Unreal Patch v227j Full Changelog

.Content index

.Version 227j

.TL;DR version

- Audio:
  - Tons of fixes to OpenAL and SwFMODEx.
  - Added Speech Volume in order to control the volume of monster grunts, RTNP Marines and custom dialogue voicelines.
- Editor:
  - New Hint System which uses the comments of the UScript in order to display what certain functions do.
  - New switch in View Menu to enable/disable lighticon in lightcolor.
  - Now it's possible to build very large and complex spheres without the Editor crashing!
  - SkeletalMeshes can now be copypasted.
  - Actors can be copy+pasted in selected position via RMB menu.
  - ucc packageflag doesn't generate corrupt .unr files anymore.
  - Brush Scaling and Brush Stretching were extended to Actor Scaling and Actor Stretching.
  - With RMB the builder brush can transform the current shape into a dynamic zone.
  - New RMB menu option to replace the actor while keeping the same modifier properties.
  - Moved all hidden (uncategorized) properties to a "Hidden properties" category in properties window, also color coded to red.
  - Added an option (enabled by default) to make rotation angles show in radii angles (0-360 range) rather then UU angles (0-2^16 range). Setting is at: UseRegularAngles=False under [Core.System].
  - Maps that have missing packages are now allowed to be partially loaded (they will still give a map load warning at first and ask Y/N if you want to load them anyway).
  - Added a portal directional arrow for WarpZoneInfo to help identify what direction the portal is currently facing at.
  - Added Actor.bBlockAISight which makes AI not be able to see through the actor (must have bCollideActors enabled but can be non-blocking).
  - Added support for custom actors to be placed now with a Brush collision shape attached to it (Actor must have bSpecialBrushActor enabled).
  - Added a 'Play from here' right-click menu option to editor 3D viewport.
  - Added right-click menu for properties window in order to allow user to copy variables information to clipboard.
  - Made UnrealEd track modified packages and ask if each one of them should be saved on exit.
  - Made non-compiled script classes non-placeable in editor.
  - Added Tooltip for object classes to show header comments of the class you're hovering over.
  - Added functionality to rename asset groups too (texture/sound/mesh etc).

- Added right-click menu for sound browser.
- Added a minimum properties window size to avoid 1 pixel size properties window bug.
- Disabled in-game keybindings from firing inside editor.
- Fixed editor from losing focus when closing a sub-window after using alt-tab (win 10).
- Made sounds not reset nor music stop when you paste actors in editor.
- Added a song section slider to Music Browser so you can listen to specific sections of the song.
- Added support for custom editor keybindings (see Unreal.ini, EditorEngine.KeyBindings).
- Added right-click BSP surface options to Merge faces (on a tesselated surface) and Flip faces (incase they are facing the wrong direction).
- Added editor option to disable saving of LevelSummary object (if you want save map as pre-226 compatibility, found in Advanced Options).
- Made editor save all modified packages to System/Backup folder if editor crashes.
- Added support for importing 2-bit or 4-bit PCX textures.
- Added LiftExit variant LiftJumpDest which tells AI to lift-jump to an area.
- Made editor "Show paths" show exact path size values of routes with selected path nodes.
- Implemented OBJ Brush exporter/importer.
- Made DumpIntCommandlet merge changes with all languages.
- Tons of other minor fixes and additions.

- Engine:
  - Made player footsteps cause wave ripplets on FluidSurfaceInfo as you walk in it.
  - Due to complaints about lighting differences in 227, especially because of the feedback from people doing some conversion from UT maps, added a new switch which enables SpecularLighting effect for all maps unless a specific new flag is set: var() bool bSpecularLight;. This means that 227 mappers would want to disable this flag to have the known and more accurate 227 lighting, but old and already ported maps have the 226/UT behavior. Maybe should have done from the begin with, because this means now more work for 227 mappers, especially for already finished maps, but this allows everyone to take the preferred method after all.
  - Added triggered MaterialSequence animation.
  - Fixed some timer issue for windows version, depending on appSeconds(), which causes mostly a problem for servers in relation with MaxTickRate. It seems that only a few systems are affected by this problem (maybe depending on hardware/bios). This bug is existing in all versions before, including 224,225 and 226.
  - Added path builder information with customizable variables to LevelInfo to allow mappers building maps with different settings or for other UEngine games with UED2.1.
  - Fixed: Server doesn't inform clients about closing old connections on map change.
  - Fixed: Bug in banning system may cause duplicate (false) entries.

- Added delta compression for saved games, also made saved games files contain save information along with a screenshot preview to show on load game menu.
- Added commands "STAT THREAD" to display number of running threads and "THREADS" to display a list and description of running threads in memory (for Unreal).
- Added Sound compression/decompression support for all audio drivers. You can compress sounds in sound browser or in UnrealScript by defining "OGG=X" (X is the quality in range of 0-11, 0 = worst, 11 = best) in #exec audio import line or then you can directly import ogg files as sounds.
- Added support for defining HD HUD textures, by either setting Texture.HDTexture or in UnrealScript add "HD=XXX" to #exec texture/font import lines. HD texture will be used if texture is being stretched out by at least +25% of its original size.
- Made it possible to enter spaces in names in properties and defaultproperties blocks.
- Added ZoneInfo.MinWalkableZ to tell physics engine how steep slopes Pawns can walk on.
- Implemented Alt+F4 hotkey to exit the game instantly.
- Improved Log window to support command history browsing (up/down arrows) also properly support selecting/replacing parts of text or insert text in middle of command.
- Added support for Sub-Levels architecture (where you can link multiple maps together into one with seamless level transition). See LevelInfo.SubLevels array.
- Added GrassDeco actor for a Source Engine style grass sprites in levels (also GrassDecoSchool for an automatic grass patch).
- Added support for custom mouse cursors in-game (directly use regular Textures), see Engine.UnrealCursor class.
- Disable initial garbage collection when booting up the game (waste of time).
- Made it impossible to try to load package 'nul' to prevent malicious loading crashes.
- Added Volumes support from UnrealEngine 2+ (which can be used to create brush shape collision shapes or triggers to levels).
- Made LadderTrigger a Volume instead, also various fixes to ladder:
- Player properly hops off the ladder now at top.
- Added a flag to allow player to strafe on ladder (you must have bDirectional enabled).
- Added a flag to enable/disable directional ladder which makes you drop off the ladder if you look away from it (must also have bDirectional enabled).
- Pressing jump key while holding backwards now makes you now jump off ladder backwards.
- Added a value to control climbing speed on the ladder.
- Added INI file option 'LangPath' which can be used to specify directory of language files (like .int).

- Added a variable to SkyZoneInfo which allows you to make sky move in relative to player camera position.
- Added support for static light static mesh movers.
- Added ClientMover actor for a client-side background looping movers (which doesn't interact with gameplay in any way).
- Added TransporterVolume which can be used to transport all actors touching that volume to a relative offset in another place in level. Similar to Transporter but with better control.
- Disabled Password logging by default game modes.
- Fixed INI file writer failing if trying to write really long ini lines.
- Added PhysX physics support (with Physics=PHYS_RigidBody and PhysicsData reference in actor).
- Added a Pawn PhysicsAnimation object which can be used to give pawns automatic physics based animations.
- Made package loader load meshes as any mesh type (regular Mesh can now load as LodMesh/SkeletalMesh/StaticMesh etc...).
- Added a new BSP surface flag 'invisible occluder' which works like an anti-portal surface to simply occlude view.
- Fixed players unable to walk inside warpzones.
- Added option to MusicEvent to disable cross-level song changes (when using sub-levels).
- Added PhysX option for Emitter particles to make them rigid.
- Added particle fade style option (to make particles change style only while fading).
- Added LevelInfo.MaxPortalDepth to control how many layers warpzones/mirrors/skyboxes can draw.
- Made UnknownXX keybindings not configurable, and therefore wont show up in User.ini.
- Added consolecommand "DUMPSTATES <classname>" to dump state execution information of all matching actors currently in level.
- Fixed Advanced Options window to correctly handle sub-section naming conflicts (such as <Advanced Options - Advance> vs <Advanced Options - Editor - Advance>).
- Added a LevelInfo flag bUTZoneVelocity to make ZoneVelocity behave like in UT games (defaults to True on levels saved in UT editor).
- Added GameInfo.GameMaxChannels which controls the max number of channels 227j+ clients can have on the server (valid range is 256-65536).
- Added an INI/INT formatting for color codes, works with following formatting: (#HTML color tag), as example: (#FFFF00) = yellow color.
- Made string properties with some special characters gets written with a special text tag into INI (tags are \%HEX or \%n or \%").
- Fixed a memory error when a Mesh or LodMesh has a surface with a vertex index that goes out of range.
- Also added a Teleporter.bUTRotationMode flag which is set to true also when loading UT maps. To make teleporters behave exactly like in UT.
- Made ExtremeDGen and EndGame usable in network servers.
- Fixed game from crashing if you delete a navigationpoint that is referenced by a reachspec.

    - Made game de-reference deleted lights and zoneinfos on map load.
    - Fixed loading of some UT maps that have incorrect LightEffect=LE_Sunlight reference, making level oddly bright (CTF-November).
    - Implemented a hack fix for RedeemYourSpace map pack, to force player to touch any teleporters under their MaxStepHeight while standing on a mover moving downwards.
    - Added features for JumpPads: a flag to disable it for monsters or players, made it trigger an event when used and triggering it will toggle if its enabled or disabled.
    - Added support for multiple language paths.
    - Added to LevelInfo flags to disable level specific rigidbody physics (bDisableRbPhysics/bDisableSubLevelRbPhys - if you know you wont need it).
    - Fixed so players never spawn as DemoRecSpectator when you load game from a save in which you've demo recorded on.
    - Made so game/editor doesn't crash on boot if splash screen texture is missing.
    - Fixed 'First time startup wizard' menu to have functional back button again.
- UI:
    - New languages: Portuguese, Catalan, Dutch.
    - Overall improvements and updates to German, French, Spanish, Italian, Russian and Polish.
    - All languages updated with the latest strings and their problems fixed. Any inconsistency, problem, etc. should be reported here, with pull-requests if possible: https://github.com/NeonKnightOA/unreal-localization
    - Added "stat video" command to enable DrawStats(Frame) in RenDev.
    - UMenu Preferences -> Video:
    - Changed video driver selection shows all installed drivers now (select one and press "Restart").
    - Added checkbox for fullscreen/window mode.
    - Added checkbox for trilinear buffering.
    - Added checkbox for precaching.
    - UMenu Preferences -> Audio: Changed audio driver selection shows all installed drivers now (select one and press "Restart").
    - UMenu Preferences -> Game Settings: Added language selection for the main localization of the game (select one and press "Restart").
    - UMenu Load/Save Menu: Fixed localization support.
    - UMenu Botmatch/Multiplayer:
    - Reorganized the entries in the game type combobox in botmatch and new network game dialog.
    - Added settings and rules tab for coop game type.
    - UMenu Preferences -> HUD: Added subtitles fontsize combobox.
    - Classic Menu:
    - Added all new difficulty settings.
    - Added coop settings in botmatch/multiplayer menu.
    - Added support for advanced options localization without breaking the tree.

    - Made weapons menu always show custom weapons.
- Gameplay:
    - Fixed: disappearing Slith carcass bug (slith carcass in corrosive zones, such as slimezone).
    - Fixed: RTNP SpaceMarines, when a SpaceMarine leaves a spawn point on level Crashsite2 prematurely, bad things may happen: he may have blue aura, wrong fatness and mass or be completely invisible, his weapon may have wrong fatness too.
    - Fixed: Problems related to MarineMatch and SpaceMarines.
    - MarineMatch game immediately ends when TimeLimit is greater than zero.
    - MarineMatch doesn't modify difficulty level according to the bot config.
    - Gender of SpaceMarines is not properly assigned.
    - Made save game menu support unlimited slots.
    - Added option to LevelInfo/bEnhancedIcePhysics, that will fix walking speed on ice.
    - Fixed playerpawn viewbob to cap view bobbing if player is moving at very high speeds.
    - Fixed ZoneVelocity behave like in Unreal v 230+ versions, with an exception if ZoneVelocity.Z is non-zero it will null out zone gravity (legacy map support, Zora's ZM3WaterBridge lava jump).
    - Fixed ParentBlob to be able to obtain new enemy once old enemy is dead or has disappeared out of sight.
    - Made some decos, inventory and fragments orient to floor direction when landing.
    - Made package loader keep loading packages even if a single package was missing to throw full list of missing packages when finished trying.
    - Added SkeletalMesh/LodMesh load-time error checking in case of broken mesh.
    - Fixed Movers with StopOnEncroach to network stopped movement to 227j clients.
    - Fixed Movers with ContantLoop state to properly handle Stop/ReturnOnEncroach types.
    - Fixed orientation errors when passing through WarpZones with different orientations (gimbal lock error).
    - Made weapons force stop firing while entering feign death mode.
    - Made Pawns walking along ledges a little less janky.
    - Made user consolecommand history get saved in User.ini to be reusable on next session again.
    - Made UWindow combo boxes auto-scroll to currently selected item when opened.
    - Added to MusicMenu a shuffle playlist checkbox.
    - Added a button to MusicMenu to add every music found in music folder.
    - Fixed NaliRabbits/BiterFish pick destination to not pick world origin.
    - Made UMenu PlayerSetup/Weapon menu downscale view model by window height to properly fit model on screen in widescreen modes.
    - Made UMenu warn user if you try to play singleplayer with bonus 227 difficulty levels.

- Made bonus difficulty levels show in red on classic menu.
- Changed mechanics of bonus difficulty levels to not increase health of monsters, but to alter their AI behaviour.
- Fixed an ScriptWarning when Weapon InstantFire ends up killing player owner.
- Fixed some mismatching font characters on LargeFont and BigFont (? and + characters being inverted).
- Made Translucent and Modulated decorations not cast shadows.
- Added an option to run game with "classic balance" so gameplay behaves exactly like in pre-227 (pupaes can't attack upwards, projectiles spread works like before and new 227 difficulty levels work like in 227i version).
- Fixed Ultra shadow detail mode option not working on UMenu.
- Set a default playerclass to UBrowser to use in-case user reset their config and try to join a server without visiting their playersetup.
- Added more options to Decoration > Cannon to have more useful options for mappers to use.
- Fixed a bug where if you stood in a knee-high painzone and walked into a wall it would instantly kill you because it would constantly spam you entering the zone every frame.
- Fixed RTNP not ending intro maps correctly if they were translated to a different map title.
- Fixed SCUBA gear from remaining active after mapchange.
- Added option to run realtime shadows on single core.
- Made UMenu level preview info loader fallback to read mapinfo from int files.
- Added new HD icons and textures from Krull0r and Sly.
- Added a secret mirror mode gamemode.
- Added support for face skin selection in classic player select menu.
- Added Mutator selection menu for classic menu.
- Fixed game window not always starting focused.
- Fixed game window moving mouse cursor to center of screen even if it wasn't in focus.

- Render:
  - Added XOpenGL OpenGL3 / OpenGL4 renderer. Note: You need to have at least an OpenGL 3.3 capable card to use this renderer!
  - Fixed: Coronas being blocked by BlockAll (ignores bHidden).
  - Added a catch for crippled meshes could crash renderer (d3d9/opengl) when using hwclipping.
  - New Quadshot firstperson model and animations.
  - Fixed Software render to draw projectors and trail emitters.
  - Changed Software render to draw alpha blended textures as masked (masking invisible parts).
  - Fixed static meshes (or any other bShadowCast actors) to cast shadows over dynamic actors.
  - Added PortalModifier object which can be assigned to Texture (to be applied when drawing portal through it), ZoneInfo (to be applied when camera is inside it), or PlayerPawn (for mod override), which can be used to replace several rendering factors, such as disable lighting/fog/modify

distance fog or replace textures etc. (see Object > PortalModifier).
- Fixed Direct3D9 flickering with BSP and distance fog.
- Added support for hardware clipping planes to D3D9 and OpenGL.
- Made actors that pass through zones with different zone ambient light, fade between the zone light colors rather then instantly swap them.
- Made mesh actors that have their origin inside level geometry to not turn unlit and still accept lighting from nearby light sources.
- Added option to upscale HUD (HUD.HudScaler) to make canvas draw HUD as it were in low res but still keep high in-game resolution.
- Fixed coronas and sprites to fade out from distance fog.
- Optimized render by disabling surface sorting when not needed.
- Fixed a bug where skybox would turn into a HOM when viewed through a mirror reflection.
- Made windows mouse pointer usable in-game on fullscreen mode (mainly for UMenu).
- Added support for DynamicCorona to have directional light ray texture (which is only visible from the sides of the corona).
- Added custom LensFlares option for DynamicCoronas.
- Fixed ZoneInfo.Min/MaxLightcount to be functional again.
- Fixed Armor mesh missing bottom polygons and adjusted several LOD meshes to have less agressive LOD. (Thanks Krull0r)
- Added various HD icons for when you use hi-res HUD mode. (Thanks Krull0r)
- Fixed zoneportals from hiding skybox in additive maps.
- Fixed so render supports unlimited screen res Y size (was limited to 2880).
- Fixed two sided meshes to receive lighting from both sides again.
- Made volumetric fogs draw on zoneportals and sky (sky fog one is optional and can be changed from video settings between high quality/low quality/disabled options).
- Made game draw volumetric fogs in fogzones even if player is standing outside of any fogzones.
- Made volumetric fogging to be rendered on unlit actors.
- Made map projectors properly project on any actors it touches (if bOnlyAttachStaticActors is False).
- Made projectors project on first person weapon mesh too if it projects on player model itself.
- Made shadow bitmaps ignore translucent mesh surfaces so they don't cast shadows (so pawn muzzle flashes dont draw ridiculous shadows).
- Made shadow bitmaps render as STY_Modulated for Software render device (or any other render devices that don't support alpha blending).
- Fixed a render glitch with regular decals and made them functional on Software render device.
- Made OpenGL and D3D9 reset texture cache for an texture if their format or resolution changes but their CacheID remains the same (instead of corrupting the texture).
- Fixed meshes disappearing on mirror reflections on a lot of cases.
- Added a separate MeshDetailTextures option for render drivers to

disable/enable detail textures on meshes.
    - Made Emitter particles reset their lighting information whenever particles respawn (so mesh emitter particles don't fade lighting from previous position particle).
    - Corrected blob shadow scale for RocketCans and RazorBlades ammo.
    - Fixed so that invisible static meshes don't block lighting in-game.
    - Added a CPU usage limit to skybox fogging.
    - Corrected Krall mesh mouth clamps not being masked.
    - Changed when player teleports to a new area it wont fade-in lighting to new area, but instead instantly swaps it.
    - Added render option to how aggressively it should reduce BSP lightmap framerate detail ("Lightmap LOD" in UMenu).
    - Added actor shadow occlusion distance ("Shadow draw distance" in UMenu).
    - Fixed weapon muzzle flashes to be drawn during translucent render pass instead of same time as owner player itself, prevents muzzle flash from being drawn behind terrain meshes.
    - Fixed decals to not to render on rotating mover surfaces to prevent them from glitching out.
    - Implemented some HD game fonts by Krull.
    - Fixed a bug where HUD scaling would stop working on HUD messages while UWindow was open.
    - Fixed a bug where Unreal custom cursor would stop working while RawHIDInput was enabled.
    - Made zero drawscale actors not drawn at all.
    - Software render: Fixed an overflow crash when drawing too huge tiles on too large screen resolution.
    - Made mesh lighting less flickery on bad framerate.
    - Added new HD icons and fonts by Krull0r and Sly.
    - Added a toggle option to enable/disable HD textures (for HUD and some mesh textures, found in WinDrv.WindowsClient as UseHDTextures=True).
    - Lots of fixes on already existing D3D9Drv, D3D10Drv, OpenGL.
    - Lots of fixes on meshes.
- Physics: Added PhysX physics support (simple rigidbodies and joints).
- UnrealScript: Too many to count. Check the main Release Notes.
- Server:
    - Updated webadmin interface, now it is partially native codes based, new native features include:
    - Image/binary file web response.
    - Binary file downloading from client.
    - Better support for huge data handling.
    - Advanced options property listing.
    - New web pages:
    - File upload (for simple mod uploading).
    - Advanced options.
    - Web admin accounts manager.
    - Misc features/improvements:
    - Fixed problem where sometimes messaging spectator wouldn't spawn,

thus didn't allow for viewing chat log.
    - Added webadmin privilege levels for accounts (0-minimum, 255-maximum privileges).
    - Added DoS connection detection (to block off users that rapidly try to connect to webadmin with different passwords).
    - Fixed UCC.exe Server commandlet to accept command input.
    - Made clients verify cache files GUID version before using them.
    - Added server command: UVerifyClient <ID> to manually verify a single or all clients packages with HASH values to verify they are in network sync.
    - Fixed GameInfo to not send game options to clients when switching levels.
    - Increased maximum MaxClientRate value to 1,000,000.
- New in-game features:
    - Added NoRunAway option into Unreal.ini [Core.System] to enable old crash behavior with RunAwayLimit, to help modders in developing and debugging.
    - New commands:
    - CacheRip <package>: Rips a specific package
    - CacheRipAll: Rips all packages from the server you are currently in.
    - CacheRipMap: Rips current map and dependencies. In order to avoid confusion, it just grabs the info from the server and uses it to rip the stuff from local cache directory. No server overhead is produced.
    - Note that determining the file extension correctly is only working on 227 servers, older servers don't provide the information needed. This way only maps can be identified, any other file will get the extension .u - which will work for Unreal but the disadvantage of that is obvious.
    - Added a new option ContinuousKeyEvents as a client setting to enable old behavior to continue running when typing
    - New supported prefixes for URL sharing via game chat: mailto:, http://, https://, ftp://, www., ftp., unreal://
    - Added flag to level info for SupportsCrouchJumping, which will enable crouch-jumping.
    - Added FAKELAG <ping> consolecommand for server to fabricate lag (for debugging).
    - Made HTTP downloading clients notify server of their download progress (for UnrealScript to catch serverside).
    - Fixed LevelInfo.bCheckWalkSurfaces to work. It will make Texture.Friction scale Pawns walking friction (if you use Friction 10, it will function as a ladder texture, similar to Half-Life).
- v469 Backports:
    - Backported fixed Pawn walk along ledges code.
    - Implemented some UTF-8 format fixes.
    - Pulled some render buffer overflow handling code to fix some specific render crashes.
    - Fixed editor to remember better what viewports had what render devices selected.
    - Implemented shoulder button support (aka Button4 and Button5) for Win32 mouse input.

- Updated DirectInput from version 3 to 8. The former is utterly broken on Win10/11. The latter sort of works.
- Fixed various DirectDraw issues on Win10/11.
- Added InhibitWindowsHotkeys option to UWindowsClient. If set to its default value of FALSE, the game no longer blocks the Alt+Esc, Alt+Tab, Ctrl+Esc, and Ctrl+Tab hotkeys. Setting the option to TRUE restores the original UE1 behavior.
- Removed SlowVideoBuffering from UWindowsClient because this option no longer works on Win7+
- Made the DirectInput mouse handling code read the full DIMOUSESTATE2
- Made DXGI renderers (i.e., D3D10Drv and D3D11Drv) no longer mess up your WindowedViewportX and WindowedViewportY settings while ALT+Entering
- Disabled EnhancedPointerPrecision by default, except in Unreal Editor
- Fixed several bugs that broke mouse input in Unreal Editor if you had raw input enabled in-game
- Fixed a bug that could cause GetClipboardText to read invalid data from the clipboard
- Made UWindowsViewport::ShutdownAfterError send a WM_FORCEMINIMIZE to the viewport window. This fixes a bug where the viewport remains visible if the game crashes with raw input active
- Made lighting rebuilds use a minimal null renderer (WinDrv.TemporaryRenderDevice). This should speed up lighting rebuilds
- Fixed several bugs that caused raw input not to capture the mouse correctly when switching between windowed and fullscreen mode
- Fixed several bugs that could cause the windows mouse cursor to remain visible after switching between windowed and fullscreen mode
- Fixed a bug that made TryRenderDevice crash the game when you tried to switch to a renderer that is already active
- Made SoftDrv correctly reinitialize the DibSection after changing the viewport resolution

- Linux:
  - In order to fix some old crappy problems and missing abilities in Linux, added SDL2Drv and SDL2Launch for Linux. In order to do that all old SDL stuff becomes deprecated and won't make it into 227j: SDLDrv, SDLLaunch, SDLGLDrv, SDLSoftDrv. That's nothing to worry about though, since these have been kept mostly as reference and have no features which could be compared to OpenGLDrv, which can be used in any Linux distro with at least Mesa. Details can be found here: http://www.oldunreal.com/cgi-bin/yabb2/YaBB.pl?num=1424370983/0#0
  - Updated Linux SDL to SDL2Drv.
  - Added SDL2 store window position
  - Added Linux ARM port.
  - SDL2Drv added support for RefreshRate
  - Fixed relaunch command.

.Audio

- (03/03/13) Some fixes for SwFMod (FModEx) audio driver:
- Fixed ambient sounds to use game volume and not play at max volume at all time.
- Fixed so ambient sound effects with unknown sound source (net games) to stop sound once out of hearing distance (instead of keeping the sound until mapchange).
- Fixed consolecommands GetMusicOffset, SetMusicOffset and GetMusicLen to use correct music offset values for modules.
- (10/29/13) Switched in OpenAL from FMod Music output to libxmp. This player is not only free and open source (GNU LESSER GENERAL PUBLIC LICENSE), but better portable and is yet actively maintained unlike most other players. It also plays some tracker files "more correctly" than old FMod3. OGG playing is now done directly via ogg vorbis (BSD-License).
- (10/29/13) Added CrossFading ability in OpenAL when using MTRAN_Segue.
- (05/26/15) ALAudio:
- added AL_METERS_PER_UNIT to ALAudio to be more precise with Unreal UU's
- removed ALAudio.u as its not needed anymore for 227 but added EFX.u which is a separate package in case one wants to use customized audio effects instead of presets. This file is also independent from ALAudio.dll, means if you use it in your maps anyone can download the package even although not using or having ALAudio.dll (it won't work then of course but doesn't do any harm either). This will allow also any mapper to make use of ALAudio in other UEngine games without having dll dependencies (builds will follow soon). Thanks to Han for that change!
- It also contains a couple of cleanups again and a new libxmp 4.3.8 (see http://xmp.sourceforge.net/ for update details, fixes/improvements).
- (01/06/15) Added SpeechVolume setting (ALAudio, SwFMOD, Galaxy): It sets volume of all SLOT_Talk events, to set these separately if wanted and is giving a finer control for overall sound experience. In-game, usually monster sounds use it (grunt, death sounds etc.), but also the speech of the marines known from RTNP. Should be also helpful for example in dialog sequences in custom maps & mods.
- (03/09/15) Fixed: Improved ALAudio looping wav code. Thanks Shivaxi for pointing it out!
- (12/19/20) Anth: Fixed Galaxy audio accessing outside of memory bounds when playing some sound effects.
- (03/24/21) Anth: Fixed choppy Galaxy/ALAudio playback.

.Editor

- (02/20/13) Added a hint system to UED2.1 for in game variables, which displays the content of the comment out of uscript. Simply right mouse click on any variable in the property window will open a messagebox with the corresponding help text. Left or right mouse button will close it again.
- (08/16/13) UED2.1: added a switch in View Menu to enable/disable the

lighticon in lightcolor feature.
- (10/29/13) Added function to create a new UnrealED.ini from DefaultUnrealED.ini if the ini is missing. Behaves the same like Unreal or User.ini, can be used also in case the ini is messed up in some way, just delete the ini and let UnrealED create a new one.
- (10/29/13) UED2.1: fixed default save directory in MusicBrowser.
- (11/30/13) UED2.1: Fixed filter function at the bottom of the texture browser
- (11/30/13) UED2.1: Fixed building of very large and complex spheres (which did work yet in UED 2.0 due to compiler reasons, but crashed in 2.1)
- (01/05/14) UED2.1: added export and batchexport for SkeletalMeshes.
- (02/06/15) UED2.1: added rmb menu cut/copy and paste (on click position) for actors.
- (06/06/15) UED2.1:
    - Changed in "Add Special" dialog Transparent to Translucent to fit the flag and added Modulated/AlphaBlend/Environment.
- (07/20/16) Fixed: UCC PackageFlag creates a corrupted unr package.
- (07/20/16) UED2.1:
    - added it to create "exe name".ini and "exe name".log, while I have to admit I always hated to have UnrealEd.exe but Editor.log anyway. So in future it's UnrealEd.log as default.
    - also added UEDINI=bla.ini ability, to add as commandline parameter (or shortcut that is), for multiple setups.
- (08/07/17) Fixed UED2.1: Sheer tool not working (properly)
- (08/07/17) UED2.1: Enhanced Brush Scaling and Brush Stretching to Actor Scaling and Actor Stretching.
- (11/11/19) Added to editor texture browser a right click option to compress/decompress textures and to add/remove mipmaps.
- (11/11/19) Fixed StaticMesh simple collision model creator properly handle correctly different coordinate spaces between the mesh and builder brush.
- (11/11/19) Made LevelInfo.TimeSeconds never save from editor to avoid unneccessary space wasted.
- (11/11/19) Fixed editor to reset Runaway loop counter every frame so it wont eventually always fail to run UScript code in Editor.
- (11/11/19) Fixed editor to properly handle certain Canvas functions in render callbacks (i.e: WorldToScreen, DrawBox).
- (11/11/19) Improved add actor ontop of a mesh actor tool.
- (11/11/19) Improved visuals for editor path network render.
- (11/11/19) Added right click option for builder brush to transform current shape into a dynamic zone (and added brush shape as an option for dynamic zone shape).
- (11/11/19) Made ed mesh browser 'show bones' tool also draw bounding sphere and bounds for vertex meshes. Made 'show bone names' tool show

vertex numbers for vert meshes (to help mod devs find vert indices).
- (11/28/19) Improved editor map rebuilder progress bar be more detailed.
- (11/28/19) Enabled map builder progress bar 'Cancel' button to abort building.
- (11/28/19) Added multi-threading support for BSP partitioning and light raytracing.
- (11/28/19) Fixed editor not drawing icons on ortho viewports in Direct3D and OpenGL.
- (11/28/19) Fixed an issue which made it impossible to assign to properties any objects with weird names (such as GenFX.Flarel~6).
- (12/16/19) Fixed a bug when where selecting movers in some old maps warps them to different location (i.e: Trench).
- (12/16/19) Added right-click option to replace actor while keeping same modifier properties.
- (12/16/19) Altered BSP rebuilder to bake lights to zone leafs only to visible leafs, not based on distance alone (optimizes render and improves map performance in-game).
- (01/04/20) Made editor save viewport coordinates with the maps.
- (06/01/20) Fixed DebugLines to draw in editor too.
- (06/01/20) Added to MeshBrowser a menu option to export mesh import lines to clipboard (incase you tweak them in editor).
- (06/01/20) Made properties window automatically pop-up tooltip of the property when you hover the mouse over the line for a second.
- (06/01/20) Fixed properties window to show correctly struct member values if multiple actors are selected.
- (06/01/20) Fixed properties window color selection preview color if Alpha value was non-zero.
- (06/01/20) Made edit constant variables red colored in properties window and made the textlines selectable so that you can copy the value to clipboard (such as Name or Class name).
- (06/01/20) Moved all hidden (uncategorized) properties to a "Hidden properties" category in properties window, also color coded to red.
- (06/01/20) Added an option (enabled by default) to make rotation angles show in radii angles (0-360 range) rather then UU angles (0-2^16 range). Setting is at: UseRegularAngles=False under [Core.System].
- (06/01/20) Added an option (disabled by default) to allow Actor properties window close itself if no actors are selected. This is located at CloseEmptyProperties=True under [Core.System].
- (06/01/20) Optimized Editor group browser (also added a second checkbox to lock actors, in order to keep them visible but have them unselectable).
- (06/01/20) Added a warning if you try to open a map outside of working directory.

- (06/01/20) Fixed so that maps that have missing packages to be allowed to be partially loaded (it will still give a map load warning at first and asks Y/N if you want to load anyway).
- (07/14/20) Updated toolbar icons.
- (07/14/20) Fixed LE_SpotLight/LE_StaticSpot to work correctly with light rebuild with static meshes.
- (07/14/20) Added variable TargetLevelID for Teleporter/WarpZone/Trigger/Counter to link it up with a sub-level.
- (07/14/20) Added Editor.EdGUI_XXX classes for creating custom editor windows in UnrealScript.
- (07/14/20) Added a portal directional arrow for WarpZoneInfo to help identify what direction the portal is currently facing at.
- (09/21/20) Fixed snap to grid buttons at bottom bar of editor to save changes to ini.
- (09/21/20) Also made the icon buttons have more clear graphic when grid is disabled.
- (09/21/20) Made Texture properties window update its own resolution when source texture res was changed in real time.
- (09/21/20) Fixed a visual bug with Actor class browser when you try to create class under a native class and it fails.
- (09/21/20) Made BrushBuilder also accept a Texture as icon rather then BMP only.
- (09/21/20) Added Actor.bBlockAISight which makes AI not be able to see through the actor (must have bCollideActors enabled but can be non-blocking).
- (09/21/20) Added support for custom actors to be placed now with a Brush collision shape attached to it (Actor must have bSpecialBrushActor enabled).
- (09/21/20) Added an in-editor event callback (EdBrushDeployed) which is called when actor was added to level with a brush.
- (09/21/20) updated PackageFlagCommandlet to change package flag bits only instead of loading and resaving the packages.
- (12/19/20) Updated script editor to show multiple error lines whenever it makes multiple compiler errors.
- (12/19/20) Added editor skeletal mesh socket preview (to test mesh attachments offsets).
- (12/19/20) Fixed an editor crash when trying to rebuild lighting of a static mesh that hasn't been rendered yet on camera.
- (12/19/20) Fixed mouse cursor clicking inaccurancy in ortho viewports with D3D9 and OpenGL drivers.
- (12/19/20) Added a 'Play from here' right-click menu option to editor 3D viewport.
- (12/19/20) Added right-click menu for properties window in order to allow

user to copy variables information to clipboard.
- (12/19/20) Made UnrealEd track modified packages and ask if each one of them should be saved on exit.
- (12/19/20) Made non-compiled script classes non-placeable in editor.
- (12/19/20) Fixed a crash with Projectors when exporting/importing maps where projector is attached to an actor.
- (12/19/20) Added new ShowFlags for: Static Meshes/Event lines/Projectors.
- (12/19/20) Added new actor view mode to show actors visibility as in-game.
- (12/19/20) Optimized Ortho viewport render.
- (12/19/20) Fixed drag selection box snapping to nearby actor if you click on an actor when beginning to drag box.
- (12/19/20) Updated Copy/Paste object properties in Class browser to keep proper property formatting.
- (12/19/20) Added Tooltip for object classes to show header comments of the class you're hovering over.
- (12/19/20) Added a minimum window size when opening properties windows.
- (12/19/20) Fixed 2D shape editor to properly init box with grey colored lines (and not with random leftover junk from memory).
- (03/24/21) Made font importer try to remap missing font characters with similar characters (i.e: Engine LargeFont remaps lowercase characters now to uppercase ones).
- (03/24/21) Fixed StaticMeshes resetting their static lightmap when selecting the mesh after map load.
- (03/24/21) Anth: Fixed surface properties tabs now switching properly the first time around.
- (03/24/21) Anth: Fixed couple of properties windows problems and crashes.
- (03/24/21) Added a minimum properties window size to avoid 1 pixel size properties window bug.
- (03/24/21) Added functionality to rename asset groups too (texture/sound/mesh etc).
- (03/24/21) Added right-click menu for sound browser.
- (03/24/21) Added to texture/sound right-click menu an option to copy their full name to clipboard.
- (06/13/21) Disabled in-game keybindings from firing inside editor.
- (06/13/21) Fixed editor from losing focus when closing a sub-window after using alt-tab (win 10).
- (06/13/21) Made sounds not reset nor music stop when you paste actors in editor.
- (06/13/21) Added a song section slider to Music Browser so you can listen to specific sections of the song.
- (06/13/21) Added support for custom editor keybindings (see Unreal.ini, EditorEngine.KeyBindings).

- (06/13/21) Made editor mesh browser draw meshes with lighting enabled.
- (06/13/21) Added right-click BSP surface options to Merge faces (on a tesselated surface) and Flip faces (incase they are facing the wrong direction).
- (08/02/21) Fixed when exporting LODMesh as OBJ to keep correct offset.
- (08/02/21) Made editor warn user if they save a package with same name as another package, but at different extension.
- (08/02/21) Added progress bar for "PATHS BUILD" command.
- (08/02/21) Changed all path building log lines type to DevPath.
- (08/02/21) Modified TexRotator tool when holding alt key, it rotates it snapped.
- (08/02/21) Added editor option to disable saving of LevelSummary object (if you want save map as pre-226 compatibility, found in Advanced Options).
- (08/02/21) Added LevelInfo.bRequireHighChannels which if enabled, makes 227j clients use 4X max channels on the level.
- (10/19/21) Made it possible to type in any math operations on properties window (for example Nali.Health*2 -> 80).
- (10/19/21) Added Mesh editor (under Tools menubar) which can be used to edit _d.3d and .psk mesh files to edit their polyflags or textures.
- (10/19/21) Made editor save all modified packages to System/Backup folder if editor crashes.
- (02/23/22) Added Actor.bBlockTextureTrace which can be used to tell if the actor should block footstep traces to detect footstep sounds.
- (02/23/22) Corrected default FireTexture palette from having no alpha blending value set.
- (02/23/22) Fixed editor to not crash if a BSP polygon had invalid polylink.
- (02/23/22) Optimized StaticMesh OBJ importer to work a little faster.
- (02/23/22) Optimized Brush -> StaticMesh converter to delete any polygons that get merged by vertex merging.
- (02/23/22) Added support for importing 2-bit or 4-bit PCX textures.
- (02/23/22) Fixed a BSP rebuilder error when BSP surface texture U/V mapping was too far away off from surface origin, which ended up causing BSP holes.
- (02/23/22) Made BSP rebuilder auto-tesselate BSP surfaces that aren't fully flat (instead of making BSP errors).
- (02/23/22) Fixed BSP builder giving incorrect bounding box size for dynamic BSP in some cases (ending up making movers or blocking volumes being non-solid at some edges if too far off from actor origin).
- (02/23/22) Added a right-click option to rebuild dynamic BSP model for an actor (if you for example vertex edit an mover) (Transform -> Rebuild dynamic BSP model).
- (02/23/22) Improved Tools > Scale lights editor option.

- (02/23/22) Improved Edit -> Search for actors menu.
- (02/23/22) Added LiftExit variant LiftJumpDest which tells AI to lift-jump to an area.
- (02/23/22) Fixed editor to load texture mipmaps before trying to export then (so that BatchExport textures don't crash editor).
- (02/23/22) Made editor "Show paths" show exact path size values of routes with selected path nodes.
- (02/23/22) Fixed Mover WorldTraceKey to cast shadow of the mover at correct position.
- (02/23/22) Importing SkeletalMesh animation sequences can be set without defining StartFrame value for every animation, editor will remember offset of previous animation and continue from there.
- (02/23/22) Fixed broken support for bPlayerOnly paths.
- (03/07/22) Fixed BSP builder:
    - Sometimes semi-solid brushes gets cut off from visibility bounds (disappearing at specific view angles).
    - BSP holes caused by some math imprecision.
    - Added Brush.bStrictMerging actor flag which tells builder to strictly merge it with other geometry in case where there are many surfaces parallel to each other.
    - Added some BSP debug mode options under Build Menu -> Stats to help track down source of some BSP errors.
- (03/07/22) Fixed an editor crash when trying to load old format 2D files for 2D shape editor.
- (03/07/22) Implemented OBJ Brush exporter/importer.
- (03/07/22) Fixed an editor bug where newly imported brush didn't update its bounding box (thus making red builder brush disappear from view at wrong offset).
- (03/07/22) Made brush importer less strict with excluding tiny polygons.
- (03/07/22) Fixed editor to support importing OBJ files without U/V mapping information.
- (03/07/22) Change brush intersection/deintersection tool to operate with selected brushes only if you have other brushes (other then red builder brush) selected.
- (05/29/22) Fixed editor to support core OBJ commands.
- (06/08/22) Made BSP rebuilder triangulate concave BSP surfaces using same algorithm as 2D shape builder.
- (06/08/22) Made DumpIntCommandlet merge changes with all languages.
- (06/08/22) Made Selection preview also draw on ortho viewports (for example Dispatcher event or jump pad trajectory lines).
- (06/08/22) Made BatchMeshExportCommandlet only load mesh objects from packages (so its more likely to work on outdated Unreal packages).

- (06/08/22) Added a right-click menu option to preview interpolation path in editor.
- (06/08/22) Fixed editor to cleanup post-light building junk data from BSP (reduce map filesize).
- (06/08/22) Made editor better at packing light data into BSP (reduce map filesize).
- (06/08/22) Added a progress bar for BSP zoning build step.
- (06/08/22) Fixed 8bit paletted texture export as BMP.
- (06/08/22) Fixed UnrealScript class exporter to include sub-objects in their defaultproperties.
- (06/08/22) Improved editor performance when dragging actors while moving viewport with it (Ctrl+Shift+Drag).

.Engine

- (12/04/12) Fixed: Fixed a bug with FluidSurfaceInfo where it had wrong shading on larger resolution.
- (12/09/12) Fixed: Made player footsteps cause wave ripplets on FluidSurfaceInfo as you walk in it.
- (12/09/12) Reworked all of the code on ProceduralMesh so now it's fully functional and bug-free.
- (12/09/12) Changed so InterpolationPoint.GameSpeedModifier does not modify Level.TimeDilation in a multiplayer game environment.
- (12/27/12) Changed CloudZone to avoid crashes with monsterspawners not avoiding to spawn monsters there.
- (10/28/13) Due to complaints about lighting differences in 227, especially because of the feedback from people doing some conversion from UT maps, added a new switch which enables SpecularLighting effect for all maps unless a specific new flag is set: `var() bool bSpecularLight;`. This means that 227 mappers would want to disable this flag to have the known and more accurate 227 lighting, but old and already ported maps have the 226/UT behavior. Maybe should have done from the begin with, because this means now more work for 227 mappers, especially for already finished maps, but this allows everyone to take the preferred method after all.
- (11/30/13) Added triggered MaterialSequence animation.
- (12/01/13) Fixed so "GoodCollision" movers don't clip through player if large enough and moves fast enough into player. Also changed so if you are inside the mover you can leave from it without it blocking you.
- (05/22/14) Fixed some timer issue for windows version, depending on appSeconds(), which causes mostly a problem for servers in relation with MaxTickRate. It seems that only a few systems are affected by this problem (maybe depending on hardware/bios). This bug is existing in all versions

before, including 224,225 and 226.

- (05/26/15) Added path builder information with customizable variables to LevelInfo to allow mappers building maps with different settings or for other UEngine games with UED2.1:

```
// Pathbuilder - these settings are for advanced pathing or usage of UED2.1 for other UEngineGames. Changing these settings can cause extremely weird behavior, Don't mess with it if you don't know what you are doing.
var(PathBuilder) config int MaxCommonRadius;//      max radius to consider in building paths, default 70
var(PathBuilder) config int MaxCommonHeight;//      max typical height for non-human intelligent creatures, default 70
var(PathBuilder) config int MinCommonHeight;//  min typical height for non-human intelligent creatures, default 40
var(PathBuilder) config int MinCommonRadius;//  min typical radius for non-human intelligent creatures, default 24
var(PathBuilder) config int CommonRadius;      //  max typical radius of intelligent creatures, default 52
var(PathBuilder) config int HumanRadius;      //  normal player pawn radius, default 18
var(PathBuilder) config int HumanHeight;    //      normal playerpawn height, default 39

/*
//Example values

// DeusEx:
MAXCOMMONRADIUS 115
MAXCOMMONHEIGHT 79
MINCOMMONRADIUS 12
COMMONRADIUS    52
HUMANRADIUS     22
HUMANHEIGHT     51

// RUNE
MAXCOMMONRADIUS 70
MAXCOMMONHEIGHT 70
MINCOMMONHEIGHT 50
MINCOMMONRADIUS 50
COMMONRADIUS    50
HUMANRADIUS     43
HUMANHEIGHT     20

// Disneys BearBrothers
MAXCOMMONRADIUS 18
MAXCOMMONHEIGHT 20
MINCOMMONHEIGHT 39
MINCOMMONRADIUS 18
COMMONRADIUS    18
HUMANRADIUS     18
```

```
HUMANHEIGHT     39

// KHG
MAXCOMMONRADIUS 70
MAXCOMMONHEIGHT 70
MINCOMMONHEIGHT 48
MINCOMMONRADIUS 24
COMMONRADIUS    52
HUMANRADIUS     18
HUMANHEIGHT     39

// NERF!
MAXCOMMONRADIUS 50
MAXCOMMONHEIGHT 60
MINCOMMONHEIGHT 48
MINCOMMONRADIUS 35
COMMONRADIUS    40
HUMANRADIUS     35
HUMANHEIGHT     52

// DS9:TF
#define MAXCOMMONRADIUS 40
#define MAXCOMMONHEIGHT 80
#define MINCOMMONHEIGHT 14
#define MINCOMMONRADIUS 14
#define COMMONRADIUS    22
#define HUMANRADIUS     22
#define HUMANHEIGHT     52

// Undying.
#define MAXCOMMONRADIUS 150 //max radius to consider in building paths
#define MAXCOMMONHEIGHT 150
#define MINCOMMONHEIGHT 70 //min typical height for non-human
intelligent creatures
#define MINCOMMONRADIUS 30 //min typical radius for non-human
intelligent creatures
#define COMMONRADIUS    52 //max typical radius of intelligent
creatures
#define HUMANRADIUS     18 //normal player pawn radius
#define HUMANHEIGHT     65 //normal playerpawn height

// WOT
#define MAXCOMMONRADIUS 30 //max radius to consider in building paths
#define MAXCOMMONHEIGHT 61
#define MINCOMMONHEIGHT 61 //min typical height for non-human
intelligent creatures
#define MINCOMMONRADIUS 30 //min typical radius for non-human
intelligent creatures
#define COMMONRADIUS    30 //max typical radius of intelligent
creatures
#define HUMANRADIUS     17 //normal player pawn radius
#define HUMANHEIGHT     46 //normal playerpawn height
```

*/

- (07/20/16) Fixed: Server doesn't inform clients about closing old connections on map change.
- (07/20/16) Fixed: Bug in banning system may cause duplicate (false) entries.
- (11/11/19) Added multi-threading support for memory allocator and log writer.
- (11/11/19) Fixed a memory leak when accessing destroyed actors string values.
- (11/11/19) Heavily improved skeletal mesh functions (and added option to specify transformation spaces with bone modifier functions). Also including support to use multiple animation sets per actor.
- (11/28/19) Added delta compression for saved games, also made saved games files contain save information along with a screenshot preview to show on load game menu.
- (11/28/19) Changed all objects created in-game use their class name as object name instead of unique numbered names (to conserve memory).
- (11/28/19) Made object allocator never try to replace object with same name but different class (to fix Can't replace A with B crash).
- (11/28/19) Changed order of code in DestroyActor code by setting bDeleteMe first before any cleanup code, to ensure no physics values are set on UnrealScript exit events, thus crashing later with GC/physics.
- (01/16/20) Fixed StatLogFile and TimeDemo objects to properly flush and close log file if object is destroyed (level change garbage collector or game exit).
- (01/16/20) Made engine fill out Touching array in inverse order and sort pawns to be last (legacy code support, Zora's ZM3WaterLab cannon).
- (01/27/20) Added commands "STAT THREAD" to display number of running threads and "THREADS" to display a list and description of running threads in memory (for Unreal).
- (01/27/20) Added Sound compression/decompression support for all audio drivers. You can compress sounds in sound browser or in UnrealScript by defining "OGG=X" (X is the quality in range of 0-11, 0 = worst, 11 = best) in #exec audio import line or then you can directly import ogg files as sounds.
- (01/27/20) Added support for defining HD HUD textures, by either setting Texture.HDTexture or in UnrealScript add "HD=XXX" to #exec texture/font import lines. HD texture will be used if texture is being stretched out by at least +25% of its original size.
- (06/01/20) Added IK solvers for SkeletalMeshes for easy animation blending (such as head turning, limb rotation or foot placement).
- (06/01/20) Added support for SkeletalMesh hitboxes for individual bones (and group them to bodypart type for modders to use).
- (06/01/20) Made it possible to enter spaces in names in properties and

defaultproperties blocks.
- (06/01/20) Added ZoneInfo.MinWalkableZ to tell physics engine how steep slopes Pawns can walk on.
- (06/01/20) Fixed preferences window to correctly save config to ini if you add/remove dynamic array entries.
- (06/01/20) Improved Physics PHYS_Spider to work better at finding walkable surfaces and attempt to yaw actor towards desired rotation.
- (06/01/20) Fixed a rare AActor::BeginTouch/EndTouch crash.
- (06/01/20) Implemented Alt+F4 hotkey to exit the game instantly.
- (06/01/20) Improved Log window to support command history browsing (up/down arrows) also properly support selecting/replacing parts of text or insert text in middle of command.
- (07/14/20) Added support for Sub-Levels architecture (where you can link multiple maps together into one with seamless level transition). See LevelInfo.SubLevels array.
- (07/14/20) Added GrassDeco actor for a Source Engine style grass sprites in levels (also GrassDecoSchool for an automatic grass patch).
- (07/14/20) Moved PawnList updating to be fully in C++ codes only.
- (07/14/20) Added a hidden ReplicationInfo linked list which gets setup by C++ codes (for more optimial ScoreBoard code).
- (07/14/20) Heavily optimized Navigation AI code.
- (07/14/20) Optimized server network codes.
- (07/14/20) Added support for custom mouse cursors in-game (directly use regular Textures), see Engine.UnrealCursor class.
- (07/14/20) Reworked Inventory travel code to be more optimal, and added functions GameInfo.OnPrepareTravel (event notification when it should save inventory) and GameInfo.SaveTravelInventory (to save inventory) and GameInfo.DeleteTravelInventory (to manually delete inventory).
- (09/21/20) Changed download error messages to be also printed to the HUD/console.
- (09/21/20) Made AttachMover also attach actors on clientside (mainly to fix elevator corona on Terralift).
- (09/21/20) Added an option to player console to enable script error printing to in-game console.
- (09/21/20) Added Monochrome bitmap compression (mainly for single colored masked textures like fonts).
- (09/21/20) Fixed various bugs with demorecording (added support for older 224-225 demos, added HUD features, fixed demo scoreboard etc).
- (09/21/20) Made it also possible to pause demo playback (also made it skip pauses during demo playback).
- (09/21/20) Fixed a rare crash with trying to call SetOwner onto an invalid actor or causing an infinite owner chain (Actor A owns B which owns A etc).

- (09/21/20) Compressed int replication a bit more when replicating low values.
- (09/21/20) Disable initial garbage collection when booting up the game (waste of time).
- (09/21/20) Made it impossible to try to load package 'nul' to prevent malicious loading crashes.
- (09/21/20) Made commands 'Start' and 'Open' always delay mapchange until end of tick to prevent 'Admin' command from crashing the server when using it (i.e: Admin Open). Also made said commands pass in as Level.ServerTravel event when called on server to allow clients properly to reconnect to new map.
- (09/21/20) Added Volumes support from UnrealEngine 2+ (which can be used to create brush shape collision shapes or triggers to levels).
- (09/21/20) Made LadderTrigger a Volume instead, also various fixes to ladder:
    - Player properly hops off the ladder now at top.
    - Added a flag to allow player to strafe on ladder (you must have bDirectional enabled).
    - Added a flag to enable/disable directional ladder which makes you drop off the ladder if you look away from it (must also have bDirectional enabled).
    - Pressing jump key while holding backwards now makes you now jump off ladder backwards.
    - Added a value to control climbing speed on the ladder.
- (09/21/20) Added INI file option 'LangPath' which can be used to specify director of language files (like .int).
- (09/21/20) Added a variable to SkyZoneInfo which allows you to make sky move in relative to player camera position.
- (09/21/20) Added support for static light static mesh movers.
- (12/19/20) Made package loader remove references to deleted actors.
- (12/19/20) Added last camera coordinates to crash history message whenever game crashes.
- (12/19/20) Fixed a client-side download manager crash when connection was caneled or aborted during downloading.
- (12/19/20) Added ClientMover actor for a client-side background looping movers (which doesn't interact with gameplay in any way).
- (12/19/20) Added TransporterVolume which can be used to transport all actors touching that volume to a relative offset in another place in level. Similar to Transporter but with better control.
- (12/19/20) Disabled Password logging by default game modes.
- (12/19/20) Fixed INI file writer failing if trying to write really long ini lines.
- (12/19/20) Added support for object archetype references (in order to save delta modified EditInLine objects).

- (12/19/20) Added movement interpolation between actors network updates.
- (12/19/20) Added PhysX physics support (with Physics=PHYS_RigidBody and PhysicsData reference in actor).
- (12/19/20) Added a Pawn PhysicsAnimation object which can be used to give pawns automatic physics based animations.
- (12/19/20) Fixed SecurityData to properly sanitize player names with special characters.
- (12/19/20) Made 'GETPING' consolecommand also functional client-side.
- (12/19/20) Fixed Quat Rotator conversion errors.
- (12/19/20) Heavily optimized Garbage collector.
- (12/19/20) Fixed GameInfo server travel to not send out game options to clients in server that is traveling.
- (03/24/21) Fixed an emitter crash with particle combiners.
- (03/24/21) Fixed emitter crash with SetMaxParticles function.
- (03/24/21) Replaced various emitter static arrays with dynamic arrays (NOTE: Mods making direct references to these will need to be recompiled).
- (03/24/21) Added log supression for DevPhysics in default ini.
- (03/24/21) Anth: Reduced game deltaseconds clamping to allow game run in higher FPS without speeding up the game.
- (03/24/21) Made package loader load meshes as any mesh type (regular Mesh can now load as LodMesh/SkeletalMesh/StaticMesh etc...).
- (03/24/21) Fixed map travelled inventory to have their bHeldItem flag set to true.
- (03/24/21) Added a new BSP surface flag 'invisible occluder' which works like an anti-portal surface to simply occlude view.
- (06/13/21) Added to DynamicCorona CoronaFadeTimeScale to control how fast the corona should fade in or out when pops in and out of view.
- (06/13/21) Added TriggerCorona which is a DynamicCorona that can be triggered on or off.
- (06/13/21) Removed duplicate TriggerLight actor from UnrealShare package.
- (06/13/21) Fixed players unable to walk inside warpzones.
- (06/13/21) Added simple collision model support for any meshes.
- (06/13/21) After making various emitter static arrays into dynamic arrays, made old mods functional with them.
- (06/13/21) Fixed an error with Actor.TraceSurfInfo which ended up crashing during APawn::ProcessLanded.
- (06/13/21) Added more safety checks and optimized ActorChannel management.
- (06/13/21) Allowed Actors and Pawns to perform physics outside world geometry if event FellOutOfWorld doesn't kill them.
- (06/13/21) Fixed a specific crash that could happen during client disconnection from server if exiting code would attempt to call a RPC function.

- (06/13/21) Added more validity checks to GarbageCollector.
- (06/13/21) Fixed a crash when attempting to load corrupted meshes.
- (06/13/21) Added option to MusicEvent to disable cross-level song changes (when using sub-levels).
- (06/13/21) Made ScriptWarnings dump UnrealScript call stack to log (you can suppress this with Suppress=ScriptStack).
- (06/13/21) Made default objects use name 'Default' to more easily detect errors on error stack dump during crashes.
- (06/13/21) Added Texture.SuperGlow to make texture glow with some fog overlay with bright lights (for meshes).
- (06/13/21) Added support for floating point UV texture mapping for SkeletalMeshes.
- (06/13/21) Added PhysX option for Emitter particles to make them rigid.
- (06/13/21) Added particle fade style option (to make particles change style only while fading).
- (06/13/21) Added LevelInfo.MaxPortalDepth to control how many layers warpzones/mirrors/skyboxes can draw.
- (08/02/21) Disallowing actors from moving into invalid location (division by zero, infinite location).
- (08/02/21) Fixed Pawn Z height fighting when walking on collision cylinder thats exactly the same height as BSP.
- (08/02/21) Made save game code wait until end of tick before saving the game, so it doesn't save state mid-tick, thus could break state code for a single actor.
- (08/02/21) Made UnknownXX keybindings not configurable, and therefore wont show up in User.ini.
- (08/02/21) Added consolecommand "DUMPSTATES <classname>" to dump state execution information of all matching actors currently in level.
- (08/02/21) Fixed already deleted actors from reciving one extra tick (which would happen if they have an owner and gets a delayed tick, but it destroyed by another actor before they recieve it).
- (08/02/21) Made Pawns stop their movement latent actions if their physics is PHYS_None.
- (08/02/21) Fixed inventory travel code to allow for unlimited variable data size.
- (08/02/21) Fixed Advanced Options window to correctly handle sub-section naming conflicts (such as <Advanced Options - Advance> vs <Advanced Options - Editor - Advance>).
- (08/02/21) Made editor not save actors with their state frame information (as game does reset that on map load anyway).
- (08/02/21) Added a LevelInfo flag bUTZoneVelocity to make ZoneVelocity behave like in UT games (defaults to True on levels saved in UT editor).

- (08/02/21) Added GameInfo.GameMaxChannels which controls the max number of channels 227j+ clients can have on the server (valid range is 256-65536).
- (10/19/21) Added more failsafe checks to garbage collector to reduce chance of gc crashes.
- (10/19/21) Fixed movers to not stuck pawns/players as easily if crushes pawn into ceiling.
- (10/19/21) Added a failsafe mechanism to Movers where if a pawn gets stuck inside mover it will try a couple of ticks before it completely ignores collision with the pawn and tries to push pawn off.
- (10/19/21) Fixed non-zero extent traces vs BSP to not to be allowed to pull-back trace result for more than 4 UU (where it used to be 10 % of full trace distance).
- (10/19/21) Reworked networking code to not use UObjects for channels but instead own internal struct classes for better server performance.
- (10/19/21) Fixed some rare NaN vector (division by zero) errors in physics code.
- (02/23/22) Added an INI/INT formatting for color codes, works with following formatting: (#HTML color tag), as example: (#FFFF00) = yellow color.
- (02/23/22) Made Package loading log lines log load-times from start of current load operation instead of from app start time.
- (02/23/22) Made string properties with some special characters gets written with a special text tag into INI (tags are \%HEX or \%n or \%").
- (02/23/22) Fixed a memory error when a Mesh or LodMesh has a surface with a vertex index that goes out of range.
- (02/23/22) Added a new system to prevent even more version mismatch errors when connecting to server online (if a package is already loaded and is mismatching with a package on a server, it will create a clone of it inside the game memory).
- (02/23/22) Made game remember window position when playing on windowed mode so it always keep it whenever you boot the game.
- (02/23/22) Optimized how Emitter handles RainRestrictionVolumes with weather emitters.
- (02/23/22) Added a server option to specify how many clients are needed in server before it starts to saturate network updates (TcpNetDriver.LanModeClientsCount).
- (02/23/22) Added some UE3-4 networking fixes and optimizations.
- (02/23/22) Tweaked some actor networking values.
- (02/23/22) Disabled client netstat commands to prevent server from flooding with network messages to client.
- (02/23/22) Made server not replicate small vector/rotation changes that are too unsignificant to be any different from previous value when compressed.

- (02/23/22) Fixed a crash if a textures dimensions were 1x1.
- (02/23/22) Fixed an error where properties in child classes would eat up saved values from parent classes with same name (i.e: GameReplicationInfo.Region would eat value of Actor.Region).
- (02/23/22) Changed actors to check for current zone only on end of current physics step (related to player dying when walking into a wall while standing in pain zone).
- (02/23/22) Made PHYS_Trailer actors follow Owner to different sub-levels.
- (02/23/22) Changed PHYS_Trailer on emitters to behave same way as they were DT_Mesh instead of DT_Sprite (so they don't trail behind scaled by actor Mass distance).
- (02/23/22) Corrected so when loading UT maps teleporters default their bChangesYaw to True.
- (02/23/22) Also added a Teleporter.bUTRotationMode flag which is set to true also when loading UT maps. To make teleporters behave exactly like in UT.
- (02/23/22) Made dropped inventory visible to AI on navigation network.
- (02/23/22) Changed ScriptedPawn default resting physics to PHYS_Walking instead of PHYS_None so that they will correctly fall if they are pushed off a ledge by a mover.
- (02/23/22) Added more Reset functionality for ExplodingWall, ArrowSpawner and some decals.
- (02/23/22) Fixed objects in EndGame to be more optimized for network play.
- (02/23/22) Made ExtremeDGen and EndGame usable in network servers.
- (02/23/22) Made sure that pre-227 clients don't crash when they see a default bStatic/bNoDelete actor in level that isn't static or non-delete.
- (02/23/22) Fixed game from crashing if you delete a navigationpoint that is referenced by a reachspec.
- (02/23/22) Made game de-reference deleted lights and zoneinfos on map load.
- (02/23/22) Fixed object network replication where when you try to replicate an object reference that isn't in current sandbox to replicate as 'None' instead of some random object from the network table.
- (02/23/22) Fixed actor replication with large arrays to not to send malformed packets during channel initial replication.
- (02/23/22) Fixed loading of some UT maps that have incorrect LightEffect=LE_Sunlight reference, making level oddly bright (CTF-November).
- (02/23/22) Added an option for maximum ScriptWarnings per function (System.MaxWarnPerFunc).
- (02/23/22) Implemented a hack fix for RedeemYourSpace map pack, to force player to touch any teleporters under their MaxStepHeight while standing on a mover moving downwards.

- (02/23/22) Added features for JumpPads: a flag to disable it for monsters or players, made it trigger an event when used and triggering it will toggle if its enabled or disabled.
- (02/23/22) Made player water-jump only check with blocking actors, so you don't try to jump out of water by random triggers or projectiles in water.
- (05/29/22) added a catch to Texture loader if invalid Palette and changed log a bit in to see where it fails to import
- (06/08/22) Added support for multiple language paths.
- (06/08/22) Made newly spawned players touch any overlapping triggers (only if spawning in air).
- (06/08/22) Added to LevelInfo flags to disable level specific rigidbody physics (bDisableRbPhysics/bDisableSubLevelRbPhys - if you know you wont need it).
- (06/08/22) Fixed so players never spawn as DemoRecSpectator when you load game from a save in which you've demo recorded on.
- (06/08/22) Fixed a crash that could happen if you tried to resolve a really long URL (InternetLink.Resolve).
- (06/08/22) Made so game/editor doesn't crash on boot if splash screen texture is missing.
- (06/08/22) Fixed 'First time startup wizard' menu to have functional back button again.
- (06/08/22) Increased level max BSP node count to 262,144.
- (06/12/22) Added editor support to create colored font objects.
- (06/12/22) Improved editor texture properties.
- (06/12/22) Made fonts selectable and previewable in editor.
-

.UI

- New languages: Portuguese, Catalan, Dutch.
- Overall improvements and updates to German, French, Spanish, Italian, Russian and Polish.
- All languages updated with the latest strings and their problems fixed. Any inconsistency, problem, etc. should be reported here, with pull-requests if possible: https://github.com/NeonKnightOA/unreal-localization
- (04/06/14) Added "stat video" command to enable DrawStats(Frame) in RenDev.
- (04/06/14) New changes in UMenu:
    - Preferences -> Video:
        - Changed video driver selection shows all installed drivers now (select one and press "Restart").
        - Added checkbox for fullscreen/window mode.
        - Added checkbox for trilinear buffering.

        - Added checkbox for precaching.
    - Preferences -> Audio:
        - Changed audio driver selection shows all installed drivers now (select one and press "Restart").
    - Preferences -> Game Settings:
        - Added language selection for the main localization of the game (select one and press "Restart").
    - Load/Save Menu:
        - Fixed localization support.
- (06/13/14) New feature: Localized Subtitles. Full localized subtitles in mission pack "Return to Na Pali" intro & cutscenes.
- (01/09/15) Added checkboxes in "HUD" tab to view cutscenes in cinematic style with/without subtitles.
- (05/27/15) UMenu:
    - Botmatch/Multiplayer:
        - Reorganized the entries in the game type combobox in botmatch and new network game dialog.
        - Added settings and rules tab for coop game type.
    - Preferences -> HUD:
        - Added subtitles fontsize combobox.
- (05/27/15) Classic Menu:
    - Added all new difficulty settings.
    - Added coop settings in botmatch/multiplayer menu.
    - Added subtitles settings to video menu.
- (09/21/20) Made log window (through SHOWLOG or dedicated server window) use a separate editbox for inputting commands.
- (05/29/22) Added support for advanced options localization without breaking the tree.
- (05/29/22) Fixed some UMenu Botmatch/Multiplayer menu script warnings.
- (05/29/22) Made weapons menu always show custom weapons
- (06/12/22) Added FontColored object that simply modifies color of a source font.
- (06/12/22) Compressed UColoredWindowFonts packages using the said Colored font objects.
- (06/12/22) Made Unreal main menu draw with smoothing enabled.
- (06/12/22) Made SoftDrv render premultiplied alpha tiles correctly. We need this if we want to render UT fonts.

.Gameplay

- (08/16/13) Fixed: blood drip rotating wrong when standing still.
- (08/17/13) Fixed: disappearing Slith carcass bug (slith carcass in corrosive

zones, such as slimezone).
- (12/03/13) Fixed: RTNP SpaceMarines, when a SpaceMarine leaves a spawn point on level Crashsite2 prematurely, bad things may happen: he may have blue aura, wrong fatness and mass or be completely invisible, his weapon may have wrong fatness too.
- (06/29/14) Fixed: RealCrouching caused falling through static meshes.
- (10/03/14) Fixed: Improved AI pathfinding even more by storing previous anchor path. So this should reduce the chance of having bots run back and forth between 2 pathnodes on a slope.
- (07/20/16) Fixed: Problems with pawn bleeding.
    - Immediate pawn bleeding takes place when it's not supposed to happen.
    - Postponed pawn bleeding works only once.
- (07/20/16) Fixed: Problems related to MarineMatch and SpaceMarines.
    - MarineMatch game immediately ends when TimeLimit is greater than zero.
    - MarineMatch doesn't modify difficulty level according to the bot config.
    - Gender of SpaceMarines is not properly assigned.
- (11/28/19) Made pawns network their location relative to their mover base (only between 227j clients and servers).
- (11/28/19) Made heavy pawns that crush smaller pawns not bounce off the small pawn, but instead fall through (only if it gibs the victim).
- (11/28/19) Made save game menu support unlimited slots.
- (01/16/20) Added option to LevelInfo/bEnhancedIcePhysics, that will fix walking speed on ice.
- (01/16/20) Fixed playerpawn viewbob to cap view bobbing if player is moving at very high speeds.
- (01/16/20) Fixed ZoneVelocity behave like in Unreal v 230+ versions, with an exception if ZoneVelocity.Z is non-zero it will null out zone gravity (legacy map support, Zora's ZM3WaterBridge lava jump).
- (09/21/20) Fixed ParentBlob to be able to obtain new enemy once old enemy is dead or has disappeared out of sight.
- (12/19/20) Improved TexModifiers (TexRotator/Scaler/Oscallitor) to modify U/V map instead of stretching out pixels with itself (rotation/scaling is not for BSP textures).
- (12/19/20) Made some decos, inventory and fragments orient to floor direction when landing.
- (12/19/20) Made package loader keep loading packages even if a single package was missing to throw full list of missing packages when finished trying.
- (12/19/20) Added SkeletalMesh/LodMesh load-time error checking in case of broken mesh.

- (12/19/20) Fixed Movers with StopOnEncroach to network stopped movement to 227j clients.
- (12/19/20) Fixed Movers with ContantLoop state to properly handle Stop/ReturnOnEncroach types.
- (12/19/20) Fixed orientation errors when passing through WarpZones with different orientations (gimbal lock error).
- (12/19/20) Futher optimized SkeletalMesh render code.
- (12/19/20) Fixed an inaccurancy error with footstep sound tracing.
- (06/13/21) Made weapons force stop firing while entering feign death mode.
- (06/13/21) Made player select/weapon menu draw meshes with lighting enabled.
- (08/02/21) Made Teleporters interrupt player interpolation.
- (08/02/21) Improved Movers ability to push back actors without just encroaching on them.
- (08/02/21) Made Pawns walking along ledges a little less janky.
- (10/19/21) Made user consolecommand history get saved in User.ini to be reusable on next session again.
- (10/19/21) Made UWindow combo boxes auto-scroll to currently selected item when opened.
- (10/19/21) Added to MusicMenu a suffle playlist checkbox.
- (10/19/21) Added a button to MusicMenu to add every music found in music folder.
- (10/19/21) Fixed NaliRabbits/BiterFish pick destination to not pick world origin.
- (10/19/21) Made UMenu PlayerSetup/Weapon menu downscale view model by window height to properly fit model on screen in widescreen modes.
- (10/19/21) Added mesh rotation controls to Weapon menu (by clicking and dragging the mesh).
- (10/19/21) Made UMenu warn user if you try to play singleplayer with bonus 227 difficulty levels.
- (10/19/21) Made bonus difficulty levels show in red on classic menu.
- (10/19/21) Changed mechanics of bonus difficulty levels to not increase health of monsters, but to alter their AI behaviour.
- (10/19/21) Made Emitters with missing particle texture draw nothing instead of particles with default texture.
- (10/19/21) Made Ternary conditions use precendence 50, which is highest of all operators, and first bool condition work as coerce parameter.
- (02/23/22) Corrected so that respawning inventory doesn't adjust away from wall sometimes.
- (02/23/22) Fixed an ScriptWarning when Weapon InstantFire ends up killing player owner.
- (02/23/22) Fixed some mismatching font characters on LargeFont and

BigFont (? and + characters being inverted).
- (02/23/22) Made Translucent and Modulated decorations not cast shadows.
- (02/23/22) Added an option to run game with "classic balance" so gameplay behaves exactly like in pre-227 (pupaes can't attack upwards, projectiles spread works like before and new 227 difficulty levels work like in 227i version).
- (02/23/22) Fixed Ultra shadow detail mode option not working on UMenu.
- (02/23/22) Fixed BigBiogel not playing impact sound online.
- (02/23/22) Set a default playerclass to UBrowser to use in-case user reset their config and try to join a server without visiting their playersetup.
- (02/23/22) Made clients connecting to server to load into the level as a client-side spectator until it receives a playerpawn from server (before you would be stuck in CONNECTING in entry).
- (02/23/22) Added more options to Decoration > Cannon to have more useful options for mappers to use.
- (02/23/22) Fixed a bug where if you stood in a knee-high painzone and walked into a wall it would instantly kill you because it would constantly spam you entering the zone every frame.
- (02/23/22) Fixed RTNP not ending intro maps correctly if they were translated to a different map title.
- (02/23/22) Fixed SCUBA gear from remaining active after mapchange.
- (05/29/22) Added option to run realtime shadows on single core
- (05/29/22) Added various optimizations to Emitters.
- (05/29/22) Added support to attach TrailEmitters to a normal emitter particles.
- (05/29/22) Made UMenu level preview info loader fallback to read mapinfo from int files
- (05/29/22) Added new HD icons and textures from Krull0r and Sly.
- (05/29/22) Made AI avoid trying to use navigationpoints with zero outgoing routes as entry path.
- (06/08/22) Added a secret mirror mode gamemode.
- (06/08/22) Added support for face skin selection in classic player select menu.
- (06/08/22) Added Mutator selection menu for classic menu.
- (06/08/22) Added support to attach trail emitter trails to regular particles on any emitters.
- (06/08/22) Fixed game window not always starting focused.
- (06/08/22) Fixed game window moving mouse cursor to center of screen even if it wasn't in focus.
- (06/15/22) Made game try to load LocalMap instead if AltLocalMap does not existing (if patch applied onto Unreal classic version).
- (06/15/22) Made game init simply always fallback to Entry.unr if

localmap/altlocalmap fails to load.
- (06/18/22) Fixed SkaarjTrooper sometimes not spawning weapon in their hand if they bounce off another monster on spawn.

.Render

- (02/10/13) Fixed: Coronas being blocked by BlockAll (ignores bHidden).
- (03/09/15) Added a catch for crippled meshes could crash renderer (d3d9/opengl) when using hwclipping.
- (07/20/15) Fixed: ViewFlash set in delayed (f.e. with amplifier or underwater fog) - this ancient bug was discovered by Shivaxi and debugged by Han - thanks friends!
- (07/20/16) Fixed: Bad weapon offset when viewing from a player with hidden DispersionPistol.
- (07/20/16) Fixed: Client crash with projector decals (Critical: AProjector:DrawDecalBSP).
- (07/20/16) Fixed: TimeDemo shows incorrect FPS values when Level.TimeDilation != 1.
- (07/20/16) Fixed: [D3D9Drv] Fog rendering is buggy.
- (07/20/16) Added XOpenGL OpenGL3 / OpenGL4 renderer. Note: You need to have at least an OpenGL 3.3 capable card to use this renderer!
- (07/20/16) added Exp and Exp2 fog mode as new DistanceFog equation modes. (See here: https://msdn.microsoft.com/en-us/library/windows/desktop/bb324452%28v=vs.85%29.a)
- (11/28/16) Render fixes:
  - AlphaBlend Meshes are not affected by fog. Affected renderers: Direct3D9, OpenGL, XOpenGL.
  - TrailEmitters render incorrectly. Affected renderers: XOpenGL.
  - Beam Emitters are affected by fog, making the beam a solid color. Affected renderers: Direct3D9, OpenGL, XOpenGL.
  - DistanceFog affects emitter particles. Making particles look like squares on things like SpriteEmitters. Affected renderers: Direct3D9, OpenGL, XOpenGL.
  - Switching to XOpenGL breaks texture filtering on BSP (everything looks like the No Smooth flag is set) for the first launch. Closing the game and restarting fixes the texture filtering. Affected renderers: XOpenGL (duh).
  - DistanceFog affects TrailEmitters. Making the trail a solid color. Affected renderers: XOpenGL.
  - DistanceFog causes artifacts on a Fake Backdrop/Skybox. Affected renderers: XOpenGL.

    - Environment map doesn’t work on meshes. Affected renders: Direct3D9, OpenGL, XOpenGL.
    - Masked texture don’t work on meshes. Affected Renderers: Direct3D9. Notes: looks like UseFragmentProgram was broken upon implementing exp and exp2 fog. Can't be fixed unless one writes corresponding fragment programs for D3D9, so mask problem on meshes is now fixed, but no exp and exp2 fog for D3D9 users with UseFragmentProgram enabled.
    - DistanceFog renders outside of correct zones even with bZoneBasedFog enabled. Affected renderers: XOpenGL.
    - Meshes always look unlit. Affected renderers: XOpenGL.
    - EnvironmentColor does noting and always renders Environment BSP like it is set to X=1 Y=1 Z=1. Affected renderers: XOpenGL.
    - AlphaBlend BSP is not affected by fog. Affected renderers: Software, Direct3D9, Direct3D10, OpenGL, XOpenGL.
- (08/07/17) Several fixes on meshes:
    - Decoration:
        - MonkStatue: fixed stretched UV coordinates
        - Lantern: fixed stretched UV coordinates and added faces to the arm
        - Sconce: added faces to the arm
        - Lamp4: now two-sided and faces were added to the bottom
        - Vase: added faces to the bottom
        - Urn: added faces to the bottom
    - Pickup:
        - Armor: now two-sided and have thickness.
        - Health: added faces to the bottom
        - Clip: added faces to the bottom
        - Stingerammo: added faces to the bottom
        - Shells: added faces to the bottom
        - Sludge: added faces to the bottom
        - New Quadshot firstperson model and animations.
- (10/12/17) Fixed none masked beta skin of devilfish.
- (11/28/19) Optimized mesh/lodmesh/skeletalmesh render (by precaching surface normals and simplifying some operations).
- (11/28/19) Fixed Software render to draw projectors and trail emitters.
- (11/28/19) Changed Software render to draw alpha blended textures as masked (masking invisible parts).
- (11/28/19) Slightly optimized decals and projectors render code.
- (12/16/19) Fixed Sprite/Mesh emitters sprite animation mode to function correctly (loop/play once/reverse animation).
- (12/16/19) Heavily optimized lighting render, also fixed static meshes (or any

other bShadowCast actors) to cast shadows over dynamic actors.
- (12/16/19) Added PortalModifier object which can be assigned to Texture (to be applied when drawing portal through it), ZoneInfo (to be applied when camera is inside it), or PlayerPawn (for mod override), which can be used to replace several rendering factors, such as disable lighting/fog/modify distance fog or replace textures etc. (see Object > PortalModifier).
- (12/16/19) Fixed Direct3D9 flickering with BSP and distance fog.
- (01/16/20) Added support for hardware clipping planes to D3D9 and OpenGL.
- (01/16/20) Made mesh wireframe color to match ActorRenderColor (if non-zero) in game.
- (01/27/20) Made actors that pass through zones with different zone ambient light, fade between the zone light colors rather then instantly swap them.
- (01/27/20) Made mesh actors that have their origin inside level geometry to not turn unlit and still accept lighting from nearby light sources.
- (01/27/20) Added option to upscale HUD (HUD.HudScaler) to make canvas draw HUD as it were in low res but still keep high in-game resolution.
- (06/01/20) Fixed coronas and sprites to fade out from distance fog.
- (06/01/20) Optimized render by disabling surface sorting when not needed.
- (06/01/20) Fixed a bug where skybox would turn into a HOM when viewed through a mirror reflection.
- (06/01/20) Made windows mouse pointer usable in-game on fullscreen mode (mainly for UMenu).
- (06/01/20) Added support for DynamicCorona to have directional light ray texture (which is only visible from the sides of the corona).
- (07/14/20) Optimized RenderIterator code (for Emitters etc).
- (07/14/20) Added custom LensFlares option for DynamicCoronas.
- (07/14/20) Implemented DT_VerticalSprite and DT_RopeSprite DrawTypes.
- (07/14/20) Fixed TrailEmitter to work with ColorTimeScale.
- (12/19/20) Fixed ZoneInfo.Min/MaxLightcount to be functional again.
- (12/19/20) Added ZoneInfo.LightSharpnessFactor/LightNormalMinAng variables to modify lighting method of meshes inside that zone.
- (12/19/20) Added support for animloop blending between 2 looping animations.
- (12/19/20) Made LightMap render code to use C++ style memory management and also reduce LightMap framerate in case of high BSP lightmap count on screen.
- (12/19/20) Fixed static lightmap movers to properly grab lighting at the BrushTraceKey location.
- (12/19/20) Added VisibilityRadius culling support for Decals and Projectors (also made object shadows fade-out as its about to get culled).
- (03/24/21) XOpenGL/Smirf: Removed SyncToDraw option and implemented various other bugfixes and optimizations.

- (03/24/21) Fixed a bug where meshes or sprites could not be drawn if their render bounding box were parallel to nearby BSP splitting plane.
- (03/24/21) Krull0r: Fixed Armor mesh missing bottom polygons and adjusted several LOD meshes to have less agressive LOD.
- (03/24/21) Made Canvas.DrawText support alpha blended font textures.
- (03/24/21) Added a time-based light flicker for LightType Strobe and Flicker so they can be slowed down by TimeDilation too.
- (06/13/21) Fixed zoneportals from hiding skybox in additive maps.
- (06/13/21) Fixed so render supports unlimited screen res Y size (was limited to 2880).
- (06/13/21) Fixed two sided meshes to receive lighting from both sides again.
- (08/02/21) Made Actor.VisibilityHeight as obsolete and Actor.VisibilityRadius work as spherical visibility radius.
- (10/19/21) Made volumetric fogs draw on zoneportals and sky (sky fog one is optional and can be changed from video settings between high quality/low quality/disabled options).
- (10/19/21) Made game draw volumetric fogs in fogzones even if player is standing outside of any fogzones.
- (10/19/21) Made volumetric fogging to be rendered on unlit actors.
- (10/19/21) Made map projectors properly project on any actors it touches (if bOnlyAttachStaticActors is False).
- (10/19/21) Made projectors project on first person weapon mesh too if it projects on player model itself.
- (10/19/21) Made shadow bitmaps ignore translucent mesh surfaces so they don't cast shadows (so pawn muzzle flashes dont draw ridiculous shadows).
- (10/19/21) Made shadoe bitmaps render as STY_Modulated for Software render device (or any other render devices that don't support alpha blending).
- (10/19/21) Fixed a render glitch with regular decals and made them functional on Software render device.
- (10/19/21) Changed Projectors/Decals with Style STY_None to use their textures render style.
- (10/19/21) Made translucent actor attachments get drawn in later pass if drawn on solid meshes.
- (10/19/21) Changed RenderIterator to obtain render list as linked list instead of iterating a list (and added an UnrealScript basecode for RenderIterator).
- (10/19/21) Fixed meshes and decals turning invisible when they're past FogDistanceEnd if FogMode is on non-linear mode.
- (02/23/22) Made OpenGL and D3D9 reset texture cache for an texture if their format or resolution changes but their CacheID remains the same (instead of corrupting the texture).
- (02/23/22) Fixed meshes disappearing on mirror reflections on a lot of cases.
- (02/23/22) Added a seperate MeshDetailTextures option for render drivers to

disable/enable detail textures on meshes.

- (02/23/22) Made Emitter particles reset their lighting information whenever particles respawn (so mesh emitter particles don't fade lighting from previous position particle).
- (02/23/22) Corrected blob shadow scale for RocketCans and RazorBlades ammo.
- (02/23/22) Fixed so that invisible static meshes don't block lighting in-game.
- (02/23/22) Added a CPU usage limit to skybox fogging.
- (02/23/22) Corrected Krall mesh mouth clamps not being masked.
- (02/23/22) Changed when player teleports to a new area it wont fade-in lighting to new area, but instead instantly swaps it.
- (02/23/22) Added render option to how aggressively it should reduce BSP lightmap framerate detail ("Lightmap LOD" in UMenu).
- (02/23/22) Added actor shadow occlusion distance ("Shadow draw distance" in UMenu).
- (02/23/22) Fixed weapon muzzle flashes to be drawn during translucent render pass instead of same time as owner player itself, prevents muzzle flash from being drawn behind terrain meshes.
- (02/23/22) Fixed decals to not to render on rotating mover surfaces to prevent them from glitching out.
- (02/23/22) Implemented some HD game fonts by Krull0r.
- (02/23/22) Fixed a bug where HUD scaling would stop working on HUD messages while UWindow was open.
- (02/23/22) Fixed where Unreal custom cursor would stop working while RawHIDInput was enabled.
- (02/23/22) XOpenGL: Tweaked zNear (first person weapon projected onto screen) and ScreenFlash to match up with other render drivers.
- (05/29/22) Made Canvas set proper render flags for specific fonts.
- (06/08/22) Made zero drawscale actors not drawn at all.
- (06/08/22) Software render: Fixed an overflow crash when drawing too huge tiles on too large screen resolution.
- (06/08/22) Made light effects LE_NonIncidence and LE_Cylinder behave more the same as on Meshes as with BSP.
- (06/08/22) Made mesh lighting less flickery on bad framerate.
- (06/08/22) Optimized Dynamic BSP (movers) and made it less flickery.
- (06/08/22) Added new HD icons and fonts by Krull0r and Sly.
- (06/08/22) Added a toggle option to enable/disable HD textures (for HUD and some mesh textures, found in WinDrv.WindowsClient as UseHDTextures=True).

.Physics

- (03/24/21) Added PhysX physics support (simple rigidbodies and joints).
- (03/24/21) Added new actor Physics mode PHYS_RigidBody.
- (03/24/21) Added Engine.PhysicsEngine ini setting to chose which physics engine to use (defaults to PhysX).
- (02/23/22) Fixed some PhysX memory errors.
- (02/23/22) Fixed so PhysX objects respects Actor.CollisionGroup/CollisionFlags values.
- (02/23/22) Fixed PhysX objects material friction and resitution to work.
- (02/23/22) Changed PhysX emitter particles to not to collide with each other or other PhysX objects (prevents a contact overflow crash).
- (02/23/22) Fixed PhysX objects not touching triggers.
- (02/23/22) Corrected PlayersOnly to also pause PhysX simulation.
- (02/23/22) Suppressed some useless PhysX geometry building warnings.
- (05/29/22) Added XRopeDeco that has PhysX simulated rope physics.

.UnrealScript

- (12/09/12) Added a new native function to PlayerPawn to get current client version: `native final function int GetClientVersion();`.
- (12/09/12) Added a new PlayerInterpolating state to PlayerPawn to be used on 227 clients when entering "PlayerPath" on SpecialEvent, it allows for online smooth interpolation movement.
- (06/07/13) Added some new features:
  - Variable keyword EditInLine lets you create new objects of desired metaclass and edit its variables in properties window, looks like this: www.klankaos.com/downloads/Properties.jpg
  - Variable keyword AutoDestruct will destroy and cleanup the object instantly from memory when actor is destroyed. This way you can store additional data on objects for actors and have it released from memory as actor dies.
  - Any object or class variables have a dropdown box to select from a reference in memory (in properties window).
  - Added new function for ScriptedTexture: DrawPortal, it uses software render device to draw a portal from any point in level on a texture (and works really fast).
  - Added PostTouch/PendingTouch functionality from UT as-well.
- (01/07/14) Fixed: Memory leak in dynamic array handling, improvements (DynArrayElement/Array_Insert/Array_Remove)
- (10/03/14) Added new compiler directive for UnrealScript, a C++ style variable evaluator: `<bool statement> ? <value if true> : <value if`

false>. So you can write for example: `KillerName = Killer!=None ? Killer.MenuName : "Someone";`. However, this will make your mod unloadable in pre-227 clients, crashing with unknown token error.

- (02/08/15) Changed PerObjectConfig support to output format like in UnrealEngine 2+:
  ```
  [SomeClass ClassName]
  Config values...
  ```
  It will convert old config entries it finds in old to new format to prevent problems with ini updates. This should also prevent some possible abuse with using PerObjectConfig to overwrite some other config.
- (02/08/15) Added 2 new native functions:
  - `native static final function ClearConfig();`
    Removes the PerObjectConfig ini entry for the object.
  - `native static final function array<string> GetPerObjectNames( string ININame, optional string ObjectClass, optional int MaxResults /*unlimited if unspecified*/ );`
    Finds all specified PerObjectConfig ini entries from an ini file.
- (05/26/15) Fixed: upak predators may crash or freeze the game (thanks dots).
- (06/22/15) Added missing color operators:

```
// Color operators
native(549) static final operator(20)  color - ( color A, color B );
native(550) static final operator(16) color *  ( float A, color B );
native(551) static final operator(20) color + ( color A, color B );
native(552) static final operator(16) color * ( color A, float B );
```

- (08/10/15) Added autoloading of UFactories and UExporters based on int entries to UCC. This way you can extend the capabilities of the batchexport commandlet, like adding an *.3ds file exporter for Meshes. I used this in Deus Ex to add the capability to export Text packages using batchexport.

```
[Public]
(Name=RevisionEditor.RevisionExtStringExporterTXT,Class=Class,MetaClass=Core.Exporter
)
```

- (08/10/15) New native functions `native(273) final function StopSoundSlot( ESoundSlot Slot );` and `native(274) final function StopAllSoundSlots();`. These functions stop the specified or all sound slots for an Actor. These functions are network transparent, but unlike PlaySound() use a reliable network replication pattern and do no distance checks. They share the same behaviour regarding simulated functions and invokation on network clients as PlaySound() does. This includes that you need to run those inside a simulated function when PlaySound() for that slot was called inside a simulated function (note that the server can invoke playback of a sound inside the same slot as the client for an given actor and they will both be played at

the same time, so sound slots are not mutually exclusive in this situation!). In order to avoid problems in netplay you should ensure that there is a slight amount of time between PlaySound() and StopSoundSlot()/StopAllSoundSlots() call, so both will likely be execute in order on the network client (note that the same argument applies to multiple PlaySound() calls for the same channel, so this is no new issue introduce by these functions).

- (08/15/15) Merged in some commandlets out of HTK:
  - AudioPackageCommandlet: Brought from *Unreal II*, it's an automated sound package generator. If you specify a directory with .wav files in it, this commandlet will build an .uax file with those Wave file imported.
  - MusicPackagesCommandlet: Batchconverts a directory full of music source files into a directory full of umx files.
  - ReduceTexturesCommandlet: Moves the compressed textures to the slot of the uncompressed ones, so you can have pure s3tc packages.
  - SaveEmbeddedCommandlet: Extracts embedded objects from a package.
  - ListObjectsCommandlet: Lists the Objects inside a package.
  - DumpTextureInfoCommandlet: Prints out a detailed info about textures inside a package.
- (10/26/15) Added `VClampMode=VClamp` and `UClampMode=UClamp` texture import parameters, to be able to set ClampMode at compile time, for example: `#exec TEXTURE IMPORT NAME=Bg11 FILE=Textures\Bg-11.pcx GROUP="Icons" MIPS=OFF VClampMode=VClamp UClampMode=UClamp`.
- (10/28/15) Editor.EditorEngine now optionally uses ini:Engine.Engine.EditorRender/ini:Engine.Engine.EditorViewportManager settings instead of ini:Engine.Engine.Render/ini:Engine.Engine.ViewportManager settings if those are specified in the ini file.
- (03/15/16) Added from DeusEx the idea for `var(Frob) bool bIsFrobable;` (to manipulate or adjust, to tweak this actor over Frob() interface.) which is placed in actor, with the additions, in Pawn, as new helper functions for mods, of:
  - `var() float MaxFrobDistance;`
    Max distance frob checks are being done.
  - `var transient Actor FrobTarget`
  - `function HighlightCenterObject()`
    Checks to see if an object can be frobbed, if so, then highlight it.
  - `function bool IsFrobbable( Actor Actor )`
  - In order to make this work, these new functions have been added to actor:

```
//
// Fixed and extended version of TraceTexture() which also returns
// the Texture hit (Hint: There are Sound properties on Texture).
// Extended version of TraceActors which returns additional Texture info on Mover /
LevelInfo hit.
//
native(1760) final iterator function TraceTextures
(
        Class<Actor> BaseClass,
        out Actor    HitActor,
        out Texture  Texture,
        out Name     TextureName,
        out Name     TextureGroup,
        out int      TextureFlags,
        out Vector   HitLocation,
        out Vector   HitNormal,
        Vector       TraceEnd,
        optional Vector TraceStart,
        optional Vector TraceExtent
);
//
// First Hit variant of TraceTextures.
// (The same as Trace is to TraceActors)
//
native(1761) final function Actor TraceSingleTexture
(
        Class<Actor> BaseClass,
        out Texture  Texture,
        out Name     TextureName,
        out Name     TextureGroup,
        out int      TextureFlags,
        out Vector   HitLocation,
        out Vector   HitNormal,
        Vector       TraceEnd,
        optional Vector TraceStart,
        optional bool bTraceActors,
        optional Vector TraceExtent
);
```

- (07/20/16) Fixed: PlayerCanSeeMe may return true for spectating users whose PlayerReplicationInfo.bIsSpectator is true.
- (07/20/16) Changes/additions to the following functions:
  - `native(237) static final function int Asc ( string S, optional int Pos );`
    Determines if a string Str starts with other string SubStr.
  - `native(392) static final function bool StartsWith(coerce string Str, coerce string SubStr, optional bool bCaseInsensitive);`
- (11/28/16) Fixed: AlphaBlending when using SetTile3DOffset (by using Texture.Alpha).
- (08/10/17) Added missing import parameter BUMPTEX= and BUMP= (like DETAILTEX= or DETAIL=) for textures.
- (11/11/19) Added support for 'skip' parameters to regular native functions (not just operators second parameter).
- (11/11/19) Made "replace to" parm in "ReplaceStr" skippable, meaning it

wont get evaluated if there are no replacements found.
- (11/11/19) Fixed ternary operators (<condition> ? <value if true> : <value if false>) to support l-values (and fixed a compiler bug with them).
- (11/11/19) Added "typecast" compiler keyword (can be used to hard cast any value to any type), for example: int i = typecast<int>(Class) will grab pointer value of 'Class'. Using this feture is compatible with any Unreal version.
- (11/11/19) Added struct constructor function: plane p = construct<plane>(2,0,1,0) or plane p = construct<plane>(X=2,Z=1) (both methods works), which is faster then manually assigning each struct value in script but slower then using vect or rot constants.
- (11/11/19) Fixed FindObject so that if searched object contains a dot in the name, it will seek for it as a full object name instead of any object with matching name.
- (11/11/19) Added Actor.FindSpot native function that seeks for empty space for a box extent space (same way as bCollideWhenPlacing works).
- (11/11/19) Improved Engine FindSpot to find free spot better and more efficiently.
- (11/11/19) Fixed PointCheck to only check at blocking actors.
- (11/11/19) Added Actor.bNetInitialRotation to make server network Actor rotation along with initial spawn information.
- (11/11/19) Added function Actor.OnDrawActor to be called if bCustomDrawActor is true.
- (11/11/19) Added function Actor.ReplicationEnded to notify when a bNoDelete actor channel has been closed (Actor.PostNetBeginPlay will be called when re-opened again).
- (11/11/19) Fixed Actor.PostNetBeginPlay not getting called on bStatic or bNoDelete actors.
- (11/11/19) Changed 'if' conditions to compile as coerce parms to easily allow cast them into bool comparisions.
- (11/11/19) Changed bool operators to also take coerce parameters.
- (11/11/19) Fixed a compiler bug where when it would compile a sub expression it would automatically truncate float into int, as example: int i = (1.f/10.f)*100.f would compile as int(int(1.f/10.f)*100.f), thus resulting in value 0.
- (11/11/19) Fixed ConsoleCommands to properly handle multi-parameter exec functions, if using multiple string parameters then they will be split by spaces, unless they are enclosed with quotation marks or apostrophes.
- (11/11/19) Made return value from exec function return to ConsoleCommand output.
- (11/11/19) Made if first parameter of exec function is "PlayerPawn Caller", then that will be set to the PlayerPawn executing the command (or None if ran on any other Actor).

- (11/11/19) Made ExecFunctionStr work same way as ConsoleCommand with parameters.
- (11/11/19) Added functions to LevelInfo: static native final function PlayerPawn GetLocalPlayerPawn() / static native final function LevelInfo GetLevelInfo() - To easily find active PlayerPawn and Level in the game.
- (11/11/19) Added variables to LevelInfo: RealTimeSeconds (actual TimeSeconds unmodified by TimeDilation), NetTimeSeconds (network synced between server and client TimeSeconds, only for 227j clients and when LevelInfo.bNetworkTimeSeconds is true), LastDeltaTime (current frame DeltaTime), LastRealDeltaTime (current frame DeltaTime unmodified by TimeDilation).
- (11/11/19) Added Actor.bTickRealTime to allow Actor tick by real DeltaTime unmodified by TimeDilation.
- (11/11/19) Fixed Texture trace to support trace with collision extent.
- (11/11/19) Added function Pawn.WalkTextureChange which is called if LevelInfo.bCheckWalkSurfaces is true.
- (11/11/19) Removed redundant functions Actor.SpawnAct and Actor.NativeExec.
- (11/11/19) Added parms to Actor.Spawn: SpawnInstigator/SpawnTemplate/bNoCollisionFail from prevous SpawnAct function.
- (11/11/19) Added new parm to Actor.SetTimer: CallbackObject: To call the timer function on any external object.
- (11/11/19) Made defaultproperties importer detect if same variable is imported multiple times (and throw compiler warnings).
- (11/11/19) Added feature to allow defaultproperties with numeric values contain math evaluations in them, ex: Health=Nali.Health*2 or SomeFlag=(1 << 3)
- (11/11/19) Added support for simple preprocessed maths in script code (encompassed with {}), as example: Health+={Nali.Health/2}, which will compile as Health+=20;
- (11/11/19) Changed GameRules.ModifyThreat to take third parm as enum instead of byte.
- (11/11/19) Added Actor.RenderPass to set manual render order.
- (11/11/19) Made ScriptCompiler avoid saving references to super function unless you explicitly call Super in your code.
- (11/11/19) Added a new parm to Canvas.SetTile3DOffset, bWorldOffset, to allow easily draw canvas stuff anywhere in world space.
- (11/11/19) Added variables Canvas.Mirror (to detect if current draw is inside mirror reflection) and Canvas.Recursion (to detect portal depth of current canvas call).
- (11/11/19) Added new parm to Canvas.DrawTile, vector WorldPosition, to

make it draw the tile as a sprite in level.
- (11/28/19) Added Actor.CollisionGroups and Actor.CollisionFlag which can be used to let modders set custom collision rules.
- (11/28/19) Added function bool Actor.IsBlockedBy(Actor Other) to check if another actor can block that actor.
- (11/28/19) Added Pawn.bShovePawns (and Pawn.ShoveCollisionRadius to specify internal radius) which if enabled, allows pawns to push back other pawns (kinda like Team Fortress style) depending on their mass differences.
- (11/28/19) Added function GameInfo.AllSavedGames to grab a list of all savegame slots used along with their information.
- (11/28/19) Exposed Object ObjectIndex to unrealscript and added function Object.FindObjectIndex to find an object by its index.
- (11/28/19) Changed Object.FindObject return in same metaclass type as specified in first parm, same way as 'Spawn'.
- (12/16/19) Added UnrealScript function hooks, to redirect function calls/override code (check out Object > ScriptHook).
- (12/16/19) Added event AppShutdown() to notify ALL objects when game is about to exit (not crash).
- (01/04/20) Added support for map properties (map<typeA,typeB> MapValue) aswell as accessors to it:
  - `out typeB MapValue[typeA]`
    Simple read value by key value, creates new key if not found.
  - `bool MapValue.Has(typeA, optional out typeB)`
    Check if has (and optional grab key value).
  - `bool MapValue.Remove(typeA)`
    Remove a key from the map.
  - `MapValue.Empty()`
    Empty map.
- (01/04/20) Added dynamic array call macros (it will compile as Array_Size/Array_Insert/Array_Remove calls):
  - `int Array.Size()`
    Grab array size.
  - `int Array.SetSize(new size)`
    Change array size and return new array size.
  - `int Array.Empty()`
    Empty array and return new array size.
  - `bool Array.Insert(index,count)`
    Insert new entires at offset of array (return true if succeed).
  - `bool Array.Remove(index,count)`
    Remove entires at offset of array (return true if succeed).
- (01/04/20) Added Map and Dynamic array iterators: foreach Map(out typeA, out typeB) or foreach Array(out valuetype).

- (01/04/20) Fixed script compiler to prevent modifing constant dynamic arrays.
- (01/04/20) Made Object.GetParentClass a static function and parameter to be optional (if parameter is not set it will return current class parent).
- (01/04/20) Changed Object.GetDefaultObject return in same metaclass type as input class parm (same way as Actor.Spawn does).
- (01/04/20) Added 'SizeOf(Object class)' compiler directive to return bytes of size of an object class.
- (01/04/20) Added support for declaring 'private' functions.
- (01/04/20) Added Object > LogHandler object to allow UScript to sniff and block log lines.
- (01/16/20) Added a vector2d definition (for canvas drawings).
- (01/16/20) Corrected some script warning log types to be actually ScriptWarning instead of Warning or Error.
- (01/16/20) Added functions to canvas for free polygon drawing, draw rotated textures and to push or remove custom clipping planes.
- (01/16/20) Fixed a ScriptWarning with Counter and counter messages if there is no Event Instigator.
- (01/16/20) Added variable Pawn/WalkingPct which will control walking speed of players.
- (01/27/20) Added functions to Canvas:
    - SetZTest(ERenderZTest ZTest) = Change hardware render Z-testing.
    - PushCanvasScale(float Scale, optional bool bAbsolute) = Push canvas scaling value.
    - PopCanvasScale() = Pop canvas scaling.
- (01/27/20) Added Canvas.FontScale to change scale of upcoming DrawText/TextSize calls.
- (01/27/20) Fixed Canvas.StrLen to support unlimited text size.
- (01/27/20) Fixed Canvas.TextSize to ignore color codes.
- (06/01/20) Added actor function: function bool HasMeshHitBoxes() to quickly check if actor uses SkeletalMesh with HitBoxes (to make Trace check against HitBoxes instead of collision cylinder, set Actor.bTraceHitBoxes=true, to check which HitBox bodypart type was hit by the trace, check Actor.LastHitBox.
- (06/01/20) Added a new parameter for DrawDebugLine function for making them permanent in level (or until you call ClearDebugLines): static function ClearDebugLines().
- (06/01/20) Fixed #exec directives to support unlimited characters length for the command.
- (06/01/20) Added Pawn.bNoPhysicsRotation to free Pawn from physics based rotations.
- (06/01/20) Optimized some of UnrealScript code by moving some of base Actor code to C++ codes (TriggerEvent/HurtRadius).

- (06/01/20) Added function to Pawn: iterator function AllInventory to allow for fast and safe iteration of Pawn inventory chain.
- (06/01/20) Fixed Y/N questions in UCC.exe to not skip second Y/N question if multiple ones comes after each other.
- (06/01/20) Added an event to Object: event OnPropertyChange(name Property, name ParentProperty) to notify of a property change in the object (called from properties window, SetPropertyText or SET consolecommand), also called in editor.
- (06/01/20) Added to LevelInfo: array<Object> CleanupDestroyedNotify to make custom objects receive cleanup destroyed notifications (to change destroyed actors to None in order to avoid crashes, player console is added by default to this array upon map load).
- (06/01/20) Made defaultproperties importer throw a warning if you reference to an object that wasn't found.
- (07/14/20) Added new dynamic array operators: Add/AddUnique/Find/RemoveValue.
- (07/14/20) Made Object.FindObject function same way as ObjectProperty text importer (accept format like Nali'NyLeve.Nali1').
- (07/14/20) Added new parm for Object.StringToName to set it to only find name entry, which makes it return None if that name entry is not registered yet.
- (07/14/20) Added function LevelInfo.GetSelectedObject to get selected object by type in UnrealEd.
- (07/14/20) Optimized AllActors to use PawnList or ReplicationInfo list codes whenever possible.
- (07/14/20) Added variable Actor.bCrossLevelNetwork, for which when combined with bAlwaysRelevant, allows engine to network actors from one sub-level to another.
- (07/14/20) Added function Actor.SendToLevel to send an actor to another sub-level.
- (07/14/20) Added new parm to Actor.TriggerEvent/UnTriggerEvent to allow you to send it to a specific sub-level.
- (07/14/20) Added new parm to AllActors iterator to search for actors in all sub-levels too.
- (07/14/20) Added LevelInfo.bShouldStasisLevel/bShouldChangeMusicTrack where bShouldStasisLevel controls if a sub-level should go to a paused state if no players are active in it and bShouldChangeMusicTrack controls if player entering the level should start hearing that levels Song.
- (07/14/20) Added variable Actor.NetUpdateTime which is next RealTimeSeconds server should update that actor to client.
- (07/14/20) Added new property type 'Button' for showing a simple button in object properties window.

- (07/14/20) Added new operator Vector~=Vector for approx vector comparison (allows for a ~0.0001 floating point error, now used by WarpZoneInfo for better approx comparison of coordinates).
- (09/21/20) Expanded Inventory travel code to also support travel other actors such as GameInfo and Mutators.
- (09/21/20) Added Actor.bOnlyOwnerRelevant to force actors to only network to Owner player.
- (09/21/20) Improved AnimationNotify object to be more useful and functional.
- (09/21/20) Added Array.Sort operator which is an easier access version of SortArray function call.
- (09/21/20) Added OwnerChanged event callback which is called on both client and serverside when actor owner was changed or lost.
- (09/21/20) Made Owner change to none when it was destroyed.
- (09/21/20) Added new 'Any' property type which can be used to store an arbitary property value.
- (09/21/20) Added Actor.UserData (map<name,any>) to store custom mod data with an actor.
- (09/21/20) Added 'nowarn' variable type to tell compiler to not print name conflict warnings during compilation.
- (09/21/20) Added 'protected' variable type which functions as const modifier but allows you to modify value within same class itself.
- (09/21/20) Added 'noexport' support for UnrealScript constants to prevent it from being exported to c++ header file.
- (09/21/20) Added function Engine.GetEngine to grab current engine reference in UnrealScript.
- (09/21/20) Added new parm to SetTimer to pass in data to timer callback function ('any' data).
- (09/21/20) Added Actor.bForceNetUpdate to instantly force server to replicate that actor next tick.
- (09/21/20) Added new parm to RadiusActors iterator to only check from Collision Hash.
- (09/21/20) Added new parm TraceFlags to Trace functions to allow you to define what actors to trace after.
- (12/19/20) Added CodePath variable to Unreal.ini to specify custom source code path for when compiling UnrealScript packages.
- (12/19/20) Reworked UnrealScript compiler to properly continue compiling whenever possible if it encounters errors.
- (12/19/20) Added RotAng constant which takes real angles as rotator values (i.e: rotang(360,180,0) is same as rot(65536,32768,0)).
- (12/19/20) Added support for string/name constant literals (i.e: "Some\nThings\xFF" or 'Hello\tWorld!').

- (12/19/20) Fixed another error with terniary operators (WARNING: this breaks binary compatibility with codes built on previous 227 builds that used terniary operators).
- (12/19/20) Added IntProperty variants 'DWORD' and 'UINT' to make it export to cpp as different property type.
- (12/19/20) CacoFF: Added Outer object parm for AllObjects iterator, also if you use a Struct as Outer you can iterate struct properties.
- (12/19/20) Added event RanInto for when an encroacher pushed an actor away.
- (12/19/20) Added NoDuplicate variable modifier to make it not create a clone of EditInLine object when new object is in different package.
- (12/19/20) Added NoEdSave variable modifier to tell editor to never save this variable.
- (03/24/21) Added support to call super function in a class pointer (i.e: PlayerOwner.Super(Pawn).TakeDamage(...)).
- (03/24/21) Added 'NoDuplicate' variable modifier to tell editor to never clone that editinline object to newly instanced object.
- (06/13/21) Added so that if you run UCC.made <packagename> it will force compile that as mod.
- (06/13/21) Added 'invariant' function modifier, which basically is a global function that does (mostly) only work with parameter variables. You are only allowed to call other invariant functions/variables or static functions/default variables within it, or then Object functions/variables. You can call these functions without having to reference to the class containing it.
- (06/13/21) Made Actor color operators invariant so they can be used anywhere. Also made GameInfo MakeColorCode/StripColorCodes and LevelInfo GetLocalPlayerPawn invariant for easier access.
- (06/13/21) Fixed crash with switcing states during execution of subexpression, also fixed crash with 'Can't find function XXX in YYY' (incase you switch states before the call).
- (06/13/21) Added more script functions carrying decorations.
- (06/13/21) Added 'noexport' modifier support for enums to prevent it from getting exported to C++ header.
- (06/13/21) Added function to Object: static native final function GetCallStack(out array<USScriptCallPair> Stack) - To grab current UnrealScript call stack.
- (06/13/21) Added function Engine.ConsoleCommand to call engine commands anywhere.
- (06/13/21) Added functions to Canvas: SetCustomLighting/AddCustomLightSource/ClearCustomLightSources - To control custom lighting mode for upcoming DrawActor calls on meshes.
- (08/02/21) Fixed Canvas.ScreenToWorld to transform output to correct screen rotation so you don't need to do that manually.

- (10/19/21) Made Spawn function not check collision with world for Pawns if their bCollideWorld is false or if bNoFail is enabled.
- (10/19/21) Added to ScriptCompiler a new option to compile code in optimization mode (UCC Make -optimize), which currently will only strip any Super function calls to empty function declarations (disable Suppress=DevCompile to see what it optimizes).
- (10/19/21) Made ScriptCompiler merge any string constants and solve any 'Chr(XX)' constant functions (as example Chr(13)$Chr(10) will compile into "\r\n").
- (10/19/21) Also made ScriptCompiler solve any constant math equations it finds (as example float(Asc("A")/2.f) -> 32.5).
- (10/19/21) Made LoadPackageContents only load objects of specified type user desired so it doesn't actually load entire package if you for example load only sound FX.
- (10/19/21) Added a special hack to LoadPackageContents so it's only allowed to load music objects from zxFire (Zora's map-pack music) package (as apperently it includes entire Fire.u contents from 224v).
- (10/19/21) Added legacy mod support for mods that used SpawnAct function.
- (10/19/21) Added support for C++ style for loops (for(;;) or for(i=0,j=0; i<5; i++,j+=5) etc).
- (10/19/21) Renamed PlayerPawn.GetClientVersion to GetNegotiatedVersion to match what it actually returns).
- (10/19/21) Added Mesh sockets which can be used to give skeletal mesh bones aliases for attachments with specific orientation, also to define attachment positions to vertex meshes.

Sockets can be setup in editor on mesh browser, mesh properties, under MeshSockets/PreviewSockets, then export mesh properties to clipboard to put them in code.

- (10/19/21) Added LeftHand and Head sockets for all default player models (i.e: #exec MESH SOCKET MESH=Male1 SOCKETNAME="L_Hand" VERT0=248 VERT1=72 VERT2=10 X=1 Y=0 Z=2 PITCH=-34 YAW=-20 ROLL=57).
- (10/19/21) Added PostRender2D for projectiles.
- (10/19/21) Added function Actor.GetBoundingBox to obtain actor render (or collision) bounding box.
- (10/19/21) Made #exec MESH IMPORT allow you to set any mesh properties.
- (10/19/21) Added new vector math functions: VSizeSq, VSize2D, VSize2DSq, Normal2D (VSize squared and vector 2D which ignores Z axis).
- (10/19/21) Made UnrealScript out parameters (function ABC(out int D)) function as pointer reference, so it doesn't duplicate the variable in memory.
- (10/19/21) Added 'IsValid' function syntax to check if optional out parameter has been specified, otherwise it'll cause an Accessed none ScriptWarning

when accessing it.
- (10/19/21) Added "const reference" support for UnrealScript as const out parameter, which works like a pointer but will create a temporary copy of the variable if you pass in a r-value.
- (10/19/21) Made various string functions in Object read their parameter as pointer reference if you pass in an l-value.
- (02/23/22) Removed Access None error when accessing 'optional out' parms that hasn't been assigned to anything.
- (02/23/22) Added a new alias for IsValid -> IsDefined (can be used with 'optional out' parameters).
- (02/23/22) Added function Object.RandIntRange which supports up to 30-bits of random numbers for Windows.
- (02/23/22) Added functions Object.LerpVector, LerpRotation and SlerpRotation to lineraly interpolate a vector or rotation or then to spherical rotate a rotation.
- (02/23/22) Added function PlayerPawn.CanNetworkObject to check if server or client can replicate an object reference (returns false if its not in current sandbox).
- (02/23/22) Fixed Script compiler from crashing on specific compiler errors.
- (02/23/22) Added a shortcut ClearZ function to Canvas.
- (02/23/22) Added variable modifier 'repnotify' which will make client callback OnRepNotify event when received from server.
- (02/23/22) Added variable modifier 'norepnotify' which tells client to not call PostNetReceive event if that value was changed.
- (02/23/22) Made Actor.BecomeViewTarget to get called client-side too (called by C++ codes when client receives a new ViewTarget from server).
- (02/23/22) Fixed a compiler error when you tried to use AllObject(Class'Class') iterator.
- (02/23/22) Added new line extent size parameter to Actor.DrawDebugLine to draw an axis aligned box along line.
- (02/23/22) Added function Actor.SuggestFallVelocity from UE2+ which estimates a jump/fall velocity for actor in order to land a point in level.
- (02/23/22) Cleaned up MakeCommandlet out window.
- (02/23/22) Added function GameInfo.ConsoleMessage which is triggered by server console 'SAY' command.
- (02/23/22) Fixed an UnrealScript crash where if a function returns a struct that contains dynamic arrays or string components and the function caller doesn't assign the value to anything (this will produce a new OpCode unique to 227j, so any mods compiled with such conditions wont be usable on pre-227j).
- (02/23/22) Added a compiler error if you try to compile a class with 2 replication blocks.

- (02/23/22) Added function modifiers 'reliable'/'unreliable' 'server'/'client' to have compiler add those functions directly to replication blocks, can even be used to override a replication polarity in a child class ('client' modifier also makes function simulated).
- (02/23/22) Added another function modifier 'nonetwork' to disable a replicated function from being replicated in a child class.
- (02/23/22) Added Object PlayerInteraction class which can be assigned to a player to override player input/movement/render.
- (02/23/22) Made compiler throw a compiler error if you try to use a dynamic array function on a static array.
- (02/23/22) Made Canvas.PushCanvasScale/PopCanvasScale also effect Console.FrameX/Y size.
- (02/23/22) Added function GameInfo.SanitizeString to make input playername and chat safe for logs and HUD.
- (02/23/22) Fixed Canvas.DrawText bugging out with NewLine characters.
- (03/07/22) Fixed a bug with ScriptHook where it wouldn't overwrite result value correctly.
- (06/08/22) Added a new parm to Localize function for optional default string if localization failed.
- (06/08/22) Added new functions to export/import object properties:

```
function string ExportFullProperties(optional bool bDelta /*=true*/)
function ImportFullProperties(string S)
```

- (06/08/22) Fixed Canvas.DrawClippedActor to take HUD scaling into account.
- (06/08/22) Added new optional render FOV parms to Canvas.DrawActor/DrawClippedActor.
- (06/08/22) Fixed a compiler crash if you had in code #exec obj load file=xxx package=yyy where file wasn't found or failed to load.
- (06/08/22) Fixed a script warning if PHeart actor had a swapped mesh.
- (05/29/22) Added new localize feature to Object.Localize to make it optional.
- (06/15/22) Fixed OBJ REFS debug command not being able to pick specific object aswell as made it output more useful information like variable names that makes reference to said objects.
- (06/18/22) Corrected InternetLink default encoding for better old mod compatibility (and added UnrealScript control over how text is encoded for data transfer).
- (06/18/22) Added function InternetInfo.GetLocalIP from UT.

.Server

- (11/29/14) WebAdmin improvements:
    - Updated webadmin interface, now it is partially native codes based, new native features include:

        - Image/binary file web response.
        - Binary file downloading from client.
        - Better support for huge data handling.
        - Advanced options property listing.
    - And new web pages:
        - File upload (for simple mod uploading).
        - Advanced options.
        - Web admin accounts manager.
    - Misc features/improvements:
        - Fixed problem where sometimes messaging spectator wouldn't spawn, thus didn't allow for viewing chat log.
        - Added webadmin privilege levels for accounts (0-minimum, 255-maximum privileges).
        - Added DoS connection detection (to block off users that rapidly try to connect to webadmin with different passwords).
- (12/16/19) Fixed UCC.exe Server commandlet to accept command input.
- (03/24/21) Fixed WarlordRockets spawning smoke and SeekingRockets spawning decals on servers.
- (03/24/21) Made clients verify cache files GUID version before using them.
- (03/24/21) Added server command: UVerifyClient <ID> to manually verify a single or all clients packages with HASH values to verify they are in network sync.
- (03/24/21) Fixed GameInfo to not send game options to clients when switching levels.
- (05/29/22) Increased maximum MaxClientRate value to 1,000,000.

.Optimizations

- (12/04/12) Altered many of Engine classes to use TouchingActors iterator instead of the Touching list array for better results and fewer bugs.
- (11/11/19) Optimized Real crouching code.
- (11/11/19) Optimized serverside networking code.
- (11/11/19) Made vectors better compressed over network between 227j+ clients only.
- (11/11/19) Made all meshes animations lookup use a mapped hash value instead of an array lookup for a huge performance boost in animation lookup code.
- (11/11/19) Made Actor shadows use multi-threading for rendering it.
- (11/11/19) Optimized and improved dynamic zones.
- (11/11/19) Added a projector feature (Projector.bBuildStaticMap) to make it generate pre-rendered model for BSP and bStatic StaticMeshes for a huge performance boost with MapProjectors.

.Bug fixes

- (11/11/19) Fixed server to remember what variables has been networked when a bNoDelete actor becomes network irrelevant and then relevant again.
- (11/11/19) Fixed a security flaw with networking which clients could abuse to leak memory on server.
- (11/11/19) Fixed a division by zero error with BigRock.
- (11/11/19) Improved stability with decals and projectors on save/load games.
- (11/11/19) Fixed a bug with TcpLink and UdpLink which would make it shut down socket twice upon destroy (thus throwing a windows warning).
- (11/11/19) Fixed SunlightCoronas from shining through static meshes (and optimized it).
- (11/11/19) Fixed weather emitter particles to also hit static meshes.

.New in-game features

- (02/20/13) Added NoRunAway option into Unreal.ini [Core.System] to enable old crash behavior with RunAwayLimit, to help modders in developing and debugging.
- (03/03/13) New commands:
  - CacheRip <package>: Rips a specific package
  - CacheRipAll: Rips all packages from the server you are currently in.
  - CacheRipMap: Rips current map and dependencies. In order to avoid confusion, it just grabs the info from the server and uses it to rip the stuff from local cache directory. No server overhead is produced.
    Note that determining the file extension correctly is only working on 227 servers, older servers don't provide the information needed. This way only maps can be identified, any other file will get the extension .u - which will work for Unreal but the disadvantage of that is obvious.
- (07/20/16) added a new option ContinuousKeyEvents as a client setting to enable old behavior to continue running when typing
- (07/20/16) New supported prefixes for URL sharing via game chat:
  - mailto: (new)
  - http://
  - https://
  - ftp:// (new)
  - www.
  - ftp. (new)
  - unreal:// (new)
- (11/11/19) Added flag to level info for SupportsCrouchJumping, which will enable crouch-jumping.
- (11/11/19) Added FAKELAG <ping> consolecommand for server to fabricate

lag (for debugging).
- (11/11/19) Made HTTP downloading clients notify server of their download progress (for UnrealScript to catch serverside).
- (11/11/19) Fixed LevelInfo.bCheckWalkSurfaces to work. It will make Texture.Friction scale Pawns walking friction (if you use Friction 10, it will function as a ladder texture, similar to Half-Life).

.v469 backports

- (06/13/21) Ported over from network fixes: if packet loss prevented channel from closing it will be resent, made channels open in a sequential order to prevent client->server RPC call from switching actor refs on parms incase actor happens to get destroyed inbetween.
- (10/19/21) Backported fixed Pawn walk along ledges code.
- (10/19/21) Pulled some render buffer overflow handling code to fix some specific render crashes.
- (02/23/22) Implemented some UTF-8 format fixes.
- (02/23/22) Implemented an Access None error fix which fixes a crash if Access None happens before an Iterator or inside a Switch condition.
- (02/23/22) Added unicode conversion options for StatLogFile.
- (06/08/22) Fixed editor to remember better what viewports had what render devices selected.
- (06/08/22) Implemented shoulder button support (aka Button4 and Button5) for Win32 mouse input.
- (06/08/22) Updated DirectInput from version 3 to 8. The former is utterly broken on Win10/11. The latter sort of works.
- (06/08/22) Fixed various DirectDraw issues on Win10/11.
- (06/08/22) Added InhibitWindowsHotkeys option to UWindowsClient. If set to its default value of FALSE, the game no longer blocks the Alt+Esc, Alt+Tab, Ctrl+Esc, and Ctrl+Tab hotkeys. Setting the option to TRUE restores the original UE1 behavior.
- (06/08/22) Removed SlowVideoBuffering from UWindowsClient because this option no longer works on Win7+
- (06/08/22) Made the DirectInput mouse handling code read the full DIMOUSESTATE2
- (06/08/22) Made DXGI renderers (i.e., D3D10Drv and D3D11Drv) no longer mess up your WindowedViewportX and WindowedViewportY settings while ALT+Entering
- (06/08/22) Disabled EnhancedPointerPrecision by default, except in Unreal Editor
- (06/08/22) Fixed several bugs that broke mouse input in Unreal Editor if you had raw input enabled in-game

- (06/08/22) Fixed a bug that could cause GetClipboardText to read invalid data from the clipboard
- (06/08/22) Made UWindowsViewport::ShutdownAfterError send a WM_FORCEMINIMIZE to the viewport window. This fixes a bug where the viewport remains visible if the game crashes with raw input active
- (06/08/22) Made lighting rebuilds use a minimal null renderer (WinDrv.TemporaryRenderDevice). This should speed up lighting rebuilds
- (06/08/22) Fixed several bugs that caused raw input not to capture the mouse correctly when switching between windowed and fullscreen mode
- (06/08/22) Fixed several bugs that could cause the windows mouse cursor to remain visible after switching between windowed and fullscreen mode
- (06/08/22) Fixed a bug that made TryRenderDevice crash the game when you tried to switch to a renderer that is already active
- (06/08/22) Made SoftDrv correctly reinitialize the DibSection after changing the viewport resolution

.Linux

- (03/29/13) Fixed Linux: relaunch command.
- (05/17/13) Cleaned up OpenAL to work better with OpenALSoft in Linux, changed some bad coding, improved EFX error handling. Maybe will fix some of the problems in windows people experienced with non CreativeLabs cards as well.
- (02/19/15) In order to fix some old crappy problems and missing abilities in Linux, added SDL2Drv and SDL2Launch for Linux. In order to do that all old SDL stuff becomes deprecated and won't make it into 227j: SDLDrv, SDLLaunch, SDLGLDrv, SDLSoftDrv. That's nothing to worry about though, since these have been kept mostly as reference and have no features which could be compared to OpenGLDrv, which can be used in any Linux distro with at least Mesa. Details can be found here: http://www.oldunreal.com/cgi-bin/yabb2/YaBB.pl?num=1424370983/0#0
    - In the stats page changed the order from V / U to U / V
- (08/07/17) Linux:
    - Updated Linux SDL to SDL2Drv.
    - Added SDL2 store window position
    - Added Linux ARM port.
- (02/23/22) Fixed a Linux server crash when map contained a PathNode route reference to itself.
- (02/23/22) Made Object.Rand function limit number to 15-bits for Linux (so Titan boulders don't spin with an insane speed).
- (05/29/22) SDL2Drv added support for RefreshRate

.Version 227i

Release date: November 11, 2012

- Added RawHIDInput for most precise mouse input and to fix the problems with DirectInput.
- Added UED2: a reset button for the brush builders.
- Added UED2: real-time preview (for selected pathnodes) so that you can see how pathnodes will be bound with each other (it also draws a cylinder around selected pathnode so that you can see how far it can bind).
- Added UED2: moved parts of path building code from hardcoded C++ codes into UnrealScript, that includes the special paths for LiftExit/LiftCenter/Teleporter/InventorySpot etc...
- Added UED2: modified 227 JumpPad's to draw a yellow line preview of the throw trojectory also modified so AI paths won't get bound reverse up the jump pad path.
- Added UED2: 2 variables to NavigationPoint; ForcedPathSize and MaxPathDistance.
- Added UED2: ForcedPathSize will be the path "size" when using "ForcedPaths" list, so that mappers can limit smaller pawns to use them.
- Added UED2: MaxPathDistance is the maximum distance the pathnode will seek for path around itself.
- Added UED2: render device selection menu for TextureBrowser and MeshBrowser. Most people didn't know that it could be set up in Unreal.ini with "WindowedRenderDevice". Softdrv is in many cases not sufficient (like AlphaBlend Textures or for StaticMeshes) to display everything correctly and now it can be easily switched between SoftDrv/D3D9 and OpenGL during runtime.
- Added UED2: Added 2DShapeEditor GridSize 128/256, asks to save when creating or opening new shape.
- Added New animation notify code. Just extend class AnimationNotify and define notify:

```
AnimationNotify(0)=
AnimName=RunSm,FunctionName=Testing,KeyFrame=4,NotifyEval=ANE_Greater,
bOwnerCall=true,bCallOncePerLoop=true,bCalculatedFrame=true)
```

- Added option which will make server compress normal server downloads (this will only work for 227i clients connecting to 227i servers!) before they are being sent, thus reducing the download times greatly. The files server compresses before being sent to client are saved in CompressDir, you can even use UCC.exe Compress commandlet and move the compressed files in to that folder to prevent hitches on server (for first time downloads to files). It can be set on Unreal.ini:

```
[Engine.ChannelDownload]
UseCompression=True
CompressDir="..\Compress\"
```

- Added option for AI strafing movement while moving along paths (AI: bDoAutoSerpentine, also in NavigationPoint: bNoStrafeTo to stop AI from walking to this node like that). Which means AI will strafe left/right while moving along paths making them harder to hit. This will be enabled on higher difficulty bots.
- Added crawling option for Pawns (Movement: bIsCrawler), which means pawns like pupaes can actually roll and pitch according to the floor slopeness.
- Added pathfinding option HuntOffDistance, which when combined with bHunting, AI will try to find closest path to a visible point to their enemy which is inside that distance, rather than trying to find path leading to the actual enemy (will also now be used by higher difficulty bots).
- Added Meshes support all different LightEffects too now.
- Added altered LightEffect LE_OmniBumpMap for a slower radial light which does some real-time light raytracing (meaning BSP can cast shadows around it).
- Modified all dynamic light effects (except for LE_NonIndicence) to not lit up backfaces on BSP (looks more realistic that way).
- Replaced the 227h variable bDayLight with LightEffect LE_Sunlight (and made Sunlight render slightly more optimized).
- Added actor variable (Display: bUseSpriteLit) to make DrawType DT_Sprite to get lit up by lights like meshes.
- Added UED2: Modified light actor icons to be rendered in editor in the color of their light source (to show a small preview of their light color like in UE2+).
- Made Galaxy audio support Ogg/Midi and other audio music formats now like OpenAL/FMod (using FMod library to play them, but still uses original Galaxy for the old music formats).
- Added UED2: for meshes an option "ShadowMesh" which is used for computing shadows instead of the original mesh, so it is possible use specific meshes for more detailed shadows (useful for trees for example).
- Added UED2: auto disable realtime preview when test playing map.
- Added UED2: confirmation, asks to save when opening/loading from MRU/ exiting UED and a map with unsaved changes is still open. Also added "Cancel" to those dialog windows.
- Added throttle for downloading from servers DownloadSpeedLimit.
- Updated security and anticheat system. Switched from MD5 to SHA256 for anticheat, added better support for older clients. Using a separate ini file:
    - Server installed packages can be included (whitelisted) and updated with `ucc engine.shaupdate` for 227 clients.
    - For older clients anticheat support there is now the commandlet `ucc engine.linkerupdate <directory>` which will add all checksum of the files within the directory automatically to the SHALinkerCache.ini. This way it should be very easy to secure and update the server also for custom packages such as textures.
    - If you want to whitelist a package in general you just need to update a line f.e. with `GenFluid=`.
    - Note that full anticheat protection is only possible with 227i clients due to technical limitations of the older versions. Only a basic protection can be guaranteed with 224,225 and 226 clients.

- Added double Nodelimit
- Added actor Lighting option bDarkLight, which will turn the light source into subtractive light instead of additive.
- Added a new parameter for SetTimer function (optional name InFunc) where you can specify callback function which will also allow you to start multiple timers at once.
- Added logging to see which file could be responsible in case of Bad name index -50/55 crashes. This crash can be caused by crippled cache files. To enable it, Suppress=DevLoad in [Core.System] needs to be removed or commented out.
- Added new defaultproperties macros for dynamic arrays: Array.Add(Value), Array.Remove(Value), Array.Empty
- Improved static mesh render performance by ~20 % for some hardware render drivers (OpenGL, D3D9).
- Added support for static mesh lighting for dynamic (but yet static like TriggerLights or pulsing lights) light sources.
- Added so all meshes with bShadowCast + bUseMeshCollision cast BSP shadows that can pass through masked parts of the texture surfaces (for more realistic tree leafs shadows).
- Added support for localized female death messages in UnrealIGame (for languages like Polish or French).
- Modified actors to use a dynamic array for Touching actors list (to prevent bugs with triggers touching too many actors), it will still sync up the dynamic array with the static Touching array list to keep mod compatibility.
- Added screenshots preview for default Unreal maps for UMenu.
- Added new emitter options:
    - `XParticleEmitter.EmVisibility.bNotOnPortals`: Don't draw particles through portals or mirrors.
    - `XEmitter.EmVisibility.CullDistanceFadeDist`: This will make particles 'fade out' based on camera distance to the particle (it is in relation to CullDistance value).
    - `XEmitter.EmVisuals.Scale3DRange`: Particle 3D scale range.
    - `XWeatherEmitter.WallHitEmitter`: When matched another Emitter's Tag it will spawn those particles upon wallhit.
    - `XWeatherEmitter.WaterHitEmitter`: When matched another Emitter's Tag it will spawn those particles upon impact with water surface.
- Added UED2: "Align Viewport Cameras to 3D viewport" as right mouse button menu selection and button on the left side (misc section)
- Added UED2: "Rebuild BSP only", so people can build Maps like in UED1 with every step separately if wanted.
- Added exit information to appRequestExit, if Unreal exited by unknown reason such as maybe mod failures. So it is possible to see at least where or why it did that.
- Added SwFMOD, some new sounddevice like OpenAL or FMod, but unlike old FMod (FMOD 3) its based on FModEx (FMOD 4). Dots reworked some old code I found for DeusEX and updated it to newest version while also fixing a lot of bugs, also the ambient presets known out of OpenAL are supported. It's new and of course there is a good chance of new bugs, but it has a lot of potential,

not only for Windows but also for Linux users.
- Added UED2.1: "Batchexport to BMP from Package" and "Batchexport to PCX from Package" for textures
- Added UED2.1: "Batchexport to WAV from Package" for sounds
- Added UED2.1: for ActorBrowser "Find..." which prints out the location of a class (example "Inventory Pickup Health" for Bandages). Should safe some time especially if digging for some class in unknown mod packages. Also selects the class found, for use in ClassEditor or to place into map, but does not expand it in the browser itself.
- Added editor option (AlwaysPermanentBrush=True/False) to always transform all brushes permanently every time you scale or rotate them (pretty much like in UnrealEd 3+) to minimize floating point transformation imprecision in BSP.
- Added new canvas functions:
  - `native(1758) final function SetTile3DOffset( bool bEnable, optional vector Offset, optional rotator RotOffset, optional bool bFlatZ, optional float Scale );`
    You can set all canvas DrawTile (or text) calls to have a 3D coordinates off screen. This allows you to create a HUD with a slight angle and depth like in most modern games.
  - `native(477) final function DrawTileStretched( Texture Tex, int X1, int Y1, int X2, int Y2 );`
    You can draw a stretched tile which works like the one in UnrealEngine 2.
- Added new factories (to be used with BatchExport commandlet):
  - Mesh txt/uc factory: Can be used to write #exec import lines for the meshes and skeletal meshes.
  - SkeletalMesh psk/psa factories: Can export skeletal meshes and animations from package
- Added for UED2.1: export for all classes in MyLevel.
- Improved FluidSurfaceInfo to use a faster math algorithm and added a few extra features. Also added a FluidSurfaceOscillator class to generate constant waves at one spot. WARNING: Had to move FluidSurface from Engine to Fire due to dependencies the updates required and to clean up some things.There are not many maps out there using it yet, if at all, but now the performance is significantly improved it will be used more in future I hope. Just keep in mind that you need to redo 227h Fluids in your map if used.
- Added OldClientCompatMode. This enables a hack which fixes the connection issues with 226f clients, meaning that every old version (224,225,226b,226f) can connect to 227 servers now. Epic and Legend broke the compatibility between classic Unreal 226f and UnrealGold, where both versions were conformed against 225, which made 225 the only server version for both these two and the following versions. 227 suffered all the time from this, but since the mistake happened long in the past there was no suitable fix for it yet. The feature was tested intense and very carefully, it seems to work without any problems or side -effects. It is still a hack and will be globally enabled for all old clients once active, so it can be disabled if necessary. Note that this will not fix any 226f specific bugs and mods that are already broken

on 226f. While I still can only encourage everyone to move on 227 nobody needs to feel forced to with this option and server admins can make use of the advantages of 227 without being afraid of losing players.
- UED2.1: Added a switch to disable "Are you sure you want to save new filesize kb, old filesize kb)" save dialog. [Editor.EditorEngine] AskSave=True
- UED2.1: Added RightMouseButton on Surface -> Tesselate Surface (3D view surface context menu)
- Fixed: the "famous" execClientHearSound crashbug. Only very rarely happening (while some gametypes like MonsterHunt trigger it, but in a usual game it almost never happens). Seems to be some ancient bug and you can find reports for Unreal, UT, Unreal2 and UT2k3.
- Fixed: Bots intelligence set to BRAINS_HUMAN
- Changed: Removed UltraResShadow option. Can be still set manually in ini, but caused to many trouble because it just eats a lot performance, even on nowadays machines and people disregard the warnings about it.
- Fixed: Playermodels fit in Player Setup Window in UMenu
- Fixed: "Sockets" command trigger a crash mostly on Servers. 227 provides newer commands and make it obsolete, so Servers with updated mods are not affected, but some older mods still use it.
  Note: Even with this fix it can still trigger crashes, but the chances are lowered drastically. Nevertheless, updating older mods is recommended.
- Fixed "Ending Symbol" not showing up in coop mods (f.e. JCoopz)
- Fixed some outputs in Linux build (f.e. obj list).
- Fixed umod install
- Fixed a small flaw in the setting of the BufferSize for Music output in both OpenAL and FModAudio. Mostly Linux users should benefit from this fix. Windows users usually don't need to change this value manually.
- Fixed download redirect where if one file failed on redirect it would download rest of the files off server (instead of proceeding with the redirect download).
- Fixed FMod and ALAudio audiodrivers to loop correctly sounds with looping points.
- Fixed actor reachability checks to function correctly with smaller pawns (such as Pupaes) so that they can use pathnodes if their AI flags allow that.
- Fixed UED 2: snapped brush scaling sometimes not snapping anymore with changed pivot
- Fixed UED 2: snapped brush scaling overflow when scaling "smaller than possible"
- Fixed UED 2: actors visible in level through wall when used in skybox (and not only in fakebackdrop)
- Fixed UED 2: "Save Brush as" / "Open Brush" (NOT import/export) function. Stores position of the brush as well as texture information.
- Fixed UED 2: StaticMesh behavior in front of FakeBackdrop surfaces (Z issue).
- Fixed UED 2: RightMouseButton on Surface -> Reset (was entirely nonfunctional in UED2 and 2.1, UT and Unreal version).
- Fixed download redirect where if one file failed on redirect it would download rest of the files off server (instead of proceeding with the redirect download).
- Fixed FMod and ALAudio audiodrivers not looping correctly sounds with loop points.

- Fixed actor reachability checks to function correctly with smaller pawns (such as Pupaes) so that they can use pathnodes if their AI flags allow that.
- Fixed pathfinding to work with pawn that is on PHYS_Falling too.
- Fixed UED2: 2DShapeEditor shows the selected GridSize in Menu.
- Fixed Cannon, now it even really shots! (optional)
- Fixed UED2 importing sounds didn't accept spaces in pathname.
- Fixed UED2 "Replace with..." was setting bRemoteOwned flag causing some actors behaving strange.
- Fixed some mover bug which caused movers to hang when standing below it (online only issue).
- Fixed the final issues with meshes disappearing in some cases, this also made FullMeshRendering obsolete.
- Fixed Pawn.MoveToward where if the Pawn is moving along PathNodes and if the Pawn misses the origin of the pathnode (due to curving path) it wont turn around to touch the origin before proceeding.
- Fixed pathfinding to support smaller and larger pawns (like Pupaes and Titans).
- Fixed higher difficulty levels stick with Actor filter option bDifficulty3.
- Fixed Linux client some Emitters not working correctly.
- Fixed Emitter crashing when doing undo after delete
- Fixed Ban IP Addresses in Security.ini for older clients got overwritten always by latest client IP
- moved by heavy request on UnrealSP the bEnhancedSightCheck to pawn and removed it to be automatically enabled if difficulty >3
- Fixed UMenu match setup Friendly Fire scale slider from not working.
- Fixed EndGame level travel to FlyBy map instead of bringing up Unreal main menu.
- Fixed UED2: order of how things are build when doing "Build all", which seems to make a significant difference in reducing holes.
- Fixed UED2: import for 32bpp pcx, crashed before (but still suggest using bmp instead for that).
- Fixed a bug with NavigationPoints.
- Fixed bUseLitSprite on AlphaBlend sprites.
- Fixed UED2.1: "Select Inside" was selecting static mesh movers outside.
- Fixed a couple of possible loopholes and security flaws in UCC Masterserver commandlet to make it more reliable and stable.
- Fixed OpenAL may crash when trying to load crippled or truncated sound data.
- Fixed a flaw in "Path=" parameter in Unreal.ini section [Core.System] so it is able now to read also .int files in custom directories, like: Paths=..\SystemInt*.int.
- Fixed PlayerCanSeeMe() function.
- Fixed Aura3.Luckshot not working anymore.
- Fixed UED2: Transform permanently to keep U/V mapping perfectly intact.
- Fixed some problem in logging system which could cause a corruption in Security.ini.
- Fixed some boulder decal issues which caused the decal to be displayed cut off.

- Fixed UED2: brush clipping resulted sometimes in brushes with vertex rounding errors.
- Fixed UED2: undo to work with vertex editing.
- Fixed -strict commandline parameter to cause crashes in critical functions only. This way it can be used by modders or serveradmins who prefer a crash instead of a log only, for debugging purposes.
- Fixed interpolation point end in water.
- Fixed UPak cloak jump bug.
- Fixed Automag decal and ricochet sound.
- Fixed IPDrv on Server could cause problems with older clients using Nephthys.

.Version 227h

Release date: June 2, 2011

- Added a scalable Translator for higher resolutions.
- Added "export selected package" for classes in UED.
- Added support for projector decals (bProjectorDecals) to fix decals not working on static meshes.
- Added modified editor class browser to append star (*) after class name if the class is not placeable in level (marked abstract, nousercreate or transient).
- Added support for DrawText color codes (similar to Unreal Engine 2).
- Added 2 color code functions in GameInfo:
  - `static native final function StripColorCodes( out string S );`
    Strip color codes from a string.
  - `static native final function string MakeColorCode( color Color );`
    Make a color code string.
- Fixed OpenGL doesn't reset gamma correctly on some system.
- Fixed online some sounds not working online
- Fixed movers glitch when player is pushed into ceiling
- Fixed D3D8/9 startup freeze when using same fullscreen resolution as desktop resolution.
- Fixed UPak Rocket Launcher secondary fire showed no decal online.
- Fixed Client didn't disconnect correctly on exit
- Fixed UED2.1 frame disappearing when exiting game from inside UED started Map.
- Fixed decals not fogged with distance fog
- Fixed missing frameborder when switching to windowed mode in OpenGL
- Fixed Titan weird behavior when running against cliffs.
- Fixed UPak "Bounds of Foundry" trigger broken because of NaliC2 fix.
- Fixed some cache crash when loading files (with dependencies) from cache
- Fixed another mover bug and Linux save game blows with .rut translation
- Fixed fragment program breaks fogging on meshes (such as gibs) if set to true in OpenGL.
- Fixed Movers where triggered when leaving them (especially DM maps were affected)
- Fixed bug with struct import text where it failed to import if struct contained

static arrays. While at it made any property text imports show name of property on log upon import failure (instead of just ImportText: Failed because of blabla).
- Fixed editor class browser to show display properly 2 classes with same name but different packages.
- Fixed missing "Unreal" gui for UMenu from UGold
- Fixed UPak intro to use the original "Papyrus" Font again.
- Fixed decals not working on StaticMeshes. Requires to switch to projector decals (bProjectorDecals), which need more performance especially on StaticMeshes. Using projector decals fixes the very very rare decal crash bug too.

.Version 227g

Release date: April 5, 2011

- Added: "Change audio device" button to unreal recovery mode (for windows), just in case some audio device causes trouble.
- Added: Coords maths functions (for advanced rotations):

```
// Rotate coords A by B
native(330) static final operator(16) Coords * ( Coords A, Coords B );
native(331) static final operator(34) Coords *= ( out Coords A, Coords B );
//Rotate coords by rotator
native(332) static final operator(34) Coords *= ( out Coords A, rotator B );
//Inverted rotate coords by rotator
native(333) static final operator(34) Coords /= ( out Coords A, rotator B );
//Rotate vector by coords
native(334) static final operator(34) Coords *= ( out vector A, Coords B );
//Get default forward direction coords.
native(335) static final function Coords GetUnitCoords();
```

- Added flag for Actor to enable multiple "Enviroment mapped" textures on 1 mesh actor.
- Added enhanced SkeletalMesh support; caching skeletal pose, set/get custom bone rotation/offset scale, play multiple animations at once.
- Added better support for "Drop Detail FPS" to reduce FX count with specific effects.
- Added third person recoil animation support for all standard Unreal weapons. Changed package loader so that it does not give `"Package 'ABC' version mismatched"` in online games that much anymore.
- Added UED2: ew option into Editor section of Unreal ini: **FreeMeshView**, when enabled you can fly around in mesh viewer the same way like in 3D view (no fixed positioning anymore)
- Added new menu items in context menu (Right Mouse Button) **Align to wall around X axis** and **Align to wall around Y axis** in order to fix alignment

problems.
- Added search function so that Unreal can use Cache.ini to find in cache some specific files that are missing (like when trying to open some maps and you are missing some textures or whatever). This way, files don't need anymore to be moved out of cache, renamed, etc, etc if you want to load, for example, a specific map which needs textures or sounds you have downloaded from a server.
- Added new function to Actor which can be used to get BSP surface information (was used as fix replacement for footstep textures):

```
function bool TraceSurfHitInfo( vector Start, vector End, optional out
vector HitLocation, optional out vector HitNormal, optional out
Texture HitTex, optional out int HitFlags );
```

- Added UED2: "Remove script" function to classes browser; and changed this HIGHLY ANNOYING behavior that it shrinks the menu down to "actor" when adding a new class. It now stays where the new class was added.
- Added UED2: Textures Clamp/Wrap option to avoid borders in corners of skyboxes when using Textures > 256. Works in D3D8/D3D9 and OpenGL.
- Added: DXT3/DXT5 support and the necessary additions to UED2. Any format can be chosen during import of bmp files (32bpp) directly in UED now.
- Added UED2: "Do you really want to compile all scripts?" dialog to avoid recompiling everything (which needs a lot of time) when clicking "Compile All" in class editor by accident.
- Added "Flat Shading" support for mesh rendering.
- Added **USECPU=** to startup command parameters. This way, a server admin can choose which server runs on which CPU core (on multicore or multiprocessor systems) when having multiple server instances on one machine. Also maybe useful if running both client and server on one machine simultaneously.
- Added functions **AttachActorToBone** and **DetachFromBone** to attach specific actors to bones on skeletal meshes.
- Re-implemented support for curvy meshes and made use of it in a couple of default Unreal meshes (however, there's still an option to disable this). In order for meshes to be Curvy, it no longer uses **Actor.bMeshCurvy** (in order to avoid unexpected results with mods); instead, in order to enable it, you need to set on Mesh import parms **Curvy=1**.
- Added new actor called **FluidSurfaceInfo** for creating a wavy water mesh for water surface.
- Added functionality for surface flag "Environment" on BSP surfaces, in order to add an environment mapped texture overlay on the surface.
- Added a couple new functions for Object. Convert string to lower case, replace part of some string, sort dynamic and static arrays (in a very performance effective way):

```
static native(238) final function string Locs( string InStr );
native(239) static final function string ReplaceStr( string Text,
string FindStr, string ReplaceWith, optional bool bCaseInsensitive );
```

```
native(240) static final function bool SortArray( ArrayProperty Prop,
Function SortCode );
native(241) static final function bool SortStaticArray( Property Prop,
Function SortCode, optional int SortSize );
```

- Added UED2: Texture property **PaletteTransform** in order to allow change palette color range to that desired color (useful for changing FireTexture color instead of having to import some dummy palette texture).
- Changed screenshot names to **MapName-Timestamp.bmp** in order to have a more useful name and information. This also removes the limit of max 1024 screenshots.
- Improvement performance of "trace" code.
- Added new actor flag **bWorldGeometry** in order to make the actor be treated as part of world geometry (block visibility sight and stop explosion radius etc...).
- Added a shadow bitmap texture render that projects some mesh actor in level (used in pawn shadow as an option).
- Added: decals render as wires in wireframe mode (RMode 1).
- Added UED2: **Select All Actors** and **Select Inside Actors** buttons. These were buttons already available in UED1 but disappeared in UED2 for unknown reason.
- Added for mappers:
    - LevelInfo:

```
var() bool bSupportsRealCrouching;
Support crouching f.e. through tunnels with half player height
var() bool bEnhancedSightCheck;
If enabled, AI can see through transparent/masked BSP etc. Also
faster. Enhanced check is enabled for new difficulty levels by
default.
```

    - Pawns:

```
var(AI) float SightDistanceMulti;
Multiply this AI's sight distance over the hardcoded limit with this.
Useful for very big maps.
```

- Reintegrated **Small Diagonal** and **Big Diagonal** rotations from UED1.
- Added Preprocessor commandlet updated to version 0.5.296.
- Added new commandlet **Editor.CompareInt <Int file> <Misc language file>**, used for comparing 2 language files to add/remove lines/groups that's missing from the other file.
- Added 4 simple texture modifiers. You should be able to dynamically create these in-game and switch their textures or values (however editor preview may act up when switching source textures):
    - Texture scaler, simply rescale some other texture render scale.
    - Texture panner, makes some texture panning.
    - Texture oscillator, makes a texture wavy stretching or panning.
    - Texture rotator, rotates a source texture.

- Added UED2: **AlphaBlend** option when importing 32bpp bmp's as DXT3/5. Notice that during import only AlphaBlend or Masked can be used. DXT1 and 256 color indexed textures can't use this feature for obvious reasons. Also added STY_AlphaBlend for usage f.e. in sprites to make them fade in / fade out.
- Added new class **VisibilityNotify**. It's used for stuff like security cameras/portals etc... in order to make them work online so that you see stuff behind the other side. In order to use it, simply add 2 of those actors (one in each side of the portal), set a collision radius/height that covers a visible area around there and make sure they have the same Tag set.
- Added new functions for Actor:

```
native final function bool TraceThisActor
{
     vector              TraceEnd,
     vector              TraceStart,
     optional out vector HitLocation,
     optional out vector HitNormal,
     optional vector     Extent
};
```

    - Simply returns true and gives out hit information if a single trace did hit this actor only (will now be used by projectiles for performance).
    - `native final iterator function IntDescIterator( string ClassName, optional out string EntryName, optional out string Desc, optional bool bSingleNames );`
      An iterator version of "GetNextIntDesc", which works a lot faster than other function and provides some extra features (will now be used by some menus).
- Added Linux: case insensitive search for filenames during import when building with UCC. This was added in order to make it possible to compile mods in Linux which were mostly written in Windows, which is case insensitive. Because of that, most filenames are not correct and failed during import. This addition is only for **IMPORT FILE=""** in classes and does NOT include a case insensitive search for folder names, these have to be fixed manually, but compared to filenames this can be done very fast. This makes it possible to fully use UCC in linux, except font import, which is unfortunately an entirely windows based routine at the moment.
- Added **Actor.PostNetBeginPlay** event to notify client whenever actor has just been spawned and all initial variables has been replicated.
- Added support for multiple redirect websites, to use that add (in Unreal.ini):

```
DownloadManagers=IpDrv.HTTPDownload http://www.site1.com/redirect/
DownloadManagers=IpDrv.HTTPDownload http://www.site2.org/redirect/
DownloadManagers=IpDrv.HTTPDownload
DownloadManagers=Engine.ChannelDownload
```

- Added UED2: Editor Mesh Browser shows mesh package name as well.
- Added Moved headshot detection from weapons to Pawn.IsHeadshot in order

to allow pawns themselves decide whenever they got headshotted. This doesn't affect the original gameplay.
- Added new config setting for OpenAL and FMod (mainly for Linux) **ProbeDevicesOnly** which can be used to start up Unreal and let it detect only the available sounddevices instead of trying to use them directly. Since blocked or not correctly chmod'ed devices can cause a segfault directly, this can be used to determine the devices in a failsafe way. The available devices depend on the chosen Output (see below). The device list can be read in `Unreal.log/UnrealLinux.bin.log`.
- Added new configuration for preferred OpenAL Output.
    - Windows user can choose between WINMM, DSOUND, A3D
    - Linux users can choose OSS, ALSA, ESD
- Added: FMOD device and output selection for OpenAL's music output (which is based on FMOD), this makes it more configurable and hopefully will fix some remaining problems, especially with some Linux systems.
- Added Color flags for FontImport (`new TrueTypeFontFactory PACKAGE="URedWindowFonts" Name=RedTahoma10 FontName="Tahoma" Height=10 AntiAlias=0 UseGlyphs=1 R=255 G=0 B=0`). You can now set R / G / B during import to affect the color of the imported font. Valid values are between 0 and 255.
- Added UED2: Brush manipulation buttons known from UED1: **Sheer, Scale and Stretch** (while the in UED2 already existing Scale button was in UED1 "SnapScale" and to explain its function correctly it is renamed now accordingly like in UED1).
- Added UED2: **All Textures In Use** Button for TextureBrowser which makes the Browser show only the textures of a package which are used in a map
- Added UED2: Added Next Frame and Previous Frame buttons in MeshBrowser to browse through the mesh frame by frame.
- Added: Z-sorting for meshes with AlphaBlend so that meshes can be used with DXT3/5 alpha blended textures. However, sorting can never be perfect so that gaps can appear with many overlapping objects. So its always a trade off with smooth blending or sharp edges when using masked instead.
- Added UED2: 5 customizable context menu entries (right mouse button) like **Add Light Here**. Those can be entered into UnrealED.ini:

```
[ContextMenuClassAdd]
Custom1=Engine.Playerstart
Custom2=Engine.Pathnode
Custom3=...
Custom4=...
Custom5=...
```

- Added: **LadderTrigger** actor for making a climbable spot.
- Added: **ForcedPaths** and **ProscribedPaths** to NavigationPoint, in order to force add/remove path bindings while rebuilding paths (you input Name or Tag of the desired path nodes).
- Added: MapLists use dynamic array (to support unlimited maps to be put on cycle).

- Added: "Shuffle" option for maplists to play the maps in random order.
- Added: Allowed ServerActors to modify ANY properties on spawn, rather than limited to 'config' variables.
- Added: Moved Translator render code from UnrealHUD to Translator, allowing modified Translator classes to override the looks of it.
- Added support for additive BSP mapping.
- Added additional buttons for Music Menu: An own Music volume slider, Play random music track, Music play time (once music played X amount of minutes, switch to next random music track).
- Added: Made web links on UWindow console to be clickable with your mouse pointer.
- Added: option **bUseMeshCollision** to Actor; putting this to true will make actor use mesh zero animation frame as collision shape.
- Added: option **bUseGoodCollision**, to use an alternative method for collision calculating (gets rid of invisible collision hull problems on complex movers). However a note for mod authors (if you use mesh collision): after modifing some values (drawscale, mesh or the flag itself), you MUST call `SetCollisionSize(CollisionHeight,CollisionRadius);` in order for it to update collision hash bounds for the actor.
- Added: "CollisionPlane" actor for setting an simple easy collision rectangle for your map (for simple collision boundries).
- Added: 2 options for servers inventory travelling system:
  - **bDeleteTravelInvOnLoad**, deletes player inventory data from travel actors list after player gets their inventory once (to prevent players from reconnecting in order to stock on inventory).
  - **bServerSaveInventory**, save/load Inventory data to an INI file (to keep inventory even after server crash).
- Added some UnrealScript events (to GameInfo) for server inventory travelling handling:
  - **GetPlayerTravelID**, gets player identification for travelling inventory (by default: player name).
  - **ModifyTravelList**, can modify the list of actors that's about to be exported for travelling.
  - **SpawnTravelActor**, spawns an incoming travelling actor to the level.
- Added PowerMac G5 PPC port.
- Added Mac: control for cpu affinity, implemented some platform specific stuff for integrity check, fixed some masking problem due to byte order and added some high resolution timer for more precise appSeconds.
- Added Linux: added control for cpu affinity and added some high resolution timer for more precise appSeconds.
- Added Windows: added control for cpu affinity and added some high resolution timer for more precise appSeconds.
- Added: flag for Actor to enable multiple "Enviroment mapped" textures on 1 mesh actor.
- Added: enhanced SkeletalMesh support; caching skeletal pose, set/get custom bone rotation/offset/scale, play multiple animations at once.
- Added third person recoil animation support for all standard Unreal weapons.
- Added UED2: 2 new menu items in context menu (Right Mouse Button):

Align to wall around X axis and **Align to wall around Y axis**.
- Added server option to disable third person recoil animations.
- Changed weapon third person mesh render code to spawn a separate actor (a WeaponAttachment class) that updates location and rotation to weapon triangle/bone. This fixes issues with having first person mesh and separate third person mesh while viewing yourself at mirror, makes game uncache/recache mesh render data every frame twice over and over... Allows UnrealScript mod authors to add emitters or something on third person mesh position. Do basicly whatever you want with the third person mesh on weapons, as it's no longer as hardcoded as it used to be.
- Added new option to manipulate music pan separation for FMOD and OpenAL because there were some complains about it would be separated to hard.
- Added better support for "Drop Detail FPS" to reduce FX count with specific effects.
- Added UED2: switch for realtime preview also. So you can choose separate if you want to have backdrop for normal and for realtime preview or not. Defaults are as it was before, settings are saved at exit.
- Added option for **StaticMeshActors bBuildStaticLights** which means editor will raytrace per vertex light data for it (so other objects in map can cast shadows on those static meshes) but it also means that their lightmap will be static (only dynamic lights can be applied on top of that).
- Added UED2 the out of UED1 known check: "An Actor class named Example already exists! Do you want to replace it?" dialog for duplicate class names, in order to prevent messing up an existing class by accident.
- Added UED2 ability to import meshes from .obj (Wavefront) files.
- Added UED2 support for import multiple .obj files at once.
- Added UED2 display Texture size 512 and 1024 (will only work if window size is big enough)
- Added in PlayerPawn:

```
//Returns true if player is holding down that key number.
native(549) final function bool IsPressing( byte KeyNum );
```

- Added in GameInfo:

```
// Servers only) Force to load travelling inventory for a client
(returns true if inventory was found)
native(920) final function bool LoadTravelInventory( PlayerPawn
Other );
```

- Updated network codes, prevents some client crashes as well as improves server performance.
- Added replication modifiers. These replication modifiers are not visible in UScript replication blocks, only in C++ codes:
    - Actor:
        - **bSkipActorReplication** - Do not replicate any actor properties.
        - **bRepAnimations** - Do replicate animation updates.
        - **bRepAmbientSound** - Do replicate ambient sound.

        - **bSimulatedPawnRep** - Replicate physics like for Pawns.
        - **bRepMesh** - Do replicate mesh + skins.
    - Pawn:
        - **bRepHealth** - Do replicate health to others than player himself.
    - ZoneInfo:
        - **bRepZoneProperties** - Do replicate all standard zoneinfo properties.
    - Inventory:
        - **bRepMuzzleFlash** - Do replicate muzzle flash.
        - **bRepPlayerView** - Do replicate player view properties.
- Added mesh import feature to define polygon flags upon mesh import by adding FLAGS=XXX after settexture exec lines.
- Added to zoneinfo 'FadeTime' for distance fog, so distance fog blends smoothly to new fog properties.
- Added new texture modifiers:
    - **MaterialSequence** (good for screenshots),
    - **TexCombiner** (to combine 2 textures in specific way),
    - **ScriptedTextureX** (more advanced scriptedtexture).
- Added UED2 editor "Copy defaultproperties" and "Paste defaultproperties" to copy/paste defaultproperties block in Class Browser.
- Added network fix which should prevent specific network incompatibilities (namely from S3TC textures), but it only works with 227g+ clients on 227g+ servers.
- Added UED2 "Load Entire Package" button in Texture, Sound and Meshbrowser
- Added missing Open/Close Package button in Meshbrowser
- Added MeshMover UED2 support in MoverMenu
- Added UED2 "Lighting only" mode for 3D viewport.
- Added UED2 "Export to bmp" which can export DXT Textures as 24bpp bmp as well.
- Added UED2 "Are you sure you want to save new filesize kb, old filesize kb)" to save dialog to see and avoid map/package corruption if new filesize is smaller than old.
- Added UED2 EXPORTFONT/IMPORTFONT commandlet for fontworks,available in command (log) window
- Added UPAK (RTNP) decal support
- Added UPAK (RTNP) UPakFix mutator to fix online RTNP Coop gaming issues. It's located in the helpfolder.
- Updated MusicMenu to have play offset scrollbar (only works in FMod/ALAudio drivers).
- Updated C++ definition for native functions 'Out' parameters to take less game performance.
- Added new audio (FMod/ALAudio) consolecommands:
    - **GetMusicOffset** - Outputs currently playing music offset (Modules in currently playing row, Streaming tracks in MS).
    - **SetMusicOffset <Row/MS>** - Set currently playing music offset to.
    - **GetMusicLen <Music name>** - Get music length (Modules in number of rows, Streaming tracks in MS).

    - **GetMusicType <Music name>** - Get the type of music (IT/S3M/MIDI/STREAM).
- Updated map T3D exporter to write less redundant data.
- Updated T3D importer to load up necessary texture packages used by BSP rather than slapping default-texture all over them (if not currently loaded on editor).
- Added to map rebuilder to cleanup small map errors that might been caused from copy/pasting/undo/redo to avoid editor from freezing during build.
- Added whenever a package/map is saved in editor it asks if you are sure that you want to overwrite old version if new one is smaller in filesize.
- Added UED2 whenever package/map saving failed, it pops up a window rather than showing it in log only.
- Added UED2 export and batchexport for vertex meshes
- Reverted so that texture Specular value defines the "glowing" of the mesh lighting (the way lighting behaves in pre 227f).
- Fixed: non western European chars and that makes it now also possible to change the fontsize for the game. So if you play with a very high resolution you may want to increase fontsize. The definition is language specific and can be found in the Engine.[localized] files.
- Fixed: Improved performance of some UMenu menus.
- Fixed: Made all transient objects have transient names in game (transient names as instead of: ActorChannel0, ActorChannel1, ActorChannel2, they get ActorChannel, ActorChannel, ActorChannel etc..) in order to save memory usage for server and clients.
- Fixed: Changed so that coronas can be seen through invisible/masked/transperent BSP surfaces.
- Fixed: Changed so that ServerPackages/ServerActors no longer crash the server if they fail to load or are not found.
- Fixed: Replaced the 2 last Trace function parameters with BSPTraceFlags and bTraceBSP.
- Fixed: bleeding.
- Fixed: specialdamage not working correctly.
- Fixed: Changed package loader so that it does not give "Package 'ABC' version mismatched" in online games that much anymore.
- Fixed: LevelInfo.GetClientPort to return client port instead of server port.
- Fixed: Warning in ScriptedPawn.SetEnemy and Bots.Roaming.
- Fixed: Coding error in SentryGun and Translocator.
- Fixed: support for Decals to be visible/invisible on panning/unlit surfaces.
- Fixed: Changed render DrawActor to handle RenderIterator. This way you can use Canvas DrawActor to render entire particle emitter on HUD instead of just the icon of it.
- Fixed: the scoreboard to show for spectators "Press [Fire] to view from different player" instead of "You are dead, press [Fire] to respawn".
- Fixed: Improved UMesh::GetFrame to avoid some specific errors.
- Fixed: Reimplemented function BeginState for Bots GameEnded state to avoid conflict with mods thats compiled in 226.
- Fixed: `Canvas.DrawActor(None,,true);` so it can be used to clear render Z without crashing the game.

- Fixed: a crash with Emitter's trigger spawn particles.
- Fixed: error where friendly creatures hate players who hurt themselves.
- Fixed: PlayerPawn GameEnded state to not update actor rotation online for third person players.
- Fixed: Changed so that bots behaviour with special shoot open doors (like in DMCurse) so that they don't spam projectiles while running into the doors.
- Fixed Warlords from instantly disappearing when they should teleport out.
- Fixed: problems with DEP (Data Execution Prevention) on newer OS.
- Fixed: UED2 Framecount in Meshviewer
- Fixed: "random wandering" glitch with scripted pawns
- Fixed issue where when you don't have any other selectable inventory and toss final one you get messages in a bad order.
- Fixed double expired messages bug (such as with flashlight when the batteries drain out).
- Fixed so player don't become visible when they type 'walk' cheat command when they are dead.
- Fixed issue with some specific inventory such as the Amplifier, when it runs out it makes you select next item even when Amplifier is not selected.
- Fixed: Brute rocket speed problem online.
- Fixed: ASMD Tazerproj becomes invisible when shooting against f.e. masked sheets.
- Fixed another small security flaw for both server and client
- Fixed an error that happens when level contains an actor that had bStatic/bNoDelete false when in their defaults its true (such as the boulders in EndGame.unr).
- Fixed: stalls during package download from server via Engine.ChannelDownload (connecting to a server without http redirect). Fixes this problem for all clients.
- Fixed UED2: can't delete Textures.
- Fixed: Trace for Footstepsounds not working correctly.
- Fixed a crash that could sometimes happen on decal DeattachDecal.
- Fixed UED2: changed a crash to a warning message which happens in some rare occasions with tesselated cubes.
- Fixed UED2: crash when selecting "Export" in Texture Browser while no Texture is selected.
- Fixed: water/iced/scripted textures not working with DXT textures.
- Fixed: monk statue push sound loops forever.
- Fixed: DXT compressed textures preview on editor to no longer show as garbage on software render (until you flush view).
- Fixed: lighting issue with mirror viewed meshes (showing mirrored lighting).
- Fixed: UMenu to use UMenu load game menu instead of classic load game menu.
- Fixed: UMenu to share same saved games list as classic save/load game menu does (so you don't lose saved games when you switch console).
- Fixed critical security leak for servers.
- Fixed UED2: Got ride of the annoying "Actor dosen't fit there" error in Editor when trying to add actor to level.
- Fixed: Translocator Target from hitting triggers and other non-blocking stuff.

- Fixed: an issue with mesh specular lighting where surfaces would get lit up by light sources behind the surface itself.
- Fixed: another mesh lighting error where shadow/lit direction was mirrored on mirrors.
- Fixed: rotation network replication error (which is clearly seen on pupaes that have been hanging on walls).
- Fixed: Made RMode 1 render the world in geometry view as well in game (easier to check up on map errors).
- Improved mesh actor rendering even more so it surely doesn't disappear from sight when its clearly in front of you.
- Fixed network replication so that you can see actors through transperent/masked surfaces in online games (such as windows or grates).
- Fixed decals not rendering in mirror view.
- Fixed UCC: Ctrl-C causes the appearance of windows crash-window (incl. annoying blam! sound) during "ucc make" and exits smoothly now
- Fixed: DXT support for Linux internal engine decompression and SDLSoftDrv.
- Fixed Linux: UnrealXLinux.bin, XDrv and XMesaGLDrv for testing purposes, not many features, not very fast. But maybe for some experimental use.
- Fixed TTFImport and added some Texture replacement function which replaces Fonts regarding the language corresponding Engine.[localized] file. So it can be easily a font-texture created with TTFontImport to replace the ingame whitefont, medfont, largefont, bigfont with a fontset which supports the necessary chars. Also created a new UWindowFonts.utx which should contain most needed chars directly. Note, this is Windows only!
- Fixed: `UCCLinux.bin make` segfault crash when finished compiling.
- Fixed bmp import in Linux.
- Fixed Linux: most likely cause of mysterious segfaults when issues with MasterServer.
- Fixed: UMenu maplist from being empty all the time.
- Fixed Linux: a couple of reasons for segfaults caused by AudioDevices, all OS: code cleanup for FMOD and OpenAL.
- Fixed Linux: EFX support for ambient presets for OpenAL's EAXZoneInfo. So any ambient and reverb should now work in Linux also.
- Fixed lighting/shadowmap bug in Linux for fully optimized builds. Increases overal Linux performance greatly.
- Fixed DXT3/5 compression size.
- Added UED2: missing "Texture Browser" to view menu.
- Fixed UED2: changes in Texture properties not automatically visible when using OpenGL or D3D8/9.
- Fixed UED2: crash on exit when using Menu->File->Exit.
- Fixed UED2: Replace Texture tool can't be re-opened anymore after one time usage.
- Fixed UED2: 2D Shape Editor Buttons Revolved Shape, Extruded Shape, Extrude to Point, Extrude to Bevel not accessible anymore after closing.
- Fixed UED2: shape vanishing after moving with right mouse button pressed over window borders.
- Fixed UED2: "Extude" typo for to Point and to Bevel.
- Fixed UED2: display of fonts in TextureBrowser by group selection.

- Fixed UED2: a rare crash with Mesh Browser that could happen on startup if the mesh that was going to be first on mesh list had same name as a class name.
- Fixed UED2: SMPDev fixed selection in D3D9.
- Fixed Linux: changed logging so that it doesn't segfault anymore when starting fails due missing packages. Instead it prints the error now to console and closes nicely the logfile. Also added some additional logging info for console which hopefully helps if segfaults happen because of c++ exceptions.
- Fixed UED2: context menu -> View -> Show Coordinates to show current camera position.
- Fixed UED2: context menu -> View -> Show Backdrop to really show skyboxes without actually being in realtime mode.
- Fixed UED2: If you select a light, go into its properties, LightColor, Color, try to access the 3D viewport, press OK - your viewports no longer respond, and when you try to close the editor, it crashes.
- Fixed UED2: Merging vertices with vertex editing results in a "not enough vertices" crash, but that has already been reported before. This is stupid to do, but if you create a new material, for example, TexPanner, and set it to pan itself, the editor crashes in the way that even the crash system fails (you get a Windows crash message and the log is cut off).
- Fixed: Reduced the chance for crash failure at "TouchTo", which should fix some errors with projectile spamming mods.
- Fixed: ammo pickups, when you pick up 'child' ammo type it didn't show any pickup message unless the pickup will respawn (good example; clips).
- Fixed: inventory pickup messages to always show the pickup's own pickupmessage, rather than the carried item pickupmessage (usually noticed when mappers have modified the pickup message for some items).
- Fixed: Made UnrealShare.Suits item to destroy themselves and throw a warning on log if added to the map (as they should never be used anyway).
- Fixed: UnrealScript compiler to show properly 'error' count.
- Fixed: Third person weapon mesh from sometimes having wrong display properties.
- Fixed: HUD to show properly multi-lined messages (so that next message don't overleap the second line).
- Fixed: HUD to render messages after the armor icons (so that event/death messages don't overleap armor icons).
- Fixed: Changed HUD name identifier to show the name of the player in front of CAMERA location rather than PLAYER location (fixes spectator name identifier).
- Fixed Linux: sometimes crashes at startup. Also speeds up start.
- Fixed "Dirty Shadows" surface flag functional. Since never implemented it was of no use so far, although many mysteries came up to this, it never had any effect. (No, not in Unreal and no, not in UT). Now it creates some random dark shadow spots if used. Maybe for virtually "uneven" surfaces. The factor can be set in LevelInfo with DirtyShadowLevel.
- Fixed so that rolling/pitching movers properly update their attached actors location/rotation in relation to the delta rotation.
- Fixed UED2 bsp building so that way less bsp holes arise. So far no known

side effects (except slightly more polygons).
- Fixed shadowing on meshes. Now decal/projector shadows work on meshes too.
- Fixed execDrawPortal Sprite offset.
- Fixed Decals can crash Client if Mirror is present.
- Fixed disappearing Meshes in some rare view angles. Fixes also Playermodel vanishing in Mirror if to close.
- Fixed rescaled Movers collision.
- Fixed another security bug which can cause a server crash.
- Fixed UED2 crash which happened if you clicked on a brush builder directly after a map import.
- Fixed: Attitude_Follow.
- Fixed a memory leak in FMod.
- Fixed Krallbolt translucency to STY_Translucent.
- Fixed now (hopefully) the "vanishing mesh" bug completely (as addition there is now a bAlwaysRenderFlag to ensure it for problematic places.
- Fixed Demorecording
- Fixed movers from appearing at level origin when you just connected to a network server or in demorec playback.
- Fixed "Admin" consolecommand to output each "line" from engine commands to output as own clientmessage, instead of output all in one long message. This includes "Admin Sockets", "Admin UHelp", etc...
- Fixed a client mapswitch crash which could happen on servers with custom playerpawn classes (with function Destroyed/EndState).
- Fixed Demo Recording to work on offline mode, as server host and as a client on server.
- Fixed Unicode support for Linux.
- Fixed/Updated network replication so bNoDelete/bStatic actors with replication do not re-replicate variables that are changed from actor default values by the mappers. This will reduce server bandwith usage a bit as-well as fix some obscure bugs such as that super fast spinning windmill in NaliBoat.
- Fixed for Linux XMesaGLDrv in combination with XLaunch and XDrv. Although it stays unmaintained in general I found it reasonable to make it at least working, since new renderer versions can now be created with XDrv instead of SDLDrv also.
- Fixed a memory leak in UnrealScript struct compare.
- Fixed UED2 exporting Fire/Water/Ice Textures crashes UED. Now exporting a stillshot.
- Fixed UED2 "Export to PCX..." only accepts now non DXT Textures instead exporting a 0x0 sized invalid texture/file.
- Fixed Linux version could crash if some masterserver was failing (both Server and Client)
- Fixed Linux version crash when trying to open an URL without any network cards available in system.
- Fixed UPAK (RTNP) Grenade Launcher flies "through" wooden boxes etc.
- Fixed some UPAK (RTNP) replication problems, but couldn't fix entirely because of backwards compatibility with Unreal Gold.

- Fixed UED2 "Reset Pivot" to really reset it when pressed and not later when Brush is reselected.
- Fixed UED2 Surface Properties and Build Properties pages are not refreshed after hiding/closing.
- Fixed Linux UCCLinux.bin didn't accept parameters correctly.
- Fixed some very rare happening crash bug with crippled messages.
- Fixed Galaxy to only log invalid sounds instead of crashing.
- Fixed actors getting beginplay events twice if spawned while level is being brought up.

.Version 227f

Release date: May 14, 2009

- Fixed: Dynamic array accessing bug: `Code: local array<string> SomeArr; SomeArr[0] = "Value 0";`
- Fixed: Bugs in banning system and allowing IP-range banning.
- Fixed: additional crashes from AI navigation on maps with broken path network.
- Fixed: TeamChat messages to show sender player name aswell as show the message in yellow rather than white.
- Fixed: Volumetric lighting to not mess up when being "overused" (as example with Aura weapons).
- Fixed: Script warning that Fly and SinglePlayer(game) would sometimes cause.
- Changed GameRules to have different flags for different functions to be called on it (in order to save server resources).
- Upgraded banning/temp banning/client logger to be based on a dynamic array to support unlimited amount rather than some static amount (such as max 1024 log entires).
- Fixed banning so it's possible to ban older clients by IP address aswell as it supports IP range banning (i.e: 1.2.0.0-1.3.255.255).
- Changed music menu musics list and favorites server list in order to be based on dynamic arrays, as well for same reasons as above.
- Added: Changed music menu file browser; when you double click the music file, it will automatly load the package and search for musics in them.
- Fixed bug with player skin selection menus, when you have skeletal mesh skins and switch between 2 that have different animation sets, their animation sets could go wrong.
- Fixed UED2: OpenGL selection problems and S3TC support.
- Fixed UED2: Select surfaces by group in.
- Fixed UED2: Red selection brush and movers can't be seen through walls in OpenGL.
- Fixed UED2: Tooltips not working.
- Fixed: If ghosting out of a mover (such as nali boat) you were still "bound" to it.
- Fixed Linux bug: new resolution setting was not saved in ini when using ubrowser console or setres command.
- Fixed: Some MasterServer issue with "Unknown Error Processing Port".

- Fixed: Old console causing "out of memory error".
- Fixed: Squid not attacking correctly.
- Fixed: UED2: D3D in does not show Zoning or BSPCuts.
- Fixed: ANIM NOTIFY for skeletal meshes.
- Fixed: Linux client crashing when disconnecting from server.
- Fixed: D3D8, D3D9 and OpenGL are not updating mover lighting (f.e. in NaliBoat).
- Fixed: OccludeBSP in-game crashes in very large maps.
- Fixed: feature: LightEffect LE_Spotlight and LE_StaticSpot lits up meshes correctly now.
- Fixed: WarpZones from generating odd errors in offline/online games when walking through them.
- Fixed Linux: lightmap bug.
- Fixed Linux: version crashing if wet/ice/scripted textures were made with S3TC textures.
- Added new option for renderers: **FullMeshRendering (True/False)** - This solves the problem with disappearing meshes (like Trees if you are very close) - could be a heavy performance drain on low end machines, so it's optional.
- Dynamic Array support has been improved: Added dynamic array length accessing functions:

```
native(640) static final function int GetArraySize( ArrayProperty
ArProp );
native(641) static final function bool InsertArrayIdx( ArrayProperty
ArProp, int Offset, optional int Count );
native(642) static final function bool RemoveArrayIdx( ArrayProperty
ArProp, int Offset, optional int Count );
```

- Added: server downloader's info is automatically broadcasted to all admins (+ all other players if enabled).
- Added: for GameRules modifier to desire whatever if client is allowed to download some file off server or not. Can be used like this:

```
function TestFunction()
{
     local array<int> TestArr;

Log(GetArraySize(ArrayProperty'MyMod.MyClass.TestFunction.TestArr'));
}
```

- Added 2 new commandlets:
  - **Editor.StripSource** - strips the text buffers from script packages.
  - **Editor.DumpInt** - generates a *.int file out of some desired package.
- Added UED2: "OpenGL" into selection menu for rendering devices.
- Added: new menu "Admin menu" which lets server admins get an easier control over kicking/banning/listing.
- Added: **"Grab"**, **"AdminMenu"** and **"Toggle Behindview"** keys in "configure keys" menus.
- Added: Servers now report server OS on serverinfo (Windows/Linux).

- Added: support for ALAudio/FMod to set Ogg music looping points.
- Added: UWindow menu kick message support (using clientmessage and message type 'Networking').
- Added: math functions for Coords (for advanced rotation support).
- Added UScript preprocessor (Details: UE1PreProcessorCommandlet) In order to use preprocessor you have to call ucc with following parameters: ucc `uengineppc.parse project=[<project_dir>/<project_file>] [-option...] [-globals...]`
- Added: distance fog support for mappers to use (you can simply define in ZoneInfo distance fog start/end distance and color).
- Added: new page to the setup wizard which allows to chose the sound device at first start (OpenAL/FMod/Galaxy).
- Added: "Change audio device" button to Unreal recovery mode (for windows), just in case some audio device causes trouble.
- Added **bSoundAttenuate [True/False]**: Attenuate Soundsources which are behind walls etc. for OpenAL.
- Added Dynamic Corona: a corona which can be spawned/moved/deleted in game aswell as has some additional properties (ideas from UT2004 such as directional corona, close/ranged distance colors, max size).
- Added Sunlight Corona: another corona but it renders as sunlight corona, the direction of actor defines sun position in sky (only renders when you have direct contact with the sky. As in addition you can add lensflares with it.
- Added **ZoneInfo->ZoneTimeDilation**: lets you slow down/speed up the actors locally within 1 zone only.
- Added: A new query method has been implemented to speed up serverbrowsing when multiple masterservers are used.
- Added: new admin command **uknownnames**: prints out known and used names from players currently on server since some string length limitation its not able to use it for ingame admin purposes (except directly on the server console), but for mods.
- Added **DynamicZoneInfo** actor, a zoneinfo actor which can be spawned/destroyed/moved in game. It works using simple zone shapes. This will allow mod authors to make some summonable lava/water blocks or mappers to make some rising water effect easy.
- Added Master of the Woods also known as Woodruff. Small addition to have some alternative for the bandage. It's a small and fast growing plant with +5 healing.
- Added new UnrealScript native function AppSeconds, usage is as follows:
  - `native(643) Final Function AppSeconds(Out Float OutTime);`
    Returns how long the application has been running in seconds, ie 1.1 = 1.1 seconds. This can be useful f.e. for server uptime.
  - `native(197) static final function float Acos ( float A );`
    The missing Acos function, also known as "ArcCos".
  - `native(126) static final function int InStr ( coerce string S, coerce string t , optional int Start );`
    If Start is not defined, less or equal 0 or greater then number of characters in S, then standard InStr is performed (if the string T is found inside S, the number of characters in S before the first occurance

of T is returned). If Start is greater then 0 and less then number of characters in S then function will try to find T in S after first Start characters. If function fails to find T in S, then -1 is returned. It will not break backward compatibility.

- `native(1718) final function bool AddToPackagesMap( optional string PackageName, optional bool bSkipSecurityCheck );`
  Adds some package to sandbox (server packages) temporarly for current map only (very much alike UT2004' one).
- `native(1719) final function bool IsInPackageMap( optional string PackageName );`
  Check if some package is current available in sandbox (server packages).
- `native(1720) final function vector GetVertexPos( int iVert, bool bAnimatedFrame );`
  Return world position of a single vertex on meshed actor.
- `native(1721) final function int GetVertexCount();`
  Return the amount of vertexes in single mesh actor.
- `native(1722) final iterator function AllFrameVerts( out vector iVertex, bool bAnimatedFrame );`
  Iterate through whole frame of vertexes in a mesh actor.
- `native(1723) final function int GetClosestVertex( vector CheckPos, bool bAnimatedFrame, out vector ResultVert );`
  Get the closest vertex to some point in world position.
- `native(1724) final function int GetBestTraceLineVertex( float MinDot, vector CheckPos, vector CheckDir, bool bAnimatedFrame, out vector ResultVert );`
  Check for best vertex result in line check.
- `native(1725) final function bool MeshTrace( vector Start, vector End, out vector HitNormal, out vector HitLocation );`
  Perform a 3D mesh line check, returns False if did not hit.
- `native(636) static final iterator function AllLinkers( out name PackageName, out string FileName, out string GUID, out int NmCount, out int ImpCount, out int ExpCount, out int FileSize );`
  Iterate through all linkers and get some general information about the packages (very much alike **ObjLinkers** but in a faster method).
- `native(637) static final function Object GetDefaultObject( Class<Object> ObjClass );`
  Get the default object of some object/actor.
- `native(635) static final function int Ceil( float f );`
  Returns f rounded off to the next higher whole number.
- `native static final function bool Divide( coerce string Src, string Divider, out string LeftPart, out string RightPart );`
  Divides a string and returns the two parts.
- `native static final function bool ExtractString(string Src, string LeftDivider, string RightDivider, out string MidString, optional bool bAdvanced);`
  Returns a string from between Left and Right Divider:
  - if bAdvanced is false:

```
Src = string1(string2)string3
LeftDivide = (RightDivider = )
result MidString = string2
```

- if bAdvanced is true it takes occurance of LeftDivider into account:

```
Src = string1(string2(string3)string4)string5
LeftDivide = (RightDivider = )
result MidString = string2(string3)string4
```

- `native(257) static final function bool LoadPackageContents( string PackageName, Class<Object> ListType, out array<Object> PckContents );`
  Load an enitre package and give out the contents of it (returns false if package failed to load).

.Version 227e

Release date: July 19, 2008

- Fixed: OpenAL in Linux version
- Fixed: Loading savegames in Linux version crashed game.
- Fixed: UBrowser Menu (classic) for Linux
- Fixed: Galaxy reverb crashes.
- Fixed: bleeding damage ignoring settings
- Fixed: Slow download speed for server without redirect.
- Added: New option **AllowFastDownload**: if true, the D/L speed is like older servers, if false, the speed is reduced like pre-227e servers for low bandwidth connection servers.
- Added: Built-in ban manager has been changed to save all bans in a new file called "Security.ini"
- Fixed: Last jump causing a "ghost" jump in online games.
- Fixed: Support with multiple jumpboots (custom mods).
- Fixed: Broken shield effect on online games.
- Fixed UED2: Crashed when using MyLevel.
- Fixed UED2: Crashed when closing it.
- Fixed UED2: Group browser checkbox was inverted
- Fixed UED2: After you load script with longer name, with first letter of script with name you are trying to load you CAN'T load it (for example, you cant loading Male after loading MaleOne)
- Added: Support for playing special Nali/Male 1/Female 2 skins.
- Added: Support for importing and playing Ogg Vorbis music files (in editor on Music Browser or in game as map music). (For OpenAL and FMod use only!)
- Added: FMod: ALAudio alike "AStat Audio" and "AStat Detail" commands. Fixed the reverb issue with maps like QueenEnd.
- Added: New option to enable Sound Attenuate (sounds that are played behind walls are heard in lower volume).
- Added: IP/Name/ID playerlogger for Servers and messaging it for server admins/server log which will be stored in *Security.ini* too if **GameInfo**

bMessageAdminsAliases/bLogNewPlayerAliases are true. Like the banning system, this feature is only available for 227 clients.
- Added Web-Administration support with administration page. Read the included UWebReadme.txt for more info. Features:
    - Restart map - Restart currently playing map.
    - Switch map - Switch map to selected map/game/mutators (note that maplists configure must be enabled to access maplist).
    - Current game - You see list of players (ID/Name/Ping/Score/IP) and controls to Kick/Ban them.
    - Server console - See a list of chat messages in server aswell a command line where you can chat with the players to execute a command on server.
    - Banlist - See a full list of all banned clients, also lets you unban them.
    - Defaults - Server configure page:
    - Main Game config - Lets you configure basic game rules (such as max players/server packages/server actors/redirecting etc).
    - ServerInfo config - Lets you configure public server rules (such as server name/admin name/MOTD).
    - Maplists - lets you configure maplists (for DeathMatch based games, NOT for Coop).
    - Mod Configures - Custom mods configures page (by default it contains configures for DeathMatch, TeamGame, Cooperative).
    - Changed to little better interface and added a config option for the new 227e AllowFastDownload flag.
- Added **OverridePrelogin** function for GameRules to modify prelogin allowance or something (its called directly after GameInfo Prelogin).
- Added new class **PlayerClassManager** which allows custom mods to temporarly adding in to player settings some new player classes/skins for that game only.
- Added in Actor a new variable bNetNotify and a new event PostNetReceive, can be used by mod authors for notifing whenever a replicated variable has been changed on client (whenever bNetNotify is True).
- Added in UBrowser a config option to set initial startup page (Advanced Options > Networking > UBrowser > InitialPage).
- Added some new native functions to enhance Uscripting:
    - Object:
        - `native final function Object FindObject( Class ObjClass, string ObjectName );`
          Finds an object based on object name/class.
        - `native final function Class GetParentClass( Class ObjClass );`
          Get Parent class of a desired class.
        - `native final iterator function AllObjects( class BaseClass, out Object Obj );`
          Iterate through all objects in game.
        - `native final iterator function AllFiles( string FileExtension, string FilePrefix, out string FileName );`
          Iterate through all Unreal Package files (u, umx, utx, uax...).

- Actor:
  - `native final function Actor SpawnAct( Class ActClass, vector Loc, optional rotator Rota, optional name ActName, optional Actor Own, optional Pawn Instigat, optional Actor Template, optional bool bMayColFail );`
    Spawn an actor class with some additional parameters.
  - `native(1722) final function bool CanReachPoint( vector Start, vector End, float ColRadius, float ColHeight, float JumpZ, float XYSpeed );`
    A point reachability function for AI use.
  - `static native final function NativeExec( string Cmd );`
    Pretty much same as 'ConsoleCommand' except it's static function.
- LevelInfo:
  - `static native final function byte GetConState( NetConnection Other );`
    Get connection state out of a net connection.
  - `static native final function string GetConIP( NetConnection Other, out int Port )`
    Get net connection IP aswell as Port.
  - `static native final function string GetConOpts( NetConnection Other );`
    Get net connection options (?Name=(Leader)-Dante?Class=UnrealShare.Male3?etc...).
  - `native final function bool HasDownloaders();`
    Fast check if current game has some downloaders.
  - `native final iterator function AllConnections( out NetConnection Connect );`
    Iterate through all connections currently on server.
  - `native final iterator function AllDownloaders( out NetConnection Connect, out string File, out int Sent, out int TotalSz );`
    Iterate through all downloaders on server.
  - `native final function PointRegion GetLocZone( vector Pos );`
    Get the zone out of a desired location.
  - `native final function Object AllocateObj( Class ObjClass );`
    `native final function FreeObject( Object Obj );`
    For storing temporary objects/actors that can be reused later.
- NavigationPoint:
  - `native final function int GenReachSpec( Actor Start, Actor End, int Dist, int ColR, int ColH, int RchFlgs, bool bPruned );`
    Generate reachspecs for current map.
  - `native final function bool EditReach( int Idx, optional Actor Start, optional Actor End, optional int Dist, optional ColR, optional int ColH, optional int RchFlgs, optional bool bPruned );`
    Edit an excisting reachspec in current map.

        - `native final function bool RemoveReachSpec( int Idx );`
          Remove a reachspec from the current map.
    - Canvas:
        - `native final function Draw2DLine( Color Col, vector Start, vector End );`
          Draw a 2D line on the screen.
        - `native final function Draw3DLine( Color Col, vector Start, vector End );`
          Draw a 3D line in world.
        - `native final function vector WorldToScreen( vector WorldPos );`
          Convert world coordinates to screen coordinates.
        - `native final function vector ScreenToWorld( vector ScreenPos );`
          Convert screen coordinates to world coordinates.
        - `native final function DrawPathNetwork( bool bOnlyWalkable );`
          Render current map's path network (with 3D lines).
        - `native final function coords GetCameraCoords();`
          Get the current camera location and rotation.
- Added new command for server admins: **admin UGetConnections**, shows current connection IPs, player names, and downloading status.
- Added: new emitter particle system (Emitter.u/Emitter.dll) which is handled 99 % in C++ scripts (giving a huge speed boost), with realtime preview support in Editor. For full features list, check Particle Emitter page.

.Version 227d

Release date: April 5, 2008

- Fixed: Slithprojectile not vanishing on dedicated servers.
- Fixed: Spawning many Skaarj Player Bots may crash *Unreal*.
- Fixed: Server wandering port - essential fix for server admin.
- Fixed: DmRetrospective not showing gibs.
- Fixed: Inventory HUD bug, charge being drawn from one side to another.
- Fixed: Quadshot pickup sound and switch to other weapon if 0 ammo. Not being able to select/deselect when 0 ammo but still ammo loaded in the barrels, size adjusted.
- Fixed: Black screen bug caused by division by 0, an obscure bug, will help a few mods.
- Fixed: Client not directly connecting after downloading a map.
- Added: Changed console behavior to display messages without calling it once first.
- Added: Translator message for hints.

.Version 227c

Release date: March 30, 2008

- Fixed: Quadshot accessed nones.
- Fixed: Player "fell out of the world!" when walking over decorations.
- Fixed: SoftwareRendering didn't work in fullscreen mode.
- Fixed: Coop server item respawn bug.
- Fixed: Can't find ByteProperty'VoicePitch'.
- Fixed: Vortex2 Bug (Playerstart).
- Fixed: Sound pitch changes.
- Fixed: ExtremeLab tubeportals (warpzones) are too slow.
- Fixed: OldWeapons.u didnt make it into 227b although promised (sorry for that one).
- Fixed: Widescreen support is not being applied online.
- Fixed: UMenu HUD no preview.
- Fixed: Error handler not working.
- Fixed: UMenu - Playersetup - Teamcolor wasn't read out of user settings.
- Fixed: Not being able to see PreLogin error on Mid-Screen (just as server is at capacity).
- Fixed: Player spidering movement issues.
- Fixed: Player spidering animations to better one.
- Fixed: NaliPlayer to use UT'99 NaliPlayer model (one which has weapon triangle).
- Fixed Linux: shift key not working by default in (ini setting).
- Fixed: Possible CoopGame server crash when monster is attacking a player while leaving.
- Fixed: Bug with Skaarj not behaving correctly in vortex2.
- Fixed Linux: SDLGLDrv for Linux. Although may be a bad idea to use it for playing (it lacks many features and is really not very far advanced), it may be useful for debugging, or for some low-end systems. In order to use it, you need to start *Unreal* with "`./UnrealLinux.bin -noforcesdldrv`".
- Fixed Linux: UBrowser select input keys not working.
- Fixed Linux and added SDLSoftDrv - now there are OpenGL and Software rendering for Linux versions.
- Fixed: UWindow Win95 LookAndFeel having some rendering errors (with menu backgrounds and buttons).
- Fixed: Some problems with older clients and mods.
- Added: Never switch on pickup option, so client can chose whatever they automatly switch to best weapon on pickup.
- Fixed: TraverseForm crash (on some maps).
- Fixed: OpenAL some sounds not playing correctly.
- Fixed: Some weird warpzones crashing Unreal when playing online.
- Added: EngineSubVer variable to LevelInfo to define version number (1=a, 2=b, 3=c).
- Added: ServerQuery to add EngineSubVer for GameVersion number (227 + a/b/c/d etc..).
- Added: Older 227 clients (227a, 227b) can't join newer 227 version servers

(227c). It's a precaution because this will crash the older client. Unfortunately there is no way to fix this problem without losing compatibility with older 224/225/UGold clients, so its out of question.
- Fixed: Removed "SpawnPlayActor" and "Incoming travelling actor" logging.
- Fixed: Console set to ";" key in Linux.
- Fixed: UMenu player class selection mesh scale - adjusted to work for NaliPlayer as well.
- Fixed: Another few server fixes for reported crashes (details of these wouldn't help anyone here :))
- Added: Enhanced the UWindow EditBox to make it possible to select text with mouse and Copy/Paste text in it.
- Added: a Music Player Menu (now default key to open it on F8), which lets you listen to custom musics while playing. Be aware that many anti-cheat mods may not like the custom musics!
- Added: On UMenu.UnrealConsole, you can enable UWindow debug mode (Tools > Debug mode) to debug your own menus while making them.
- Added: UMenu: Win 95 "LookAndFeel" as a selectable GUI interface option now.

.Version 227a (initial) and v227b

Release dates: December 26, 2007 (227a) and February 10, 2008 (227b)

- A native Linux port is now availible, which runs with OpenGL and SDLSoftDriver for grafix and FMOD for sound. Built in SuSE, works on LFS as well, but needs glibc2.3.
- The OpenGL, D3D8 and D3D9 renderers have been completely reworked and heavily improved with the files from [1]. These renderers support S3TC Textures and make Unreal compatible with most recent grafix cards.
- New Sound Drivers OpenAL and FMOD:
  - OpenAL - Provides hardware support for recent soundcards (best for creative cards), EFX (EAX) reverb soundeffects and full support for multispeaker setups.
  - FMOD - Provides hardware support for recent soundcards, original Unreal reverb soundeffects and full support full support for multispeaker setups.
- Added commands for banning and kickbanning:
  - **uhelp:** Prints the explanations below
  - **uplayers:** shows for all players the name, ID, IP-Address, IdentNr and Identity
  - **ukickid:** kicks a player with a given ID
  - **ubanid:** kicks and bans a player with a given ID (full ban by IP and Name, even after a restart of the game/server)
  - **ubanlist:** shows a list with all banned players
  - **uunban:** unbans a player with the number X (see in banlist for the ban-number)
  - **utempbanid:** kicks and bans a player until server is restarted
  - **utempbanlist:** current list of temp-banned players
  - **utempunban:** unbans a tempbanned player with the number X (see in

tempbanlist for the ban-number)
    - **utempunbanall:** unbans all tempbanned players
- Added: Support for footstepsounds and footprints. However, in order to use these new features, the texture properties need to be changed; there are now 4 slots for different sounds (FootStepSound) the texture produces when walking on, and one variable (Footprint) which defines if the texture should display (or not) a footprint when walking on.
- Added UI-FX - Particle Emitters, Weather Simulator, Vegetation Generator and many things more. Details can be found in the forums.
- Added support for skeletal meshes like *UT*.
- Added support for HTTP-redirect for map downloading, like in *UT*.
- Widescreen users can now adjust the FOV in order to adjust *Unreal* better for their screens (Maybe some mods will override this setting).
- New setting in NetDrv: **AllowOlderClients True/False**. If set to True, 224, 225 and UGold Clients can join the server, if False, they will get the Upgrade Message. Either way, there's a problem: since the upgrade URL is defined in UpgradeMenu.uc, old clients will be redirected to the old dead upgrade page, and there's nothing I can do about that.
- Added Coop options for easy configuration of Weapon, Items, Flares and Seeds spawn time:
    - **bInstantWeaponRespawn**
    - **bInstantItemRespawn**
    - **bHighFlareAndSeedRespawn**
    - FlareAndSeedRespawnTime
- New and fully working QuadShot, which replaces the old unusable and unfinished quadshot.
- Decals have been added to all *Unreal* weapons (including the monster weapons): Decals can be added to custom mods as well, but can be turned off if wanted. All weapon decals (or subclasses of Scorch) have a configurable variable called **DecalLifeSpan**. This variable affects ONLY clients, servers need not apply. This is located under "Decals" then "Lifespan" in preferencesDefault is -1.
    - -1 = default behavior, the effects will disappear upon not being rendered for some amount of time.
    - 0 = infinite. Decals will NEVER disappear. use at your own risk.
    - >0 = time in seconds, 1.1 is 1.1 seconds. 30 is 30 seconds.
- Added support for new difficulty levels (up to 6) for Coop & Singleplayer: they needs to be set with the commandline parameter **?difficulty=X** (X=4, 5 or 6)
- New parameter for commandline: -timestamplog, forces *Unreal* to put a Timestamp after the logfile name, so if you would simply start Unreal with -timestamplog, the logfile will be *UnrealYear_Month_Day_Hour_Minute_Second.log* If you start it up with **-log=server.log -timestamplog**, the logfile will be: *ServerYear_Month_Day_Hour_Minute_Second.log*.
- New Blood effects: Gibs now have blood impact decals. If enabled on client, and the server is 227, dynamic blood splatter will occur on all standard (and most mods) pawns/carcasses. If enabled on server, bleeding will occur, damage is optional. Blood pools will spawn for killed pawns.

- Added 2 new engine consolecommands:
  - GameInfo.ConsoleCommand **GetPreLoginAddress**, can be used during Event Prelogin call for retreiving the connecting client's IP
  - PlayerPawn.ConsoleCommand **UGetIP**, smilar to GetPing but returns instead the Client IP.
- Added new parameter in ZoneInfo: If a new damage type and **zoneDamageString** is supplied in a damaging zone, players will now receive that death message.
- Updated UBrowser Server browser: Bigger default vertical size of that window to show full info of all servers (servername, players, ping, IP, port etc...), without having to resize it.
- Added server version number, which appears on the server list like *UT*.
- Added "Join with password" button option, when you right click at a server from the list. This window has an ability to save the password so next time you normally connect to that server (through UBrowser) it automatically uses that password. When connecting to an "unknown" server (one where you haven't inputted any password for) it will join with a randomly generated password to protect you from password stealing.
- Added **U227GameRules** class: can enhance serverside mutators, such as modifying player spawn point, modifying damage, preventing deaths, blocking/modifying chat messages etc... Note that some of these functions may **not** work with some custom game types that modifies the game. Also using this class will force your mod to be for 227 use only, so try to keep it in use only on server (a nondownloadable package) so older clients can still join the server aswell.
- Added new JumpPad class, UJumpPad.
- Added new setting for **Mousehandling** to enhance precission with high-resolution mice: Mouse smoothing can be turned off.
- 2 new menus are available: Unreal classic-style and new UMenu (Unreal Gold) style.
- Naliplayer can now be played ingame.
- New implementation allows to use localized chars such as "ö,ä,ü" or other language specific symbols while using say.
- UnrealED 2 used for map editing.

.UnrealEd 2.1 notes

As some of you maybe know, 227 contains UED2.1 - many fixes have been made, improvements were added, 227 features included and some missing functions out of UED1 have been reintroduced, it was about time to give it a name.

Still I'm curious if there are some things you are missing in UED or if you are aware about some bugs which need to be fixed yet. Of course its not possible to add every wish, but maybe some things can be still realized.

Here is a list of whats fixed and new in UED2.1:

.Fixes

1. Fixed crashed when using MyLevel
2. Fixed crashed when closing it
3. Fixed group browser checkbox was inverted
4. Fixed after loading a script with longer name, you can't load another script with the same first letter(s) in name
5. Fixed OpenGL selection problems and S3TC support
6. Fixed select surfaces by group
7. Fixed for red selection brush and movers can't be seen through walls in OpenGL
8. Fixed Tooltips not working
9. Fixed D3D does not show Zoning or BSPCuts
10. Fixed Framecount in Meshviewer
11. Fixed can't delete Textures
12. changed a crash to a warning message which happens in some rare occasions with tesselated cubes. See: http://www.oldunreal.com/cgi-bin/yabb2/YaBB.pl?num=1251918599
13. Fixed crash when selecting "Export" in Texture Browser while no Texture is selected.
14. Fixed water/iced/scripted textures not working with DXT textures (not directly UED related but important for mapping maybe)
15. Fixed changes in Texture properties not automatically visible when using OpenGL or D3D8/9
16. Fixed crash on exit when using Menu->File->Exit
17. Fixed Replace Texture tool can't be re-opened anymore after one time usage.
18. Fixed 2D Shape Editor. For buttons: Revolved Shape, Extruded Shape, Extrude to Point, Extrude to Bevel not accessible anymore after closing, fixed shape vanishing after moving with right mouse button pressed over window borders. Also fixed "Extude" typo for to Point and to Bevel
19. Fixed bug which caused random shadows appear in a map when using BrightCorners a lot.
20. Fixed display of fonts in TextureBrowser by group selection
21. Fixed selection in D3D9. Thx to SMPDev (UTGLR) for this update.
22. Fixed that a couple of windows were blocking UED2 entirely if closing them with "x" (close button) in the right upper corner
23. Fixed context menu -> View -> Show Coordinates to show current camera position
24. Fixed context menu -> View -> Show Backdrop to really show skyboxes without actually being in realtime mode.
25. Fixed BSP building to reduce the number of BSP holes very much.
26. Fixed crash with brush builders after map import.

27. Fixed crash when trying to view Upak.RL3RD mesh in the mesh browser.
28. Fixed exporting Fire/Water/Ice Textures crashes UED. Now exporting a stillshot.
29. Fixed "Export to PCX..." only accepts now non DXT Textures instead exporting a 0x0 sized invalid texture/file.
30. Fixed "Reset Pivot" to really reset it when pressed and not later when Brush is reselected.
31. Fixed Surface Properties and Build Properties pages are not refreshed after hiding/closing.
32. Fixed clip brush causing slight imprecision in coordinates (float accuracy problems).
33. Fixed snapped brush scaling sometimes not snapping anymore with changed pivot
34. Fixed snapped brush scaling overflow when scaling "smaller than possible"
35. Fixed actors visible in level through wall when used in skybox (and not only in fakebackdrop)
36. Fixed "Save Brush as" / "Open Brush" (NOT import/export) function. Stores position of the brush as well as texture information.
37. Fixed StaticMesh behavior in front of FakeBackdrop surfaces (Z issue).
38. Fixed RightMouseButton on Surface -> Reset (was entirely nonfunctional in UED2 and 2.1, UT and Unreal version).
39. Fixed 2DShapeEditor shows the selected GridSize in Menu.
40. Fixed importing sounds didn't accept spaces in pathname.
41. Fixed "Replace with..." was setting bRemoteOwned flag causing some actors behaving strange.
42. Fixed import for 32bpp pcx, crashed before (but still suggest using bmp instead for that).
43. Fixed "Select Inside" was selecting static mesh movers outside.
44. Fixed brush clipping resulted sometimes in brushes with vertex rounding errors.
45. Fixed undo to work with vertex editing.
46. Fixed order of how things are build when doing "Build all", which seems to make a significant difference in reducing holes.
47. Fixed Transform permanently to keep U/V mapping perfectly intact

.Additions

Added "OpenGL" , "Direct3D8" and "Direct3D9" into selection menu for rendering devices(Please note that selecting is only working correctly in SoftDrv,OpenGL and D3D9, while some buggy ATI drivers don't even work with OpenGL too, but still it can be used to have a look how things look in game then)

1. Added to new menu items in context menu (Right Mouse Button): Align to

wall around X axis and Align to wall around Y axis to fix alignment problems like in: http://www.oldunreal.com/cgi-bin/yabb2/YaBB.pl?num=1249683850
2. Added missing "Texture Browser" to view menu
3. Added EAX ambient presets for use with OpenAL (already known out of the OMP package)
4. Added a new option into Editor section of Unreal ini. FreeMeshView: when enabled you can fly around in mesh viewer the same way like in 3D view (no fixed positioning anymore)
5. Added a "remove script" function to classes browser and changed this HIGHLY ANNOYING behavior that it shrinks the menu down to "actor" when adding a new class. It now stays where the new class was added.
6. implemented DXT3/DXT5 support and the necessary import additions. Any format can be chosen during import directly in UED now. Needs 32bpp bmp.
7. implemented AlphaBlending for DXT3/5 based textures. Can be used instead of simple masking and is selected with a new checkbox during import.
8. Added support and preview for native emitter usage and STY_AlphaBlend so you can make f.e. fade in / fade out on emitter effects such as smoke. Also contains a weather emitter for rain and snow and many more things.
9. Added support and preview for skeletal meshes, show naming of bones, bonescaling and much more
10. Added "Do you really want to compile all scripts?" dialog to avoid recompiling everything (which needs a lot of time) when clicking "Compile All" in class editor by accident
11. Added functionality for surface flag "Environment" on BSP surfaces to add an environment mapped texture overlay on the surface.
12. Added a new actor called "FluidSurfaceInfo" for creating a wavy water mesh for water surface.
13. Added a Texture property 'PaletteTransform' to allow change palette color range to that desired color (useful for changing FireTexture color instead of having to import some dummy palette texture).
14. Added a new actor flag "bWorldGeometry" to make actor be threated as part of world geometry (block visibility sight and stop explosion radius etc...).
15. Added Select All Actors and Select Inside Actors buttons. These buttons were already available in UED1 but disappeared in UED2 for unknown reason.
16. out of UED1 re-integrated "Small Diagonal" and Big Diagonal" to rotation.
17. Added to UED2 the brush manipulation buttons known from UED1: Sheer, Scale and Stretch (while the in UED2 already existing Scale button was in UED1 "SnapScale" and to explain its function correctly it is renamed now accordingly like in UED1).
18. Added AllTexturesInUse Button for TextureBrowser which makes the Browser show only the textures of a package which are used in a map .
19. Added Next Frame and Previous Frame buttons in MeshBrowser to browse

through the mesh frame by frame.
20. Added DDS import with and without mipmaps.
21. Added for Meshes on right mouse button dialog "Add ... here" functionality.
22. Added support for static Meshes.
23. Added support for Export/Import of static Meshes or mesh frames.
24. Added support for renaming meshes.
25. Added UED1 feature for disabling Grid and custom Grid size.
26. Added support for mesh movers into right mouse button menu of the mover button.
27. Added support for opening and saving of static mesh packages.
28. Added open/save buttons for opening and saving static mesh packages (.usm).
29. Added ability to import Meshes from .obj (WaveFront) files.
30. Added support for fog fade in times.
31. Added a bool for bAlwaysRender, should finally fix any mesh disappearing especially with huge static meshes for terrains in maps. Can decrease performance significantly if overused.
32. Added missing Load/Save Package button in MeshBrowser.
33. Added Load Entire Package button in Texture, Sound and MeshBrowser
34. Added "Lighting only" mode for 3D viewport.
35. Added "Export to bmp" which can export DXT Textures as 24bpp bmp as well.
36. Added "Are you sure you want to save new filesize kb, old filesize kb)" to save dialog to see and avoid map/package corruption if new filesize is smaller than old.
37. Added EXPORTFONT/IMPORTFONT commandlet for fontworks,available in command (log) window
38. Updated map T3D exporter to write less redundant data.
39. Updated T3D importer to load up necessary texture packages used by BSP rather than slapping default-texture all over them (if not currently loaded on editor).
40. Added to map rebuilder to cleanup small map errors that might been caused from copy/pasting/undo/redo to avoid editor from freezing during build.
41. Added whenever a package/map is saved in editor it asks if you are sure that you want to overwrite old version if new one is smaller in filesize.
42. Added export and batchexport for vertex meshes
43. Added whenever package/map saving failed, it pops up a window rather than showing it in log only.
44. Added render device selection menu for TextureBrowser and MeshBrowser. Most people didn't know that it could be set up in Unreal.ini with "WindowedRenderDevice". Softdrv is in many cases not sufficient (like AlphaBlend Textures or for StaticMeshes) to display everything correctly and

now it can be easily switched between SoftDrv/D3D9 and OpenGL during runtime.

45. Moved parts of path building code from hardcoded C++ codes into UnrealScript, that includes the special paths for LiftExit/LiftCenter/Teleporter/InventorySpot etc...
46. Added 2 variables to NavigationPoint; ForcedPathSize and MaxPathDistance. ForcedPathSize will be the path "size" when using "ForcedPaths" list, so that mappers can limit smaller pawns to use them. MaxPathDistance is the maximum distance the pathnode will seek for path around itself.
47. Added a reset button for the brush builders.
48. Added real-time preview (for selected pathnodes) so that you can see how pathnodes will be bound with each other (it also draws a cylinder around selected pathnode so that you can see how far it can bind).
49. Added modified 227 JumpPad's to draw a yellow line preview of the throw trajectory. Also modified so AI paths won't get bound reverse up the jump pad path.
50. Added render device selection menu for TextureBrowser and MeshBrowser. Most people didn't know that it could be set up in Unreal.ini with "WindowedRenderDevice". Softdrv is in many cases not sufficient (like AlphaBlend Textures or for StaticMeshes) to display everything correctly and now it can be easily switched between SoftDrv/D3D9 and OpenGL during runtime.
51. Added 2DShapeEditor GridSize 128/256, asks to save when creating or opening new shape.
52. Added Modified light actor icons to be rendered in editor in the color of their light source (to show a small preview of their light color like in UE2+).
53. Added for meshes an option "ShadowMesh" which is used for computing shadows instead of the original mesh, so it is possible use specific meshes for more detailed shadows (useful for trees for example).
54. Added auto disable realtime preview when test playing map.
55. Added confirmation, asks to save when opening/loading from MRU/ exiting UED and a map with unsaved changes is still open. Also added "Cancel" to those dialog windows.
56. Added bDarkLight option for lights, which removes light instead making it.
57. Added "Align Viewport Cameras to 3D viewport" as right mouse button menu selection and button on the left side (misc section)
58. Added "Rebuild BSP only", so people can build Maps like in UED1 with every step separately if wanted.
59. Added "Batchexport to BMP from Package" and "Batchexport to PCX from Package" for textures
60. Added "Batchexport to WAV from Package" for sounds
61. Added for ActorBrowser "Find..." which prints out the location of a class

(example "Inventory Pickup Health" for Bandages). Should safe some time especially if digging for some class in unknown mod packages. Also selects the class found, for use in ClassEditor or to place into map, but does not expand it in the browser itself.

62. Added editor option (AlwaysPermanentBrush=True/False) to always transform all brushes permanently every time you scale or rotate them (pretty much like in UnrealEd 3+) to minimize floating point transformation imprecision in BSP.
63. Added export for all classes in MyLevel
64. Added a switch to disable "Are you sure you want to save new filesize kb, old filesize kb)" save dialog. [Editor.EditorEngine] AskSave=True
65. Added RightMouseButton on Surface -> Tessellate Surface (3D view surface context menu)

Note that some functions are specifically 227 dependent and maybe "ripped out of the context" a bit.