

[Project Home](#)[Downloads](#)[Wiki](#)[Issues](#)[Source](#)[Summary](#) [People](#)

Last updated 2008-04-14



An **open-source**  
JavaScript Date Library

## ⚙ Overview

[Datejs](#) is an open source JavaScript Date library for parsing, formatting and processing.

The last 'official' release was Alpha-1 on November 19th, 2007. The project has been under active development since that time and many revisions and additions have occurred. It is highly recommended that you obtain a copy of the latest source from [SVN](#).

Technical support is provided through the Datejs [Discussion Group](#).

A test suite is available within [SVN](#) (/trunk/test/) or viewable online at <http://www.datejs.com/test/>.

View [Change Log](#).

## ⚙ Getting Started

NOTE Please be sure to check out our blog post "[Getting Started with Datejs](#)".

We recommend including one of the .js files from within the /build/ folder.

```
<script type="text/javascript" src="date.js"></script>
```

Within the /build/ folder, a date.js file has been created for each of the 150+ supported Cultures. Changing the Culture of the library is as easy as changing the date.js file.

```
<!-- Set the CultureInfo to de-DE (German/Deutsch) -->
<script type="text/javascript" src="date-de-DE.js"></script>
```

Before minification, the Datejs library is contained in five (6) separate JavaScript files. Each of the files can be included individually.

The following is a list of precedence if including the files individually. Each file requires the one above it. For example, core.js requires a CultureInfo file.

- |                 |   |
|-----------------|---|
| 1. CultureInfo  | Contains all Globalized strings and culture specific properties. Debug versions available within the /trunk/source/globalization/ folder. |
| 2. core.js      | All core Date processing logic.   |
| 3. parser.js    | All parsing logic.  |
| 4. sugarpak.js* | All syntactical sugar.  |
| 5. time.js**    | TimeSpan and TimePeriod classes.  |
| 6. extras.js**  | PHP/Unix date format conversion functions.  |
- \* The parser.js file is not required for sugarpak.js
- \*\* The time.js and extras.js files are optional and are not included in the compiled /build/ versions.

## ⚙ Example Usage

### Syntax Overview

```

Date.today() // Returns today's date, with time set to 00:00 (start
of day).

Date.today().next().friday() // Returns the date of the next Friday.
Date.today().last().monday() // Returns the date of the previous Monday.

new Date().next().march() // Returns the date of the next March.
new Date().last().week() // Returns the date one week ago.

Date.today().is().friday() // Returns true|false if the day-of-week matches.
Date.today().is().fri() // Abbreviated day names.

Date.today().is().november() // Month names.
Date.today().is().nov() // Abbreviated month names.

Date.today().is().weekday() // Is today a weekday?

Date.today().addDays(1) // Add one day (+1).
Date.today().addMonths(-3) // Subtract three months (-3).

Date.today().add(1).day() // Add one (+1) day. Supports all date parts (year,
month, day, hour, minute, second, millisecond, and weeks)
Date.today().add(-3).months() // Subtract three (-3) months.

(1).day().fromNow() // One (1) day from now.
(3).months().ago() // Three (3) months ago.

var n = 6;
n.months().fromNow() // Six (6) months from now.

Date.monday() // Returns Monday of the current week.
Date.mon() // Abbreviated version of Date.monday()

Date.march() // Returns March 1st of this year.
Date.mar() // Abbreviated version of Date.march()

Date.today().first().thursday() // Returns the first Thursday of the current month.
Date.today().second().thursday() // Returns the second Thursday of the current month.

Date.march().third().thursday() // Returns the third Thursday in March of the
current year.
Date.october().fourth().sunday() // Returns the fourth Sunday in October.

Date.today().fifth().sunday() // Returns the fifth Sunday in the current month,
or throws a RangeError exception if there are not 5 Sundays in the current month.
Date.october().final().sunday() // Returns the final Sunday in October.

Date.january().first().monday() // Returns the first Monday of the current year.

```

```
Date.december().final().friday() // Returns the last Friday of the current year.
```

```
Date.today().at("6:15pm"); // Returns today's date at 6:15pm.
```

```
var time = {hour: 18, minute: 15};
Date.today().at(time); // Set time with a config object.
```

```
var birthdayParty = {month: 1, day: 20, hour: 20, minute: 30};
Date.today().set(birthdayParty); // Set date and time with a config object.
```

## Parsing

The following list is only a small subset of hundreds of string formats which can be parsed correctly without providing a date format. All parsing is fully Globalized by including the appropriate [CultureInfo](#) file.

The [CultureInfo](#) file contains all the strings used for parsing and formatting.

All [CultureInfo](#) files can be found in the `/trunk/source/globalization/` folder.

The following `.parse()` samples use the en-US.js [CultureInfo](#) file.

```
Date.parse('t') // Returns today's date.
Date.parse('today') // Returns today's date.
Date.parse('tomorrow') // Returns tomorrow's date.
Date.parse('yesterday') // Returns yesterday's date.
```

```
Date.parse('next friday') // Returns the date of the next Friday.
Date.parse('last monday') // Returns the date of the previous Monday.
```

```
Date.parse('July 8th, 2004') // Thu Jul 08 2004
Date.parse('15-Jan-2004') // Thu Jan 15 2004
```

```
Date.parse('7/1/2004') // Thu Jul 01 2004
Date.parse('7.1.2004') // Thu Jul 01 2004
Date.parse('07.15.04') // Thu Jul 15 2004
```

```
Date.parse('July 23rd 2004') // Fri Jul 23 2004
Date.parse('Sat July 3, 2004') // Sat Jul 03 2004
```

```
Date.parse('10:30 PM EST') // Wed Oct 31 2007 20:30:00
Date.parse('10PM') // Wed Oct 31 2007 22:00:00
```

```
Date.parse('t + 5d') // Adds 5 days to today.
Date.parse('today - 1 month') // Subtracts 1 month from today.
```

## Project Information

- [Project feeds](#)
- 
- Code license
- [MIT License](#)
- 
- Labels  
javascript, date, library,  
parse, parser, duration,  
datetime, time, timespan,  
prototype

## Members

[ge...@ext.net](#)  
[, d...@zeraweb.com](#)

•

## Featured

## Downloads

- [Datejs-all-Alpha1.zip](#)
- [date.js](#)
- [Show all »](#)
- 

## Wiki pages

- [APIDocumentation](#)
- [FormatSpecifiers](#)
- [Show all »](#)
- 

## Links

- Blogs
- [Datejs Blog](#)
- 
- External links
- [Datejs Website](#)
- [Unit Tests](#)
- [API Documentation](#)
- [Change Log](#)
- 
- Groups
- [Datejs Forums](#)
- [SVN Commits](#)
- 



```
Date.parse('+')           // Add 1 day to today = tomorrow.
Date.parse('- 3months')   // Subtract 3 months.

Date.parse('+1year')      // Add a year to today.
Date.parse('- 12 months') // Subtract 12 months (1 year) from today.

Date.parse(' July 4th')   // July 4th of this year.
Date.parse(' 15')         // 15th day of current month/year.
```



```
Date.parse(' July 8th, 2004, 10: 30 PM') // Thu Jul 08 2004 22: 30: 00
Date.parse(' 2004-07- 15T06: 45: 00')    // Thu Jul 15 2004 06: 45: 00
Date.parse(' Thu, 1 July 2004 22: 30: 00 GMT') // Thu Jul 01 2004 16: 30: 00

Date.parse(' 1997-07- 16T19: 20: 15')    // ISO 8601 Formats
Date.parse(' 1997-07- 16T19: 20: 30+01: 00') // ISO 8601 with Timezone offset
Date.parse(' 1985-04- 12T23: 20: 50Z')    // RFC 3339 Formats
```



## Chaining

```
Date.today().add({ months: 1, days: 5 }).is().fri() // Add 1 month and 5 days,
then check if that date is a Friday.
Date.parse(' 10- July-2004').next().friday().add(-1).month() // Take in a date, then move
to the next Friday and subtract a month.
```

## Comparison

```
Date.today().equals( Date.parse(' today')) // true|false
Date.parse(' last Tues').equals(Date.today()) // true|false

Date.equals(Date.today(), Date.parse(' today')) // true|false
Date.compare(Date.today(), Date.parse(' today')) // 1 = greater, -1 = less
than,

Date.today().compareTo(Date.parse(' yesterday')) // 1 = greater, -1 = less
than, 0 = equal
Date.today().between(startDate, endDate) // true|false
```

## Converting to String

Note The format parameter is optional with the . toString() function. If no format is provided, the native JavaScript Date . toString() function will be called.

A detailed list of supported [FormatSpecifiers](#) is listed in the [Wiki](#) documentation.

## Standard Date and Time Format Specifiers

Format	Description	Example
s	The seconds of the minute between 0-59.	"0" to "59"
ss	The seconds of the minute with leading zero if required.	"00" to "59"
m	The minute of the hour between 0-59.	"0" or "59"
mm	The minute of the hour with leading zero if required.	"00" or "59"
h	The hour of the day between 1-12.	"1" to "12"
hh	The hour of the day with leading zero if required.	"01" to "12"
H	The hour of the day between 0-23.	"0" to "23"
HH	The hour of the day with leading zero if required.	"00" to "23"
d	The day of the month between 1 and 31.	"1" to "31"
dd	The day of the month with leading zero if required.	"01" to "31"
ddd	Abbreviated day name. <code>Date.CultureInfo.abbreviatedDayNames</code> .	"Mon" to "Sun"
dddd	The full day name. <code>Date.CultureInfo.dayNames</code> .	"Monday" to "Sunday"
M	The month of the year between 1-12.	"1" to "12"
MM	The month of the year with leading zero if required.	"01" to "12"
MMM	Abbreviated month name. <code>Date.CultureInfo.abbreviatedMonthNames</code> .	"Jan" to "Dec"
MMMM	The full month name. <code>Date.CultureInfo.monthNames</code> .	"January" to "December"
yy	Displays the year as a two-digit number.	"99" or "07"
yyyy	Displays the full four digit year.	"1999" or "2007"
t	Displays the first character of the A.M./P.M. designator. <code>Date.CultureInfo.amDesignator</code> or <code>Date.CultureInfo.pmDesignator</code>	"A" or "P"
tt	Displays the A.M./P.M. designator. <code>Date.CultureInfo.amDesignator</code> or <code>Date.CultureInfo.pmDesignator</code>	"AM" or "PM"
S	The ordinal suffix ("st", "nd", "rd" or "th") of the current day.	"st", "nd", "rd" or "th"

## Custom Date and Time Format Specifiers

Format	Description	Example
d	The <a href="#">CultureInfo</a> shortDate Format Pattern	"M/d/yyyy"
D	The <code>CultureInfo</code> longDate Format Pattern	"dddd, MMMM dd, yyyy"
F	The <code>CultureInfo</code> fullDateTime Format Pattern	"dddd, MMMM dd, yyyy h:mm:ss tt"

m	The CultureInfo monthDay Format Pattern	"MMMM dd"
r	The CultureInfo rfc1123 Format Pattern	"ddd, dd MMM yyyy HH:mm:ss GMT"
s	The CultureInfo sortableDateTime Format Pattern	"yyyy-MM-ddTHH:mm:ss"
t	The CultureInfo shortTime Format Pattern	"h:mm tt"
T	The CultureInfo longTime Format Pattern	"h:mm:ss tt"
u	The CultureInfo universalSortableDateTime Format Pattern	"yyyy-MM-dd HH:mm:ssZ"
y	The CultureInfo yearMonth Format Pattern	"MMMM, yyyy"

Separator Characters

Character	Name
/	forward slash
	space
.	period dot
-	hyphen dash
,	comma

```
new Date().toString() // "Wed Oct 31 2007 16:18:10 GMT-0700 (Pacific Daylight Time)"
new Date().toString('M/d/yyyy') // "10/31/2007"

Date.today().toString('d-MMM-yyyy') // "31-Oct-2007"
new Date().toString('HH:mm') // "16:18"

Date.today().toString('MMMM dS, yyyy') // "April 12th, 2008"

Date.today().toShortDateString() // "10/31/2007". Culture specific as per
Date.CultureInfo.shortDatePattern.
Date.today().toLongDateString() // "Wednesday, October 31, 2007". Culture
specific as per Date.CultureInfo.longDatePattern.

new Date().toShortTimeString() // "4:18 PM". Culture specific as per
Date.CultureInfo.shortTimePattern.
new Date().toLongTimeString() // "4:18:34 PM". Culture specific as per
Date.CultureInfo.longTimePattern.
```

Core

```
Date.today().set({ day: 15 }) // Sets the day to the 15th of the current
```

month and year. Other object values include year|month|day|hour|minute|second.

```
Date.today().set({ year: 2007, month: 1, day: 20 })
```

Date.today().add({ days: 2 }) // Adds 2 days to the Date. Other object values include year|month|day|hour|minute|second.

```
Date.today().add({ years: -1, months: 6, hours: 3 })
```

```
Date.today().addYears(1) // Add 1 year.
```

```
Date.today().addMonths(-2) // Subtract 2 months.
```

```
Date.today().addWeeks(1) // Add 1 week
```

```
Date.today().addHours(6) // Add 6 hours.
```

```
Date.today().addMinutes(-30) // Subtract 30 minutes
```

```
Date.today().addSeconds(15) // Add 15 seconds.
```

```
Date.today().addMilliseconds(200) // Add 200 milliseconds.
```

```
Date.today().moveToFirstDayOfMonth() // Returns the first day of the current month.
```

```
Date.today().moveToLastDayOfMonth() // Returns the last day of the current month.
```

```
new Date().clearTime() // Sets the time to 00:00 (start of the day).
```

```
Date.today().setTimeToNow() // Resets the time to the current time (now).
```

The functional opposite of .clearTime()

## ISO 8601

```
// Parse ISO 8601 string
```

```
Date.parse('\\"1997-07-16T19:20:15\\"') // ISO 8601 string format with wrapping double-quotes
```

```
// Convert date to ISO 8601 string
```

```
new Date().toISOString() // Returns ISO 8601 string of date converted to it's UTC value. "2007-10-31T16:18:00Z"
```

```
// Get UTC converted ISO week number
```

```
Date.today().getISOWeek() // Returns ISO 8601 week of year. Returns "01" to ("52" | "53") depending on the year. See also .getWeek()
```

## Misc

```
Date.getMonthNumberFromName('March') // 2 - CultureInfo specific. <static>
```

```
Date.getDayNumberFromName('sat') // 6 - CultureInfo specific. <static>
```

```
Date.isLeapYear(2008) // true|false. <static>
```



```
Date.getDaysInMonth(2007, 9)           // 31 <static>

Date.today().getWeek()                  // Returns week of year. Returns 1 to (52 |
53) depending on the year
Date.today().setWeek(1)                  // Sets the week of the year to the Monday of
the week set.

var test = new Date();                  // Do something... like run a test...
test.getEl apsed()                       // Returns millisecond difference from now.

Date.today().isDayl ightSavi ngTi me()   // true|false. Is within the Daylight Saving Time.
Date.today().hasDayl ightSavi ngTi me()  // true|false. Is Daylight Saving Time observed.
```

Some icons provided by fantabulous [FamFamFam](#).

[Terms](#) - [Privacy](#) - [Project Hosting Help](#)

Powered by [Google Project Hosting](#)