Práctica 02 – Battle Arena – v1

Objetivo académico

El fin de la práctica es que demuestres que eres capaz de resolver un problema que requiere el uso de tecnologías HTML, CSS i JavaScript antes no propuesto en los ejercicios de evaluación continua/prácticas/clase. Al final de esta documentación encontrarás la rúbrica de evaluación.

Juego Battle Arena

El concepto del juego es muy sencillo. En un espacio cerrado los jugadores deben sobrevivir a los ataques de los otros jugadores. El jugador ganador es el que menos muertes ha sufrido.

Battle Arena es un juego de rol al más puro estilo dungeoncrawler 2D. El principal objetivo será mantenerse en vida.

Las funcionalidades

El juego debe contemplar, como mínimo, las siguientes acciones:

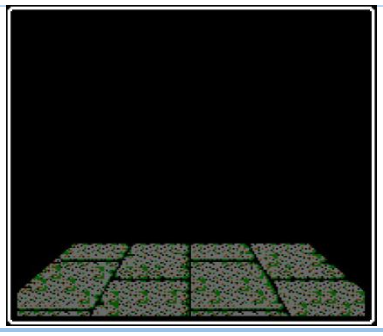
- Moverse por la Arena: adelante, atrás, girar vista a la derecha y girar vista a la izquierda.
- Atacar enemigos. Por omisión, cualquier personaje jugador se considera enemigo.
- Un visor de navegación en el que visualizar la Arena.
- Un minimapa que muestre al jugador, a los enemigos y objetos, así como la dirección a la que se está mirando y están mirando los enemigos.
- Una brújula.
- Un espacio donde ver las características del jugador.
- Nuevo jugador, Revivir jugador, Eliminar jugador

El visor de navegación: lo que ve el jugador

La mazmorra es un lugar muy oscuro. A simple vista el jugador solo puede ver una posición por delante. Inicialmente el visor, se pueden añadir más visores y posiciones, puede mostrar 2 posiciones distintas: oscuridad o entidad. Se considera una entidad todo aquello que no sea un espacio vacío, como puede ser un enemigo o una pared.

El jugador puede moverse hacia el Norte (N), Sur (S), Este (E) u Oeste (O). Cuando se mueve el jugador hacia una dirección, este se quedará mirando hacia dicha dirección.

Ejemplos de situaciones según posición:

		
Sin pared delante	Con pared delante	Con enemigo delante

Para poder ir hacia la izquierda o a la derecha se deben utilizar las opciones de movimiento del jugador: girar la vista a la izquierda y girar la vista a la derecha. Esto significará que para ir a la izquierda el jugador deberá girar a la izquierda y avanzar. El mapa tiene 40x40 casillas.

El log de acciones: lo que sucede en el entorno

Toda la información que interese a un jugador deben visualizarse en un espacio de texto: Enemigo muerto, objetos en el suelo, información detallada de enemigos...

El jugador: características

Cualquier jugador tiene las siguientes características (entre paréntesis se indica el nombre de la variable en los resultados de las APIs):

- **Identificador único (token)**
- **Nombre (name)**
- **Posición horizontal (x)**
- **Posición vertical (y)**
- **Dirección donde mira el jugador (d)**
- **Ataque (attack)**
- **Defensa (defense)**
- **Puntos vitales (vp)**
- **Imagen (image)**
- **Objeto (object)**

Objetos

Los objetos tienen un nombre, una imagen, un valor de ataque y un valor de defensa. Estos valores puedes ser 0 siendo un objeto normal, positivos siendo un objeto mágico o negativos siendo un objeto maldito.

Solo se puede llevar un objeto (espada, escudo, talismán, orbe, anillo…) teniendo en cuenta que:

- Cuando un jugador se crea aparece sin objeto.
- Cuando se equipa un objeto este no se puede dejar, solo se puede cambiar.
- Cuando un jugador muere, el objeto cae en la celda en la que murió.
- Cuando se elimina un jugador se pierde el objeto para siempre.

La lucha: morir o sobrevivir

El sistema de lucha se basa en tiempo real:

- El jugador que ataque primero tiene más posibilidades de sobrevivir.
- Cuando se ataca y hay más de un enemigo en la casilla de enfrente, el sistema selecciona el objetivo a atacar al azar.

Muerte: el jugador es vencido

Cuando un jugador muere se convierte en un fantasma y cualquier ataque recibido resultara en vano. Puede recorrer la Arena, pero no puede interactuar. Para poder jugar, se deberá eliminar el jugador de la partida y crear uno de nuevo, o revivirlo.

Partida: nueva, revivir, eliminar

El usuario puede crear un nuevo jugador. Si ya existe un jugador deberá eliminarlo o de revivirlo en caso de muerte. **Un usuario solo puede tener un jugador activo.**

API

Condiciones de uso de la API del Battle Arena: Para poder llamar a la API hay que estar conectado a Eduroam desde dentro de LaSalle o utilizar la VPN desde fuera de LaSalle

API: remove

- **GET**:
  - Descripción: Elimina el jugador indicado.
  - Parámetros:
    - token: identificador único del jugador
  - Códigos de retorno:

        - 200 Si se ha eliminado correctamente
    - Contenido de retorno:
        - Sin contenido
    - Formato de llamada:
        - http://<direccion>/arena/api/remove/<token>

API: spawn

- **GET**:
    - Descripción: Genera un nuevo jugador.
    - Parámetros:
        - name: nombre del jugador
    - Códigos de retorno:
        - 200 Si se ha generado correctamente
    - Contenido de retorno:
        - El token único del jugador
    - Formato de llamada:
        - http://<direccion>/arena/api/spawn/<nombre>

API: respawn

- **GET**:
    - Descripción: Regenera un nuevo jugador, actualizando posición, imagen y puntos de vida.
    - Parámetros:
        - token: Identificador único del jugador
    - Códigos de retorno:
        - 200 Si se ha regenerado el jugador correctamente
    - Contenido de retorno:
        - Sin contenido
    - Formato de llamada:
        - http://<direccion>/arena/api/respawn/<token>

API: players & objects

- **GET**:
    - Descripción: Devuelve información detallada de jugadores y objetos, si estos están en la celda del jugador o colindante.
    - Parámetros:
        - token: indentificador único del jugador

- Códigos de retorno:
  - 200 Si se han podido consultar los enemigos y objetos
- Contenido de retorno:
  - JSON con la información de los enemigos y objetos
- Formato de llamada:
  - http://<direccion>/arena/api/players/<token>

API: map

- **GET**:
  - Descripción: Devuelve la posición de los enemigos y objetos que hay en todo el mapa.
  - Parámetros:
    - sin parámetros
  - Códigos de retorno:
    - 200 Si se han podido consultar los enemigos y objetos
  - Contenido de retorno:
    - JSON con la posición de los enemigos y objetos
  - Formato de llamada:
    - http://<direccion>/arena/api/map

API: player

- **GET**:
  - Descripción: Devuelve información detallada del jugador.
  - Parámetros:
    - token: Identificador único del jugador.
  - Códigos de retorno:
    - 200 Si se han podido consultar la información del jugador
  - Contenido de retorno:
    - JSON con la información del jugador
  - Formato de llamada:
    - http://<direccion>/pwi/arena/api/player/<token>

API: move

- **GET**:
  - Descripción: Mueve el jugador hacia el Norte, Sur, Este u Oeste.
  - Parámetros:
    - token: Identificador único del jugador
    - direccion: La letra coincidente con la dirección objetivo (N, S, E, O).

- Códigos de retorno:
  - 200 Si se han podido mover al jugador
  - 500 Si ha habido algú error en el movimiento, como intentar atravesar una pared.
- Contenido de retorno:
  - Sin contenido o mensaje de error
- Formato de llamada:
  - http://<direccion>/arena/api/move/<token>/<direccion>

API: attack

- **GET**:
  - Descripción: Ataca al primer enemigo, en la dirección indicada.
  - Parámetros:
    - token: Identificador único del jugador
    - direccion: La letra coincidente con la dirección donde atacar (N, S, E, O).
  - Códigos de retorno:
    - 200 Si se ha podido realizar el ataque
    - 500 Si ha habido algún error en el ataque, como intentar atacar si se está muerto, no hay ningún enemigo hay una pared delante.
  - Contenido de retorno:
    - Los puntos de vida que se quitan en el ataque
  - Formato de llamada:
    - http://<direccion>/arena/api/attack/<token>/<direccion>

API: craft

- **GET**:
  - Descripción: Crea un objeto. Si el jugador tiene un objeto se equipará el nuevo y se eliminará el viejo. Solo se puede crear un objeto si el jugador está vivo.
  - Parámetros:
    - token: Identificador único del jugador
    - name: El nombre del objeto
    - image: La dirección web de la imagen del objeto
    - attack: Valor del ataque al usar objeto. Puede ser negativo si el objeto está maldito.
    - defense: Valor de la defensa al usar objeto. Puede ser negative si el objeto está maldito.
  - Códigos de retorno:
    - 200 Si se ha podido crear le objeto.
  - Contenido de retorno:

        - Sin contenido o mensaje de error.
    - Formato de llamada:
        - http://<direccion>/pwi/arena/api/craft/<token>

API: pickup

- **GET**:
    - Descripción: Recoge un objeto. Solo se puede coger un objeto si el jugador está vivo. Si el jugador tiene un objeto se equipará el objeto recogido y se soltará el viejo.
    - Parámetros:
        - token: Identificador único del jugador
        - name: El nombre del objeto
        - image: La dirección url de la imagen del objeto
        - attack: Valor del ataque al usar objeto. Puede ser negativo si el objeto está maldito.
        - defense: Valor de la defensa al usar objeto. Puede ser negative si el objeto está maldito.
    - Códigos de retorno:
        - 200 Si se ha podido coger el objeto.
    - Contenido de retorno:
        - Sin contenido o mensaje de error
    - Formato de llamada:
        - http://<direccion>/pwi/arena/api/craft/<token>

Requisitos

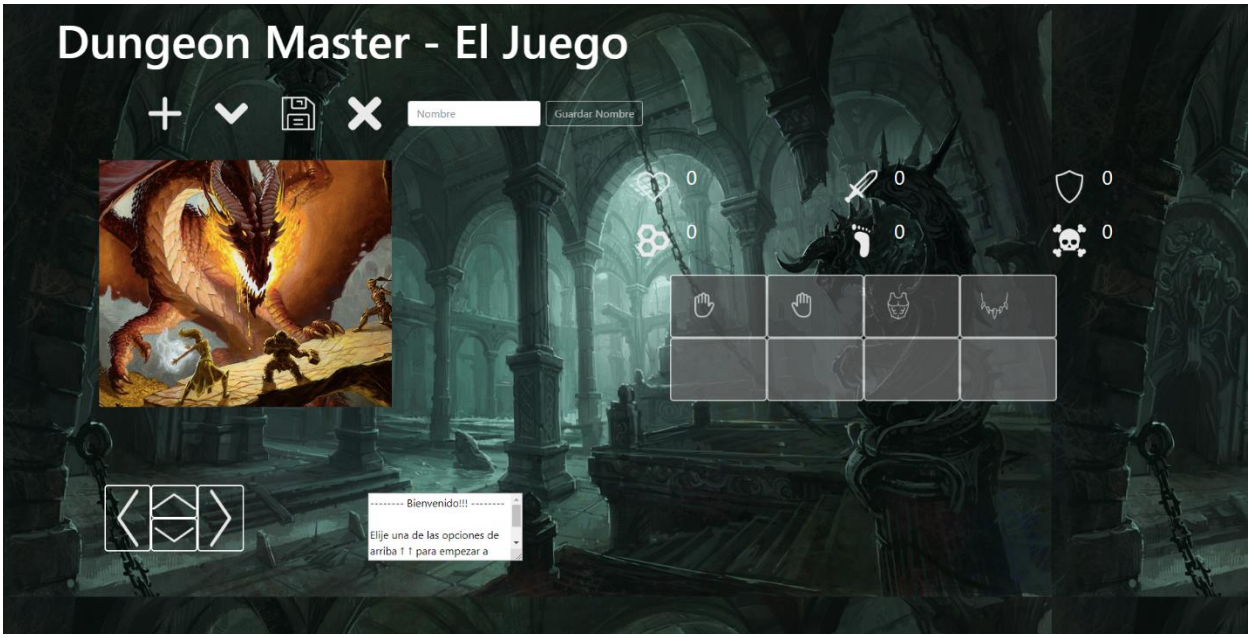
La práctica debe cumplir los siguientes requisitos:

- Debéis entregar un **.zip** con:
    - Carpeta **dev** (con todos los archivos fuente y compilados).
    - Carpeta **build** (listo para jugar).
- El nombre del archivo .zip debe ser **NOMBRE_APELLIDO_PAC2.zip o GRUPO_#_PAC2.zip.**
- No puede aparecer ningún error de consola, ni al cargar el juego ni durante el mismo.

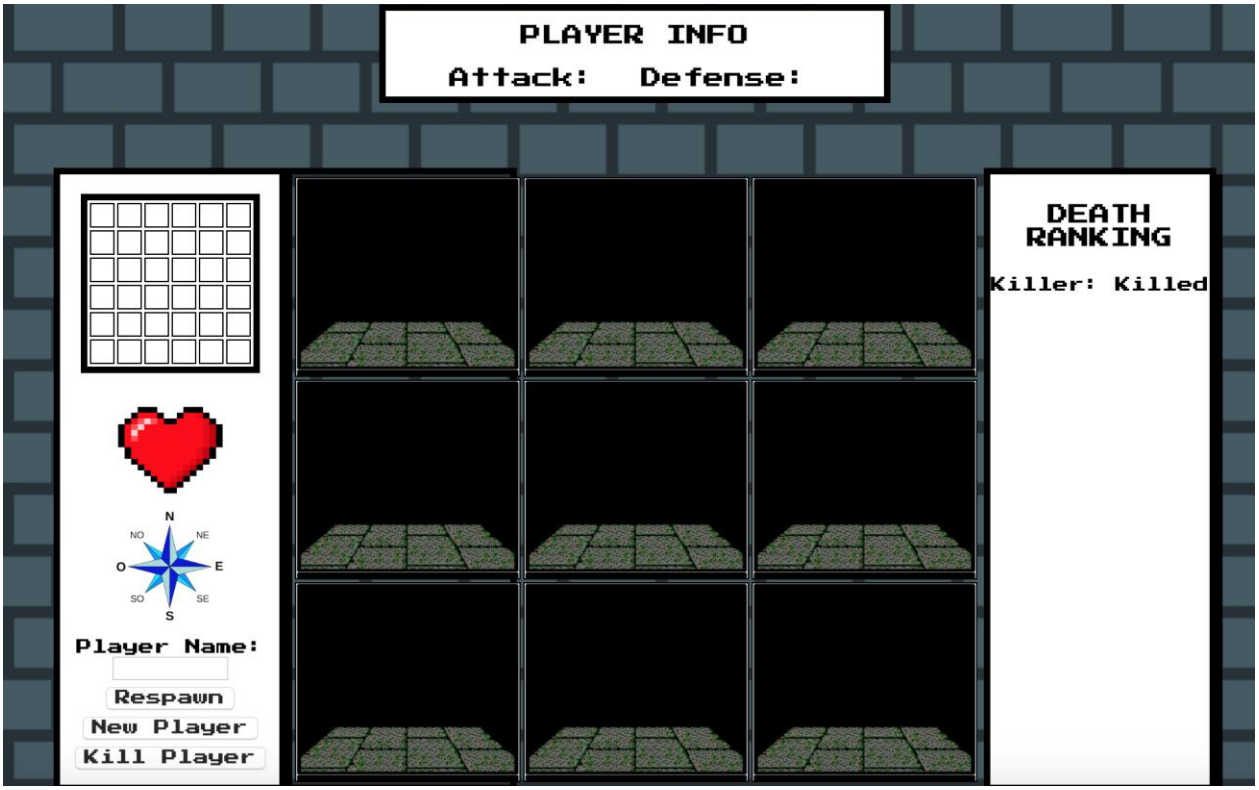
Ejemplos de interfaces de prácticas anteriores:


laSalle dungeon
Ramon Llull University
Nueva partida
Cargar partida
Xp
Ataque
Defensa
Vida
Mapa
Nivel 0
laSalle
Ramon Llull University
dungeon
Controles
SALTAR (DOS PASOS)
Consola
[17:13:42] ¡Bienvenido a LaSalle Dungeon! Selecciona "Nueva partida" o "Cargar partida" para empezar a jugar
Equipo
Mano izquierda
Mano derecha
Mochila
Mochila vacía

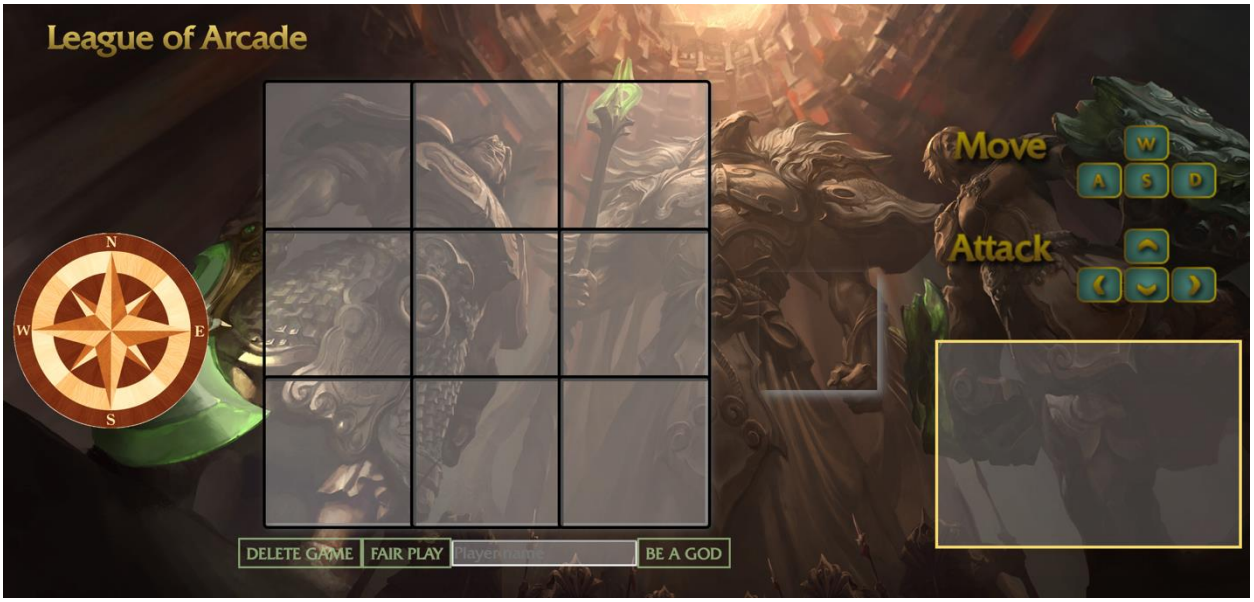

STRANGER THINGS
Dungeon Master
Comenzar partida
Cargar partida
Guardar partida
Eliminar partida
Navegación
Controles
Brújula
Jugador
Nivel:
XP:
Ataque:
Defensa:
Vida:
Equipo
Mano Izquierda
Mano Derecha
Mochila
ESPACIO LIBRE


Dungeon Master - El Juego
Nombre
Guardar Nombre
0
0
0
0
0
0
-------- Bienvenido!!! --------
Elije una de las opciones de arriba ↑ ↑ para empezar a


Nueva partida
Cargar partida
Guardar partida
Borrar partida
Tutorial
Movimiento:
Puedes clicar para moverte
O
Pulsar Shift + WASD
Shift
W
A
S
D
Informacion sobre hechizo:
Pon el cursor encima del boton.
Objetos:(click izquierdo para usar)
Armas:
Click izquierdo para IZQUIERDA
Ctrl + Click izquierdo para DERECHA
Close
(Puedes usar Shift + WASD)
Eventos
Batalla
Otros


PLAYER INFO
Attack: Defense:
DEATH RANKING
Killer: Killed
N
NO
NE
O
E
SO
SE
S
Player Name:
Respawn
New Player
Kill Player


Vida:
Attack:
Defense:
MENU
Nuevo Jugador
Ver ranking
Revivir Jugador
Ayuda
18:40 - El toke recibido es null
18:40 - ¡Te has movido una casilla hacia el undefined!
MAP
N
Leyenda mapa
Jugador
Jugador
Jugador
Enemigo desc.
Vacía



Evaluación:

Requisitos

Todo lo que no está especificado en esta rúbrica se define a discreción de los desarrolladores. Para que la práctica pueda evaluarse debe cumplir los siguientes requisitos:

- Los archivos HTML, CSS, SCSS y JavaScript propios deben contener comentarios claros.
- Los archivos HTML, SCSS y JavaScript deben estar libre de errores.
- Los archivos HTML y CSS deben validar por W3 sin errores.
- Debe haber una carpeta src con el código fuente de la práctica (la rúbrica TECNOLOGIA evaluará esta carpeta).
- Debe haber una carpeta build con los archivos suficientes para jugar a la práctica.
- **Usar un refresco de un mínimo de 1 segundo. Cualquier otro refresco puede saturar al servidor con las consecuencias derivadas y no se aceptará como entrega.**

Si se cumplen los requerimientos anteriores se aplicará la siguiente rúbrica:

Categoría	**3**	**2**	**1**	**0**
		JUEGO		
Movimiento	Realiza todos los movimientos básicos y añade nuevos movimientos (huida...).	Realiza todos los movimientos básicos.	Realiza alguno de los movimientos básicos.	No se mueve.
Lucha	Hace llamada API correctamente, captura respuesta y trata datos devueltos.	Hace llamada API correctamente y captura respuesta.	Hace la llamada API pero respuesta no capturada, capturada incorrectamente o llamada incorrecta.	No hace llamadas a las API del juego.
Jugabilidad	Hay elementos del juego que permiten jugar con facilidad y estar informado sin complicaciones.	El usuario puede jugar con normalidad aunque hay opciones de juego escondidas o dificiles de acceder.	El usuario puede jugar, aunque hay elementos confusos.	No es jugable, confuso y no se informa al usuario de ninguna o poca información
Visor	Hay un visor que muestra todos los elementos, información de las APIs, información transformada (porcentages…) o múltiples visores.	Hay un visor que muestra todos los elementos de juego más información adicional conseguida de las APIs.	Hay un visor que no muestra todos los elementos de juego (paredes, jugadores, enemigos).	No hay visor.
Minimapa	Muestra minimapa con todos los elementos de juego y información aumentada (colores distintos según puntos de vida…).	Muestra minimapa con todos los elementos de juego.	Muestra minimapa pero faltan elementos de juego.	No muestra minimapa.
Información de datos del jugador y estado de la partida	Además muestra información adicional a la básica (porcentajes, número de enemigos muertos...)	Muestra toda la información actualizada a tiempo real (mínimo 1 segundo).	Muestra la información o alguna pero no toda.	No muestra información.

Diseño	Además añade animaciones y efectos (visuales y/o sonoros).	El juego es completamente visual.	Añade algún elemento visual.	No usa interfaz gráfica.
		TECNOLOGIA		
API	Demuestra que envía, recibe y hace tratamiento de los códigos devueltos en asíncrono.	Envía y recibe en JSON en síncrono.	No envía o no recibe información.	No envía y no recibe información del servidor
JSON	Hace uso de los datos tratados.	Hace tratamiento de los datos con JSON.parse	Hace tratamiento de los datos con eval()	No hace tratamiento de los datos
Framework HTML/CSS	Usa un framework de desarrollo web sin errores con una justificación de uso	Usa un framework de desarrollo web sin errores.	Usa un framework de desarrollo web con errores.	No usa ningún framework.
JavaScript	Utiliza como dos librerías externas JavaScript, una local y otra en CDN, y usa todas las estructuras de programación vistas en clase.	Utiliza todas las estructuras de programación vistas en clase (objectos, clases, arrays, funciones, callbacks).	Utiliza algunas de las estructuras de programación vistas en clase.	Errores consola
Bases de datos locales	Usa correctamente base de datos local sin expirar sesión.	Usa base de datos local sin expiración de sesión con errores.	Usa base de datos local con expiración de sesión.	No usa base de datos.
Carpetas Src & Build	Entrega las dos carpetas necesarias y contienen los archivos correctos.	Entrega las dos carpetas necesarias pero no contienen los archivos correctos.	Entrega carpetas pero no existe la src o build.	No entrega carpetas.
Comentarios en código HTML, CSS, JAVASCRIPT	Hay comentarios de calidad en todo el código.	Hay comentarios, algunos confusos y otros de calidad.	Hay comentarios pero son confusos o no ejemplifican el código (no son de calidad).	No hay comentarios.