C. de Valdes Salas, 11
Tel. 98 510 27 96
eii@uniovi.es
33007 Oviedo

INFORME - AMPLIACIÓN NAVES

Guillermo Facundo Colunga (UO236856).

PROFESORADO:

Jordán Pascual Espada - pascualjordan@uniovi.es

Resumen. En este documento se recoge un informe con las ampliaciones realizadas sobre el juego de naves visto en clase. Dicho informe se compone del número de ampliación, una breve descripción y aquellos fragmentos de código más representativos de cada ampliación.

Palabras clave: java script, videojuegos, software de entretenimiento.

1. Lista de ampliaciones realizadas

A continuación se detallan las ampliaciones realizadas:

ID	Nombre	Descripción
2	Enemigos con capacidad de disparo.	Se añade a los enemigos la capacidad de disparar, cada disparo resta una vida al jugador.
3	Jugador con vida.	El jugador tiene 3 vidas al principio, cada colisión o disparo que impacte en el enemigo resta una vida.
4	Jugador con disparos finitos.	El jugador posee un número de disparos finitos, cada vez que dispara se resta uno de los disparos disponibles. También se añadirá un elemento de munición al juego que permita recargar en 10 unidades los disparos disponibles.
6	Monedas	En el juego aparece un nuevo elemento llamado monedas, cada vez que el jugador colisione contra una moneda esta sumará una vida, hasta un máximo de 3, y un punto.
10	Bombas	Otro elemento del juego serán las bombas, permiten eliminar todos los enemigos y sus disparos de la pantalla actual. Al usar las bombas no se sumarán puntos por los enemigos eliminados.

2. Enemigos con capacidad de disparo

GameLayer.js

```
. . .
this.disparosEnemigos = [];
. . .
// DisparoEnemigo - Jugador
for (var i = 0; i < this.disparosEnemigos.length; i++) {
      if (this.jugador.colisiona(this.disparosEnemigos[i])) {
            if (this.vidas.valor == 1) {
                  this.iniciar();
            } else {
                  this.vidas.valor--;
                  this.disparosEnemigos.splice(i, 1);
            }
      }
}
. . .
// Actualizando disparos de enemigos.
for (var i = 0; i < this.disparosEnemigos.length; i++) {
      this.disparosEnemigos[i].actualizar();
}
. . .
// Generando los disparos de ls enemigos.
for (var i = 0; i < this.enemigos.length; i++) {
      var nuevoDisparo = this.enemigos[i].disparar();
      if (nuevoDisparo != null) {
            this.disparosEnemigos.push(nuevoDisparo);
      }
}
. . .
// Dibujando disparos de enemigos.
// Dibujando disparos de enemigos.
for (var i = 0; i < this.disparosEnemigos.length; i++) {
      this.disparosEnemigos[i].dibujar();
}
```

DisparoEnemigo.js

```
class DisparoEnemigo extends Modelo {

    constructor(x, y) {
        super(imagenes.disparo_enemigo, x, y);
        this.vx = -2;
    }

    actualizar() {
        this.x = this.x + this.vx;
    }
}
```

Enemigo.js

```
. . .
this.cadenciaDisparo = 200;
this.tiempoDisparo = 0;
. . .
disparar() {
     if (this.tiempoDisparo == 0) {
          // reiniciar Cadencia
          this.tiempoDisparo = this.cadenciaDisparo;
          return new DisparoEnemigo(this.x, this.y);
     } else {
          return null;
     }
}
. . .
```

3. Jugador con vida

GameLayer.js

```
. . .
// Creando el icono de vidas.
this.fondoVidas = new Fondo(imagenes.corazon_pequenio, 480 * 0.05, 320 * 0.05);
this.vidas = new Texto(3, 480 * 0.1, 320 * 0.07);
. . .
this.fondoVidas.dibujar();
this.vidas.dibujar();
. . .
if (this.vidas.valor == 1) {
     this.iniciar();
} else {
     this.vidas.valor--;
     this.enemigos.splice(i, 1);
}
. . .
if (this.vidas.valor < 3) {
     this.vidas.valor++;
}
. . .
```

4. Jugador con disparos finitos

GameLayer.js

```
. . .
// Creando el icono de disparos disponibles.
this.fondoDisparos = new Fondo(imagenes.bala, 480 * 0.65, 320 * 0.05);
this.disparos = new Texto(10, 480 * 0.7, 320 * 0.07);
. . .
this.powerups = [];
. . .
// Actualizando powerups.
for (var i = 0; i < this.powerups.length; i++) {
     this.powerups[i].actualizar();
}
. . .
// PowerUp - Jugador
for (var i = 0; i < this.powerups.length; i++) {
     if (this.jugador.colisiona(this.powerups[i])) {
          this.disparos.valor += 10;
          this.powerups.splice(i, 1);
     }
}
. . .
for (var i = 0; i < this.powerups.length; i++) {
     this.powerups[i].dibujar();
}
. . .
this.fondoDisparos.dibujar();
this.disparos.dibujar();
. . .
if (controles.disparo) {
     if (this.disparos.valor > 0) { // AMPLIACION 4 - JUGADOR CON DISPAROS FINITOS.
          var nuevoDisparo = this.jugador.disparar();
          if (nuevoDisparo != null) {
               this.disparosJugador.push(nuevoDisparo);
               this.disparos.valor–;
          }
     }
}
. . .
```

PowerUps.js

```
class PowerUp extends Modelo {

     constructor(x, y) {
          super(imagenes.bala, x, y);

          this.vy = 0;
          this.vx = 1;
     }

     actualizar() {
          this.vx = -1;
          this.x = this.x + this.vx;
     }
}
```

5. Monedas

GameLayer.js

```
. . .
this.monedas = [];
. . .
// Generando Monedas
if (this.iteracionesCrearMonedas > 300) {
      var rX = Math.random() * (600 - 500) + 500;
      var rY = Math.random() * (300 - 60) + 60;
      this.monedas.push(new Moneda(rX, rY));
      this.iteracionesCrearMonedas = 0;
}
. . .
// Actualizando monedas.
for (var i = 0; i < this.monedas.length; i++) {
      this.monedas[i].actualizar();
}
. . .
// Moneda - Jugador
for (var i = 0; i < this.monedas.length; i++) {
      if (this.jugador.colisiona(this.monedas[i])) {
            this.puntos.valor++;
            if (this.vidas.valor < 3) { // AMPLIACION 3 - VIDAS.
            this.vidas.valor++; // AMPLIACION 3 - VIDAS.
      }
            this.monedas.splice(i, 1);
      }
}
. . .
// Pintando las monedas.
for (var i = 0; i < this.monedas.length; i++) {
      this.monedas[i].dibujar();
```

```
}
. . .
```

Moneda.js

```
class Moneda extends Modelo {

    constructor(x, y) {
        super(imagenes.moneda, x, y);

        this.vy = 0;
        this.vx = 1;
    }

    actualizar() {
        this.vx = -1;
        this.x = this.x + this.vx;
    }
}
```

6. Bombas

GameLayer.js

```
. . .
this.bombas = [];
. . .
if (this.iteracionesCrearBombas > 300) {
     var rX = Math.random() * (600 - 500) + 500;
     var rY = Math.random() * (300 - 60) + 60;
     this.bombas.push(new Bomba(rX, rY));
     this.iteracionesCrearBombas = 0;
}
. . .
for (var i = 0; i < this.bombas.length; i++) {
     this.bombas[i].actualizar();
}
. . .
for (var i = 0; i < this.bombas.length; i++) {
     this.bombas[i].dibujar();
}
. . .
// Bomba - Jugador
for (var i = 0; i < this.bombas.length; i++) {
     if (this.jugador.colisiona(this.bombas[i])) {
          this.enemigos = [];
          this.bombas = [];
          this.disparosEnemigos = [];
     }
}
```

Bomba.js

```
class Bomba extends Modelo {

    constructor(x, y) {
        super(imagenes.bomba, x, y);

        this.vy = 0;
        this.vx = 1;
    }

    actualizar() {
        this.vx = -1;
        this.x = this.x + this.vx;
    }
}
```