Technical Manual

About GameZoneRtx

By Alan Gómez Mireles

20130820

Mail: alanyahir63@gmail.com

Contact: 8714153150

Play games…

Advanced programming topics

May 17 , 2022

INDEX

Contenido

Introduction

As part of the project unit 3 of the subject Advanced Topics in Programming, we were asked to develop a client-server program, to which my project ZoneGameRtx arises as a response.

Objective

The objective of this manual is to inform the user interested in the realization of a program using servlet and local or remote servers, the concepts, steps, classes and things necessary for its realization, as well as a guide of steps to know how to realize this specific program.

System Requirements

The necessary system requirements are:

Hardware

- Keyboard, mouse, monitor
- Processor 1.4 Ghz minimum
- RAM memory min 2GB

Software

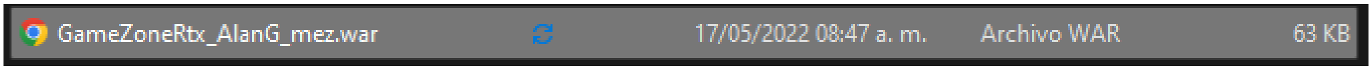
- 15 mb memory space
- Java 8.0

Other requirements

- Have installed Apache Tomcat 8.0.27.0
- Have Netbeans 8.2 installed
- Have a browser installed
- Optional (Xampp control panel v3.3.0)

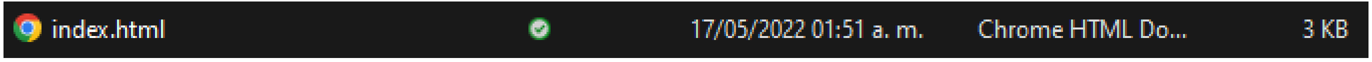
Processing

To enter the program it is necessary to download the program file, which contains a .war file that will contain our index as well as our games, made for html.
All is content into de carpet dist:



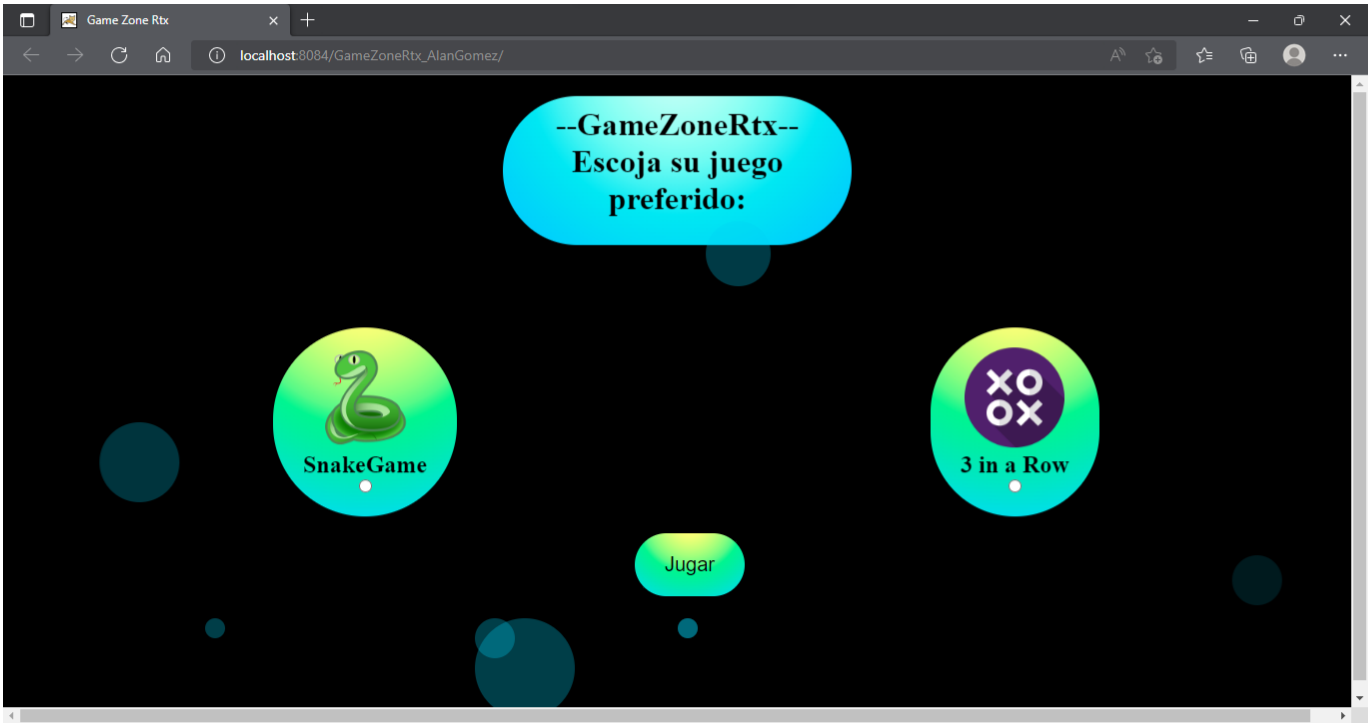
GameZoneRtx_AlanG_mez.war 17/05/2022 08:47 a. m. Archivo WAR 63 KB

Open our file index.html



index.html 17/05/2022 01:51 a. m. Chrome HTML Do... 3 KB

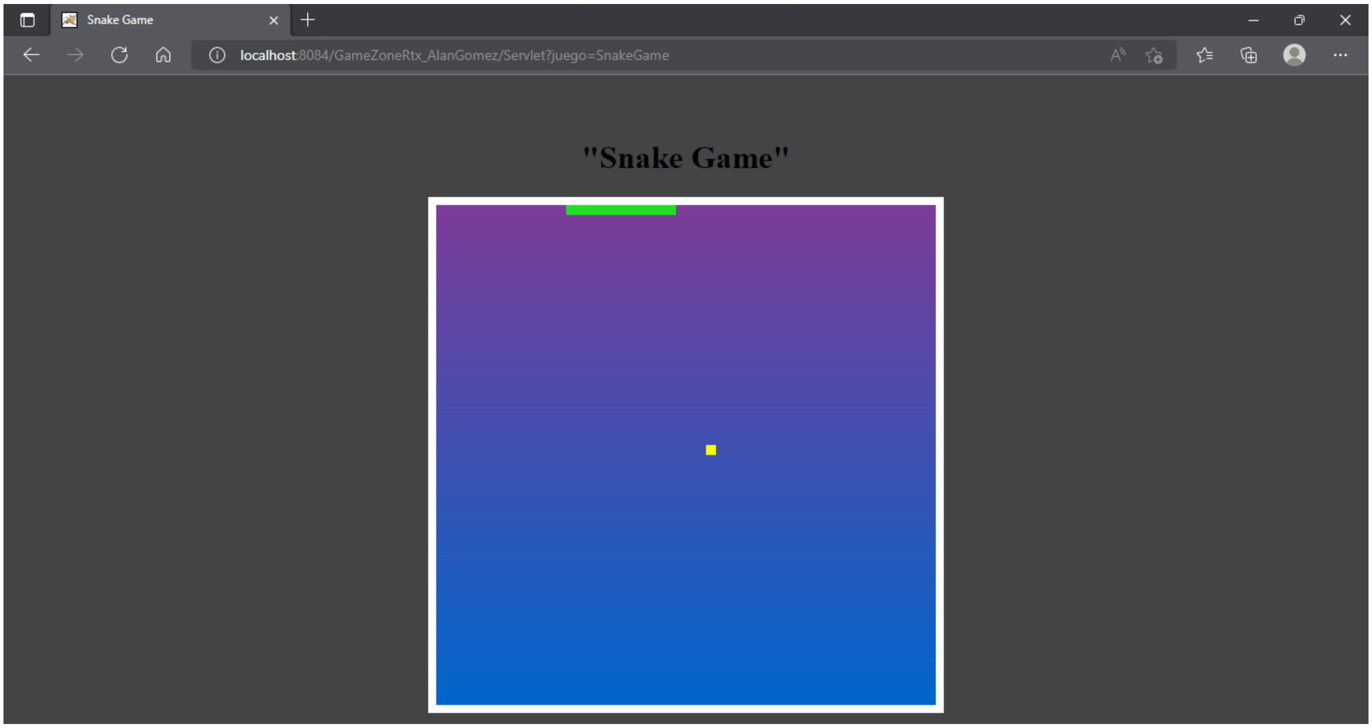
Our principal page is that:



--GameZoneRtx--
Escoja su juego preferido:

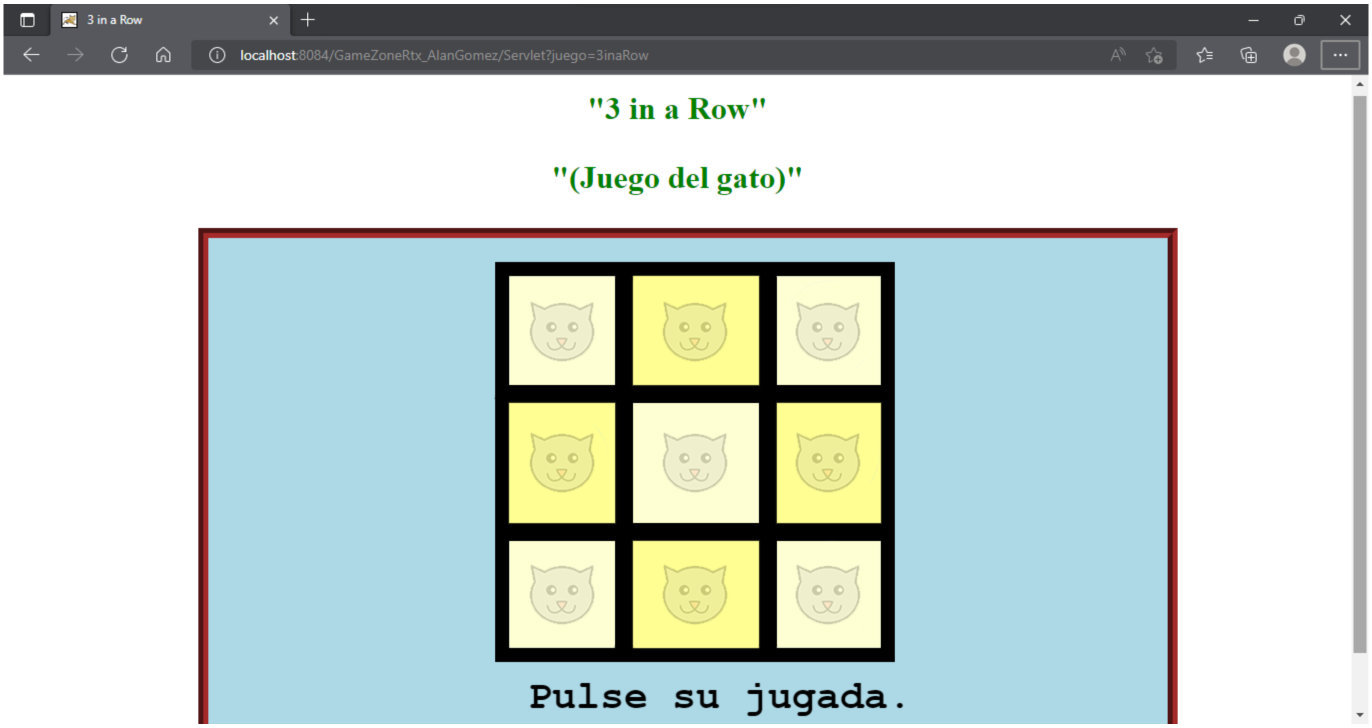
Once there, choose your favorite game by clicking on the circles below the titles, and then click on the play button.

Which will direct us to our chosen game:


Snake Game
localhost:8084/GameZoneRtx_AlanGomez/Servlet?juego=SnakeGame

"Snake Game"

Juego de la serpiente (Snake Game)



"3 in a Row"

"(Juego del gato)"

Pulse su jugada.

Juego del gato (3 in a Row)

How to play the Snake Game?

To play the snake game it is necessary to perform the basic movements with the keyboard, using the letters a,s,d,w which correspond to a=left, s=down, d=right and w=up.

The method of the game is simple, it is about directing the snake to the point so that it eats and grows but without hitting the edge of the container or colliding with itself because then it will lose.

How to play 3 in a Row?

To play the 3 in a row game, it is necessary to click on one of the 9 squares shown, to give your move, before this, the server will make random moves to try to win.

The game ends if you win, draw or lose, to win it is necessary to have 3 in a row, it is a draw if you don't win or lose, and it is lost if the server generates 3 in a row. To play again, press the button to play again. In case you get stuck, click on the replay button.

Necessary files:

Files .html:

- Index.html
- Index.css
- 3inaRow.html
- SnakeGame.html

Files .png:

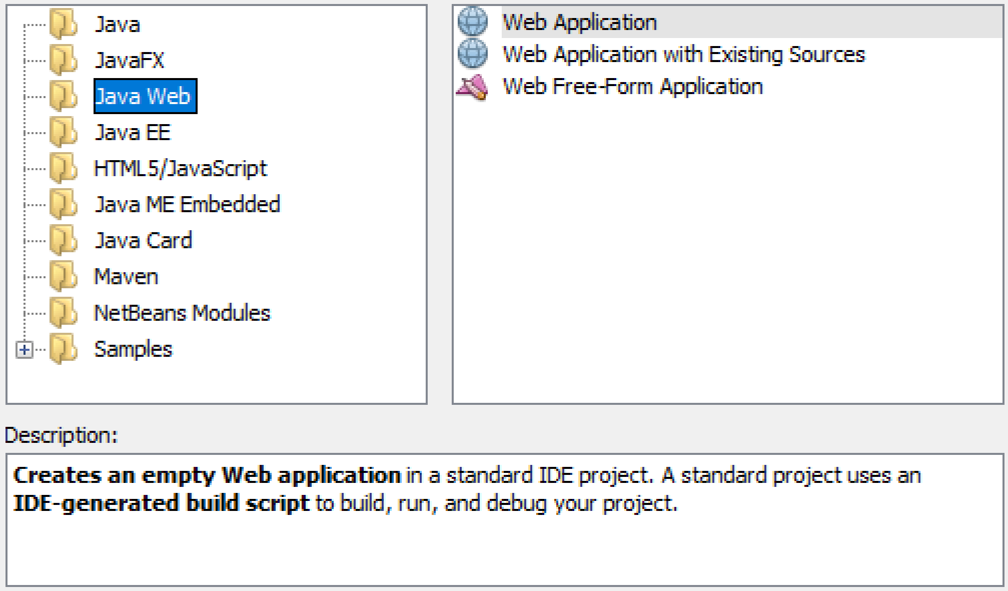
- Tablero.png

Package Servlet:

- Servlet.java

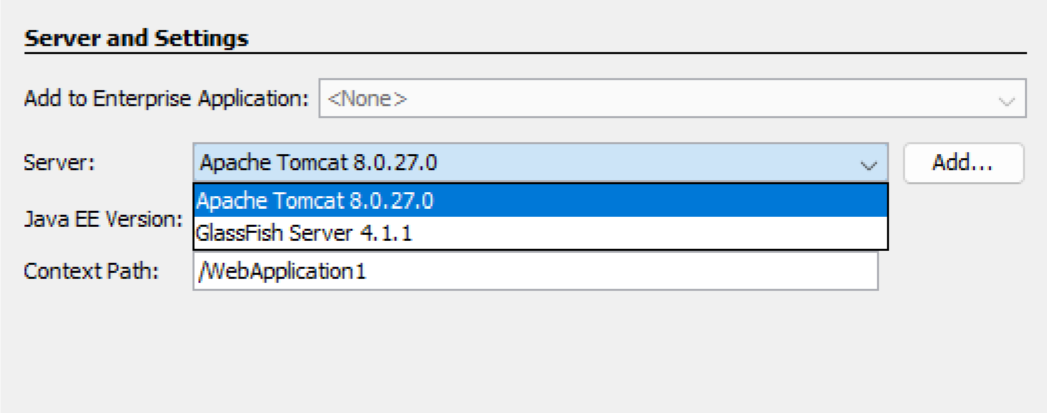
¿How to create a program that this type?

For the creation of this program it is necessary to create a java file of web type:



Creates an empty Web application in a standard IDE project. A standard project uses an **IDE-generated build script** to build, run, and debug your project.

Then search the server to use:

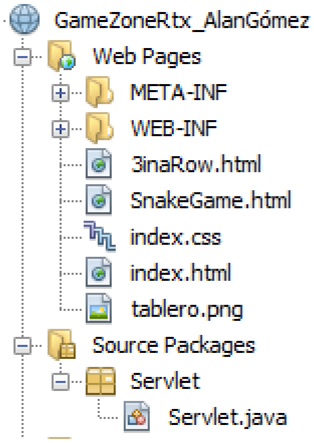


Add to Enterprise Application: <None>

Server: Apache Tomcat 8.0.27.0

Context Path: /WebApplication1

And we will give finalize, which will create a template that will contain an index.html:



```
GameZoneRtx_AlanGómez
  Web Pages
    META-INF
    WEB-INF
    3inaRow.html
    SnakeGame.html
    index.css
    index.html
    tablero.png
  Source Packages
    Servlet
      Servlet.java
```

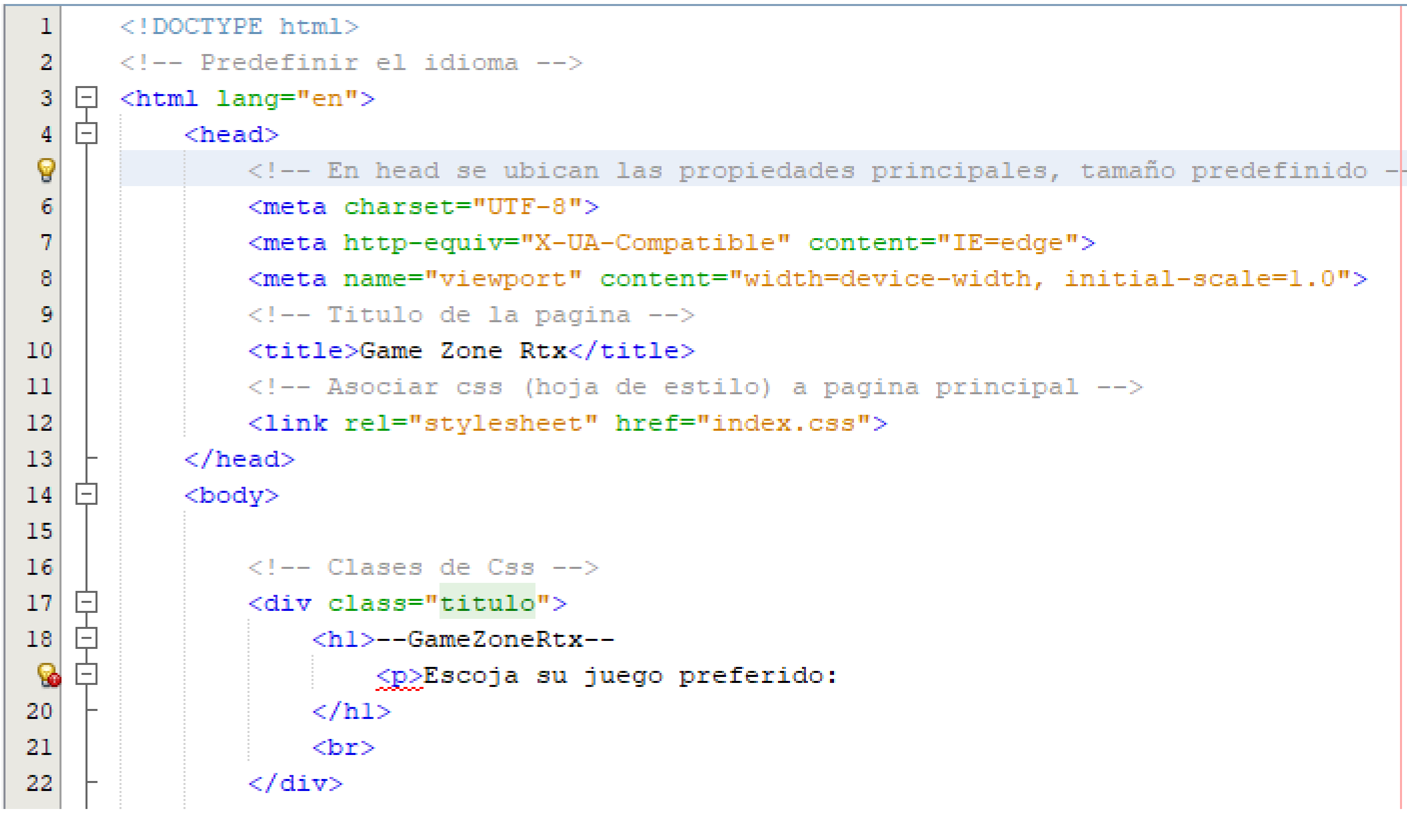
The html of the games, must be imported from the download folder, or in case you want to make them, import their html.

The index.css file must be imported from the download folder, or in case you want to make your own games, evaluate your design and import their files.

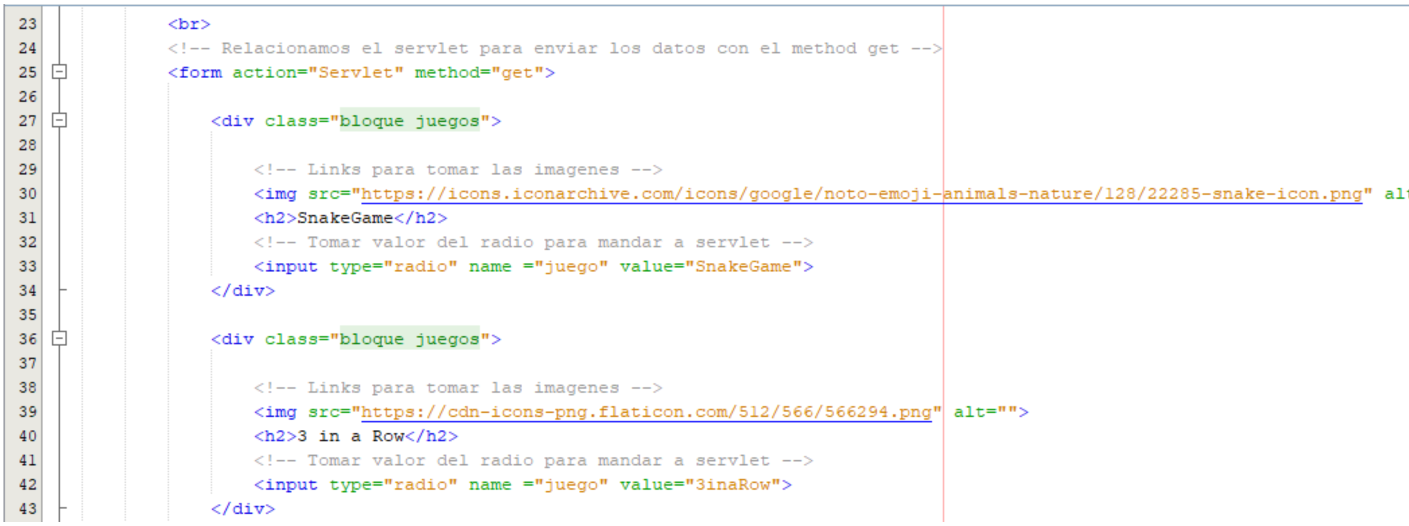
The Servlet package must be created, and create a servlet type class, which will contain the template to make a httppServlet.

The codes are shown below:

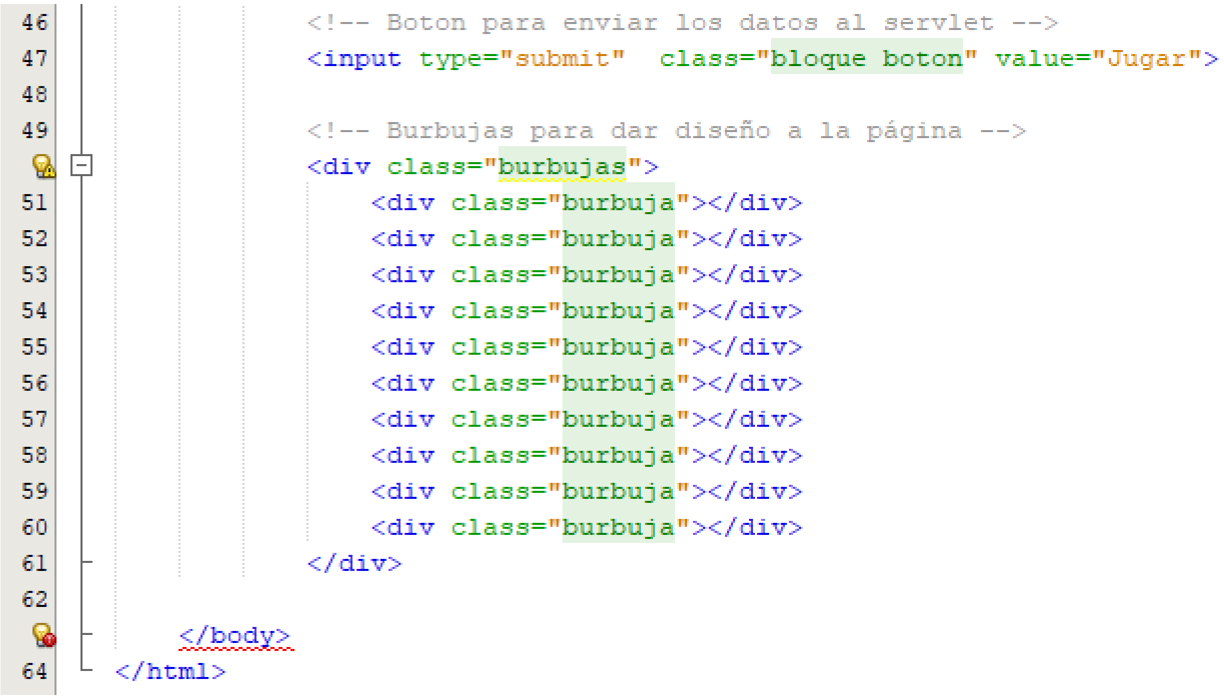
Index.hmt Code:



```
<!DOCTYPE html>
<!-- Predefinir el idioma -->
<html lang="en">
    <head>
        <!-- En head se ubican las propiedades principales, tamaño predefinido --
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <!-- Titulo de la pagina -->
        <title>Game Zone Rtx</title>
        <!-- Asociar css (hoja de estilo) a pagina principal -->
        <link rel="stylesheet" href="index.css">
    </head>
    <body>

        <!-- Clases de Css -->
        <div class="titulo">
            <h1>--GameZoneRtx--
                <p>Escoja su juego preferido:
            </h1>
            <br>
        </div>
```

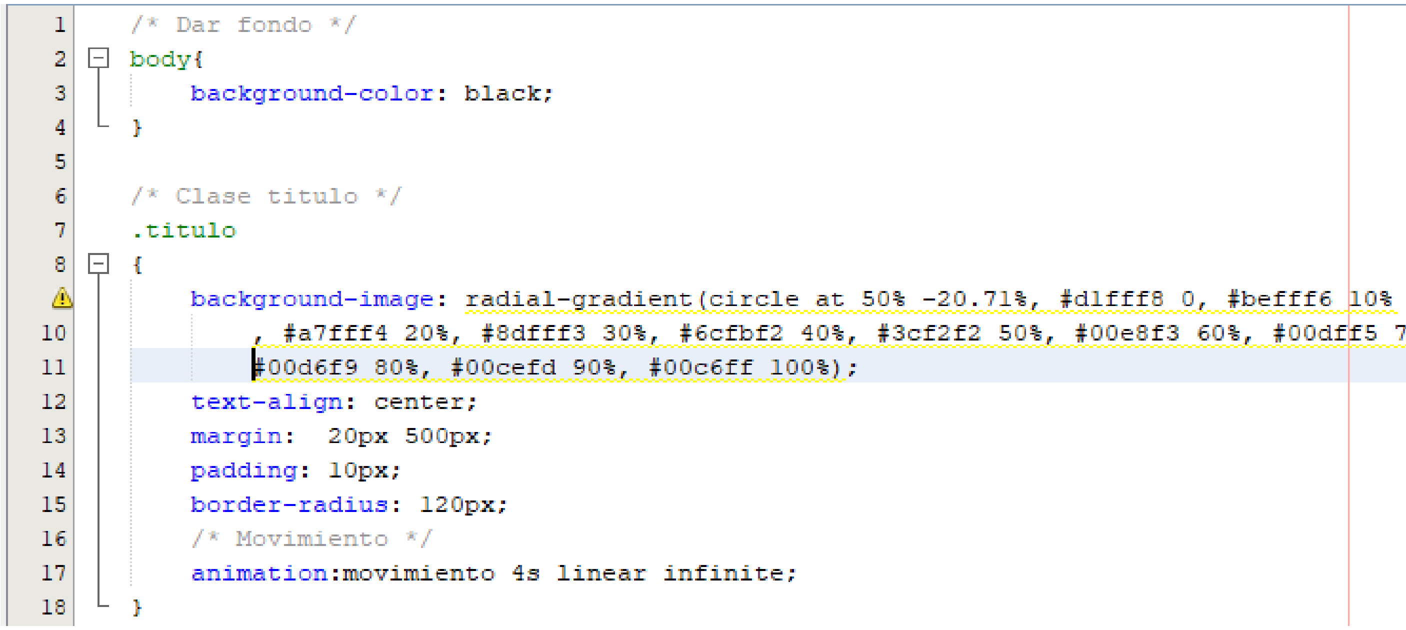


```
        <br>
        <!-- Relacionamos el servlet para enviar los datos con el method get -->
        <form action="Servlet" method="get">

            <div class="bloque juegos">

                <!-- Links para tomar las imagenes -->
                <img src="https://icons.iconarchive.com/icons/google/noto-emoji-animals-nature/128/22285-snake-icon.png" al
                <h2>SnakeGame</h2>
                <!-- Tomar valor del radio para mandar a servlet -->
                <input type="radio" name ="juego" value="SnakeGame">
            </div>

            <div class="bloque juegos">

                <!-- Links para tomar las imagenes -->
                <img src="https://cdn-icons-png.flaticon.com/512/566/566294.png" alt="">
                <h2>3 in a Row</h2>
                <!-- Tomar valor del radio para mandar a servlet -->
                <input type="radio" name ="juego" value="3inaRow">
            </div>
```

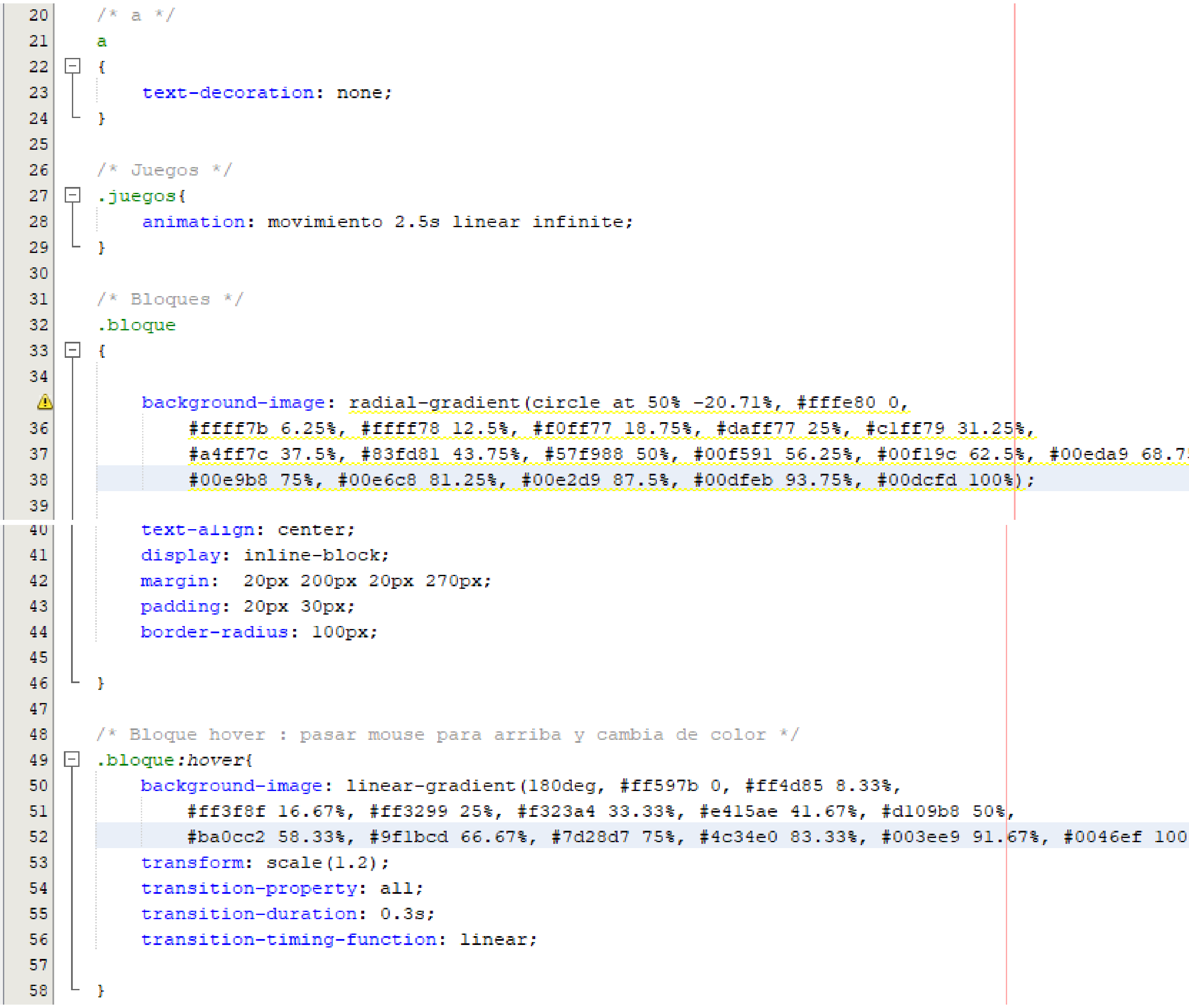


```html
            <!-- Boton para enviar los datos al servlet -->
            <input type="submit"  class="bloque boton" value="Jugar">

            <!-- Burbujas para dar diseño a la página -->
            <div class="burbujas">
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
                <div class="burbuja"></div>
            </div>

    </body>
</html>
```

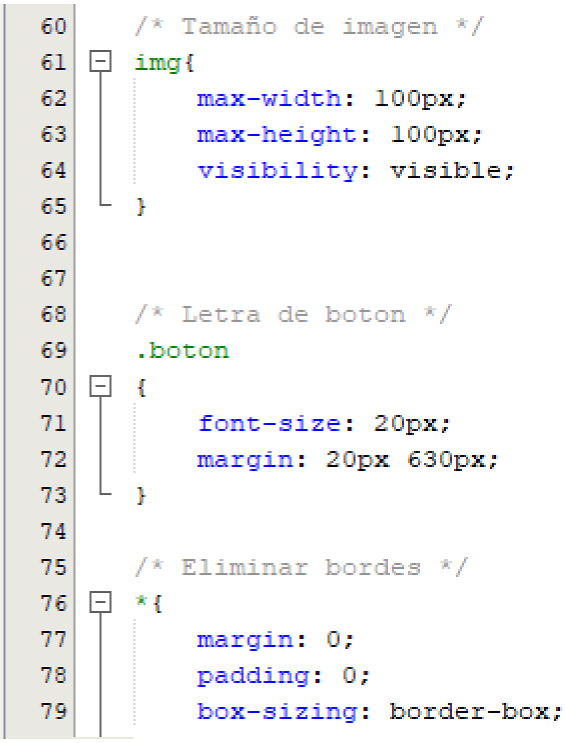
Index.css Code:



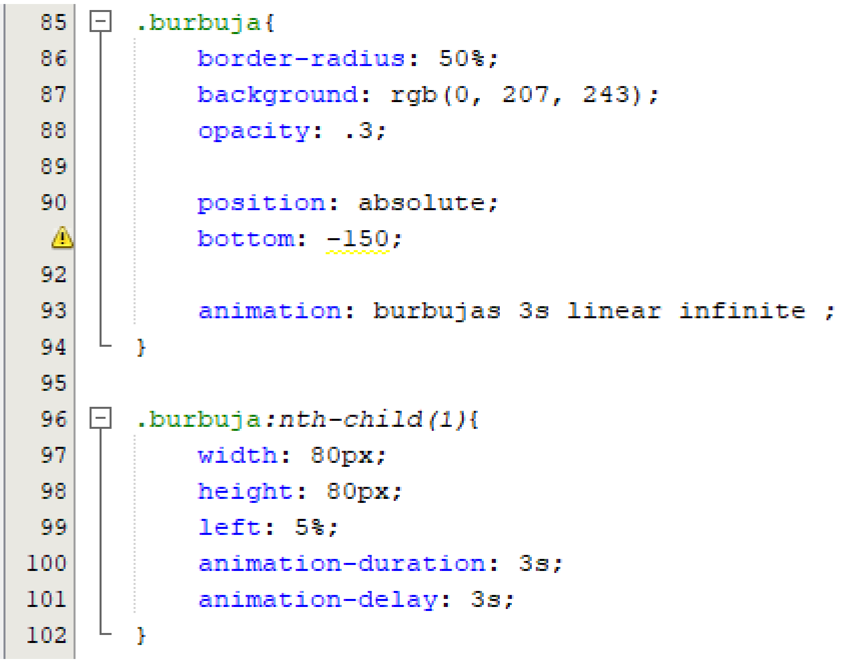
```css
/* Dar fondo */
body{
    background-color: black;
}

/* Clase titulo */
.titulo
{
    background-image: radial-gradient(circle at 50% -20.71%, #d1fff8 0, #befff6 10%
        , #a7fff4 20%, #8dfff3 30%, #6cfbf2 40%, #3cf2f2 50%, #00e8f3 60%, #00dff5 7
        #00d6f9 80%, #00cefd 90%, #00c6ff 100%);
    text-align: center;
    margin:  20px 500px;
    padding: 10px;
    border-radius: 120px;
    /* Movimiento */
    animation:movimiento 4s linear infinite;
}
```



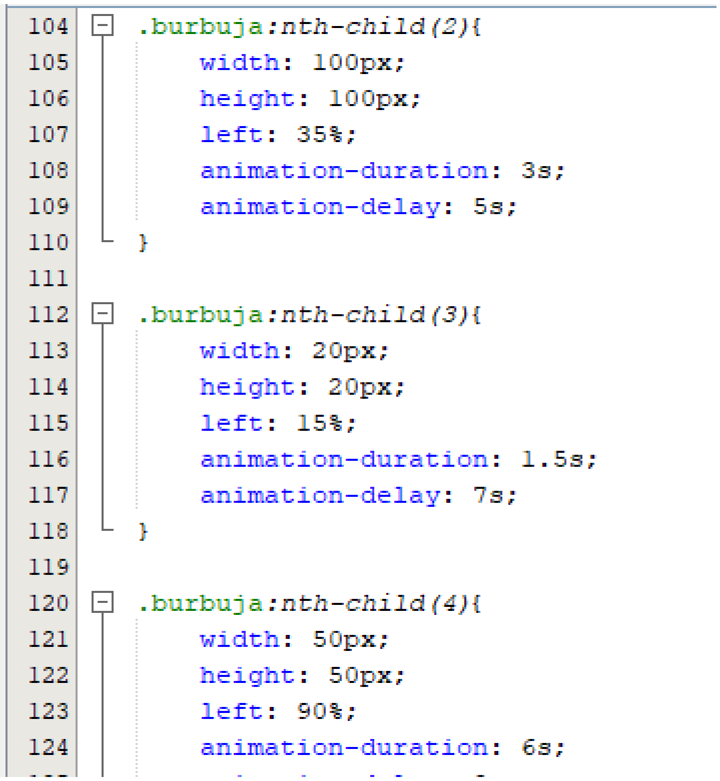
```css
/* a */
a
{
    text-decoration: none;
}

/* Juegos */
.juegos{
    animation: movimiento 2.5s linear infinite;
}

/* Bloques */
.bloque
{

    background-image: radial-gradient(circle at 50% -20.71%, #fffe80 0,
        #ffff7b 6.25%, #ffff78 12.5%, #f0ff77 18.75%, #daff77 25%, #c1ff79 31.25%,
        #a4ff7c 37.5%, #83fd81 43.75%, #57f988 50%, #00f591 56.25%, #00f19c 62.5%, #00eda9 68.7
        #00e9b8 75%, #00e6c8 81.25%, #00e2d9 87.5%, #00dfeb 93.75%, #00dcfd 100%);

    text-align: center;
    display: inline-block;
    margin:  20px 200px 20px 270px;
    padding: 20px 30px;
    border-radius: 100px;

}

/* Bloque hover : pasar mouse para arriba y cambia de color */
.bloque:hover{
    background-image: linear-gradient(180deg, #ff597b 0, #ff4d85 8.33%,
        #ff3f8f 16.67%, #ff3299 25%, #f323a4 33.33%, #e415ae 41.67%, #d109b8 50%,
        #ba0cc2 58.33%, #9f1bcd 66.67%, #7d28d7 75%, #4c34e0 83.33%, #003ee9 91.67%, #0046ef 100
    transform: scale(1.2);
    transition-property: all;
    transition-duration: 0.3s;
    transition-timing-function: linear;

}
```



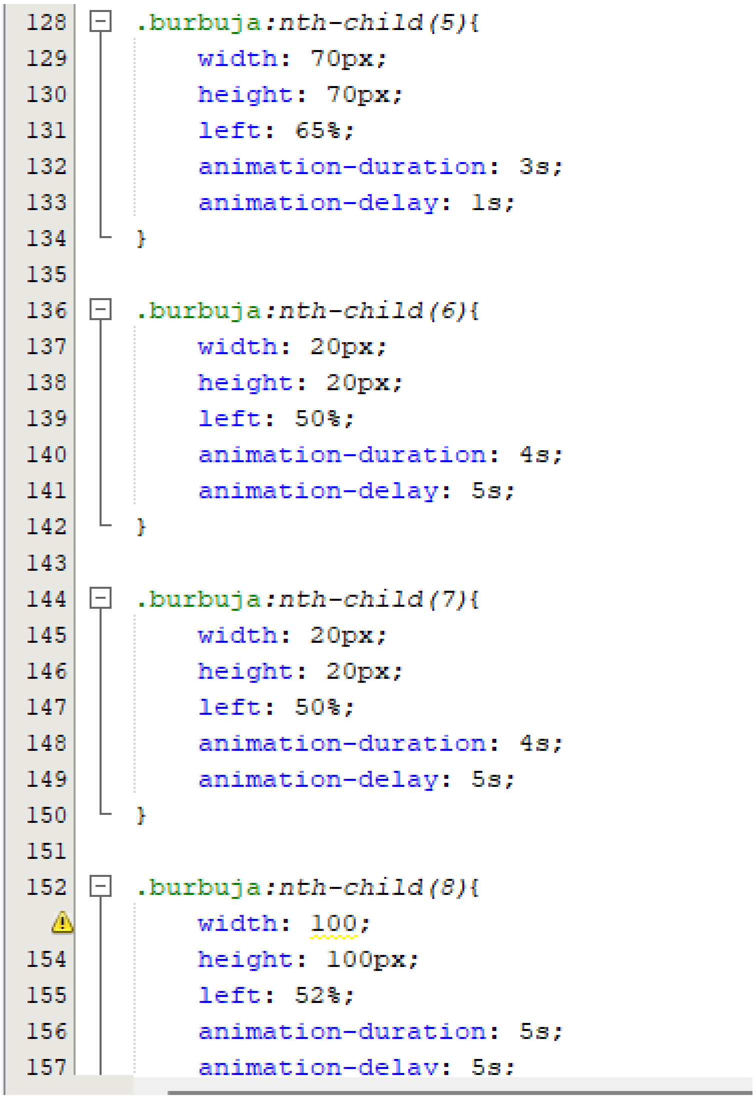
```css
/* Tamaño de imagen */
img{
    max-width: 100px;
    max-height: 100px;
    visibility: visible;
}


/* Letra de boton */
.boton
{
    font-size: 20px;
    margin: 20px 630px;
}

/* Eliminar bordes */
*{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
```



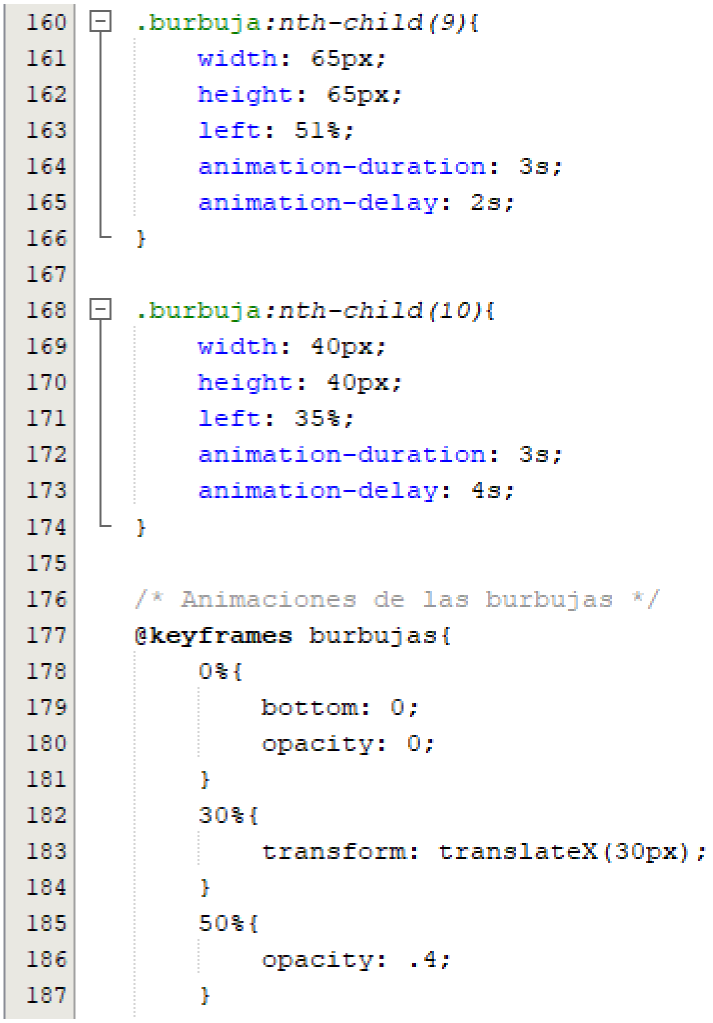
```css
.burbuja{
    border-radius: 50%;
    background: rgb(0, 207, 243);
    opacity: .3;

    position: absolute;
    bottom: -150;

    animation: burbujas 3s linear infinite ;
}

.burbuja:nth-child(1){
    width: 80px;
    height: 80px;
    left: 5%;
    animation-duration: 3s;
    animation-delay: 3s;
}
```



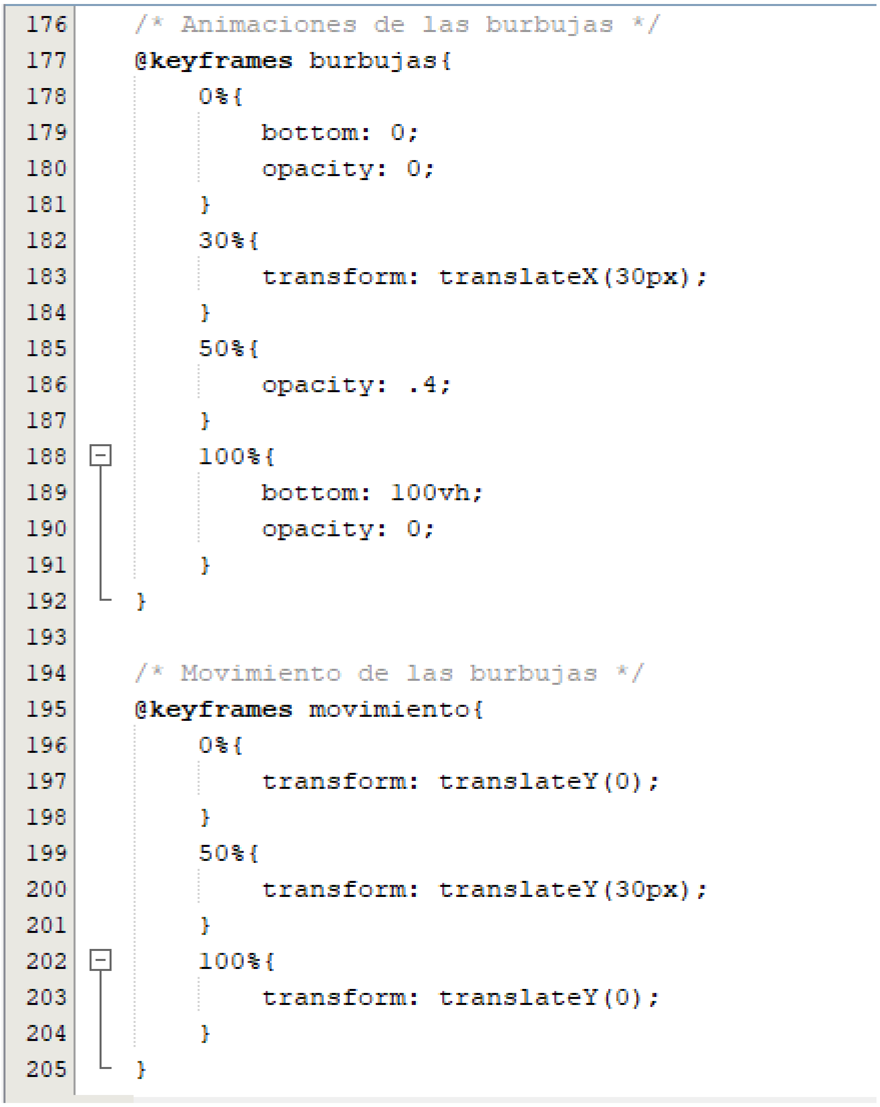
```
.burbuja:nth-child(2){
    width: 100px;
    height: 100px;
    left: 35%;
    animation-duration: 3s;
    animation-delay: 5s;
}

.burbuja:nth-child(3){
    width: 20px;
    height: 20px;
    left: 15%;
    animation-duration: 1.5s;
    animation-delay: 7s;
}

.burbuja:nth-child(4){
    width: 50px;
    height: 50px;
    left: 90%;
    animation-duration: 6s;
```



```
.burbuja:nth-child(5){
    width: 70px;
    height: 70px;
    left: 65%;
    animation-duration: 3s;
    animation-delay: 1s;
}

.burbuja:nth-child(6){
    width: 20px;
    height: 20px;
    left: 50%;
    animation-duration: 4s;
    animation-delay: 5s;
}

.burbuja:nth-child(7){
    width: 20px;
    height: 20px;
    left: 50%;
    animation-duration: 4s;
    animation-delay: 5s;
}

.burbuja:nth-child(8){
    width: 100;
    height: 100px;
    left: 52%;
    animation-duration: 5s;
    animation-delay: 5s;
```



```css
.burbuja:nth-child(9){
    width: 65px;
    height: 65px;
    left: 51%;
    animation-duration: 3s;
    animation-delay: 2s;
}

.burbuja:nth-child(10){
    width: 40px;
    height: 40px;
    left: 35%;
    animation-duration: 3s;
    animation-delay: 4s;
}

/* Animaciones de las burbujas */
@keyframes burbujas{
    0%{
        bottom: 0;
        opacity: 0;
    }
    30%{
        transform: translateX(30px);
    }
    50%{
        opacity: .4;
    }
```



```
/* Animaciones de las burbujas */
@keyframes burbujas{
    0%{
        bottom: 0;
        opacity: 0;
    }
    30%{
        transform: translateX(30px);
    }
    50%{
        opacity: .4;
    }
    100%{
        bottom: 100vh;
        opacity: 0;
    }
}

/* Movimiento de las burbujas */
@keyframes movimiento{
    0%{
        transform: translateY(0);
    }
    50%{
        transform: translateY(30px);
    }
    100%{
        transform: translateY(0);
    }
}
```

Snake Game code:

Código

```
<!DOCTYPE html>
<!--
Juego de la serpiente en Html
-->
<html>
  <head>
    <title>Snake Game</title>
    <style>
      body {
        background: #444;
        text-align: center;
        margin: 64px 0 0 0;
      }
      canvas {
        border: solid 8px white;
        background: linear-gradient(
          0deg,
          rgba(0, 102, 204, 1.0) 0%,
          rgba(125, 60, 152, 1.0) 100%
          );
      }
    </style>
  </head>
  <body>
    <h1>"Snake Game"</h1>
    <!-- Los canvas se utilizan para dibujar gráficos a través de sencuencias de comandos -->
    <canvas width="500" height="500"></canvas>

    <!-- Los scripts nos ayudan a hacer referencia a un ejecutable en un documento html -->
    <script>
      const STATE_RUNNING = 1;
      const STATE_LOSING = 2;

      const TICK = 80;
      const SQUARE_SIZE = 10;
      const BOARD_WIDTH = 50;
      const BOARD_HEIGHT = 50;
      const GROW_SCALE = 10;
      const DIRECTIONS_MAP = {
        'A': [-1, 0],
  'D': [1, 0],
  'S': [0, 1],
  'W': [0, -1],
  'a': [-1, 0],
  'd': [1, 0],
  's': [0, 1],
  'w': [0, -1],
};

let state = {
  canvas: null,
  context: null,
  snake: [{x: 0, y: 0}],
  direction: {x: 1, y: 0},
  prey: {x: 0, y: 0},
  growing: 0,
  runState: STATE_RUNNING
};

function randomXY() {
  return {
    x: parseInt(Math.random() * BOARD_WIDTH),
    y: parseInt(Math.random() * BOARD_HEIGHT)
  };
}

function tick() {
  const head = state.snake[0];
  const dx = state.direction.x;
  const dy = state.direction.y;
  const highestIndex = state.snake.length - 1;
  let tail = {};
  let interval = TICK;

  Object.assign(tail,
      state.snake[state.snake.length - 1]);

  let didScore = (
      head.x === state.prey.x
      && head.y === state.prey.y
      );

  if (state.runState === STATE_RUNNING) {
    for (let idx = highestIndex; idx > -1; idx--) {
      const sq = state.snake[idx];

      if (idx === 0) {
        sq.x += dx;
        sq.y += dy;
      } else {
        sq.x = state.snake[idx - 1].x;
        sq.y = state.snake[idx - 1].y;
      }
    }
  } else if (state.runState === STATE_LOSING) {
    interval = 10;

    if (state.snake.length > 0) {
      state.snake.splice(0, 1);
    }

    if (state.snake.length === 0) {
      state.runState = STATE_RUNNING;
      state.snake.push(randomXY());
      state.prey = randomXY();
    }
  }

  if (detectCollision()) {
    state.runState = STATE_LOSING;
    state.growing = 0;
  }

  if (didScore) {
    state.growing += GROW_SCALE;
    state.prey = randomXY();
  }

  if (state.growing > 0) {
    state.snake.push(tail);
    state.growing -= 1;
  }

  requestAnimationFrame(draw);
  setTimeout(tick, interval);
}
```

```
function detectCollision() {
  const head = state.snake[0];

  if (head.x < 0
      || head.x >= BOARD_WIDTH
      || head.y >= BOARD_HEIGHT
      || head.y < 0
      ) {
    return true;
  }

  for (var idx = 1; idx < state.snake.length; idx++) {
    const sq = state.snake[idx];

    if (sq.x === head.x && sq.y === head.y) {
      return true;
    }
  }

  return false;
}

function drawPixel(color, x, y) {
  state.context.fillStyle = color;
  state.context.fillRect(
      x * SQUARE_SIZE,
      y * SQUARE_SIZE,
      SQUARE_SIZE,
      SQUARE_SIZE
      );
}

function draw() {
  state.context.clearRect(0, 0, 500, 500);

  for (var idx = 0; idx < state.snake.length; idx++) {
    const {x, y} = state.snake[idx];
    drawPixel('#22dd22', x, y);
  }

  const {x, y} = state.prey;
  drawPixel('yellow', x, y);
```

```
      }

      window.onload = function () {
        state.canvas = document.querySelector('canvas');
        state.context = state.canvas.getContext('2d');

        window.onkeydown = function (e) {
          const direction = DIRECTIONS_MAP[e.key];

          if (direction) {
            const [x, y] = direction;
            if (-x !== state.direction.x
                && -y !== state.direction.y)
            {
              state.direction.x = x;
              state.direction.y = y;
            }
          }
        }

        tick();
      };

    </script>
  </body>
</html>
```

3 in a Row code:

Código:

```
<!DOCTYPE html>
<!--
Juego del gato o 3 en raya creado en Html
-->
<html>
   <head>
      <meta charset="utf-8">
      <title>3 in a Row</title>
      <style type="text/css">
         body{
            width: 960px;
            margin: 0 auto;
         }
         h1{
            text-align: center;
            color: green;
         }
         #pantalla{
            border: groove 10px brown;
            background: lightblue;
         }
         #boton{
            background-color: red;
            color: white;
            font-size: 20px;
            text-align: center;
            font-weight: bolder;
            padding: 3px;
            border: solid 2px black;
         }
         #boton:hover{
            background-color: lightcoral;
            font-size: 22px;
         }

      </style>
      <script>
         var ctx, canvas;
         var fichas_array = new Array();
         var COLUMNAS = 3;
var RENGLONES = 3;
var fichas_X = 0;
var fichas_O = 0;
var tiradas = 0;
var gameOver = false;
var lados = 120;

window.onload = function () {
  /*Verificar canvas y ejecutarlo si esta bien o msj si error*/
  canvas = document.getElementById("pantalla");
  if (canvas && canvas.getContext) {
    ctx = canvas.getContext("2d");
    if (ctx) {
      /*Si canvas se ejecuto bien*/
      gato();
      mensaje("Pulse su jugada.");
      canvas.addEventListener("click", seleccionUsuario, false);
    } else {
      /*Si error*/
      alert("Error al crear el contexto!");
    }
  }
}

function gato() {
  /*Estas funciones hacen que se cargue la imagen del tablero en el canvas*/
  var imagen = new Image();
  function procesaImagen() {
    ctx.drawImage(imagen, 0, 0);
  }
  imagen.src = "tablero.png";
  imagen.onload = function (e) {
    procesaImagen();
  }
  /*Insertando fichas en array y las dibujamos   F,R,C */
  fichas_array.push(new Ficha(288, 29, lados, lados, 0, 0, 0));
  fichas_array.push(new Ficha(423, 29, lados, lados, 1, 0, 1));
  fichas_array.push(new Ficha(556, 29, lados, lados, 2, 0, 2));
  fichas_array.push(new Ficha(288, 160, lados, lados, 3, 1, 0));
  fichas_array.push(new Ficha(420, 160, lados, lados, 4, 1, 1));
  fichas_array.push(new Ficha(560, 160, lados, lados, 5, 1, 2));
  fichas_array.push(new Ficha(286, 295, lados, lados, 6, 2, 0));
  fichas_array.push(new Ficha(424, 295, lados, lados, 7, 2, 1));
```

```
    fichas_array.push(new Ficha(560, 295, lados, lados, 8, 2, 2));
}

function mensaje(cadena) {
    var lon = (canvas.width - (20 * cadena.length)) / 2;
    ctx.strokeStyle = "blue";
    ctx.clearRect(0, 420, canvas.width, 100);
    ctx.font = "bold 40px Courier";
    ctx.fillText(cadena, lon, 470);
}


function Ficha(x, y, w, h, i, ren, col) {
    this.x = x;
    this.y = y;
    this.w = w;
    this.h = h;
    this.i = i;
    this.ren = ren;
    this.col = col;
    this.peso = 0;
    this.valor = "";
    this.pinta = pintaFicha;
}


function pintaFicha(valor) {
    this.valor = valor;
    ctx.font = "bold 100px Arial";
    ctx.fillStyle = "blue";
    ctx.fillText(valor, this.x + 30, this.y + 100, this.w, this.h);

}


function ajustar(xx, yy) {
    var posCanvas = canvas.getBoundingClientRect();
    var x = xx - posCanvas.left;
    var y = yy - posCanvas.top;
    return {x: x, y: y}
}
```

```
function seleccionUsuario(e) {
   /*Ajusta la posicion en coordenadas del click*/
   var pos = ajustar(e.clientX, e.clientY);
   var x = pos.x;
   var y = pos.y;
   /*Ciclo verificacion si se ha dado en el lugar correcto y si esta vacio*/
   var ficha;
   for (i = 0; i < fichas_array.length; i++) {
      ficha = fichas_array[i];
      /*Aqui se verifica si el click esta dentro del espacio correcto*/
      if (ficha.x > 0) {
         if ((x > ficha.x) &&
               (x < ficha.x + ficha.w) &&
               (y > ficha.y) &&
               (y < ficha.y + ficha.h)) {
            tiradas++;  //aumenta uno a las tiradas
            break;  //Saca del ciclo
         }
      }
   }
   /*comprobar si aun se puede pintar ficha*/
   if (i < fichas_array.length) {
      /*Pintando la jugada del usuario (si esta vacio el lugar)*/
      if (ficha.valor == "") {
         ficha.pinta("X");
         /*Lanza un timer para que tire la pc en un segundo*/
         setTimeout(tiraMaquina, 1000);
      }
   }
   /****Verificamos si ganamos*******/
   verificaRenglones(true);
   verificaColumnas(true);
   verificaDiagonal1(true);
   verificaDiagonal2(true);
   if (gameOver == false && tiradas < 9) {
      /***Si no ganamos****/
      mensaje("Pensando...");
      canvas.removeEventListener("click", seleccionUsuario, false);
   } else {
      /*****Si ganamos desactivamos el listener******/
      if (gameOver == false) {
         mensaje("Empatados!!!");
      }
```

```
    }
}


function verificaFin(O, X) {
    fin = false;
    if (X == 3) {
        fin = true;
        mensaje("Felicidades, has ganado!!!");
        canvas.removeEventListener("click", seleccionUsuario, false);
    } else if (O == 3) {
        fin = true;
        mensaje("Lo siento, ha Perdido.");
        canvas.removeEventListener("click", seleccionUsuario, false);
    }
    return fin;
}


function buscaFicha(i, j) {
    for (k = 0; k < fichas_array.length; k++) {
        ficha = fichas_array[k];
        if (ficha.ren == i && ficha.col == j) {
            break;
        }
    }
    return ficha;
}


function verificaRenglones(calculaPeso) {
    if (gameOver == false) {
        for (i = 0; i < RENGLONES; i++) {
            fichas_X = 0;
            fichas_O = 0;
            for (j = 0; j < COLUMNAS; j++) {
                ficha = buscaFicha(i, j);
                fichas_X += (ficha.valor == "X" ? 1 : 0);
                fichas_O += (ficha.valor == "O" ? 1 : 0);
            }
            if (calculaPeso) {
                for (j = 0; j < COLUMNAS; j++) {
                    ficha = buscaFicha(i, j);
```

```
               pesoFicha(ficha.i, fichas_O, fichas_X);
            }
         }
         gameOver = verificaFin(fichas_O, fichas_X);
         if (gameOver)
            break;
      }
   }
}
/*********************/

/****************Verificar Columnas*******************/
function verificaColumnas(calculaPeso) {
   if (gameOver == false) {
      for (j = 0; j < COLUMNAS; j++) {
         fichas_X = 0;
         fichas_O = 0;
         for (i = 0; i < RENGLONES; i++) {
            ficha = buscaFicha(i, j);
            fichas_X += (ficha.valor == "X" ? 1 : 0);
            fichas_O += (ficha.valor == "O" ? 1 : 0);
         }
         if (calculaPeso) {
            for (i = 0; i < RENGLONES; i++) {
               ficha = buscaFicha(i, j);
               pesoFicha(ficha.i, fichas_O, fichas_X);
            }
         }
         gameOver = verificaFin(fichas_O, fichas_X);
         if (gameOver)
            break;
      }
   }
}
/******************************************************/

/*******Verificar diagonal desendiente********////
function verificaDiagonal1(calculaPeso) {
   if (gameOver == false) {
      fichas_X = 0;
      fichas_O = 0;
      for (i = 0; i < RENGLONES; i++) {
         ficha = buscaFicha(i, i);
         fichas_X += (ficha.valor == "X" ? 1 : 0);
         fichas_O += (ficha.valor == "O" ? 1 : 0);
      }
      if (calculaPeso) {
         for (i = 0; i < RENGLONES; i++) {
            ficha = buscaFicha(i, i);
            pesoFicha(ficha.i, fichas_O, fichas_X);
         }
      }
      gameOver = verificaFin(fichas_O, fichas_X);
   }
}
/****************************************************/

/*********Verificar Diagonal Acendente******************/
function verificaDiagonal2(calculaPeso) {
   if (gameOver == false) {
      fichas_X = 0;
      fichas_O = 0;
      j = 2;
      for (i = 0; i < RENGLONES; i++) {
         ficha = buscaFicha(i, j);
         fichas_X += (ficha.valor == "X" ? 1 : 0);
         fichas_O += (ficha.valor == "O" ? 1 : 0);
         j--;
      }
      if (calculaPeso) {
         j = 2
         for (i = 0; i < RENGLONES; i++) {
            ficha = buscaFicha(i, j);
            pesoFicha(ficha.i, fichas_O, fichas_X);
            j--;
         }
      }
      gameOver = verificaFin(fichas_O, fichas_X);
   }
}

function pesoFicha(i, fichas_O, fichas_X) {
   ficha = fichas_array[i];
   if (ficha.valor == "") {
      if (fichas_O == 2 && fichas_X == 0) {
         ficha.peso += 10;
      } else if (fichas_O == 0 && fichas_X == 2) {
         ficha.peso += 6;
      } else if (fichas_O == 1 && fichas_X == 0) {
         ficha.peso += 3;
      } else {
         ficha.peso += 1;
      }
   } else {
      ficha.peso = 0;
   }
}

function tiraMaquina() {
   tiradas++;
   console.log("Tirada numero: " + tiradas);
   /*Comprobar si hay ganador*/
   if (gameOver == false) {
      /*Verificar con el peso*/
      verificaRenglones(true);
      verificaColumnas(true);
      verificaDiagonal1(true);
      verificaDiagonal2(true);
      /*Seleccionar mejor jugada*/
      fichas_X = 0;
      fichas_O = 0;
      mejorJugada = 0;
      for (i = 0; i < fichas_array.length; i++) {
         ficha = fichas_array[i];
         if (ficha.peso > mejorJugada) {
            mejorJugada = ficha.peso;
            ii = i;
         }
      }
      /*Realizar jugada de pc*/
      ficha = fichas_array[ii];
      ficha.pinta("O");
      verificaRenglones(false);
      verificaColumnas(false);
      verificaDiagonal1(false);
      verificaDiagonal2(false);
      /*Verificamos si alguien gano*/
      if (gameOver == false) {
         /*Si nadie gano activa el listener de click*/
```

```
            if (tiradas < 9) {
              canvas.addEventListener("click", seleccionUsuario, false);
              mensaje("Pulse su jugada...");
            }
          }
        }
      }


    </script>
  </head>
  <body>
    <h1>"3 in a Row"
      <p>"(Juego del gato)"</h1>
    <canvas id="pantalla" width="960px" height="500px">
      Tu navegador no soporta Canvas.
    </canvas>
    <button id="boton" type="reset" onclick="javascript:window.location.reload();">Volver a
Jugar</button>
  </body>
</html>
```

Glossary of Concepts

- Web Application - A set of pages that run on the Internet; these pages are those that the user views through an Internet browser (Microsoft Internet Explorer, Chrome, Mozilla Firefox, etc.).

- Client Computer - Computer that connects to the server via the Internet.

- Server Computer - A computer that makes Web applications accessible through the Internet.

- Controller - The software that processes the user's requests. It decides which module will have the control to execute the next task.

- Framework: It is a skeleton for the development of an application. The frameworks define the structure of the application, that is to say, the way in which the files are organized and even the names of some of the files and the programming conventions.
- HTTP - HyperText Transfer Protocol. It is a protocol through which the server software communicates with the browser installed on the client computer.

- JSP - JavaServer Page is an HTML page in which Java code is embedded.

- Model - Contains the core functionality, that is, it executes the "business logic". Business logic is the way in which the information is processed to generate the expected results. The model connects to the database to store and retrieve information.

- Dynamic Web page - A Web page in which the server processes the information provided by the user and displays the results of this processing.

- Static Web page - An informative Web page, in which the user cannot make modifications.

- Servlet - A Java class (child of the HttpServlet class) that runs on the server.

- Server Software - It is the software that controls the execution of Web applications, and runs on the Server Computer.

- View - These are the SW modules involved in the interface with the user.

Suggestions

- Preferably use Edge or Chrome browsers for a better adaptation of the page.
- It is recommended to have the zoom at 90% -100%.
- Keep the server open.
- To return to the main menu, it is necessary to use the browser arrows located on the top left, used to return.
- Keep the classes in place, download and place them correctly.
- In case of errors contact the owner, data on the cover page.
- Use default libraries and specific versions required.
- Apache Tomcat is an option, can also be used with Glassfish 4.1.1.
- Open Xampp for better server management.

Tags html:

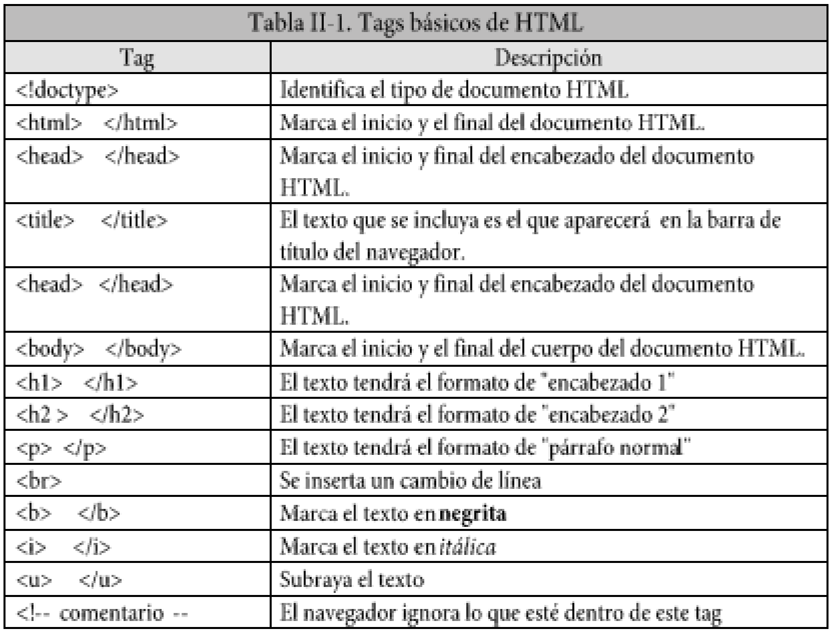


Tabla II-1. Tags básicos de HTML

Tag	Descripción
<!doctype>	Identifica el tipo de documento HTML
<html> </html>	Marca el inicio y el final del documento HTML.
<head> </head>	Marca el inicio y final del encabezado del documento HTML.
<title> </title>	El texto que se incluya es el que aparecerá en la barra de titulo del navegador.
<head> </head>	Marca el inicio y final del encabezado del documento HTML.
<body> </body>	Marca el inicio y el final del cuerpo del documento HTML.
<h1> </h1>	El texto tendrá el formato de "encabezado 1"
<h2 > </h2>	El texto tendrá el formato de "encabezado 2"
<p> </p>	El texto tendrá el formato de "párrafo normal"
 	Se inserta un cambio de línea
<b> </b>	Marca el texto en **negrita**
<i> </i>	Marca el texto en *itálica*
<u> </u>	Subraya el texto
<!-- comentario --	El navegador ignora lo que esté dentro de este tag

References

Creation servlet:

- https://www.youtube.com/watch?v=zlib-2j7yHs&ab_channel=SOPROTI
- http://www.cua.uam.mx/pdfs/revistas_electronicas/libros-electronicos/2017/java/Java.pdf

Creation Gato o 3 en raya:

- https://www.youtube.com/watch?v=olLNys4rP9A&ab_channel=Proyectoodnet

General:

- Basham B., Sierra K., Bates B. , Head First Servlets and JSP, O Reilly & Associates, 2nd Ed,USA, 2008.
- Deitel P., Deitel H., Internet & World Wide Web, How to program. Pearson-Prentice-Hall,USA, 2008.