IES MARQUES DE COMARES

GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL

Junta de Andalucía
Consejería de Educación y Deporte

Unión Europea-
NextGenerationEU.

UNIDAD 2-1

Cadenas, Arrays y Funciones

IES MARQUES DE COMARES

GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL

Junta de Andalucía
Consejería de Educación y Deporte

Unión Europea-
NextGenerationEU.

ÍNDICE:

1.- Introducción

Tanto las cadenas de texto como los arrays son, en realidad, objetos con sus propiedades y métodos.

2- Cadenas

- **Declaración de cadenas**

Para declarar una cadena basta con asignar una cadena a una variable.

Ej: let nombre=“miguel”;

Pero, ya hemos dicho que en realidad una cadena es un objeto, en este caso String, con lo cual podríamos también declararlo con su constructor.

Ej: let nombre=new String(“hola”);

La única diferencia es cómo se muestran en consola.

Normalmente se suele utilizar la primera forma.

Cadenas

- **Propiedades**

Tienen dos únicas propiedades prototype (prototipo) y length (longitud de la cadena).

El prototipo es a su vez un objeto, y por tanto, a sirve para que a dicho objeto se le pueden añadir nuevas propiedades y métodos.

- **Métodos.**

Tienen muchos métodos, pero nombremos algunos interesantes: toUpperCase(), toLowerCase(), include(),

IES MARQUES DE COMARES

Cadenas

- **Métodos.**

Tienen muchos métodos, pero nombremos algunos interesantes:

toUpperCase(): pasa a mayúsculas. Ej: nombre.toUpperCase()
toLowerCase(): pasa a minúsculas.
include(arg): comprueba y devuelve true si la cadena incluye al argumento.
Ej: texto.include("el");
trim(): quita los espacios en blanco al inicio y final de la cadena
split(arg): convierte la cadena en un array usando como separador lo pasado como argumento

IES MARQUES DE COMARES

Cadenas

- **Concatenación e interpolación en cadenas**

La concatenación de cadenas podemos hacerla con el operador “+”

Ej:

```
let marca=“Ford”; let modelo=“Mondeo”;
console.log (“mi coche es de la marca “+marca+” del modelo  “+modelo+”.”)
```

La interpolación de variables significa meter en una cadena de texto el valor dinámicamente que tiene una variable, lo cual haremos incluyendo ${variable} dentro de las comillas invertidas “ ` ” (template String)

Ej:

```
let marca=“Ford”; let modelo=“Mondeo”;
console.log(`Mi coche es ${marca} del modelo ${modelo}.`);
```

Igualmente, cuando queramos imprimir código HTML en una variable utilizar comillas invertidas.

Cadenas

- **Además las comillas invertidas (template String) me permite introducir tabuladores e intros en las cadenas de texto.**

Ej:

```
ul=`    <ul>
            <li> DWEC</li>
            <li> DWES</li>
            <li>DIW</li>
        </ul>`
```

3- Funciones

Una función es una porción de código reutilizable en cualquier momento

- Declaración de funciones

```
function  [nombrefuncion]([args]){
        Sentencias de código;
        [return valor];
        }
```

Los parámetros primitivos (number, etc) se pasan a las funciones **por valor,** pero si se pasa valor no primitivo (Array, objeto, etc) se pasa por referencia, es decir, si la función cambia las propiedades del objeto, ese cambio es visible fuera de la función

- Invocación de funciones

```
        nombrefuncion(argumentos);
```

IES MARQUES DE COMARES

Funciones

Importante:

Si declaramos una función, ésta automáticamente se declara como global y puede ser invocada en cualquier parte del programa, incluso antes de su declaración.

Sin embargo, si declaramos una función como función Expresada y no declarada, ya sí que debe ser invocada después de su declaración.

Ej:

```
funcionDeclarada(); //invocación correcta
function funcionDeclarada(){console.log("funcion declarada")};
funcionExpresada();  //invocación errónea
funcionExpresada =function (){console.log("función expresada")};    //se declara como función anónima
funcionExpresada();// invocación correcta
```

4- Arrays

Declaración: Se pueden declarar con const, var o let

- **const vector=[]; //array vacío**
- **var vector2=[“a”,2,”b”,3,[4,”c”,4]]; //array inicializado**
- **let vector3=Array.of(“A”,”B”,”C”); //a partir de ESC6**
- **let vector4=Array(5).fill(“A”);**
- **const vector5=new Array(“A”,2,”C”); // ya en desuso**

Todos los arrays comienzan con la posición 0. Para acceder a cualquiera de los elementos del array utilizamos la notación nombrrearray[pos-1]. Para acceder a un array dentro de otro array, accedemos con doble corchete y asi sucesivamente dependiendo de los arrays anidados.

Ej con vector2: **console.log(vector2[4][1])imprimiría la letra c**

IES MARQUES DE COMARES

Arrays

Agregar elementos

Para agregar elementos al final podemos utilizar el método push

Ej: vector2.push(5);

Para quitar elementos del final podemos utilizar el método pop.

Ej: vector2.pop();

Nota: Podemos ver todos los métodos aplicables en el prototipo en la consola.

Arrays

Metodo interesante forEach

Permite ejecutar una función por cada uno de los elementos del array.

Ej:

```
let vector3= Array.of("A","B","C");
//cuidado con las comillas invertidas para usar interpolación
vector3.forEach(function(el,index){
            console.log(`<li id='${index}'>${el}</li>`);});
```