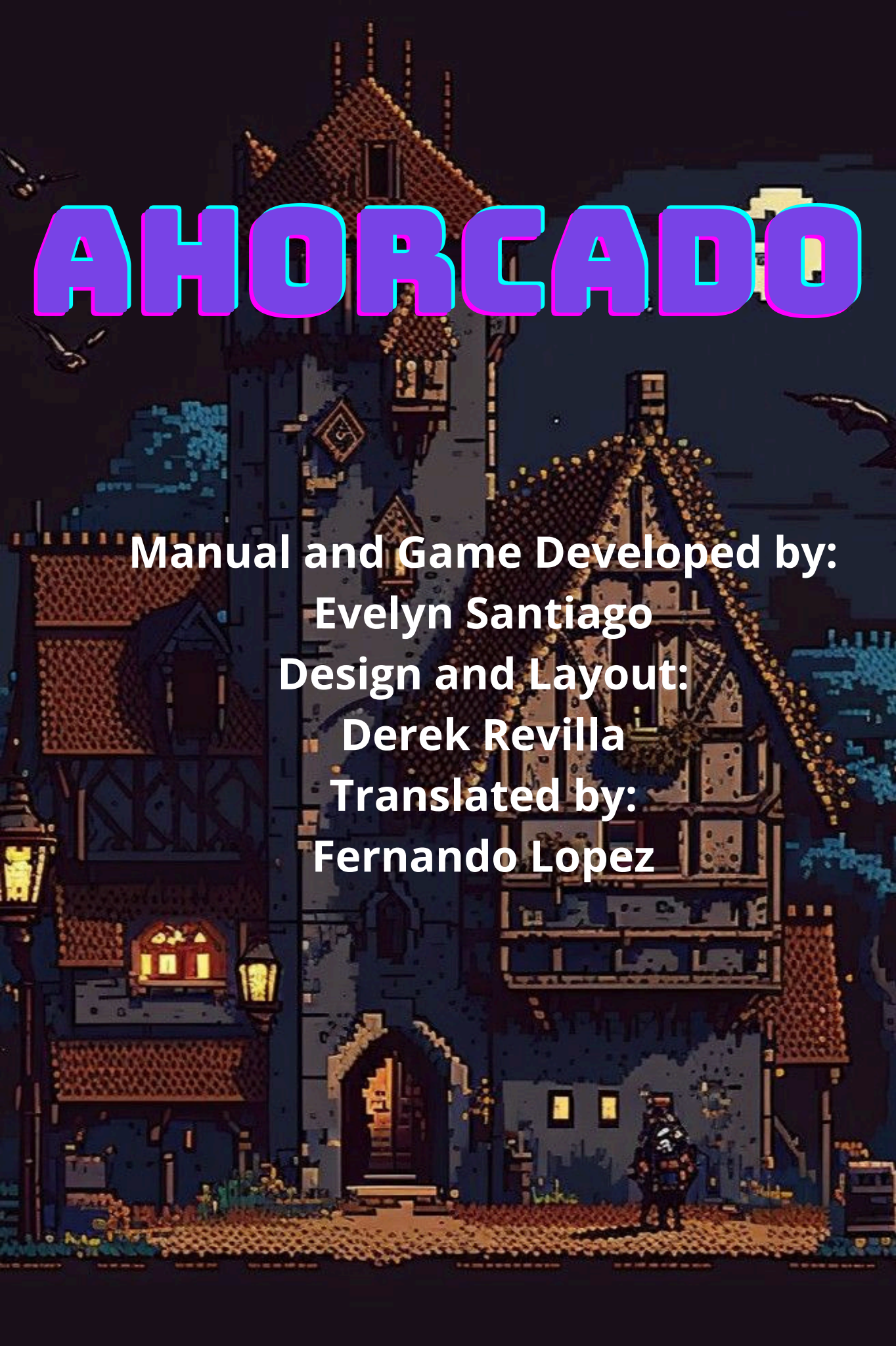

Manual and Game Developed by:
Evelyn Santiago
Design and Layout:
Derek Revilla
Translated by:
Fernando Lopez

DETAILED MANUAL OF THE GAME "HANGED"

Before using the different file types that the game is made of, it's important to understand the project purpose and the involved languages to develop it.
The hanged game is an interactive application design to be used directly on the browser. Its main purpose is to allow the player to guess a secret word by selecting the alphabet words. For every mistake,a hanged man image will be complete, until the player guesses the word or loses the game.
For the creation of this game, its used three fundamental technologies:

- HTML (HyperText Markup Language): its used to structure the page content, in other words, create the visual elements like the title, the canvas, and the restart button.
- CSS (Cascading Style Sheets): it's in charge of the visual part, providing colors, sizes, positions, and element effects to make the game more entertaining and visually functional.
- JavaScript: is the language in charge of the logic part, and the game 's behaviour. Allowing to generate a keyboard, detect the selected keys, check the hits and mistakes and the control of the hanged images.

These three languages combine to offer a fully functional and entertaining game experience
First, the HTML file sets the site's structure; then the CSS sets the appearance, and finally, JavaScript provides movement, interactivity and game logic.
Each part has an essential function that, together, makes possible the hanged game in any up-to-date browser

1.HTML STRUCTURE

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Juego del ahorcado</title>
```

- <!doctype html> Indicates that the file uses HTML5, the newest version of the language
- <html> opens the HTML file.
- <head> contains all the game settings(not directly visible).
- <meta charset="utf-8"> sets the type of coding characters"UTF-8".
- <title> sets the name that will appear in the browsers title: "Juego del ahorcado".

2. STYLES CSS (VISUAL DESIGN)

```css
body {
    width: 960px;
    margin: 0 auto;
}
```

- ESTABLISHES THAT THE WHOLE CONTENT HAS A 960 PIXELS WIDTH AND CENTERED IN THE SCREEN (MARGIN: 0 AUTO).

```css
.imagen-derecha {
  position: absolute;
  top: 40px;
  left: 60px;
  width: 90px;
}
```

- SETS THE IMAGE POSITION AND SIZE THAT WILL BE PLACED IN A FIXED WAY. POSITION: ABSOLUTE;ALLOWS TO PLACE THE EXACT COORDINATES (40PX FROM TOP, 60PX FROM LEFT).

```css
h1 {
    text-align: center;
}
```

CENTERS THE MAIN TITLE GAME (“EL JUEGO DEL AHORCADO”).

```css
#pantalla {
    border: groove 8px gold;
    background: lightgreen;
}
```

- #PANTALLA THE CANVAS IS WHERE THE GAME IS DRAWN.
- BORDER: GROOVE 8PX GOLD; CREATES AN EMBOSSED GOLD EDGE .
- BACKGROUND: LIGHTGREEN; GIVES THE BACKGROUND A LIGHT GREEN COLOR.

```css
#boton {
    background-color: red;
    color: white;
    font-size: 20px;
    text-align: center;
    font-weight: bolder;
    padding: 3px;
    border: solid 2px black;
}
```

- SETS THE BUTTONS VISUAL STYLE “VOLVER A JUGAR”.
- USES RED WITH WHITE LETTERS, CENTERED TEXT AND A BLACK EDGE.

```css
#boton:hover {
    background-color: lightcoral;
    font-size: 22px;
    border: groove 4px red;
}
```

- :HOVER IT ACTIVATES WHEN THE CURSOR PASSES OVER.
- CHANGES THE COLOR AND THE FONT SIZE, GIVING THE FEELING OF INTERACTION (VISUAL DYNAMIC EFFECT).

3. HTML VISIBLE CONTENT

```
<h1>-El juego del ahorcado-</h1>
<canvas id="pantalla" width="960px" height="450px">
    Tu navegador no soporta Canvas.
</canvas>
```

- <H1>: MAIN TITLE GAME.
- <CANVAS>: IS THE CANVAS WHERE THE LETTERS, KEYS AND IMAGES WILL BE DRAWN.

THE WITHIN TEXT (“YOUR BROWSER DOESN´T SUPPORT CANVAS.”)APPEARS ONLY IF THE BROWSER IS OLD AND DOESN’T SUPPORT THE LABEL <CANVAS>.

```
<button id="boton" type="reset" onclick="javascript:window.location.reload();">Volver a
```

- CREATES A BUTTON THAT REFRESHES ALL THE PAGE, REBOOTING THE GAME
- WINDOW.LOCATION.RELOAD() IT LOADS THE FILE HTML FROM SCRATCH.

4. SCRIPT JAVASCRIPT (GAMES LOGIC)

HERE ARE CREATED THE FUNCTIONS THAT ALLOW TO DRAW, DETECT CLICKS, SHOW LETTERS, ETC.

MAIN VARIABLES

```
var ctx;
var canvas;
var palabra;
var letras = "QWERTYUIOPASDFGHJKLÑZXCVBNM";
```

- CANVAS Y CTX: ARE USED TO DRAW IN THE <CANVAS>.
- CTX (CONTEXT) IS WHAT REALLY ALLOWS US TO USE DRAWING TOOLS (TEXT, RECTANGLES,IMAGES).
- PALABRA: STORES THE SECRET GAME’S WORD.
- LETRAS: CONTAINS ALL THE AVAILABLE LETTERS OF THE KEYBOARD (IN CAPITAL LETTERS).


```
var ctx;
var canvas;
var palabra;
var letras = "QWERTYUIOPASDFGHJKLÑZXCVBNM";
```


- SETS THE KEYBOARD’S COLORS, POSITIONS AND SIZES WHEN IT APPEARS ON THE SCREEN.
- INICIOX Y INICIOY: COORDINATES WHERE THE LETTERS ARE STARTING TO BE DRAWN.
- LON: TAMAÑO DE CADA TECLA (35PX).
- MARGEN: ESPACIO ENTRE TECLAS.

MAIN ARRANGEMENTS

```
var teclas_array = new Array();
var letras_array = new Array();
var palabras_array = new Array();
```

- TECLAS_ARRAY: STORES ALL THE “TECLA” TYPE ELEMENTS.
- LETRAS_ARRAY: STORES THE PLACES OF THE ALREADY GUESSED LETTERS.
- PALABRAS_ARRAY: CONTAINS THE PLAYABLE WORD-LIST.

AVAILABLE WORDS

```
palabras_array.push("LEON");
palabras_array.push("CABALLO");
...
palabras_array.push("AGUILA");
```

- IT ADDS ARRAY WORDS <PALABRAS_ARRAY>.
- .PUSH()PUTS A NEW ELEMENT AT THE END
- THESE ARE THE POSSIBLE WORDS THAT WILL APPEAR RANDOMLY.

5. OBJECT CONSTRUCTORS

THESE CREATE “TEMPLATES” FOR LETTERS AND KEYS

KEYS

```
function Tecla(x, y, ancho, alto, letra){
    this.x = x;
    this.y = y;
    this.ancho = ancho;
    this.alto = alto;
    this.letra = letra;
    this.dibuja = dibujaTecla;
}
```

- SETS A KEY WITH ITS POSITION, SIZE AND LETTER THAT REPRESENTS.
- IT ALSO KEEPS WHAT FUNCTION WILL BE USING TO DRAW. (DIBUJATECLA)

LETTERS

```
function Letra(x, y, ancho, alto, letra){
    this.x = x;
    this.y = y;
    this.ancho = ancho;
    this.alto = alto;
    this.letra = letra;
    this.dibuja = dibujaCajaLetra;
    this.dibujaLetra = dibujaLetraLetra;
}
```

- REPRESENTS EACH BLANK SPACE IN THE WORDS.
- IT CAN DRAW ITS “EMPTY BOX” AND ALSO SHOW THE CORRECT LETTER WHEN GETTING IT RIGHT.

- Represents each blank space in the words.
- It can draw its “empty box” and also show the correct letter when getting it right.

6. DRAWING FUNCTIONS

Draw keys

```
function dibujaTecla(){
    ctx.fillStyle = colorTecla;
    ctx.strokeStyle = colorMargen;
    ctx.fillRect(this.x, this.y, this.ancho, this.alto);
    ctx.strokeRect(this.x, this.y, this.ancho, this.alto);

    ctx.fillStyle = "white";
    ctx.font = "bold 20px courier";
    ctx.fillText(this.letra, this.x+this.ancho/2-5, this.y+this.alto/2+5);
}
```

- Draws a key with a dark grey rectangle and red edge.
- Places the letter in white over it, centered,

Drawing the word’s letters.

```
function dibujaLetraLetra(){
    ctx.fillStyle = "black";
    ctx.font = "bold 40px Courier";
    ctx.fillText(this.letra, this.x+this.ancho/2-12, this.y+this.alto/2+14);
}
```

- It executes when the player gets a word right.
- Shows the letter in black according to the box

Draw the empty letter box

```
function dibujaCajaLetra(){
    ctx.fillStyle = "white";
    ctx.strokeStyle = "black";
    ctx.fillRect(this.x, this.y, this.ancho, this.alto);
    ctx.strokeRect(this.x, this.y, this.ancho, this.alto);
}
```

- Draws the white spaces (empty) where the secret word letters go.

7. CLUE FUNCTION

```
function pistaFunction(palabra){
    let pista = "";
    switch(palabra){
        case 'LEON':
            pista = "Ruge y es fuerte";
            break;
        ...
        default:
            pista="No hay pista aun xP";
    }
    ctx.fillStyle = "black";
    ctx.font = "bold 20px Courier";
    ctx.fillText(pista, 10, 15);
}
```

- Shows the clue at the top of the canvas according to the selected word.
- Uses switch to choose the correct phase.

8. DRAW KEYBOARD

```
function teclado(){
    var ren = 0;
    var col = 0;
    ...
    for(var i = 0; i < letras.length; i++){
        letra = letras.substr(i,1);
        miLetra = new Tecla(x, y, lon, lon, letra);
        miLetra.dibuja();
        teclas_array.push(miLetra);
        ...
    }
}
```

- Draw all the keys (from A to Z even ñ).
- It is placed in three rows, with calculated dynamically coordinates.
- Each key is keeped in the array teclas_array.

9. CHOOSE AND DRAW THE WORD

```
function pintaPalabra(){
    var p = Math.floor(Math.random()*palabras_array.length);
    palabra = palabras_array[p];
    pistaFunction(palabra);
    ...
}
```

- Chooses a random word (Math.random()).
- Uses pistaFunction() to show the correct clue.
- Draws the empty spaces of each letter at the center of the canvas.

10. DRAW THE HANGED MAN

```
function horca(errores){
    var imagen = new Image();
    imagen.src = "imagenes/ahorcado"+errores+".png";
    imagen.onload = function(){
        ctx.drawImage(imagen, 390, 0, 230, 230);
    }
}
```

- Shows the hanged man image according to the number of mistakes.
- Each image (ahorcado0.png, ahorcado1.png...) represents a drawing stage.

11. CLICK DETECTORS

```
function selecciona(e){
    var pos = ajusta(e.clientX, e.clientY);
    ...
}
```

- It activates when the player clicks over a drawn key.
- Detects in which canvas coordinates the click happened .
- Checks if that position matches any key.

If the letter matches:

```
if (letra == tecla.letra){
    caja = letras_array[i];
    caja.dibujaLetra();
    aciertos++;
}
```

- Draws the letter in a correct square and sums a success .

If it doesn't:

```
errores++;
horca(errores);
if (errores == 5) gameOver(errores);
```

- Increases the mistakes counter and the hanged man image updates.
- If you reach 5 mistakes it's a game over.

12. GAME OVER.

```
function gameOver(errores){
    ctx.clearRect(0, 0, canvas.width, canvas.height);
    ctx.fillStyle = "black";
    ctx.font = "bold 50px Courier";
    ...
}
```

- Cleans the whole screen.
- Shows the final message:
  - If the word is guessed, appears "muy bien".
  - If not "Lo sentimos".
- Draws again the whole word and the gallows.

13. AUTOMATIC RESTART

```
window.onload = function(){
    canvas = document.getElementById("pantalla");
    if (canvas && canvas.getContext){
        ctx = canvas.getContext("2d");
        if(ctx){
            teclado();
            pintaPalabra();
            horca(errores);
            canvas.addEventListener("click", selecciona, false);
        }
    }
}
```

- It executes automatically when the page ends loading.
- Starts the canvas and the context (ctx).
- Draws the keyboard, the image and the word.
- Activates the click event to play

GENERAL CONCLUSION

This code combines HTML (structure), CSS (visual design) and JavaScript (games logic).

The use of the canvas allows to graphically represent the keyboard, the word, and the images of the hanged without using extra HTML elements

Each function has a clear responsibility:

- draws key / draws the key box: visual control.
- Selects: click detector.
- Gallow: visual retroalimentation.
- Game Over: end game.