JS

Date
Page

* Hoisting

→ Is a state where we can use function and variable before the declaration.

* Constant objects And Arrays

→ It doesnot define a constant value. It defines a constant reference to a value.

You can NOT
- Reassign a constant value
- Reassign a constant array
- Reassign a constant object

But you CAN
- change the element of constant array.
- change the properties of constant object.

* JavaScript Type Operator

typeof → Returns the type of a variable
instanceof → Return true if an objects is an instance of an object type.

Exponential Notation

```
let y = 123e5 ;
let x = 123e-5 ;
```

IN JS objects are KING

(i) object are objects
(ii) math are objects
(iii) Function are objects
(iv) Dates are objects
(v) Arrays are objects
(vi) Maps are objects
(vii) sets are objects

Object Displaying

→ by name = person.age, person.name
→ by a loop = for (let x in person) { text }
→ by object.values() = object.values(person);
→ JSON.stringify() = JSON.stringify(person);
→ By object.entries() =
(let [fruits, value] of object.entries(arrays))

JS Events

onclick → user clicked
onchange → element has been changed
onmouseover → mouse over
onmouseout → mouse away
onkeydown → pushes key down
onload → browser has finished loading the page

* Template Strings ⇒ ` `, `${x}`

* Escape Character ⇒ \' \" \\

\b ⇒ Backspace
\f ⇒ Form Feed
\n ⇒ new Line
\r ⇒ Carriage Return
\t ⇒ Horizontal Tabulator
\v ⇒ vertical tabulator

The 6 escape character above were originally designed to control typewriters, teletypes, and fax machines. They do not make any sense in HTML

* let x = "John"; → string
let y = new String("John");

→ new keyword complicates the code and slows down execution speed. String objects can produce unexpected results.

NAN : not a number

```
let x = 123;
let x = new number(123);
```

(i) MAX_SAFE_INTEGER
(ii) MIN_SAFE_INTEGER

```
Number.isInteger(10);
Number.isSafeInteger(10000000000000000);
```

* Array Element can be objects

```
myArray[0] = Date.now;
myArray[1] = myFunction;
myArray[2] = myCars;
```

properties & method

```
cars.length;
cars.sort();
```

How to find is that array or not

- Array.isArray(fruits);
- ~~Array~~(fruits instanceof Array)

Nested Arrays and objects

Values in objects can be arrays and value in array can be objects.

JS Dates

new Date(); → current date.

new Date("2022-03-25"); → give that date in format

new Date(year, month, day, hours, minutes, seconds, ms)

new Date(date string)

new Date(milliseconds)

d.toString()

d.toDateString()

d.toUTCString();

d.toISOString();

JS math

properties

Math.E ⇒ return Euler's number
math.PI ⇒ return PI
Math.SQRT2 ⇒ return root of 2
Math.SQRT1-2 ⇒ return root of 1/2
Math.LN2 ⇒ logarithm of 2
Math.LN10 ⇒ logarithm of 10
math.LOG2E ⇒ logarithm base 2 of E
math.LOG10E ⇒ logarithm base 10 of E

method

math.round(x) → rounded to its nearest integer
math.ceil(x) → rounded up to its nearest integer
math.floor(x) → rounded down to its nearest integer
math.trunc(x) → give the x value
math.sign(x) → 1, -1, 0, (-, +, 0)
math.pow(8,2)
math.sqrt(x)
math.abs(x)
math.sin(x)
math.cos(x)
math.max()
math.min()
math.random()
math.log(x)

JS Random

math.floor(math.random()*10);

JS loop

for → loops code number of times.

for-in → loops through the properties of an object.

for-of → loops through the iterable object.

while → loops while a specified condition is true.

do-while → loops while a specified condition is true.

for In loop

↳ loops through the properties of objects.

```
const person = { fname: " ", lname: " ", age: 25 };

for (let x in person) {
    text += person[x];
}

Array.forEach(myFunction);
```

* For of loop

→ loops through the values of an iterable object.

```
for(let x of Arrays){
    text t = x;
}
```

JS sets

→ collection unique values.
→ each value can only occur once in a set.
→ can be of any type, primitive values or objects.

How to create sets

(i) passing array to new set()
(ii) create empty set and use add()

eg: const letters = new set(["a", "b", "c"]);
letters.size;

eg. const letters = new set();
letters.add("a");
letters.add("b");

We can use **for-of** for list element.

JS Maps

⟶ A map object hold key-value pairs key can be any datatype

How to create map

(i) new map
(ii) map.set()

e.g: const fruits = new map([
["apples", 500],
["bananas", 300],
]);

e.g: fruits.set("oranges", 200);

e.g fruits.get("apples"); ⇒ 500

fruits instanceof map; → true
fruits instanceof object; → true

Javascript Iterables

→ An iterables can be iterated over with for-of loop.

sytax:

```
for (variable of iterable){

}
```

Iterating over → string
Array
Set
map

→ iterator protocol defines how to produce a sequence of value from object.

→ object became iterator when implements a next().

next() method return:

(i) value → value return
(ii) done → true / false if completes or can con-

```
obj[Symbol.iterator] = function{
    next(){
    }
}
```

JS iterator

→ provides a standard way to access elements sequentially.

→ Iterator protocol: , must have next() method function

drop() → skip a specified number of elements before yielding the rest.

every()

filter() → new iterator containing element that satisfy a filter function.

find() → return element the satisfies a test function.

from() → creates iterator object from an existing iterable or iterator obj

map() → new iterator with transformed by a map function.

take() → return a new iterator that yields at most a specified number of element.

reduce() →

note: null is a primitive value, however, typeof return 'object'. this is a well-known bug in JS.

Type Coversion

(i) number()
(ii) parseFloat()
(iii) parseInt()
(iv) Unary operator (+)
(v) string()

toExponential()
toFixed()
toPrecision()

JS Error

try - catch block

- throw → exception is caught by the catch statement and custom error message is displayed.

HTML validation

<input id="demo" type="number" min="5" max="10" step="1">

finally statement

→ execute the code after try-catch block.

Error Name Values

EvalError → An error has occurred in the eval() function
RangeError → A number "out of range" has occured
ReferenceError → illegal reference has occured
SyntaxError → syntax error has occurred
URIError → An error in encodeURI() has occured

newer JS do not throw EvalError use SyntaxError instead.

The nodeName property

* nodeName
  - nodename is read-only.
  - nodeName of an element node is the same as tag name.
  - " " " attribute node is the attribute name.
  - " " the document node is always = # document.
  - " " a text ~~document~~ node is always = # text.

* nodeValue
* nodeType

document.getElementById("demo").nodeType;

Dom Navigation

everything in an HTML document is a node.

- The entire document is a document node.
- Every HTML element is an element node.
- The text inside HTML elements are text nodes
- Every HTML attribute is an attribute node (dep
- All comment are comment nodes.

parentNode
childNodes[nodenumber]
firstChild
lastChild.
nextSibling
previousSibling

document.getElementById("box").firstChild.nodeValue;

document.body → body document
document.documentElement → full document

Dom Event Listenner

Date
Page

Bubbling → event first run on that element then on its parent then upward through the dom.

addEventListener() → adding event / attached event handler
removeEventListener() → removing event

Syntax:

element.addEventListener(event, function, useCapture);

↙ boolean value whether to use event bubblings or event.

Note: dont use "on" prefix: use "click" instead of "onclick".

Capturing → event goes from root to child
capturing is rarer, used for special cases like: intercepting events before they hit the target.

use the function:

element.addEventListener(event, function, useCapture);

useCapture => false bubbling
useCapture => true capturing

DOM CSS

→ changing the style of css.

document.getElementById(id).style.property = new style

Dom Animation

id = setInterval(frame, 5),

function frame(){ }

Dom Event

Example of event.

(i) user clicked the mouse
(ii) web page has loaded
(iii) image has been loaded
(iv) mouse move over element
(v) input field is changed
(vi) HTML form is submitted
(vii) user stokes a key.

event are :

(1) onclick, onload, onunload,
- onload & onunload is used for cookies.

(2) oninput, onchange, → when leave the input field

(3) onmouseover & onmouseout & onmousedown & onmouseup & onclick,

Changing HTML Content

- innerHTML => modify the content of HTML element.
- attribute => change the value of HTML attributes
  e.g -> document.getElementById("image").src = "image.jpg";

document.write() => used to write directly to the HTML output stream.

HTML Constraint Validation

- constraint validation HTML Input Attribute.
- constraint validation CSS pseudo selector.
- Constraint validation DOM properties and methods.

* constraint validation HTML input Attributes
→ disabled, max, min, pattern, required, type

* Constraint validation CSS pseudo selectors
  :disabled =>
  :invalid =>
  :optional =>
  :required =>
  :valid =>

DOM JS

- In dom all html element are defined as obj.
- interface is the properties and methods of each object.

action =
value =
properties = get & set
method = action we gonna use

getElementById => method
innerHTML => property

Element select method

(i) by id
(ii) by tag name
(iii) by class name
(iv) by CSS selectors
(v) by HTML object collections

document.getElementById("")
document.getElementByTagName("")
document.getElementByClassName("")
document.getElementsByAll("")
document.forms("")

Array Iteration

- forEach
- map
- flatMap()
- reduce()
- reduceRight()
- every()
- filter()
- some()
- from()
- keys()
- entries()
- with()
- spread() ... array
- Rest()

JS function

- function call
- function apply
- function bind ⇒ we can it on W3school

JS Object

need be learn more deep.

JS Array Method

(i) length
(ii) tostring()
(iii) at()
(iv) join('*')
(v) push(), pop(),
(vi) shift()
(vii) unshift()
(viii) concat() → merging array
(ix) copyWithin()
(x) flat() reduce dimension
(xi) flatMap()
(xii) splice() → add new item
(xiii) slice() → slice out a piece
(XIV)
(XV)
(XVI)

Search Method

→ indexof()
→ find()
→ lastIndexOf()
→ Array includes()
findIndex()
findLast()
arrayfindLastIndex()

Alphabetic sort

Array sort()
Array reverse()
Array toSorted()
Array toReverse()
Array objects

Numeric sort

Numeric sort
Random sort
math.min()
math.max()

note: "use strict"; defines that javascript code should be executed in "strict mode".

JS modules

export (name, age);

export default filename, function_name;

import $name, age y from "./person.js";

import

JS JSON

JSON.parse() to convert the string into a JS object.