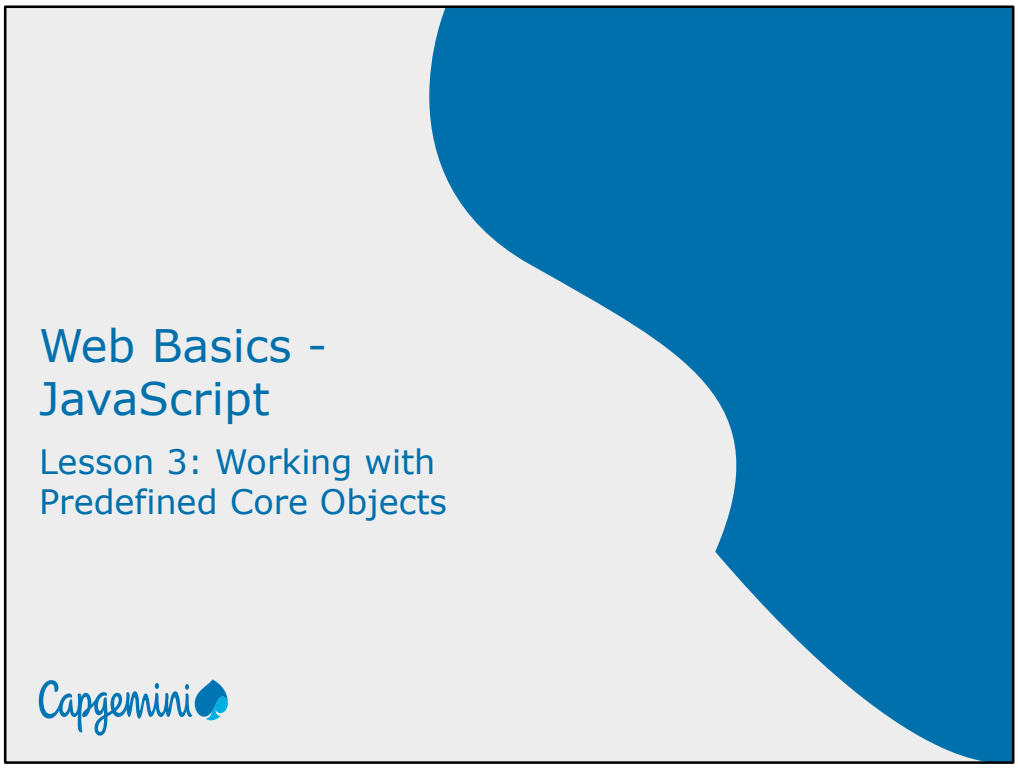

Web Basics -
JavaScript
Lesson 3: Working with
Predefined Core Objects
Capgemini

Lesson Objectives

- Data Types in JavaScript
- String Object
- Math Object
- Date Object

3.1: Data Types in JavaScript

Data Types in JavaScript

- JavaScript has predefined objects and uses standard browser objects. Some of them are discussed here

- Predefined objects in JavaScript are:
  - String
  - Math
  - Date

3.2: String Object

String Object

- Properties of a string object:
  - Length: The length property returns the number of characters in a string.
  - Syntax : stringObject.length
  - "Lincoln".length // result = 7

String Objects: Creating a String

A string consists of one or more standard text characters between matching quote marks. JavaScript is forgiving in one regard: You can use single or double quotes, as long as you match two single quotes or two double quotes around a string. JavaScript draws a fine line between a string value and a string object. Both let you use the same methods on their contents, so by and large, you do not have to create a string object (with the new String() constructor) every time you want to assign a string value to a variable. A simple assignment operation (var myString = "fred") is all you need to create a string value that behaves on the surface very much like a full-fledged string object.

```
var myString = new String("characters")
var myString = "fred"
```

Properties of String Objects:

Length:

The most frequently used property of a string is length. To derive the length of a string, extract its property as you would extract the length property of any object. The length value represents an integer count of the number of characters within the string. Spaces and punctuation symbols count as characters. Any backslash special characters embedded in a string count as one character, including such characters as newline and tab.

```
"Four score".length // result = 10
"One\ntwo".length // result = 7
"".length // result = 0
```

3.2: String Object

String Object(Parsing Methods)

- charAt(index): The charAt() method returns the character at a specified position.
  - Syntax: String charAt(index)
  - "HelloWorld".charAt(5)// result "W"
- concat()
  - The concat() method is used to join two or more strings
  - One or more string objects to be joined to a string
  - Syntax: stringObject.concat(stringX,stringX,...,stringX)

String Objects (Parsing Methods):

String.chartAt(index)

Use the string.charAt() method to extract a single character from a string when you know the position of that character. For this method, you specify an index value in the string as a parameter to the method. The index value of the first character of the string is 0. If your script needs to get a range of characters, use the string.substring() method. It is a common mistake to use string.substring() to extract a character from inside a string, when the string.charAt() method is more efficient.

string.concat(string2)

Returns: Combined string.

"abc".concat("def") // result: "abcdef"

JavaScript's add-by-value operator (+=) provides a convenient way to concatenate strings. N4 and IE4, however, introduces a string object method that performs the same task. The base string to which more text is appended is the object or value to the left of the period. The string to be appended is the parameter of the method, as the example demonstrates. Like the add-by-value operator, the concat() method doesn't know about word endings. You are responsible for including the necessary space between words if the two strings require a space between them in the result.

3.2: String Object

String Object(Parsing Methods) Contd…

- match(regExpression)
  - Searches for a specified value in a string
  - Syntax: string.match(regExpression)
- replace(regExpression, replaceString)
  - Replaces some characters with some other characters in a string.
  - Syntax: string.replace(regExpression, replaceString)
- substr(start [, length])
  - Extracts a specified number of characters in a string, from a start index .
  - Syntax: string.substr(start [, length])

String Objects (Parsing Methods contd..):

string.match(regExpression) **:**Returns Array of matching strings.
The string.match() method relies on the RegExp (regular expression) object. The parameter must be a regular expression object. This method returns an array value when at least one match turns up; otherwise the returned value is null.
string.replace(regExpression, replaceString): Returns Changed string.
Regular expressions are commonly used to perform search-and-replace operations. JavaScript's string.replace() method provides a simple framework in which to perform this kind of operation on any string.

```
var str = “To be, or not to be: that is the question.”
var regexp = /be/
str.relace(regexp, “exist”)
```

string.substr(start [, length]) :Returns Characters of the string from the first character of the given string to the specified length.

3.2: String Object

String Object(Converting Methods)

- toLowerCase()
  - Displays a string in lowercase letters
  - string.toLowerCase()
- toUpperCase()
  - Displays a string in uppercase letters
  - string.toUpperCase()

String Objects (Parsing Methods contd..):

string.toLowerCase() and string.toUpperCase()

Returns: The string in all lower- or uppercase, depending on which method you invoke.

A great deal of what takes place on the Internet (and in JavaScript) is case-sensitive. URLs on some servers, for instance, are case-sensitive for directory names and filenames. These two methods, the simplest of the string methods, convert any string to either all lowercase or all uppercase. Any mixed-case strings get converted to a uniform case. If you want to compare user input from a field against some coded string without worrying about matching case, you should convert both strings to the same case for the comparison.

3.2: String Object

String Object (Formatting Methods)

- Formatting Methods:
  - string.bold() : Displays a string in bold
  - string.italics() : Displays a string in italic
  - string.fontcolor (colorValue) : Displays a string in a specified color
  - string.fontsize(integer1to7) : Displays a string in a specified size
  - string.big() : Displays a string in a big font
  - string.small() : Displays a string in a small font

String Objects (Formatting Methods):

Now we come to the other group of string object methods, which ease the process of creating the numerous string display characteristics when you use JavaScript to assemble HTML code.

You can still use the standard HTML tags instead of by calling the string methods in your web pages. The choice is up to you. One advantage to the string methods is that they never forget the ending tag of a tag pair.

string.fontsize() and string.fontcolor() also affect the font characteristics of strings displayed in the HTML page. The parameters for these items are pretty straightforward —an integer between 1 and 7 corresponding to the seven browser font sizes and a color value (as either a hexadecimal triplet or color constant name) for the designated text.

3.2: String Object

URL String Encoding and Decoding

- JavaScript includes two functions for encoding & decoding
  - escape()
  - encodes the string that is contained in the string argument to make it portable.
  - So it can be transmitted across any network to any computer that supports ASCII characters.
- unescape()
  - Use the unescape function to decode an encoded sequence that was created using escape.

URL String Encoding and Decoding

escape("Howdy Pardner") // result = "Howdy%20Pardner"
unescape("Howdy%20Pardner") // result = "Howdy Pardner"

When browsers and servers communicate, some nonalphanumeric characters that we take for granted (such as a space) cannot make the journey in their native form. Only a narrower set of letters, numbers, and punctuation is allowed. To accommodate the rest, the characters must be encoded with a special symbol (%)and their hexadecimal ASCII values. For example, the space character is hex 20 (ASCII decimal 32). When encoded, it looks like %20. You may have seen this symbol in browser history lists or URLs.

JavaScript includes two functions, escape() and unescape(), that offer instant conversion of whole strings. To convert a plain string to one with these escape codes, use the escape function, as in escape("Howdy Pardner") // result = "Howdy%20Pardner". The unescape() function converts the escape codes into human-readable form.

Demo

- string_len.html
- string_method.html
- string_style.html

3.3: Math Object

Math Object - Properties & Methods

Property/ Method	Description
Math.PI	PI (3.141592653589793116)
Math.SQRT2	Square root of 2 (1.4142)
Math.random()	Random number between 0 and 1
Math.round(val)	N+1 when val >= n.5; otherwise N
Math.max(val1, val2)	The greater of val1 or val2
Math.min(val1, val2)	The lesser of val1 or val2

Math Object – Properties & Methods:

The Math object allows you to perform mathematical tasks. All properties and methods can be called without creating the Math object.

You can use them in your regular arithmetic expressions since they return constant values. For example, to obtain the circumference of a circle whose diameter is in variable d, you use the statement shown below.

circumference = d * Math.PI

The Math.random() method returns a floating-point value between 0 and 1. If you are designing a script to act like a card game, you need random integers between 1 and 52; for dice, the range is 1 to 6 per die. To generate a random integer between zero and any top value, use the first formula shown where n is the top number.

To generate random numbers between a different range use the second formula where m is the lowest possible integer value of the range and n equals the top number of the range minus m. In other words n+m should add up to the highest number of the range you want. For the dice game, use the third formula for each throw of the die.

Math.round(Math.random() * n)

Math.round(Math.random() * n) + m

newDieValue = Math.round(Math.random() * 5) + 1

Demo

- Rand_fun.html

3.4: Date Object

Date

- Properties and Methods:
- var dateObject = new Date([parameters])

Properties	Description
dateObj.getTime()	Milliseconds since 1/1/70 00:00:00 GMT
dateObj.getYear()	Specified year minus 1900
dateObj.getMonth()	Month within the year (January = 0)
dateObj.getDate()	Date within the month

Date Object: Creating a Date object

```
var dateObject = new Date([parameters])
new Date("Month dd, yyyy hh:mm:ss")
new Date("Month dd, yyyy")
new Date(yy,mm,dd,hh,mm,ss)
new Date(yy,mm,dd)
new Date(milliseconds)
```

The Date object evaluates to an object rather than to some string or numeric value. If you leave the parameters empty, JavaScript takes that to mean you want today's date and the current time to be assigned to that new Date object. To create a Date object for a specific date or time, you have five ways to send values as a parameter to the new Date() constructor function.
The Date object has only a prototype property, which enables you to apply new properties and methods to every Date object created in the current page.

3.4: Date Object

Date and Time Arithmetic

- To simplify the tasks of formatting and manipulating dates, JavaScript provides a Date object along with some extra functions that help you work with dates.

```
var oneMinute = 60 * 1000
var oneHour = oneMinute * 60
var oneDay = oneHour * 24
var oneWeek = oneDay * 7
targetDate = new Date()
dateInMs = targetDate.getTime()
dateInMs += oneWeek
targetDate.setTime(dateInMs)
```

Date and time arithmetic:

You may need to perform some math with dates for any number of reasons. Perhaps you need to calculate a date at some fixed number of days or weeks in the future or figure out the number of days between two dates. When calculations of these types are required, remember the *lingua franca* of JavaScript date values: the milliseconds.

What you may need to do in your date-intensive scripts is establish some variable values representing the number of milliseconds for minutes, hours, days, or weeks, and then use those variables in your calculations. On the slide you can see an example that establishes some practical variable values, building on each other.

With these values established in a script, you can use one to calculate the date one week from today. Following is the complete example.

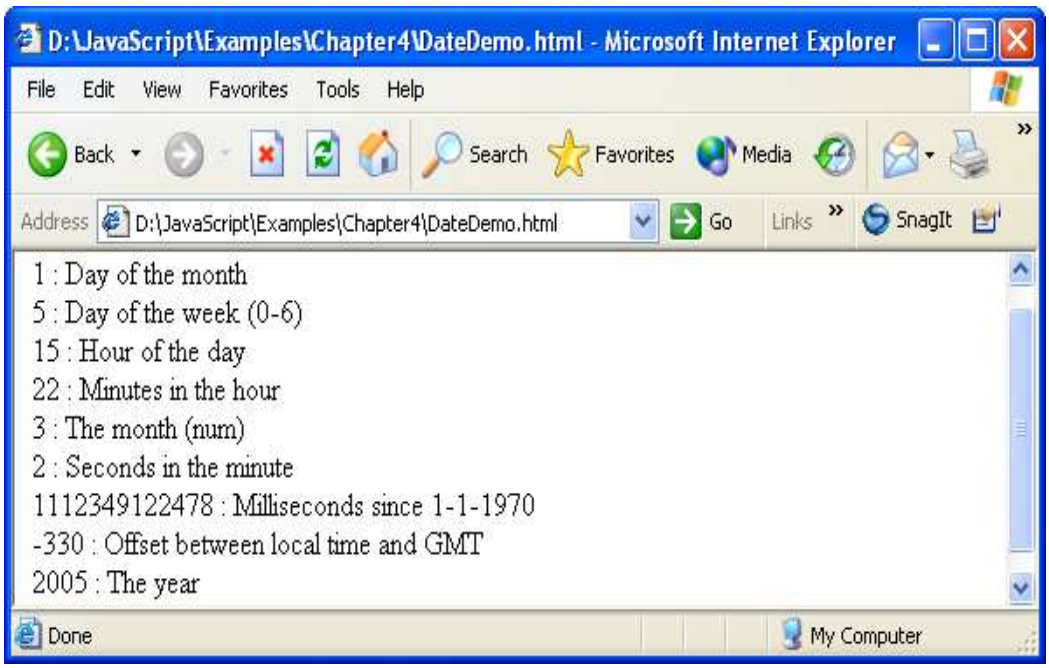
Date and time arithmetic (contd..):

```
<!DOCTYPE html>
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
<script>
dateinfo = new Date();
document.write(dateinfo.getDate() + " : Day of the month" + "<br>")

document.write(dateinfo.getDay()              +" : Day of the week  (0-6)" +
"<br>" )
document.write(dateinfo.getHours()+" : Hour of the day " + "<br>")

document.write(dateinfo.getMinutes()+" : Minutes in the hour" + "<br>")
document.write(dateinfo.getMonth()+" : The month              (num)" +
"<br>" )
document.write(dateinfo.getSeconds()+ " : Seconds in the minute " +
"<br>")
document.write(dateinfo.getTime()+" : Milliseconds since 1-1-1970" +
"<br>")
document.write(dateinfo.getTimezoneOffset() +" : Offset between local time
and GMT" + "<br>")
document.write(dateinfo.getYear() +" : The year" + "<br>")
</script>
</BODY>
</HTML>
```

And it produces the output as:



Demo

- Date_info.html
- DateDifference.html
- Utc_ex.html

Add the notes here.

Lab

- Lab 3:
  - Working with Predefined Core Objects

Add the notes here.

Summary

- Predefined objects of JavaScript like String, Math and Date
- How to use predefined objects
- How to manipulate their properties and invoke methods

Review Question

- Question 1: Which is the method to extract a single character from a string when you know the position of that character.
  - Option 1: string.charAt()
  - Option 2: string.charAtIndex()
- Question 2: getDate() returns the day within the month.
  - True/False
- Question 3: To convert a plain string to one with these escape codes, use the _____ function