Collision Commander

User Guide

Collision Commander — User Guide

Collision Commander is a single dockable editor panel opened via **Window** → **Collision Commander**. It surfaces the complete collision picture for your Unreal Engine project in one place, replacing the need to jump between Project Settings, Details panels, and per-component properties.

The plugin has five tabs: Matrix, Collision Compare, Actor Inspector, Experiment, and Validation.

Getting Started

Installation

Install Collision Commander from your Fab.com library via the Epic Games Launcher. Enable it in your project under **Edit** → **Plugins** → **search "Collision Commander"** → **Enable** → **restart the editor**.

A C++ project (or stub module) is required for the plugin to compile. Blueprint-only projects are fully supported once this prerequisite is met. The plugin itself adds no C++ requirement to your project code.

Opening the Panel

Go to **Window** → **Collision Commander** in the main menu bar, or click the toolbar button added to the Level Editor toolbar. The panel is a dockable Nomad Tab — drag it anywhere in the editor layout.

Understanding Query vs Physics

Unreal Engine stores two independent sets of collision responses per preset. The toggle at the top right of most panels controls which set you're viewing:

- **Query**: governs line traces, shape sweeps, and overlap events. Use this when diagnosing trigger volumes, hit detection, and line-of-sight checks.
- **Physics**: governs rigid-body simulation. Use this when diagnosing whether objects physically push against each other, land on surfaces, or pass through geometry.

> *A preset can Block a channel for physics (objects collide) while Ignoring it for queries (traces pass through). This is intentional UE design. Collision Commander is the fastest way to confirm which mode is configured how.*

How Pairwise Resolution Works

UE resolves interactions using the minimum of both actors' responses:

```
result = min(A's response to B's ObjectType, B's response to A's ObjectType)
```

Where Ignore (0) < Overlap (1) < Block (2). If either side is Ignore, the result is always Ignore regardless of the other side. Collision Commander applies this rule for every cell in the matrix.

Panel 1: Matrix

What it shows

An N×N grid where every row is a preset (Actor A) and every column is a preset (Actor B). Each cell shows the resolved interaction between that pair using UE's pairwise resolution rule.

Color legend:

- Red = BLOCK: physical or query contact occurs
- Yellow = OVERLAP: overlap event fires (if enabled)
- Grey = IGNORE: no interaction
- Dark blue = Diagonal: same preset vs itself

Visual separator

A thick border divides UE's 14 built-in presets (top-left block) from your custom presets (bottom-right block). This helps you spot when a custom preset is misconfigured relative to the built-ins.

Hover tooltip

Hovering any non-diagonal cell shows:

- PresetA's response to PresetB's Object Type channel
- PresetB's response to PresetA's Object Type channel
- The resolved result (the minimum of both)

Clicking a cell

Left-clicking any non-diagonal cell switches to the Collision Compare panel and pre-populates it with the row preset as the focus and the column preset as the first comparison entry.

Scrolling

- **Scroll wheel**: vertical scroll
- **Shift + Scroll wheel**: horizontal scroll

Both work from anywhere in the panel, including over headers.

Query vs Physics toggle

Use the **Query / Physics** toggle in the top bar to switch which response set is displayed. Both sets are always stored separately. The toggle only changes what you see.

Refresh

Click **Refresh** to re-read UCollisionProfile and rebuild the matrix after changing presets in Project Settings.

Panel 2: Collision Compare

What it's for

The Matrix shows the full picture, but hunting down a specific pairing in a large grid gets old fast. Collision Compare gives you a focused view: pick one preset as your focus, then add however many others you want to compare against. One table, all the pairings you care about.

Good example: set your Projectile preset as the focus and add all your character and enemy presets. You'll see in one shot what it blocks, overlaps, or ignores across the board.

Building your compare list

- Select a focus preset from the dropdown at the top
- Add comparison presets to the list below it
- Remove any entry with the x button on its row
- Hit Refresh after a Project Settings change to re-evaluate

Reading a result row

Each row shows the comparison preset name, a color-coded BLOCK / OVERLAP / IGNORE badge, the A->B raw response, and the B->A raw response. Hover any row for the full tooltip.

Overlap event warning

When a pairing resolves to OVERLAP, a warning icon appears on that row. This is a reminder that for BeginOverlap to actually fire at runtime, both actors need Generate Overlap Events enabled. A resolved OVERLAP result is necessary but not sufficient.

Navigating from the Matrix

Left-clicking any data cell in the Matrix tab automatically switches to Collision Compare, sets the row preset as the focus, and adds the column preset to the compare list. One click to drill into any pairing.

Panel 3: Actor Inspector

What it's for

A single actor can have a capsule, a skeletal mesh, a hitbox sphere, and a weapon trace component, each with its own preset and settings. Checking them individually in the Details panel is slow and gives you no combined view.

There's also a hidden problem worth knowing about: if you manually edit a component's collision responses in the Details panel without assigning a named preset, it gets set to **Custom**. That bypasses the preset system entirely and makes that component's collision harder to audit. The Actor Inspector calls that out explicitly.

How to use it

- Pick a Blueprint actor from the level or the Content Browser
- The panel lists every collision-capable component with its preset, enabled mode, and object type
- Build a list of target presets or channels below, and the panel renders a per-component interaction table
- Hover any cell for the tooltip with both contributing responses and the resolved result
- Hit Refresh after loading a new actor or changing Project Settings

Component table columns

- **Component**: Name and class
- **Preset**: Assigned collision preset name
- **Warning icon**: Shown if the preset is "Custom" (manually overridden)
- **Enabled**: No Collision / Query Only / Physics Only / Query and Physics
- **Object Type**: The channel this component identifies as

Custom preset warning

Any component using a Custom profile gets a warning icon with a tooltip explaining the issue. These warnings are also forwarded to the Validation tab automatically when you refresh.

Scrolling

- **Scroll wheel**: vertical scroll through component rows
- **Shift + Scroll wheel**: horizontal scroll to see more target columns

---

Panel 4: Experiment

What it's for

When you're thinking about adding a new preset or changing how an existing one behaves, there's no safe way to preview that in vanilla UE without actually editing Project Settings and then undoing if it's wrong. Experiment is a sandbox for that. Design a preset, tweak its channel responses, and see exactly how it would interact with everything else in your project before you commit to anything.

How to use it

- Load an existing preset as a starting point, or start from scratch
- Set a name, object type, collision enabled mode, and per-channel responses
- The interaction table at the bottom updates live as you change values
- Nothing is saved anywhere until you hit **Create Preset**

Committing

When you're happy with the result, press **Create Preset**. You'll get a confirmation dialog before anything writes. If you loaded from an existing preset and kept the same name, you'll be asked to confirm an overwrite. That's the only point in Collision Commander where data gets written back to Project Settings.

Visual indicator

When an experiment is in progress the panel shows a visible indicator so you always know you're in a modified state. Close without committing and everything is discarded.

> *Note: Creating a preset writes to your project's DefaultEngine.ini collision config, the same as doing it manually in Project Settings -> Collision. Check the Matrix after committing to confirm the result looks right.*

Scrolling

The whole panel scrolls vertically so all sections remain accessible at any window height.

Panel 5: Validation

What it's for

Collision misconfigurations are silent. A preset that ignores WorldStatic causes actors to fall through the floor. An asymmetric Block/Ignore pair behaves differently depending on which actor initiates the query. Overlap callbacks that never fire waste engineering time. These issues often go unnoticed until a specific gameplay scenario exposes them.

Severity levels

- **Error**: Misconfiguration that will silently break collision. Fix before shipping.
- **Warning**: Likely unintended setup, worth investigating. May be intentional.
- **Info**: Observation, not necessarily wrong.

When it runs

Validation runs automatically on every Refresh and after every Experiment commit. The tab strip badge in the top bar always shows the current error, warning, and info counts, visible from any tab.

All 10 validation rules

Rule ID	Description
CS_CHAN_BUDGET_WARN	14+ of 18 custom channel slots in use. Audit before adding more.
CS_CHAN_BUDGET_CRIT	17+ slots used. Consolidate immediately, you're near the limit.
CS_CHAN_UNUSED	Custom channel defined but no preset responds non-Ignore. Remove or reserve.

CS_PRESET_DUPLICATE	Two presets with identical ObjectType and response arrays. Consolidate.
CS_PRESET_ALL_IGNORE	A preset ignores every channel. Likely misconfiguration.
CS_STATIC_IGNORED	Dynamic ObjectType preset ignores WorldStatic. Actor will fall through the floor.
CS_OVERLAP_MISMATCH	A wants to Block B, but B ignores A. The Block is silently overridden to Ignore.
CS_NAME_CONFLICT	Custom preset name matches a UE built-in exactly. Rename to avoid lookup ambiguity.
CS_OVERLAP_EVENT	Two presets resolve to Overlap but Generate Overlap Events may not be enabled.
CS_VISIBILITY_BLOCK	A preset blocks the Visibility channel. This affects AI sight, shooting traces, and camera.

Reading results

Each row shows the rule ID, a description, and a suggested fix. Hover for the full tooltip including affected preset and channel names. Double-click any result to jump to the Matrix tab.

Component warnings

When an actor is loaded in the Actor Inspector and Refresh is clicked, any components using the Custom profile are forwarded to the Validation panel automatically as CS_COMPONENT_CUSTOM_PRESET entries.

Per-rule toggles

Each rule can be individually enabled or disabled via **Edit -> Project Settings -> Collision Commander**.

Summary

Tab overview:

- **Matrix**: Live N×N grid showing every preset-vs-preset interaction at a glance
- **Collision Compare**: Focused view — audit one preset against a chosen list of others
- **Actor Inspector**: Full collision surface of a Blueprint actor, component by component

- **Experiment**: Design and preview a new preset safely before committing it
- **Validation**: Automatically detect common collision misconfigurations

Open the panel: **Window -> Collision Commander**

For support, email support@paracosm.gg
