﻿Functional Programming in JavaScript

Higher Order Functions:

These functions are values like any other values that is taken as parameters for other functions and these functions are also called object of array.

For example: Filter, Map, Reduce, Sort etc.

Map: Map is another higher order function just like filter. It takes the each item as parameter on callback/directly of the array and we can do something with it for return without changing the original array.

Syntax

array.map(function(currentValue, index, arr), currentValue)

Example 01:

employees: [

{"name":"Bablu Ahmed", "title":"Programmer"},

{"name":"Abdul Kuddus", "title":"Vue JS Expert"},

{"name":"Sakibur Rahman", "title":"Marketing Manager"},

]

Let’s try first using for loop as follows:

var names = [];

for (var i = 0; i<employees.length; i++)

{

names.push(employees[i].name);

}

console.log(names);

output:

["Bablu Ahmed", "Abdul Kuddus", "Sakibur Rahman"]

But we can do exact using Map and so much less code:

var names = animals.map(functions(animal){

return animal.name;

});

In arrow function:

var names = animals.map((animal)=> return animal.name);

Example 02:

var numbers = [4, 9, 16, 25];

function myFunction() {

x = document.getElementById("demo")

x.innerHTML = numbers.map(Math.sqrt);

}

Output:

2,3,4,5

Some other functions:

Split, replace, slice, filter,

Split:

slice:

**The Spread (…)**

**Bind, Call and Apply in JavaScript:**

function foo() {

this.a = 1;

}

foo(); // using function as a regular function. The context is window.

console.log(window.a); // prints "1"

foo.call(window); // explicitly specify execution ctx. The same as just foo() call

var instance = new foo(); // using foo as a constructor

console.log(instance.a); // prints "1"

// actually you can do it without new keyword

var instance = {}; // manually create new object

foo.call(instance); // manually call foo against this object

console.log(instance.a); // prints "1"

However, the code above is not strictly equivalent to the code using new. It omits the concept of prototype, but it's enough for our current task.

**Constructor concept in javascript:**

Any function in JavaScript can act as a constructor when the function is invoked with new operator.

Now, what a constructor does ? it creates/instantiate an object from the constructor function.

It explains the fundamentals very clearly.

What is this ?

when this constructor function is invoked with new, this points to the new object created at that invocation. and in that object we set firtName and lastName (it is the initialization of the new object created).

Now when we add methods to the prototype of the constructor , that is being shared between all the objects created using the constructor function(picture explains it lit bit more)

and regarding your last query "And also in one of the blogs I was studying if the file name is Samplescript.js and if a function is written using the same name inside this like var Samplescript=function(){}, will this function be considered a constructor? Please clarify me this"

any function in JavaScript can act as a constructor when the function is invoked with new operator, and its not the way that blog says.

So please stop reading that blog, the link i provided is a very good starting point