Node.js 소개 및 활용

Node.js는 Chrome V8 JavaScript 엔진을 기반으로 한 서버 사이드 플랫폼입니다. 비동기 이벤트 기반 프로그래밍을 지원하여 웹 서버, API 서버, 실시간 애플리케이션 등 다양한 분야에서 활용됩니다.

2020875029 소프트웨어학과 신성우

Node.js의 사용 사례

웹 서버

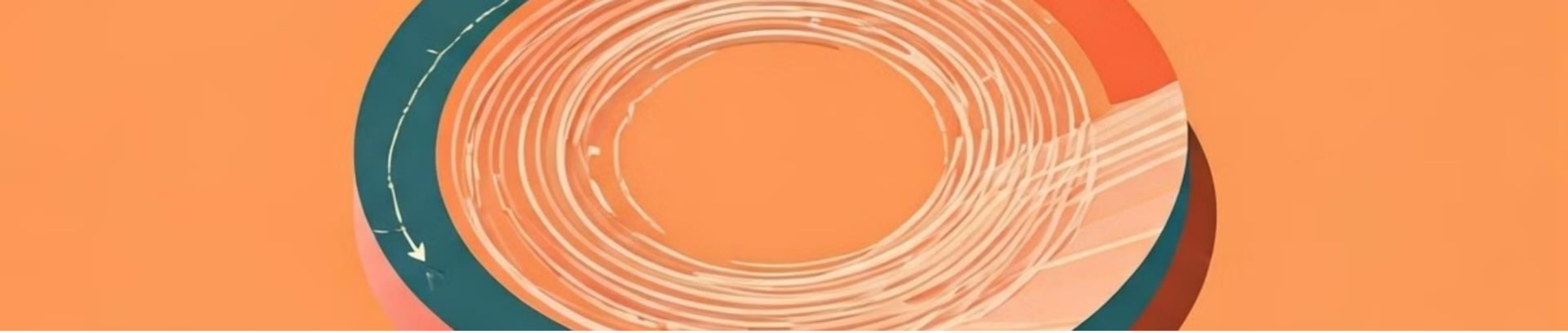
Express.js를 이용한 빠르고 확장 가능한 웹 서버 구축

실시간 애플리케이션

Socket.io를 활용한 실시간 채팅 앱 개발

API 서버

RESTful API 서버 구현으로 효율적인 데이터 교환



Node.js의 주요 특징

1

비동기 I/O 처리

높은 성능과 확장성을 제공하여 대규모 애플리케이션 개발에 적합합니다.

2

단일 스레드 이벤트 루프

효율적인 리소스 사용으로 서버 자원을 최적화합니다.

3

npm (Node Package Manager)

방대한 패키지 생태계로 개발 생산성을 크게 향상시킵니다.

Node.js 기본 개념

모듈

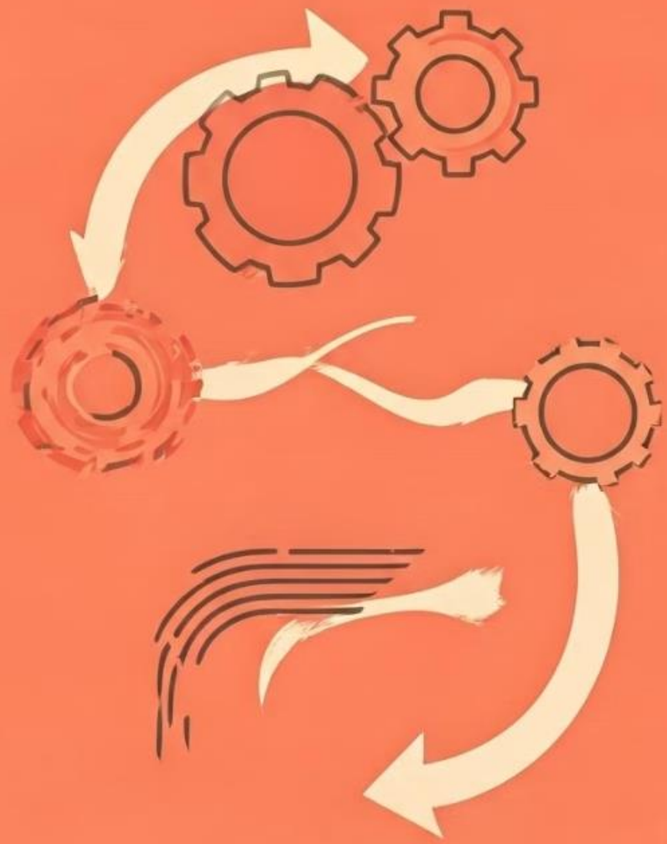
코드의 재사용을 위한 기본 단위로, 기능을 캡슐화하고 의존성을 관리합니다.

이벤트

비동기 처리를 위한 핵심 개념으로, 이벤트 기반 프로그래밍을 가능하게 합니다.

콜백

비동기 함수의 결과를 처리하는 방법으로, 비동기 작업의 완료를 알립니다.



Node.js 모듈 시스템

CommonJS

Node.js는 CommonJS 모듈 시스템을 사용합니다. 'require' 함수로 모듈을 불러오고, 'module.exports'로 모듈을 정의합니다.

모듈 정의

module.exports = { 함수명: 함수 }; 형식으로 모듈을 정의합니다.

모듈 사용

const 모듈명 = require('./모듈경로'); 형식으로 모듈을 불러와 사용합니다.

Node.js의 보안 고려사항

1

입력 검증

SQL 인젝션을 방지하기 위해 사용자 입력을 철저히 검증합니다.

2

HTTPS 사용

데이터 암호화를 위해 HTTPS 프로토콜을 사용합니다.

3

환경 변수 관리

민감한 정보는 환경 변수로 관리하여 보안을 강화합니다.

Node.js 성능 최적화

1

비동기 코드 활용

I/O 작업은 비동기로 처리하여 블로킹을 방지합니다.

2

캐싱 전략

자주 사용되는 데이터는 메모리에 캐싱하여 응답 시간을 단축합니다.

3

클러스터 모듈 사용

멀티 코어 CPU를 활용하여 애플리케이션의 처리량을 증가시킵니다.

결론 및 Q&A

1 **중요성**

Node.js는 현대 웹 개발에서 중요한 역할을 합니다.

2 **장점**

비동기 처리와 높은 성능으로 효율적인 애플리케이션 개발이 가능합니다.

3 **Q&A**

질문을 통해 Node.js에 대한 더 깊이 있는 이해를 도모합니다.