|  |
| --- |
| var ZOMBIES = { |
|  | lastLoopRun: 0, |
|  | timeInit: 0, |
|  | HEROWIDTH: 80, |
|  | HEROHEIGHT: 80, |
|  | enemies: [], |
|  | LEVEL: 1, |
|  | LEVEL\_SPEED\_Enemy\_Generate: 500, |
|  | LEVEL\_SPEED\_Enemy\_Move: 2, |
|  | iterations: 0, |
|  | LEFT\_KEY: 65, |
|  | UP\_KEY: 87, |
|  | RIGHT\_KEY: 68, |
|  | DOWN\_KEY: 83, |
|  | SPACE\_KEY: 32, |
|  | LAZER\_SPEED: 25, |
|  | laserArray: [], |
|  | ENEMYHEIGHT: 40, |
|  | ENEMYWIDTH: 40, |
|  | FINISH:false, |
|  | SCORE:0, |
|  |  |
|  | init: function (options) { |
|  |  |
|  | ZOMBIES.hero = new Hero('assets/images/heros/male-hero.png', ZOMBIES.HEROHEIGHT, ZOMBIES.HEROWIDTH); |
|  | ZOMBIES.scoreDiv = document.getElementById('score'); |
|  | document.onkeydown = function (evt) { |
|  | ZOMBIES.toggleKey(evt.keyCode, true); |
|  | }; |
|  |  |
|  | document.onkeyup = function (evt) { |
|  | ZOMBIES.toggleKey(evt.keyCode, false); |
|  | }; |
|  |  |
|  | ZOMBIES.loop(); |
|  | } |
|  | , |
|  | loop: function () { |
|  |  |
|  | if (new Date().getTime() - ZOMBIES.lastLoopRun > 40) { |
|  | ZOMBIES.updatePositions(); |
|  | ZOMBIES.hero.handelControllers(); |
|  | ZOMBIES.checkCollisions(); |
|  |  |
|  | ZOMBIES.addEnemy(); |
|  |  |
|  | // ZOMBIES.showSprites(); |
|  |  |
|  | ZOMBIES.lastLoopRun = new Date().getTime(); |
|  | ZOMBIES.iterations++; |
|  | } |
|  |  |
|  | if(!ZOMBIES.FINISH) |
|  | setTimeout('ZOMBIES.loop();', 41); |
|  | } |
|  | , |
|  | updatePositions: function () { |
|  | for (var i = 0; i < ZOMBIES.laserArray.length; i++) { |
|  | if (ZOMBIES.laserArray[i].isGotOut) { |
|  | ZOMBIES.laserArray[i].remove(); |
|  | ZOMBIES.laserArray.splice(i, 1); |
|  | } else { |
|  | ZOMBIES.laserArray[i].y -= ZOMBIES.LAZER\_SPEED; |
|  | } |
|  | } |
|  | for (var i = 0; i < ZOMBIES.enemies.length; i++) { |
|  | ZOMBIES.enemies[i].y += ZOMBIES.LEVEL\_SPEED\_Enemy\_Move; |
|  | if (ZOMBIES.enemies[i].isGotOut) { |
|  | ZOMBIES.enemies[i].remove(); |
|  | ZOMBIES.enemies.splice(i, 1); |
|  | } else { |
|  | ZOMBIES.enemies[i].y += ZOMBIES.LEVEL\_SPEED\_Enemy\_Move; |
|  | } |
|  | } |
|  |  |
|  | }, |
|  | toggleKey: function (keyCode, isPressed) { |
|  | if (keyCode == ZOMBIES.LEFT\_KEY) { |
|  | ZOMBIES.hero.moveLeft(isPressed); |
|  | } |
|  | if (keyCode == ZOMBIES.RIGHT\_KEY) { |
|  | ZOMBIES.hero.moveRight(isPressed); |
|  | } |
|  | if (keyCode == ZOMBIES.UP\_KEY) { |
|  | ZOMBIES.hero.moveUp(isPressed); |
|  | } |
|  | if (keyCode == ZOMBIES.DOWN\_KEY) { |
|  | ZOMBIES.hero.moveDown(isPressed); |
|  | } |
|  | if (keyCode == ZOMBIES.SPACE\_KEY && !ZOMBIES.finish && !isPressed) { |
|  | if (ZOMBIES.laserArray.length < 3) { |
|  | ZOMBIES.laserArray[ZOMBIES.laserArray.length] = new Laser('assets/images/heros/male-hero.png', 20, 5, ZOMBIES.hero.x + (ZOMBIES.hero.w / 2) - 3, ZOMBIES.hero.y); |
|  | } |
|  | } |
|  | } |
|  | , |
|  | checkCollisions: function () { |
|  | for (var i = 0; i < ZOMBIES.enemies.length; i++) { |
|  |  |
|  | var laser = ZOMBIES.getIntersectingLaser(ZOMBIES.enemies[i]); |
|  | if (laser) { |
|  |  |
|  | (function () { |
|  | var i2 = i; // closure of i (lexical scope: for-loop) |
|  |  |
|  | // Hassan Edit, removed the enemy immediatly after it die instead of in timeout because of a BUG ! |
|  | ZOMBIES.enemies[i2].onDie(); |
|  | var temp = ZOMBIES.enemies[i2] ; |
|  | ZOMBIES.enemies.splice(i2, 1); |
|  | setTimeout(function () { |
|  |  |
|  | temp.remove(); |
|  | }, 500) |
|  | })(); |
|  |  |
|  |  |
|  | i--; |
|  | laser.y = -laser.h; |
|  | ZOMBIES.SCORE += 100; |
|  | ZOMBIES.scoreDiv.textContent = ZOMBIES.SCORE ; |
|  | } else if (ZOMBIES.intersects(ZOMBIES.hero, ZOMBIES.enemies[i])) { |
|  |  |
|  | ZOMBIES.enemies[i].onDie(); |
|  | ZOMBIES.gameOver(); |
|  | } |
|  | } |
|  | }, |
|  | getIntersectingLaser: function (enemy) { |
|  | var result = null; |
|  | for (var i = 0; i < ZOMBIES.laserArray.length; i++) { |
|  | if (ZOMBIES.intersects(ZOMBIES.laserArray[i], enemy)) { |
|  | result = ZOMBIES.laserArray[i]; |
|  | break; |
|  | } |
|  | } |
|  | return result; |
|  | } |
|  |  |
|  | , |
|  | gameOver: function () { |
|  | ZOMBIES.FINISH = true; |
|  | var element = document.getElementById(hero.id); |
|  | element.style.visibility = 'hidden'; |
|  | //element.parentElement |
|  | // element = document.getElementById('gameover'); |
|  | // element.style.visibility = 'visible'; |
|  | } |
|  | , |
|  | intersects: function (a, b) { |
|  | return a.x < b.x + b.w && a.x + a.w > b.x && a.y < b.y + b.h && a.y + a.h > b.y; |
|  | } |
|  | , |
|  | helpers: { |
|  | getRandom: function (maxSize) { |
|  | return parseInt(Math.random() \* maxSize); |
|  | } |
|  | } |
|  | , addEnemy: function () { |
|  | ZOMBIES.interval = 50; |
|  |  |
|  | // if (ZOMBIES.iterations > ZOMBIES.LEVEL \* ZOMBIES.LEVEL\_SPEED\_Enemy\_Generate) { |
|  | // ZOMBIES.interval = 5; |
|  | // } else if (ZOMBIES.iterations > ZOMBIES.LEVEL \* ZOMBIES.LEVEL\_SPEED\_Enemy\_Generate) { |
|  | // ZOMBIES.interval = 20; |
|  | // } else if (ZOMBIES.iterations > ZOMBIES.LEVEL \* ZOMBIES.LEVEL\_SPEED\_Enemy\_Generate) { |
|  | // ZOMBIES.interval = 35; |
|  | // } |
|  |  |
|  | if (ZOMBIES.helpers.getRandom(ZOMBIES.interval) == 0) { |
|  | var elementName = 'enemy' + ZOMBIES.helpers.getRandom(10000000); |
|  | var enemy = new Enemy('assets/images/heros/male-hero.png', ZOMBIES.ENEMYHEIGHT, ZOMBIES.ENEMYWIDTH); |
|  | ZOMBIES.enemies.push(enemy); |
|  | } |
|  | } |
|  |  |
|  | } |
|  | ; |
|  | ZOMBIES.init(); |