

Performance Monitoring HUD

Outline/Proposal

Let's create an in-game, heads-up-display for runtime performance information.

The goal is to provide at-a-glance, discipline and asset-specific runtime information for artists to help them maintain performance targets.

We'll design this to give a definitive, go/no-go answer to the question, "is my work performant?"

Aggregating the Best Stats for the Task

At its most basic level, this is simply a tool for organizing existing runtime UE stats. As it stands, stats are scattered all over the engine, have confusing names, and are often opaque in terms of describing what they measure.

A first pass at this tool is creating clear, Tech Artist-designed templates for specific use cases. For example, a VFX-focused template would display the total time used by VFX elements, a "you're over-budget" indicator, and the asset names of the of worst offending elements

Features Needed

- Template HUD UI
- Configurable mechanism for selecting existing stats
- Logic for determining pass/fail, best/worst
- Buckets/discipline budgets (e.g. 2.5ms for VFX, 5ms for Lighting, etc)
- CPU/GPU/DRAM/VRAM display

Quick-reference Buttons/Hotkeys

While developing assets, we can often get a back-of-the-envelope idea about problem areas by engaging a debug mode. For example, if you pause rendering, you can easily decide if a bottleneck is CPU vs. GPU. Often, only Tech Artists know these tricks. Let's make these more accessible.

Features Needed

- In general, configurable feature on/off switches in templates
- Pause rendering button/hotkey
- More examples:
 - Toggle lighting groups
 - Toggle VFX
 - Toggle material features

Stat Relationships, Optimization Techniques

Often, Tech Artists employ their own, personal techniques or algorithms for optimization. These could involve things like aggregating parallel stats measures over asset types, measuring stat dependencies, or general "min/maxing" over a set of things. Ultimately, the goal of this tool is to encapsulate these techniques into formats we can easily display in the HUD.

Features Needed

- API for managing stat relationships, displaying in HUD
- A/B stat snapshots, comparisons

Metadata

Sometimes, the quickest way to debug a performance issue is to take a screenshot and send it to a Tech Artist. Let's make this easy.

Features Needed

- Screenshot or video captures with specific, burned-in info for Tech Artists
- Quick launch an Insight trace with annotated events; upload to specific cloud location

Extra Features, Goals

If this tool is successful, there are things we could add.

Features Needed

- Additional engine stats for problem areas
- User-friendly UI for template creation (assuming the initial pass will be limited in terms of flexibility)
- Configure by stat picker checkbox (rather than console command)
- Ensure cross-platform compatibility, especially on consoles
- Emulate low-spec PC