Diagnosing Performance Problems

1.The Profiler window

2.Diagnosing performance problems using the Profiler window

The CPU usage profiler

Frame rate is a standard measure of performance in games.

Frame rate, or how fast the frames are being rendered, is measured in frames per second (FPS).

There are three types of thread involved in Unity’s rendering process: the main thread, the render thread and worker threads.

Vertical Synchronization, known as VSync, is used to match a game’s frame rate to the refresh rate of the screen it is being displayed on.

Rule out VSync

Hiding VSync information in the CPU usage profiler

Disregarding VSync information in the Hierarchy view

Whenever we see a function called WaitForTargetFPS in the Hierarchy view, this means that our game is waiting for VSync.

Disabling VSync

VSync cannot be disabled on all platforms: many (such as iOS) enforce its use.

Edit > Project Settings > Quality > VSync Count > Don’t Sync.

Renderer Profiler

Identifying if our game is GPU bound

The quickest way to identify if our game is GPU bound is by using Unity’s GPU usage profiler. Unfortunately, not all devices and drivers support this profiler.

Look at the area in the centre of the screen where the CPU and GPU times for the currently selected frame are displayed.

If the GPU time is excess of the CPU time, we have identified that our game is GPU bound at this point.

If the GPU usage profiler is not available for our target hardware, we can still identify if our game is GPU bound.

If the function Gfx.WaitForPresent is taking the longest in the CPU Usage profiler, this indicates that the CPU is waiting for the GPU. This means that our game is GPU bound at this point.

If we have identified that our game is GPU bound, we should read the tutorial on Fixing Performance Problems: Optimizing graphics rendering in Unity games.

Identifying if our game is CPU bound

Select the Time ms column to sort the functions by time in milliseconds.

If the selected function is a rendering function, the CPU profiler graph will highlight it as part of the Rendering display.

If we have identified that our game is CPU bound due to rendering issues, we should read the tutorial, Fixing Performance Problems: Optimizing graphics rendering in Unity games

Garbage Collection profiling

Select the Time ms column to sort the functions by time in milliseconds.

If the function GC.Collect() appears and is taking up excessive CPU time, then we have identified a garbage collection problem with our game.

Physics profiling

If we have ruled out rendering and garbage collection, let’s check to see whether complex physics calculations are the cause of our performance problem.

Slow scripts profiling

Other causes of performance problems

Although we have covered four of the most common causes of performance problems, our game may have a performance problem that is not related to these areas. If this is the case, we should follow the same approach as for the problems above: gather data, investigate with the CPU usage profiler and find out the name of the function that is causing the problem.