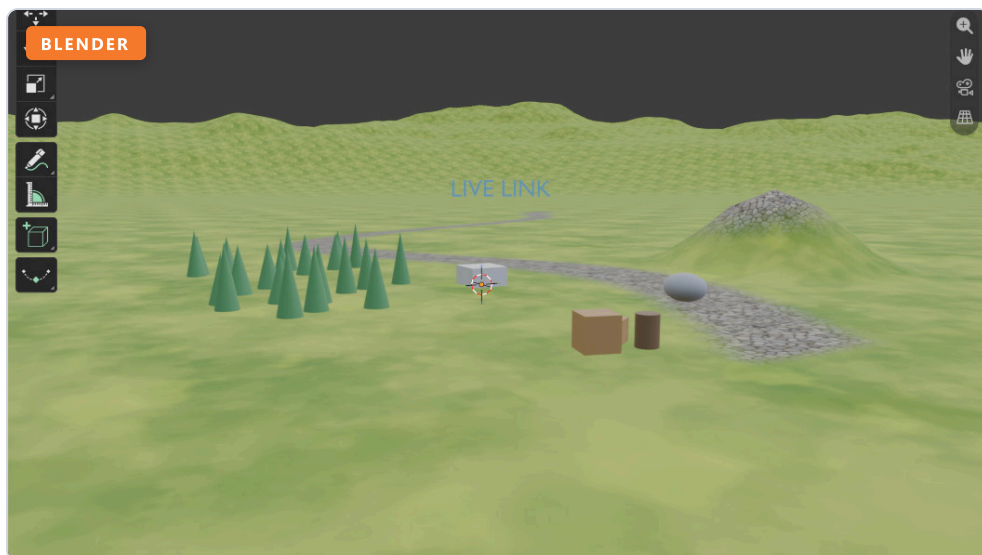
Unreal closed: your work is kept and goes across when the level is open again.

Safe on both sides

- What comes from Unreal is **one undo step** in Blender, and each move from Blender is one in Unreal: Ctrl+Z on either side undoes that side's last step, never the other app's edits.
- **One Blender per editor**: a second one is told; *Connect* there takes the link over.
- The link **never runs inside Unreal's own saves**, progress dialogs or undo.
- A mesh edited in Unreal (Modeling Mode, LOD or Nanite settings) goes to Blender again.

Set dressing, live

Use Blender's tools on the real level. Left: Blender's view; right: Unreal at the same moment, same camera.



- 1 Move, rotate or scale with **G** **R** **S** or the gizmo - the actor follows while you drag.
- 2 **Shift** + **D** duplicates: Unreal makes a **real duplicate** of the actor (overrides and every setting kept).
- 3 **F2** renames the actor. Move objects between child collections to change Outliner folders.
- 4 **Shift** + **A** a mesh, light, camera, curve or text: it becomes an actor in Unreal.
- 5 **X** deletes the actor - one **Ctrl** + **Z** in Unreal brings it back.

SAFE BY DESIGN

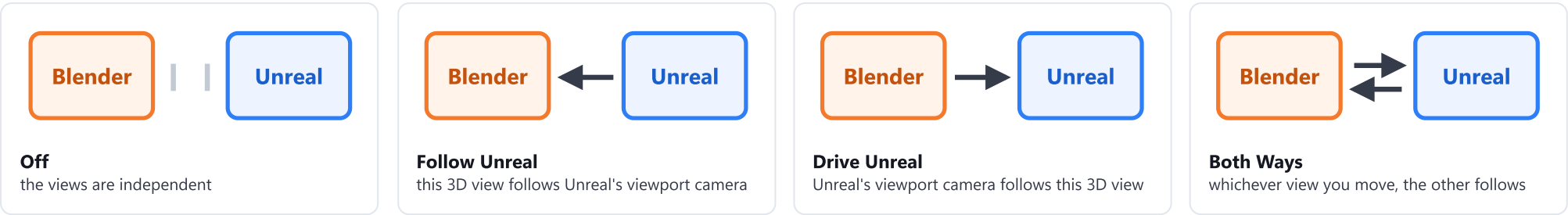
Undo in Blender is safe: undoing a move moves the actor back, undoing a delete re-adds it. More than 25 objects deleted at once wait for **Delete in Unreal** or **Restore**. Every change is one undo step in Unreal (*Edit* → *Undo History*: "Blender Link: ...").

BIG LEVELS

Level tab → **What to Mirror** → **Camera Radius** mirrors only actors near Unreal's camera (they stream in and out); **Only Folders** / **Skip Folders** pick Outliner folders.

View Sync, lights and cameras

Both viewports can follow each other, so you always look at the same thing in both apps.

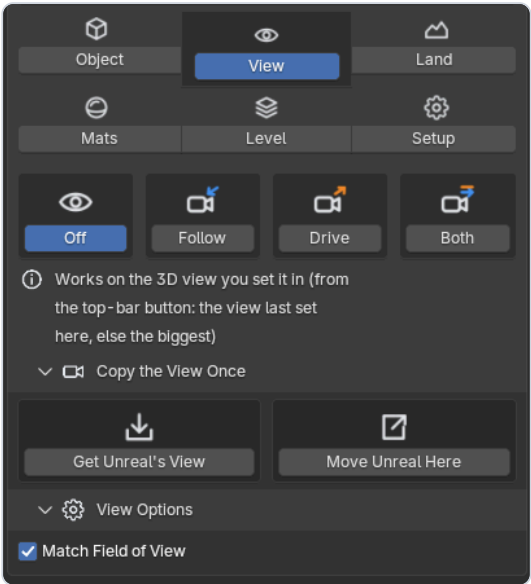


Set it up

- 1
- Hover the 3D view you want synced, open the **View** tab and pick a mode.
- 2
- Turn on **Match Field of View** so both lenses agree and compositions line up.
- 3
- Copy the View Once: Get Unreal's View / Move Unreal Here** copy the view once, without staying synced.

LIGHT IN BLENDER, JUDGE IN LUMEN

Use **Drive Unreal** while you place lights in Blender: Unreal shows the final, Lumen-lit result from exactly Blender's angle - on a second monitor it feels like one app.



Lights and cameras

BLENDER	UNREAL
Point light	Point Light
Spot light	Spot Light
Area light	Rect Light
Sun light	Directional Light
Camera	Cine Camera Actor

Colour, intensity (unit-correct), radius, spot angles, size and shadows go both ways; cameras keep their focal length.

Two monitors

Blender on one screen, Unreal on the other, **Drive Unreal**: model and light in Blender, judge the final frame in Unreal.

Frame a shot

Add a camera in Blender and set its focal length: it is a **Cine Camera** actor in Unreal, ready for Sequencer.

Fly in Unreal, edit in Blender

Follow Unreal: wherever you fly in Unreal's viewport, Blender's view is already there when you switch.

Materials and textures

Full PBR materials with their textures, live while you paint.

Three update speeds, picked automatically

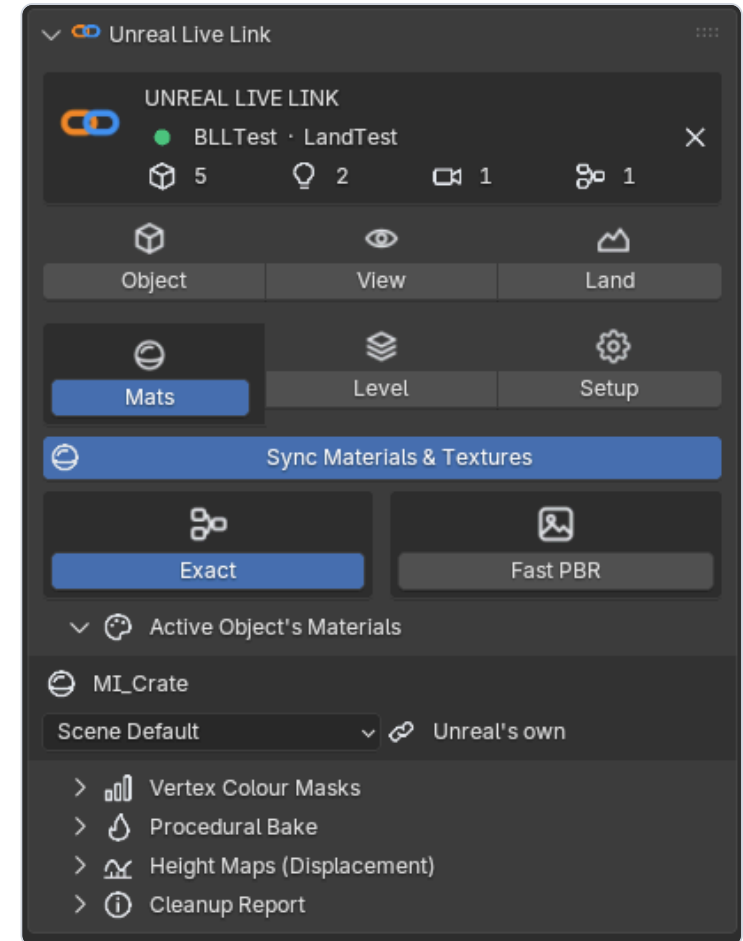
Value change	~1 ms	colour, roughness, metallic, emission... only the parameter changes in Unreal - no re-import.
Texture painting	~8 ms	the painted image is encoded on a background thread and that one texture is re-imported, while you keep painting.
Structure change	~1 s	new image, node layout or slot: the material is re-sent as a small carrier (GLB, or .gltf + images with FBX).

Paint a texture live

- 1 Give the mesh UVs and a material with an Image Texture.
- 2 Switch to **Texture Paint** and paint. Each pause re-imports that one texture in Unreal.
- 3 Drag the roughness or colour sliders: Unreal follows in about a millisecond.

Good to know

- **PBR texture sets** (BlenderKit, Poly Haven, Substance): base colour, roughness, metallic (packed), normal (OpenGL flipped for Unreal), emissive, alpha. Mapping-node tiling, offset, rotation carry over.
- Assigning a material in Blender assigns it in Unreal: on the asset for Blender meshes, as a **per-actor override** on Unreal's meshes (the asset is never changed).
- Objects that **share one mesh** can each have their own material: a slot set to *Link: Object* becomes a per-actor override in Unreal.
- Unreal's overrides arrive as object-linked slots; an Unreal material stays the real one.
- **Height maps** → Displacement node → real Nanite tessellation; Midlevel and Scale update in ~1 ms.



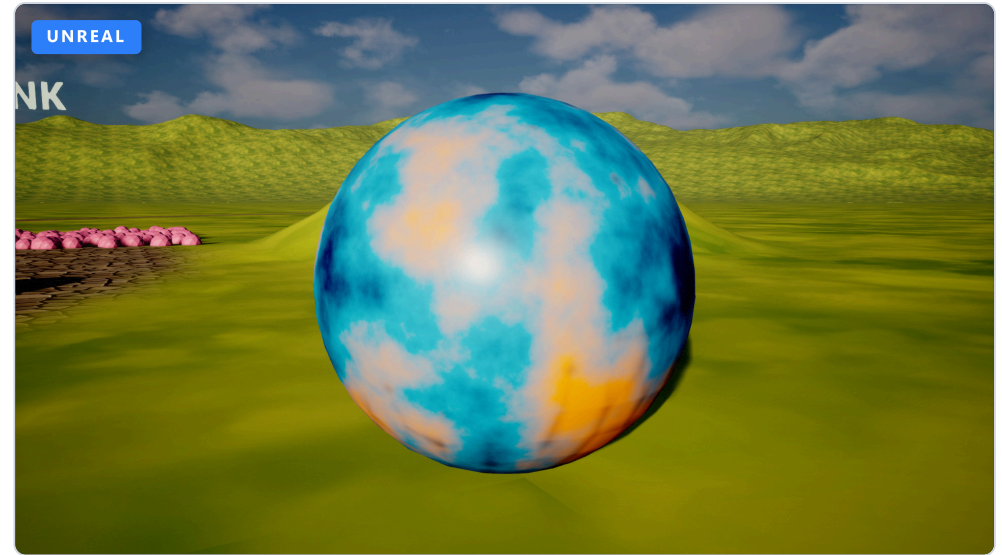
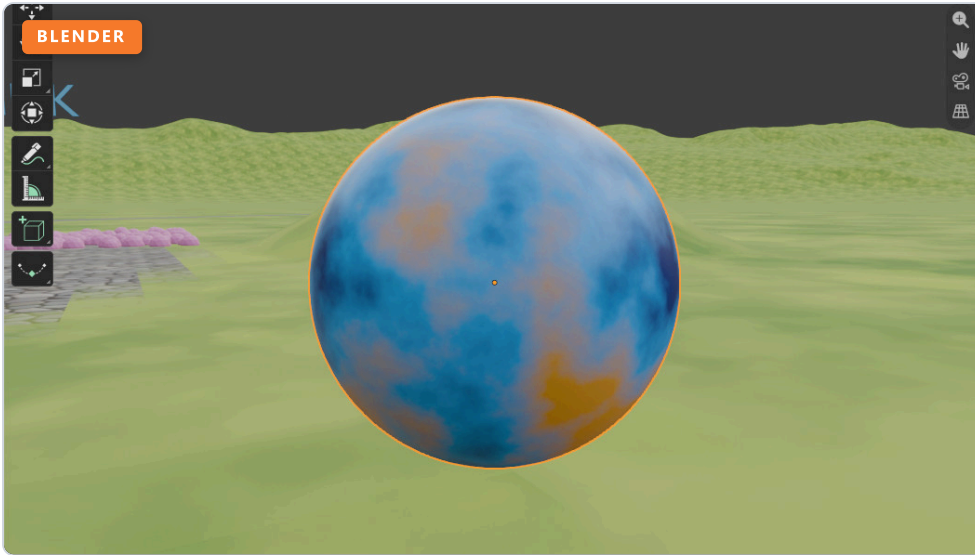
Mats tab: the crate's material is Unreal's own (*Unreal's own*), with a Scene Default / Exact / Fast PBR override.

WHERE THEY GO

Each Blender material becomes a material instance in `/Game/BlenderLink/Materials/<Material>/` with its textures. Reconnecting doesn't re-send materials Unreal already has.

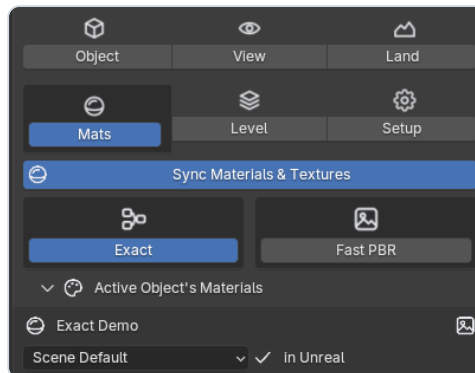
Exact Materials: Blender's node tree in Unreal

Unreal evaluates your nodes with the same code as Cycles, ported to HLSL: the same numbers - not an approximation, not a bake. Left: Blender's Material Preview; right: the same ball in Unreal under Lumen.



Use it

- 1 **Mats** tab → Material Mode **Exact** (the default for new scenes).
- 2 Build the material as usual - here a Noise → Color Ramp base colour and a Voronoi → Bump normal.
- 3 Per material, the dropdown next to it overrides the mode: Scene Default / Exact / Fast PBR.



The ball's material: Exact, **in Unreal**, with a procedural badge.

What's live

- **Values** - unlinked sockets, ramp stops and colours, Value / RGB nodes - become parameters: ~1 ms, no shader compile.
- **Painting** a texture re-imports just that texture.
- **Layout changes** (a node, a link, a dropdown) rebuild the shader: a few seconds of compiling.
- Unreal gets `.../Materials/<Name>/Graph/M_<Name>`, the instance `MI_<Name>` on your actors and `Textures/`.

Exact Materials: verified, node by node

One Custom-node shader per material, generated from the node tree and checked against Cycles in two test benches.



Verified 1:1 with Cycles

- **Image bench**, 325 node tests: each graph baked by Cycles vs rendered by Unreal - median worst-pixel difference 0.000003, largest 0.002.
- **Mesh bench**, 61 tests on a rotated, non-uniformly scaled sphere moved off its pivot: all within 0.002 - the same on Nanite meshes.
- **True displacement** on Nanite: within 0.6 mm of Cycles for a ± 6 cm displacement.

FOR BIT-EXACT RESULTS

Nanite meshes: Explicit Tangents on, Normal Precision 15, Tangent and Position Precision 12. Normal-mapped meshes: Full Precision UVs + High Precision Tangent Basis.

Translated nodes



Not reproducible (reported per material)

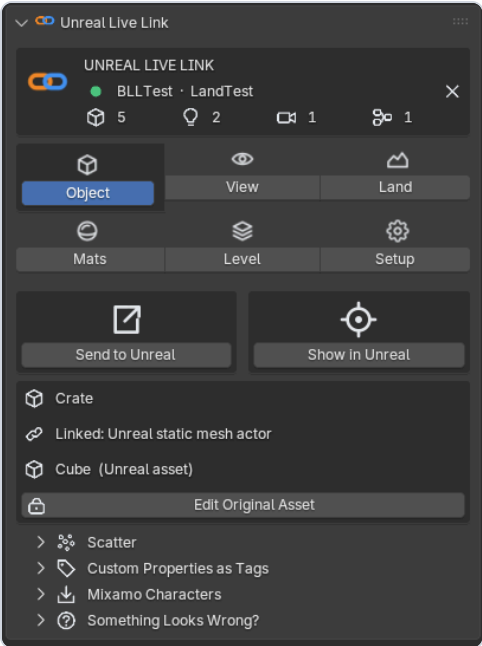


Shaders and geometry

- **Principled BSDF** → Unreal's Default Lit inputs: base colour, metallic, roughness, specular, normal, emission, alpha, coat → Clear Coat. Diffuse, Glossy, Metallic, Emission, Transparent, Mix and Add Shader map onto the same inputs.
- **Geometry the way Cycles sees it**: normals through the object's inverse transpose, MikkTSpace tangents for normal maps, Cycles' bump differentials, the object's pivot for Object Info.
- **True displacement** (Material → Settings → Displacement) becomes Nanite tessellation.
- Lighting is Unreal's own (Lumen): the *material* matches, the final image follows Unreal's lights.

Meshes and object data

Geometry, UVs, vertex colours, weight masks and tags - and a guard so Unreal's own assets are never damaged by accident.



Object tab for an Unreal mesh: its actor, its asset and **Edit Original Asset**.

Made in Blender

updates its Unreal asset **in place**: every actor using it changes.

From Unreal

editing it makes a **separate copy** `.../<Mesh>_Edit` - the original can't be damaged by accident.

Edit Original Asset

overwrites the real asset instead (asks first). Slots, Nanite and collision stay; a backup goes to `_Backups/`.

DATA	BLENDER → UNREAL	UNREAL → BLENDER
UV maps	all of them, in order (the one the materials sample first)	all channels (UVMap ... UVMap7)
Vertex colours	the render colour attribute	colour attribute <code>Color</code>
Weight painting	vertex groups you pick → R / G / B / A of the vertex colours	-
Custom properties	→ actor tags	actor tags → custom properties
Normals	custom split normals	per-corner normals

Weight-paint a material mask

- 1 **Mats** tab → **Vertex Colour Masks** → pick a vertex group for R (and G, B, A).
- 2 Weight-paint: that channel updates in Unreal live.
- 3 Read it with a **Vertex Color** node - the same value as Blender's Color Attribute (± 0.004).

Tags from custom properties

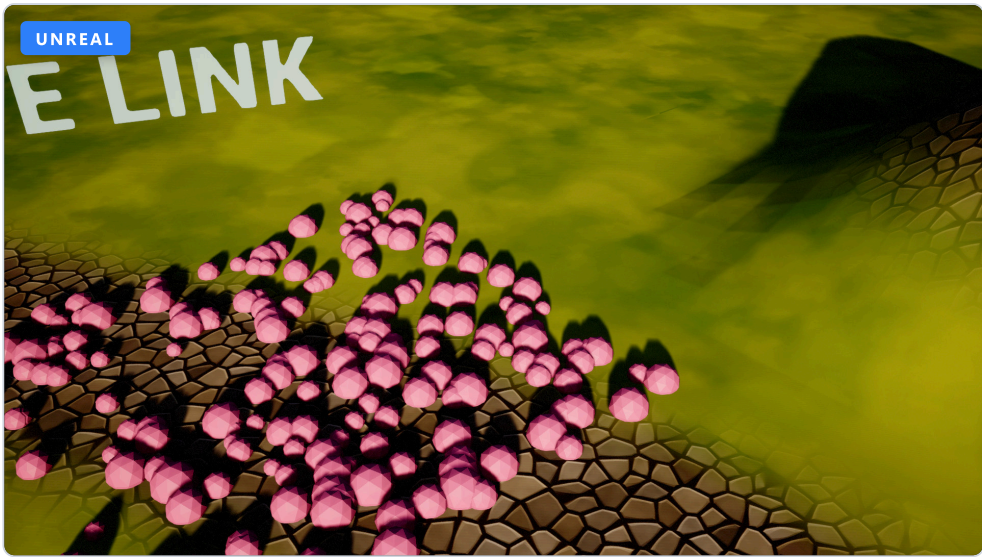
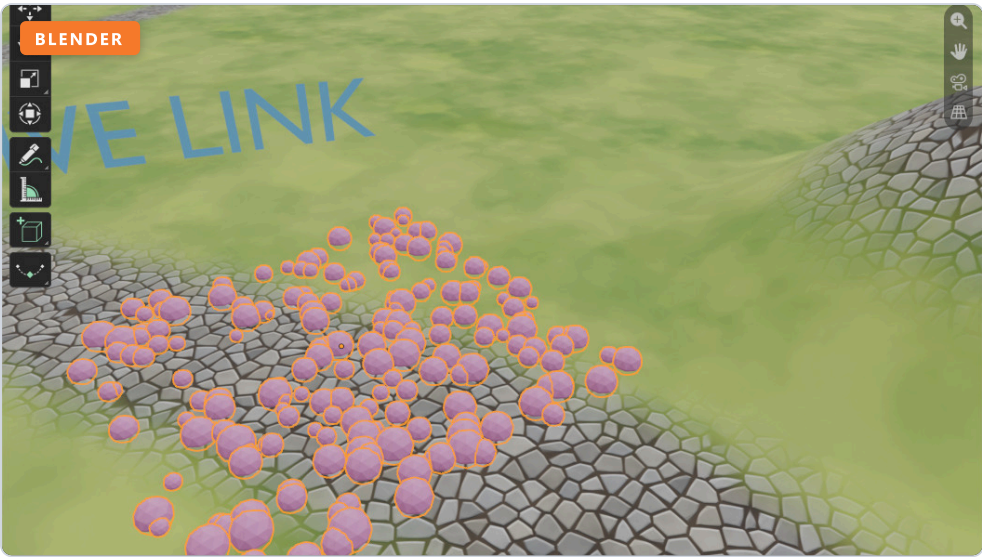
A number or text property becomes the actor tag `key=value` ; a True checkbox the tag `key` . Actor tags come back the same way (numbers as numbers). Edits on either side update the other.

KEPT FOR YOU

Your vertex groups survive when Unreal updates a mesh; meshes with your shape keys aren't refreshed from Unreal.

Geometry Nodes scatter → instanced meshes

Scatter in Blender; Unreal gets efficient instanced static meshes on the object's actor, not one huge merged mesh.



Scatter something

- 1 Add a plane (the ground) in **Unreal**: `<Level>` and a Geometry Nodes modifier.
- 2 **Distribute Points on Faces** → **Instance on Points** (an Ico Sphere, Object Info or Collection Info as the instance).
- 3 Drag density, seed or the scale range: Unreal follows within a fraction of a second.

What gets instanced

- A mesh **from Unreal** (one of your level's meshes) is instanced as that asset - never copied.
- A mesh **made in Blender** is exported once to `/Game/BlenderLink/<Mesh>/` and updated in place: every instance follows.
- Geometry made **inside the node tree** goes as `<Object>_geo0 , _geo1 ...`
- One Instanced Static Mesh component per kind of mesh, Nanite when the mesh is.

Fast, even when big

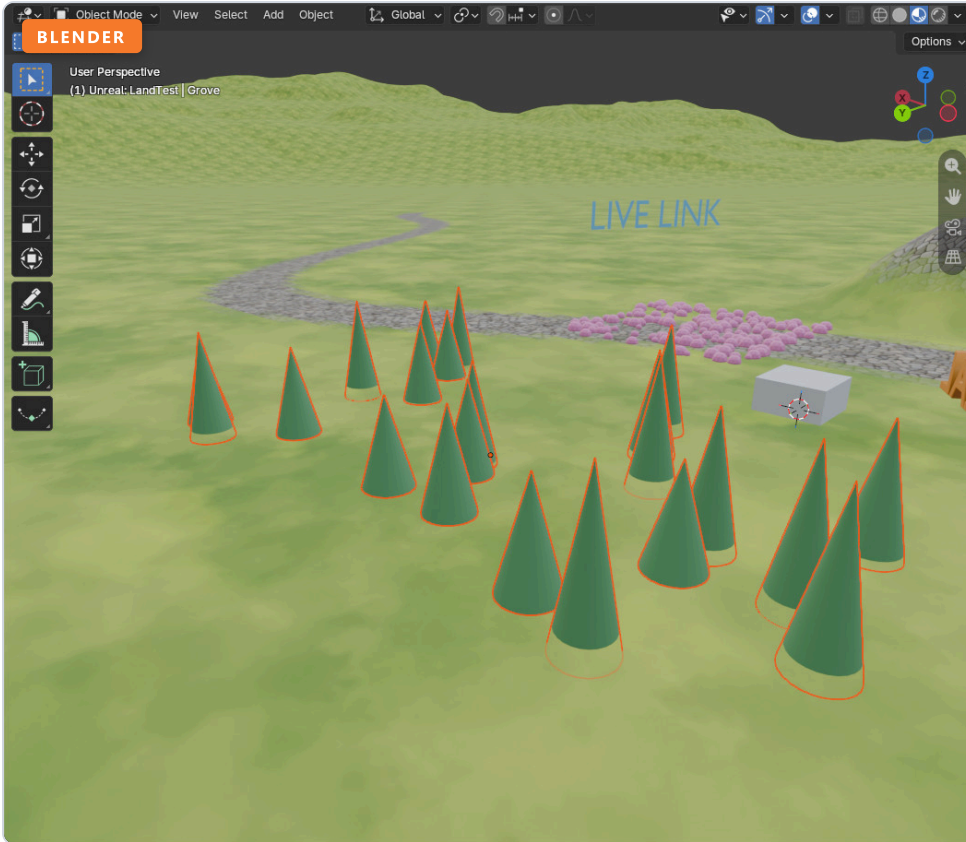
INSTANCES	BLENDER	UNREAL
~2,000	~12 ms	~50 ms
~19,000	~260 ms	~400 ms

GOOD TO KNOW

Up to 100,000 instances per object. The components are named `BlenderInst_<source>` ; edits made to them in Unreal are replaced by the next update.

Edit Unreal foliage in Blender

Foliage painted in Unreal and instanced components come over as points you can move, copy and delete.



A Hierarchical Instanced Static Mesh (the **Grove** cones) in Edit mode: every instance is one point drawn as the real mesh; a third of them selected (outlined), ready to move with **G**.

- 1 Click a foliage or ISM / HISM instancer - one object per component.
- 2 Press **Tab** for Edit mode. **G** moves instances, **Shift** + **D** adds copies, **X** deletes.
- 3 **Tab** out: the changes go to Unreal as **one undo step**; Unreal answers with the component as it is now and Blender rebuilds it from that.

FOLIAGE STAYS FOLIAGE

Edited instances are placed one by one through the Foliage Helper, so they keep (or get) their attachment to the landscape or mesh they stand on, and still follow Unreal's sculpting. Instances moved into another World Partition cell move to that cell's foliage actor.

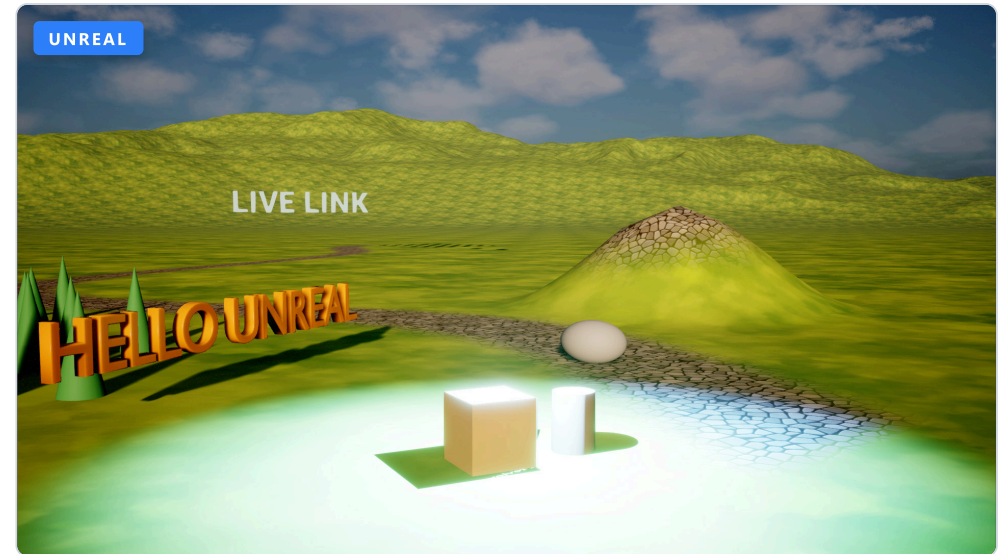
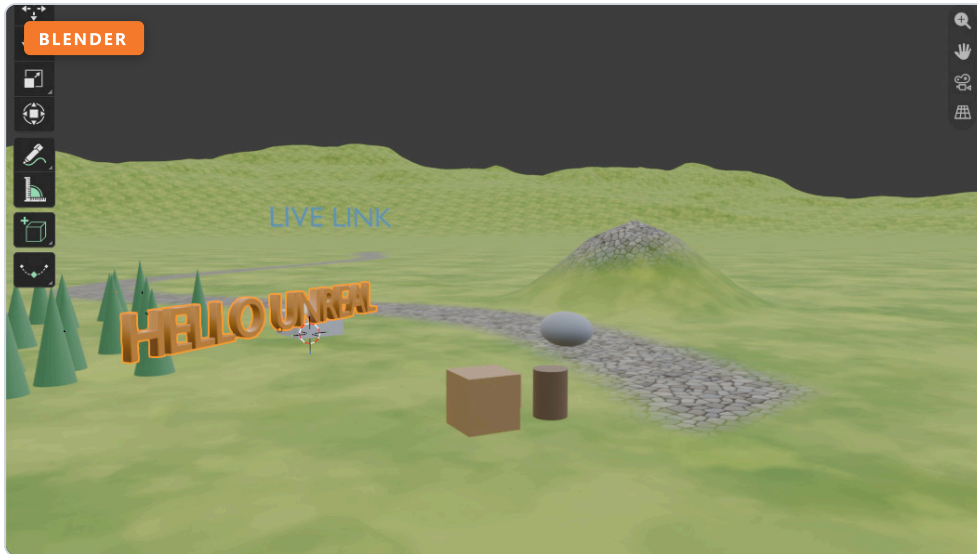
- ISM / HISM components of ordinary actors use Unreal's own instance functions - no helper needed.
- Each instance keeps its rotation and scale: moving points moves instances.
- If Unreal changed the same component while you edited it, your edit is refused - nothing is half-applied.

READ-ONLY CASES

Without the Foliage Helper, foliage is read-only (Blender says so). Components a Blueprint's construction script creates are rebuilt by the Blueprint: edit the Blueprint.

Curves and text

Type in Blender, see the 3D text in Unreal - live, typing included.



Curves, text, surfaces and metaballs BLENDER → UNREAL

- 1 **Shift** + **A** → **Text** (or any curve) in the level collection; give it some extrude or bevel.
- 2 **Tab** and type: the static mesh updates within half a second of each pause - bevel, extrude, offset and modifiers applied.
- 3 Assign materials as on any mesh. A curve without a surface yet waits until it has one.

Text Render actors BOTH WAYS

- Unreal's Text Render actors arrive as text objects: same text, size (world size = text size × 100), alignment, colour and facing.
- Edit the text, its size, alignment or the material's Base Color: the Text Render follows live; **Shift** + **D** makes a Text Render copy.
- Blender shows its own font (Unreal font assets don't carry over); Unreal's text stays flat.

Blueprint splines: fences, roads, rails

Edit a Blueprint's spline points with Blender's curve tools - Unreal rebuilds whatever the Blueprint builds along it.



Reshape a spline Blueprint

- 1 Click the Blueprint - its spline component is a Bezier curve part of the actor.
- 2 **Tab** into Edit mode: move points and handles, Extrude / Subdivide to add points, **X** to delete, toggle cyclic.
- 3 Unreal's spline follows live; the construction script reruns and the posts or road mesh rebuild along it - and come back to Blender.

What Unreal gets

- Exact points and tangents: straight (Vector) handles become linear points, moved handles custom tangents; untouched points keep Unreal's own type.
- The component is set to **Override Construction Script**, so the Blueprint keeps your points.

NEEDS

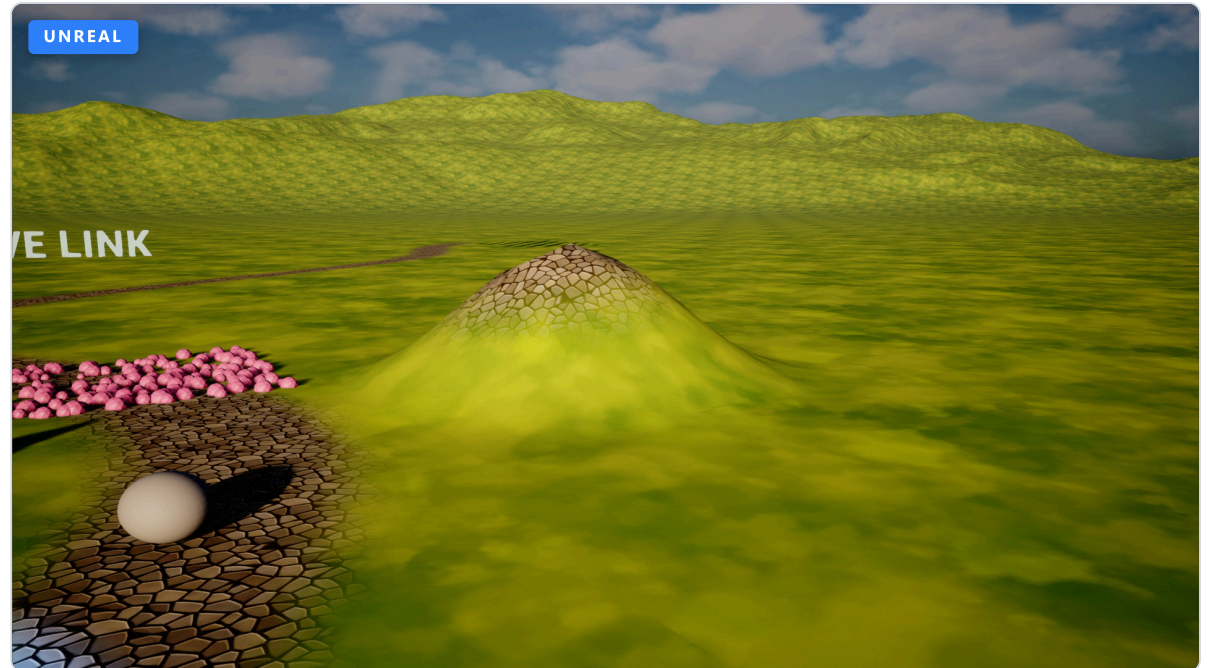
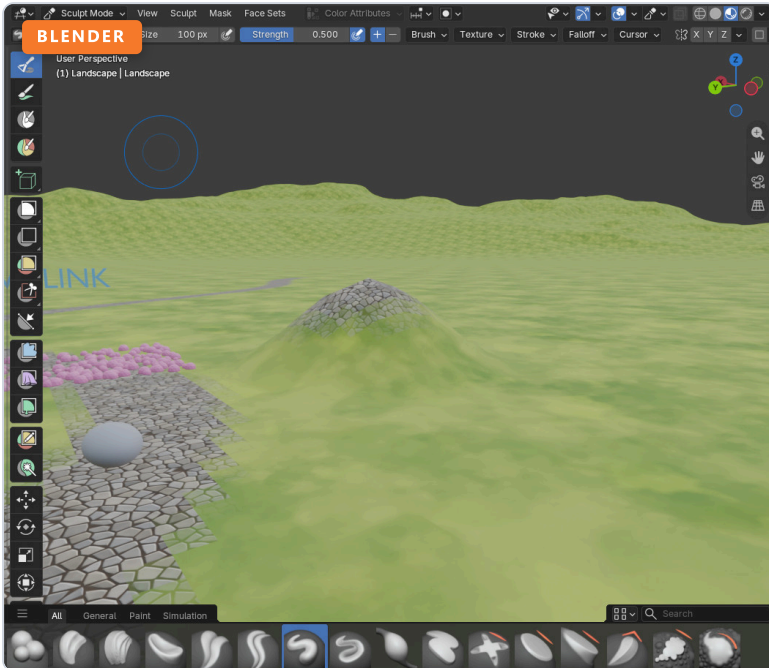
The Foliage Helper reruns the construction script (Python can't).

LIMITS

Splines a construction script makes itself are read-only. One spline component = one spline. The rest of a Blueprint moves as a whole.

Landscapes: exact heights, sculpted in Blender

A Landscape (World Partition proxies too) arrives as one terrain object with Unreal's exact heights. Left: a hill sculpted in Blender; right: the same moment in Unreal.



- 1 Select the terrain → **Sculpt mode** (Draw, Clay, Smooth, Flatten, Grab...).
 - 2 After each stroke the heights go to Unreal as **one undoable transaction**.
 - 3 Sculpted in Unreal instead? Switch to Blender: the terrain refreshes within ~3 s.
- Unreal's 16-bit heightmap arrives vertex by vertex. Over ~1.1 million vertices every 2nd (4th...) one is mirrored - your sculpt still lands exactly.
 - **Edit layers:** the sculpt goes into the base layer as base + (sculpt – result): the final landscape is exactly your sculpt.
 - Only height goes back; adding or deleting vertices disconnects the edit.

FOLIAGE FOLLOWS?

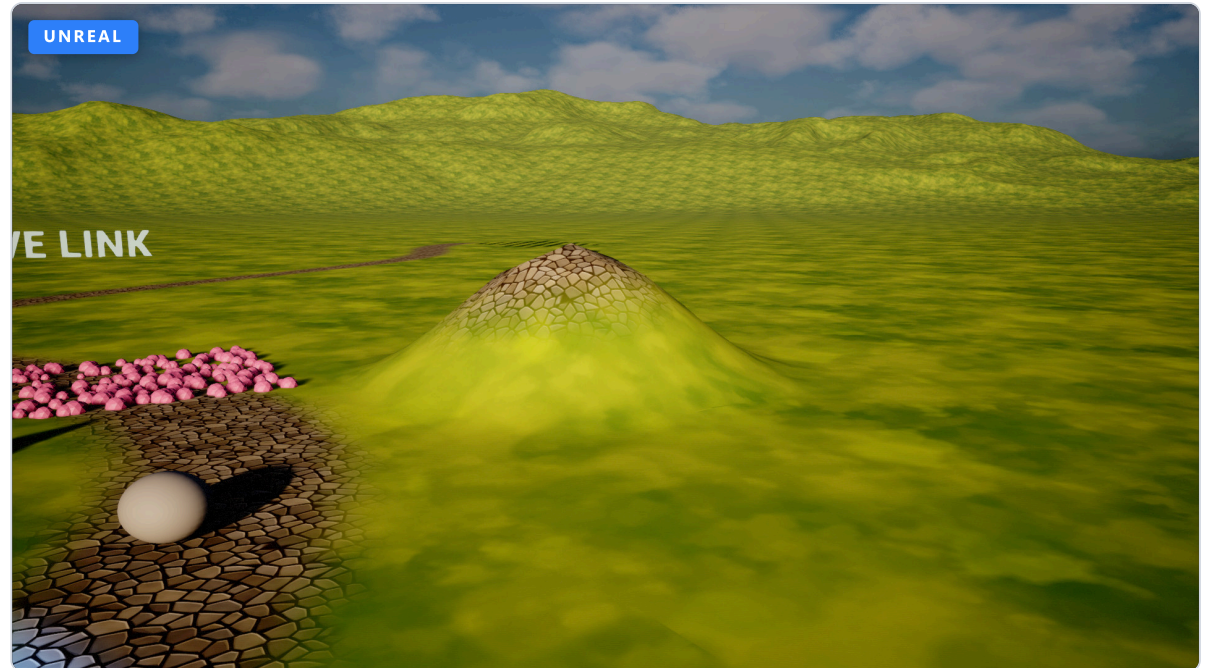
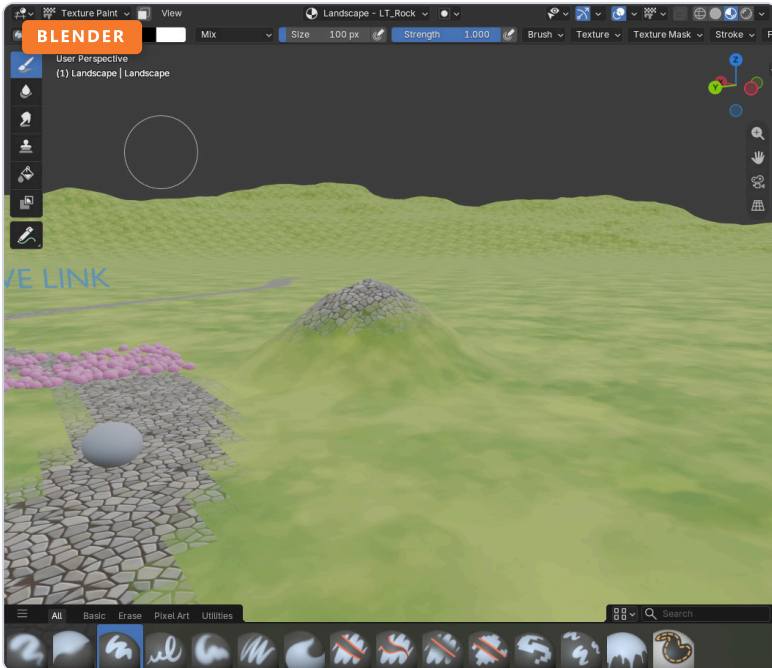
Set **Foliage Landscape Snapping Mode** (Project Settings → Engine → Landscape) to *Snap Under Brush Only*; UE 5.8's default is Don't Snap.

LANDSCAPE GRASS - A READ-ONLY MIRROR

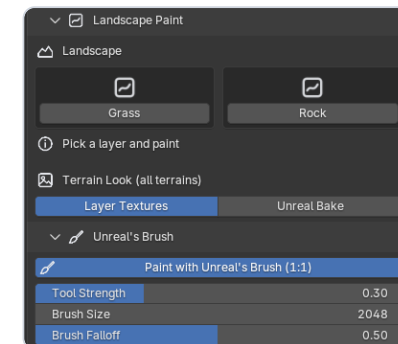
Grass types in the material's **Grass Output** arrive as one instancer per variety, at Unreal's exact transforms, and regrow as you sculpt or paint. **Grass Limit** (Land tab) and **Camera Radius** keep big landscapes light (needs the Foliage Helper 1.8).

Paint the landscape's layers

Every paint layer is an editable greyscale image with Unreal's exact weights, and the terrain shows the real layer textures. Left: the rock layer painted onto the hill in Blender; right: Unreal.



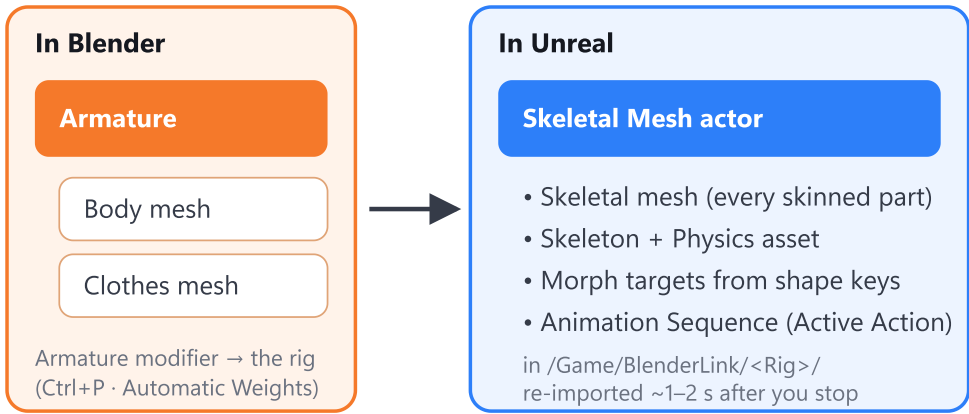
- 1 **Land** tab → **Landscape Paint** → click a layer's card. Blender selects the terrain and enters Texture Paint on that layer's image.
 - 2 Paint. With **Unreal's Brush** (the default) a stroke lands exactly as the same stroke made in Unreal: **Shift** erases, **Ctrl** paints without taking from the other layers. Every stroke shows at once.
 - 3 Each stroke goes to Unreal when it ends, as one undo step. **Done Painting** leaves.
- **Layer Textures** (default): each layer's real texture with its tiling - or its plain colour - mixed like Unreal: weight-blended layers averaged, the others laid over in order.
 - **Unreal Bake**: Unreal's rendered colour (height blends, tints, glow).
 - Weight-blended layers stay at 100% together, like Unreal's paint tool.
 - **Blender's brushes** (Unreal's Brush off): white adds the layer, black removes it; sent when you pause.



Land tab: a card per layer, the look and Unreal's Brush. New layers are made in Unreal.

Rigs → skeletal meshes

An armature with its skinned meshes becomes one Skeletal Mesh actor - poses and actions play in the Unreal editor.



Make a rig live

- 1 Put an armature in **Unreal: <Level>** with its meshes skinned to it (**Ctrl** + **P** → With Automatic Weights).
- 2 Together they become **one Skeletal Mesh actor**. The armature object is the actor: move it and the actor follows; delete it and the actor goes.
- 3 Edit the mesh, weights, bones (rest pose), shape keys or materials: the rig is re-imported in place. A bone drag sends once, when you let go.

Pose and animation

Object tab → **Rig (Skeletal Mesh)** → **Plays in Unreal:**

MODE	WHAT UNREAL SHOWS
Action or Pose	the active action if the armature has one, else the current pose (default)
Active Action	the action as an Animation Sequence that plays, looping, right in the editor viewport - edit keys and it updates
Current Pose	pose bones in Blender; the actor takes the same pose
Rest Pose	no animation

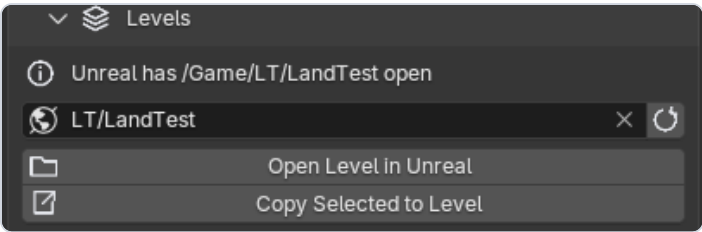
- Shape keys arrive as morph targets; materials sync like any other.
- Rigs always travel as GLB (glTF skin), whatever the exchange format.
- Skinned meshes inside the link collection are parts of the rig, not separate static meshes.

ONE DIRECTION

Unreal's own skeletal meshes aren't mirrored into Blender - rigs go from Blender to Unreal.

Levels: browse, open, send

Every level of the project, one click away - and objects from any Blender scene can go into any of them.



Level tab → Levels: search the project's levels, open one, or copy the selection into it.

- 1 **Level** tab → **Levels** lists every level (map) of the project - type to search.
- 2 **Open Level in Unreal** switches Unreal to that level; Blender follows with its own **Unreal: <Level>** collection. The others are hidden and kept; switching back syncs like a reconnect.
- 3 **Copy Selected to Level** copies the selected objects - from any Blender scene - into the chosen level as new actors, opening it first. The originals stay where they are.

UNSAVED WORK IS SAFE

If the level open in Unreal has unsaved changes, Blender asks first: **Save & Open, Open Without Saving** or cancel - Unreal's scripted level switch would drop them without asking.

What travels to another level

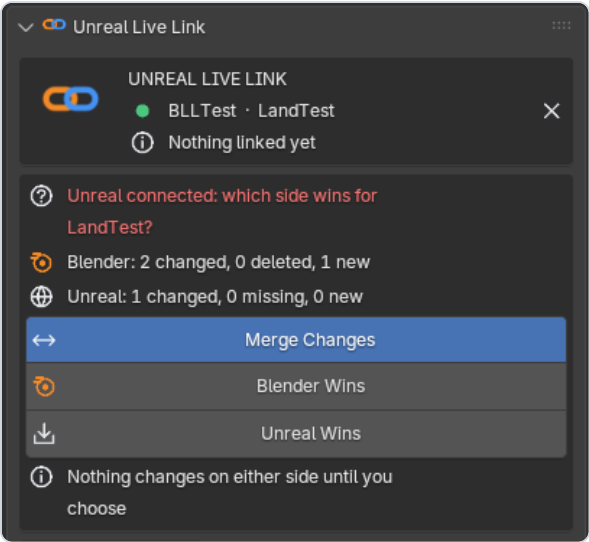
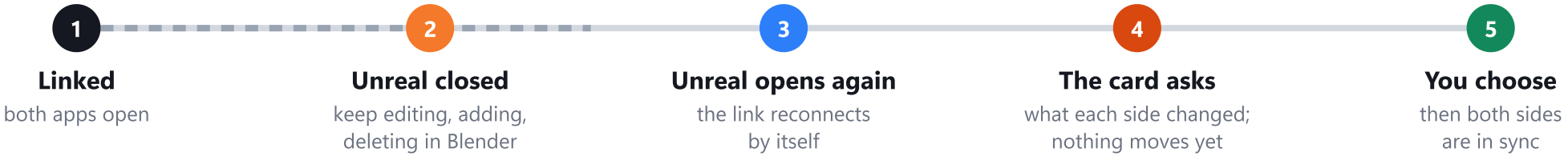
OBJECT	IN THE OTHER LEVEL
Meshes	the same Unreal asset, placed again
Curves, text, lights, cameras	their own copy of their data
Material slots, custom properties	kept (custom properties as actor tags)
A Blueprint's mesh part	a plain mesh actor
Blueprints, rigs, foliage / grass, terrain	can't be copied from Blender

UNREAL OPENED ANOTHER LEVEL?

When Unreal starts on its startup map instead of yours, Blender doesn't switch: a card asks first (page 29).

Work while Unreal is closed

Close Unreal whenever you like and keep working in Blender. When the project opens again the link reconnects by itself - and asks which side wins before it changes anything.



The card after a reconnect: Blender moved two objects and added one; Unreal moved one.

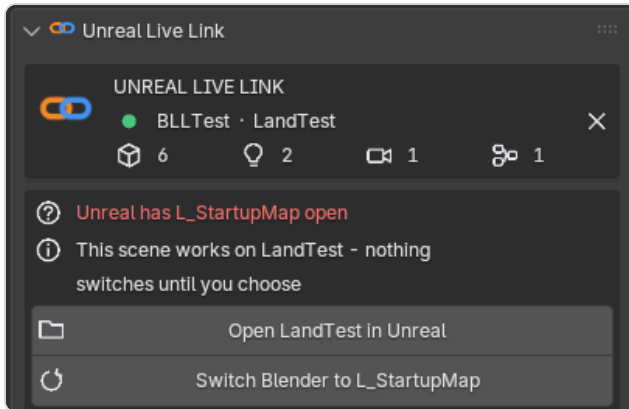
NOTHING HAPPENS UNTIL YOU CHOOSE
No object is built, sent or removed on either side; the top-bar **Unreal** button stays red. Your last answer is shown pressed next time.

CHOICE	WHAT HAPPENS
Merge Changes	Per object: what you changed in Blender goes to Unreal - position, light / camera / text settings, name, folder, custom properties, mesh, materials; objects you added are created, objects you deleted are deleted (more than 25 wait for <i>Delete in Unreal / Restore</i>). Everything you didn't touch follows Unreal, including what changed there. Changed on both sides: Blender's version wins.
Blender Wins	Every linked object goes to Unreal as it is in Blender; objects deleted here are deleted there, objects missing there are made again.
Unreal Wins	Blender's copy of the level is rebuilt from Unreal: changes to linked objects are undone, deleted objects come back. New objects you made here are kept and sent.

WHICH ONE?
Not sure? **Merge Changes** - it keeps the work of both sides. Every connect asks (a reconnect too) whenever the .blend already has that level; a first connect has nothing to lose and isn't asked.

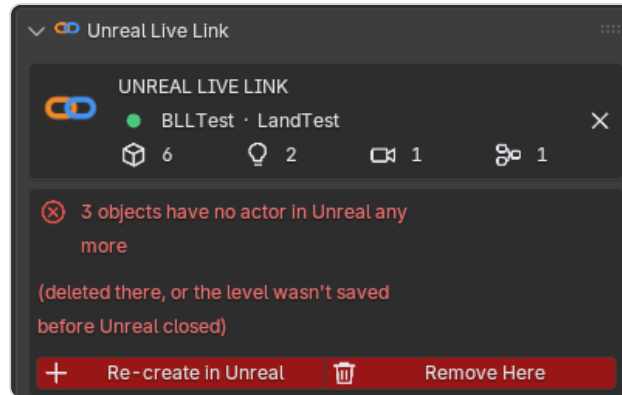
Offline work: the other cards

Two more situations the link asks about instead of guessing.



Unreal starts on another level

When Unreal opens its startup map instead of your level, Blender **doesn't switch** - so nothing unsaved can be lost. Choose **Open <your level> in Unreal** (Unreal asks first if its open level has unsaved changes) or **Switch Blender** to Unreal's level (your level's collection is hidden and kept).



Actors that are gone

Objects whose actor no longer exists - deleted in Unreal, or the level closed with *Don't Save*: an object you changed meanwhile is re-created automatically; untouched ones wait on this card with **Re-create in Unreal** and **Remove Here**. Nothing you made is silently deleted from Blender.

Deleted, but not loaded yet

Deleted something whose actor isn't loaded or in the link's scope when Unreal returns - an unloaded World Partition region, outside the **Camera Radius**, a skipped folder? The deletion is kept in the .blend and carried out when the actor comes in. It never comes back by itself (*Unreal Wins* drops it).

Keep the sync state

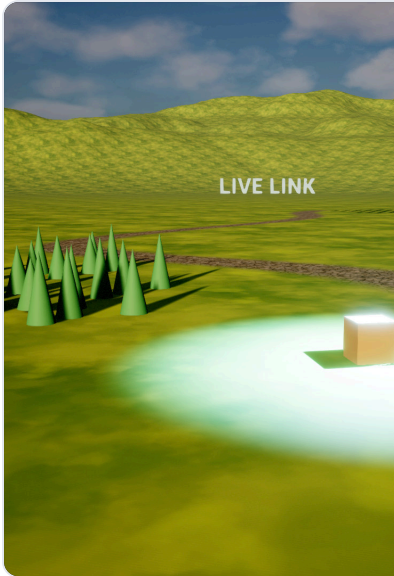
What each object was when last in sync is stored on it in the .blend (on disconnect and when you save while linked) - so closing Blender too is fine: **save the .blend first**.

ALWAYS AVAILABLE

Resync from Unreal re-reads the whole level and always takes Unreal's version.

Use cases

Eight ways teams use the link - each with a three-step recipe.



1 Blockout & set dressing

Rough out a level with Blender's modelling tools on the real level, then dress it with the project's own assets - in place, at exact scale.

1. Connect; **Shift** + **A** boxes and shapes - each is an actor at once.
2. Duplicate Unreal meshes from the level: real Unreal duplicates, overrides kept.
3. Snap, align and array with Blender's tools; sort into collections = folders.

Live mirror · duplicates · folders

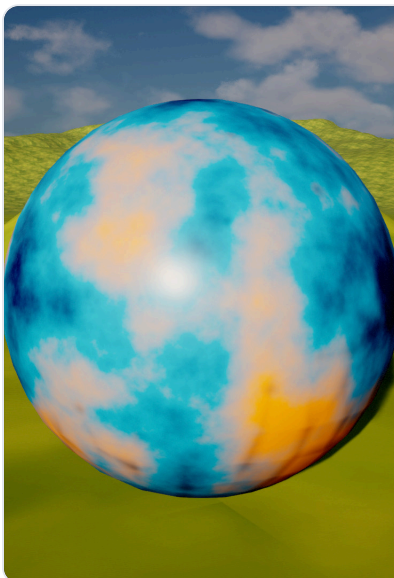


2 Terrain art pass

Shape landforms with Blender's sculpt brushes, then paint grass, dirt and rock with the real layer textures showing.

1. Sculpt mode on the terrain - each stroke lands in Unreal.
2. Land tab → a layer's card → paint with Unreal's brush.
3. Unreal regrows its grass; set foliage to Snap Under Brush Only.

Landscapes · paint layers · grass

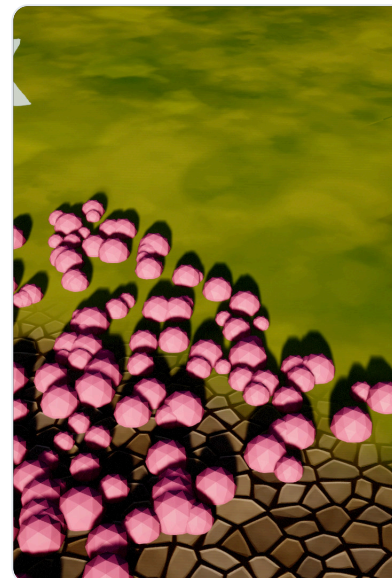


3 Look-dev under Lumen

Build materials in Blender's node editor and judge them under Unreal's lighting, instantly.

1. Material Mode: Exact.
2. View Sync → Drive Unreal, Unreal on a second monitor.
3. Drag values (~1 ms) and paint textures (~8 ms) while watching Unreal.

Exact Materials · View Sync · texture paint



4 Procedural vegetation & props

Scatter rocks, flowers or debris with Geometry Nodes; Unreal gets lean instanced components.

1. Geometry Nodes on a ground plane: Distribute Points → Instance on Points.
2. Instance meshes from the level - they stay Unreal's assets.
3. Tune density, seed and scale live; Nanite when the mesh is.

Scatter · ISM · Nanite

Use cases, continued

Splines, text, rigs - and working without the project open.



5 Roads, fences and rails

Reshape spline Blueprints with Blender's curve tools and watch them rebuild.

1. Click the Blueprint; **Tab** into its spline.
2. Move points and handles, extrude new ones.
3. The construction script reruns: posts and road meshes follow.

Blueprint splines · Foliage Helper

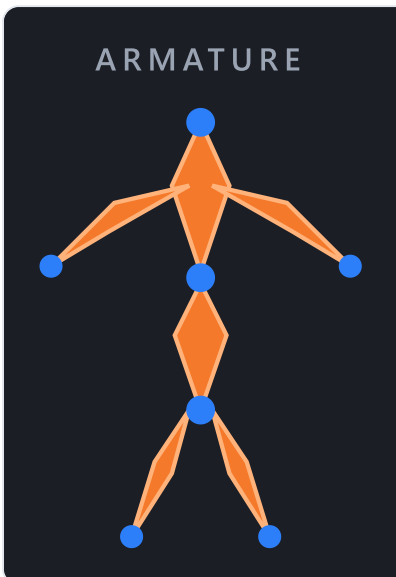


6 Signage, logos and titles

3D text and logo shapes from Blender curves, and Text Render actors edited as text.

1. **Shift** + **A** → Text; extrude and bevel.
2. **Tab** and type - the mesh follows as you pause.
3. Edit Unreal's Text Renders from Blender, colour included.

Curves & text

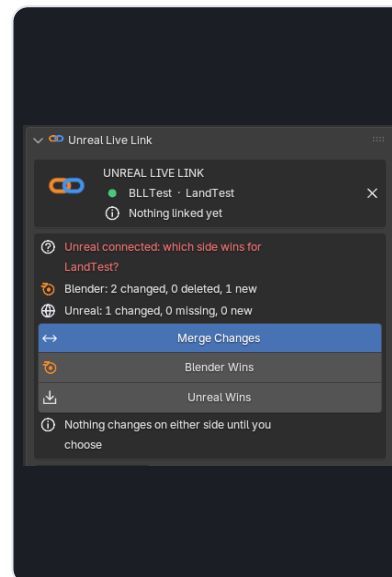


7 Character and prop previs

Rig and animate in Blender, see the skeletal mesh and its action loop inside the level.

1. Armature + meshes, **Ctrl** + **P** → With Automatic Weights.
2. Rig → Plays in Unreal: Active Action.
3. Edit keys or weights: the actor updates in the Unreal viewport.

Rigs · morph targets · animation



8 Away from the project

Take the .blend to work on a laptop or while Unreal is closed - merge when you're back.

1. Save the .blend while linked.
2. Close Unreal; keep editing, adding, deleting.
3. Reopen the project → card → Merge Changes.

Offline work · choice cards

Best practices

Habits that keep sessions fast and your work safe.



Do

Keep both parts on the same version

The add-on and the plugin belong together - update them together.

Save in Unreal as usual

Moves from Blender mark the level dirty; assets made from Blender save themselves.

Save the .blend while linked

The next connect is faster and the sync state is kept for offline work.

Scope big levels

Level tab: Camera Radius around Unreal's camera, Only / Skip Folders. Land tab: Grass Limit.

Give meshes UVs

Texture painting, bakes and Unreal's materials read the first UV channel.

Install the Foliage Helper

Foliage editing, landscape grass and spline rebuilds need it.

Choose Merge Changes when unsure

It keeps the work of both sides; Resync from Unreal is always there.

Set foliage snapping before sculpting

Snap Under Brush Only keeps trees on the ground you reshape.

Use Exact Materials for look-dev

The same numbers as Cycles; values update without shader compiles.



Avoid

Edit Blueprint internals in Blender

Move Blueprints as a whole; only their splines are editable.

Hand-edit grass instancers

They are read-only: paint the grass layer or sculpt, Unreal regrows it.

Edit Blender scatter components in Unreal

BlenderInst_ components are replaced by the next update.

Rotate or scale single foliage instances

Moving points moves instances; rotation and scale stay per instance.

Expect Unreal skeletal meshes in Blender

Rigs go from Blender to Unreal only.

Rely on automatic cleanup

The link deletes no assets: unused material copies are listed for you to delete.

Edit while Unreal plays (PIE)

The link pauses; Blender changes apply when PIE ends.

Link one scene to two editors

One Blender scene links to one Unreal editor at a time; a second editor uses another port.

Expect Unreal edits to become curves

Curves and text go to Unreal as meshes: edit the curve in Blender.

Safety rules & performance

Guards on both sides, and numbers that keep both editors responsive.

What the link will and won't do

- **Unreal decides what exists.** On connect objects re-link to their actors by a stored id; anything missing is rebuilt; transforms come from Unreal.
- **Nothing is pushed until the level is fully loaded** in Blender, and Unreal refuses adds and deletes from a client that hasn't confirmed it's in sync.
- **Mass-delete guard:** more than 25 objects deleted at once wait for *Delete in Unreal* or *Restore*. **Mass-push guard:** more than 25 objects losing their actor at once are not re-sent.
- **Undo is per app:** what came from Unreal is one undo step in Blender, each change from Blender one in Unreal ("Blender Link: ..."); undo never deletes Unreal's own actors.
- **One Blender per editor;** the link never runs inside Unreal's own saves, progress dialogs or undo.
- **No assets are deleted** by the link in this version; unused material copies are only listed.
- **Play In Editor** pauses the link. **Opening another .blend** disconnects first. **Switching levels** makes a new collection; the old one is hidden and kept.
- **Offline edits** wait for your choice on reconnect - nothing changes on either side until then.

Measured on a 270-actor level

	TYPICAL	WORST
Unreal editor, per frame	~1.1 ms	4.9 ms
Blender, per tick (50 Hz)	0.07 ms	5 ms
Blender structure scan (every 0.4 s)	3.5 ms	
Material value → Unreal	~1 ms	
Painted texture → Unreal	~8 ms	
Scatter, 2,000 instances (Blender / Unreal)	~12 / ~50 ms	
Scatter, 19,000 instances (Blender / Unreal)	~260 / ~400 ms	
Displacement Midlevel / Scale → Unreal	~1 ms	
Grass update, 120,000 instances (Unreal / Blender)	~25 / ~75 ms	

Unreal is checked in small time-boxed slices; Blender only re-reads what its dependency graph reports as changed. Heavy work runs in the background or while you are in the other app.

Troubleshooting

Most messages tell you what to do; here are the common ones.

PROBLEM	FIX
“Waiting for Unreal on 127.0.0.1:9920”	Unreal isn't open, or the server is off: Tools → VenomBridgeLink → Start . A second editor uses 9921+: set that port in Setup → Connection.
Objects are grey boxes	Their meshes are still loading (the card shows the count). Textures come while you aren't working in Unreal. Boxes left over from a failed load ask again on the next connect.
A procedural material arrives flat	Use Exact Materials, or turn on Mats tab → Procedural Bake and give the mesh UVs.
“Another Blender is linked to this Unreal editor”	Another Blender window has the link: press Connect here to take it over (the other one then waits), or disconnect there.
“Unreal is busy: ...” on the status card	Unreal is importing, exporting or building; its editor is blocked meanwhile. The link continues when it is done.
Red “lost their Unreal actor” error	The mass-push guard held objects back: click Resync from Unreal .
Changes stop while Unreal plays	Expected: PIE pauses the link. They apply when you stop playing.
The top-bar Unreal button is red	A card waits for you (which side wins, another level, missing actors) or a problem is listed - open the panel.
“Unreal plugin not activated on this PC”	Setup → License → Enter License Key... (or Start Free 1-Day Trial , once per PC). A friend key must be made for the machine code shown there.
“reached the activation limit” / “could not be reached for 30 days”	Deactivate a PC you no longer use / connect that PC to the internet once.
“foliage edits need the ... Foliage Helper”	Copy <code>VenomBridgeLinkFoliage</code> into <code><YourProject>/Plugins/</code> and restart the editor once (1.8 or later for grass).
No grass in Blender	Unreal grows grass near its camera only: look at the landscape in Unreal once. Check Land tab → Landscape Grass (Mirror Landscape Grass, Grass Limit) and the Camera Radius.
Trees float or sink after sculpting	Set Foliage Landscape Snapping Mode to Snap Under Brush Only (Project Settings → Engine → Landscape).
“Unreal could not take the sculpt exactly”	The landscape's height limits (0-65535) or a locked layer cut it off; Blender shows what Unreal made of it.
Deleted while Unreal was closed, still in Unreal	Its actor wasn't loaded or in scope when Unreal came back (World Partition region, Camera Radius, skipped folder): the deletion waits in the .blend and is carried out when the actor loads.
Asked which side wins on every connect	By design: every connect to a level this .blend already has asks first, so no work is lost. Your last answer is pre-selected.

Limits & compatibility

What the link leaves alone, and where it runs.

Not mirrored (yet)

- Landscape splines and particles (heights, paint layers, grass and spline components are).
- Landscape grass is read-only and only what Unreal grows near its camera.
- Blueprint actors are read-only apart from their spline components; foliage / ISM instances can be moved, added and deleted, not rotated or scaled one by one.
- Unreal materials appear in Blender as a baked approximation (base colour, roughness / metal, normal, emissive); Exact Materials go the other way.
- Unreal's own skeletal meshes don't come into Blender.
- Mesh data Unreal static meshes can't hold: shape keys, vertex groups themselves, more than one colour attribute, creases / bevel weights.
- One Blender scene links to one Unreal editor at a time.

Compatibility

Tested with	Unreal Engine 5.8 · Blender 5.2 · Windows 11
Plugin core	compiled for Windows; Mac / Linux need their own build
Foliage Helper	built for UE 5.8 Win64; rebuild from Source for another engine version
This guide	version 1.11.0 - protocol Unreal 15 / Blender 7

Two parts, one tool

The Blender add-on **VenomBridgeLink** is free (GPL-3.0-or-later). The Unreal plugin **VenomBridgeLink** is the paid part: a licence key, activated once per PC - see page 9.

New in 1.11.0

- **Six workflow tabs:** Object, View, Land, Mats, Level, Setup - each a picture over its name (page 11).
- **Unreal's Brush:** paint landscape layers with Unreal's own brush - a stroke lands 1:1 as in Unreal.
- **Own materials on shared meshes:** objects sharing one mesh keep their own material in Unreal (per-actor overrides).
- **Fixes:** undoing a delete keeps the material; no duplicate materials after an Unreal restart; the last edits made during an upload reach Unreal; and more (see the changelog).

UNREAL ENGINE 5.8 · BLENDER 5.2

Build in Blender. See it in Unreal.

Open your level, click Connect, and work in whichever app is better for the job. The link keeps both in step - live, exact and safe.

Install: page 8

Quick start: page 10

Troubleshooting: page 34

