

# PIEDRA PAPEL TIJERA



# Índice

Índice.....	2
1. Detalles.....	2
2. Enunciado de la actividad.....	2
3. Pruebas.....	3
2.1. Introducción de usuario con datos no válidos.....	3
2.2. Introducción de cantidad de partidas con datos no válidos.....	5
2.3. Acceso a la aplicación con datos válidos.....	7
2.4. Seleccionar una de las opciones y jugar al menos 5 partidas.....	8
2.5. Pulsar el botón RESET y jugar al menos 3 partidas.....	10
4. Bibliografía.....	11
5. Copyright.....	11

## 1. Detalles

**Nombre:** Manuel Bolaños García Noblejas

**ID estudiante:** 133145

**Asignatura:** Desarrollo web entorno cliente

**Prueba:** PAC 5 (UF3) - PAC de Desarrollo (D)

**Fecha:** Abril 2025

## 2. Enunciado de la actividad

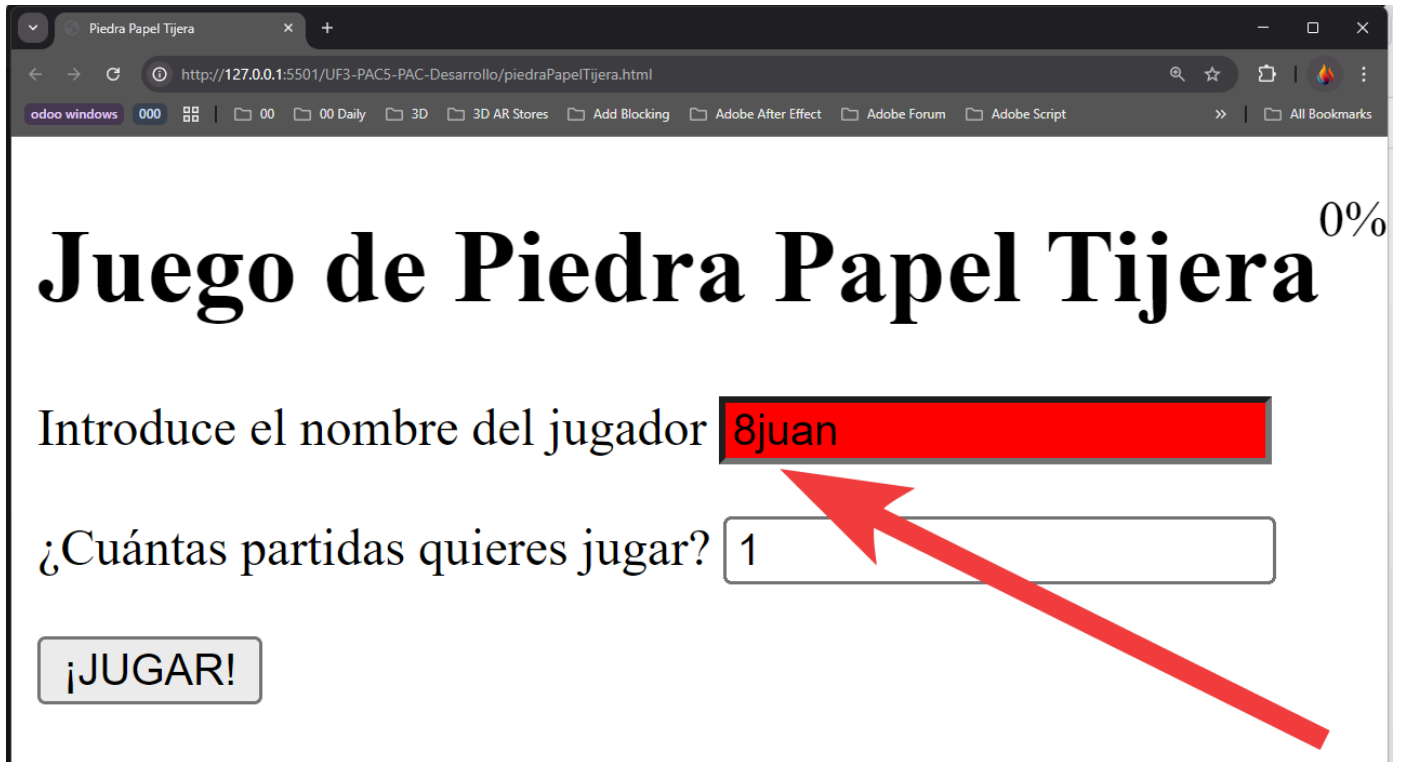
Se quiere desarrollar una aplicación web que permita a sus usuarios jugar a Piedra, Papel o Tijera contra la máquina, eligiendo entre las diferentes opciones y permitiendo llevar un historial de los últimos resultados.

### 3. Pruebas

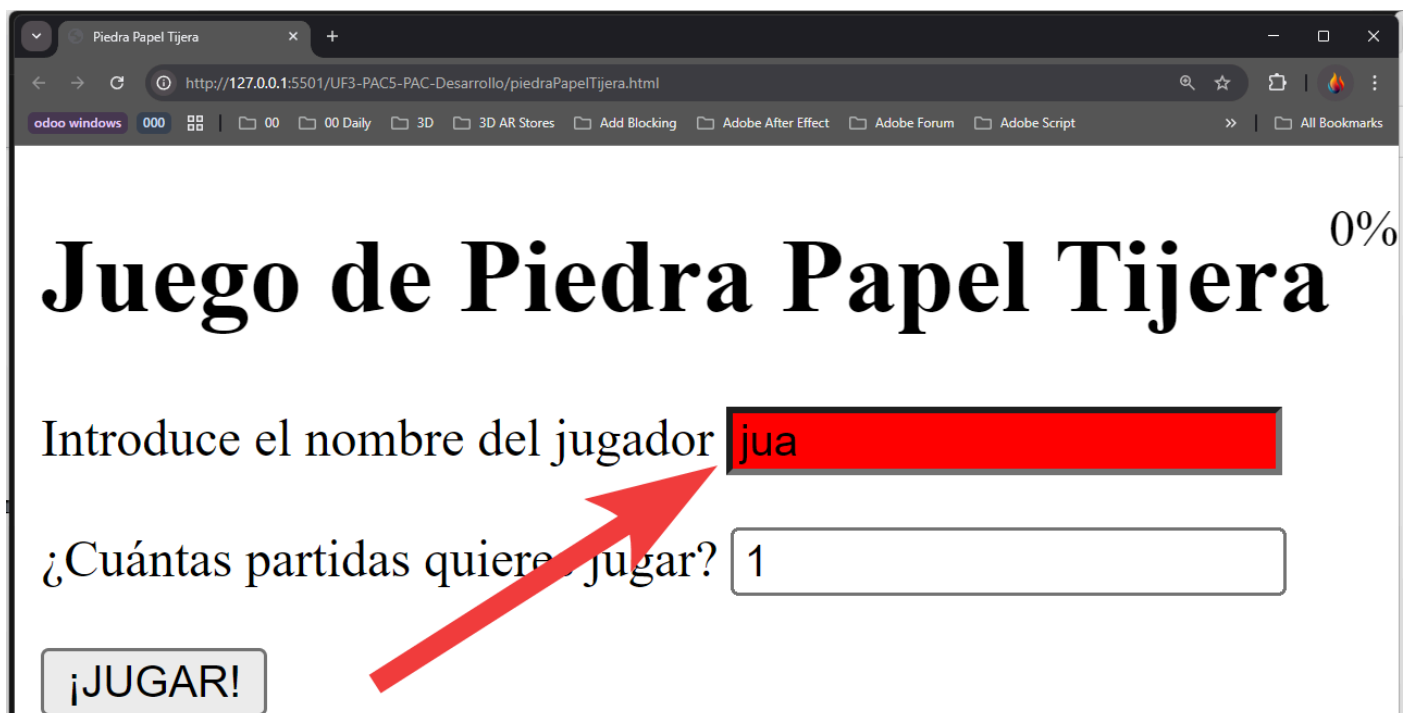
#### 2.1. Introducción de usuario con datos no válidos.

La aplicación del formulario de validación de usuario cumple con 2 requisitos:

- Primer carácter no puede ser un número
- Más de 3 caracteres

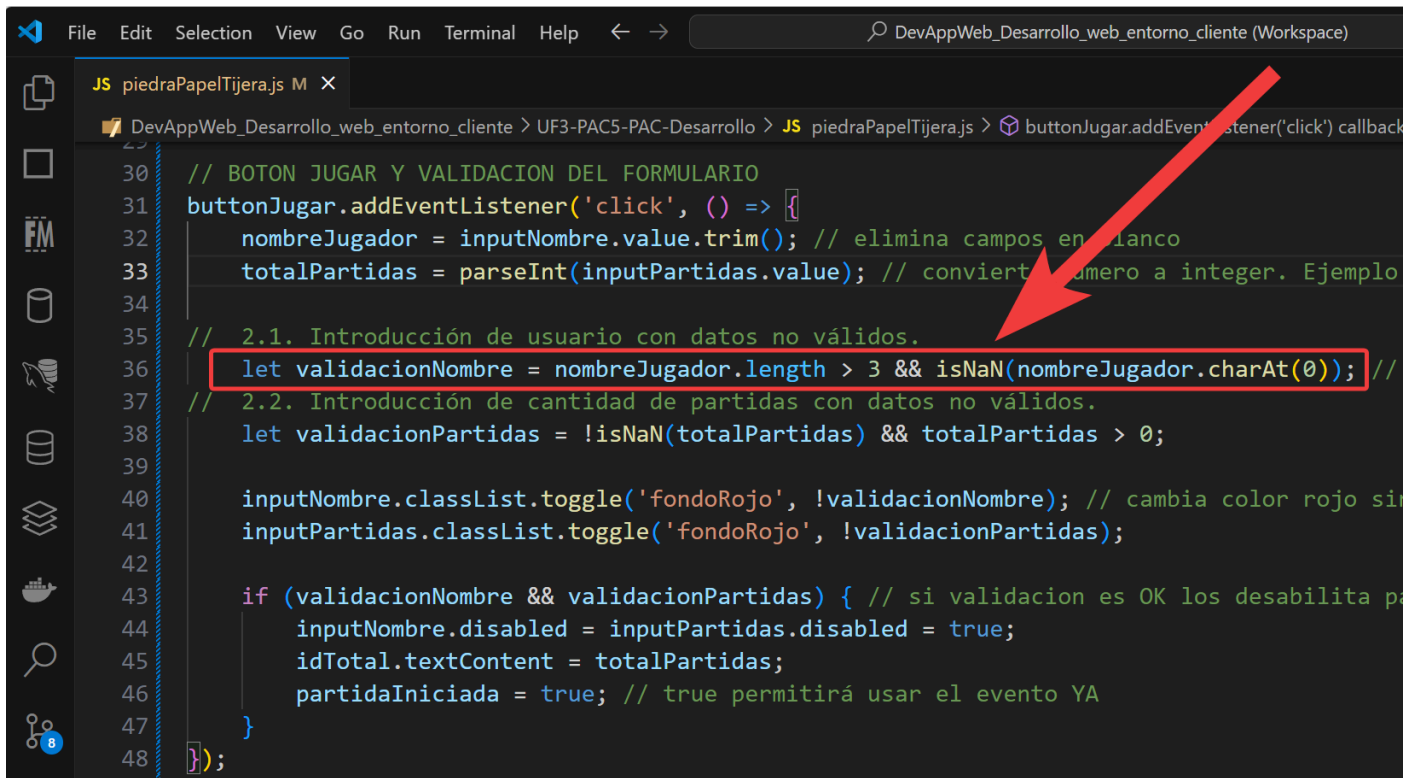


The screenshot shows a web browser window with the title 'Piedra Papel Tijera'. The URL is <http://127.0.0.1:5501/UF3-PAC5-PAC-Desarrollo/piedraPapelTijera.html>. The page content includes the title 'Juego de Piedra Papel Tijera' with a '0%' progress indicator. Below the title, there is a label 'Introduce el nombre del jugador' followed by a text input field containing '8juan'. The input field is highlighted with a red background. Below this, there is a label '¿Cuántas partidas quieres jugar?' followed by a text input field containing '1'. At the bottom left, there is a button labeled '¡JUGAR!'. A red arrow points from the '8' in the username field to the '¿Cuántas partidas quieres jugar?' label.



The screenshot shows the same web browser window as the previous one. The page content is identical, but the text input field for the username now contains 'jua'. The input field is highlighted with a red background. A red arrow points from the 'jua' input field to the '¿Cuántas partidas quieres jugar?' label.





```
30 // BOTON JUGAR Y VALIDACION DEL FORMULARIO
31 buttonJugar.addEventListener('click', () => {
32     nombreJugador = inputNombre.value.trim(); // elimina campos en blanco
33     totalPartidas = parseInt(inputPartidas.value); // convierte numero a integer. Ejemplo
34
35 // 2.1. Introducción de usuario con datos no válidos.
36 let validacionNombre = nombreJugador.length > 3 && isNaN(nombreJugador.charAt(0)); //
37 // 2.2. Introducción de cantidad de partidas con datos no válidos.
38 let validacionPartidas = !isNaN(totalPartidas) && totalPartidas > 0;
39
40 inputNombre.classList.toggle('fondoRojo', !validacionNombre); // cambia color rojo si
41 inputPartidas.classList.toggle('fondoRojo', !validacionPartidas);
42
43 if (validacionNombre && validacionPartidas) { // si validacion es OK los deshabilita p
44     inputNombre.disabled = inputPartidas.disabled = true;
45     idTotal.textContent = totalPartidas;
46     partidaIniciada = true; // true permitirá usar el evento YA
47 }
48 });
```

Esta línea de código realiza la validación del nombre del jugador introducido en el campo de texto. A continuación se explica los detalles más importantes:

- `nombreJugador.length > 3`

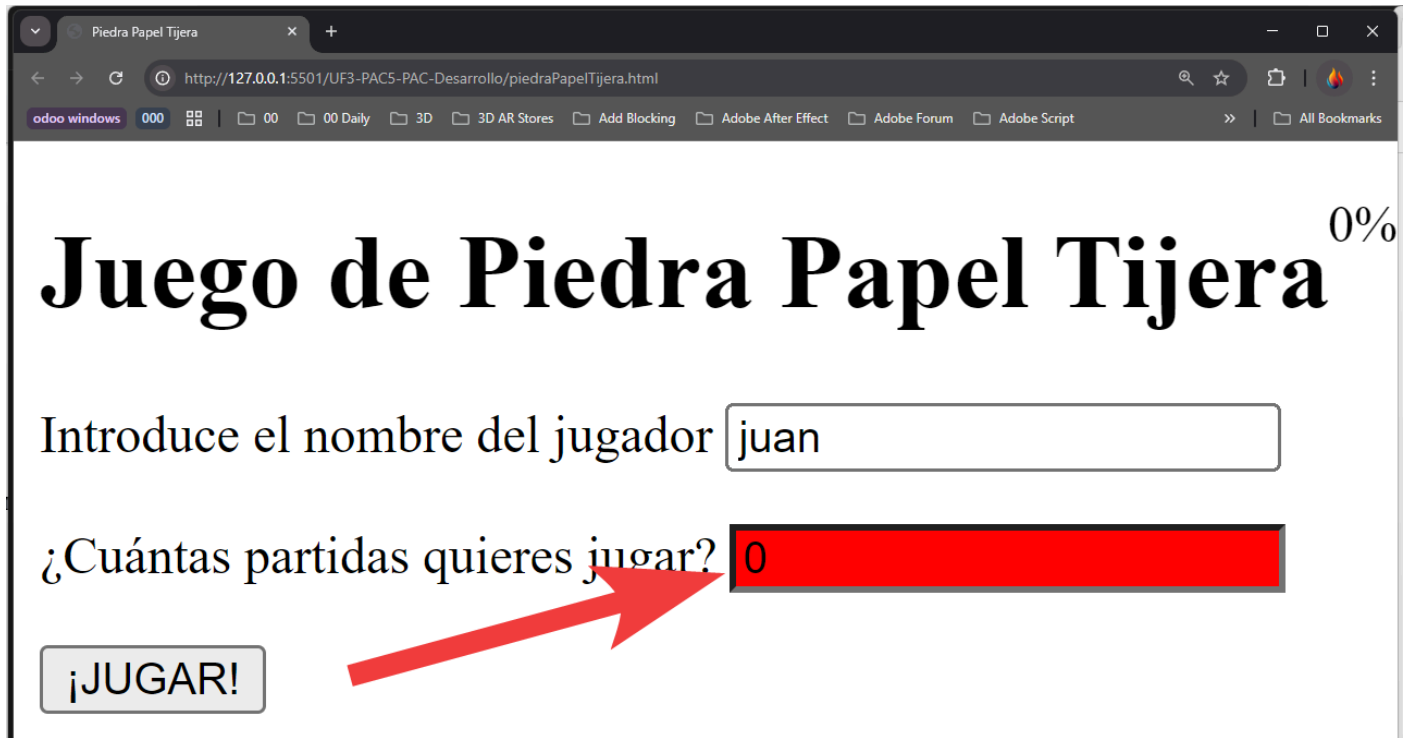
- Esta parte verifica que la longitud del texto introducido como nombre de jugador (`nombreJugador`) sea mayor que 3 caracteres. Esto asegura que el nombre tenga una longitud mínima razonable.

- `isNaN(nombreJugador.charAt(0))`

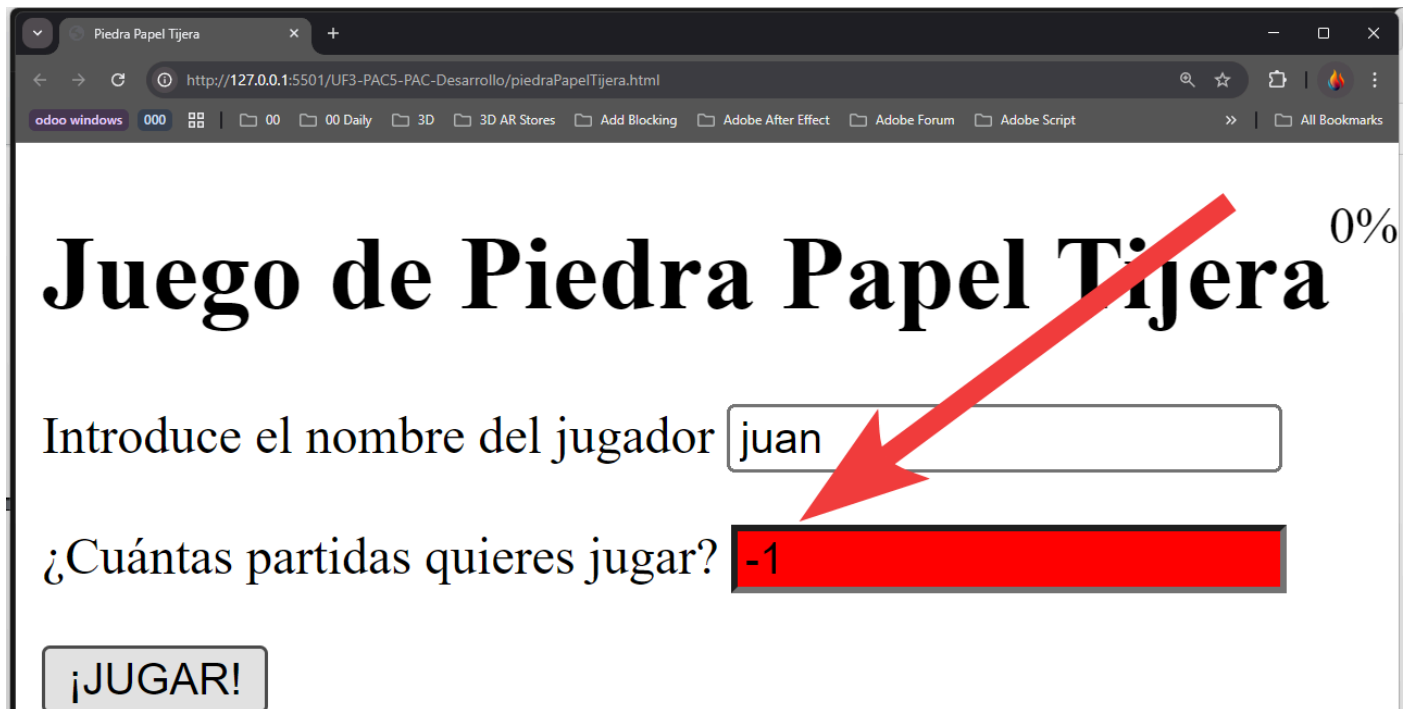
- Aquí se comprueba si el primer carácter del nombre del jugador (`nombreJugador.charAt(0)`) No es un número (**isNaN significa "Is Not a Number"**).
- **charAt(0)** extrae el primer carácter de la cadena.
- **isNaN()** devuelve true si el valor no es un número, y false si es un número. Por lo tanto, esta condición es true si el primer carácter del nombre no es un número, y false si lo es.
- **&&** Este es el operador lógico "AND". Significa que ambas condiciones deben ser verdaderas para que la variable `validacionNombre` sea true.

## 2.2. Introducción de cantidad de partidas con datos no válidos.

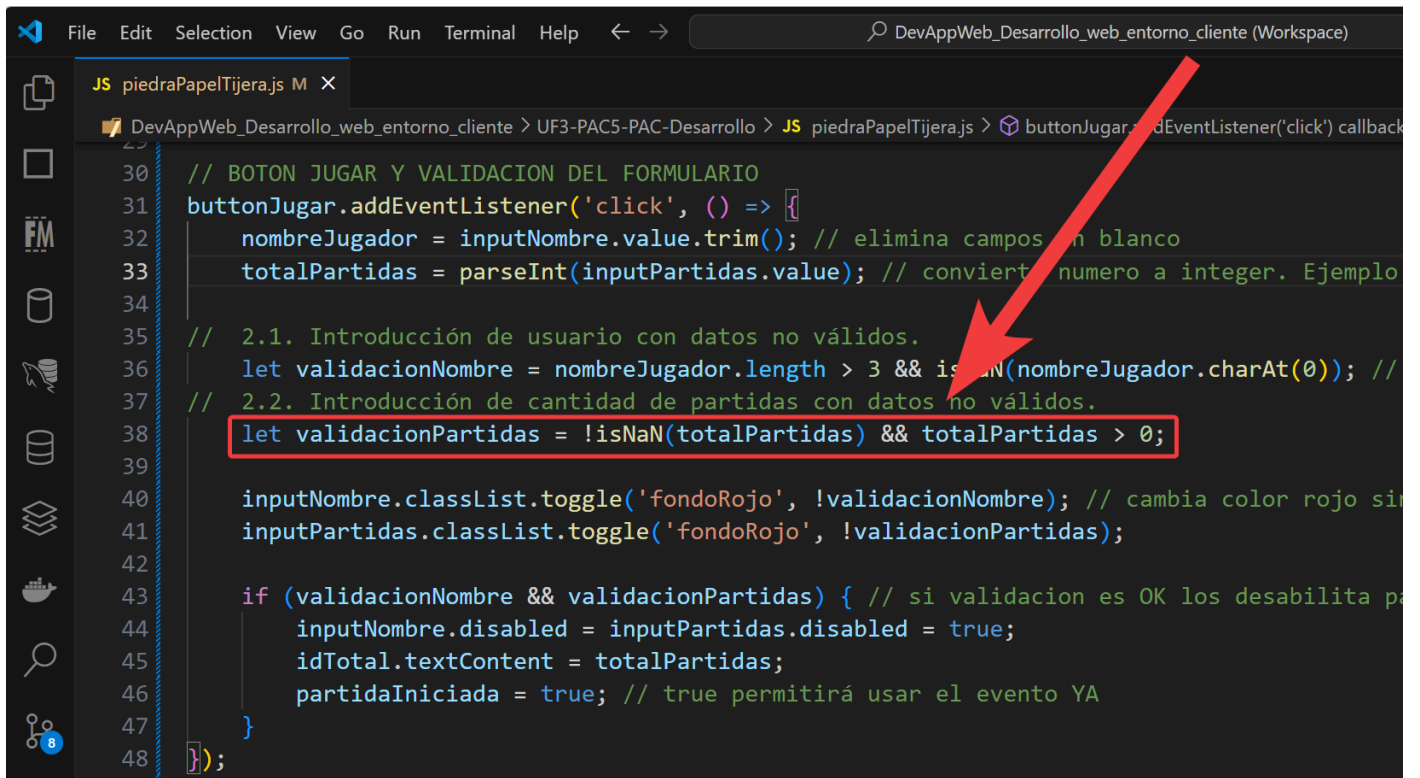
Previamente nos hemos asegurado de convertir el número introducido a integer, para evitar que un usuario introduzca números decimales.



A screenshot of a web browser displaying the 'Juego de Piedra Papel Tijera' application. The title 'Juego de Piedra Papel Tijera' is at the top right with a '0%' indicator. Below it, the text 'Introduce el nombre del jugador' is followed by a text input field containing 'juan'. The next line asks '¿Cuántas partidas quieres jugar?' followed by a red input field containing '0'. A red arrow points from the '¡JUGAR!' button to the '0' in the input field. The browser's address bar shows 'http://127.0.0.1:5501/UF3-PAC5-PAC-Desarrollo/piedraPapelTijera.html'.



A screenshot of the same web application, but with an invalid input. The text 'Introduce el nombre del jugador' is followed by a text input field containing 'juan'. The next line asks '¿Cuántas partidas quieres jugar?' followed by a red input field containing '-1'. A red arrow points from the '¡JUGAR!' button to the '-1' in the input field. The browser's address bar shows 'http://127.0.0.1:5501/UF3-PAC5-PAC-Desarrollo/piedraPapelTijera.html'.



```
30 // BOTON JUGAR Y VALIDACION DEL FORMULARIO
31 buttonJugar.addEventListener('click', () => {
32     nombreJugador = inputNombre.value.trim(); // elimina campos en blanco
33     totalPartidas = parseInt(inputPartidas.value); // convierte numero a integer. Ejemplo
34
35     // 2.1. Introducción de usuario con datos no válidos.
36     let validacionNombre = nombreJugador.length > 3 && !isNaN(nombreJugador.charAt(0)); //
37     // 2.2. Introducción de cantidad de partidas con datos no válidos.
38     let validacionPartidas = !isNaN(totalPartidas) && totalPartidas > 0;
39
40     inputNombre.classList.toggle('fondoRojo', !validacionNombre); // cambia color rojo si
41     inputPartidas.classList.toggle('fondoRojo', !validacionPartidas);
42
43     if (validacionNombre && validacionPartidas) { // si validacion es OK los deshabilita p
44         inputNombre.disabled = inputPartidas.disabled = true;
45         idTotal.textContent = totalPartidas;
46         partidaIniciada = true; // true permitirá usar el evento YA
47     }
48 });
```

- **`!isNaN(totalPartidas)`**

- comprueba si el valor de **totalPartidas** no es un número. El operador ! (NOT lógico) invierte el resultado de `isNaN()`.

- **`totalPartidas > 0`**

- Esta parte verifica si el valor de **totalPartidas** es mayor que 0. Evitamos así que el usuario introduzca un número negativo o cero.

## 2.3. Acceso a la aplicación con datos válidos.

En esta etapa el usuario ha superado la validación del formulario y como resultado los campos se han deshabilitado para de este modo impedir su modificación durante el funcionamiento de la aplicación. Esto se consigue por medio de **disable** como puede verse en la línea de código.

Juego de Piedra Papel Tijera 0%

Introduce el nombre del jugador

¿Cuántas partidas quieres jugar?

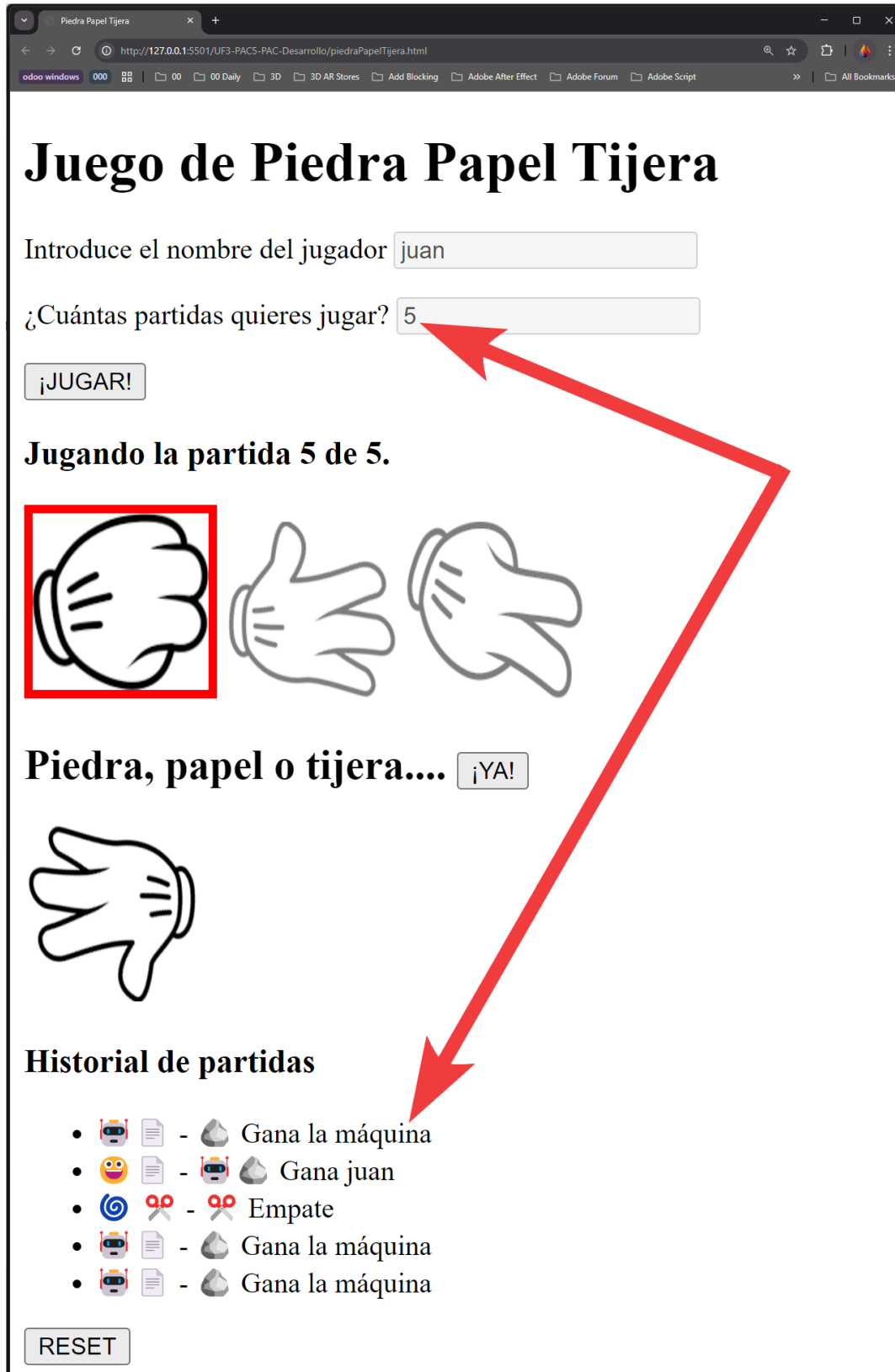
Jugando la partida 0 de 5.

```
30 // BOTON JUGAR Y VALIDACION DEL FORMULARIO
31 buttonJugar.addEventListener('click', () => {
32     nombreJugador = inputNombre.value.trim(); // elimina campos en blanco
33     totalPartidas = parseInt(inputPartidas.value); // convierte numero a integer. Ejempl
34
35 // 2.1. Introducción de usuario con datos no válidos.
36 let validacionNombre = nombreJugador.length > 3 && isNaN(nombreJugador.charAt(0)); //
37 // 2.2. Introducción de cantidad de partidas con datos no válidos.
38 let validacionPartidas = !isNaN(totalPartidas) && totalPartidas > 0;
39
40 inputNombre.classList.toggle('fondoRojo', !validacionNombre); // cambia color rojo s
41 inputPartidas.classList.toggle('fondoRojo', !validacionPartidas);
42
43 if (validacionNombre && validacionPartidas) { // si validacion es OK los deshabilita
44     inputNombre.disabled = inputPartidas.disabled = true;
45     idTotal.textContent = totalPartidas;
46     partidaIniciada = true; // true permitirá usar el evento YA
47 }
48 }
```

## 2.4. Seleccionar una de las opciones y jugar al menos 5 partidas

Observamos nuestra aplicación en funcionamiento, en una partida de 5 jugadas, donde se muestra todos los casos posibles; gana máquina, jugador y empate.

Para mejorar la experiencia de usuario y al mismo tiempo no incumplir los requisitos de nuestra aplicación, donde sólo podemos editar el archivo javascript, se ha optado por introducir una serie de emoticonos logrando un resultado visual mucho más divertido para nuestros jugadores.





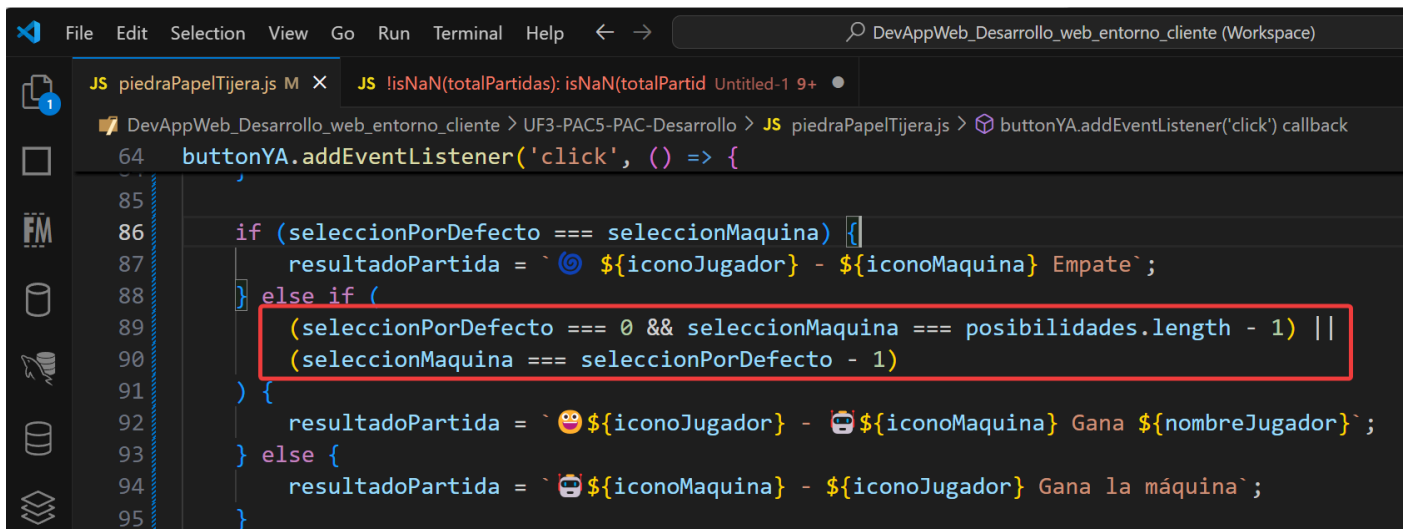
Sin duda, esta línea es el mayor reto de dificultad que presenta la aplicación ya que requiere de una parte lógica avanzada. Una de las claves es el orden en que están posicionados los elementos en el array **posibilidades**:

```
var posibilidades = ["piedra", "papel", "tijera"];
```

```
0 = piedra
```

```
1 = papel
```

```
2 = tijera
```



```
64 buttonYA.addEventListener('click', () => {
85
86     if (seleccionPorDefecto === seleccionMaquina) {
87         resultadoPartida = `🤖 ${iconoJugador} - ${iconoMaquina} Empate`;
88     } else if (
89         (seleccionPorDefecto === 0 && seleccionMaquina === posibilidades.length - 1) ||
90         (seleccionMaquina === seleccionPorDefecto - 1)
91     ) {
92         resultadoPartida = `😄 ${iconoJugador} - 🤖 ${iconoMaquina} Gana ${nombreJugador}`;
93     } else {
94         resultadoPartida = `🤖 ${iconoMaquina} - ${iconoJugador} Gana la máquina`;
95     }
96 }
```

```
• (seleccionPorDefecto === 0 && seleccionMaquina ===  
posibilidades.length - 1) ||
```

- Aquí la clave es **"piedra le gana a tijera"**.
- ambas condiciones dentro del paréntesis deben ser verdaderas. Es decir, el jugador debe haber elegido "piedra (0)", y la máquina debe haber elegido "tijera".

```
• (seleccionMaquina === seleccionPorDefecto - 1)
```

- Evalúa si la máquina ha seleccionado la opción que pierde contra la elección del jugador, cubriendo los casos donde el jugador gana con papel sobre piedra (**jugador=papel, máquina=piedra**) y con tijera sobre papel (**jugador=tijera, máquina=papel**).
- Esta condición, junto con la anterior, abarca todas las posibles victorias del jugador en el juego.

## 2.5. Pulsar el botón RESET y jugar al menos 3 partidas

El objetivo principal de este botón es reiniciar el juego a su estado inicial, permitiendo al jugador comenzar una nueva partida y mostrar en nuestro historial un texto que así lo confirme.

```
• inputPartidas.disabled = inputNombre.disabled = false;
```

- Quitamos el **disable** para que el usuario pueda de nuevo introducir valores en el formulario.

```
• inputPartidas.value = 0;
```

- Establecemos el valor inicial a cero para comenzar de nuevo.

```
• li.textContent = '★ Nueva partida';
```

- Indicamos visualmente en nuestro historial que comenzamos una nueva partida.



## 4. Bibliografía

[w3schools.com](https://www.w3schools.com)

[javascript.info](https://javascript.info)

[stackoverflow.com](https://stackoverflow.com)

## 5. Copyright

[Leonardo.ia](#) (Imagen de la portada)

Usuarios de nivel gratuito: si bien Leonardo.ai posee los derechos de las imágenes que usted crea, se le otorga una licencia no exclusiva y libre de regalías para usar el contenido generado con fines comerciales.

[Emoticonos Licencia](#) (emoticonos)