

Day 6: JavaScript Dates

 by [AvmnuSng](#)

[Table of Contents](#)[Date](#)[Creating Date](#)[Date getters](#)

Dates in JavaScript

Date

A JavaScript Date instance represents a single moment in time based on the number of milliseconds elapsed since **1 January, 1970 UTC**.

Creating Date Instance

There are four constructors we can use to create a *Date* object, defined below.

1. Using `new Date()`

The *default constructor* creates a JavaScript *Date* object for the current date and time (according to your system settings).

2. Using `new Date(value)`

This constructor has a parameter, *value*, which is an integer representing the number of milliseconds elapsed since **1 January 1970 00:00:00 UTC** (this is a [Unix Epoch](#), though you should keep in mind that most Unix timestamp functions count in seconds).

3. Using `new Date(dateString)`

This constructor has a parameter, *dateString*, which is a String describing a date. The *dateString* must be in a format recognized by the `Date.parse()` function, such as `MM/DD/YYYY` or `Month Day, Year`. For example, `01/01/1980` and `Jan 1, 1980` are both strings that can be successfully parsed using the *parse* function.

4. Using `new Date(year, month, day, hour, minutes, seconds, milliseconds)`

This constructor has the following parameters:

- **year**: An integer denoting the calendar year. Values from **0** through **99** map to the years **1900** through **1999**.
- **month**: A one or two digit integer denoting the zero-indexed month. This means that **0** denotes January and **11** denotes December.
- **day**: Optional. An integer denoting the specific day number within the calendar month.
- **hour**: Optional. An integer denoting the hour of the day.
- **minute**: Optional. An integer denoting the minute segment of a time.
- **second**: Optional. An integer denoting the second segment of a time.
- **millisecond**: Optional. An integer denoting the millisecond segment of a time.

EXAMPLE

Input Format

The first line contains *date1*.
The second line contains *date2*.

[Table O](#)[Date](#)[Creating](#)[Date ge](#)

```
1 'use strict';
2 process.stdin.on('data', function (data) {
3     main(String(data).split("\n"));
4 });
5
6 function main(input) {
7     var date1 = new Date(input[0]);
8     console.log(date1.toString());
9     console.log(date1.toISOString());
10
11    var date2 = new Date(input[1]);
12    console.log(date2.toString());
13    console.log(date2.toISOString());
14 }
```

Input

Run

Output

The code above produces the following output:

```
Sun Oct 11 2009 00:00:00 GMT+0000 (UTC)
2009-10-11T00:00:00.000Z
Fri Apr 22 2016 00:00:00 GMT+0000 (UTC)
2016-04-22T00:00:00.000Z
```

Date get Methods

1. Date.getTime()

Get the time in milliseconds elapsed since **January 1, 1970**.

2. Date.getFullYear()

Get the four-digit year (**yyyy**).

3. Date.getMonth()

Get the *Date* object's month as a zero-indexed number (**0 — 11**).

4. Date.getDate()

Get the *Date* object's day as a number (**1 — 31**).

5. Date.getDay()

Get the *Date* object's weekday as a number (**0 — 6**).

6. Date.getHours()

Get the *Date* object's hour (**0 — 23**).

7. Date.getMinutes()

Get the *Date* object's minutes (**0 — 59**).

8. Date.getSeconds()

Get the *Date* object's seconds (**0 — 59**).

9. Date.getMilliseconds()

Get the *Date* object's milliseconds (**0 — 999**).

[- EXAMPLE](#)

Click *Run* below to see this in code.

Input Format

A single date string in a format recognized by `Date.parse()`.

```
1 'use strict';
2 process.stdin.on('data', function (data) {
3     main(String(data));
4 });
5 /**** Ignore above this line. ****/
6
7 function main(input) {
8     let date = new Date(input);
9
10    console.log("date: " + date);
11    console.log("date.getDate(): " + date.getDate());
12    console.log("date.getDay(): " + date.getDay());
13    console.log("date.getFullYear(): " + date.getFullYear());
14    console.log("date.getHours(): " + date.getHours());
15    console.log("date.getMilliseconds(): " + date.getMilliseconds());
16    console.log("date.getMinutes(): " + date.getMinutes());
17    console.log("date.getMonth(): " + date.getMonth());
18    console.log("date.getSeconds(): " + date.getSeconds());
19    console.log("date.getTime(): " + date.getTime());
20    console.log("date.getYear(): " + date.getYear());
21    console.log("date.toString(): " + date.toString());
22 }
```

Input

Feb 3, 1987 12:34:56:789

Run

Output

The code above produces the following output:

```
date: Tue Feb 03 1987 12:34:56 GMT+0000 (UTC)
date.getDate(): 3
date.getDay(): 2
date.getFullYear(): 1987
date.getHours(): 12
date.getMilliseconds(): 789
date.getMinutes(): 34
date.getMonth(): 1
date.getSeconds(): 56
date.getTime(): 539354096789
date.getYear(): 87
date.toString(): Tue Feb 03 1987
```

You could also create a date object for the date given as input using the following date constructor and arguments:

```
// Date(year, month, day, hour, minutes, seconds, milliseconds)
let date = new Date(1987, 1, 3, 12, 34, 56, 789);
```

Table O

Date

Creating

Date ge

