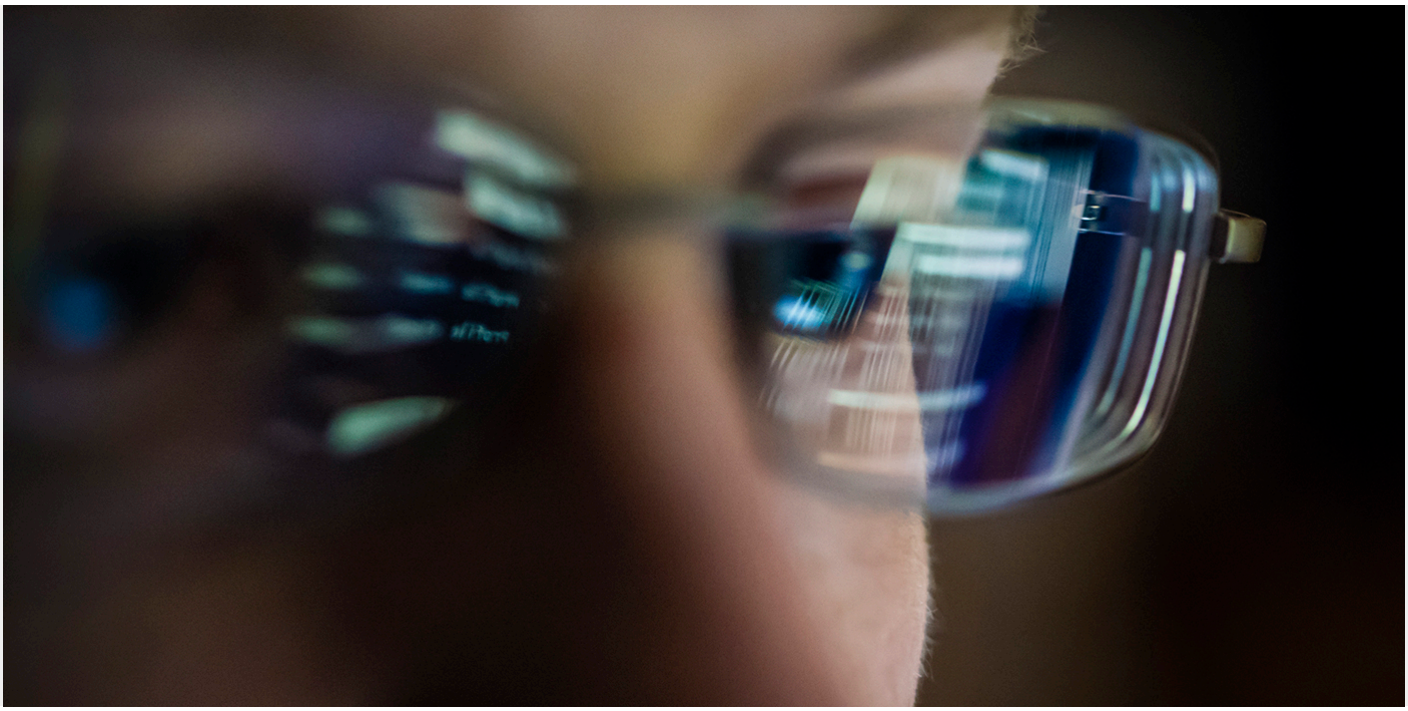
NODEJS VS PYTHON: WHEN & HOW TO USE BOTH



NodeJS and Python are two of the most popular technologies for application development. Python is one of the widely adopted programming languages, facilitating developments in many areas. On the other hand, NodeJS is a runtime environment.

Both are excellent for their intended purposes with overlapping use cases. In this post, we will dig into Python and NodeJS to understand the similarities and differences between the two technologies.

What is Python?

Python is an open-source, high-level, and dynamic programing language. Python is a general-purpose language, meaning that it's not specialized for a specific area or task. It can be used for any development purpose, from building websites and software, automation to data analytics and machine learning, etc.

This flexibility and user-friendliness have made Python one of the leading programming languages.

Advantages of Python

- **Versatility.** As a general-purpose language, Python can be used to accommodate a wide variety of programming needs, from simple scripting to machine learning.
- **Ease of use.** Python is one of the simpler languages to learn, with a low barrier to entry while offering all its powerful capabilities.
- **Echo-system.** Python has thousands of libraries and frameworks to facilitate any kind of functionality. Thus, you can easily find packages to extend the functionality of Python. The best

part is that all these libraries and frameworks can be easily installed via the Python package manager called pip.
- **Extensibility.** Python can be easily integrated with other languages such as C, C++, and Java. It helps to utilize the functionality of Python within programs developed using other languages.
- **Cross-platform support.** Programs can be run on any operating system, including Windows, Linux, and macOS.
- **GUI support.** Unlike some other languages, Python has multiple fully developed GUI frameworks like Tkinter and Pygame to create GUI applications.

What is NodeJS?

NodeJS is a single-threaded, open-source JavaScript runtime environment that enables developers to build scalable server-side applications. Node is built on the open-source V8 JS runtime engine and written in C, C++, and JavaScript.

The main difference between NodeJS and Python is that Python is a fully flagged programming language while Node is a runtime environment designed to run JavaScript outside the browser.

Advantages of NodeJS

- **Simplicity.** Since NodeJS uses the popular JavaScript language as the base, developers can easily use it in their applications and use JavaScript for both client-side and server-side developments.
- **Scalability.** The single-threaded nature of NodeJS helps to scale NodeJS applications easily by enabling it to handle a large number of simultaneous connections with high throughput.
- **Ecosystem.** NPM offers thousands of packages to extend the functionality of NodeJS.
- **Speed & efficiency.** NodeJS can run relatively faster than other tools and runtimes as it is developed using C and C++. When this speed is coupled with the scalability of its runtime, the speed of NodeJS applications is increased further.
- **Multi-platform.** Node has cross-platform support allowing users to develop web, desktop, and mobile applications.

Comparing NodeJS vs Python

Now that we understand the basics of Python and NodeJS, let's compare them to identify the intricacies of this programming language and runtime environment.

Use cases

The first thing to compare is the use cases. While both NodeJS and Python are excellent back-end technologies, they have many ways to use them:

Node is ideal for scalable application developments, especially when dealing with real-time data and event-driven architectures. The features and speed of Node have made its runtime an excellent choice to power REST APIs, IoT, single-page applications, data streaming, etc. Additionally, NodeJS can also be used to create desktop and mobile applications with tools like Electron, Ionic, and Flutter.

As a general-purpose language, Python can be used for virtually any kind of development. These developments range from developing desktop applications to web applications using frameworks like Flask, Django, and Pyramid. As a scripting language, Python can be used to add additional functionality to software developed using other programming languages and as a language to create automation scripts.

Additionally, Python has gained immense popularity with data science as one of the leading languages for data analytics, machine learning, neural networks, and artificial intelligence projects. Even though mobile development support is one area that Python lacks, frameworks like Kivy and Beeware can be used for mobile development.

Importantly, however, Python lacks features and tools when compared to other options such as React Native and Flutter.

Architecture

Good architecture is vital for any software application or tool to function properly in an efficient manager. Architecture defines the underlying behavior, components, and the relationships between components.

NodeJS is based on a single-threaded event loop model to handle multiple client requests simultaneously. Its architecture is designed to reduce resource usage, leading to relatively lightweight processes with fast executions. The non-blocking nature of NodeJS also allows handling multiple concurrent connections.

Python converts its code into bytecode and later machine code using an interpreter. This approach leads to slow code execution times compared to other languages. However, there are new interpreters like PyPy that increase the speed of Python as an alternative to the default CPython.

Python also does not support multi-threading—the underlying CPython interpreter does not support true multi-core execution via multi-threading. However, it does not limit the functionality of Python as libraries like Asyncio can be used to build asynchronous applications.

Performance

Speed, scalability, and efficiency are key parameters when considering the overall performance of any tool or service. A faster and more efficient platform will lead to more stable and responsive applications.

NodeJS executes its code outside of the constraints of the browser, allowing it to be faster and more resource efficient. The non-blocking nature of the architecture allows increasing the speed further.

Node applications can easily scale up or down depending on the application architecture and requirements. Moreover, NodeJS can easily facilitate scalable architectures with fast execution times as well as lightweight communication between each process.

Python is slower than NodeJS as an interpreted language. As Python does not support multi-threading natively, the scalability of Python applications can be limited compared to NodeJS. The Python interpreter is unable to execute multiple tasks simultaneously. However, there are

implementations like the PyPy, which is a new interpreter that increases speed. Additionally, there are features like Stackless Python to integrate thread-based programming using Python.

Extensibility

The ability to extend functionality outside of the core capabilities is crucial when deciding on a tool for development. Extensibility without impacting the existing features or functions and having an extensive echo system are key pillars to enable extensibility. Both NodeJS and Python have excellent extensibility options.

NodeJS can be easily extended and integrated with various packages and tools. Node package manager (NPM) provides developers access to thousands of packages to add new capabilities to an application. NPM has the largest open-source package library with over a million packages.

NodeJS also provides an inbuilt API for developing HTTP and DNS servers. Furthermore, frameworks like React, Vue, and Angular allow developers to create web applications easily.

Python also has an extensive package library that allows developers to add new functionality to Python via its pip repositories. It features an extensive list of frameworks from web development to data analytics and machine learning. Here, the extensibility of Python plays a key role as it can be easily integrated with other programming languages.

A good example of this is to use Python binding to call functions and pass data from Python to languages like C and C++. It allows developers to take advantage of the strengths of both languages and provides a good solution to overcome the relative slowness of Python.

Ease of use

With straightforward syntax and programming structure, both technologies are easy to learn, particularly compared to other languages like Java, C++, and C#. However, Python has the edge here as it's much more readable than NodeJS.

Additionally, Python has a slight edge over NodeJS with beginner-friendliness as it is easy to learn and get started.

NodeJS vs Python: Comparison summary

bmc

NodeJS vs Python: Comparison

	NodeJS	Python
Type	Runtime environment	High-level programing language
Architecture	Uses the V8 JavaScript engine	Uses Cpython as default interpreter; supports other interpreters like PyPy
Maturity	Created in 2009	Created in 1981
Package manager	Node Package Manager (NPM)	PIP Package Manager (Recursive acronym for "Pip installs packages")
Scalability	Highly scalable both horizontally & vertically, with ability to handle multiple concurrent requests	Comparatively limited scalability; requires 3rd-party packages to increase the scalability
Extensibility	Highly extensible	Highly extensible
Performance	Faster than Python	Relatively slower (interpreted language)
User friendliness	Beginner friendly and simple to learn	Higher levels of user friendliness; simpler syntax & code structure with better readability
Primary use cases	Frontend & backend web development; Mobile & desktop development	Backend web Development; desktop development; automation; data science

What to Choose for Your Development?

Both NodeJS and Python are excellent tools for their targeted development use-cases. NodeJS will be ideal if you want a unified runtime environment to create cross-platform applications for web, mobile, and desktop.

However, it does mean Python cannot be used for these types of developments as it is a popular choice for powering many back-end services. Moreover, Python has a clear edge over NodeJS when

it comes to other requirements like automation scripting, data analytics, and machine learning. It is also the go-to language for many DevOps and data science projects.

Related reading

- BMC DevOps Blog
- The Lua Programming Language Beginner's Guide
- React JavaScript Library: Concepts & Tutorials for Getting Started
- SRE vs DevOps: What's The Difference?
- Today's Top Trends in Software Development
- Top Conferences for Programming & Software Development