

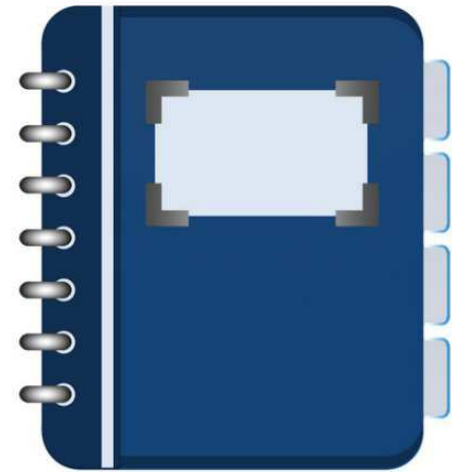
# Tema 5.1

# Objetos JavaScript

# Introducción y Objeto String



# Definición



Entre los objetos de Java Script distinguiremos los objetos predefinidos del lenguaje y los objetos del navegador que se crean cuando el cliente WWW carga un documento HTML.

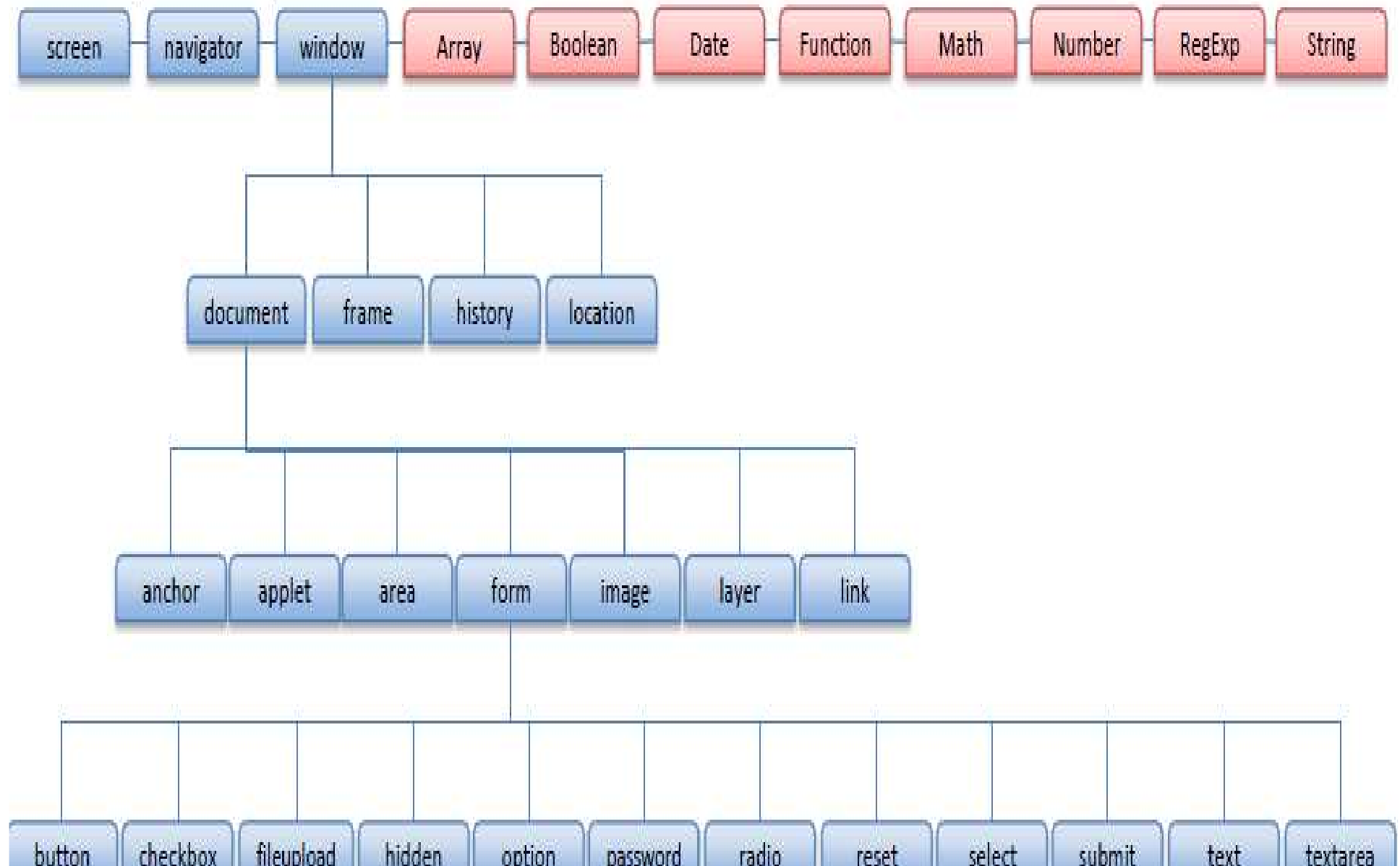
Los objetos tienen PROPIEDADES y METODOS.

# Referencia

En JavaScript se accede a las propiedades y a los métodos de los objetos mediante el operador punto (“.”):

- `mi_objeto.nombre_propiedad;`
- `mi_objeto.nombre_función([parámetros]);`

# Jerarquía



# Objetos del lenguaje:STRING

- `s_obj = new String("foo");` // crea un objeto String
- `s_prim = String("bar");` //crea una Cadena primitiva
- `s_also_prim = "foo";` // crea una Cadena primitiva

Javascript convierte automáticamente entre cadenas primitivas y objetos de cadena, podemos llamar a cualquiera de los métodos del objeto String en una cadena primitiva. JavaScript convierte automáticamente la cadena primitiva en un objeto String.

- El objeto String – Métodos y propiedades:

Métodos				Propiedades
anchor()	fixed()	link()	strike()	Lenght
big()	fontcolor()	match()	sub()	
blink()	fontsize()	replace()	substr()	
bold()	fromCharCode() )	search()	substring()	
charAt()	indexOf()	slice()	sup()	
charCodeAt()	italics()	small()	toLowerCase()	
concat()	lastIndexOf()	split()	toUpperCase()	

# toString

- Conversión de un array en una cadena.

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.toString();
```

The result of *fruits* will be:

```
Banana,Orange,Apple,Mango
```

# Objetos del lenguaje:STRING

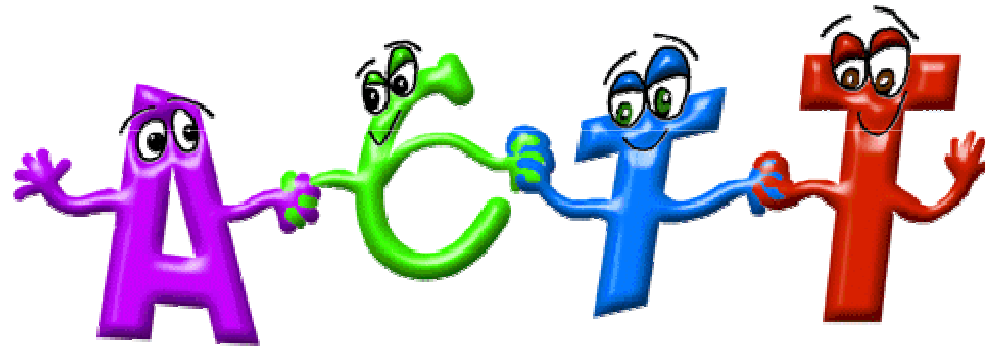
El objeto **string** posee una sola propiedad, la propiedad ***length***, que contiene el tamaño de la cadena de caracteres representada por dicho objeto.





# String

- Los métodos asociados al objeto **string** se pueden clasificar en dos categorías



- Métodos de **Manipulación**
- Métodos de **Formato**

# String-Métodos de manipulación

## Acceder a caracteres individuales de una cadena

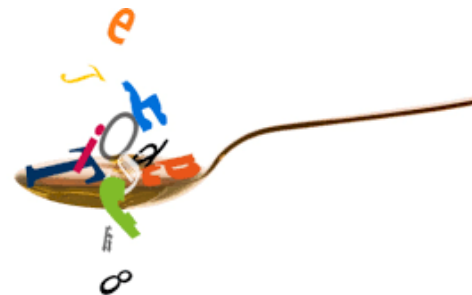
- **charAt(índice)**

Extrae un carácter representado por el índice especificado. Índice está entre 0 y *length*-1.

`'gato'.charAt(1) // devuelve "a"`

El otro modo es tratar la cadena como un arreglo (vector), donde cada índice se corresponde con un carácter individual:

`'gato'[1] // devuelve "a"`



# Comparación de cadenas

- Sólo se usan los operadores menor que y mayor que:

```
1  var a = "a";
2  var b = "b";
3  if (a < b) // true
4      document.write(a + " es menor que " + b);
5  else if (a > b)
6      document.write(a + " es mayor que " + b);
7  else
8      document.write(a + " y " + b + " son iguales.");
```

# String-Métodos de manipulación

- **charCodeAt(indice)**

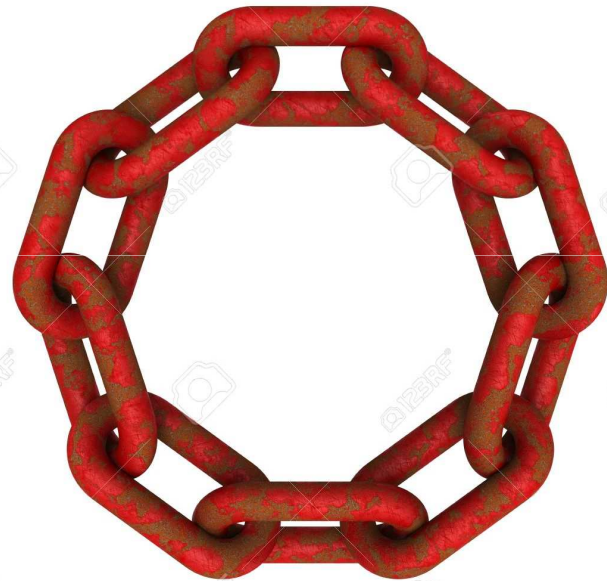
Nos devuelve el valor Unicode del carácter que este en la posición que le enviemos.

- **fromCharCode(valor unicode)**

Nos devuelve el carácter representado por el valor unicode que le enviemos.

# String

- **Concat()**
- Concatena dos cadenas
- ```
var str1 = "Hello ";  
var str2 = "world!";  
var res = str1.concat(str2);
```



# String

- **indexOf(carácter)**
- **indexOf(carácter, índice)**
- Devuelve el índice de la primera ocurrencia del carácter. También se le puede especificar el lugar por el que se quiere que empiece a buscar.

# String

- **lastIndexOf(cadena de caracteres)**
- Devuelve la última ocurrencia de la cadena de caracteres.

substring  
bstrin  
str

## String

- **substring(indice principio, indice fin)**

Extrae una cadena de caracteres entre un valor de índice inicial y otro final.

- **Substr(indice inicial , número de caracteres)**

Extrae una cadena de caracteres desde un valor de índice inicial y número de caracteres que quiero extraer.

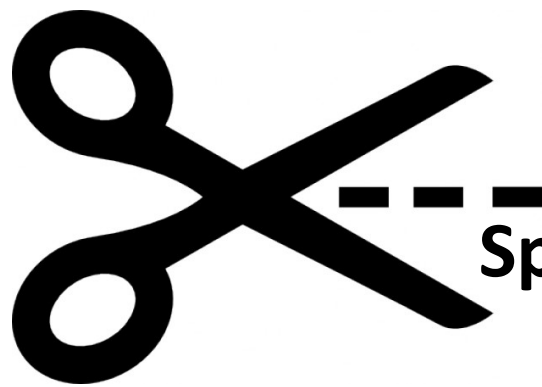
substring  
bstrin  
str



# String

- **toLowerCase()** Transforma una cadena de caracteres en minúsculas.
- **toUpperCase()** Transforma una cadena de caracteres en mayúsculas.





# String

## Split (delimitador)

- Nos parte la cadena en tantos elementos como coincidencias haya con el delimitador propuesto. Con estos elementos nos crea un array.
- `var nombre="Pepe Gómez Ruíz"`
- `Cortar=nombre.split(" ");`
- `cortar[0]`-→Pepe
- `Cortar[1]`-→Gómez

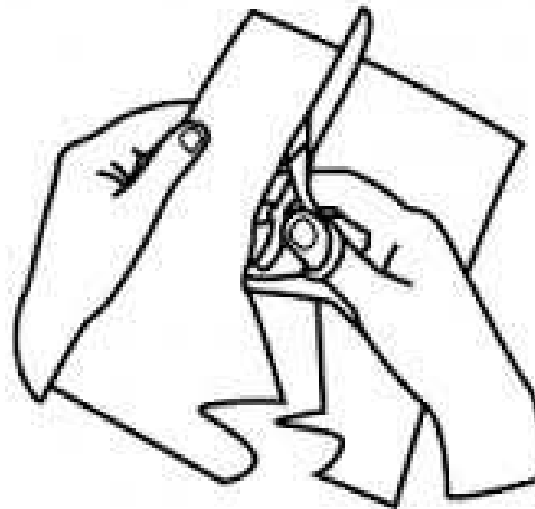
# método replace()

- Busca un valor dentro de una cadena y devuelve una nueva cadena en la que se sustituyen los valores especificados por un nuevo valor indicado previamente.

```
var cadena = "cadena de texto",  
    patron = "texto",  
    nuevoValor = "oro",  
    nuevaCadena = cadena.replace(patron, nuevoValor);  
  
console.log(nuevaCadena); // la consola devolverá: cadena de oro
```

# Eliminar espacios en blanco

- *cadena.trim()*
- *cadena.trimLeft()*
- *cadena.trimRight()*



# Ejercicios cortos



- Disponemos de una matriz de datos llamada **minus** con los días de la semana escritos en minúscula. A partir de minus crear otro array **mayus** con las cadenas al siguiente formato:

LuneS.....(primera y ultima letra en mayúsculas)

- Crea un script que identifique el número de palabras que hay en este texto: “por cien cañones por banda poema de Espronceda” y después le añada un punto al final de la frase antes del nombre del autor.



# Ejercicio String

- Realizar un script que nos pide la introducción de una cadena por teclado que tenga mas de 10 caracteres y a continuación visualizar:
  - 1.El octavo carácter
  - 2.En que posición aparece la primera vez ese carácter.
  - 3.En que posición aparece la última vez ese carácter.
  - 4.Extraer la cadena de caracteres que está entre la primera y la última aparición de “nuestro carácter”.
  - 5.Transformar en esa cadena los caracteres pares a mayúsculas y los impares a minúsculas. Visualizar la cadena completa.

# Ejercicio String



- Introducir un texto por teclado. Cada frase estará del punto.
- Extraer las frases que existan.(.)
- Visualizar
- Visualizar cuantas frases hay.
- Extraer las palabras y la cantidad que hay en cada frase.
- Visualizar
- ¿qué frase es la más larga?
- Ahora elegimos por teclado una frase (número de frase) y extraemos sus palabras. Elegimos una palabra y la deletreamos  
Visualizar en una tabla el deletreo.
- Visualizar la palabra a la inversa en otra tabla.

# String-Métodos de formato

- Los siguientes métodos permiten insertar cadenas de caracteres en marcas HTML. Sirven para generar código HTML mediante scripts.



# String-Métodos de formato

- **anchor(etiqueta)** Transforma la cadena de caracteres en un ancla. Etiqueta es la cadena de caracteres que llevaría el atributo NAME.

# String-Métodos de formato

- **sub()** Inserta la cadena de caracteres en la marca SUB.
- **sup()** Inserta la cadena de caracteres en la marca SUP.

# String-Métodos de formato

- Ejemplo de uso:

```
<script language="JavaScript">  
var superior = "i+j";  
var expr = "(a+b)" + superior.sup();  
document.write(expr);  
</script>
```

# String-Métodos de formato

- Realizar una pequeña aplicación probando los distintos métodos de formato.