What is JavaScript?

JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.

The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Name some of the JavaScript features.

Following are the features of JavaScript −

- JavaScript is a lightweight, interpreted programming language.
- JavaScript is designed for creating network-centric applications.
- JavaScript is complementary to and integrated with Java.
- JavaScript is is complementary to and integrated with HTML.
- JavaScript is open and cross-platform.

What are the advantages of using JavaScript?

Following are the advantages of using JavaScript −

- **Less server interaction −** You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- **Immediate feedback to the visitors −** They don't have to wait for a page reload to see if they have forgotten to enter something.
- **Increased interactivity −** You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- **Richer interfaces −** You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

What are disadvantages of using JavaScript?

We can not treat JavaScript as a full fledged programming language. It lacks the following important features −

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript can not be used for Networking applications because there is no such support available.
- JavaScript doesn't have any multithreading or multiprocess capabilities.

Is JavaScript a case-sensitive language?

Yes! JavaScript is a case-sensitive language. This means that language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

How can you create an Object in JavaScript?

JavaScript supports Object concept very well. You can create an object using the object literal as follows −

```
var emp = {
   name: "Zara",
   age: 10
};
```

How can you read properties of an Object in JavaScript?

You can write and read properties of an object using the dot notation as follows −

```
// Getting object properties
emp.name  // ==> Zara
emp.age   // ==> 10
// Setting object properties
emp.name = "Daisy"  // <== Daisy
emp.age  =  20      // <== 20
```

How can you create an Array in JavaScript?

You can define arrays using the array literal as follows −

```
var x = [];
var y = [1, 2, 3, 4, 5];
```

How to read elements of an array in JavaScript?

An array has a length property that is useful for iteration. We can read elements of an array as follows −

```
var x = [1, 2, 3, 4, 5];
for (var i = 0; i < x.length; i++) {
   // Do something with x[i]
}
```

What is a named function in JavaScript? How to define a named function?

A named function has a name when it is defined. A named function can be defined using function keyword as follows −

```
function named(){
   // do some stuff here
}
```

How many types of functions JavaScript supports?

A function in JavaScript can be either named or anonymous.

How to define a anonymous function?

An anonymous function can be defined in similar way as a normal function but it would not have any name.

Can you assign a anonymous function to a variable?

Yes! An anonymous function can be assigned to a variable.

Can you pass a anonymous function as an argument to another function?

Yes! An anonymous function can be passed as an argument to another function.

What is arguments object in JavaScript?

JavaScript variable arguments represents the arguments passed to a function.

How can you get the type of arguments passed to a function?

Using typeof operator, we can get the type of arguments passed to a function. For example −

```
function func(x){
   console.log(typeof x, arguments.length);
}
func();                    //==> "undefined", 0
func(1);                   //==> "number", 1
func("1", "2", "3");       //==> "string", 3
```

How can you get the total number of arguments passed to a function?

Using arguments.length property, we can get the total number of arguments passed to a function. For example −

```
function func(x){
   console.log(typeof x, arguments.length);
}
func();                    //==> "undefined", 0
func(1);                   //==> "number", 1
func("1", "2", "3");       //==> "string", 3
```

How can you get the reference of a caller function inside a function?

The arguments object has a callee property, which refers to the function you're inside of. For example −

```
function func() {
   return arguments.callee;
}
func();                    // ==> func
```

What is the purpose of 'this' operator in JavaScript?

JavaScript famous keyword this always refers to the current context.

What are the valid scopes of a variable in JavaScript?

The scope of a variable is the region of your program in which it is defined. JavaScript variable will have only two scopes.

- **Global Variables −** A global variable has global scope which means it is visible everywhere in your JavaScript code.
- **Local Variables −** A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

Which type of variable among global and local, takes precedence over other if names are same?

A local variable takes precedence over a global variable with the same name.

What is callback?

A callback is a plain JavaScript function passed to some method as an argument or option. Some callbacks are just events, called to give the user a chance to react when a certain state is triggered.

What is closure?

Closures are created whenever a variable that is defined outside the current scope is accessed from within some inner scope.

Give an example of closure?

Following example shows how the variable counter is visible within the create, increment, and print functions, but not outside of them −

```
function create() {
   var counter = 0;
   return {
      increment: function() {
         counter++;
      },

      print: function() {
         console.log(counter);
      }
   }
}
var c = create();
c.increment();
c.print();     // ==> 1
```

Which built-in method returns the character at the specified index?

charAt() method returns the character at the specified index.

Which built-in method combines the text of two strings and returns a new string?

concat() method returns the character at the specified index.

Which built-in method calls a function for each element in the array?

forEach() method calls a function for each element in the array.

Which built-in method returns the index within the calling String object of the first occurrence of the specified value?

indexOf() method returns the index within the calling String object of the first occurrence of the specified value, or −1 if not found.

Which built-in method returns the length of the string?

length() method returns the length of the string.

Which built-in method removes the last element from an array and returns that element?

pop() method removes the last element from an array and returns that element.

Which built-in method adds one or more elements to the end of an array and returns the new length of the array?

push() method adds one or more elements to the end of an array and returns the new length of the array.

Which built-in method reverses the order of the elements of an array?

reverse() method reverses the order of the elements of an array −− the first becomes the last, and the last becomes the first.

Which built-in method sorts the elements of an array?

sort() method sorts the elements of an array.

Which built-in method returns the characters in a string beginning at the specified location?

substr() method returns the characters in a string beginning at the specified location through the specified number of characters.

Which built-in method returns the calling string value converted to lower case?

toLowerCase() method returns the calling string value converted to lower case.

Which built-in method returns the calling string value converted to upper case?

toUpperCase() method returns the calling string value converted to upper case.

Which built-in method returns the string representation of the number's value?

toString() method returns the string representation of the number's value.

What are the variable naming conventions in JavaScript?

While naming your variables in JavaScript keep following rules in mind.

You should not use any of the JavaScript reserved keyword as variable name. These keywords are mentioned in the next section. For example, break or boolean variable names are not valid.

JavaScript variable names should not start with a numeral (0-9). They must begin with a letter or the underscore character. For example, 123test is an invalid variable name but _123test is a valid one.

JavaScript variable names are case sensitive. For example, Name and name are two different variables.

How typeof operator works?

The typeof is a unary operator that is placed before its single operand, which can be of any type. Its value is a string indicating the data type of the operand.

The typeof operator evaluates to "number", "string", or "boolean" if its operand is a number, string, or boolean value and returns true or false based on the evaluation.

What typeof returns for a null value?

It returns "object".

Can you access Cookie using javascript?

JavaScript can also manipulate cookies using the cookie property of the Document object. JavaScript can read, create, modify, and delete the cookie or cookies that apply to the current web page.

How to create a Cookie using JavaScript?

The simplest way to create a cookie is to assign a string value to the document.cookie object, which looks like this −

Syntax −

```
document.cookie = "key1 = value1; key2 = value2; expires = date";
```

Here expires attribute is option. If you provide this attribute with a valid date or time then cookie will expire at the given date or time and after that cookies' value will not be accessible.

How to read a Cookie using JavaScript?

Reading a cookie is just as simple as writing one, because the value of the document.cookie object is the cookie. So you can use this string whenever you want to access the cookie.

The document.cookie string will keep a list of name = value pairs separated by semicolons, where name is the name of a cookie and value is its string value.

You can use strings' split() function to break the string into key and values.

How to delete a Cookie using JavaScript?

Sometimes you will want to delete a cookie so that subsequent attempts to read the cookie return nothing. To do this, you just need to set the expiration date to a time in the past.

How to redirect a url using JavaScript?

his is very simple to do a page redirect using JavaScript at client side. To redirect your site visitors to a new page, you just need to add a line in your head section as follows −

```
<head>
<script type="text/javascript">
<!--
   window.location="http://www.newlocation.com";
//-->
</script>
</head>
```

How to print a web page using javascript?

JavaScript helps you to implement this functionality using print function of window object. The JavaScript print function window.print() will print the current web page when executed.

What is Date object in JavaScript?

The Date object is a datatype built into the JavaScript language. Date objects are created with the new Date( ).

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

What is Number object in JavaScript?

he Number object represents numerical date, either integers or floating-point numbers. In general, you do not need to worry about Number objects because the browser automatically converts number literals to instances of the number class.

Syntax −

Creating a number object −

```
var val = new Number(number);
```

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

How to handle exceptions in JavaScript?

The latest versions of JavaScript added exception handling capabilities. JavaScript implements the try...catch...finally construct as well as the throw operator to handle exceptions.

You can catch programmer-generated and runtime exceptions, but you cannot catch JavaScript syntax errors.

What is purpose of onError event handler in JavaScript?

The onerror event handler was the first feature to facilitate error handling for JavaScript. The error event is fired on the window object whenever an exception occurs on the page.

The onerror event handler provides three pieces of information to identify the exact nature of the error −

- **Error message −** The same message that the browser would display for the given error.
- **URL −** The file in which the error occurred.
- **Line number −** The line number in the given URL that caused the error.