Adrián Jiménez Villarreal

71904816P | uo227602

Videojuegos-Infinite

Ampliaciones

Contenido

[Ampliaciones 2](#_Toc468658677)

[Varias vidas 2](#_Toc468658678)

[Modificación del Jugador 2](#_Toc468658679)

[Modificación de los controles 2](#_Toc468658680)

[Modificación de la escena de juego 2](#_Toc468658681)

[Scroll en el eje Y 2](#_Toc468658682)

[Modificación de la escena de juego 3](#_Toc468658683)

[Capacidad de disparo 3](#_Toc468658684)

[Nuevo tipo de enemigo 3](#_Toc468658685)

[Turbo 3](#_Toc468658686)

[Pinchos 4](#_Toc468658687)

[Niveles 4](#_Toc468658688)

[Control el modo plataformas 4](#_Toc468658689)

# Ampliaciones

## Varias vidas

Para realizar esta ampliación primero deberemos modificar el objeto Jugador, posteriormente la capa de controles y finalmente la escena del juego.

### Modificación del Jugador

Añadir una variable vidas inicializada a 3 y la siguiente función.

|  |
| --- |
| }, restaVida: **function**(){  **this**.vidas--; } |

### Modificación de los controles

Habrá que crear una etiqueta para las vidas, e inicializarla de la siguiente manera.

|  |
| --- |
| *// Contador Vidas* **this**.etiquetaVidas = **new** cc.LabelTTF(**"Vidas: 3"**, **"Helvetica"**, 20); **this**.etiquetaVidas.setPosition(cc.p(size.width - size.width+90, size.height - 20)); **this**.etiquetaVidas.fillStyle = **new** cc.Color(0, 0, 0, 0); **this**.addChild(**this**.etiquetaVidas); |

También habrá que crear un método para actualizar esta etiqueta cada vez que pierda una vida el jugador.

|  |
| --- |
| },actualizarVidas:**function**(vidas){  **this**.etiquetaVidas.setString(**"Vidas: "** + vidas); } |

### Modificación de la escena de juego

Habrá que modificar la colisión del jugador con enemigos, para que pierda vidas cuando chocas con ellos.

|  |
| --- |
| ,collisionJugadorConEnemigo:**function** (arbiter, space) {  **var** capaControles = **this**.getParent().getChildByTag(idCapaControles);  **this**.jugador.restaVida();  **var** shapes = arbiter.getShapes();  **this**.formasEliminar.push(shapes[1]);  **if**(**this**.jugador.vidas<=0){  cc.director.pause();  cc.director.runScene(**new** GameOverLayer());  }  capaControles.actualizarVidas(**this**.jugador.vidas); } |

## Scroll en el eje Y

Para realizar esta ampliación simplemente deberemos modificar como se mueve la cámara, no solo en el eje x sino también en él y.

### Modificación de la escena de juego

En el método update se controlará la cámara de la siguiente manera, así seguirá al jugador tanto en el eje horizontal como el vertical.

|  |
| --- |
| *// Ampliacion Scroll eje Y* **var** posicionXCamara = **this**.jugador.body.p.x - **this**.getContentSize().width/2; **var** posicionYCamara = **this**.jugador.body.p.y - **this**.getContentSize().height/2;  **if** ( posicionXCamara < 0 ){  posicionXCamara = 0; } **if** ( posicionXCamara > **this**.mapaAncho - **this**.getContentSize().width ){  posicionXCamara = **this**.mapaAncho - **this**.getContentSize().width; }  **if** ( posicionYCamara < 0 ){  posicionYCamara = 0; } **if** ( posicionYCamara > **this**.mapaAlto - **this**.getContentSize().height ){  posicionYCamara = **this**.mapaAlto - **this**.getContentSize().height ; }  **this**.setPosition(cc.p( - posicionXCamara , - posicionYCamara)); |

## Capacidad de disparo

Para poder realizar esta ampliación necesitaremos crear una clase Disparo, modificar la capa de controles y añadir colisiones extra para los disparos con los enemigos y el suelo.

### Creación Disparo

|  |
| --- |
| **var** Disparo = cc.Class.extend({  space:**null**,  sprite:**null**,  body:**null**,  shape:**null**,  layer:**null**, ctor:**function** (space, posicion, layer) {  **this**.space = space;  **this**.layer = layer;   *// Crear animacion* **var** framesAnimacion = [];  **for** (**var** i = 1; i <= 4; i++) {  **var** str = **"disparo\_0"** + i + **".png"**;  **var** frame = cc.spriteFrameCache.getSpriteFrame(str);  framesAnimacion.push(frame);  }  **var** animacion = **new** cc.Animation(framesAnimacion, 0.2);  **var** actionAnimacionBucle =  **new** cc.RepeatForever(**new** cc.Animate(animacion));   *// Crear Sprite - Cuerpo y forma* **this**.sprite = **new** cc.PhysicsSprite(**"#disparo\_01.png"**);  *// Cuerpo estÃ¡tica , no le afectan las fuerzas* **this**.body = **new** cp.Body(5, Infinity);   **this**.body.setPos(posicion);  **this**.body.setAngle(0);  **this**.sprite.setBody(**this**.body);  *// Se inserta el cuerpo al espacio* **this**.space.addBody(**this**.body);   *// forma* **this**.shape = **new** cp.BoxShape(**this**.body,  **this**.sprite.getContentSize().width,  **this**.sprite.getContentSize().height);  *// agregar forma dinamica* **this**.shape.setSensor(**true**);  **this**.space.addShape(**this**.shape);  **this**.shape.setCollisionType(tipoDisparo);  *// aÃ±adir sprite a la capa   // ejecutar la animaciÃ³n* **this**.sprite.runAction(actionAnimacionBucle);   layer.addChild(**this**.sprite,10);    }, eliminar: **function** (){  *// quita la forma* **this**.space.removeShape(**this**.shape);   *// quita el cuerpo* **this**.space.removeBody(**this**.shape.getBody());   *// quita el sprite* **this**.layer.removeChild(**this**.sprite);  } }); |

### Modificación ControlesLayer

Habrá que añadir el botón y después controlar si se pulsa, además de añadir un tiempo entre disparo y disparo.

Insercion del botón.

|  |
| --- |
| *// BotonDisparo* **this**.spriteBotonDisparo = cc.Sprite.create(res.boton\_disparar\_png);  **this**.spriteBotonDisparo.setPosition(  cc.p(size.width\*0.8, size.height\*0.2));   **this**.addChild(**this**.spriteBotonDisparo); |

Control boton

|  |
| --- |
| **if** (cc.rectContainsPoint(areaDisparo,  cc.p(event.getLocationX(), event.getLocationY()))  && **new** Date().getTime() - instancia.tiempoDisparar > 1000 ){  instancia.tiempoDisparar = **new** Date().getTime();  **var** gameLayer = instancia.getParent().getChildByTag(idCapaJuego);  **var** disparo = **new** Disparo(gameLayer.space,  cc.p(gameLayer.jugador.body.p.x, gameLayer.jugador.body.p.y),  gameLayer);  disparo.body.vx = 600;  gameLayer.disparos.push(disparo); } |

### Modificación GameScene

Habrá que añadir las nuevas colisiones y tratar los objetos que se deben eliminar como se tratan los demás.

### Gestión nuevas colisiones

En la llamada se tratará después de la colisión no antes.

|  |
| --- |
| },colisionDisparoConEnemigo:**function** (arbiter, space) {  **var** shapes = arbiter.getShapes();   **this**.formasEliminar.push(shapes[1]);  **this**.formasEliminar.push(shapes[0]); },colisionDisparoConSuelo:**function** (arbiter, space) {  **var** shapes = arbiter.getShapes();   **this**.formasEliminar.push(shapes[0]); } |

## Nuevo tipo de enemigo

Para realizar esta ampliación se ha usado el mismo sprite del cuervo, simplemente que esta vez se mueve entre unas barreras y no puede pasar de ahí, además este dispara. Además, se ha modificado la clase disparo creada anteriormente para que se creen los disparos de tipo enemigo o jugador, para tratar las colisiones.

### Creación del nuevo enemigo

|  |
| --- |
| **var** NuevoEnemigo = cc.Class.extend({  direccion: **"derecha"**,  space:**null**,  sprite:**null**,  shape:**null**,  layer:**null**,  tiempoDisparo:0, ctor:**function** (space, posicion, layer) {  **this**.space = space;  **this**.layer = layer;   *// Crear animación* **var** framesAnimacion = [];  **for** (**var** i = 1; i <= 8; i++) {  **var** str = **"cuervo"** + i + **".png"**;  **var** frame = cc.spriteFrameCache.getSpriteFrame(str);  framesAnimacion.push(frame);  }  **var** animacion = **new** cc.Animation(framesAnimacion, 0.2);  **var** actionAnimacionBucle =  **new** cc.RepeatForever(**new** cc.Animate(animacion));   *// Crear Sprite - Cuerpo y forma* **this**.sprite = **new** cc.PhysicsSprite(**"#cuervo1.png"**);  *// Cuerpo estática , no le afectan las fuerzas  // Cuerpo dinámico, SI le afectan las fuerzas* **this**.body = **new** cp.Body(5, Infinity);   **this**.body.setPos(posicion);  **this**.body.setAngle(0);  **this**.sprite.setBody(**this**.body);  *// Se añade el cuerpo al espacio* **this**.space.addBody(**this**.body);   *// forma* **this**.shape = **new** cp.BoxShape(**this**.body,  **this**.sprite.getContentSize().width,  **this**.sprite.getContentSize().height);  *// agregar forma dinamica* **this**.space.addShape(**this**.shape);  **this**.shape.setCollisionType(tipoEnemigo);  *// añadir sprite a la capa   // ejecutar la animación* **this**.sprite.runAction(actionAnimacionBucle);   layer.addChild(**this**.sprite,10);   }, moverAutomaticamente: **function**(){  *// invertir direccion   // Velocidad baja ha colisionado con algo,* **if** ( **this**.body.vx < 3 && **this**.body.vx > -3 ) {  **if** (**this**.direccion == **"derecha"**){  **this**.direccion = **"izquierda"**;  **this**.body.p.x = **this**.body.p.x -10; *// Para que salga de la colisión* **this**.sprite.scaleX = 1;  } **else** {  **this**.direccion = **"derecha"**;  **this**.body.p.x = **this**.body.p.x + 10; *// Para que salga de la zona de colisión* **this**.sprite.scaleX = -1;  }  }   *// Dar impulsos para mantener la velocidad* **if** (**this**.direccion == **"izquierda"** && **this**.body.vx > -100){  **this**.body.applyImpulse(cp.v(-100, 250), cp.v(0, 0));  }  **if** (**this**.direccion == **"derecha"** && **this**.body.vx < 100){  **this**.body.applyImpulse(cp.v(100, 250), cp.v(0, 0));  }     } , eliminar: **function** (){  *// quita la forma* **this**.space.removeShape(**this**.shape);   *// quita el cuerpo* **this**.space.removeBody(**this**.shape.getBody());   *// quita el sprite* **this**.layer.removeChild(**this**.sprite);  }, disparar: **function**(){  **if**(**new** Date().getTime() - **this**.tiempoDisparo > 1000 ){  **this**.tiempoDisparo = **new** Date().getTime();  **var** disparo = **new** Disparo(**this**.space,  cc.p(**this**.body.p.x, **this**.body.p.y),  **this**.layer,**true**);  **if**(**this**.direccion==**"derecha"**)  disparo.body.vx = 350;  **else** disparo.body.vx = -350;  **this**.layer.disparos.push(disparo);  }  } }); |

### Modificación del GameScene

Para empezar, se han creado las colisiones siguientes

|  |
| --- |
| },colisionEnemigoConContencion:**function**(arbiter,space){  **var** shapes = arbiter.getShapes();  *// shapes[0] es el enemigo* **var** formaEnemigo = shapes[0];  formaEnemigo.body.vx = 0; *// Parar enemigo* },colisionDisparoConJugador:**function** (arbiter, space) {  **var** capaControles = **this**.getParent().getChildByTag(idCapaControles);  **this**.jugador.restaVida();  **var** shapes = arbiter.getShapes();  **this**.formasEliminar.push(shapes[0]);  **if**(**this**.jugador.vidas<=0){  cc.director.pause();  cc.director.runScene(**new** GameOverLayer());  }  capaControles.actualizarVidas(**this**.jugador.vidas); } |

Y ya posteriormente se ha llamado a los métodos que tienen los nuevos enemigos en el update.

|  |
| --- |
| *// Mover enemigos:* **for**(**var** i = 0; i < **this**.nuevosEnemigos.length; i++) {  **var** enemigo = **this**.nuevosEnemigos[i];  enemigo.moverAutomaticamente();  enemigo.disparar(); } |

Con todo esto tendríamos enemigos que disparasen y se moviesen de izquierda a derecha en un entorno limitado.

## Turbo

Para realizar esta ampliación se necesita insertar un botón de turbo en la capa de controles, controlar la pulsación sobre este y una función en el jugador que le dé un impulso.

### Inclusión del botón

|  |
| --- |
| *// BotonTurbo* **this**.spriteBotonTurbo = cc.Sprite.create(res.boton\_turbo\_png); **this**.spriteBotonTurbo.setPosition(  cc.p(size.width\*0.8, size.height\*0.8));  **this**.addChild(**this**.spriteBotonTurbo); |

### Control de la pulsación

|  |
| --- |
| procesarMouseDown:**function**(event) {  **var** instancia = event.getCurrentTarget();  **var** areaBoton = instancia.spriteBotonSaltar.getBoundingBox();  **var** areaTurbo = instancia.spriteBotonTurbo.getBoundingBox();   *// La pulsación cae dentro del botón* **if** (cc.rectContainsPoint(areaBoton,  cc.p(event.getLocationX(), event.getLocationY()) )){   *// Accedemos al padre (Scene), pedimos la capa con la idCapaJuego* **var** gameLayer = instancia.getParent().getChildByTag(idCapaJuego);  *// tenemos el objeto GameLayer* gameLayer.jugador.saltar();  }  **if** (cc.rectContainsPoint(areaTurbo,  cc.p(event.getLocationX(), event.getLocationY()) )){  **var** gameLayer = instancia.getParent().getChildByTag(idCapaJuego);  gameLayer.jugador.turbo();  } |

### Inclusión de la función para impulsar

Se debe crear primero una variable con los usos máximos del turbo.

|  |
| --- |
| },turbo: **function**(){  **if**(**this**.usosTurbo>0){  **this**.body.applyImpulse(cp.v(5000, 0), cp.v(0, 0));  **this**.usosTurbo--;  } } |

## Pinchos

## Niveles

## Control el modo plataformas