We are very pleased to bring you the 5.5.0 Preview 5 release. Our Dev Team have been busy improving on the earlier previews that were released on 20 March 2018, 03 May 2018, 24 May 2018 and 07 June 2018.

As always we really do welcome further feedback and comments from the CRYENGINE Community, but please can these come through our normal channels i.e. the Github Issue Reporter and the 5.5 Feedback Thread.

Accessing the 5.5.0 Preview 5 Release

Engine versions are now available through the CRYENGINE Launcher.

1. Log into the CRYENGINE Launcher.
2. Go to **LIBRARY -> My Engines** where your currently installed and available Engines are listed. From there Engines can be updated (installed).

Github

Preview Engine versions are also available from Github, however Sandbox Editor source code is only available via Github.

1. Go to: https://github.com/CRYTEK/CRYENGINE/releases/5.5.0_preview5
2. Download CRYENGINE_preview_5.5.0.298_pc.zip
3. Unzip it somewhere and open the directory “CRYENGINE_preview_5.5.0.298_pc”
4. Double click on InstallEngine.bat

For more information, see the Important CRYENGINE 5.5 Data and Code Changes article.

If you are upgrading from CRYENGINE 5.4, please read this topic: Migrating from CRYENGINE 5.4 to CRYENGINE 5.5.

- Reflections are broken if a material is attached to a object. Can be tweaked with "r_ssreflsamples" and "r_ssrefldistance".
- Game will crash on startup with Vulkan.
- Legacy PFX can cause a crash shortly after being added to a level.
- Changes to a particle effect in the new Particle Editor are not directly applied to the Particle Effect instances in the level (currently a level reload is necessary).
- Editor crashes if a particle effect is placed in the level and is opened in Particle Editor.
- Designer objects may be invisible after levelload/chainload.
  Workaround: Enable, then disable **e_permanentRenderObjects 0/1**.
- C#: Using the CRYENGINE C# extension for Visual Studio 2017 to start the GameLauncher, Sandbox Editor or Server can cause an error when used in the latest version of Visual Studio 2017. More information and a fix for this issue can be found here.

Animation General

- **Fixed:** Bug where m_fZoomDistanceSq would change during successive render passes and cause culling of attachments during skinning.
- **Fixed:** (Renderer) Fix to keep support for per-attachment nearest flag.

Middleware Updates

- Updated to Wwise SDK v2017.2.6 build 6636.
- Updated to Fmod Studio 1.10.06.
- Updated Oculus spatializer plugin for Wwise 1.27.0.

Audio General

- **New:** View dist ratio for clip volumes.

Engine General

- **New:** CryAssert now breaks on the erroneous line of code and not 2 calls deeper. Also added CryVerify.
- **Fixed:** HeapAllocator.Array now aligns both the beginning and end of allocations (for safe SIMD access).

Common

- **Fixed:** log2 calculation for values of "1".
- **Tweaked:** Added 64bit atomic increment (needed for atomic time-value accumulation).

System

- **Fixed:** (Screenshot Grabber) 3 race conditions and now allow multiple consumers.
- **Fixed:** Thaw Render thread and do not enforce exclusive log access for crash screenshot.
- **Fixed:** Use ::PostMessage instead of ::ShowWindow from non-owning thread for window minimization.
- **Fixed:** DynArray initalizer_list ctor.
- **Fixed:** (XboxOne) Missing era.xvd (attempt 2).

- **Tweaked:** Division operators for vectors (which failed badly with integers).
- **Tweaked:** (XboxOne) Update XDK to version April 2018.

Default Entities

- **Fixed:** Linking an entity with a Light Component to a clip volume doesn't work.
- **Fixed:** Map load causes collider component settings to change.
- **Fixed:** Decals are not visible upon level load.
- **Fixed:** Cannot assign lens flares to entity with 'point light' component.

Movie System

- **Fixed:** Deactivate screenfader when AnimScreenFaderNode becomes inactive.

Renderer General

- **New:** Multiple modes for refraction resolve.
- **New:** Profile section to detect GetOrCreateInputLayout's shader reflection request.
- **New:** Screenfader post-process effect.
- **New:** Added DXGI memory statistics to DX11.
- **New:** Proper safety check for TerrainSectorTextureInfo pointer in instance buffer update.
- **New:** Added graceful handling of running out of constant-buffer memory.
- **New:** Added frameID to groups in the RenderPipelineProfiler / PipelineProfiler can now return detailed frame info stats.
- **Refactored:** SceneRenderPass profiling refactor.
- **Refactored:** Stopped all resource tracking on the high level.
- **Refactored:** Store transformed AABBs of the render element for permanent and temporary render objects.
- **Refactored:** Pass particles' AABBs to CREParticle via jobs.
- **Refactored:** Statistics code.
- **Refactored:** Cleaned up constant buffer slots and reflection CVars.
- **Refactored:** Removed reverse-z CVar.
- **Refactored:** Transparent renderitem list into above-water transparent and below-water transparent.
- **Refactored:** Remove displayContextHandle and replace it with a type coherent key.
- **Refactored:** Added UAV supporting clear functions in clear region/surface passes.
- **Refactored:**Particle constant buffer race condition and move tiled shading shadow frustum slices generation to 3D Engine.
- **Refactored:** Exposed CVar for changing frame latency. Make sure frame fence blocks before other fences in case we're GPU bound.
- **Refactored:** Removed dx9 style texture semantics from various shaders.
- **Refactored:** Removed reserved stencil value for MSAA. Sort clip volumes by view distance before adding to renderer to reduce visual impact when max clip volume count is exceeded.
- **Refactored:** Replaced r_shadersorbis, r_shadersdx10, r_shadersdx11, r_shadersGL4, r_shadersGLES3, r_shadersdurango, r_shadersVulkan CVars with r_ShaderTarget CVar which specifies the target which shaders must be built for.
- **Refactored:** Storing detailed profiler stats double buffered now instead of temporary to avoid race condition / Removed need to access using frameID from the outside.
- **Optimized:** Removed possible WaterOcean displacement upload sync (because of pending uploads).
- **Optimized:** Use only as many back-buffers as configured latency (instead of max.) - reduced memory consumption.
- **Optimized:** Make resources which are temporal (addref/release within a frame) available to recycling immediately after release (no race possible).
- **Optimized:** Enable permanent objects for refractive objects and use the tighter AABB for resolve passes.
- **Optimized:** Fixed most of the clear-values to be correct.
- **Optimized:** Topologically sort transparent items, injecting the minimum refraction resolve passes required.
- **Optimized:** (CResourceBindingInvalidator) O(1) RemoveInvalidateCallbacks() instead of linear time and a single pass InvalidateDeviceResource().
- **Optimized:** Support direct access constant buffers on DirectX 11.1
- **Optimized:** Tell DXGI to reserve 7/8 of video memory for CRYENGINE.
- **Optimized:** Disabled state deduplication in DX11 wrapper (redundant).
- **Optimized:** (Renderer, DX11) Removed local scope array constructions (overhead).
- **Optimized:** Reduced number of branches and memory accesses in various locations.
- **Optimized:** Disabled profiling markers in device wrapper functions as they are currently not very helpful and produce significant overhead.
- **Optimized:** Fixed profile labeling to properly display wait time on render thread instead of loading system textures.
- **Optimized:** Optimized CCompiledRenderObject (sort member by access, prefetch compiled object-pointer contents).
- **Optimized:** Removed redundant locks from SProfilerThreads::PushSection.

- **Fixed:** WaterOcean displacement upload and foam-texture usage.
- **Fixed:** Missing ParticleData in pixel-shader.
- **Fixed:** Problem of zero-handle assumed to be "invalid" devbufman handle (causes nullptr access crash in GetD3D).
- **Fixed:** Missing WaterDDN in vertex-shader.
- **Fixed:** Transform gizmo (in Character Tool) is now rendered after grid instead of before.
- **Fixed:** Bug where objects that failed compilation will not update their instance matrix (resulting in missing meshes).
- **Fixed:** Bump shader cache version.
- **Fixed:** (Shader Cache) mfGetCacheItem open disk cache file if not already open.
- **Fixed:** Memory leaks.
- **Fixed:** Raw offset calculation for views when the number of elements is 1.
- **Fixed:** (Sky) Do a full update when updating for the first time.
- **Fixed:** Some bugs in ChangeOutputIfNecessary() regarding swap-chain recreation.
- **Fixed:** Let the GPU run dry before recreating swap-chains (convention is to allow deletion of held SC-resources immediately).
- **Fixed:** Uninitialized batchflags when allocating renderviews.

- **Fixed:** Statistics accumulated twice when frames stall on GPU.
- **Fixed:** Data-race when tracking multi-threaded sections (data bleed into overlaid profile-sections).
- **Fixed:** Missing gpu-timestamps to measure multi-threaded renderpasses.
- **Fixed:** Cpu/gpu-time smoothing when sections appear for a single frame only.
- **Fixed:** Texture upload cpu-cost calculation.
- **Fixed:** Frame-timing mismatch in regards to profile-sections.
- **Fixed:** (Time-smoothing) Use the same weight for sections and summaries (otherwise you could get >100% times in r_profiler 1).
- **Fixed:** Use the tighter AABB for renderitems sorted by distance (only works for temporary objects).
- **Fixed:** GBuffer-rendering called from secondary Viewports.
- **Fixed:** (Tiled Shading) Cubemaps come first, then the sun, then the rest.
- **Fixed:** Crash when resizing the display context, where the zBuffer texture during GBufferMinimal pass was being set while still invalidated.
- **Fixed:** Integration merge issues by overwriting RenderOutput with latest from main.
- **Fixed:** Shader reflection memory leak.
- **Fixed:** r_GraphicsPipelineMobile x (loop progression and sampling function).
- **Fixed:** Durango and Orbis autocompile issue.
- **Fixed:** Shader precache ignores everything but vertex and pixel shaders.
- **Fixed:** Pixel shaders precache behaviour different to other shaders.
- **Fixed:** Fullscreen fader, correct src blend factor.
- **Fixed:** Assign correct depth target and reset clear mask on CRenderOutput::EndRendering().
- **Fixed:** Allocate a temporary depth target for display context RenderOutputs that do not receive a depth target from the display context.
- **Fixed:** Cubemap rendering.
- **Fixed:** Clean up some ECGParams and fix ParticleImposter shader.
- **Fixed:** Clouds shadowgen not generating correct reprojection matrices for multi-viewport rendering.
- **Fixed:** Do not cache display context in CRenderAuxGeomD3D::PreparePass as display context can be destroyed or recreated.
- **Fixed:** SVOGI in combination with forward tiled.
- **Fixed:** CVar description of r_EnableDebugLayer.
- **Fixed:** Remove references to non-existing CVars; r_Beams, r_ShadowBlur, r_ImposterRatio. Remove non-functional CVars r_ShadowsUseClipVolume, r_shadowbluriness and hide e_ParticlesMotionBlur CVar.
- **Fixed:** Character attachments disappearing because of early IB/VB stream check before it was created.
- **Fixed:** Time-sliced updates for any dynamic light.
- **Fixed:** Temporal AA jitters.
- **Fixed:** Account for temporary depth targets when enumerating textures.
- **Fixed:** Precache more default shadersm and correctly read to in-memory shader cache.
- **Fixed:** Medium-spec assert in watermask-pass construction.
- **Fixed:** Rain.
- **Fixed:** DX12 FillDescriptorBlock() overwriting destination descriptor with each chunk. Fixed DX12 device release.
- **Fixed:** Managed resource memory leak.
- **Fixed:** Wait for particle compute jobs before compiling particle render items.
- **Fixed:** Allow dynamic growth of the particle buffer sets.
- **Fixed:** Clean up ParticleBufferSet and use frame id.
- **Fixed:** Persistend map() when substituting resources.
- **Fixed:** Wrongly applied instancing.
- **Fixed:** Streaming would never complete when generating cubemaps.
- **Fixed:** Add check for invalidated output resources to CPrimitiveRenderPass::InputChanged.
- **Fixed:** Added Vegetation_CustomRenderPassPS function that uses vegetation shader's alpha test function for proper silhouettes.
- **Fixed:** Amend previoud commit: Readd eRC_RenderScene.
- **Fixed**: Change PostEffects pipeline, re-enables silhouettes and sharpnening.
- **Fixed:** Check ALL streams needed by CRenderMesh when compiling CCompiledRenderObject.
- **Fixed:** Check streaming status of rendermesh before initiating lod transition. Check for presence of all required vertex streams before marking permanent render object compilation as complete.
- **Fixed:** Clamp refraction sampler.
- **Fixed:** clouds.cfx compilation error.
- **Fixed:** Correct value for overriding e_LodTransitionTime during cubemap generation.
- **Fixed:** Detach render context resolution from wild resize events, Fix QImage pixel format to mache requirements of renderer::ReadFrameBuffer(), set preview widget to be invisible, commented out strange'background' texture that rendered stripes over the corner of the desired image.
- **Fixed:** Disable lod dissolve for cubemap generation. Clean up CVar overrides for cubemap generation.
- **Fixed:** Fix clouds.cfx compilation errors for all rt combinations.
- **Fixed:** Fix deadlock due to nested message processing from DXGI in CRenderDisplayContext::SetFullscreenState.
- **Fixed:** Fix edge case in shader cache generation where removal of duplicate shaders could end up referencing the wrong shader.
- **Fixed:** Fixed ref-counting of used depth-stencil/render-target resources.
- **Fixed:** Fixed reutilization of named temporary texture-resources.
- **Fixed:** Fixed screen-space reflections for Illum.
- **Fixed:** Fixed structured buffers being cleared with INT/FLOAT, despite not being possible.
- **Fixed:** Fixed zprepass/dissolve mismatch between hunt's zpass and main's zprepass.
- **Fixed:** Fixes Release compilation when ENABLE_RENDER_AUX_GEOM is not defined.
- **Fixed:** Fixes vulkan shader cache generation. The encoded resource layout size is reduced by half. It is base64 encoded and sent with request line to remote shader compiler. This way the resource layout description which is needed for Vulkan shader compilation is known during shader cache generation.
- **Fixed:** Fix extension removal from shader file name in CShaderMan::mfReloadFile.
- **Fixed:** Fix forward minimal emissive rendering.
- **Fixed:** Fix input dirty checks on various primitive passes.
- **Fixed:** Fix logic error preventing cubemap generation tasks from clearing from the task schedule.
- **Fixed:** Fix minimal forward emissive objects blending with background.
- **Fixed:** Fix refractive brushes.
- **Fixed:** Fix scenepass profile labels.
- **Fixed:** Fix sorting order for permanent render objects (fixes transparent forward pass).
- **Fixed:** Fix UIFlashElement logic.
- **Fixed:** Fix water shader recursive rendering initiation.

- **Fixed:** Initialize user folder shader cache during startup to avoid runtime stalls due to file IO.
- **Fixed:** Inject partial resolve passes between refractive render items.
- **Fixed:** Integrate Marcel's GlassHunt changes to CE.
- **Fixed:** Keep a shared_ptr to flash player instances and enable flash update during level load.
- **Fixed:** Make sure to remove list items from CRenderMesh::s_MeshModified list during update to avoid double delete when render mesh is destroyed.
- **Fixed:** Precache level shaders and generate hardware instances during precache.
- **Fixed:** Release CRenderDisplayContext resources correctly.
- **Fixed:** Remove #pragma optimize("", off).
- **Fixed:** Remove global CRendererResources::s_ptexPrevBackBuffer and let PostAA stage handle its own resources.
- **Fixed:** Remove hidden dependency of LDR PostAA film grain on the (Dep) Min exposure and (Dep)Max exposure parameters, which, in certain common combinations, would render grain invisible. Now PostAA grain is controlled solely by the grainAmount parameter, reducing confusion and marginally improving performance. The previous behavior can be re-enabled for legacy content support by turning on the cvar r_GrainEnableExposureThreshold.
- **Fixed:** Resolve pass: Ensure we don't copy out of bounds.
- **Fixed:** Restore original cryengine batch flag behaviour.
- **Fixed:** Shader parser: Add annotations to textures and remove old samplers parsing pipeline.
- **Fixed:** Streamed CGF meshes now correctly analyze foliage skinning data on loadup.
- **Fixed:** Sunshafts broken when sys_spec < High.
- **Fixed:** Wait for streaming jobs to finish when generating cubemaps.
- Fixed: D12 device removal due to slightly mismatched texture copy targets.
- **Tweaked:** Rename m_RenderTargetCount for consistency.
- **Tweaked:** (Shader Cache) Check lookup data on cache activation.
- **Tweaked:** Set clear value of initial exposure to 1.0f (neutral) so as to not mess with auto-eye-adaption in the first few frames.
- **Tweaked:** (RenderView) Small clean up.
- **Tweaked:** Remove unused global textures.
- **Tweaked:** Draw resolve passes debug rectangles.
- **Tweaked:** Added multi-threaded timings to r_profiler=2 (non-additive, for inspection).
- **Tweaked:** Add dissolve to glass shader.
- **Tweaked:** DrawToCommandList() takes a commandlist explictly.
- **Tweaked:** Fix GPUParticles compile error.
- **Tweaked:** Slightly better scope labels for SRenderThread commands.
- **Tweaked:** CRenderPassBase::InputChanged() is now a variadic template and can accept any trivial data.
- **Tweaked:** Adjusted MAX_CLIPVOLUMES constant to be in sync with the amount defined in engine code.
- **Tweaked:** (Android) Reinstate android trybuilds, following some fixes.
- **Tweaked:** Added functionality to forward transparent pass to write depth / velocity for hunt weapon scopes.
- **Tweaked:** Disable asynchronous state check when not enabled.
- **Tweaked:** DX12: omit state tracking on graphics queue when D3D11_COPY_NO_OVERWRITE_CONC is set.
- **Tweaked:** Improved water volume reflections quality.
- **Tweaked:** Log mfPrecacheAllCombinations() requests.
- **Tweaked:** Omitted windows code for XBox.
- **Tweaked:** Optimized SRendItem.
- **Tweaked:** Rename InputChanged to IsDirty and remove uneeded checks.
- **Tweaked:** Safety check for potentially empty vector. The Release compilation does not crash, but the Debug will raise asserts.
- **Tweaked:** Use downscaled source and targets for sunshafts, depending on sys_spec.

Vulkan

- **New:** Annotating HLSL shader to include resource layout descriptors.
- **New:** Local shader compilation in Windows platform.
- **New:** Enable Vulkan renderer for the Sandbox Editor.
  NOTE: This is just to enable it and currently it cannot be guaranteed that it works.
- **New:** DX11/12/Vulkan Depth Bounds.
- **Refactored:** Support for DXC and GLSLANG HLSL to SPIRV compilers and compiler switching feature.
- **Refactored:** Support for using new SPIRV-Cross.
- **Refactored:** Adjustments to support shader cache generation and remote shader compiler with the new shader compiler.
- **Refactored:** Changing the swap chain's color buffer format from RGBA to BGRA (Apparently validation errors were triggering because of RGBA formats on both NVIDIA and AMD).
- **Refactored:** SVOGI's full resolution kernels are now utilized for Vulkan since this is no longer a limitation.
- **Refactored:** Registering encoded layout for a resource set only - when it is not registered already.
- **Fixed:** Not treating warnings as an error for SPIRV-cross.
- **Fixed:** Fixes Android crash due to null output.
- **Fixed:** DXC shader compilation errors which fail to provide an implicit cast between two user defined structures, and handling samplerCUBE.
- **Fixed:** Reverting back the image layout after copying the image.
- **Tweaked:** Compile Vulkan renderer on Linux using gcc. (disabled Vulkan compilation on Linux).

3D Engine

- **Optimized:** Per-objects shadow maps ported to one-pass scene graph traversal.
- **Optimized:** Recalculated entity clip volume only when entity position changed. Removed list of registered rendernodes from CClipVolume.
- **Fixed:** Do not defer rendernode deletion on level unload and shutdown.
- **Fixed:** Clip volumes update for render nodes.
- **Fixed:** Clip volumes for fog volumes
- **Fixed:** Rendering multiple Viewports invalidating all persistent render-node data (by switching render-resolution).
- **Fixed:** MergedMeshes properly cleaning up tempData during PostRender.

Particles

- **New:** Enhanced grouping capabilities in the new ActivateRandom feature; now supports all possibilities of old SecondGen features.
- **New:** Replaced SecondGen features (on parents) with child features (on children, who decide when they activate). Replaced SecondGen 'Random' parameters with new ComponentActivateRandom feature. Added AddSubInstances and CullSubInstances features, removed KillParticles feature. Parent relationship is now explicitly serialized. Added Component-to-Component node linking in Particle Editor.
- **New:**Components do not need any feature to be able to work properly - adding default size, time and spawn features.
- **New:** Added modifier Global access and added level time into it.
- **New:** Added TimeOfDay and ExposureValue global domains.
- **New:** GPU particle counts.
- **New:** Added FeatureAnglesAlign World mode. Simplified AlignType construction.
- **New:** Added feature light lens flare.
- **New:** Refactored pfx1 and pfx2 stats to inherit from NumberVector - allows averaging over frames with float and int versions. pfx1 stats now display below pfx2 stats in the same format. Added emitter stats to pfx1 along with Component. Restored CollectEffectStats function with sorting.
- **Refactored:** ParticleDataType is now type-templated - provides more type-safety and easier interface with Container Streams; now supports Vec3 etc. directly rather than float3. Streams now have [] operators. Vec3 and Quat streams are template specializations. Moving from specialized stream names to generic TIStreams and TIOStreams. Simplified ParamMod Init and Update functions.
- **Refactored:** Keep Features in .cpp only files. Refactored default and exclusive feature implementations. Duplicate features are auto-disabled.
- **Refactored:** Renamed modifiers Time Source to Domain.
- **Refactored:** Replace iteration macros with range-based loops.
- **Refactored:** Standardized implementation of params which have infinite/disabled state: edited with an enable toggle, otherwise serialized with possible infinite value. Refactored TValue<TTraits> templates, unifying limits and conversion. Removed soft limits for now as they are not supported by PropertyTree.
- **Refactored:** Restored proper unit scale for Lighting Diffuse value.
- **Refactored:** Infinite values now displayed as 'Infinity' in the Sandbox Editor. Moved version fix code to VisibilityParams. Compile fixes.
- **Refactored:** Combined all Life features into one compute Lifetime and InvLifetime.
- **Refactored:** Scheduling optimizations: emitters rendered for the first time after being unseen now get a priority boost to their update jobs. Emitters with DeferredRender components (meshes, decals, etc) get a further boost. DeferredRender jobs are now grouped by emitter, not component. Removed e_ParticlesThread=1 mode, moved modes 2 through 4 down a slot, mode 4 has new render priorities. Added stats for components delayed on syncing.

- **Optimized:** Vectorized stream Fill functions. Reversed inheritance of IStream and IOStream. Removed not needed members from IOStream.
- **Optimized:** STempInitBuffer simplifies data initialization, only allocates data for spawned particles.
- **Optimized:** More DeferredRender scheduling optimisation.
- **Optimized:** Removed unnecessary SUpdateContext, replaced it and other signatures consistently with CParticleComponentRuntime and added convenience functions to CParticleComponentRuntime. (Makes it easier to unify particle and subemitter data in next step.)
- **Optimized:** Use EDataDomain flags for both ParticleDataTypes and IParamModContext, eliminating IParamModContext struct, and several virtual functions.

- **Fixed:** Emitters not activating properly (muzzle flashes etc.).
- **Fixed:** Editing effects again updates existing emitters; Sandbox Editor uses standard particle system to load effects instead of creating a duplicate.
- **Fixed:** Properly unregister emitters when resetting them during sys_spec change.
- **Fixed:** Crash when editing effects destroyed child hierarchy.
- **Fixed:** Crash when adding new Components to running effect.
- **Fixed:** Crash on adding invalid child feature to emitter features. Refactored feature registration to remove invalid features from emitter features list.
- **Fixed:** Issues in pfx1 conversion. Fixed TValue serialization to remove improper zeroing of not-found values (caused ViewDistanceMultiplier to always be 0); updated for new Child features; added default SortMode; reduced intermediate XML attributes.
- **Fixed:** Default Sprite Component had all lighting values = 0.
- **Fixed:** Several errors involving Emitter Features. Altered effect is now completely cloned for proper parenting. Component.SetParent cleans up children. Clearing Emitter Features now updates emitter. Emitter Features no longer flagged as changed when empty.
- **Fixed:** Emitter activate issues. Parent particle now added only once.
- **Fixed:** Broken Audio Component.
- **Fixed:** pfx1 conversion issues. AudioSource, LifeImmortal and added node positions.
- **Fixed:** Array overflow in GetExtents.
- **Fixed:** CParamMod now correctly converts values during serialization.
- **Fixed:** Invalid data stream initialization.
- **Fixed:** pfx1 conversion fixes. Fixed emission direction. Don't replace already-loaded effects. Don't create inactive effects. Refactored some interfaces.
- **Fixed:** Orientation fix for placed pfx1-converted emitters.
- **Fixed:** Restored function of e_ParticlesMaxScreenFill.
- **Fixed:** Enforce strict priorities on VR plugins.
- **Fixed:** Fixed inconsistent entity code, unhiding emitters works properly.
- **Fixed:** Added null-pointer check to CharInstance.GetRandomPoints
- **Fixed:** Rollback previous fix for unhiding emitters. New fix ensures EntitySlot.bRegisteredRenderNode is consistently set.
- **Fixed:** Computed bounds now use proper particle size for meshes and lights. Removed separate ComputeBounds feature. Tweaked bounds pdate code. Component camera culling is now done in ComputeVertices, with current bounds.
- **Fixed:** Emitters with instant immortal particles never updated, due to incorrect "stable" test.
- **Fixed:** Fixed crash, added more type safety to TParticleId and TParticleGroupId.
- **Fixed:** Fixed invalid float values in padded particle data: CopyData now copies padded data, age values are clamped to non-zero.
- **Fixed:** Imposed more collision iteration limits on particles, fixing stalls.
- **Fixed:** Made RenderMeshes piece selection, positioning, and scaling more consistent.

VR

- **Fixed:** Wrong depth texture used for Sky pass.
- **Fixed:** Editor throws exception when renaming a variable.

C#.Core

- **Fixed:** C# solution not being generated for non-C# projects in the Sandbox Editor (when the first C# file is added).
- **Fixed:** Modifying files is triggering multiple compilation runs in the Sandbox Editor.
- **Fixed:** Crash when releasing textures that are still in use.
- **Fixed:** Resizing the canvas of the C# UI on an entity not resizing the UI as well.
- **Tweaked:** Changed the name of the C# project generated for C# assets - so that it doesn't conflict with the project of managed plugins.

Physics

- **Fixed:** Initialize m_sweepGap for bodies activated mid-step.

Lobby

- **Fixed:** To assert() instead of crash with access violation.
- **Fixed:** Bounding boxes of render node objects are oversized.
- **Fixed:** Substance Instance Editor still shows information even if **File -> Close** was used.

Editor General

- **Fixed:** Removing a Component while several entities are selected only works for one of them.
- **Fixed:** Input is buggy when Sandbox Editor is launched via Renderdoc.
- **Fixed:** Low performance Physics/AI mode issue. The DoFrame was called with the wrong argument which was causing continuous resizing of textures.
- **Fixed:** Renaming group crashes the Sandbox Editor if the Flow Graph Tool is open.
- **Fixed:** Crash in obtaining help info in python module after VS 2017.7 compilation. Autocomplete data is generated by pythoneditor. generate_pythoneditor_autocomplete_files.
- **Fixed:** .level.cryasset file is not generated after upgrading levels from .cry to .level.
- **Fixed:** Console Command Tooltips for functions with params get description from arbitrary place of memory (sometimes it may crash the Sandbox Editor).

Tools General

- **Tweaked:** Transfer unsmoothed cpu/gpu times (statoscope has built-in smoothing).
- **Tweaked:** Return gpu-timestamps instead of timestamp-indices (indices are not useful).

Resource Compiler

- **Tweaked:** Make default position format the exporter once (repeatedly calling RC will produce identical CGF's).
- **New:** Added CVar to reduce the range that text in e_debugdraw is displayed on screen.
- **New:** (Renderer, DX11) Added DX11 device, fences, scheduler and heap.
- **New:** Enable true texture streaming for texture based lights.
- **New:** Expose CVar to disable clip volumes.
- **Refactored:** (Renderer General) (Shadows) Shadow CVar clean up.
- **Reafcatored:** Removed non functional r_ShadowsDepthBoundNV, r_UseShadowsPool, r_ShadowsMaskDownScale, r_ShadowsStencilPrePass, e_ShadowsMasksLimit, e_ShadowsConstBiasHQ and e_ShadowsSlopeBiasHQ.
- **Refactored:** Hide r_ShadowsMaskResolution as this is currently not supported (very likely to come back).
- **Refactored:** Renamed r_ShadowGen to r_ShadowMapsUpdate.
- **Refactored:** Extend e_Shadows and r_ShadowsMask to allow selective switching off of shadows for sun or point lights.
- **Refactored:** Using SDisplayContextKey instead of HWND (platform dependent) in ISystem::DoFrame and ISystem::RenderBegin.
- **Refactored:** (Renderer, XBox) Prevent access to DXGI 1.4 header.
- **Refactored:** (Renderer, Durango) Fixed case mismatch in variable name after merge.
- **Refactored:** (Renderer, Water) Skip WaterMask stage PSO creation when not active (sys-spec dependent).
- **Optimized:** Minimize memory utilization of textures used by lights.
- **Optimized:** (Streaming) Respect minimum-upload size CVar for texture streaming.
- **Fixed:** Cubemap race condition.
- **Fixed:** cpp check 'selfinitialization' errors.
- **Fixed:** Texture streaming rework.
- **Fixed:** Crash - >0 comparison of UInt32 causing GetOrCreateRasterState() failure and crash.
- **Fixed:** Pass SSReflHalfres info into shader to fix bug reading from full-res depth buffer with half res target.

- **Fixed:** Turn off rendering of helpers and aux. geo. temporarily when rendering cubemaps.
- **Fixed:** (Streaming) Fixed unclamped texture transfer function parameters (validation error).
- **Fixed:** Not enough occlusion if offline voxelization for GI is used.
- **Fixed:** Prevent concurrent device access due to renderview transition on main thread. Make sure all members are properly reset in CRenderView::Clear(). Remove unused SRenderingPassInfo::pJobState.
- **Fixed:** Pass screensize directly to BlendWeightSMAA_PS functions - rather than relying on side-effect in global PS_ScreenSize which was being modified illegally w/out dx9 compatibility mode in main PS function.
- **Fixed:** Replace dysfunctional legacy downsample with StretchRegion operation, fixing CD3D9Renderer::CaptureFrameBufferfast() and thus Statoscope screenshots.
- **Fixed:** Re-enable resizing code in 2D Viewport.
- **Fixed:** Remove inter-parameter value checks that break the UI. Force users to set the oscillator params before setting moveType.
- **Fixed:** Prevent gamma 'snap back' on quit.
- **Fixed:** Temporarily resize render Viewport during sequence capture to match requested output resolution.
- **Fixed:** Fix random crash on exit, replace with an assert.
- **Fixed:** Protect CREWaterVolume against crash (due to geometry-less water volumes in the Airfield level).
- **Fixed:** Many profiler issues. Functions from all threads are now properly combined and display the thread count. Replaced history bins with running averages for profiler self time, total time and call counts; they now all properly compute average, min, max and variance based on profile_smooth time.
- **Fixed:** Preserve all 64 mask bits in MaterialProperties SetFlag() function.
- **Fixed:** Force cubemap TIFF alpha to 1.0.
- **Fixed:** Crash follows assert when painting a terrain layer with proc vegetation.
- **Fixed:** Shadows differ between e_OnePassOctreeTraversal 0/1.
- **Fixed:** (Renderer, General) Added CustomRenderPass to vegetation shader to support silhouette highlighting.
- **Fixed:** (Renderer, DX12) Drop TEARING flag when TEST flag is used on Present() (incompatible flags).
- **Fixed:** (Renderer, DX12) Possible objects-leak when root-signatures fail to compile.
- **Fixed:** (Renderer, 3DHUD) Removed jitter from HUD positioning when AA is used.
- **Fixed:** (Renderer, Caustics) Moved light-volume calculation before caustics (caustics need it).
- **Fixed:** (Renderer, Vulkan) Resource-assignment when substituting resources.
- **Fixed:** (Renderer, DX12) Time-stamp issue (wrong command list).
- **Fixed:** Update shadowmap stage before CTiledLightVolumesStage::GenerateLightList() to ensure shadow matrices are up to date when light list is generated.
- **Tweaked:** (Build) Bootstrap Scaleform revision update (legacy libs added for XBox compilation).
- **Tweaked:** Improved e_debugdraw 1 mode readability.
- **Tweaked:** (3DRenderAuxGeom) Remove default parameter value.
- **Tweaked:** (Streaming) Remove ulp based stream-out request and allow requesting minimum number of mips (< vs. <=).
- **Tweaked:** Rename late camera injection example's class.
- **Tweaked:** (Renderer, Flash) Render flash dyntextures before rendering (not after).
- **Tweaked:** (Renderer, Vulkan) Execute TickDestruction() before GarbeCollect() (views hold ref-counts on resources).
- **Tweaked:** (Renderer, DX11) Add NO_OVERWRITE mode to constant-buffer updates when available under 11.0/1.
- **Tweaked:** (Shaders) Broken instancing workaround.
- **Tweaked:** (Renderer, XBox, DX12) Make DX12 compile with XDK.