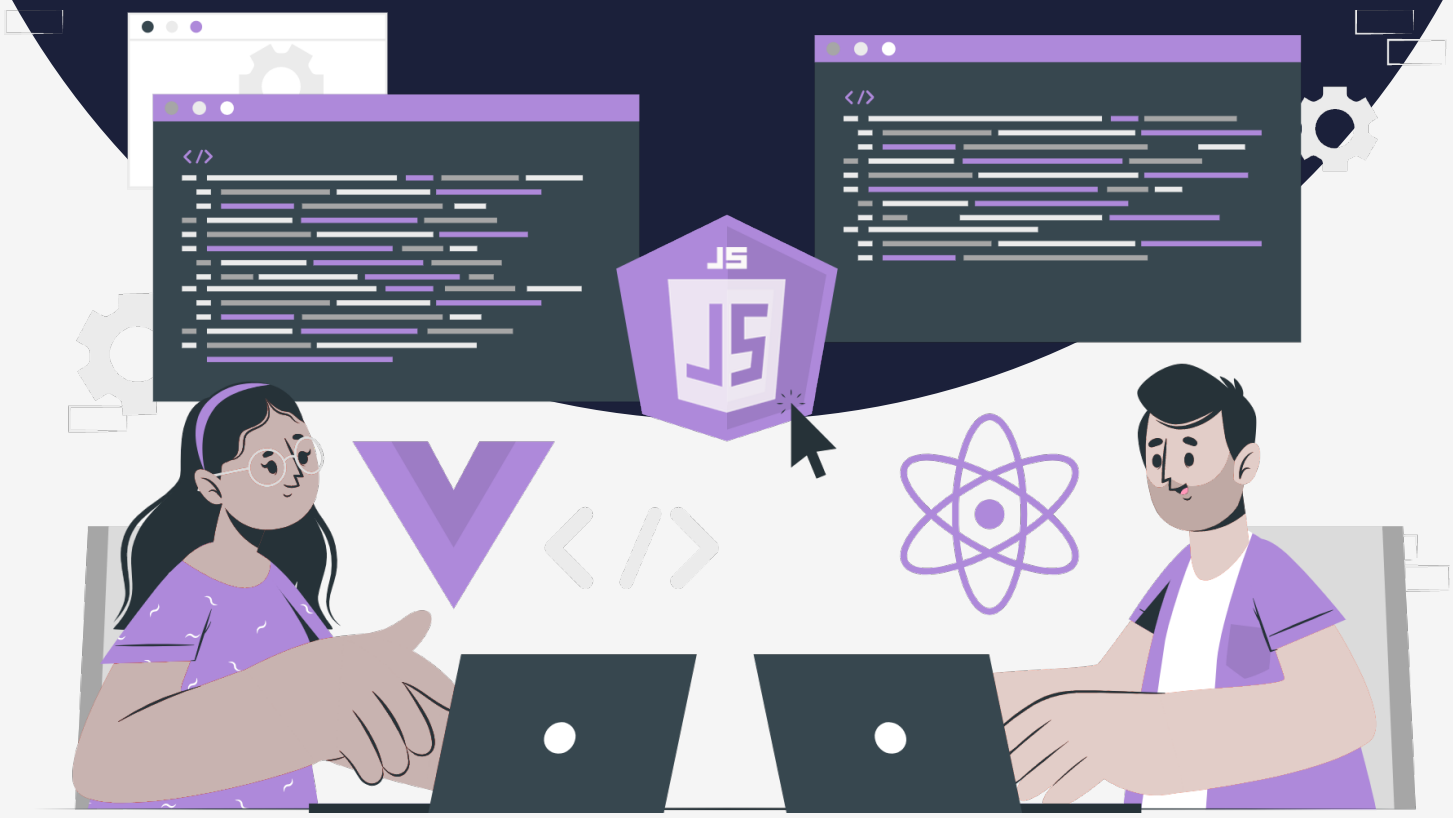
Lesson:

OS Module



List of content:

1. Useful commands to fetch information from OS
2. Examples and output

The OS module in Node.js is utilized to retrieve details about the operating system of a computer. It offers methods for communication with the computer's operating system, including retrieving the hostname and the amount of available system memory in bytes.

Let us have a look at the important commands:

```
// Include os module and create its object
var os = require('os');

// for cpu architecture
console.log("CPU architecture: " + os.arch());

// For fetching the amount of free system memory in bytes
console.log("Free memory: " + os.freemem());

// For fetching the total amount of system memory in bytes
console.log("Total memory: " + os.totalmem());

// For fetching the list of network interfaces
console.log('List of network Interfaces: ' + os.networkInterfaces());

// It returns the operating systems default directory for temp files.
console.log('OS default directory for temp files : ' + os.tmpdir ());
```

```
CPU architecture: x64
Free memory: 11969703936
Total memory: 21243420672
List of network Interfaces: [object Object]
OS default directory for temp files : C:\Users\ACER\AppData\Local\Temp
```

1. The os.arch() method in Node.js is used to retrieve the architecture of the computer's CPU. It returns a string that indicates the processor's architecture, eg. x32, x64 etc.

2. The os.freemem() method in Node.js is used to retrieve the amount of free system memory in bytes. It returns a number that indicates the number of bytes of free memory available in the system. This will output the amount of free memory available in the system, in bytes.

3. The os.totalmem() method in Node.js is used to retrieve the total amount of system memory in bytes. It returns a number that indicates the total number of bytes of memory available in the system. This will output the total amount of memory available in the system, in bytes.

4. The os.networkInterfaces() method in Node.js is used to retrieve information about the network interfaces available on the system. It returns an object containing information about each network interface, including its name, address family (IPv4 or IPv6), and IP addresses.

5. The os.tmpdir() method in Node.js is used to retrieve the default directory path for temporary files on the current operating system. It returns a string that represents the path of the default directory for temporary files.

Let us look at a few more important and frequently used commands:

```
// Include os module and create its object
var os = require('os');

// For fetching the endianness of system
console.log("Endianness of system: " + os.endianness());

// For fetching the hostname of system
console.log("Hostname: " + os.hostname());

// For fetching the operating system name
console.log("Operating system name: " + os.type());

// For fetching the platform of os
console.log('operating system platform: ' + os.platform());

// For fetching the operating systems release.
console.log('OS release : ' + os.release());
```

```
Endianness of system: LE
Hostname: alka-5564
Operating system name: Windows_NT
operating system platform: win32
OS release : 10.0.22621
```

1. The os.endianness() method in Node.js is used to retrieve the endianness of the computer's CPU. Endianness refers to the order in which bytes are stored in memory. A little-endian CPU stores the least significant byte first, while a big-endian CPU stores the most significant byte first.

The os.endianness() method returns a string that indicates the endianness of the CPU. It will return either "BE" (big-endian) or "LE" (little-endian).

2. The os.hostname() method in Node.js is used to retrieve the hostname of the computer on which the Node.js process is running. It returns a string that represents the hostname of the computer. This will output the hostname of the computer on which the Node.js process is running. The hostname is typically a name that is used to identify the computer on a network.

3. The os.type() method in Node.js is used to retrieve the operating system name on which the Node.js process is running. It returns a string that represents the name of the operating system.This will output the name of the operating system on which the Node.js process is running, such as "Windows_NT", "Linux", or "Darwin" (for macOS). The output may vary depending on the operating system being used.

4. The os.platform() method in Node.js is used to retrieve the platform on which the Node.js process is running. It returns a string that represents the platform, such as "win32", "linux", or "darwin" (for macOS). This will output the platform on which the Node.js process is running, such as "win32", "linux", or "darwin". The output may vary depending on the platform being used.

5. The os.release() method in Node.js is used to retrieve the operating system release on which the Node.js process is running. It returns a string that represents the release of the operating system. This will output the release of the operating system on which the Node.js process is running. The output may vary depending on the operating system being used.