Copy from register

Set Timeout returns a numeric value.

Calback can also take a value as input and return any operations performed on it

Now suppose you are performing a network call and you do not know the number of arguments (parameters) passed in the callback.

Spread Operator (three dots prefix to the variable)

Var myCb = (…arguments){

Console.log(…arguments)

}

It takes all the data and displays it in a NEW and open array..

Like if a was and array and b was an array it concatenates them..

Var c = […a,…b];

Similarly for objects too.

We could also make a new array by using forEach and push but the problem is that it takes a lot of time for longer arrays/objects.

In ES5 you would use slice for arrays. Or .toString and then split(‘,’)

In ES6

For objects we Use

Var a = Object.assign(b);

This will reference the b object in the new object a;

Can also concatenate 2 arrays in a reference.

But if we want to create a new object or array which is not referenced then we define the data type required at the end (breaking the reference)

Var a = Object.assign(b,c, []);

Old browsers do not run es6 so we have libraries that convert es6 into es5.

JSON.stringify() bhi use kr skte hain.

TO convert object values to values..

Var { name, age } = obj // which is an object having name, age and location as properties.

Var { name, age , …extra} = obj // this saves all the leftover properties in the extra variable which is an array

API Call (network calls)

2 methods used in HTML GET & POST

Data from a API call is usually in JSON (objects & arrays) and XML( old apis ).

Call an API and you get an object.

api.github.com/users/hassan-imran/

api.github.com/users/hassan-imran/followers/ //etc

# Old Method – XHR (returns JSON file and callback)

Then

# JQUERY – library (returns CallBack, promise)

First call jquery library from the internet (script src in absolute path (also called CDN paths) from jquery.com)

$ 🡺 represents the usage of Jquery

$.get(“link”, cb); // get can be called by browser and POST,PUT, cannot be called by browser

Where var cb = (a,b) {

Console.log(a,b); // in a it gives the JSON data (object) and b gives success or error //(promise)

}

This takes place in the form of asynchronous programming (the rest of the program runs while it also listens to the callback at the same time)

But now In JS we have fetch (Almost all browsers but mostly chrome)

Higher order functions

Returns a functions from within a function

Function myFunc () {

Return () => {

Return 123;

}

}

Either call in 2 steps.

Var a = myFunc();

a();

or simply call

myFunc()();

eg multiplier(2)(3); // returns 6

now if we want to use a multiplier with a specific value we use..

var double = multiplier(2);

now when we call double (3).. it doubles 2

shorthand:

function multiplier(x) => return (factor) => return (x\*factor)

Closure (kinda like local vars)

If we store a function in a variable and the same function in another variable..

The value in the function is not overwritten.. but the var holds, stores the parameter passed in the functions in that specific variable.

Closure issue

Var arr = [5,4,3,2,1];

For (var i=0; i<arr.length;i++) {  
 var x = setTimeout

}

Referential transparency

Does not change any value (in the data) (it does the work on the values but does not change it) and return the same data.

This

Lambda functions => fat arrow funcs

Fat arrow function also uses the this of its parent (baap)

In normal fu