****

Módulo Profesional 06:  
Desarrollo web en entorno cliente **Actividad UF2**

**CICLO FORMATIVO DE GRADO SUPERIOR EN**

**DESARROLLO DE APLICACIONES WEB**

**MODALIDAD ONLINE**

**Juan Bello Fernández  
2º DAW**

**Actividades evaluables de la UF2**

|  |  |
| --- | --- |
| Objetivos de la actividad | Metodología |
| Evaluar los conocimientos en el desarrollo web en entorno cliente sobre sintaxis del lenguaje y objetos predefinidos del lenguaje. | * Preparación individual. |
| Criterios de evaluación | Entrega |
| Conocimiento en temas relacionados con:   * Identificación y aplicación de los objetos predefinidos del lenguaje. * Escribir código, identificando y aplicando las funcionalidades aportadas por los objetos predefinidos del lenguaje. | * Fecha límite el 09/11/2022. * Subir en archivo comprimido con el documento de la actividad y la carpeta de código fuente. |
| Dedicación estimada | Documentos de referencia |
| 180 minutos. | Libro de texto y recursos disponibles en el aula. |

**Descripción de la actividad**

A continuación, se presenta una serie de ejercicios de contenido teórico y práctico. En el primero de los casos, se proponen tanto preguntas de opción múltiple (utilizar color de resaltado de texto para la respuesta elegida), como preguntas de respuesta breve. Por el otro lado, se pide escribir código a través de unas actividades propuestas.

**Desarrollo de la actividad**

**Preguntas de opción múltiple (25%).**

1. **¿Qué función utilizaríamos para convertir una cadena de caracteres (string) en un número de coma flotante?**
2. eval
3. **parseFloat**
4. isNaN
5. unescape
6. **Para llamar una función definida por el usuario, ¿qué de las siguientes afirmaciones es cierta?**
7. Se tiene que invocar la función a primeros de la página.
8. **Es puede hacer el llamamiento antes de la declaración de la función aunque estén en diferentes bloques de scripts.**
9. La declaración de la función se tiene que hacer primero que el llamamiento cuando estén en diferentes bloques de scripts.
10. Se tiene que invocar la función al final de la página.
11. **Utilizaremos un *Array* cuándo...**
12. **sea necesario acceder a la información mediante un índice.**
13. se necesite guardar una cadena de texto muy larga.
14. todas las variables sean del mismo tipo.
15. se tenga que simplificar el valor de varías variables.
16. **Indica si la siguiente afirmación se verdadera o falsa: “El código *arrNotes[1]* devuelve la primera posición de *la Array* arr*Notas*”**
17. Verdadera
18. **Falsa**
19. **Para poder recorrer un Array bidimensional utilizaremos...**
20. **dos estructuras de control *for*, una dentro del otro.**
21. una función de usuario, con la posición de *la Array* como parámetro.
22. un *Array* asociativo para acceder a las dos dimensiones.

**Preguntas de respuestas breves (25%)**

1. **Describe los diferentes conceptos que se indican a continuación.**
2. **Función**
3. **Array asociativo y Array bidimensional**

**-Array asociativo:** es un array cuyos índices no son numéricos sino cadenas de texto (claves). En JavaScript no existen realmente arrays asociativos, pero podemos simularlos creando objetos y accediendo a sus propiedades.

Se utilizan llaves {} para generar el array de elementos clave: valor. Podemos añadir elementos de dos maneras

a)  
var coche = new Array();

coche["color"] = "rojo";

coche["marca"] = "seat";

coche["modelo"] = "leon";

b)  
var coche = {"color":"rojo","marca":"seat","modelo":"leon"};

En los arrays indexados accedemos con el índice o la posición dentro del array. En los arrays asociativos accedemos con la clave

coche[“color”]

- **Array bidimensional:** es un array de arrays, es decir, cada elemento del array es, a su vez, otro array. Los arrays de 2 dimensiones guardan sus valores, por decirlo de alguna manera, en filas y columnas y por ello necesitamos dos índices para acceder a cada una de sus posiciones.

Un ejemplo sencillo de un array de 2 X 2

var nuevoArray = new Array(2);

//En cada posición de nuevoArray guardamos un nuevo array de dos posiciones

nuevoArray[0] = new Array(2);

nuevoArray[1] = new Array(2;

Para acceder a cada elemento debemos indicar la posición en la que se encuentra. Si pensamos en filas y columnas indicaremos primero la fila y después la columna

//Accedemos a la fila 0, columna 1

nuevoArray[0][1];

Aunque se pueden crear arrays bidimensionales de manera manual como en el ejemplo anterior, lo normal es hacerlo mediante bucles. Aquí un ejemplo de un array bidimensional de 2 x 2

//Declaración del array de 2 posiciones

var nuevoArray = new Array(2);

//Bucle para meter en cada posición otros array de 2

for(var i=0; i<2; i++) {

nuevoArray[i] = new Array(2);

}

Para recorrer un array bidimensional necesitamos dos bucles for, uno dentro de otro.

Cuando no conocemos el tamaño del array utilizamos el método length() en el segundo argumento del bucle for():

//Bucle que recorre el primer array

for(var i=0; i<nuevoArray.length; i++) {

//Bucle que recorre el array que está en la posición i

for(var j=0; j<nuevoArray[i].length; j++) {

document.write(nuevoArray[i][j]);

}

}

**Actividad práctica (50%)**

1. **Un circuito de carreras de Karts nos pide que programemos una serie de scripts para guardar información sobre las carreras que realizan y los pilotos que corren.**
2. Se quiere tener un índice de la clasificación de cada carrera, a medida que los pilotos cruzan la línea de meta. Se quiere guardar, por orden de llegada, el número de Kart, el nombre del piloto y el tiempo que ha tardado al realizar la carrera. Escribe el código necesario para guardar esta información utilizando Arrays y mostrarla por pantalla. Utiliza información ficticia para guardar la clasificación de una carrera donde han participado 5 pilotos.
3. **Realiza un documento web con temática libre, en la que incluyas elementos tanto a nivel de HTML como de CSS y JS. Se requiere aplicar alguna de las funcionalidades aportadas por los objetos predefinidos del lenguaje. Envía una carpeta de código fuente.**