Adrian Sigala Chavez

Ms. Gifford

ELA IV

February 21, 2025

The History and Development of 2-Dimensional Games

Think about the first two-dimensional (2D) game that was played. Most do not recall what it was called. However, at the time, most enjoyed playing it. Was it a platformer, a first person shooter, or a type of board game? 2D games have been a cornerstone of gaming since the 1950's, starting with Tennis for Two and evolving into iconic titles like Super Mario Bros. and Celeste (Bramble.) These games, though simple in appearance, have captivated players for decades with their creativity, storytelling, and emotional depth. The way two-dimensional games look and are developed has changed drastically over the years to better adapt to new genres and emotions, driven by advancements in technology, design, and sound.

The origins of 2D games can be traced back to 1958, when physicist William Higinbotham created Tennis for Two, the first 2D game (Bramble.) This simple tennis simulation laid the foundation for future games. While it was rudimentary by today's standards, Tennis for Two demonstrated the potential of interactive entertainment. In 1972, Pong revolutionized the gaming industry, becoming one of the most recognized 2D games of all time. Developed by Atari, Pong was a commercial success and introduced millions to the concept of video games. The rise of arcade games in the 1970s and 1980s further cemented 2D games as a cultural phenomenon. Titles like Pac-Man and Donkey Kong became household names, captivating players with their addictive gameplay and colorful visuals. By the 1980s, 2D games transitioned into home consoles, becoming a staple of early gaming systems like the Atari 2600 and the

Nintendo Entertainment System (NES) (Bycer.) These consoles brought 2D games into living rooms, allowing players to experience the joy of gaming without visiting an arcade.

As technology advanced, so did the design of 2D games. Early games relied on simple pixel art, but over time, developers began experimenting with more intricate, hand-drawn styles. For example, Super Mario Bros. (1985) used pixel art to create a vibrant, colorful world filled with memorable characters and levels. The game's success solidified Nintendo's place in gaming history and set a standard for future platformers. In contrast, Hollow Knight (2017) featured detailed, hand-drawn animations that brought its dark, atmospheric world to life. The game's art style, combined with its challenging gameplay and emotional storytelling, earned it critical acclaim and a dedicated fanbase. The 2010s saw a resurgence of 2D games, particularly in the indie game scene, where titles like Celeste, Shovel Knight, and Dead Cells emphasized creativity and storytelling over flashy graphics. These games proved that 2D games could still innovate and captivate players in an era dominated by 3D graphics.

The development of 2D games relies heavily on game engines, which provide the tools needed to bring these games to life. Unity is one of the most popular engines for 2D development, offering tools for sprite management, physics, and more (Bramble.) Its versatility and ease of use have made it a favorite among indie developers and large studios alike. Games like Cuphead and Ori and the Blind Forest were created using Unity, showcasing the engine's ability to handle complex animations and beautiful visuals. GameMaker is another popular choice, known for its beginner-friendly features and used in games like Undertale and Hyper Light Drifter (Bramble.) These games demonstrate how accessible tools can empower developers to create unique and memorable experiences. Other engines, such as the Source Engine and

Godot, also play a role in 2D game development, offering flexibility and customization for developers who want to push the boundaries of what 2D games can achieve.

Programming languages are another critical component of 2D game development. C# is one of the most widely used languages in 2D game development, particularly for games built with the Unity engine. Unity's robust tools and extensive documentation make it a favorite among indie developers and large studios alike. C# is an object-oriented language, meaning it organizes code into reusable components called classes and objects. This makes it ideal for managing complex game systems, such as character movement, enemy behavior, and inventory management. For example, in Celeste, C# was used to create the game's precise platforming mechanics and responsive controls (Tran.) Unity's built-in physics engine and sprite management tools also rely heavily on C#, allowing developers to create dynamic and interactive worlds without starting from scratch. Additionally, C# is beginner-friendly, with a syntax that is easy to learn and understand. This accessibility has made it a popular choice for developers who are new to game development but want to create polished, professional-quality games.

C++ is another popular language for 2D game development, particularly for high-performance titles. Known for its speed and efficiency, C++ gives developers greater control over memory management and optimization, making it ideal for games that require complex calculations or real-time rendering. For example, Hollow Knight was developed using C++ to handle its intricate animations and large, interconnected world (Tran.) C++ is also commonly used in game engines like Unreal Engine, which supports both 2D and 3D development. While C++ is more difficult to learn than C#, its performance benefits make it a preferred choice for developers who want to push the limits of what 2D games can achieve.

However, its complexity means that it is often reserved for experienced programmers or larger teams with the resources to manage its steep learning curve.

Java is another language commonly used in 2D game development, particularly for mobile and web-based games. Its “write once, run anywhere” philosophy makes it ideal for creating games that can run on multiple platforms, including Windows, macOS, and Android. Java’s object-oriented design and extensive libraries, such as LibGDX, provide developers with the tools they need to create 2D games quickly and efficiently. For example, Minecraft (initially a 2D prototype) was developed using Java, showcasing the language’s versatility and power (Tran.) Java’s garbage collection feature also simplifies memory management, reducing the risk of crashes and performance issues. However, Java’s reliance on the Java Virtual Machine (JVM) can lead to slower performance compared to languages like C++, making it less suitable for high-performance games. Despite this, Java remains a popular choice for developers who prioritize cross-platform compatibility and ease of use.

Lua is a lightweight scripting language often used in 2D game development for its simplicity and flexibility. Unlike C# or C++, Lua is not a full-fledged programming language but rather a scripting language, meaning it is used to control specific aspects of a game rather than build it from the ground up. Lua is commonly embedded in game engines like LÖVE and Corona SDK, where it is used to handle game logic, user input, and animations. For example, Celeste used Lua for its scripting, allowing the developers to quickly prototype and iterate on gameplay mechanics (Tran.) Lua’s lightweight design makes it ideal for mobile games and smaller projects, where performance and resource usage are critical. Additionally, Lua’s simple syntax and ease of integration with other languages make it a popular choice for developers who want to add scripting capabilities to their games. However, Lua’s lack of built-in tools and

libraries means that developers often need to rely on external resources or custom solutions, which can increase development time.

Art and animation are essential to the success of 2D games. Player character sprites, NPCs, and enemies must be memorable and visually appealing to engage players (Bramble.) A well-designed character can become iconic, like Mario from Super Mario Bros. or the Knight from Hollow Knight. Animation techniques, such as frame-by-frame and skeletal animation, bring these characters to life. Frame-by-frame animation involves creating individual frames for each movement, resulting in smooth and detailed animations. Skeletal animation, on the other hand, uses a rigged model to simulate movement, making it easier to create complex animations. Generative Adversarial Networks (GANs) are now being used to automate sprite generation and animation, making the process more efficient and allowing developers to focus on creativity (Choi et al.) The choice of art style, whether pixel art, hand-drawn, or vector art, also plays a significant role in shaping the player's emotional connection to the game (Bramble.) For example, the pixel art in Celeste complements its retro-inspired gameplay, while the hand-drawn art in Cuphead evokes the style of 1930s cartoons.

Sound design is another crucial element of 2D games. Sound effects, such as footsteps or weapon fire, must align with on-screen actions to create an immersive experience ("How Game Sound Design Enhances Player Experience and Immersion.") For example, the sound of footsteps changing based on the terrain—metal clanging or grass rustling—provides players with auditory cues about their surroundings ("How Game Sound Design Enhances Player Experience and Immersion.") Music sets the mood, shifting from calm to intense to guide players through a rollercoaster of emotions. For example, The Legend of Zelda: Breath of the Wild uses dynamic sound design to create an auditory landscape that feels alive, while DOOM (2016) integrates

heavy metal riffs to keep adrenaline levels high ("How Game Sound Design Enhances Player Experience and Immersion.") Adaptive audio and 3D audio technology further enhance immersion, even in 2D games. Adaptive audio shifts the game's music and sound effects based on player actions or game events, creating a more dynamic experience. 3D audio technology places sounds in a three-dimensional space, making players feel like they're truly part of the game world.

World-building and environment design are also key to creating memorable 2D games. Unlike 3D environments, 2D worlds may lack visual depth, but they have their own charm and require less computational power to render (Dave.) When designing for PC and consoles, developers must prioritize design, details, and visual clarity to ensure compatibility with larger screens (Dave.) For example, Hollow Knight features a sprawling, interconnected world filled with secrets and lore, encouraging players to explore every corner. NPCs and enemies contribute to world-building through voice acting and memorable designs, while objectives must feel natural within the game's environment (Bramble.) For example, in Undertale, the NPCs and enemies are integral to the story, with each character having a unique personality and role in the world.

Despite their simplicity, 2D games face several challenges in development. Animation requires a balance between fluidity and performance, as too many frames can slow down the game. Sound design must adapt to diverse player setups, from home theaters to smartphones, ensuring that the experience is consistent across devices ("How Game Sound Design Enhances Player Experience and Immersion.") Coding challenges include learning new languages like C# or Lua and optimizing code for performance on various platforms (Tran.) For example, Celeste faced challenges in optimizing its physics engine to ensure smooth gameplay across all devices.

The future of 2D games is bright, with technological advancements like AI and machine learning shaping the way games are developed (Choi et al.) Emotional storytelling has become a growing focus, with games like Undertale proving that 2D games can tell powerful stories that would not work in 3D. Accessibility and creativity ensure that 2D games remain popular, offering players unique experiences that 3D games cannot replicate. For example, Celeste uses its 2D platforming mechanics to explore themes of mental health and self-discovery, creating a deeply personal and emotional experience.

In conclusion, the history of 2D games shows a progression from simple mechanics to complex, emotionally driven experiences. Advances in engines, programming, art, and sound have transformed how 2D games are developed and experienced. As technology evolves, 2D games will continue to adapt, offering players new ways to connect with stories and worlds. Whether through nostalgic pixel art or innovative storytelling, 2D games will always hold a special place in the hearts of gamers.

Work Cited

Bramble, Ross. “Everything You Need to Know Before You Make a 2D Game.” *GameMaker,* 5 Mar. 2023, gamemaker.io/en/blog/how-to-make-a-2d-game. Accessed 13 Dec. 2024.

Bycer, Josh. “From Retro to Modern: Exploring the evolution of video game consoles.” *Game Wisdom*, Game Wisdom, 20 12 2023, https://game-wisdom.com/general/retro-modern-exploring-evolution-video-game-consoles. Accessed 19 2 2025.

Choi, Jong-In, Kim Soo-Kyun, and Kang Shin-Jin. "Image Translation Method for Game Character Sprite Drawing." *Computer Modeling in Engineering & Sciences*, vol. 131, no. 2, 2022, pp. 747-762. *ProQuest*, https://www.proquest.com/scholarly-journals/image-translation-method-game-character-sprite/docview/2646008907/se-2, doi:https://doi.org/10.32604/cmes.2022.018201.

Dave, Ankit. “Game Environment Modeling - All You Need to Know.” *300Mind Blog*, 7 July 2023, 300mind.studio/blog/game-environment-modeling-guide/.

“How Game Sound Design Enhances Player Experience and Immersion - Pyramind Institute.” *Pyramind Institute - Music and Audio Production: School & Recording Studios*, 12 May 2024, pyramind.com/how-game-sound-design-enhances-player-experience-and-immersion/.

Tran, Trung. “Top 6 Programming Languages for Game Development.” *Pyramind*, 12 Apr. 2024, www.orientsoftware.com/blog/gaming-programming-language/#:~:text=JavaScript%20is%20mainly%20used%20for,both%202D%20and%203D%20elements. Accessed 18 Dec. 2024.