

**INSTITUTO POLITECNICO NACIONAL**

**UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERIAS Y TECNOLOGIAS AVANZADAS**

Trabajo:

**Proyecto Final**

**“Videojuego en Unity”**

Carrera:

**Ingeniería en Mecatrónica**

Asignatura:

**Programación Avanzada**

Presenta:

**Villafranca Tobón Cesar**

**Tapia Romero Roberto Ángel**





**INTRODUCCIÓN**

¿Qué es Unity?

Unity es un [motor de videojuego](https://es.wikipedia.org/wiki/Motor_de_videojuego) multiplataforma creado por Unity Technologies. Unity está disponible como plataforma de desarrollo para [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows), [OS X](https://es.wikipedia.org/wiki/OS_X), [Linux](https://es.wikipedia.org/wiki/Linux). La plataforma de desarrollo tiene soporte de compilación con diferentes tipos de plataformas

Unity puede usarse junto con [Blender](https://es.wikipedia.org/wiki/Blender), [3ds Max](https://es.wikipedia.org/wiki/3ds_Max), [Maya](https://es.wikipedia.org/wiki/Maya_(programa)), [Softimage](https://es.wikipedia.org/wiki/Softimage), Modo, [ZBrush](https://es.wikipedia.org/wiki/ZBrush), [Cinema 4D](https://es.wikipedia.org/wiki/Cinema_4D), Cheetah3D, [Adobe Photoshop](https://es.wikipedia.org/wiki/Adobe_Photoshop), [Adobe Fireworks](https://es.wikipedia.org/wiki/Adobe_Fireworks) y Allegorithmic Substance. Los cambios realizados a los objetos creados con estos productos se actualizan automáticamente en todas las instancias de ese objeto durante todo el proyecto sin necesidad de volver a importar manualmente.

El motor gráfico utiliza [OpenGL](https://es.wikipedia.org/wiki/OpenGL) (en Windows, Mac y Linux), [Direct3D](https://es.wikipedia.org/wiki/Direct3D) (solo en Windows), [OpenGL ES](https://es.wikipedia.org/wiki/OpenGL_ES) (en Android y iOS), e [interfaces](https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones) propietarias (Wii). Tiene soporte para [mapeado de relieve](https://es.wikipedia.org/wiki/Mapeado_topol%C3%B3gico), mapeado de reflejos, [mapeado por paralaje](https://es.wikipedia.org/wiki/Mapeado_por_paralaje), oclusión ambiental en espacio de pantalla, sombras dinámicas utilizando mapas de sombras, render a textura y efectos de post-procesamiento de pantalla completa.

Se usa el lenguaje [ShaderLab](https://es.wikipedia.org/w/index.php?title=ShaderLab&action=edit&redlink=1) para la creación de [sombreadores](https://es.wikipedia.org/wiki/Shader), similar a [Cg](https://es.wikipedia.org/wiki/Cg_(lenguaje_de_programaci%C3%B3n))/CgFx y [DirectX](https://es.wikipedia.org/wiki/DirectX) [HLSL](https://es.wikipedia.org/wiki/HLSL) Effects (.Fx). Pueden escribirse shaders en tres formas distintas: como [Surface shaders](https://es.wikipedia.org/w/index.php?title=Surface_shaders&action=edit&redlink=1), como [Vertex](https://es.wikipedia.org/w/index.php?title=Vertex&action=edit&redlink=1) and [Fragment shaders](https://es.wikipedia.org/w/index.php?title=Fragment_shaders&action=edit&redlink=1), o como shaders de función fija. Un shader puede incluir múltiples variantes y una especificación declarativa de reserva, lo que permite a Unity detectar la mejor variante para la tarjeta de vídeo actual y si no son compatibles, recurrir a un shader alternativo que puede sacrificar características para una mayor compatibilidad.

El soporte integrado para [Nvidia](https://es.wikipedia.org/wiki/Nvidia) (antes Ageia), el motor de física [PhysX](https://es.wikipedia.org/wiki/PhysX), (a partir de Unity 3.0) con soporte en tiempo real para mallas arbitrarias y sin piel, ray casts gruesos, y las capas de colisión.

El scripting viene a través de [Mono](https://es.wikipedia.org/wiki/MonoDevelop). El script se basa en Mono, la implementación de código abierto de [.NET Framework](https://es.wikipedia.org/wiki/.NET_Framework). Los programadores pueden utilizar UnityScript (un lenguaje personalizado inspirado en la sintaxis [ECMAScript](https://es.wikipedia.org/wiki/ECMAScript)), [C#](https://es.wikipedia.org/wiki/C) o [Boo](https://es.wikipedia.org/wiki/Boo_(programaci%C3%B3n)) (que tiene una sintaxis inspirada en [Python](https://es.wikipedia.org/wiki/Python)). A partir de la versión 3.0 añade una versión personalizada de [MonoDevelop](https://es.wikipedia.org/wiki/MonoDevelop) para la depuración de scripts.

Esta última parte es importante debido a que C# fue uno de los lenguajes implementados en el curso, a continuación se explicará lo que es éste lenguaje.

C#

Es un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) [orientado a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos) desarrollado y estandarizado por [Microsoft](https://es.wikipedia.org/wiki/Microsoft) como parte de su plataforma [.NET](https://es.wikipedia.org/wiki/Microsoft_.NET), que después fue aprobado como un estándar por la [ECMA](https://es.wikipedia.org/wiki/Ecma_International). C# es uno de los lenguajes de programación diseñados para la [infraestructura de lenguaje común](https://es.wikipedia.org/wiki/Infraestructura_de_lenguaje_com%C3%BAn).

Su sintaxis básica deriva de [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n))/[C++](https://es.wikipedia.org/wiki/C%2B%2B) y utiliza el modelo de objetos de la plataforma .NET, similar al de [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)), aunque incluye mejoras derivadas de otros lenguajes.

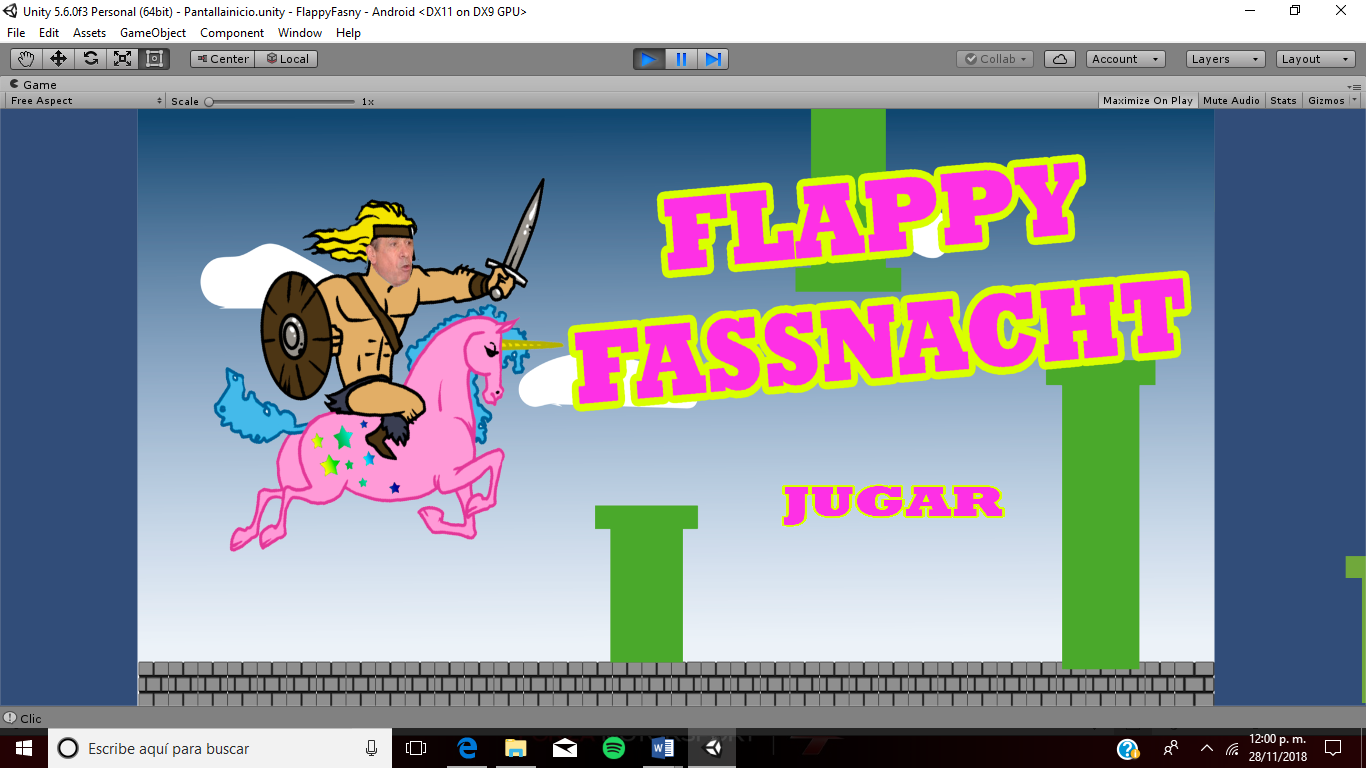
El nombre C Sharp fue inspirado por el signo '#' que se compone de cuatro signos '+' pegados.

Aunque C# forma parte de la plataforma .NET, ésta es una [API](https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones), mientras que C# es un lenguaje de programación independiente diseñado para generar programas sobre dicha plataforma. Ya existe un [compilador](https://es.wikipedia.org/wiki/Compilador) implementado que provee el marco [Mono](https://es.wikipedia.org/wiki/Proyecto_Mono) - [DotGNU](https://es.wikipedia.org/wiki/DotGNU), el cual genera programas para distintas plataformas como [Windows Microsoft](https://es.wikipedia.org/wiki/Microsoft_Windows), [Unix](https://es.wikipedia.org/wiki/Unix), [Android](https://es.wikipedia.org/wiki/Android), [iOS](https://es.wikipedia.org/wiki/IOS_(sistema_operativo)), [Windows Phone](https://es.wikipedia.org/wiki/Windows_Phone), [Mac OS](https://es.wikipedia.org/wiki/Mac_OS) y [GNU/Linux](https://es.wikipedia.org/wiki/GNU/Linux).

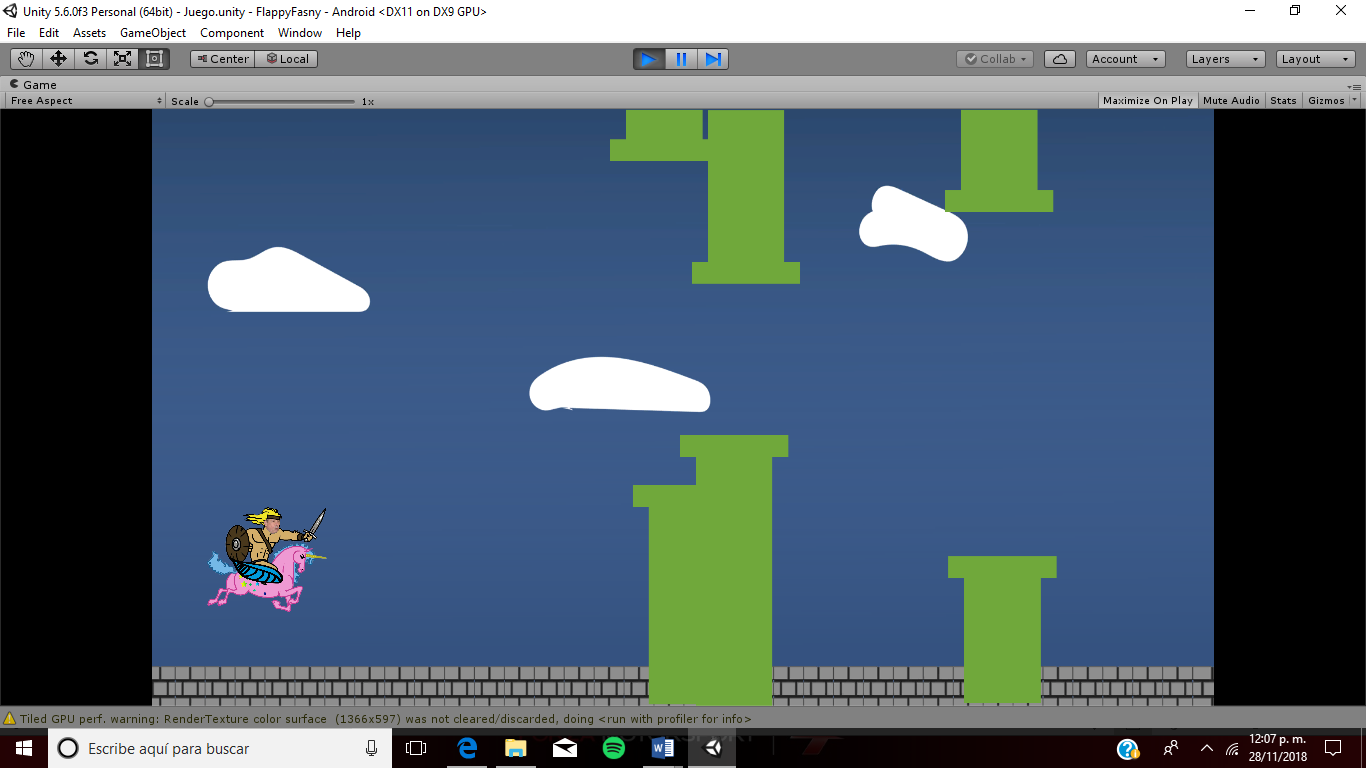
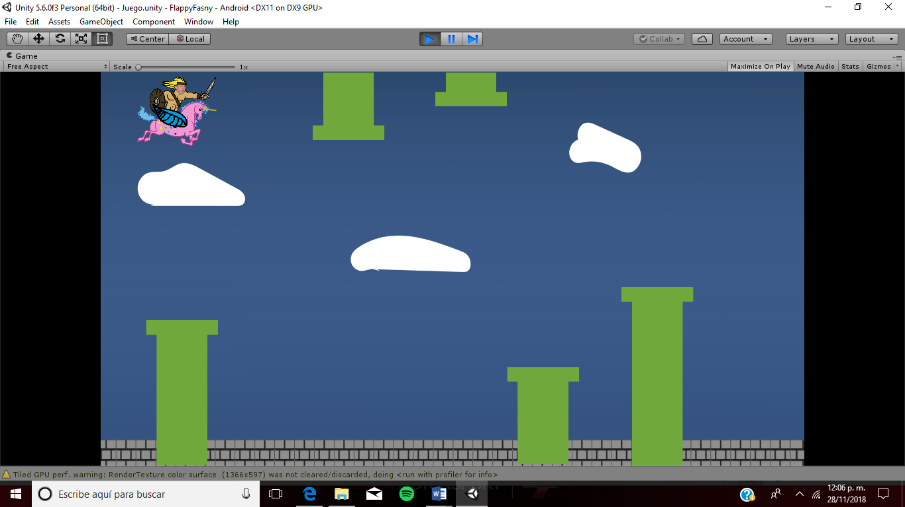
**Desarrollo**

El juego se realizó bajo la plataforma de Unity y utilizando el lenguaje de programación C#, el título del videojuego es “Flappy Fasny”, es un juego en 2D donde se tiene el objetivo de esquivar obstáculos variados que apareces de forma aleatoria y sobrevivir el mayor tiempo posible.

La siguiente imagen muestra la pantalla de inicio, el juego comenzará en cuanto el usuario realice un clic en el botón “JUGAR”.



El personaje se mueve en el eje X a una velocidad ya establecida y el movimiento en el eje Y se permite cambiar utilizando un “clic” haciendo que el personaje simule un salto, la distancia que salta el personaje por cada “clic” que da el usuario ya viene establecida, es posible dar muchos “clics” consecutivos para ganar altura pero en cuanto el usuario deje de presionar dicho botón, el personaje caerá debido a que se programo una fuerza de gravedad que va a afectar solo al personaje.

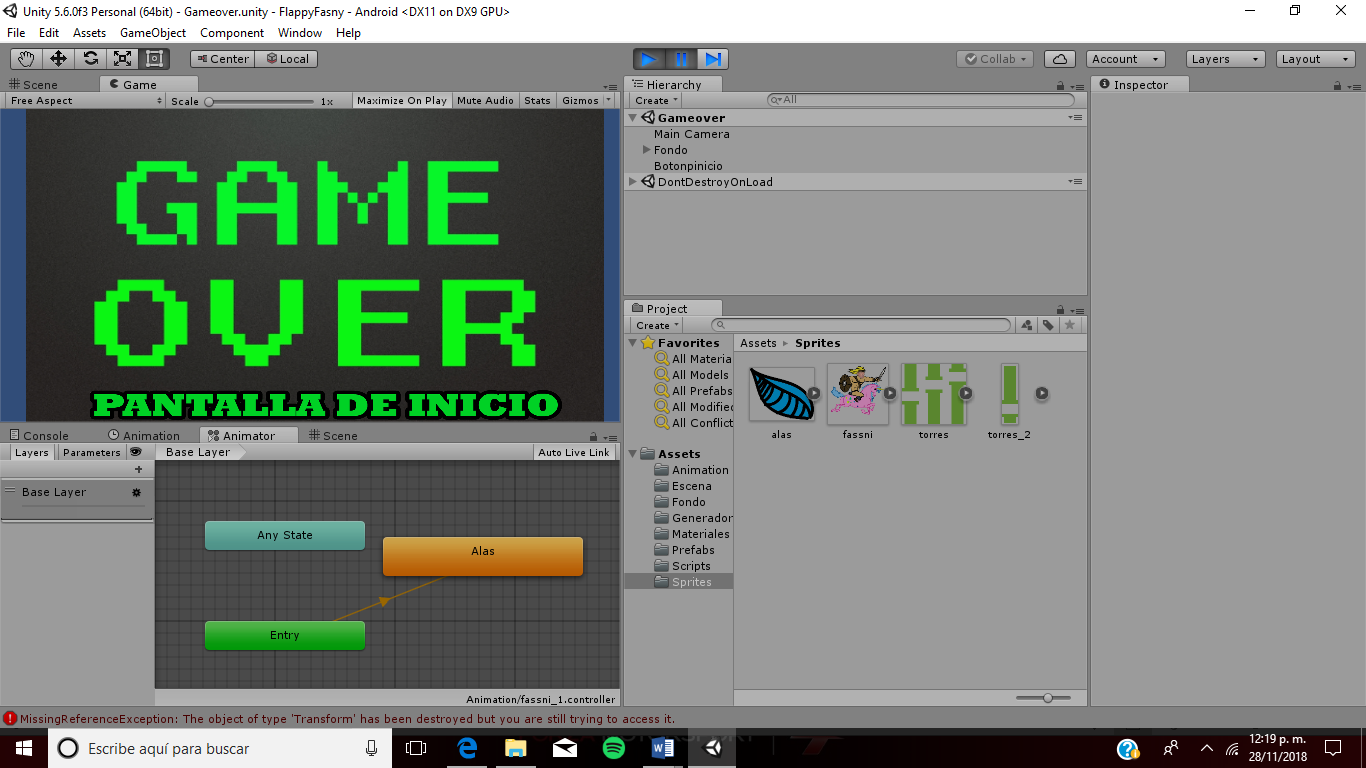
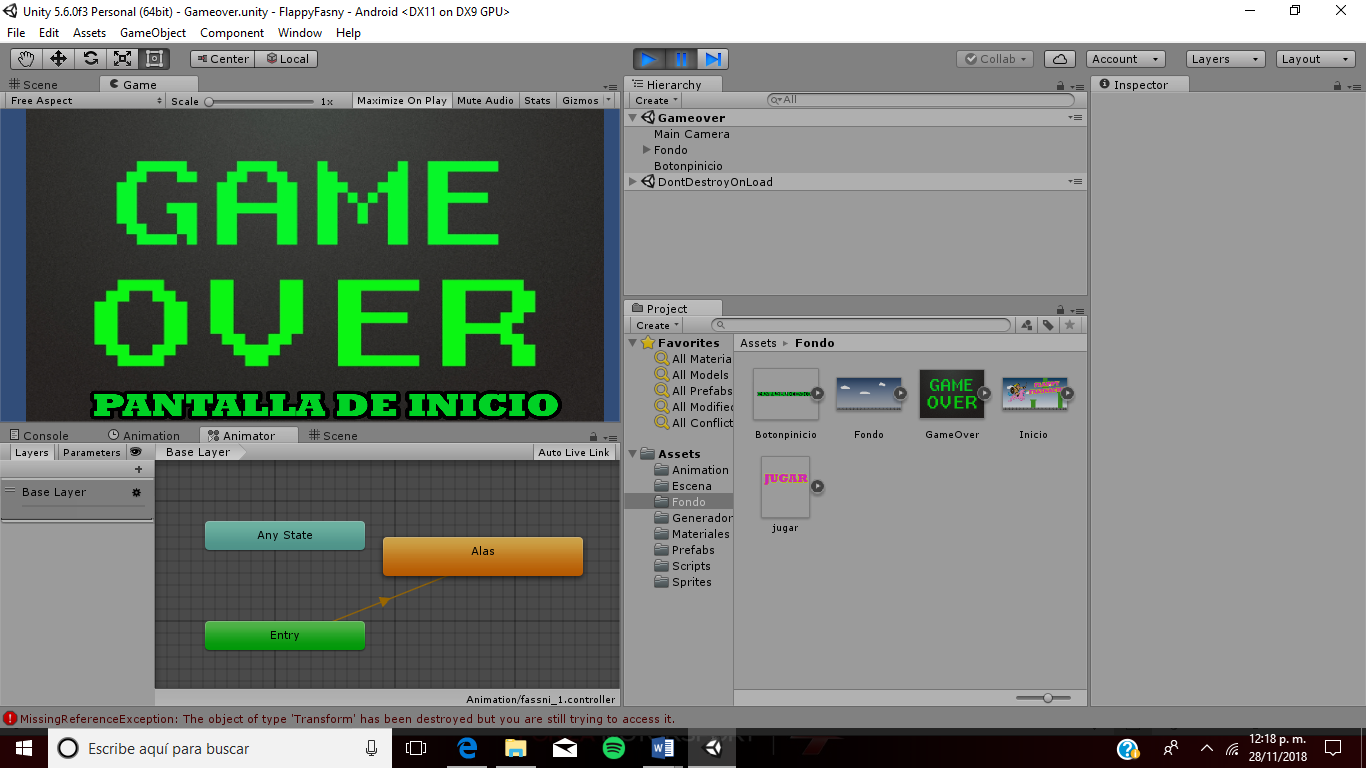


El juego continuará mientras el personaje no impacte con ninguno de los obstáculos, si pasa lo contrario el usuario habrá perdido y será enviado a la siguiente ventana.

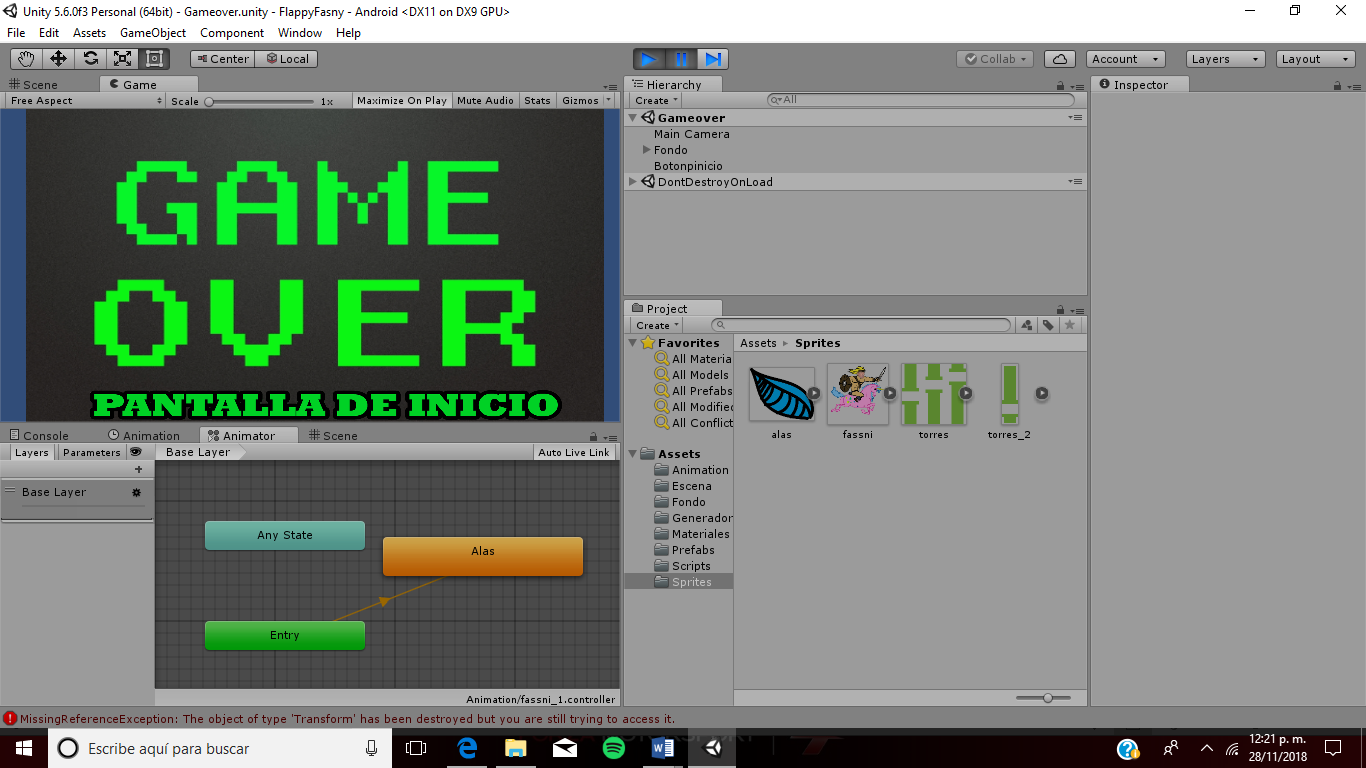


Si el usuario pierde, podrá volver a jugar dando clic en el botón de pantalla de inicio, que lo direccionará a dicha pantalla y ahí podrá presionar nuevamente el botón jugar.

Para realizar por completo este juego no solo se realizó programación, también se tuvieron que diseñar distintos elementos como el personaje, los fondos, obstáculos y botones.



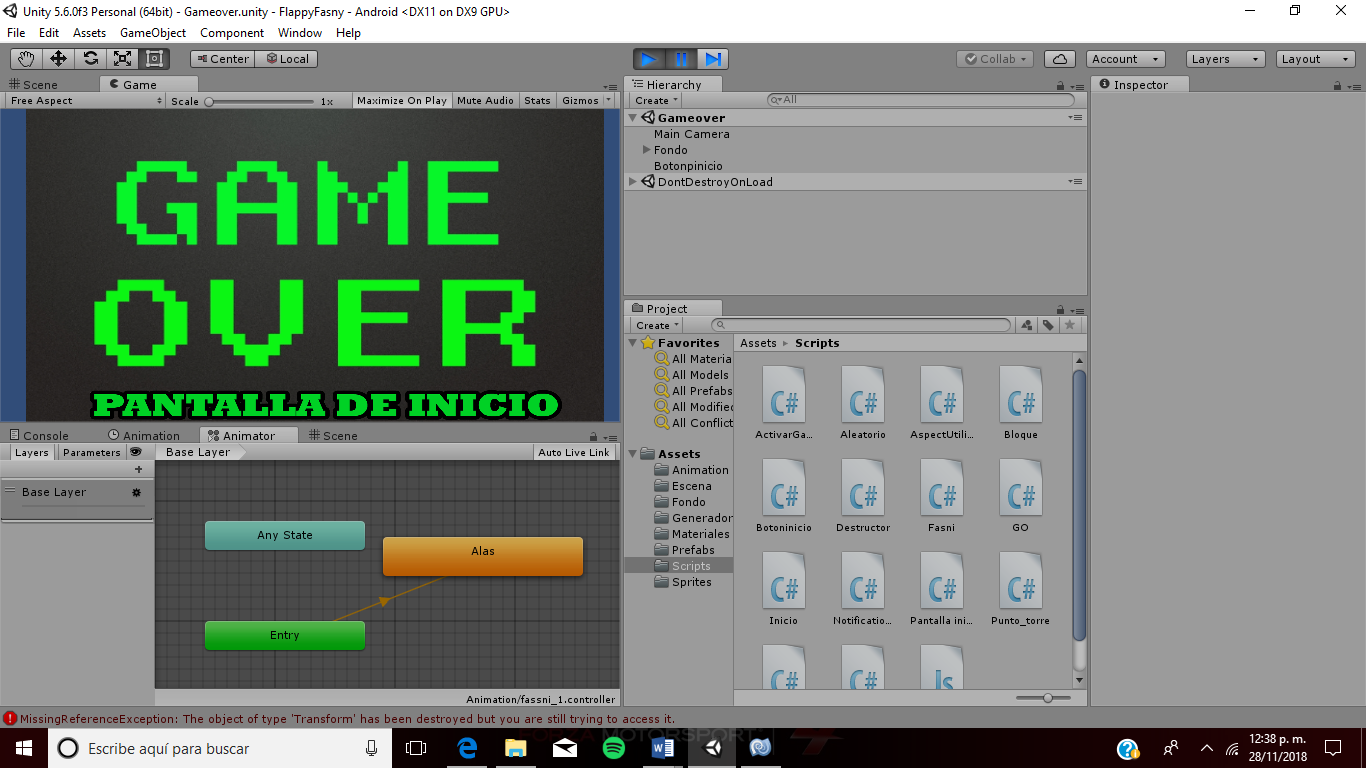
También se realizó una animación para el personaje, este movimiento se puede apreciar mientras el usuario se encuentre jugando, la programación de dicha animación se realizó de la siguiente forma:



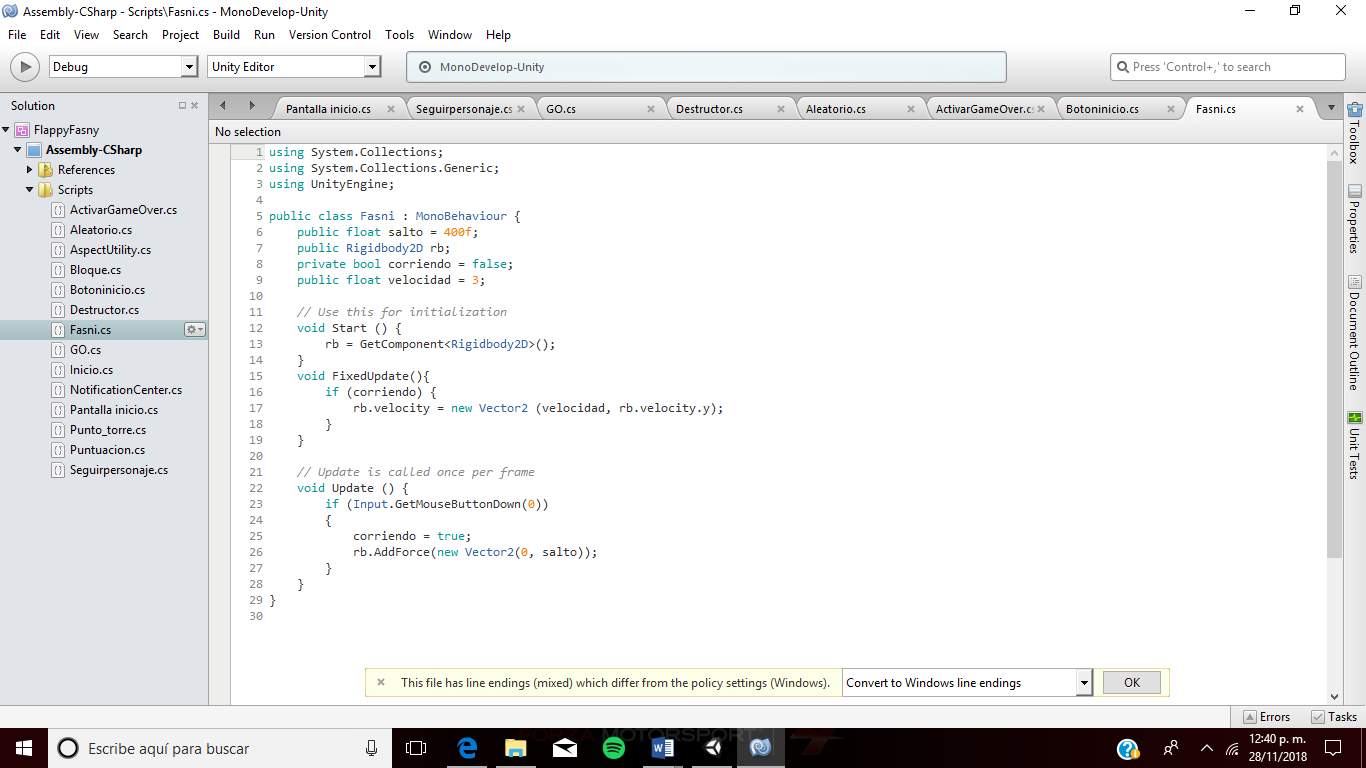
También se programaron coliders destructores, que avanzan conforme va avanzando el personaje, lo que realiza es que destruye todo lo que va haciendo contacto con el para de esta forma ir liberando espacio en la memoria.

Temas de clase puestos en práctica

Se utilizaron la mayoría de los temas de la unidad 1 y de la unidad 2, de la primera se utilizaron temas como clases, objetos, atributos, herencia, entre otras cosas que no vienen especificadas en el programa del curso, pero fueron utilizadas durante las prácticas de clase, todos estos temas fueron utilizados gracias a que se hizo una programación orientada a objetos.



Todos los programas se realizaron en el compilador de Unity



De la unidad dos se utilizaron los siguientes temas, ambiente de desarrollo, este ambiente fue creado completamente en Unity, áreas de trabajo y menús es otro tema, esto se puede apreciar en las ventanas para iniciar el juego y en la de fin de juego, también en estas ventanas se aprecian botones, que son otro tema llamado controles básicos, el último tema empleado es el de eventos, ya que se utiliza el clic para utilizar todo el juego.

**Conclusiones**

Para la elaboración de programas es muy importante la orientación a objetos, es una de las bases de la programación, así como la realización de la interfaz.

Realizar este proyecto ayudó mucho para reafirmar los conocimientos adquiridos durante las sesiones de clase de Programación Avanzada, ya que se vio una aplicación real y conforme se iba avanzando en el proyecto se fueron observando los diversos temas del curso, tal vez al comienzo no se pensaba que se utilizarían ciertos temas, pero con el paso de la construcción se fueron observando y asociando.

Este curso nos mostró un panorama diferente de lo que es la programación y gracias a eso logramos realizar este proyecto.