

# Documentación del proyecto Scape Room de Mairena del Alcor



Nombre: María Jiménez Conzález

Curso: 2ºDAW

Módulo profesional: Desarrollo web entorno servidor

Índice:

<b>Documentación del proyecto scape room de Mairena del Alcor</b>	<b>3</b>
Explicación de las pruebas	3
Explicación del código	3
index.php	3
interactividad.js	4
prueba1.php	4
prueba2.php y prueba3.php	4
funciones.php	4
final.php	5
Soluciones de las pruebas	5

## Explicación de las pruebas

### 1. Prueba 1: La inscripción de la piedra:

Este desafío comienza en el castillo de Luna de Mairena del Alcor. El usuario deberá escribir lo que está escrito en la imagen que se proporciona. La palabra que se encuentra en ella es *Luna*, que hace honor al Castillo de Mairena y la Familia a la que pertenecía.

### 2. Prueba 2: La plegaria de la ermita:

Este desafío tiene lugar en la ermita de San Sebastián, lugar céntrico de Mairena que posee el nombre del Patrón y de la cual sale la Patrona, Nuestra Señora de los Remedios, el día de la Romería Mairenera.

La prueba consistirá en decir qué día se celebra esta fiesta en concreto, la Romería. La respuesta será: *el primer domingo de septiembre*.

### 3. Prueba 3: La feria:

Este desafío tiene lugar en la Feria de Mairena, una de las fiestas más importantes del pueblo, además de la primera Feria de toda Sevilla y considerada “La madre de las ferias”.

La prueba consistirá en decir quién y en qué año la fundó originalmente.

La respuesta es: *Juan II de Castilla en 1441*, quien fundó la primera feria de ganado en Mairena del Alcor.

## Explicación del código

### index.php

Este es el primer archivo; será lo primero que vea el usuario al entrar en nuestro juego.

Lo primero que encontramos es un código PHP con el inicio de sesión que se encarga de que, al pulsar desde cualquier página el botón “Volver a empezar”, vuelva a aparecer esta página con una sesión reiniciada.

Después encontramos el HTML de esta página, con su imagen y su texto para ponerte en situación de lo ocurrido. Debajo encontraremos un botón con el que podremos dar inicio a la historia y pasar a la prueba 1.

Al final de la página tenemos la llamada al archivo *interactividad.js*. Esta se realiza al final del documento porque, si el .js se cargara en el *head* o antes de que el formulario existiera todavía, esos elementos no se habrían creado y el .js fallaría o no haría nada.

### interactividad.js

Este archivo se encarga de pedir la confirmación de que queremos la pista extra.

Si el usuario pulsa aceptar, se cambia el valor del campo *confirm* y se envía “1”.

Esto hará que se muestre la pista.

La llamada a esta función se encuentra dentro de cada prueba.

## prueba1.php

Este archivo define el funcionamiento de la prueba 1, muy similar al de las otras dos.

Lo primero que encontramos es una instrucción que obliga a que exista el archivo *funciones.php*, ya que muchas funciones trabajan sobre él. Luego inicializamos los textos de mensaje y pista a vacío e indicamos el número de prueba que es; esto último nos servirá para indicarle al programa cuál es la respuesta correcta del usuario o qué pista debe darse si se solicita.

Lo siguiente es comprobar si la pista extra ha sido solicitada; de ser así, se llamará a la función *obtenerPista()* de *funciones.php*, donde enviaremos el número de prueba.

Después encontramos el apartado que revisa la respuesta y la marca como correcta o no.

Si es correcta, pasará automáticamente a la prueba 2.

Si no lo es, se dan al usuario hasta 4 intentos; el propio programa indica cuántos quedan. Si se pierden todos los intentos el programa lleva al archivo *perdiste.php*, donde se informa de la derrota y se ofrece volver a empezar.

Por último, pasamos al HTML de la página. En él se ve el texto que pone en situación de la prueba, la imagen, las secciones que muestran información como los intentos restantes o la pista (si está activada), y el formulario para rellenar.

## prueba2.php y prueba3.php

Su funcionalidad es prácticamente idéntica a la primera prueba, con la excepción de que estas usan respuestas compuestas. Es decir, hay dos campos de respuesta a rellenar. Esto se gestiona en *funciones.php* separando ambas respuestas mediante un símbolo “|”.

## funciones.php

Este archivo se encarga de la gestión de todas las funciones de la aplicación.

Empieza iniciando la sesión. Luego se define un array con las respuestas correctas, asegurándonos de que en las pruebas 2 y 3, que tienen respuesta múltiple, ambas partes estén separadas con “|”. También se define un array con las pistas para cada prueba.

A continuación aparece la validación de las respuestas. Se divide en caso simple o compuesto, pero en ambos casos el final es el mismo:

Si la respuesta introducida coincide exactamente con la almacenada en el array, la función devuelve *true* y se pasa a la siguiente prueba. También se enviará la pista si se ha solicitado.

El programa también lleva la cuenta de los intentos: cada vez que se falle o se recargue la página, se resta un intento.

También se incluyen funciones como marcar la prueba como correcta, sanear el *input* (uso muy frecuente, por lo que se crea una función para evitar repeticiones) o resetear el estado.

## final.php

Si se aciertan todas las pruebas antes de que se acaben los intentos, se redirige a esta página. En ella aparece la enhorabuena y la conclusión de la historia. También se puede volver a empezar o ver esta documentación completa mediante un botón.

Si por el contrario se acaban los intentos en alguna prueba, se dirige al archivo *pierdes.php*, donde aparece un mensaje indicando que no quedan intentos y se ofrece comenzar de nuevo.

## Soluciones de las pruebas

- **Prueba 1:** Luna
- **Prueba 2:**
  - **Día:** domingo
  - **Mes:** septiembre
- **Prueba 3:**
  - **Año:** 1441
  - **Nombre del fundador:** Juan II de Castilla