www.matthewball.vc /all/epicprimer1

Epic Games Primer (Pt I): Epic's Flywheel & Unreal Engine — MatthewBall.vc

May 18 Written By Matthew Ball & Jacob Navok

28-35 minutes

An Introduction to Epic Games & The 'Epic Games Primer'

This is designed to be an "Explain Like I'm (Twenty) Five" edition of Epic Games. Or if you want a throwback to the 1990s, when then-Potomac Computer Systems was founded, "Epic Games For Dummies". Technical terms and other details have been eschewed and simplified accordingly.

Epic Games was founded by Tim Sweeney and Mark Rein in 1991. Sweeney is the CEO and majority/controlling shareholder, while Tencent owns roughly 40%. As a private company, Epic does not publicly disclose its financials. However, it was valued at roughly $15B in 2018 (when it last raised capital) and is currently raising more at a "significantly higher" price, per Bloomberg.

Compared to Facebook, Amazon, Apple, Microsoft and Google, which are worth $600B to $1.4T, Epic's valuation is modest. But to many, Epic has the potential to become one of the largest, most influential tech companies in the world. This might seem hyperbolic to those who know Epic only as the marker of the hit video game *Fortnite: Battle Royale*. In fact, even long-time fans of Epic's games might find such a pronouncement odd given *Fortnite* has generated more revenue in three years than the rest of Epic has in almost as many decades. But behind the scenes, it looks increasingly likely that Epic will be at the very center of society's digital future.

This potential stems from the strength of Epic's core business, the Unreal Engine, which is already used to produce many of the world's leading games, movies and virtual experiences. This includes, of course, Epic's *Fortnite*, which is one of the most played and profitable online words globally. The popularity of this "game" is hard to miss. However, its success has enabled Epic to rapidly transform and expand its business. This includes the 2018 launch of Epic Games Store, the 2019 beta-release of Epic Online Services and acquisition of social video app Houseparty, and the 2020 unveiling of Epic Games Publishing. And with 350MM+ users and 2.3B social connections, Epic now operates one of the Internet's largest and fastest growing social networks.

Epic's growing strength has already forced enormous change in the media and entertainment industry. This includes forcing blue chip giants like Sony and Microsoft to open up parts of their closed gaming

ecosystems, convincing storied Hollywood giants to allow their franchises to intermingle, and showing the world that experiences once thought to be "IRL" only, like going to a concert with friends, had the potential to be even better when online-only.

But if Epic is successful in building out its 'flywheel', it will even more dramatically reshape the digital world - from data and privacy rights, to emergent technical standards, the distribution of profits, and the very ways in which humans work and relax. And all of this is critical to Sweeney's long-term vision of society's future: the Metaverse.

The purpose of this primer is to walk readers through each area of Epic Games. Specifically, we'll detail why it exists, how it works, what it hopes to achieve, why it matters, and how it will strengthen Epic Games overall. This first part focuses on the Unreal Engine.

The Epic Games Primer, Part I: The Unreal Engine (1995–)

What Is a Game Engine?

Games, virtual worlds, and digital simulation run on foundational code, or "engines". These manage everything from the processing of decision logic, rules, and physics (e.g. the X button was pressed, so a bullet was fired, which traveled from point A to point B, hit player 2, etc.), to real-time visuals, sound production, artificial intelligence, memory and network management, and so forth.

What Are Game Engine Specialists?

While every game requires an engine, the majority of game makers (developers and publishers) build and operate their games based on a third-party engine.

A good way to think about this is to consider movies and TV shows. Companies like Marvel and Disney don't need to manufacture, let alone design, custom cameras, editing software, digital storage, etc., to "make" their films and TV series. Instead, they use off-the-shelf equipment, physical technologies, and software designed by third-party specialists, such as Red Digital Cinema (cameras), Maya (computer animation software), and Technicolor (color processing).

This approach allows content and production companies to focus only on what they are best at — literally managing creative processes such as writing and casting, as well as production (filming), distribution (i.e. marketing, home video sales), etc. And should Marvel or Disney ever need something truly required for production, they could just build *that* and then integrate it into existing software, etc. An example here is Marvel's proprietary de-aging technologies.

In a simplified sense, Unreal Engine is an extensive suite of tools and technology that allows third parties to produce virtual experiences without needing to engineer the code that's needed to make them "work". In fact, they need not even *know* how they work. Instead, a developer can just focus on the creative. Just as a writer doesn't need to know how film is captured from a camera, or how video files are streamed over the Internet.

> *"...The bigger aim [with Unreal Engine 5] is to make content creators more productive, so that you can build next-generation games with a small team and produce incredibly high-quality results without a vast budget. The limitations of the technology are really only one of the game industry's problems. The other is the limits of people's time and the economics of building games, which is always a limiting factor for all game developers". - Tim Sweeney*

Notably, however, the complexity and scope of a game engine suite goes far beyond the film/TV analogy. Capturing frames of video that are played back in a single order is far simpler than managing the physics of a multiplayer game in real time. In addition, the technical capabilities of a game engine are constantly growing in both depth (e.g. how good the graphics are) and range (e.g. adding AR, VR, live voice transcription). Furthermore, these enhancements need to be deployed against hundreds of games and up to thousands of different devices without breaking any games or experiences. There is no "update Unreal but only for Game X".

What Are the Major Engines?

Today, there are only two engines that are commonly used: Unreal and Unity.

Unreal is typically considered best for visually rich games, especially multiplayer ones played on consoles/PCs. Unreal makes most of its money via a licensing fee, typically as a percentage of revenues (usually 5%). This fee can typically be bought out upfront, too.

Unity is easier than Unreal to use and lighter (i.e. game file sizes can be smaller) but also more technically limited. This tradeoff typically makes it a better fit than Unreal for mobile games, and, to this point, it's the dominant engine used for mobile games. This also means it's the most widely deployed engine overall, too. Importantly, Unity does not charge a revenue share — you only pay a small monthly subscription to use the software, similar to most design software such as Adobe Photoshop or Autodesk's Autocad.

Unity also operates a mobile ad network that allows developers to easily (i.e. "flip a switch") monetize their Unity-based apps without a need to find, integrate, and manage third-party ad networks, let alone hire their own ad sales force. Accordingly, they can focus even more exclusively on just the game design/creative. Each month, Unity delivers 23 billion ads across 2.2 billion devices.

Valve, which operates the world's largest digital gaming storefront (Steam) also offers the Source engine, but it's not widely used. Amazon has operated the Lumberyard engine since 2016, having bought a license from and forked Crytek's CryEngine, but it has seen little use.

Who Uses a Third-Party Game Engine?

The major publishers, such as Activision Blizzard or Take-Two, typically use proprietary engines over those of Unreal or Unity.

There are a few reasons for this. For example, most of the games sold by the major publishers can/will generate hundreds of millions or even billions in revenues. As a result, Unreal's revenue-based licensing fee can quickly feel "expensive," and thus there's often a desire to insource versus royalty-out. In addition, the number of games these publishers operate, the enormity of their revenues, and the size of their engineering teams mean that they can also afford to build or develop a proprietary engine. In fact, many have several.

The engines made by the major publishers are less capable overall than those of specialists like Unreal and Unity. However, the consequences are easy to overstate. While cutting edge graphical fidelity or physics can help differentiate a game's release marketing, games succeed because they're fun. Furthermore, revenue models today mostly depend on ongoing engagement, not the ability to sell a copy.

The bestselling game of the current, eighth generation of consoles, 2013's *Grand Theft Auto: V*, was made for and released on the seventh. Accordingly, it looks much "worse" than almost every AAA game it has outsold. And that's fine! *GTA:V* succeeds because it's fun and diverse, not because it's a visually realistic portrayal of organized and anarchist crime. Similarly, the most played AAA games globally, *Minecraft* and *Roblox*, look like games from the early 2000s and could mostly have been made then, too. They're about creation, not simulating reality.

In addition, publisher engines are typically purpose-built for a specific game or genre. Each year, for example, Activision Blizzard focuses on improving *Call of Duty* engine's visuals and physics, but primarily for the same look (gritty war simulation), fixed style of gameplay (fast-twitch first person shooters) and limited number of devices (e.g. PC and console). This isn't to diminish the quality or ease of these improvements, but the engine isn't designed to stretch far beyond the experience it needs to generate because it isn't required; *Call of Duty* doesn't need cel-shading. As a result, these engines can offer specific experiences that are close-enough to what Unreal might enable.

In addition, Unreal is not necessarily the right fit for all types of games. And even when it is, some developers prefer greater control over the "engine" code. In other cases, what a developer wants to do might not yet be supported by Unreal, and thus they'd rather build it themselves than wait.

Many publishers also talk about the strategic risk of relying on a third party for such critical technology. In five years, some publishers argue, Unreal or Unity might suddenly increase their revenue share rates. Even if these companies offer "grandfathered" rates on old games, a price hike would cut the profit margins of all future games. And the cost of changing engines mid-development is enormous (delays, budget increases, new hiring). In other cases, a developer might want or need something that Unreal or Unity doesn't yet support and won't get to in time. And even if these features or capabilities are planned, they could be cancelled or delayed. Additionally, Unreal's licensing terms also mean that in some cases technology or tools built by a developer are then freely licensed back to Unreal, which can then provide these innovations to the developer's competitors.

Though these concerns are understandable, usage of third-party engines has grown considerably in the last 15 years, and today even major publishers use third-party engines. For example, Nintendo's 2019 Switch game, *Yoshi's Crafted World*, uses Unreal, as does Square Enix's *Kingdom Hearts 3* and *Final Fantasy 7 Remake*. Activision Blizzard's *Hearthstone*, meanwhile, is based on Unity. Such exceptions typically result from constrained timelines, limited budgets or a mismatch between what a game needs and the capabilities of the publishers' own engines.

More important, however, is the fact that the majority of newly formed game studios are using Unreal or Unity. It's one thing for decades' old giants like Activision Blizzard and Nintendo to keep up with the leading independent engines. Trying to do so from scratch while also trying to build a new hit game is nearly impossible.

The only company to get close is Niantic. However, Niantic didn't build a game engine like Unreal or Unity. In fact, *Pokémon Go* runs on Unity, as does *Harry Potter: Wizards Unite*. Instead, Niantic offers a suite of "real world" platform tools that can hook into and be used to build atop other engines. Furthermore, this platform wasn't built anew, either. In 2015, Niantic spun out of Google's Maps group and presumably received an impossible-to-buy and generous long-term technology license from its former parent (though, of course, it has built up its technology considerably since). In addition, the company's first hit game, *Pokémon Go*, is based on a license from the highest-grossing media franchise in history.

> *"The cost of building an engine now is hundreds of millions of dollars, representing the work of hundreds of people for many years. You can't build an engine that's just good for one type of game anymore. The economics don't work". - Tim Sweeney*

In addition, it's important to note that Riot Games, maker of the world's most popular AAA game *League of Legends*, decided to use Unreal for its highly anticipated new shooter, *Valorant*. The fact that Riot is owned by Tencent, which owns ~40% of Epic, is obviously relevant to this decision. However, Riot is a highly profitable, technically skilled, and large publisher that's close to 15 years old. It is fully capable of building a new engine or adapting an old one. The decision to shift from an engine they fully own and operate to Unreal is therefore instructive.

Why Unreal's Technical Advantages Are Likely to Grow

As the complexity of video games grows, the incentives/need to use a third-party engine *specialist* are growing, too.

For gaming's first few decades, most games were purpose built for a specific device such as a dedicated arcade unit, Super Nintendo or PlayStation. This made engine design fairly simple - Square Enix, for example, knew the exact device its game(s) would run on for years. Over time, it became common for a publisher to release a title on two platforms (e.g. PlayStation and Xbox), but their technical hardware and input devices remained broadly similar. PCs were much more difficult, of course, as there were scores of manufacturers, each of which mixed-and-matched their devices with different components, most of which changed annually. This is why independent engines like Unreal first served the PC market. However, complexity here was managed by high performance requirements that meant the vast majority of PCs couldn't play a game.

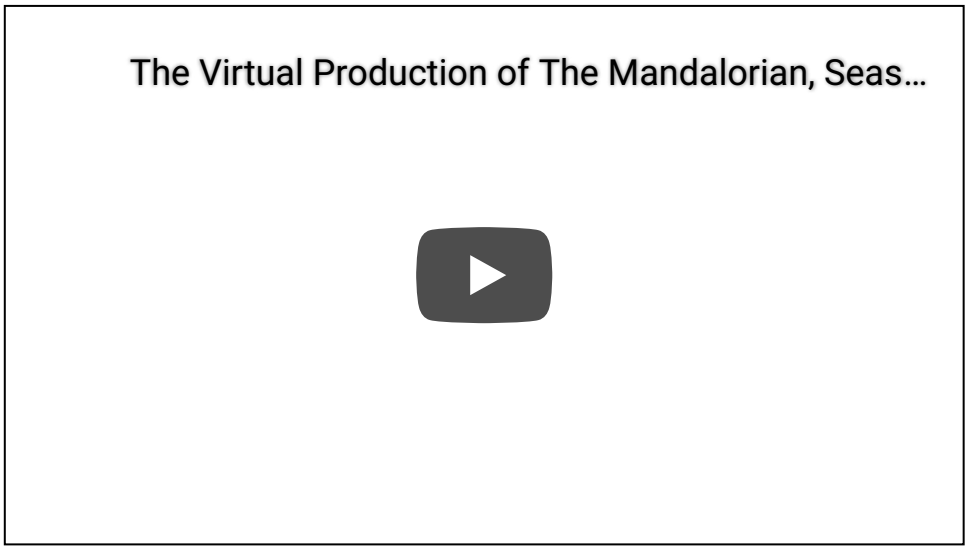
Today, however, most games aspire to work on as many platforms as possible - and to support cross-play between these platforms. What's more, there are more platforms than ever and several have not just multiple editions (e.g. iPad v. iPad Air v. iPad Pro), but refresh annually. This makes engine design considerably harder. It's one thing to make a game that works just on PlayStation 3, another to have it work on PlayStation 3 and Xbox 360, and another still to support PlayStation 4 and PlayStation 5 and Xbox One and Xbox Series X and various generations of iPads, iPhones and Android… and allow players on each device to play with one another. Developing on Unreal not only allows a publisher to focus more of its time on creative/production but also eliminates the worry about whether their game will work (and continue to work) well across all platforms and the enormous cost/burden of ensuring so.

As a widely deployed engine specialist, Unreal also benefits from an ever-growing number of unique feedback loops that expand its technical lead.

As mentioned above, Unreal's licensing terms also mean that in some cases technology or tools built by a developer are then freely licensed back to Unreal. This means that the engine benefits from the work of thousands of developers while proprietary publisher engines max out at a few hundred. In addition, Unreal's engineering team gets access to multiple times more player/usage/performance data, which in turn allows for better informed development and investment.

And although publisher engines are typically sufficient for the games they're designed for, this approach nevertheless limits innovation, investment, and quality. The best way to consider this feedback loop is to look beyond gaming:

Many believe that just as hand-drawn cartoons and practical effects shifted to CGI, so-called "real-time virtual production" game engines are the future of both animation and live action video. To this end, two of the biggest video releases of 2019, *The Lion King* and *The Mandalorian*, were shot entirely on Unity and Unreal respectively. This significance of this technological shift is hard to overestimate. The entirety of *The Mandalorian*, from its unnamed ice world, to the desert planet Nevarro and the forested Sorgan — and every set within them — was almost exclusively shot on a single virtual stage in Manhattan Beach, California. This was critical to the show's ability to offer film-grade visuals at half a film budget and more than twice the running time. However, it also offered unique cinemagraphic flexibility. Director Jon Favreau had perfect control over the elements of every shot — affording the perfect sunset (and perfectly positioned clouds) at any time. And rather than need to pick camera angles based on how a set looks backed by a greenscreen, he could literally move around the fully rendered set to pick (and change) the truly perfect angle in real time. The benefits here are particularly large given COVID-related restrictions on travel and the number of people on set.



And as Unreal expands in film and TV, it's also expected to become a leader in live music/events, too. Unreal also runs much of Disney's *Star Wars: Galaxy's Edge* theme park attraction. (Notably, Epic's CTO, Kim Libreri, was previously SVP Technology at Lucasfilm.) Sony Music, for example, has already announced it's building a technology team for Unreal-based concerts. This move is particularly telling as Sony Music's sister company Sony Interactive Entertainment has its own gaming platform (PlayStation) and numerous proprietary engines.

Unreal goes beyond entertainment, too. Over the past five years, a number of urban planning, architecture, and automotive engineering firms have shifted their workflows and designs to Unreal Engine or Unity. This allows for not just easier and more sophisticated rendering of their models, but also the ability to present fully functional simulations based around them. For example, Hong Kong International Airport uses Unity to simulate changes in passenger volumes. An architectural firm, meanwhile, can easily offer a would-be client a model that can actually be explored via touchscreen, mouse, or VR headset, or tested under various weather conditions. And if selected, its client could then launch a digital replica of this store using this same model — and do so even before the physical version opens.

Many secondary schools now use Unreal as part of their curricula, too (Epic offers lesson planners based around students designing Rube Goldberg machines, building sustainable cities, animating alternate endings to classic novels, etc). Epic's game engine is even used by the US military for simulations.

The scale and scope of Unreal's business means that Epic can invest in capabilities, technologies, and acquisitions that are impractical for companies that focus only on creating or marketing a video game.

Examples here include Cubic Motion (2020), 3Lateral (2019), and Quixel (2019). The first two companies are leaders in the creation of realistic, often produced-in-real-time digital humans. Quixel, meanwhile, creates "megascans" of real-world environments that can be easily adapted into and then built upon via Unreal. Any publisher can try to replicate these sorts of technologies in house or buy competing startups. However, they're competing not with other game makers like Nintendo or Capcom, but instead with Amazon's AWS, Facebook's Horizons, or Microsoft's Azure/Xbox.

To this end, it's no coincidence that eight of thirteen launch titles for Microsoft's newest console are Unreal-based; no engine is as ready for next generation hardware.

The ever-growing number of Unreal-based experiences also speaks to a final advantage: network effects. Different products/worlds/experiences don't need to be on the same engine to "talk to another". However a Porsche car designed in Unreal is quicker to drop into *Fortnite* than one designed on proprietary software. And similarly, it's much easier to drop an Unreal-based Porsche design into *Fortnite* than *Call of Duty* (which runs on a proprietary Activision Blizzard engine). The fact that the fictitious worlds of *The Mandalorian* were created in Unreal makes it much easier for the series' sets and characters to transition into a video game or virtual reality experience, too.

These examples may seem inessential today. However, an increasingly large portion of the gaming economy now runs on virtual goods. The ability to take items, outfits, and more from one game to another will obviously increase the price a player is willing to pay on these goods. And if a user ever wanted to jump across digital worlds (as is portrayed in *Ready Player One*), it helps if all worlds use the same "physics" and "logic".

How to Use Unreal

There are basically three tiers/access points to Unreal-based creation via Epic:

1. **Unreal:** This is for complicated games/simulations and is primarily code-based (and thus for computer engineers).

2. **TwinMotion:** This is for professional applications that aren't strictly about simulations but more about visualization/renders, such as auto design or architecture. TwinMotion is icon-based in the same way Windows is icon-based v. DOS. The company was bought by Epic in mid-2019.

3. **Fortnite Creative Mode**: This is similar to *Minecraft* in the sense that it gives non-experts pre-fabricated assets (e.g. trees, stairs) and heavily simplified tools for game making. Compared to TwinMotion or Unreal, creation is fairly limited (e.g. designs are mostly tied to *Fortnite*'s art style and items), but a child can use it, and fully immersive, narrative games can still be told. Here are some good examples of Creative Mode creation (#1, #2).

In a simplified sense, Unreal is like building a dollhouse using raw inputs like uncut lumber and pigment dyes using a professional workshop. TwinMotion is closer to building a dollhouse out of all possible Ikea parts. Creative Mode, meanwhile, would be like using Lego.

Epic hopes to make it easier for those using Fortnite Creative Mode or TwinMotion to make the sorts of complex, highly customized experiences that today can only be made directly via Unreal Engine 4 and Epic.

To this same end, it's important to identify that Creative Mode speaks to the growing popularity of "no code" game engines such as *Roblox* and *Minecraft*. Although these titles are commonly thought of as "games", they are really end-to-end platforms that allow anyone to create a game (engine), distribute it (store), attract players (i.e. *Roblox* and *Minecraft* users), and monetize (Robux and Minecraft Coins). This means that the *Minecraft* and *Roblox* game engines are only available for game creation within Minecraft and Roblox. *Fortnite Creative Mode*, meanwhile, is just one way to use Unreal.

This is great for driving more people into the game creation business. However, there are real impediments for professional/commercial game makers. As a result, it's not *yet* right to consider them competitors to Unreal like Unity or Valve's Source.

For example, a developer is limited only to the features, capabilities, and graphics afforded by these no-code platforms. They cannot add bespoke technologies or change the core code themselves. Similarly, developers must use all of the platform's game services (e.g. its account system, voice chat), which prevents forming a direct relationship with the player and limits data collection. What's more, their

games can't even be played or distributed outside the "parent game" (e.g. *Roblox*). This restricts the size of the realizable player base to that parent game's user base, complicates marketing, and results in considerable supplier risk.

Most challenging, however, are the revenue sharing rates, which are particularly rough here. *Roblox*, for example, nets developers less than 30% of the revenue they generate (Sony and Apple give 70%). This is in part because *Roblox* first needs to pay the stores/platforms it operates on (e.g. Sony and Apple).

The Strategic Value, Influence, and Durability of Unreal

Although game engines don't own the end consumer like a game publisher (e.g. Activision Blizzard) or platform (Sony), they are uniquely valuable because of how they have access to and can influence the developers that make games in the first place.

What Epic decides to invest in, improve, or create, for example, immediately impacts not just what experiences developers will (or can make), but also the hardware that's made, the standards deployed, and the content consumers buy, and so on.

If Epic chooses to invest into real-time ray tracing (a technology which changes the way computers create an image, resulting in greater realism), they allow developers to make games with higher visual fidelity. This, in turn, gives gamers reasons to buy Nvidia GPUs with real-time ray tracing support and PC manufacturers reasons to integrate these GPUs. If Epic invests into VR and AR technologies, it becomes much easier for developers to deploy games with those technologies onto Oculus or Apple products. This means there are more reasons to buy these devices, which leads to greater sales, thereby increasing the addressable playerbase of VR/AR gamers, which leads to more developers producing VR/AR games. And, of course, the gaming space is rapidly growing in terms of players, playtime, and importance globally.

In addition, engine-makers can use their reach to push and accelerate many other businesses. As mentioned above, Unity used its reach in mobile gaming to launch one of the largest mobile app ad networks in the world (it ranks third, fourth, or fifth depending on the metric used, such as ads server v. impressions, etc.). As will be detailed in parts 2 and three, Epic was also able to use Unreal to launch a games store, as well as an online services and a publishing arm.

Unreal is so deeply deployed today that it's also difficult for its market leadership position to quickly erode. In theory, an airplane could have its engines swapped mid-flight. A video game cannot change its engine after it's released. And as games shift from packaged sales of play-and-finish games to live-operated unending ones, the lifespan of a single title continues to grow. As a result, building on Unreal today can mean committing to it through 2035. Note, too, that many of the games that may soon lead the industry, such as *Valorant*, use Unreal and have yet to even launch.

Taking a further step back, it's critical to recognize that whole generations of developers have now been trained in Unreal and spent up to several decades becoming experienced in it. As a group, developers are incredibly reluctant to learn new engines; there's little upside, lots of risk, and a huge investment required. And even if a team does convert, it takes four or more years to release a major AAA title. This means that even if a new, fully featured Unreal competitor emerged, it would take years to harm Epic's market share.

To this end, we can consider the entry of Amazon. In 2015, Amazon licensed the CryEngine for $50–70MM and began to redevelop it as "Amazon Lumberyard". Lumberyard is designed to drive Amazon's AWS business (and, presumably, its rumoured cloud gaming service). Accordingly, it's free to use for

developers, unlike Unreal and Unity, which charge licensing fees. However, almost no third-party developers have chosen to use it.

The next part of this series will publish May 19 and focus on Epic Games Store and Epic Game Publishing. On May 20th, we will cover Epic Online Services. May 21st will cover Fortnite. May 22nd will then explain how everything Epic does fits into Sweeney's vision for the future, and how it's transforming the way its competitors, partners and consumers think, too.

Matthew Ball (@ballmatthew) & Jacob Navok (@jnavok)