**1. What is Node.JS?**

Node.js is a very popular cross-platform runtime environment that is based on Chrome’s V8 JavaScript code. It is an open-source platform that enables developers to create real-time web applications. Because of its features and non-blocking asynchronous architecture, Node.js has become very popular in a very short period. Giants like Uber, Netflix, Walmart, and eBay are using the benefits of Node.js for their web apps. It makes the web development process very smooth and efficient.

**2. What is Node.js Used for?**

There are many reasons to use Node.js for your different projects. Here are some:

**2.1 API**

A RESTful or GraphQL API can be created with relative ease, as  node.js    has a robust library support system. In the past, Node.js was only suggested for low-volume, high-CPU-intensive applications. Worker threads have made this possible since version 10.5.0.

**2.2 Streaming Web Applications**

Videos and music are often consumed via streaming platforms. Node.js has a built-in streams module that allows you to process large amounts of data sequentially. This eliminates the need for any kind of temporary storage or caching in your program memory. Since video encoding can be done on the server, clients won’t have to worry about it.

**2.3 Video Gaming**

The fast reaction time of the platform is particularly useful for multiplayer games and virtual emulators since it simplifies communication between a large number of participants and the execution of different scenarios.

**2.4 Command Line Tools**

NPM offers a wide array of packages that may be used to create more complex tools in addition to standard command-line utilities.

**2.5 Real-Time Web Applications**

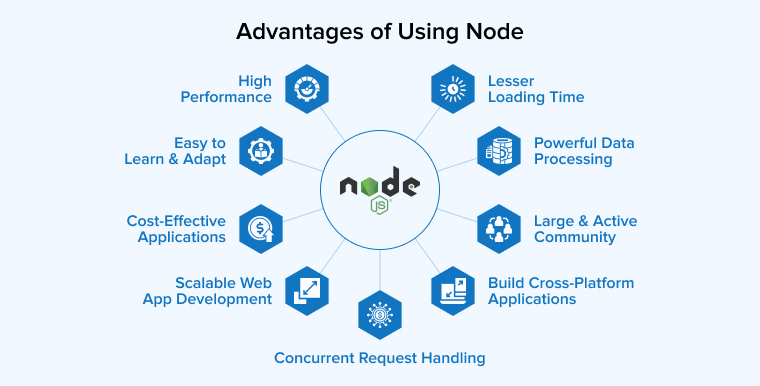
Real-time online applications, such as a chat room where users can communicate with one another in real-time, can be built using Node.js since it can manage a large number of I/O operations efficiently. Node’s Web Sockets and Events API make this a reality. When it comes to building websites, Node js is at ease.

**2.6 Microservices**

This approach to developing apps has gained a lot of traction in the past several years. Since Node.js is lightweight and simple to implement, it can be used to quickly expand your solution. When developing and managing our microservices, Node.js is a go-to technology. Many businesses choose Node.js as part of their transition to microservices.

**3. Advantages of Using Node**

As we all know, Node.js is continuously growing and gaining massive popularity for web app development, but now its capabilities have grown even more because it offers a variety of benefits for backend development. It is a great platform for real-time application development and managing huge amounts of information.



Let’s discuss the advantages of Node.js in detail:

**3.1 Scalable Web App Development**

Node.js was built with scalability in mind. It allows multiple threads to run simultaneously and interact with each other which is superior to other web backend development solutions. It has a cluster module that is responsible for load balancing for every CPU-bound task. This is one of the main Node.js advantages. You can run more than one node at a time and the cross-platform runtime environment will automatically balance the workload.

Node.js allows horizontal partitioning and divides your app into small instances. You can show various versions of the mobile apps to users based on their interests, age, location, language, etc. which reduces workload and increases personalization. Node.js child processes operate quickly and interact with each other to share the same origin.

**3.2 High Performance**

Node.js is popular for providing excellent performance and facing considerable amounts of data to process at the same time. It is a perfect choice for backend development that interprets the JavaScript code using Google’s V8 JavaScript engine. Node.js web app development engine helps to compile the code into machine code directly, making it easier and faster to write server-side code and implement it.

The backend Node.js technology accelerates code execution and its runtime environment supports non-blocking I/O operations. As a single-threaded event loop, it uses multiple worker threads in the background to execute the asynchronous programming code and nonblocking which means the callback function is delegated to the event loop but executed by different threads.

**3.3  Cost-Effective with Full-Stack JavaScript**

With Node.js, JS developers can effortlessly create both server-side and client-side JavaScript applications. One of the finest things about node.js is how it increases development speed and reduces the cost and effort needed to build a project by eliminating the need to hire two resource teams.

After migrating from Ruby on Rails to Node.js, LinkedIn’s development team saw a significant boost in performance. They went from thirty web servers to only three. They began more effectively allocating and utilizing available resources and moved their attention from problem-solving to app development.

**3.4 Powerful Data Processing**

Node.js is a backend development language and runtime environment capable of processing thousands of requests simultaneously. It is a perfect choice for efficient data-handling tools. Many backend development platforms use synchronous processing for handling requests where they each wait to process CPU-bound tasks one by one. It refuses the traditional system and adopts an asynchronous structure for upcoming requests to process together with the first one.

There are two approaches to software development: linear and event-based programming. A linear tool helps to execute algorithms in a queue regardless of users’ actions. Event-based programming doesn’t have a specific output order and there are useful background operations.

**3.5 Easy to Learn and Quick to Adapt**

If a programmer is familiar with JavaScript, making the switch to Node.js for server-side work is a simple process. This is appealing to novice programmers since the runtime environment is simple to pick up.

It’s more like a general-purpose language in the software development industry. Even if the programmer is a newbie or junior developer, they won’t need to spend any additional time studying NodeJS.

**3.6 Concurrent Request Handling**

With Node.js, handling multiple concurrent users simultaneously in a single go is possible. And this is one of the biggest advantages of Node.js. Its capacity to handle multiple requests at once for any type of complex and extensive web application has made it popular among Javascript developers. Besides, the majority of [eCommerce businesses prefer Node.js](https://www.tatvasoft.com/outsourcing/2021/07/node-js-to-develop-an-ecommerce-application.html) for their application for this reason.

**3.7 Lesser Loading Time**

Node.js comes with a caching module that helps in reducing the overall loading time. Developers can execute the code without any hassle. Besides, having a caching module helps the web app get a request easily and then transfer it to in-app memory. This enables the user to get results on web pages in a few microseconds without waiting much.

All these benefits of node.js contribute towards making nodejs a favorable choice for developers for their development projects. On the other hand, there are certain disadvantages of node.js that will affect your decision to choose a node. These disadvantages of node js are as under.

**3.8 Large and Active Community Support**

With thousands of professionals actively participating in the Node.js community, you can rely on assistance from specialists all around the world to help you work through even the most unusual programming issues.

For JavaScript, the most used package manager registry is NPM. It provides easy access to a wide variety of free tools and libraries that can be integrated into your work. There is a wealth of developer-created resources on GitHub that can help you get more done while spending less time and money.

Many of the world’s most well-known technology companies, including Amazon, Google, Facebook, and Netflix, are strong supporters of the node.js community and have contributed significantly to its success. They’ve contributed much by creating several open-source alternatives. The future of technology depends on it being adopted by major tech companies and the development community.

**3.9 Helps in Building Cross-Platform Applications**

Using frameworks like Electron and NW.js, Node.js enables the development of cross-platform, real-time online apps. This simplifies the time-consuming process of creating multiple sets of code for various desktop operating systems. Your team won’t require experts in Objective-C, C#, or any other language in the future to build native desktop applications.

In addition, the Google Chrome Developer Tools can potentially be used by your Node.js developers for any purpose, including coding, debugging, and building. This efficient method is fundamental to modern web development using Javascript front-end frameworks like AnjularJS, Vue.js, and React and back-end frameworks like Node.js. It has made it possible for full-stack developers to reap the benefits of and explore the opportunities presented by node.js.

**4. Disadvantages of Using Node**

**4.1 Asynchronous Programming Model**

To get a scalable app the developer needs a technology that has an asynchronous programming model. However, some developers may feel that it is a difficult programming model in comparison to the linear blocking I/O programming. Therefore,  fewer programmers are choosing Node.js as their development framework.

**4.2 Doesn’t have a Strong Library Support System**

**JavaScript does not come with a well-equipped library system compared to other universal** programming languages. This results in developers being forced to take support of a common library for various tasks like processing the images, Object-Relational Mapping (ORM), handling database operations, and more.

**4.3 Unstable API**

In Node.js there are frequent  API changes. API stands for Application Programming Interface and it keeps on changing. Every new API comes with backward-incompatible changes. This means that the developers will have to change the code to make it compatible with the latest version of the Node.js API.

**4.4 Reduces Performance when Handling Heavy Computing Tasks**

Node.js has certain benefits from being single-threaded and event-driven, but it has several drawbacks when it comes to doing heavy CPU-intensive tasks.

Node.js will use all of its CPU power to complete a large CPU-driven job received in the event loop. This slows down the application’s event loop as a whole, making the user interface less responsive.

To address this issue, Node.js added the “worker threads” module in 2018. This made it possible for programmers to run several threads at once. However, you will still need other tools to cope with CPU-bound computing problems, and this method does not provide them altogether.

5. Conclusion

As seen in this blog, Node.js has a lot of advantages rather than disadvantages and it is because of its architecture, features, and active community support. This is why developers prefer to use Node.js in the maximum number of projects that fall in its requirement range. Besides, as Node.js has JavaScript, it can be used for both frontend and backend servers which is a big plus. So, if you want to take your client’s project to a greater height, Node.js is the best technology.

**What is the Most Important Feature of NodeJS?**

These are a few of the most appreciated features in NodeJS

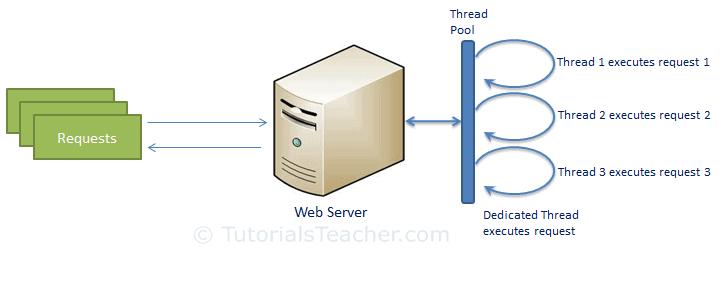
* Built with JavaScript
* Single-Threaded Operation
* Speedy Streaming
* Reduced or No Buffering
* Multiple-platform Compatibility

**Node.js Process Model**

In this section, we will learn about the Node.js process model and understand why we should use Node.js.

**Traditional Web Server Model**

In the traditional web server model, each request is handled by a dedicated thread from the thread pool. If no thread is available in the thread pool at any point of time then the request waits till the next available thread. Dedicated thread executes a particular request and does not return to thread pool until it completes the execution and returns a response.

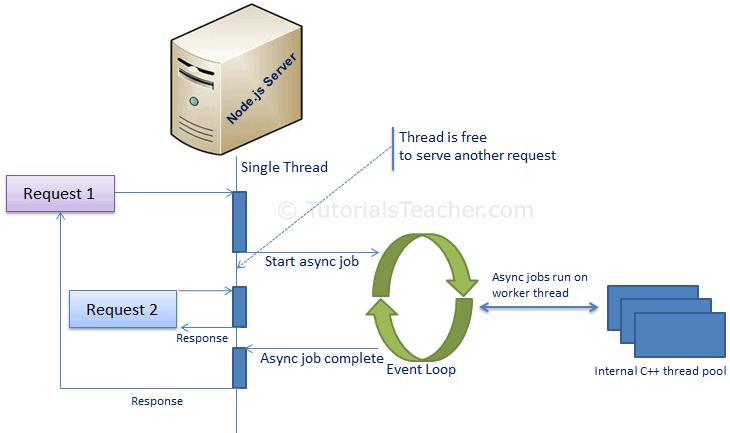


**Node.js Process Model**

Node.js processes user requests differently when compared to a traditional web server model. Node.js runs in a single process and the application code runs in a single thread and thereby needs less resources than other platforms. All the user requests to your web application will be handled by a single thread and all the I/O work or long running job is performed asynchronously for a particular request. So, this single thread doesn't have to wait for the request to complete and is free to handle the next request. When asynchronous I/O work completes then it processes the request further and sends the response.

An event loop is constantly watching for the events to be raised for an asynchronous job and executing callback function when the job completes. Internally, Node.js uses libev for the event loop which in turn uses internal C++ thread pool to provide asynchronous I/O.

The following figure illustrates asynchronous web server model using Node.js.



Node.js process model increases the performance and scalability with a few caveats. Node.js is not fit for an application which performs CPU-intensive operations like image processing or other heavy computation work because it takes time to process a request and thereby blocks the single thread.

**Installation of Node.js**

**Step 1: Download Node.js Installer**

* Visit the [official Node.js](https://nodejs.org/en/download/) website to download the Node.js ‘.msi’ installer
* Download the Windows Installer based on your system architecture (32-bit or 64-bit)

**Step 2: Run the Installer**

* Locate the downloaded**.msi** file and double-click to run it.
* Follow the prompts in the setup wizard, accept the license agreement, and use the default settings for installation.
* Select features to install such as:
  + npm: to manage packages for Node.js applications
  + Native modules: for building native C++ modules

**Step 3: Finish Setup and Install Node.js and NPM**

* The installer may prompt you to “install tools for native modules”. Select “Install” to complete the process.

**Step 4: Verify the Installation**

Open **Command Prompt** or **PowerShell** > Check the installed versions by running these commands:

* Type **node -v** and press Enter to check the Node.js version.
* Type **npm -v** and press Enter to check the npm version.
* Both commands should return version numbers, confirming successful installation.