Looping

It is also called iteration.

The process of executing a instruction /block of instruction repeatedly multiple times.

Note: when we design a loop it is a responsibility of a programmer to break the loop after achieving the desire task, if not the

program get into infinite loop statement.

If we don't break loop it will be infinite and control will not come out of loop.

Condition is use to break loop after some certain

Loop statements

1.While

a. Syntax

while(condition){

statement to be repeated;

}

2.Do-while :

The do-while loop loops through a

The do-while loop loops through a block of code once, then the condition is evaluated. If the condition is

true the statement is repeated as long as specified condition is true.

a. number of iteration in do while is 1

Syntax

do{

// statement;

}

while(condition);

Note:

a. in do while loop the body of the loop is executed first then the condition is evaluated.

b. If the condition evaluated is true the body of the loop inside the do statement is executed again.

c. The statement inside the do statement is executed again.

d. This process continue until the condition evaluates to false. Then the loop stops.

3. For

4. For-in etc

Functions

• Function is a block of instruction which is used to perform a specific task

• A function get executed only when it called.

• The main advantage of function is we can achieve code reusability.

• To call a function we need its reference and ()

Note: in javascript functions are beautiful, every function is nothing but an object.

Syntax to create a function.

Generally we can create a function in 5 ways:

1. Function declaration statement(function statement).

2. Function expression.

Function declaration / statement(function statement).

Syntax :

function identifier ([list_of_parameter,...])

{

statements;

}

1.

Note:

i. Function is object

ii. Name of function is variable which holds the reference of function object.

iii. Creating a function using function statement supports function hoisting.

iv. Therefore we can also call a function before function declaration.

v. when we try to log function name the entire function defination is printed.

```
console.log('start');
console.log(test);
function test(){
console.log('Hello');
}
console.log('start');
```

test - it is a function variable - it will return body of function

test () - invoking a function.

To call a function - function_name();// we can also pass data

Function_name(arguments_list,.....);

```
console.log('start');
function test(){
console.log('Hello');
}
test();
test();
test();
console.log('start');
```

parameter(placeholder: variable to hold data when function is called)

```
console.log('start');
function test(a){
console.log('Hello');
console.log(a);
}
test(10); argument : values which are passed.
console.log('start');
```

Parameters

The variables declared in the function defination is known as parameters.

The parameters have local scope (can be used only inside function body).

Parameters are used to hold the values passed by calling a function.

Eg:

```
function sum(a, b){
console.log(a + b)
}
// a and b are variables local to the function sum
```

Arguments

The values passed in the method call statement is known as arguments.

Note : An argument can be a literal, variable or an expression which gives a results.

Eg 1:

```
sum(10,20); // 10,20 are literals used as arguments
```

Eg 2:

```
sum(-10+3,-20);//-27
```

Eg 3:

```
sum(a ,b); //a and b are variable used as argument
```

return

1. It is a keyword used as control transfer statement in a function.

Return will stop the execution of the function and transfer control along with data

to the caller.

2.

```
function toMeters(cms){
return (cms/100);
```

```
}
var cms =2546;
console.log(toMeters(cms));
var m =toMeters(cms );
console.log(m);
```

The ability of a function to use as value is called first class function

2. Function as Expression

Syntax :

```
var/let/const identifier = function (){
}
```

The function is used as value

Disadvantage :

- function is not hoisted
- We cannot use a function before declaration.

```
a();// error
var a = function (){ /* a has address of function*/
console.log("fun");// will get entire function definition
}
```

In the above example function is not hoisted instead variable is hoisted and assigned with default value undefined. therefore type of a is not function it is undefined.

This

It is a keyword used as variable.

It holds the address of global window object.

Therefore with the help of this variable we can use members of global window object.

Whenever we call a function a new execution context is created. Inside that all local variable declare inside will be there and 1 more "this" will be there which is different from this

Inside this it have address of which function belongs to.

a.In javascript 'this' is a property of every function.(every function will have this keyword)

b. Generally this contains the current execution context to which the function belongs.

3.Arrow Functions

• Arrow function was introduce from ES6 OF JS.

• The main function of using arrow function is to reduce syntax

Syntax

(parameter_list,...) => {}

Note:

1. Parameter is optional.

2. If function has only one statement , then block is optional.

3.it is mandatory to create a block if return keyword is used.

Error if we write like this

```
const c =(n1+n2) => return n1+n2;
console.log(c(10,20));
```

Correct solution

```
const c =(n1, n2) => { return n1+n2};
console.log(c(10,20));
```

Operation taking parameter using arrow function

4.IIFE-(Immediate invocation function)

When a function is called immediately as soon as the function object is created.

Steps to achieve

• Treat a function like a expression by declaring in pair of bracket.

Add another pair pf braces next to it which behaves like function call statement.

○ Eg1

```
(function abc(){
console.log("Hi");
})();
```

○ Eg2

```
let a= ( ()=>{console.log("hello"); return 10})();
console.log(a);
```

5.Anonymous function

The function declare without any name which is called as anonymous function

Syntax:

```
function (){
instruction
}
```

//OBJECTS

-js objects is designed on simple object based paradigm.An object is a collection of properties, and a properties is association between a name(or key) and a value.

We can can create an object by using 3 ways

1.By using literals

-Create a single object , using an object literals

Syntax: var obj={

color:"white",

width:123,

}

2.create an object by using new keyword

-Create a single object ,using new keyword.

Syntax: var obj1=new Object();

3.create an object by using constructor function

Syntax:

function person(name,age,sal){

this.name=name;

this.age=age;

this.sal=sal;

}

//Date object

-Js date object represent a single moment in time in a platform

independent format.

-date object contains numbers that represents miliseconds since 1 january

1970

Date object methods

```
//date object
    // console.log("date object for regular date");
    var a=new Date();
    // console.log(a);
    //date (year,mon,day,hr,min,sec) 0=jan ,feb=1
    // console.log("manual date");
    // var b=new Date(2022,11,5,2,34,33);
    // console.log(b);
    // // //1sec=1000milisec
    // var c=new Date(1000);
    // console.log(c);
    // //date (date should be in string formate manual input)
    // var d=new Date("25,jan,2022");
    // console.log(d);
    // //get the date
    console.log(a.getDate());
    console.log(a.getMonth()+1);
    console.log(a.getFullYear());
    console.log(a.getTime());
    console.log(a.getHours());
    console.log(a.getDay());
    // //25/12/2022
    console.log(a.getDate()+"/"+a.getMonth()+"/"+a.getFullYear());
    console.log("27"+"/"+"jan"+"/"+"2022");
```

```
//Math objects
1310868812
-Math is inbuilt object that has properties and methods for mathematical
```

constant and functions.its not a function object

-math is work with number type

math object methods

```
var a= Math.PI;
      console.log(a);
      document.write(a);
      //round
      var b=Math.round(12.74);//round the nearest int value
      console.log(b);
      //power
     var c= Math.pow(2,3);
     console.log(c);
     //absolute it can conver -value to positive
     var d=Math.abs(-12);
     console.log(d);
     //Square root
     var e=Math.sqrt(16);
     console.log(e);
     //floor give 12.49 or 12.99 it will give the floor value 12
     var f=Math.floor(12.99);
     console.log(f);
     //ceil give 12.49 or 12.99 it will give the next vaule (ceil)
     var g=Math.ceil(12.44);
     console.log(g);
     //max
     var h=Math.max(22,33,44,55,66,77,88,11);
     console.log(h);
     //min
     var i=Math.min(22,33,44,55,66,77,88,11);
```

```
 console.log(i);
 //random
var j= Math.random();
console.log(j);
```