Unity Game Engine

What is Unity Game Engine?

Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Worldwide Developers Conference as a Mac OS X game engine. The engine has since been gradually extended to support a variety of desktop, mobile, console and virtual reality platforms. It is particularly popular for iOS and Android mobile game development and is considered easy to use for beginner developers and is popular for indie game development.

The engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering, construction, and the United States Armed Forces.

Game Engine Architecture

What Is a Game?

We probably all have a prett y good intuitive notion of what a game is. The

general term “game” encompasses board games like chess and Monopoly, card games like poker and blackjack, casino games like roulett e and slot machines, military war games, computer games, various kinds of play among children, and the list goes on. In academia we sometimes speak of “game theory,” in which multiple agents select strategies and tactics in order to maximize their gains within the framework of a well-defi ned set of game rules. When used in the context of console or computer-based entertainment, the word “game” usually conjures images of a three-dimensional virtual world featuring a humanoid, animal, or vehicle as the main character under player control. (Or for the old geezers among us, perhaps it brings to mind images of two-dimensional classics like Pong, Pac-Man, or Donkey Kong.) In his excellent book, A Theory of Fun for Game Design, Raph Koster defi nes a “game” to be an interactive experience that provides the player with an increasingly challenging sequence of patt erns which he or she learns and eventually masters. Koster’s assertion is that the activities of learning and mastering are at the heart of what we call “fun,” just as a joke becomes funny at the moment we “get it” by recognizing the patt ern.

For the purposes of this book, we’ll focus on the subset of games that comprise two- and three-dimensional virtual worlds with a small number of players (between one and 16 or thereabouts). Much of what we’ll learn can also be applied to Flash games on the Internet, pure puzzle games like Tetris, or massively multiplayer online games (MMOG). But our primary focus will be on game engines capable of producing fi rst-person shooters, third-person action/platform games, racing games, fi ghting games, and the like.

What Is a Game Engine?

The term “ game engine” arose in the mid-1990s in reference to fi rst-person shooter (FPS) games like the insanely popular Doom by id Soft ware. Doom was architected with a reasonably well-defi ned separation between its core soft - ware components (such as the three-dimensional graphics rendering system, the collision detection system, or the audio system) and the art assets, game worlds, and rules of play that comprised the player’s gaming experience. The

value of this separation became evident as developers began licensing games and re-tooling them into new products by creating new art, world layouts, weapons, characters, vehicles, and game rules with only minimal changes to the “engine” soft ware. This marked the birth of the “mod community ”—a group of individual gamers and small independent studios that built new games by modifying existing games, using free toolkits provided by the original developers. Towards the end of the 1990s, some games like Quake III Arena

and Unreal were designed with reuse and “ modding” in mind. Engines were made highly customizable via scripting languages like id’s Quake C, and engine licensing began to be a viable secondary revenue stream for the developers who created them. Today, game developers can license a game engine and reuse signifi cant portions of its key soft ware components in order to build games. While this practice still involves considerable investment in custom soft ware engineering, it can be much more economical than developing all of

the core engine components in-house. The line between a game and its engine is oft en blurry. Some engines make a reasonably clear distinction, while others make almost no att empt to separate the two. In one game, the rendering code might “know” specifi - cally how to draw an orc. In another game, the rendering engine might provide

general-purpose material and shading facilities, and “orc-ness” might

be defi ned entirely in data. No studio makes a perfectly clear separation between the game and the engine, which is understandable considering that the defi nitions of these two components oft en shift as the game’s design solidifi es.

Arguably a data-driven architecture is what differentiates a game engine from a piece of software that is a game but not an engine. When a game contains hard-coded logic or game rules, or employs special-case code to render specific types of game objects, it becomes difficult or impossible to reuse that software to make a different game. We should probably reserve the term “game engine” for software that is extensible and can be used as the foundation for many different games without major modification.

Why use unity game engine?

Unity 1.0 was released in June 2005, when developers Nicholas Francis, Joachim Ante and David Helgason created the first version of what would become one of the most widespread game engines. The three set up to create a software that cared for developers’ needs, and with every new version that they launched, the game engine got increasingly better.

Today, Unity is used worldwide and is one of the most popular cross-platform game engines out there. Its community embodies over 2.5 million registered developers, and is a place where both hobbyists and experienced users can share ideas, ask for advice or browse through documentation.

To better understand why Unity is so popular in the game development world, let’s elaborate on some of the main ingredients that contribute to its notoriety.

1. Multiple platforms

Although it was first released only for Mac OS X, Unity is now supporting over 25 different platforms, which enable developers to make their game more accessible for the public. Creators have the possibility to deploy on all popular platforms and devices, from Android and iOS to consoles and web plugins, everyone can download their creation, no matter the gadget that they own.

2. 2D, 3D, VR and AR

Unity gives developers the opportunity to create not only 3D games, but, since November 2013, specialists can design 2D games as well. And as far as mobile gaming goes, Unity dominates more than half of the market, as creators continue to rely on this game engine to build the most successful games. Unity holds the supremacy for AR and VR as well, with over 60% of the developed content.

3. Cost

Another great thing about Unity is that it allows anyone that is interested in game creation, to download a free version. Small, independent developers are encouraged to pursue their vision and are provided with various tools and techniques to help make their ideas come to life. For those who are looking for extra features, there is also a Pro Business plan available, that goes for 150 dollars a month, and includes priority customer service.

4. Great Graphics

When it comes to graphics, Unity is known for its high quality, next-level visual effects. The highly customizable rendering technology that the game engine offers, along with a variety of intuitive tools, facilitate the creation of fantastic looking games. In 3D games, developers can generate natural, smooth movements of objects and there are also lots of tutorials to choose from, that explain how to do it all.

5. Unity Asset Store

Launched in November 2010, Unity’s Asset Store makes game development even easier and more cost-effective. And creators have a variety of assets to choose from; whether they’re looking for specific sounds, 3D structures, patterns, textures or animations, developers can use them to make their vision come alive. These assets are available for both free and paid versions and, furthermore, designers can sell their assets and collect 70 percent of the revenue.

6. Less coding

The fact that you don’t need to have a great deal of coding knowledge or experience, is really convenient for those who are at the beginning of their game development career or are just looking to start a hobby. Unity uses C#, JavaScript and Boo, but you can also develop a game from start to finish without writing a single line of code. At the same time, for those who do have coding skills, learning programmes and documentation are available, and can come in handy.

7. Strong community

Having such a large community, Unity developers can ask questions and find solutions to their issues by asking other members, all this in a friendly and supportive environment. Moreover, Unity engineers meet with community members annually, in an organised event named after the game engine. During this conference, developers share experiences, exchange ideas, and connect with each other, consolidating the community.