

**MOUSE**

import processing.serial.\*; // línea que accede a la ruta para la counicacion puerto serie

Maze maze;

// El puerto serial

//Serial myPort; //Declaracion Variable “myPort para la comunicación Puerto Serie (COM), en este caso con la tarjeta Arduino”

void setup() {

// Tamaño de la ventana

size(400, 400);

// Tamaño fuente

textSize(20);

// Color de fondo de la ventana

background(51);

// Alineación del texto

textAlign(CENTER, CENTER);

// Crea una nueva cuadrícula de 6 x 6

maze = new Maze(6);

// Abre el puerto serial con velocidad de 9600 baudios

//myPort = new Serial(this, "/dev/ttyUSB1", 9600);

//myPort.clear();

// Ejemplo: Coloca pared en norte y este de la celda 2,2

maze.setWalls(2,2, Maze.NORTH | Maze.EAST );

}

void draw() {

maze.display();

/\*while (myPort.available() > 0) {

byte[] inBuffer = new byte[11];

if( myPort.readBytesUntil('\n', inBuffer) > 0){

String myString = new String(inBuffer);

print(int(myString.substring(0,3)));

print(int(myString.substring(3,6)));

println(int(myString.substring(6,9)));

}

}\*/

}

**MAZE**

class Maze {

public static final int NORTH = 0x01;

public static final int EAST = 0x02;

public static final int SOUTH = 0x04;

public static final int WEST = 0x08;

private int gSize;

private int[][] gArray;

private int[][] gWalls;

private int widthSquare = 60;

private int mPosx, mPosy;

// Constructor

public Maze(int size){

gSize = size;

gArray = new int[size][size];

gWalls = new int[size][size];

mPosx = mPosy = 0;

int even = (size % 2)==0 ? 1 : 0;

// Inicializa la cuadrícula

for (int i = 0; i <= size/2-even; i++) {

for (int j = 0; j <= size/2-even; j++) {

gArray[i][j] = size-j-i-1-even;

gArray[size-i-1][j] = size-j-i-1-even;

gArray[i][size-j-1] = size-j-i-1-even;

gArray[size-i-1][size-j-1] = size-j-i-1-even;

}

}

// Inicializa las paredes

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

gWalls[i][j] = 0;

if( j == 0 )

gWalls[i][j] = NORTH;

if( j == size-1 )

gWalls[i][j] = SOUTH;

if( i == 0 )

gWalls[i][j] |= WEST;

if( i == size-1 )

gWalls[i][j] |= EAST;

}

}

}

// Obtiene el valor de una celda

public int getValue(int x, int y){

if( (x<gSize) && (y<gSize) )

return gArray[x][y];

else

return 0;

}

// Actualiza el valor de una celda

public void setValue(int x, int y, int value){

if( (x<gSize) && (y<gSize) )

gArray[x][y] = value;

}

// Actualiza la posición del mouse

public void setMouse(int x, int y){

if( (x<gSize) && (y<gSize) ){

mPosx = x;

mPosy = y;

}

}

// Actualiza los muros de una celda

public void setWalls(int x, int y, int walls){

if( (x<gSize) && (y<gSize) ){

gWalls[x][y] = walls & 0xF;

}

}

// Obtiene los muros de una celda

public int getWalls(int x, int y){

if( (x<gSize) && (y<gSize) )

return (gWalls[x][y] & 0xF);

else

return 0;

}

public void display(){

// Dibuja la cuadrícula

for (int x = 0; x < gSize; x++) {

for (int y = 0; y < gSize; y++) {

// Color de relleno para las figuras

fill(255);

stroke(153);

strokeWeight(1);

// Dibuja un rectángulo en la posición

// x, y de un ancho de 30 x 30

rect( (x\*widthSquare)+20, (y\*widthSquare)+20, widthSquare, widthSquare);

// Color del texto

fill(1);

// Coloca los números en la cuadrícula

text(gArray[x][y], (x\*widthSquare)+50, (y\*widthSquare)+50);

}

}

// Color de línea

stroke(204, 102, 0);

// Ancho paredes

strokeWeight(2);

// Dibuja las paredes

for (int x = 0; x < gSize; x++) {

for (int y = 0; y < gSize; y++) {

// Dibuja las paredes

if( (gWalls[x][y]&NORTH) == NORTH)

line((x\*widthSquare)+20, (y\*widthSquare)+20, (x\*widthSquare)+widthSquare+20, (y\*widthSquare)+20);

if( (gWalls[x][y]&EAST) == EAST)

line((x\*widthSquare)+widthSquare+20, (y\*widthSquare)+20, (x\*widthSquare)+widthSquare+20, (y\*widthSquare)+widthSquare+20);

if( (gWalls[x][y]&SOUTH) == SOUTH)

line((x\*widthSquare)+20, (y\*widthSquare)+widthSquare+20, (x\*widthSquare)+widthSquare+20, (y\*widthSquare)+widthSquare+20);

if( (gWalls[x][y]&WEST) == WEST)

line((x\*widthSquare)+20, (y\*widthSquare)+20, (x\*widthSquare)+20, (y\*widthSquare)+widthSquare+20);

}

}

stroke(1);

// Dibuja el mouse

rect( (mPosx\*widthSquare)+30, (mPosy\*widthSquare)+30, widthSquare-20, widthSquare-20, 10);

}

}