

**SNEST**

**DGEST**

**SEP**

INSTITUTO TECNOLOGICO DE TOLUCA

**Documentación del Juego 2D**

****

PRESENTA

**JESÚS GUZMÁN MONDRAGÓN**

**HÉCTOR MORALES PALMA**

**SANDRA MARLEN GUTIÉRREZ PINEDA**

PROFESOR

**ELIZABETH PULIDO**

MATERIA:

TALLER DE INVESTIGACION I

METEPEC, ESTADO DE MÉXICO, OCTUBRE DE 2012

**Presentación de MazeAgram**

MazeAgram por sus siglas en inglés Maze – Laberinto y la abreviación de Anagrama – Agram, es un juego dinámico, al cual lo conforma un laberinto generado aleatoriamente y una relación entre letras – imagen.

**Objetivo**

De los desarrolladores: Crear un juego interactivo a base de herramientas 2D que contenga lo solicitado como el manejo de polígonos (figuras básicas), personajes e interfaces totalmente diseñados por nosotros, utilizar composiciones, colisiones, transparencias, sonido y transformaciones bidimensionales, así como toda herramienta 2D que nos sea de utilidad. Logrando con ello un juego didáctico.  
  
Del juego: Ser un juego dinámico e interactivo, que pueda ser entretenido para los niños que lo jueguen, y al jugar éste les permita desarrollar el sentido de percepción espacial, aprendizaje de palabras, así como la relación palabra – imagen.

De jugar:

Parte 1. Adentrarse al laberinto y lograr juntar todas las letras que se encuentren en él, así como encontrar la solución de éste mismo, es decir la salida!

Parte 2. Con las letras recolectadas, ordenar cada una en el lugar correspondiente para formar una palabra, para ello el usuario tendrá la ayuda de una imagen que te permitirá saber cual es la palabra que tienes que formar con esas letras.

**Justificación**

Se pretende con éste juego lograr desarrollar cualidades específicas en ambos participantes, es decir, tanto en:

El desarrollador (aplicación y ejecución de la parte teórica de programación, aprendizaje de herramientas 2D, graficación de polígonos y transformaciones bidimensionales, así como el trabajo en equipo);

El usuario, interacción con un juego de computadora, desarrollo de imaginación espacial, aprendizaje de palabras, capacidad de relación de palabra – imagen.

Todo esto en un entorno sano, pretendiendo que las cualidades en cada uno sean aprendidas exitosamente.

**Delimitación**

Temporal: A partir de la fecha en que sea presentado y los desarrolladores lo aprueben.  
Espacial: Instituto Tecnológico de Toluca  
Demográfico: Para toda persona, aunque es enfocado a niños de 6 a 8 años, que sepan leer.

**Diseño y Diagrama de Soluciones**

El soporte y mantenimiento, son factores clave al momento de evaluar el éxito y la sustentabilidad del juego.  
  
Estos puntos se comienzan a definir desde el principio mismo del proyecto junto con el diseño de la solución, donde se establecen las políticas del juego a y las características generales de instalación, así como de ejecución del mismo,  que permitan el crecimiento a futuro tanto de las cualidades ya aprendidas en el desarrollador y el usuario como la oportunidad en el desarrollador de mejorar los proyectos similares a éste, así como solucionar cualquier percance que se llegue a presentar.  
Algunas de éstas son:

\* Conocer perfectamente el juego para poder solucionar algún percance.

\* Entorno e interfaces de calidad

\* Instrucciones claras de cómo manejar el juego

\* Fácil gestión

\* Soluciones Rápidas



**Código Relevante**

Este hilo se encuentra en la clase Pantalla y es el que mueve el laberinto a través del personaje, así como detectar las colisiones con la salida, los cofres y las paredes del laberinto. Es por esta razón que nos parece que es una de las partes más relevantes dentro del juego.

public void run() { //hilo para mover el laberinto y el personaje

int bandera1=0; //bandera para el personaje

while(true){ //iniciamos el ciclo

Rectangle aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

//creamos un rectangulo de las posicion del laberinto actual

if(pnj.arriba){ // si se presiona w movemos hacia arriba

this.mueveCofresAb(); //movemos los cofres

maze.inicioy+=velocidad; maze.salida.y+=velocidad; //movemos el laberinto y la salida

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

//actualizamos las posiciones del laberint

while(isPixelCollide(pnj.pnjimg1,maze.loglab,pnj.bounds,aux)){ //detectamos collision por pixel

maze.inicioy-=velocidad; maze.salida.y-=velocidad; // si ha colision nos regresamos

pnj.arriba=false;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

removeKeyListener(aux1);

this.mueveCofresA();

} addKeyListener(aux1);// pnj.pnjimg=pnj.generarImagen();

}

if(pnj.abajo){//movemos el laberinto

maze.inicioy-=velocidad; maze.salida.y-=velocidad;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresA();

while(isPixelCollide(pnj.pnjimg1,maze.loglab,pnj.bounds,aux)){

removeKeyListener(aux1);

maze.inicioy+=velocidad; maze.salida.y+=velocidad;

pnj.abajo=false;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresAb();

} addKeyListener(aux1);

}

if(pnj.izquierda){

maze.iniciox+=velocidad; maze.salida.x+=velocidad;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresI();

while(isPixelCollide(pnj.pnjimg1,maze.loglab,pnj.bounds,aux)){

removeKeyListener(aux1);

maze.iniciox-=velocidad; maze.salida.x-=velocidad;

pnj.izquierda=false;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresD();

} addKeyListener(aux1);

}

if(pnj.derecha){

maze.iniciox-=velocidad; maze.salida.x-=velocidad;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresD();

while(isPixelCollide(pnj.pnjimg1,maze.loglab,pnj.bounds,aux)){

removeKeyListener(aux1);

pnj.derecha=false; maze.salida.x+=velocidad;

maze.iniciox+=velocidad;

aux= new Rectangle(maze.iniciox,maze.inicioy,maze.labimg.getWidth(),maze.labimg.getHeight());

this.mueveCofresI();

} addKeyListener(aux1);

}

if (System.currentTimeMillis()-tiempo>70) { //cambiamos el radio del clipping cada .7 seg

if(bandera){

iluminacionradio+=10;

bandera=false;

} else{iluminacionradio-=10; bandera=true;}

tiempo=System.currentTimeMillis();

} if(pnj.bounds.intersects(maze.salida)&&maze.chests.length==chestabiertos.size()){ //verificamos si el personaje llego a la salida y si ya encontró todos los cofres

System.out.println("LLegaste a salida");

finalizado=true;

ventanaJuego.terminarHiloPL();//

ventanaJuego.establecesJuego2();//

break; //terminamos el hilo

}

if (System.currentTimeMillis()-tiempo2>100) { // cada segundo cambiamos la magen del personaje

if(bandera1==0){

pnj.pnjimg=pnj.pnjimg2;

bandera1++;

} else if(bandera1==1){pnj.pnjimg=pnj.pnjimg3; bandera1++;}

else if(bandera1==2){ pnj.pnjimg=pnj.pnjimg4; bandera1++;

} else{pnj.pnjimg=pnj.pnjimg1; bandera1=0;}

tiempo2=System.currentTimeMillis();

}

colicofres();//verificamos las colisiones con los cofres

update(getGrap()); //actualizamos la pantalla

}

}

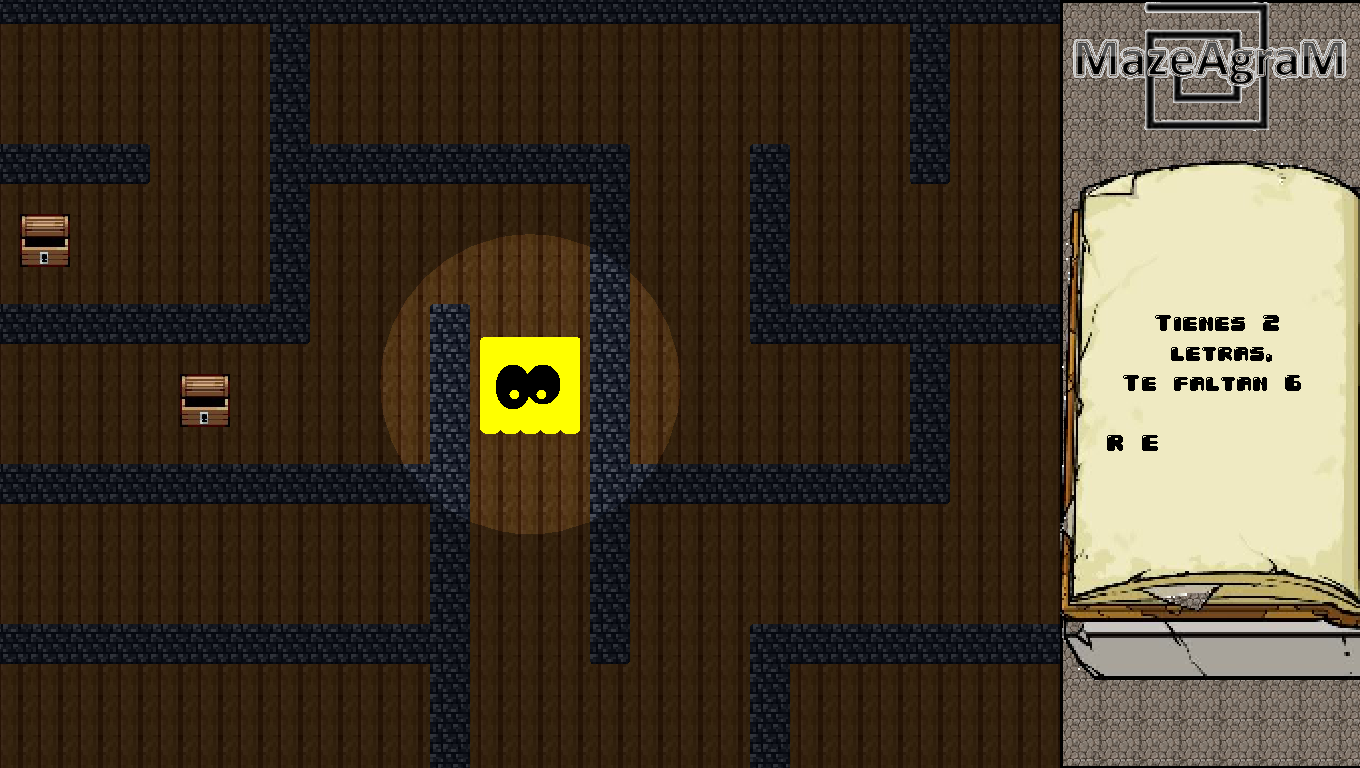
**Impresiones de Pantalla**

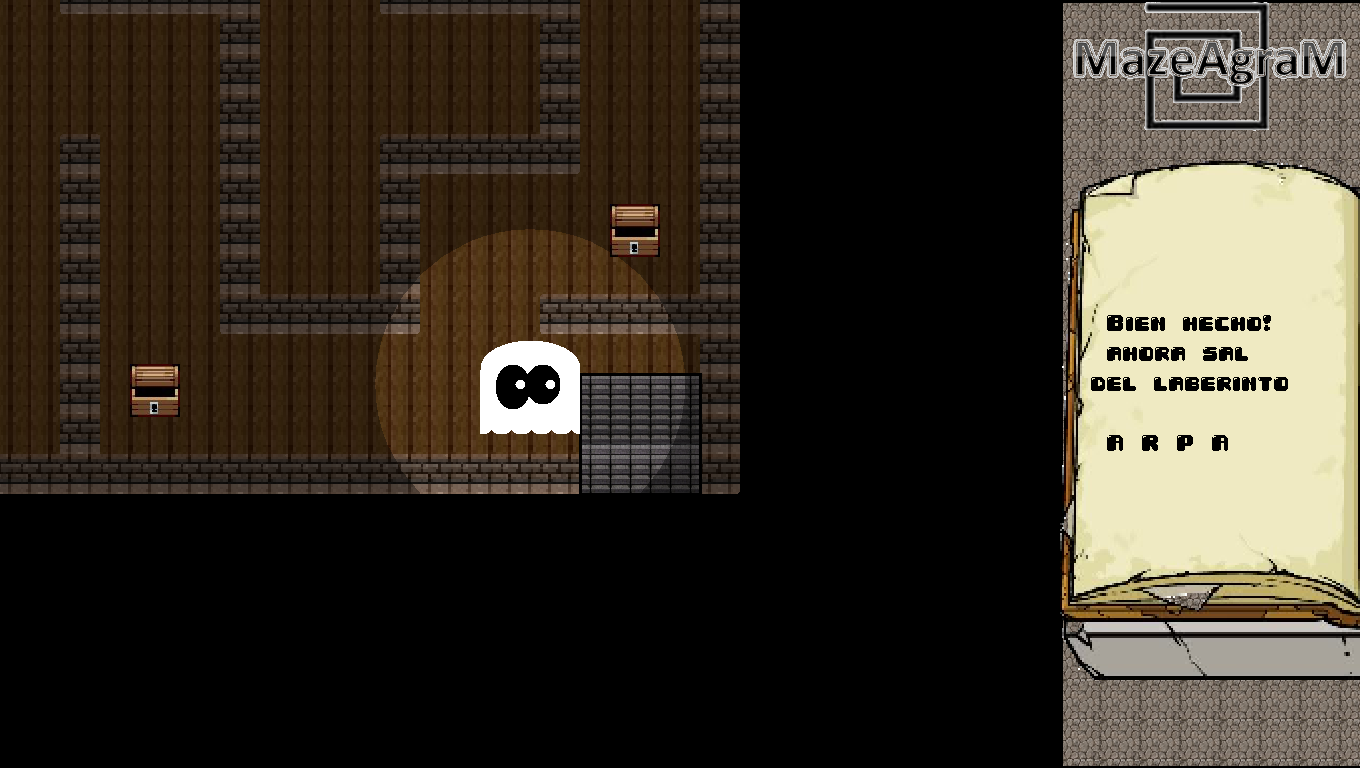
** 1**

**2 **

** 3**

**4 **

** 5**

**6 **

** 7**

**8 **

** 9**



**Dificultades Presentadas**

-Quitar el parpadeo en la primera fase del juego, para evitar el parpadeo a la hora de mover el laberinto, así como a la hora de mover las letras, lo cual se logró implementando un doble buffer.

-La creación de la clase MazeGenerator, que genera tanto el laberinto lógico, como la imagen que es utilizada para representarlo. El laberinto es generado aleatoriamente utilizando el método de Backtracking.

-La animación del personaje, la cual fue lograda utilizando varias imágenes que son cambiadas mediante el hilo.

-En la primera fase, la creación de imágenes a través del uso de los canvas, lo cual fue solucionado creando una BufferedImage temporal y mandándola al método paint de algunos canvas.

En la parte del juego denominado anagrama se presentó la dificultad de mantener la presentación de movimientos de las nubes y del rehilete al mismo tiempo. Para ello solo se utiliza un hilo el cual hace varias funciones tanto de aumentar la posición de las nubes y de aumentar la rotación del rehilete.

while(banderaHilo){

try {

anguloRotacion=Math.toRadians(n);

inicioNubes+=2; //velocidadNubes

Thread.sleep(100l);

n+=10;

if(n>=370)

n=0;

if(inicioNubes==0)

inicioNubes=-400;

repintar();

} catch (InterruptedException ex) {

System.out.println("GATOMAN estuvo aqui");

}

}

Para ello primero se aumenta el ángulo de rotación, después se aumenta la posición de las nubes en aumento de 2. Se hace un sleep para ver la transición de la imagen.

Se procede hacer el aumento de la variable n y hacer su validación de que no aumento a mayor de 370.

Después de esto se hace un repintado al canvas para ver el efecto de cambio.

**Conclusiones**

Jesús Guzmán Mondragón.

La creación de un juego en dos dimensiones puede ser muy larga y tediosa, incluso más la creación de las imágenes y sprites que se muestran el mismo, el manejo de los hilos es fundamental para todo tipo de juego. Ahora se valoran detalles en otros juegos que antes parecían inverosímiles o muy fáciles de hacer. Hay muchísimos factores en la creación de un videojuego que deben ser tomados en cuenta, desde el guion, la practicidad, los gráficos, la interacción con el usuario, y todo ello debe de ser combinado de manera creativa para que el juego cumpla su objetivo, incluyendo, si lo tiene, el educativo.

Héctor Morales Palma.

La importancia de desarrollar un juego educativo para niños de primaria es de que desarrollen imaginación espacial, y puedan relacionar imágenes con una palabra todo ello con la interacción para con la máquina utilizando el teclado y el ratón, lo que da un plus ya que no solo aprenden si no que aumentan sus habilidades con la computadora.

Sandra Marlen Gutiérrez Pineda.

El haber logrado elaborar un juego para niños que desarrolle cualidades como programador para poner a prueba los conocimientos que debieron adquirirse durante el curso es grato, pero que a parte de ello se logren desarrollar cualidades en los usuarios de corta edad como interacción con un juego de computadora, desarrollo de imaginación espacial, aprendizaje de palabras, capacidad de relación de palabra – imagen, es aún más satisfactorio.

**Bibliografía/Biografía/Sitios Web**

Recursos como texturas así como música, todos bajo licencia GNU/GPL.

www.openart.com

**Anexos**

En la segunda parte del juego, (relación palabras – imagen) se utilizaron imágenes de libros de primaria y algunas del juego de lotería, las cuales fueron escaneadas, esto con el fin de que los niños tengan alguna familiaridad con las imágenes y con ello facilitar su relación con las letras en desorden.