



## EXAMEN UD1-3

### Los RA's que evaluaremos son:

- RA1.- Selecciona las arquitecturas y tecnologías de programación sobre clientes Web, identificando y analizando las capacidades y características de cada una.
- RA2.- Escribe sentencias simples, aplicando la sintaxis del lenguaje y verificando su ejecución sobre navegadores Web.
- RA3.- Escribe código, identificando y aplicando las funcionalidades aportadas por los objetos predefinidos del lenguaje.
- RA4.- Programa código para clientes Web analizando y utilizando estructuras definidas por el usuario.

## Fiesta en Fondo de Bikini

### PARTE I (5,5 puntos)

#### 1. Clase Personaje (1 punto)

Representa a los habitantes de Fondo de Bikini.

##### Atributos (privados):

**nombre** → String. Debe comenzar por mayúscula y tener al menos 4 caracteres. Si no cumple la condición, se lanza un error.

##### Métodos:

**Constructor (nombre)**

**toString()** → Devuelve toda la información.

**Getters y Setters** con validaciones.

#### 2. Clase Humano (hereda de Personaje) (1 punto)

Representa a los personajes humanos que visitan Fondo de Bikini (por ejemplo, el pirata Parches el Pirata).

##### Atributos adicionales (privados):

**profesion** → String. No puede estar vacía y debe terminar en "o" o "a" (ejemplo: "cocinero", "pirata").

**edad** → Number. Debe ser un número entre 16 y 120. Si no lo es, se lanza error.

##### Métodos:

**Constructor (nombre, profesion, edad)**

**toString()** → Incluye la información heredada y propia.

**Getters y Setters** con validaciones.

#### 3. Clase Animal (hereda de Personaje) (1 punto)

Representa a los animales que viven en Fondo de Bikini (Bob Esponja, Patricio, Calamardo, etc.).

##### Atributos adicionales (privados):



**especie** → String. Solo puede tomar los valores:

"esponja", "estrella de mar", "calamar", "cangrejo", "ardilla".

Si no coincide con ninguno, se lanza error.

**trabajo** → String. Debe contener al menos una palabra y tener más de 3 letras (por ejemplo, "cocinero", "músico"). Si no cumple, se lanza error.

**Métodos:**

**Constructor (nombre, especie, trabajo)**

**toString()** → Incluye la información heredada y propia.

**Getters y Setters** con validaciones.

#### 4. Clase Fiesta (1,5 puntos)

Representa una gran celebración en Fondo de Bikini.

**Atributos (privados):**

**nombre** → String, mínimo 3 caracteres, no puede contener números.

**lugar** → String, no vacío.

**fecha** → String con formato "YYYY-MM-DD".

Si no coincide con ese formato, se lanza error.

**alInvitados** → Array que almacenará objetos de tipo Personaje (animales o humanos).

**Métodos:**

**Constructor (nombre, lugar, fecha)**

**anadirInvitado(objetoPersonaje)**

- Añade el personaje si no estaba repetido (mismo nombre).
- Devuelve true si se añade, false si ya existía.

**eliminarInvitado(nombre)**

- Elimina el invitado con ese nombre si existe. Devuelve true o false.

**toString()**

- Devuelve la información general de la fiesta y el número de invitados.

**Getters y Setters** correspondientes.

#### - Consideraciones generales -

- Cada clase debe ir en un archivo independiente (.js).
- Usa sintaxis ES6 (class, import/export).
- Todos los atributos deben ser privados y el acceso a los mismos se hará por sus métodos getters y setters.
- Las validaciones se harán en los setters.
- Si un valor no cumple las condiciones, debe lanzarse una excepción (throw).
- Usa bloques try...catch para controlar errores al crear objetos.
- Usa métodos de arrays (forEach, some, filter, findIndex, etc.).
- Se recomienda usar funciones flecha en las callbacks.
- El código debe estar comentado con claridad, sobre todo en las acciones que requieran más complejidad. (0,5 puntos)



## PARTE II (4 puntos)

### Recursos previos

- HTML entregado por el profesorado con cuatro botones.
- Módulos de clases ya implementados en la parte de POO: personaje.js, humano.js, animal.js, fiesta.js.

#### 1) Botón Crear objetos (1 punto)

- o Al pulsarlo se deben crear:
  - ☐ 2 objetos de Humano.
  - ☐ 3 objetos de Animal.
  - ☐ 1 objeto de Fiesta.
- o Añadir todos los humanos y animales a la fiesta mediante su método correspondiente.
- o Guardar el estado en variables accesibles para el resto de botones (mismo módulo).
- o Mostrar un breve mensaje de confirmación (alerta o salida por consola).

#### 2) Botón Mostrar humanos (0,75 puntos)

- o Requisito previo: si no se han creado objetos, informar y no continuar.
- o Abrir una ventana secundaria con window.open.
- o En esa ventana, indicar el número de humanos creados (consultarlo en tiempo de ejecución; no puede ser un valor escrito de forma estática).

Nota: para mostrar texto en una nueva ventana emergente:

```
ventanaHumanos.document.body.innerText = "N.º Humanos invitados:" + NumHumanos +  
"¡En esta ventana se cerrará automáticamente en 10 segundos...";
```

- o La ventana debe cerrarse automáticamente a los 10 segundos.
- o Si ya existe una ventana de humanos abierta, cerrarla antes de abrir una nueva.
- o Dar foco a la ventana secundaria al abrirla.

#### 3) Botón Mostrar animales (0,75 puntos)

- o Requisito previo: si no se han creado objetos, informar y no continuar.
- o Abrir una ventana secundaria (independiente de la de humanos).
- o En esa ventana, indicar el número de animales creados (consultarlo en tiempo de ejecución; no puede ser un valor escrito de forma estática).
- o La ventana debe cerrarse automáticamente a los 10 segundos.
- o Si ya existe una ventana de animales abierta, cerrarla antes de abrir una nueva.
- o Dar foco a la ventana secundaria al abrirla.

#### 4) Botón Mostrar fiesta (1,5 puntos)

- o Requisito previo: si no se han creado objetos, informar y no continuar.
- o Abrir una ventana secundaria (independiente de las anteriores).
- o Mostrar un resumen de la fiesta:
  - ☐ toString() de la fiesta.
  - ☐ Número total de invitados.
  - ☐ Listado de cada invitado en una línea.
- o La ventana debe cerrarse automáticamente a los 10 segundos.



- o Si ya existe una ventana de fiesta abierta, cerrarla antes de abrir una nueva.
- o Dar foco a la ventana secundaria al abrirla.