JavaScript

Javascript is the duct tape of the Internet.
- Charlie Campbell

YET ANOTHER
CODING CLUB

What can js do ?

- Helps in creating dynamic
- Run a game inside the browsers.
- Helps in making PPTs online.
- Make a real time chat possible.
- We can run js on server side as well as client side.
- Who uses : LinkedIn, Netflix, NASA and list goes on.

JavaScript is the only language that I'm aware of that people feel they don't need to learn before they start using it.

- *Douglas Crockford*

YET ANOTHER
CODING CLUB

Language fundamentals

Variables : Containers that store values. You start by declaring a variable with the let keyword, followed by the name you give to the variable.

```
let myVariable = "Bob"; var myVariable = "Bob";

const myVariable = "Bob";
```

- JavaScript is case sensitive. This means `myVariable` is not the same as `myvariable`
- Variable names **cannot contain spaces**. Variables **cannot be the same as reserved keywords** such as if or const . By convention, JavaScript variable names are written in camelCase.

Comments : Use to describe some part of the code.

```
// This is a comment

/* Everything in between is a comment.*/
```

Data types

- String : This is a sequence of text

```
let myVariable = 'Bob'; or let myVariable  "Bob";
```

- Number : This is a number.

```
let myVariable = 10;
```

- Array : This is a structure that allows you to store multiple values in a single reference.

```
let myVariable = [1,'Bob','Steve',10];
```

  Refer to each member of the array like this : `myVariable[0], myVariable[1]`

- Object : Consists of unordered key-value pairs

```
let school = { name: 'xyz school', address: '123 street', grade: 10 }
```

YET ANOTHER
CODING CLUB

Operators

- Addition :

```
let myVariable = 3 + 4;

let myVariable  "Bob" + ‘Alice’; // string addition
```

- Subtraction, Multiplication, Division :

```
let myVariable = 3 - 4;

let myVariable = 3 * 4;

let myVariable = 3 / 4;
```

- Strict equality `(===)` : This performs a test to see if two values are equal and of the same data type. It returns a true/false (Boolean) result.
- Not Equal `(!==)` : Checks two values are equal or not.

YET ANOTHER
CODING CLUB

Conditionals

```
let iceCream = "mango";

if (iceCream === "mango")  {

  alert("Yay, I love mango ice cream!");

} else if (iceCream === "Strawberry")  {

  alert("Awwww, but Strawberry is my favorite");

}

else {

  alert("I don’t like icecream much.");

}
```

YET ANOTHER
CODING CLUB

Loops

```
for (let i = 0; i < 10; i++)  {
   // some code
}

 while (condition) {
    // some code
 }
```

Q. Find the sum of first 20 even numbers using both loops.

Array methods

- .toString() : converts an array to a string of (comma separated) array values.
- .pop() : removes the last element from an array.
- .push(value) : adds a new element to an array (at the end).
- .length : gives the length of array
- .map() :

```
function myFunction(num) {
      return num * 10;
}
const numbers = [65, 44, 12, 4];
const newArr = numbers.map(myFunction)
```

YET ANOTHER
CODING CLUB

Practise questions

Q1. Create a array of 8 numbers and store it in variable named *arr*.

Q2. Find maximum number of *arr.*

Q3. Find sum of all elements of *arr.*

Functions

- Way of packaging functionality that you wish to reuse.

```
function multiply(num1, num2) {
        let result = num1 * num2;
        return result;
}
```

- Q. Write a function that takes two number and returns their Greatest common divisor.

YET ANOTHER
CODING CLUB

Arrow Functions

- Way of packaging functionality that you wish to reuse.

```
const multiply = (num1, num2) => {
      let result = num1 * num2;
      return result;
}
```

- Q. Write a function that takes two number and returns their Greatest common divisor.

YET ANOTHER CODING CLUB

Call back functions

- A callback is a function passed as an argument to another function. This technique allows a function to call another function. A callback function can run after another function has finished.

```
function sayHello()  {

  console.log("Hello, world!");

}

setTimeout(sayHello, 2000);

setInterval(sayHello, 2000);
```

YET ANOTHER
CODING CLUB

Thank you

Practical use cases of javascript

- Asynchronous Programming
- Functional Programming
- Ability to Write Cross-browser Code
- ReactJS
- Node JS
- TypeScript
- jQuery

YET ANOTHER
CODING CLUB

```
// Q1

// let arr = [1, 2, 3, 5, 6, 7];

// var ans = 0

// for( let i = 0; i < 6; i++ ) {

//     ans += arr[i] + arr[i-1]

//     ans += arr[i+1] + arr[i]

// }

// console.log(ans)




// Q2

// var a = 1;

// var b = 0;

// while (a <= 3)

// {

//   a++;
```

YET ANOTHER
CODING CLUB