2.4 OBJECTS IN JAVASCRIPT

JavaScript is an Object Oriented Programming (OOP) language. A programming language is called object-oriented if it has the following four basic capabilities: encapsulation, aggregation, inheritance and polymorphism. All the objects will have properties and methods.

Object Properties

Object properties can be any of the primitive data types or abstract data types. Object properties are usually variables that are used internally in the object's methods, but can also be globally visible variables that can be used throughout the page.

objectName.objectProperty = propertyValue;

Example: varstr = document.title;

Object Methods

The **methods** are functions that let the object do something or let something be done to it. A function is a standalone unit of statements and a method is attached to an object and can be referenced by the keyword.Methods are useful for everything from displaying the contents of the object to the screen to performing complex mathematical operations on a group of local properties and parameters.

Example: document.write("This is a method of the object document");

User-Defined Objects

Apart from built-in objects, the users can also create their own objects. All user-defined objects and built-in objects are descendants of an object called **Object**. The new operator is used to create a new object.

The new Operator

The new operator is used to create an instance of an object. To create an object, the new operator is followed by the constructor method.

Example:var books = new Array("Thirukural", " Geethai");

In the above example Array() is a built-in object. Books is the instance of the object Array().

The Object() Constructor

A **constructor** is a function that creates and initializes an object. The Object() constructor is used to build an object. The return value of the Object() constructor is assigned to a variable.The variable contains a reference to the new object. The properties assigned to the object are not variables. These properties can be accessed only through objects.

Objectname.property

Creating an object

```
<html><head><script type="text/javascript">
var book = new Object(); // Create the object
 book.name = "PonniyinSelvan"; // Assign properties to the
      object
book.author = "Kalki";
</script></head>
<body><script   type="text/javascript">
document.write("Book name is : " + book.name + "<br>");
document.write("Book author is : " + book.author + "<br>");
</script></body></html>
```

Book name is : Ponniyin Selvan
Book author is : Kalki

JavaScript Native Objects

JavaScript has several built-in or native objects. These objects are accessible anywhere in the program as the other objects. The following are some of the important JavaScript Native Objects:JavaScript Number Object, JavaScript Boolean Object, JavaScript String Object, JavaScript Array Object, JavaScript Date Object, JavaScript Math Object, JavaScript RegExp Object

2.4.1 JavaScript Number Object

The Number object represents numerical date, either integers or floating-point numbers. The browser automatically converts number literals to instances of the number class.

varval = new Number(number);

If the argument cannot be converted into a number, it returns NaN (Not-a-Number).

Number Properties

Property	Description
MAX_VALUE	The largest possible value a number in JavaScript (1.7976931348623157E+308).
MIN_VALUE	The smallest possible value a number in JavaScript can have (5E-324)
NaN	Equal to a value that is not a number.
NEGATIVE_INFINITY	A value that is less than MIN_VALUE.

POSITIVE_INFINITY	A value that is greater than MAX_VALUE
Prototype	A static property of the Number object. This is used to assign new properties and methods to the Number object in the current document.

Number Methods

Method	Description
Number()	Returns the number object.
toExponential()	Forces a number to display in exponential notation.
toFixed()	Formats a number with a specific number of digits to the right of the decimal.
toLocaleString()	Returns a string value version of the current number in a format that may vary according to a browser's locale settings.
toPrecision()	Defines how many total digits to display of a number (including digits to the before and after the decimal).
toString()	Returns the string representation of the number's value.
valueOf()	Returns the number's value.

2.4.2 JavaScript String object

varval = new String(string); // The string parameter is series of characters.

String Properties

- length -Returns the length of the string.
- Prototype-The prototype property allows you to add properties and methods to an object.

String Methods

Method	Description
String()	Returns the string object.
charAt(i)	Returns the character at the specified index.

charCodeAt(index)	Returns a number indicating the Unicode value of the character at the given index.
concat()	Combines the text of two strings and returns the combined string.
indexOf()	Returns the index of the first occurrence of the String object of or -1 if not found.
lastIndexOf()	Returns the index within the calling String object of the last occurrence of the specified value, or -1 if not found.
localeCompare()	Returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order.
match()	Used to match a regular expression against a string.
replace()	Used to find a match between a regular expression and a string, and to replace the matched substring with a new substring
search()	Executes the search for a match between a regular expression and a specified string.
slice()	Extracts a section of a string and returns a new string.
split()	Splits a String object into an array of strings by separating the string into substrings.
substr()	Returns the characters in a string beginning at the specified location through the specified number of characters.
substring()	Returns the characters in a string between two indexes into the string.
toLocaleLowerCase()	The characters within a string are converted to lower case while respecting the current locale.
toLocaleUpperCase()	The characters within a string are converted to upper case while respecting the current locale.
toLowerCase()	Returns the calling string value converted to lower case.
toString()	Returns a string representing the specified object.
toUpperCase()	Returns the calling string value converted to uppercase.
valueOf()	Returns the primitive value of the specified object.

String HTML wrappers

The functionalities of these wrappers are similar to the HTML tags.

Method	Description
anchor()	Creates an HTML anchor that is used as a hypertext target.
big()	Creates a string to be displayed in a big font as if it were in a <big> tag.
blink()	Creates a string to blink as if it were in a <blink> tag.
bold()	Creates a string to be displayed as bold as if it were in a <b> tag.
fixed()	Causes a string to be displayed in fixed-pitch font as if it were in a <tt> tag.
fontcolor()	Causes a string to be displayed in the specified color as if it were in a <font color="color"> tag.
fontsize()	Causes a string to be displayed in the specified font size as if it were in a <font size="size"> tag.
italics()	Causes a string to be italic, as if it were in an <i> tag.
link()	Creates an HTML hypertext link that requests another URL.
small()	Causes a string to be displayed in a small font, as if it were in a <small> tag.
strike()	Causes a string to be displayed as struck-out text, as if it were in a <strike> tag.
sub()	Causes a string to be displayed as a subscript, as if it were in a <sub> tag.
sup()	Causes a string to be displayed as a superscript, as if it were in a <sup> tag.

JavaScript Array object

The Array object is used store multiple values in a single variable.

Example:var fruits = new Array("apple", "orange", "mango");

The Array parameter is a list of strings or integers. The maximum length allowed for an array is 4,294,967,295.

Array Properties

- index - The property represents the zero-based index of the match in the string

- input - This property is only present in arrays created by regular expression matches.
- length - Reflects the number of elements in an array.
- Prototype - The prototype property allows you to add properties and methods to an object.

Array Methods

Method	Description
Array()	Returns a reference to the array function that created the object.
concat()	Returns a new array comprised of this array joined with other array(s) and/or value(s).
every()	Returns true if every element in this array satisfies the provided testing function.
filter()	Creates a new array with all of the elements of this array for which the provided filtering function returns true.
forEach()	Calls a function for each element in the array.
indexOf()	Returns the first (least) index of an element within the array equal to the specified value, or
join()	Joins all elements of an array into a string.
lastIndexOf()	Returns the last (greatest) index of an element within the array equal to the specified value
map()	Creates a new array with the results of calling a provided function on every element in this array.
pop()	Removes the last element from an array and returns that element.
push()	Adds one or more elements to the end of an array and returns the new length of the array.
reduce()	Apply a function simultaneously against two values of the array (from left
reduceRight()	Apply a function simultaneously against two values of the array (from right
reverse()	Reverses the order of the elements of an array

shift()	Removes the first element from an array and returns that element.
slice()	Extracts a section of an array and returns a new array.
some()	Returns true if at least one element in this array satisfies the provided testing function.
toSource()	Represents the source code of an object
sort()	Sorts the elements of an array.
splice()	Adds and/or removes elements from an array.
toString()	Returns a string representing the array and its elements.
unshift()	Adds one or more elements to the front of an array and returns the new length of the array.

JavaScript Math Object

The math object provides the properties and methods for mathematical constants and functions. Unlike the other global objects, Math is not a constructor. All properties and methods of Math are static and can be called by using Math as an object without creating it.

varpi_val = Math.PI; //Math object need not be created

varsine_val = Math.sin(30);

Math Properties

Property	Description
E	Euler's constant and the base of natural logarithms, approximately 2.718.
LN2	Natural logarithm of 2, approximately 0.693.
LN10	Natural logarithm of 10, approximately 2.302.
LOG2E	Base 2 logarithm of E, approximately 1.442.
LOG10E	Base 10 logarithm of E, approximately 0.434.
PI	Ratio of the circumference of a circle to its diameter, approximately 3.14159.
SQRT1_2	Square root of 1/2; equivalently, 1 over the square root of 2, approximately 0.707.
SQRT2	Square root of 2, approximately 1.414.

Math Methods

Method	Description
abs()	Returns the absolute value of a number.
acos()	Returns the arccosine (in radians) of a number.
asin()	Returns the arcsine (in radians) of a number.
atan()	Returns the arctangent (in radians) of a number.
atan2()	Returns the arctangent of the quotient of its arguments.
ceil()	Returns the smallest integer greater than or equal to a number.
cos()	Returns the cosine of a number.
exp()	Returns EN, where N is the argument, and E is Euler's constant,the base of the natural logarithm.
floor()	Returns the largest integer less than or equal to a number.
log()	Returns the natural logarithm (base E) of a number.
max()	Returns the largest of zero or more numbers.
abs()	Returns the absolute value of a number.
min()	Returns the smallest of zero or more numbers.
pow()	Returns base to the exponent power, that is, base exponent.
random()	Returns a pseudo
round()	Returns the value of a number rounded to the nearest integer.
sin()	Returns the sine of a number.
sqrt()	Returns the square root of a number.
tan()	Returns the tangent of a number.
toSource()	Returns the string "Math".

Regular expressions and regular objects

A regular expression is an object that describes a pattern of characters. The JavaScript RegExp class represents regular expressions, and both String and RegExp define methods that

use regular expressions to perform powerful pattern-matching and search-and-replace functions on text.

```
var pattern = new RegExp(pattern, attributes);

var pattern = /pattern/attributes;
```

- **pattern**: A string that specifies the pattern of the regular expression or another regular expression.
- **attributes:** An optional string containing any of the "g", "i", and "m" attributes that specify global, case-insensitive, and multiline matches, respectively.

Brackets:

Brackets ([]) have a special meaning when used in the context of regular expressions. They are used to find a range of characters.

- [...] - Any one character between the brackets.
- [^...] - Any one character not between the brackets.
- [0-9] - It matches any decimal digit from 0 through 9.
- [a-z] - It matches any character from lowercase a through lowercase z.
- [A-Z] - It matches any character from uppercase A through uppercase Z.
- [a-Z] - It matches any character from lowercase a through uppercase Z.

Quantifiers:

The frequency or position of bracketed character sequences and single characters can be denoted by a special character. Each special character have a specific connotation. The +, *, ?, and $ flags all follow a character sequence.

- p+ -It matches any string containing at least one p.
- p*-It matches any string containing zero or more p's.
- p?-It matches any string containing one or more p's.
- p{N} -It matches any string containing a sequence of N p's
- p{2,3}-It matches any string containing a sequence of two or three p's.
- p{2, } -It matches any string containing a sequence of at least two p's.
- p$-It matches any string with p at the end of it.
- ^p-It matches any string with p at the beginning of it.

RegExp Properties

Property	Description
Global	Specifies if the "g" modifier is set.
ignoreCase	Specifies if the "i" modifier is set.
lastIndex	The index at which to start the next match.
Multiline	Specifies if the "m" modifier is set.
Source	The text of the pattern.
Global	Specifies if the "g" modifier is set.

RegExp Methods

Method	Description
exec()	Executes a search for a match in its string parameter.
test()	Tests for a match in its string parameter.
toSource()	Returns an object literal representing the specified object; you can use this value to create a new object.
toString()	Returns a string representing the specified object.

JavaScript Date Object

The Date object is a data type built into the JavaScript language. Once a Date object is created, a number of methods are available to operate onit.

- new Date( ) - This constructor creates a Date object set to the current date and time.
- new Date(milliseconds)- When one numeric argument is passed, it is taken as the internal numeric representation of the date in milliseconds, as returned bythe getTime( ) method. For example, passing the argument 5000 creates a date that represents five seconds past midnight on 1/1/70
- new Date(datestring) - When one string argument is passed, it is a string representation of a date, in the format accepted by the Date.parse( ) method.
- new Date(year,month,date[,hour,minute,second,millisecond]) -The parameters on square brackets are optional . The description of the arguments are listed below:

1. year: Integer value representing the year. For compatibility (in order to avoid the Y2K problem), you should always specify the year in full; use 1998, rather than 98.
2. month: Integer value representing the month, beginning with 0 for January to 11 for December.
3. date: Integer value representing the day of the month.
4. hour: Integer value representing the hour of the day (24-hour scale).
5. minute: Integer value representing the minute segment of a time reading.
6. second: Integer value representing the second segment of a time reading.
7. millisecond: Integer value representing the millisecond segment of a time reading.

Date Properties:

- constructor- Specifies the function that creates an object's prototype.
- Prototype- The prototype property allows you to add properties and methods to an object

Date Methods:

Method	Description
Date()	Returns today's date and time
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.
etMilliseconds()	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 of the specified date as the number of milliseconds since January 1, 1970, 00:00:00 UTC.
Date()	Returns today's date and time
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.
getTimezoneOffset()	Returns the time
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 in the specified date according to universal time.
getUTCSeconds()	Returns the seconds in the specified date according to universal time.
getYear() Deprecated	Returns the year in the specified date according to local time.
getTimezoneOffset()	Returns the time

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.
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 Date object to the time represented by a number of milliseconds since January 1, 1970, 00:00:00 UTC.
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.
setUTCDate()	Sets the day of the month for a specified date according to universal time. setUTCFullYear()
toLocaleDateString()	Returns the "date" portion of the Date as a string, using the current locale's conventions.
toLocaleFormat()	Converts a date to a string, using a format string.
toLocaleString()	Converts a date to 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 a string representing the source for an equivalent Date object; you can use this value to create a new object.
toString()	Returns a string representing the specified Date object.
toTimeString()	Returns the "time" portion of the Date as a human
toUTCString()	Converts a date to a string, using the universal time convention.
valueOf()	Returns the primitive value of a Date object.
setUTCDate()	Sets the day of the month for a specified date according to universal time. setUTCFullYear()
toLocaleDateString()	Returns the "date" portion of the Date as a string, using the current locale's conventions.
toLocaleFormat()	Converts a date to a string, using a format string.
toLocaleString()	Converts a date to 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 a string representing the source for an equivalent Date object; you can use this value to create a new object.

Date objects

```
<html><body><script type="text/javascript">
var d = new Date()
document.write(d.getDate())document.write(".")
document.write(d.getMonth() + 1)document.write(".")
document.write(d.getFullYear())d.setFullYear("1990")document.write(".")
document.write(d.getUTCHours())document.write(".")
document.write(d.getUTCMinutes() + 1)document.write(".")
document.write(d.getUTCSeconds())
varweekday=new
```

```
Array("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")
document.write("Today is " + weekday[d.getDay()])
varweekday=new
Array("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")
varmonthname=new
Array("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec")
document.write(weekday[d.getDay()] + " ")
document.write(d.getDate() + ". ")
document.write(monthname[d.getMonth()] +" ")
document.write(d.getFullYear())
</body></html>
```

22.12.2014.10.9.11Today is SaturdaySaturday 22. Dec 1990