SUNSTONE

PLACEMENT
READINESS

Backend Development using Node.JS

Placement Readiness

"Your chance to level up — every interview is a new skill unlocked!"

Interview Dynamics

1. Infosys – 2023

Question: What is Node.js and how is it different from traditional server-side platforms?

Smart Answer:

Node.js is an open-source, cross-platform runtime environment built on Chrome's V8 JavaScript engine. It allows you to run JavaScript on the server side. Unlike traditional server-side platforms like PHP or Java, Node.js uses a single-threaded, non-blocking, event-driven model. This makes it highly efficient and suitable for handling concurrent I/O operations.

Example:

Node.js can handle thousands of concurrent requests with minimal thread management, which is ideal for building APIs and real-time apps like chat applications.

2. Wipro – 2022

Question: How do you install Node.js and npm on a system?

Smart Answer:

You can download the Node.js installer from the official website nodejs.org. It includes npm (Node Package Manager) by default. After installation, you can verify the installation using the following terminal commands:

node -v to check Node.js version

npm -v to check npm version

3. TCS – 2024

Question: Explain a basic "Hello World" application in Node.js.

Smart Answer:

To create a basic "Hello World" server using Node.js, use the built-in http module. Here's a minimal example:

```
const http = require('http');

const server = http.createServer((req, res) => {

  res.write('Hello World');

  res.end();

});

server.listen(3000);
```

This code creates a server that responds with "Hello World" on port 3000.

4. Capgemini – 2023

Question: Name some core modules in Node.js and explain the fs module.

Smart Answer:

Some commonly used core modules in Node.js include: fs, http, path, os, and events.

The fs module allows interaction with the file system such as reading, writing, and deleting files.

Example:

```
const fs = require('fs');
fs.readFile('example.txt', 'utf8', (err, data) => {
  if (!err) console.log(data);
});
```

5. Accenture – 2024

Question: How do you create a basic web server using the http module in Node.js?

Smart Answer:

You can use http.createServer() to create a server and listen on a specific port.

Example:

```
const http = require('http');
http.createServer((req, res) => {
  res.end('Server is running');
}).listen(3000);
```

6. IBM – 2022

Question: What is the purpose of the path module in Node.js?

Smart Answer:

The path module provides utilities for working with file and directory paths in a cross-platform way.

Example:

```
const path = require('path');
console.log(path.join(__dirname, 'folder', 'file.txt'));
```

This ensures correct path formatting regardless of OS.

7. Cognizant – 2023

Question: What is asynchronous programming in Node.js?

Smart Answer:

Asynchronous programming allows operations (like file I/O or network requests) to be executed without blocking the main execution thread. Node.js uses callbacks, promises, and async/await for handling async code.

8. Mindtree – 2024

Question: Explain how callbacks work in Node.js.

Smart Answer:

A callback is a function passed to another function to be executed later, once a task is completed.

Example:

```
fs.readFile('data.txt', 'utf8', function(err, data) {
  if (!err) console.log(data);
});
```

Here, the function is called when the file is finished reading.

9. HCL – 2023

Question: What is a Promise in Node.js and how is it different from callbacks?

Smart Answer:

A Promise is an object representing the eventual completion or failure of an asynchronous operation. Unlike callbacks, Promises can be chained and help avoid "callback hell."

Example:

```
fs.promises.readFile('data.txt', 'utf8')
  .then(data => console.log(data))
  .catch(err => console.error(err));
```

10. LTI – 2024

Question: What is async/await and how does it simplify asynchronous code?

Smart Answer:

async/await is syntactic sugar over Promises that makes asynchronous code easier to read and write, similar to synchronous code.

Example:

```
async function readFileData() {
  try {
    const data = await fs.promises.readFile('file.txt', 'utf8');
    console.log(data);
  } catch (err) {
    console.error(err);
  }
}
readFileData();
```

11. Tech Mahindra – 2023

Question: What is the Event Loop in Node.js?

Smart Answer:

The Event Loop is a core part of Node.js that enables non-blocking I/O operations by offloading operations to the system kernel and using a single-threaded loop to check for and execute callback functions.

12. Deloitte – 2022

Question: Explain event-driven architecture in Node.js.

Smart Answer:

Node.js applications are designed around events. Components emit events, and listeners handle them. This is useful for managing asynchronous operations efficiently.

Example:

```
const EventEmitter = require('events');
const emitter = new EventEmitter();
emitter.on('start', () => {
  console.log('Started!');
});

emitter.emit('start');
```