



UNITY

GAMES ENGINE AND SCRIPTING

MARCOS JODAR GÓMEZ

19 DE FEBRERO DE 2024

ÍNDICE

DESCRIPCIÓN DEL PROYECTO	3
MOTORES DE JUEGOS	4
Unity	5
Unreal Engine	9
Godot Engine	12
CryEngine	16
Lumberyard	20
COMPARACIÓN entre los distintos motores	23
TENDENCIAS	24
EVALUACIÓN DE LOS MOTORES DE JUEGOS	25
UTILIZACIÓN DE GDD	34
PLAN DE DESARROLLO	34
RUSHED PENGUIN	43
PRUEBAS	43
EVALUACIÓN Y MEJORAS DE LA DEMO	46
Rendimiento y Evaluación	47
WEBGRAFÍA	49

DESCRIPCIÓN DEL PROYECTO

Para asegurar mi idoneidad para el puesto en el estudio de videojuegos, me embarcaré en dos actividades fundamentales. En primer lugar, llevaré a cabo un exhaustivo informe que analizará la evolución de diversos motores de juegos. En este análisis, evaluaré sus características y arquitecturas, abordando aspectos como monetización, análisis, multijugador, gráficos, animación, física y más. Asimismo, revisaré juegos de diversos géneros y plataformas para explicar las características de los motores incorporados en cada uno.

Como segunda actividad, crearé un prototipo jugable utilizando un motor de juegos específico. Este prototipo demostrará implementaciones comunes en videojuegos, como animaciones, colisiones y física. Además, evaluaré y planificaré mejoras que se implementarán en futuras versiones del prototipo. Estas actividades no solo resaltan mi habilidad para analizar y mejorar motores de juegos, sino también mi destreza en el desarrollo y perfeccionamiento de experiencias de juego.

MOTORES DE JUEGOS

Hay varios motores de juegos populares y otros no tanto, cada uno con sus propias características y ventajas. Algunos de ellos:

Unity



Unity es uno de los motores de juegos más utilizados y versátiles. Es conocido por su facilidad de uso y es ideal tanto para desarrolladores principiantes como para profesionales.

Cuenta con una amplia compatibilidad de plataformas, gráficos de alta calidad, soporte para realidad virtual y aumentada, además de una gran comunidad de usuarios.

A continuación, veremos sus ventajas, fortalezas y debilidades:

© Ventajas:

- ❖ Fácil de aprender, con ciertos tutoriales de Youtube puedes aprender rápidamente.
- ❖ Cuenta con una gran cantidad de recursos y activos disponibles en la tienda de Unity Asset.
- ❖ Tiene un soporte para múltiples plataformas.

© Fortalezas:

Facilidad de Uso:

- Es muy conocido por su interfaz de usuario intuitiva y su fácil uso, como ya hemos dicho. Siendo accesible para desarrolladores principiantes y fácil de aprender.

Compatibilidad de Plataformas:

- Es altamente compatible con una amplia gama de plataformas, como PC, consolas, dispositivos móviles y realidad virtual. Esto facilita el traspaso de los juegos a diferentes dispositivos.

Gran Comunidad y Soporte:

- cuenta con una comunidad de usuarios muy grande y activa. Haciendo una abundancia de recursos en línea, tutoriales y activos disponibles en la tienda de Unity Asset, como ya hemos dicho.

Ecosistema de la Tienda de Assets:

- Hablando de la tienda Unity Asset, esta proporciona una amplia variedad de recursos y herramientas que aceleran su desarrollo al ofrecer activos predefinidos y soluciones.

Desarrollo Multiplataforma:

- Unity permite el desarrollo multiplataforma, para crear juegos para diferentes Sistemas Operativos y dispositivos fácilmente.

© Limitaciones:

Gráficos Iniciales Menos Potentes:

- Muchos desarrolladores consideran que, en comparación con *Unreal Engine*, motor que veremos próximamente, la calidad gráfica inicial puede ser inferior.

Licencia Gratuita Limitada:

- Unity ofrece una versión gratuita, pero las características avanzadas y servicios requieren una suscripción o pago adicional, lo que limita a desarrolladores con presupuestos ajustados.

Optimización Necesaria:

- La optimización podría requerir más esfuerzo en comparación con otros motores.

Cambios Frecuentes:

- Las numerosas actualizaciones de Unity, que aunque mejoran el motor, causan ciertos inconvenientes a los desarrolladores que necesitan adaptarse constantemente a cambios en la plataforma.

Origen y Tipo de Unity:

Unity fue desarrollado inicialmente por *Unity Technologies* y fue lanzado públicamente en junio de 2005. Su objetivo principal era proporcionar una plataforma de desarrollo de juegos accesible y versátil, dando un motor de juegos multiplataforma que admitiera el desarrollo de juegos para una amplia variedad de plataformas, incluyendo PC, consolas, dispositivos móviles y realidad virtual.

Cronológicamente en cuanto a sus versiones cuenta con multitud de ellas:

- **Versión 1.0 (2005):** El lanzamiento inicial de Unity se centró en ofrecer un entorno de desarrollo amigable y accesible.
- **Versión 2.0 (2007):** *Unity 2.0* introdujo mejoras significativas, como un soporte para *Windows* y *Mac OS X*, así como una interfaz de usuario mejorada. En esta etapa, Unity empezó a ganar popularidad entre los desarrolladores.
- **Versión 3.0 (2009):** *Unity 3.0* trajo mejoras en cuanto a gráficos y rendimiento, así como el soporte para plataformas adicionales como *Android*, ampliando aún más la versatilidad del motor.
- **Versión 4.0 (2013):** *Unity 4.0* se enfoca en mejorar la calidad visual y la capacidad de renderizado. Además de introducir el sistema de animación *Mecanim*.
- **Versión 5.0 (2015):** *Unity 5.0* mejoró la iluminación global, introdujo el motor de física 2D y se ofreció soporte para *WebGL*. Estos avances fortalecieron la capacidad del motor para crear experiencias visuales más impresionantes.
- **Versión 2017 (Unity 2017):** *Unity 2017* introdujo *Scriptable Render Pipeline* o SRP (controlador de gráficos), lo que permitió un mayor control sobre el proceso de renderizado y abrió la puerta a gráficos más avanzados.
- **Versión 2018 (Unity 2018):** *Unity 2018* trajo mejoras en el rendimiento, la calidad gráfica y la capacidad de producción.
- **Versiónes posteriores (2020 y más):** Unity continúa evolucionando con actualizaciones periódicas que introducen características avanzadas, mejoras en el rendimiento y soporte para las últimas tecnologías. Como su última versión, la *Unity 2022.1 TECH Stream*, que

cuenta con actualizaciones, como una mayor productividad, editor expandible y con un flujo de trabajo más rápido.

Títulos Influyentes y su Impacto en el Diseño y Desarrollo de Videojuegos:

- **Monument Valley (2014):** Este juego indie utilizó Unity para su desarrollo, destacando la capacidad del motor para crear experiencias visuales únicas y envolventes, que combina geometría imposible con paisajes oníricos.
El juego ofrece una serie de niveles diseñados con ingenio y creatividad, desafiando al jugador con rompecabezas que desafían las leyes de la física.
La historia del juego se cuenta a través de la interacción del jugador con el entorno, creando una experiencia narrativa envolvente y evocadora.



- **Hollow Knight (2017):** Otra joya indie, Hollow Knight, muestra cómo Unity puede ser utilizado para crear juegos de plataformas 2D con un diseño artístico distintivo, con su arte pixelado y su atmósfera oscura. Ofrece un vasto mundo subterráneo lleno de áreas interconectadas y secretos por descubrir, dando una experiencia de exploración profunda y satisfactoria.
El juego presenta un sistema de combate fluido y desafiante, con una variedad de enemigos y jefes únicos que requieren habilidad y estrategia para superar.



- **Pokémon Go (2016):** Aunque no se desarrolló exclusivamente con Unity, *Pokémon Go* utilizó el motor para algunas partes de su desarrollo, destacando la capacidad del motor para juegos móviles de gran escala y popularidad. Utiliza la tecnología de realidad aumentada

(AR) para superponer criaturas virtuales en el mundo real, lo que ofrece una experiencia de juego única y envolvente.

Los jugadores pueden explorar el mundo real en busca de Pokémons, promoviendo la actividad física y la interacción social en entornos urbanos y naturales.

El objetivo principal es la captura de Pokémons y participar en batallas virtuales en gimnasios y otros eventos.



Impacto en las Expectativas de los Jugadores:

El uso generalizado de Unity en juegos populares ha influido en las expectativas de los jugadores en cuanto a la accesibilidad, calidad visual y versatilidad de plataformas. La capacidad de Unity para facilitar el desarrollo de juegos ha contribuido a una gama de diversidad de títulos disponibles, satisfaciendo las demandas de una audiencia ansiosa de nuevas experiencias. Tanto así, que Unity se metió en proyectos tanto de realidad virtual como de aumentada, llevándolos a generar una mayor expectativa y experiencias inmersivas por parte de los jugadores.



**UNREAL
ENGINE**

Unreal Engine

Motor desarrollado por **Epic Games** y siendo conocido por sus gráficos de alta fidelidad y su potente motor de renderizado.

Cuenta con gráficos realistas, herramientas de desarrollo avanzadas, un amplio soporte para plataformas y un sistema de secuencias cinematográficas.

◎ Ventajas:

- ❖ Cuenta con una prodigiosa potencia gráfica.
- ❖ Teniendo una mayor flexibilidad en el diseño de sus juegos.
- ❖ En cuanto al Desarrollo, utiliza la creación de juegos AAA, es decir, títulos de alta calidad y alto presupuesto.



◎ Fortalezas:

Gráficos de Alta Fidelidad:

- Como ya hemos dicho, tiene una impresionante capacidad gráfica. Sus capacidades de renderizado permiten la creación de entornos y personajes visualmente sorprendentes, haciéndose muy popular.

Herramientas de Desarrollo Avanzadas:

- Ofrece herramientas de desarrollo avanzadas, como por ejemplo, un sistema de secuencias cinematográficas (*Sequencer*) que facilita la creación de cinematográficas y escenas complejas, como en una película.

Blueprints y Visual Scripting:

- La implementación de *Blueprints* permite a los desarrolladores crear lógica de juego SIN necesidad de programación, siendo obviamente una gran innovación en la creación y diseño de prototipos.

Comunidad Activa y Soporte Continuo:

- Unreal Engine cuenta con una comunidad activa, que comparte sus conocimientos y cuestiones. Además de un amplio soporte en línea, y una documentación oficial plenamente extensa.

Desarrollo Multiplataforma:

- Al igual que Unity, Unreal Engine es capaz de producir juegos para una variedad de plataformas, incluyendo PC, consolas y dispositivos móviles.

◎ Limitaciones:

Curva de Aprendizaje Pronunciada:

- Si nos centramos en el aprendizaje, puede ser muy costoso, en comparación con algunos otros motores. Sus numerosas características avanzadas pueden requerir mucho tiempo para dominarse.

Requisitos de Hardware:

- Los títulos desarrollados con Unreal Engine a menudo tienen requisitos de hardware más elevados, por lo que se destinan a una audiencia con hardware más limitado.

Por Hardware nos referimos a:

- **Procesador:** Intel Core i5 de séptima generación o equivalente
- **Memoria RAM:** 8 GB
- **Tarjeta Gráfica:** NVIDIA GeForce GTX 1060 o AMD Radeon RX 580 (o superior) con al menos 4 GB de VRAM
- **Almacenamiento:** 50 GB de espacio disponible en disco
- **Sistema Operativo:** Windows 10 (64-bit)

Licencia con Regalías:

- Aunque Unreal Engine es gratuito para el desarrollo, se aplica una cierta tarifa a los proyectos comerciales que superen ciertas cifras, lo que afecta a la rentabilidad de ciertos desarrollos independientes.

Mayores Recursos Iniciales Requeridos:

- Unreal Engine requiere de mayores recursos iniciales de tiempo y habilidades para configurar un proyecto, especialmente para desarrolladores iniciados.

Tamaño de Archivos de Juego:

- A menudo, los juegos tienen tamaños de archivo más grandes, afectando los tiempos de descarga y almacenamiento, especialmente en plataformas móviles.

Origen y Tipo de Unreal Engine:

Unreal Engine tuvo su origen en el año 1998 con el lanzamiento de su primera versión.

Originalmente fue diseñado para el juego ***Unreal***, pero el motor evolucionó para convertirse en una herramienta completa de desarrollo de juegos. Actualmente, Unreal Engine es clasificado como un motor de juegos de **primera persona (FPS)**, demostrando ser versátil para diversos géneros y experiencias, incluyendo juegos de acción, aventuras, simuladores y mucho más.

Evolución Cronológica:

- ***Unreal Engine 1 (1998)***: La primera versión del motor se utilizó para el juego "Unreal", siendo un BUM, un FPS que destacó por sus gráficos avanzados en ese momento.
- ***Unreal Engine 2 (2000)***: Se implementaron mejoras gráficas y se introdujo soporte para juegos de consola. Títulos notables incluyen *Unreal Tournament 2003* y *Unreal Championship*.
- ***Unreal Engine 3 (2006)***: Esta versión obtuvo capacidades avanzadas de renderizado y física. Sacando juegos influyentes como *Gears of War*, *Bioshock* y *Mass Effect* se desarrollaron con Unreal Engine 3.
- ***Unreal Engine 4 (2012)***: Con un énfasis en la accesibilidad y la calidad visual, UE4 se convirtió en un referente en la industria. Títulos destacados incluyen *Fortnite*, *Street Fighter V* y *Kingdom Hearts III*.
- ***Unreal Engine 5 (2021 - En adelante)***: La última iteración, UE5, ha introducido tecnologías innovadoras como *Nanite* para detalles geométricos y *Lumen* para iluminación global dinámica. Con tanto hasta dia de hoy con su última actualización, **UE 5.3**, añadiendo la opción de implementar sus juegos en aún más plataformas, como *ARKit*, *ARCore*, *OpenXR*, *SteamVR*, *Oculus* y *SteamDeck*.

Títulos Influyentes y su Impacto:

Gears of War (2006): Este título, desarrollado con Unreal Engine 3, destacó la capacidad del motor para crear juegos de acción intensa y frenética con gráficos impresionantes, influyendo en el estándar visual de la industria. Tiene un combate en tercera persona, que incluye enfrentamientos con hordas de enemigos y batallas contra jefes.

El juego ofrece modos cooperativos y multijugador que permiten a los jugadores unirse en línea o en red local para completar la historia o competir en modos PvP.

Gears of War presenta una historia profunda y una ambientación postapocalíptica, con personajes memorables y un mundo devastado por la guerra.



Fortnite (2017): Desarrollado con Unreal Engine 4, *Fortnite* ha sido un fenómeno cultural global. Su éxito ha influido en el diseño de juegos, estableciendo nuevos récords de comunidad y eventos en el juego.

Fortnite se popularizó por su modo Battle Royale, donde cientos de jugadores en línea luchan en un mapa hasta que solo queda uno. Este modo ofrece una experiencia competitiva y emocionante. Además cuenta con una característica distintiva, su mecánica de construcción, que permite a los jugadores crear estructuras defensivas para protegerse y ganar ventaja durante los combates. El juego se actualiza regularmente con eventos especiales y temporadas que introducen nuevas mecánicas de juego y otros cambios.



Street Fighter V (2016): La icónica serie de juegos de lucha, con controles precisos, combos elaborados y mecánicas de lucha estratégica que desafían a los jugadores a dominar las habilidades de cada personaje, adoptó Unreal Engine 4, mostrando que el motor no solo es apto para juegos de disparos, sino también para géneros variados como juegos de lucha. El juego cuenta con una amplia gama de personajes, cada uno con su propio conjunto de movimientos especiales, estilos de lucha y habilidades únicas, lo que permite a los jugadores encontrar un estilo que se adapte a sus preferencias.



Impacto en las Expectativas de los Jugadores:

Unreal Engine ha tenido un impacto significativo en las expectativas de los jugadores, elevando el listón en términos de **gráficos, físicas y experiencias altamente inmersivas**. Los títulos desarrollados con Unreal Engine se han convertido en juegos muy reconocidos en la industria, contribuyendo a la demanda de juegos visualmente impresionantes, influyendo en el estándar visual de la industria, elevando las expectativas de los jugadores en términos de calidad visual y realismo. La introducción de Unreal Engine 5 con tecnologías revolucionarias ha generado expectativas aún mayores y se espera que esto siga afectando positivamente las experiencias de juego futuras, con los desarrolladores llevando a cabo proyectos ambiciosos.



Godot Engine

Godot es un motor de juegos de **código abierto y gratuito**, adecuado para desarrolladores independientes y pequeños equipos, ganado popularidad en la comunidad de desarrollo de juegos indie.

Cuenta con un sistema de nodos para diseño visual, soporte 2D y 3D, editor integrado y completamente gratis, sin tarifas a futuro.

Aquí se presenta una análisis crítica de Godot Engine, considerando tanto sus ventajas y fortalezas como sus posibles limitaciones:

◎ Ventajas:

- ❖ Completamente gratuito y de código abierto
- ❖ Comunidad activa
- ❖ Fácil de aprender
- ❖ Adecuado para proyectos de diferentes tamaños.

◎ Fortalezas:

Código Abierto y Gratuito:

- Naturaleza de código abierto + su licencia gratuita. Siendo accesible para desarrolladores independientes y equipos con presupuestos ajustados.

Lenguaje de Propio:

- Utiliza su propio lenguaje de scripting, ***GDScript***, diseñado para el desarrollo de juegos. Es **fácil de aprender** y ofrece una sintaxis clara, facilitando la programación para aquellos que no tienen experiencia.

Soporte para Múltiples Plataformas:

- Permite exportar juegos a una amplia variedad de plataformas, como *PC*, *consolas*, *dispositivos móviles* y la *web*. Llegando a diversas audiencias.

Motor 2D y 3D Integrado:

- Ha mejorado mucho su soporte 3D en versiones más recientes. Permitiendo crear una amplia gama de juegos, desde simples 2D hasta proyectos 3D más complejos.

Nodo y Sistema de Escena:

- Sistema de nodos y escenas en Godot facilita la organización y estructuración del juego. Permite crear jerarquías de objetos y gestionar interacciones intuitivamente.

◎ Limitaciones:

Menor Base de Usuarios y Recursos:

- Ha tenido un crecimiento significativo, pero su comunidad es más pequeña en comparación con Unity o Unreal Engine. Por tener una cantidad menor de recursos y tutoriales disponibles en línea.

Menos Recursos Gráficos Predefinidos:

- Carece de una tienda de activos tan extensa. Necesitando crear más activos desde cero o depender de la creación interna.

Menor Adopción en la Industria AAA:

- Es muy popular entre desarrolladores indie y para proyectos más pequeños. Su adopción en la industria AAA es muy limitada en comparación con otros motores.

Actualizaciones de Características más Lentas:

- Algunas actualizaciones de características pueden ser más lentas en comparación con motores más grandes y financiados.

Origen y Tipo de Godot Engine:

Godot Engine fue creado por **Juan Linietsky** en el año **2007**. Su objetivo principal era proporcionar una herramienta accesible y poderosa para el desarrollo de juegos, especialmente orientada a **proyectos independientes y pequeños estudios**. Siendo un motor multipropósito con juegos 2D y 3D, con su propio lenguaje de scripting llamado ***GDScript***.

Evolución Cronológica:

- ***Godot Engine 1.0 (2007)***: Su primer lanzamiento estableció las bases para el motor de código abierto. Con herramientas para el desarrollo en 2D y una arquitectura modular.
- ***Godot Engine 2.0 (2016)***: La versión 2.0 introdujo el soporte para juegos 3D. Se mejoraron las capacidades y la interfaz de usuario, ampliando la versatilidad del motor.
- ***Godot Engine 3.0 (2018)***: Esta versión trajó mejoras en el rendimiento, la capacidad de renderizado 2D y 3D, así como la introducción de *GDScript* como lenguaje principal.

- **Godot Engine 3.1 (2019):** Se introdujeron más características y mejoras, incluyendo soporte para *Vulkan*(mejor desarrollo en hardware), mejoras en el editor visual y una mayor eficiencia en el uso de recursos.
- **Godot Engine 3.2 (2020):** Continuó con mejoras en la estabilidad, rendimiento y funcionalidades. Se agregaron herramientas de animación avanzadas y un soporte mejorado para móviles.
- **Godot Engine 3.3 (2021):** Se enfocó en la estabilidad y la mejora.
- **Godot Engine 3.5(2022):** Versión más estable y con más mejoras
- **Godot Engine 4.2.1(2023 - más allá):** Versión que incluye soporte para .NET y C#1.

Títulos Influyentes y su Impacto:

Debido a su enfoque en juegos independientes y proyectos más pequeños, Godot no tiene títulos AAA influyentes. Sin embargo, ha sido utilizado en una variedad de juegos indie exitosos, incluyendo:

- **Fist of the Forgotten.** El juego se centra en la mecánica de movimiento basada en el impulso, deslizamiento y balanceo para cubrir grandes distancias. El juego también cuenta con una animación fluida y una física de alta velocidad, permitiendo una experiencia de juego suave incluso en monitores que muestran más de 100 fotogramas por segundo. El juego ofrece un sistema de combate profundo y satisfactorio, donde los jugadores pueden ejecutar una variedad de movimientos y combos fluidos para derrotar a sus enemigos. Cuenta con exploración de un vasto mundo abierto lleno de secretos, misterios y peligros. Desde ruinas antiguas hasta densos bosques. La trama del juego sigue la historia de un héroe olvidado que busca redimir su pasado y desentrañar los secretos de un antiguo mal que amenaza con destruir el mundo. A medida que avanza, el jugador descubre nuevos personajes, giros inesperados y momentos emocionantes.



- **Brotato:** Es un videojuego indie que combina la jugabilidad de los *roguelike* con el estilo de supervivencia por oleadas, en el que eres una patata que mata monstruos.



- **Usagi Shima:** Es un juego casual desarrollado por Godot Engine. Centrado en la relajación y en distraerte acariciando conejos. Juegos para decorar y personalizar a tu gusto, tu granja de conejos y así ir ganando más fama.



Impacto en las Expectativas de los Jugadores:

Godot Engine ha impactado las expectativas de los jugadores al demostrar que un motor de código abierto y gratuito puede competir en términos de funcionalidad y rendimiento con motores más comerciales. Su adopción en juegos indie, ha llevado a la creación de juegos creativos y únicos que han ganado reconocimiento en la industria. Aunque no ofrece la misma capacidad gráfica que algunos motores AAA, ha influido en que la percepción de libertad creativa. La creciente comunidad y el continuo desarrollo del motor han contribuido a cambiar las expectativas sobre lo que es posible lograr, especialmente para proyectos más pequeños y equipos independientes. La elección de Godot dependerá de los objetivos específicos del proyecto y del tipo de juego que los desarrolladores desean crear.

Desarrollado por **Crytek**, CryEngine es conocido por su impresionante **calidad visual**, siendo muy utilizado en el desarrollo de juegos con mucho énfasis en gráficos avanzados. Cuenta con gráficos de alta calidad, iluminación avanzada, herramientas de desarrollo potentes.

© **Ventajas:**

- ❖ Excelente para juegos con altos estándares visuales.
- ❖ Integración fácil con herramientas externas.



© **Fortalezas:**

Impresionantes Gráficos y Efectos Visuales:

- Cuenta con gran capacidad gráfica. Ofrece un renderizado avanzado y efectos visuales de alta calidad, destacándose en la creación de entornos realistas.

Física Realista:

- Implementa física realista, permitiendo interacciones detalladas con el entorno. Trayendo experiencias en juegos más inmersivos.

Entornos Amplios y Detallados:

- Es eficaz para crear entornos extensos y detallados, como juegos de mundo abierto y exploración.

Herramientas de Desarrollo Potentes:

- Tiene herramientas de desarrollo avanzadas para crear mundos complejos y efectos visuales impresionantes.

Ecosistema de la Comunidad:

- Cuenta con una comunidad activa que comparte recursos, tutoriales y activos, facilitando el proceso de desarrollo y la solución de problemas.

© **Limitaciones:**

Curva de Aprendizaje Pronunciada:

- Al igual que Unreal Engine, puede ser muy complicado de aprender, dificultando que los desarrolladores principiantes se familiaricen rápidamente con todas sus capacidades.

Requisitos de Hardware Elevados:

- Para aprovechar al máximo su potencial gráfico, se requiere de un hardware de gama alta, limitando su accesibilidad para algunos usuarios y desarrolladores.

Menos Flexibilidad en Géneros de Juegos:

- Es un motor versátil, pero le cuesta adaptarse a diversos géneros de juegos.

Licencia y Costos:

- Utiliza un modelo de licencia que incluye el pago de regalías en función de los ingresos generados por el juego. Afectando la rentabilidad de proyectos independientes.

Menos Documentación Disponible:

- En comparación con motores como Unity o Unreal Engine, la documentación y recursos disponibles son menos extensos, complicando la resolución de problemas.

Origen y Tipo de CryEngine:

CryEngine, desarrollado por **Crytek**, tuvo su origen en el **2002**, inicialmente creado para el juego *Far Cry* lanzado en 2004. Este motor, también se clasifica como un motor de juegos de primera persona (FPS) y se ha destacado por su capacidad gráfica avanzada, sobre todo en entornos extensos y detallados.

Evolución Cronológica:

- ***CryEngine (2004)***: El lanzamiento de *Far Cry* fue el debut de CryEngine. Este juego se destacó por sus impresionantes gráficos y entornos abiertos, estableciendo el estándar para el motor en términos de calidad visual.
- ***CryEngine 2 (2007)***: Se mejoraron los gráficos y la capacidad de renderizado. Con la salida de *Crysis* se convirtió en un título influyente, conocido por su impacto visual y su demanda de hardware de gama alta.
- ***CryEngine 3 (2009)***: Se introdujo el soporte para consolas, con mejoras gráficas y herramientas de desarrollo. *Crysis 2* y *Crysis 3* destacaron por sus efectos visuales y diseño.
- ***CryEngine V (2016)***: Se anunció como una versión mejorada y más accesible, con un enfoque en la facilidad de uso y una mayor diversidad de géneros de juegos. *Prey* y *Sniper Ghost Warrior 3* fueron algunos títulos destacados.
- ***CryEngine 5.6 (2021)***: Crytek ha pasado ya varios desafíos financieros, aun así han continuado actualizando el motor en términos de gráficos y rendimiento.

Títulos Influyentes y su Impacto:

Far Cry Series (2004 en adelante): Serie de videojuegos de acción en primera persona distribuida por **Ubisoft**. Su éxito de la serie *Far Cry* ayudó a consolidar la reputación de CryEngine en la creación de entornos abiertos y gráficos avanzados.

Cada juego ofrece una nueva historia y personajes, pero todos comparten elementos comunes de gameplay y temática. Desde las aventuras de *Jack Carver* en el archipiélago de Micronesia en el 1, pasando por la misión de un mercenario para eliminar al proveedor de armas *Far Cry 2* (2008), hasta el viaje de *Jason Brody* para rescatar a sus amigos en *Far Cry 3* (2012), cada juego ofrece una experiencia única en un mundo abierto.



Crysis Series (2007 en adelante): La serie *Crysis* es conocida por su acción intensa y su enfoque en la acción de disparos en primera persona combinada con elementos de ciencia ficción. La trama gira en torno a soldados equipados con nanotrajes avanzados que les otorgan habilidades sobrehumanas. Los juegos se desarrollan en entornos de mundo abierto

y ofrecen una combinación de combate táctico, exploración y enfrentamientos contra enemigos alienígenas y fuerzas militares enemigas. Se convirtió en un referente para los juegos de disparos en primera persona, destacando en los gráficos y su física realista.



Ryse: Son of Rome (2013): Es un juego de acción y aventuras, desarrollado en la Roma antigua durante el apogeo del Imperio Romano. El juego sigue la historia de *Marius Titus*, un soldado romano que busca vengar la muerte de su familia y restaurar el honor de Roma. Ryse combina combates cuerpo a cuerpo con secuencias cinematográficas y elementos de estrategia táctica. El juego se centra en la lucha contra bárbaros y otros enemigos. Es conocido por su impresionante diseño visual y sus mecánicas de combate fluidas. Aunque recibió opiniones mixtas, es una experiencia visual super inmersiva, especialmente en el ámbito histórico y cinematográfico.



Impacto en las Expectativas de los Jugadores:

CryEngine ha tenido un impacto significativo en las expectativas de los jugadores al establecer un estándar visual elevado en el diseño y desarrollo de videojuegos. Sus títulos más influyentes, han contribuido a la demanda de experiencias de juego visualmente impactantes. La capacidad del motor

para representar entornos expansivos y detallados ha influido en la expectativa de mundos de juego inmersivos y realistas. Pero, su exigencia de hardware de gama alta y su complicado aprendizaje lo han limitado en su aceptación en comparación con otros motores más accesibles.



Lumberyard

Desarrollado por **Amazon**, *Lumberyard* es un motor de juegos **gratuito** y potente, enfocado en juegos en la nube y experiencias multijugador.

Integra servicios en la nube de Amazon, herramientas de desarrollo robustas y un amplio soporte para plataformas.



© Ventajas:

- ❖ Enfoque en juegos en la nube
- ❖ Integración con servicios web de Amazon
- ❖ Gratuito.

© Fortalezas:

Integración con Servicios de Amazon Web Services (AWS):

- Integración con los servicios de **AWS**. Permitiendo a los desarrolladores aprovechar servicios en la nube como el almacenamiento, la IA y la gestión de usuarios de manera fácil.

Gráficos y Física de Alta Calidad:

- Ofrece capacidades gráficas avanzadas y un sistema de físicas. Siendo muy beneficioso para proyectos que requieren gráficos de alta calidad y simulación realista.

Herramientas de Creación de Niveles y Cinemáticas:

- El motor proporciona herramientas para la creación de niveles y cinemáticas, facilitando a los desarrolladores el diseño y la implementación de mundos y escenas más complejas.

Sopporte Multiplataforma:

- Puede mandar juegos a varias plataformas, como *PC*, *consolas* y *dispositivos móviles*. Brindando flexibilidad a los desarrolladores para llegar a distintas audiencias.

Licencia Gratuita y Sin Regalías:

- Es totalmente gratuito y NO impone regalías sobre los ingresos generados, siendo diferente y atractivo para estudios independientes y desarrolladores con presupuestos bajos.

© Limitaciones:

Menor Comunidad y Documentación:

- Lumberyard tiene menos documentación disponible y una comunidad mucho más pequeña, en comparación con motores como Unity o Unreal Engine. Esto resulta más complejo, a la hora de resolver problemas y obtener ayuda.

Dificultad de Aprendizaje:

- No solo eso, sino que también es difícil de aprender, con herramientas y una interfaz especialmente complicada para desarrolladores novatos.

Menos Activos y Recursos Disponibles:

- Su cantidad de activos y recursos disponibles es menor que otros motores. Requiriendo de la creación de activos desde cero o cojer recursos externos.

Menor Adopción en la Industria:

- Su adopción es menor en la industria en comparación con motores más establecidos como Unity o Unreal Engine.

Origen y Tipo de Lumberyard:

Amazon *Lumberyard* es un motor de juegos lanzado por Amazon en 2016. Su origen se encuentra en la adquisición de la empresa de desarrollo de juegos **Crytek**, que resultó en la adopción de la tecnología *CryEngine* como base para Lumberyard. Se clasifica como un motor de juegos de alta gama, con un enfoque en gráficos avanzados, física realista y la integración con servicios de la nube de Amazon Web Services (AWS).

Evolución Cronológica:

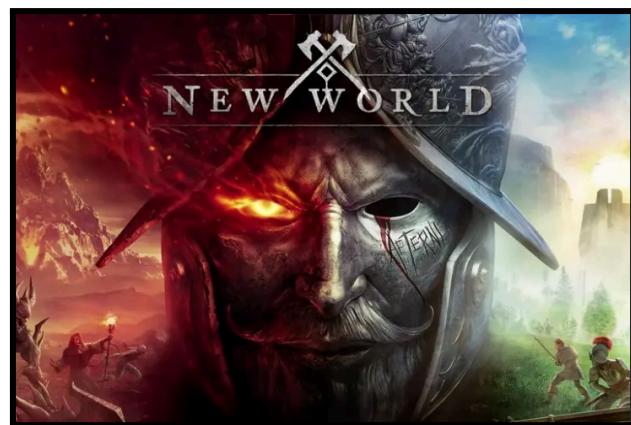
- **Lumberyard 1.0 - Lanzamiento Inicial (2016):** La versión inicial de Lumberyard heredó muchas características de CryEngine y se centró en la creación de juegos con gráficos de alta calidad y gran simulación física.
- **Actualizaciones y Mejoras Continuas:** A lo largo de los años, Lumberyard ha tenido varias actualizaciones para mejorar la estabilidad, rendimiento y características, usando distintas versiones como **1.2, 1.4, 1.5...** Se han añadido herramientas para la creación de niveles, cinemáticas y la integración con servicios en la nube de AWS.
- **Lumberyard 1.20 - 1.28 (2021 - en adelante):** Lanzamiento de New World, que influyó en la percepción de Lumberyard en la industria. Sacando más versiones, hasta la 1.28. Y anunciando recientemente el lanzamiento de una nueva versión renombrada como **O3DE** (Open 3D Engine), siendo el sucesor de Lumberyard. Este software se publicará bajo la licencia de código abierto **Apache-2.0**.

Títulos Influyentes y su Impacto:

New World (septiembre de 2021): MMORPG (Juego de rol multijugador en línea) desarrollado por *Amazon Game Studios*.

El juego está ambientado en un mundo de fantasía del siglo XVII, en América. Y cuenta con una jugabilidad de exploración, combate y construcción. Los jugadores pueden formar facciones, forjar alianzas y enfrentarse a otros jugadores en combate PvP (jugador contra jugador), o explorar mazmorras y enfrentarse a criaturas peligrosas en entornos PvE (jugador contra entorno).

El juego se centra en la supervivencia y la conquista territorial, donde se compite por el control de diferentes regiones del mundo. Recibe actualizaciones y expansiones regularmente, para mejorar y expandir la experiencia de juego.



Crucible (Mayo de 2020): Era un shooter(juego de disparos) en tercera persona que combinaba elementos de MOBA (partidas competitivas en tiempo real entre dos equipos de jugadores) y juegos de disparos. Ambientado en un mundo alienígena, los jugadores luchaban en equipos para completar objetivos y derrotar a criaturas y enemigos controlados por otros jugadores. Sin embargo, el juego NO recibió buenas críticas y Amazon retiró el juego de la venta en noviembre de 2020.



Impacto en las Expectativas de los Jugadores:

El impacto de Lumberyard en las expectativas de los jugadores es más limitado en comparación con algunos motores más establecidos. Su integración con AWS y capacidades gráficas avanzadas son aspectos positivos, pero la menor adopción en la industria y la falta de una comunidad tan escasa afecta a la percepción general de los jugadores.

COMPARACIÓN entre los distintos motores

Vamos a comparar los motores de juegos mencionados ([Unity](#), [Unreal Engine](#), [Godot Engine](#), [CryEngine](#) y [Lumberyard](#)) en varios aspectos:

	Unity	Unreal Engine	Godot Engine	CryEngine	Lumberyard
Modelo de Licencia	Propietario, versión gratuita y planes de suscripción.	Propietario, gratuito con regalías en ingresos.	Código abierto, gratuito y sin regalías.	Propietario, gratuito con regalías en ingresos.	Gratis y sin regalías.
Accesibilidad y Aprendizaje	Fácil de aprender, amplia documentación y gran comunidad.	Más complejo, pero potente, aprendizaje complicado.	Fácil de aprender, con una comunidad activa.	Aprendizaje medio/alto, con menos documentación.	Aprendizaje complicado.
Gráficos y Capacidad 3D y 2D	Capaz en 3D, pero inicialmente se centró en juegos 2D.	Excelente en gráficos 3D, destacando en la industria.	Mejoras recientes en capacidades 3D y fuerte en 2D.	Potente en gráficos realistas, especialmente en entornos exteriores.	Fuerte en gráficos 3D y física avanzada.
Servicios en la Nube	Ofrece servicios en la nube <i>Unity Cloud</i> , pero no tan integrados como en otros motores.	Puede integrarse con servicios en la nube de terceros, pero no propiamente.	No tiene una integración nativa con servicios en la nube.	No cuenta con servicios en la nube.	Integración directa con Amazon Web Services (AWS).
Comunidad y Ecosistema	Gran comunidad, extensa tienda de activos y abundantes recursos en línea.	Comunidad activa, amplia documentación y tienda de activos.	Comunidad creciente, con recursos y activos disponibles.	Comunidad más pequeña, menos recursos disponibles en comparación con otros.	Comunidad más pequeña y menos recursos en comparación con otros motores.
Títulos Influyentes	Gran versatilidad, sacando muchos títulos exitosos en diversas plataformas.	Creador de los juegos más icónicos y exitosos de la última década.	Ha ganado popularidad entre los juegos indie, produciendo títulos notables.	Reconocido por su gran renderizado gráfico y utilizado en juegos de renombre.	Opción viable para el desarrollo de juegos, dentro del ecosistema de Amazon.

La elección del motor de juegos depende de factores como la experiencia del desarrollador, los objetivos del proyecto y la preferencia de gráficos y servicios:

Unity y **Unreal Engine** ⇒ son líderes establecidos con fortalezas en diferentes áreas.

Godot ⇒ Es accesible y versátil.

CryEngine ⇒ Destaca en gráficos.

Lumberyard ⇒ Se integra estrechamente con los servicios en la nube de AWS.

Cada motor de juegos se ha utilizado para una amplia variedad de géneros y estilos de juegos. Por ejemplo, juegos como "*Cuphead*" y "*Hollow Knight*" fueron desarrollados con Unity, aprovechando su versatilidad y facilidad de uso para juegos 2D.

"*Fortnite*" y "*Gears of War*" son ejemplos notables de juegos desarrollados con Unreal Engine, aprovechando sus capacidades gráficas y de mundo abierto.

TENDENCIAS

Las tendencias y tecnologías actuales en la industria de los videojuegos proporcionan una gran vista al futuro de los motores de juegos y cómo podrían impactar en diseñadores, desarrolladores y jugadores. Los diseñadores y desarrolladores, deberán adaptarse a nuevas tecnologías y paradigmas de juegos, mientras que los jugadores pueden esperar experiencias más inmersivas, personalizadas y socialmente sensatas. La flexibilidad y la adaptabilidad serán muy importantes para el éxito en la industria, en constante cambio de los videojuegos.

Algunas de esas tendencias y tecnologías, junto con sus posibles implicaciones futuras:

1) Realidad Virtual (VR) y Realidad Aumentada (AR):

Actualmente

Para aumentar las experiencias inmersivas en los juegos, ha llevado al gran crecimiento de la realidad virtual y aumentada.

Predicción Futura

Los motores de juegos deberán adaptarse para ofrecer un mejor soporte, además de herramientas específicas para VR y AR. Probablemente, esto afecte a los desarrolladores y diseñadores, quienes necesitarán habilidades adicionales, y una mayor creatividad y diseño de niveles.

2) Plataformas de Streaming de Juegos:

Actualmente

Existen plataformas de streaming con servicios de suscripción de juegos como *Google Stadia* y *Xbox Game Pass*, que están ganando popularidad. Estas plataformas son como un Netflix o HBO, pero de juegos. Con estas plataformas, en las que o bien puedes suscribirte durante un período de tiempo o comprar juegos individuales, puedes jugar con una amplia variedad de juegos, siendo así como una biblioteca. Incluso hace bien poco Netflix publicó gran parte de la saga *Grand Theft Auto* en su plataforma, para que puedas jugar con ellos a través de un dispositivo móvil, además de juegos interactivos como el *Preguntados*.

Predicción Futura

Los motores de juegos podrían evolucionar, optimizando el rendimiento a través de la transmisión y adaptarse a modelos de entrega de contenido en streaming. Esto hará que tanto los desarrolladores

creen juegos más eficientes en cuanto al ancho de banda, como los diseñadores consideren una experiencia de juego más continua y que obviamente no haya problemas.

3) Inteligencias Artificiales y Aprendizaje Automático:

Actualmente

La IA se utiliza para mejorar la experiencia e inmersión en un juego, personalizando la experiencia del usuario y creando personajes no jugables(NPC) más realistas, sobre todo en mundos abiertos y libres, como un *Grand Theft Auto*.

Predicción Futura

Los motores de juegos podrían integrar aún más capacidades de IA para crear experiencias de juego más dinámicas y personalizadas. Afectando a los desarrolladores al requerir habilidades de programación de IA y a los diseñadores, que pueden aprovechar la IA para crear narrativas más elaboradas.

Esto se ha aplicado, según filtraciones, al próximo juego de la saga *Grand Theft Auto*. En el GTA 6, los NPC estarán dotados de una inteligencia artificial **avanzada** (*AI/Memory*), indicando que recordarán las decisiones que tomemos y nuestras interacciones con ellos, influyendo en su comportamiento hacia nosotros a lo largo del juego. Además, los animales en el juego también serán más inteligentes y se comportarán de manera más realista (*Animal Intelligence*)

4) Gráficos Ray Tracing:

Actualmente

La tecnología de ray tracing es una técnica avanzada de renderización utilizada para simular cómo interactúan los rayos de luz con los objetos. Esta tecnología está mejorando la calidad visual de los juegos con efectos de iluminación más realistas. Aunque actualmente, si no tienes un buen ordenador, puede complicar tu experiencia de juego, provocando que disminuya el rendimiento, siendo muy exigente en cuanto a recursos.

Predicción Futura

Los motores de juegos se centrarán en la implementación eficiente de ray tracing para mejorar aún más los gráficos. Los desarrolladores y diseñadores deberán colaborar para aprovechar al máximo estas capacidades y crear experiencias visuales impactantes. Además de mejorar su rendimiento, sin la necesidad de un hardware potente.

5) Desarrollo Sostenible y Diversidad:

Actualmente

Estos últimos años ha habido una creciente conciencia sobre la sostenibilidad en el desarrollo de juegos, incluida la diversidad y la inclusión. Esto se ha debatido y criticado numerosas veces, llegando a clasificar juegos dependiendo de sus líneas narrativas, como con la inclusión.

Predicción Futura

Los motores de juegos pueden incorporar herramientas para facilitar el desarrollo sostenible. Además de que los desarrolladores y diseñadores se enfrentarán a presiones para crear juegos socialmente diversos e inclusivos, pudiendo influir en la elección de determinados motores que faciliten estas prácticas. y otros que podrán añadirlo como una opción que se pueda desactivar, ofreciendo un buen contenido a distintas audiencias, sin ser criticados.

EVALUACIÓN DE LOS MOTORES DE JUEGOS

Unity, Unreal Engine, Godot Engine, CryEngine y Lumberyard son motores de juegos populares utilizados en la industria del desarrollo de videojuegos, como ya hemos dicho. Ahora, estudiaremos detalladamente dos incógnitas que no hemos resuelto aún, **¿Cuál es su finalidad? Y ¿Cómo funciona?**

UNITY

Finalidad:

Unity es un motor de juegos que desarrolla juegos *2D, 3D*, además de en realidad virtual (*VR*) y realidad aumentada (*AR*).



Funcionamientos:

Se centra en la creación de escenas donde los desarrolladores pueden **colocar y manipular objetos, personajes, luces y cámaras** de manera visual, utilizando un sistema de *arrastrar y soltar*.

Su *Lenguaje de Programación* principal ⇒ **C#**



Con un Entorno de Desarrollo Integrado (**IDE**) → Permite la creación, prototipado y despliegue de juegos en **múltiples plataformas**.

Los desarrolladores pueden **programar el comportamiento** de los **objetos y personajes** mediante **Scripts** escritos en **C#**, brindando un alto grado de control sobre la interactividad y la dinámica del juego.

Unity permite **compilar y empaquetar** juegos para su distribución en **diversas plataformas**, como PC, consolas, dispositivos móviles y sistemas de realidad virtual. Esto hace que llegue a un amplio número de jugadores.

UNREAL ENGINE

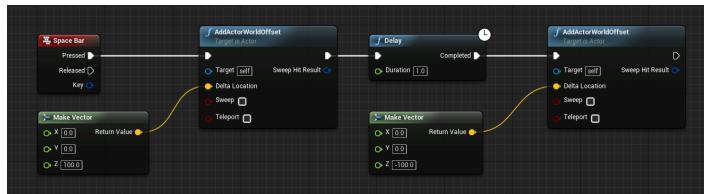
Finalidad:

A Unreal Engine se le conoce por su capacidad para desarrollar **juegos AAA con gráficos impresionantes**, jugabilidad avanzada y producción de **primer nivel**. Además de utilizar en sus títulos

una amplia gama de experiencias interactivas, como *simulaciones*, aplicaciones de entrenamiento y espacios con estructuras y edificaciones.

Funcionamientos:

Utiliza el sistema de *scripting visual*, **Blueprints**, que permite crear lógica de juego y comportamientos sin necesidad de programar con código. Este sistema permite una prototipación rápida y eficiente en el desarrollo de juegos.

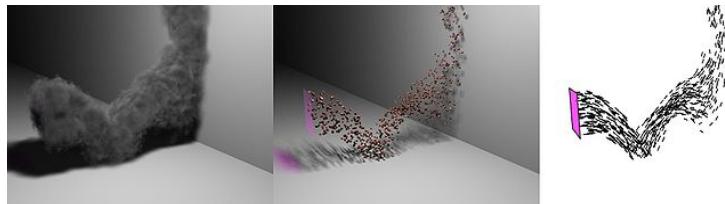


Como Lenguaje de Programación ⇒ **C++**

Lenguaje de programación avanzado para personalización profunda, que brinda a los desarrolladores un mayor control sobre el rendimiento y funcionalidad del juego, así como otras características avanzadas.

Se proporcionan **herramientas** avanzadas para la *creación de mundos, animaciones y efectos visuales*, pudiendo crear experiencias visuales impresionantes.

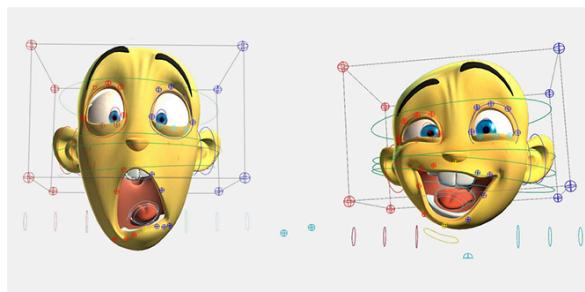
- **Sistemas de partículas**



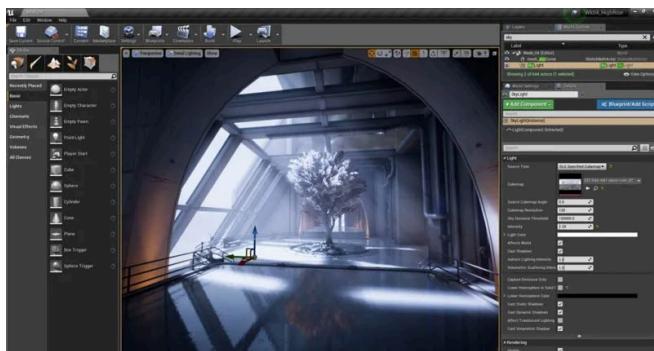
Física avanzada

$$F = m \cdot a$$

Animación rigging ⇒ Animar personajes 3D realistas y controlar sus movimientos.



Sistemas de iluminación



Renderizado de vanguardia ⇒ Técnicas y tecnologías avanzadas para generar imágenes, logrando resultados visuales de alta calidad y realismo.



Unreal Engine es **compatible** con una amplia variedad de **plataformas**, incluyendo PC, consolas de videojuegos, dispositivos móviles y realidad virtual. También, ofrece herramientas para el desarrollo colaborativo simultáneo, facilitando la colaboración e interacción en equipo.

GODOT ENGINE

Finalidad:

Godot Engine está diseñado para crear juegos en **2D y 3D**, aplicaciones interactivas y simulaciones. Su flexibilidad lo hace adecuado para una amplia gama de proyectos, desde juegos indie hasta proyectos comerciales de gran escala.

Funcionamientos:

Godot utiliza su propio Lenguaje de Programación ⇒ **GDScript**.

Diseñado exclusivamente para el desarrollo de juegos, con una sintaxis similar a **Python** que facilita su escritura de código y la creación de **scripts** para controlar el comportamiento de los objetos en el juego.



Godot es compatible con otros Lenguajes de Programación, incluyendo **C#** y **C++**. Proporcionando a los desarrolladores la flexibilidad para elegir el lenguaje que mejor se adapte a sus necesidades y preferencias, así como acceso a funcionalidades avanzadas y rendimiento optimizado.



Permite a los desarrolladores trabajar de manera colaborativa y flexible en sus proyectos. Esto se logra mediante el uso de **Blueprints** y escenas que facilitan la organización y la gestión de los elementos, además de la integración de assets.

Como Godot Engine es un motor de código abierto, la comunidad de desarrolladores pueden personalizar y extender el motor según sus necesidades. Además, estos proporcionan recursos, tutoriales y soporte técnico , aprovechando al máximo el potencial de Godot Engine.

CRYENGINE

Finalidad:

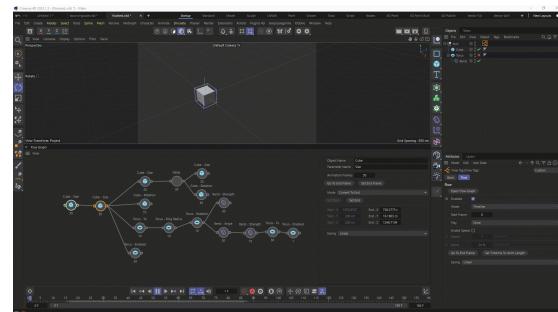
CryEngine tiene la capacidad para crear juegos inmersivos con buenos gráficos, que sumergen al jugador en mundos detallados y realistas. Se centran en la calidad visual y la inmersión, convirtiéndolo en una opción popular para desarrolladores que buscan experiencias de alta gama.

Funcionamientos:

Tiene capacidades avanzadas de animación y renderizado que permiten la creación de personajes y entornos altamente detallados y realistas. Sus sistemas de animación facial, animación de cuerpos completos y física de personajes, brindan a los desarrolladores un gran control sobre el aspecto y el comportamiento de los personajes en el juego.



CryEngine utiliza un sistema de *scripting* llamado **Flowgraph**, es exactamente igual que *Blueprints*, solo que este sistema pertenece a CryEngine y que es un poco mas complicado de aprender.

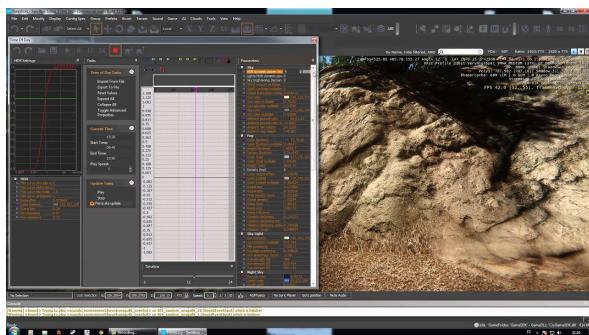


CryEngine es compatible con **C++**, permitiendo a los desarrolladores realizar programación avanzada y personalización profunda.



Tiene un conjunto de **herramientas** avanzadas para el diseño de niveles:

Editores de terreno



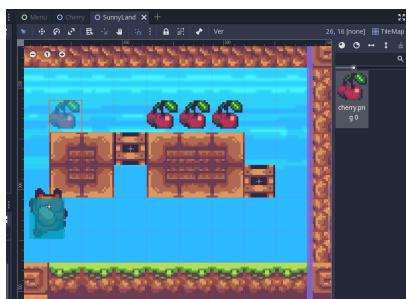
Herramientas de modelado de terreno

- Esculpido de Terreno
- Texturización de Terreno
- Materiales de Terreno
- Capas de Terreno
- Importador de Mapas de Altura de Terreno
-

Sistemas de iluminación dinámica y estática



Herramientas de colocación de objetos.



Estas herramientas permiten a los diseñadores crear entornos detallados y dinámicos, dando vida al mundo del juego.

CryEngine ofrece un **soporte** para una **variedad** de **plataformas**, incluyendo *PC, consolas de videojuegos y dispositivos móviles*. Además, proporciona herramientas y técnicas de optimización de rendimiento que permiten a los desarrolladores maximizar el rendimiento del juego y garantizar una experiencia de juego suave y fluida en todas las plataformas.

LUMBERYARD

Finalidad:

Lumberyard se especializa en la creación de juegos con servicios en la nube, autorizando a los desarrolladores a implementar características como **almacenamiento en la nube, análisis de datos,**

administración de usuarios y más. También ofrece herramientas para el desarrollo de experiencias **multijugador**, lo que facilita la creación de juegos en línea con modos de juego **cooperativo** y **competitivo**.

Funcionamientos:

Lumberyard utiliza un editor visual, **Lumberyard Editor**, para editar el contenido visualmente. Este editor proporciona una interfaz gráfica de usuario que facilita la creación y manipulación de elementos del juego, como objetos, terrenos, luces y cámaras.

Utiliza el Lenguaje de Programación ⇒ **C++**.



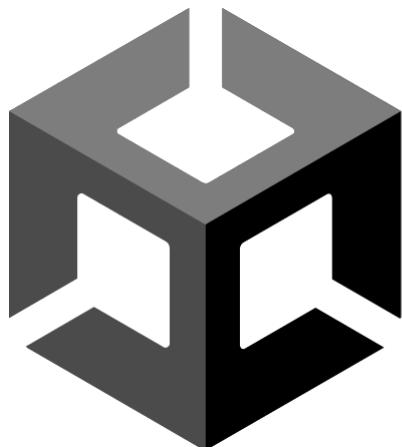
Proporciona un amplio repertorio de características para la creación de mundos, animaciones, efectos visuales y sistemas de juego en **línea**. Estas características son **editores de terreno**, **sistemas de partículas**, **herramientas de animación**, **efectos de post-procesamiento**, **sistemas de física**, etc. Estas herramientas permiten a los desarrolladores crear entornos detallados y dinámicos, personajes realistas y efectos visuales impresionantes.

Lumberyard viene integrado con *Amazon Web Services* (AWS), así los desarrolladores pueden aprovechar los numerosos servicios de la nube para agregar características avanzadas a sus juegos, como **análisis en tiempo real**, **transmisión de contenido en vivo**, **almacenamiento escalable** y más. Esta unión favorece el desarrollo de juegos escalables y altamente personalizables que pueden crecer y adaptarse a las necesidades cambiantes de los jugadores y empresas.



Pero bien, ahora debemos elegir un motor de acuerdo a nosotros para la realización de la actividad práctica.

En mi caso voy a elegir hacerlo con Unity. Pero ¿Por qué?



Unity

En primer lugar, Unity permite la creación de juegos en 2D, 3D, así como en realidad virtual (VR) y realidad aumentada (AR). Esto me permite generar cualquier tipo de género y estilo de juego, pudiendo hacer juegos de móvil fácil y simples, o juegos más complejos y de alta gama.

En segundo lugar, Unity es fácil de usar y aprender. Tiene un entorno de desarrollo integrado o solo IDE, con una interfaz intuitiva que me permite, aunque sea un novato, poner en marcha mis ideas rápidamente. La cantidad de recursos y activos disponibles en la *Asset Store* también me ayudan y me facilitan el desarrollo del juego, proporcionando acceso a una gran cantidad de contenido que puedo reutilizar, desde modelos 3D hasta scripts o efectos visuales.

Por otro lado, Unity tiene el soporte de la comunidad, personas como yo, que buscan mejorar el motor, ya sea con tutoriales u otros recursos para ayudar a diferentes trabajos y aprender nuevas técnicas. Si no, también cuenta con una buena documentación, lo que facilita la búsqueda de ayuda y la resolución de problemas.

Otro factor importante es su amplia compatibilidad multiplataforma. Permite desplegar los juegos creados en muchas plataformas diferentes, como PC, consolas de videojuegos, dispositivos móviles y sistemas de realidad virtual, siendo más accesible.

Por último, considero que es el más rápido para mí, ya que es de los pocos que he aprendido y lo considero más conocido, que los demás.

UTILIZACIÓN DE GDD

PLAN DE DESARROLLO:

Objetivo:

Mi objetivo es crear un juego en 2D y coger un personaje, que vaya esquivando objetos que caigan del cielo. Necesito un terreno firme, y un personaje al que le pueda cambiar de animación. Y debe ser en poco tiempo. Quiero que el personaje salte manualmente, en cuanto toque el suelo, así el jugador solo podrá moverse de izquierda a derecha con las flechas.

Recursos Disponibles:

Tengo a mi disposición: los assets de la página web, internet y tutoriales muy parecidos a lo que quiero hacer. Además de un gran desarrollador.

Desglose del Trabajo:

1. Conseguiremos el cuerpo de nuestro personaje, de los objetos y del entorno, ya sea en la asset store o en internet..
2. Empezaremos organizando nuestra carpeta de Asset en carpetas más pequeñas, como *entorno, script, scenes y el personaje*.
3. Encontrar un tutorial acorde a nuestras preferencias y necesidades, que sea parecido a nuestros objetivos
4. Crear GameObjects necesarios y sus características.
5. Empezar a crear el juego, en partes:
 - a) Página de juego
 - b) Página de *Game Over*
 - c) Página de Inicio,.

Asignación de Recursos:

Utilizar *Visual Studio Code*, como plataforma para la creación de Scripts.

Planificación de repeticiones:

A medida que vayamos avanzando en el desarrollo del juego se irá probando su rendimiento, jugabilidad y funcionamiento.

Revisión y Ajuste:

A medida que avancemos habrá múltiples cambios en el funcionamiento del juego. Lo cual serán causados por fallos no deseados, falta de tiempo o mejoras adicionales que hacen mejor el juego.

Un GDD (Game Design Document), o Documento de Diseño de Juego, es un **documento de guía**

detallada para el desarrollo de un videojuego. Este recurso fundamental, *describe* todos los aspectos del juego:

1. Concepto inicial
2. Mecánica
3. Historia
4. Personajes
5. Niveles
6. Arte
7. Sonido
8. Jugabilidad y mucho más.

Su **objetivo principal** es proporcionar una indicación, aclaración o una IDEA centralizada y coherente para mi como desarrollador o para los miembros de mi equipo de desarrollo, incluyendo diseñadores, artistas, programadores, sonidistas y más trabajadores.

Algunos de los elementos comunes que pueden estar presentes en un GDD incluyen:

El GDD a lo largo del desarrollo del juego se revisará y actualizará conforme el juego se desarrolla y se perfecciona. Es una buena herramienta para garantizar la coherencia y la visión del juego durante todo el proceso de producción, ayudando a mantener a todos los miembros del equipo en la misma página en términos de objetivos y expectativas.

PASOS:

Para planificar:

Paso 1: Revisión del GDD

Paso 2: Definición de Objetivos y Alcance de la Demo

Paso 3: Planificación del Desarrollo

Paso 4: Desarrollo de la Demo

Paso 5: Pruebas y Ajustes

Paso 6: Revisión y Pulido

Paso 7: Preparación para el Lanzamiento

Paso 8: Evaluación Post-Lanzamiento

JUEGO:

RUSHED PENGUIN

OBJETIVOS DEL JUEGO	34
HISTORIA	34
CONTROLES DEL JUEGO	34
PARTE FRONTAL DEL JUEGO	34
CUTSCENE DESCRIPTION	34
NO HAY ATTRACT MODE DESCRIPTION	35
TÍTULO/PANTALLA DE INICIO	35
PANTALLA DE JUEGO	35
OTRAS PANTALLAS	36
PANTALLA DE CARGA	36
MECÁNICAS UNIVERSALES	36
PERSONAJE DEL JUGADOR	38
MEDIDAS DEL JUGADOR	39
HABILIDADES DEL JUGADOR	39
HERRAMIENTAS DE INVENTARIO DEL JUGADOR	39
POTENCIADORES/MODIFICADORES DE ESTADO	39
VIDA	39
SCORE	39
RECOMPENSA Y ECONOMÍA	39
NO VEHÍCULOS	39
PERSONAJES PRINCIPALES DE LA HISTORIA	39
ESQUEMA DE PROGRESIÓN DEL JUEGO	39
CLASIFICACIÓN DE LA JUGABILIDAD	40
PANTALLA DE RESUMEN DEL MUNDO/SELECCIÓN DE NIVEL/NAVEGACIÓN	40
NIVELES DEL JUEGO	40
HUB	40
REGLAS GENERALES EN LOS ENEMIGOS	40
NO ENEMIGOS ESPECÍFICOS	40
NO JEFES	40
NON-PLAYER CHARACTERS	40
SIN COLECCIONABLES	40
SIN ESCENAS SECUNDARIAS	40
NI MÚSICA, NI EFECTOS DE SONIDO	40
PUNTOS Y PREOCUPACIONES	40

OBJETIVOS DEL JUEGO

Es del personaje, un pingüino que debe recoger diamantes que caen del cielo para aumentar sus puntos. Pero si en vez de coger un diamante, coje una bomba se termina y se vuelve a empezar.

HISTORIA

AUNQUE LA HISTORIA NO ESTÁ DESARROLLADA COMO TAL EN EL PROPIO JUEGO, YA QUE ES UN PROTOTIPO MUY POCO AVANZADO Y FALTA PRESUPUESTO Y TIEMPO, SI TIENE UN TRASFONDO.
Es un Pingüino, abandonado por su familia, que espera volver con ellos siendo rico y así él espera, que lo volverán a aceptar.

Pero como todo juego, tiene ciertos obstáculos, siendo bombas que lo separan de su meta, siendo esta coger los máximos diamantes posibles, aunque los problemas aumentarán a medida que se acerque a su objetivo.

En el Final el Pingüino consigue todos los diamantes y así gana y es rico, dándose cuenta que está mejor solo.

CONTROLES DEL JUEGO

Descripción general: El jugador se moverá de izquierda a derecha y de arriba a abajo, pudiendo dar saltos.

PARTE FRONTAL DEL JUEGO

El juego se iniciará, una vez se de inicio.

CUTSCENE DESCRIPTION

NO HABRÁ CINEMÁTICAS.

NO HAY ATTRACT MODE DESCRIPTION

TÍTULO/PANTALLA DE INICIO

Lo primero que se ve al abrir el juego, será el menú inicial.



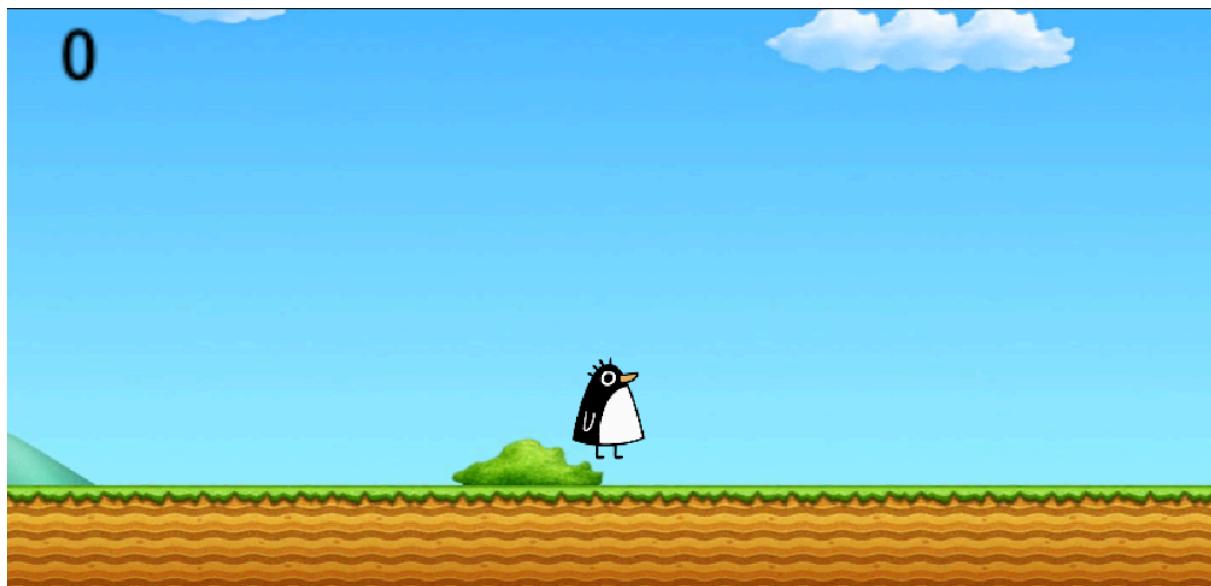
Contará con nuestro protagonista y el nombre del juego, “RUSHED PENGUIN”, que significa *Pinguino Apresurado*. Posteriormente, tendrá dos botones, los cuales habrá que pinchar con el ratón, uno para INICIAR el Juego, y otro para SALIR de él.

Además contará con el logo del creador del juego. *JodyMania*, el nuevo mejor motor de juegos.



PANTALLA DE JUEGO

Al pulsar el botón de Jugar con el ratón, en la pantalla de inicio, ya nos llevará a la pantalla de juego.



Sabes que ha comenzado el juego, si el personaje llega al suelo. Ahí comenzará el juego.

OTRAS PANTALLAS

Cuando hayas perdido, se te mostrará una pantalla pegada a la de juego, para decirte que has perdido. Con dos botones, que habrá que pinchar con el ratón, uno para VOLVER A JUGAR y otro para VOLVER AL MENÚ DE INICIO.



PANTALLA DE CARGA

De momento, al ser un prototipo, no contiene una pantalla de carga.

MECÁNICAS UNIVERSALES

• Descripción

El jugador deberá reunir tantos Diamantes como le sean posibles, para reunir puntos, mientras esquiva a las bombas, si no se va a la Interfaz de Game Over, con un botón de Reinicio. En esa Interfaz no podrás mover al personaje.

Del cielo estarán cayendo las Bombas y los Diamantes. Una vez lleguen al suelo, tardarán unos segundos en desaparecer.

• Jugador

El jugador, se encargará de dar movimiento al pingüino, para que sobreviva el mayor número de tiempo posible.

Objetos no rompibles

Encontramos dos Objetos:

1. Diamantes:

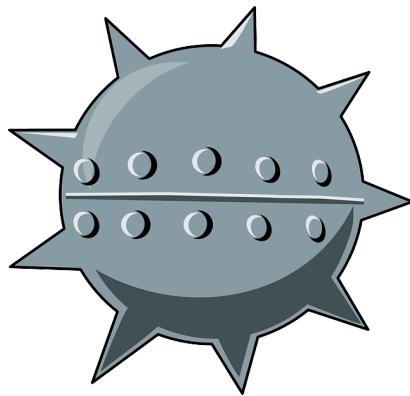
Los Diamantes son los responsables de que se te añadan puntos.

Pueden caer de arriba en cualquier lado del pequeño mapa. Y al tocar el suelo, tardan 4 segundos en desaparecer.



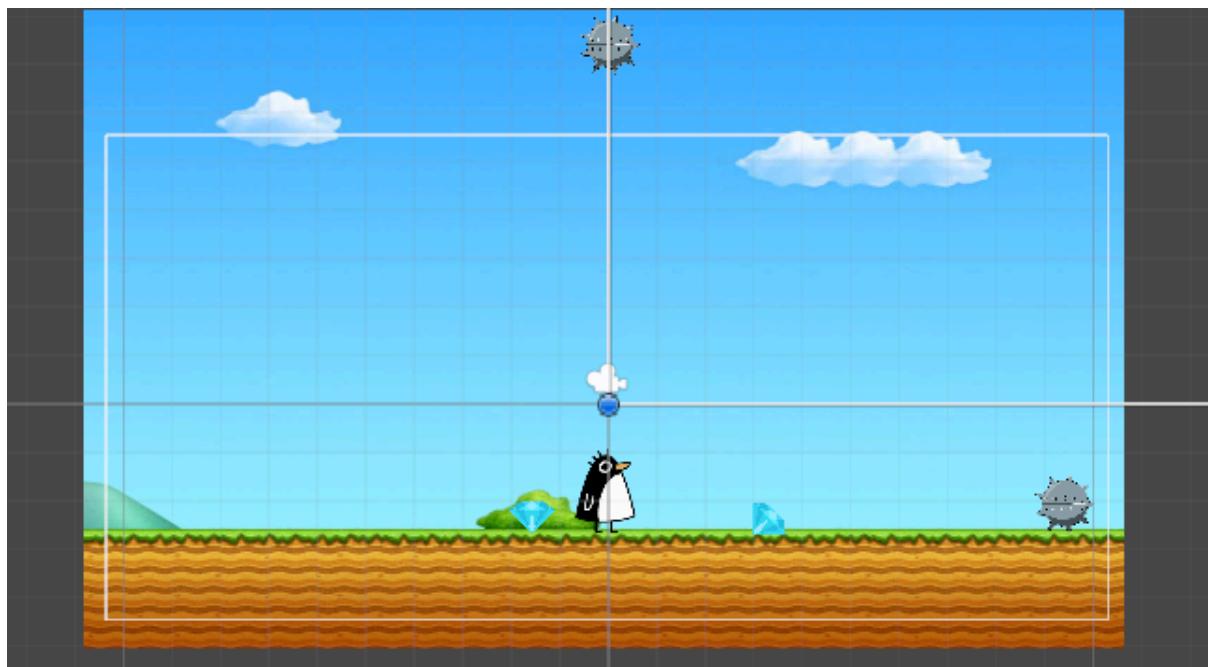
2. Bombas

Las Bombas serán las que harán perder, sacando la Interfaz de Reinicio. Estas bombas, al igual que los Diamantes, caen aleatoriamente por el mapa. y tardarán 5 segundos en desaparecer.



Cámara:

La perspectiva que se utilizará, será de frente, al jugador, viéndose todo el terreno de 2D y los objetos caer.



Para lograr moverse se necesita, mínimo un teclado y ratón , para que el personaje pueda tener movimiento, y para poder acceder al juego desde el menú de inicio.

Controles:

Para poder mover el personaje se necesitan las teclas de la flecha a un lado y al otro. Esto le moverá de izquierda a derecha con un paso seguido. Además de saltar, con la tecla espacio del teclado, únicamente cuando esté tocando el suelo, no pesa mucho así que al saltar y girar a uno de los lados, parece que planea.

Puntuación:

Para que el jugador pueda conseguir sus puntos debe chocar con los Diamantes, de esta manera, arriba a la izquierda se sumarán de uno en uno, hasta que tu jugador perezca, en todo caso se detendrá ahí.

Guardar/Cargar:

El prototipo a dia de hoy no tiene ningún punto para guardar, pero se está mirando una forma de guardar tus puntos acumulados, en records, y cada vez que lo superes que lo pongo como puntos superados. Como si fuera un speedrun, con un mensaje de tu máxima puntuación.

PERSONAJE DEL JUGADOR

El Nombre del jugador, aunque no se especifica en el prototipo, se llamará Pingui.



MEDIDAS DEL JUGADOR

En comparación con otros elementos, como las bombas o los diamantes, el personaje es más o menos igual.

Se mueve de derecha a izquierda.

Cuando el personaje toque un Diamante, además de notar la colisión, el diamante desaparecerá y se sumará un punto.

Cuando el personaje toque una Bomba se le llevará a la escena de *Game Over*, y tendrá la opción de volver a Jugar.

HABILIDADES DEL JUGADOR

SIN HABILIDADES, SOLO MOVIMIENTO

HERRAMIENTAS DE INVENTARIO DEL JUGADOR

EL JUGADOR NO CONTARÁ CON UN INVENTARIO

POTENCIADORES/MODIFICADORES DE ESTADO

NO POTENCIADORES

VIDA

El jugador contará con una única vida, que será quitada una vez choque con una de las Bombas.

SCORE

El sistema de puntos sirve para algo:

Cada que el jugador vaya reuniendo puntos de **5 en 5** el tiempo en destruirse las bombas irá aumentando, además del tiempo del Spawn, que aumentará un 0.5s cada vez. También en el tiempo de los Diamantes, que aumentará en el tiempo de destrucción 0.4 s cada vez.

RECOMPENSA Y ECONOMÍA

PRÓXIMAMENTE SE AÑADIRÁ UN SISTEMA PARA GENERAR RECOMPENSAS CADA CIERTOS PUNTOS.

NO VEHÍCULOS

PERSONAJES PRINCIPALES DE LA HISTORIA

SOLO ESTA PINGUI.

ESQUEMA DE PROGRESIÓN DEL JUEGO

ACTUALMENTE SOLO EVOLUCIONARÁ EL JUEGO MEDIANTE EL SISTEMA DE PUNTOS.

CLASIFICACIÓN DE LA JUGABILIDAD

Este juego, se le podría describir como un juego **casual**, debido a su falta de historia y trasfondo, es un juego que puedes jugar en cualquier franja de cualquier horario para pasar el rato, sin necesidad de dedicarle mucho rato.

PANTALLA DE RESUMEN DEL MUNDO/SELECCIÓN DE NIVEL/NAVEGACIÓN

El jueggo no irá por niveles, si no por puntuación.

Al jugador una vez empiece el juego, empezaran a caerle los objetos del cielo, deberá ser observador y rápido, teniendo que moverse de izquierda a derecha repetidas veces, ya que si al personaje le dices que vaya para un lado, pinchando una flecha, irá hasta ese lado hasta el infinito y mas.

NIVELES DEL JUEGO

SIN NIVELES, EL JUEGO CAMBIARÁ A MEDIDA DE LA RECOLECCIÓN DE PUNTOS.

HUB

Información relevante sobre los juegos

AUN NO HAY HUB

REGLAS GENERALES EN LOS ENEMIGOS

Spawn:

Las bombas spawnean cada 4 segundos, pero cada 5 puntos se aumentará el 0.5 s.

Derrota:

Si tocas una bomba mueres.

Si llegas a muchos puntos, digamos 30 acabarás lleno de bombas.

NO ENEMIGOS ESPECÍFICOS

NO JEFES

NON-PLAYER CHARACTERS

SIN NPCs

SIN COLECCIONABLES

SIN ESCENAS SECUNDARIAS

NI MÚSICA, NI EFECTOS DE SONIDO

PUNTOS Y PREOCUPACIONES

APÉNDICE

La única animación que hay es cuando cambias de lado, que el personaje se gira.

PRUEBAS:

Al crear planes de desarrollo y pruebas detallados, podrás gestionar de manera efectiva el desarrollo de tu juego y asegurarte de que cumple con los estándares de calidad y funcionalidad deseados.

Requisitos:

Debemos probar distintas características del juego:

- 1) Probar colisiones: considerar las colisiones del suelo y del personaje.
- 2) Botones de movimiento: Derecha a izquierda (teclas de flecha), y de salto (espacio).
- 3) Salto: que el jugador solo pueda saltar, únicamente cuando toque el suelo.
- 4) Generador de bombas y diamantes: debemos probar, que caen del cielo, y en un sitio aleatorio, además de su desaparición.
- 5) Límites: Probar paredes del límite del mapa.
- 6) Aumento del contador: cada vez que se toque un diamante, el contador de puntos de arriba a la izquierda debe aumentar de uno en uno.
- 7) Aparición de la página de *Game Over*: Cuando se choque con una bomba debe aparecer la página de inicio del juego.
- 8) Botón de reinicio: Al salir la página de reinicio, tocar el botón de reinicio de nivel, reiniciando el juego de nuevo.
- 9) Sistema de puntos: Aumentar el tiempo en segundos en cuanto a la destrucción de las bombas al caer al suelo y diamantes, además de disminuir su tiempo, en la generación de los mismos.
- 10) Botones del menú Inicial: Probar botón de *JUGAR* para iniciar el juego. Probar el botón *SALIR*, para salir de la aplicación.
- 11) Botón volver: botón de la página *Game Over*, de *VOLVER* al menú de inicio.

Recursos:

4 Scripts con cs:

- MovimientoPingui
- Bombas
- diamantes

MenulInicial

Desarrolla Casos de Pruebas:

1. Probar Colisiones:

El personaje debe chocar contra el suelo.

- Acción: Soltar al personaje en el aire y observar si colisiona con el suelo.

2. Botones de Movimiento:

El jugador presiona las teclas de flecha y la barra espaciadora para mover al personaje.

- Acción: Presionar las teclas de flecha para mover de izquierda o derecha y la barra espaciadora para saltar, en distintas combinaciones.
- Resultado esperado: El personaje se mueve hacia la derecha, la izquierda y saltar según las teclas presionadas.

3. Salto:

El jugador intenta saltar en diferentes situaciones.

- Acción: Presionar la barra espaciadora mientras el personaje está en el suelo y en el aire.
- Resultado esperado: El jugador solo puede hacer que el personaje salte cuando está en el suelo.

4. Generador de Bombas y Diamantes:

Bombas y diamantes caen del cielo aleatoriamente y desaparecen después de un tiempo.

- Acción: Observar la caída de bombas y diamantes en diferentes partes del mapa.
- Resultado esperado: Las bombas y diamantes caen aleatoriamente del cielo y desaparecen después de un cierto tiempo.

5. Límites:

El personaje intenta atravesar los límites del mapa.

- Acción: Mover al personaje hacia los límites del mapa, con las flechas del teclado.
- Resultado esperado: El personaje no debe poder atravesar los límites del mapa.

6. Aumento del Contador:

El jugador recoge diamantes.

- Acción: Hacer que el personaje recoja diamantes.
- Resultado esperado: El contador de puntos en la parte superior izquierda aumenta en uno, por cada diamante recogido.

7. Aparición de la Página de Game Over:

El personaje choca con una bomba.

- Acción: Hacer que el personaje colisione con una bomba.
- Resultado esperado: La página de Game Over aparece al chocar con la bomba, mostrando la opción de reiniciar el juego.

8. Botón de Reinicio:

El jugador selecciona el botón de reinicio en la página de Game Over.

- Acción: Hacer clic en el botón de reinicio.
- Resultado esperado: Al pulsar el botón, el juego debe reiniciarse y volver al estado inicial.

9. Sistema de Puntos:

El jugador recolecta diamantes y evade las bombas.

- Acción: Recolectar diamantes y evitar bombas durante un período de tiempo. Además de cronometrar el tiempo de destrucción y respawn.

- Resultado esperado: Cada vez que se recolectan 5 diamantes, el tiempo de destrucción de las bombas y diamantes debe ajustarse y aumentar, según las reglas del juego.

10. Botones del Menú Inicial:

El jugador interactúa con los botones del menú inicial.

- Acción: Hacer clic en los botones *JUGAR* y *SALIR*.
- Resultado esperado: El botón *JUGAR* lleva a iniciar el juego, y el botón *SALIR* cierra la aplicación.

Botón Volver:

El jugador selecciona el botón *VOLVER* en la página de Game Over.

- Acción: Hacer clic en el botón *VOLVER*.
- Resultado esperado: El juego debe volver al menú inicial después de seleccionar el botón *VOLVER* en la página de Game Over.

EVALUACIÓN Y MEJORAS DE LA DEMO

Puntos Fuertes:

- ★ **Premisa del juego clara y simple:** un pingüino que recolecta diamantes y debe evitar las bombas. Esto facilita la comprensión y la jugabilidad para los jugadores.
- ★ **Mecánicas Básicas:** Las mecánicas básicas del juego, como mover al pingüino, recolectar diamantes y evitar bombas, están implementadas y son funcionales.
- ★ **Interfaz de Usuario Intuitiva:** La pantalla de inicio y sus transiciones entre pantallas son claras y fáciles de entender para los jugadores. Los botones de inicio y salida son directos y cumplen su función. Además del botón de reinicio.
- ★ **Progresión en cuanto a Puntuación:** La progresión del juego funciona en función de la cantidad de puntos acumulados en el sistema de puntuación, añadiendo un elemento de desafío y motivación para los jugadores. Para que cada vez se esfuerzen más en conseguir un nuevo record.

Puntos Débiles:

- **Falta de Profundidad:** El juego carece de elementos que aumenten su profundidad y lo hagan más interesante a largo plazo. Esto es debido a su simplicidad, aunque con su ausencia de historia y de mecánicas adicionales, puede hacer que el juego resulte monótono en poco tiempo.
- **Limitación de Contenidos:** La falta de elementos adicionales como pantallas de carga, música, efectos de sonido y animaciones adicionales puede afectar la experiencia general del jugador y hacer que el juego parezca incompleto o poco pulido. Pero es un prototipo, así que tampoco es para tanto.

- **Escasa Variedad de Objetos:** Solo hay dos tipos de objetos: diamantes y bombas. Si añadieramos más objetos podría mejorar la diversidad y la complejidad del juego.

Áreas Mejorables y Oportunidades:

- **Desarrollo de la Historia:** Explorar y desarrollar la historia del pingüino podría agregar profundidad emocional al juego y brindar una experiencia más envolvente para los jugadores. Se podría hacer una cinematografía cortita, aunque en un juego de este calibre no creo que sirva de mucho. Aun así demostraría al jugador, que su desarrollador le tendría cariño a su juego.
- **Ampliación de Contenidos:** Agregar más elementos como pantallas de carga, música , efectos de sonido y animaciones mejoraría la presentación e inmersión del juego en gran medida.
- **Introducción de Nuevas Mecánicas:** Incorporar nuevas mecánicas de juego, como obstáculos móviles, obstáculos más grandes o más pequeños y su presentación, obstáculos grandes que se separan en mas pequeños al llegar al suelo, obstáculos que rebotan, vidas extras, inmunidad, dobles puntos, todo esto para aumentar la variedad y el desafío del juego.
- **Personalización del Personaje:** Permitir la personalización del pingüino con diferentes prendas u objetos adicionales. O la introducción de diferentes personajes jugables, esto podría agregar un elemento de diversión adicional y atractivo para los jugadores.
- **Añadir records:** Cuando el jugador muera y aparezca el menú de *Game Over*, debería aparecer su puntuación. Además de ver un apartado en el menú de inicio o una mención del récord de puntuación más alta conseguida.
- **Menú de Opciones:** Añadir en el menú inicial del juego una página de opciones, donde podrás encontrar:
 - **Opciones de Audio**, para ajustar el volumen de la música de fondo, de los efectos de sonido y opción para habilitar/deshabilitar la música y efectos de sonido.
 - **Opciones de Gráficos**, para configurar la calidad gráfica, ya sea baja, media o alta. Para activar o desactivar efectos visuales, como sombras, partículas, etc.
 - **Configuración de Controles**, para poder reasignar teclas y botones para adaptarse a las preferencias del jugador. Además de una opción para cambiar la sensibilidad del mouse.
 - Selección del **idioma** del juego.
 - **Ajustes de Jugabilidad**, de la dificultad del juego, ya sea fácil, normal o difícil.
 - **Información del Juego**, para que muestre la información del juego, incluidos desarrolladores, artistas, músicos, etc. Ademas de enlaces a sitios web o redes sociales relacionadas con el juego para obtener más información o soporte.

Rendimiento y Evaluación:

Pruebas de Jugabilidad Exitosas:

El juego funciona bien y es dinámico. Es funcional, está bien ya que te genera un reto interno. Es un juego algo complicado, aunque parezca fácil es complicado calcular la caída de las bombas, además de que cada vez son más. Lo mismo con los diamantes, cada vez tardan menos en desaparecer, pero

con el aumento considerable de bombas esto no es muy bueno que digamos. terminando el juego si o si.

Respuesta de los jugadores: Los jugadores probablemente responderían según sus preferencias individuales y experiencia de juego, de manera variada a las mecánicas, controles y dinámica del juego.

Mecánicas: Algunos jugadores podrían encontrar las mecánicas simples y fáciles de entender, permitiéndoles sumergirse rápidamente en la experiencia del juego.

Controles: La mayoría de los jugadores podrían encontrar los controles intuitivos, permitiéndoles mover al pingüino con facilidad y precisión. Aunque otros preferirían opciones de personalización de controles para asignar diferentes teclas o controles, dependiendo de sus preferencias individuales de juego. Esto si añadimos la opción de Opciones.

Dinámica del juego: Los jugadores podrán disfrutar de la dinámica rápida y frenética del juego, mientras intentan recolectar diamantes y esquivar bombas en un entorno cada vez más desafiante. Aunque podría parecer repetitivo después de un tiempo. aunque con mejoras constantes podría mejorar.

Satisfacción del Jugador: La retroalimentación de los jugadores podría mejorarse con comentarios positivos sobre la jugabilidad intuitiva y entretenida, manteniendo a los jugadores comprometidos a medida que avanzaban en el juego. Los gráficos y el diseño visual serán bien recibidos para la comunidad específica en 2d, por su estilo colorido, que mejora la atmósfera del juego.

Comparación con Otros Juegos: Al comparar el rendimiento de "RUSHED PENGUIN" con otros juegos similares en el mercado, se pueden identificar varias áreas donde nuestro juego se destaca:

- La simplicidad y la accesibilidad de su mecánica de juego le hacen atractivo, diferenciándolo de juegos más complejos.
- Su combinación de elementos de recolección de puntos y desafíos de esquivar obstáculos ofrece una experiencia única que se destaca entre otros juegos casuales disponibles.
- La falta de niveles y la progresión basada en la puntuación también brindan una experiencia de juego dinámica y en constante evolución, para traer desafíos rápidos y gratificantes.

Plan de Mejora Continua: A pesar de ser un buen juego, aún es un prototipo y debemos hacer mejoras continuas del juego con actualizaciones periódicas que agreguen contenido nuevo y características emocionantes, además de arreglar fallos de ser encontrados. Consideraremos la expansión de la historia del juego para agregar profundidad a la experiencia del jugador, así como la incorporación de efectos de sonido envolventes para mejorar la inmersión. Estamos explorando opciones para agregar elementos de progresión del jugador, como desafíos y logros, que mantendrán a los jugadores comprometidos a largo plazo.

Nuestro objetivo es asegurarnos de que "RUSHED PENGUIN" siga siendo atractivo y relevante para los jugadores, brindándoles una experiencia de juego satisfactoria y emocionante en el futuro.

WEBGRAFÍA

Vandal, (24-feb-2023). Motores de juegos. [Consultado el 9 de febrero de 2024] Disponible en:
<https://vandal.elespanol.com/reportaje/los-11-mejores-motores-graficos-para-introducirse-en-el-desarrollo-de-videojuegos>

Youtube (15-oct-2021). BravePixelG, [Consultado el 16 de febrero de 2024]. Disponible en:
<https://www.youtube.com/watch?v=sJUBoFgO7Ng>

Youtube, (13-jul-2023). DigelusDev, [Consultado el 12 de febrero de 2024]. Disponible en:
Primera parte:

<https://www.youtube.com/watch?v=cNCu-MA52vc>

Segunda parte:

<https://www.youtube.com/watch?v=Vzmx-JAPrL0>

Youtube, (2-dic-2022). Game Maker's Toolkit, [Consultado el 12 de febrero de 2024]. Disponible en:
<https://www.youtube.com/watch?v=XtQMytORBmM>