22. JavaScript Date Object

The Date object is used to work with dates and times.

22.1. Create a Date Object

The Date object is used to work with dates and times.

Date objects are created with the Date() constructor.

There are four ways of initiating a date:

```
new Date() // current date and time
new Date(milliseconds) //milliseconds since 1970/01/01
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

Most parameters above are optional. Not specifying, causes 0 to be passed in.

Once a Date object is created, a number of methods allow you to operate on it. Most methods allow you to get and set the year, month, day, hour, minute, second, and milliseconds of the object, using either local time or UTC (universal, or GMT) time.

All dates are calculated in milliseconds from 01 January, 1970 00:00:00 Universal Time (UTC) with a day containing 86,400,000 milliseconds.

Some examples of initiating a date:

```
var today = new Date()
var d1 = new Date("October 13, 1975 11:13:00")
var d2 = new Date(79,5,24)
var d3 = new Date(79,5,24,11,33,0)
```

22.2. Set Dates

We can easily manipulate the date by using the methods available for the Date object.

In the example below we set a Date object to a specific date (14th January 2010):

```
var myDate=new Date();
myDate.setFullYear(2010,0,14);
```

And in the following example we set a Date object to be 5 days into the future:

```
var myDate=new Date();
```

```
myDate.setDate(myDate.getDate()+5);
```

Note: If adding five days to a date shifts the month or year, the changes are handled automatically by the Date object itself!

22.3. Compare Two Dates

The Date object is also used to compare two dates.

The following example compares today's date with the 14th January 2100:

```
var x=new Date();
x.setFullYear(2100,0,14);
var today = new Date();

if (x>today)
  {
  alert("Today is before 14th January 2100");
  }
else
  {
  alert("Today is after 14th January 2100");
  }
```

22.4. More Examples

a) Return today's date and time. How to use the Date() method to get today's date.

```
<!DOCTYPE html>
<html>
<body>

<script>

var d=new Date();
document.write(d);

</script>

</body>
</html>
```

b) getFullYear(). Use getFullYear() to get the year.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to display the full
year of todays date.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
var d = new Date();
var x = document.getElementById("demo");
x.innerHTML=d.getFullYear();
}
</script>

</body>
</html>
```

c) getTime(). getTime() returns the number of milliseconds since 01.01.1970.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to display the number
of milliseconds since midnight, January 1,
1970.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
var d = new Date();
var x = document.getElementById("demo");
x.innerHTML=d.getTime();
}
</script>

</body>
```

```
</html>
```

d) setFullYear(). How to use setFullYear() to set a specific date.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to display a date
after changing the year, month, and day.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
var d = new Date();
d.setFullYear(2020,10,3);
var x = document.getElementById("demo");
x.innerHTML=d;
}
</script>

<p>Remember that JavaScript counts months from 0 to
11.
Month 10 is November.</p>
</body>
</html>
```

e) toUTCString(). How to use toUTCString() to convert today's date (according to UTC) to a string.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to display the UTC
date and time as a string.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
```

```
var d = new Date();
var x = document.getElementById("demo");
x.innerHTML=d.toUTCString();
}
</script>

</body>
</html>
```

f) getDay(). Use getDay() and an array to write a weekday, and not just a number.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to display todays day
of the week.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
var d = new Date();
var weekday=new Array(7);
weekday[0]="Sunday";
weekday[1]="Monday";
weekday[2]="Tuesday";
weekday[3]="Wednesday";
weekday[4]="Thursday";
weekday[5]="Friday";
weekday[6]="Saturday";

var x = document.getElementById("demo");
x.innerHTML=weekday[d.getDay()];
}
</script>

</body>
</html>
```

g) Display a clock. How to display a clock on your web page.

```
<!DOCTYPE html>
<html>
<head>
<script>
function startTime()
{
var today=new Date();
var h=today.getHours();
var m=today.getMinutes();
var s=today.getSeconds();
// add a zero in front of numbers<10
m=checkTime(m);
s=checkTime(s);
document.getElementById('txt').innerHTML=h+":"+m+":
"+s;
t=setTimeout(function(){startTime()},500);
}

function checkTime(i)
{
if (i<10)
  {
  i="0" + i;
  }
return i;
}
</script>
</head>

<body onload="startTime()">
<div id="txt"></div>
</body>
</html>
```

22.5. Complete Date Object Reference

For a complete reference of all the properties and methods that can be used with the Date object, go to our complete Date object reference. The Date object is used to work with dates and times.

Date objects are created with new Date().

There are four ways of instantiating a date:

```
var d = new Date();
var d = new Date(milliseconds);
var d = new Date(dateString);
var d = new Date(year, month, day, hours, minutes, seconds, milliseconds);
```

Date Object Properties

Property	Description
constructor	Returns the function that created the Date object's prototype
prototype	Allows you to add properties and methods to an object

Date Object Methods

Method	Description
getDate()	Returns the day of the month (from 1-31)
getDay()	Returns the day of the week (from 0-6)
getFullYear()	Returns the year (four digits)
getHours()	Returns the hour (from 0-23)
getMilliseconds()	Returns the milliseconds (from 0-999)
getMinutes()	Returns the minutes (from 0-59)
getMonth()	Returns the month (from 0-11)
getSeconds()	Returns the seconds (from 0-59)
getTime()	Returns the number of milliseconds since midnight Jan 1, 1970
getTimezoneOffset()	Returns the time difference between UTC time and local time, in minutes
getUTCDate()	Returns the day of the month, according to universal time (from 1-31)
getUTCDay()	Returns the day of the week, according to universal time (from 0-6)
getUTCFullYear()	Returns the year, according to universal time (four digits)
getUTCHours()	Returns the hour, according to universal time (from 0-23)
getUTCMilliseconds()	Returns the milliseconds, according to universal time (from 0-999)
getUTCMinutes()	Returns the minutes, according to universal time (from 0-59)
getUTCMonth()	Returns the month, according to universal time (from 0-11)

getUTCSeconds()	Returns the seconds, according to universal time (from 0-59)
getYear()	Deprecated. Use the getFullYear() method instead
parse()	Parses a date string and returns the number of milliseconds since midnight of January 1, 1970
setDate()	Sets the day of the month of a date object
setFullYear()	Sets the year (four digits) of a date object
setHours()	Sets the hour of a date object
setMilliseconds()	Sets the milliseconds of a date object
setMinutes()	Set the minutes of a date object
setMonth()	Sets the month of a date object
setSeconds()	Sets the seconds of a date object
setTime()	Sets a date and time by adding or subtracting a specified number of milliseconds to/from midnight January 1, 1970
setUTCDate()	Sets the day of the month of a date object, according to universal time
setUTCFullYear()	Sets the year of a date object, according to universal time (four digits)
setUTCHours()	Sets the hour of a date object, according to universal time
setUTCMilliseconds()	Sets the milliseconds of a date object, according to universal time
setUTCMinutes()	Set the minutes of a date object, according to universal time
setUTCMonth()	Sets the month of a date object, according to universal time
setUTCSeconds()	Set the seconds of a date object, according to universal time
setYear()	Deprecated. Use the setFullYear() method instead
toDateString()	Converts the date portion of a Date object into a readable string
toGMTString()	Deprecated. Use the toUTCString() method instead
toISOString()	Returns the date as a string, using the ISO standard
toJSON()	Returns the date as a string, formated as a JSON date
toLocaleDateString()	Returns the date portion of a Date object as a string, using locale conventions
toLocaleTimeString()	Returns the time portion of a Date object as a string, using locale conventions

toLocaleString()	Converts a Date object to a string, using locale conventions
toString()	Converts a Date object to a string
toTimeString()	Converts the time portion of a Date object to a string
toUTCString()	Converts a Date object to a string, according to universal time
UTC()	Returns the number of milliseconds in a date string since midnight of January 1, 1970, according to universal time
valueOf()	Returns the primitive value of a Date object