BUILDING CLASSICS

In terms of number of games released, Build is one of the most successful game engines ever, with over twelve games developed for it. Duncan Rule delves into the engine, its games, and the engine's creator Ken Silverman

When id Software unleashed the seminal *Doom* in 1993, it sent waves of truly colossal proportions though the world of PC gaming. From that point onwards, the question on the lips of every gaming publication and fan worldwide was "Which game will become the *Doom*-killer?" Over the following few years, many tried to build upon what id had done with *Doom* and create the "next big thing". Unsurprisingly, the majority of these were poorly-made, rushed efforts hoping to cash-in on *Doom*'s much-deserved success, and wound up as nothing more than distinctly bland bargain-bin-fillers destined to collect dust for the rest of time.

During this period, programmer Ken Silverman was busy creating what many would come to consider his magnum opus: the Build engine. Build promised to do everything *Doom*'s technology (now officially known as id Tech 1) did, and much, much more. Silverman's engine operated on many of *Doom*'s principles. Maps were still not fully three-dimensional (this wouldn't appear in an FPS until id released *Quake* in 1996), but rather constructed out of two-dimensional shapes known as "sectors". These sectors are given an added component which specifies a floor and ceiling height, and gives the impression of a three-dimensional (but rather limited) world when rendered. As with *Doom*, Build's world is then populated with two-dimensional sprites in order to add detail, objects and enemies to a game.

However, unlike id's baby, Build allowed for additional features such as sloping floors and ceilings, dynamically altered sector information (which made destructible environments possible) and, perhaps most importantly, overlapping sectors (provided only one sector could be viewed at any one time). This meant that designers could create areas that gave the illusion of the coveted "room-above-room" scenario, which was just not possible with a so-called "2.5D" engine. Sectors could also be given tags to perform functions such as teleporting players to other areas of the map. This could be used for traditional "teleporters" in games, but also more subtly to create the illusion of falling into a pit or diving underwater.

Build was one of the most-licensed engines in the history of gaming, and almost anyone who owned a PC in the mid-to-late '90s has encountered a game that used it at one point or another. The last game to officially use the engine appeared in 1999, but Silverman's work lives on today, in no small part thanks to a release of the source code back in June 2000. But enough about the engine itself; let's take a look at the games…

Duke Nukem 3D

Developer: 3D Realms
Publisher: Apogee Software
Year: 1996

Duke Nukem 3D is undoubtedly the most well known and well-loved of all the Build engine games, having been ported to pretty much every system available at the time, including a little known Brazilian Mega Drive port that looks more like *Wolfenstein 3D* than *Duke*. Created in 1996 by 3D Realms, Duke was the primary vehicle for Build, and the focus for much of the engine's development. *Duke Nukem 3D* picks up directly from where *Duke Nukem II* left off, and transplants Apogee's beloved platform-hopping hero into a brave new world of three-dimensional alien-blasting mayhem.

As with pretty much every FPS of its era, *Duke 3D* is light on the story and heavy on the action. Aliens are taking over planet earth and generally harassing good and honest folk (including a disturbingly large number of nude women), and it's up to the Duke to stick his foot up their collective ass and give 'em a taste of hot lead. That's about as complex as the plot gets here, and the rest of the game simply involves traversing cities, shops, strip clubs, bases, canyons and space stations, looking for increasingly huge guns to destroy increasingly nasty aliens with.

While it may seem like nothing more than *Doom* with sunglasses and a blonde buzz-cut, *Duke* delivers a hell of a lot more than its hellspawn-slaughtering predecessor. One of the most attractive aspects of the game, and unquestionably one which separated it from the dreck of the time, is its level design. Gone are the generic, nondescript mazes which defined the genre, and instead we're treated to all manner of convincing real-world locales to blast though. Comedy also plays an important role alongside the gibs and guns, and Duke's frequent one-liners have been firmly cemented into the annals of PC gaming. What's not to love about a game that lets you relieve your bladder into a variety of pixellated toilets in the midst of the action, or a protagonist who defecates down the neck of a recently deceased boss monster while reading the paper? Who could forget the infamous "Shake it, baby!", or a weapon that lets you shrink enemies down to pintsize proportions before stamping on them like a bug?

Duke Nukem 3D isn't ever going to win an award for sophistication. It's rude, crude, obnoxious, offensive and hilariously good fun. Plus it can now be had for next to nothing from GOG.com and in a new incarnation on Xbox Live Arcade. A true classic that everyone and... well, maybe not their mum... should play.

Blood

Developer: Monolith Productions
Publisher: GT Interactive
Year: 1997

Blood was never going to have an easy time of it. After all, it was a sprite-based game in a post-*Quake* world where everyone was too busy marvelling over new-fangled fully 3D engines to worry about the perfect atmosphere and wonderful design of what was seen as yet another Build game. But despite being rather less well known than *Duke* (for precisely these reasons), *Blood* has nevertheless managed to maintain a loyal cult following over the years, thanks in no small part to its utter uniqueness.

Superficially, *Blood* appears to be very similar to *Duke* in many respects, mainly thanks to the engine. However, *Blood* succeeded in taking the FPS genre to places it had never been before with its unusual setting and thematic elements. Taking place some time during the 1920s, *Blood* puts you in the boots of Caleb, an undead gunslinger hell-bent on exacting revenge on the dark god who slaughtered him in his first life. The game begins with the heavy stone lid sliding off your tomb as Caleb utters the words "I live...

again!", and from there on out it's just you against an army of zombies, cultists, gargoyles, spiders, and all manner of hideous occult beasties as you battle your way through four episodes of dark, gothic levels to reach your goal. It's certainly different from the usual "You are Sgt. Muscles McBadass; save the world!" approach of most FPS games.

Blood is a real treat for any fans of horror fiction, and the game is absolutely littered with references (some subtle and some not-so-subtle) to a myriad of horror movies and books. You'll hear quotes from *The Evil Dead*, find Zombies encased in metal barrels a la *Return of the Living Dead*, and even a rather frosty Jack Nicholson in the middle of a hedge maze. Unfortunately, *Quake*-mania meant that *Blood* never really received the attention it deserved (and no doubt would have received had it been released a year or two earlier), and the game today is reserved to cult status. The acquisition of Monolith by Infogrames (and subsequently Atari) seems to have destroyed the hope of any possible source code release in the future. Still, *Blood* is a great piece of gore-soaked fun, and a game well worth picking up if you can find it.

Shadow Warrior

Developer: 3D Realms
Publisher: GT Interactive
Year: 1997

3D Realms' second and final game to be made with the Build engine was *Shadow Warrior*; *Duke Nukem*'s slightly more refined, feature-laden Japanese cousin. You play as Lo Wang (pun intended); a master assassin and stereotypically Asian Duke clone who is tasked with, you guessed it, saving the world (or at least Japan) from an evil corporation known as Zilla Enterprises.

If you loved *Duke*, you'll find plenty to like with *Shadow Warrior*, as it provides an extremely similar gaming experience. Wise-cracking protagonist? Check. A host of pop-culture references? Check. Lashings of tongue-in-cheek humour? Check. Plenty of big weapons to demolish the bad guys with? Oh yes! *Shadow Warrior* focuses on a Japanese/Samurai/Ninja/martial arts-type setting, and the locations, weapons and enemies are designed accordingly. You can hack baddies apart with a razor-sharp katana, pepper them with shurikens (unless you own the British version which replaces these with darts, thanks to our friends over at the BBFC), or batter them with your fists of fury in a pinch. There's also plenty of standard weaponry on offer, such as twin Uzis and a riot shotgun, along with the obligatory ridiculous instruments of death such as a dismembered head that shoots fire and a miniature nuclear missile.

Shadow Warrior also brought a few engine enhancements to the table such as voxel items, transparent water, and primitive drivable vehicles. It's probably not a game deserving of "classic" status, but it's the next logical step for anyone who's finished *Duke* and wanting more of the same—a solid old-style shooter that doesn't take itself too seriously. *Shadow Warrior*'s source code was also released in 2005, and there are now ports available for a variety of modern operating systems.

Redneck Rampage

Developer: Xatrix Entertainment
Publisher: Interplay
Year: 1997

Completing the "big four" Build game line-up is *Redneck Rampage*, a truly memorable FPS that delivers "all the killin', twice the humour [and] half the intelligence" of its contemporaries.

Like many of the Build engine games, *Redneck Rampage* manages to inject a surprising degree of freshness into a genre that was, at the time, almost entirely comprised of uninspired "clones". The action takes place in the (unfortunately fictional) backwoods town of Hickston, Arkansas, and centres on two brothers; Leonard (the player) and Bubba, as they attempt to retrieve their prized pig Bessie from aliens who've invaded Earth and dun swiped 'er. It's certainly one of the stupider plots out there, and the rest of the game matches it beautifully. Health and armour are replaced by junk food and alcohol respectively, and players are forced to strike a delicate balance with their consumption of each. Too much food and you risk alerting enemies to your presence with an unfortunately-timed fart as you creep up on them; too much alcohol and your view and controls become garbled as you try to deal with the effects of drunkenness. It's certainly a refreshing take on the tried-and-tested systems of old, and adds the need for some strategic item usage to the game. *Redneck Rampage* also introduces a novel way to end the level: locate your halfwit of a brother (who somehow manages to make it to the end of the stage miles ahead of you without killing a single enemy) and whack him in the face with your crowbar.

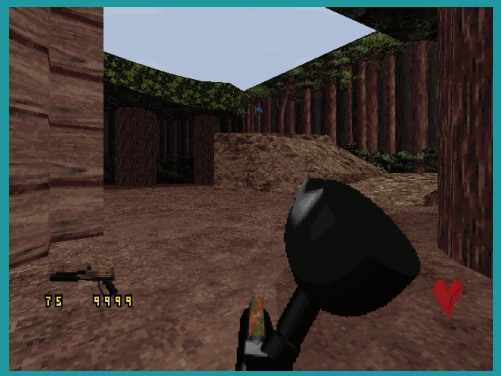
The enemies are an amusing bag of alien clones of the townsfolk (the Skinny Old Coot and his cries of "Git awf mah laynd!" being one of the game's most memorable moments), huge alien guards, dominatrix-styled "vixens" complete with machine-gun breasts, "turd minions" who pelt you with faeces, and an assortment of non-alien Deep South nasties such as vicious guard dogs and mosquitoes the size of a dinner plate. The weapon selection ranges from the tried-and-true pistol/shotgun/machine gun set through to more bizarre offering such as the vixens' machine-gun bra and a crossbow that fires sticks of dynamite (the ultimate redneck rocket launcher).

Redneck Rampage took the crude humour of *Duke 3D* and upped the ante considerably to never-before-seen levels of tastelessness. The end result is an insanely fun and memorable FPS, the likes of which we aren't likely to see again any time soon. The game did however receive an expansion and sequel (essentially a standalone expansion), along with a deer hunting game, *Redneck Deer Huntin'*, also based upon Build's technology.

Witchaven

Developer: Capstone Software
Publisher: IntraCorp Entertainment
Year: 1995

Witchaven was one of the earlier (pre-*Duke 3D*) Build games, and as a result was built upon a slightly less-refined version of the engine than its shotgun-toting descendent. With *Witchaven*, Capstone attempted to combine the fast paced action of the nascent FPS genre with elements associated with CRPGs (such as breakable weapons and experience points), all within a traditional fantasy setting. The resulting game is a bit of a mixed bag. The levels are pretty vast in comparison to *Duke*, but the unpolished feel of the game's presentation (such as enemies that appear to have been created initially as models while others were rendered entirely with software) combined with slippery, awkward controls make the whole experience a pretty forgettable one. Capstone released a sequel, *Witchaven II: Blood Vengeance*, a year later that adds a few extras such as human enemies, the ability to wield dual weapons, and a level editor. Unfortunately, the game underneath remains unchanged and the sequel failed to win over many (if any) new fans.



Extreme Paintbrawl

Developer: Creative Carnage
Publisher: Head Games
Year: 1998

The only thing extreme about *Extreme Paintbrawl* is its crappiness. As the title suggests, the game is an attempt at a paintball sim, which seems to be a somewhat odd choice of pastime to bother recreating in the electronic medium. It manages to fluctuate between two extremes of difficulty which combine to create one extra-rancid whole. The AI in *Extreme Paintbrawl* is, on the whole, god awful, and you'll often see both teammates and enemies stuck helplessly on bits of terrain as they attempt to throw strategy completely out of the window and fire blindly at you in a straight charge across the map. If you're lucky enough to encounter an enemy who isn't stuck on the scenery, you'll usually feel the wrath of their perfect aim as they hit you with the first shot they fire, ending the round. In addition, despite being a DOS game, *Extreme Paintbrawl* will only run in Windows thanks to an extremely out of place frontend (the only Build game with such a requirement). Oh, and it came in out 1998. This is budget software of the worst kind. Avoid, avoid, avoid.

Legend of the Seven Paladins

Developer: Accend, Inc.
Publisher: Accend, Inc.
Year: 1994

Probably the least known Build game on account of its extreme obscurity, *Legend of the Seven Paladins* was developed by a Chinese company who had been in talks with 3D Realms to license Silverman's technology. The deal fell through, but the team apparently pressed ahead with the game regardless. It was originally thought that *Legend of the Seven Paladins* only existed in demo form, but at least one retail copy has since been spotted on a Taiwanese auction site.

William Shatner's TekWar

Developer: Capstone Software
Publisher: IntraCorp Entertainment
Year: 1995

…or should that be *William Shatner's SteamingPile*? A strong contender for the worst Build game this side of *Extreme Paintbrawl*, *TekWar* is based upon Shatner's ghost-written book series of the same name. You play an ex-police agent turned hitman whose mission is to eliminate "tek" dealers. While the game is notable for utilizing a hub-based level system a la *Hexen*, and featuring non-hostile NPCs, the end product is an amazingly poor exercise in game design. Don't bother with this stinker, even if you're a Shatner fan.

Exhumed

Developer: Lobotomy Software
Publisher: BMG Interactive
Year: 1997

This curious entry into the FPS genre saw the light of day on the PC, PSX and Saturn at the time of release. However, while the Playstation and Saturn incarnations use a true 3D engine built specifically for the game, the PC version was inexplicably built upon Ken Silverman's engine instead. Known as *PowerSlave* in the US (an obvious nod to Iron Maiden's seminal 1984 album of the same name), the game follows the Build tradition of unorthodox setting and places the player in ancient Egyptian city of Karnak, complete with all the stereotypical monsters you'd expect. It's not a bad game by any means, but definitely not up there with the best of the Build alumni.

WWII GI

Developer: Team TNT
Publisher: GT Interactive
Year: 1999

In 1999 (would you believe), Team TNT followed up *NAM* with the similarly themed *WWII GI*, which this time had the player take the role of a soldier during the D-Day invasion of France. On the whole it's really not that different to its predecessor, save for the graphical style, and being released as late as '99 makes this one a good contender for one of the last ever commercially released DOS games.

NAM/Napalm

Developer: Team TNT
Publisher: GT Interactive
Year: 1998

NAM originally began life as a total conversion (TC) for *Duke Nukem 3D* known as "*Platoon TC*". Created by Team TNT of *Final Doom* fame, *NAM* places you in the combat boots of a GI during the Vietnam War. The game was released in 1998, and as a result seems even more dated than some of its brethren, as games like *Quake II*, *Half-Life* and *Unreal* were coming out around the same time. It does however appear to be one of the earliest war-based FPS games, and adds a few features such as friendly team-mates, airstrikes and booby traps that force the player to adopt a more strategic style of play. An interesting game doomed to obscurity from the start.

The lost Build games

There were also two in-development Build games that never saw the light of day. The first of these was known as *Fate*, and was developed by DogBone Software, a subsidiary of IntraCorp. Despite IntraCorp's less-than-stellar reputation, *Fate* did actually look to be a promising title and appeared much more polished than the majority of the company's earlier output. However, *Fate* (shown on the right) only made it to demo stages before IntraCorp went bankrupt and the game disappeared into the void for eternity.

The second "lost" Build game was interestingly enough also tied to IntraCorp, albeit this time coming from the studio behind *Witchaven*, Capstone Software. *Corridor 8: Galactic Wars* was a planned sequel to Capstone's earlier *Wolfenstein 3D* clone *Corridor 7: Alien Invasion*. The game disappeared as IntraCorp went under, and not even a demo managed to surface to the public. The game's source code, along with a prototype, was however released onto the net in 2005 by one of the original dev team.

KEN SILVERMAN INTERVIEW

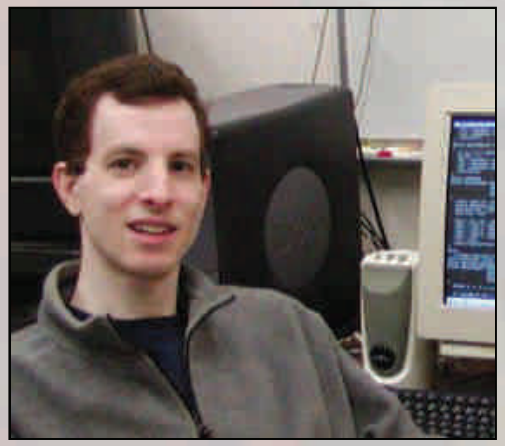
Duncan Rule recently had the chance to catch up with Build's creator, Ken Silverman, to chat about his time in the games industry, and his most famous creation

"Build" games that Ken was heavily involved with:

Duke Nukem 3D (3DRealms, GT Interactive)
Shadow Warrior (3DRealms, GT Interactive)
Blood (Monolith, GT Interactive)

"Build" games that Ken has done some kind of limited support for:

Exhumed - *a.k.a. Powerslave* (Lobotomy, Playmates Interactive Ent.)
Witchaven (Capstone, Intracorp)
Witchaven II: Blood Vengeance (Capstone, Intracorp)
TekWar (Capstone, Intracorp)
Fate (Capstone, Intracorp)
Redneck Rampage (Xatrix, Interplay)
Redneck Rampage Rides Again (Xatrix, Interplay)
Redneck Deer Huntin' (Xatrix, Interplay)

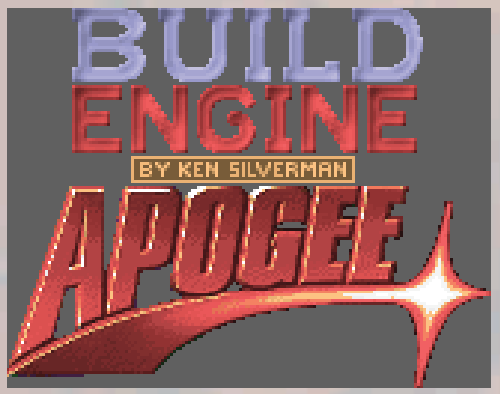


Duncan Rule: Tell us about your first exposure to computers. What was the first system you used and/or owned? Were you interested in gaming from an early age?
Ken Silverman: Throughout the 80s, my dad would take my older brother and I to his lab on occasion. To kill time, we would find an unoccupied Unix terminal and play games on it. My first computer was a TI-99/4a, which we got in December 1983.

DR: When did you develop an interest in programming and what was the first system you programmed for?
KS: My interest in programming began when I got the TI-99/4a. In the early days, I would work with my brother, such as designing mazes for a *Pacman* game. Later, my obsession with programming would overtake my brother's. I used TI BASIC, which was very slow; most of the games I wrote for it involved a monster (a red dot) chasing you (a green dot) that you would control with the joystick. Later, my dad borrowed a Kaypro II from work—it was like DOS with no graphics modes. I wrote some grid-based text adventures on that one. He later borrowed an HP Series 200, which had EGA-like graphics. I wrote some board games, some card games, and some other simple 2D games on it.

DR: When did you get your first PC, and which programming languages were you using initially?
KS: I got my first PC in November 1988. I probably installed *QuickBasic* on the first day. My parents tried to get me into C programming early on, but I was more comfortable with *QuickBasic*. It wasn't until 1990 that I started to realize the benefits of C.

DR: How and when did you enter the games industry in a professional capacity?
KS: That would be January 1, 1993, the day I released the shareware version of Ken's Labyrinth. I got a lot of calls after that. One of them was Epic MegaGames, who later sold the game under its own label. Later that year, I signed a contract with Apogee Software to develop the Build engine.



The logo used for the Build Engine games

DR: Tell us about Build's genesis and what you wanted to accomplish with it. Was it intended to better what had been done by John Carmack with the *Doom* engine?
KS: I started playing around with *Doom* tech soon after seeing id Software's original press release about it. I wrote an early demo with angled walls—PICROT4.BAS dated March 29, 1993. I have a copy of it on my website. At first, I was just trying to copy the features in the press release. When *Doom* (classic) was released, the goal changed to improving on the features of *Doom* in order to stay competitive.

DR: Did 3D Realms commission you to create the engine, or was it already a

"When *Doom* (classic) was released, the goal changed to improving on the features of *Doom* in order to stay competitive"

work-in-progress when they became involved?
KS: I had already named Build and written a demo when they hired me. They would not have offered me a deal otherwise. You can find this demo on my website—search for "grid-based BUILD engine". In January 1994, I did a complete rewrite of the engine, using the sector idea.

DR: To what extent were 3D Realms involved in the development of the engine itself? Did they specify the things they'd like to see included?
KS: Everybody had ideas and suggestions, but I did all the programming on the engine and tools. I didn't actually share the engine source code until late in the project. Many ideas were obvious, such as reminding me to copy features of other games. Many of them were impractical. Ultimately, it was up to me to select what ideas were most practical and useful to work on each day.

DR: Was Build originally intended to be used for *Duke Nukem 3D*?
KS: Build was not originally intended for any specific team. It was up to Apogee Software to find interested parties. One of the earliest users of Build was Nick Newhard, the head programmer of *Blood*. Todd [Replogle] and Allen [Blum] of *Duke Nukem 3D* didn't start with the engine until January 1994.

DR: How did Capstone come to use the engine for *Witchaven* before *Duke* was released? Was the engine unfinished at this time?
KS: I'm not sure how the Capstone deal came about. I only found out about it after the deal was done. Apparently, their contract had no restriction on release date. Notably missing from *Witchaven* were the sloped ceilings and floors. I was still adding features to Build in 1997. There is no specific finish date.

DR: What was the biggest technical challenge you came across when programming Build? Was there anything you had to leave out due to its complexity or feasibility?
KS: The biggest challenge would have to be the network code. It would have been nice if I had it all figured out in my original sample game. But I knew little about networking, and had to figure things out in the middle of the project. Once I got it working in my own test game, I had to figure out how to incorporate the changes into each game (*Duke 3D*, *Blood*, and *Shadow Warrior*). I had to learn three different code bases.

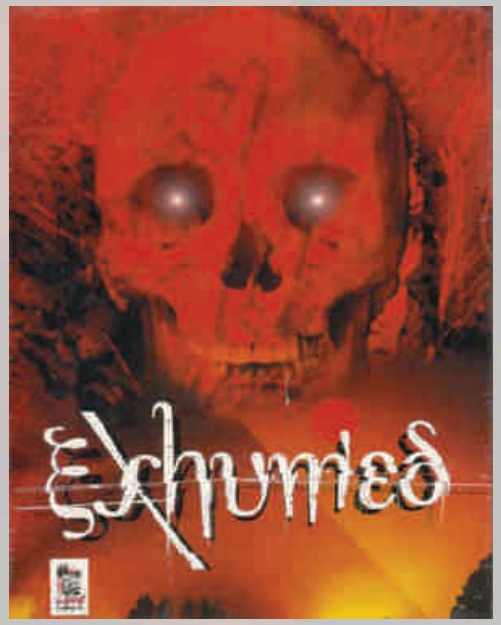
"I never wrote a design document for Build. I don't work that way. I plan things in my head, and find something useful to work on each day"

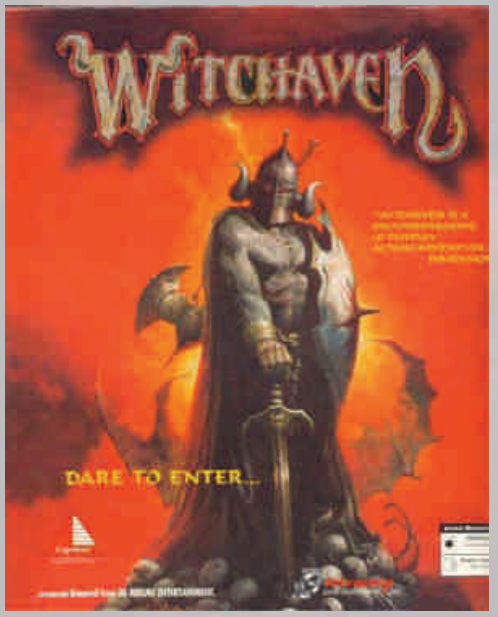
DR: Did 3D Realms give you a deadline for Build? Is there anything you didn't include that you'd have liked to? Did it live up to your expectations?
KS: That's one thing that's nice about working for 3D Realms—there are no deadlines, only milestones.

Here are some of the bigger things I wanted to do but never had time for:
* True perspective look up/down
* Native support for sector-over-sector
* Fancy lighting system
* Drop-in networking

I never wrote a design document for Build. I don't work that way. I plan things in my head, and find something useful to work on each day. It's a process of evolution. I can't say that I had expectations.

DR: Did you personally make any



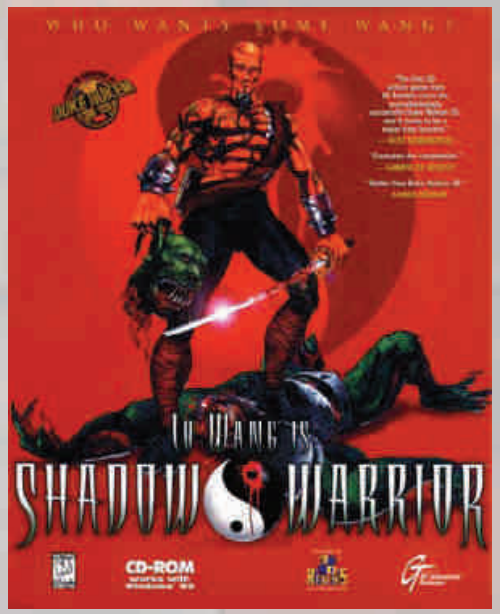


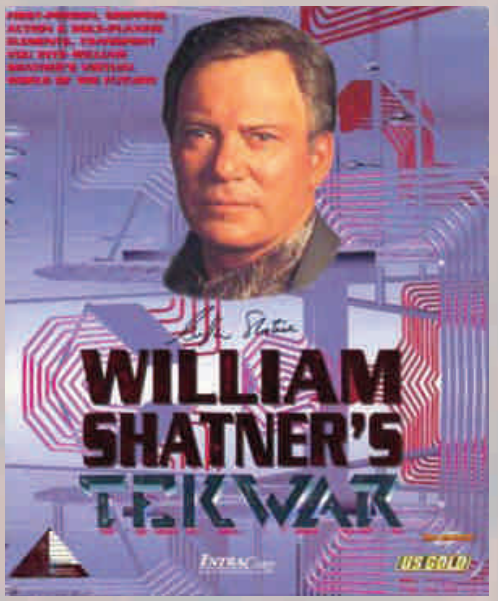
changes for *Shadow Warrior*, *Blood*, *Redneck Rampage* or any other Build games?
KS: I did several enhancements after *Duke 3D*. The most notable features are support for room over room and voxel sprites. *Shadow Warrior* also had transparent floors. I did very little to support *Redneck Rampage*, as they were not an internal team at any time.

DR: Which of the Build engine games are your favourite(s)? Are there any that you've never played?
KS: I always thought *Blood* had the best graphics; *Shadow Warrior* had the most features, and *Duke 3D* got points mainly for being the first of the big games. I have not played many Build games besides these three, and I don't think I've played through any of them without cheating.

DR: In 2000 you took the step of releasing Build's source code to the public. What compelled you to do this? Do you feel this is something that more programmers and developers should be doing with their older material?
KS: I released the code at that time for several reasons:
1. id Software set a precedent by releasing the *Doom* code.
2. Fans were pressuring me to release it.
3. I wanted to finish college before releasing it—to avoid a barrage of email during classes.
4. I felt that there was little value left in the Build code.

If people want to release code, that's great. But it takes time to prepare code for release. First you have to select a license that you're happy with. Then if you care, you





should document some things, like how to compile the code. The hardest part is removing or replacing anything that may be inappropriate, such as copyright textures or sounds that you did not make.

DR: Have you played any of the enhanced ports of *Duke* and *Shadow Warrior*? Have you tried the Xbox LIVE Arcade version of *Duke*?
KS: I wrote some of the systems in JonoF's ports, so obviously I have played those. I have yet to see the XBLA version in action.

DR: What's new in the world of Ken Silverman in the post-Build era? What have you developed since Build? What are you working on at the moment?
KS: I put many of my recent projects on my website (http://advsys.net/ken). I guess the most

"I always thought *Blood* had the best graphics; *Shadow Warrior* had the most features, and *Duke 3D* got points mainly for being the first of the big games"

popular ones have been: *Voxlap*, *PNGOUT*, and *Evaldraw*. I still do plenty of programming, but I don't like to announce things before they are finished.

DR: What's your view on the games industry of today?
KS: The industry has grown a lot since my time. I've gotten some offers, but I'm really not interested in returning to the industry. I'd rather work on my own ideas.

DR: Are you a retro gamer? Do you still find time to play the classics now and then?
KS: When I have time to waste, it is usually one of my own games that I play. I've written a few arcade classics for *Evaldraw*, such as *Breakout* and *Pacman* clones. I am always working on new things, but I do not wish to announce anything at this time.

DR: Lastly, do you think *Duke Nukem Forever* will ever see the light of day?
KS: I have no inside information other than knowing the tendencies of a few people who used to work there when dinosaurs roamed the earth.

If you're interested in seeing more of Ken's work, you can check out his personal website here: http://www.advsys.net/ken/

Many thanks to Ken Silverman and Jonathon Fowler for their help and contributions.