Javascript

What is Javascript ?

Javascript is a programing language for web application. It has nothing to do with `Java`.

Linking

1. Internal Linking: Javascript code is written inside the `HTML` code.
2. External Linking: Javascript code is written in a separate `js` file. This is written without script tags.

```
<script>
console.log("This is how javascript is written inside HTML code");
</script>
```

Javascript code is interpreted in browser you can see the output using

```
console.log()
```

which can be shown in the console. Javascript file is often named as script.js

Variables in Javascript

There are three types of variables in Javascript `var`, `let` and `const`. Variables in javascript are auto selected in datatype this will be discussed further. .

Var

This is old way of declaring a variable and is function scoped. It doesn't have a scope in the `block` in which it was declared. It can be updated again and again . At the time of it's declaration if any values is not assigned then it is `undefined`.

```
//Variable
var age = 20;
// here 22 is an integer so the variable will also be an integar
var name = "Ali";
//here the datatype will change into a string datatype

if(true)
```

```
{
    var a = 10;
}
console.log(a);
// Here I would be able to access the variable when though it is declared
outside a block
```

Let

It is a new way of declaring a variable. It is the same as a `var` except it is it is local scoped. It is only accessible in the `block` in which it was declared in. It is recommend to use let.

```
if(true)
{
    let a = 20;
    console.log(a);
    // only here would I be able to access the variable
}
console.log(a);
// I can't access the variable from here
```

Const

The scope of const is the same as let. Once declared it's value cannot be reassigned. We also won't be able to change it's datatype.

```
const a = 20;
a = 12;
// here the value can't be reassigned It is read-only. this will throw an
error
```

Scope

Scope defines the accessibility for difference variable and functions who are part of the function. The scope is different for variables like `var`, `const` and `let`. The scope of these variables have been discussed before.

Local Scope	Global Scope
A variable declared inside a function or block belongs to that scope.	A variable declared outside a function or a block belongs to this scope.

Local Scope	Global Scope
It cannot be accessed outside the function or block.	It can be accessed from anywhere on the code.

Datatypes in Javascript

Javascript supports a number of datatypes which will be discussed further. The datatypes supported in javascript are as follows:

- Numbers
- Strings
- Booleans
- Undefined
- NULL
- Objects

Numbers

All integers and floats are numbers and we can perform arithmetic operations like +, -, * and / on them. The datatype of a variable is chosen automatically assigned in javascript

```
let a = 12;
let b = 2.32;
console.log(typeof(a));
console.log(typeof(b));
//Output
//number
//number
```

Strings

Textual data is stored inside a string variable enclosed in ' ' and " " etc. Alphanumeric characters are stored in a string.

```
let name = "Ali";
console.log(name);
//Output: Ali
```

Booleans

Boolean represents true or false values.

```
let a = false;
console.log(a);
//output: false
```

Null

Nothing is stored inside the variable. It is used to represent not assigned.

```
let a = null;
console.log(a);
//Output: null
```

Undefined

A variable that has been declared but a value is not assigned yet.

```
let a;
console.log(a);
//Output: undefined as no value has been assigned yet
```

Object

Object in javascript is quite difference from objects in other languages like C++. It is a collection of key value pairs. You can store different type of values in them

```
const person = {
    firstName : "Ali",
    secondName : "Abdullah",
    age : 20,
    signedIn : true,
    lastSignIn : null
}
```

Conversion in Javascript

They are two types of datatype conversion in javascript. Implicit and explicit.

Implicit conversion:

Javascript automatically converts data types.

```
console.log("5" -2);// Output:3
console.log("5"  + 2); //Output: "52"
console.log("10" * "2");//Output:20
console.log(5 == "5");//Outpue:true
```

Explicit conversion:

You have to manually convert the data types.

```
console.log(Number("5") - 2);//Output: 3
console.log(String(5) + "2");//Output: "52"
console.log(Boolean(0));//Output: false
console.log(parseInt("10.5"));//Outpue: 10
console.log(parseFloat("10.5"));//ouput: 10.5
```

Conditional Statements

They allow the execution of different code based on the conditions.

`If` and `else` statement

Executes the block of code when the condition is true. Executes the other block of code if the condition is not true.

```
let age = 23;
if(age<=18)
{
console.log('U r an adult');
}else
{
console.log('u r not an adult');
}
```

`else If` statement

Used when there are multiple conditions.

```
let marks = 85;
if(marks>=90) {
console.log("Grade:A");
}else if(marks>=75){
```

```
console.log("Grade:B");
}else if(marks>=60){
console.log("Grade:C");
}else{
console.log("Grade:F");
}
```

Ternary operator

Used as a short for `if` and `else` statements.

```
let age = 90;
age>=90? console.log("Yes"): console.log("No");
// you can also assign using it
let result = age>=90? "Yes": "No";
```

Switch Case in JavaScript

Used for multiple fixed conditions. `break` stops execution after a match. `default` runs if no case matches.

```
let option = 3;

switch (day) {
    case 1:
        console.log("Assalam u Alaium");
        break;
    case 2:
        console.log("Hello");
        break;
    case 3:
        console.log("Konichiwa");
        break;
    default:
        console.log("Invalid input");
}
```