Part 5

Other Objects

In addition to the objects we have already encountered, there are a number of other objects that form part of the JavaScript language. Among the more important and useful of these are the `Date` and `Math` objects.

The `Date` object

The `Date` object allows us to obtain the current date and time, and to perform various timing operations.

In order to use the `Date` object, we must first create a new 'instance' of it. This is done in the following way:

```
var myDateObject = new Date;
```

This creates an object called `myDateObject` that contains information about the date, time, etc., ***at the instant it was created***. The information in `myDateObject` doesn't change as time passes, so if you want to know the correct time a few minutes later you will have to create a new instance of the `Date` object.

Once you have created an instance of the `Date` object, you can use any of the methods below to obtain information from it:

`getFullYear()` Returns the current year as a four-digit number (e.g., 2000). For example:

```
var myDateObject = new Date;
var currentYear = myDateObject.getFullYear();
alert(currentYear);
```

Get Year Click here to see this example working.

`getMonth()` Returns the current month as an integer from 0-11 (e.g., November = 10). For example:

```
var myDateObject = new Date;
var currentMonth = myDateObject.getMonth();
alert(currentMonth);
```

Get Month Click here to see this example working.

`getDate()` Returns the day of the month as an integer between 1 and 31. For example:

```
var myDateObject = new Date;
var currentDate = myDateObject.getDate();
```

```
alert(currentDate);
```

Get Date Click here to see this example working.

`getDay()` Returns the day of the week as an integer between 0 and 6, starting from Sunday (e.g., Tuesday = 2). For example:

```
var myDateObject = new Date;
var currentDay = myDateObject.getDay();
alert(currentDay);
```

Get Day Click here to see this example working.

`getHours()` Returns the hour of the day as an integer between 0 and 23. For example:

```
var myDateObject = new Date;
var currentHour = myDateObject.getHours();
alert(currentHour);
```

Get Hour Click here to see this example working.

`getMinutes()` Returns the number of minutes since the beginning of the hour as an integer. For example:

```
var myDateObject = new Date;
var currentMinute = myDateObject.getMinutes();
alert(currentMinute);
```

Get Minute Click here to see this example working.

`getSeconds()` Returns the number of seconds since the start of the minute as an integer. For example:

```
var myDateObject = new Date;
var currentSecond = myDateObject.getSeconds();
alert(currentSecond);
```

Get Second Click here to see this example working.

In order to use the `Date` object, it is often necessary to convert the data it produces, e.g., to obtain the names of days and months rather than just numbers. To see an example of the `Date` object in use, click here.

The `Date` object also has methods to obtain time intervals as small as milliseconds, to convert between various time systems, to parse dates and times in various formats into individual elements, and to perform various other time-related operations.

The `Math` object

The `Math` object allows us to perform various mathematical operations not provided by the basic operators we have already looked at.

Its methods include the following:

`sqrt(x)` Returns the square root of *x*. For example:

```
var inputValue = prompt("Please enter a value",
"");
var squareRoot = Math.sqrt(inputValue);
alert(squareRoot);
```

Find Square Root Click here to see this example working.

`log(x)` Returns the natural logarithm of *x*. For example:

```
var inputValue = prompt("Please enter a value",
"");
var logOfX = Math.log(inputValue);
alert(logOfX);
```

Find Logarithm Click here to see this example working.

`max(x,y)` Returns whichever is the larger of *x* and *y*. For example:

```
var inputX = prompt("Please enter a value for X",
"");
var inputY = prompt("Please enter a value for Y",
"");
var largerOfXY = Math.max(inputX, inputY);
alert(largerOfXY);
```

Find The Larger Click here to see this example working.

`min(x,y)` Returns whichever is the smaller of *x* and *y*.
Works in a similar way to `max(x,y)`, above.

`round(x)` Returns the value of *x* rounded to the nearest integer. For example:

```
var inputvalue = prompt("Please enter a value",
"");
var roundedValue = Math.round(inputValue);
alert(roundedValue);
```

Round Click here to see this example working.

`ceil(x)` Returns the absolute value of *x* rounded **up** to the next integer value. Works in a similar way to `round(x)`, above.

`floor(x)` Returns the absolute value of *x* rounded **down** to the next integer value. Works in a similar way to `round(x)`, above.

`abs(x)` Returns the absolute value of *x*. For example:

```
var rawValue = prompt("Please enter a value", "")
var absValue = Math.abs(rawValue)
alert(absValue);
```

Get Absolute value Click here to see this example working.

`pow(x,y)` Returns the value of *x* raised to the power *y*. For example:

```
var baseValue = prompt ("Please enter base value",
"");
var expValue = prompt ("Please enter exponent",
"");
var baseToPower = Math.pow(baseValue, expValue);
alert(baseToPower);
```

Raise To Power Click here to see this example working.

The `Math` object also has methods to perform trigonmetrical operations such as `sin(), cos(), tan(),` etc., and a set of properties that include values for *pi* and other constants.