**LC3 BreakOut**

# **Brick.asm**

Este archivo contiene codigo que manaja el funcionamiento de los ladrillos, a grandes rasgo contiene las funciones que chequean sus colisiones, y su reaccion a ellas. Además de poseer sus dimensiones.

## **Estructura en memoria**

struct ladrillo {

int x;

int y;

unsigned int muerto:1;

unsigned int estado:15;

}

## **Funciones**

### **BRICK\_ON\_COLLISION**

Es la funcion que se ejecuta cuando la pelota choca con un ladrillo. Disminuye el estado en uno y lo rerenderea acorde.

### Pseudocodigo

void BRICK\_ON\_COLLISION(struct Ladrillo\* ladrillo) {

ladrillo->estado -= 1;

if (ladrillo->muerto) {

BRICK\_UNRENDER(ladrillo);

} else {

BRICK\_RENDER(ladrillo);

}

}

## **BRICK\_CHECK\_COLL**

Es la funcion que chequea si la pelota colisiono con un ladrillo.

### Pseudocodigo

void BRICK\_CHECK\_COLLISION(struct Ladrillo\* ladrillo, struct Pelota\* pelota) {

if (ladrillo->muerto) {

return false;

}

int l\_x = ladrillo->x;

int l\_y = ladrillo->y;

int p\_x = BITSHIFT8\_RIGHT(pelota->x);

int p\_y = BITSHIFT8\_RIGHT(pelota->y);

if (p\_x + BALL\_WIDTH >= l\_x && p\_x <= l\_x + BRICK\_WIDTH && p\_y + BALL\_HEIGHT >= l\_y && p\_y <= l\_y + BRICK\_HEIGHT) {

return true;

}

return false;

}

## **Constantes**

### **BRICK\_HEIGHT**

El alto de los ladrillo.

#### Pseudocodigo

int BRICK\_HEIGHT = 4;

### **BRICK\_WIDTH**

El ancho de los ladrillos.

#### Pseudocodigo

int BRICK\_WIDTH = 16;