VConnect Quickstart

A short guide to get VConnect running with your VTuber setup, import assets, and create your first trigger.

If you have used **T.I.T.S.** before, you will likely adapt quickly: same general idea (overlay + items + event triggers), with a more flexible node-based system and a Setup Wizard for common setups.

Table of contents

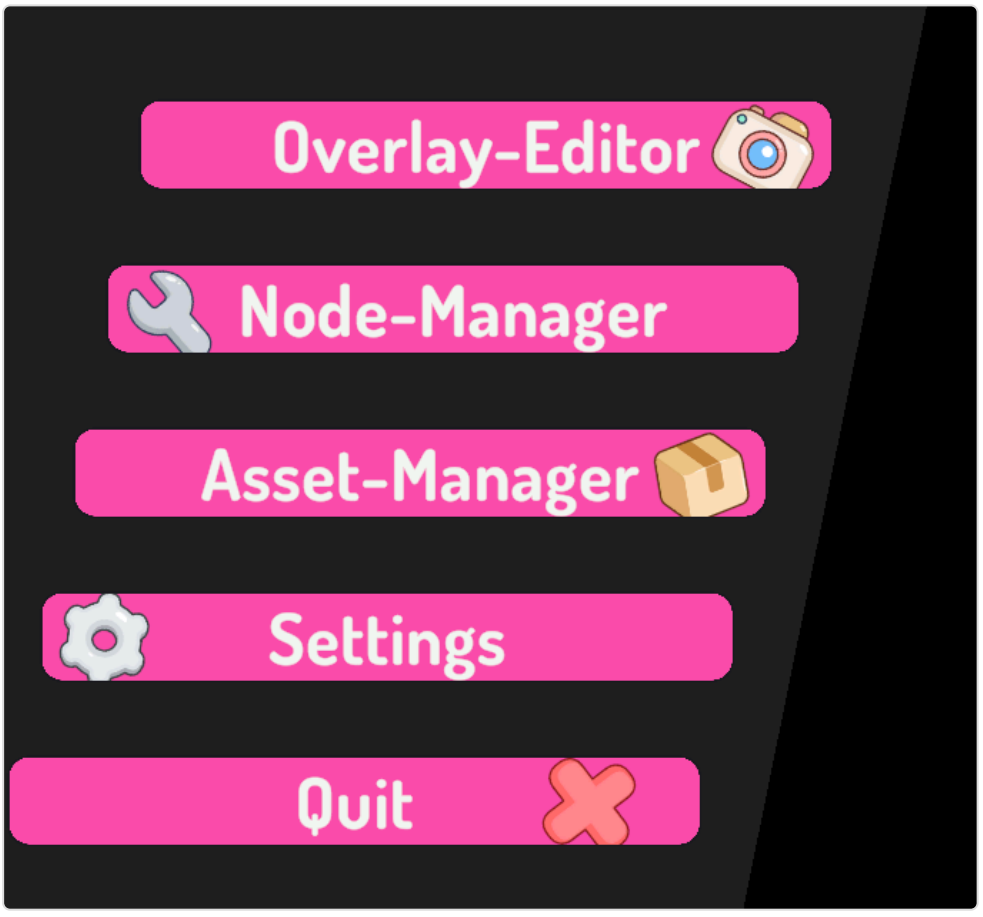
1. What is VConnect?
2. Limitations
3. Recommended setup: VTS / VSF + Spout2
    - Steps
    - OBS: add the Spout2 source
    - Linux: Spout2 through PipeWire
    - Optional: split output “sandwich”
    - Alternate path: VTS full-screen in OBS
4. Settings
5. Overlay Editor
6. Asset Manager
7. Importing from T.I.T.S.
8. Node Manager
9. Your first working effect
10. Cheat sheet
11. Advanced: Node Graph Concepts
    - Coordinate System
    - Conversion nodes
    - Node Groups
    - Variables

What is VConnect?

VConnect is a VTubing overlay and automation tool. You connect your avatar (typically via **VTube Studio** or **VSeeFace**), bring stream events into the app (Twitch, YouTube, donations, OBS, and more), and build **triggers** that spawn stickers, throw items, drive VTS effects, play audio, and so on.

The app has four main screens, reached from the side menu shown below:

Screen	Role
Overlay Editor	Align the overlay with your avatar; manage scenes, targets (pins), and hitbox/model settings
Node Manager	The heart of VConnect — event-driven node graphs (triggers) and the Setup Wizard
Asset Manager	Import, organize, edit, and export images, 3D models, video, and audio
Settings	Connect services (Twitch, VTS, OBS, Spout/NDI, ...) and app preferences

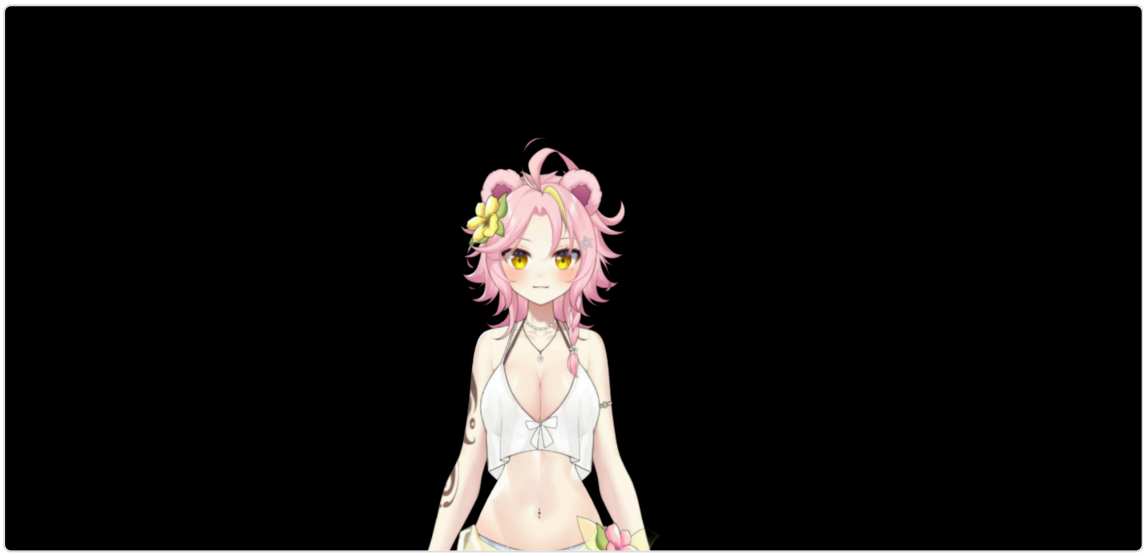


Top to bottom: Overlay Editor, Node Manager, Asset Manager, Settings — each covered in its own section below.

Overlay Mode (F3)

Press **F3** anywhere in VConnect to enter **Overlay Mode**: the editor UI is hidden and only the overlay is shown. Press **F3** again to return to the full UI.

- Use Overlay Mode when you **Game Capture** or **Window Capture** VConnect in OBS, so the editor chrome is not visible on stream.
- If you use **Spout2** or **NDI** senders, the sent image is already the overlay content (not the editor UI), so F3 is optional for that path.



Limitations

VConnect is in early access. A few platform limitations are known and being worked on — noting them up front so they don't feel like bugs:

Limitation	Details	Workaround
Spout2 on Linux needs extra setup	Spout2 itself is Windows-only. On Linux the Spout2 card uses **PipeWire** instead, and VTube Studio (running in Proton) and OBS each need a small add-on to talk to it.	Follow Linux: Spout2 through PipeWire.
NDI on Linux is laggy	NDI works on Linux, but with noticeable latency.	Use Spout2 through PipeWire instead, or capture the VConnect window in OBS (**Game Capture** / **Window Capture**) and press **F3** to hide the editor UI.
VP9 webm is not supported on Linux	Unity's Linux VideoPlayer backend only decodes **VP8**. Importing a VP9-encoded `.webm` will fail.	Re-encode/export video as **VP8** `.webm` before importing on Linux.
Drag-and-drop does not work on Linux	Dragging files (assets, packs) into VConnect from the OS file manager has no effect.	Use the in-app **Import** button in the Asset Manager instead.

Some of these can be worked around by running VConnect under **Proton** on Linux, but results vary a lot by setup, so this guide doesn't include a Proton walkthrough.

Recommended setup: VTS / VSF + Spout2

This is the default path for most users on Windows.

Intended use: Put **VConnect** in OBS as your avatar / effects source — **instead of** VTube Studio. VConnect receives the avatar (via Spout/NDI), runs effects there, and you capture **one** VConnect source in OBS. That single-source setup is the best usecase.

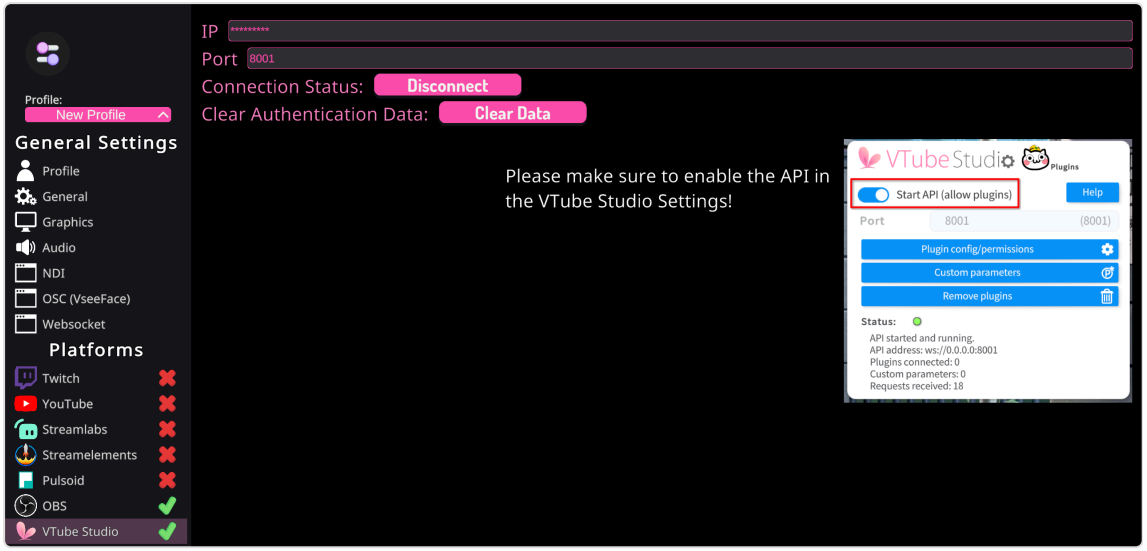
You can still keep VTube Studio in OBS temporarily to check alignment, but for streaming you normally leave VTS out of the final scene and only show VConnect. If you **really** need VTube Studio visible in OBS alongside effects (for example layering behind and in front of the avatar), see Optional: split output “sandwich” — that is a workaround, not the intended workflow.

Idea: Bring your avatar into VConnect with Spout2, run effects inside VConnect, then capture **VConnect** in OBS (instead of capturing VTS for the final effects layer).

```
VTS or VSF  --Spout2-->  VConnect  --Spout2-->  OBS
```

Steps

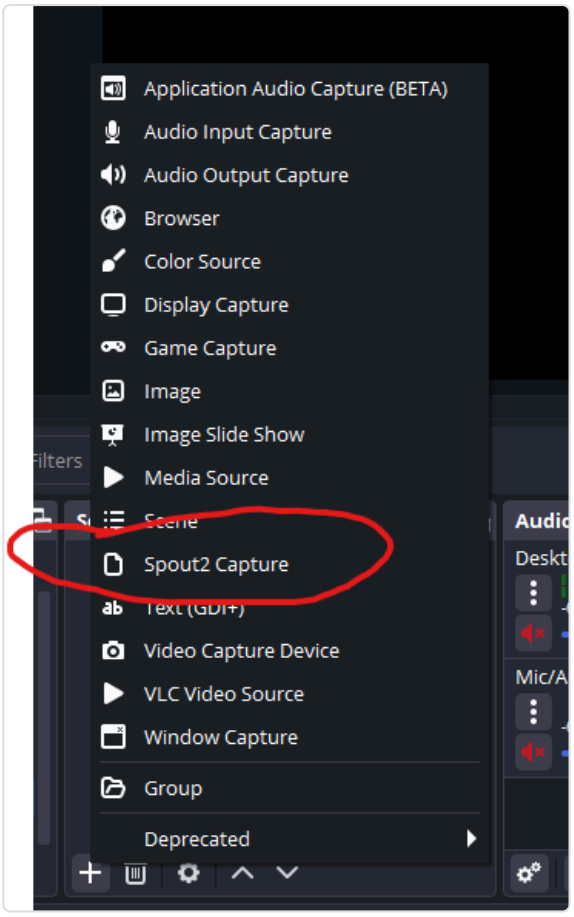
1. Launch **VTube Studio** or **VSeeFace** and enable its Spout / Spout2 output (name depends on that app’s settings). On **Linux**, launch it through spout2pw first — see Linux: Spout2 through PipeWire.
    - In VTS, set background to **black** and enable **Transparency** — best results for clean alpha when VConnect receives the feed.
2. Open VConnect → **Settings**.
3. Connect **VTube Studio** (IP + port; use the values shown in VTS’s API settings if you are not on the same machine defaults).
4. In the **Spout2** card:
    - Enable the **receiver** and select your VTS/VSF Spout source so the avatar appears in VConnect for alignment.
    - Enable the **sender**. The default sender name is `VConnectSpout`.
5. In **OBS**, add a **Spout2 Capture** source and select `VConnectSpout` (or whatever sender name you set). See the OBS steps below. Prefer this as your **only** avatar/effects source (not VTS + VConnect both on stream).
6. (Optional) Press **F3** only if you are capturing the VConnect window instead of using Spout.



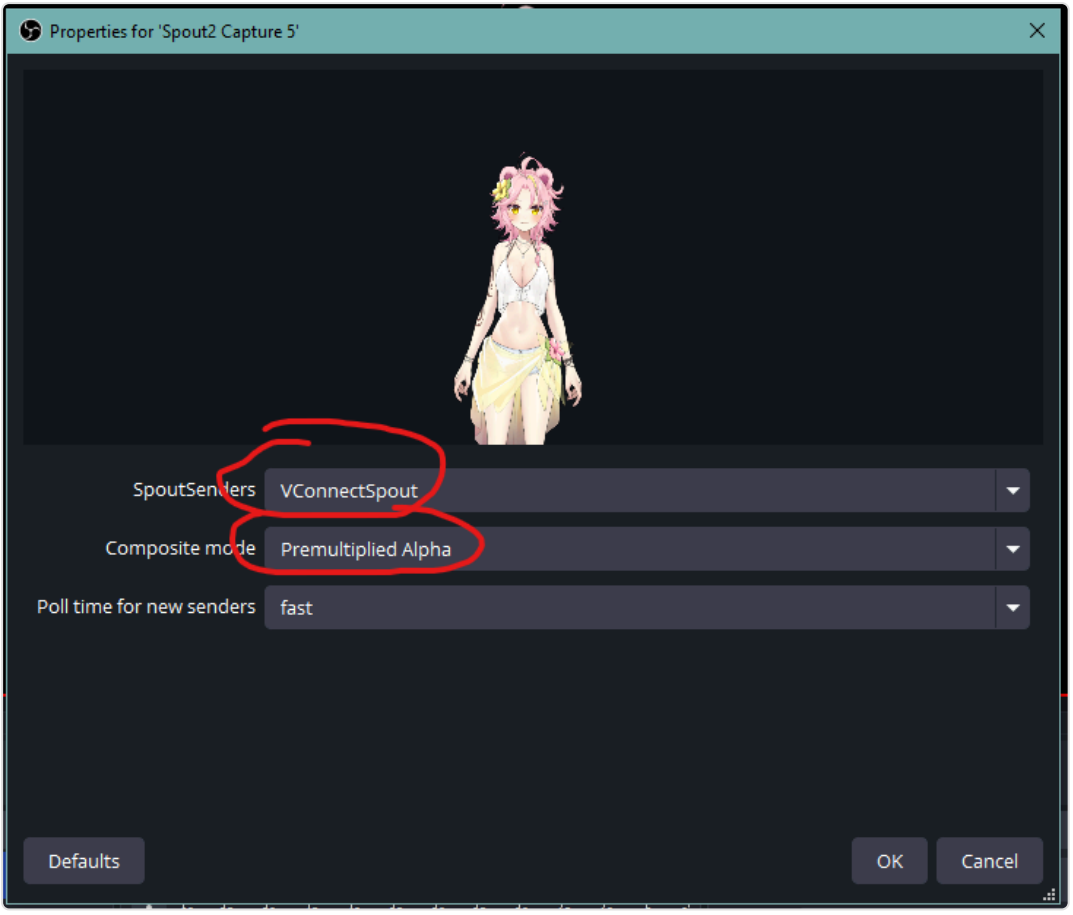


OBS: add the Spout2 source

1. In OBS, click **+** under Sources and choose **Spout2 Capture**.



2. Open the source properties and set:
   - **SpoutSenders** → `VConnectSpout` (or your sender name)
   - **Composite mode** → **Premultiplied Alpha** (recommended so transparency looks correct)



Linux: Spout2 through PipeWire

Spout2 itself only exists on Windows. On Linux, VConnect's **Spout2 card** works the same way, but uses **PipeWire** (Linux's built-in audio/video sharing) to move the image between apps. The names stay the same as on Windows: your avatar still arrives as `VTubeStudioSpout`, and VConnect still sends `VConnectSpout`.

```
VTS (in Proton) --spout2pw--> VConnect --PipeWire--> OBS ("PipeWire Video" source)
```

VTube Studio and OBS don't speak PipeWire video on their own, so each needs one free add-on:

Add-on	What it does	Needed for
spout2pw	Runs next to VTube Studio in Proton and turns its Spout2 output into a PipeWire stream	Getting your avatar into VConnect
obs-pwvideo	Adds a **PipeWire Video** source to OBS	Showing VConnect in OBS

VConnect itself needs nothing extra.

Before you start, check:

- **PipeWire 1.2.7 or newer.** Run `pipewire --version` in a terminal. Most current distros ship something newer.
- **GPU driver:**

    - **NVIDIA:** driver **555 or newer**, with `nvidia-drm` modeset enabled. Fedora, Arch, Nobara and Bazzite enable modeset by default.
    - **AMD and Intel:** any recent Mesa.
- **Proton:** VTube Studio runs with **Proton Experimental**, or **GE-Proton 10-29 or newer**. Proton 11 works.

1. Install spout2pw (once)

1. Download the newest `spout2pw-bin-x.x.x.tar.gz` from the spout2pw releases page. Not the `dynamic-experimental` one.
2. Extract it into a folder called `spout2pw` in your home folder. The archive has no folder of its own, so create one first:

   ```
   mkdir -p ~/spout2pw
   tar -xzf ~/Downloads/spout2pw-bin-*.tar.gz -C ~/spout2pw
   ```

   `~/spout2pw` should now contain `spout2pw.sh`, `spout2pw.inf` and a `spout2pw-dlls` folder.

Don't move or rename anything inside that folder. To update later, extract a newer release into the same folder.

2. Launch VTube Studio through spout2pw

First, find out which Steam you have: open Steam → **Settings** → **Storage**. If the path contains `.var/app/com.valvesoftware.Steam`, you have the **Flatpak** version.

Regular (non-Flatpak) Steam:

1. In your Steam library, right-click **VTube Studio** → **Properties**.
2. Under **General** → **Launch Options**, enter:

   ```
   ~/spout2pw/spout2pw.sh %command%
   ```

3. Under **Compatibility**, leave it on the default (Proton Experimental). If spout2pw later shows an error box about the Proton/Wine version, come back here, tick **Force the use of a specific Steam Play compatibility tool**, and pick a recent GE-Proton.
4. Start VTube Studio from Steam. If it was already running, close it and start it again.

Flatpak Steam: Steam can't see your home folder, so do one of these first, then follow the steps above with the same launch option:

- Move the `spout2pw` folder into `~/.var/app/com.valvesoftware.Steam/`, or
- Run this once: `flatpak override --user com.valvesoftware.Steam --filesystem=~/spout2pw`

If VTube Studio silently doesn't start at all, it's almost always this Flatpak issue.

3. In VTube Studio

- Open **Settings** → **Camera** tab → **Spout2 Config** → turn on **Activate Spout2**.
- Set the background to **Color Picker**, pure black, and turn on **Transparent in capture**. This gives clean edges around your avatar.

4. In VConnect

Open **Settings** → **Spout2** card (the same card as on Windows):

- Turn on the **receiver** and pick `VTubeStudioSpout` from the source list. The list shows everything spout2pw is currently sharing.
- Turn on the **sender**. The default name is `VConnectSpout`.

Start order doesn't matter: VConnect waits for VTube Studio to appear, and reconnects by itself if you restart VTS.

5. In OBS

Install **obs-pwvideo** to match how you installed OBS. To check: in OBS go to **Help** → **Log Files** → **View Current Log**; if it mentions **Flatpak Branch**, you have the Flatpak version.

Your OBS	Install obs-pwvideo with
Flatpak (also the usual case on SteamOS, Bazzite and other immutable distros)	`flatpak install com.obsproject.Studio.Plugin.OBSPWVideo`
Fedora package	`sudo dnf install obs-studio-plugin-pwvideo`
Anything else	Download `obs-pwvideo-X.X.X-x86_64-ubuntu-gnu.tar.xz` from the obs-pwvideo releases page, extract it, and move the `obs-pwvideo` folder into `~/.config/obs-studio/plugins/`

Restart OBS. Then click **+** under Sources → **PipeWire Video**, and select `VConnectSpout`.

If something doesn't work

Problem	What to check
VConnect shows **PipeWire Sender/Receiver Unavailable**	The popup says why. Usually it's an outdated GPU driver or NVIDIA modeset being off; fix that and restart VConnect.
`VTubeStudioSpout` isn't in the receiver list	Check that VTS was started through spout2pw (step 2) and that **Activate Spout2** is on. To see what's being shared, run `pw-dump | grep '"node.nick"'` in a terminal; `VTubeStudioSpout` should be listed.
VTube Studio won't start	See the Flatpak note in step 2. For details, change the launch option to `~/spout2pw/spout2pw.sh -debug %command%`; a log named `steam-*.log` then appears in your home folder.
Glitches after resizing the VTS window	Resizing a Spout2 sender is a known weak spot of spout2pw. Set the VTS window size before you go live.
Coloured halo or haze around your avatar	The VTS background must be **pure black** with **Transparent in capture** on (step 3).

More help: the spout2pw troubleshooting page.

VSeeFace works the same way: launch it through spout2pw. For transparency, add `--background-color '#000000'` to its launch arguments.

Other options: NDI (laggy on Linux; default sender name `VConnectNdi`) and **Game / Window Capture** of the VConnect window, with **F3** to hide the editor UI, both still work if you prefer them.



Optional: split output “sandwich”

This is not the intended use of VConnect. Split output is mainly a **workaround** for setups that insist on keeping VTube Studio (or another avatar source) in OBS and sandwiching VConnect effects around it. Prefer the single VConnect source path above whenever you can.

For layering effects **behind and in front of** your avatar in OBS, enable **Split Output** in Profile settings (Settings → profile / camera-related options).



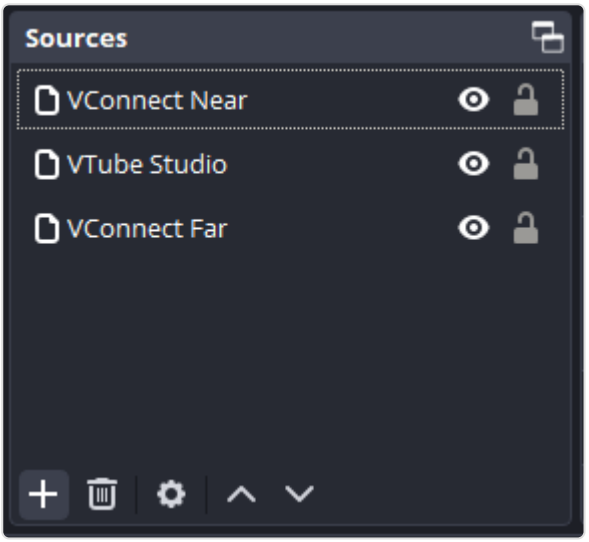
When Split Output is on, Spout/NDI exposes **two** senders from your base name:

Role	Default Spout names	Default NDI names
Near (front)	`VConnectSpout-Near`	`VConnectNdi-Near`
Far (back)	`VConnectSpout-Far`	`VConnectNdi-Far`

Important: With Split Output enabled, **disable the Spout and/or NDI receiver** in Settings. Your avatar comes from the VTS (or other) source in OBS instead; receiving it into VConnect as well is unnecessary for this path and can confuse the setup.

Typical OBS stack (top = in front):

1. VConnect **Near** (in front)
2. VTS / avatar source
3. VConnect **Far** (behind)



> ***Performance:*** *Split output **will** cost performance. It runs **two cameras** (Near and Far) so both layers render. Most users should stay on a **single** Spout/NDI sender and only use Split Output if they really need VTube Studio in OBS.*

Alternate path: VTS full-screen in OBS

If VTube Studio already fills your OBS canvas and you treat VConnect more like a classic overlay on top, you can skip a lot of Overlay Editor work. Scenes and hitbox tweaks matter more when your avatar is **not** full-screen or when you need precise alignment with a partial VTS source.

Again: the preferred setup is still **VConnect alone in OBS**. Keeping VTS full-screen in OBS is closer to a classic overlay workflow; if you need effects both behind and in front of that VTS source, use Split Output (and disable the Spout/NDI receiver as noted above).

Settings

You mostly open Settings to **connect services**. Day-to-day streaming rarely needs constant tweaking here.

Services

Connect what you use (the Platforms list is on the left of the Settings screen — see the VTS connection screenshot above):

- **Twitch**, **YouTube**, **Streamlabs**, **StreamElements**, **Pulsoid**
- **VTube Studio** (avatar / model data)
- **OBS** (scene linking in the Overlay Editor, and OBS nodes later)

Also available as protocol cards:

- **Spout2** / **NDI** (video I/O)
- **OSC**, **WebSocket**

> *Some services may not work yet if they still need server-side setup on the VConnect side. If a connect button fails, ask in* `#testing-discussion`.

Other Settings notes

- **Advanced Mode** (per profile, also available from Node Manager): unlocks more nodes and ports in the node menu. Leave it off until you need them.
- **Save location** (Global vs Local): where profiles and data are stored. Switching can require a restart and may show a migration prompt. Prefer leaving the default unless you know you need a local copy.
- **Graphics / Audio**: resolution, FPS cap, volumes, output device — change only if something looks or sounds wrong.

OBS tip: connect OBS in Settings if you want Overlay **scenes** to follow OBS scene changes, or if you plan to use OBS nodes in graphs.

Overlay Editor

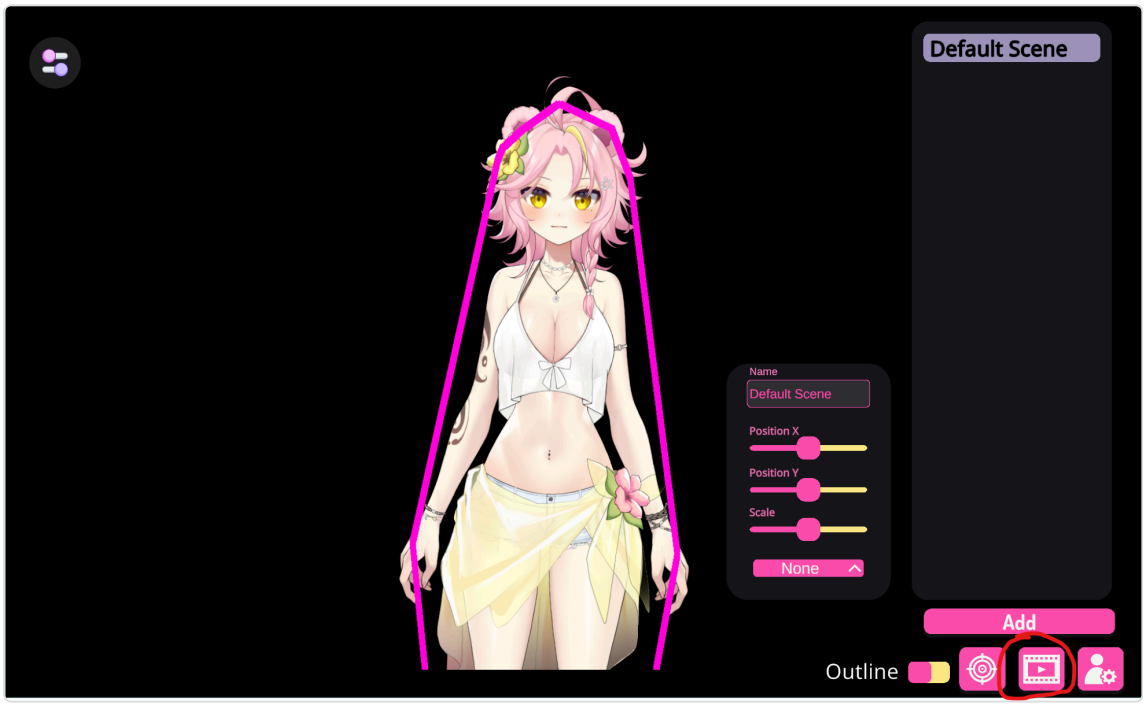
Configure how the live overlay lines up with your avatar.

You can largely skip this screen if VTS is full-screen in OBS and effects already look correct. Otherwise, use it to match layout and targeting.

Scenes

Scenes are for when your avatar / VTS source moves or scales between OBS layouts (same idea as in T.I.T.S.).

- Add, rename, remove scenes; set position, scale, and rotation.
- Link a VConnect scene to an **OBS scene** so VConnect switches automatically when OBS does (OBS must be connected in Settings).
- If VTS is always full-screen in OBS, you may never need more than the default scene.



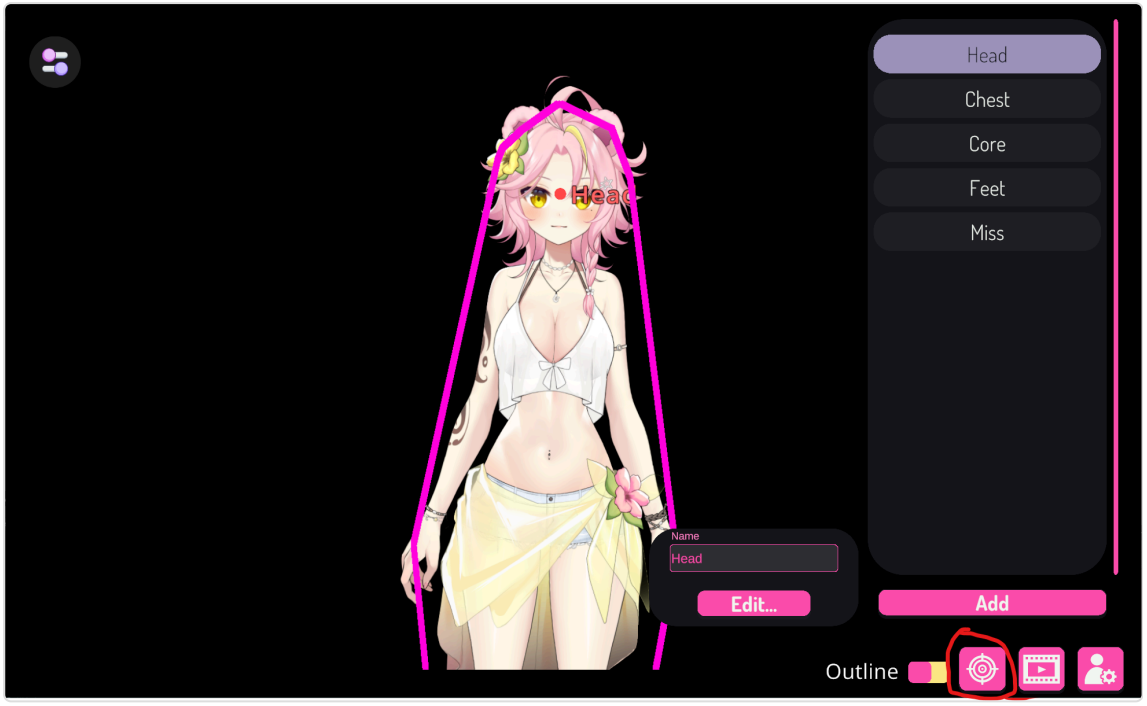
Targets

Think of a **target** as a **pin** on your model: a named position relative to the hitbox, used for throwing, sticking, anchoring, and other behavior.

Default targets:

- Head, Chest, Core, Feet, Miss

You can add, remove, rename, click-to-place in the preview, and reset to defaults.

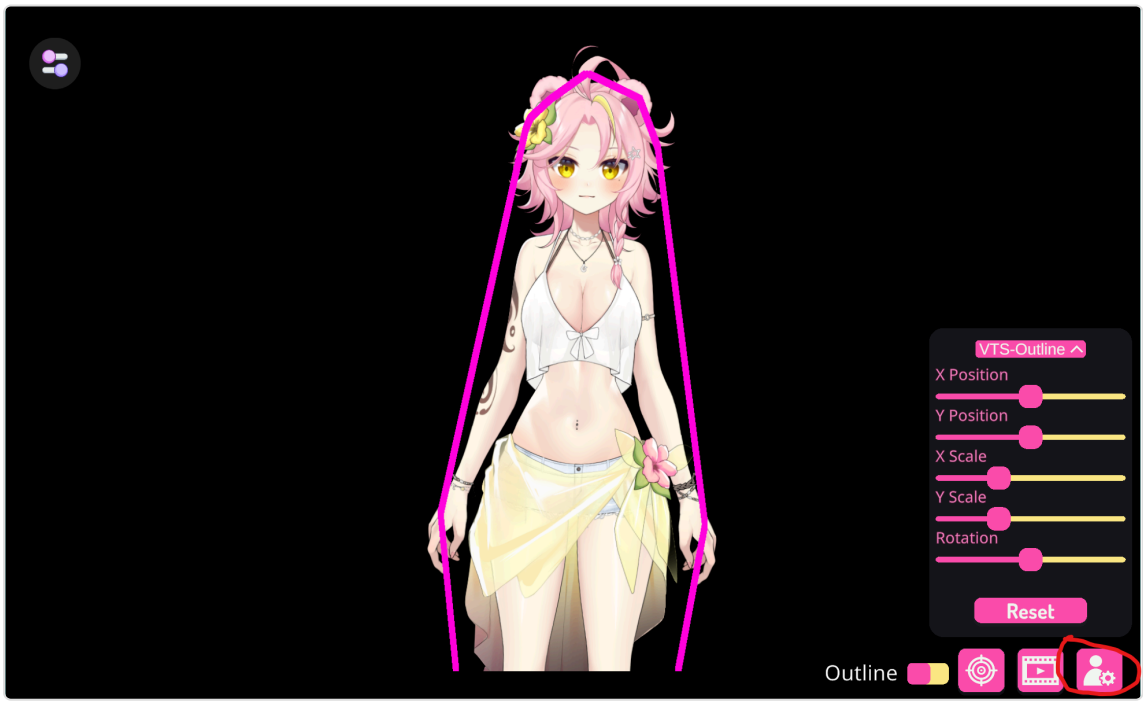


Model settings / hitbox

Adjust position offset, scale, rotation, and **display mode**:

- Human, Sphere, **VTS-Outline**, Cube

With VTS connected, you often leave this alone; use it when the hitbox does not match your model well, or when you are not using VTS.



Asset Manager

Import media into VConnect, organize it, edit properties, and share packs.

Import

Import detects type from the **file extension**. Sounds and throwable items live in the same library (unlike older T.I.T.S. splits).

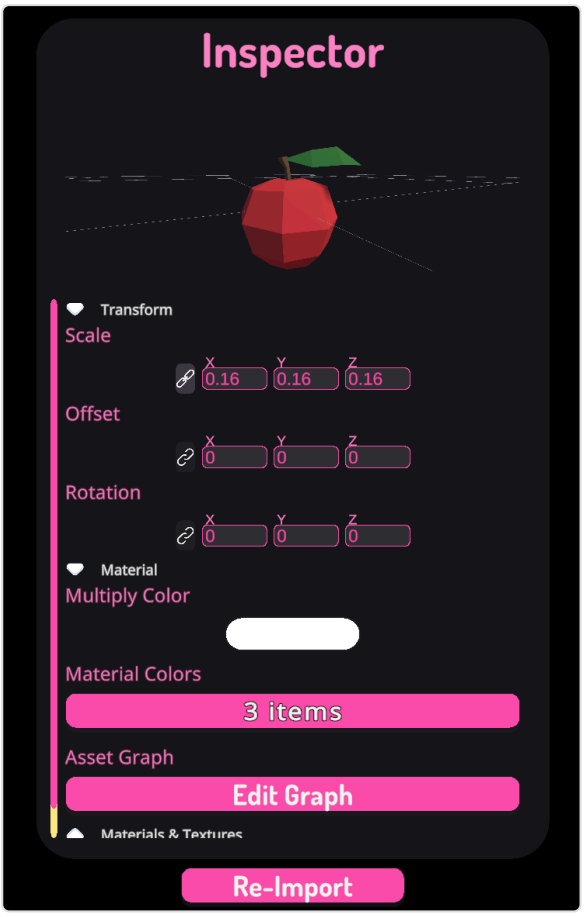
Common types:

Kind	Extensions (typical)
Images	`.png`, `.gif`
Audio	`.ogg`, `.wav`, `.mp3`
3D	`.fbx`, `.obj`, `.gltf`, `.glb`, … (archives like `.zip` may be unpacked on import)
Avatars	`.vrm`
Video	`.mp4`, `.webm` — on **Linux**, MP4 import may be unavailable and `.webm` must be **VP8**-encoded (VP9 is not supported, see Limitations)
Bundles	`.vasset` (VConnect asset export)

Use folders, search, move, rename, copy/paste, and the **inspector** on the side to edit asset properties.



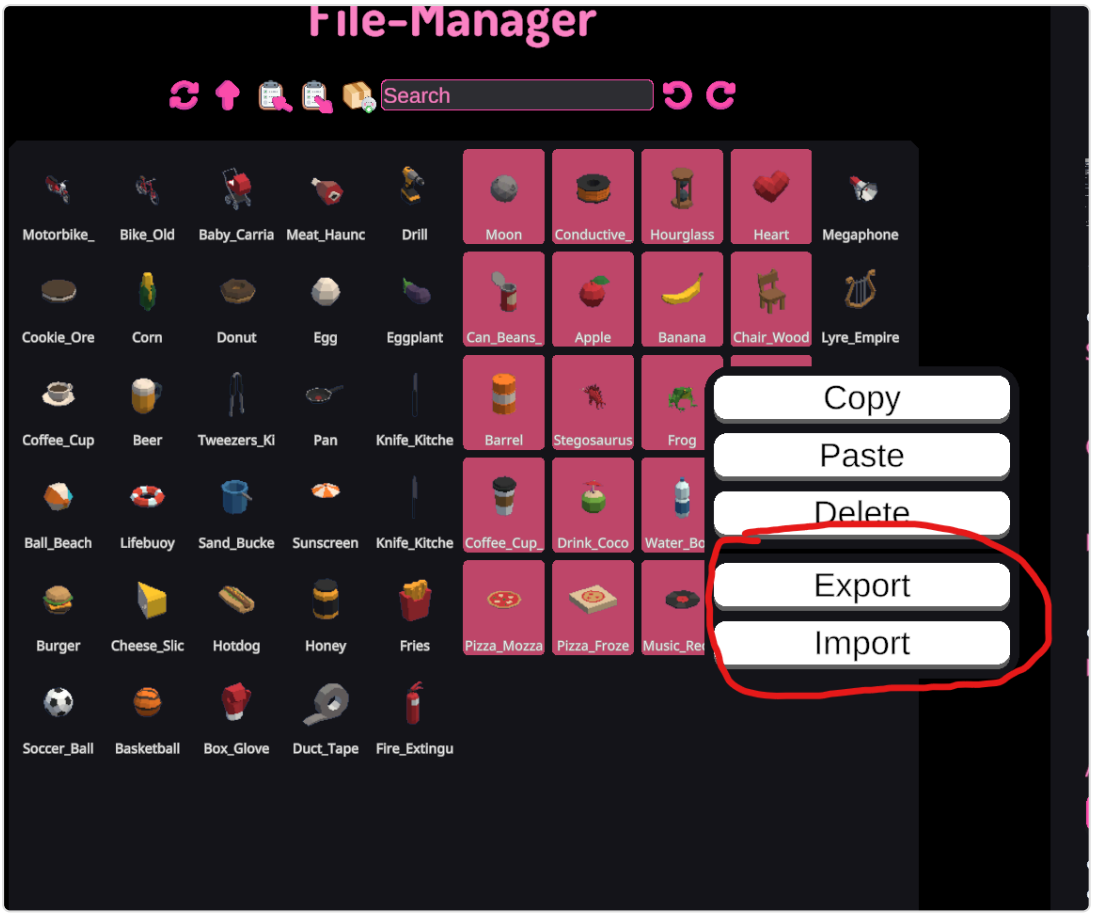
The browser: folders, search, and the imported-asset grid.



The inspector panel — properties for whatever asset is selected in the browser.

Export / share assets

Select one or more assets in the browser, then **right-click** to open the context menu and choose **Export**. That creates a `.vasset` bundle (optionally including referenced assets such as linked sounds). Others can **Import** that file the same way and get the same assets and settings.



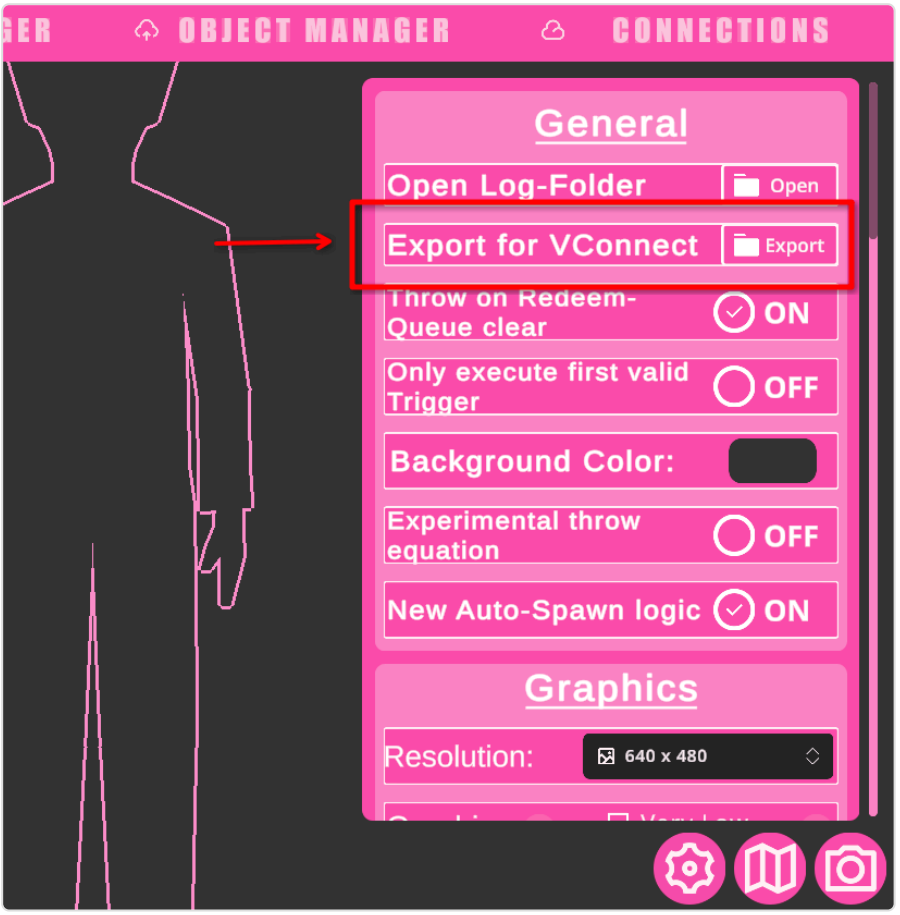
Optional: emote import

You can bulk-import emotes from **Twitch**, **BTTV**, **FFZ**, and **7TV** from the Asset Manager (handy for sticker-style assets).

Importing from T.I.T.S.

If you're coming from **T.I.T.S.** (Twitch Integrated Throwing System), you don't have to rebuild your setup by hand — you can export it from T.I.T.S. and import it directly into VConnect.

1. In T.I.T.S. (version **1.2.3.6** or newer), use the **Export** menu item to export your setup.



2. The export can take a while — be patient. Once it's done, the output folder opens automatically with a `.tits` file inside.

3. Import that `.tits` file into VConnect: either use **Import** in the Asset Manager, or drop the file directly onto the app window and it will import automatically. (Drag-and-drop is not available on Linux — see Limitations — so use the Asset Manager import there.)

Importing creates a **new profile** from the T.I.T.S. data rather than merging into your current one — if your existing triggers seem to have disappeared after an import, they haven't: they're still on their original profile, just not the one you're currently viewing.

This importer is still in **beta**. It makes a lot of "best guesses" translating T.I.T.S. concepts into VConnect's node system, since not everything carries over 1:1 — expect some setups to need manual touch-ups after import. It's being actively worked on and will keep improving. One known gap: the **bonk** feature is not currently supported by the importer (also in progress).

Node Manager

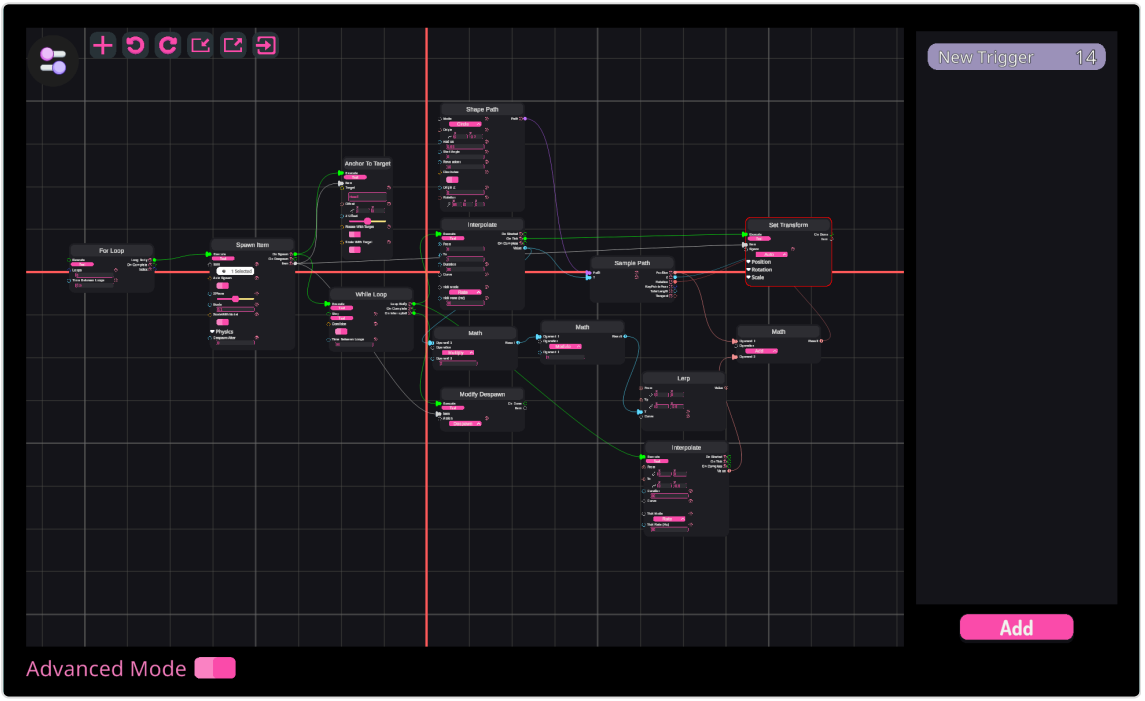
This is the core of VConnect. Behavior is driven by **triggers**: graphs that start from an event node (for example **On Channel Point Redeem**) and then run actions.

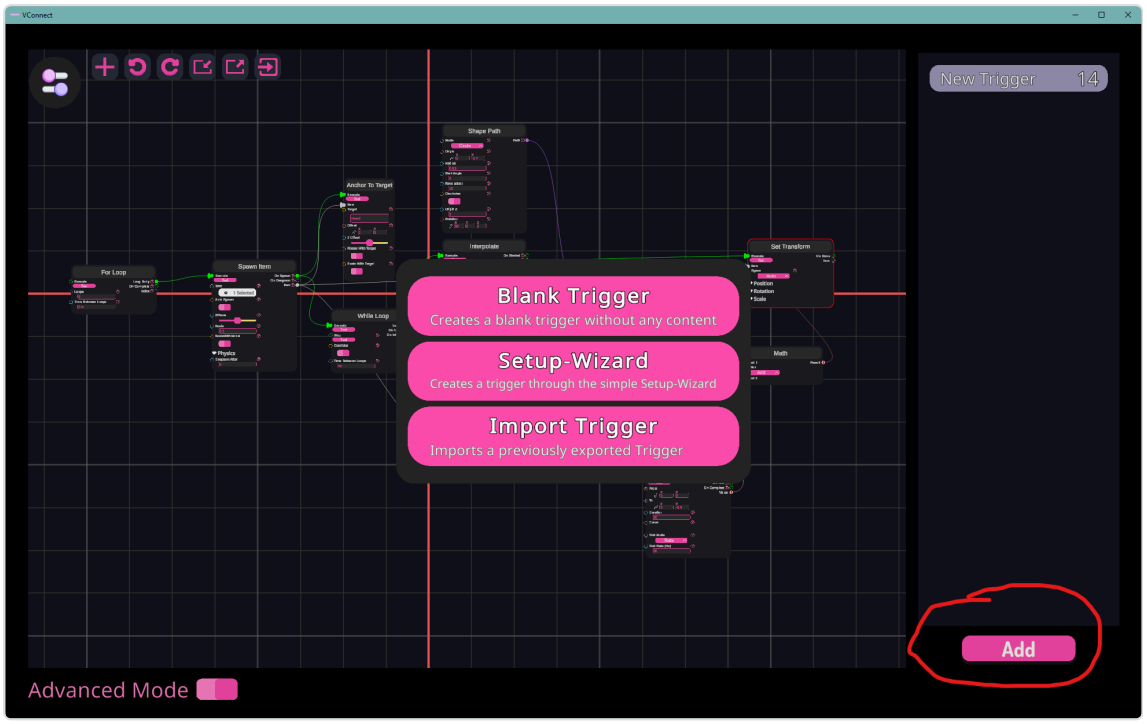
Node graphs are powerful — and can feel overwhelming. Use the **Setup Wizard** first; explore free-form nodes when you are ready.

Adding a trigger

1. Open **Node Manager**.
2. Click **Add** (bottom right).
3. Choose how to add the trigger — for beginners, pick the **Setup Wizard**.

You can also create a blank trigger and build the graph by hand, or **import** a shared trigger (see below).





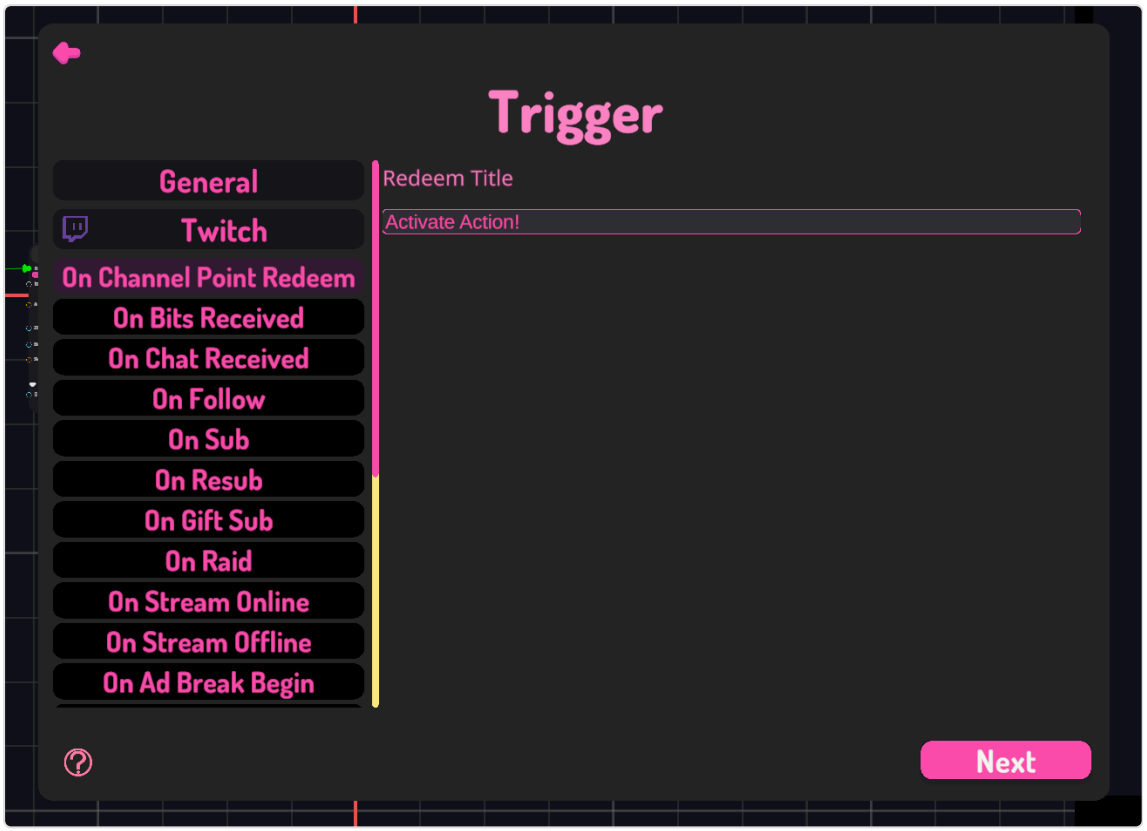
Setup Wizard

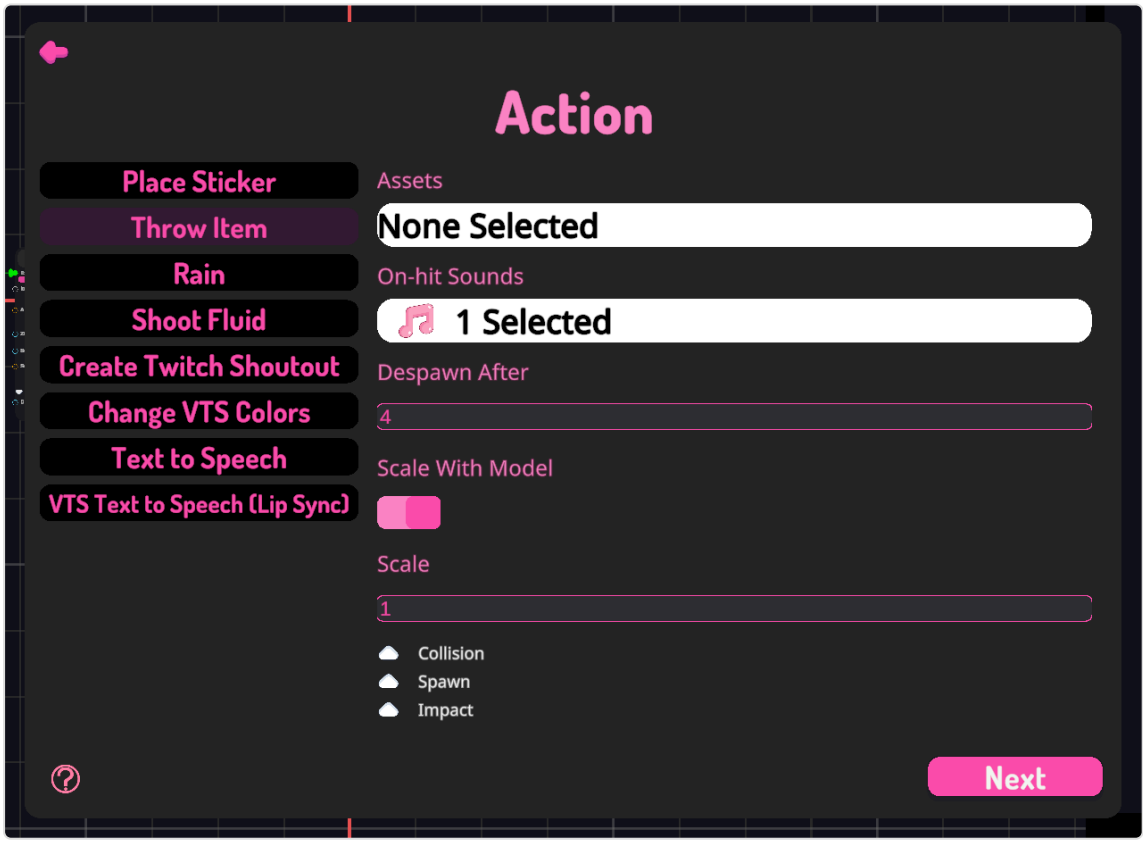
The wizard walks you through:

1. **Trigger** — what event starts the graph (Twitch redeem, bits, chat, YouTube super chat, donations, hotkey, OBS scene change, ...)
2. **Action** — what happens (see list below)
3. **Global options** — cooldown, delay, repeat

Built-in wizard actions include:

- Place Sticker
- Throw Item
- Rain
- Shoot Fluid
- Change VTS Colors
- Text to Speech
- VTS Text to Speech (Lip Sync)
- Create Twitch Shoutout


Trigger
General
Twitch
On Channel Point Redeem
On Bits Received
On Chat Received
On Follow
On Sub
On Resub
On Gift Sub
On Raid
On Stream Online
On Stream Offline
On Ad Break Begin
Redeem Title
Activate Action!
Next


Action
Place Sticker
Throw Item
Rain
Shoot Fluid
Create Twitch Shoutout
Change VTS Colors
Text to Speech
VTS Text to Speech (Lip Sync)
Assets
None Selected
On-hit Sounds
1 Selected
Despawn After
4
Scale With Model
Scale
1
Collision
Spawn
Impact
Next

Editing graphs

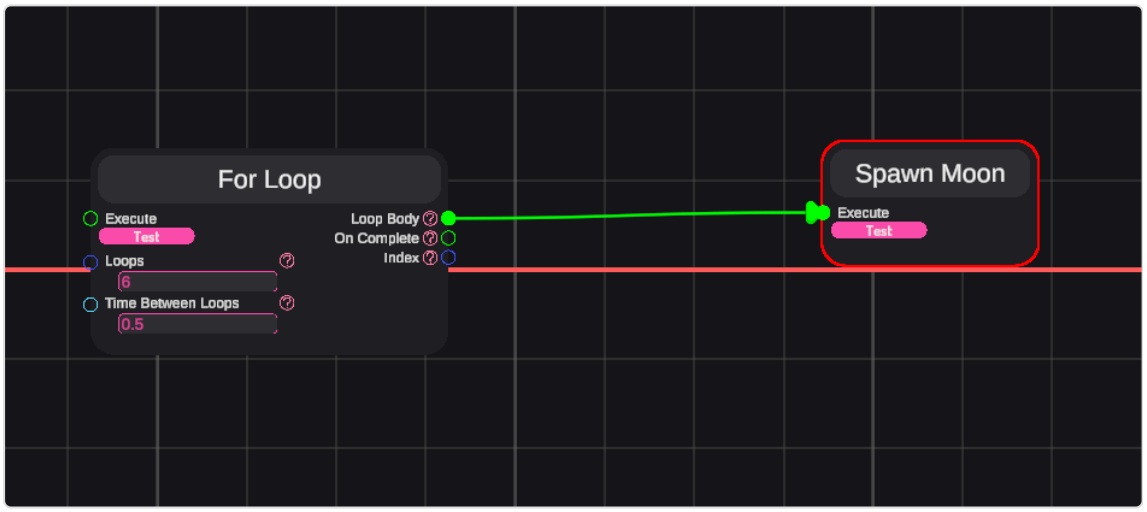
- Pan/zoom the canvas; open the **node menu** to add nodes by category (Twitch, VTS, Items, OBS, Flow Control, ...).
- Wire ports between nodes: drag from an **output** to add a wire, or from an **input** to move an existing one; hold **Shift** (or long-press) to invert. Dragging a new connection into empty space can open the node menu to continue the chain.
- **Advanced Mode** shows additional nodes and ports.
- Spawnable assets can also have their own small graphs (advanced); you do not need this for the quickstart.

Node groups

Select several nodes → create a **Node Group** to wrap them into something that behaves like a single reusable node.

- **Local Groups** — for the current profile
- **Global Groups** — available across profiles

Use groups when a chunk of logic repeats or gets too large to read.

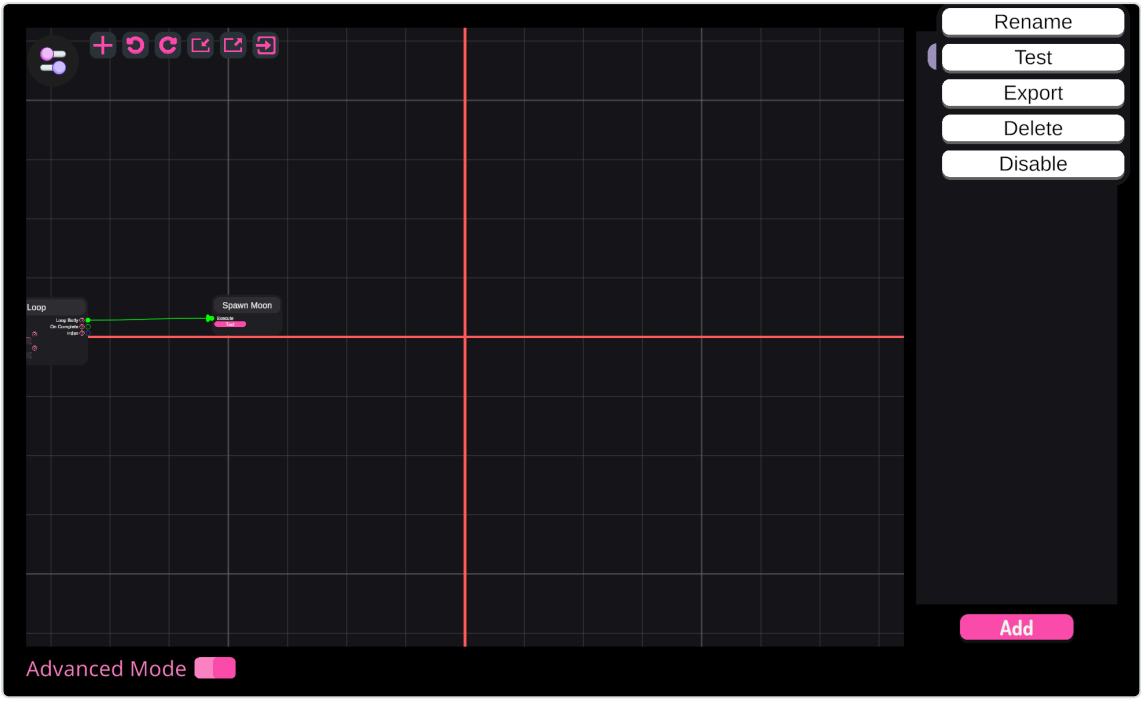


Import / export triggers

Format	Use
`.vtrigger`	Recommended — trigger **plus** referenced assets, so imports "just work"
`.json`	Trigger graph only (assets must already exist or be imported separately)

To export: **right-click** a trigger in the list (or use the **Export** action on the selected trigger) and save as `.vtrigger`.

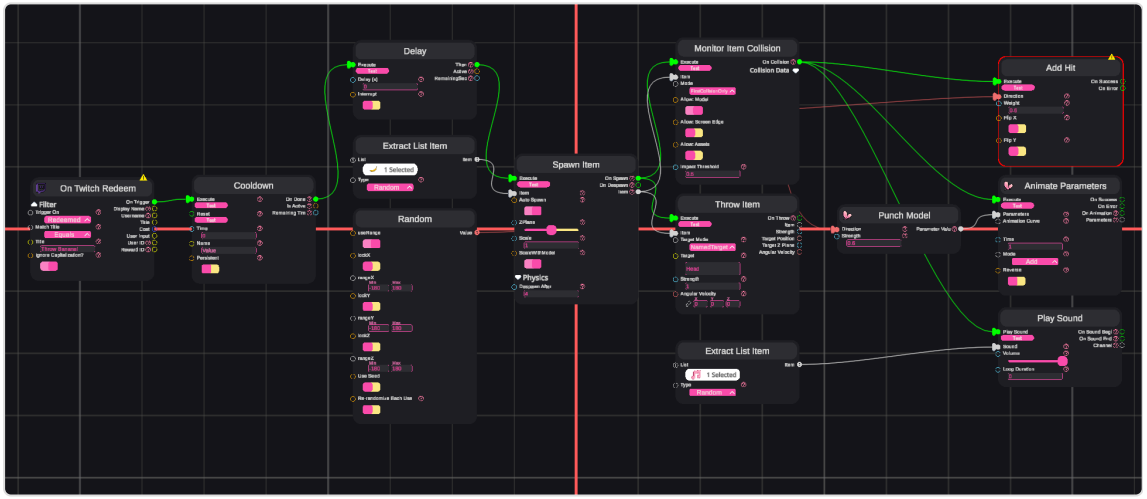
Share cool setups in Discord `#trigger-sharing`. Suggest new nodes in `#program-suggestions`.


Rename
Test
Export
Delete
Disable
Loop
Spawn Moon
Add
Advanced Mode

Your first working effect

End-to-end path for a Twitch channel-point throw:

1. **Settings**
   - Connect **Twitch**
   - Connect **VTube Studio**
   - Enable Spout2 **receiver** (avatar) and **sender** (`VConnectSpout`). On Linux this is the same card, see Linux: Spout2 through PipeWire
2. **OBS** — add Spout2 Capture for `VConnectSpout` (Premultiplied Alpha), or on Linux a **PipeWire Video** source for `VConnectSpout`
3. **Asset Manager** — import a PNG/GIF (or 3D item) you want to throw
4. **Node Manager** → **Add** → **Setup Wizard**
   - Trigger: **On Channel Point Redeem** (pick / filter your redeem as prompted)
   - Action: **Throw Item** — select the asset and a target (e.g. Head)
   - Set a short cooldown if you want
5. Redeem the channel point on Twitch and confirm the item appears on the overlay / in OBS
6. (Optional) Export the trigger as **`.vtrigger`** and share it



Cheat sheet

Topic	Reminder
Four screens	Overlay Editor · Asset Manager · Node Manager · Settings
Overlay Mode	**F3** toggles UI off/on (overlay only)
Best capture path	VTS/VSF → Spout into VConnect → **one** Spout `VConnectSpout` into OBS (**instead of** VTS in OBS)
Linux	Spout2 card works through PipeWire: launch VTS via **spout2pw**, use OBS's **PipeWire Video** source (obs-pwvideo) — see Linux: Spout2 through PipeWire; webm must be VP8; no drag-and-drop, use Import button (see Limitations)
Split output	Workaround only; disable Spout/NDI **receiver**; `…-Near` / `…-Far` sandwich in OBS; **2 cameras** → costs performance
Asset pack	`.vasset` (right-click in Asset Manager → Export)
Trigger pack	`.vtrigger` (right-click trigger → Export; includes assets)
Beginner triggers	Setup Wizard via **Add** in Node Manager
Help	`#testing-discussion`
Share triggers	`#trigger-sharing`
Suggest nodes	`#program-suggestions`

Advanced: Node Graph Concepts

Everything below is optional deeper material for people building their own node graphs. You don't need it to get a first trigger working — skip straight to Your first working effect if that's all you need today.

Coordinate System

VConnect uses **normalized coordinates**, the same convention as VTube Studio: `(-1, -1)` is the bottom-left of the overlay, `(1, 1)` is the top-right, and `(0, 0)` is the center. That's the space every **Position** port works in (X and Y range −1…1; a third Z value controls depth — lower is closer to the camera).

Position, Rotation, Scale, and Direction each use the space that makes them behave correctly, and they are **not** all the same space:

Property	Space	Why
Position	Normalized `[-1, 1]`	Matches VTube Studio directly, and the edges are always exactly ±1 no matter the resolution or aspect ratio.
Rotation	Degrees, **clockwise-positive**	A plain angle — there's no "normalized rotation" the way there is for a position.
Scale	World-space-consistent	Scale is tied to your model's actual on-screen size, not to the −1…1 box — so an item sized to match your model stays the right size no matter the overlay's aspect ratio.
Direction	Angle (degrees), not a position delta	Directions answer "which way," not "where" — built from angles (collision bounces, the direction-picker dial, applied forces), not by subtracting two positions.

The one place this bites: building a Direction by subtracting one normalized Position from another (`Target − Model`) inherits Position's −1…1 skew and reads wrong on non-square overlays. Use the **Math** node's on-screen subtract mode, or the **Screen Aspect** node below, instead of a plain subtract for that case.

Conversion nodes

Both live in the **Vector Math** node category and work in both directions via a **Mode** dropdown:

- **Screen Aspect** — converts a normalized vector to/from aspect-corrected "on-screen" space, where equal magnitude reads as equal on-screen distance in any direction. Use it before feeding a position-difference into Distance/Magnitude-style math, so the result isn't skewed by a non-square overlay.
- **Pixel Coordinate** — converts a normalized vector to/from raw pixel coordinates (top-left origin, Y down — the same convention OBS's own Transform panel uses), relative to VConnect's current overlay resolution. Use it to hand a VConnect position to an **OBS: Set Source Transform** node's Position port. This is accurate when your OBS canvas matches VConnect's overlay resolution — the standard setup from Your first working effect.

Node Groups

Select several nodes and turn them into a **Node Group** to reuse a chunk of graph logic without rebuilding it each time.

- **Local Groups** live on the current profile only.
- **Global Groups** are shared across every profile.

Placing a group in a graph is a **reference**, not a copy: every place you've dropped that group points back to the same central definition. Edit the group's internals once, and **every instance of it, in every graph, updates immediately** — this is powerful for maintaining shared logic, but also means a change is never scoped to just one instance.

Variables

Set Variable and **Get Variable** nodes let you stash a value under a name and read it back elsewhere — including in a different trigger's graph.

- Identify a variable by a **Key** (a string you choose) and a value type (number, bool, text, vector, color, or asset reference).
- By default, variables are **session-only** — cleared on restart.
- Toggle **Persistent** on to save a variable to disk so it survives an app restart.

Important: the **Persistent** toggle must match between the Set and the matching Get node for the same Key — a persistent Set and a non-persistent Get (or vice versa) are treated as two completely separate variables, and the Get will silently report "not found" rather than erroring.