Date ___/___/___

Question Nodejs: Introduction, Features, execution architecture.

answer
→

⊛ introduction:-

Nodejs is an open-source, cross-platform Javascript runtime environment that allows developers to execute Javascript code outside of a web browser. It uses the V8 Javascript engine from google chrome to interpret and run Javascript code on the server-side providing a powerful and efficient platform for building scalable network application.

⊛ features:-

→ Asychronous and Non-blocking :-

Node.JS employs an event-driven, non-blocking I/O model enabling it to handle a large number of concurrent connection efficiently without blocking the execution of other tasks

→ Lightweight and Fast.

Date __/__/____

Saathi

→ NPM (Node package manager)

Node.js has a vast ecosystem of third party libraries and modules available through NPM, allowing developers to easily integrate various functionalities into their application.

→ Server-side and client-side:

Node.js is commonly used on the server-side to build web servers and APIs but, it can also be used on the client side to manage build process and package management for web applications.

⊛ Execution :-

Node.js application are executed using the 'node' command followed by the name of the javascript file containing the application code for example.

node myapp.js

Date ___/___/___

once executed, the Node.js runtime starts interpreting the code and the application can respond to incoming requests and perform various tasks ~~asy~~ asynchronously.

Architecture of Node.js

Node.js follows a single-thread event-driven architecture with a non-blocking I/O model when a Node.js application starts it initializes an event loop that listens for incoming events such as HTTP requests or file system operations. when an event is triggered Node.js asynchronously processes it, allowing the application to continue handling other event or tasks without for the first one to complete.

Node.js also employs the concept of callbacks and uses an event driven programming paradigm, when functions are passed as argument and called when certain events occur this approach allows for efficient handling of multiple concurrent operation without the need for multiple threads.

Date ___/___/___

Saathi

Question = Note on modules with example

answer → module are a fundamental concept that allows you to organize your code into reusable and maintable component. Modules enable you to break down your application into smaller, manageable pieces, making it easier to maintain and scale. Node.js uses the CommonJS module system where each file is treated as a module with its own scope, and you can export and import functionalities between modules.

⊛ Creating modules :-

To create a module in Node.js you define your functions classes or variable in a file and use the 'exports' object to make them accessible to other parts of your application.

example :-

Date __/__/__

circle.js

```
const calculateCircleArea = (radius) =>
{
    return Math.PI * radius * radius;
};
exports.calculateCircleArea = calculateCircleArea;
```

* using the module:-

app.js

```
const circleModule = require('./circle');
const radius = 5;
const area = circleModule.calculateCircleArea(radius);
console.log('The area of the circle with radius ${radius} is : ${area.toFixed(2)}');
```

* output:-

The area of the circle with radius 5 is : 78.54

Question Note on package with example

Answer → A package refers to collection of module and metadata such as package name, version, dependencies etc.) bundled together. Packages can be easily managed and installed using the node package manager (NPM), NPM is a powerful tool that allows developers to discover, share, and use various packages to enhance their Node.js application functionality.

⊗ Creating Packages:-

to create a 'package.json' file open your project directory in the terminal and run the following command

```
npm init
```

⊗ Installing and using packages.

To install lodash package, run the following command.

```
npm install lodash
```

Date ___/___/___

⊛ using the 'lodash' package in your code:

```
const _ = require('lodash');
const numbers = [1, 2, 3, 4, 5];
const sum = _.sum(numbers);
console.log('The sum of numbers is ${sum}');
```

Question = use of package.json and package-lock.json

Answer →

⊛ package.json

→ Dependency management:- one of the main use of 'package.json' is to list all the dependencies required for the project. These dependencies are typically external packages available on NPM that the project relies on to function correctly.

→ ~~vers~~ versioning :-

package.json specifies the version of the project allowing other developers or tools to identify the specific version used in a project.

Page No.

Date ___/___/___

(Saathi)

* scripts:-

you can identify define custom scripts in the package.json making it convenient to run common tasks. such as starting the application running tests building the project etc.

* project information.

the file include important information about the project such as the author licence.

* initiating New project:-

the package.json can be used to initialize a new project with the 'npm init' command where you provide information about the project interactively.

* package-lock.json :-

→ Dependency version locking:-
the primary purpose of package-lock.json is to provide a snapshot of the exact versions of all installed dependencies including their sub dependencies.

Saathi

Date __/__/__

→ Faster and reproducible installs:

By using package-lock.json npm can quickly install dependencies without needing to resolve versions repeatedly, making installation faster and ensuring that everyone working on the project gets the same dependencies.

→ Collaboration.

allows multiple developers to collaborate on a project with the certainly that they are using the same versions of dependencies, reducing potential compatibility issues and inconsistencies.

Question = NodeJS packages

Answer → NPM:-

Node package manager (NPM) is the default package manager for Node.js it allows developers to easily install, manage, and share packages with others.

→ Package.json

each Node.js project contains a

Date ___/___/_____

package.json file that serves as a manifest for the project. it lists all the dependencies (packages) required for the project to run, along with other metadata like the project name, version, author and scripts.

→ installing packages:-

to use a package in a Node.js project you can install it via NPM using the 'npm install' command followed by the package name.

→ Require() :-

to use the functionality provided by a package in your Node.js code, you can use the 'require()' function to import the package and its functionalities into your application.

→ CommonJS modules:

Node.js uses the CommonJS module system, allowing developers to create and import module using the 'module.exports' and require() syntax respectively.

Date __/__/__

Question = npm introduction and commands with its use.

Answer → Introduction to NPM (Node package Manager).

NPM (Node package manager) is a command line tool and the default package manager for node.js it allows developers to easily install manage and share Node.js packages or modules with other developers worldwide, NPM is a crucial part of the Node.js ecosystem providing access to a vast library of open-source packages that enhance the functionality and efficiency of Node.js applications.

* NPM commands and their use in Node.js :-

1. 'npm init' :- Initialize a new Node.js project and create a 'package.json' file. It guide you through a series of prompts to set up essential information about the project, such as the project name, version, description, entry point etc.

Page No.

Date ___/___/___

2. 'npm install' :- install all the dependencies listed in the 'package.json' file for the current project. When run without specifying a package name, it installs all the dependencies listed in the dependencies section of the 'package.json'

3. 'npm install <package_name>' :- installs a specific package and adds it to the dependencies section of the 'package.json' file. example 'npm install express'.

4. 'npm uninstall <package_name>' : uninstall a specific package and removes it from the 'dependencies' section of the 'package.json' file. example 'npm uninstall lodash'.

5. 'npm update' :- update all the packages to their latest versions specified in the 'package.json' it will modify the 'package.json' file accordingly.

Date : / /

Question Describe use and working of following Nodejs packages:

answer
->

(1). Express:

- use :- express is a popular web application framework for Node.js that simplifies the process of building robust and scalable web application and APIs.

- working :- Express provides a set of functions and middleware to handle HTTP request and response, routing, handle different HTTP methods, and integrate middleware for tasks like authentication, error handling and more.

(2). Mongoose:-

→ Use:- Mongoose is an object data model (ODM) library for MongoDB in Node.js it simplifies the interaction with MongoDB and provides a structured schema-based approach to model application data.

Date ___/___/___

- Working :- mongoose allows you to define schemas for your data models and provide an easy-to-use API for crud operation (create, Read, update, delete) and other database interaction.

3. ~~Axios~~:-

- use :- Lodash is a utility that provide a collection of convenient functions to work with arrays, objects, strings, and more in Node.js.

- ~~working~~:- Lodash simplifies complex data manipulation and provide higher order functions like 'map', 'filter', 'reduce', and various utility methods to perform tasks efficiently.

4. Lodash:-

- use :- lodash is utility bi library that provide a collection of convinient functions to work with arrays, objects, strings, and more in Node.js.

- working :- lodash simplifies complex data manipulation and provide higher-order

Page No.

Date ___/___/___

functions like map, filter reduce and various utility methods to perform tasks efficiently. it enhance code readability and improves developement productivity.

5. Axios :-

- Use :- axios is a popular HTTP client for Node.js used for making HTTP request to servers, APIs or other components. endpoint

- working :- axios provide a straightforward API to perform HTTP requests using methods like 'get' post, put, delete etc.

Question = Important properties and method, and relevant programs

Answer →

1. Process Properties :-

- process.argv :- array containing command line arguments passed to the Node.js process.

- process.env :- Object containing environment variables.

Page No.

Date ___/___/___

- process.pid :- process id of the Node.js process.

2. File system methods (fs module) :-

- 'fs.readFile()' :- asynchronously reads the contents of a file.

- 'fs.writeFile' :- asynchronously write data to file, replacing the file if it already exists.

- 'fs.existSync()' :- synchronously tests whether a path exists.

3. HTTP Module :-

- 'http.createServer()' :- Creates an HTTP server.

- http.ServerResponse :- Represent the response from an HTTP server.

- http.IncomingMessage :- Represent an incoming HTTP request.

4. Events (Event Emitter) :-

- events.EventEmitter :- core class enabling event-driven programming in Node.js

- emitter.on(eventName, listener) :- Adds a listener function to the specific event.

- emitter.emit(eventName, [args]) :- Emits an event with the given name and optional arguments.

Question :- URL.

'url' module is a built-in module that provide utilities for working with URLs (Uniform Resource Locators), it allows developers to parse, formats, and manipulate URLs in a simple and efficient manner.

- url.parse() :- this function parses a URL string and returns an object containing its various components, such as protocol, hostname, port, path, query, parameter and more.

- url.format() :- the formatter function takes a URL object and returns a formatted URL string.

- url.resolve() :- this function resolve a target URL relative to a base URL and returns the absolute URL.

Question = process, pm2 external package

⊗ process :-

- the process object is a global object in Node.js that provide information and control over the current Node.js process.

- it allows access to information like command-line arguments, environment variables and process specific details.

⊗ pm2 : process manager 2 :- external package :-

- pm2 is an external package used as a process manager for running and managing Node.js application in production environment.

Date ___/___/___

Question readline

readline module is a built-in module that provides an interface for reading input from readable streams, such as the terminal or a file, line by line it simplifies the process of getting user input or reading large files in a line-by-line manner.

→ Interface creation :- The readline module allows you to create a specific interface to read input from the terminal or any readable stream.

→ Line-by-Line Reading :- it facilitates reading input line by line which is useful for handling user input or processing large files without loading the entire file into memory.

Question fs

→ fs module is a built-in module that stands for file system it provides functions for interacting with the file system, allowing developers to read, write and manipulate files and directories it is a core module

Page No.

Date ___/___/___

and does not require installation.

Question events

- events module is a core module that enables event-driven programming, it provides the 'EventEmitter' class, allowing developers to create, emit, and handle custom events.

Events-driven programming allows different parts of the application to communicate asynchronously enhancing and efficiency and scalability of Node.js application.

Question Console.

→ console is a built-in module used for basic input and output operations it provides functions like console.log() to print messages to the console allowing developers to debug and display information during program execution.

The console module is essential for inspecting variables logging errors and tracking the flow of execution in Node.js application.

Page No.

Date ___/___/___

Question buffer

Buffer is a built-in class that represent binary data, it allows developer to work with raw binary data directly and is used for handling streams, file operation and other low-level data manipulation tasks.

The buffer class is particularly useful when working with network operations and file I/O, where data needs to be processed in its raw binary form.

Question querystring

querystring module is a built-in module that provide utilities for parsing and formatting URL query strings. It allow developer to convert query parameters to object and vice-versa.

The querystring module simplifies working with query strings, enabling easy extractions of data from URLs and handling URL-encoded data in HTTP request and responses.

Page No.

Question http.

http module is a built-in module that allows developers to create HTTP servers and make HTTP requests, it provides functions to handle incoming HTTP requests and send HTTP response.

The http module is the foundation for building web servers and interacting with web services in Node.js applications.

Question v8

v8 is an open-source Javascript engine developed by Google, it is the underlying engine that interprets and executes Javascript code in Node.js applications. v8 is highly optimized for performance and memory usage, making Node.js fast and efficient in handling Javascript code, making it suitable for building scalable and high-performance applications.

Date __/__/__

Question os

os module is a built in module that provide operating system-related utilities. it allows developers to access information about the operating system, such as the platform, architecture, hostname network interface and more.

The 'os' module is useful for building platform-independant application that can adapt to different operating systems and environments

Question zlib

zlib module is a built in module that provides compression and decompression functionalities using the zlib compression algorithm. it allows developers to compress data to save space and decompress it for efficient data transmission or storage.

The zlib module offers methods like zlib.gzip() for compression and zlib.gunzip() for decompression, making it useful for handling data streams and reducing file sizes in Node.js applications.

Page No.