Event Loop JavaScript

JavaScript is a single-threaded programming language. In other words, it can do only one thing at a time.

One request takes more than 2 minutes to execute?
Will the user be kept waiting for ever?

Thankfully,
JavaScript's Event loop architecture comes to the rescue, as it provides the ability to process multiple requests asynchronously.

```
The general algorithm of the engine:

1. While there are tasks:
    - execute them, starting with the oldest task.
2. Sleep until a task appears, then go to 1.
```

Still Confused?

Let’s understand with an example.

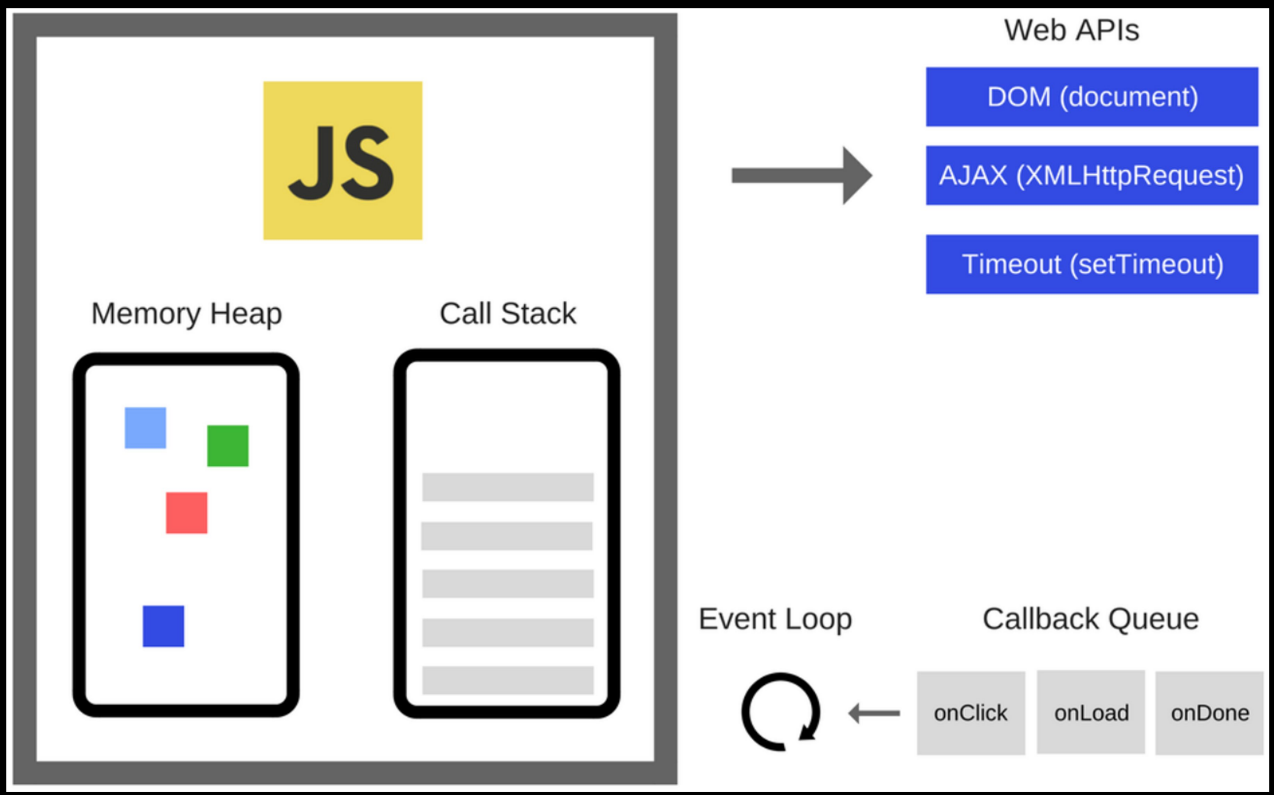
```
const foo = () => console.log("First");
const bar = () => setTimeout(() => console.log("Second"), 500);
const baz = () => console.log("Third");

bar();
foo();
baz();
```

```
Output:
First
Third
Second
```

What just Happened?

To understand that, we need to understand this.



- Memory Heap
- Call Stack
- Web APIs
- Callback Queue

Memory Heap:

Data stored randomly and memory allocated.

Call Stack:

- Call stack is a function which keeps track of all other functions executed in run time.
- Function get popped out of the stack after a function's purpose gets over.

That is,
bar() will come to call stack, it executed and poped out of stack once completed, then, foo() will come to call stack, then, baz().

However, there is a catch. While execution of bar() fn setTimeout() will go to Web API.

Web APIs:

Since the 'setTimeout' function is part of the Web API, it gets moved to the Web API.
Ideally, you would have anticipated that the function is returned back to the Call Stack for execution. However, instead, it adds it to the **Event Queue**.

Event Queue/Callback Queue:

Event Queue is a data structure similar to Stack, which holds the data temporarily and the important thing to note is that the data added first is processed first.

The **function of Event Loop** can simply be explained as **connecting** the Event Queue to the Call Stack.

Event Loop:

Event Loop does the following:

- Checks if the Call Stack is empty, i.e., if all the functions have completed their execution and they have been popped off the Call Stack.
- Once the Call Stack is empty, it moves the first item from the Event Queue to the Call Stack.

That's all there is to know about Event Loop. Hope the explanation with illustrations provided a way to understand Event Loop in a simple manner.

FOLLOW FOR MORE SUCH CONTENT

DID YOU FIND THIS POST USEFUL?

Let me know in the Comments below!