# Ubisoft tech UX director David Lightbown explores the history of Unreal Engine



**Gamasutra**
Gamasutra analyzes the classic Unreal engine with Ubisoft's David Lightbown

00:05:50                                        01:24:48
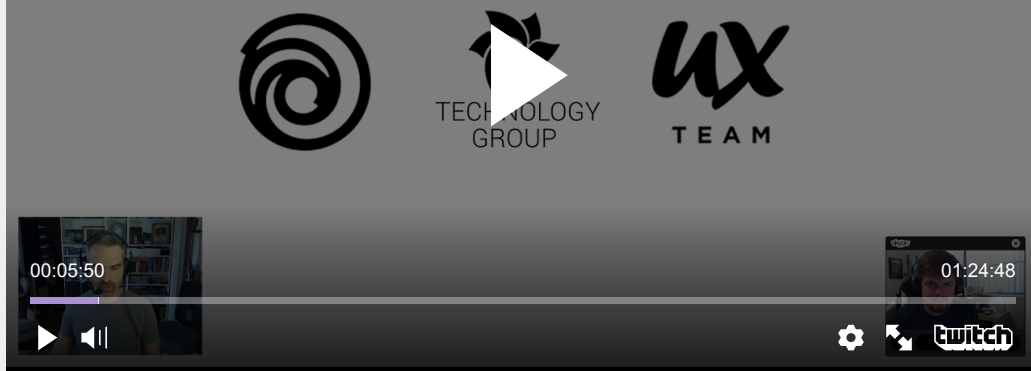
December 14, 2017 | By **Bryant Francis**

The Unreal Engine editor is one of the longest-running toolsets in game development history. From 1995 until today, it's been many a game developer's way in to the weird world of making games, and it's set a standard that other tool developers have learned from until now.

Today on the Gamasutra Twitch channel, we were lucky enough to be joined by Ubisoft Technology Group UX director David Lightbown, who's become something of an informal historian of game engines and the programmers who created them. After interviewing John Carmack a few months back, Lightbown talked to Epic co-founder Tim Sweeney about the history of the original Unreal Engine, and joined us to dig into several older coding tools that informed the engine's design.

You can (and definitely should) watch the conversation up above if you're making tools for your fellow game developers, but in case you're still on the clock and working on those tools right now, here's a few quick takeaways from our chat with Lightbown.

**Unreal is the product of several decades of improving coding tools**

While we only spent a few minutes in Unreal proper, much of our time with Lightbown was spent popping open older coding tools and examining how older UX advances helped define what would become Epic's flagship business. Lightbown pointed out that the auto-compiler of Turbo Pascal, the ability to play builds in Epic's *ZZT* editor, and the double-clicking and drag-and-drop functions of Visual Basic, all inform how modern-day Unreal developers make games.

As we noted during the stream, it was worth revisiting these tools because it's sometimes shocking that as recently as 2 decades ago, basic workflow functions that impact how we organize game files and mechancics just didn't exist, and it took specialized design work to bring them to life.

**Stop reinventing the wheel**

Before diving into the different tools, Lightbown pontificated on the need to study history so UX designers aren't doomed to repeat it. In his words, there are a lot of 'solved problems' for developers building proprietary tools that are sometimes buried in older software. Since users often have software behaviors they've been building for years with other programs, it's important for tool designers to take note of those behaviors so they don't accidentally demand they learn something new when they don't need to.

**Don't build one big tool when smaller specific ones will do**

At one point toward the end of our chat, a questioner from the stream queried Lightbown about how tools impact user creativity. What followed was a discussion about how sometimes it's tempting for tool creators to solve user problems by building 1 "do everything" tool when in reality, several more specific tools would do better.

For example, Lightbown talked about how Microsoft Word users sometimes complain they can't add more cells to a grid chart in that program...at which point, instead of trying to make a fix, he'd suggest they boot up Microsoft Excel, which is designed for grid charts. Lightbown conjured the image of a nightmare program called "Microsoft WordExcelPowerpoint," and after pausing for laughs, pointed out that some tools in game development fall into that exact trap, which developers should endeavor to avoid.