


# 18. Métodos para el manejo de fechas

## Date - JavaScript | MDN

Los objetos Date representan en JavaScript un momento fijo en el tiempo en un formato independiente. El objeto Date contiene un Number que representa los milisegundos transcurridos desde el 1 de Enero de 1970 UTC.

 [https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global\\_Objects/Date](https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Date)

 mdn web docs

JavaScript proporciona el objeto `Date` para trabajar con fechas y horas. Aunque su funcionalidad es básica en comparación con bibliotecas como **moment.js** o **date-fns**, es esencial para operaciones comunes. A continuación, exploraremos los métodos más utilizados.

## 1. Creación de Fechas

### Constructor `Date`

```
// Fecha actual
const ahora = new Date(); // Ej: 2023-10-05T12:34:56.789Z

// Fecha a partir de una cadena
const fechaCadena = new Date("2023-10-05"); // AAAA-MM-DD

// Fecha a partir de valores (año, mes, día, horas, minutos, segundos, ms)
// ¡Los meses son 0-indexados! (0 = enero, 11 = diciembre)
const fechaValores = new Date(2023, 9, 5, 15, 30, 0); // 5 de octubre de 2023, 15:30:00

// Fecha a partir de milisegundos (desde 1/1/1970 UTC)
const fechaTimestamp = new Date(1696523400000);
```

## 2. Métodos para Obtener Componentes de la Fecha

Método	Descripción	Ejemplo
<code>getFullYear()</code>	Año (4 dígitos)	2023
<code>getMonth()</code>	Mes (0-11)	9 (octubre)
<code>getDate()</code>	Día del mes (1-31)	5
<code>getDay()</code>	Día de la semana (0-6, donde 0=domingo)	4 (jueves)
<code>getHours()</code>	Hora (0-23)	15
<code>getMinutes()</code>	Minutos (0-59)	30
<code>getSeconds()</code>	Segundos (0-59)	0
<code>getTime()</code>	Milisegundos desde 1/1/1970 (timestamp)	1696523400000

**Ejemplo:**

```
const fecha = new Date();
console.log(fecha.getFullYear()); // 2023
console.log(fecha.getMonth()); // 9 (octubre)
```

### 3. Métodos para Modificar Componentes de la Fecha

Método	Descripción
<code>setFullYear(año)</code>	Establece el año
<code>setMonth(mes)</code>	Establece el mes (0-11)
<code>setDate(día)</code>	Establece el día del mes
<code>setHours(horas)</code>	Establece la hora
<code>setMinutes(minutos)</code>	Establece los minutos
<code>setSeconds(segundos)</code>	Establece los segundos
<code>setTime(milisegundos)</code>	Establece la fecha desde un timestamp

#### Ejemplo:

```
const fecha = new Date();
fecha.setFullYear(2024); // Cambia el año a 2024
fecha.setMonth(0); // Cambia el mes a enero
```

### 4. Formateo de Fechas

#### Métodos Integrados

```
const fecha = new Date();

// Formato ISO 8601 (UTC)
console.log(fecha.toISOString()); // "2023-10-05T15:30:00.000Z"

// Fecha legible para humanos
console.log(fecha.toDateString()); // "Thu Oct 05 2023"

// Hora local
console.log(fecha.toLocaleTimeString()); // "3:30:00 PM"
```

#### Formateo Manual

```
const fecha = new Date();
const día = fecha.getDate().toString().padStart(2, "0");
const mes = (fecha.getMonth() + 1).toString().padStart(2, "0"); // Mes +1 (no es 0-index)
const año = fecha.getFullYear();
```

```
console.log(`${dia}/${mes}/${año}`); // "05/10/2023"
```

## 5. Cálculos con Fechas

### Diferencia entre Fechas

```
const fecha1 = new Date("2023-10-05");
const fecha2 = new Date("2023-10-10");

// Diferencia en milisegundos
const diferencia = fecha2.getTime() - fecha1.getTime(); // 432000000 ms (5 días)
```

### Añadir/Sustraer Días

```
const hoy = new Date();
const mañana = new Date(hoy);
mañana.setDate(hoy.getDate() + 1); // Añade 1 día
```

## 6. Validación de Fechas

```
function esFechaValida(fecha) {
  return fecha instanceof Date && !isNaN(fecha);
}

console.log(esFechaValida(new Date("2023-02-30"))); // false (febrero no tiene 30 días)
```

## 7. Métodos UTC

Para trabajar con fechas en formato UTC (Tiempo Universal Coordinado):

```
const fecha = new Date();

// Obtener componentes en UTC
console.log(fecha.getUTCFullYear()); // Año en UTC
console.log(fecha.getUTCMonth()); // Mes en UTC (0-11)
```

## Ejemplo Práctico: Contador de Días

```
function diasHasta(fechaObjetivo) {
  const ahora = new Date();
  const diferencia = fechaObjetivo.getTime() - ahora.getTime();
  return Math.ceil(diferencia / (1000 * 60 * 60 * 24));
}
```

```
}
```

```
const navidad = new Date("2023-12-25");  
console.log(`Faltan ${diasHasta(navidad)} días para Navidad`);
```

## Limitaciones y Recomendaciones

1. **Zonas Horarias:** El objeto `Date` usa la zona horaria del navegador. Para operaciones complejas, usa bibliotecas como **Luxon** o **date-fns**.
2. **Formato de Cadenas:** Evita analizar fechas con cadenas (ej: `new Date("2023-10-05")`), ya que el formato puede variar entre navegadores.
3. **Mutabilidad:** Los métodos `setX()` modifican la fecha original. Si necesitas inmutabilidad, crea una copia:

```
const nuevaFecha = new Date(fechaOriginal.getTime());
```