LAB 10

Question 1:

Write the necessary Node script to make this code work for all arrays:

[1,2,3,4,5,6,7,8].even(); // [2,4,6,8]

[1,2,3,4,5,6,7,8].odd(); // [1,3,5,7]

Test your code in Node.JS CLI

//even

Array.prototype.even = function () {

let arr = this;

return arr.filter(num => num % 2 === 0);

}

//odd

Array.prototype.odd = function () {

let arr = this;

return arr.filter(num => num % 2 == 1);

}

const arr = [1, 2, 3, 4, 5, 6, 7, 8];

console.log(arr.even());

console.log(arr.odd());

Question 2:

1. Explain why do we want sometimes to use setImmediate instead of using setTimeout?

setImmediate runs in check phase, setTimeout runs in timer phase. We can use setImmediate whenever we needs to run just before close phase.

1. Explain the difference between process.nextTick and setImmediate?

|  |  |
| --- | --- |
| Process.nextTick | setImmediate |
| It is processed at the starting of the event loop and between each phase of the event loop. | It is processed in the check phase |
| Process.nextTick get executed multiple chances in one iteration, then it has higher priority over than setImmediate.  setImmediate only get executed once in one cycle. | |
| If it is called in a given phase, all the callbacks passed to it will be resolved before the event loop continues. This will block the event loop and create I/O Starvation if it is called recursively. | It's callbacks will not be executed when using recursive for process.nextTick() because of blocking event loop. |
|  | Recursive calls to setImmediate won’t block the event loop, because every recursive call is executed only on the next event loop iteration. |
| API provided natively by Nodejs | Provided by libuv |

1. Does Node.js has window object?

Node.js don’t have window object. Instead Node provides us with global modules and methods that are automatically created for us: module, global, process, buffer, require, setInterval, setTimeout, clearInterval, clearTimeout.