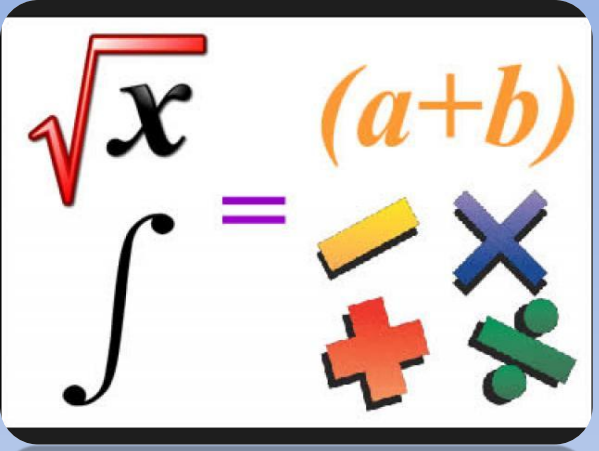
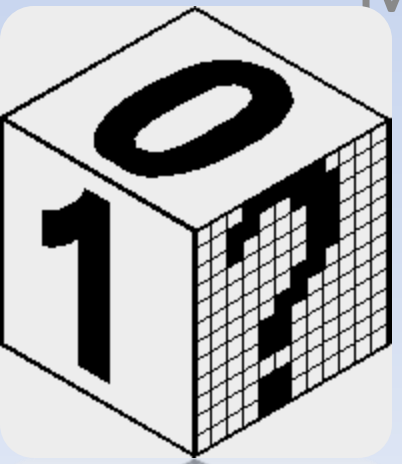
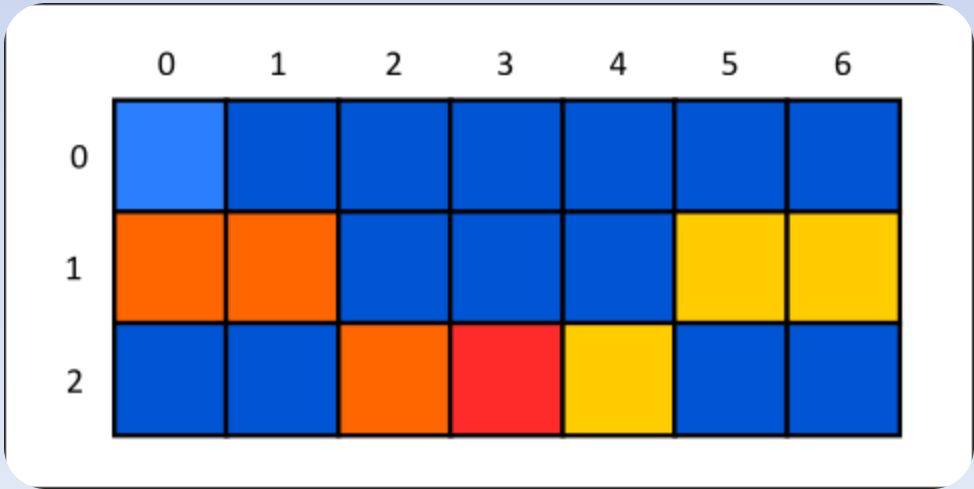


Objetos del lenguaje Javascript 5.2

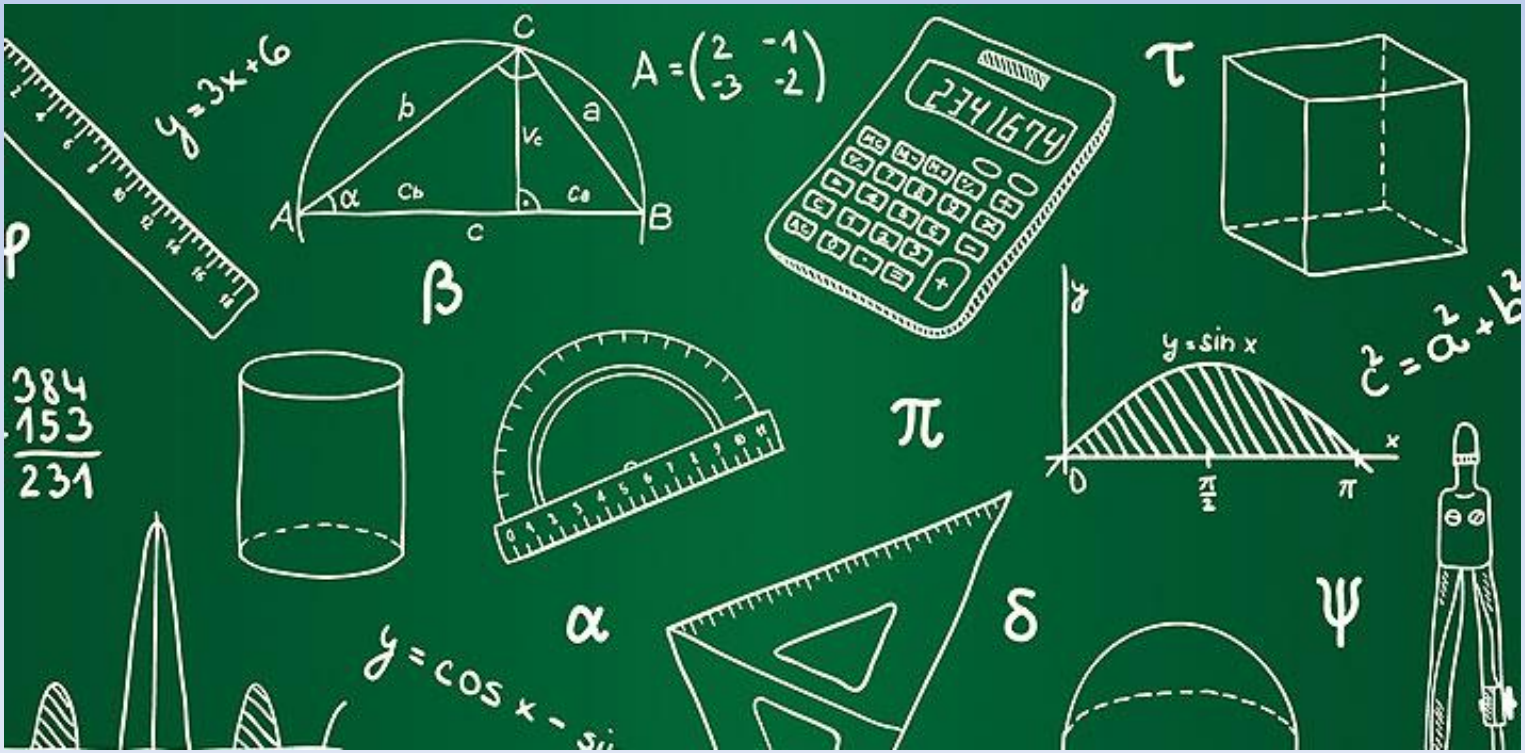
Math-Date-Array-Boolean-Number





Math

- El objeto Math predefinido posee propiedades y métodos asociados a las constantes y funciones matemáticas.



El objeto Math – Métodos y propiedades:

Métodos		
abs()	exp()	random()
acos()	floor()	round()
asin()	log()	sin()
atan()	max()	sqrt()
ceil()	min()	tan()
cos()	pow()	

Propiedades
E
LN2
LN10
LOG2E
LOG10E
PI
SQRT1_2
SQRT2

Math-Propiedades

- **E** Constante matemática que representa al número e. Su valor es 2.71828...
- **LN2** Constante matemática que representa al número resultado de calcular el logaritmo neperiano de 2. Su valor es 0.693...
- **LN10** Constante matemática que representa al número resultante de calcular el logaritmo neperiano de 10. Su valor es 2.303...
- **PI** Constante matemática que representa al número PI . Su valor es 3.1416...
- **SQRT1_2** Constante matemática que representa al número resultante de calcular la raiz cuadrada de 1/2. Su valor es 0.707...
- **SQRT2** Constante matemática que representa al número resultante de calcular la raiz cuadrada de 2. Su valor es 1.414213...

Math-Métodos

- **abs(número)** Devuelve el valor absoluto del número pasado por parámetros.
- **ceil(número)** Redondeo superior.
- **floor(número)** Devuelve la parte entera del número pasado por parámetros.
- **max(número1, número2)** Devuelve el máximo de dos valores.
- **min(número1, número2)** Devuelve le mínimo de dos valores.
- **random()** Devuelve el valor aleatorio entre cero y uno.

Math.floor(Math.random() * (MAX - MIN + 1)) + MIN;

- **round(número)** Redondeo inferior.

Math-métodos

- **acos(número)** Devuelve el arco coseno del número pasado por parámetros.
- **asin(número)** Devuelve el arco seno del número pasado por parámetros. **atan(número)** Devuelve la arco tangente del número pasado por parámetros.
- **cos(número)** Devuelve el coseno del número pasado por parámetros.
- **exp(número)** Devuelve *e* elevado al número pasado por parámetros.

Math-Métodos

- **log(número)** Devuelve el logaritmo neperiano del número pasado por parámetros
- **pow(base, exponente)** Devuelve el resultado de elevar la base al exponente.
- **sin(número)** Devuelve el seno del número pasado por parámetros.
- **sqrt(número)** Devuelve la raíz cuadrada del número pasado por parámetros. **tan(número)** Devuelve la tangente del número pasado por parámetros

Date

- El lenguaje JavaScript no posee variables tipo fecha, pero el objeto predefinido **Date** permiten la manipulación de datos que representen fechas.
- JavaScript gestiona las fechas como el lenguaje Java y utilizan la misma referencia:
- 1 de enero de 1970.

Date

- Cuenta con una serie de métodos divididos en tres subconjuntos:
    - Métodos de lectura.
    - Métodos de escritura.
    - Métodos de conversión.

Date

- El objeto Date – Métodos:

Métodos				
getDate()	getTime()	getUTCMonth()	setMonth()	setUTCMonth()
getDay()	getTimezoneOffset()	getUTCSeconds()	setSeconds()	setUTCSeconds()
getFullYear()	getUTCDate()	parse()	setTime()	toDateString()
getHours()	getUTCDay()	setDate()	setUTCDate()	toLocaleDateString()
getMilliseconds()	getUTCFullYear()	setFullYear()	setUTCFullYear()	toLocaleTimeString()
getMinutes()	getUTCHours()	setHours()	setUTCHours()	toLocaleString()
getMonth()	getUTCMilliseconds()	setMilliseconds()	setUTCMilliseconds()	toTimeString()
getSeconds()	getUTCMinutes()	setMinutes()	setUTCMinutes()	toUTCString()

Date

- Se utilizará la sintaxis siguiente para crear un objeto de tipo de **Date**:
- NombreVariable = new Date(parámetros)
- var d = new Date();
  var d = new Date(*milliseconds*);
  var d = new Date(*dateString*);
  var d = new Date(*year*, *month*, *day*, *hours*, *minutes*, *seconds*, *milliseconds*);

Date

El constructor *Date* admite los parámetros siguientes:

- **ninguno,** para crear un objeto que tenga la hora actual.
  hoy = new Date();

- **una cadena de caracteres** con el formato.
  "Mes día, año horas:minutos:segundos"
  Por ejemplo:
  fecha = new Date("January 6, 1996 14:50:00");
- **tres enteros** que representan el año, el mes y el día. Por ejemplo:

  fecha = new Date(79,2,12);

- **seis enteros** que representan el año, el mes, el día, la hora, los minutos y los segundos. Por ejemplo:

  fecha = new Date(79,2,12,6,45,0);

Date

Clasificación de métodos:

- Aquellos que nos permiten **acceder** a la fecha y hora
- **fijar** la fecha y hora
- **analizar** las cadenas de caracteres que representan las fechas
- **convertir** las fechas en una cadena de caracteres

Date-Métodos para acceder

- **getDate()** Devuelve el día del mes de 1 a 31.
- **getDay()** Devuelve el día de la semana de 0 (domingo) a 6 (sábado).
  getHours() Devuelve la hora de 0 a 23.
- **getMinutes()** Devuelve los minutos de 0 a 59.
- **getSeconds()** Devuelve los segundos de 0 a 59.
- **getMonth()** Devuelve el mes de un objeto Date. La salida es un valor entero entre 0 (Enero) y 11 (Diciembre).

Date-Métodos para acceder

- **getTime()** Devuelve el valor numérico correspondiente al objeto *Date* que lo llama, el valor de salida va referido al 1 de Enero de 1970 a las 00:00:00 y es el valor en milisegundos, con signo positivo si la fecha es posterior a la de referencia y con signo negativo si la fecha está antes de ésta.

- **getTimezoneOffset()** Devuelve la diferencia con la zona GMT en minutos.

- **parse(ristra)** Devuelve el número de milisegundos de una string que representa a una fecha con respecto a la ya conocida fecha de referencia del 1 de Enero de 1970 a las 00:00:00 horas. Para referenciar este método no hemos de crear ninguna instancia del objeto Date, bastará con realizar lo siguiente:
  var miliseg = Date.parse(string);

- **getFullYear() , devuelve el año**

Date-Métodos para establecer

- **setDate(número)** Asigna el día del mes, donde número será un valor entre 1 y 31.
- **setHours(número)** Asigna la hora, donde número será un valor entre 0 y 23.
- **setMinutes(número)** Asigna los minutos, donde número será un valor entre 0 y 59. **setSeconds(número)** Asigna los segundos, donde número será un valor entre 0 y 59.
- **setTime(número)** Fija en valor numérico correspondiente al objeto *Date* que lo llama, el valor de entrada va referido al 1 de enero de 1970 a las 00:00:00 y es el valor en milisegundos, con signo positivo si la fecha es posterior a la referencia y negativo en caso contrario.
- **setFullYear(número)** Asigna el año.

Date-Métodos para analizar y convertir

- **toGMTString()** Convierte una cadena de caracteres que representa un fecha al formato GMT.

- **toLocaleString()** Convierte una fecha al formato local.

- **toString()** Convierte convierte una fecha en una cadena de caracteres.

- **UTC(año, [mes], [día], [horas], [minutos], [segundos], [milisegundos])** Devuelve el número de milesegundos desde la fecha que se toma como referencia 1 de enero de 1970 a las 00:00:00, hasta la fecha del argumento de entrada. Decir que los campos horas, minutos y segundos son opcionales y si se omiten se considerarán cero. No es necesario crear una instancia del objeto de Date para usar este método, ya que es un método estático. Por tanto, el método se utilizará de la siguiente manera:

  var miliseg = Date.UTC(*parámetros*);

Boolean

- **El Objeto Boolean**
- Este objeto nos permite crear booleanos, esto es, un tipo de dato que es cierto o falso, tomando los valores true o false. Podemos crear objetos de este tipo mediante su constructor.
- 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'

Objeto boolean: métodos

Descripción

toString ()

Devuelve una representación de cadena de un Boolean.

valueOf ()

Obtiene una referencia al Boolean.

Number

El objeto Number:

- Permite realizar tareas relacionadas con tipos de datos numéricos.

Objetos nativos de JavaScript

- El objeto Number – Métodos y propiedades:

Métodos
toExponential()
toFixed()
toPrecision()

Propiedades
MAX_VALUE
MIN_VALUE
NaN
NEGATIVE_INFINITY
POSITIVE_INFINITY

Number

- Se crea mediante el constructor:

- a = new **Number**(valor);

Number

- Propiedades del Objeto Number
- **MAX_VALUE:** Valor máximo que se puede manejar con un tipo numérico
- **MIN_VALUE:** Valor mínimo que se puede manejar con un tipo numérico
- **NaN:** Representación de un dato que no es un número
- **NEGATIVE_INFINITY:** Representación del valor a partir del cual hay desbordamiento negativo (underflow)
- **POSITIVE_INFINITY** Representación del valor a partir del cual hay desbordamiento positivo (overflow)
- Para utilizar estas propiedades hay que trabajar directamente sobre el objeto number.

Objeto Number: Métodos

toFixed

```
var num = new Number(123);
 var fix = num.toFixed();
 document.write(fix);
document.write("<br/>");
num = new Number(123.456);
fix = num.toFixed(5); document.write(fix);
```

Objeto Number: Métodos

toLocaleString()

```
var n, s;
n = new Number(1000000);
s = "Current locale value is: ";
s += n.toLocaleString();
document.write(s);
```

Objeto Number: Métodos

toPrecision

```
var num = new Number(123);
 var prec = num.toPrecision();
   document.write(prec);
   document.write("<br/>");
num = new Number(123.456);
prec = num.toPrecision(5);
   document.write(prec);
```