

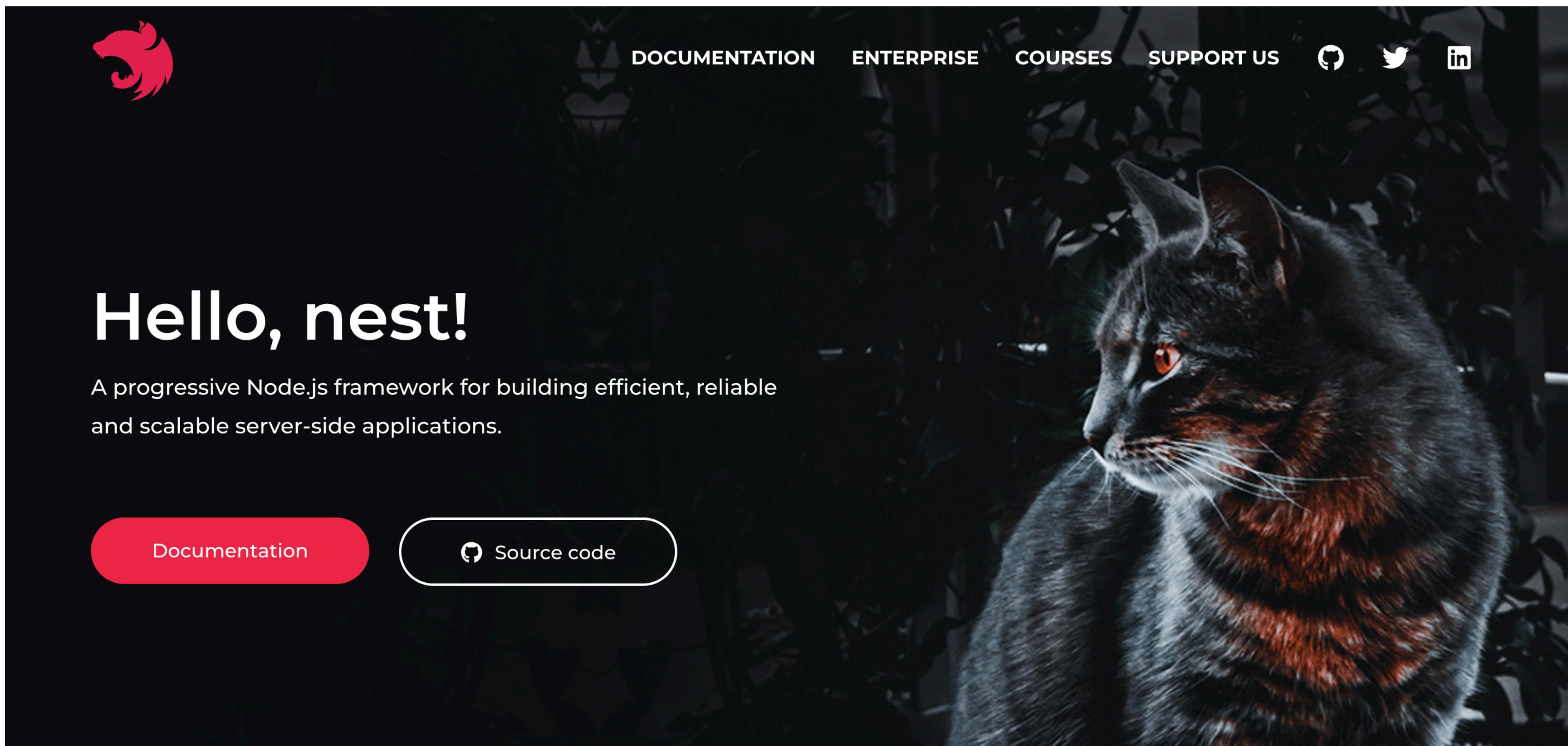
Backend-NestJs 03

**Nestjs 자세히 살펴보기 / NodeJS와 javascript의 언어적 특징 / curl 과
Postman / 간단한 api 서버 구축**

컴퓨터학과 17 홍석민

- <https://github.com/honghyeong>

NestJs 란 ?



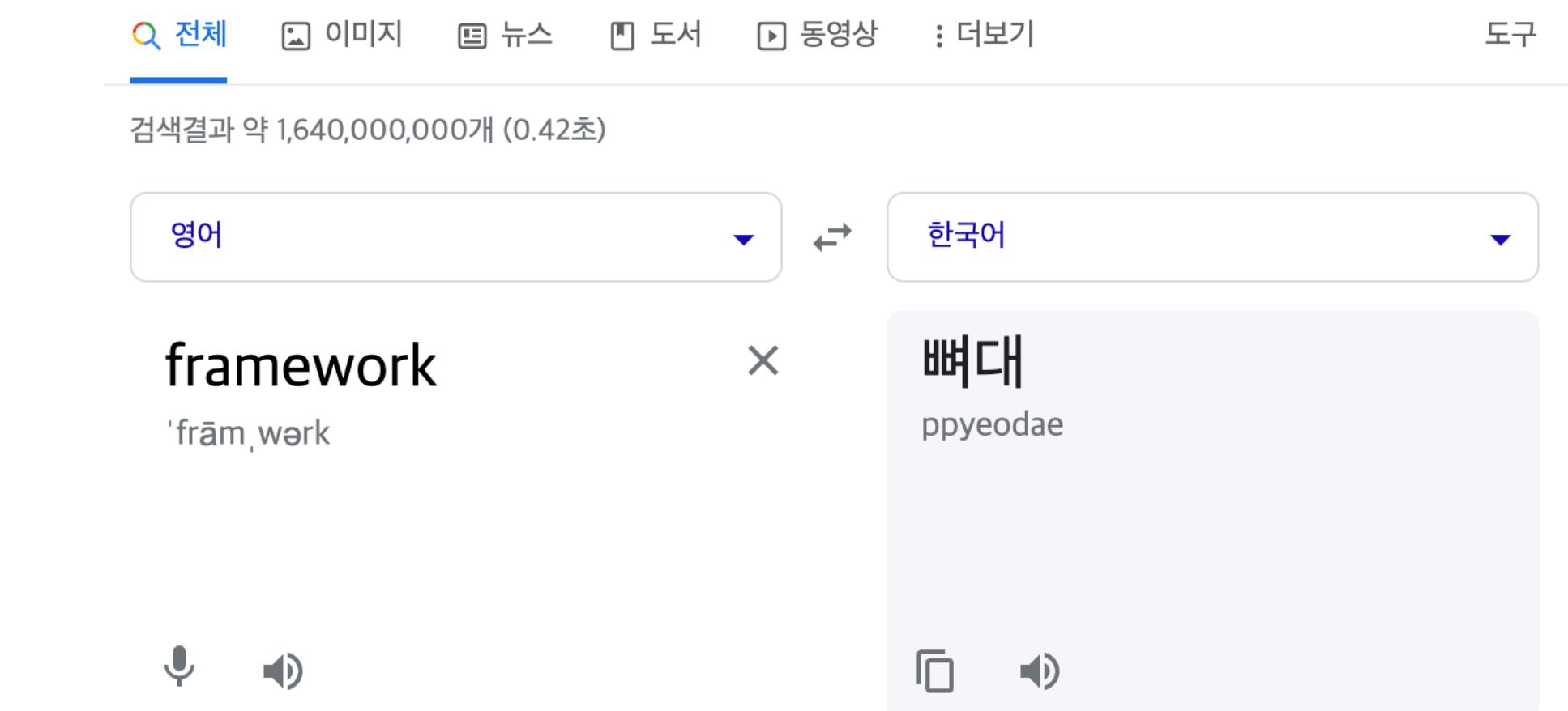
NestJs 란 ?

- NestJs는 Node.js에 기반을 둔 웹 API 프레임워크로써 Express 또는 Fastify 프레임워크를 래핑하여 동작 (Default : Express)
- **Node.js**
 - 장점 : 손쉬운 사용 + 확장성 + 유연성
 - 단점 : 과도한 유연함으로 인한 구조적 문제, 적절한 라이브러리 서칭 어려움
- **NestJs**
 - 장점 : 데이터베이스, ORM, Validation Check 등 수많은 기능을 standard 로 제공하여 따로 라이브러리를 찾을 필요가 없음.
 - 모듈 / 컴포넌트
 - IoC, DI, AOP 와 같은 객체 지향 개념을 도입
 - TypeScript : Typing

NestJs 란 ? : web framework?

- 웹 개발에 필수적인 요소들을 묶어 개발자들이 쉽게 쓸 수 있게 한 것.
- 웹 개발에 필요한 것을 일일이 개발자가 할 필요 없게 되어 개발자의 시간과 노력이 감소함.
 - 데이터베이스 연결
 - 유효성 검사
 - 로깅
 - ...

프레임워크	개발언어	설명
React	Javascript TypeScript	SPA(Single Page Application)나 모바일 앱 개발에 사용된다. 최근 몇 년간 가장 인기있는 프론트엔드 프레임워크 가상 DOM(Virtual Document Object Model)을 사용한다 페이스북이 주도하는 공동체에서 유지 보수되고 있다
Vue.js	Javascript TypeScript	React.js와 함께 인기있는 프론트엔드 프레임워크 SPA를 구축할 수 있다 MVVM 패턴에서 VM(View Model)에 해당하는 라이브러리 속도가 빠르다
Angular	TypeScript	SPA를 위한 프레임워크면서 SSR을 지원한다 모듈/컴포넌트 기반으로 작성하며 재사용성 높은 SW를 만들수 있다 2021년 말 Angular 1.x인 AngularJS의 LTS 지원이 중단되었다.
Svelte	Javascript TypeScript	React, Vue.js를 제치고 최근 인기순위 1위로 등극한 프론트엔드 프레임워크 가상 DOM을 사용하지 않는다 정은 코드로 구현이 가능하고 이를 인한 유지 보수성이 좋다
Express	Javascript TypeScript	가장 많은 사용자를 가지고 있는 Node.js 기반 백엔드 프레임워크 가볍게 서버를 구동시킬 수 있다 NestJS와 같이 Express를 기반으로 하는 프레임워크도 존재한다
Spring	Java Kotlin	국내에서 인기가 높은 자바기반 프레임워크. 전지정부프레임워크를 이용하는 프로젝트를 수행하기 위해서는 스프링을 알아야 하기 때문에 그 영향이 크다 IoC, DI, AOP와 같은 객체 지향 프로그래밍 기법을 쉽게 적용할 수 있다
STRUTS	Java	자바기반의 JSP만을 위한 프레임워크 MVC 모델 기반의 웹 애플리케이션을 쉽게 작성할 수 있게 해 준다 한 때 JSP가 많이 사용되었으나 처음 배우는 독자라면 추천하지 않는다
Django	Python	파이썬 기반 웹 프레임워크의 표준이라 할 수 있다 MVC 패턴을 사용한다
Gin	Go	Golang 기반의 웹 프레임워크



'framework'의 번역

NestJs 란 ? : NodeJs 기반의 웹 프레임워크

Node.js 기반 웹 프레임워크가 갖춰야 할 필수 기능이라면 다음과 같은 것들이 있습니다.

- 최신 Ecma Script 지원
- Typescript (선택사항이나 사용 추세가 계속 늘어나고 있음)
- CORS
- HTTP 헤더 보안 (Express는 helmet을 사용)
- Configuration
- Interceptor
- Middleware
- Scheduling
- Logging
- Testing
- Swagger 문서화
- ORM

NestJs 란 ? : Express vs. NestJS

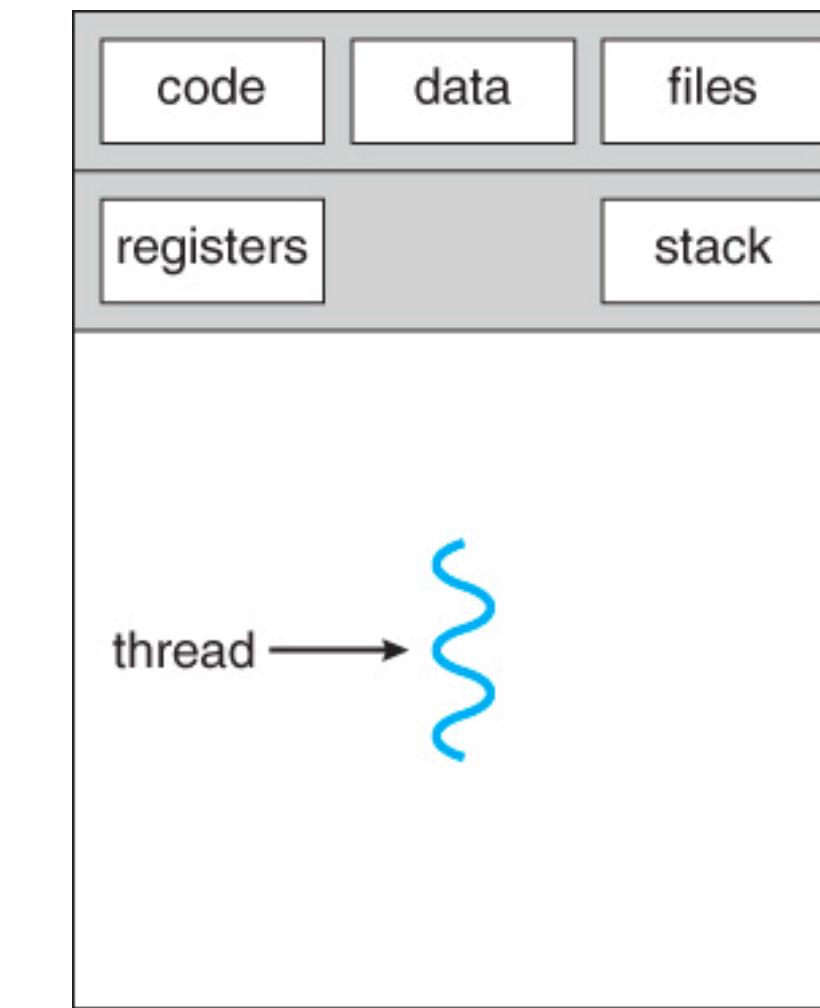
구분	Express	NestJS
유연함, 확장성	<p>Express는 가볍게 테스트용 서버를 띄울수 있습니다. 아이디어를 빠르게 검증하는 데에는 좋겠지만 단순하고 자유도가 높은 만큼 자기에게 맞는 라이브러리를 찾기 위해 제품을 팔아야 합니다. 보일러 플레이트를 미리 얹어 놓은 깃허브 리포지토리들이 있으니 이를 활용해도 좋습니다.</p>	<p>미들웨어, IoC, CQRS 등 이미 많은 기능을 프레임워크 자체에 포함하고 있습니다. 사용자는 문서를 보고 쉽게 따라할 수 있습니다. 원하는 기능이 없다면 다른 라이브러리를 적용해서 사용하면 됩니다.</p>
TypeScript 지원	<p>추가 설정을 통해 사용 가능합니다.</p>	<p>기본 설정입니다. 바닐라 자바스크립트¹로도 작성 가능합니다.</p>
커뮤니티	<p>가장 큅니다.</p>	<p>꾸준히 증가하고 있습니다.</p>

NestJs 란 ? : NodeJs 란?

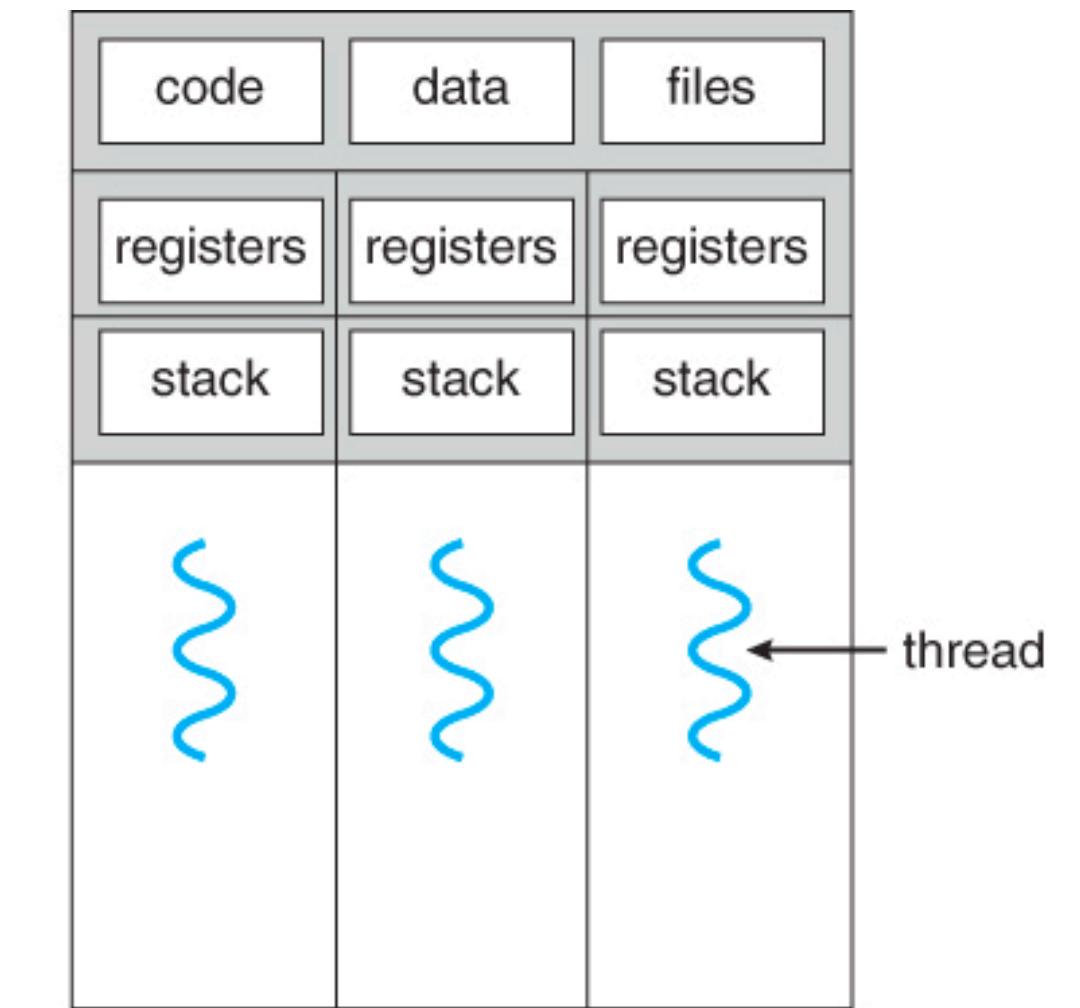
- NestJs 의 소스코드 => Express 소스코드
- 따라서, Node.js의 동작원리를 이해하는 것은 중요하다.
- Node.js
 - Javascript는 원래, 브라우저에 내장된, 화면을 동적으로 구성하기 위한 스크립트 용 프로그래밍 언어
 - Javascript를 브라우저 밖에서도 실행할 수 있도록하는 Runtime => Node.js
 - HTML – <script>에서 벗어나 서버를 구동할 수 있게됨
 - Single Thread, But 비동기 / 동시성 / Event-driven / Non-blocking I/O

Node.js와 Javascript 의 특징 : 비동기와 동시성

- Process / Single vs Multi Thread
- Javascript == Single Thread : 한 번에 하나의 작업만, 끼어들기 불가능
 - 동시성, Non-blocking 은 어떻게 ? => Node.js (Javascript Runtime / Browser)
 - Thread Pool (libuv)
- Single Thread :
 - 특징 및 장점
 - 하나의 스레드를 이용하는 것처럼 코딩
 - Clustering => Process fork 도 가능 !
 - 멀티스레드에 비해 높은 효율성
 - 컴파일러 언어의 처리 속도에 비해 낮은 성능



single-threaded process



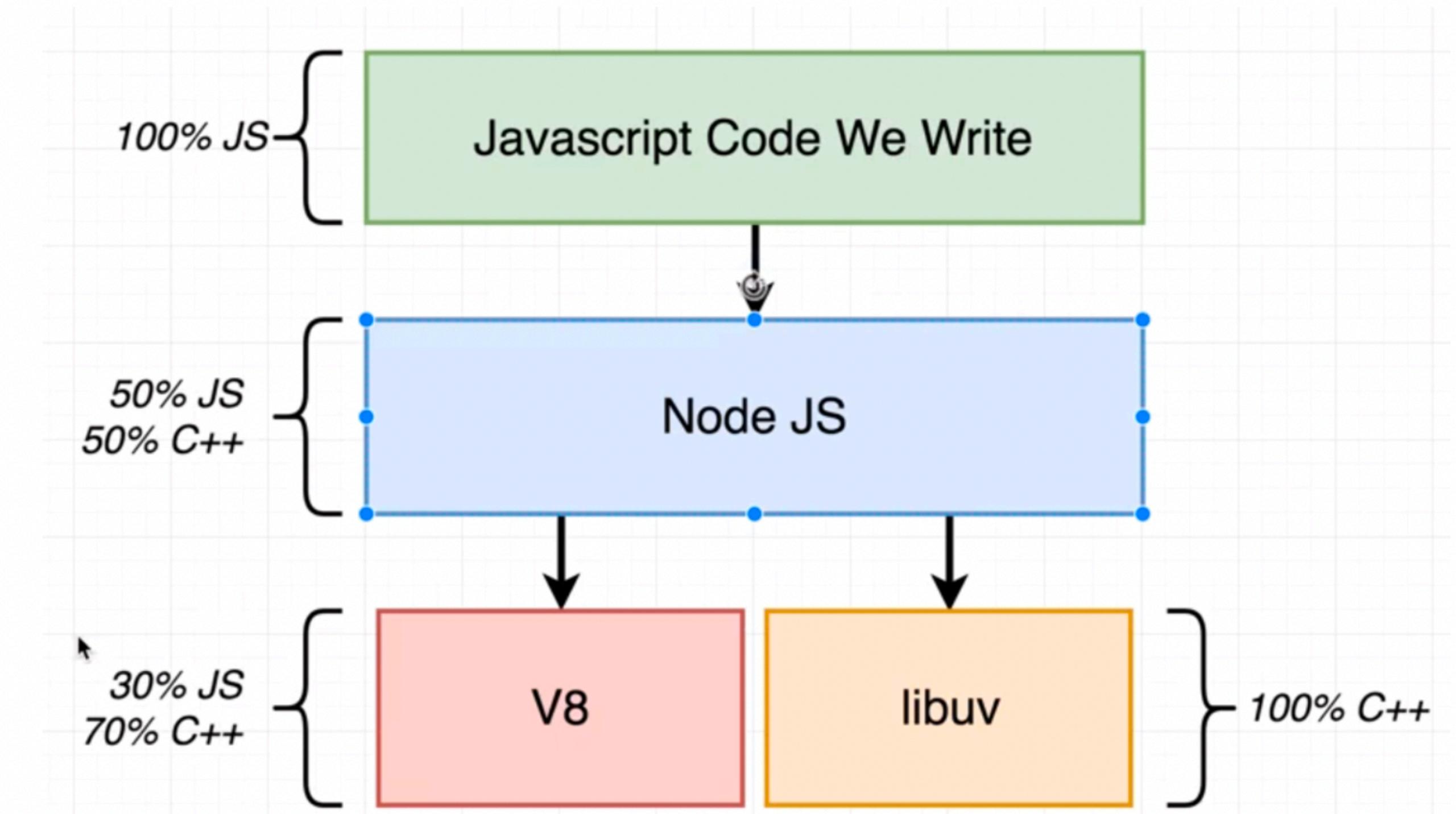
multithreaded process

Node.js와 Javascript 의 특징 : 비동기와 동시성

```
const foo = () => {
  bar()
  console.log('foo')
}

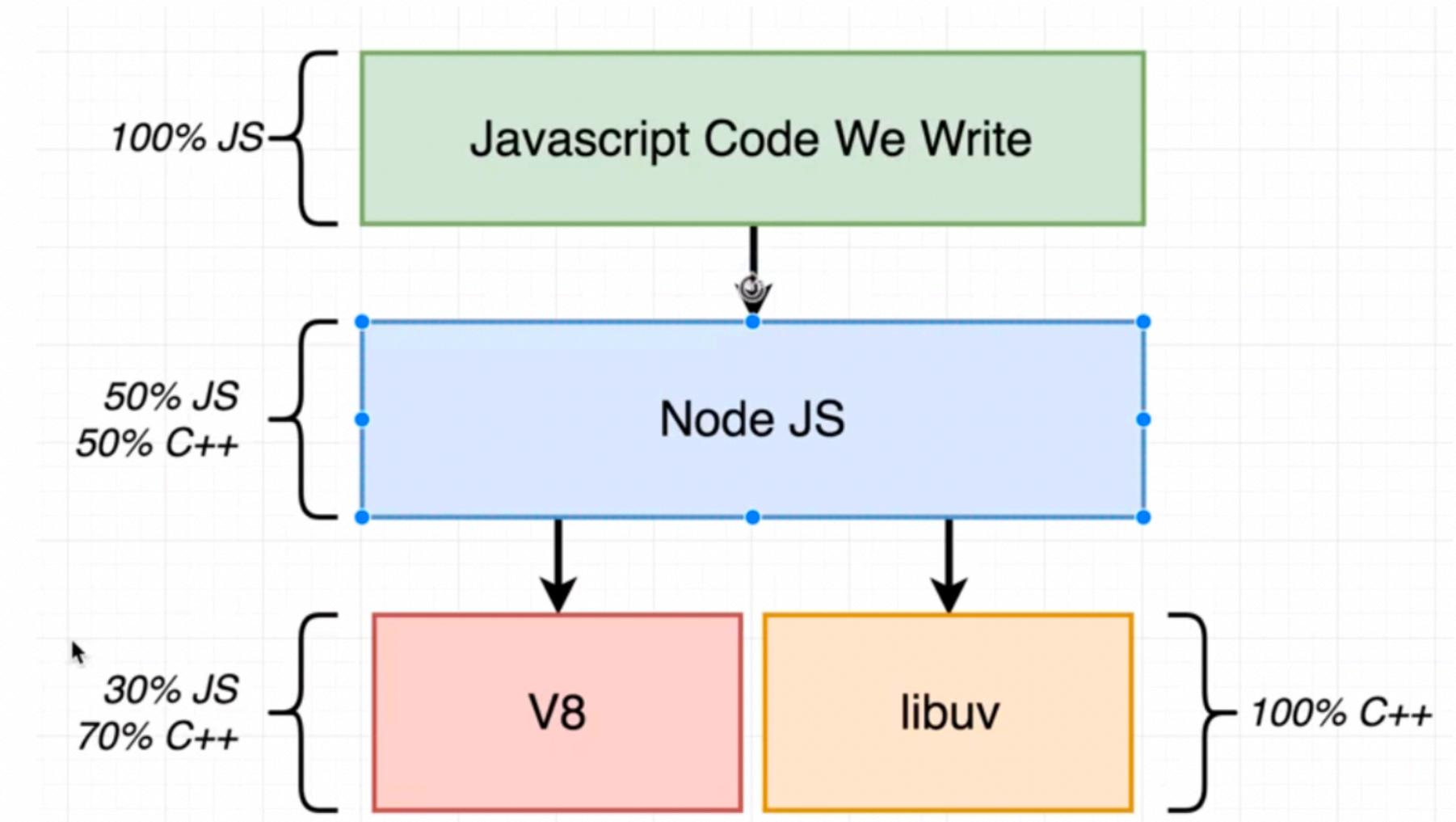
const bar = () => {
  console.log('bar')
}

foo();
console.log('foo and bar')
```



Node.js와 Javascript 의 특징 : Event-driven

- Single Event Loop + Single Thread
- Libuv
 - Event-driven, Non-blocking I/O
 - Callback function

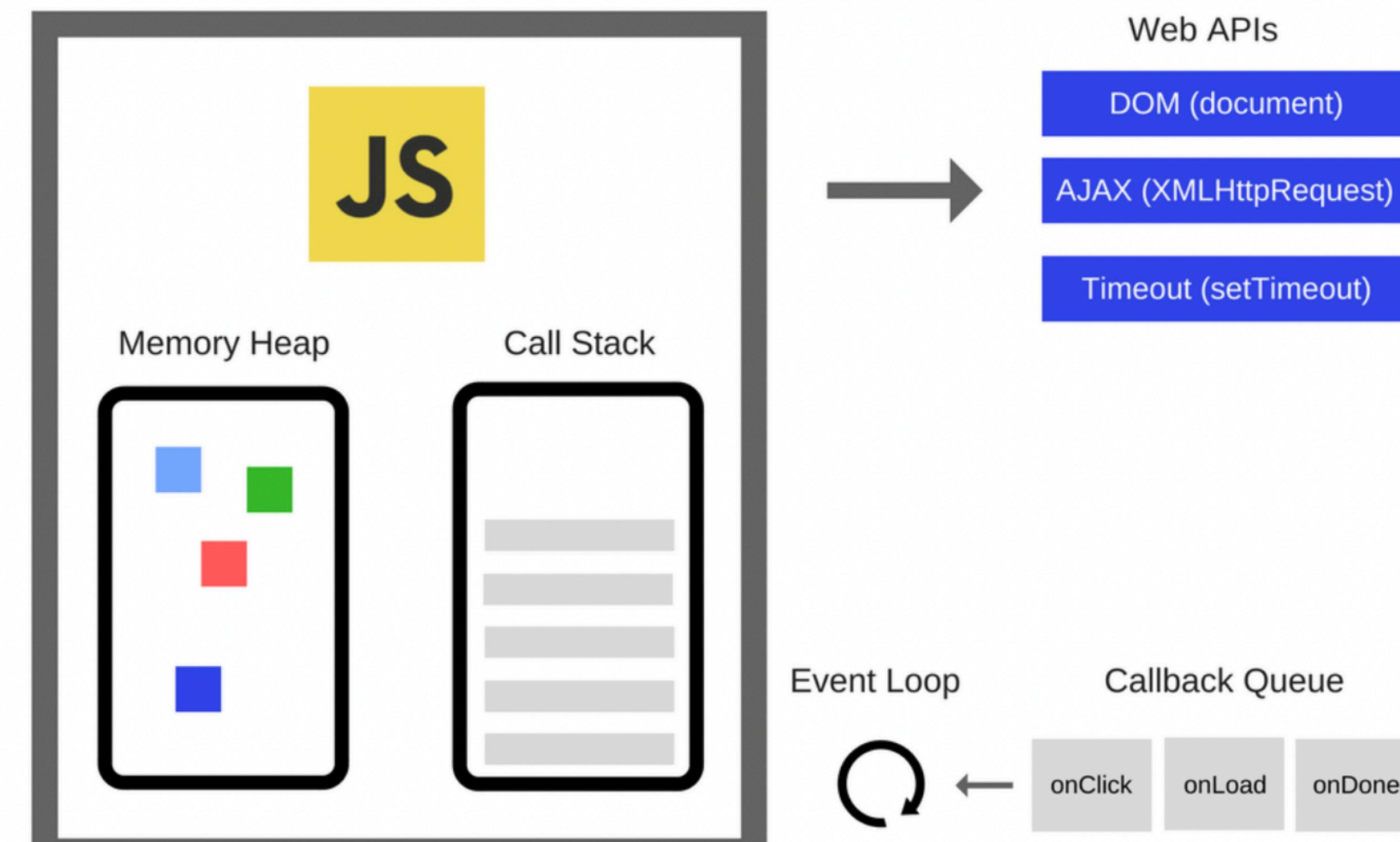


```
console.log('첫번째로 실행됩니다. ');

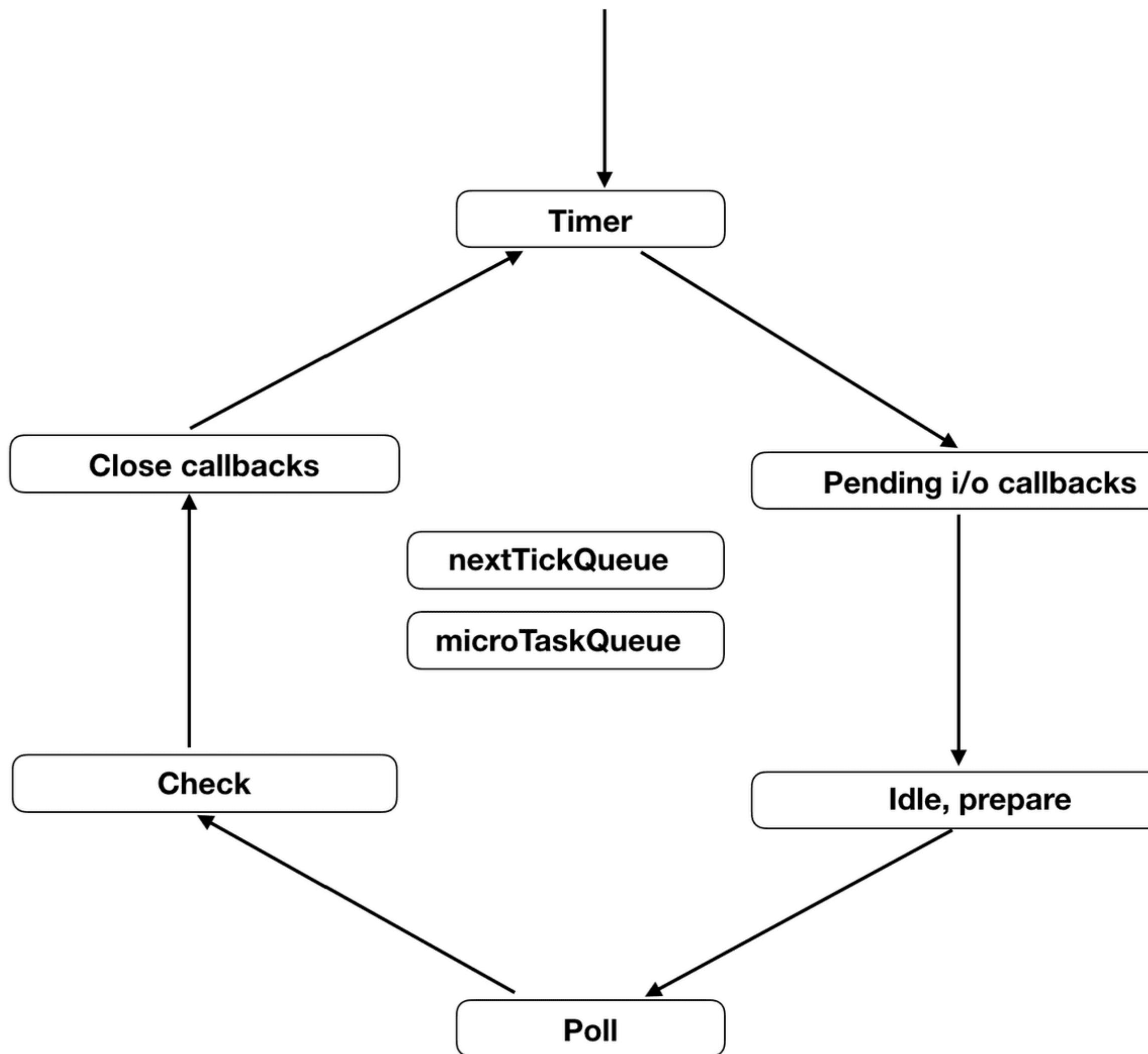
setTimeout(() => console.log('최소 0초보다 늦게 실행됩니다.'), 0);

console.log('언제 실행될까요?');
```

Node.js와 Javascript 의 특징 : Event-driven



Node.js와 Javascript 의 특징 : Event Loop



```
setTimeout(() => {  
  console.log('timeout');  
}, 0);
```

```
setImmediate(() => {  
  console.log('immediate');  
});
```

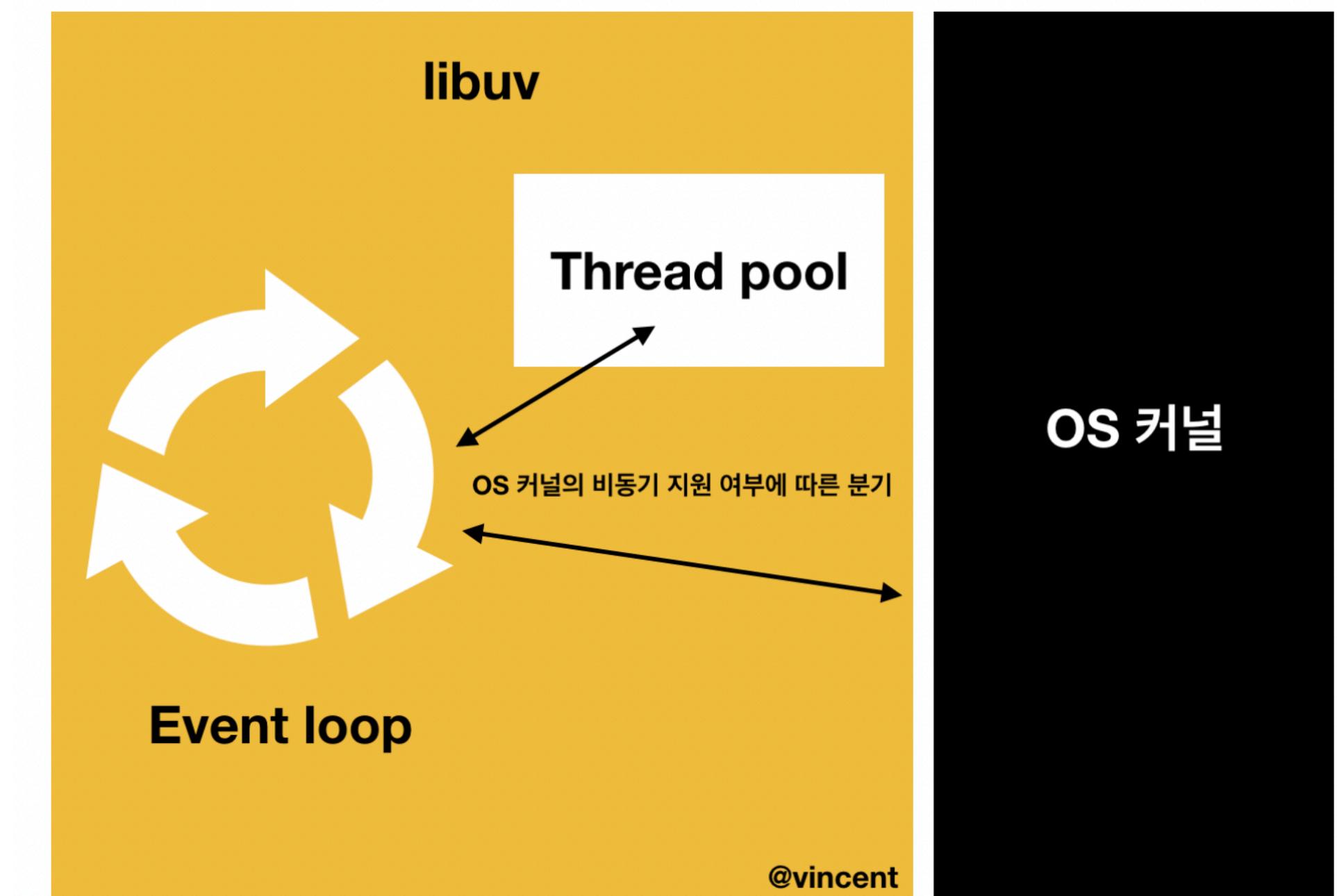
```
const fs = require('fs');  
  
fs.readFile(__filename, () => {  
  setTimeout(() => {  
    console.log('timeout');  
  }, 0);  
  setImmediate(() => {  
    console.log('immediate');  
  });  
});
```

Node.js와 Javascript의 특징 : Non-blocking I/O

- Non-blocking I/O 이벤트 기반 비동기 방식
 - 입력은 하나의 스레드에서 받지만, 입력받은 순서대로 작업을 처리하지 않고, 먼저 처리된 결과를 이벤트로 반환 (Event Loop, Callback Queue, Call Stack)
 - Callback function : Promise, Async/Await

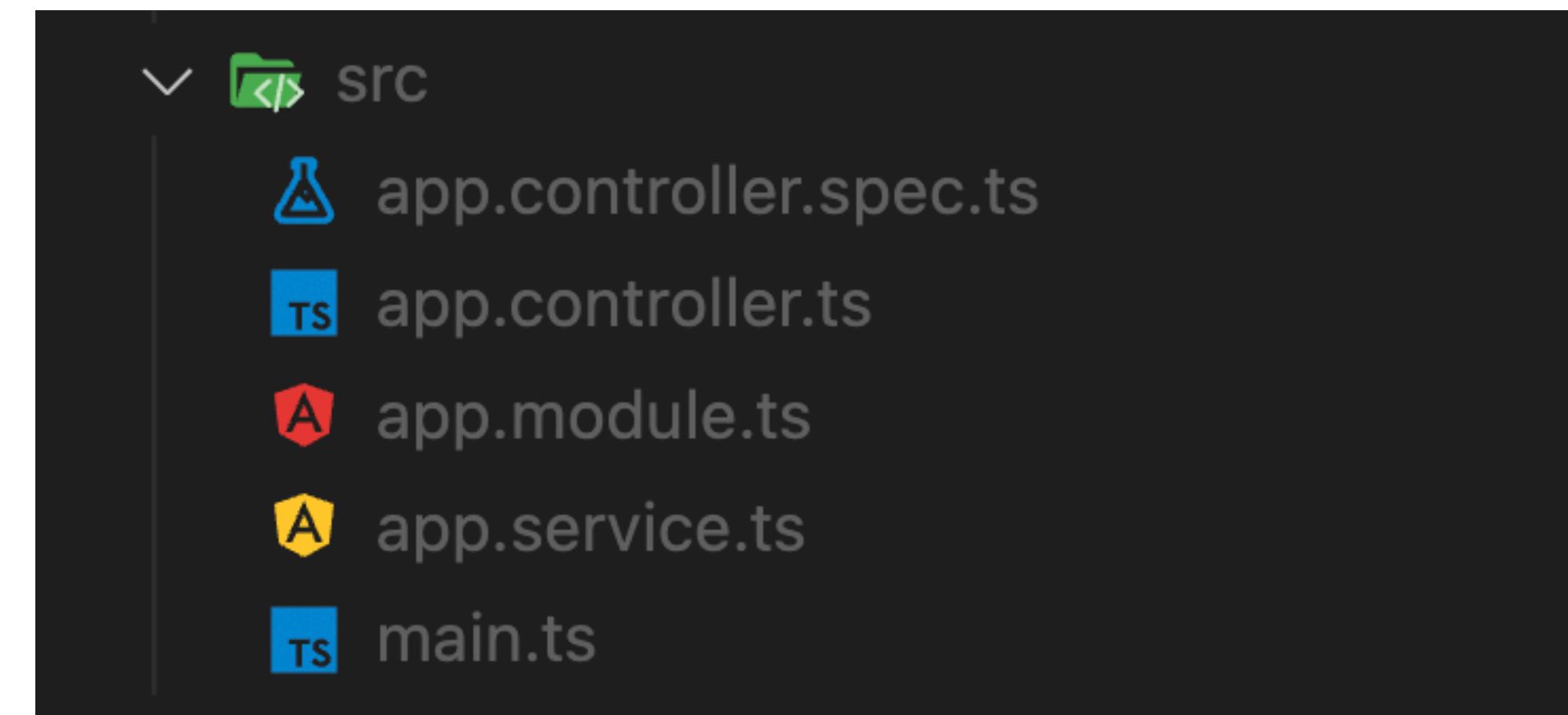
- 각각의 작업에 대해 걸리는 시간은 다음과 같고, 아래와 같은 순서로 작업에 대한 I/O 요청이 들어왔다고 하자.

A : 10초
B : 3초
C : 10000초
D : 10초



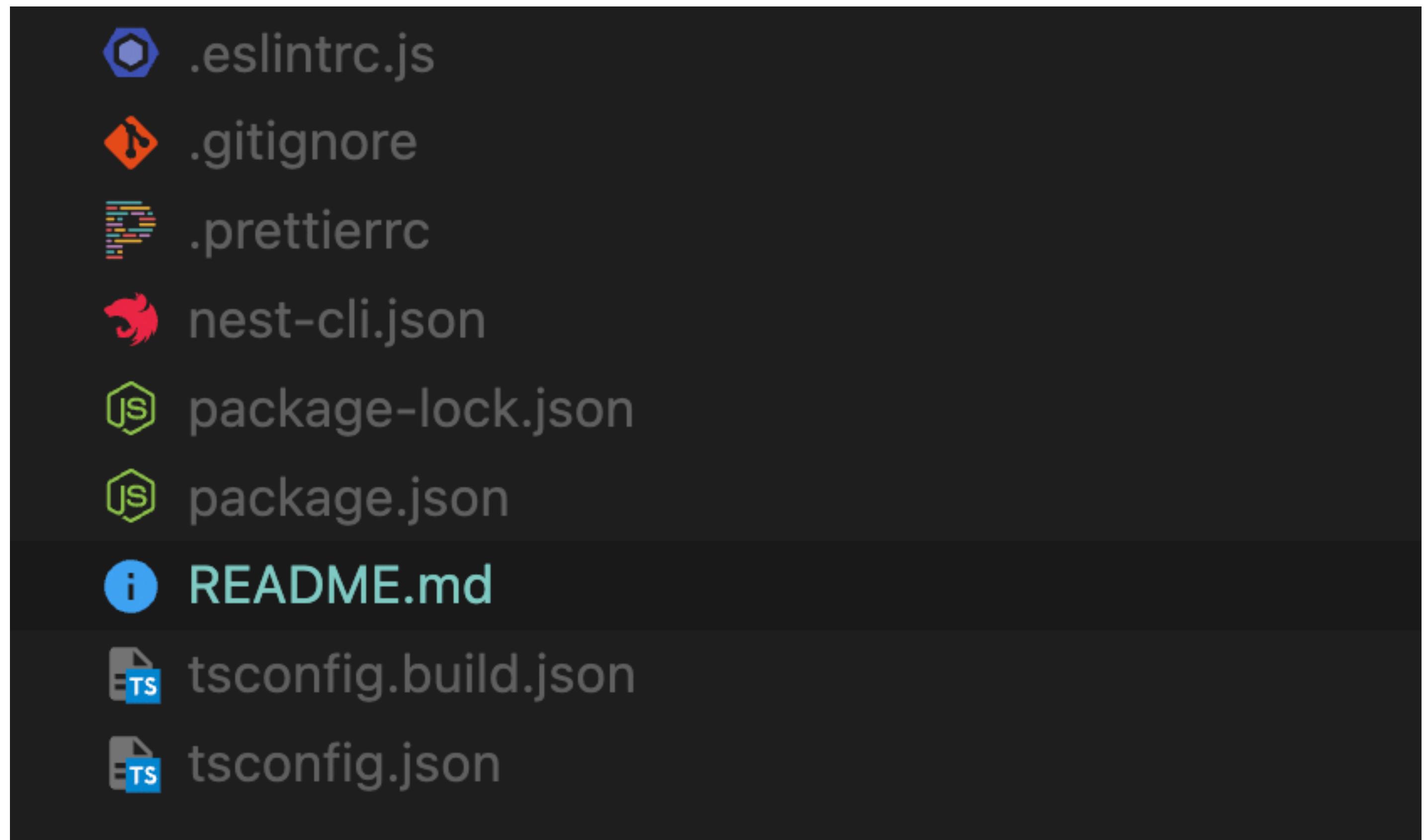
NestJs의 구조 : Module / Component 구조

- **src** : 실제 서버를 구성하는 **Code**와 **Test Code**
 - **app.module.ts**
 - **app.controller.ts**
 - **app.service.ts**
 - **main.ts**
- ***.spec.ts / ./test** : unit test, e2e test, integration test를 할 수 있는 code



NestJs의 구조

- **.eslintrc.js**
- **.gitignore**
- **.prettierrc**
- **nest-cli.json**
- **package-lock.json**
- **package.json**
- **README.md**
- **tsconfig.build.json**
- **tsconfig.json**

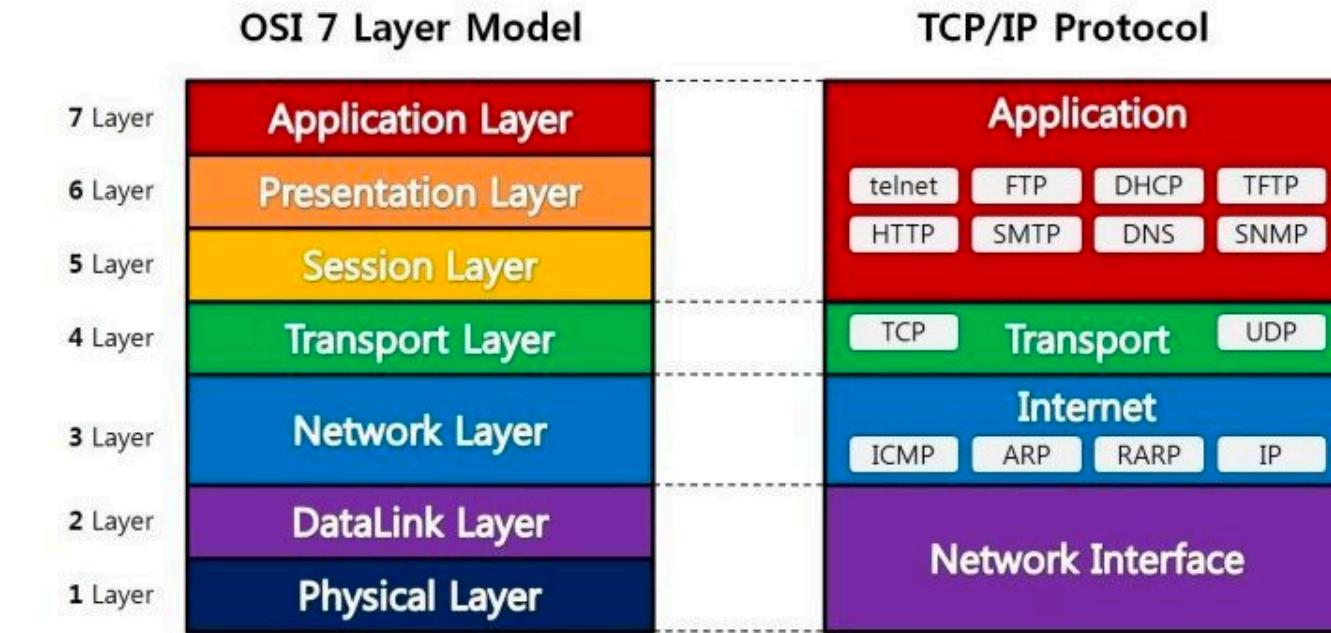


NestJs의 구조 - NPM ? Yarn ?

- NodeJS 패키지 매니저
- 프로젝트에 사용되는 library의 버전을 관리하고 / 라이브러리를 설치하고 / 라이브러리 간 의존성을 관리하기 위한 패키지 매니저.
- NPM init
- NPM install (npm i)
- package.json
- package-lock.json
- Yarn 은 조금 더 보안적으로 업그레이드 된 패키지 매니저

CURL

- CURL ?
 - Command line 용 데이터 통신 / 웹 개발 tool
 - HTTP/HTTPS/FTP/SMTP 등 주요한 Application Layer 프로토콜을 지원
 - Proxy, Header, Cookie 등의 세부 옵션까지 쉽게 설정 가능
 - 언제 쓰느냐 ? Client를 코딩하기 전에, curl 명령어로 서버 동작을 먼저 확인하여 빠르게 개발을 진행할 수 있다.
 - Linux 와 Mac OS에서는 기본적으로 설치 돼있다!



CURL Example

```
curl [options...] <URL>
```

Short Option	Long Option	Description
-o	—output FILE	curl 은 remote에서 받아온 데이터를 기본적으로는 콘솔에 출력한다. -o 옵션 뒤에 FILE을 적어주면 해당 FILE로 저장한다. (download 시 유용)
-H	—header	HTTP Header에 헤더 추가 <KEY : VALUE>
-X	—request	request 시 사용할 method 종류(GET, POST, PUT, PATCH, DELETE)를 기술한다.
-d	—data	HTTP Post data
-A	—user-agent <name>	Send User-Agent <name> to server
-O	—remote-name	file 저장시 remote의 file 이름으로 저장한다. -o 옵션보다 편리하다.
-T	—upload-file <file>	Transfer local FILE to destination

CURL Example

서버의 응답을 결과물로 출력

```
# 서버 응답의 헤더를 -i 옵션을 사용하여 출력할 수 있습니다  
$ curl -i https://www.naver.com
```

서버의 파일 이름 변경 없이 다운로드

```
$ curl -O https://example.com/test.zip
```

Http Header에 Bearer Token을 포함하여 POST

- http://example.com/api로 Bearer Token을 Authorization 값으로 실어서 data.json 내용으로 POST 하는 명령어입니다. OAuth 2.0 명령어 테스트 시 유용합니다.

```
curl -X POST \  
-H Content-Type:application/json \  
-H Authorization: Bearer <BEARER TOKEN> \  
-d @data.json \  
http://example.com/api
```

Postman

Curl 을 쉽게, 시각적으로

- Postman Installation
 - <https://www.postman.com/>
- Postman 으로 구글에 HTTP GET 요청을 해봅시다.

Assignment 2 : User API 개발

- Feature/assignment2 브랜치에서 commit convention을 지키면서 작업
- 오늘 배운 NestJs의 구조를 바탕으로 설계하는 User API
- 원래는 User를 Database에서 CRUD 하지만 Memory의 Dummy Data를 이용해서 조회.
- User Module, User Controller, User Service
- Module : **nest g module users**
- Controller : **nest g controller users**
- Service : **nest g service users**

Reference

Nest

- [**https://github.com/dextto/book-nestjs-backend**](https://github.com/dextto/book-nestjs-backend)
- [**https://wikidocs.net/147248**](https://wikidocs.net/147248)
- [**https://nestjs.com/**](https://nestjs.com/)