|  |
| --- |
| Palacios Limón, Guillermo Barchín Rubio, Álvaro Rodriguez Jañez, Eloy Matellanes Ramón, Miguel Pérez López, Xabier 2018-2019 |

|  |
| --- |
| Tecnología de Videojuegos |
| BEYOND SOFTWARE |
| Documento Técnico |



# Introducción

A continuación se explicarán las distintas clases que se han creado a lo largo del videojuego, así como las funciones cuyo funcionamiento no sea evidente.

Para facilitar la explicación y con el objetivo de mantener un orden, se catalogarán las clases en varios bloques:

* ***Historia***
* ***Introducción***
* ***Niveles***
* ***Jugador***
* ***Jefes finales***
* ***Enemigos***
* ***Objetos***

# Clases para la Historia y para la Introducción

## Historia1. java, Historia2. java, historia3. java, Historia4. java

### Función

El juego está formado por varios niveles entre los cuales se mostrará en pantalla una imagen en la que aparecerá una frase relacionada con el juego y, en alguna de ellas, algún consejo para el jugador

### Atributos

El único atributo que será necesario será el atributo *image*, que contendrá la imagen que se debe mostrar por pantalla entre cada nivel.

Será inicializada dentro de la función ***public void init();***

### Métodos

Los métodos pertenecientes a estas clases son los que se enumeran a continuación:

* ***public int getID():*** devuelve el número que identifica a cada estado para poder realizar la transición de éstos.
* ***public void init():*** inicializa los atributos que posea la clase. En este caso solo se inicializa el atributo *image*.
* ***public void render():*** esta función está destinada a imprimir texto e imágenes por pantalla. Dentro de esta función se llamará a las funciones *g.drawImage();* y *g.drawString();*
* ***public void update():*** puesto que la única funcionalidad de estas clases es mostrar una imagen por pantalla, cuando el jugador presione la tecla *ENTER*, se pasará al siguiente estado, que será el siguiente nivel según corresponda. Esta función será implementada dentro de este método.

## Historia5. java, Historia6. java, historia7. java, Historia8. java, Historia9. java

### Función

Una vez derrotemos al jefe final, estas clases, funcionando a modo de estados, darán una última explicación de la historia al jugador. Los atributos y los métodos serán los mismos que en las clases anteriores.

### Atributos

*Véase 2.1.2 Atributos*

### Métodos

*Véase 2.1.3 Métodos*

## Logo. java y Logo\_juego. java

### Función

La función de estas clases es mostrar por pantalla el logotipo del equipo de desarrollo (Logo.java) y el logo del videojuego (Logo\_juego.java), antes de mostrar el menú principal del videojuego.  
Los atributos y los métodos serán los mismos que en los casos anteriores con la salvedad de que se añade, dentro de *Logo\_juego.java* el atributo *historia*, de tipo *Music*

### Atributos

Además de los atributos ya mencionados (*Véase 2.1.2 Atributos*) la clase *Logo\_juego.java* incluye el atributo *private Music historia*.

### Métodos

Todos los atributos mantienen su función (*Véase 2.1.3 Métodos*) salvo el método ***public void init()*** que en que este caso también inicializa el atributo *historia* y es cuando pulsamos *ENTER* dentro de la función ***public void update();*** cuando comienza a reproducirse con la instrucción *historia.loop();*

## Menu. java

### Función

Esta clase mostrará por pantalla un fondo con 3 opciones que podrá elegir el usuario para comenzar una nueva partida, cargar partida o salir del juego.  
Al pulsar nueva partida, se mostrará la introducción del videojuego.

### Atributos

* ***private int selected:*** opción elegida por el usuario.
* ***private String[] options = new String[]{“ ”,” ”,” ”}:*** variable tipo string que almacena las opciones que puede pulsar el jugador. En este caso aparecerán vacías porque las opciones están impresas sobre la imagen de fondo.
* ***private StateBasedGame game***
* ***private Image image***
* ***private Sound sound:*** almacena el sonido que debe reproducirse cuando el jugador se desplace entre las distintas opciones.

### Métodos

* ***public int getID();***
* ***public void init():*** dentro se inicializan los atributos *image* y *sound*.
* ***public void render():*** imprime la imagen de fondo y dibuja el recuadro que se superpone a la opción que esté seleccionando el jugador en cada momento.
* ***public void update():*** dentro contiene una máquina de estados implementada con un switch-case. Con el primer estado se iniciará una nueva partida, con el segundo estado se cargará la partida en función del nivel en el que nos hayamos quedado y con el último estado el jugador saldría del juego.
* ***public void keyReleased():*** con esta función, pulsando las flechas arriba o abajo podemos desplazarnos entre las tres opciones disponibles.

## Intro1. java, Intro2. java, Intro3. java, Intro4. java, Intro5. java, Intro6. java, Intro7. java

### Función

Dan al jugador una breve introducción de la historia para ponerlo en contexto. Puesto que puede ser que no sea la primera vez que el jugador comienza el juego, se ha decidido dar la opción de que pulsando el botón *ESPACIO*, el jugador podrá saltar toda la introducción y saltar

### Atributos

Se añade a los atributos anteriores (*Véase 2.1.2 Atributos*) el atributo *nivel1*, de tipo *Music*.

### Métodos

Dentro del método ***public void update*** al pulsar *ESPACIO* (o *ENTER también en la clase Intro7.java)* se comenzará a reproducir la música del primer nivel.

# Niveles