**DECLARACIÓN DE CLASES**

*Ricardo José Portnoy Tobías*

*Luis Alberto Ardila Páez*

1. **INTRODUCCIÓN**

En el presente documento se explicará al detalle y se mostrará la declaración de las clases necesarias para la realización del proyecto final, videojuego de carreras en el espacio *Across Lighspeed*.

1. **DECLARACIÓN**

**II.I PERSONAJE**

La clase *PERSONAJE* va a ser la encargada de modelar el comportamiento del piloto controlado por el usuario, asimismo, va a ser la clase padre de *ENEMIGOS*, la cual va a aprovechar la gran mayoría de atributos y funciones de esta.

Como se menciona anteriormente, *PERSONAJE* nos permitirá definir y gestionar las diferentes características de cada individuo en el juego, con atributos enteros como Vida, Fricción, Distancia de salto, Velocidad y Aceleración, encargados de definir como se van a comportar cada una de las funciones principales de las naves en el juego planteado. Además, esta clase heredará públicamente los distintos métodos de la clase padre *QGraphicsRectItem*, que nos ayudarán a determinar la posición y la *hitbox* de los objetos en todo momento.

Personaje posee gran cantidad de métodos en los cuales encontramos:

Saltar: mediante la distancia de salto y la velocidad se calculará la posición en la cual el auto caerá para después hacer una animación de salto, colocándose en la posición calculada.

Acelerar: cuando el coche acelera, su velocidad varia por lo cual cambiará su posición en el mapa, a mayor velocidad, mayor el cambio de posición del vehículo, el cambio de velocidad está determinado por la fricción y la aceleración, también producirá una animación de aceleración.

Frenar: al igual que acelerar este método cambiara la velocidad, para que esta disminuya progresivamente dependiendo de la fricción generando una actualización de su Sprite para que frene.

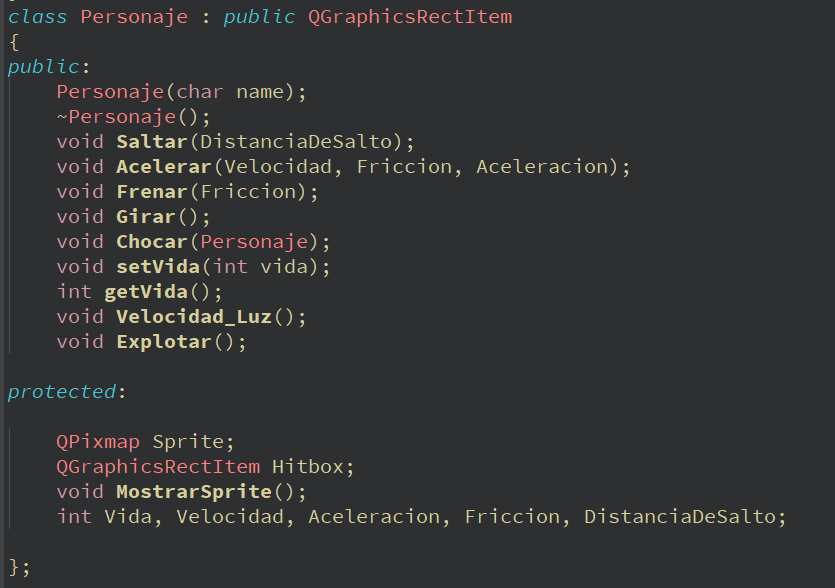
Girar: el vehículo cambia de dirección generando una animación de giro.

Chocar: el vehículo podrá chocar con otro personaje para ello se tienen en cuenta sus posiciones y el tamaño del Sprite de tal manera que se puede calcular sí hay una colisión, en el caso de que se presente este fenómeno la velocidad se vera afectada.

Velocidad Luz: en el momento que el vehículo obtenga una velocidad dada, realizara una animación y su velocidad aumentara exponencialmente a coste que su vida disminuya al mismo ritmo con el que acelera.

Explotar: en el momento que la vida del personaje llegue a valores negativos este explotara y todas sus demás variables se volverán 0, a su vez, se visualizara una animación de explosión.

***Ilustración clase PERSONAJE***

****

**II.II *BACKGROUND***

Los objetos tipo *BACKGROUND* son los encargados de representar los fondos de cada uno de los niveles del videojuego. Al ser el fondo un apartado sumamente importante en el tipo de juego que se planea construir, poseer una clase que establezca los fondos y distintos métodos para interactuar con ellos es indispensable. *BACKGROUND* esta conformada por atributos como un *QImage* el cual contendrá la imagen del mapa del nivel, cuatro enteros, un par representará las dimensiones de la pantalla del juego y el otro par las dimensiones de la imagen del nivel, y un *QGraphicsScene* en donde se establecerá el fondo y sus distintas modificaciones. Los métodos de la clase corresponden *a INIT, TRANSFORMAR* y *ACTUALIZAR,* además de los distintos métodos que la clase heredara públicamente de la clase padre *QMainWidget.*

INIT: establece la escena del mapa en el *GraphicsView*

Transformar: dado el mapa ya establecido, este método se encargará de hacer los respectivos cambios a la imagen para dar la sensación de perspectiva.

Actualizar: constantemente cambiara la escena del *GraphicsView* para seguir los movimientos del jugador, es decir, cada vez que el jugador cambie de posición se actualizara la imagen para que la sensación de perspectiva sea constante.

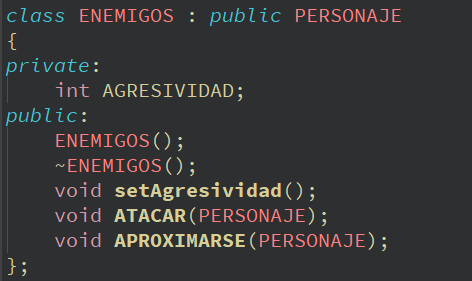
*****Ilustración clase BACKGROUND***

**II.III ENEMIGO**

Enemigo es la clase encargada de representar a los distintos adversarios del juego, esta clase heredara públicamente todos los atributos y métodos de *PERSONAJE* ya que, los enemigos al igual que el jugador están expuestos a las mismas características e interacciones como chocar, acelerar, frenar o derrapar. La principal cualidad de los objetos enemigo es su nivel de dificultad, a medida que avanza el juego los enemigos varían su comportamiento dependiendo del nivel de dificultad que estos posean, además que tienen ciertos métodos únicos como *ATACAR* y *APROXIMARSE*. Los enemigos podrán atacar o aproximarse a un personaje indicado.

Atacar: se necesita un personaje el cual atacar, de aquí calcularemos su posición y dicho objeto enemigo variara su velocidad para provocar un choque.

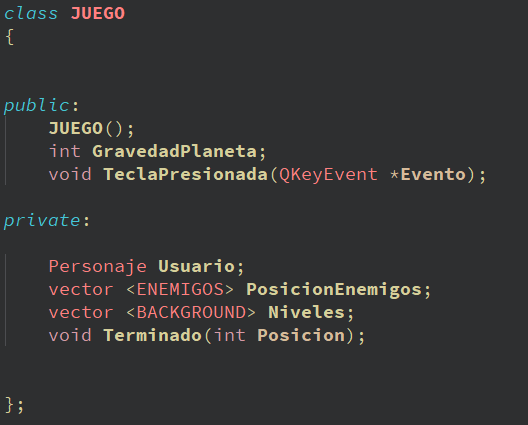
Aproximarse: este método permite que los enemigos no sean superados tan fácilmente, sí la distancia entre un personaje y dicho enemigo es considerablemente alta el enemigo cambiara su posición en el mapa para estar cerca del personaje al cual quiera aproximarse.

***Ilustración clase ENEMIGO***

**II.IV JUEGO**

La clase *JUEGO* es responsable de coordinar todos los elementos del juego, desde el control de las teclas presionadas por el jugador, hasta el manejo de los personajes, enemigos y escenarios. Proporciona una interfaz de interacción y encapsula la lógica interna necesaria para que el proyecto funcione correctamente, como GravedadPlaneta, variable esencial a la hora de inicializar cada uno de los niveles, ya que esta nos indicará las distintas interacciones que tendrá tanto el piloto controlado por el usuario como los enemigos; además de tener almacenadas en un vector cada una de las pistas que se mostrarán en el transcurso del videojuego.

***Ilustración clase JUEGO***

******