JavaScript

https://www.facebook.com/workspace.mnnit

Ravi Shankar, 20124025
ravishankarkumar@live.com

What is javascript?

JavaScript is a programming language

javascript syntax is similar to “C”

Variable types in JS

- Undefined/null
- Number
- boolean
- String
- Array
- Object

Declaring variable in JS

```
var varName;
```

```
//note that in js, all the six types of variable are declared with var only, we don't need int/char/bool etc
```

Default value of a variable

undefined

```
var a;
console.log(a);

// it will print undefined
```

Type undefined

```
var rs;
console.log(rs);
```

//it will output undefined

(undefined.js)

Type number, string, boolean

```
var myNum=12;
var myString=”shades”;
var myNo=false;
console.log(myNum);
console.log(myString);
console.log(myNo);
```

(cool.js)

Addition/ Subtraction/ Multiplication/ Division

```
var myNum1=12, myNum2= 15, myNum3;
myNum3 = myNum1+myNum2;
console.log(myNum3);

myNum3 = myNum1-myNum2;
console.log(myNum3);

myNum3 = myNum1*myNum2;
console.log(myNum3);

myNum3 = myNum1/myNum2;
console.log(myNum3);
```

(operator.js)

concat

```
var firstVar = "Workspace";
var confusion = firstVar + " MNNIT";
console.log(confusion);

//workspace.js
```

Exercise

WRITE and run a javascript code for adding two numbers.

Exercise

Write and run a javascript code that takes two variables for yourFirstName and yourSecondName. After that it concatanate them and print them like this.

yourFirstName yourSecondName

Array

```
var myStudents=[ ];
myStudents=["Abhijeet", "Abhinav"];
for(var i=0; i<2;i++){
    console.log(myStudents[i]);
}

// for loop is explained later
// array can hold different types of values also
//array.js
```

Array (Example of mixed)

```
var myStudents=[ ];
myStudents=["Abhijeet", 5];
for(var i=0; i<2;i++){
    console.log(myStudents[i]);
}


// array can hold different types of values also
//array-mixed.js
```

Array: addition and deletion

```
var myStudents=[ ];
myStudents=["Abhijeet", “Aanchal”];
console.log(myStudents);
var ind = myStudents.indexOf(“Aanchal”);
myStudents.splice(ind, 1);
console.log(myStudents);
myStudents.push(“Dixita”);
console.log(myStudents);
myStudents = [];
console.log(myStudents);

// array can hold different types of values also
//array-operation.js
```

Exercise

Write a program to initialise an array with five characters (s, n, y, a, d);
Remove 'a', and add 'r';

//array-ex.js

Looping

While loop

```
var r = 0;
while(r<10){
    console.log(r);
    r++;
}

//while.js
```

Looping

for loop

```
for(var r = 0; r<10; r++){
    console.log(r);
}
```

//for.js

Conversion to boolean

Undefined: false

Null: false

Boolean: The result equals the input argument (no conversion).

Number: The result is false if the argument is +0, -0, or NaN; otherwise the result is true.

String: The result is false if the argument is the empty String (its length is zero); otherwise the result is true.

Object: true

//falsy.js

==

== is a soft equality operator

It does type correction and then compares values

Following will evaluate to true

42 == "42"

0 == false

0 == ""

[] == ""

{} == "[object Object]"

"1" == true

===

=== does not do type correction while comparing

Following will evaluate to false

42 === "42"

0 === false

0 === ""

[] === ""

{} === "[object Object]"

"1" === true

!= !==

It must be quite obvious by now

!= means 'not equal'

!== means 'not equal' or 'not equal type'

5!=”5” if false

and 5!==”5” is true

//negative.js

If statement

```
var rs="ravi shankar";
if(rs == "ravi shankar"){
  console.log("This guy is "+rs);
}
var num = 8;
if(num){
  console.log("The non-zero number is true");
}
//conditionals.js
```

If else statement

```
var luck;
if(luck){
  console.log("Something must be wrong as undefined evaluated to true");
} else {
  console.log("undefined evaluates to false");
}
//conditionals-2.js
```

function

```
var fun1 = function(){
  console.log("U r good");
}
function fun2(){
  console.log("U r nice");
}

fun1();
fun2();

//function.js
```

Exercise

Write a function to take a number and print its twice.
Ex: print 4 if 2 is passed to the function

//function-sol.js

Objects

Javascript has an object data type
ex:- {fName:”Bajirao”, sName: “Mastani”};

//objects can have anything as a property, even objects, functions and array
var obj = {name: “node”};

//here obj is the name of object
//name is a property of obj
// node is the value of name

Accessing object properties

```
var obj = {name: “node”, teacher: “Ravi”};

console.log(obj);

console.log(obj.teacher);


//objectifying.js
```

Adding & modifying object properties

```
var obj = {name: “node”, teacher: “Ravi”};
console.log(obj);

obj.context = “MNNIT”;
console.log(obj);

obj.teacher = “Prashant”;
console.log(obj);


//moreobjectifying.js
```

deleting object properties

```
var obj = {name: "node", teacher: "Ravi", context: "MNNIT", year: "2015", sem: "second"};
console.log(obj);

 delete obj.sem;
 console.log(obj);

delete obj['year'];
console.log(obj);

var prop = "context";
delete obj[prop];
console.log(obj);


//lessobjectifying.js
```

Exercise

Make an object like this:

{context: "MNNIT", year: "2015", sem: "second"}

delete year property
update context property to “CP” from MNNIT

Object should become:

{context: "CP", sem: "second"}

//doctored-object.js

object properties: function

```
var obj = {
  name : "Ravi Shankar",
  showName : function(){
    return this.name;
  }
};

//this will tell you that it ia a function
console.log(obj.showName);

//this will actually do the intended tast
console.log(obj.showName());

//funobject.js
```

object properties: array

```
var obj = {
  students : ["abhijeet", "aanchal"]
};

console.log(obj.students);
console.log(obj.students[0]);

//arrobject.js
```

Exercise

Write a function that print whether the input parameter was true or false.

Call this function two times.

Once by passing an attributes that evaluates to false.

Next time by passing attribute that evaluates to true.

//sr.js

Exercise

Write a function that prints the factorial of a number.

Print factorial of a small number using this function.

//facto.js

&& and ||

A series of statements bound by these operators have a return value!

&&
- Returns First False
- or returns last true

||
- Returns First true
- Returns last false

Exercise

Anitesh got two passes for Indo-Aus T20 match. He asked four of his friends (through a node.js app), but he can take only one of them with him to the match.

The app has 4 variables for friends(var f1, f2, f3, f4;) for shalu, sneha, suraj, sumit. If shalu accepts the offer, f1 will become “shalu” or it will become false, similarly for others. Anitesh's preference order is shalu, sneha, suraj, sumit respectively.

Write js code without any conditional to find who will end-up going with Anitesh.

//short-circuit.js

Short-circuit.js

The technique used in short-circuit.js is called short circuiting. This is a very popular technique. Node.js uses this technique to select port number.

Exercise

Write a program to print the truth value of a variable without using conditionals.

Ex:
if s = “”, print false
if s = “r”, print true

//truthy.js
// technique used in this solution is also very popular with node developers

setTimeout function

```
SetTimeout(function(){
    console.log("u make me wait, i hate u");
}, 5000);
```

after 5000 millisecond ie 5 second, the first argument, that is a function will be called.

So, "u make me wait, i hate u", will be printed after 5 seconds.

Exercise

Write a program which will print your name after 2 second

What will be the output of the following program?

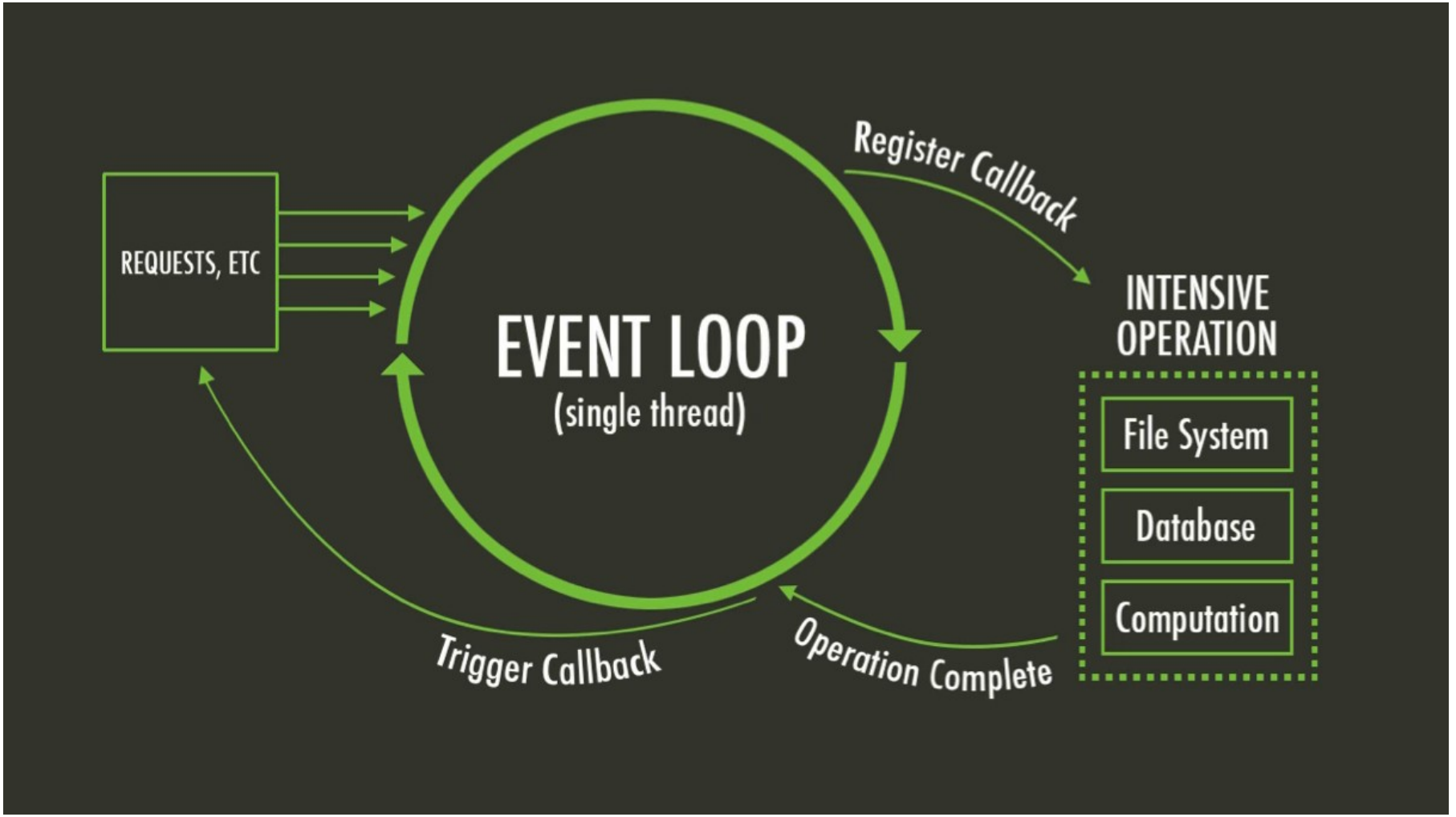
```
Console.log( “a” );
setTimeout(function () {
    Console.log( “c”)
}, 500 );
setTimeout(function () {
    Console.log( “d” )
},  500);
setTimeout(function () {
    Console.log( “e” )
},  500);
Console.log( “b” );
//bulbul.js
```

The answer is

a
b
c
d
e

yeah, it is not(a,c,d,e,b)
// the reason in async nature of js
//bulbul.js

Event Loop



https://www.facebook.com/workspace.mnnit

First callback program

```
function buji(i, done){
  i += 500;
  done(i);
}
var gt = 20;
buji(gt, function(val){
  console.log(val);
});
//fcall.js
```