JavaScript - 8

Array methods →

⊗ arr.forEach (some function definition or name) ;

⇒ it apply the callback function to array's every element.

⊗ newarr = arr. map (some function definition or name);

⇒ it return elements after do some operation by callback function that are stored in new array. of same sized.

⊗ Let newArr = arr. filter (some function definition or name);

⇒ It store the elements in new array after some ~~operations~~ of callback function to filter the elements for new Array.

Conditions

• if the conditions of callback function are true. than the element is filtered for new array.

classmate

DATE

⊗ Every

arr. every (some function definition or name);

⇒ Returns true if every element of array gives true for callback function. else return false.

(like And)

⊗ Some

arr. some (some function definition or name)

Returns true if some elements of array give true for some function. else return false

(like OR)

⊗ Reduce

arr. reduce (reducer function with 2 variables (accumulator, element));

reduces the array to a single value.

default Parameters →

Giving a default value to the arguments if the value is not passed.

• use default parameters for lasts arguments so that you not get "NaN".

SPread →

Expands an iterable (array, strings etc.) into multiple

classmate PAGE

Values.

Spread (array literals) ⇒ after spread can assign these values to a new array.

Let newarray = [... array];

Spread (object literals) ⇒ we can spread an object in another object and incase of array or strings they can also spread but in need of key : value there is only value so define key by the index of array or string values.

Rest → Allows a function to take an indefinite number of arguments and bundle them in an array.

{ every function has a default collection for variable of arguments that is "arguments" but it is not a array }

Syntax ⇒
```
function sum (... args) {
        return args.reduce((add, el) => add+el);
}
```

• In this function we can give infinite args. they will be stored in args named array

(additional)

- if we need extra parameters or arguments than we ~~can~~ have to put them firstly so that we will not get error.

<u>Destructuring →</u>

Storing values of arrays into multiple variables

syntax → let [variable 1, var2, var 3] = arrayname ;

- the value of 1st, 2nd, 3rd index are stored in these variables respectively.

<u>Destructuring (objects) →</u>

Syntax → let { variable1 , var2 } = student ;

- In this case variable1, var2 are checked for same named keys and than the value of these keys of student object is stored in var 1, var 2 .

syntax → let { Key1: var1 , key: var2 } = student ;
(Object)

- In this case the value of Key 1, key 2 is stored in the var1, var 2.

Syntax → let { Key 1 : var 1 = "value 1" } = student ;

- in this case if ~~value is~~ key 1 is not in the

classmate

DATE

student object than the value of var 1 is value1