Seminar Report on Asynchronous JavaScript

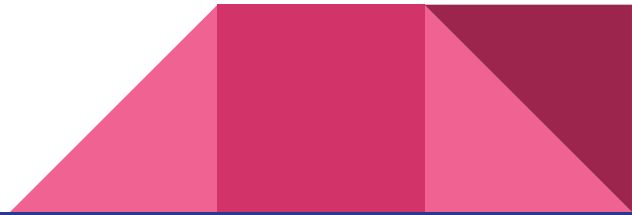
Name: Pratham Dogra
Roll No: 04015002817

Who am I?

- Hi, I am Pratham Dogra.
- B.Tech from MSIT, New Delhi.
- I am a Web Developer
- Former Intern at Paisabazaar.

What is Asynchronous or Asynchrony?

Asynchrony in computer programming refers to the occurrence of events independently of the main program flow and ways to deal with such events.



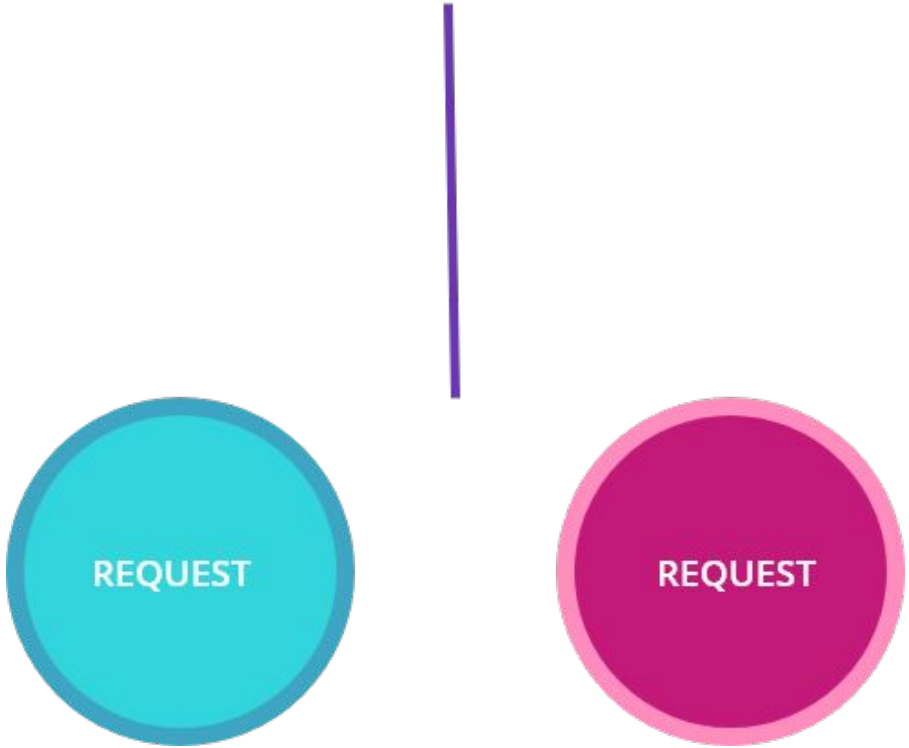

Asynchronous vs Synchronous

Is JavaScript Synchronous or Asynchronous?

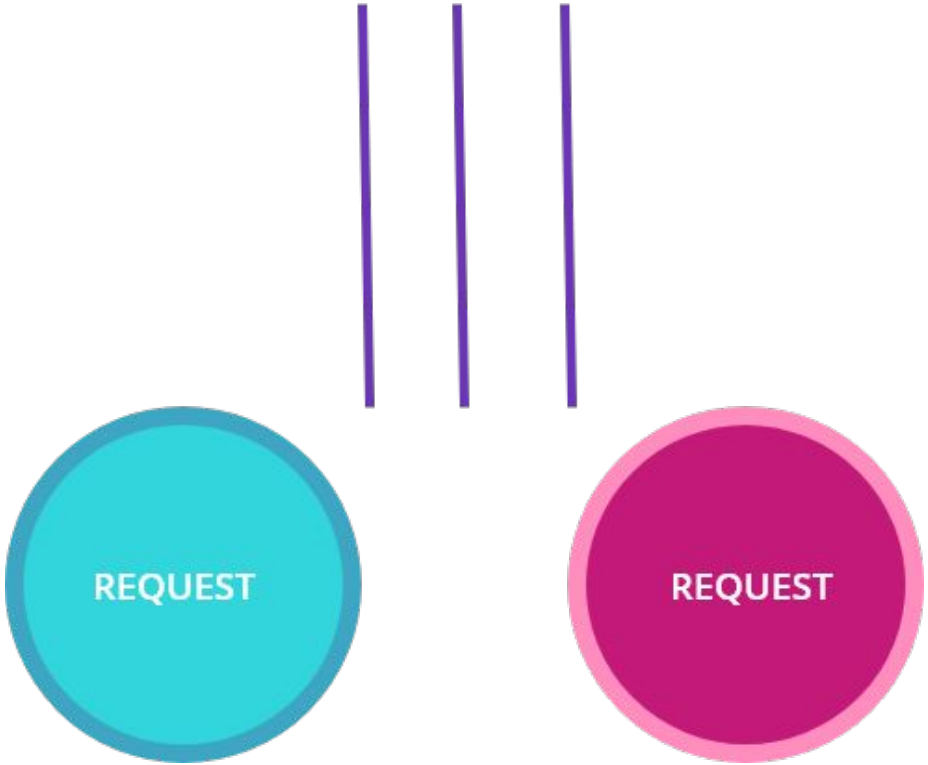
It is Synchronous

- Single threaded.
- It works sequentially.

SINGLE THREAD

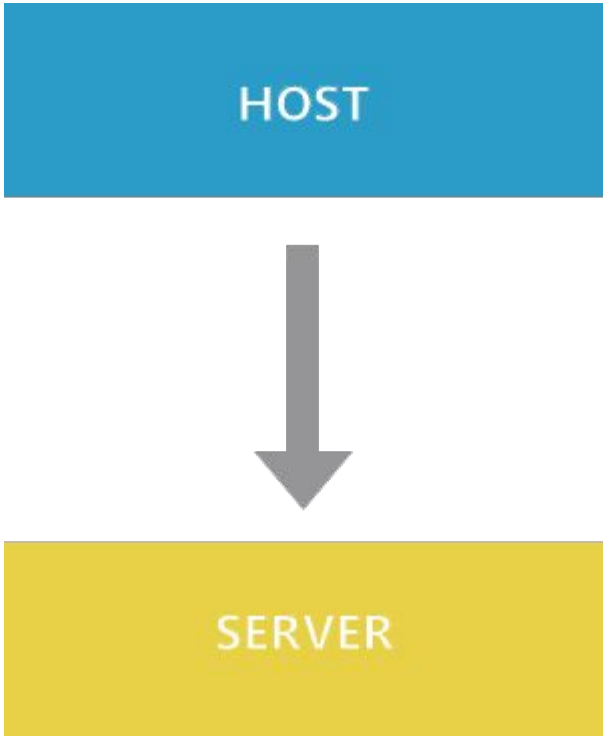


MULTI THREAD



Problem due to Single Thread

```
function delayBySeconds(sec){
    let start = now = Date.now()
    while (now-start < (sec*1000)){
        now = Date.now();
    }

    console.log("Hello World");
}

delayBySeconds(5);
```



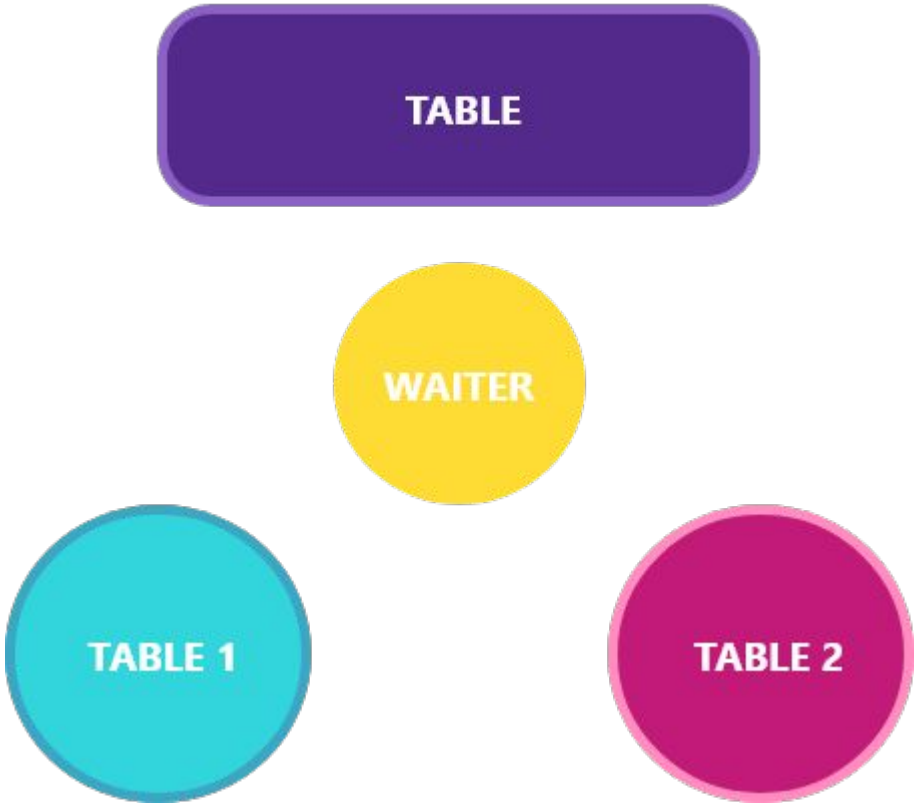

```
makeNetworkRequest()

while(!networkRequestReturned()){

  // Just Keep Looping

}

//runs after request returned
```

How does it handle multiple requests in parallel?

Asynchronous non-blocking I/O Model

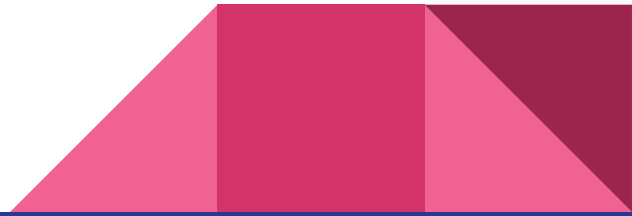
What does that mean?

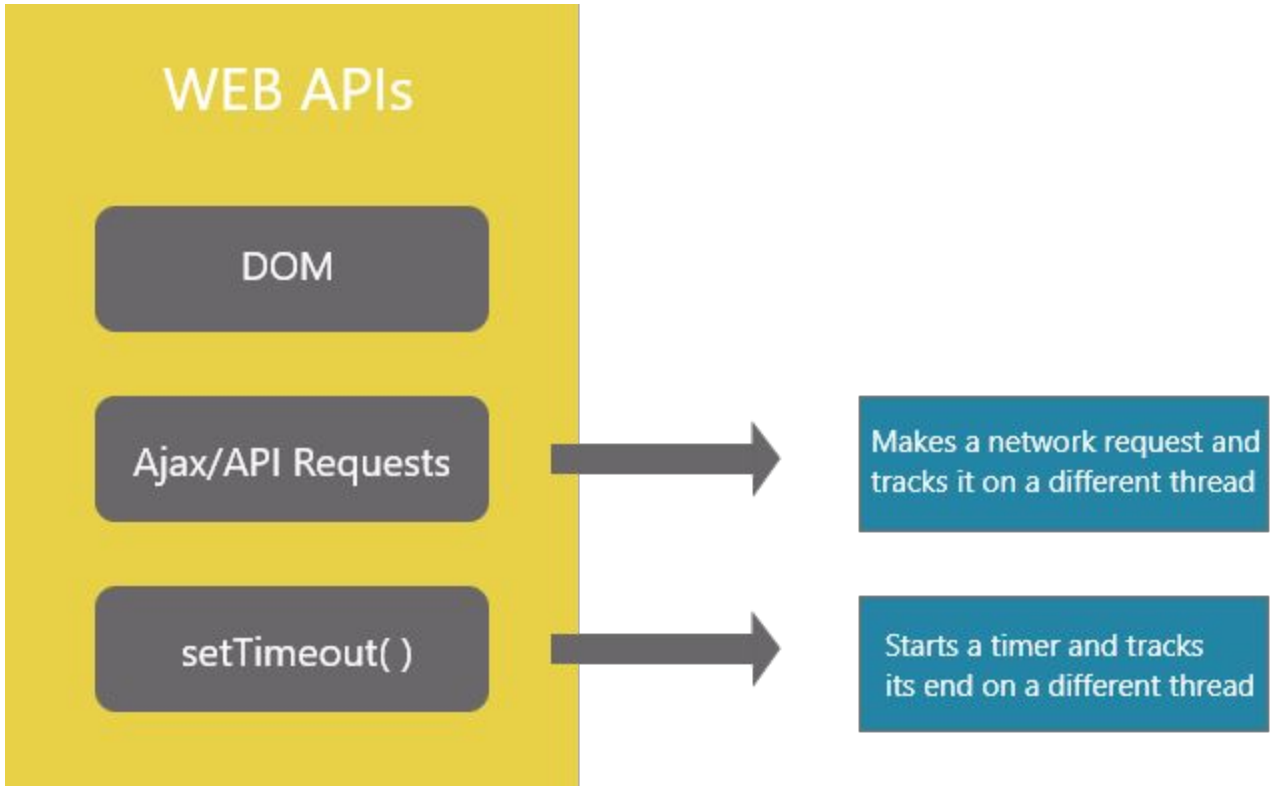
Restaurant Example

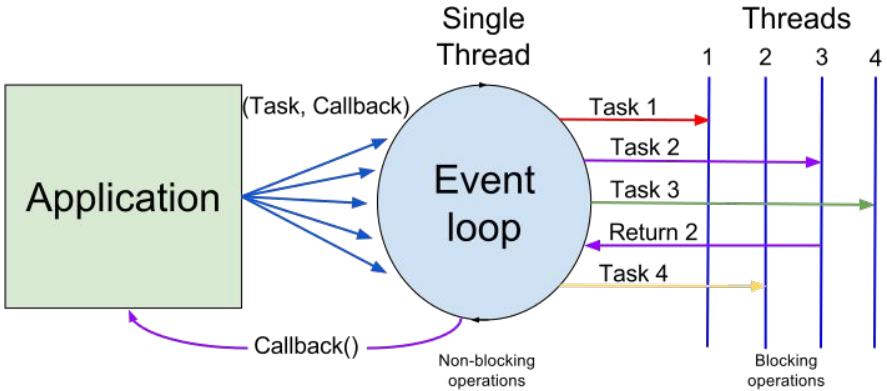


- While the execution of JavaScript is blocking, I/O operations are not.

- I/O operations can be fetching data over the internet with Ajax or REST, querying data from a database or accessing the filesystem with the Node.js “fs” module.



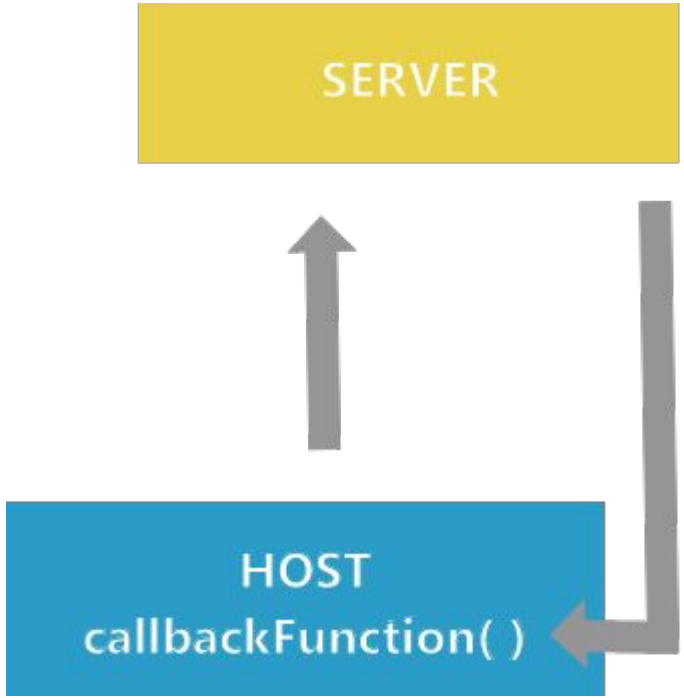

WEB APIs
DOM
Ajax/API Requests
setTimeout( )
Makes a network request and tracks it on a different thread
Starts a timer and tracks its end on a different thread



- Callbacks
- Promises
- Async/Await

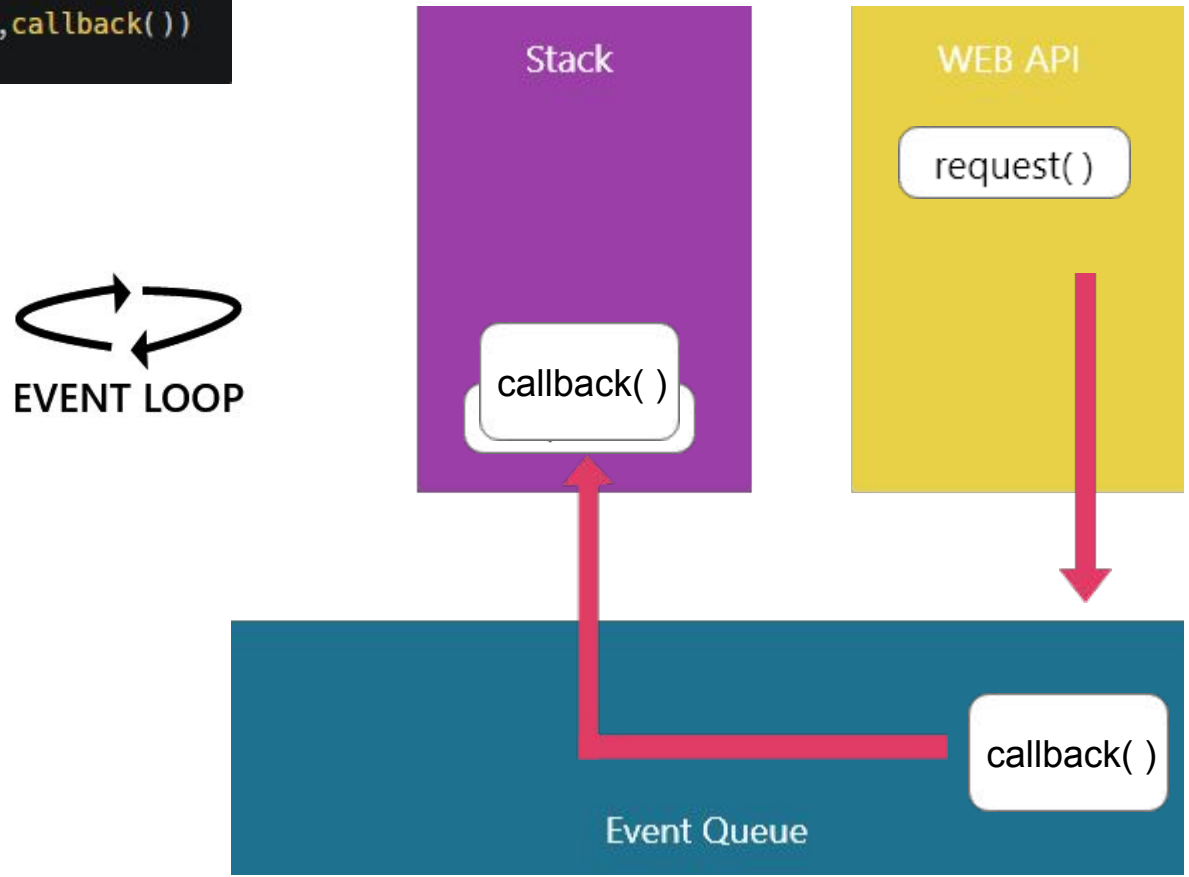
Callback functions

```
const request = require('request');
request('https://www.somepage.com', function (error, response, body) {
  if(error){
    // Handle error.
  }
  else {
    // Successful, do something with the result.
  }
});
```



The Event Loop

```
request('http://www.somepage.com',callback())
```



Promises

```
const axios = require('axios');
axios.get('http://www.somepage.com')
.then(function (response) { // Reponse being the result of the first request
    // Returns another promise to the next .then(..) in the chain
    return axios.get(`http://www.somepage.com/${response.someValue}`);
})
.then(function response { // Reponse being the result of the second request
    // Handle response
})
.catch(function (error) {
    // Handle error.
});
```

Async / Await

```
const request = require("request");

const apiCall = async () => {
  const result = await request("https://jsonplaceholder.typicode.com/posts/1");
  console.log(result);
};
console.log('Hello world');
// Some other Code
apiCall();
```

Questions?

Thank You

Let’s connect?

 /prathamdogra

 /PrathamDogra

 /@DograPratham