Introduction to Java Script
Object Oriented

By

Vaibhav P. Vasani

Assistant Professor

Department of Computer Engineering

K. J. Somaiya College of Engineering

Somaiya Vidyavihar University

vaibhav.vasani@gmail.com

Object Oriented Programming (OOP) language

- H
- E
- M
- A
- C
- P
- T

Class ?

Object ?

Polymorphism ?

Object Properties

- Object properties are usually variables that are used internally in the object's methods, but can also be globally visible variables that are used throughout the page.

Syntax and Example

- syntax
- objectName.objectProperty = propertyValue;

- **example**
- var str = document.title;

Object Methods

- document.write(“Welcome to MERN lab");
-

User-Defined Objects

- All user-defined objects and built-in objects are descendants of an object called **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.

```
var employee = new Object();
var names= new Array(“Krishna", “Mohan", “Madhusudhan");
var bday = new Date(“November 08, 1987");
```

The Object() Constructor

- A constructor is a function that creates and initializes an object.
- JavaScript provides a special constructor function called **Object()** to build the object.
- The return value of the **Object()** constructor is assigned to a variable.

Example#01

```
<html>
  <head>
    <title>User-defined objects</title>
    <script type = "text/javascript">
      var book = new Object();   // Create the object
      book.subject = “MERN”;     // Assign properties to the object
      book.author  = “SCP”;
    </script>
  </head>
  <body>
    <script type = "text/javascript">
      document.write("Book name is : " + book.subject + "<br>");
      document.write("Book author is : " + book.author + "<br>");
    </script>
  </body>
</html>
```

Output

?

Example#2

```
html>
   <head>
   <title>User-defined objects</title>
      <script type = "text/javascript">
         function book(title, author) {
            this.title = title;
            this.author  = author;
         }
      </script>
   </head>
   <body>
      <script type = "text/javascript">
         var myBook = new book(“Cryptography", “MMP”);
         document.write("Book title is : " + myBook.title + "<br>");
         document.write("Book author is : " + myBook.author + "<br>");
      </script>
   </body>
</html>
```

Output

?

Defining Methods for an Object

16_Methods_for_an_Object.html

The 'with' Keyword

- The **‘with’** keyword is used as a kind of shorthand for referencing an object's properties or methods.
- The object specified as an argument to **with** becomes the default object for the duration of the block that follows.
- The properties and methods for the object can be used without naming the object.

Syntax

- with (object) { properties used without the object name and dot }

17_with_keyword.html

JavaScript Native Objects

- JavaScript has several built-in or native objects.
- These objects are accessible anywhere in your program and will work the same way in any browser running in any operating system.

The 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.

- Syntax
- var val = new Number(number);

- Students will practice on same

Number Properties

Sr.No.	Property & Description
1	MAX_VALUEThe largest possible value a number in JavaScript can have 1.7976931348623157E+308
2	MIN_VALUEThe smallest possible value a number in JavaScript can have 5E-324
3	NaNEqual to a value that is not a number.
4	NEGATIVE_INFINITYA value that is less than MIN_VALUE.
5	POSITIVE_INFINITYA value that is greater than MAX_VALUE
6	prototypeA static property of the Number object. Use the prototype property to assign new properties and methods to the Number object in the current document
7	constructorReturns the function that created this object's instance. By default this is the Number object.

```
<html>
   <head>
      <script type = "text/javascript">
         <!--
            function showValue() {
               var val = Number.MAX_VALUE;
               document.write ("Value of Number.MAX_VALUE : " + val );
            }
         //-->
      </script>
   </head>
   <body>
      <p>Click the following to see the result:</p>
      <form>
         <input type = "button" value = "Click Me" onclick = "showValue();" />
      </form>
   </body>
</html>
```

Output

```
Value of Number.MAX_VALUE : 1.7976931348623157e+308
```

Number Methods

Sr.No.	Method & Description
1	toExponential()Forces a number to display in exponential notation, even if the number is in the range in which JavaScript normally uses standard notation.
2	toFixed()Formats a number with a specific number of digits to the right of the decimal.
3	toLocaleString()Returns a string value version of the current number in a format that may vary according to a browser's local settings.
4	toPrecision()Defines how many total digits (including digits to the left and right of the decimal) to display of a number.
5	toString()Returns the string representation of the number's value.
6	valueOf()Returns the number's value.

- <html>
- <head>
- <title>JavaScript toPrecision() Method </title>
- </head>
- <body>
- <script type = "text/javascript">
- var num = new Number(7.123456);
- document.write("num.toPrecision() is " + num.toPrecision());
- document.write("<br />");
- document.write("num.toPrecision(4) is " + num.toPrecision(4));
- document.write("<br />");
- document.write("num.toPrecision(2) is " + num.toPrecision(2));
- document.write("<br />");
- document.write("num.toPrecision(1) is " + num.toPrecision(1));
- </script>
- </body>
- </html>

```
num.toPrecision() is 7.123456
num.toPrecision(4) is 7.123
num.toPrecision(2) is 7.1
num.toPrecision(1) is 7
```

The Boolean Object

- The **Boolean** object represents two values, either "true" or "false".
- If *value* parameter is omitted or is 0, -0, null, false, **NaN,** undefined, or the empty string (""), the object has an initial value of false.
- var val = new Boolean(value);

Students will practice on same

Boolean Properties

Sr.No.	Property & Description
1	Constructor Returns a reference to the Boolean function that created the object.
2	prototypeThe prototype property allows you to add properties and methods to an object.

Boolean Properties: Constructor

Return Value
Returns the function that created this object's instance.

```
• <html>
•    <head>
•       <title>JavaScript constructor() Method</title>
•    </head>
•
•    <body>
•       <script type = "text/javascript">
•          var bool = new Boolean( );
•          document.write("bool.constructor() is:"+bool.constructor);
•       </script>
•    </body>
• </html>
```

Output

```
bool.constructor() is : function Boolean() {
[native code] }
```

Boolean Methods

Sr.No.	Method & Description
1	toSource()Returns a string containing the source of the Boolean object; you can use this string to create an equivalent object.
2	toString()Returns a string of either "true" or "false" depending upon the value of the object.
3	valueOf()Returns the primitive value of the Boolean object.

The Strings Object

- The **String** object lets you work with a series of characters
- As JavaScript automatically converts between string primitives and String objects

- var val = new String(string);

- WAP to find length of string
- Use following Javascript String object methods
- Split()
- substr()
- toLocaleLowerCase()

String Properties

Sr.No.	Property & Description
1	constructorReturns a reference to the String function that created the object.
2	lengthReturns the length of the string.
3	prototypeThe prototype property allows you to add properties and methods to an object.

String Methods

Sr.No.	Method & Description
1	charAt()Returns the character at the specified index.
2	charCodeAt()Returns a number indicating the Unicode value of the character at the given index.
3	concat()Combines the text of two strings and returns a new string.
4	indexOf()Returns the index within the calling String object of the first occurrence of the specified value, or -1 if not found.
5	lastIndexOf()Returns the index within the calling String object of the last occurrence of the specified value, or -1 if not found.
6	localeCompare()Returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order.
7	match()Used to match a regular expression against a string.
8	replace()Used to find a match between a regular expression and a string, and to replace the matched substring with a new substring.
9	search()Executes the search for a match between a regular expression and a specified string.
10	slice()Extracts a section of a string and returns a new string.
11	split()Splits a String object into an array of strings by separating the string into substrings.
12	substr()Returns the characters in a string beginning at the specified location through the specified number of characters.
13	substring()Returns the characters in a string between two indexes into the string.

String HTML Wrappers

Sr.No.	Method & Description
1	anchor()Creates an HTML anchor that is used as a hypertext target.
2	big()Creates a string to be displayed in a big font as if it were in a <big> tag.
3	blink()Creates a string to blink as if it were in a <blink> tag.
4	bold()Creates a string to be displayed as bold as if it were in a <b> tag.
5	fixed()Causes a string to be displayed in fixed-pitch font as if it were in a <tt> tag
6	fontcolor() Causes a string to be displayed in the specified color as if it were in a <font color="color"> tag.
7	fontsize()Causes a string to be displayed in the specified font size as if it were in a <font size="size"> tag.
8	italics()Causes a string to be italic, as if it were in an <i> tag.
9	link()Creates an HTML hypertext link that requests another URL.
10	small()Causes a string to be displayed in a small font, as if it were in a <small> tag.
11	strike()Causes a string to be displayed as struck-out text, as if it were in a <strike> tag.
12	sub()Causes a string to be displayed as a subscript, as if it were in a <sub> tag
13	sup()Causes a string to be displayed as a superscript, as if it were in a <sup> tag

String - anchor() Method

```
<html>
  <head>
    <title>JavaScript String anchor() Method</title>
  </head>

  <body>
    <script type = "text/javascript">
      var str = new String("Hello world");
      alert(str.anchor( "myanchor" ));
    </script>
  </body>
</html>
```

Output

```
<a name = "myanchor">Hello world</a>
```

- Practice program

The Arrays Object

- The **Array** object lets you store multiple values in a single variable.
- It stores a fixed-size sequential collection of elements of the same type.
- An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

```
var FSD = new Array( “MERN", “MEAN", “Python“, “JAVA” );
```

Array Properties

Property & Description
Constructor : Returns a reference to the array function that created the object.
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

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 -1 if none is found.

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, or -1 if none is found.

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-to-right) as to reduce it to a single value.

reduceRight(): Apply a function simultaneously against two values of the array (from right-to-left) as to reduce it to a single value.

reverse(): Reverses the order of the elements of an array -- the first becomes the last, and the last becomes the first.

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.

The Date Object

- a built datatype
- reated with the **new Date( )**
- Syntax

- new Date( ) new Date(milliseconds)
- new Date(datestring)
- New Date(year,month,date[,hour,minute,second,millisecond])

- **No Argument** − With no arguments, the Date() constructor creates a Date object set to the current date and time.
- **milliseconds** − When one numeric argument is passed, it is taken as the internal numeric representation of the date in milliseconds, as returned by the getTime() method. For example, passing the argument 5000 creates a date that represents five seconds past midnight on 1/1/70.
- **datestring** − When one string argument is passed, it is a string representation of a date, in the format accepted by the **Date.parse()** method.
- **7 agruments** − To use the last form of the constructor shown above. Here is a description of each argument −
    - **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.
    - **month** − Integer value representing the month, beginning with 0 for January to 11 for December.
    - **date** − Integer value representing the day of the month.
    - **hour** − Integer value representing the hour of the day (24-hour scale).
    - **minute** − Integer value representing the minute segment of a time reading.
    - **second** − Integer value representing the second segment of a time reading.
    - **millisecond** − Integer value representing the millisecond segment of a time reading.

Date Properties

Sr.No.	Method & Description
1	Date()Returns today's date and time
2	getDate()Returns the day of the month for the specified date according to local time.
3	getDay()Returns the day of the week for the specified date according to local time.
4	getFullYear()Returns the year of the specified date according to local time.
5	getHours()Returns the hour in the specified date according to local time.
6	getMilliseconds()Returns the milliseconds in the specified date according to local time.
7	getMinutes()Returns the minutes in the specified date according to local time.
8	getMonth()Returns the month in the specified date according to local time.
9	getSeconds()Returns the seconds in the specified date according to local

Date Static Methods

Sr.No.	Method & Description
1	Date.parse( )Parses a string representation of a date and time and returns the internal millisecond representation of that date.
2	Date.UTC( )Returns the millisecond representation of the specified UTC date and time.

The Math Object

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

- var pi_val = Math.PI;
- var sine_val = Math.sin(30);

Math Properties

Sr.No.	Property & Description
1	E \Euler's constant and the base of natural logarithms, approximately 2.718.
2	LN2Natural logarithm of 2, approximately 0.693.
3	LN10Natural logarithm of 10, approximately 2.302.
4	LOG2EBase 2 logarithm of E, approximately 1.442.
5	LOG10EBase 10 logarithm of E, approximately 0.434.
6	PIRatio of the circumference of a circle to its diameter, approximately 3.14159.
7	SQRT1_2Square root of 1/2; equivalently, 1 over the square root of 2, approximately 0.707.
8	SQRT2Square root of 2, approximately 1.414.

Math Methods

1	**abs()Returns the absolute value of a number.**
2	acos()Returns the arccosine (in radians) of a number.
3	asin()Returns the arcsine (in radians) of a number.
4	atan()Returns the arctangent (in radians) of a number.
5	atan2()Returns the arctangent of the quotient of its arguments.
6	ceil()Returns the smallest integer greater than or equal to a number.
7	cos()Returns the cosine of a number.
8	exp()Returns E^N, where N is the argument, and E is Euler's constant, the base of the natural logarithm.
9	floor()Returns the largest integer less than or equal to a number.
10	log()Returns the natural logarithm (base E) of a number.
11	max()Returns the largest of zero or more numbers.
12	min()Returns the smallest of zero or more numbers.
13	pow()Returns base to the exponent power, that is, base exponent.
14	random()Returns a pseudo-random number between 0 and 1.

Regular Expressions and RegExp Object

- A regular expression is an object that describes a pattern of characters.

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

  or simply

- 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 multi-line 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.

Sr.No.	Expression & Description
1	**[...]** Any one character between the brackets.
2	**[^...]** Any one character not between the brackets.
3	**[0-9]** It matches any decimal digit from 0 through 9.
4	**[a-z]** It matches any character from lowercase **a** through lowercase **z**.
5	**[A-Z]** It matches any character from uppercase **A** through uppercase **Z**.
6	**[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 has a specific connotation. The +, *, ?, and $ flags all follow a character sequence.

Sr.No.	Expression & Description
1	**p+** It matches any string containing one or more p's.
2	**p*** It matches any string containing zero or more p's.
3	**p?** It matches any string containing at most one p.
4	**p{N}** It matches any string containing a sequence of **N** p's
5	**p{2,3}** It matches any string containing a sequence of two or three p's.
6	**p{2, }** It matches any string containing a sequence of at least two p's.
7	**p$** It matches any string with p at the end of it.
8	**^p** It matches any string with p at the beginning of it.

Examples##

- [^a-zA-Z]
- It matches any string not containing any of the characters ranging from a through z and A through Z.
- p.p
- It matches any string containing p, followed by any character, in turn followed by another p.
- ^.{2}$
- It matches any string containing exactly two characters.
- <b>(.*)</b>
- It matches any string enclosed within <b> and </b>.
- p(hp)*
- It matches any string containing a p followed by zero or more instances of the sequence hp.

Literal characters

Sr. No.	Character & Description
1	**Alphanumeric :** Itself
2	**\0:** The NUL character (\u0000)
3	**\t:** Tab (\u0009
4	**\n:** Newline (\u000A)
5	**\v:** Vertical tab (\u000B)
6	**\f** Form feed (\u000C)
7	**\r** Carriage return (\u000D)
8	**\xnn** The Latin character specified by the hexadecimal number nn; for example, \x0A is the same as \n
9	**\uxxxx** The Unicode character specified by the hexadecimal number xxxx; for example, \u0009 is the same as \t
10	**\cX** The control character ^X; for example, \cJ is equivalent to the newline character \n

Metacharacters

Sr.No.	Character & Description
1	**.** a single character
2	**\s** a whitespace character (space, tab, newline)
3	**\S** non-whitespace character
4	**\d** a digit (0-9)
5	**\D** a non-digit
6	**\w** a word character (a-z, A-Z, 0-9, _)
7	**\W** a non-word character
8	**[\b]** a literal backspace (special case).
9	**[aeiou]** matches a single character in the given set
10	**[^aeiou]** matches a single character outside the given set
11	**(foo|bar|baz)** matches any of the alternatives specified

For instance, you can search for a large sum of money using the '\d' metacharacter: **/([\d]+)000 /**, Here **\d** will search for any string of numerical character.

Modifiers

Sr.No.	Modifier & Description
1	**i** Perform case-insensitive matching.
2	**m** Specifies that if the string has newline or carriage return characters, the ^ and $ operators will now match against a newline boundary, instead of a string boundary
3	**g** Performs a global matchthat is, find all matches rather than stopping after the first match.

RegExp Properties

Sr.No.	Property & Description
1	constructorSpecifies the function that creates an object's prototype.
2	globalSpecifies if the "g" modifier is set.
3	ignoreCaseSpecifies if the "i" modifier is set.
4	lastIndexThe index at which to start the next match.
5	multilineSpecifies if the "m" modifier is set.
6	Source The text of the pattern.

RegExp Methods

Sr.No.	Method & Description
1	exec()Executes a search for a match in its string parameter.
2	test()Tests for a match in its string parameter.
3	toSource()Returns an object literal representing the specified object; you can use this value to create a new object.
4	toString() Returns a string representing the specified object.

Thank You