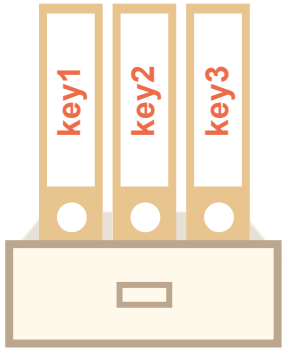


Objetos predefinidos en JavaScript

- Y algo más -

- (Objetos de usuario y funciones) -

Chema Durán

Objetos predefinidos - Boolean

Instanciación

```
// Convierte un valor no booleano a un valor booleano
a = new Boolean(); // asigna a 'a' el valor 'false'
a = new Boolean(0); // asigna a 'a' el valor 'false'
a = new Boolean(""); // asigna a 'a' el valor 'false'
a = new Boolean(false); // asigna a 'a' el valor 'false'
a = new Boolean(numero_distinto_de_0); // asigna a 'a' el valor 'true'
a = new Boolean(true); // asigna a 'a' el valor 'true'
```

Propiedades

Propiedad	Descripción
constructor	Devuelve la función que creó el objeto Boolean.
prototype	Te permitirá añadir propiedades y métodos a un objeto.

Métodos

Método	Descripción
toString()	Convierte un valor Boolean a una cadena y devuelve el resultado.

Objetos predefinidos - Date

Instanciación

```
var d = new Date();
var d = new Date(milisegundos);
var d = new Date(cadena de Fecha);
var d = new Date(año, mes, día, horas, minutos, segundos, milisegundos);
// (el mes comienza en 0, Enero sería 0, Febrero 1, etc.)
```

Objetos predefinidos - Date

Propiedades

Propiedad	Descripción
constructor	Devuelve la función que creó el objeto Date.
prototype	Te permitirá añadir propiedades y métodos a un objeto.

Métodos

Método	Descripción
getDate()	Devuelve el día del mes (de 1-31).
getDay()	Devuelve el día de la semana (de 0-6).
getFullYear()	Devuelve el año (4 dígitos).
getHours()	Devuelve la hora (de 0-23).
getMilliseconds()	Devuelve los milisegundos (de0-999).
getMinutes()	Devuelve los minutos (de 0-59).
getMonth()	Devuelve el mes (de 0-11).
getSeconds()	Devuelve los segundos (de 0-59).

Objetos predefinidos - Date

Métodos

Método	Descripción
getTime()	Devuelve los milisegundos desde media noche del 1 de Enero de 1970.
getTimezoneOffset()	Devuelve la diferencia de tiempo entre GMT y la hora local, en minutos.
getUTCDate()	Devuelve el día del mes en base a la hora UTC (de 1-31).
getUTCDay()	Devuelve el día de la semana en base a la hora UTC (de 0-6).
getUTCFullYear()	Devuelve el año en base a la hora UTC (4 dígitos).
setDate()	Ajusta el día del mes del objeto (de 1-31).
setFullYear()	Ajusta el año del objeto (4 dígitos).
setHours()	Ajusta la hora del objeto (de 0-23).

Objetos predefinidos - Date

Métodos

Método	Descripción
getTime()	Devuelve los milisegundos desde media noche del 1 de Enero de 1970.
getTimezoneOffset()	Devuelve la diferencia de tiempo entre GMT y la hora local, en minutos.
getUTCDate()	Devuelve el día del mes en base a la hora UTC (de 1-31).
getUTCDay()	Devuelve el día de la semana en base a la hora UTC (de 0-6).
getUTCFullYear()	Devuelve el año en base a la hora UTC (4 dígitos).
setDate()	Ajusta el día del mes del objeto (de 1-31).
setFullYear()	Ajusta el año del objeto (4 dígitos).
setHours()	Ajusta la hora del objeto (de 0-23).

Objetos predefinidos - Date

Ejercicio: U3T1 - Date

- **Crea un programa que pida el número de días que quedan desde hoy hasta el fin de curso (por ejemplo, el 24 de junio).**

Recuerda que los meses empiezan desde el número 0.

Objetos predefinidos - Date

Ejercicio: U3T1 - Date

- **Crea un programa que pida por parámetro tu cumpleaños (no hace falta el año) y saque todos los años en que tu cumpleaños va a caer en domingo desde este año hasta el año 2100.**

Recuerda que los meses empiezan desde el número 0.

Objetos predefinidos - Date

Ejercicio: U3T1 - Date

- **Crea un programa que muestre la fecha actual en diferentes formatos, según el valor que meta el usuario por parámetro:**

- **17/02/2016**
- **Miércoles, 17 de febrero de 2016.**
- **Wednesday, February 17, 2016.**

Objetos predefinidos - Date

Ejercicio: U3T1 - Date

- Crea un programa que muestre la hora actual en diferentes formatos, según el valor que meta el usuario por parámetro:

- 14:35:07 (hora detallada con minutos y segundos)
- 02:35 PM o 02:35:07 AM (hora con minutos y AM o PM según sea antes o después del medio día).

Objetos predefinidos - Math

Instanciación

```
// No es un constructor, no podemos crear objetos de tipo Math.
// Pero sí podemos  llamar a propiedades o métodos.

var pi = Math.PI;
var x= Math.sqrt(16);
```

Objetos predefinidos - Math

Propiedades

Propiedad	Descripción
E	Devuelve el número Euler (aproximadamente 2.718).
LN2	Devuelve el logaritmo neperiano de 2 (aproximadamente 0.693).
LN10	Devuelve el logaritmo neperiano de 10 (aproximadamente 2.302).
LOG2E	Devuelve el logaritmo base 2 de E (aproximadamente 1.442).
LOG10E	Devuelve el logaritmo base 10 de E (aproximadamente 0.434).
PI	Devuelve el número PI (aproximadamente 3.14159).
SQRT2	Devuelve la raíz cuadrada de 2 (aproximadamente 1.414).

Objetos predefinidos - Math

Métodos

Método	Descripción
abs(x)	Devuelve el valor absoluto de x.
acos(x)	Devuelve el arcocoseno de x, en radianes.
asin(x)	Devuelve el arcoseno de x, en radianes.
atan(x)	Devuelve el arcotangente de x, en radianes con un valor entre -PI/2 y PI/2.
atan2(y,x)	Devuelve el arcotangente del cociente de sus argumentos.
ceil(x)	Devuelve el número x redondeado al alta hacia el siguiente entero.
cos(x)	Devuelve el coseno de x (x está en radianes).
floor(x)	Devuelve el número x redondeado a la baja hacia el anterior entero.
log(x)	Devuelve el logaritmo neperiando (base E) de x.

Objetos predefinidos - Math

Métodos

Método	Descripción
abs(x)	Devuelve el valor absoluto de x.
acos(x)	Devuelve el arcocoseno de x, en radianes.
asin(x)	Devuelve el arcoseno de x, en radianes.
atan(x)	Devuelve el arcotangente de x, en radianes con un valor entre -PI/2 y PI/2.
atan2(y,x)	Devuelve el arcotangente del cociente de sus argumentos.
ceil(x)	Devuelve el número x redondeado al alta hacia el siguiente entero.
cos(x)	Devuelve el coseno de x (x está en radianes).
floor(x)	Devuelve el número x redondeado a la baja hacia el anterior entero.
log(x)	Devuelve el logaritmo neperiando (base E) de x.

Objetos predefinidos - Math

Ejercicio: U3T2 - Math

- Crea un programa que pida al usuario que elija una opción del siguiente menú:
- 1) Potencia.
- 2) Raíz.
- 3) Redondeo.
- 4) Trigonometría.
- Si el usuario introduce 1, se le deberá pedir una base y un exponente y se mostrará el resultado en pantalla (La potencia de X elevado a Y es:)
- Si el usuario introduce 2, se le pedirá un número (no negativo) y se mostrará el resultado en pantalla (La raíz de X es:)
- Si el usuario introduce 3, se le pedirá un decimal por pantalla y se mostrará el redondeo al entero más próximo, al alta y a la baja.
- Si el usuario introduce 4, se le pedirá un ángulo (entre 0 y 360) y se le mostrarán por pantalla los valores trigonométricos del seno, coseno y tangente.

Objetos predefinidos - Math

Ejercicio: U3T2 - Math

- Crea un programa que pida al usuario el valor del radio y muestre por pantalla:
- El valor del radio.
- El valor del diámetro.
- El valor del perímetro de la circunferencia.
- El valor del área del círculo.
- El valor del área de la esfera.
- El valor del volumen de la esfera.
- El valor de Pi debes obtenerlo del objeto Math, no introducirlo manualmente.
- Debes escribir al lado si son cm, o cm2, o cm3.

- Como datos de muestra, si metes 5, deberías obtener aproximadamente: 5 / 10 / 31,41 / 78,54 / 314,15 / 523,59.

Objetos predefinidos - Number

Instanciación

```
// No se suele utilizar: normalmente asignamos valores numéricos
// a una variable:

var num = new Number (value);

// Si el parámetro que se pasa al constructor no se puede
// convertir devuelve NaN.
```

Objetos predefinidos - Number

Propiedades

Propiedad	Descripción
constructor	Devuelve la función que creó el objeto Number.
MAX_VALUE	Devuelve el número más alto disponible en JavaScript.
MIN_VALUE	Devuelve el número más pequeño disponible en JavaScript.
NEGATIVE_INFINITY	Representa a infinito negativo (se devuelve en caso de overflow).
POSITIVE_INFINITY	Representa a infinito positivo (se devuelve en caso de overflow).
prototype	Permite añadir nuestras propias propiedades y métodos a un objeto.

Objetos predefinidos - Number

Métodos

Método	Descripción
toExponential(x)	Convierte unnúmero a su notación exponencial.
toFixed(x)	Formatea un número con x digitos decimales después del punto decimal.
toPrecision(x)	Formatea un número a la longitud x.
toString()	Convierte un objeto Number en una cadena. • Si se pone 2 como parámetro se mostrará el número enbinario. • Si se pone 8 como parámetro se mostrará el número enoctal. • Si se pone 16 como parámetro se mostrará el número en hexadecimal.
valueOf()	Devuelve el valor primitivo de un objeto Number.

Objetos predefinidos - Number

Ejercicio: U3T3 - Number

Crea un programa que pida al usuario un número entero por pantalla y muestre:

- **Su valor exponencial.**
- **El número con 4 decimales.**
- **El número en binario.**
- **El número en octal.**
- **El número en hexadecimal.**
- **Utiliza para ello los métodos del objeto Number.**

Como datos de muestra, si metes 50, deberías obtener: 5e1 / 50.0000 / 00110010 / 62 / 0x32.

Objetos predefinidos - String

Instanciación

```
// Podemos utilizar dos maneras:
var txt = new String("string");
var txt = "string";

// Podemos utilizar comillas dobles o simples:
var cadena = '<input type="checkbox" name="coche" />Audi A6';
var cadena = "<input type='checkbox' name='coche' />Audi A6";

// Podemos utilizar caracteres de escape:
var cadena = "<input type=\"checkbox\" name=\"coche\" />Audi A6";
```

Objetos predefinidos - String

Instanciación

```
// Podemos concatenar cadenas muy largas con +=
var nuevoDocumento = "";
nuevoDocumento += "<!DOCTYPE html>";
nuevoDocumento += "<html>" ;
nuevoDocumento += "<head>";
nuevoDocumento += '<meta http-equiv="content-type"';
nuevoDocumento += ' content="text/html;charset=utf-8">';

// Podemos concatenar cadenas con variables:
nombreEquipo = prompt("Introduce el nombre de tu equipo favorito:","");
var mensaje= "El " + nombreEquipo + " ha sido el campeón de la Copa del Rey!";
alert(mensaje);
```

Objetos predefinidos - String

Propiedades

Propiedad	Descripción
length	Contiene la longitud de una cadena.

Métodos

Método	Descripción
charAt()	Devuelve el carácter especificado por la posición que se indica entre paréntesis.
charCodeAt()	Devuelve el Unicode del carácter especificado por la posición que se indica entre paréntesis.
concat()	Une una o más cadenas y devuelve el resultado de esa unión.
fromCharCode()	Convierte valores Unicode a caracteres.
indexOf()	Devuelve la posición de la primera ocurrencia del carácter buscado en la cadena.
lastIndexOf()	Devuelve la posición de la última ocurrencia del carácter buscado en la cadena.

Objetos predefinidos - String

Métodos

Método	Descripción
match()	Busca una coincidencia entre una expresión regular y una cadena y devuelve las coincidencias o null si no ha encontrado nada.
replace()	Busca una subcadena en la cadena y la reemplaza por la nueva cadena especificada.
search()	Busca una subcadena en la cadena y devuelve la posición dónde se encontró.
slice()	Extrae una parte de la cadena y devuelve una nueva cadena.
split()	Divide una cadena en un array de subcadenas.
substr()	Extrae los caracteres de una cadena, comenzando en una determinada posición y con el número de caracteres indicado.
substring()	Extrae los caracteres de una cadena entre dos índices especificados.
toLowerCase()	Convierte una cadena en minúsculas.
toUpperCase()	Convierte una cadena en mayúsculas.

Objetos predefinidos - String

Ejercicio: U3T4 - String

Crea un programa que pida al usuario su nombre y apellidos y muestre:

- **El tamaño del nombre más los apellidos (sin contar espacios).**
- **La cadena en minúsculas y en mayúsculas.**
- **Que divida el nombre y los apellidos y los muestre en 3 líneas, donde ponga Nombre: / Apellido 1: / Apellido 2:**
- **Una propuesta de nombre de usuario, compuesto por la inicial del nombre, el primer apellido y la inicial del segundo apellido: ej. Para José María García Durán sería jgarciad.**
- **Una propuesta de nombre de usuario compuesto por las tres primeras letras del nombre y de los dos apellidos: ej. josgardur.**

Objetos predefinidos - String

Ejercicio: U3T4 - String

Crea un programa que pida al usuario una propuesta de contraseña y compruebe si cumple con los siguientes requisitos.

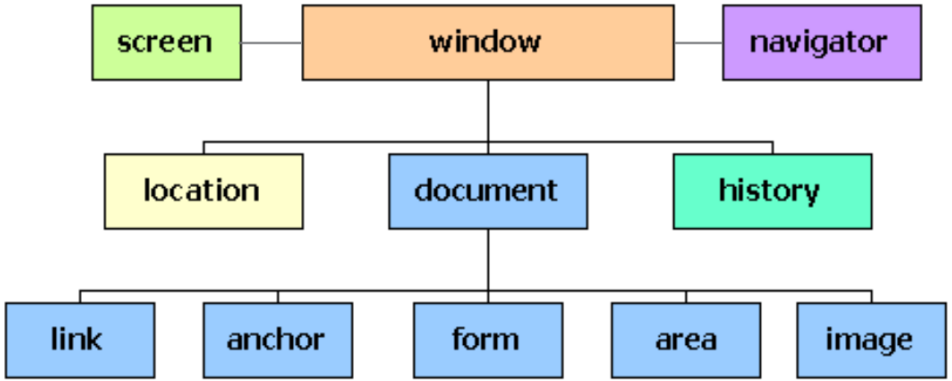
- **Tiene entre 8 y 16 caracteres.**
- **Tiene una letra mayúscula.**
- **Tiene una letra minúscula.**
- **Tiene un número.**
- **Tiene uno de los siguientes valores: guión alto, guión bajo, arroba, almohadilla, dólar, tanto por ciento o ampersand.**

Si cumple con todos los requisitos se considera una contraseña segura, de lo contrario mostrará que es una contraseña no segura.

Objetos predefinidos - Window

Introducción

Mediante el Browse Object Model (BOM) se podían modificar las propiedades del navegador. Con Document Object Model (DOM) no hay ninguna entidad que trate de estandarizarlo.



Objetos predefinidos - Window

Acceso a propiedades y métodos

```
window.nombrePropiedad
window.nombreMetodo( [parámetros] )

// Como contiene el resto de objetos podemos omitir su nombre:
nombrePropiedad
nombreMetodo( [parámetros] )
```

Objetos predefinidos - Window

Propiedades

Propiedad	Descripción
closed	Devuelve un valor Boolean indicando cuando una ventana ha sido cerrada o no.
defaultStatus	Ajusta o devuelve el valor por defecto de la barra de estado de una ventana.
document	Devuelve el objeto document para la ventana.
frames	Devuelve un array de todos los marcos (incluidos iframes) de la ventana actual.
history	Devuelve el objeto history de la ventana.
length	Devuelve el número de frames (incluyendo iframes) que hay en dentro de una ventana.
location	Devuelve la Localización del objeto ventana (URL del fichero).

Objetos predefinidos - Window

Propiedades

Propiedad	Descripción
name	Ajusta o devuelve el nombre de una ventana.
navigator	Devuelve el objeto navigator de una ventana.
opener	Devuelve la referencia a la ventana que abrió la ventana actual.
parent	Devuelve la ventana padre de la ventana actual.
self / window	Devuelve la ventana actual.
status	Ajusta el texto de la barra de estado de una ventana.
top	Nombre alternativo de la ventana del nivel superior

Objetos predefinidos - Window

Métodos

Método	Descripción
alert(mensaje)	Muestra una ventana emergente de alerta y un botón de aceptar.
blur()	Elimina el foco de la ventana actual.
clearInterval(id)	Resetea el cronómetro ajustado con setInterval().
setInterval(expresion,tiempo)	Llama a una función o evalúa una expresión en un intervalo especificado (en milisegundos).
clearTimeOut(nombre)	Cancela el intervalo referenciado por 'nombre'.
setTimeOut(expresion,tiempo)	Evalua la expresión después de que hayan pasado un intervalo de tiempo(en milisegundos).
close()	Cierra la ventana actual.
confirm(mensaje)	Muestra una ventana emergente con un mensaje, un botónde aceptar y un botón de cancelar.

Objetos predefinidos - Window

Métodos

Método	Descripción
focus()	Coloca el foco en la ventana actual.
moveBy(x,y)	Mueve la ventana actual el número de píxeles especificados.
moveTo(x,y)	Mueve la ventana a la posición especificada.
open(url,nombre,características)	Abre una nueva ventana de navegación.
prompt(mensaje,respuesta_defecto)	Muestra una ventana de diálogo para introducir datos.
scrollBy(x,y)	Mueve el scroll de la ventana el número de píxeles indicados.
scrollTo(x,y)	Mueve el scroll de la ventana a la posición indicada.

Objetos predefinidos - Window

Características

Método	Descripción
toolbar = [yes|no|1|0]	Indica si la ventana tendrá o no barra de herramientas (yes=1; no=0).
location = [yes|no|1|0]	Indica si la ventana tendrá campo de localización o no (yes=1; no=0).
directories = [yes|no|1|0]	Indica si la ventana tendrá botones de dirección o no (yes=1; no=0).
status = [yes|no|1|0]	Indica si la ventana tendrá barra de estado o no (yes=1; no=0).
menubar = [yes|no|1|0]	Indica si la ventana tendrá barra de menús o no (yes=1; no=0).
scrollbars = [yes|no|1|	Indica si la ventana tendrá barras de desplazamiento o no (yes=1; no=0).
resizable = [yes|no|1|0].	Indica si la ventana se podrá redimensionar o no (yes=1; no=0).
width = px / heigth = px	Ancho y alto de la ventana en píxeles
outerWidth = px / outerHeigth = px	Ancho y alto total de la ventana en píxeles.
top = px / left = px	Distancia de la ventana desde la parte superior e izquierda en píxeles.

Objetos predefinidos - Window

Ejercicio: U3T5 - Window

Crea un programa que tenga botones para permitir modificar las siguientes propiedades de una ventana:

Abrir una ventana nueva:

- Debes preguntar al usuario si está de acuerdo o no, y solo si acepta se abrirá la nueva ventana.
- La nueva ventana tendrá las siguientes propiedades: no tendrá barrade herramientas, ni location, ni barra de menú, ni será redimensionable. Tendrá 200x80 píxeles y se posicionará en 500x500 píxeles.
- La nueva ventana incluirá un pequeño texto y un botón que al hacer clic cerrará la ventana.
- Cerrar la ventana creada: si la ventana está cerrada mostrará un mensaje de error.
- Mover la ventana 10 píxeles a la derecha y abajo.
- Mover la ventana a la posición 100,100.
- Aumentar el tamaño de la ventana 10 píxeles de ancho y largo.
- Aumentar el tamaño de la ventana a 400x200.
- Colocar el scroll de la ventana arriba del todo
- Colocar el scroll de la ventana a 10 píxeles de la parte superior.

Todos los botones, excepto el primero y el segundo, los puedes programar directamente mediante la propiedad onClick, por ejemplo:

<input type="button" value="Imprimir" onClick="print()"/>

Recuerda que no es necesario utilizar "window" delante de la propiedad.

Arrays

Colección de valores homogénea o no, y ordenada.

Crear un array usando el constructor del objeto Array:

```
miArray = new Array(“Hola”,30, 3.14); // elementos heterogéneos
miArray2 = new Array(“Viento”, “Lluvia”, “Fuego”, “Tierra”); //elementos  homogéneos
miArray3 = new Array(5); // Array vacío con 5 elementos
```

Crear un array usando corchetes:

```
var frutas = [”Manzana", "Platano"];
console.log(frutas.length); // 2
```

Objetos predefinidos - Arrays

Arrays

Acceder a un elemento de un array:

```
var primero = frutas[0]; // Manzana
var ultimo = frutas[frutas.length - 1]; // Platano
```

Recorrer un array:

```
frutas.forEach(function (item, index, array) {
  console.log(item, index);
});
// Manzana 0
// Platano 1
```

Objetos predefinidos - Arrays

Propiedades:

Propiedad	Descripción
length	Devuelve la longitud del array
prototype	Permite agregar al objeto Array las propiedades que queramos.

Ejemplo de prototype:

```
Array.protoype.descriptor = null;
dias = new Array ('lunes', 'Martes', 'Miercoles', 'Jueves', 'Viernes');
dias.descriptor = "Dias laborables de la semana";
```

Objetos predefinidos - Arrays

Métodos:

Método	Descripción
valueOf(elemento)	Es el método por defecto de un objeto array y lo devuelve.
indexOf(elemento)	Devuelve la primera posición en la que se encuentra un elemento.
push(elemento)	Añade un elemento al final del array (aumenta su longitud).
pop()	Elimina un elemento al final del array y lo devuelve (disminuye su longitud)
unshift(elemento)	Añade un elemento al principio del array (aumenta su longitud)
shift()	Elimina el primer elemento del array y lo devuelve (disminuye su longitud).
reverse()	Modifica el array colocando los elementos en orden inverso.
sort()	Ordena los elementos del array según orden lexicográfico.

Objetos predefinidos - Arrays

Métodos:

Método	Descripción
concat(elemento/s)	Concatena elementos de varios arrays.
join(separador)	Une todos los elementos en una cadena de texto (lo contrario que el split) unidos por un separador.
slice(inicio,fin)	Toma elementos de un array (sin modificar el original) desde la posición inicio a fin.
splice (inicio,numero_elementos,[reemplazo])	Elimina un número de elementos de un array empezando por la posición inicio. SI tiene el tercer parámetro, además, los sustituye por él.

Objetos predefinidos - Arrays

Ejemplos de uso: Partimos del array `frutas[“Manzana”, “Platano”];`

Obtener la posición de un elemento del array.

```
var posicion= frutas.valueOf(”Platano");
// Devuelve 1, porque plátano es la 2ª posición, pero empezamos por el 0.
```

Añadir al final del array:

```
var nuevoTamano = frutas.push("Naranja");
// ["Manzana", "Platano", "Naranja"] y devuelve el tamaño del array
```

Eliminar del final del array:

```
var ultimo = frutas.pop(); // borra Naranja (del final) y lo devuelve
// ["Manzana", "Platano"];
```

Añadir al principio del array:

```
var nuevoTamano = frutas.unshift("Fresa") // añade al principio y devuelve
// el  tamaño
// ["Fresa", “Manzana”, "Platano"];
```

Objetos predefinidos - Arrays

Ejemplos de uso:

Eliminar del principio del array:

```
var primero = frutas.shift(); // borra Fresa(del principio) y lo devuelve
// [“Manzana”, "Platano"];
```

Ordena según el orden lexicográfico:

```
var frutas=[“Manzana”,”Platano”,”Kiwi”,”Higo”,”Pera”,”Ciruela”];
var frutasOrdenado = frutas.sort ();
//frutas=[“Ciruela”, “Higo”, “Kiwi”, “Manzana”, “Pera”, “Platano”]
//Devuelve el array ordenado
```

Invierte el orden de los elementos en el array:

```
var frutas=[“Manzana”,”Platano”,”Kiwi”,”Higo”,”Pera”,”Ciruela”];
var frutasReves = frutas.reverse();
//frutas=[“Ciruela”, “Pera”, “Higo”, “Kiwi”, “Platano”, “Manzana”
//Devuelve el array ordenado del revés
```

Objetos predefinidos - Arrays

Ejemplos de uso:

Concatenar dos arrays:

```
var frutas1 = [”Manzana", "Platano"];
var frutas2 = [“Ciruela”];
frutas1.concat[“Kiwi”,”Higo”,”Pera”];
//frutas1=[“Manzana”,”Platano”,”Kiwi”,”Higo”,”Pera”];  frutas1.concat(frutas2);
//frutas1=[“Manzana”,”Platano”,”Kiwi”,”Higo”,”Pera”,”Ciruela”];
```

Unir los elementos de un array en una cadena:

```
var frutas=[“Manzana”,”Platano”,”Kiwi”];
var cadena = frutas.join(“:”);
// cadena = “Manzana:Platano:Kiwi”;
```

Objetos predefinidos - Arrays

Ejemplos de uso:

Tomar elementos según su posición (inicial y final):

```
var frutas=[“Manzana”,”Platano”,”Kiwi”, “Pera”,”Higo”];
var seleccion = frutas.slice(0, 1); // seleccion = [”Manzana”,”Platano"]
var ultimos = frutas.slice(--2); // seleccion = [”Pera”,”Higo"]
```

Eliminar elementos según su posición (inicio, número de elementos):

```
var frutas=[“Manzana”,”Platano”,”Kiwi”, “Pera”,”Higo”];
var eliminado= frutas.splice(2, 2); // frutas = [”Manzana”,”Higo"]
var ultimos = frutas.splice(-1);// frutas = [“Manzana”]
```

Objetos predefinidos - Arrays

Ejercicio U3T6 – Parte 1:

Vamos a gestionar una lista de países haciendo uso de Arrays. Para ello necesitarás crear un archivo arrays.js que incluya las siguientes funciones:

- **Mostrar el número de elementos del array.**
- **Mostrar todos los elementos del array.**
- **Muestra los elementos del array en sentido inverso.**
- **Muestra los elementos del array ordenados alfabéticamente (pero no los ordena).**
- **Añadir un elemento al principio del array.**
- **Añadir un elemento al final del array.**
- **Borrar un elemento al principio del array (y decir cuál se ha borrado).**
- **Borrar un elemento al final del array (y decir cuál se ha borrado).**
- **Muestra el elemento que se encuentra en una posición que el usuario indica.**
- **Muestra la posición en la que se encuentra un elemento que le indica el usuario.**
- **Muestra los elementos que se encuentran en un intervalo que el usuario indica.**

Objetos predefinidos - Arrays

Ejercicio U3T6 – Parte 2:

Ten en cuenta que el array será una variable global y que se pasará por parámetro en todas las funciones. Cuando el usuario cargue la página, se cargará un prompt donde se mostrarán (en el prompt, no en la página) las opciones:

- **Mostrar número de países.**
- **Mostrar listado de países (y le preguntará si quiere mostrarlos en el orden que se encuentran en el array, del revés u ordenados alfabéticamente).**
- **Mostrar un intervalo de países (y le pedirá que introduzca el intervalo en formato inicio-fin; luego deberás extraer el valor inicio y fin).**
- **Añadir un país (y le preguntará si quiere añadir al principio o al final).**
- **Borrar un país (y le preguntará si quiere borrar al principio o al final).**
- **Consultar un país (y le preguntará si quiere consultar por posición o por nombre).**

Todas las operaciones que se realicen se irán mostrando en la página con su título.

Objetos predefinidos – Arrays multidimensionales

Son arrays que almacenan en cada posición otros arrays a su vez. Es posible crear arrays bidimensionales, tridimensionales, etc.

Crear un array bidimensional mediante otros arrays:

```
var datos = new Array();
datos[0] = new Array (“Programacion”, “1DAW”, 10);
datos[1] = new Array (“DWEC”, “2DAW”, 9);
datos[2] = new Array (“DIW”, “2DAW”, 5);
```

Crear un array bidimensional usando corchetes:

```
 var datos = [
     [“Programacion”, “1DAW”, 10],
     [“DWEC”, “2DAW”, 9],
     [“DIW”, “2DAW”, 5]
     ];
```

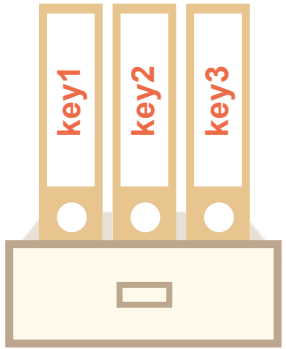
Objetos predefinidos – Arrays multidimensionales

Acceder a una posición de un array multidimensional. Tenemos que indicar tantas posiciones como dimensiones (índices) tenga el array:

```
var elemento = datos [0][2];
```

Imprimir datos de un array multidimensional. Utilizamos tantos bucles for anidados como dimensiones (índices) tenga el array.

```
for (i = 0; i < datos.length; i++) {
  for (j = 0; j < datos[i].length; j++) {
    document.write("Elemento " + i + "," + j + ": " + datos[i][j]);
  }
}
```



Funciones predefinidas en JavaScript

Chema Durán

Funciones predefinidas - eval

eval(cadena [,objeto])

Cadena: **Una cadena que representa una expresión, sentencia o secuencia de sentencias en JavaScript. La expresión puede incluir variables y propiedades de objetos existentes.**

Objeto: **Un argumento opcional; si se especifica, la evaluación se restringe al contexto del objeto especificado.**

```
eval(new String("2 + 2")); // devuelve un objeto String que contiene "2 + 2”
eval("2 + 2"); // devuelve 4
var expresion = new String("2 + 2");
eval(expresion.toString());
```

Funciones predefinidas - parseInt

parseInt(cadena, base);

Cadena: Una cadena que representa el valor que se desea convertir.

Base: Un entero que representa la base de la mencionada cadena.

Si se encuentra con un carácter que no se corresponde con la base especificada devuelve NaN.

Si no se especifica la base o se especifica como 0, JavaScript asume lo siguiente:

Si el parámetro cadena comienza por "0x", la base es 16 (hexadecimal).

Si el parámetro cadena comienza por "0", la base es de 8 (octal). Esta característica está desaconsejada.

Si el parámetro cadena comienza por cualquier otro valor, la base es 10 (decimal).

Funciones predefinidas - parseInt

parseInt(cadena, base);

Ejemplos:

```
parseInt("F", 16); //Devuelve 15
parseInt("17", 8); ; //Devuelve 15
parseInt("15", 10); ; //Devuelve 15
parseInt(15.99, 10); ; //Devuelve 15
parseInt("Hello", 8); // No es un número en absoluto. Devuelve NaN
parseInt("0x7", 10); // No es de base 10. Devuelve NaN
```

Funciones predefinidas - parseFloat

parseFloat(cadena);

Cadena: **Una cadena que representa el valor que se desea convertir.**

Si el primer carácter no se puede convertir a número, devuelve NaN.

Ejemplos:

```
parseFloat("3.14"); // Devuelve 3,14
parseFloat("314e-2"); // Devuelve 3,14
parseFloat("0.0314E+2"); // Devuelve 3,14
var cadena = "3.14"; parseFloat(cadena); // Devuelve 3,14
parseFloat("3.14más caracteres no dígitos"); // Devuelve 3,14
parseFloat("FF2"); // Devuelve NaN
```

Funciones predefinidas - isNaN

isNaN(valor)

Valor: El valor que se desea evaluar.

IsNaN quiere decir: Is Not a Number.

Intenta convertir el argumento a número; si no puede, devuelve true; en caso contrario false.

Ejemplos:

```
isNaN(NaN) // devuelve true
isNaN("string") // devuelve true
isNaN("12") // devuelve false
isNaN(12) // devuelve false
```

Funciones predefinidas - isFinite

isFinite(numero)

Valor: El valor que se desea evaluar.

Si el argumento es NaN, infinito positivo o negativo devuelve false; en caso contrario true.

Ejemplos:

```
isFinite(Infinity); // falso
isFinite(NaN); // falso
isFinite(-Inifinity); //falso
isFinite(0); // verdadero
isFinite(2e64); // verdadero
isFinite("0"); // verdadero, hubiera sido falso en el caso de usar Number.
               // isFinite("0") que es  mas robusta.
```

Funciones predefinidas – Number / String

Number(objeto) / String(objeto)

Objeto: el objeto que quiere pasarse a Number o String.

Convierten el objeto pasado por parámetro a Number o String.

Ejemplos:

```
var hoy = new Date();
document.write(String(hoy));
```

Funciones definidas por el usuario

Son un conjunto de sentencias que realizan una tarea específica.

Conviene definirlas en el `<head>`, y luego llamarlas desde cualquier punto de la web con un script.

```
function nombre([argumentos]) {
    // Sentencias
    // [return dato]
}
```

`function`: **palabra reservada para definir una función.**

`nombre`: **palabra que determina el nombre de la función.**

`argumentos` **(opcional): parámetros que pueden utilizarse internamente en la función.**

`return dato`: **la función puede o no devolver un valor, que estaría indicado después de la palabra reservada return.**

Funciones definidas por el usuario

Ejemplos:

```
function mifuncion(cadena) {
  document.write("<P><BLINK>" + cadena + "</BLINK>");
}

function resta(a, b) {
  return a - b;
}
```

Funciones definidas por el usuario

Ejercicio U3T7:

- **Crea un archivo funciones.js en el que implementes las funciones siguientes (busca la fórmula en Internet):**
- **Imc (Índice de masa corporal)**
- **Fcm (Frecuencia cardíaca máxima)**

- **Crea un html donde incluyas el archivo anterior y dos enlaces. Cada uno (mediante el método onClick) ejecutará una función.**

Objetos definidos por el usuario

En Javascript, un objeto es una colección de propiedades que a su vez pueden ser datos o métodos.

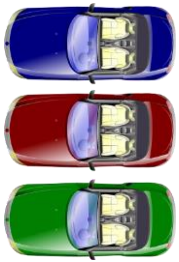
Constructor: es una función especial para crear un objeto. Empieza en mayúscula:

```
function Coche() {
    //Propiedades y métodos
}
```

Creación de un objeto:

```
var unCoche = new Coche();
```

Coche()



```
var cocheazul = new Coche();
var cocherojo = new Coche();
var cocheverde = new Coche();
```

Objetos definidos por el usuario

Definir propiedades de un objeto:

Se crean en el constructor precedidas por la palabra this.

"Constructor" sin parámetros:

```
function Coche() {
  this.marca = ""; // Vacío
  this.modelo = ""; // Vacío
  this.combustible = "Diesel"; // Inicializado
  this.cantidad = 0; //Inicializado
}
```

"Constructor" con parámetros:

```
function Coche(marca, modelo, combustible, cantidad) {
  this.marca = marca;
  this.modelo = marca;
  this.combustible = combustible;
  this.cantidad = cantidad;
  // Cada propiedad toma los valores recibidos por parámetro
}
```

Objetos definidos por el usuario

Definir propiedades de un objeto:

Crear un objeto vacío (sin propiedades):

```
var cocheVacio = new Coche();
```

Cambiar valores en las propiedades:

```
cocheVacio.marca = "Seat";
cocheVacio.modelo = "Ibiza";
cocheVacio.combustible = "Diesel";
cocheVacio.cantidad = 40;
```

Crear un objeto inicializado (con propiedades):

```
var miCoche = new Coche("Seat", "Ibiza", "Diesel", 40);
```

Acceder a las propiedades:

```
document.write(" Mi coche es un " + miCoche.marca + "  " + miCoche.modelo);
```

Objetos definidos por el usuario

Definir métodos de un objeto:

Permiten acceder y modificar las propiedades de los objetos. Empiezan en minúscula.

Método que modifica la cantidad de combustible:

```
function rellenarDeposito(litros) {
  this.cantidad = litros;
}
```

Referencia al método fuera del objeto (¡no recomendado!)

```
function Coche(marca, modelo, combustible, cantidad) {
  //Propiedades
  this.marca = marca;
  this.modelo = marca;
  this.combustible = combustible;

  //Métodos
  this.rellenardeposito = rellenardeposito;
}
```

Objetos definidos por el usuario

Definir métodos de un objeto:

Permiten acceder y modificar las propiedades de los objetos. Empiezan en minúscula.

Método que modifica la cantidad de combustible:

```
function rellenarDeposito(litros) {
  this.cantidad = litros;
}
```

Referencia al método fuera del objeto (¡no recomendado!)

```
function Coche(marca, modelo, combustible, cantidad) {
  //Propiedades
  this.marca = marca;
  this.modelo = marca;
  this.combustible = combustible;

  //Métodos
  this.rellenardeposito = rellenardeposito;
}
```

Objetos predefinidos - Arrays

Definir métodos de un objeto:

Referencia al método dentro del objeto

```
function Coche(marca, modelo, combustible, cantidad) {
  //Propiedades
  this.marca = marca;
  this.modelo = marca;
  this.combustible = combustible;
  //Métodos
  this.rellenardeposito = function (litros) {
    this.cantidad = litros;
  };
}
```

Objetos definidos por el usuario

Objetos literales:

Un literal es un valor fijo. Está formado por parejas de tipo nombre:valor.

Ejemplo:

```
var coche = {marca:"Ibiza", modelo:"Seat", combustible:"diesel", cantidad:40};
```

Ejemplo equivalente:

```
var coche = new Object();
coche.marca = "Ibiza";
coche.modelo = "Seat";
coche.combustible = "diesel";
coche.cantidad = 40;
```

Acceso:

```
coche.marca;
coche["marca"];
```

Objetos definidos por el usuario

Ejercicio U3T8 – Parte 1:

Necesitamos almacenar en un programa todos los discos de música que tenemos en casa. Ahora que sabemos crear nuestros propios objetos es el mejor modo de guardar esta información.
Crea un objeto "disco" que almacene la siguiente información:

- **Nombre del disco.**
- **Grupo de música o cantante.**
- **Año de publicación.**
- **Tipo de música (podrá ser "rock", "pop", "punk" o "indie");**
- **Localización: almacenará un número de estantería.**
- **Prestado: almacenará un valor booleano. Por defecto será false.**

Además tendrá los siguientes métodos:

- **Un "constructor" sin parámetros (las 4 primeras propiedades serán cadenas vacías, la localización será 0 por defecto y prestado estará a false).**
- **Un método que permita incluir las cinco primeras propiedades; la propiedad prestado seguirá a false.**
- **Un método que permitirá cambiar el número de estantería en la localización.**
- **Un método que permitirá cambiar la propiedad Prestado.**
- **Un método que muestre toda la información de un disco.**

Guarda todo el código en un archivo llamado disco.js

Objetos definidos por el usuario

Ejercicio U3T9 – Parte 2:

Carga en tu página el archivo de arrays que hicimos en la práctica anterior.
Crea un array vacío para almacenar los discos.
Cuando el usuario cargue la página, se cargarán las opciones:

- **Mostrar número de discos.**
- **Mostrar listado de discos(y le preguntará si quiere mostrarlos en el orden que se encuentran en el array, del revés u ordenados alfabéticamente).**
- **Mostrar un intervalo de discos(y le pedirá que introduzca el intervalo en formato inicio-fin; luego deberás extraer el valor inicio y fin).**
- **Añadir un disco (y le preguntará si quiere añadir al principio o al final).**
- **Borrar un disco (y le preguntará si quiere borrar al principio o al final).**
- **Consultar un disco (y le preguntará si quiere consultar por posición o por nombre).**

Todas las operaciones que se realicen se irán mostrando en la página con su título.

REUTILIZA EL CÓDIGO DE LA PRÁCTICA ANTERIOR.

END

prof.jduran@iesalixar.org