Desarrollo de un entorno virtual de aprendizaje resaltando la importancia de las energías renovables: energía eólica.

Autor:

Victor Manuel Herrera Martinez

Asesor:
Profesor: Diego Rivera

Universidad Pedagógica Nacional
Departamento de tecnología
2020

Resumen

En este artículo se da a conocer el proceso de desarrollo de un Entorno Virtual de Aprendizaje enfocado en la divulgación de los beneficios de la energía eólica.

Se desarrolla un videojuego dinámico, interactivo y realista, desde el motor gráfico Unity 2018.2.12f1 3D. Para la evaluación de su usabilidad se toma en cuenta las pautas de la norma ISO 25000.

Este ambiente virtual se desarrolla en aras de dar una experiencia combinada con varios usuarios en interconexión, para ello se usó Photon Unity como intermediario entre los usuarios y la plataforma (el proceso e implementación se describe en el anexo).

Abstract

In this article, the process of developing a Virtual Learning Environment focused on spreading the benefits of wind power will be made known.

A dynamic, interactive and realistic videogame is developed from the graphics engine Unity 2018.2.12f1 3D. For the evaluation of its usability the guidelines of ISO 25000 are considered.

This virtual environment is developed in order to provide a combined experience with many users in interconnection, for which Photon Unity was used as an intermediary between the users and the platform (the process and implementation is described in the annex).

Introducción

Pierre Levy postula en su libro la *inteligencia colectiva* que el conocimiento absoluto es imposible, que el desarrollo de la humanidad se dio gracias a la distribución y especialización del conocimiento en cada persona de la colectividad, y que esta inteligencia se potenciaba a medida que la comunicación mejoraba, así que, con la llegada de las nuevas tecnologías, se puede decir que la *inteligencia colectiva* está en el punto más alto de su desarrollo.

Estas nuevas tecnologías se abren camino en medio de la educación, se busca continuamente nuevos espacios en donde el aprendizaje tenga un papel protagónico, es en estos espacios donde nacen estrategias como el *electronic learning (e-learning).* Estrategias que surgen con la masificación de las nuevas tecnologías (TIC)[1]. Esta Cibercultura[2] que permea y muta la cultura actual nos obliga a generar mediaciones[3] de carácter tecnológico y en este caso virtual para generar un entorno de interés, agradable y a la par con las TICs que se están desarrollando.

El concepto e-learning nace de la mano con la internet[4] en 1996, a partir de esta fecha ha venido evolucionando con nuevas herramientas que se han venido incluyendo (chat, video, llamada de voz). Pero, a pesar de la inclusión de estas herramientas el e-learnig se queda corto ante las infinitas posibilidades que nos abre las nuevas tecnologías.

Este avance tecnológico, genera un aumento en la exigencia de los estudiantes que a su vez hace que sea imperativo el cambio y la actualización de nuestros medios para

---

[1] Tecnologías de la información y comunicación
[2] Pierre Levy: Implicaciones culturales de las tecnologías digitales de comunicación e información.
[3] Vigotsky, proceso de enseñanza-aprendizaje.
[4] El concepto en si nace en 1924 cuando Sídney L. Presley inventa la primera máquina que suministraba preguntas de múltiple respuesta.

enseñar. El libro innovación tecnológica[5] hace referencia como aquellas organizaciones que no han avanzado junto a los cambios tecnológicos sufren una reducción en el desenvolvimiento de sus capacidades reflejándose en falta de competitividad (Revilla, 2001).

Así que evidencié en los videojuegos las herramientas necesarias para realizar ese cambio. Los videojuegos en la actualidad han evolucionado mucho en muy poco tiempo, al volverse una industria multimillonaria se ha convertido en una de las opciones más viables y más desarrolladas en la generación de ambientes virtuales hiperrealistas, en la web podemos encontrar distintos motores gráficos para el uso e implementación de videojuegos en este caso de carácter pedagógico.

"Los videojuegos arte, ciencia y tecnología; involucran una plétora de habilidades y conocimientos en distintas disciplinas, desde ciencias formales hasta ciencias sociales que van más allá del típico proyecto de software e implican al mismo tiempo la creatividad y la imaginación." (Morales, Nava, Fernández & Rey, 2010)

[5] Innovación Tecnológica: ideas básicas. Fundación Cotec

Diseño del entorno virtual

Preproducción

La idea del desarrollo de un videojuego para la divulgación de los beneficios de la implementación de la energía eólica, surge durante el curso de las materias de la Maestría en Tecnologías de la Información Aplicadas a la Educación, en ambas materias encontré las herramientas necesarias para poner en marcha el proyecto.

Pautas

Seleccioné así una serie de cánones que debería tener el proyecto:

- El objetivo principal del entorno virtual era la difusión de los beneficios y usos de las energías renovables específicamente de la energía eólica.
- Debería ser desarrollado sobre una plataforma grafica que permitiera la interacción con otros usuarios.[6]
- El entorno virtual debería ser lo más realista posible, pero con requerimientos de hardware básicos que permitieran funcionar el entorno virtual con bajos requerimientos funcionales.

Selección de plataforma de desarrollo

[6] Juego multijugador en línea: Es un videojuego donde pueden interactuar varios jugadores en el mismo entorno virtual en línea.

La plataforma grafica se seleccionó en base a los ítems anteriores. Las dos principales opciones por su accesibilidad y facilidad de manejo fueron: Sansar y Unity.

Se hizo una breve exploración de ambas plataformas; Sansar al ser una plataforma social en VR[7] cumplía el segundo ítem a cabalidad pero esa misma condición hacia que no en todos los computadores se desempeñara de manera fluida por los requerimientos funcionales que demanda.

A diferencia de Samsar en donde todos los objetos, funciones y hasta la interconexión en red traía ya prefabricada Unity daba la posibilidad de realizar todo este desarrollo desde ceros. Se realizó una exploración de funcionalidad y requerimientos y decidí dar inicio al proyecto usando Unity como el motor gráfico de desarrollo del entorno virtual.

Producción

Se inicia el desarrollo del proyecto con la generación del ambiente (mapa). El proceso de diseño se inicia con bosquejos básicos para tener una breve noción de la organización que debería llevar.[8]

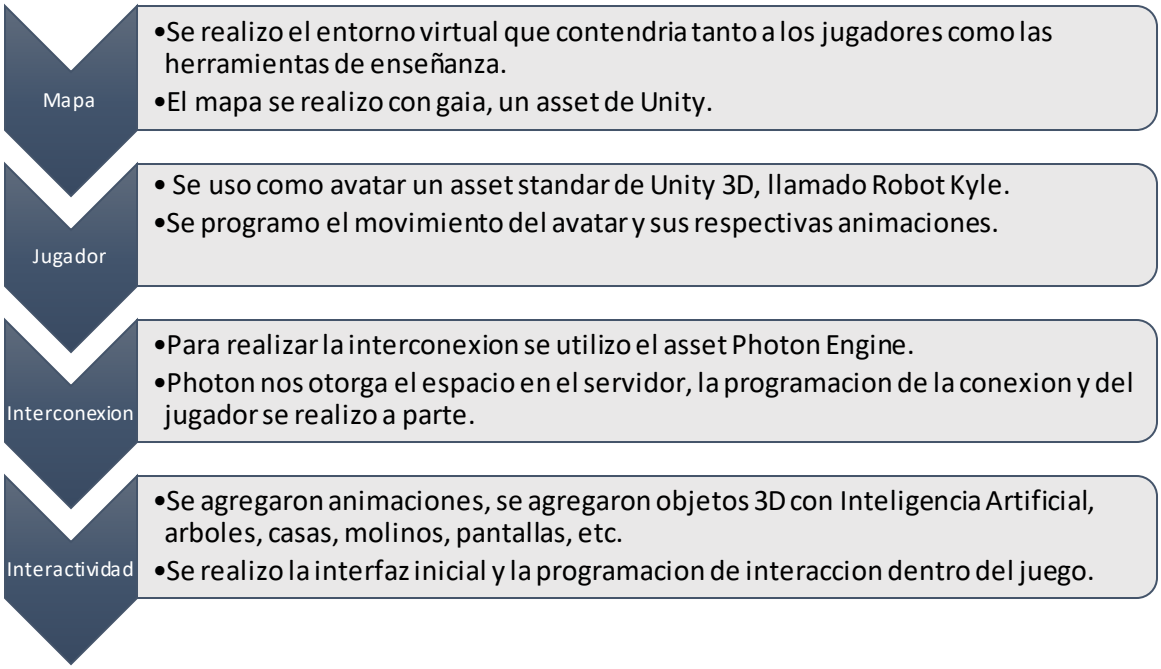
Se exploraron distintos assets[9] teniendo en cuenta los siguientes factores:

- Herramientas básicas
- Desarrollo del Mapa

[7] Plataforma de Realidad Virtual
[8] En el anexo titulado: Proceso de desarrollo videojuego energía eólica, podremos encontrar el paso a paso detallado del proceso de desarrollo.
[9] En Unity los assets son ítems, herramientas o paquetes de ítems que pueden venir de archivos creados desde fuera de Unity, podrían ser: modelos 3D, audios, paquetes de programación.

- Conexión en red
- Player (jugador)
- Entorno e interacción.

Una vez seleccionado cada assets inicie el desarrollo que llevo la siguiente secuencia:



Cabe resaltar que la selección de assets y el proceso en detalle se describe en el documento anexo.

Guía de software

Recomendaciones

La plataforma digital de videojuegos Steam[10] nos sugiere las siguientes precauciones básicas al momento de hacer uso de plataformas digitales y videojuegos:

[10] Steam es una plataforma de distribución digital, gestión digital de derechos, comunicaciones y servicios multijugador.

- No se siente demasiado cerca de la pantalla y colóquese a la distancia máxima que le permita el cable del mando.
- Juegue con videojuegos preferiblemente en una pantalla pequeña.
- No juegue si se encuentra cansado.
- Asegúrese de que la habitación en la que está jugando esté bien iluminada.
- Descanse durante 10-15 minutos por cada hora que pase jugando con un videojuego.

Requerimientos

OS: Windows® 7 SP1+, macOS® 10.11+, Ubuntu 12.04+, SteamOS+

Tarjeta de video: Con capacidad para DX10 (shader modelo 4.0).

CPU: compatible con el conjunto de instrucciones SSE2.

Espacio disponible disco duro: 1.0 GB

Instalación

Se recomienda cerrar cualquier programa en ejecución para que no interfiera al momento de extraer el archivo. El archivo del proyecto se pasará en extensión .zip por el tamaño del mismo. Así que el primer paso sería extraer el archivo (clic derecho->extraer aquí).

El último paso es darle doble clic al archivo de extensión .exe para ejecutarlo.

Menú principal

Una vez ejecutado el archivo entraremos a la pantalla principal:



Ilustración 1. Menú principal.

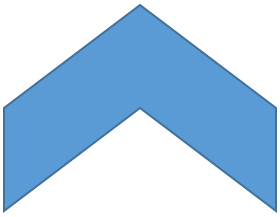
Play: Botón que dará inicio al juego.
Options: Aquí podremos modificar el volumen del sonido.
Quit: Botón para cerrar el juego.



Minimapa: La función principal del mini mapa es la de guiar al usuario alrededor de los puntos de interés, además de mostrar de manera ortogonal una vista general del mapa.



Indicador minimapa: Dentro del entorno virtual existen múltiples indicadores de puntos de interés siendo este el más notable pues aparece directamente sobre el mini mapa



.

Orientación: Esta imagen estará indicando continuamente la orientación de nuestro jugador sobre el mapa.



Indicador panorámico: Este indicador se da a manera de partícula que se eleva se manera constante lo que nos permite visualizarlo desde una buena distante en la cámara principal. Indicador de puntos de interés.



Guía: Este drone se encontrará a lo largo de todo el recorrido, pues es el encargado de brindarnos información y guía en cada estación de información.

Controles

Los controles de mando del avatar es un trabajo combinado entre teclado y mouse. Mientras el teclado controla el movimiento y la activación de ciertos elementos el mouse nos otorga visión y orientación del movimiento.



Las teclas *W* y *S* son usadas para el desplazamiento *adelante* y *atrás* respectivamente. Y las teclas *Q* y *E* desplazamiento izquierda y derecha. Las teclas *A* y *D* se utilizan para rotar al avatar sobre su propio eje.



En cuanto al mouse sosteniendo el **clic derecho** y girando al tiempo rotamos el cuerpo del avatar y a la vez su dirección de desplazamiento.

Sosteniendo el **clic izquierdo** y girando rotamos la cámara principal mas no su dirección de desplazamiento.

Con el **botón scroll**, desplazándolo hacia atrás aleja la cámara transformándola de una cámara de primera a tercera persona y desplazándolo hacia al frente tendría el efecto contario.

Dinámica del juego

Al comienzo del juego te encuentras en una isla con un drone ayudante, el té guiara esperándote en cada estación de información. Habrá una breve explicación de los objetos de interacción y como activarlos. Durante todo el recorrido habrán estaciones con información acerca de la energía eólica, su origen, beneficios y su uso en la actualidad

Plantilla de diseño

En la siguiente plantilla *Huddle*[11] se hará un recuento organizado de la información recopilada hasta el momento:

CAMPO	DESCRIPCIÓN
CONCEPTO	
Titulo	WindPower
Diseñadores	Victor Manuel Herrera Martinez
Genero	Simulación, educativo.
Plataforma	Unity 2018.2.12f1 3D
Versión	6.0

Sinopsis y mutabilidad	El ambiente se sitúa en una isla santuario de la energía eólica. Es una isla santuario porque es el último vestigio de humanidad que queda en la tierra, el resto del mundo sucumbió ante la contaminación. El primer encuentro será con *Andru* un androide quien será el guía por toda la isla e ira describiendo en cada estación los objetos y dará preámbulo a cada estación de información.
Categoría	De entre los juegos con protagonismo de la energía eólica WindPower es el único virtual y en 3D así que se puede decir que el antecedente ante este ambiente virtual es muy pobre.
Controles	En la sección de *controles* se explica el desplazamiento del avatar sobre el mapa. El ambiente virtual aparte de contar con indicadores en ambas cámaras del juego, cuenta también con tele trasportadores que

[11] Esta plantilla es el resultado de la recopilación y análisis de distintos procesos de desarrollo de software adaptados al desarrollo de videojuegos. Realizada en el 2010 por Morales, Nava, Fernández & Rey.

	hacen mucho más sencillo el desplazamiento entre punto y punto de interés.
Tecnología	Computadora personal con tarjeta de video con capacidad para DX10, OS:. Windows® 7 SP1+, macOS® 10.11+, Ubuntu 12.04+, SteamOS+
Publico	Este juego va dirigido a población adolescente y adulta. Por el tipo de información y la complejidad de la interacción.
Historial de versiones	Versión 1.0 En esta versión se realizó el mapa inicial. Versión 2.0 Se agregó el personaje (avatar) Versión 3.0 Programación de animaciones de movimiento del personaje principal. Versión 4.0 Programación de interconexión de red. Versión 5.0 Programación Inteligencia Artificial del entorno (aves, robots, sonidos, etc.) Versión 5.1 Se agregó los objetos representativos del juego (turbinas eólicas, molinos, etc.) Versión 5.2 Objetos de interacción y videos. Versión 5.3 Se programa la activación de los objetos de interés Versión 5.4 Se organiza el recorrido. Versión 5.5 Se programa el reproductor de sonido y video. Versión 5.6 Se anima los objetos de interés. Versión 6.0 Se agrega la escena del Menú principal

MECÁNICA
El jugador podrá navegar libremente sobre el mapa del juego, interactuar con los objetos y criaturas del mapa. Existe una ruta sugerida que se puede realizar con ayuda de tele trasportadores para ayudarnos a desplazar de un lugar a otro, la ruta sugerida va organizada como línea de tiempo tratando la energía eólica como eje central.
VISIÓN GENERAL

Para nadie es un secreto que los problemas de contaminación que está viviendo el mundo es por el consumo desenfrenado que lleva actualmente la humanidad, el sistema económico se encarga de poner en la cima de la pirámide económica las ganancias y deja en segundo plano la generación excesiva de desechos. Según Francisco Vargas Subdirector General de Sanidad Ambiental de Madrid España en los países industrializados una incidencia total de las enfermedades puede atribuirse a la contaminación y 1 de cada 3 muertes se les atribuye a exposiciones ambientales esto en grupo de edad de 0-19, una cifra alarmante al considerarse un promedio de edad saludable.

Estos problemas medio ambientales han adquirido tal magnitud que se hace imposible no evidenciar los efectos ocasionados, la sociedad ha cobrado conciencia y ha tratado de buscar alternativas y soluciones. Una de las soluciones es cambiar los agentes productores de desecho nocivos para salud, es el caso de los hidrocarburos principal productor de dióxido de carbono que está directamente relacionado con problemas respiratorios y con el calentamiento global. Es aquí cuando surgen las llamadas energías renovables, energía que se obtiene de fuentes naturales virtualmente inagotables, el viento, es una energía inagotable y ha sido aprovechada desde la antigüedad por la humanidad por medio de molinos de viento para triturar los granos de las cosechas o para impulsar los navíos a través del mar. Con el desarrollo de la tecnología actual podemos tomar esta energía y transformarla en electricidad y a su vez usarla para alimentar grandes ciudades. La idea del entorno virtual es mostrar la cantidad de inventos, tecnología, aerogeneradores que se han creado alrededor de esta problemática y que a pesar de ser una solución viable para mitigar los problemas causados por la contaminación son muy poco conocidos. En Latinoamérica somos unos de los países que menos le apuesta a la generación de energía por medio de la energía eólica según el *canalclima* quien presta servicios meteorológicos para diferentes sectores de la industria. Brasil ha invertido más de 347 millones de dólares en la generación de energía a través de parques eólicos seguido por México, Chile, Argentina y Costa rica. Colombia solo posee un parque eólico, el parque eólico Jepichari, que cuenta con 15 aeroturbinas, con una generación total de 19.5 MW de potencia a comparación de Brasil que genera 2.000MW, la diferencia denota una inversión muy pobre por parte de nuestro país en la generación de energía a través de fuentes renovables de energía.

La misiva principal del entorno virtual es divulgar y naturalizar el uso de los aerogeneradores como principales productores de energía en Colombia. Uno de los principales problemas en el país es el desconocimiento de fuentes de energía alternativas y no contaminantes y la falta de apropiación de esta información en el ámbito escolar siendo los medios de comunicación quienes toman la tarea de investigar, pero igualmente quedan cortos pues le dedican uno o dos artículos al

año acerca de los avances de la energía eólica en el continente y el mundo. Este olvido involuntario de la energía eólica genera que el estado no vea la energía eólica como una alternativa viable de producción de energía y termine apostándole a los hidrocarburos, represas e hidroeléctricas que han demostrado ser fuentes bastantes cuestionadas energía.

Paola Moreno estudiante de Gerencia Ambiental de la Universidad Libre nos demuestra en su estudio acerca de las ventajas y desventajas de la implementación de la energía eólica en Colombia que un solo un rotor de 92 m de ancho puede llegar a producir 1028kw y que a pesar de ser un país naturalmente rico en energía eólica cuenta con muy poco incentivo por parte del estado y nos invita a generar un cambio drástico de mentalidad que permita ver la verdadera riqueza de las fuentes de energía renovable no solo en el ámbito económico sino también en el de la salud y el medio ambiente. WindPower es un entorno virtual que con una expectativa ambiciosa quiere empezar a generar ese cambio de mentalidad en una parte de la población colombiana y quiere que los estudiantes se apropien de la energía eólica y apuesten por ella como una opción viable de producción de energía.

Evaluación de usabilidad

La evaluación de usabilidad se da bajo los estándar de la norma ISO/IEC 25000, conocida como SQuaRE[12] . Estos entandares son una serie de recopilación de normas que tiene como objetivo la delimitación de la calidad de un producto software.

El conjunto de normas ISO/IEC 25000 es el resultado de la evolución de dos familias ISO/IEC 9126 y ISO/IEC 14598. La primera encargada de describir las características de un modelo de calidad de un producto de software y la segunda encargada del proceso de evaluación de productos de software. Ambas se fusionaron y generaron la familia de normas ISO/IEC 25000:

[12]System and Software Quality Requirements and Evaluation



Ilustración 2. División de normas ISO/IEC 25000.

Las características principales a evaluar del entorno virtual son:

- Adaptabilidad: El usuario podrá deducir que tan conveniente es el producto para sus necesidades.
- Aprendizaje: Que tan eficaz es el producto al comunicar y promover la interiorización de la información presentada.
- Usabilidad: La facilidad con que el entorno puede ser entendido, manejado y usado.
- Errores: Generación de errores al usarlo.
- Accesibilidad: Facilidad para acceder al producto.
- Estética: Se evaluará que tan llamativo es el entorno virtual, si cumple con requisitos de estética básicos o sobrepasa las expectativas generando una interacción satisfactoria.

Plan de evaluación

Durante todo el proceso de diseño y desarrollo el videojuego estuvo monitoreado, calificado y valorado por Diego Rivera asesor y tutor del proyecto. Continuamente se hizo modificaciones y mejoras gracias a las críticas de carácter constructivo por parte del asesor, recomendaciones que permitieron al proyecto satisfacer las características básicas para un buen desempeño. Para la evaluación del producto se desarrolló como herramienta una encuesta basándose en las características anteriormente nombradas, esta encuesta se realizó de forma online en el colegio *Gimnasio Campestre San Francisco De Sales* con seis estudiantes del grado 9ª con intermediación del profesor del Profesor del área de Tecnología Jerson Escobar, quienes interactuaron 20 minutos con el entorno virtual. A continuación, se muestra el plan de evaluación y posteriormente la herramienta:

Tabla 1

Plan de evaluación

Ítems a evaluar			
Ítem	**Criterio**	**Descripción**	**Categoría**
Adaptación	Utilidad	Beneficioso al momento de usarlo	Ia
	Pertinencia	Es pertinente con las problemáticas sociales actuales	Ib
Aprendizaje	Facilidad	Se hace más sencillo el aprendizaje	IIa
	Información	Información eficaz, suficiente y argumentada.	IIb
	Conciso	Cumple sus funciones	IIc
Usabilidad	Navegación	Facilidad al desplazarse por el entorno	IIIa

	Rendimiento	Buen rendimiento operativo	IIIb
	Interfaz	Interfaz amigable con el usuario	IIIc
Errores	Funcionalidad	Presencia de errores al ejecutarlo	IV
Estética	Armonía	Presentación agradable y coherencia de diseño entre interfaz, entorno y texto.	V

Los Ítems de usabilidad fueron tomados de los parámetros de calidad de software establecido por la norma ISO/IEC 25000 subdividido en criterios se le otorga un indicador para discriminar dentro de la encuesta cada criterio y así asegurar que cada uno sea evaluado.

Herramienta de evaluación

Tabla 2

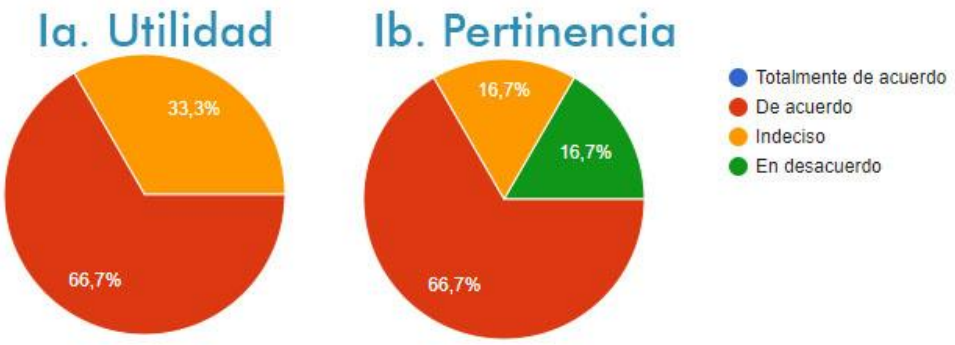
Encuesta

Categoría	Pregunta	Opción de respuesta
Ia	Respecto al tema de la energía eólica¿ Le pareció útil WindPower?	A. Totalmente de acuerdo. B. De acuerdo. C. Indeciso. D. En desacuerdo. E. Otro.
Ib	¿Es adecuado como aborda el tema el videojuego?	
IIa	¿La dificultad del tema le pareció la adecuada?	
IIb	¿La información plasmada en el juego le pareció pertinente?	
IIc	Teniendo en cuenta que el objetico del juego es divulgar los beneficios de la energía eólica ¿Se cumplió el objetivo con ud?	
IIIa	El desplazamiento por el entorno virtual se diseñó para que fuera sencillo y rápido ¿Según ud esas expectativas se cumplieron?	
IIIb	¿El juego tiene un buen rendimiento operativo al ejecutarlo?	
IIIc	¿El menú inicial y del juego son intuitivos al momento de hacer uso de ellos?	
IV	¿No se presentó ningún error durante el uso del videojuego?	
V	¿El diseño del entorno virtual es agradable y con buen nivel de detalle?	

Resultados

A continuación, se mostrará los resultados de la encuesta, herramienta usada para la medición de calidad del videojuego:

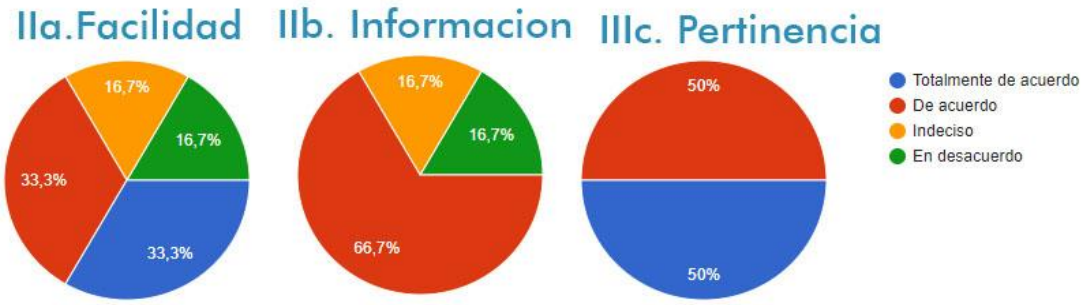
Adaptación



Ia. Utilidad: El 66,7% de los encuestados perciben al videojuego como útil al momento de dar a conocer la energía eólica.

Ib. Pertinencia: Y a la mayoría con un 66,7% perciben en el entorno virtual una adecuada manera de manejar y presentar el tema principal: la energía eólica.

Aprendizaje



IIa. Facilidad: A la mayoría de las personas encuestadas les pareció que el nivel de dificultad de la información presentada y en la forma en que se hizo fue la adecuada.

IIb. Información: En cuanto a la información presentada de la energía eólica el 66% estuvo de acuerdo en que fue la más acertada

IIIc. Pertinencia: La totalidad de los encuestados opinó que el entorno virtual cumple con el objetivo de dar a conocer la energía eólica y sus beneficios.

Usabilidad



IIIa. Navegación: Los encuestados llegaron a la conclusión que el diseño, la organización y las herramientas dispuestas en el escenario para la navegación del mismo fueron útiles y sirvieron haciendo del desplazamiento una tarea más sencilla.

IIIb: Rendimiento: En cuanto al rendimiento funcional del software todos llegaron a la conclusión de que era optimo y funcional.

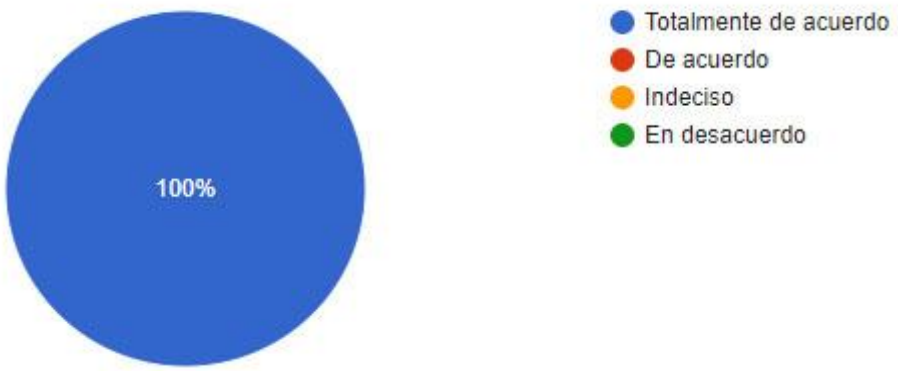
IIIc. Interfaz: En cuanto al menú inicial y los controles se notó un resultado poco satisfactorio, esto representa una posible falla en la disposición de del menú y de los controles del avatar.

Errores



IV. Funcionalidad: El 66% no tuvo problema alguno con la ejecución del juego, a algunos por problemas de hardware no les fluyo dinámicamente y tan solo el 16% tuvo problemas de conexión con el servidor durante el inicio del juego. Lo que denota que el videojuego cumplió en su mayoría con sus funciones operacionales.

Estética



V. Estética: En cuanto a los resultados de la apariencia, calidad y nivel de detalle de las gráficas, menú y los textos me atrevo a decir que fueron más que satisfactorios, en su totalidad los usuarios estuvieron más que de acuerdo con la calidad de la apariencia del juego.

Conclusiones

Los videojuegos han evolucionado rápidamente en la última era, eran considerados juguetes para niños y desarrollados por pocos programadores aficionados, pero actualmente las empresas editoriales creadas para el desarrollo de videojuegos comprometen distintas disciplinas al punto que aparte de los programadores inmiscuyen músicos, diseñadores, marketing e incluso abogados, y todo esto solo para publicar un videojuego

El amplio desarrollo tecnológico de los videojuegos que se debe en gran medida al éxito lucrativo que genera ha llevado que los motores gráficos para el desarrollo de estos proliferen, abriendo la posibilidad de ser aprovechados en este caso como medio para llegar al aprendizaje y más aun aprovechando el auge de las TIC y el acceso casi general que tiene la mayoría de la población

Los resultados anteriores evidencian una aceptación por parte de los adolescentes a interactuar y aprender en la virtualidad, la Cibercultura[13] ha facilitado y potenciado la

[13] Cibercultura: Es un informe escrito por Pierre Levy para el consejo de Europa donde habla acerca de las implicaciones culturales de las nuevas tecnologías de la información y comunicación en la sociedad.

aceptación de estos medios Virtuales para la enseñanza y más aún cuando estos se encuentran interconectados de manera colectiva. Se logró evidenciar cinco factores principales que deben tenerse en cuenta al momento de desarrollar un videojuego de carácter educativo: Estética, aprendizaje, funcionalidad, utilidad y adaptación. Todos tomados en cuenta al momento de desarrollar este proyecto enfocado a mitigar los efectos producidos por la contaminación de la no generación de energías por vías alternativas y renovables ya que como evidencia el estudio realizado por la Universidad Libre[14], en Colombia hay una muy pobre política estatal, muy baja factibilidad de proyectos de energía eólica, efectos generados en gran medida al desconocimiento del uso, beneficio y aprovechamiento territorial en Colombia de la energía eólica.

Quisiera agradecer la oportunidad que recibí para optar y cursar estas materias de maestría quienes fueron las que me dotaron de las herramientas necesarias para desarrollar este proyecto.

[14] Energía eólica: ventajas y desventajas de su utilización en Colombia. 2013

Referencias

Gutierrez (2001). *Innovación Tecnológica. Ideas Básicas*. Valladolid: Colección innovación práctica. Vol.2, cap1 La innovación. Fundación Cotec.

Moreno (2013). *Energía eólica: ventajas y desventajas de su utilización en Colombia.* Bogotá: monografía para optar el título de especialista en gerencia ambiental. Universidad Libre de Colombia.

Hernández (2018). *Diseño e Implementación de un prototipo de Entorno Virtual de Aprendizaje referente a la ubicación espacial de robots seriales holonómos: el problema cinemático directo e inverso*. Bogotá: Trabajo de grado Universidad Pedagógica Nacional.

Morales, Nava, Fernández & Rey (2010). *Procesos de desarrollo para videojuegos.* Juárez: Culcyt videojuegos, vol. 7, Universidad Autonoma de Ciudad de Juárez.

Duarte (2003). *Ambientes de aprendizaje. una aproximación conceptual*. Medellín: Estudios Pedagógicos, vol. 1 Nº 29 pag 97-113 Universidad de Antioquia.

Mendoza, Galvis (1999). *Ambientes virtuales de aprendizaje: una metodología para su creación*. Bogotá: Informática Educativa Unidades, vol. 12, No2, pag 295-317.

Cabrero (2006). *Bases pedagógicas del e-learnig.* Bogotá: Revista de Universidad y sociedad del conocimiento, vol. 3, No 1, Universidad de los Andes.

Agudelo, M (2009). *Importancia del diseño instruccional en ambientes virtuales de aprendizaje*. Medellín: Universidad de Antioquia.

Montoya (2004). *Schumpeter, innovación y determinismo tecnológico*. Pereira: Scientia et Technica, vol. 25, Universidad Tecnológica de Pereira.

Levy (1997) *Cibercultura informe al consejo de Europa*. Barcelona: Anthropos, vol.13.

Chavales (2001). *Implicaciones educativas de la teoría sociocultural de Vigotsky*. Montes de Oca: Revista Educación, vol. 25, núm. 2, Universidad de Costa Rica San Pedro.

Salgado (2013). *La gran máquina de enseñar*. Costa Rica: Research Gate.

Revilla (2001). *Innovación Tecnológica*. *Ideas Básicas*. Valladolid: Universidad de Valladolid.

Huanay, M (2017). *Tutorial Básico – Unity*. Lima: Universidad Nacional de Ingeniería.

Comité Técnico *ISO (2011). ISO / IEC 25010: 2011 Ingeniería de sistemas y software - Requisitos y evaluación de calidad de sistemas y software (SQuaRE) - Modelos de calidad de sistemas y software*. Ingeniería de software y sistemas.

Anexo

Proceso de desarrollo videojuego
Energía eólica

Victor Manuel Herrera Martinez

Profesor: Diego Rivera

Universidad Pedagógica Nacional
Departamento de tecnología

TABLA DE ILUSTRACIONES

Ilustración 1. Creando un nuevo proyecto en Unity.....32
Ilustración 2. **Descargamos e importamos** 32
Ilustración 3. Importando Assets estándar. 32
Ilustración 4. Importando assets standard.....33
Ilustración 5. Pasos para creación de objeto terreno.....33
Ilustración 6. Pasos para importar paquete de texturas.....34
Ilustración 7. Importando texturas 35
Ilustración 8 Seleccionando el pincel de texturas......36
Ilustración 9. Seleccionando las texturas necesarias. 36
Ilustración 10. Integración y posición de la cámara.....38
Ilustración 11. Ubicación de botones para relieve de terreno. 39
Ilustración 12. Depresiones sobre el terreno......40
Ilustración 13. Seleccionando nueva textura......40
Ilustración 14. Creando montañas, resultado. 41
Ilustración 15. Resultado de la nueva textura aplicada......41
Ilustración 16. Relieve aplicado en todo el mapa. 42
Ilustración 17. Textura aplicada sobre la punta de las montañas......42
Ilustración 18. Agregando grass en la lista de paint trees......43
Ilustración 19. Grass aplicadas......44
Ilustración 20. Resultado final vegetación 44
Ilustración 21. Ruta para crear el postprocesing profile. 45
Ilustración 22. Agregando postprocesing.....46
Ilustración 23. Personalizando postprocesing. 46
Ilustración 24. Postprocesing sobre el script......47
Ilustración 25. Ruta para crear efecto de sistema de partículas. 48
Ilustración 26. Características del sistema de partículas Niebla......48
Ilustración 27. Menú de importación de game kit en asset store.....51
Ilustración 28. Niveles de juego prefabricados. 52
Ilustración 29. Ellen como unico character.....52
Ilustración 30. Sprinter.....53
Ilustración 31 Grenadier 53
Ilustración 32. Chomper.....53
Ilustración 33. Ruta para crear un nuevo mapa......54
Ilustración 34. Así nos quedaría nuestro mapa inicial.....55
Ilustración 35 Planos del mapa unidos......55
Ilustración 36. Boceto mapa en game kit 56
Ilustración 37 Casa de aldea inventada con objetos de game kit. 57
Ilustración 39. Resultado de aldea. 57

Ilustración 38 Objetos interactivos. 57
Ilustración 40. Resultado de producción del bosque 59
Ilustración 41. Ruta para abrir el gaia manager. 48
Ilustración 42, Terreno "estampado". 1
Ilustración 43. Stamps de gaia para el mapa 64
Ilustración 44. Una vez arrastremos la imagen de stamp al terreno este automáticamente tomara la forma. 1
Ilustración 45. Estampado del relieve del mapa. 1
Ilustración 46. **Sabemos que ejecuto correctamente porque el spawn controller nos muestra el progreso del estampado de texturas.** 1
Ilustración 47. Resultado de la aplicación del player, water y wind. 1
Ilustración 48. Insertando el robot en el mapa 1
Ilustración 49.Ubicando cámara en el personaje, ejemplo de como debería quedar. . 1
Ilustración 50. Ventana de animación. 1
Ilustración 51. Creando un nuevo estado 1
Ilustración 52. Agregando animación HumanodeIdle. 1
Ilustración 53. Configurando transiciones de animación. 1
Ilustración 54. Creando un archivo c# script. 1
Ilustración 55. Transiciones y parámetros de los nuevos estados Running y WakingBack. 1
Ilustración 56.Control de velocidad de correr. 1
Ilustración 57. Sign in en Photon. 1
Ilustración 58. Menu para crear una nueva App 1
Ilustración 59. ID de la app 1
Ilustración 60. Vamos a dejar el Photon Type como Photon Realtime. 1
Ilustración 61- Pegamos la ID 75
Ilustración 62. Skin de estado de conexión. 1
Ilustración 63. Eliminando la cámara. 1
Ilustración 64. Prueba de conexión. 1

INTRODUCCIÓN

A continuación, haremos un recuento de nuestro proceso de producción del videojuego enfocado al proceso de enseñanza- aprendizaje de las energías limpias.

Queremos empezar diciendo que entramos con expectativas a este curso de Realidad virtual, pues siempre hemos estado inmersos en el ámbito de los videojuegos, pero, como simple consumidores, ahora se nos na la oportunidad de estar al otro lado de la pantalla como productores. Aunque siempre hemos querido producir videojuegos nuestros conocimientos acerca de la producción de estos eran muy pocos y al inicio del proyecto nos sentimos bastante limitados. Pero, como dice Pierre Levy en su libro *cibercultura,* la internet es un oceánico universo de información y nosotros somos los que navegan, nutren y pueblan este ciberespacio, así que fue cuestión de saber navegar y tomar lo necesario para aplicarlo en nuestro proyecto. ~~No fue sencillo, pero les mostraremos como lo hicimos.~~



1. Unity

Tenemos que admitir que antes de iniciar este proyecto nuestro conocimiento acerca de Unity era casi nulo, pero gracias a la gran cantidad de tutoriales que existen en la web y al facilidad que permite el software para manejarlo ayudaron a una mejor comprensión del programa y a darle un énfasis según nuestra área de desempeño que en nuestro caso

es el Diseño, nuestro proyecto estará enfocado a dar una experiencia de calidad hablando en t é rminos visuales.

El motor gr á fico usado para la realizaci ó n del videojuego es Unity creado por Unity Technologies. Esta plataforma de desarrollo puede compilar para diferentes tipos de plataforma desde Microsoft hasta Android. Uno de los beneficios del uso de esta plataforma de desarrollo para nuestro proyecto es que podemos encontrar gran cantidad de modelados en 3D para incluir en nuestro y si no lo hay nosotros como diseñadores podemos crearlo e incluirlo dentro del video juego.

Uno de los primeros pasos para la generaci ó n de nuestro proyecto fue la creaci ó n de un mapa inicial para lo que ser í a la plataforma principal del videojuego la hicimos con el manejo de herramientas b á sicas de Unity y agregamos una que otra textura, material y modelados para dar un aspecto realista al terreno a continuaci ó n mostraremos paso a paso el proceso de creaci ó n de este terreno.

La ventaja de usar Unity para la generaci ó n de nuestro proyecto es que existen cantidad de "assets" (item que puede ser usado para el videojuego) que ayudan a aquellas personas con un conocimiento b á sico en programaci ó n eso y ademas la compatibilidad que tiene con software de modelaci ó n 3D como Blender, 3ds max, maya y softimage.

No siendo m á s pasaremos a analizar cada Asset que consultamos y usamos y aunque no lo incluy é ramos en el producto final cabe decir que el tiempo dedicado a estos fue de gran ayuda en el aprendizaje a temas mas espec í ficos y es por eso que le dedicamos el espacio para traerlo a colaci ó n en el actual documento.

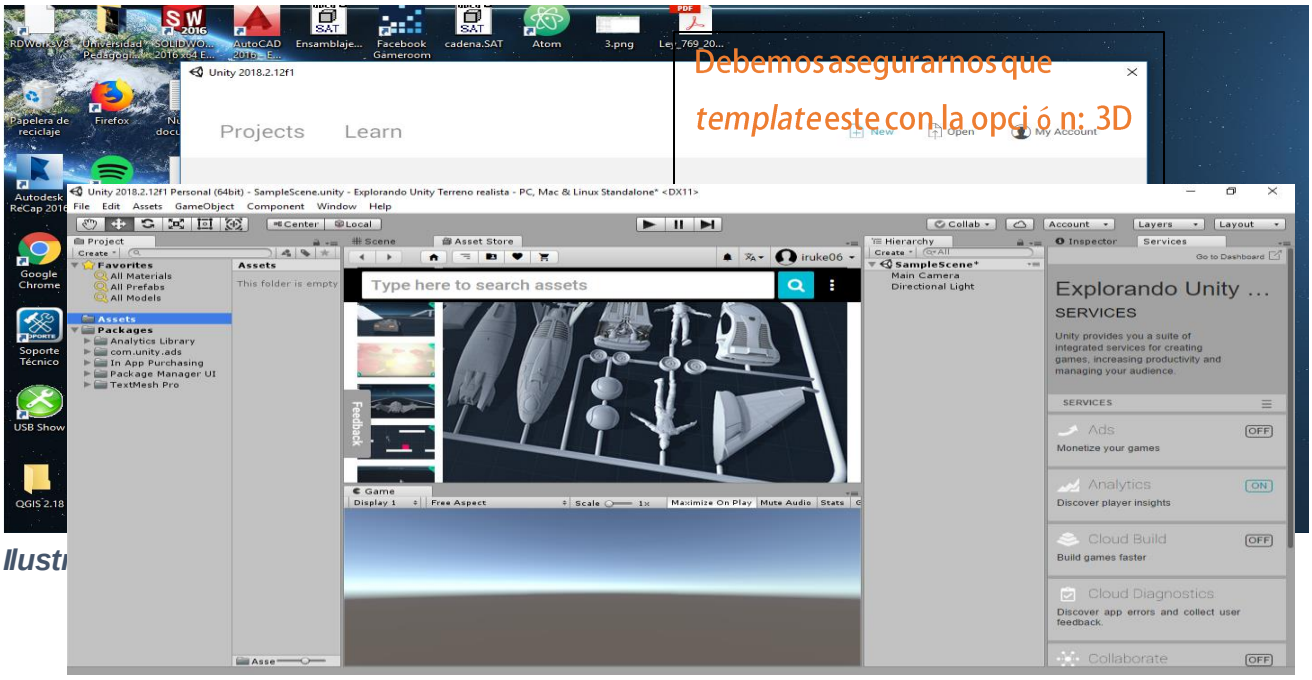
2. Mapa inicial

Esta fue la fase inicial de trabajo para nuestro proyecto del Taller de Realidad Extendida, en esta secci ó n nos dedicamos a la exploraci ó n de Unity y al aprendizaje de manejo de este software pues nuestros conocimientos respecto a este programa eran muy b á sicos por no decir nulos. As í pues nos planteamos como objetivo realizar un mapa desde cero obteniendo como resultado el proceso que van a distinguir a continuaci ó n:

i. Creando el documento

Como primer paso b á sico que mostraremos en esta fase y se dar á por entendido en las siguientes es el proceso de creaci ó n del nuevo proyecto que denominamos en este caso *Explorando Unity Terreno realista.*

ii. Importando Assets estandars



Ilustr

Ya el paso siguiente es importar algunos *Assets* est á ndar que trae Unity para poder hacer uso de ellos m á s adelante.

Ilustración 4. **Descargamos e importamos**

Ilustración 5. **Importando Assets estándar.**

Para Unity menores al 2018 los *assets estándar* ya se encuentran incluidos en el instalador para los que tengan Unity 2018 como es mi caso tendremos que descargarlos e Vale la pena recalcar que importarlos directamente desde assets store. No tiene costo alguno.

1.1.3. Creando un terreno

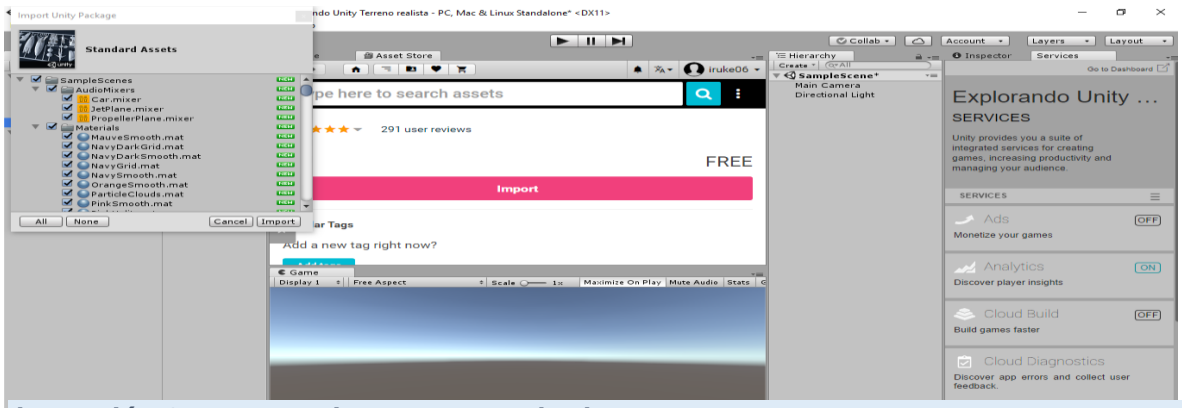


Ilustración 6. Importando *assets standard*

Ahora bien necesitamos un plano en el cual trabajar, Unity nos ofrece una opcion en el area de *Game object* pero en lugar de crear un plano como tal crearemos el objeto *terreno* o *terrain* que nos permitira editar y dar forma a este plano como a un trozo de un mapa real. Para crearlo seguiremos los pasos ilustrados en la siguiente imagen:

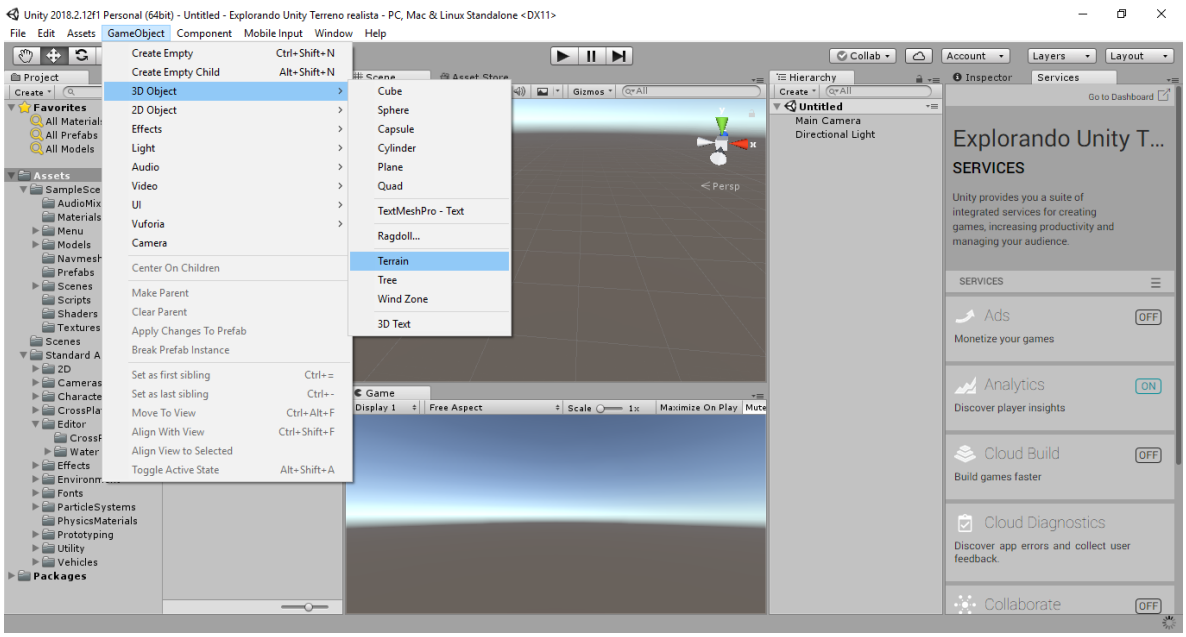


Ilustración 7.* Pasos para creación de objeto *terreno

iii. Importando y aplicando texturas al terreno

Como podemos observar el objeto *terreno* quedo como un plano com ú n y corriente, vamos entonces a importar unas texturas para aplicarle a este objeto 3D para ello seguiremos los paso que mostraremos a continuaci ó n:

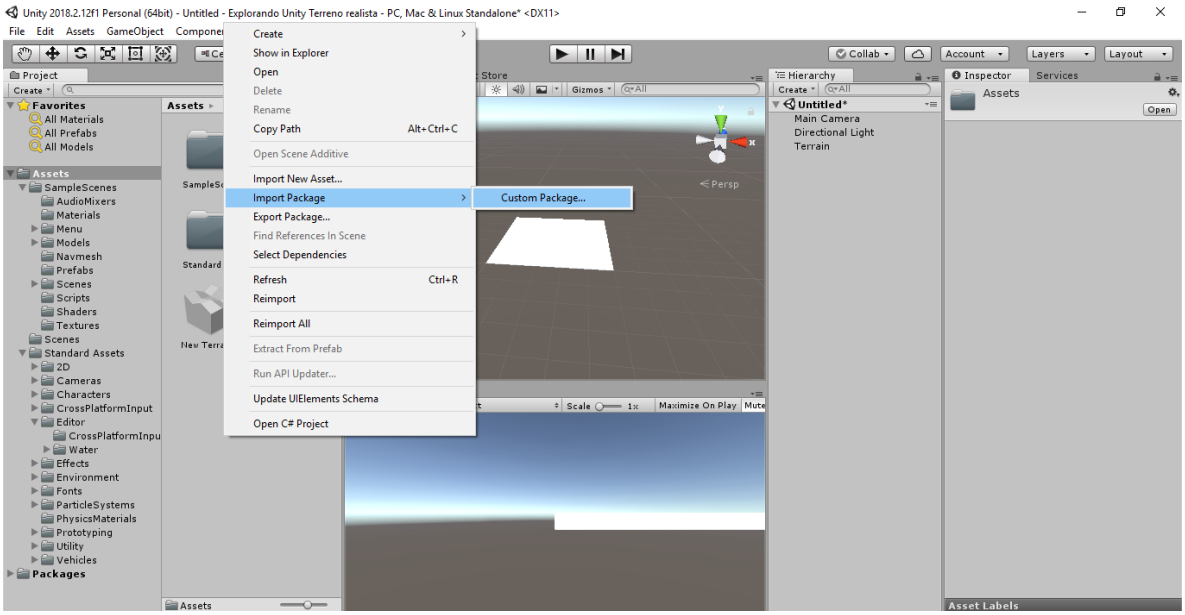


Ilustración 8. Pasos para importar paquete de texturas

Buscamos la direcci ó n en donde tenemos descargado el paquete de texturas que pueden conseguir en la web o que pueden fabricar.

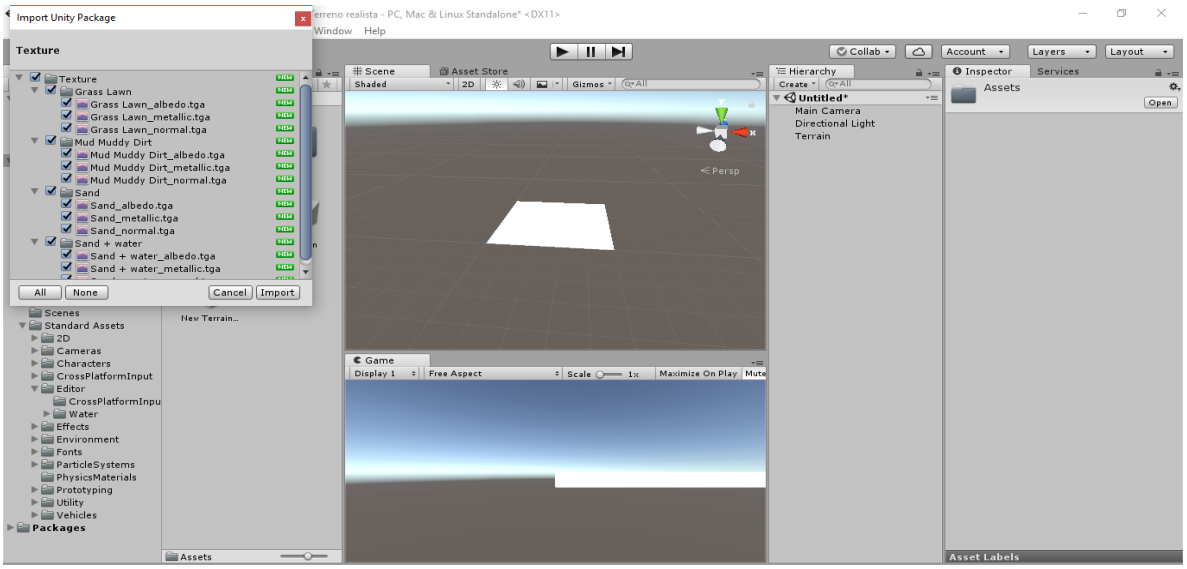


Ilustración 9. Importando texturas

Le damos click en importar y ya tenemos listo nuestro paquete de texturas para usar y pintar el objeto de *terreno.* Entonces en esta parte del proceso encontramos la diferencia entre el objeto *plano* o *plane* y elobjeto *terrain* , la diferencia radica es que con terrain podemos jugar con pinceles y aplicar por secciones diferente texturas para darle ese aspecto realista que buscamos en el momento de hacer un terreno en Unity.

Entonces vamos a seleccionar el objeto *terrain* en el men ú de *jerarqu í as* y en la secci ó n de *inspecci ó n* seleccionamos el pincel:

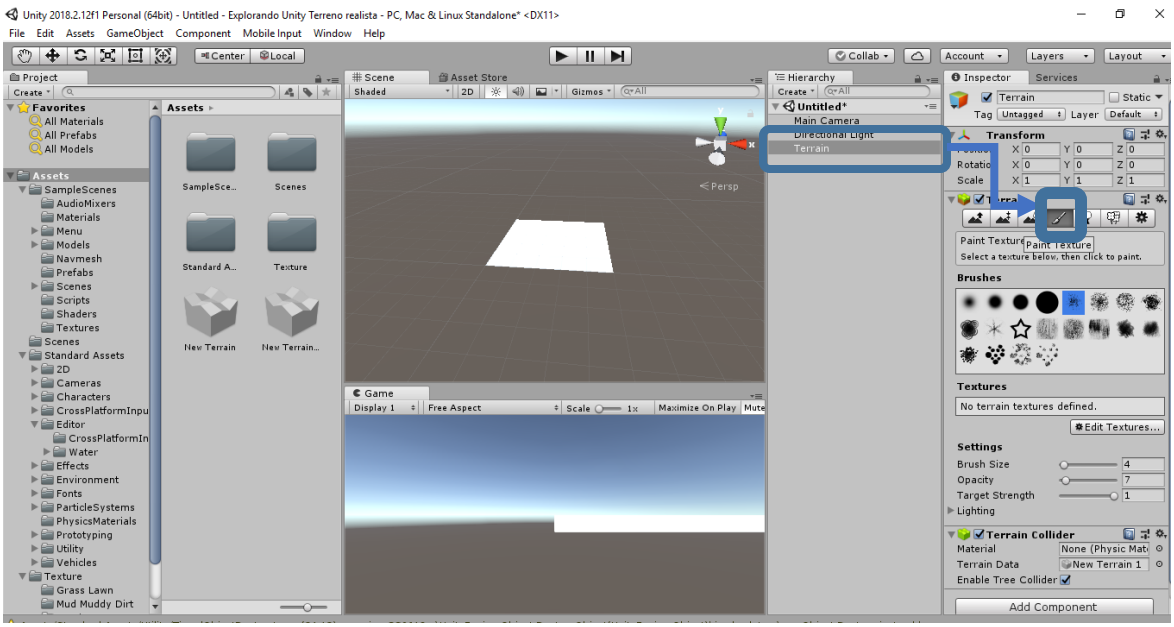
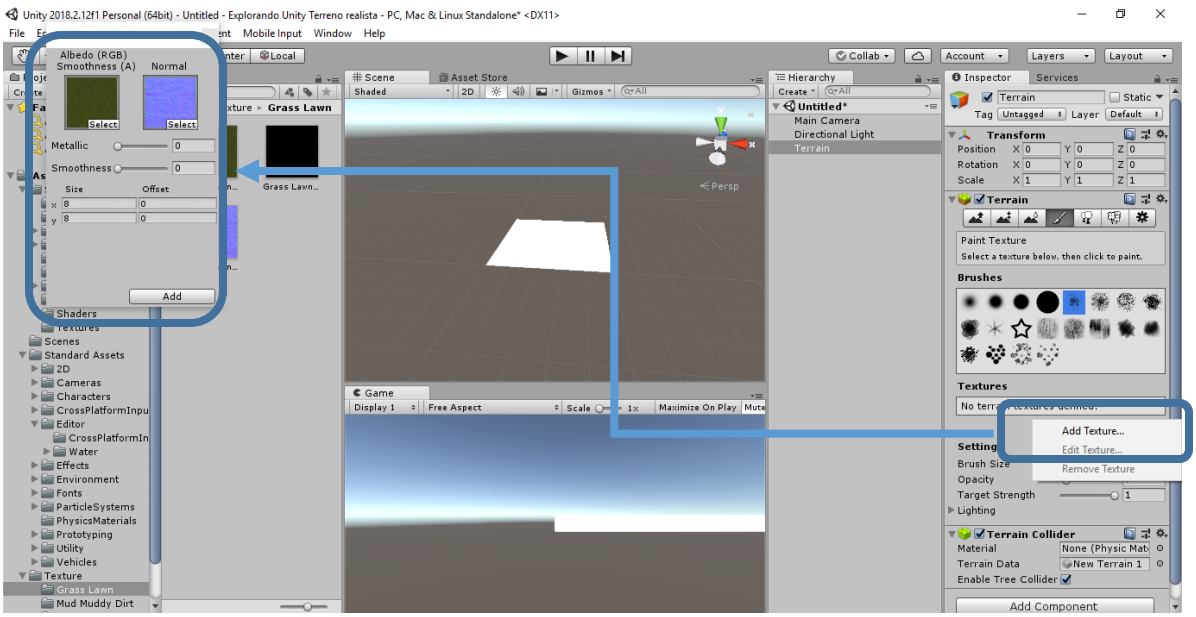


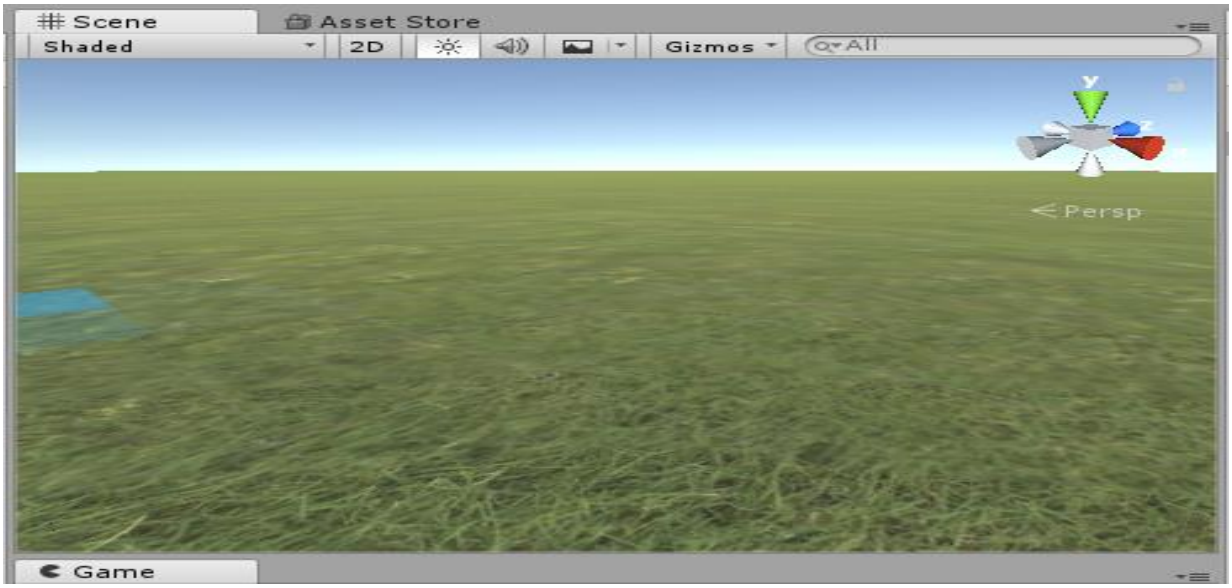
Ilustración 10 **Seleccionando el pincel de texturas.**

Ahora agregaremos las texturas al pincel que usaremos para eso en ese mismo menu debajo del pincel seleccionaremos Edit Textures -> add textures y en el menu agregaremos las dos texturas mostradas en la siguiente imagen:



Ilustraci ó n 11. Seleccionando las texturas necesarias.

Este ser í a el resultado:



iv. Agregando cámara

Ahora vamos a agregar la cámara del jugador, esta viene importada en los assets estándar que importamos en la parte inicial, lo que haremos es seguir la siguiente ruta de la ventana de *Assets:*

Standard Assets->Characters->first person->Prefabs->FPSControl.

Este último archive lo arrastramos hacia el mapa (la ventana de la escena en edición) y lo soltamos, nos aseguramos que esta capsula este por encima del terreno o al ejecutar el juego el jugador caería al vacío. Debería quedar de la forma siguiente:

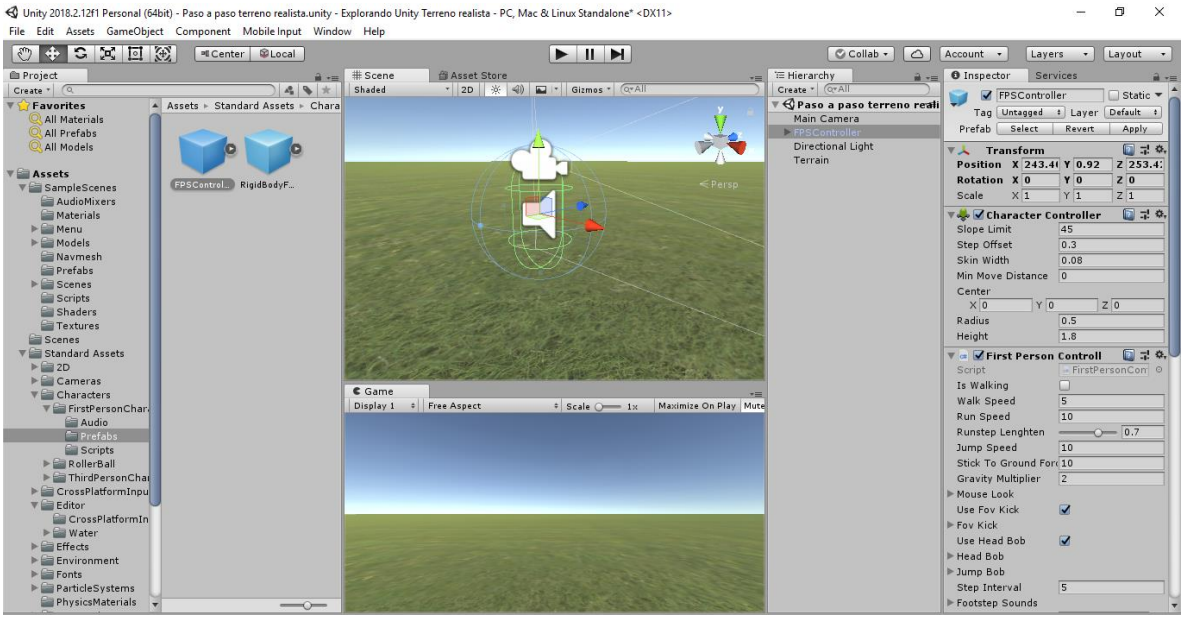
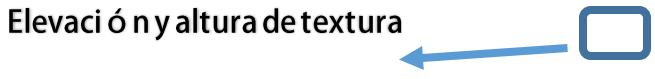


Ilustración 12. Integración y posición de la cámara

v. Editando el relieve del terreno.

Ahora nos dispondremos a editar el terreno vamos a aplicar relieves y depresiones para darle el aspecto realista que buscamos, para ello vamos a seleccionar al objeto *terrain* de la ventana de jerarquías y allí en el inspector vamos a seleccionar los botones de *altura de textura (paint heighte)* y empezamos a aplicar a todo nuestro terreno con un *brush Size y una Opcaity de 100%* y un *flatten de 10 y* con el tipo de pincel que muestra lassiguiente imagen:



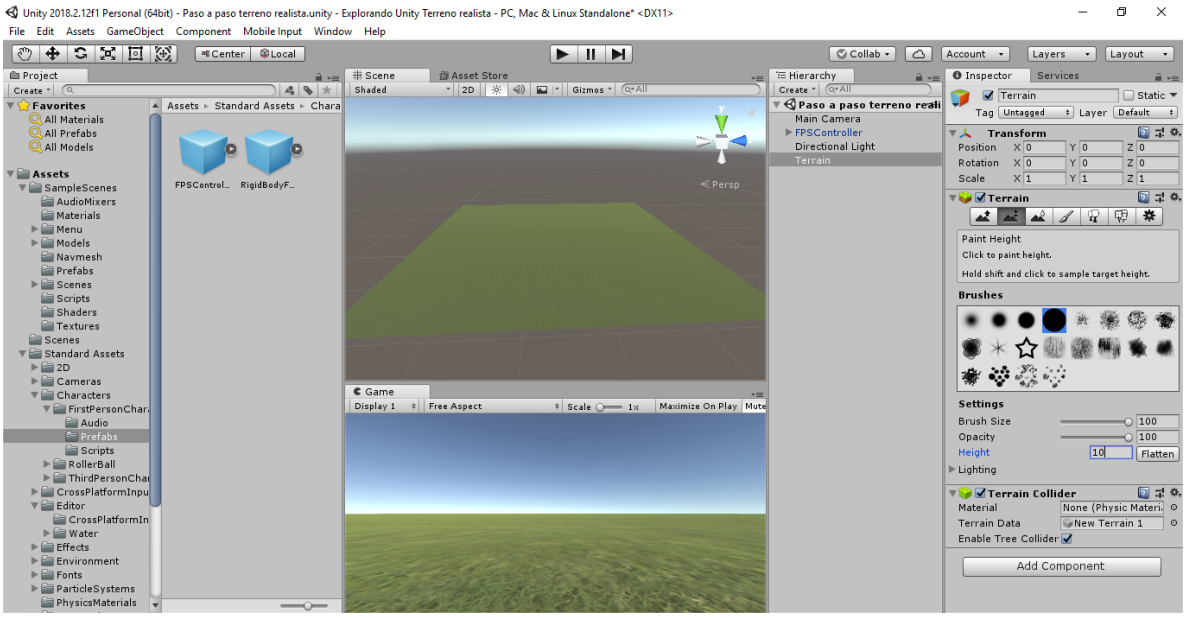


Ilustración 13. Ubicación de botones para relieve de terreno.

.

Y ahora vamos a cambiar de pincel y de *brush Size a* 4% y un *falttten* de 9. ¿Esto para qu é ? con las opciones de brush Size y una Opcaity al 100% y flatten de 10 lo que hicimos fue elevar el terreno ahora al bajarl el ancho del pincel al 4% y el *fletten a 9* le estamos diciendo a Unity que baje la altura del terreno lo que generara una especie de depresiones (huecos) de forma real sobre el terreno:

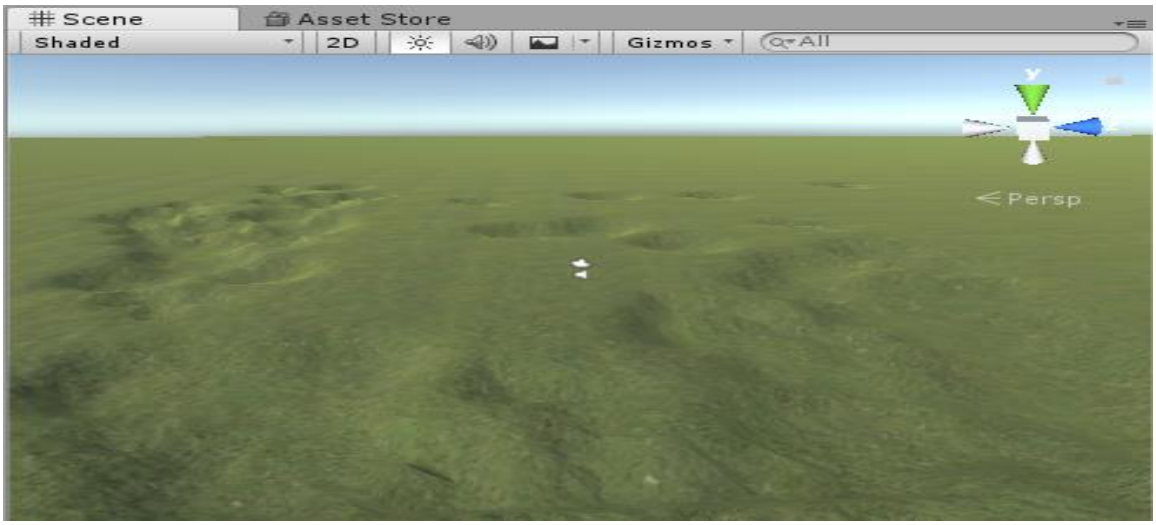


Ilustración 14. Depresiones sobre el terreno.

…siendo este el resultado.

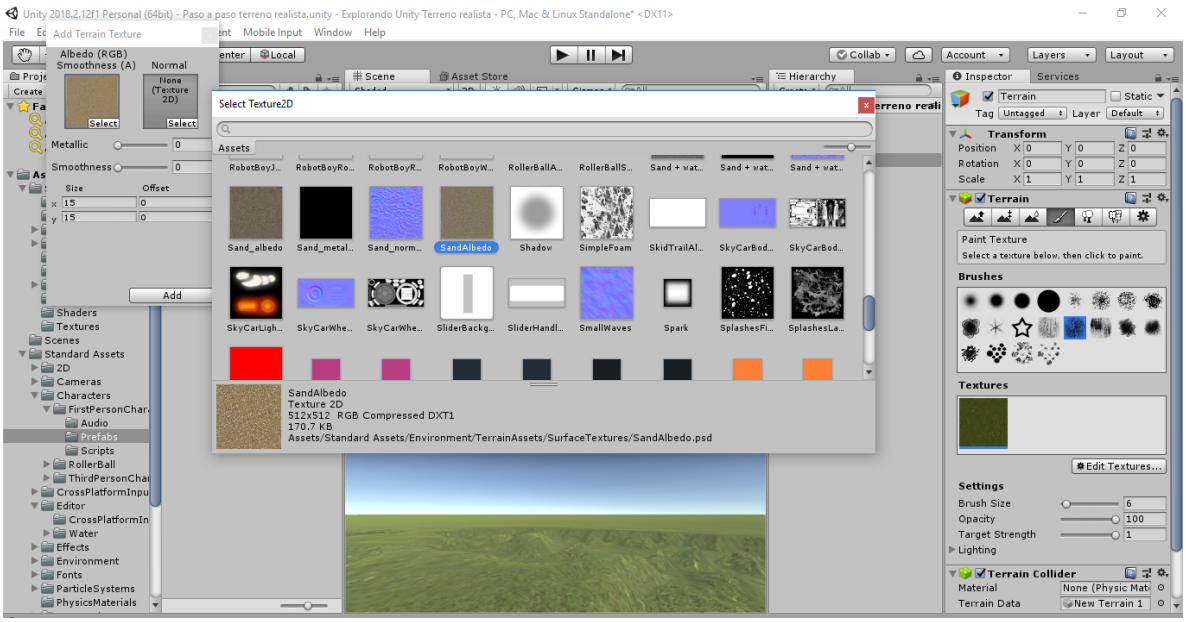


Ilustración 15. Seleccionando nueva textura.

Despu é s de seleccionada la nueva textura como lo muestra en la imagen y como explicamos en la secci ó n **1.1.3** aplicamos dentro de las depresiones que hicimos, nos quedar í a asi:

Ahora vamos a agregar monta**ñ**as para ello vamos a selecciona r el bot ó n *Raise* que



Ilustración 17. Resultado de la nueva textura aplicada.

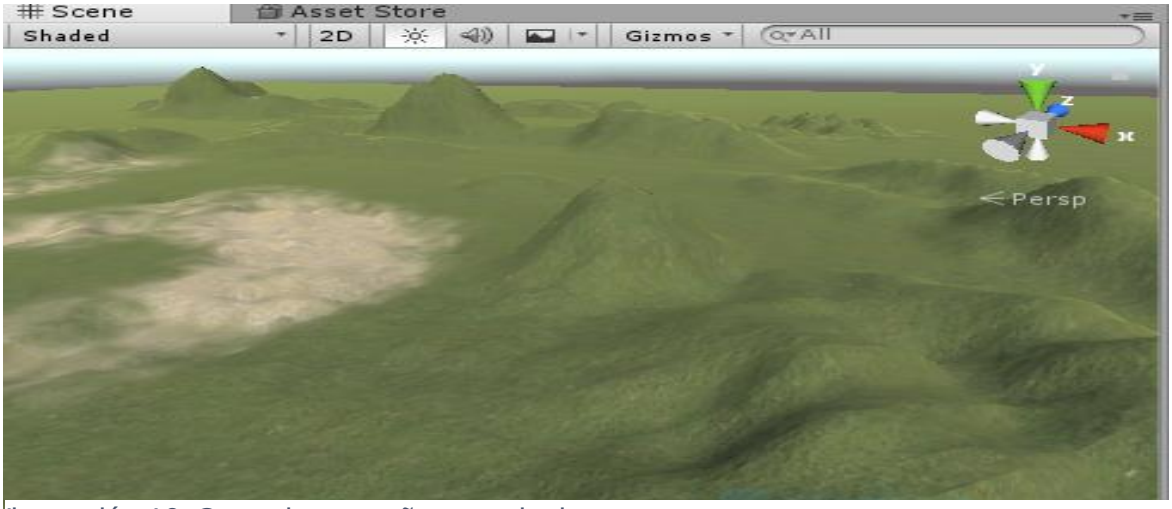
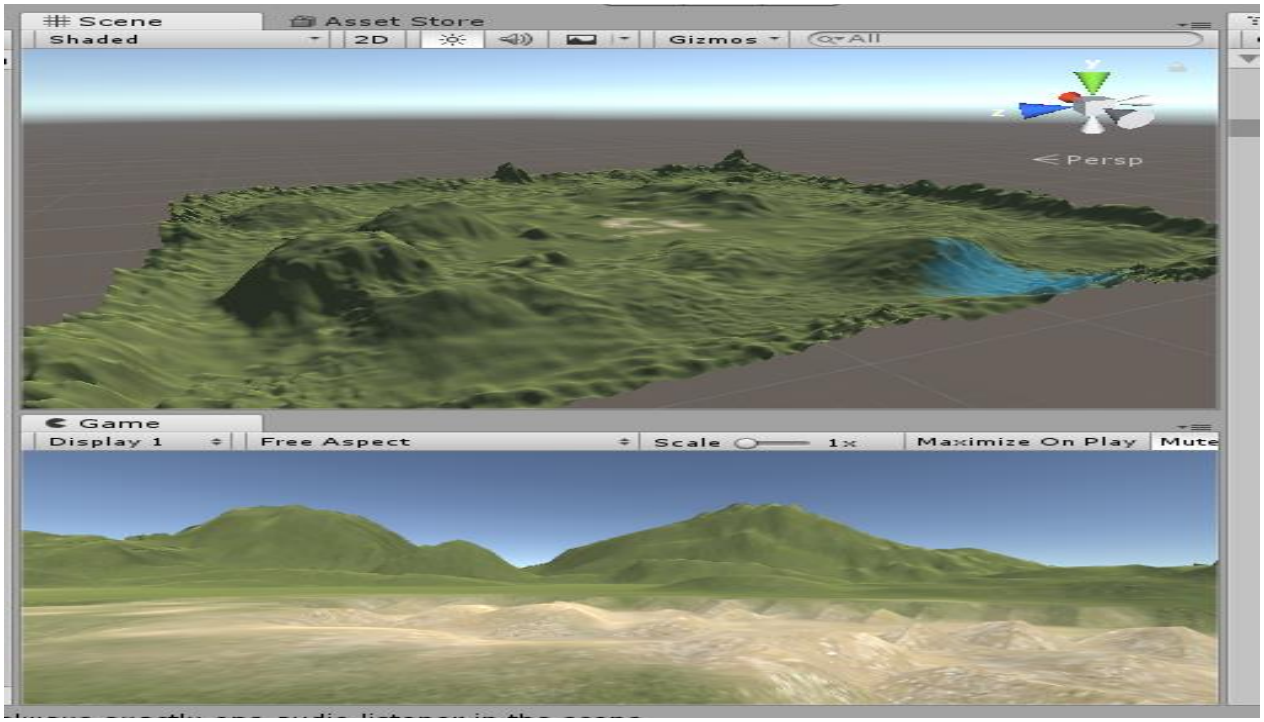


Ilustración 16. Creando montañas, resultado.

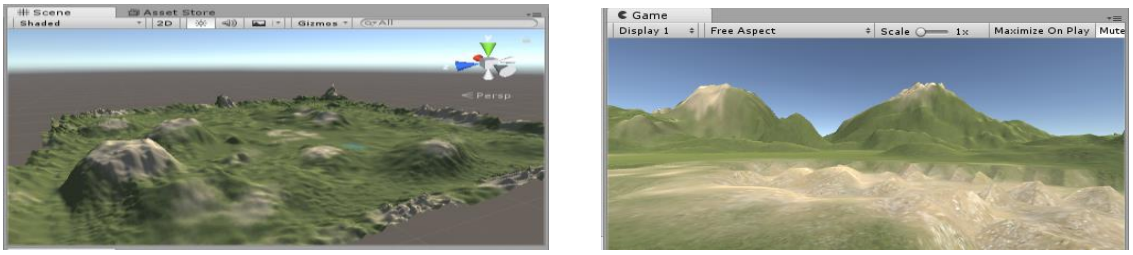
mostramos en la **ilustraci ó n 10,** ajustamos la opacidad y el ancho de pincel a nuestro gusto y aplicamos.

Para evitar que el personaje se salga del mapa en todo el borde de este bajo a generar una elevaci ó n excesiva, hacemos mas detallado los relieves, retocamos y no s quedar í a algo asi:

Ilustración 18. Relieve aplicado en todo el mapa.



Ahora cambiando de textura en el pincel, a la textura de arena. La aplicamos sobre las puntas de las montañas. Para darle un aspecto de altura y relieve natural.



Ilustraci ó n 19. Textura aplicada sobre la punta de las montañas.

vi. Vegetaci ó n

Al igual como hicimos en la seccion 1.1.3 vamos a importar un paquete de vegetacion ya sea que lo hayamos descargado o lo hayamos creado. En este caso iniciaremos con el cesped, pasto o grass que es la vegetacion base para cualquier terreno. Al igual como aplicamos texturas vamos a aplicar las grass, pero en lugar de selecciona el boton de *Raise* o *Paint Height* seleccionamos el boton de *paint trees* y agregamos como ilustramos a continuacion:

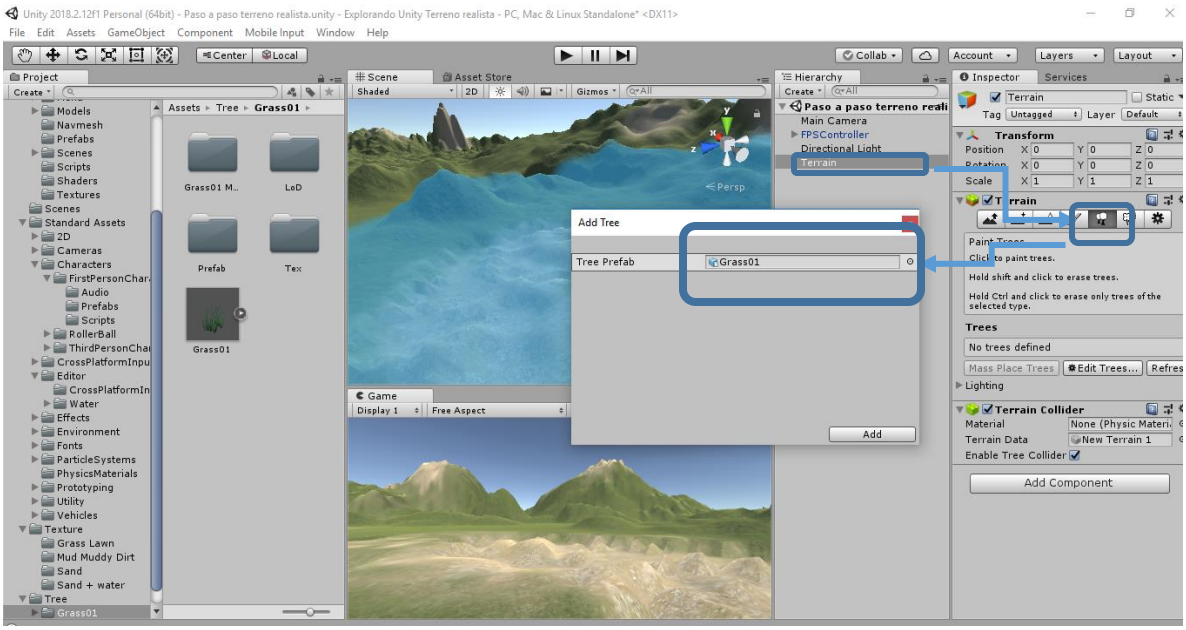
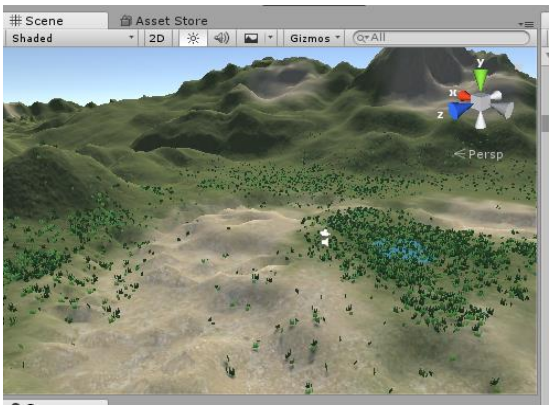


Ilustración 20. Agregando grass en la lista de paint trees.

Una vez prepara las grass arreglamos nuestro pincel a nuestro gusto y aplicamos deberia de quedar algo asi:



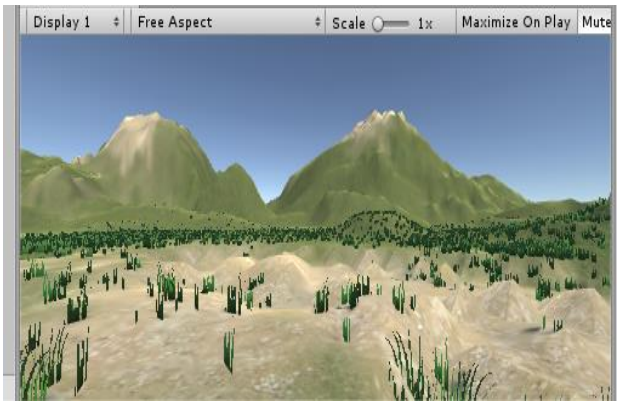


Ilustración 21. Grass aplicadas.

De la misma forma que en la ilustración 17 vamos a importar y agregar algunos arbustos y arboles a la lista de Paint tres. Y con el pincel aplicamos. Ya de aquí en adelante el trabajo depende de la creatividad del diseñador. En mi caso el resultado fue el siguiente:





Ilustración 22. Resultado final vegetación

vii. Postporcesing

Una vez terminado nuestro mapa solo nos falta agregarle algunos detalles en esta sección vamos a mostrar como agregamos el *Asset* llamado *postprocesing* que no es mas si no una serie de efectos visuales para darle un poco mas de potencia y realismo a la cámara del jugador en Unity. Empezamos importando este asset previamente descargado. Igual como hicimos en la sección 1.1.3. Una vez importado vamos a crear el posprocesing profile, para eso vamos a dar click derecho en la ventana de *assets -> created ->postprocesing profile:*

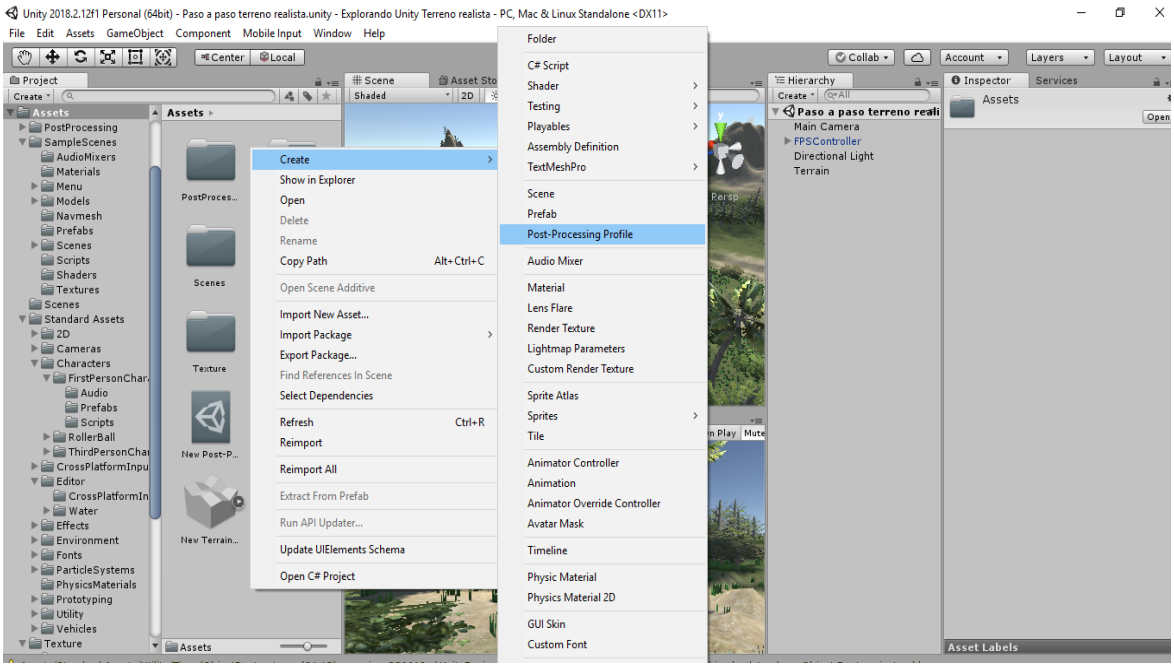


Ilustración 23. Ruta para crear el postprocesing profile.

Ahora vamos a la cámara (main cámara) en la ventana de jerarquías, la seleccionamos, damos click en el botón 

y agregamos el script 

Asi como se muestra en la siguiente imagen:

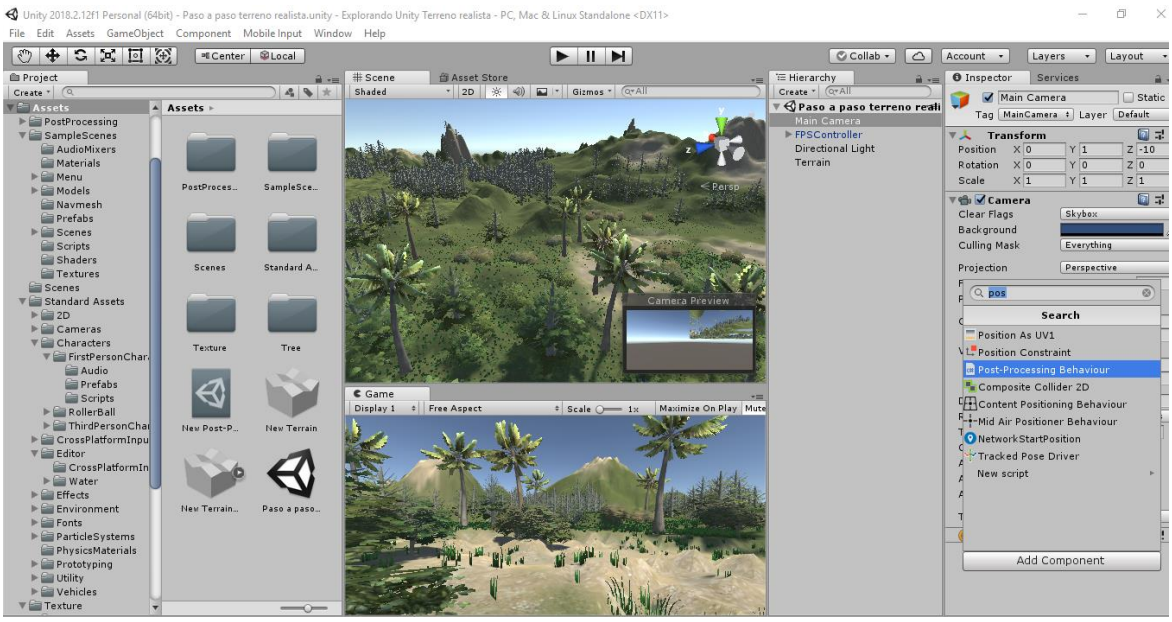


Ilustración 24. Agregando postprocesing

Listo ahora solo queda jugar con los efectos visuales de postprocesing, podemos adaptarlos a nuestro tipo de juego ya sea para darle mas nitidez a la imagen o para agregar un tipo de tono al ambiente (calido o frio) todo esto lo hacemos en el siguiente men ú .

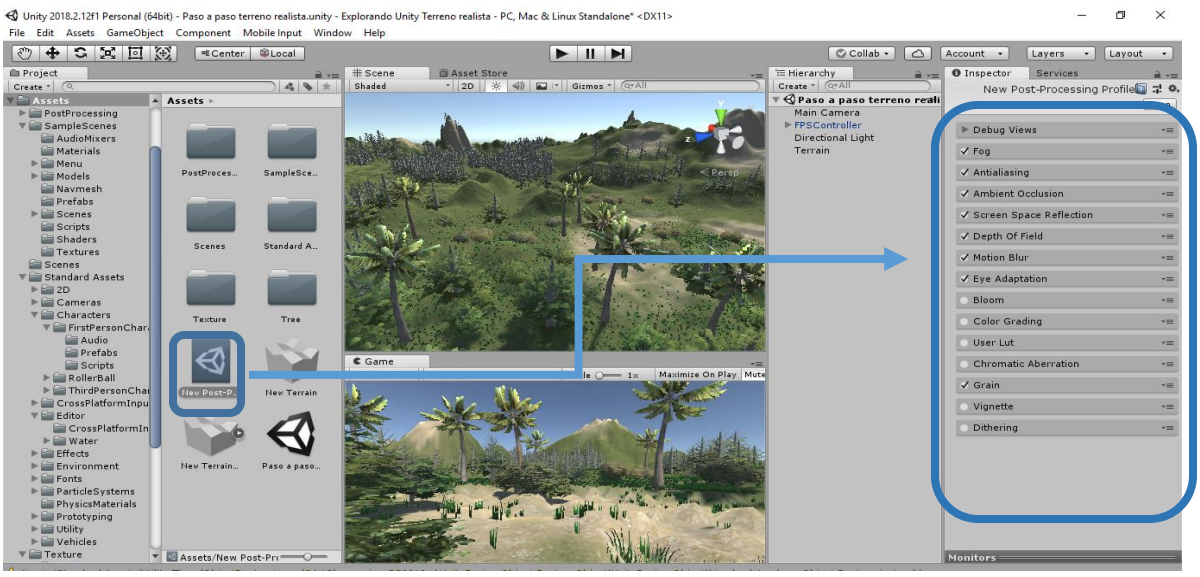


Ilustración 25. Personalizando postprocesing.

Y por ú ltimo arrastramos el postprocesing creado al espacio del script que creamos anteriormente en la c á mara, as í como e ilustra a continuaci ó n:

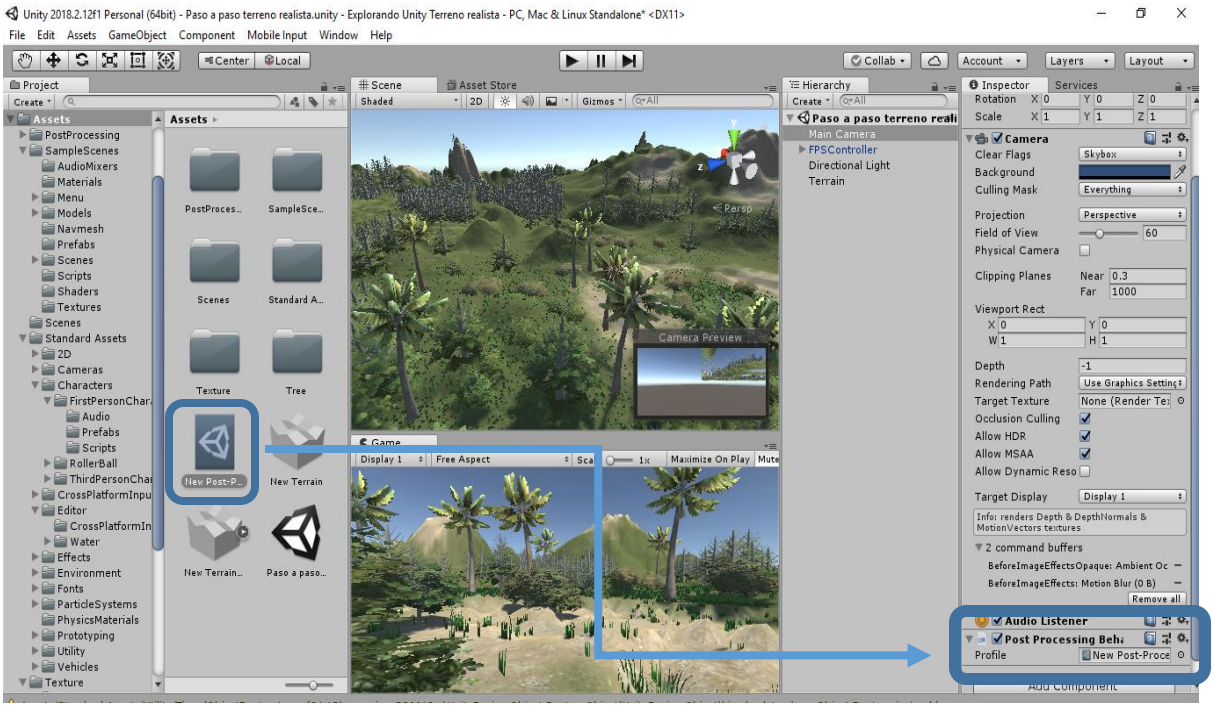


Ilustración 26. Postprocesing sobre el script.

viii. Sistema de partículas

En esta sección mostraremos como realizamos un sistema de partículas enfocado en la imitación de la niebla para agregarle aún más detalle al mapa y darle más realismo.

Empezaremos por abrir el menú *Gameobject* ->Effects -> Particle Sistem:

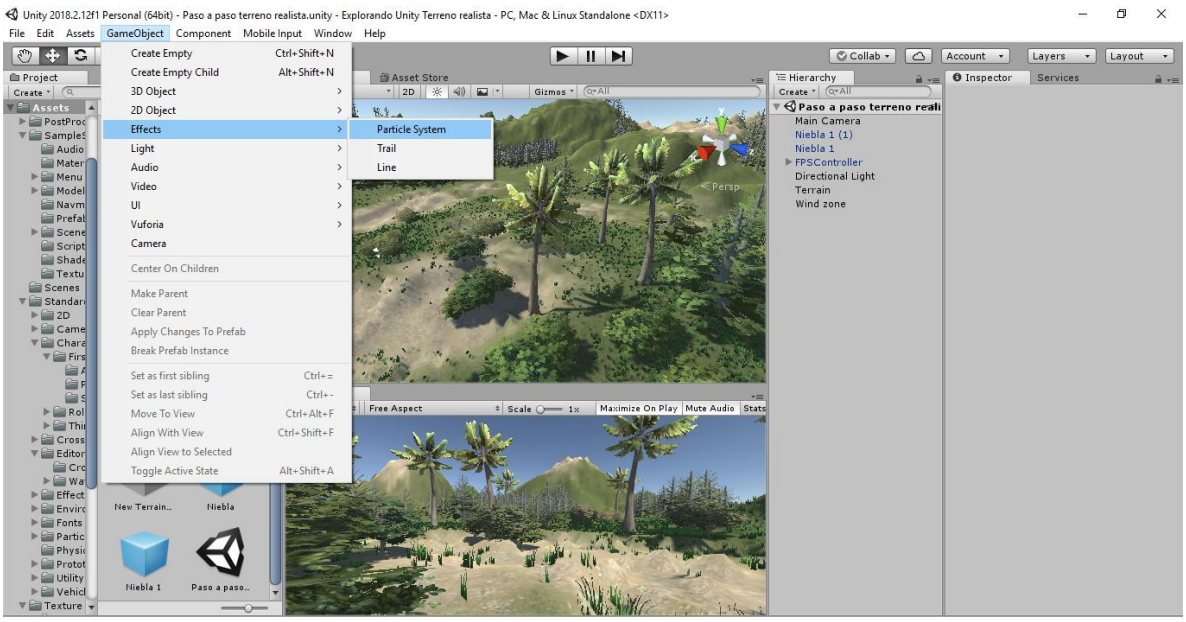


Ilustración 27. Ruta para crear efecto de sistema de partículas.

En nuestro caso lo denominamos Niebla, en el área de inspector empezamos a modificar las opciones de emisión, duración, gravedad, densidad cantidad de partículas emitidas por tiempo. Así como lo mostramos a continuación:

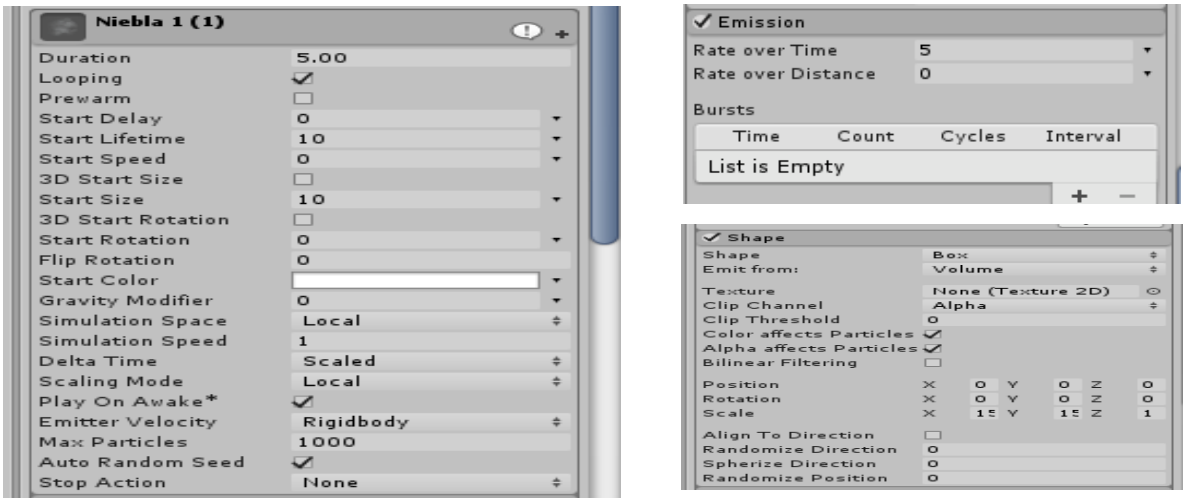
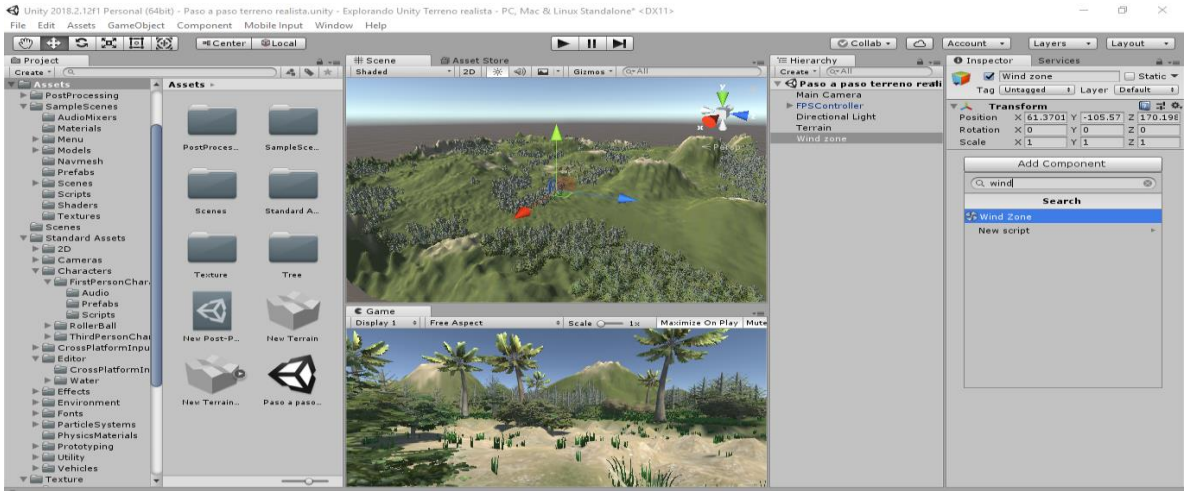


Ilustración 28. Características del sistema de partículas Niebla.

ix. Wind Zone

Por ultimo vamos a darle el efecto del viento sobre la vegetaci ó n para ello vamos a crear un objeto vac í o que llamaremos *windzone.* Y le agregaremos el componente *wind zone.* Asi como se muestra en la imagen:

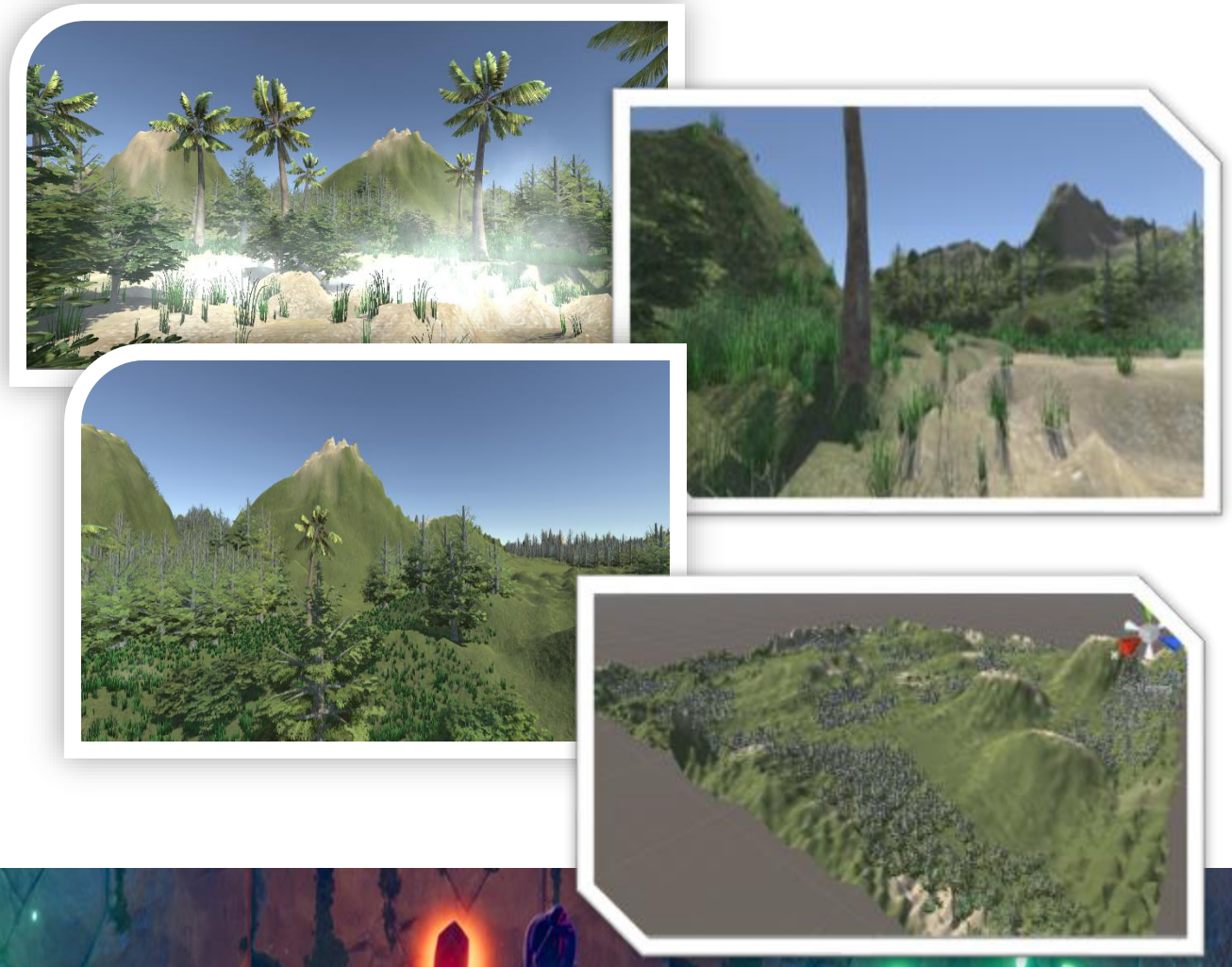
x. Aportes



En el proceso de generaci ó n de este mapa, aunque no lo usamos en el documento final, nos sirvi ó de gran ayuda para aprender a manejar Unity desde el á rea grafica. Dentro de los procesos aprendidos vale la pena mencionar importar Assets, importar objetos , agregar efectos visuales, sistemas de part í culas, movimiento, agregar c á mara (character) , modificar relieve del mapa, agregar vegetaci ó n (arbustos, arboles, grass), aplicar texturas. Conocimiento que no solo ser á aplicable para este proyecto si no para proyectos venideros en donde usaremos no solo Unity si no tambi é n motores gr á ficos para videojuegos.

xi. Mapa construido

Como ya nombramos este mapa no lo usamos en el archivo final del video juego, pero sin embargo recopilamos unas im á genes para mostrar el mapa concluimos con todo el proceso que describimos en los incisos anteriores:



3. Game kit 3D

Otro conjunto de Assets que lastimosamente quer í amos pero no pudimos incluir dentro de nuestro proyecto fue Game Kit 3D. Como su nombre nos da a entender es una seria de herramientas prefabricadas para la realizaci ó n de un juego en 3D. Ahora vamos a mostrar lo que exploramos de este conjunto de Assets, lo que manejamos y lo que logramos hacer.

i. Importando game kit

Asi como en el inciso 1.1.2 vamos a buscar a 3D game kit en el asset store y lo importamos.

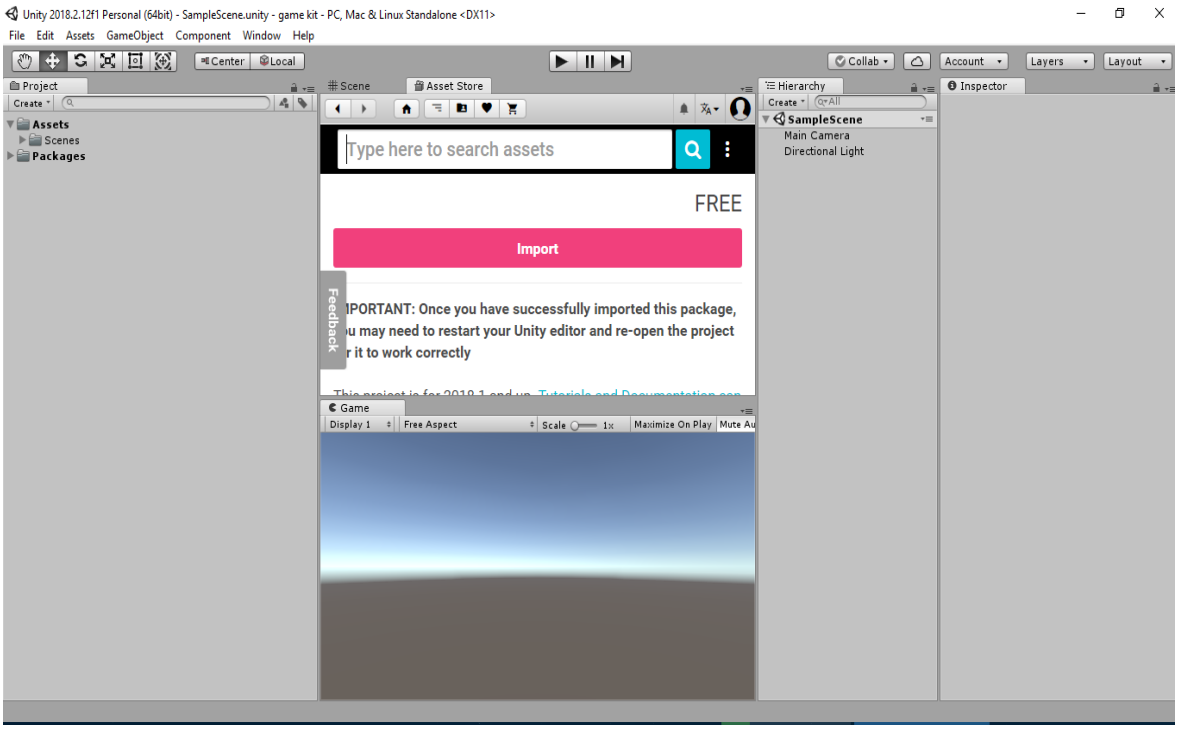


Ilustración 29. Menú de importación de game kit en asset store

Por ser un conjunto de assets tan amplio y con tantas texturas la descarga e importación de este kit de juego es bastante demorada, asi que no se sorprendan que en este solo proceso pueda llevarle horas. Una vez descargado lo importamos a nuestro proyecto.

ii. Explorando 3D game kit

Una vez descargado game kit nos dispusimos a explorar todas las carpetas que importamos a Unity allí pudimos observar que este ya trae dos niveles del juego prefabricado, dándonos a conocer las posibilidades y limitaciones del kit de juego.



Ilustración 30. Niveles de juego prefabricados.

Tambi é n encontramos bastantes prefabs, la personaje principal (Ellen, los enemigos, vegetaci ó n, rocas, paredes, columnas, objetos interactivos) algunos de los elementos que encontramos los podemos observar en las im á genes siguientes:

iii. Objetos prefabricados

Los objetos incluidos dentro del kit de juego se jerarquizan en Audio, Characters, Enemies y Environment.

1.2.3.1 Characters

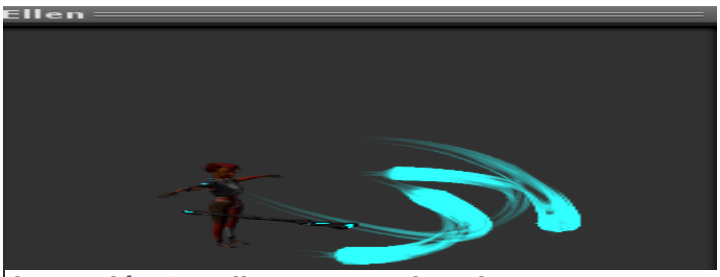


Ilustración 31. Ellen como unico character

1.2.3.2 Enemies





Ilustración 34. Chomper

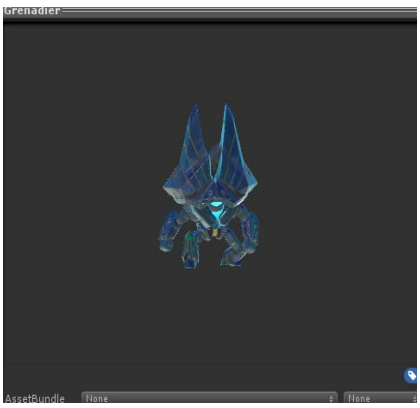


Ilustración 33 Grenadier

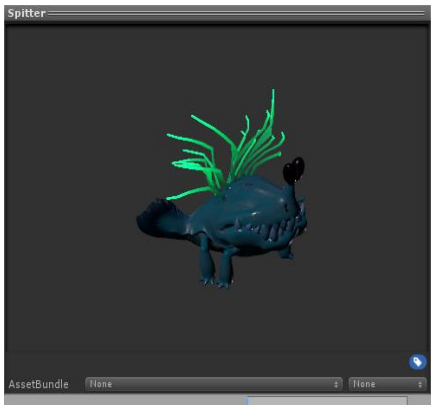
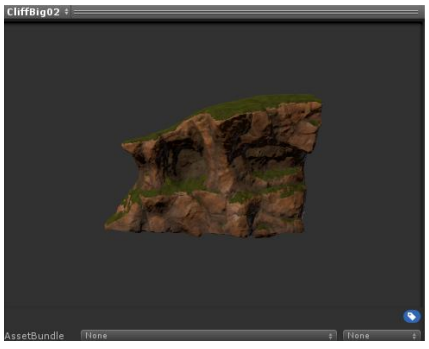


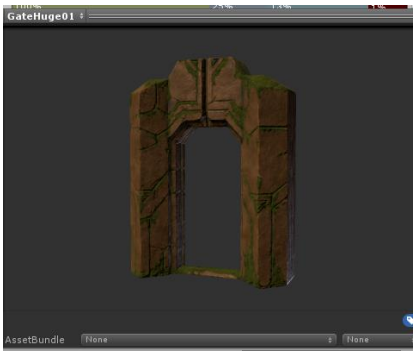
Ilustración 32. Sprinter

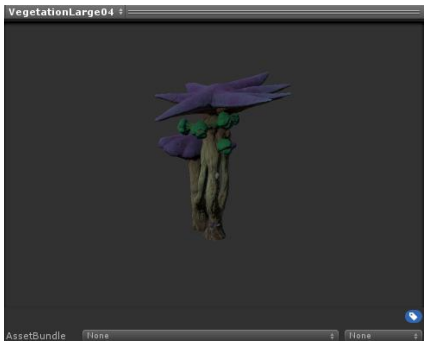
1.2.3.3 Enviroment

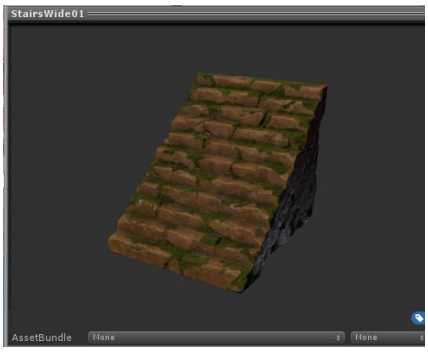
Hace referencia a todo el material de "utiler í a" algunos son interactivos, otros cumplen solo un papel est é tico. A continuaci ó n mostraremos algunos de ellos.









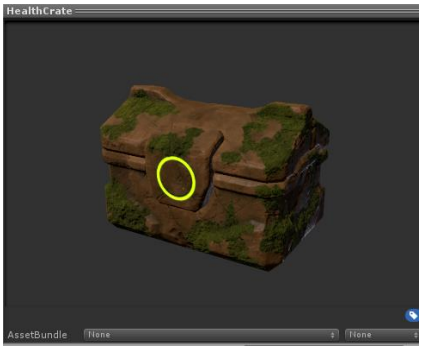


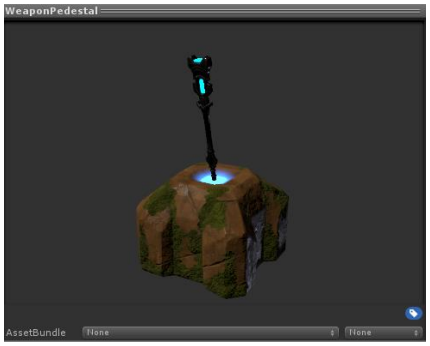


v. Creando un nuevo mapa en game kit

Despues de explorar







game kit iniciamos la creacion de nuestro propio mapa, para ello vamos al menu kit tools -> create new scena:

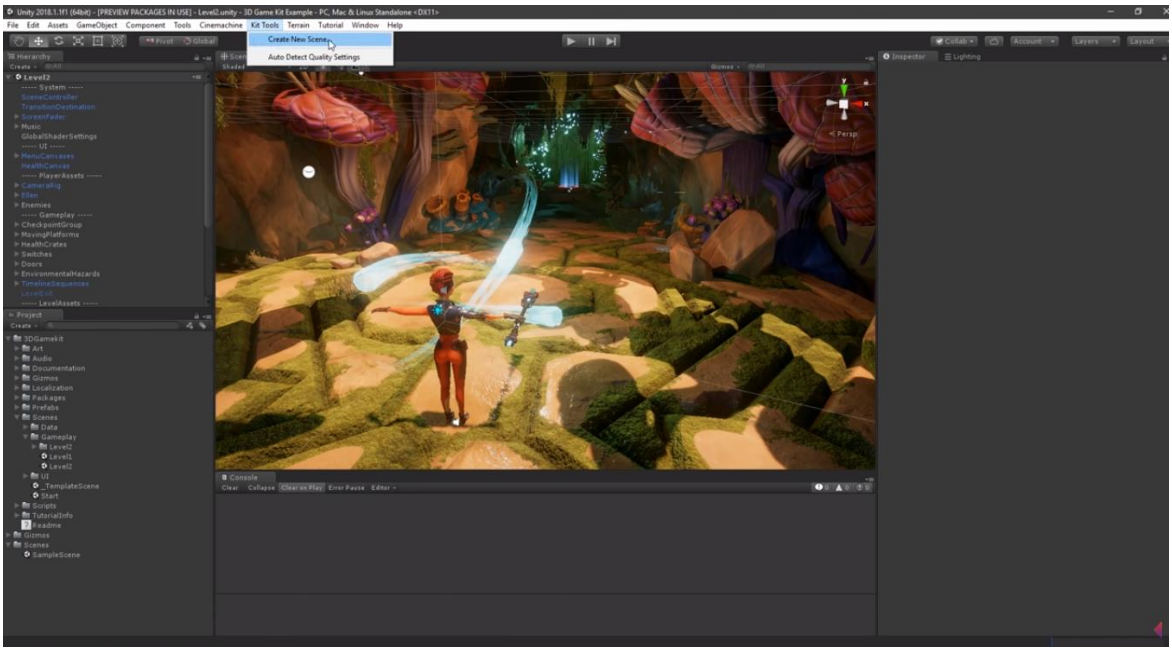


Ilustración 35. Ruta para crear un nuevo mapa.

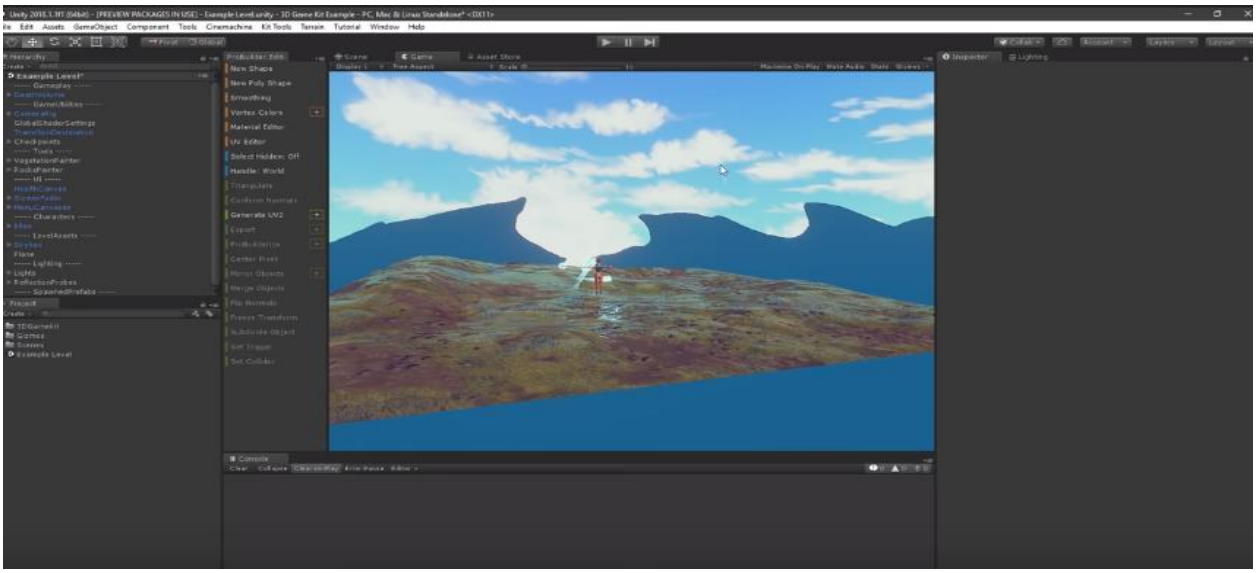


Ilustración 36. Así nos quedaría nuestro mapa inicial

Tratamos de expandir el mapa a la escala que necesitamos pero al hacerlos la textura del plano se expande tambien y perdia calidad asi que lo que hicimos fue unir retazos de este plano generando uno mas grande de la union de todos estos segementos.
Quedando algo asi:

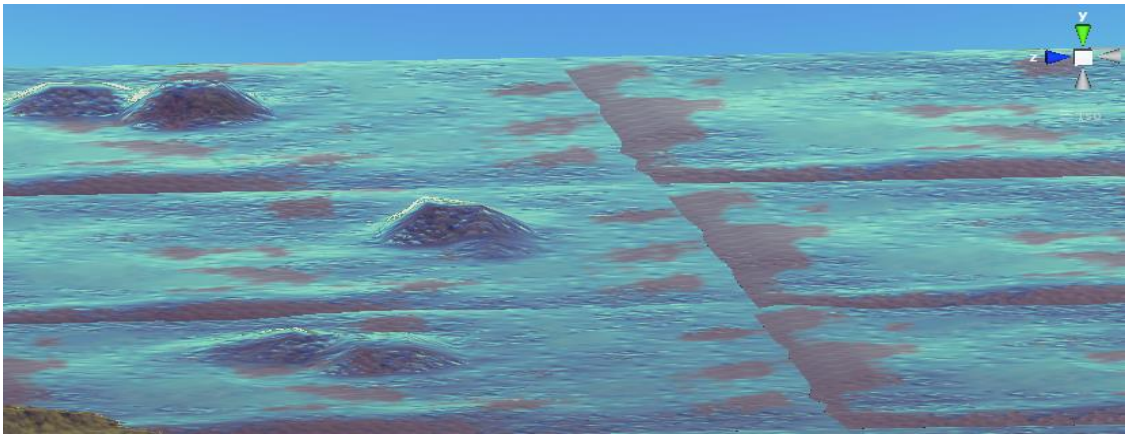


Ilustración 37 **Planos del mapa unidos.**

Una vez creada nuestra unión de planos empezamos a agregarle los límites del mapa, en la sección 1.2 el mapa inicial hicimos estos límites elevando el terreno excesivamente pero como en game kit el suelo no es un objeto de tipo *terrain* si no que es un plano, no nos permite hacer esta operación, asi que empezamos a hacer estos limites con rocas como si fueran grandes barreras.

Una vez hecho esto podemos empezar a generar nuestro mapa, pero antes de eso hicimos



un bosquejo a lápiz de cómo deberían ir nuestro mapa con game kit el resultado aunque muy poco técnico fue el siguiente:



Ilustración 38. Boceto mapa en game kit

1. Aldea

Al momento de generar nuestra aldea encontramos el primer inconveniente pues objeto como casas, edificios en si no los trae este kit, tuvimos que poner a prueba nuestra creatividad y con objetos como baldosas y piedras les fuimos dando forma a pequeñas casas.



Ilustración 39 Casa de aldea inventada con objetos de game kit.

Siguiendo el mismo patrón de construcción generamos un edificio principal, pusimos estatuas, objetos interactivos, guías y puertas corredizas.

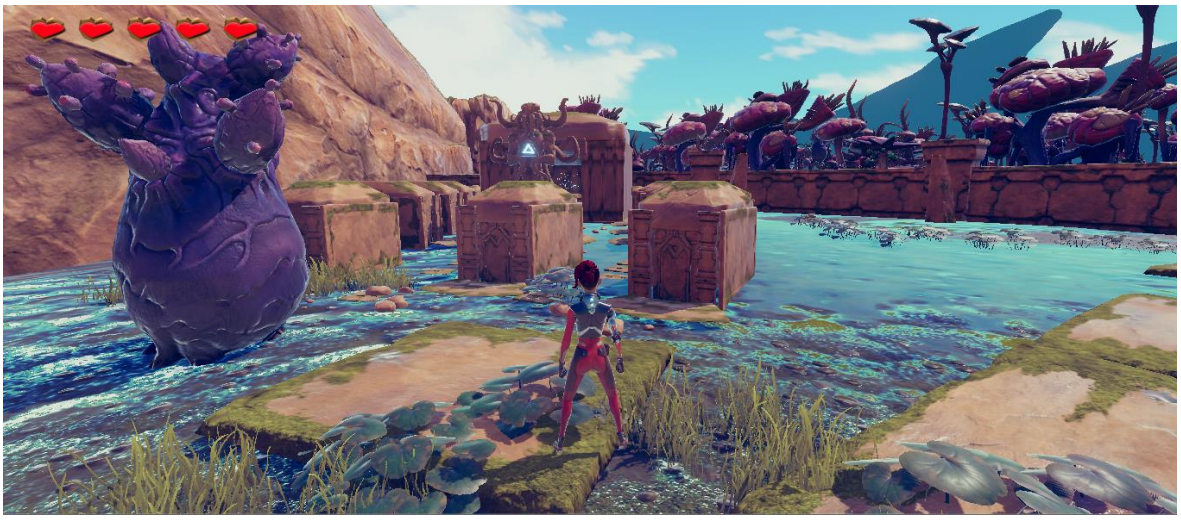


Ilustración 41. Resultado de aldea.

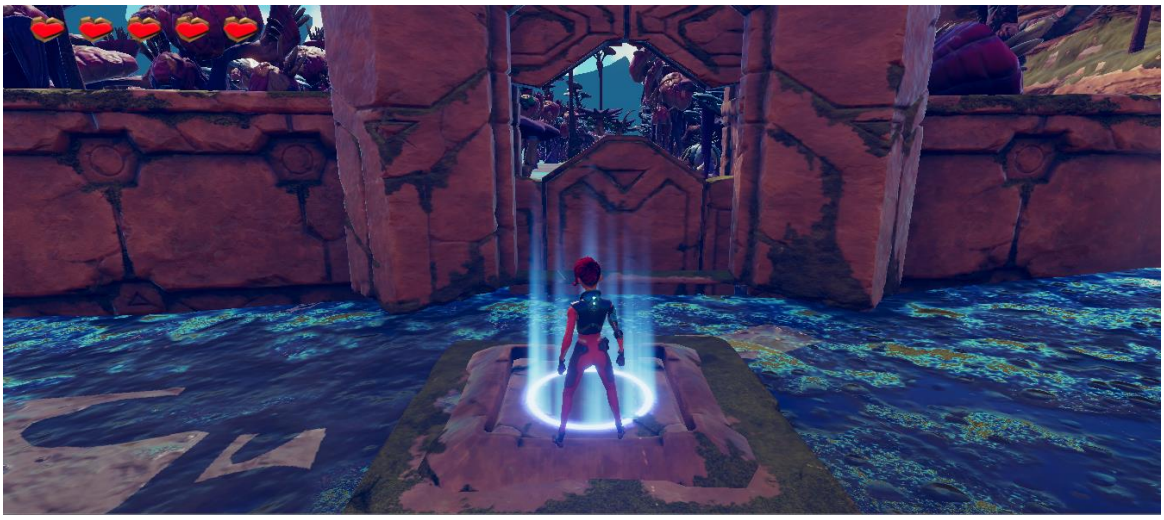
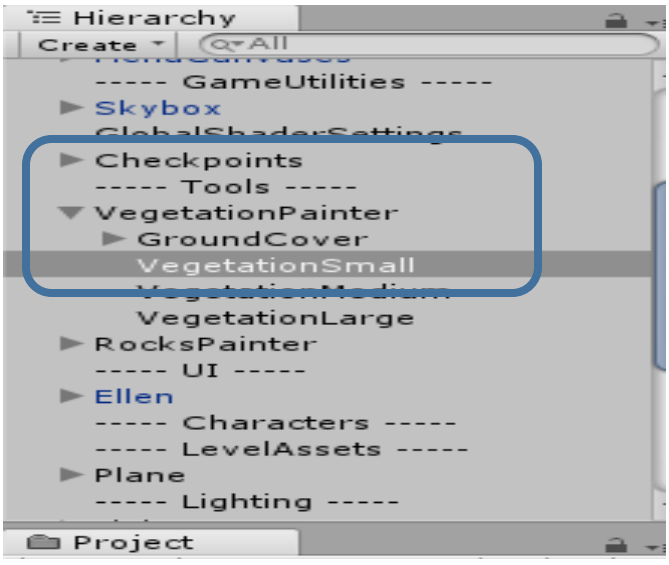


Ilustración 40 Objetos interactivos.

2. Vegetación



Al igual que hicimos en la sección 1.1 la generación de la vegetación se da por medio de pinceles en donde se modifica la cantidad de objetos por cuadro aplicados y la amplitud de alcance, pero, como lo explique en el inciso 1.2.4 el suelo en game kit consta de planos no de terrenos, entonces game kit soluciono este problema agregando el pincel en la ventana de jerarquías y lo denomino *groundcover.*



El "pincel" lo divide en tres secciones:

- Vegetación pequeña (grass, césped)
- Vegetación media (arbustos)
- Vegetación grande (arboles)

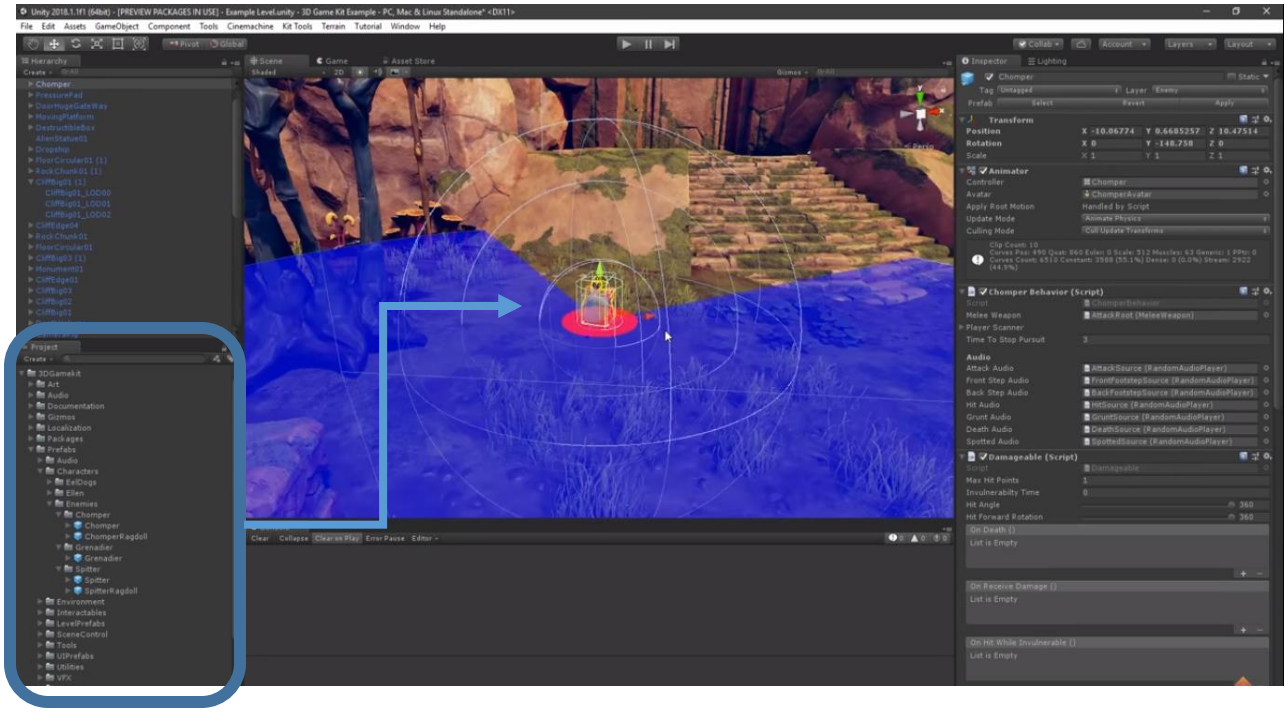
Una vez entendido como funciona nos dispusimos a aplicar vegetación en el mapa generando el bosque central como lo bosquejamos en el boceto.

Ilustración 42. Resultado de producción del bosque

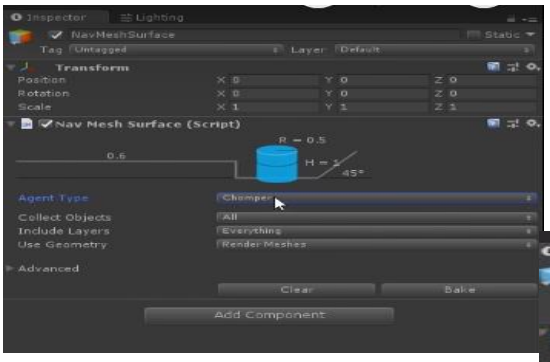
1.2.3 Criaturas

Ya tenemos el terreno, la vegetaci ó n la ambientaci ó n solo falta las criaturas enemigas, que vamos a diseminar por todo el mapa a excepci ó n de la aldea que va a ser un lugar seguro para el usuario.

Para agregar nuestros enemigos nos dirigimos a Assets->Prefabs->enemies->Chomper. Lo seleccionamos y arrastramos sobre el mapa.

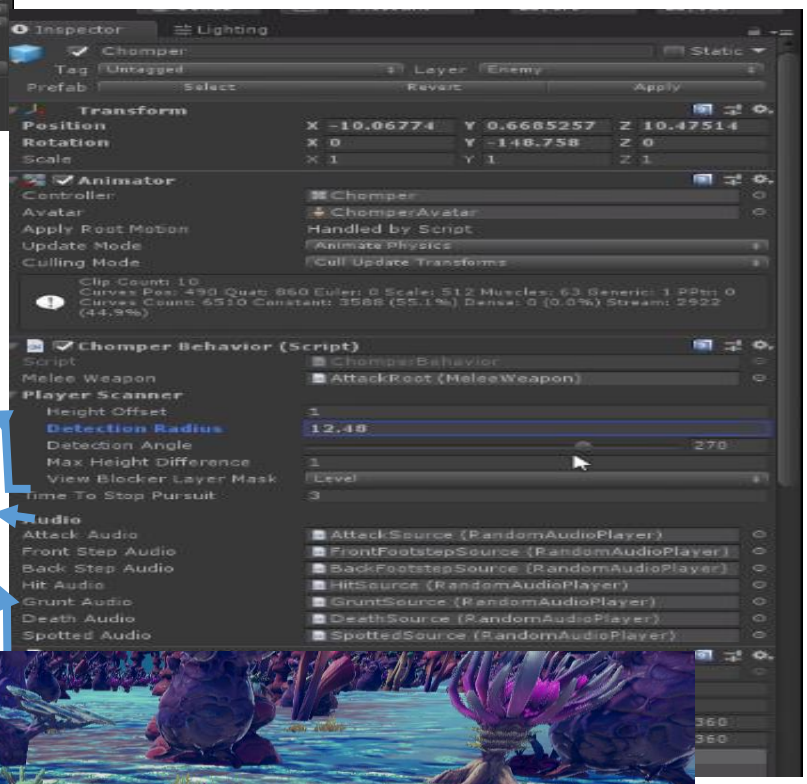


Ya quedo nuestra criatura sobre el mapa pero si ejecutamos asi nada mas el personake enemigo no puede navegar sobre el mapa, corre en el sitio pero no avanza, para solucionar ese problema vamos a crear un objeto vac í o y lo llamaremos *nave mesh Surface* y a este vamos a agregar un componente script denominado *nave mesh Surface.*



Quedaria asi, ahora en *Agent Type* seleccionamos la opcion chomper, que es el tipo de criatura que tenemos en el mapa. Ya nuestra criatura puede navegar con libertad por el mapa

Tambien podemos modificar las caracteristicas de nuestra critatura, caracteristicas como el area de detencion del enemigo, el angulo de vision y la cantidad de golpes necesarios para derribarlos.



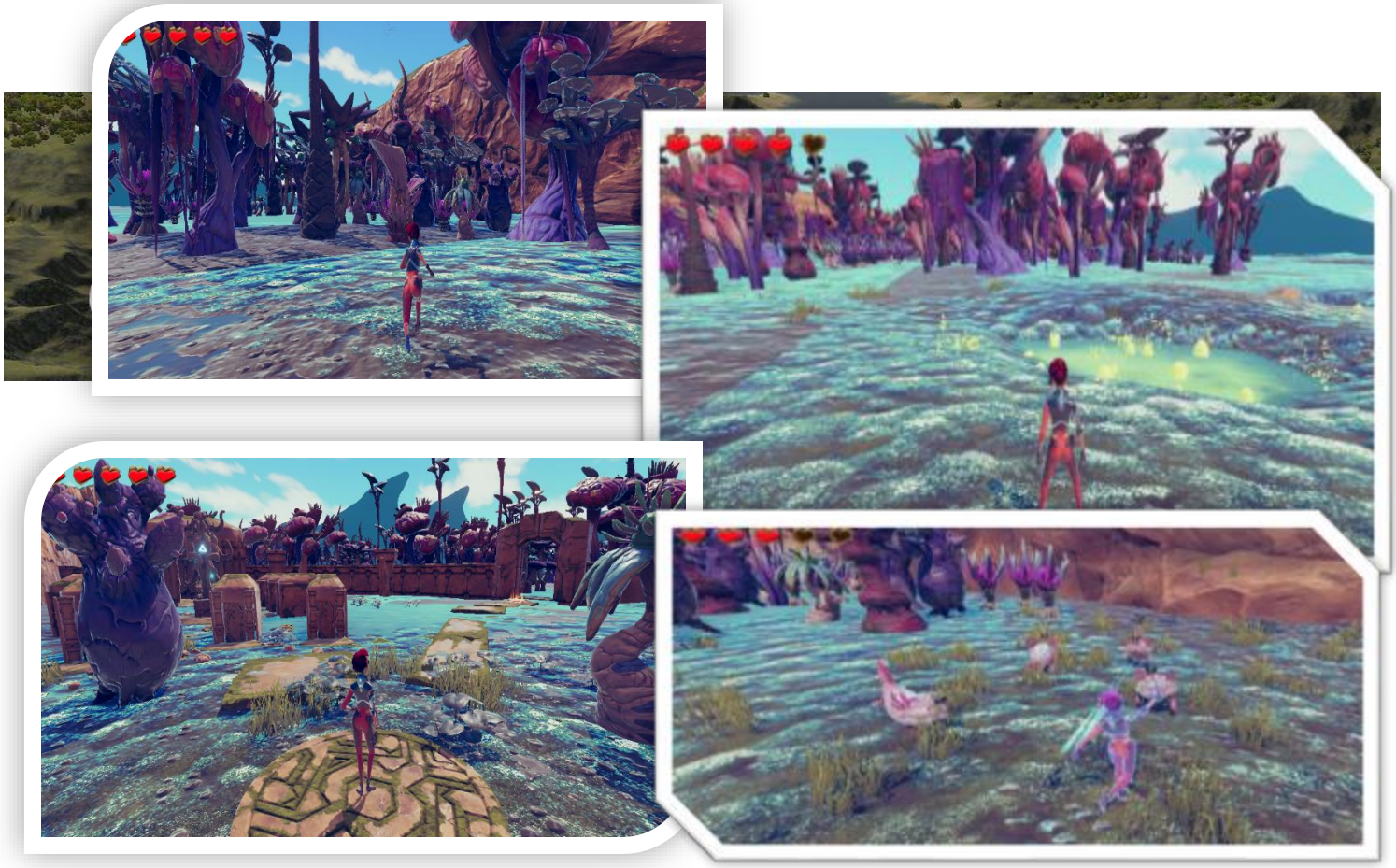


Ya agregadas nuestras criaturas el resultado fue este.

1.2.4 Aportes

Game kit nos mostro una gran cantidad de herramientas bastante útiles que nos hubiera podidos servir para nuestro mapa final, muchos objetos prefabricados ponían a prueba la imaginación del investigador que solo tenia que preocuparse por el diseño del escenario y del mapa lastimosamente al tratar de vincular game kit como otro proyecto game kit era incompatible con la Física del otro proyecto lo que generaba

incongruencias al momento de ejecutarlo. Sin embargo, aquí dejaremos imágenes del mapa generado gracias a este kit de herramientas para 3D.



4. Gaia

Este generador de terreno nos evita hacer todo el proceso que realizamos en la sección 1.1. Es una herramienta amigable para el usuario que te ayuda a generar mapas de grandes extensiones en 3D justamente lo que necesitábamos. A continuación, mostraremos el proceso que seguimos para la generación de nuestro mapa.

1.3.1 Importando Gaia

Igual como hemos hecho con las herramientas anteriores, para poder usar determinado conjunto de assets tenemos primero que todo importarlos a nuestro proyecto en Unity.

Al ser casi igual de grande que el game kit la importaci ó n de este conjunto de assets es bastante demorada tambi é n asi que no se sorprendan si en solo este paso puedan demorarse horas.

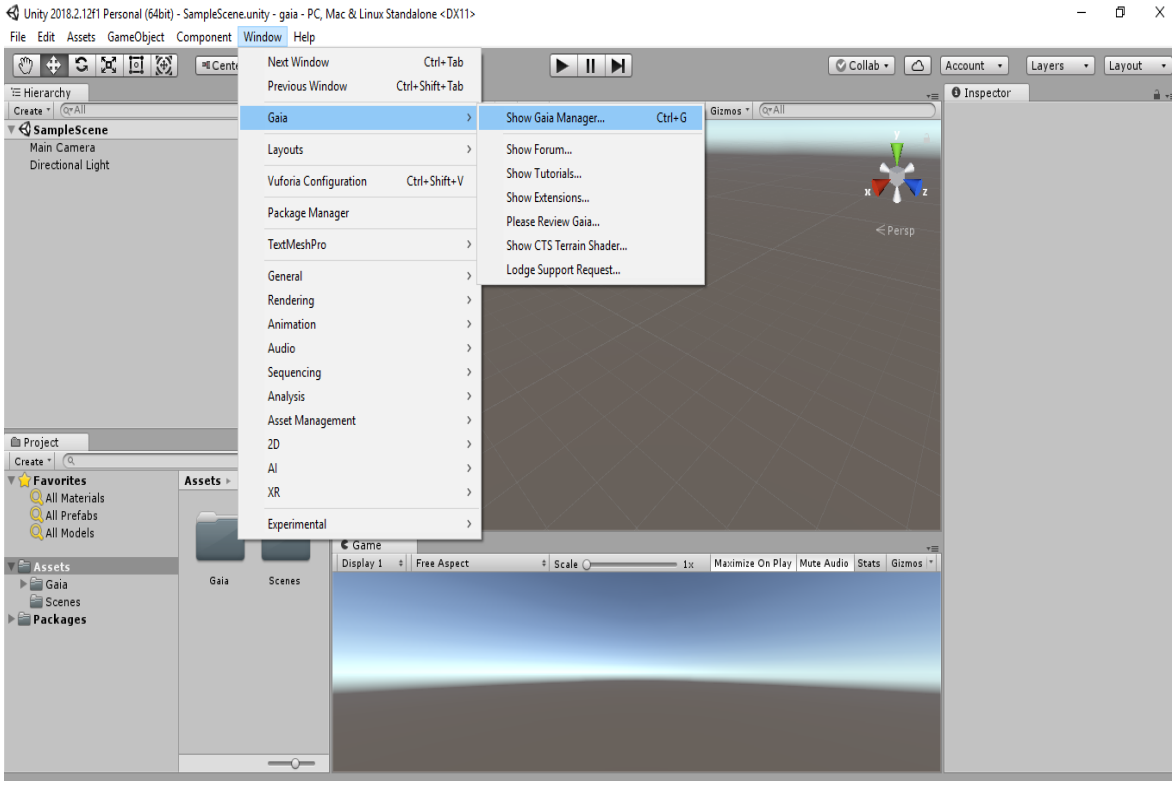
Bueno, para realizar un mapa con Gaia tenemos dos opciones:

- Hacer como en el apartado 1.1 y realizar el mapa planta por planta, con pinceladas aplicar texturas, vegetaci ó n, hacer relieve y agregar detalles.
- Por medio del Gaia Manager, el cual es una gu í a b á sica para generar un mapa con un terreno determinado, cabe resaltar que este nos ayuda a crear un mapa y su vegetaci ó n, el resto de detalles queda por parte del diseñador.

Como nosotros ya mostramos como hacer un mapa desde ceros en esta secci ó n mostraremos como hacer un mapa con el Gaia manager. No siendo mas.

1.3.2 Generando el mapa

El primer paso que debemos realizar es abrir el Gaia manager, para ello vamos a ir al menú *window->Gaia->show gaia manager.*



Este el gaia manager , el cuadro que deberia aparecer despues de seguir la ruta anterior.

En esta seccion vamos a crear un mapa Est ά ndar basico. Podemos modificar el tipo de controlador antes de empezar, el renderizado, el tamaño y el tipo de luz (el tipo de luz lo podemos dejar *movile and VR* pues nuestro proyecto sera enfocado a la realidad extendida.

1.3.2.1 Create terrain and show Stamper

En este paso Gaia nos "estampa" el terreno en Unity pero lohace de forma predeterminada de forma plana, este terreno a diferencia de la seccion 1.1 y 1.2 no es un objeto *terrain* ni *plane*. Nos quedaria asi:

Ahora vamos a darle el relieve realista para eso nos sirve el Stamp, pues gaia trae ya algunos relieves predeterminados que generan la altura del terreno dependiendo de la tonalidad de estas im á genes.

Ilustración 43. Stamps de gaia para el mapa

Ahora lo que haremos es seleccionar el "stamp" que nos sirve y arrastrarlo al area de inspeccion del Stamper de nuestro mapa, asic omo se muestra a continuacion.

Podemos alsar o mover el terreno desseleccionando el punto pase en el cuadro de cheklist del area de inspeccion.

Con esto podemos mover el terreno con total libertad, ya sea para que se vea mas alzado como una montaña o para que se vea menos terreno. Pues quedaria sumergido bajo el agua. (una vez hecho el estampado el generador gaia nos produce un plano con textura de agua)

En mi caso lo deje asi:

Ilustración 44. Estampado del relieve del mapa.

1.3.2.2 Create spwners

Este es el segundo paso que se define en agregar texturas, el generador nos ayuda a diseminar las texturas de acuerdo a la altura del relieve del terreno, al igual que en la secci ó n 1.2.4.2 (vegatacion gamekit) estos spowners son herramientas prefabricadas de gaia hechas para agregar coberturas de texturas, de objetos de juego (rocas, troncos), cantidad de arboles, arbustos, c é sped y dem á s. Las activamos en el gaia manager:

Entonces vamos a aplicar *Coverage Texture Spawner,* para ello en la ventana de jerarquía vamos a seleccionar *Coverage Texture Spawner.* Y a su vez en la ventana de inspección de este objeto le damoc click en *spawn,* asi:

Así mismo haremos con la herramienta de estampado de objetos de juego, de árboles, arbustos y césped. Es el mismo proceso, no lo mostramos para no parecer repetitivos. Vale la pena resaltar que cada *coverage* tiene en la ventana de *inspector* unas características de aplicación que pueden ser modificadas antes de ponerlas en el mapa: densidad, detalles, cantidad de una u otra planta por aplicación, ya queda a gusto del diseñador modificar esta caracteristicas.

1.3.2.3 Create player, wind, water and screenshotter

En este último paso del manager de gaia crearemos el jugador (la camara con movimiento), la zona de viento (en la sección 1.1 la hicimos manualmente con gaia se hace automático), agregaremos el agua y la pantalla.

Para ello daremos clic en el inciso tres del cuando gaia manager:

Al igual que las secciones anteriores no debemos limitarnos a lo que el generador de mapas nos haga por defecto podemos agregar mas detalles, modificando el jugador, cambiando la posición del plano Water, acelerando o disminuyendo el efecto de Wind (viento). Todo depende de nuestro proyecto y nuestros intereses.

1.3.2.4 Aportes de Gaia al proyecto

Gaia nos ofrece una gran cantidad de herramientas para el desarrollo de mapas forestales en 3D, como pueden observar esta es una gu í a r á pida de c ó mo generar uno de estos, de manera comercial gaia en sus videos nos muestra mapas de muy alta calidad y se puede llegar a ese punto siempre y cuando tambi é n tengan en su haber las extensiones de los mismos creadores, sin embargo y en comparaci ó n al primer mapa que hicimos la calidad del mapa generado con Gaia es mayor. En este caso como no tenemos las extensiones usamos Assets gratuitos para mejorar y agregar a ú n m á s detalles a nuestro mapa. Los asstes que usaremos son lo mismos que usamos en la secci ó n 1.1.2 ,1.1.7 y 1.18.

Despu é s de un trabajo detallado, en donde agregamos m á s plantas de manera individual, c é sped, sistema de part í culas , y tambi é n en donde ponemos en uso con lo anteriormente aprendido, llegamos al siguiente resultado:

5. Robot Kyle

Uno de los factores principales despu é s de tener listo el espacio de trabajo (mapa) es el *player* est á bamos en un dilema. Como el juego va proyectado hacia una visualizaci ó n en Realidad Virtual quer í amos poner la c á mara en primera persona ¿Pero que suceder í a con la poblaci ó n que es sensible a este tipo de est í mulos? Cuando una persona no esta acostumbrada al movimiento de c á mara en realidad virtual estar expuesto a estas experiencias puede producir mareos, para evitar estos giros rapidos decidimos trabajar con el *player en tercera persona.*

1.3.1 Importando Space robot Kyle

Igual como hemos hecho con todos los Assets anteriores abrimos el assets store y en la barra de b ú squeda ponemos en nombre completo del avatar.

Los descargamos y posteriormente lo importamos. Nosotros trabajamos directamente sobre el mapa de gaia que construimos en la secci ó n 1.3 de este documento.

Ahora buscamos en la ventana de assets->robotkyle->model->Robotkyle arrastramos este archivo y lo ponemos sobre el mapa bien ubicado.

Teniendo al personaje dentro del mapa vamos a crear un objeto vacio titulado Character y buscamos en la barra de jeraquias los objetos robot kyle y *maincamera* y lo arrastramos a character.

Con esto nos aseguramos de no generar incongruencias con el jugador al momento de agregar mas objetos y tambien de que la camara siga al personaje.

El objeto *maincamera* va a ser nuestra camara principal y la vamos a ubicar detr á s del robot que quede centrada para generar el efecto de tercera persona.

1.3.2 Animando a Kyle

Es hora de darle movimiento biomecánico al avatar , vamos a empezar por seleccionar dentro de la ventana de jerarquiar el objeto llamado *robot kyle* que debe de estar dentro de la carpeta *character* que acabamos de crear y borramos cualquier componen de animación que traiga por defecto en el área de *inspector*.

Creamos en la carpeta de proyecto un archivo de tipo *animator controller*:

Y lo abrimos:

En la seccion de Kyle robot llegamos hasta el punto en donde empezabamos a animar al personaje, decidimos detenernos en esa instancia ya que de ahí en adelante empezábamos a poner en práctica el conjunto de herramientas para locomoción titulado *raw mocap data for mecanim. A*si como hemos hecho en secciones anteriores lo vamos a descargar en nuestro proyecto el asset *raw mocap data for mecanim,* en el mismo proyecto que estábamos trabajando con robot kyle.

Ahora seguimos en el mismo punto en el que terminamos la ultima sección (ilustración 50), sobre esta venta del *animator* vamos a darle click derecho->cretestate->empty

Lo vamos a nombrar "Idle," este estado va a hacer referencia de cuando el *player* este en reposo:

:

En la ventana de inspección del estado *Idle* vamos a buscar la animación que corresponda:

En la ventana de animación en la pestana de *Parameters*, vamos a crear un nuevo layer llamado walk y un nuevo estado llamado walking.

E igualmente como hicimos con el *estado* anterior buscaremos una animación para caminar o que se llame walk.

Aquí lo que vamos a hacer es la animación de entrada y salida de la animación *Idle a walking* y viceversa ¿Cómo lo haremos? Vamos a darle clic derecho a l cuadro de estado *Idle*, este nos desplegara una ventana y seleccionamos la opción *make Transition*, una vez hecho esto aparecerá una fleca solo apuntamos al cuadro de estado *walking* , y hacemos lo mismo pero desde el cuadro de *walking* a *idle:*

Ahora configuramos la transici ó n haciendo clic sobre la fecha de transici ó n, en la ventana de *inspector* vamos a deseleccionar la casilla *has exit time* y a a cortar la transici ó n:

En cada transici ó n agregaremos la misma condici ó n *walk* , que hab í amos creado anteriormente la ú nica diferencia es que la transicion que va *de Idle a walk* estar á como *true* y la trancision que va de *walk a idle* la dejaremos *false.* As í :

Crearemos un documento script para el movimiento de la camara, de por si de los Standard Assets que descargamos en la seccion *1.1* existe ya un script para el movimiento de la camara con el raton pero este es muy brusco, entonces vamos a descargar el codigo poniendo en el buscador *SmoothMouseLook.* Ahora vamos a dar click derecho sobre la ventana de asstets y vamos a crear un archivo c# script, asi como se muestra en la ilustracion 53.

Abrimos el archico y pegamos el codigo de *SmoothMouseLook.* Y lo agregamos como componente en el objeto de Character.

Asi quedaria nuestro archivo script como componente dentro del objeto *character*, el player de nuestro proyecto.Ya el paso siguiente es es hacer el codigo para que funcione las animacion de nuestro personaje, para eso vamos a crear otro archico *c# script* lo llamaremos como queramos, nosotros le pusimos control, y lo abrimos, al abrirlo encontramos el siguien codico por defecto:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Control : MonoBehaviour {

	// Use this for initialization
	void Start () {

	}

	// Update is called once per frame
	void Update () {

	}
}
```

Vamos a borrar el void star y lo que vamos a hacer en el código es que decirle a unity que cuando presionemos la w , entonces se active la animación de caminar. Para eso crearemos una variable para trabajar con las animaciones (anim).

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Control : MonoBehaviour {

    public Animator Anim;

       // Update is called once per frame
       void Update () {
       if (input, Getkey(KeyCode.W)){
            Anim.SetBool("Walk", true);
                }
       }
}
```

Le pondremos ahora una variable float para ponerle la velocidad con la que va a caminar el personaje y tambien que cuando la tecla W no este activa tampoco lo este la animacion de caminar, corregimos errores:

```
using System.Collections;
using UnityEngine;

public class Control : MonoBehaviour {

    public Animator Anim;
    public float WalkSpeed;


       // Update is called once per frame
       void Update () {
       if (Input. GetKey (KeyCode.W)) {
            Anim.SetBool("Walk", true);
            transform.Translate(Vector3.forward * WalkSpeed * Time.deltaTime);
                }
        else
        {
            Anim.SetBool("Walk", false);
        }
       }
}
```

Y agregamos este archivo como componente en el personaje (objeto character), y arrastramos el *animator* a la variable *Anim:*

Al personaje tambi é n le vamos a agregar la gravedad con el componente *RigidBody.* Y le agregaremos tambi é n el componente *capsulecollider* para que no traspase los objetos como si fuera un fantasma.

Y nos vamos a la escena, como pueden la capsula de colision (*capsulecollider)* se encuentra desubicada, vamos a a cuadrarla de tal manera que quede sobre todo el cuerpo del robot, no necesariamente que abarque tambi é n los brazos.

Ejecutamos y ya debio quedar la locomocion hacia adelante y la rotacion de camara con el mouse. El paso a seguir es crear de la misma manera que acabamos de hacer la secuencia de caminar hacia atras y de correr. Para eso igual como hicimos en la ilustracion 51 empezamos creando los nuevos estados Runing y WalkingBack.

hacemos exactamente los mismos pasos para WalkingBack a excepcion de un pequeño detalle, la velocidad, en este caso sera de -1.

Los dos estados creados, mas sus respectivos parametros y lineas de transicion quedarian asi:

Volvemos al codigo y empezamos por crear el codigo de ir hacia atr á s, practicamente es el mismo codigo a excepcion de la veolocidad que en este caso sera negativa y del parametro de estado usado que en este caso sera WalkBack. Y no olvidar poner tambien que cuando no este activada la tecla S que no se active la animacion caminar hacia atr á s.

Junto con el codigo escrito anteriormente quedari asi:

```
using System.Collections;
using UnityEngine;

public class Control : MonoBehaviour {

    public Animator Anim;
    public float WalkSpeed;
```

```
        // Update is called once per frame
        void Update () {
        if (Input. GetKey (KeyCode.W)) {
              Anim.SetBool("Walk", true);
              transform.Translate(Vector3.forward * WalkSpeed * Time.deltaTime);
                  }
         else
         {
              Anim.SetBool("Walk", false);
         }
    if(Input. GetKey (KeyCode.S)) {
              Anim.SetBool("WalkBack", true);
              transform.Translate(Vector3.forward * -WalkSpeed * Time.deltaTime);
         }
         else
         {
              Anim.SetBool("WalkBack", false);
         }
        }
}
```

Ahora le agregaremos el codigo para que corra, como es usual en casi todos los juegos, presionando la tecla shift correra nuestro player. Y le crearemos la variable de la velocidad de cuando corre.

```
using System.Collections;
using UnityEngine;

public class Control : MonoBehaviour {

    public Animator Anim;
    public float WalkSpeed;
    public float RunSpeed;

        // Update is called once per frame
        void Update () {
        if (Input. GetKey (KeyCode.W)) {
              Anim.SetBool("Walk", true);
              transform.Translate(Vector3.forward * WalkSpeed * Time.deltaTime);
                  }
         else
         {
              Anim.SetBool("Walk", false);
         }
    if(Input. GetKey (KeyCode.S)) {
              Anim.SetBool("WalkBack", true);
              transform.Translate(Vector3.forward * -WalkSpeed * Time.deltaTime);
         }
         else
         {
              Anim.SetBool("WalkBack", false);
         }
         if (Input.GetKey(KeyCode.W) && Input.GetKey(KeyCode.LeftShift))
         {
              Anim.SetBool("Run", true);
              transform.Translate(Vector3.forward * RunSpeed * Time.deltaTime);
         }
```

```
        else
        {
            Anim.SetBool("Run", false);
        }
    }
}
```

Vamos a Unity y en el componente Control debio haberse habilitado la variable RunSpeed le cambiamos el valor que sea mayor que cero seg ú n nuestras necesidades.

7.

Photon es un conjunto de assets o herramientas de Unity enfocadas a la conexi ó n en redes, perfecto para crear plataformas multijugador online nos otorga un espacio en sus servidores para alojar la informaci ó n de nuestro proyecto y as í poder generar una interacci ó n a distancia en vivo y en directo.

1.4.1 Importando Photon

Igual como hemos hecho con los Assets anteriores vamos a buscar a Photon en el Assets store, ya sea por la pagina o directamente en Unity y lo importamos.

1.4.2 Creando cuenta en Photon Engine

Mientras descarga e importa el Photon podemos ir creando una cuenta en la pagina oficial de Photon ¿Esto con que fin? Es para que photon nos abra un espacio en sus servidores y poder hacer mas adelante la conexi ó n.

Ahora vamos a creas nuestra nube o como la denomina Photon una App.

Una vez creada la nueva aplicaci ó n nos arroaja un ID la vamos a copiar.

Una vez terminado de importar nuestro Photon del Assets Store nos va a pedir esta ID la pegamos alli.

Ilustración 45- Pegamos la ID

1.4.3 Scripts

Una vez importado Photon y conectado con nuestro proyecto y la app creada en Photon engine vamos a crear una carpeta en la ventana de Assets denominada *Scripts* y crearemos un objeto vacio.

Dentro de la carpeta que acabamos de crear vamos a guardar todos el script de Controller que creamos en la seccion anterior.

1.4.4 Creando la conexion

Bueno ahora seleccionamos el personaje y vamos a agregarle el componente de *photon view*. Seleccionamos el trasnform y lo arrastramos como componente dentro del *photon view* creado.

Con esto nos aseguramos de la rotacion y traslacion se emparejen la pocision y rotacion de nuestro personaje.

Ahora vamos a crear un nuevo script en C# para crear la conexión, lo titularemos NetworkMager.

En este codigo crearemos una variable de *version* para asegurarnos que aquel que tenga el juego esten jugando con el mismo mapa y con las mismas actualizaciones.

Quedaria asi:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class NetworkManager : Photon.MonoBehaviour {
    public string Version;
       // Use this for initialization
       void Start () {
        PhotonNetwork.ConnectUsingSettings(Version);
       }
}
```

Ahora vamos a crear la sala de tipo global en donde podamos alojar todos los usuarios y de paso le agregaremos el numero maximo de usuaios que pueden ingresar.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class NetworkManager : Photon.MonoBehaviour {
    public string Version;
       // Use this for initialization
       void Start () {
        PhotonNetwork.ConnectUsingSettings(Version);
       }

    void OnConnectedToMaster()
    {
        PhotonNetwork.JoinOrCreateRoom("TallerXR", new RoomOptions() {maxPlayers=25
},null);
    }
}
```

El siguiente paso es instanciar nuestro personaje, cabe la pena aclarar que debemos crear una carpeta llamada Resources en la ventana de Assets y alli guardaremos nuestro

personaje, todas las cosas que nosotros queramos instanciar en nuestro servidor tiene que estar dentro de esta carpeta.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class NetworkManager : Photon.MonoBehaviour {
    public string Version;
       // Use this for initialization
       void Start () {
         PhotonNetwork.ConnectUsingSettings(Version);
       }

    void OnConnectedToMaster()
    {
        PhotonNetwork.JoinOrCreateRoom("TallerXR", new RoomOptions() {maxPlayers=25
},null);
    }
    void OnJoinedRoom()
    {
        PhotonNetwork.Instantiate("Personaje", transform.position,
transform.rotation, 0);
    }
}
```

En la ultima seccin del codigo estamos creando la sala TallerXR. Cuando el usuario abra el juego este la creara, pero, si ya hay una sala creada con ese nombre entonces se unira a esa sala ya creada.

Creamos un objeto para poner el script creado y le ponemos la version de nustro proyecto. A este objeto tambien le agregamos un Photon View pero en lugar de arrastrar el trasnform arrastramos el objeto

Network Manager.

¿Pero como sabermos en que estado se encuentra la conexi ó n? Photon nos ayuda con esto, Photon tiene un componente que agregaremos al objeto Network manager, el componente se llama Show Status with connecting, lo buscamos y lo agregamos, el trabajo de este componente es que mediante un skin nos mostrara el estado de la conexi ó n.

Bueno entonces, en la carpeta Resources vamos a arrastrar el Personaje que teniamos en la ventana de jerarquias. Y lo borramos de la ventana de jerarquias. Ya estando en la carpeta Resources y con el codigo que hicimos se instanciara automaticamente al ejecutar el juego (lo duplicara).

Ejecutamos:

Aquí podemos verificar que efectivamente si se conecto. Ya hecha la conexión vamos ahora a arreglar unos errores que se presentan al conectarse dos o mas personas a la sala.

El primer problema es la camara, como la camara esta dentro del personaje persigue es al conjuno de codigo de personaje ¿pero que pasa cuando se conecta otro personaje que es exactamente igual? Pues la camara se confunde y se nos desconfigura.

Sacamos al personake de la carpeta de Reosurces, y lo ponemos de nuevo en la ventana de jerarquia, alli eliminamos la camara que tiene dentro.

Ahora eliminamos el personaje de la carpeta de Resources y ponemos este que esta sin camara, ejecutamos para corroborar que no se haya desconfigurado nada.

Ahora crearemos una camara aparte en la ventana de jerarquias, y a esta camara le agregaremos el componente *smoothfollow.* A este componente vamos. No faltaria si no probarlo.

Conclusión

Después del proceso anteriormente mostrado nos atrevemos a decir que las nuevas tecnologías están siendo muy poco aplicadas en el ámbito educativo, se han llevado casi a su máximo potencial, pero en el área comercial, es deber de nosotros como educadores tomar esa tecnología potenciada y aplicarla en la educación.

Si nos ponemos a revisar la mayoría de entidades de la sociedad a través del tiempo desde hace 50 años (fabricas, tecnología, hospitales) todos han cambiado

a excepción de la escuela, la escuela es el único lugar que en pleno auge tecnológico aún seguimos en clases magistrales convencionales el hecho de llevar un videojuego en 3D genera disrupción imagínense ustedes que pasaría si llevamos este mismo juego, pero en Realidad virtual.

Queremos no parar solamente en este videojuego, el acceso a la tecnología actualmente es posible, en cada hogar existe un televisor, celular, computador, herramientas educativas y nosotros podríamos sacar provecho de ello.