



# 10

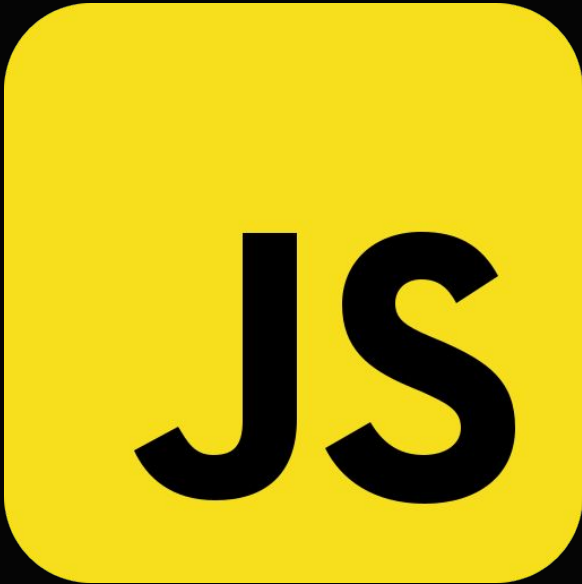
## Date, Math y conversiones

Yael Ruiz Pucheta

# Temas puntuales de la sección



- Clase / Prototipo Date
  - Constructor
  - Formatos
  - Formato Unix / epoch
  - Métodos
  - Operaciones
  - Luxon
- Clase / Prototipo Math
- Casting



JS

# Date

# Date


Es un objeto almacena la fecha, la hora, y brinda métodos para administraras.

El objeto Date contiene un Number que representa los milisegundos transcurridos desde el 1 de Enero de 1970 UTC.



```
const hoy = new Date()
```

# Date



```
new Date()
```

Sin argumentos – crea un objeto Date para la fecha y la hora actuales



```
new Date('2020-09-10');  
  
new Date(24 * 3600 * 1000)
```

Con argumentos - Crea un objeto Date con la fecha / milisegundos enviados

## Formatos de Fecha

Formato	Ejemplo
ISO Fecha	"2015-03-25" (La Norma Internacional)
Cita corta	"03/25/2015" o "2015/03/25"
Fecha larga	"Mar 25 2015" o "25 Mar 2015"
Fecha completa	"Wednesday March 25 2015"

## Formatos Unix / epoch

**Se define como la cantidad de segundos transcurridos desde la medianoche UTC del 1 de enero de 1970, sin contar segundos intercalares.**

**La época tradicionalmente corresponde a 0 horas, 0 minutos y 0 segundos (00:00:00) Tiempo Universal Coordinado (UTC) en una fecha específica, que varía de un sistema a otro.**

La mayoría de las versiones de Unix , por ejemplo, utilizan el **1 de enero de 1970** como fecha de época; Windows usa el 1 de enero de 1601; Los sistemas Macintosh usan el 1 de enero de 1904 y el Sistema de memoria virtual (VMS) de Digital Equipment Corporation usa el 17 de noviembre de 1858.



## Métodos

Método	¿Qué realiza?
<code>getFullYear()</code>	Nos devuelve el año con 4 dígitos.
<code>getMonth()</code>	Nos devuelve el número de mes del 0 (enero) al 11 (diciembre)
<code>getDate()</code>	Nos devuelve el día del mes, del 1 al 31
<code>getDay()</code>	Nos devuelve el día de la semana del 0 (domingo) al 6 (sábado)

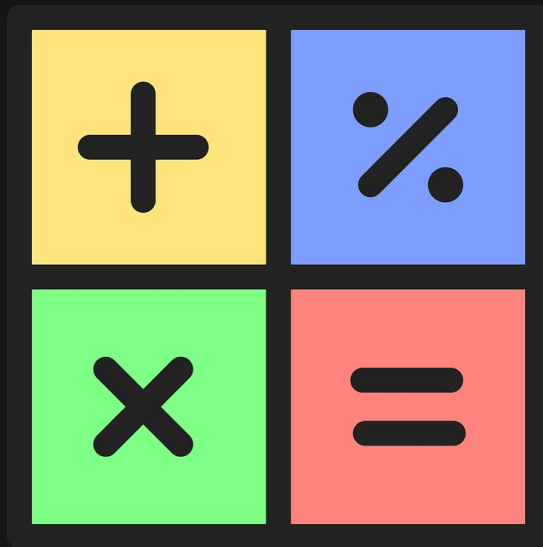


## Métodos

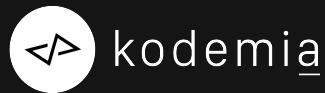
Método	¿Qué realiza?
getHours()	Nos devuelve la hora actual, desde 0 a 23 (formato 24 horas)
getMinutes()	Nos devuelve los minutos actuales, desde 0 a 59
toString()	Nos devuelve el día del mes, del 1 al 31

## Operaciones

Las operaciones que se pueden hacer con Date radican en el desarrollo de funciones / métodos creados por el propio desarrollador, con base en las necesidades que el necesite.



# Date



Sumar un día



```
const sumarDias = (fecha1, numeroDeDias) => {  
  const diasMilisegundos = numeroDeDias * 24 * 60 * 60000  
  const milisegundosFecha1 = fecha1.getTime();  
  return new Date(milisegundosFecha1 + diasMilisegundos)  
}
```

Restar un día



```
const restarDias = (fecha1, numeroDeDias) => {  
  const diasMilisegundos = numeroDeDias * 24 * 60 * 60000  
  const milisegundosFecha1 = fecha1.getTime();  
  return new Date(milisegundosFecha1 - diasMilisegundos)  
}
```

Podemos Alterar / Modificar la clase Original

```

Date.prototype.sumarDias = function(numeroDeDias){
    const diasMilisegundos = numeroDeDias * 24 * 60 * 60000
    const milisegundosFecha1 = new Date(this.valueOf()).getTime();
    return new Date(milisegundosFecha1+diasMilisegundos)
}
```

# Date

## LuxonJS

Luxon es una biblioteca / Librería para trabajar con fechas y horas en JavaScript.

Teniendo sus orígenes con MomentJS, es la nueva generación del antes mencionado utilizando el estándar más nuevo de Javascript.

Es importante conocerlo puesto que la propia documentación de Mozilla ha dedicado un post sobre esta biblioteca



# Math

# Math

Math es una de las clases nativas de Javascript. Proporciona los mecanismos para realizar operaciones matemáticas en Javascript.

La mayoría de operaciones pueden ser resueltas con operaciones aritméticas, sin embargo existen procesos que necesitan mucho más, es donde la clase Math toma relevancia.





## Métodos

Método	¿Qué realiza?
<code>abs()</code>	Devuelve el valor absoluto de un número. El valor después de quitarle el signo.
<code>ceil()</code>	Devuelve el entero igual o inmediatamente siguiente de un número. Por ejemplo, <code>ceil(3)</code> vale 3, <code>ceil(3.4)</code> es 4.
<code>pow()</code>	Recibe dos números como parámetros y devuelve el primer número elevado al segundo número.
<code>floor()</code>	Lo contrario de <code>ceil()</code> , pues devuelve un número igual o inmediatamente inferior.

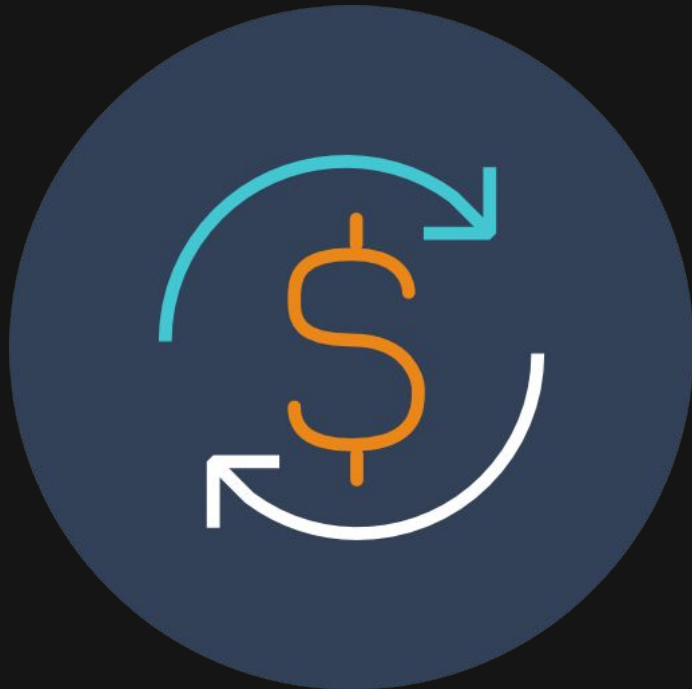
Método	¿Qué realiza?
max()	Retorna el mayor de N números.
min()	Retorna el menor de N números.
random()	Devuelve un número aleatorio entre 0 y 1.
round()	Redondea al entero más próximo.

# Math

# Casting

También conocido como **Conversión de Tipos** es un procedimiento para **transformar** una variable primitiva de un **tipo a otro**.

También se utiliza para transformar un objeto de una clase a otra clase siempre y cuando haya una relación de herencia entre ambas



# Casting



```
const numeroAString = String(10); // '10'  
const stringANumero = Number('15') // 15;  
const arregloAString = String([1,2,3]) // '1,2,3'  
const cadenaAArreglo = 'Hola,Uno,Dos'.split(',') // ['Hola','Uno','Dos']
```

# Casting

## Booleanos

Si bien el javascript puede convertir booleanos. Por otras validaciones pueden existir un “falso verdadero”



```
const transformarBoolean = Boolean('false'); // True
```

# Casting

Solo es este tipo de casos es recomendable hacer la validación del contenido



```
const tranformarBoolean = 'false' == true; // False
```

# Temas puntuales de la sección



- Clase / Prototipo Date
  - Constructor
  - Formatos
  - Formato Unix / epoch
  - Métodos
  - Operaciones
  - Luxon
- Clase / Prototipo Math
- Casting







# 10

## Date, Math y conversiones

Yael Ruiz Pucheta