Juego de Tetris con JavaScript

Juego Breakout con Canvas de HTML5 **1**
9.1. Vista general del juego rompe ladrillos o Breakout 2
9.2. Breakout: Canvas y variables 2
9.3. Breakout: Funciones 4
9.4. Breakout: Dibujar 6
9.5. Breakout: Dibujar ladrillos 7
9.6. Breakout: Mover la pelotita 9
9.7. Breakout: Mover el paddle 10
9.8. Breakout: Detectar la colisión 12

12.1. Introducción al juego

- El tablero es un conjunto de imágenes.
- Cada cuadro es una imagen.
- Podemos utilizar el apuntador del ratón.
- La información de las formas se almacenan en arreglos.

```
/*****************
 Parámetros
*****************/
//
//Número de cuadros
//
nCuadros=4;
//
//Tipos de figuras
//
nTipos=7;
//
//Tamaño del tablero
//
tableroAltura=16;
tableroAncho =10;
//
//Nivel
//
nivel=1;
//
//Velocidad
//
velocidad0=700;
velocidadK=60;
velocidad=velocidad0-velocidadK*nivel;
//
//Número de líneas
//
nLineas=0;
//
/*********************
  GLOBAL VARIABLES
**********************/
//
//Coordenadas actuales x,y
//
curX=1; curY=1;
//
//La última línea disponible
```

```
41 //
42 lineaLimite=tableroAltura-1;
43 //
44 //Serial N
45 //
46 serialN=0;
47 //
48 //Banderas
49 //
50 tableroCargado=0;
51 pausaJuego=0;
52 inicioJuego=0;
53 finJuego=0;
54 //
55 //Timer
56 //
57 timerID=null;
58 //
59 // Imágenes
60 //
61 if (document.images) {
62  Img0=new Image(); Img0.src='s0.gif';
63  Img1=new Image(); Img1.src='s1.gif';
64  Img2=new Image(); Img2.src='s2.gif';
65  Img3=new Image(); Img3.src='s3.gif';
66  Img4=new Image(); Img4.src='s4.gif';
67  Img5=new Image(); Img5.src='s5.gif';
68  Img6=new Image(); Img6.src='s6.gif';
69  Img7=new Image(); Img7.src='s7.gif';
70 }
71 //
72 // ARREGLOS
73 //
74 //Creamos arreglos de 20x20 con 0
75 //
76 f=new Array();
77 for (i=0;i<20;i++) {
78  f[i]=new Array();
79  for (j=0;j<20;j++) {
80    f[i][j]=0;
81  }
82 }
83 //
84 xBorrar =new Array(0,0,0,0);     yBorrar =new Array(0,0,0,0);
85 //
86 //Desplazamiento dx, dy, dx_, dy_
87 //
88 dx=new Array(0,0,0,0); dy=new Array(0,0,0,0);
89 dx_=new Array(0,0,0,0); dy_=new Array(0,0,0,0);
90 //
91 //dxBank, dyBank 4?
92 //
93 dxBank=new Array(); dyBank=new Array();
94 dxBank[1]=new Array(0, 1,-1, 0);  dyBank[1]=new Array(0, 0, 0, 1);
```

```
 95 dxBank[2]=new Array(0, 1,-1,-1);  dyBank[2]=new Array(0, 0, 0, 1);
 96 dxBank[3]=new Array(0, 1,-1, 1);  dyBank[3]=new Array(0, 0, 0, 1);
 97 dxBank[4]=new Array(0,-1, 1, 0);  dyBank[4]=new Array(0, 0, 1, 1);
 98 dxBank[5]=new Array(0, 1,-1, 0);  dyBank[5]=new Array(0, 0, 1, 1);
 99 dxBank[6]=new Array(0, 1,-1,-2);  dyBank[6]=new Array(0, 0, 0, 0);
100 dxBank[7]=new Array(0, 1, 1, 0);  dyBank[7]=new Array(0, 0, 1, 1);
```

Listado 12.1.1. Constantes y variables del juego, archivo tetris.js

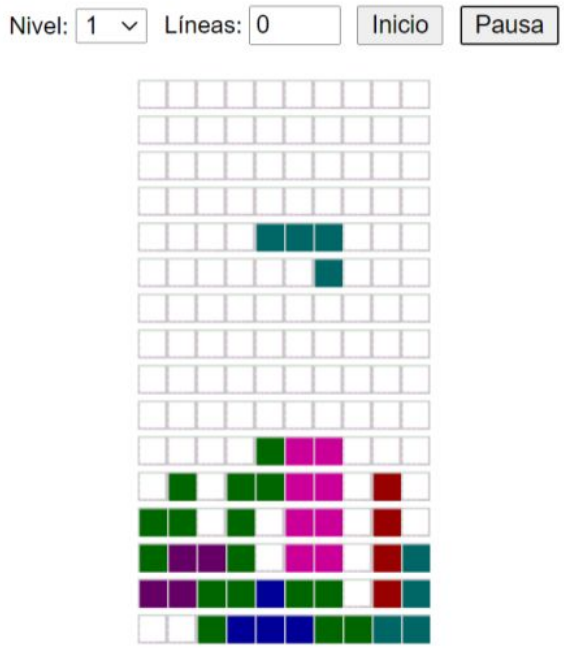

Imagen 12.1.1. Juego tetris

Imagen 12.1.2. Creado por Creado por Alekséi Pázhitnov en 1984 (Wikipedia)

12.2. Las figuras del juego

Las piezas se almacenan en arreglos o vectores y se “dibujan” dentro de la matriz del tablero. En los siguientes ejemplos tomamos como base la coordenada (10,10) y damos las coordenadas relativas en x (se suma la cantidad en los arreglos llamados **dx*Bank*** y ***dyBank***).

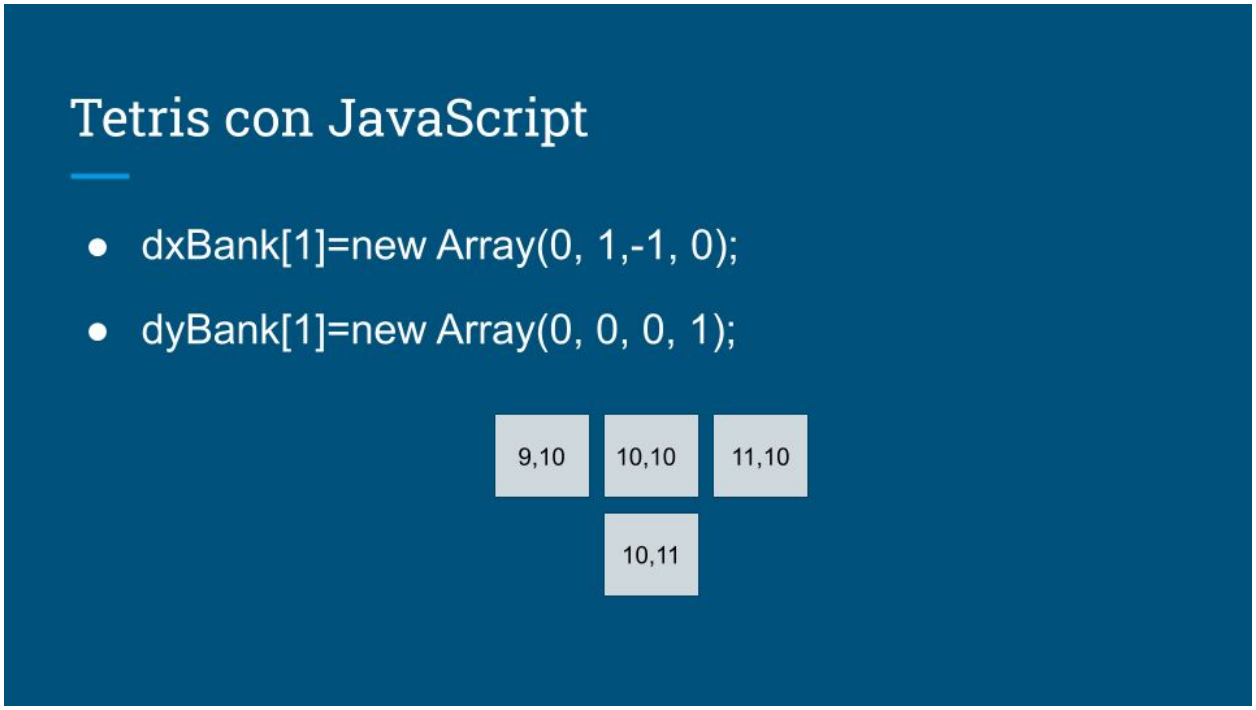

Imagen 12.2.1. Primera figura básica del Tetris

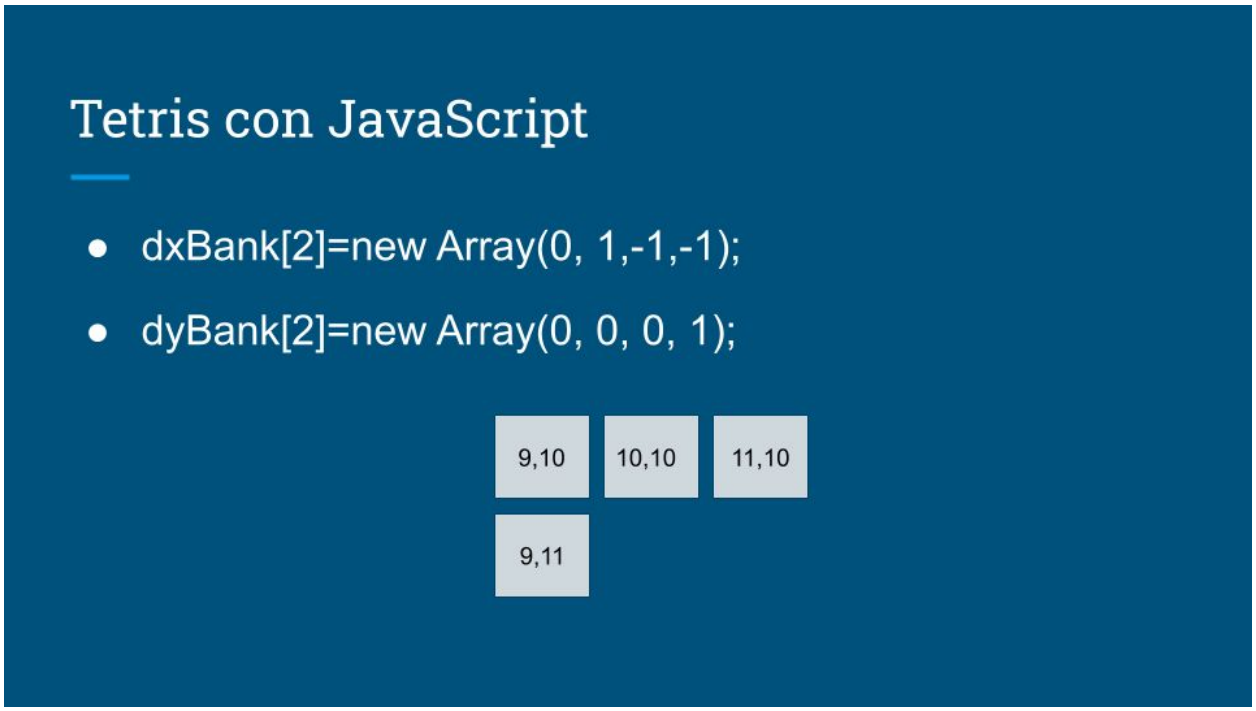

Imagen 12.2.2. Segunda figura básica del Tetris

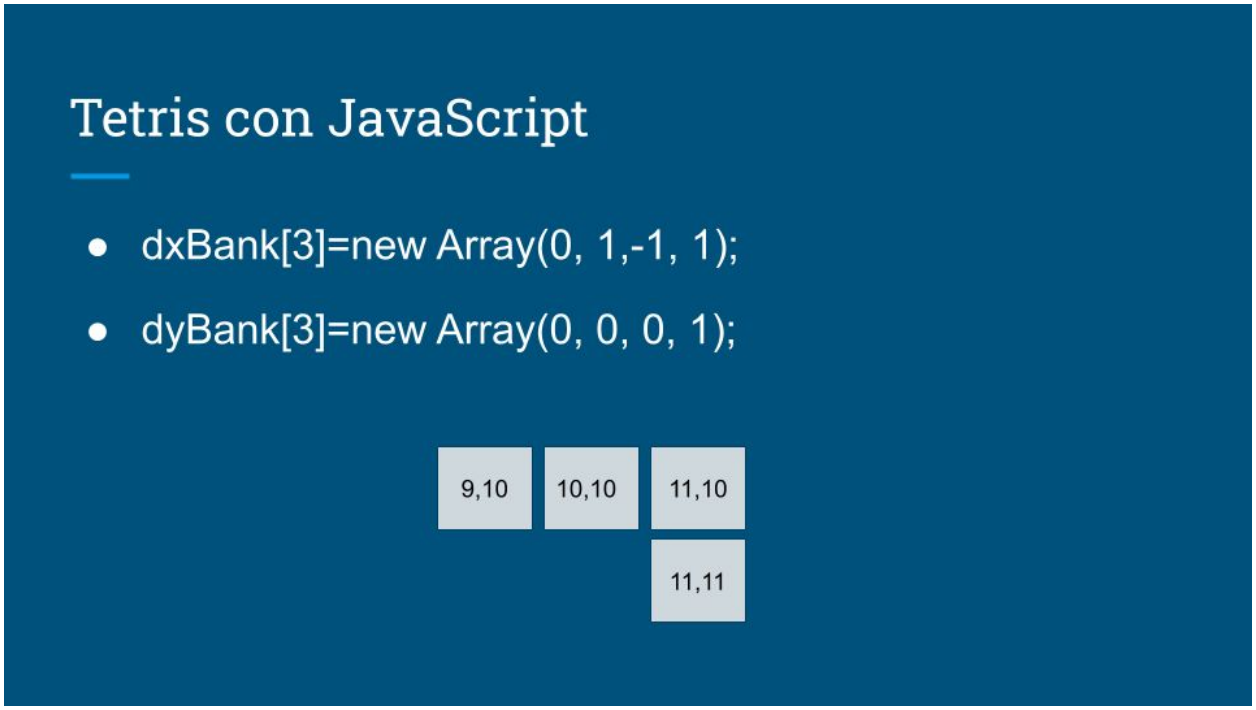

Imagen 12.2.3. Tercera figura básica del Tetris

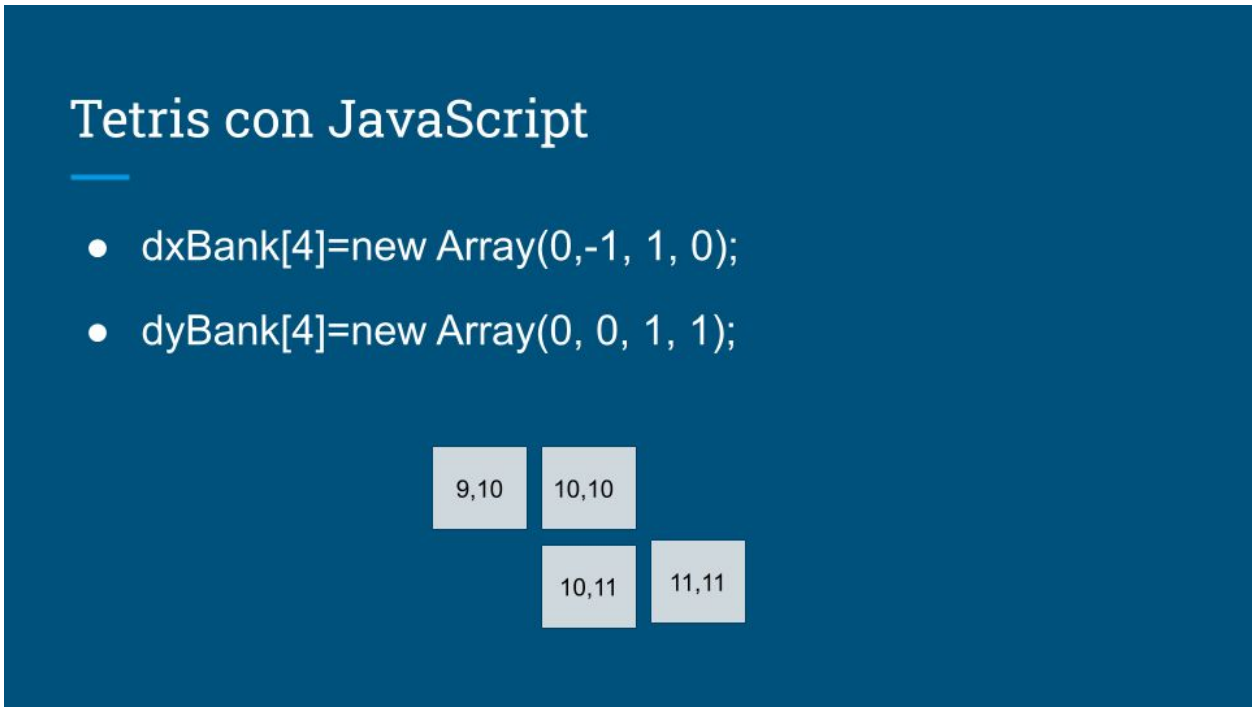

Imagen 12.2.4. Cuarta figura básica del Tetris

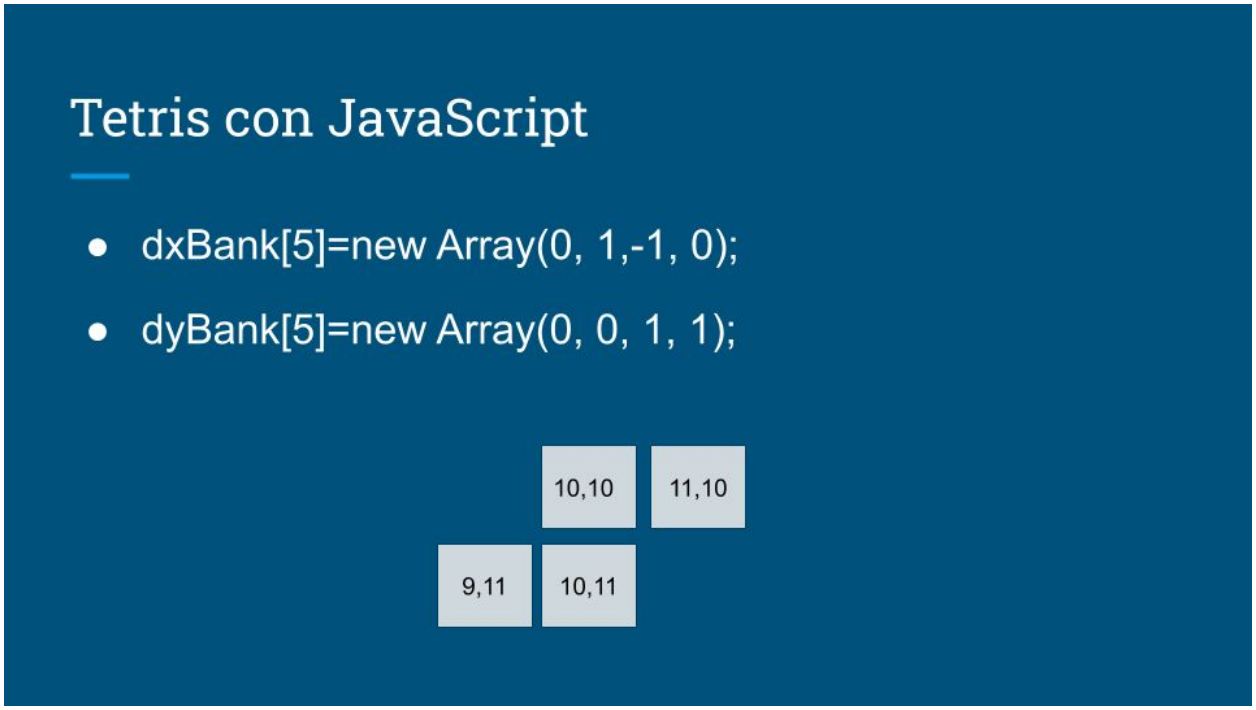

Imagen 12.2.5. Quinta figura básica del Tetris

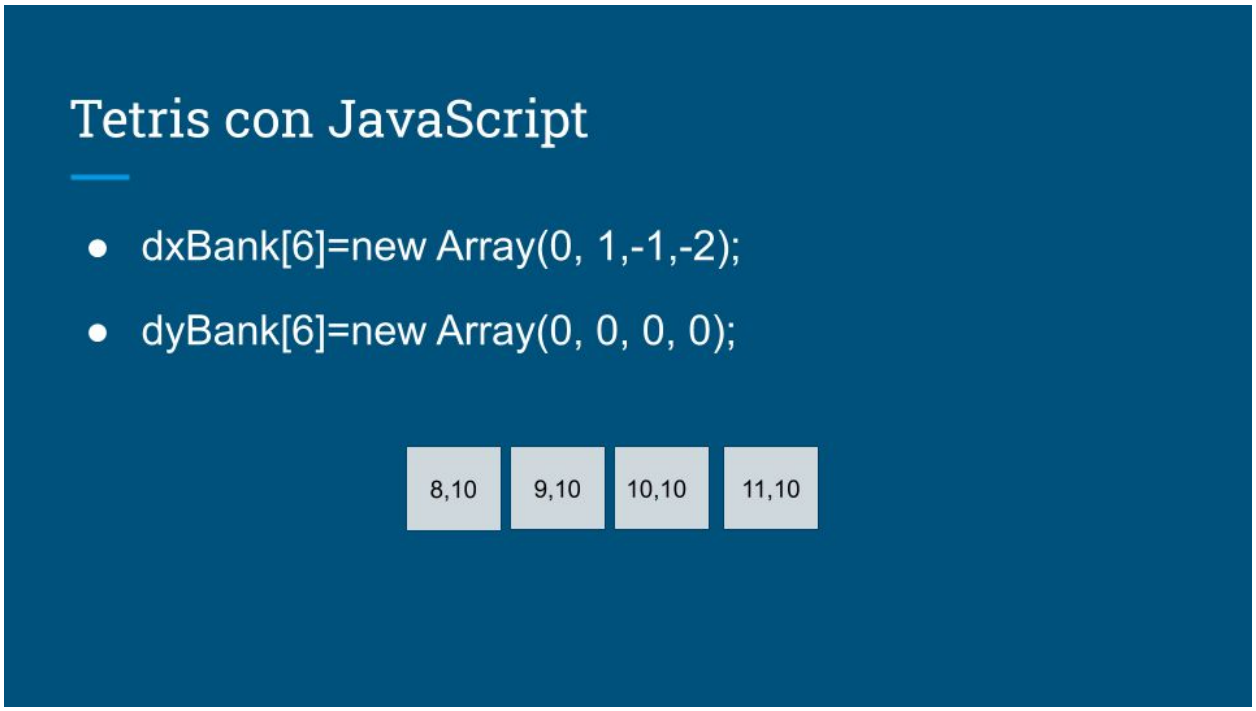

Imagen 12.2.6. Sexta imagen básica del tetris

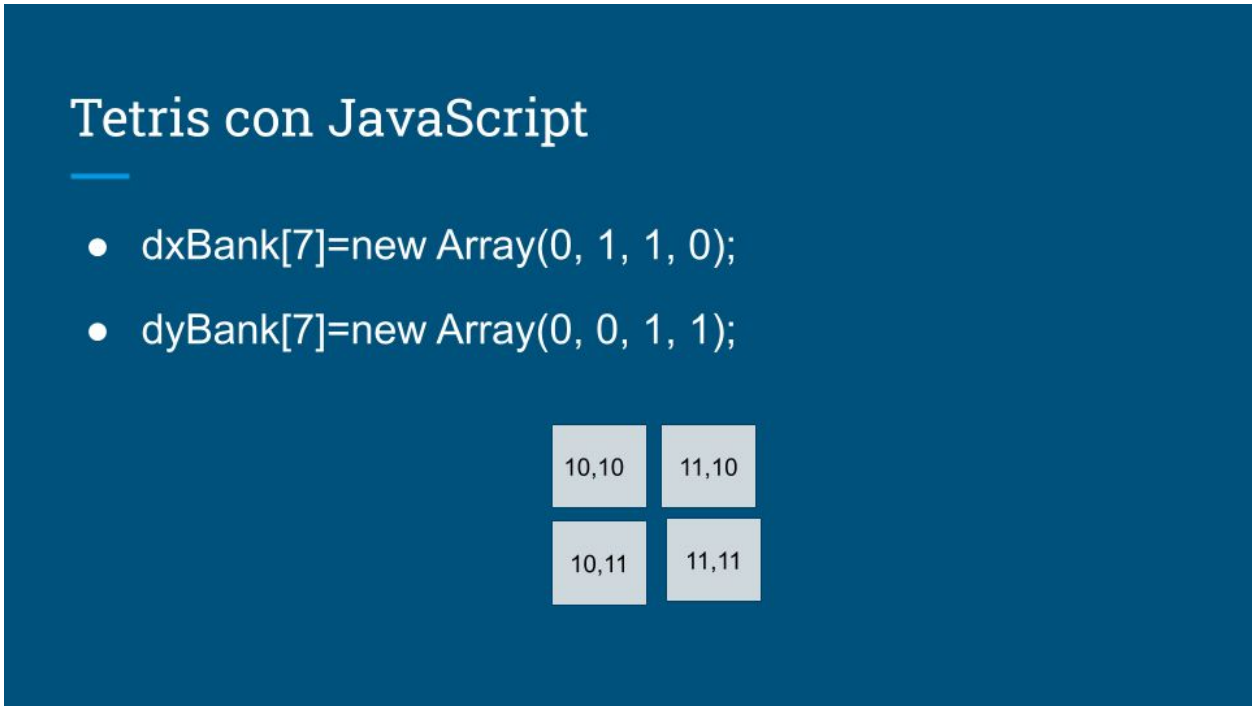

Imagen 12.2.7. Séptima figura básica del tetris

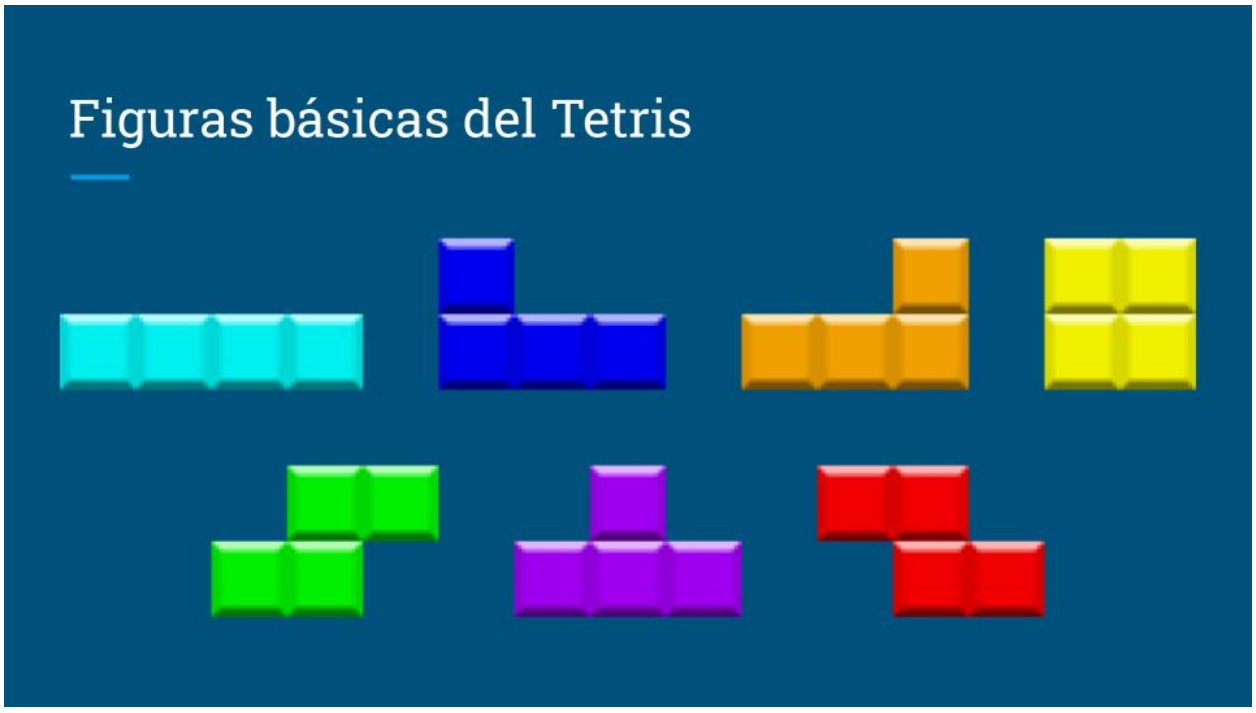

Imagen 12.2.8. Partes básicas del Tetris

12.3. Iniciar el juego

- Reiniciamos la matriz, las banderas y los contadores.
- El usuario inicia el juego presionando el botón correspondiente.

```
<!DOCTYPE html>
<html>
<meta charset="utf-8">
<head>
	<title>JavaScript Tetris 01</title>
	<style>
		a,img{margin:0;padding: 0;}
	</style>
	<script src="tetris.js"></script>
	<script>
buf='<center><form name=form1><table border=0 cellspacing=3
cellpadding=3><tr>'
+'<td><font face="Arial,Helvetica,sans-serif" size=2 point-size=10'
+'><nobr>Nivel:</font> <select name=s1
onchange="recuperarNivel();this.blur();">'
+'<option value=1 selected>1'
+'<option value=2>2'
+'<option value=3>3'
+'<option value=4>4'
+'<option value=5>5'
+'<option value=6>6'
+'<option value=7>7'
+'<option value=8>8'
+'<option value=9>9'
+'<option value=10>10'
+'</select>'
+'</nobr></font></td>'

+'<td>'
+'<font face="Arial,Helvetica,sans-serif"'
+'size=2 point-size=10'
+'><nobr>Líneas: <input name=lineas type=text value="0" size=2 readonly'
+'></nobr></font></td>'

+'<td><input type=button value="Inicio" onCLick="iniciarJuego()"></td>'
+'<td><input type=button value="Pausa" onCLick="pausarJuego()"></td>'
+'</tr></table></form>';

buf+='<pre';
for (i=0;i<tableroAltura;i++) {
 for (j=0;j<tableroAncho;j++) {
  buf+='><a href="#s" onclick="seleccionar('+i+','+j+');return false;"'
      +'><img name="s'+i+'_'+j+'" src="s'+Math.abs(f[i][j])+'.gif" width=16
height=16 border=0></a';
 }
 buf+='><br';
```

```
 47 }
 48 buf+='></pre></center>';
 49
 50 //
 51 document.writeln(buf);
 52 //
 53 //Enciende la bandera
 54 //
 55 tableroCargado=1;
 56 //
 57 window.onload = function(){
 58        inicio();
 59 }
 60 </script>
 61 </head>
 62 <body bgcolor="#FFFFFF" >
 63 </body>
 64 </html>
```

Listado 12.3.1. Estructura de las funciones

Nota: Los números de línea son aproximados.

```
447 function inicio() {
448   //
449   //Descripción: inicia el juego
450   //Parámetros:
451   //Variables locales:
452   //Variables globales:
453   //Salida:
454   //
455   //Manejo de eventos
456   //
457   document.onkeydown = pulsarTecla;
458   document.onkeyup = levantarTecla;
459   //
460   //Limpia variables
461   //
462   reiniciaJuego();
463   //
464   top.focus();
465 }
```

Listado 12.3.2. La función de inicio

```
104 function reiniciaJuego() {
105   //
106   //Descripción: Limpia las variables a cero
107   //Parámetros: ninguno
```

```
108   //Variables locales: i,j
109   //Variables globales: tableroAltura, tableroAncho, f, inicioJuego,
110   //                    pausaJuego, nLineas, serialN
111   //Salida: f,inicioJuego, pausaJuego, nLineas, serialN
112   //
113   for (var i = 0; i < tableroAltura; i++) {
114     for (var j = 0; j < tableroAncho; j++) {
115       f[i][j] = 0;
116       if (tableroCargado) {
117         //cargamos el recuadro blanco
118         eval("top.document.s"+i+"_"+j+".src='s0.gif'");
119       }
120     }
121   }
122   //
123   //Inicializa banderas
124   //
125   inicioJuego = 0;
126   pausaJuego = 0;
127   nLineas = 0;
128   serialN = 0;
129   //
130   // Ultima línea disponible
131   //
132   lineaLimite = tableroAltura-1;
133   //
134   //Numero de líneas
135   //
136   if (tableroCargado) {
137     self.document.form1.lineas.value=nLineas;
138   }
139   console.log("reiniciaJuego");
140 }
```

Listado 12.3.3. La función de reinicio

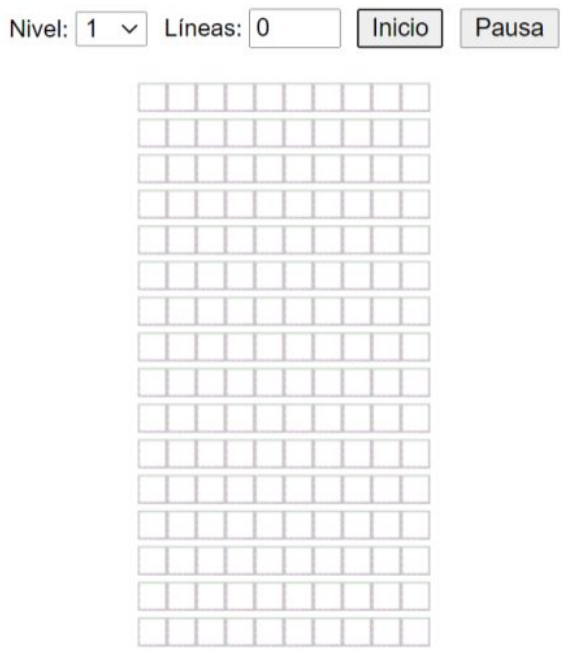

Imagen 12.3.1. Grid del juego

12.4. Iniciar el juego por el usuario

- Verificar si es el fin del juego
- Verificar si se reanuda el juego
- Generamos una pieza del Tetris
- Dibujamos la pieza
- Iniciamos banderas
- Iniciamos el Timeout

```
142 function iniciarJuego() {
143   //
144   //Descripción: Inicia el juego
145   //Parámetros: ninguno
146   //Variables locales:
147   //Variables globales: finJuego, inicioJuego, tableroCargado,
148   //                    pausaJuego, velocidad
149   //Salida:
150   //
151   //Terminar el juego
152   //
153   console.log("iniciarJuego");
```

```
154   if (finJuego) {
155     top.history.back();
156     finJuego = 0;
157   }
158   //
159   //foco()
160   //
161   top.focus();
162   //
163   //Ya se había seleccionado "inicio"
164   //
165   if (inicioJuego) {
166     if (!tableroCargado) {
167       return
168     }
169     if (pausaJuego) {
170       reanudarJuego();
171     }
172     return;
173   }
174   //
175   //Obtiene la pieza
176   //
177   obtenerPieza();
178   //
179   //Dibujar la pieza
180   //
181   dibujarPieza();
182   //
183   //Enciende la bandera de inicio
184   //
185   inicioJuego=1;
186   //
187   //Apaga la bandera de pausa
188   //
189   pausaJuego=0;
190   //
191   //Recupera el numero de líneas
192   //
193   self.document.form1.lineas.value = nLineas;
194   //
195   //Inicia el setTimeout
196   //
197   timerID = setTimeout("jugar()",velocidad);
198 }
```

Listado 12.4.1. Función iniciarJuego en el archivo tetris.js

12.5. Obtener una figura

- Recibir o generar el número de pieza
- Generar la pieza en los vectores

- Generar los arreglos temporales
- Validar pieza
- Dibujar pieza

```
370 function obtenerPieza(N) {
371   //
372   //Descripción: Calcula la pieza actual
373   //Parámetro: N, puede estar vacío y se calcula aleatoriamente
374   //Variables locales: piezaActual, k
375   //Variables globales: nTipos, curX, curY, nSqueres, dx,dy,dxBank, dyBank
376   //Llama funciones: pieceFits, dibujarPieza
377   //
378   var k;
379   var aleatorio = 1+Math.floor(nTipos*Math.random());
380   piezaActual = (obtenerPieza.arguments.length==0)?aleatorio:N;
381   console.log("obtenerPieza",piezaActual);
382   //
383   curX = 5;
384   curY = 0;
385   //
386   //Vaciamos los datos de la figura
387   for (var k = 0; k < nCuadros; k++) {
388     dx[k]=dxBank[piezaActual][k];
389     dy[k]=dyBank[piezaActual][k];
390   }
391   //
392   //Igualar los arreglos de trabajo
393   //
394   for (var k = 0; k < nCuadros; k++) {
395     dx_[k]=dx[k];
396     dy_[k]=dy[k];
397   }
398   //
399   //Valida pieza
400   //
401   if (validarPieza(curX,curY)) {
402     dibujarPieza();
403     return 1;
404   }
405   return 0;
406 }
```

Listado 12.5.1. Función **obtenerPieza()**, archivo **tetris.js**

12.6. Dibujar una figura

- Verifica si el tablero está cargado
- Verifica que la pieza no se salga el tablero
- Carga la imagen en el cuadro

- Regresa el valor en xBorrar
- Si el valor del arreglo es cero, colocamos la imagen blanca

```
279 function dibujarPieza() {
280   //
281   //Descripción: Dibuja pieza
282   //Parámetros: -Ninguno-
283   //Variables locales: k, X, Y
284   //Variables globales: nCuadros, curX, curY, dx, dy, piezaActual
285   //
286   var k, X, Y;
287   //
288   //Verifica si está cargado el tablero
289   //
290   if (document.images && tableroCargado) {
291     //
292     //Recorrer el arreglo
293     //
294     console.log("dibujarPieza",piezaActual);
295     for(k=0; k>nCuadros;k++){
296       //
297       //Recupermos valores
298       //
299       X = curX + dx[k];
300       Y = curY + dy[k];
301       //
302       //Verifica coordenadas
303       //
304       if (Y>=0 && Y<tableroAltura && X>=0 &&
305           X<tableroAncho && f[Y][X]!=-piezaActual) {
306         //
307         //dibujar pieza
308         //
309         eval("self.document.s"+Y+"_"+X+".src=Img"+piezaActual+".src");
310         //
311         //Almacenamos el valor de la pieza
312         //
313         f[Y][X]=-piezaActual;
314       }
315       //
316       //Regresa valor a borrar
317       //
318       X = xBorrar[k];
319       Y = yBorrar[k];
320       //
321       //Si el valor anterior es cero, lo "borramos"
322       //
323       if (f[Y][X]==0) {
324         eval("self.document.s"+Y+"_"+X+".src=Img0.src");
325       }
326     }
327   }
```

```
328 }
```

Listado 12.6.1. Mueve pelotita

12.7. Función Jugar

- Verifica si la pieza está en movimiento (abajo)
- Repite la función
- Si ya llegó al “tope”
- Redibuja la matrix
- Remover línea
- Determina el fin del juego
- Si no es el final -> continua
- Si es el final -> confirma

```
234 function jugar() {
235   //
236   //Descripción: ejecuta el juego
237   //Parámetros: ninguno
238   //Variables locales: activeL_, activeU_, activeR_, activeD_=0;
239   //Variables globales: timerID, velocidad, lineaLimite
240   //Funciones: moverAbajo(), jugar(), redibujarMatrix(),
241   //           removerLineas(), obtenerPieza()
242   //Salida:
243   //
244   console.log("jugar");
245   if (moverAbajo()) {
246     timerID=setTimeout("jugar()",velocidad);
247     return;
248   } else {
249     redibujarMatrix();
250     removerLineas();
251     if (lineaLimite>0 && obtenerPieza()) {
252       timerID=setTimeout("jugar()",velocidad);
253       return;
254     } else {
255       activeL_=0; activeU_=0;
256       activeR_=0; activeD_=0;
257       if (confirm("Fin del juego\n\n¿Volver a intentarlo?")) {
258         inicio();
259       } else {
260         if (finJuego) {
261           inicio();
262         } else {
263           self.close();
```

```
264         }
265       }
266     }
267   }
268 }
```

Listado 12.7.1. Fin del juego

12.8. Validar pieza

- validarPieza()
- borrarPieza()
- redibujarMatrix()

```
410 function validarPieza(X,Y) {
411   //
412   // Descripción: validar una pieza
413   //parámetros: coordenas x,y
414   //variable globales: nCuadros,dx_,dy_
415   //variables locales: miX, miY, k
416   //regresa: 1 si no se "sale" del tablero
417   //         0 si se "sale" del tablero
418   //
419   console.log("validarMatrix");
420   for (var k=0;k<nCuadros;k++) {
421     //Suma la distancia
422     miX=X+dx_[k];
423     miY=Y+dy_[k];
424     //
425     //Se sale por la izquierda o arriba
426     if (miX<0 || miX>=tableroAncho || miY>=tableroAltura) return 0;
427     //Se sale por la derecha o abajo
428     if (miY>-1 && f[miY][miX]>0) return 0;
429   }
430   //
431   //Si no se sale, regresa 1
432   //
433   return 1;
434 }
```

Listado 12.8.1. Función validaPieza()

```
388 function borrarPieza() {
389   //
390   //Descripción: remueve líneas
391   //Parámetros: ninguno
392   //Variables locales:
```

```
393   //Variables globales:
394   //Salida:
395   //
396   console.log("borrarPieza");
397   if (document.images && tableroCargado) {
398     for (var k=0;k<nCuadros;k++) {
399       X=curX+dx[k];
400       Y=curY+dy[k];
401       if (0<=Y && Y<tableroAltura && 0<=X && X<tableroAncho) {
402         xBorrar[k]=X;
403         yBorrar[k]=Y;
404         f[Y][X]=0;
405       }
406     }
407   }
408 }
```

Listado 12.8.2. Función borraPieza()

```
284 function redibujarMatrix() {
285   //
286   //Descripción: rellena la matrix
287   //Parámetros: ninguno
288   //Variables locales:
289   //Variables globales:
290   //Salida:
291   //
292   console.log("redibujarMatrix");
293   for (var k=0;k<nCuadros;k++) {
294     //
295     //Próximo cuadro
296     //
297     X=curX+dx[k];
298     Y=curY+dy[k];
299     //
300     //Verifica valores en frontera
301     //
302     if (0<=Y && Y<tableroAltura && 0<=X && X<tableroAncho) {
303       //
304       //Rellena cuadro
305       //
306       f[Y][X]=piezaActual;
307       //
308       //Línea límite
309       //
310       if (Y<lineaLimite) lineaLimite=Y;
311     }
312   }
313   top.focus();
314 }
```

Listado 12.8.3. Función redibujarMatrix()

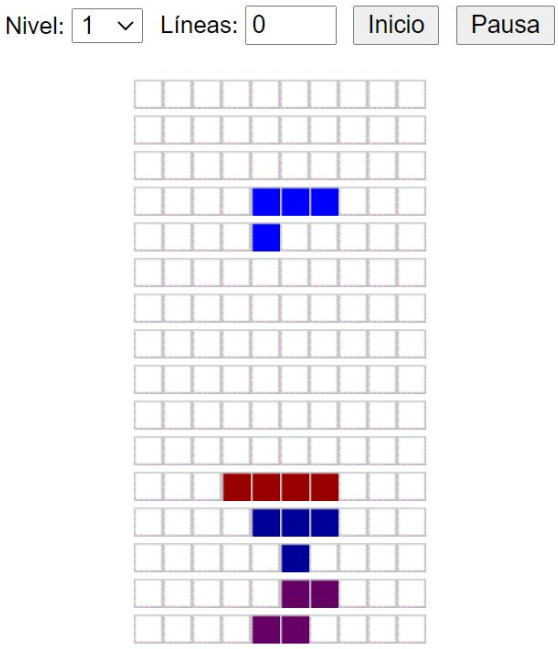

Imagen 12.8.1. Desarrollo del juego

12.9.Funciones del movimiento

En esta clase escribiremos las siguientes funciones:

- moverIzquierda()
- moverDerecha()
- bajar()
- moverAbajo()
- rotar()

```
467  function moverIzquierda() {
468    //
469    //Descripción: mueve izquierda
470    //Parámetros: ninguno
471    //Variables locales:
472    //Variables globales:
473    //Salida:
474    //
475    for (var k=0;k<nCuadros;k++) {
476      dx_[k]=dx[k];
```

```
477     dy_[k]=dy[k];
478   }
479   if (validarPieza(curX-1,curY)) {
480     borrarPieza();
481     curX--;
482     dibujarPieza();
483   }
484 }
485 
486 function moverDerecha() {
487   //
488   //Descripción: mueve pieza a la derecha
489   //Parámetros: -
490   //Variables locales: k
491   //Variables globales: nCuadros, dx, dy, dx_, dy_
492   //Salida:
493   //
494   for (var k=0;k<nCuadros;k++) {
495     dx_[k]=dx[k];
496     dy_[k]=dy[k];
497   }
498   if (validarPieza(curX+1,curY)) {
499     borrarPieza();
500     curX++;
501     dibujarPieza();
502   }
503 }
504 
505 function rotar() {
506   //
507   //Descripción: rotar
508   //Parámetros: ninguno
509   //Variables locales:
510   //Variables globales: nCuadros, dx, dy, dx_, dy_
511   //Salida:
512   //
513   for (var k=0;k<nCuadros;k++) {
514     dx_[k]=dy[k];
515     dy_[k]=-dx[k];
516   }
517   if (validarPieza(curX,curY)) {
518     borrarPieza();
519     for (var k=0;k<nCuadros;k++) {
520       dx[k]=dx_[k];
521       dy[k]=dy_[k];
522     }
523     dibujarPieza();
524   }
525 }
526 
527 function moverAbajo() {
528   //
529   //Descripción: mover abajo
530   //Parámetros: ninguno
```

```
531   //Variables locales: k
532   //Variables globales: nCuadros, dx_[k], dx[k], dy_[k], dy[k], curX, curY
533   //Salida:
534   //
535   console.log("moverAbajo",curY);
536   var k;
537   for (k=0;k<nCuadros;k++) {
538     dx_[k]=dx[k];
539     dy_[k]=dy[k];
540   }
541   if (validarPieza(curX,curY+1)) {
542     borrarPieza();
543     curY++;
544     dibujarPieza();
545     return 1;
546   }
547   return 0;
548 }
549
550 function bajar() {
551   //
552   //Descripción: bajar la pieza
553   //Parámetros: ninguno
554   //Variables locales: k
555   //Variables globales: nCuadros, dx_[k], dx[k], dy_[k], dy[k],
556   //                    curX, curY, timeID
557   //Salida:
558   //
559   var k;
560   for (k=0;k<nCuadros;k++) {
561     dx_[k]=dx[k];
562     dy_[k]=dy[k];
563   }
564   if (!validarPieza(curX,curY+1)) return;
565   clearTimeout(timerID);
566   borrarPieza();
567   while (validarPieza(curX,curY+1)) curY++;
568   dibujarPieza();
569   timerID=setTimeout("jugar()",velocidad);
570 }
```

Listado 12.9.3. Funciones de movimiento

12.10. Detección de teclas

Podemos realizar el movimiento de las piezas del juego mediante letras, números (cuando esté activado el teclado numérico) o con las flechas.

Debemos evitar que las figuras se “deslicen” por medio de una función setTimeout para cada tecla.

```
631 /*********************
632 // Manejo de teclas
633 **********************/
634 retrasoInicial_=200;
635 retrasoRepite_=20;
636 //
637 //Left, Right, Up, Down
638 //
639 activeL_=0; timerL_ = null;
640 activeR_=0; timerR_ = null;
641 activeU_=0; timerU_ = null;
642 activeD_=0; timerD_ = null;
643 activeSp=0; timerSp = null;
644 //
645 //Valores ASCII de las teclas
646 //
647 LeftIE_ =' 37 52 100 ';
648 //37  flecha a la izquierda
649 //52  4
650 //100 d
651 RightIE_=' 39 54 102 ';
652 //39  flecha a la derecha
653 //54  6
654 //102 f
655 UpIE_   =' 38 56 53 104 101 ';
656 //38  flecha arriba
657 //56  8
658 //53  5
659 //104 h
660 //101 e
661 DownIE_ =' 40 50 98 ';
662 //40  flecha abajo
663 //50  2
664 //98  b
665 SpaceIE_=' 32 ';
666 //32 barra espaciadora
667 //
668 function pulsarTecla(e) {
669   //
670   //Descripción:
671   //Parámetros: ninguno
672   //Variables locales:
673   //Variables globales:
674   //Salida:
675   //
676   var key_=0;
677   var evt = e ? e:event;
678   //
679   key_=evt.keyCode;
680   console.log("ASCII",key_);
681   if (!inicioJuego || !tableroCargado || pausaJuego) return;
682   //
```

```
683   //Mover a la izquierda
684   //
685   if (!activeL_ && LeftIE_.indexOf(' '+key_+' ')!=-1){
686     activeL_ = 1;
687     activeR_ = 0;
688     moverIzquierda();
689     timerL_=setTimeout("slideL_()",retrasoInicial_);
690   }
691   //
692   //Mover a la derecha
693   //
694   if (!activeR_ && RightIE_.indexOf(' '+key_+' ')!=-1) {
695     activeR_ = 1;
696     activeL_ = 0;
697     moverDerecha();
698     timerR_=setTimeout("slideR_()",retrasoInicial_);
699   }
700   //
701   //Rotar
702   //
703   if (!activeU_ && UpIE_.indexOf(' '+key_+' ')!=-1) {
704     activeU_ = 1;
705     activeD_ = 0;
706     rotar();
707   }
708   //
709   //Barra espaciadora
710   //
711   if (!activeSp && SpaceIE_.indexOf(' '+key_+' ')!=-1){
712     activeSp = 1;
713     activeD_ = 0;
714     bajar();
715   }
716   //
717   //Flecha hacia abajo
718   //
719   if (!activeD_ && DownIE_.indexOf(' '+key_+' ')!=-1) {
720     activeD_ = 1
721     activeU_ = 0
722     moverAbajo();
723     timerD_=setTimeout("slideD_()",retrasoInicial_);
724   }
725 }
726
727 function levantarTecla(e) {
728   //
729   //Descripción: levantar tecla
730   //Parámetros: el objeto de la tecla
731   //Variables locales:
732   //Variables globales:
733   //Salida:
734   //
735   //var KeyNN_=0;
736   var key_=0;
```

```
737
738   var evt = e?e:event;
739   key_=evt.keyCode;
740
741   if (LeftIE_.indexOf(' '+key_+' ')!=-1)  {
742     activeL_=0; clearTimeout(timerL_);
743   }
744   if (RightIE_.indexOf(' '+key_+' ')!=-1) {
745     activeR_=0; clearTimeout(timerR_);
746   }
747   if (UpIE_.indexOf(' '+key_+' ')!=-1){
748     activeU_=0; clearTimeout(timerU_);
749   }
750   if (DownIE_.indexOf(' '+key_+' ')!=-1)  {
751     activeD_=0; clearTimeout(timerD_);
752   }
753   if (SpaceIE_.indexOf(' '+key_+' ')!=-1) {
754     activeSp=0; clearTimeout(timerSp);
755   }
756   top.focus();
757 }
758
759 function slideL_() {
760   //
761   //Descripción: desliza a la izquierda
762   //Parámetros: ninguno
763   //Variables locales:
764   //Variables globales:
765   //Salida:
766   //
767   if (activeL_) {
768     moverIzquierda();
769     timerL_=setTimeout("slideL_()",retrasoRepite_);
770   }
771 }
772
773 function slideR_() {
774   //
775   //Descripción: desliza derecha
776   //Parámetros: ninguno
777   //Variables locales:
778   //Variables globales:
779   //Salida:
780   //
781   if (activeR_) {
782     moverDerecha();
783     timerR_=setTimeout("slideR_()",retrasoRepite_);
784   }
785 }
786
787 function slideD_() {
788   //
789   //Descripción: desliza abajo
790   //Parámetros: ninguno
```

```
791   //Variables locales:
792   //Variables globales:
793   //Salida:
794   //
795   if (activeD_) {
796     moverAbajo();
797     timerD_=setTimeout("slideD_()",retrasoRepite_);
798   }
799 }
```

Listado 12.10.3. Funciones de movimiento

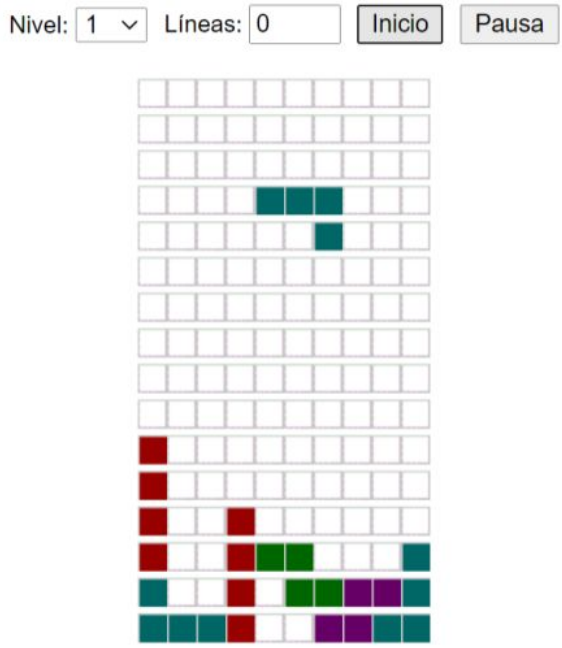

Imagen 12.10.1. Movimiento de piezas por el teclado

12.11. Manejo del ratón

- Detectamos la celda seleccionada y pasamos sus coordenadas.
- Detectamos los valores máximos y mínimos y realizamos los movimientos derecha, izquierda o abajo.
- Rotamos si se pulsó arriba.

```
607 /*****************************
608  Proceso del click del ratón
609 *****************************/
610 function obtenerMinMax() {
```

```
611   //
612   //Llamada desde: Obtener máximo y mínimo
613   //Variables globales: curX, curY, nCuadros, dx, dy
614   //Variables de trabajo: xMax, xMin, yMax
615   //Variables de salida: xMax, xMin, yMax
616   //
617   xMax=curX;
618   xMin=curX;
619   yMax=curY;
620   for (var k=1;k<nCuadros;k++) {
621     if (curX+dx[k]>xMax) xMax=curX+dx[k];
622     if (curX+dx[k]<xMin) xMin=curX+dx[k];
623     if (curY+dy[k]>yMax) yMax=curY+dy[k];
624   }
625 }
626
627 function seleccionar(yClk,xClk) {
628   //
629   //Descripción: Detecta si ya inició el juego y si el tablero
630   //ha sido cargado
631   //Llamada desde: seleccionar()
632   //Variables globales:
633   //Variables de trabajo:
634   //Variables de salida:
635   //
636   if (!inicioJuego || !tableroCargado) return;
637   //
638   //Si el juego ha sido pausado
639   //
640   if (pausaJuego) reanudarJuego();
641   //
642   //Pone el foco en el frame "top"
643   //
644   top.focus();
645   //
646   //
647   obtenerMinMax();
648   //
649   if (yClk>yMax) {moverAbajo(); return;}
650   if (xClk<xMin) {moverIzquierda(); return;}
651   if (xClk>xMax) {moverDerecha(); return;}
652   rotar(); return;
653 }
```

Listado 12.11.1. Funciones de movimiento del ratón

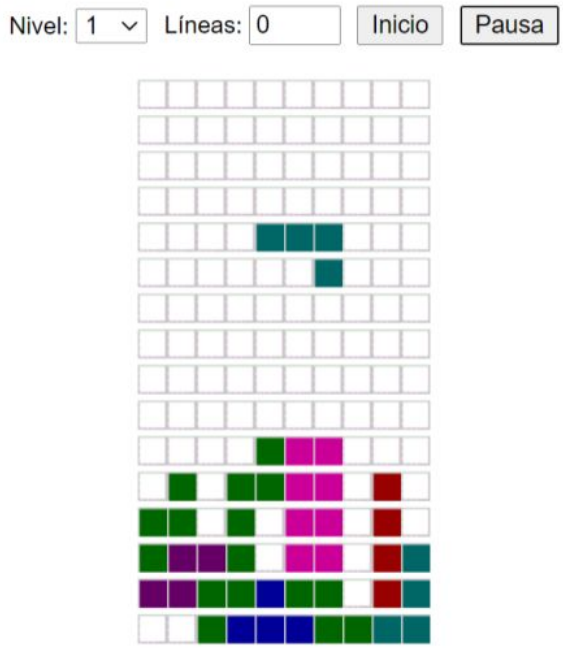

Imagen 12.11.1. Juego terminado