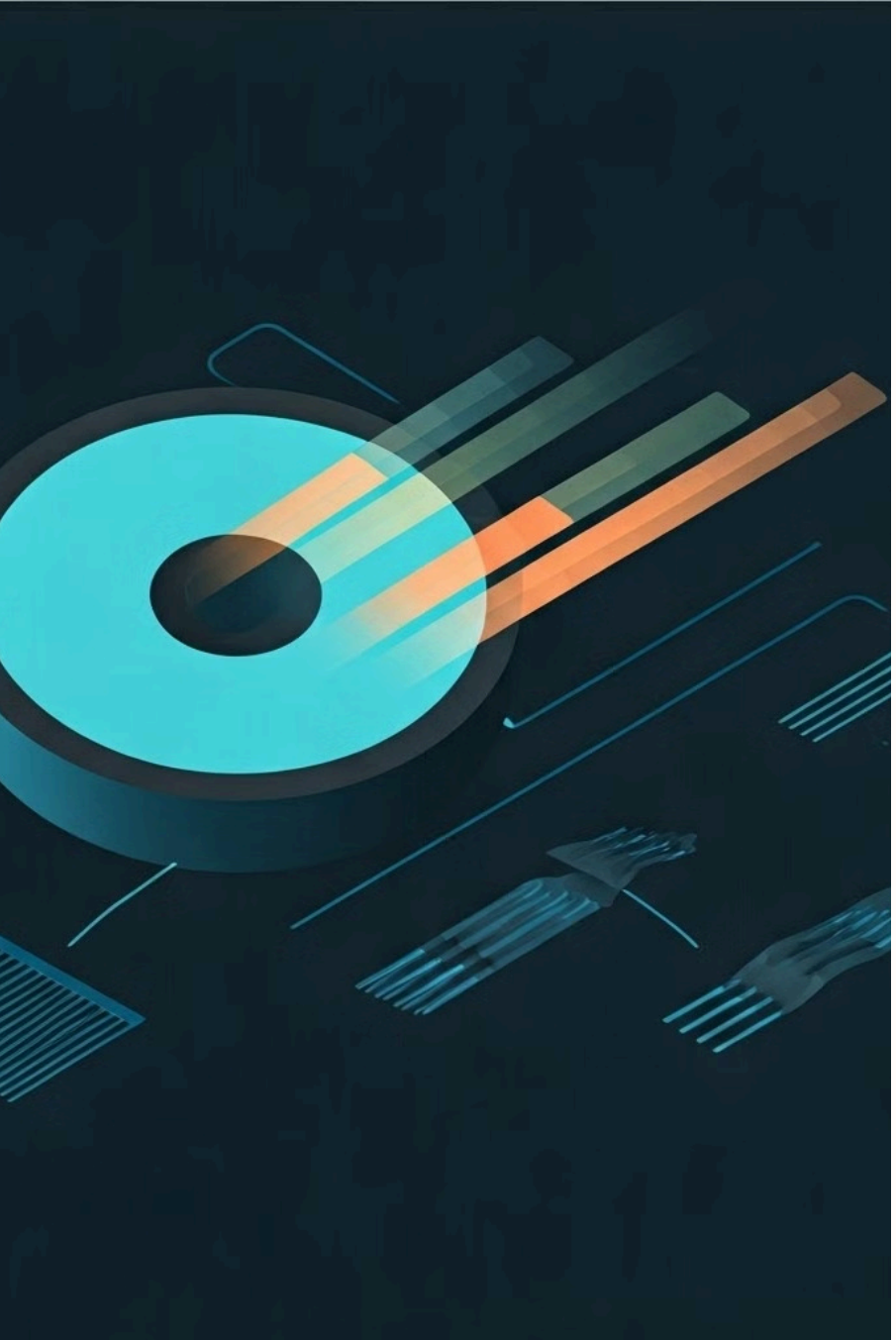
Node.js란 무엇인가?

Node.js는 JavaScript 런타임 환경으로, 서버 측 애플리케이션 개발에 널리 사용되는 강력한 플랫폼입니다. 이를 통해 개발자들은 웹 브라우저에서뿐만 아니라 서버 환경에서도 JavaScript를 사용할 수 있습니다.



Node.js의 특징

1 비동기 I/O

Node.js는 비동기 I/O 모델을 사용하여 높은 처리량과 성능을 제공합니다.

2 이벤트 기반 아키텍처

Node.js는 이벤트 기반 모델을 사용하여 효율적인 웹 애플리케이션 개발을 가능하게 합니다.

3 단일 스레드

Node.js는 단일 스레드 모델을 사용하여 간단하고 확장 가능한 애플리케이션을 만들 수 있습니다.

Node.js의 장점

성능

Node.js는 빠르고 효율적인 실행 환경을 제공하여 높은 처리량과 낮은 지연 시간을 보장합니다.

확장성

Node.js는 수평 확장이 용이하여 대규모 애플리케이션 개발에 적합합니다.

풀스택 개발

Node.js를 통해 클라이언트와 서버 측 JavaScript 코드를 공유할 수 있어 생산성이 높습니다.

Node.js의 활용 분야

웹 애플리케이션

Node.js는 고성능 웹 서버 구축에 널리 사용됩니다.

실시간 애플리케이션

채팅 앱, 게임, 실시간 데이터 스트리밍 등에 적합합니다.

IoT

센서 데이터 처리, 디바이스 제어 등의 IoT 애플리케이션 개발에 활용됩니다.

API 서버

REST API, GraphQL API 등 강력한 백엔드 서비스를 제공합니다.

Node.js의 주요 기능

1

이벤트 드리븐 엔진

Node.js는 이벤트 기반 아키텍처를 사용하여 비동기 처리를 지원합니다.

2

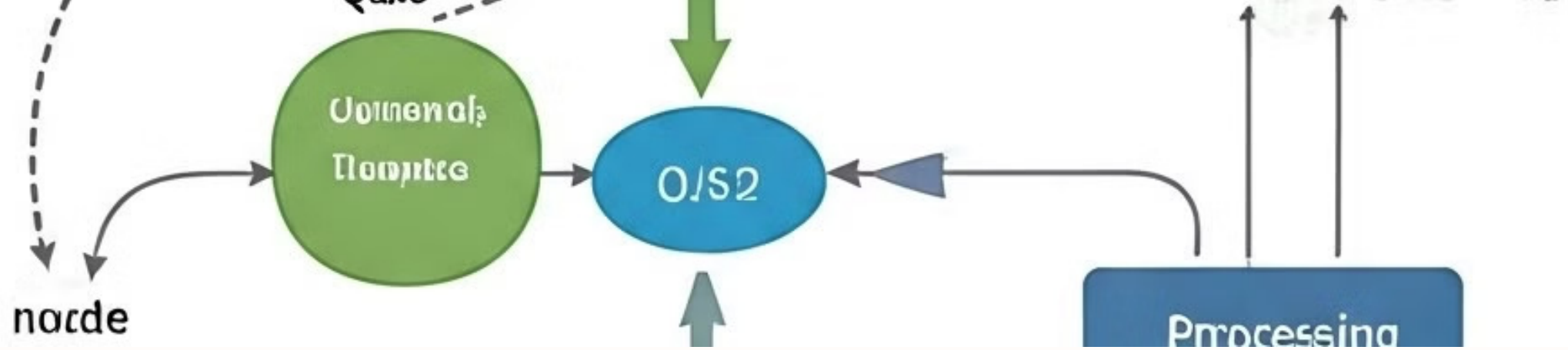
비동기 I/O

Node.js는 블로킹 없이 효율적인 I/O 작업을 수행할 수 있습니다.

3

풍부한 라이브러리

Node.js는 다양한 기능을 제공하는 수많은 모듈과 라이브러리를 가지고 있습니다.



Node.js의 비동기 처리

1

2

3

이벤트 큐

비동기 이벤트는 이벤트 큐에 저장되어 순차적으로 처리됩니다.

이벤트 루프

이벤트 루프는 이벤트 큐의 이벤트를 지속적으로 감시하고 처리합니다.

논블로킹 I/O

I/O 작업은 비동기적으로 처리되어 애플리케이션의 성능을 높입니다.

Node.js의 핵심 개념

이벤트 루프

이벤트 기반 비동기 처리의 핵심 메커니즘

논블로킹 I/O

CPU 사용을 최소화하는 효율적인 I/O 처리

모듈 시스템

재사용 가능한 코드를 체계적으로 구성

콜백 함수

비동기 작업 완료 시 실행되는 함수

Node.js 활용을 위한 팁

모듈 활용	Node.js의 방대한 모듈 생태계를 적극 활용하세요.
비동기 처리	콜백, Promise, async/await 등을 사용하여 비동기 코드를 작성하세요.
에러 처리	예외 처리와 로깅을 통해 안정적인 애플리케이션을 만드세요.
성능 최적화	이벤트 루프, 비동기 I/O, 클러스터링 등을 활용해 성능을 개선하세요.