Física para videojuegos

ONLY LEAP

El juego que se ha creado se llamado **ONLY LEAP**. Como su propio nombre indica, el juego se basa en los saltos. En este, el jugador tendrá que impulsarse con las plataformas de color rojo hacia arriba. Su objetivo es no tocar el suelo nunca, lo cual se va complicando porque las plataformas van acelerando. Podrá moverse de derecha a izquierda con las flechas y si justo presiona la tecla espacio en el momento en el que toca la plataforma, su impulso se incrementará en un 40%.

Para la programación del juego, se han creado dos clases aparte de la principal que da el programa processing (el .pde). En la clase principal, se usaron los dos métodos necesarios para la ejecución (**draw()** y **setup()**), además de 8 propios. Estos son: **menu()**, **game()**, **end()**, **reset()**, **setMap()**, **cargarPuntuacion()**, **filePrint()** y **showText()**. Todos los métodos del programa se encuentran explicados en los comentarios de este, pero también están explicados en el documento llamado: "DOCUMENTACIÓN.txt" (adjuntado con el juego). Aún así, los más importantes de la clase principal son el "**menu()**", el "**game()**" y el "**end()**". Cuando se ejecuta por primera vez el juego, la variable booleana "lose" está en false y "menu" está en true. Así cuando inicia va a empezar mostrando el menu, ejecutándose "**menu()**". Si se presiona el botón del menú, "menu" se va a poner en false, y va a empezar el juego con "**game()**". Si llega a fallar, cayéndose al suelo, "lose" se va a poner en true y va a ejecutarse "**end()**". En esta última se muestra la pantalla de que ha perdido, y el botón de volver a intentarlo. Si le da al botón se activa el método "**reset()**", con el que todas las variables se inicializan de nuevo y "lose" vuelve a ser false. Para que todo esto funcione, se ha creado dos clases aparte: **Personaje** y **Plataforma**. En la clase principal, se va a crear un objeto llamado "personaje" que va a tener la clase "Personaje", en la que en el propio constructor ya le pones una posición, velocidad y aceleración inicial. En esta está el método de "**dibujarPersonaje()**" que, como su propio nombre indica, muestra al personaje en la pantalla, además de inicializar la posición de los colliders para la detección de colisiones. También están los métodos de "**collisionImpulse()**" (retorna una nueva velocidad, que será el impulso), "**getImpulse()**" (retorna la magnitud de impulso usada en el anterior método), "**move()**" (cambia la velocidad del objeto/personaje para moverse con las flechas), "**getCoyoteTime()**" (retorna un booleano dependiendo de si los colliders están en la zona en la que puede dar un salto más

grande) y "**lose()**" (retorna un booleano cuando se pierde, es decir, toca el suelo). Por otro lado, la clase "Plataforma" se la ha asignado a los objetos que serán las plataformas. Esta tiene su constructor y sus métodos. Estos son: "**dibujarPlataforma()**" (dibuja ese objeto/plataforma), "**reset()**" (si llega a tocar el suelo su posición cambia al punto más alto) y "**Up()**" (si el personaje llega a tocar el punto más alto, su posición no cambia, aunque su velocidad sí, y las plataformas aceleran más). El mapa es generado pseudoaleatoriamente con la función de la clase "**PApplet**" llamado "**random()**". Las fuerzas aplicadas son el peso al personaje, y en su velocidad se ha puesto un impulso calculado con las velocidades relativas. Como el impulso es siempre en la dirección del eje Y negativo, el cálculo es: $\widehat{v} = (v_x, \frac{-\lambda}{m_a})$ (este cálculo está reducido de la expresión: $\widehat{p} = \widehat{p_0} + \widehat{j}$, en donde "j" es el vector de impulso). En este caso, como no se requería de la velocidad en el instante anterior (una proposición propia para que quede mejor el salto) se ha suprimido el momento lineal en el instante anterior, y solo se ha tenido en cuenta el vector de impulso. Este impulso se ha aplicado cuando se detectaba una colisión por parte de los colliders (con sus posiciones), y para el salto más grande, con la tecla espacio, se le ha aplicado el mismo impulso pero sumado a un 40% más.

En conclusión, en este juego de saltos el personaje no para de impulsarse con las plataformas hacia arriba. Cada vez que una plataforma es borrada del mapa, se pone otra distinta en el punto más alto y se consigue un punto. El sistema de físicas se centra en las colisiones y el impulso generado por el personaje, además de otros añadidos como el peso o funciones cuadráticas matemáticas utilizadas en la animación del texto de puntuación. Todas las clases y métodos (excepto los principales de processing y de java) utilizados han sido de creación propia desde cero.