Trabajo práctico “*Castlevania, Barbarianna Viking Edition*”

* Integrantes:
  + Franco Hernán Benitez
    - Email: francobentiez980@gmail.com
  + Máximo De Michieli
    - Email:
  + Gastón Rampazzo
    - Email:
* Materia: **Programación I**
* Comisión: **04**
* Año: **2021**

**Índice**

*Introducción ---------------------------------------------------------------------------------------------------------- 1*

*Descripción ----------------------------------------------------------------------------------------------------------- 2*

*Implementación ----------------------------------------------------------------------------------------------------- 3*

*Conclusiones --------------------------------------------------------------------------------------------------------- 4*

**Introducción**

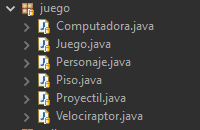
En este trabajo se llevó acabo el desarrollo de un videojuego denominado “Castlevania, Barbarianna Viking Edition”, que consiste en permitirle a Barbariana recuperar la computadora Commodore 128kb de hackeo del tiempo que le pertenecía a Hackerman y fue robada por el malvado doctor Hero. Para esto, el juego se desarrolla en un mapa que simula ser un castillo con varios pisos. En donde los velocirraptors que van a tratar de impedir que Barbariana llegue hacia la computadora, atacaran a esta, hasta tratar de eliminarla del juego. Cada Velocirraptor con tan solo tocar a Barbariana, logran eliminarla. Además, cuentan con la posibilidad de disparar rayos laser que dañan a la heroína y la dejan fuera de juego.

Por su parte, Barbariana cuenta con el martillo de Thor (el dios del trueno) permitiéndole dispararle a los dinos cuando se ve acorralada por estos mismos. Para evitar los disparos de los velocirraptors, puede saltar o agacharse.

El desarrollo esta hecho en Java, construido por distantes clases que contienen la funcionalidad para cada elemento de la aplicación. Además, cuenta con una librería externa que levanta un entorno en donde ofrece herramientas para dibujar determinadas formas en pantalla, ubicar elementos, etc.

**Descripción**

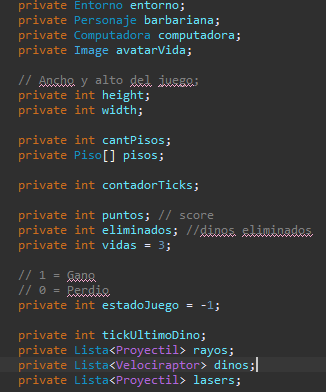
Para el desarrollo del juego, se llevaron a cabo las siguientes clases:



***Clase Juego***

Es la clase principal del proyecto. En esta clase, se crea una instancia del entorno que permite levantar un entorno grafico para que el juego se ejecute, brinda las herramientas necesarias (métodos) para dibujar los elementos en pantalla e ir actualizando constantemente el estado de los mismos.

La clase cuenta con los siguientes atributos:



* entorno: instancia del objecto Entorno
* Barbariana: instancia del objeto Personaje (personaje principal)
* Computadora: instancia del objeto Computadora que hace referencia a la computadora 128kb de Hackerman.
* avatarVida: imagen que va tener el icono de vidas
* heigth & width: Dimensiones de la ventana del juego
* cantPisos: Cantidad de pisos que tiene el juego
* Pisos: Listado que contiene cada objeto de piso
* contadorTicks:
* puntos: Acumulador de score durante el juego
* eliminados: Contador de velocirraptors eliminados
* vidas: Vidas que le quedan a Barbariana
* estadoJuego: tiene 2 estados durante el juego. 1 si gano o 0 si perdió. Permite mostrar un mensaje en especifico en pantalla cuando el juego termina.
* tickUltimoDino:
* rayos: Contiene un listado de los rayos que son disparados
* dinos: Contiene un listado de los velocirraptors que hay en pantalla
* lasers: listado de lasers que son disparados

Métodos:

* Juego(): Constructor de la clase que instancia los objetos necesarios para el correcto funcionamiento del juego.
* Tick(): Actualiza el estado del personaje, los dinos, sus disparos y láser. Además, chequea colisiones y en el caso de que se deba eliminar un objeto, lo hace.
* \_actualizarMovimientos(): Se encarga de mover al personaje según una tecla presionada y también de disparar cuando sea necesario
* \_inicializarPisos(): Crea los pisos necesarios para el juego y los guarda en la variable de instancia “pisos”
* \_dibujarPisos(): Itera cada instancia de piso que hay almacena en la variable de instancia “pisos” y los dibuja en el mapa
* \_dibujarDinos(): Dibuja el velocirraptor en pantalla chequeando que el ultimo dibujado haya bajado al siguiente piso.
* \_verificarImpactoDinoAPersonaje(): Verifica si el dino impacta sobre el personaje
* \_verificarImpactoLaserAPersonaje(): Cuando se efecuta un disparo, esta función se encarga de verificar si colisiona con barbariana.
* verificarImpactoADino(): Verifica si el personaje impacta su relámpago sobre el velocirraptor
* verificarImpactoALaser: Verifica si el personaje impacta su relámpago sobre el rayo del dino.
* moverRayo(): Mueve el rayo en la pantalla y lo elimina si sale de la pantalla.
* moverLasers(): Itera la lista de láser y en cada iteración llama a mueveLaser() pasandole como parámetro la instancia de laser correspondiente a la iteración.
* moverLaser(laser): Mueve el láser en la pantalla y lo elimina si sale de la pantalla.
* procesarRayos(): itera el listado de rayos y en cada función ejecuta el método **procesarRayoFunc().**
* procesarRayoFunc(rayo): mueve el rayo y verifica si algún elemento lo impacta
* verificarImpactoALaser(rayo): Verifica si es impacto el rayo. En caso afirmativo, suma puntos al usuario y elimina el rayo.
* moverDino(dino): Recibe un dino y lo mueve a su dirección correspondiente. Ademas, lo hace descender de piso si es necesario y lo elimina si sale de pantalla.
* procesarDinos(): itera la lista de dinos y llama a **procesarDinoFunc()**
* procesarDinoFunc(dino): Recibe un dino, lo mueve hacia donde corresponde, verifica su impacto y hace que dispare.
* finDelJuego(message,color): Muestra un mensaje con un color determinado en la parte superior de la pantalla y después de medio segundo cierra el juego. Es usado cuando el usuario gana o pierde.