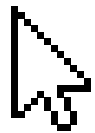
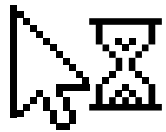
Computer Science :
JavaScript

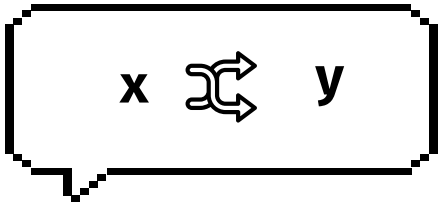


Week 6: Scope & Array

Scope

Scope defines where variable can be accessed in the code. Whether it can be used universally or restricted to a section.





01

Vocabulary

Blocks

```
function someCode {
//This is a block
}
```

Blocks are the code inside the curvy brackets

{}

Global Scope

```
const color = 'blue'

function rainbow() {
  return color;
}

console.log (rainbow());
// outputs blue
```

Global variables declared outside the block and can be accessed all throughout the code including in blocks

Block Scope

```
function rainbow() {
  const color = 'blue'
  console.log (color);
}

rainbow();
// outputs blue

console.log (color);
// outputs Error
```

When a variable is defined in a block it can only be accessed within the block {}

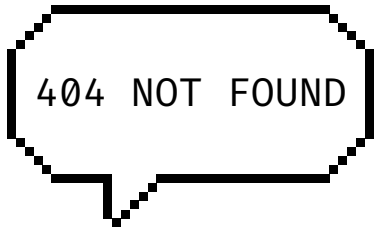
Scope Pollution

```
let color = 'blue'

function rainbow() {
  color = 'red' // this is not a
block variable bc there is not
keyword
}

console.log (rainbow());
// outputs red
console.log (color);
// color is reassigned to red
```

Avoid having too many global variables because it can cause confusion within code and it is difficult to keep track

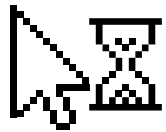
02

Array



Array

An **Array** is the method of making a list of elements and can store several data types

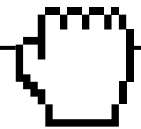


Array

[]:array literals are where you would input your array elements *(items inside an array)* and separate by commas

Ex:

```
const groceries = [‘apples’, ‘milk’, ‘onions’];
```



Practice

Create an array with 4 elements (any data type) and log (print) the array

Element Position

Elements are numbered position starting at zero
(the first item is zero)

arrayName[element number]:to call an element

Ex:

```
const groceries = ['apples', 'milk', 'onions'];
console.log(groceries[1]); //outputs milk
```

Practice

Use the array you created and log the 3rd item in the array

Changing Element Value

You choose your specific element and then assign it a new value

Ex:

```
let groceries = [‘apples’, ‘milk’, ‘onions’];
groceries[0] = ‘papayas’;
console.log(groceries); //outputs papayas, milk, onions
```

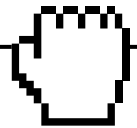
Practice

Change the value of the 2nd item in your array and then log your array

Changing Element Value pt 2

Keyword let:
You can assign an array elements and its overall value, so you can reassign a variable with a new array

Keyword const:
You can reassign the elements within the array but you can not assign the variable a new array



Practice

Given:
let cars = [‘Rivian’, ‘Volvo’, ‘Honda’, ‘Bugatti’]

Task:
Reassign ‘Honda’ to ‘Audi’ then log the array
Reassign cars to the single array ‘Audi’ then log array

Given:
const shoeSize = [7.5, 10, 6, 9.5]

Task:
Reassign 9.5 to 9 then log the array
Reassign shoeSize to the single array 9 then log the array

Nested Array

Nested Arrays are arrays stored within arrays.
Nested arrays can be grouped by index numbers
and called as such

arrayName[index number]:to call a nested array

Ex:

```
const numbers = [[1, 2], [3, 4], [5, 6]];
console.log(numbers[0]); //outputs 1, 2
console.log(numbers[2][1]); //outputs 6
```

IDENTIFIED

03

Array Features

.length

.length property provides the amount of elements in the array

```
console.log(arrayName.length)
```

.push []

.push() method allows you to add elements to an array

```
const arrayName = [‘item1’, ‘item2’]
arrayName.push(‘item3’, ‘item4’)
```

.pop []

.pop() method removes the last element
in the array and does not take arguments

```
const arrayName = [‘item1’, ‘item2’]
arrayName.pop() // removes item2
```

Practice

1. Create an array with 5 elements
2. Add 3 elements to the array
3. Log the length
4. Remove the last element
5. Log the array