JAVASCRIPT

NOTES

JAVA SCRIPT

--History of Java Script

The early to mid-1990s was an important time for the internet.

Key players like Netscape and Microsoft were in the midst of browser wars,with Netscape's Navigator and Microsoft's Internet Explorer going head to head.

In September 1995, a Netscape programmer named Brandan Eich developed a new scripting language in just 10 days. It was originally named Mocha, but quickly became known as LiveScript and, later, JavaScript.

--Why Java script

It's a most popular programming language.

It has been ranked the most commonly used programming language.

JavaScript is the programming language of the Web.

JavaScript is easy to learn.

--What Is JavaScript?

JavaScript is a scripting language that is one of the three core languages used to develop websites.

JavaScript lets you add functionality and behaviors to your website, allowing your website’s visitors to interact with content in many imaginative ways.

JavaScript is primarily a client-side language, meaning it runs on your computer within your browser.

However, more recently the introduction of Node.js has allowed JavaScript to also execute code on servers.

section 1 - fundamentals of programming language

1. Introduction of js
2. variable and values
3. keywords in js
4. identifiers
5. data types in js
6. operators
7. flow control statements
   a) decision making statements
   if
   if else
   if elseif else
   switch
   b) looping statements
   for
   while
   do while
8. functions
   normal function

section 2 --- ECMA (es6)

1. var & let & const
2. Array
   what
   how to declare
   how to use
   problems
   array built in methods
   for in loop and for of loop
3. string
   what
   how to declare string
   how to use string
   problems
   string built in methods
4. functions
   normal function
   anonymus function
   arrow function

OOPS - object oriented programming system

1. object

what

how to create

2. Math objects
3. Date objects
4. class

   how to declare class

   how to use class

   how to create object for class

5. inheritance
6. encapsulation
7. Hoisting in javascript

DOM in Java Script

Keywords

Keywords are those reserved words which will have a pre-defined meaning defined in the programming language.

Keyword should be written only in lowercase.

--values and variables

we store the value of the program inside one variables

variables are used to store the data.

syntax : var variable_name = "value";

--Naming Variables: Rules and Best Practices

The first character must be a letter or an underscore (_) or an dollar($).

You can't use a number as the first character.

The rest of the variable name can include any letter, any number, or the underscore or dollor. Can't use any other characters, including spaces.

Variable names are case sensitive.

No limit to the length of the variable name.

You can't use one of JavaScript's reserved words as a variable name.

ex: var name, var _name , var $namE,n420,n_name,n$name

--Operators

:are used to do operations on operands

1) arithematic operaotor

addition (+)

subtraction (-)

multiplication (*)

division (/)

modulus (%)

exponential (**)

increment (++)

decrement (--)

2) assignment Operators

assigning (=)

addition assigning (+=) a=a+b or a+=b

subtraction assigning (-=) a=a-b or a-=b

multiplication assigning (*=) a=a*b or a*=b

division assigning (/=) a=a/b or a/=b

modulus assigning (%=) a=a%b or a%=b

exponential assigning (**=) a=a**b or a**=b

3) comparioson Operators

equals to (==) compare only value not the datatypes

triple equals (===) compare value as well as the datatypes

not equals (!=) compare only value not the datatypes

double not equals (!==) compare value as well as the datatypes

greater than (>)

greater or equals (>=)

lesser than (<)

lesser or equals (<=)

4) logical Operators

and Operator (&&)

or operator (||)

--> Control statement & loops

Control statements are used to control the execution flow some line of codes in the program.

Here we have 2 types of control statements.

- Decision making statements
- Looping statements

--> Decision making statements are the statements used to execute the set of lines based upon some conditions

Types :

If

If and else

If ,else if and else

switch

1) if(condition)

{

body

}

--> if body will be executed whenever the condition is true.

2) if (condition)

{

body

}

else

{

body

}

--> if body will be executed whenever the condition is true

or"else" body will executed directly

3) if (condition)

{

body

}

else if(condition)

{

body

}

else

{

}

--> if body will be executed whenever the condition is true or

it will reach to "else if" and check the condition and if it is true

"else if" body will executed or "else" body will execute directly

-> Loops in javascript

Looping statements are the statements used to execute the set of lines repeatedly.

Types :

For

While

Do while

For in

For of

For loop : Whenever we know the start condition and end condition clearly we can use the for loop

Syntax : for(initialization/start ; condition/ end ; counter)

{

Body ;

}

The for body is executed repeatedly , until the end condition becomes false.

While loop : Whenever we don’t know the start condition and end condition clearly we can use the for loop

Syntax : while(condition/ end)

{

Body ;

}

The while body is executed repeatedly , until the end condition becomes false.

Do While loop : Whenever we don’t know the start condition and end condition clearly we can use the do while and it will be executed at least once even if condition is false.

Syntax : Do

{

Body ;

} while (condition/ end)

The do while body is executed repeatedly , until the end condition becomes false.

section 2 --- ECMA (es6)

var & let & const

differences---

1) Initialization declaration

2) Re initialization re declaration

3) Scope

var

Declaration and initialization can be in different line also

Re initialization is possible

Re declaration is possible.

Var is function scope

let

declaration and initialization can be in different line also

Re inizialization is possible

Re declaration is Impossible.

let is block scope

const

declaration and initialization cannot be in different line.

Re inizialization is Impossible

Re declaration is Impossible.

const is block scope.

JavaScript Function

JavaScript functions are **defined** with the function keyword.

You can use a function **declaration** or a function **expression**.

Function Declarations

```
function functionName(parameters)
{
  // code to be executed
}
```

We have different types of function

1) Normal function

2) Anonymous function

3) Arrow function

Anonymous function

Function without the function name is nothing but anonymus function

Syntax : var x = function()

```
{
  Statements;
}
```

We should call the function by using variable name we have assigned

Arrow function

Function without the function name and function keyword is nothing but Arrow function and inbetween we use fat arrow symbole (=>).

Syntax : var x = ()=>

```
{
  Statements;
```

}

We should call the function by using variable name we have assigned

Array

Array :- Array in javascript is used to store both homogeneous and heterogeneous

type of data.

syntax :- var array_name = [value1, value2, value3.....]

array values are identified by their index position, and its starts from zero

and ends at array length -1

INBUILT METHODS OF ARRAYS :

length();

:- length method will return the length of the given array.

syntax: arrayname.length

concat(array);

:- concat method will add the elements of given two array.

syntax: array1.concat(array2)

toString();

:- tostring method will give us the string representation of

the given array and it wil not convert the given array into string.

syntax: array.toString();

join();

:- join method is similar to toString method , but we can add some

symbols in between the array elements.

syntax: array.join(" + ")

pop();

:- pop method is used to delete the last element from the given array,

and every time we call the pop method last element of current array

will be deleted.

syntax: array.pop();

push();

:- push method is used to add the new element to the last position of the

given array.

syntax: array.push(value)

shift();

:- shift method will delete the first element from the given array.

whenever we call the shift method always first element will be deleted.

syntax: array.shift()

unShift(value);

:- unshift is a method where we can use it to add the new elemet in the first

position (arr[0]) of the given array.

syntax: array.unshift(value)

fill(value,position);

:- fill method will replace all the elements of an array with the given value,

and from the given position.

syntax: array.fill(value,startIndex,endIndex)

includes(value);

:- include method will search the given element in the array, if the element is

present in the array ouput is true or else false.

sort()

:-Sorts the elements of an array in place and returns the array.

splice(index,how_many)

:-Adds and/or removes elements from an array,delete the elements from the givenindex position upto the number of [how_many] given.

slice(start_index, end_index)

:- Extracts a section of the calling array and return the elements from

start index to ebd index -1.

indexOf()

:- Returns the first (least) index of an element within the array equal

to an element, or -1 if none is found.

lastIndexOf()

:- Returns the last (greatest) index of an element within the array equal to

an element, or -1 if none is found.

.reverse()

Reverses the order of the elements of an array in place. (First becomes the

last, last becomes first.

.forEach()

Calls a function for each element in the array.

.filter()

.reduce()

.map()

if we try to print deleting methods we will get

deleted value from array.

ex : console.log(a.pop());

console.log(a.shift());

if we try to print adding value methods we will get

lenght of the new array.

ex : console.log(a.push(1));

console.log(a.unshift(1));

string methods.

1. charAt(index)

: char at method will return the value present in given index

value of string.

2. substr(0, 5)

: substr method will print the characters from given index position to the numbers provided.

3. toUpperCase()

: this method will return all the characters of given string into uppercase format.

4. toLowerCase()

: this method will return all the characters of given string into uppercase format.

5. concat(str1,str2..)

: concat is used to add the one string to the another string, and we can add any number string.

6. includes("value")

: include method of the string will check the given value is present inside that sting or not.

7. indexOf(value)

: index of will give us the index position of the given character in that string and searching starts from startring position.

if not present it will return -1

8. lastIndexOf(value)

: last index of will give us the index position of the given

character

in that string and searching starts from last position.

if not present it will return -1

9. replace(search_val, replace_val)

: replace method will search the given value, and if found in the

given string it will replace that string to 2nd value.

10. trim()

: trim method will delete all the leading and trailing white spaces,

from the given string.

11. startsWith()

12. endsWith()

13. search()

14. split()

15. repeat()

OOPS - object oriented programming system

object

what

object is a pair of keys and values.

we need objects to introduce an entity to the virtual word.

what is an entity : Anything which has property and a behaviour.

1. how to create an object

By object literal { keys:values }

values inside key can be number, string, boolean,array,function, undefined,null,Nan.key is an identifier and it can also be a string.

2. How to access an object.

we can call an object by using object name.

we can access the properties of an object by 2 ways

1. objectName.KeyName

: by using dot operator

2. objectName["keyname"]

: by using array like literal

3. if you are trying to access inside the object

we should use this keyword.

THIS : this key will represent the reference of current object

3. How to modify an object.

object.key = new value;

or

object["key"]=new value;

4. How to delete a key

delete object.key;

by using delete operator

5. How to add new key to object

object.newkey = value;

CRUD operation :- create, read, update, delete

creation,accessing,(delete,reasign,newkey),delete

6. constructor function new keyword.

function constructor name(properties)

{

this.property=property;

}

to call -> new constructor_name(values)

Math objects

The JavaScript Math object allows you to perform mathematical tasks on numbers.

Method	Description
abs(x)	Returns the absolute value of x
acos(x)	Returns the arccosine of x, in radians
acosh(x)	Returns the hyperbolic arccosine of x
asin(x)	Returns the arcsine of x, in radians
asinh(x)	Returns the hyperbolic arcsine of x
atan(x)	Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radia
atan2(y, x)	Returns the arctangent of the quotient of its arguments

atanh(x)	Returns the hyperbolic arctangent of x
cbrt(x)	Returns the cubic root of x
ceil(x)	Returns x, rounded upwards to the nearest integer
cos(x)	Returns the cosine of x (x is in radians)
cosh(x)	Returns the hyperbolic cosine of x
exp(x)	Returns the value of E^x
floor(x)	Returns x, rounded downwards to the nearest integer
log(x)	Returns the natural logarithm (base E) of x
max(x, y, z, ..., n)	Returns the number with the highest value
min(x, y, z, ..., n)	Returns the number with the lowest value
pow(x, y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1

round(x)	Rounds x to the nearest integer
sign(x)	Returns if x is negative, null or positive (-1, 0, 1)
sin(x)	Returns the sine of x (x is in radians)
sinh(x)	Returns the hyperbolic sine of x
sqrt(x)	Returns the square root of x
tan(x)	Returns the tangent of an angle
tanh(x)	Returns the hyperbolic tangent of a number
trunc(x)	Returns the integer part of a number (x)

Date Object

JavaScript **Date Object** lets us work with dates:

Creating Date Objects

Date objects are created with the new Date() constructor.

There are **4 ways** to create a new date object:

new Date()
new Date(*year, month, day, hours, minutes, seconds, milliseconds*)
new Date(*milliseconds*)
new Date(*date string*)

new Date()

new Date() creates a new date object with the current date and time:

new Date(*year, month, ...*)

new Date(*year, month, ...*) creates a new date object with a specified date and time.

7 numbers specify year, month, day, hour, minute, second, and millisecond (in that order):

new Date(millisecond)

new Date(millisecond) creates a new date object with the default date and time + given milliseconds value.

new Date(*date String*)

new Date(date String) creates a new date object from a date string:

There are generally 3 types of JavaScript date input formats:

Type	Example
ISO Date	"2015-03-25" (The International Standard)
Short Date	"03/25/2015"
Long Date	"Mar 25 2015" or "25 Mar 2015"

Date methods

JavaScript Get Date Methods

Method	Description
getFullYear()	Get the **year** as a four digit number (yyyy)
getMonth()	Get the **month** as a number (0-11)
getDate()	Get the **day** as a number (1-31)
getHours()	Get the **hour** (0-23)
getMinutes()	Get the **minute** (0-59)
getSeconds()	Get the **second** (0-59)
getMilliseconds()	Get the **millisecond** (0-999)
getTime()	Get the time (milliseconds since January 1, 1970)
getDay()	Get the weekday as a number (0-6)
Date.now()	Get the time. ECMAScript 5.

JavaScript Set Date Methods

Method	Description
setDate()	Set the day as a number (1-31)
setFullYear()	Set the year (optionally month and day)
setHours()	Set the hour (0-23)
setMilliseconds()	Set the milliseconds (0-999)
setMinutes()	Set the minutes (0-59)
setMonth()	Set the month (0-11)
setSeconds()	Set the seconds (0-59)
setTime()	Set the time (milliseconds since January 1, 1970)

JavaScript Classes

ECMAScript 2015, also known as ES6, introduced JavaScript Classes.

JavaScript Classes are templates for JavaScript Objects.

Use the keyword class to create a class.

Always add a method named constructor():

Syntax

```
class ClassName

{
  constructor() { ... }
}
```

The Constructor Method

The constructor method is a special method:

- It has to have the exact name "constructor"
- It is executed automatically when a new object is created
- It is used to initialize object properties

If you do not define a constructor method, JavaScript will add an empty constructor method.

Class Methods

Class methods are created with the same syntax as object methods.

Use the keyword class to create a class.

Always add a constructor() method.

Then add any number of methods.

JavaScript Class Inheritance

Class Inheritance

To create a class inheritance, use the extends keyword.

A class created with a class inheritance inherits all the methods from another class:

The super() method refers to the parent class.

By calling the super() method in the constructor method, we call the parent's constructor method and gets access to the parent's properties and methods.

JAVASCRIPT DOM

Section 1. Getting started

Understanding the Document Object Model in JavaScript

Section 2. Selecting elements

getElementById() – select an element by id.

getElementsByName() – select elements by name.

getElementsByTagName() – select elements by a tag name.

getElementsByClassName() – select elements by one or more class names.

querySelector() – select elements by CSS selectors.

Section 3. Traversing elements

Get the parent element – get the parent node of an element.

parentNode

parentElement

Get child elements – get children of an element.

childNode

children

hasChildNodes()

childElementCount

Get siblings of an element – get siblings of an element.

nextSibling

nextElementSibling

previousSibling

previousElementSibling

Section 4. Manipulating elements

createElement() – create a new element.

appendChild() – append a node to a list of child nodes of a specified parent node.

textContent – get and set the text content of a node.

innerHTML – get and set the HTML content of an element.

innerHTML vs. createElement – explain the differences beetween innerHTML and createElement when it comes to creating new

elements.

insertBefore() – insert a new node before an existing node as a child node of a specified parent node.

insertAfter() helper function – insert a new node after an existing node as a child node of a specified parent node.

append() – insert a node after the last child node of a parent node.

prepend() – insert a node before the first child node of a parent node.

replaceChild() – replace a child element by a new element.

removeChild() – remove child elements of a node.

Section 5. Working with Attributes

HTML Attributes & DOM Object's Properties – the relatunderstandionship between HTML attributes & DOM object's properties.

setAttribute() – set the value of a specified attribute on a element.

getAttribute() – get the value of an attribute on an element.

removeAttribute() – remove an attribute from a specified element.

hasAttribute() – check if an element has a specified attribute or not.

Section 6. Manipulating Element's Styles

style property – get or set inline styles of an element.

getComputedStyle() – return the computed style of an element.

className property – return a list of space-separated CSS classes.

classList property – manipulate CSS classes of an element.

Element's width & height – get the width and height

of an element.

Section 7. Working with Events

JavaScript events – introduce you to the JavaScript events, the event models, and how to handle events.

Handling events – show you three ways to handle events in JavaScript.

1) addEventListener()

2) using onClick attribute.

3) using onclick property

Mouse events – how to handle mouse events.

Keyboard events – how to deal with keyboard events.

Scroll events – how to handle scroll events effectively.

scrollIntoView – learn how to scroll an element into view.

Focus Events – cover the focus events.

haschange event – learn how to handle the event when URL hash changes.

Event Delegation – is a technique of levering event bubbling to handle events at a higher level in the DOM than the element on which the event originated.

dispatchEvent – learn how to generate an event from code and trigger it.

Custom Events – define a custom JavaScript event and attach it to an element.

MutationObserver – monitor the DOM changes and invoke a callback when the changes occur.

Section 8. Scripting Web Forms

JavaScript Form – learn how to handle form submit event and perform a simple validation for a web form.

Radio Button – show you how to write the

JavaScript for radio buttons.

Checkbox – guide you on how to manipulate checkbox in JavaScript.

Select box – learn how to handle the select box and its option in JavaScript.

Add / Remove Options – show you how to dynamically add options to and remove options from a select box.

Handling change event – learn how to handle the change event of the input text, radio button, checkbox, and select elements.

Handling input event – handle the input event when the value of the input element changes

Topics remning

1) Different data types