SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NAAC-UGC with 'A' Grade

Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

Course Code and Name : 19TS601 FULL STACK DEVELOPMENT

Unit 3 : NODEJS AND EXPRESS

Topic : Node.js basics

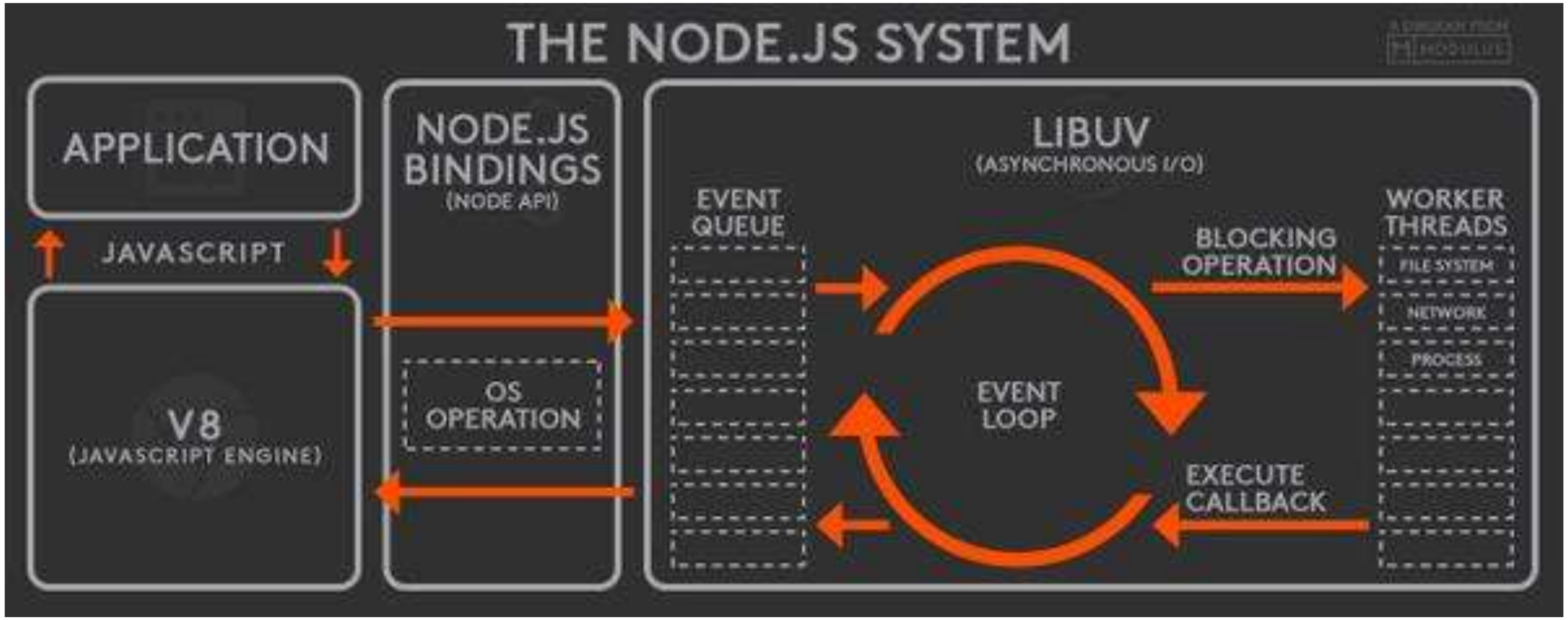
NODE.JS BASICS

- Node.js was developed by Ryan Dahl in 2009
- Node.js is an open source server framework
- Node.js allows you to run JavaScript on the server
- Node.js runs on various platforms
- Node.js uses asynchronous programming
- Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine)
- Node.js = Runtime Environment + JavaScript Library

Definition of Node.js

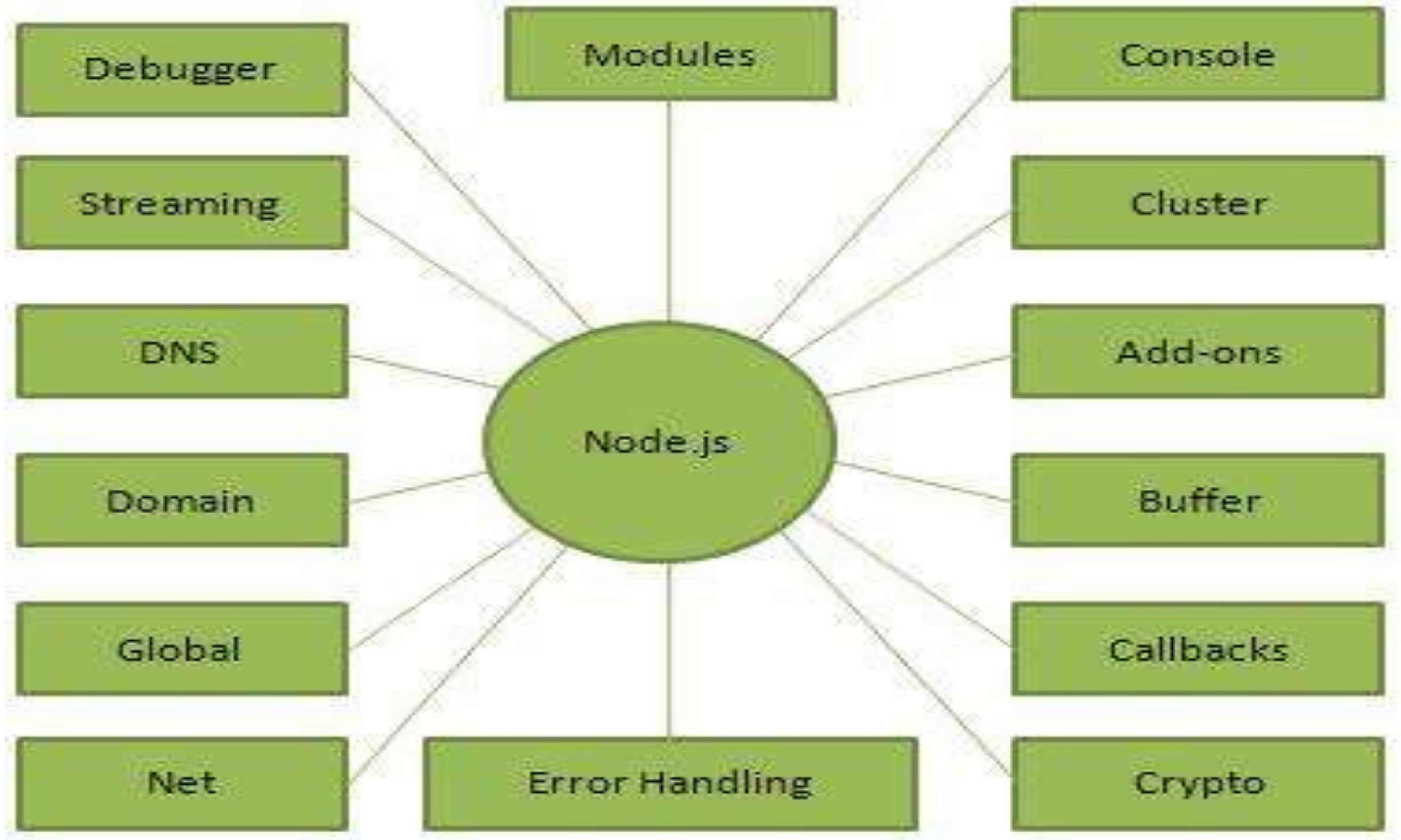
- *Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications.*
- *Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.*

Architecture



- A common task for a Web server can be to open a file on the server and return the content to the client.
- For example php, asp or jsp handles a file request in the following sequence:
- Sends the task to the computer's file system.
- Waits while the file system opens and reads the file.
- Returns the content to the client.
- Ready to handle the next request.

Components of Node.js



Whereas, Node.js handles a file request:

- Sends the task to the computer's file system.
- Ready to handle the next request.
- When the file system has opened and read the file, the server returns the content to the client.
- Node.js eliminates the waiting, and simply continues with the next request.
- Node.js runs single-threaded, non-blocking, asynchronously programming, which is very memory efficient.

Node.js can perform

- Node.js can generate dynamic page content
- Node.js can create, open, read, write, delete, and close files on the server
- Node.js can collect form data
- Node.js can add, delete, modify data in your database
- Download and install node.js from **https://nodejs.org.**

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

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World!');
}).listen(8080);
// To initiate the nodejs code execute in command line mode:
// node example1.js
// To execute in a browser:
// http://localhost:8080/example1.js
```

Basic NodeJS Concepts

- **1. Modules in NodeJS**
- **2. The Event Loop**
- **3. Asynchronous Programming**

Modules in NodeJS

- NodeJS is built around the concept of modules.
- Modules in NodeJS are reusable pieces of code that can be imported into your application.
- These can be built-in modules (like fs for file system operations, http for HTTP server, etc.) or external packages installed using NPM.

Common NodeJS Modules

- **HTTP Module:**
- The http module is used to create web servers.
- It allows you to handle requests and send responses.
- **FS (File System) Module**:
- The fs module provides an API to interact with the file system.
- It can be used to read and write files, check for file existence, etc.
- **Path Module:**
- The path module helps in handling and transforming file paths.
- It makes working with file systems easier and more cross-platform.

- **Event Module**: The events module allows objects to emit and listen to events, which helps in writing event-driven applications.
- **Express Framework:** While NodeJS provides basic capabilities, many developers use the Express framework, which simplifies routing, middleware integration, and HTTP request handling.

The Event Loop

- NodeJS operates on a single-threaded, event-driven model.
- It uses an event loop to handle asynchronous operations.
- The event loop is a process that constantly checks if any asynchronous task (such as reading a file or making an HTTP request) has been completed and then invokes the appropriate callback function.
- This allows NodeJS to handle many operations concurrently without blocking the main thread, making it efficient for I/O-heavy applications like web servers.

Asynchronous Programming

- In NodeJS, many operations, such as reading files or accessing databases, are performed asynchronously.
- This means that the program doesn’t wait for these operations to complete before moving on to the next one.
- Instead, it continues execution and provides a callback function that will be invoked once the operation finishes.

Advantages of Using NodeJS

- **High Performance:** NodeJS is optimized for performance due to its non-blocking I/O model and V8 engine, making it highly suitable for handling real-time applications and large-scale systems.
- **Scalable:** With its event-driven architecture, NodeJS can handle a large number of concurrent connections, making it highly scalable.
- **Cross-Platform:** NodeJS is cross-platform, meaning it can run on various operating systems like Windows, macOS, and Linux.
- **Active Community:** NodeJS has a large and active community that constantly contributes to its growth. This results in a vast array of open-source libraries and tools that can be easily integrated into your application.

Node.js Module

- In Node.js Application, a Module can be considered as a block of code that provide a simple or complex functionality that can communicate with external application.
- Modules can be organized in a single file or a collection of multiple files/folders.
- Almost all programmers prefer modules because of their reusability throughout the application and ability to reduce the complexity of code into smaller pieces.
- Nodejs uses the *CommonJS Module standard* implementation in its module ecosystem.

- **Types of Modules:** In Nodejs, there is 3 type of modules namely
- Core Modules
- Local Modules
- Third-Party Modules

Core Modules

- These are built-in modules that come with the Node.js installation, requiring no additional installation.
- They provide essential functionalities for various tasks. Examples include:
- http: For creating HTTP servers.
- fs: For file system operations.
- url: For URL parsing and manipulation.
- To use a core module, the require function is employed, such as:

- javascript
- const http = require('http');

Local Modules

- These are custom modules created by developers within their applications.
- They can encapsulate specific functionalities relevant to the application.
- To create a local module, you define your functions and export them using the exports object. For example:

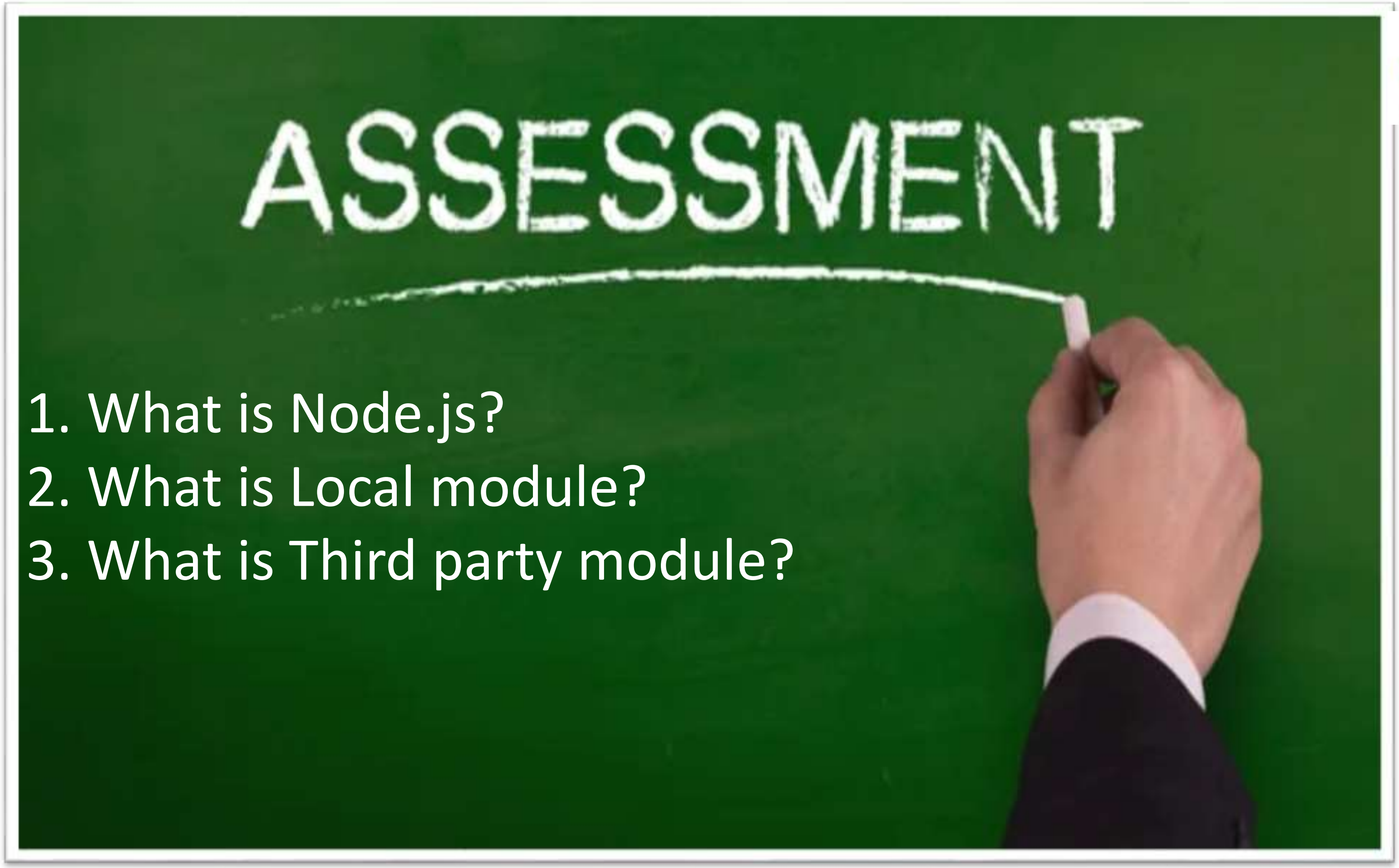
```javascript
// sum.js
exports.add = function(n, m) { return n + m; };

// index.js
const sum = require('./sum');
console.log("Sum of 10 and 20 is ", sum.add(10, 20));
```

Third-Party Modules

- These modules are developed by the community and can be installed via npm (Node Package Manager).
- They extend the functionality of Node.js applications with additional features. Popular examples include:
- express: A web application framework.
- mongoose: An ODM (Object Data Modeling) library for MongoDB.
- To install a third-party module, you would typically run:

bash

- npm install express



1. What is Node.js?
2. What is Local module?
3. What is Third party module?

Text Book:

1.Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node, Vasan Subramanian, A Press Publisher, 2019.

Reference:

David Flanagan, “Java Script: The Definitive Guide”, O’Reilly Media, Inc, 7 th Edition, 2020

2. Matt Frisbie, “Professional JavaScript for Web Developers” Wiley Publishing, Inc, 4th Edition, ISBN: 978-1-119-36656-0, 2019

Thank You!