AJAX

Rest Operator:

The idea was if you have multiple arguments to be passed inside a function and we don't know how many elements are how many arguments we are going to pass.

We are passing 4 elements, and we don't know how many elements will be passed and neither can we have multiple variables in the add() function. So, using the rest operator so that based on values passed, we can make an array out of it. So, we need to add '...' and give it a name.

```
function add() {
console.log(args)
let sum = 0
for (let index = 0; index < args.length; index++) {
sum += args[index]
}
console.log(sum)
}
add (10, 88, 77, 99) // not sure how many arguments
```

This displays me an array like structure and we can loop over it and calculate the sum of all the elements in the array. We can use the forEach loop as well for all the elements of the array.

Spread Operator:

We have obj and obj2 with 2 key-values each.

```
const obj = {
    keyA: "Value1",
    keyB: "Value2"
}
const obj2 = {
    keyC: "Value3",
    keyD: "Value4",
}
```

Now if we want to create another object that has all the values of obj and obj2. So, we will add, (...obj, ...obj2) to the joinedObj.

```
const joinedObj = {
    keyE: "Valeu5",
    ...obj,
    ...obj2
}
```

So, essentially, we are taking the keys and joining them. At the run time it would be as,

```
const joinedObj = {

keyE: "Valeu5",
    keyC: "Value3",
    keyD: "Value4",
    keyA: "Value1",
    keyB: "Value2"
}
```

We are spreading all the key inside another object.

```
console.log(joinedObj)
```

If we have another key 'nested' and this is another object which has a key 'abc': 'ASD'.

```
const obj2 = {
    keyC: "Value3",
    keyD: "Value4",
     nested: {
          abc: "ASD"
     }
}
```

When we use the spread operator, it will copy or clone, meaning it would create a different memory space for all key-value pairs. So, obj would be referring to its own sets of key-value and the same with obj2. When we join the two objects using the spread, this also created its own sets of key-value pair. If we have nested object inside obj2, then the key-value pair would be copied, but it will only create proper memory space only for the first level key-value pairs.

Ex: 01

```
const arr1 = [10, 20, 30]
const arr2 = [90, 120, 200]
const arr3 = [...arr1, ...arr2]
```

Output:

```
[10, 20, 30, 90, 120, 200]
```

Ex: 01

```
const arr1 = [10, 20, 30]
const arr2 = [90, 120, 200]
const arr3 = [...arr1, ...arr2]
const arr4 = [...arr2, arr1[2]]
```

We are not putting the spread operator as we only want to the 3rd element from arr1.

Take SS at 45:44

All the websites have 2-parts; server & client. When you go to a particular website (say Flipkart), all the things are fetched from the database which is in the server. So, we open website, that entire request (resource location) is being fetched by the browser and the server somewhere in the internet is listening to that request.

So, depending on the address you are typing it will call the server and ask the data to be shared to the browser. This is essentially a client server model.

HTTP Model:

Take SS at 51:55

When we want to get a particular request, the waiter would go to chef and inform him. This is similar to how http communication happens. The waiter is the medium. So the actually HTTP protocol when you send a resource or location, is done by http. The moment we enter a website, the browser invokes or calls the server with a get request. Another way is to post; you can use this to create data.

if we ask the chef in the restaurant to prepare a new dish, then the waiter will take the information with the chef. You are sharing data with the server to create something new.

There are two type of requests:

- Get
- Post

GET	POST
In GET method, values are visible in the URL.	In POST method, values are not visible in the URL.
GET has a limitation on the length of the values, generally 255 characters.	POST has no limitation on the length of the values since they are submitted via the body of HTTP.
GET performs are better compared to POST because of the simple nature of appending the values in the URL.	It has lower performance as compared to GET method because of time spent in including POST values to HTTP body.
This method supports only string data types.	This method supports different data types, such as string, numeric, etc.
GET results can be bookmarked.	POST results cannot be bookmarked.
GET request is often cacheable.	The POST request is hardly cacheable.
GET Parameters remain in web browser history.	Parameters are not saved in web browser history.

Synchronous

Synchronous data transmission is a data transfer method in which is a continuous stream of data signals accompanied by timing signals. It helps to ensure that the transmitter and the receiver are synchronized with each other.

This communication methods is mostly used when large amounts of data needs to be transferred from one location to the other.

Asynchronous

Asynchronous Transmission is also known as start/stop transmission, sends data from the sender to the receiver using the flow control method. It does not use a clock to synchronize data between the source and destination.

Key Differences:

- Synchronous is a data transfer method in which a continuous stream of data signals is accompanied by timing signals whereas Asynchronous data transmission is a data transfer method in which the sender and the receiver use the flow control method.
- In, synchronous transmission method users need to wait until it sending finishes before getting a response from the server. On the contrary, Asynchronous transmission method users do not have to wait until sending completes before receiving a response from the server.
- Synchronous Transmission sends data in the form of blocks or frames while Asynchronous Transmission send data in the form of character or byte.
- Synchronous Transmission is fast. On the other hand, Asynchronous transmission method is slow.
- Synchronous Transmission is costly whereas Asynchronous Transmission is economical.

How Synchronous Transmission works?

- Separate clocking lines used when the distance between the data terminal equipment (DTE) and data communications equipment (DCE) is short.
- This method uses a clocking electrical system at both transmitting and receiving stations. This ensures that the communication process is synchronized.
- Devices that communicate with each other Synchronously use either separate clocking channels.

How Asynchronous Transmission works?

- Asynchronous communication is eased by two bits, which is known as start bit as '0' and stop bit as '1.'
- You need to send '0' bit to start the communication & '1' bit to stop the Transmission.
- There is a time delay between the communication of two bytes.
- The transmitter and receiver may be function at different clock frequencies.

Synchronous	Asynchronous
Synchronous data transmission is a data transfer method in which a continuous stream of data signals is accompanied by timing signals.	Asynchronous data transmission is a data transfer method in which the sender and the receiver use the flow control method.
Synchronous handler do not return until it finishes processing the HTTP request for which it is called.	Asynchronous handler helps you to run a process independently of sending a response to the user.
Users need to wait until it sending finishes before getting a response from the server.	Users do not have to wait until sending completes before receiving a response from the server.
In this transmission method, blocks of characters are transmitted at high- speed on the transmission line.	In asynchronous transmission, the information should be transmitted character by character.
It sends data in the form of blocks or frames.	Data is sent in the form of character or byte.
Synchronous Transmission is fast.	Asynchronous transmission method is slow.
Synchronous Transmission is costly.	Asynchronous Transmission is economical.
The time interval of transmission is constant.	The time interval of transmission is random.

Synchronous Transmission does not have a gap between data.	In asynchronous transmission, there is a gap between data.
Synchronous postback renders the entire page of any postback.	Asynchronous postback renders only needed part of the page.
It does not need any local storage at the terminal end.	It requires local buffer storages at the two ends of the line to assemble blocks.
Synchronous replication should be performed when reliable and long-term storage is required.	Asynchronous replication an ideal for projects that span across long distances and have a very minimal budget.
This method does not need any synchronized clocks.	This method requires accurately synchronized clocks at both ends.
You can use it in the low-speed communication like the connection of a terminal to a computer.	You can use it in high-speed applications like the Transmission of data from one computer to another.
The voice-band and broad-band channels are mostly used in the Synchronous Transmission.	The voice-band channels that have a narrow type in the used asynchronous transfer.

Advantages of Synchronous Transmission

- It helps you to transfer a large amount of data.
- It offers real-time communication between connected devices.
- Each byte is transmitted without a gap between the next byte.
- It also reduces time timing errors.

Advantages of Asynchronous Transmission

Here are pros/benefits of Asynchronous Transmission:

- This is a highly flexible method of data transmission.
- Synchronization between the receiver and transmitter is unnecessary.
- It helps you to transmit signals from the sources which have different bit rates.

- The Transmission can resume as soon as the data byte transmission is available.
- This mode of Transmission is easy for implementation.

Disadvantages of ASynchronous Transmission

Here are cons/drawbacks of Asynchronous Transmission

- In Asynchronous Transmission, additional bits called start and stop bits are required to be used.
- The timing error may take place as it is difficult to determine synchronicity.
- It has a slower transmission rate.
- May create false recognition of these bits because of noise on the channel.

Disadvantages of Synchronous Transmission

Here are the cons/drawbacks of Synchronous Transmission.

- The accuracy of the received data depends on the receiver's ability to count the received bits accurately.
- The transmitter and receiver need to operate simultaneously with the same clock frequency.