Texture Streaming Configuration

System for loading and unloading textures into and out of memory during play.

After checking the accuracy of your built texture streaming, you may want to adjust the texture streamer's behavior and priorities. The parameters below can be adjusted through configuration files, and most of them can also be updated at runtime from the console window.

Command	Description
`r.TextureStreaming`	This enables or disables the texture streamer. When it is disabled, all texture mips will get fully loaded in memory, even if the texture is never used for rendering. You can toggle this option at runtime, if needed.

Command	Description
`r.Streaming.PoolSize`	The pool size in MB available for textures in the engine. This pool contains UI textures, NeverStream textures, cubemaps and streaming textures. On some platforms, this pool can also hold non texture resouces like GPU particle buffers and vertex buffers. When set to 0, the pool size will be unlimited.
`r.Streaming.UseFixedPoolSize`	When using a non-zero value, the texture pool size can be changed at runtime.
`r.Streaming.FramesForFullUpdate`	The number of frames between each full update of the texture streamer. Each update recomputes the required resolution of each texture and generate mip load or unload requests. Higher values reduce the texture streamer CPU usage while lower values increase its reactivity.
`r.Streaming.UseNewMetrics`	For compatibility only. When set to false, the texture streamer will proceed as per 4.12 version.
`r.Streaming.MaxTempMemoryAllowed`	The amount of temporary memory in MB allowed for updating the textures. This should be big enough to avoid starving the texture streamer while being small enough to prevent wasting (unused) memory.
`r.Streaming.HLODStrategy`	This controls the loading strategy for hierarchical LOD textures : • 0 : Allow streaming of all mips • 1 : Only stream the last mip. The other mips are always loaded. • 2 : Don't stream any mips. All mips are always loaded.

Command	Description
`r.Streaming.HiddenPrimitiveScale`	This controls a scale applied to the "wanted" resolution when the component referring to a texture is not visible (i.e. its bounding box is occluded). This only affects the resolution before it gets clamped by the max available resolution to avoid downgrading textures that are already limited. In other words, it only affects textures that have appropriate resolutions for the viewpoint.
`r.Streaming.MaxEffectiveScreenSize`	When non-zero, this will limit the screen size considered by the streamer when computing the "wanted" resolution of textures. This will prevent high resolutions from requiring significantly bigger streaming pools.
`r.Streaming.Boost`	This is a global boost affecting the "wanted" resolution of textures.
`r.Streaming.MipBias`	This is a global mip bias used to prevent the streamer from loading the biggest mips for each texture. It is used to accommodate a low streaming pool. This also affects which mips the streamer tries to load for any viewpoints, as with a small `r.Streaming.Boost` or a limiting `r.Streaming.MaxEffectiveScreenSize`. There are a few exceptions to how this bias is applied : • Terrain & Landscape textures : bias has no effect. • Hierarchical LOD textures : bias does not limit max resolution. • Lightmaps & Shadowmaps : bias only limits max resolution.
`r.Streaming.UsePerTextureBias`	This is used to limit the effect of the global mip bias by limiting its effect to the max allowed mip and not the "wanted" mips from any viewpoint. It

Command	Description
	will then apply it to textures only in order to fit in the streaming pool. Each texture has its own mip bias, ranging from 0 to `MipBias`, depending on the budget computation.
`r.Streaming.DropMips`	This is a debug option used to prevent keeping mips in memory even if the streaming pool allows it: • 0 : Drop no mips. • 1 : Drop cached mips. • 2 : Drop cached and hidden mips. The purpose of this is to assets the accuracy of the wanted mip computation, as the displayed resolution can be affected by previous load requests or hidden primitives.
`r.Streaming.FullyLoadUsedTextures`	This will stream all used textures to their max available resolution and keep them in memory for as long as the application is opened. This should be used as an alternative to disabling texture streaming completely, which would load textures that are never used and ultimately, requiring more memory usage.
`r.Streaming.UseAllMips`	Whether or not to remove all resolution limitations coming from texture group settings and cinematic settings. This should only be used when showcasing art or making promotional material.
`r.Streaming.AnalysisIndex`	This is used in Material Texture Coordinate Scale Accuracy View Mode to investigate the accuracy for individual textures.

Command	Description
`r.Streaming.CheckBuildStatus`	If enabled, the engine will check if texture streaming needs to be rebuilt, displaying a warning when simulating or in Play in Editor (PIE) mode.
`r.Streaming.DefragDynamicBounds`	If enabled, unused dynamic bounds will be removed from the update loop.
`r.Streaming.LimitPoolSizeToVRAM`	If enabled, the texture pool size will be limited to how much GPU memory is available.
`r.Streaming.MinMipForSplitRequest`	When non 0, this allows the texture streamer to first load the visible mips from a texture when non visible mips must also be loaded. This improves the visual quality by prioritizing visible mips over force loaded mips or hidden mips that could become visible.
`r.Streaming.NumStaticComponentsProcessedPerFrame`	If non-zero, this sets the number of components the engine will incrementally load per frame before the level is visible. This applies to components with their mobility set to static. The default value is set to 50.
`r.Streaming.ScaleTexturesByGlobalMyBias`	Whether the streaming texture's "wanted" resolution will be scaled down by the global mip bias.
`r.Streaming.UseMaterialData`	Whether the material texture scales and the coordinates will be used for texture streaming.