Date

Lets you work with dates and times.

Core object	
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.1: added prototype property. JavaScript 1.3: removed platform dependencies to provide a uniform behavior across platforms; added ms_num parameter to Date constructor; added getFullYear, setFullYear, getMilliseconds, setMilliseconds, toSource, and UTC methods (such as getUTCDate and setUTCDate).
ECMA version	ECMA-262

Created by

The Date constructor:

new Date()
new Date(*milliseconds*)
new Date(*dateString*)
new Date(*yr_num*, *mo_num*, *day_num*
[, *hr_num*, *min_num*, *sec_num*, *ms_num*])

Versions prior to JavaScript 1.3:

new Date()
new Date(*milliseconds*)
new Date(*dateString*)
new Date(*yr_num*, *mo_num*, *day_num*[, *hr_num*, *min_num*, *sec_num*])

Parameters

milliseconds	Integer value representing the number of milliseconds since 1 January 1970 00:00:00.
dateString	String value representing a date. The string should be in a format recognized by the Date.parse method.
yr_num, mo_num, day_num	Integer values representing part of a date. As an integer value, the month is represented by 0 to 11 with 0=January and 11=December.
hr_num, min_num, sec_num, ms_num	Integer values representing part of a date.

Description
If you supply no arguments, the constructor creates a Date object for today's date and time according to local time. If you supply some arguments but not others, the missing arguments are set to 0. If you supply any arguments, you must supply at least the year, month, and day. You can omit the hours, minutes, seconds, and milliseconds.

The date is measured in milliseconds since midnight 01 January, 1970 UTC. A day holds 86,400,000 milliseconds. The Date object range is -100,000,000 days to 100,000,000 days relative to 01 January, 1970 UTC.

The Date object provides uniform behavior across platforms.

The Date object supports a number of UTC (universal) methods, as well as local time methods. UTC, also known as Greenwich Mean Time (GMT), refers to the time as set by the World Time Standard. The local time is the time known to the computer where JavaScript is executed.

For compatibility with millennium calculations (in other words, to take into account the year 2000), you should always specify the year in full; for example, use 1998, not 98. To assist you in specifying the complete year, JavaScript includes the methods getFullYear, setFullYear, getFullUTCYear, and setFullUTCYear.

The following example returns the time elapsed between timeA and timeB in milliseconds.

```
timeA = new Date();
// Statements here to take some action.
```

```
timeB = new Date();
timeDifference = timeB - timeA;
```

Backward Compatibility

JavaScript 1.2 and earlier. The Date object behaves as follows:

-
- Dates prior to 1970 are not allowed.
- JavaScript depends on platform-specific date facilities and behavior; the behavior of the Date object varies from platform to platform.

Property Summary

Property	Description
constructor	Specifies the function that creates an object's prototype.
prototype	Allows the addition of properties to a Date object.

Method Summary

Method	Description
getDate	Returns the day of the month for the specified date according to local time.
getDay	Returns the day of the week for the specified date according to local time.
getFullYear	Returns the year of the specified date according to local time.
getHours	

	Returns the hour in the specified date according to local time.
getMilliseconds	Returns the milliseconds in the specified date according to local time.
getMinutes	Returns the minutes in the specified date according to local time.
getMonth	Returns the month in the specified date according to local time.
getSeconds	Returns the seconds in the specified date according to local time.
getTime	Returns the numeric value corresponding to the time for the specified date according to local time.
getTimezoneOffset	Returns the time-zone offset in minutes for the current locale.
getUTCDate	Returns the day (date) of the month in the specified date according to universal time.
getUTCDay	Returns the day of the week in the specified date according to universal time.
getUTCFullYear	Returns the year in the specified date according to universal time.
getUTCHours	Returns the hours in the specified date according to universal time.
getUTCMilliseconds	Returns the milliseconds in the specified date according to universal time.

getUTCMinutes	Returns the minutes in the specified date according to universal time.
getUTCMonth	Returns the month according in the specified date according to universal time.
getUTCSeconds	Returns the seconds in the specified date according to universal time.
getYear	Returns the year in the specified date according to local time.
parse	Returns the number of milliseconds in a date string since January 1, 1970, 00:00:00, local time.
setDate	Sets the day of the month for a specified date according to local time.
setFullYear	Sets the full year for a specified date according to local time.
setHours	Sets the hours for a specified date according to local time.
setMilliseconds	Sets the milliseconds for a specified date according to local time.
setMinutes	Sets the minutes for a specified date according to local time.
setMonth	Sets the month for a specified date according to local time.
setSeconds	Sets the seconds for a specified date according to local time.

setTime	Sets the value of a Date object according to local time.
setUTCDate	Sets the day of the month for a specified date according to universal time.
setUTCFullYear	Sets the full year for a specified date according to universal time.
setUTCHours	Sets the hour for a specified date according to universal time.
setUTCMilliseconds	Sets the milliseconds for a specified date according to universal time.
setUTCMinutes	Sets the minutes for a specified date according to universal time.
setUTCMonth	Sets the month for a specified date according to universal time.
setUTCSeconds	Sets the seconds for a specified date according to universal time.
setYear	Sets the year for a specified date according to local time.
toGMTString	Converts a date to a string, using the Internet GMT conventions.
toLocaleString	Converts a date to a string, using the current locale's conventions.
toLocaleDateString	Returns the "date" portion of the Date as a string, using the current locale's conventions.
toLocaleTimeString	Returns the "time" portion of the Date as a string, using

	the current locale's conventions.
toSource	Returns an object literal representing the specified Date object; you can use this value to create a new object. Overrides the Object.toSource method.
toString	Returns a string representing the specified Date object. Overrides the Object.toString method.
toUTCString	Converts a date to a string, using the universal time convention.
UTC	Returns the number of milliseconds in a Date object since January 1, 1970, 00:00:00, universal time.
valueOf	Returns the primitive value of a Date object. Overrides the Object.valueOf method.

In addition, this object inherits the watch and unwatch methods from Object.

Examples
The following examples show several ways to assign dates:

```
today = new Date()
birthday = new Date("December 17, 1995 03:24:00")
birthday = new Date(95,11,17)
birthday = new Date(95,11,17,3,24,0)
```

constructor

Specifies the function that creates an object's prototype. Note that the value of this property is a reference to the function itself, not a string containing the function's name.

Property of	Date
Implemented in	JavaScript 1.1, NES 2.0

ECMA version	ECMA-262

Description
See Object.constructor.

getDate

Returns the day of the month for the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getDate()

Parameters
None

Description
The value returned by getDate is an integer between 1 and 31.

Examples
The second statement below assigns the value 25 to the variable day, based on the value of the Date object Xmas95.

```
Xmas95 = new Date("December 25, 1995 23:15:00")
day = Xmas95.getDate()
```

See also
Date.getUTCDate, Date.getUTCDay, Date.setDate

getDay

Returns the day of the week for the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getDay()

Parameters
None

Description
The value returned by getDay is an integer corresponding to the day of the week: 0 for Sunday, 1 for Monday, 2 for Tuesday, and so on.

Examples
The second statement below assigns the value 1 to weekday, based on the value of the Date object Xmas95. December 25, 1995, is a Monday.

```
Xmas95 = new Date("December 25, 1995 23:15:00")
weekday = Xmas95.getDay()
```

See also
Date.getUTCDay, Date.setDate

getFullYear

Returns the year of the specified date according to local time.

Method of	Date

Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getFullYear()

Parameters
None

Description
The value returned by getFullYear is an absolute number. For dates between the years 1000 and 9999, getFullYear returns a four-digit number, for example, 1995. Use this function to make sure a year is compliant with years after 2000.

Use this method instead of the getYear method.

Examples
The following example assigns the four-digit value of the current year to the variable yr.

```
var yr;
Today = new Date();
yr = Today.getFullYear();
```

See also
Date.getYear, Date.getUTCFullYear , Date.setFullYear

getHours

Returns the hour for the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getHours()

Parameters
None

Description
The value returned by getHours is an integer between 0 and 23.

Examples
The second statement below assigns the value 23 to the variable hours, based on the value of the Date object Xmas95.

```
Xmas95 = new Date("December 25, 1995 23:15:00")
hours = Xmas95.getHours()
```

See also
Date.getUTCHours, Date.setHours

getMilliseconds

Returns the milliseconds in the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getMilliseconds()

Parameters
None

Description
The value returned by getMilliseconds is a number between 0 and 999.

Examples
The following example assigns the milliseconds portion of the current time to the variable ms.

```
var ms;
Today = new Date();
ms = Today.getMilliseconds();
```

See also
Date.getUTCMilliseconds , Date.setMilliseconds

getMinutes

Returns the minutes in the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getMinutes()

Parameters
None

Description
The value returned by getMinutes is an integer between 0 and 59.

Examples
The second statement below assigns the value 15 to the variable minutes, based on the value of the Date object Xmas95.

```
Xmas95 = new Date("December 25, 1995 23:15:00")
minutes = Xmas95.getMinutes()
```

See also

Date.getUTCMinutes, Date.setMinutes

getMonth

Returns the month in the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getMonth()

Parameters
None

Description
The value returned by getMonth is an integer between 0 and 11. 0 corresponds to January, 1 to February, and so on.

Examples
The second statement below assigns the value 11 to the variable month, based on the value of the Date object Xmas95.

```
Xmas95 = new Date("December 25, 1995 23:15:00")
month = Xmas95.getMonth()
```

See also
Date.getUTCMonth, Date.setMonth

getSeconds

Returns the seconds in the current time according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getSeconds()

Parameters
None

Description
The value returned by getSeconds is an integer between 0 and 59.

Examples
The second statement below assigns the value 30 to the variable secs, based on the value of the Date object Xmas95.

```
Xmas95 = new Date("December 25, 1995 23:15:30")
secs = Xmas95.getSeconds()
```

See also
Date.getUTCSeconds, Date.setSeconds

getTime

Returns the numeric value corresponding to the time for the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getTime()

Parameters
None

Description
The value returned by the getTime method is the number of milliseconds since 1 January 1970 00:00:00. You can use this method to help assign a date and time to another Date object.

Examples
The following example assigns the date value of theBigDay to sameAsBigDay:

```
theBigDay = new Date("July 1, 1999")
sameAsBigDay = new Date()
sameAsBigDay.setTime(theBigDay.getTime())
```

See also
Date.getUTCHours, Date.setTime

getTimezoneOffset

Returns the time-zone offset in minutes for the current locale.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
getTimezoneOffset()

Parameters
None

Description
The time-zone offset is the difference between local time and Greenwich Mean Time (GMT). Daylight savings time prevents this value from being a constant.

Examples
x = new Date()
currentTimeZoneOffsetInHours = x.getTimezoneOffset()/60

getUTCDate

Returns the day (date) of the month in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCDate()

Parameters
None

Description
The value returned by getUTCDate is an integer between 1 and 31.

Examples
The following example assigns the day portion of the current date to the variable d.

```
var d;
Today = new Date();
d = Today.getUTCDate();
```

See also
Date.getDate , Date.getUTCDay , Date.setUTCDate

getUTCDay

Returns the day of the week in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCDay()

Parameters
None

Description
The value returned by getUTCDay is an integer corresponding to the day of the week: 0 for Sunday, 1 for Monday, 2 for Tuesday, and so on.

Examples
The following example assigns the weekday portion of the current date to the variable ms.

```
var weekday;
Today = new Date()
weekday = Today.getUTCDay()
```

See also
Date.getDay , Date.getUTCDate , Date.setUTCDate

getUTCFullYear

Returns the year in the specified date according to universal time.

Method of	Date

Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCFullYear()

Parameters
None

Description
The value returned by getUTCFullYear is an absolute number that is compliant with year-2000, for example, 1995.

Examples
The following example assigns the four-digit value of the current year to the variable yr.

```
var yr;
Today = new Date();
yr = Today.getUTCFullYear();
```

See also
Date.getFullYear , Date.setFullYear

getUTCHours

Returns the hours in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCHours()

Parameters
None

Description
The value returned by getUTCHours is an integer between 0 and 23.

Examples
The following example assigns the hours portion of the current time to the variable hrs.

```
var hrs;
Today = new Date();
hrs = Today.getUTCHours();
```

See also
Date.getHours , Date.setUTCHours

getUTCMilliseconds

Returns the milliseconds in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCMilliSeconds()

Parameters
None

Description
The value returned by getUTCMilliseconds is an integer between 0 and 999.

Examples
The following example assigns the milliseconds portion of the current time to the variable ms.

```
var ms;
Today = new Date();
ms = Today.getUTCMilliseconds();
```

See also
Date.getMilliseconds , Date.setUTCMilliseconds

getUTCMinutes

Returns the minutes in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCMinutes()

Parameters
None

Description
The value returned by getUTCMinutes is an integer between 0 and 59.

Examples
The following example assigns the minutes portion of the current time to the variable min.

```
var min;
Today = new Date();
min = Today.getUTCMinutes();
```

See also
Date.getMinutes , Date.setUTCMinutes

getUTCMonth

Returns the month according in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCMonth()

Parameters
None

Description
The value returned by getUTCMonth is an integer between 0 and 11 corresponding to the month. 0 for January, 1 for February, 2 for March, and so on.

Examples
The following example assigns the month portion of the current date to the variable mon.

```
var mon;
Today = new Date();
mon = Today.getUTCMonth();
```

See also
Date.getMonth , Date.setUTCMonth

font face="Arial, Helvetica, sans-serif" size= "4">**getUTCSeconds**

Returns the seconds in the specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
getUTCSeconds()

Parameters
None

Description
The value returned by getUTCSeconds is an integer between 0 and 59.

Examples
The following example assigns the seconds portion of the current time to the variable sec.

```
var sec;
Today = new Date();
sec = Today.getUTCSeconds();
```

See also
Date.getSeconds , Date.setUTCSeconds

getYear

Returns the year in the specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: deprecated; also, getYear returns the year minus 1900 regardless of the year specified

ECMA version	ECMA-262

Syntax
getYear()

Parameters
None

Description
getYear is no longer used and has been replaced by the getFullYear method.

The getYear method returns the year minus 1900; thus:

-
- For years above 2000, the value returned by getYear is 100 or greater. For example, if the year is 2026, getYear returns 126.
- For years between and including 1900 and 1999, the value returned by getYear is between 0 and 99. For example, if the year is 1976, getYear returns 76.
- For years less than 1900 or greater than 1999, the value returned by getYear is less than 0. For example, if the year is 1800, getYear returns -100.

To take into account years before and after 2000, you should use Date.getFullYear instead of getYear so that the year is specified in full.

Backward Compatibility

JavaScript 1.2 and earlier versions. The getYear method returns either a 2-digit or 4-digit year:

-
- For years between and including 1900 and 1999, the value returned by getYear is the year minus 1900. For example, if the year is 1976, the value returned is 76.
- For years less than 1900 or greater than 1999, the value returned by getYear is the four-digit year. For example, if the year is 1856, the value returned is 1856. If the year is 2026, the value returned is 2026.

Examples
Example 1. The second statement assigns the value 95 to the variable year.

```
Xmas = new Date("December 25, 1995 23:15:00")
year = Xmas.getYear() // returns 95
```

Example 2. The second statement assigns the value 100 to the variable year.

```
Xmas = new Date("December 25, 2000 23:15:00")
year = Xmas.getYear() // returns 100
```

Example 3. The second statement assigns the value -100 to the variable year.

```
Xmas = new Date("December 25, 1800 23:15:00")
year = Xmas.getYear() // returns -100
```

Example 4. The second statement assigns the value 95 to the variable year, representing the year 1995.

```
Xmas.setYear(95)
year = Xmas.getYear() // returns 95
```

See also
Date.getFullYear, Date.getUTCFullYear, Date.setYear

parse

Returns the number of milliseconds in a date string since January 1, 1970, 00:00:00, local time.

Method of	Date
Static	
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
Date.parse(*dateString*)

Parameters
:

dateString	

	A string representing a date.

Description
The parse method takes a date string (such as "Dec 25, 1995") and returns the number of milliseconds since January 1, 1970, 00:00:00 (local time). This function is useful for setting date values based on string values, for example in conjunction with the setTime method and the Date object.

Given a string representing a time, parse returns the time value. It accepts the IETF standard date syntax: "Mon, 25 Dec 1995 13:30:00 GMT". It understands the continental US time-zone abbreviations, but for general use, use a time-zone offset, for example, "Mon, 25 Dec 1995 13:30:00 GMT+0430" (4 hours, 30 minutes west of the Greenwich meridian). If you do not specify a time zone, the local time zone is assumed. GMT and UTC are considered equivalent.

Because parse is a static method of Date, you always use it as Date.parse(), rather than as a method of a Date object you created.

Examples
If IPOdate is an existing Date object, then you can set it to August 9, 1995 as follows:

```
IPOdate.setTime(Date.parse("Aug 9, 1995"))
```

See also
Date.UTC

prototype

Represents the prototype for this class. You can use the prototype to add properties or methods to all instances of a class. For information on prototypes, see Function.prototype.

Property of	Date
Implemented in	JavaScript 1.1, NES 2.0
ECMA version	ECMA-262

setDate

Sets the day of the month for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
setDate(*dayValue*)

Parameters

dayValue	An integer from 1 to 31, representing the day of the month.

Examples
The second statement below changes the day for theBigDay to July 24 from its original value.

```
theBigDay = new Date("July 27, 1962 23:30:00")
theBigDay.setDate(24)
```

See also
Date.getDate, Date.setUTCDate

setFullYear

Sets the full year for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setFullYear(*yearValue*[, *monthValue*[, *dayValue*]])

Parameters

yearValue	An integer specifying the numeric value of the year, for example, 1995.
monthValue	An integer between 0 and 11 representing the months January through December.
dayValue	An integer between 1 and 31 representing the day of the month. If you specify the dayValue parameter, you must also specify the monthValue.

Description
If you do not specify the monthValue and dayValue parameters, the values returned from the getMonth and getDate methods are used.

If a parameter you specify is outside of the expected range, setFullYear attempts to update the other parameters and the date information in the Date object accordingly. For example, if you specify 15 for monthValue, the year is incremented by 1 (year + 1), and 3 is used for the month.

Examples
```
theBigDay = new Date();
theBigDay.setFullYear(1997);
```

See also
Date.getUTCFullYear ,Date.setUTCFullYear, Date.setYear

setHours

Sets the hours for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: Added minutesValue, secondsValue, and msValue parameters.
ECMA version	ECMA-262

Syntax
setHours(*hoursValue*[, *minutesValue*[, *secondsValue*[, *msValue*]]])

Versions prior to JavaScript 1.3:

setHours(*hoursValue*)

Parameters

hoursValue	An integer between 0 and 23, representing the hour.
minutesValue	An integer between 0 and 59, representing the minutes.
secondsValue	An integer between 0 and 59, representing the seconds. If you specify the secondsValue parameter, you must also specify the minutesValue.
msValue	

	A number between 0 and 999, representing the milliseconds. If you specify the msValue parameter, you must also specify the minutesValue and secondsValue.

Description
If you do not specify the minutesValue, secondsValue, and msValue parameters, the values returned from the getUTCMinutes, getUTCSeconds, and getMilliseconds methods are used.

If a parameter you specify is outside of the expected range, setHours attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes will be incremented by 1 (min + 1), and 40 will be used for seconds.

Examples
theBigDay.setHours(7)

See also
Date.getHours, Date.setUTCHours

setMilliseconds

Sets the milliseconds for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setMilliseconds(*millisecondsValue*)

Parameters

millisecondsValue	A number between 0 and 999, representing the milliseconds.

Description
If you specify a number outside the expected range, the date information in the Date object is updated accordingly. For example, if you specify 1005, the number of seconds is incremented by 1, and 5 is used for the milliseconds.

Examples
theBigDay = new Date();
theBigDay.setMilliseconds(100);

See also
Date.getMilliseconds , Date.setUTCMilliseconds

setMinutes

Sets the minutes for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: Added secondsValue and msValue parameters.
ECMA version	ECMA-262

Syntax
setMinutes(*minutesValue*[, *secondsValue*[, *msValue*]])

Versions prior to JavaScript 1.3:

setMinutes(minutesValue)

Parameters

minutesValue	An integer between 0 and 59, representing the minutes.
secondsValue	An integer between 0 and 59, representing the seconds. If you specify the secondsValue parameter, you must also specify the minutesValue.
msValue	A number between 0 and 999, representing the milliseconds. If you specify the msValue parameter, you must also specify the minutesValue and secondsValue.

Examples
theBigDay.setMinutes(45)

Description
If you do not specify the secondsValue and msValue parameters, the values returned from getSeconds and getMilliseconds methods are used.

If a parameter you specify is outside of the expected range, setMinutes attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes (minutesValue) will be incremented by 1 (minutesValue + 1), and 40 will be used for seconds.

See also
Date.getMinutes, Date.setUTCMilliseconds

setMonth

Sets the month for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: Added dayValue parameter.

ECMA version	ECMA-262

Syntax
setMonth(*monthValue*[, *dayValue*])

Versions prior to JavaScript 1.3:

setMonth(*monthValue*)

Parameters

monthValue	An integer between 0 and 11 (representing the months January through December).
dayValue	An integer from 1 to 31, representing the day of the month.

Description
If you do not specify the dayValue parameter, the value returned from the getDate method is used.

If a parameter you specify is outside of the expected range, setMonth attempts to update the date information in the Date object accordingly. For example, if you use 15 for monthValue, the year will be incremented by 1 (year + 1), and 3 will be used for month.

Examples
theBigDay.setMonth(6)

See also
Date.getMonth, Date.setUTCMonth

setSeconds

Sets the seconds for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: Added msValue parameter.
ECMA version	ECMA-262

Syntax
setSeconds(*secondsValue*[, *msValue*])

Versions prior to JavaScript 1.3:

setSeconds(*secondsValue*)

Parameters

secondsValue	An integer between 0 and 59.
msValue	A number between 0 and 999, representing the milliseconds.

Description
If you do not specify the msValue parameter, the value returned from the getMilliseconds methods is used.

If a parameter you specify is outside of the expected range, setSeconds attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes stored in the Date object will be incremented by 1, and 40 will be used for seconds.

Examples
theBigDay.setSeconds(30)

See also
Date.getSeconds, Date.setUTCSeconds

setTime

Sets the value of a Date object according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
setTime(*timevalue*)

Parameters

timevalue	An integer representing the number of milliseconds since 1 January 1970 00:00:00.

Description
Use the setTime method to help assign a date and time to another Date object.

Examples
```
theBigDay = new Date("July 1, 1999")
sameAsBigDay = new Date()
sameAsBigDay.setTime(theBigDay.getTime())
```

See also
Date.getTime, Date.setUTCHours

setUTCDate

Sets the day of the month for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCDate(*dayValue*)

Parameters

dayValue	An integer from 1 to 31, representing the day of the month.

Description
If a parameter you specify is outside of the expected range, setUTCDate attempts to update the date information in the Date object accordingly. For example, if you use 40 for dayValue, and the month stored in the Date object is June, the day will be changed to 10 and the month will be incremented to July.

Examples
```
theBigDay = new Date();
theBigDay.setUTCDate(20);
```

See also
Date.getUTCDate , Date.setDate

setUTCFullYear

Sets the full year for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCFullYear(*yearValue*[, *monthValue*[, *dayValue*]])

Parameters

yearValue	An integer specifying the numeric value of the year, for example, 1995.
monthValue	An integer between 0 and 11 representing the months January through December.
dayValue	An integer between 1 and 31 representing the day of the month. If you specify the dayValue parameter, you must also specify the monthValue.

Description
If you do not specify the monthValue and dayValue parameters, the values returned from the getMonth and getDate methods are used.

If a parameter you specify is outside of the expected range, setUTCFullYear attempts to update the other parameters and the date information in the Date object accordingly. For example, if you specify 15 for monthValue, the year is incremented by 1 (year + 1), and 3 is used for the month.

Examples
```
theBigDay = new Date();
theBigDay.setUTCFullYear(1997);
```

See also
Date.getUTCFullYear , Date.setFullYear

setUTCHours

Sets the hour for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCHours(*hoursValue*[, *minutesValue*[, *secondsValue*[, *msValue*]]])

Parameters

hoursValue	An integer between 0 and 23, representing the hour.
minutesValue	An integer between 0 and 59, representing the minutes.
secondsValue	An integer between 0 and 59, representing the seconds. If you specify the secondsValue parameter, you must also specify the minutesValue.
msValue	A number between 0 and 999, representing the milliseconds. If you specify the msValue parameter, you must also specify the minutesValue and secondsValue.

Description
If you do not specify the minutesValue, secondsValue, and msValue parameters, the values returned from the getUTCMinutes, getUTCSeconds, and getUTCMilliseconds methods are used.

If a parameter you specify is outside of the expected range, setUTCHours attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes will be incremented by 1 (min + 1), and 40 will be used for seconds.

Examples
theBigDay = new Date();
theBigDay.setUTCHours(8);

See also
Date.getUTCHours , Date.setHours

setUTCMilliseconds

Sets the milliseconds for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCMilliseconds(*millisecondsValue*)

Parameters

millisecondsValue	A number between 0 and 999, representing the milliseconds.

Description
If a parameter you specify is outside of the expected range, setUTCMilliseconds attempts to update the date information in the Date object accordingly. For example, if you use 1100 for millisecondsValue, the seconds stored in the Date object will be incremented by 1, and 100 will be used for milliseconds.

Examples
theBigDay = new Date();
theBigDay.setUTCMilliseconds(500);

See also
Date.getUTCMilliseconds , Date.setMilliseconds

setUTCMinutes

Sets the minutes for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCMinutes(*minutesValue*[, *secondsValue*[, *msValue*]])

Parameters

minutesValue	An integer between 0 and 59, representing the minutes.
secondsValue	An integer between 0 and 59, representing the seconds. If you

	specify the secondsValue parameter, you must also specify the minutesValue.
msValue	A number between 0 and 999, representing the milliseconds. If you specify the msValue parameter, you must also specify the minutesValue and secondsValue.

Description
If you do not specify the secondsValue and msValue parameters, the values returned from getUTCSeconds and getUTCMilliseconds methods are used.

If a parameter you specify is outside of the expected range, setUTCMinutes attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes (minutesValue) will be incremented by 1 (minutesValue + 1), and 40 will be used for seconds.

Examples
theBigDay = new Date();
theBigDay.setUTCMinutes(43);

See also
Date.getUTCMinutes , Date.setMinutes

setUTCMonth

Sets the month for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
setUTCMonth(*monthValue*[, *dayValue*])

Parameters

monthValue	An integer between 0 and 11, representing the months January through December.
dayValue	An integer from 1 to 31, representing the day of the month.

Description
If you do not specify the dayValue parameter, the value returned from the getUTCDate method is used.

If a parameter you specify is outside of the expected range, setUTCMonth attempts to update the date information in the Date object accordingly. For example, if you use 15 for monthValue, the year will be incremented by 1 (year + 1), and 3 will be used for month.

Examples
theBigDay = new Date();
theBigDay.setUTCMonth(11);

See also
Date.getUTCMonth , Date.setMonth

setUTCSeconds

Sets the seconds for a specified date according to universal time.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax

setUTCSeconds(*secondsValue*[, *msValue*])

Parameters

secondsValue	An integer between 0 and 59.
msValue	A number between 0 and 999, representing the milliseconds.

Description
If you do not specify the msValue parameter, the value returned from the getUTCMilliseconds methods is used.

If a parameter you specify is outside of the expected range, setUTCSeconds attempts to update the date information in the Date object accordingly. For example, if you use 100 for secondsValue, the minutes stored in the Date object will be incremented by 1, and 40 will be used for seconds.

Examples
```
theBigDay = new Date();
theBigDay.setUTCSeconds(20);
```

See also
Date.getUTCSeconds , Date.setSeconds

setYear

Sets the year for a specified date according to local time.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 Deprecated in JavaScript 1.3.

ECMA version	ECMA-262

Syntax
setYear(*yearValue*)

Parameters

yearValue	An integer.

Description
setYear is no longer used and has been replaced by the setFullYear method.

If yearValue is a number between 0 and 99 (inclusive), then the year for dateObjectName is set to 1900 + yearValue. Otherwise, the year for dateObjectName is set to yearValue.

To take into account years before and after 2000, you should use setFullYear instead of setYear so that the year is specified in full.

Examples
Note that there are two ways to set years in the 20th century.

Example 1. The year is set to 1996.

```
theBigDay.setYear(96)
```

Example 2. The year is set to 1996.

```
theBigDay.setYear(1996)
```

Example 3. The year is set to 2000.

```
theBigDay.setYear(2000)
```

See also
Date.getYear, Date.setFullYear, Date.setUTCFullYear

toGMTString

Converts a date to a string, using the Internet GMT conventions.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0 Deprecated in JavaScript 1.3.
ECMA version	ECMA-262

Syntax
toGMTString()

Parameters
None

Description
toGMTString is no longer used and has been replaced by the toUTCString method.

The exact format of the value returned by toGMTString varies according to the platform.

You should use Date.toUTCString instead of toGMTSTring.

Examples
In the following example, today is a Date object:

today.toGMTString()

In this example, the toGMTString method converts the date to GMT (UTC) using the operating system's time-zone offset and returns a string value that is similar to the following form. The exact format depends on the platform.

Mon, 18 Dec 1995 17:28:35 GMT

See also
Date.toLocaleString, Date.toUTCString

toLocaleString

Converts a date to a string, using the current locale's conventions.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
toLocaleString()

Parameters
None

Description
The toLocaleString method relies on the underlying operating system in formatting dates. It converts the date to a string using the formatting convention of the operating system where the script is running. For example, in the United States, the month appears before the date (04/15/98), whereas in Germany the date appears before the month (15.04.98). If the operating system is not year-2000 compliant and does not use the full year for years before 1900 or over 2000, toLocaleString returns a string that is not year-2000 compliant. toLocaleString behaves similarly to toString when converting a year that the operating system does not properly format.

Methods such as getHours, getMinutes, and getSeconds give more portable results than toLocaleString.

Examples
In the following example, today is a Date object:

```
today = new Date(95,11,18,17,28,35) //months are represented by 0 to 11
today.toLocaleString()
```

In this example, toLocaleString returns a string value that is similar to the following form. The exact format depends on the platform.

12/18/95 17:28:35

See also
Date.toGMTString, Date.toUTCString

toLocaleDateString

Converts a date to a string, returning the "date" portion using the current locale's conventions.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
toLocaleDateString()

Parameters
None

Description
The toLocaleDateString method relies on the underlying operating system in formatting dates. It converts the date to a string using the formatting convention of the operating system where the script is running. For example, in the United States, the month appears before the date (04/15/98), whereas in Germany the date appears before the month (15.04.98). If the operating system is not year-2000 compliant and does not use the full year for years before 1900 or over 2000, toLocaleDateString returns a string that is not year-2000 compliant. toLocaleDateString behaves similarly to toString when converting a year that the operating system does not properly format.

Methods such as getHours, getMinutes, and getSeconds give more portable results than toLocaleDateString.

Examples
In the following example, today is a Date object:

```
today = new Date(95,11,18,17,28,35) //months are represented by 0 to 11
today.toLocaleDateString()
```

In this example, toLocaleDateString returns a string value that is similar to the following form. The exact format depends on the platform.

12/18/95

See also
Date.toGMTString, Date.toUTCString

toLocaleTimeString

Converts a date to a string, returning the "date" portion using the current locale's conventions.

Method of	Date
Implemented in	JavaScript 1.0, NES 2.0
ECMA version	ECMA-262

Syntax
toLocaleTimeString()

Parameters
None

Description
The toLocaleTimeString method relies on the underlying operating system in formatting dates. It converts the date to a string using the formatting convention of the operating system where the script is running. For example, in the United States, the month appears before the date (04/15/98), whereas in Germany the date appears before the month (15.04.98). If the operating system is not year-2000 compliant and does not use the full year for years before 1900 or over 2000, toLocaleTimeString returns a string that is not year-2000 compliant. toLocaleTimeString behaves similarly to toString when converting a year that the operating system does not properly format.

Methods such as getHours, getMinutes, and getSeconds give more portable results than toLocaleTimeString.

Examples
In the following example, today is a Date object:

```
today = new Date(95,11,18,17,28,35) //months are represented by 0 to 11
today.toLocaleTimeString()
```

In this example, toLocaleTimeString returns a string value that is similar to the following form. The exact format depends on the platform.

17:28:35

See also
Date.toGMTString, Date.toUTCString

toSource

Returns a string representing the source code of the object.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
toSource()

Parameters
None

Description
The toSource method returns the following values:

-
- For the built-in Date object, toSource returns the following string indicating that the source code is not available:

```
function Date() {
  [native code]
}
```

- For instances of Date, toSource returns a string representing the source code.

This method is usually called internally by JavaScript and not explicitly in code.

See also
Object.toSource

toString

Returns a string representing the specified Date object.

Method of	Date
Implemented in	JavaScript 1.1, NES 2.0
ECMA version	ECMA-262

Syntax
toString()

Parameters
None.

Description
The Date object overrides the toString method of the Object object; it does not inherit Object.toString. For Date objects, the toString method returns a string representation of the object.

JavaScript calls the toString method automatically when a date is to be represented as a text value or when a date is referred to in a string concatenation.

Examples
The following example assigns the toString value of a Date object to myVar:

```
x = new Date();
myVar=x.toString();   //assigns a value to myVar similar to:
    //Mon Sep 28 14:36:22 GMT-0700 (Pacific Daylight Time) 1998
```

See also
Object.toString

toUTCString

Converts a date to a string, using the universal time convention.

Method of	Date
Implemented in	JavaScript 1.3
ECMA version	ECMA-262

Syntax
toUTCString()

Parameters
None

Description
The value returned by toUTCString is a readable string formatted according to UTC convention. The format of the return value may vary according to the platform.

Examples
```
var UTCstring;
Today = new Date();
UTCstring = Today.toUTCString();
```

See also
Date.toLocaleString, Date.toUTCString

UTC

Returns the number of milliseconds in a Date object since January 1, 1970, 00:00:00, universal time.

Method of	Date
Static	
Implemented in	JavaScript 1.0, NES 2.0 JavaScript 1.3: added ms parameter.

ECMA version	ECMA-262

Syntax
Date.UTC(*year*, *month*[, *day*[, *hrs*[, *min*[, *sec*[, *ms*]]]]])

Parameters

year	A year after 1900.
month	An integer between 0 and 11 representing the month.
date	An integer between 1 and 31 representing the day of the month.
hrs	An integer between 0 and 23 representing the hours.
min	An integer between 0 and 59 representing the minutes.
sec	An integer between 0 and 59 representing the seconds.
ms	An integer between 0 and 999 representing the milliseconds.

Description
UTC takes comma-delimited date parameters and returns the number of milliseconds between January 1, 1970, 00:00:00, universal time and the time you specified.

You should specify a full year for the year; for example, 1998. If a year between 0 and 99 is specified, the method converts the year to a year in the 20th century (1900 + year); for

example, if you specify 95, the year 1995 is used.

The UTC method differs from the Date constructor in two ways.

-
- Date.UTC uses universal time instead of the local time.
- Date.UTC returns a time value as a number instead of creating a Date object.

If a parameter you specify is outside of the expected range, the UTC method updates the other parameters to allow for your number. For example, if you use 15 for month, the year will be incremented by 1 (year + 1), and 3 will be used for the month.

Because UTC is a static method of Date, you always use it as Date.UTC(), rather than as a method of a Date object you created.

Examples
The following statement creates a Date object using GMT instead of local time:

```
gmtDate = new Date(Date.UTC(96, 11, 1, 0, 0, 0))
```

See also
Date.parse

valueOf

Returns the primitive value of a Date object.

Method of	Date
Implemented in	JavaScript 1.1
ECMA version	ECMA-262

Syntax
```
valueOf()
```

Parameters
None

Description
The valueOf method of Date returns the primitive value of a Date object as a number data type, the number of milliseconds since midnight 01 January, 1970 UTC.

This method is usually called internally by JavaScript and not explicitly in code.

Examples
```
x = new Date(56,6,17);
myVar=x.valueOf()      //assigns -424713600000 to myVar
```

See also
Object.valueOf

Previous **Contents** **Index** **Next**

Last Updated **May 19, 2003**