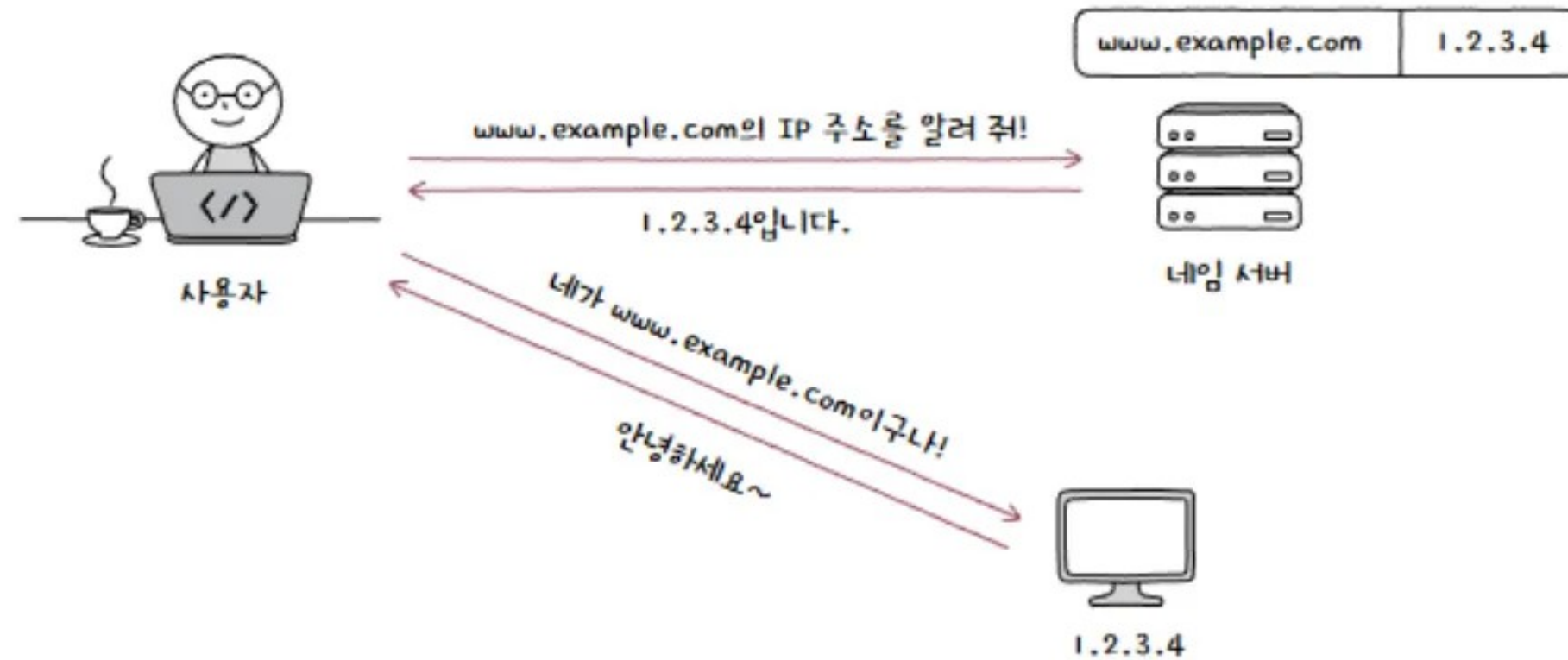


5장. 응용계층

도메인 네임과 네임 서버

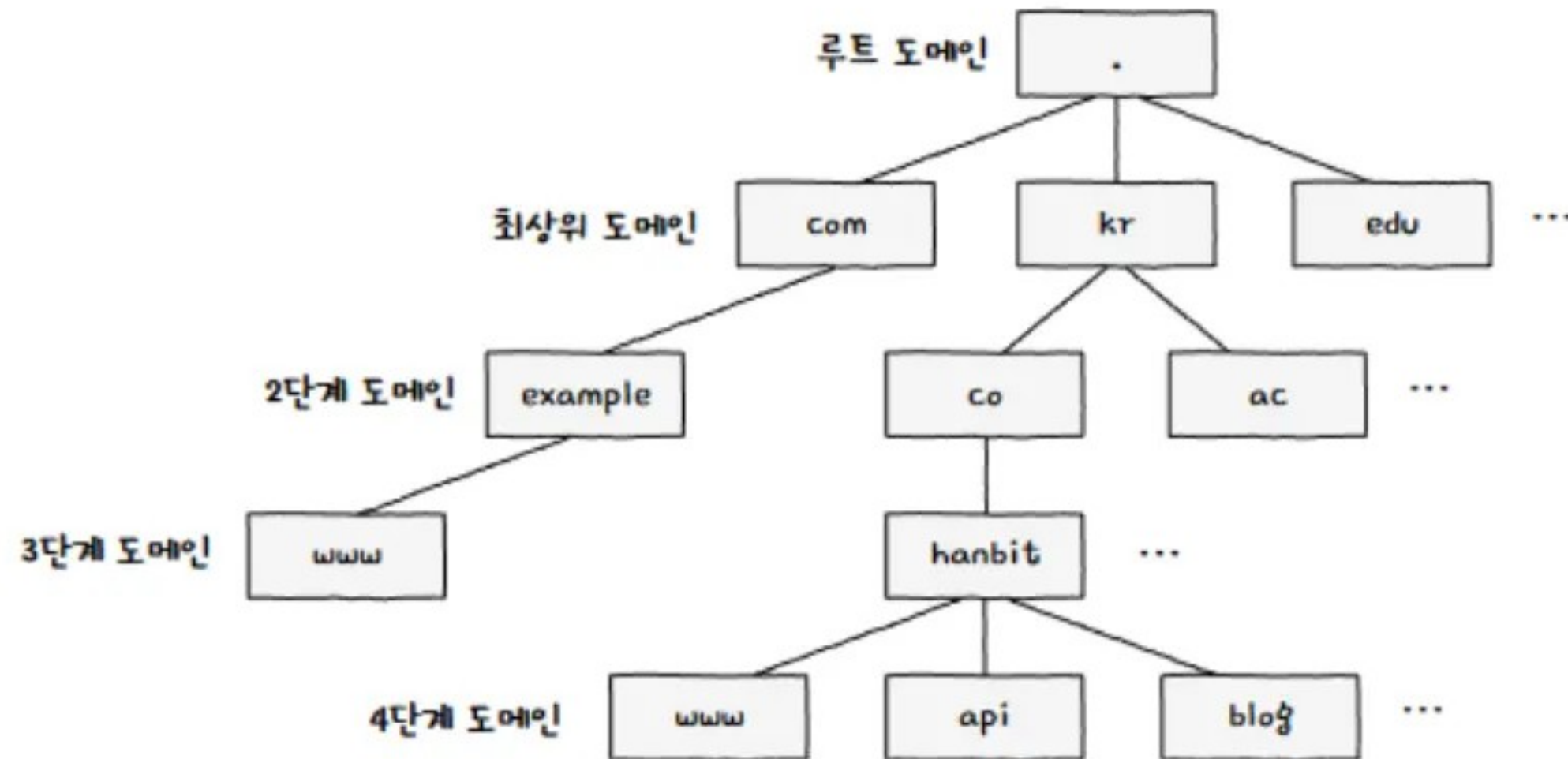


클라이언트는 상대 호스트 특징을 위해 IP 보다 도메인 네임을 사용한다.

www.naver.com과 같은 문자열이 도메인 네임

도메인 네임과 대응하는 IP 주소를 관리하는 곳이 네임 서버

도메인 네임의 구조



www.naver.com

최상위 도메인(TLD) : com

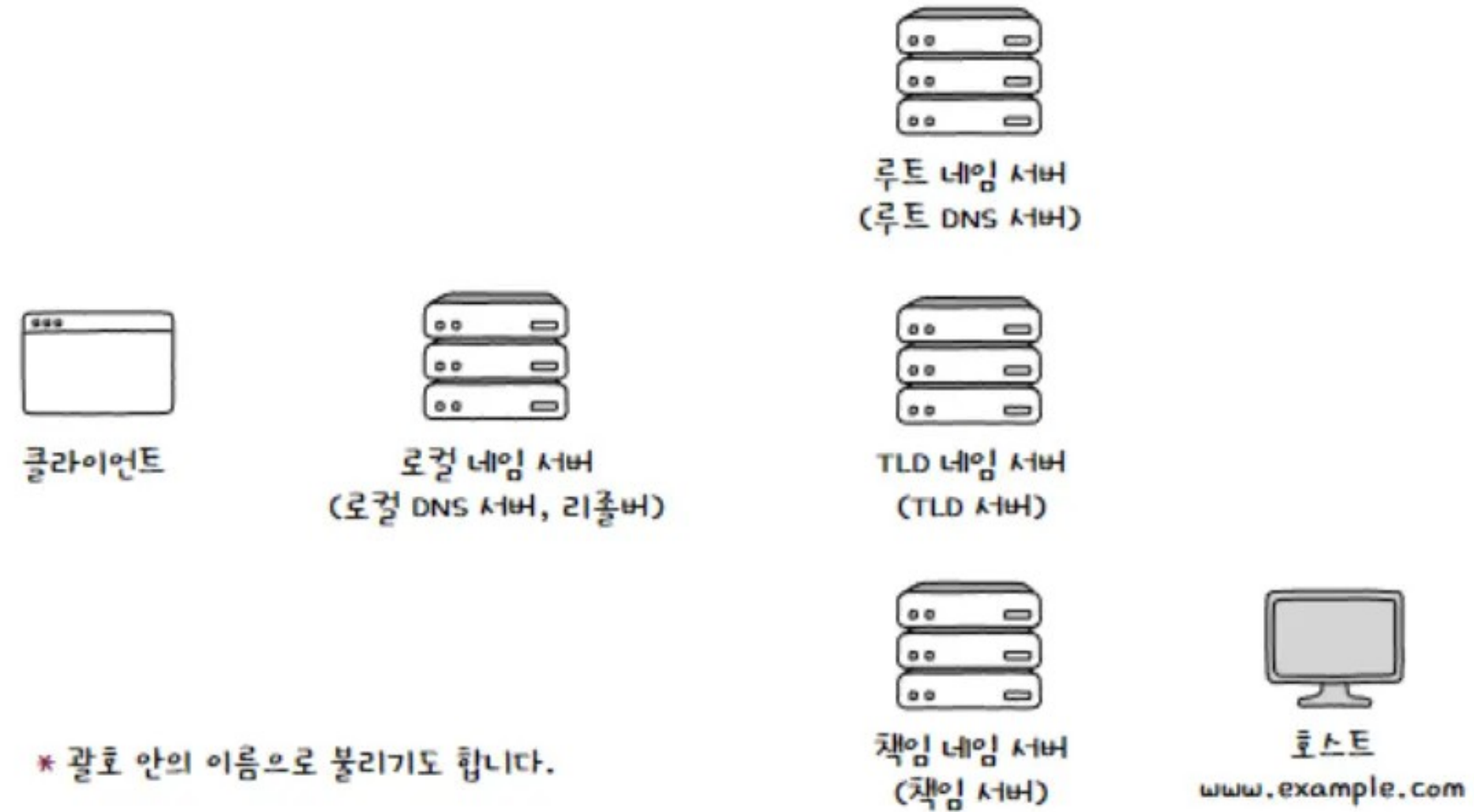
루트 도메인: .

도메인 네임의 구조



계층적이고 분산된 도메인 네임에 대한 관리 체계를
DNS(Domain Name System)이라 부름.

계층적 네임 서버



1. 로컬 네임 서버
2. 루트 네임 서버
3. TLD(최상위 도메인) 네임 서버
4. 책임 네임 서버

계층적 네임 서버 - 로컬 네임 서버



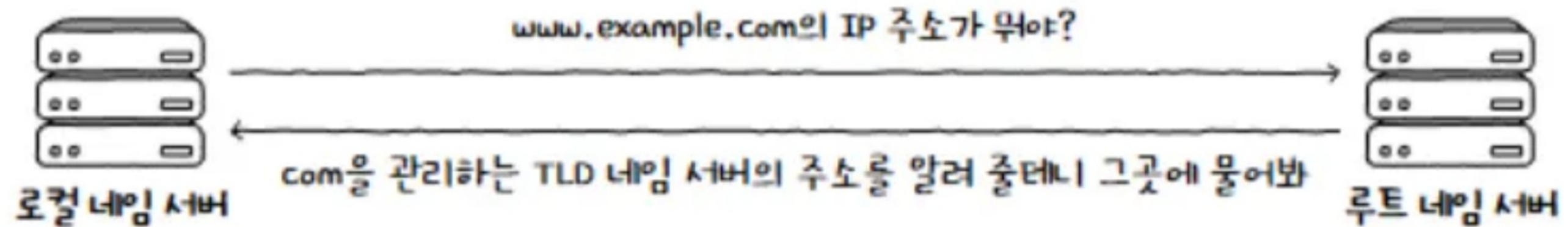
로컬 네임 서버
(로컬 DNS 서버, 리졸버)

도메인 네임을 통해 IP 주소를 알아낼 때 가장 먼저 찾는 네임 서버

로컬 네임 서버 주소는 일반적으로 ISP에서 할당

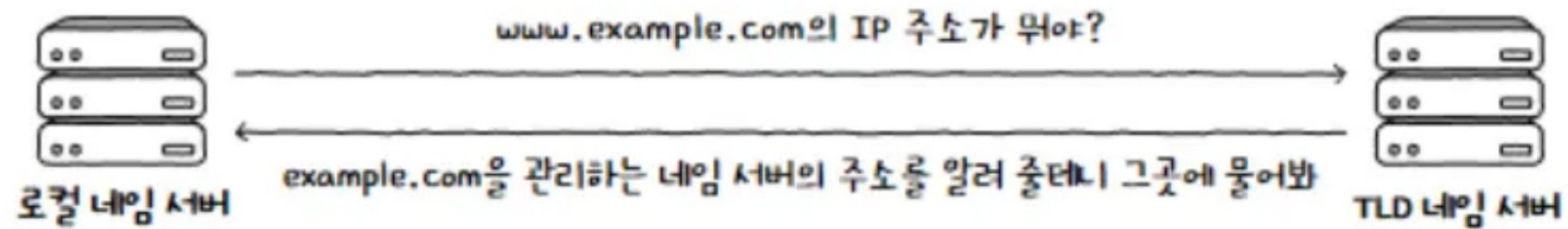
공개 DNS 서버도 이용 가능(구글 8.8.8.8 / 클라우드플레어 1.1.1.1)

계층적 네임 서버 - 루트 네임 서버



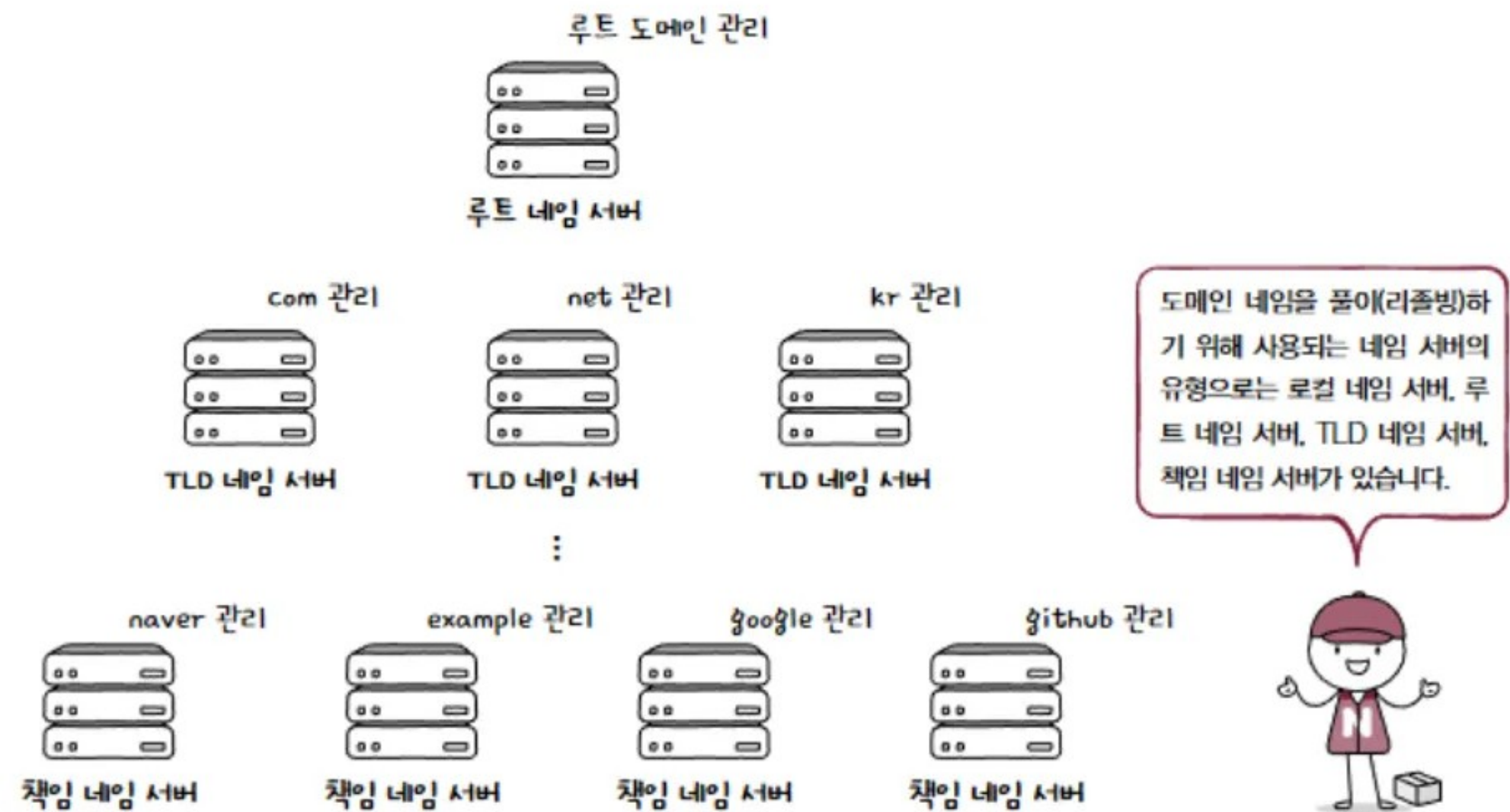
로컬 네임 서버가 대응되는 IP 주소를 모를 때,
루트 네임 서버에 도메인 네임을 질의

계층적 네임 서버 - TLD 네임 서버



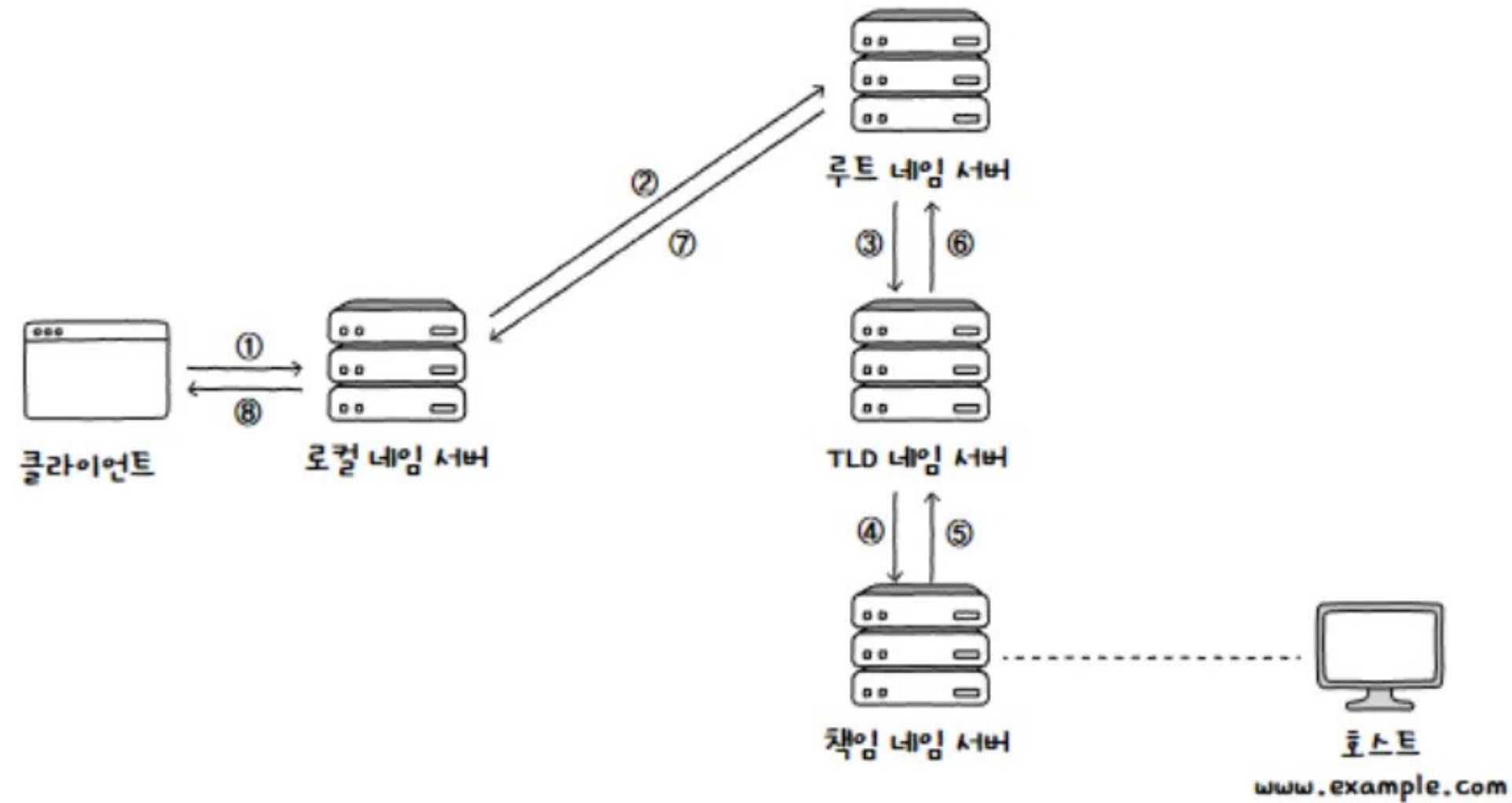
TLD의 하위 도메인 네임을 관리하는 네임 서버의 주소를 반환

계층적 네임 서버 - 책임 네임 서버



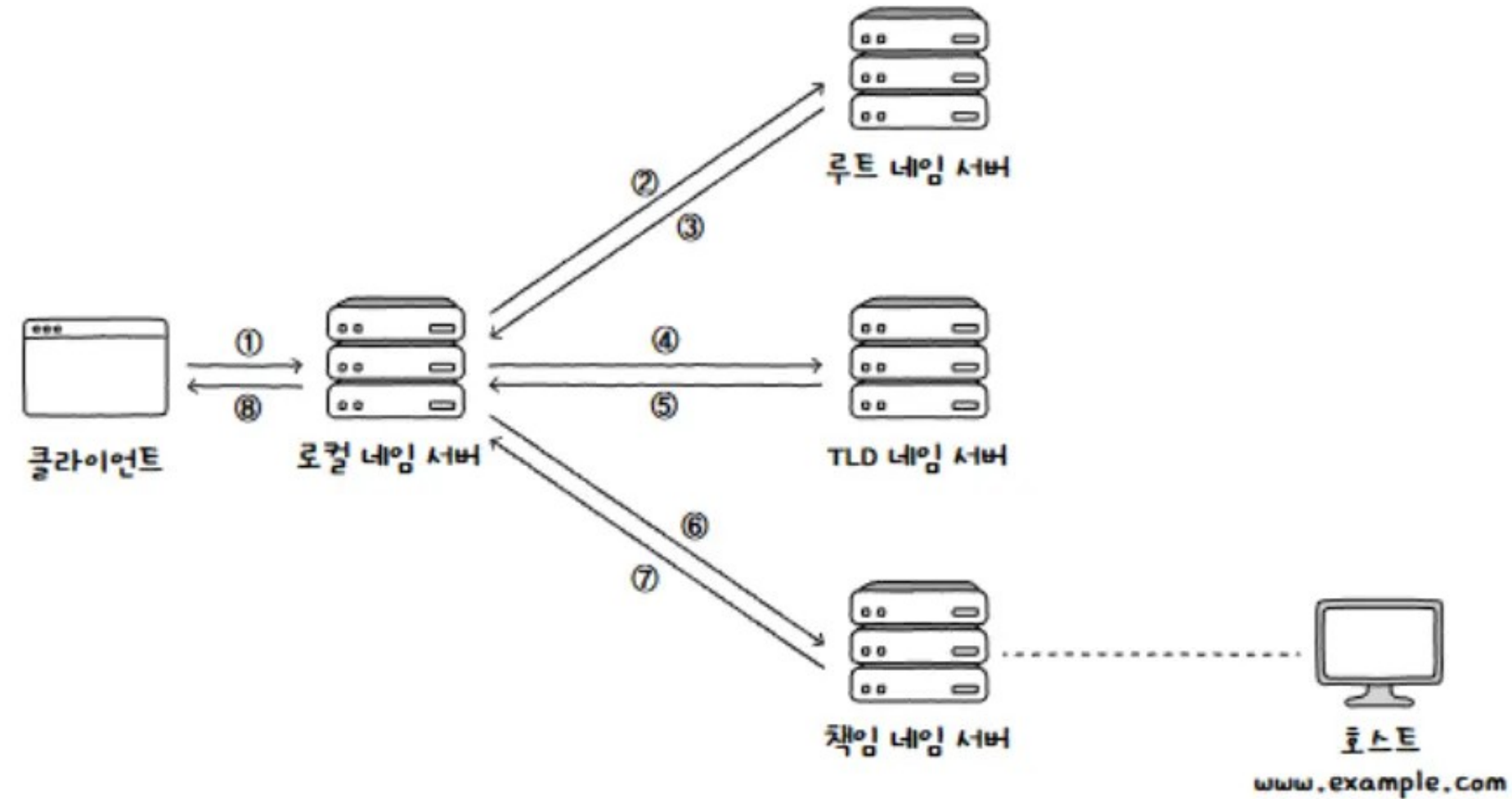
관리하는 도메인 영역의 질의는 바로 답할 수 있는 네임 서버
로컬 네임 서버가 마지막으로 질의하는 네임 서버

DNS Resolving - 재귀적 질의



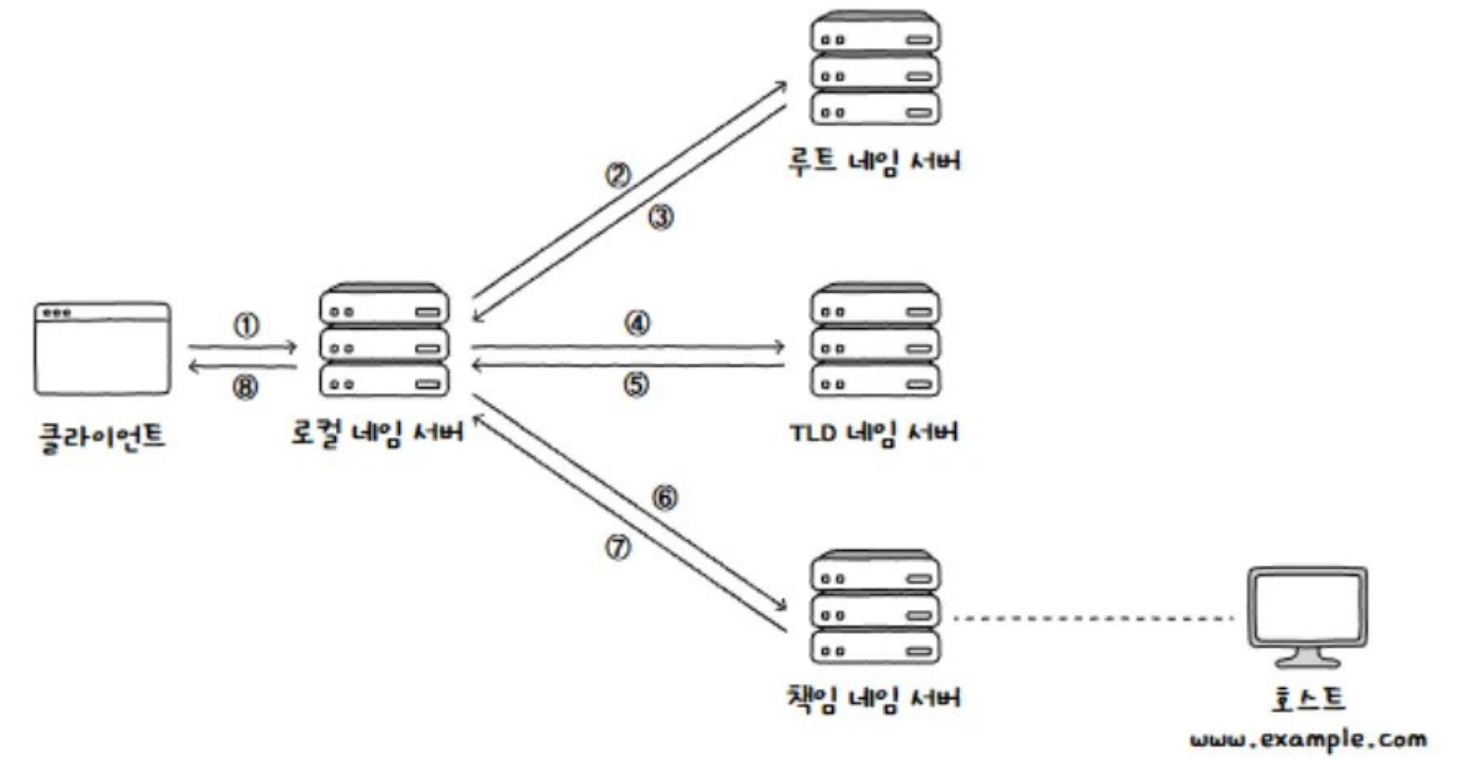
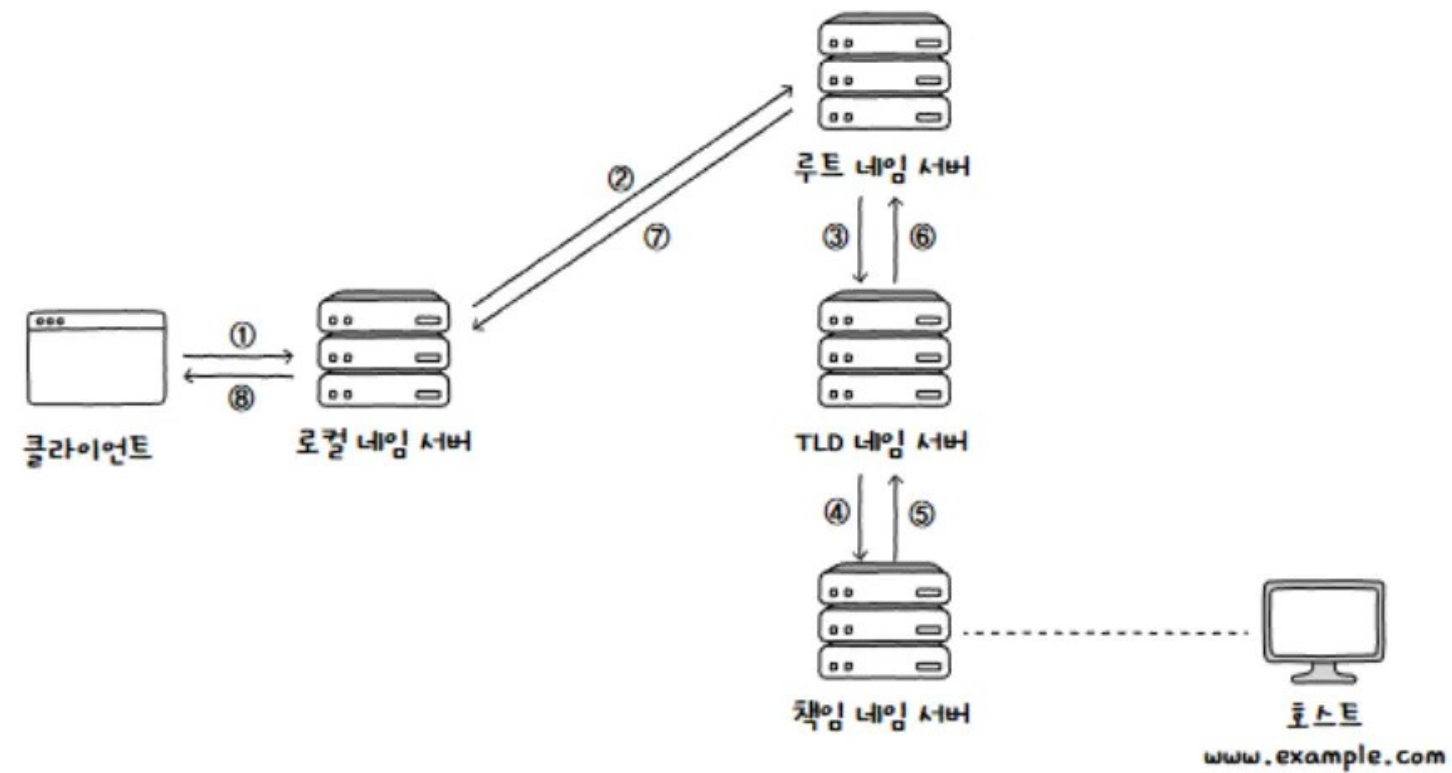
1. 클라이언트가 로컬 네임 서버에 도메인 네임 질의
2. 로컬 네임 서버가 루트 네임 서버에 질의
3. 루트 네임 서버가 TLD 네임 서버에 질의
4. TLD 네임 서버가 다음 단계에 질의

DNS Resolving - 반복적 질의



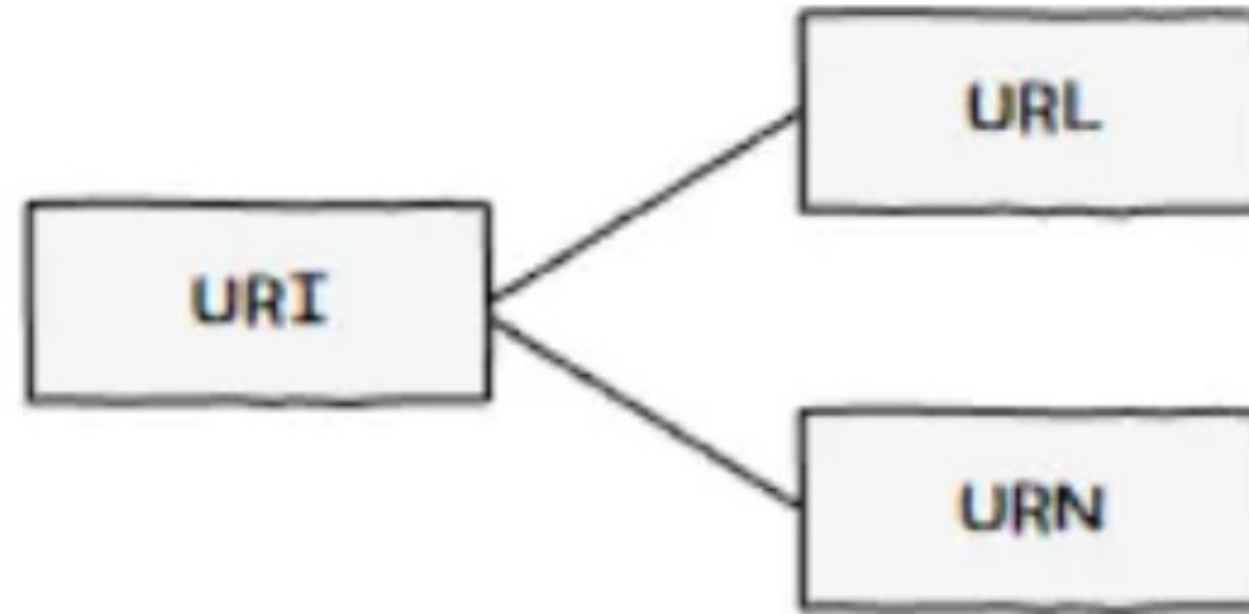
1. 클라이언트가 로컬 네임 서버에 도메인 네임 질의
2. 루트 네임 서버에 질의해서 다음으로 질의할 네임 서버 주소 응답.
3. TLD 네임 서버에 질의해 다음으로 질의할 네임 서버 주소 응답

두 방식의 문제점



하나의 DNS Resolving을 위해 8개 단계를 거침.
오랜 시간 소요, 네트워크 메시지 수 증가.
해결을 위해 실제로는 DNS 캐시 사용

자원을 식별하는 URI



자원: 네트워크상 메시지를 통해 주고받는 대상

URL: 위치 기반 자원 식별

URN: 이름 이용 자원 식별

URL(대부분 사용)

`foo://www.example.com:8042/over/there?name=ferret#nose`

The diagram shows the URL `foo://www.example.com:8042/over/there?name=ferret#nose` with brackets underneath identifying its parts: `foo` is the scheme; `//www.example.com:8042` is the authority; `/over/there` is the path; `?name=ferret` is the query; and `#nose` is the fragment.

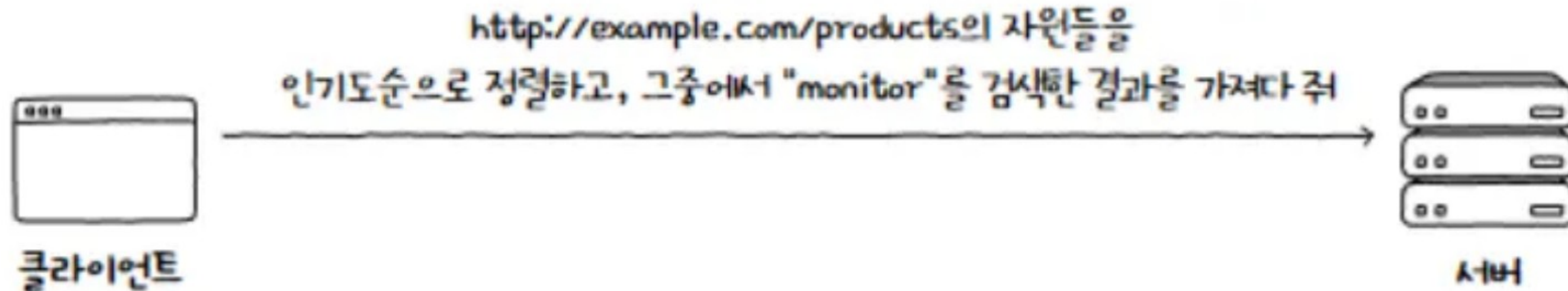
1. scheme: 자원에 접근하는 방법(http, https)
2. authority: 호스트를 특정할 수 있는 정보
3. path: 자원이 위치한 경로가 명시됨.

슬래시를 기준으로 계층적 표현

URL(대부분 사용)

`foo://www.example.com:8042/over/there?name=ferret#nose`

scheme authority path query fragment



`http://example.com/random/path?query=value&query2=value2`

4. query: ?로 시작되는 <키=값> 형태의
쿼리 파라미터를 사용해 질의하는 것.

URN(거의 사용하지 않음)

urn:isbn:9780132350884

자원에 고유한 이름을 붙이는 이름 기반 식별자
하지만 직접적인 접근 불가능. 많이 사용되지 않음.

5-2. HTTP

HTTP의 네 가지 특성



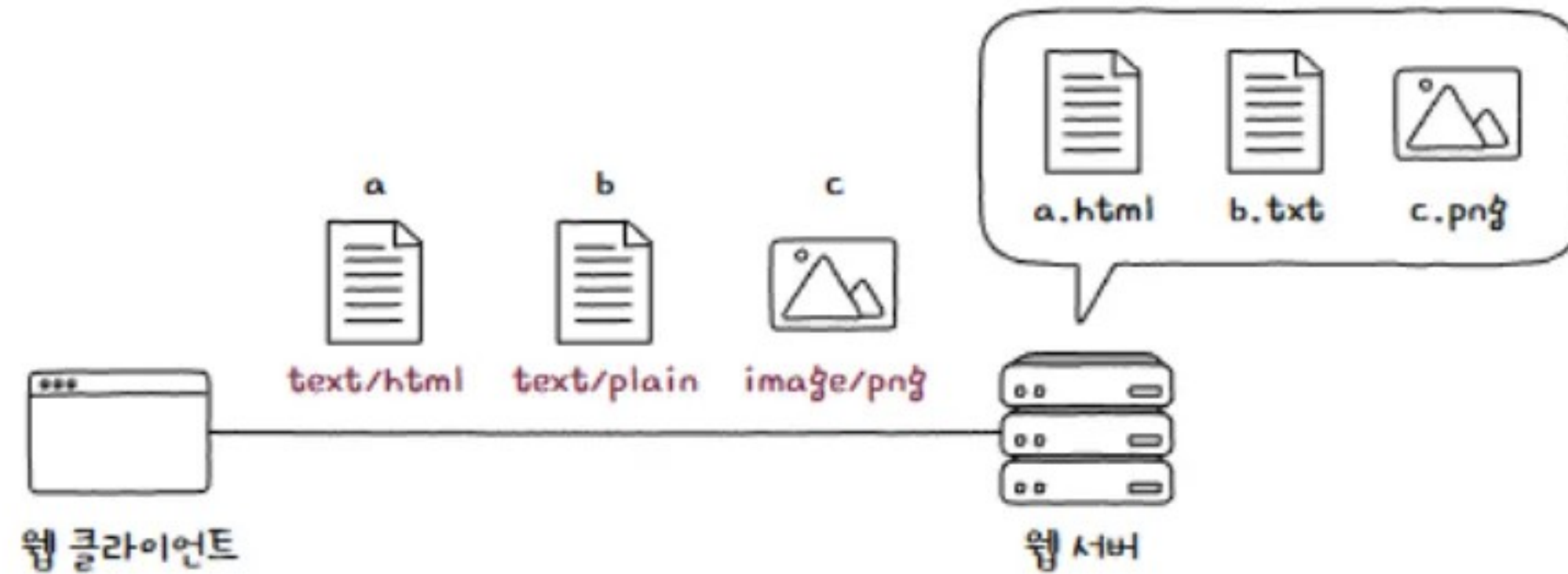
1. 요청과 응답을 기반으로 동작
2. 미디어 독립적
3. 상태를 유지하지 않음
4. 지속 연결 지원

HTTP의 특성 - 요청-응답 기반 프로토콜

Name	×	Headers	Preview	Response	Initiator	Timing
example.com		▼ General				
Request URL:		http://example.com/				
Request Method:		GET				
Status Code:		● 200 OK				
Remote Address:		93.184.216.34:80				
Referrer Policy:		strict-origin-when-cross-origin				
		▼ Response Headers		Raw		
Accept-Ranges:		bytes				
Age:		516704				
Cache-Control:		max-age=604800				
Content-Encoding:		gzip				
Content-Length:		648				
Content-Type:		text/html; charset=UTF-8				
Date:		Thu, 05 Oct 2023 04:02:07 GMT				
Etag:		"3147526947"				
Expires:		Thu, 12 Oct 2023 04:02:07 GMT				
Last-Modified:		Thu, 17 Oct 2019 07:18:26 GMT				
Server:		ECS (sec/976A)				
Vary:		Accept-Encoding				
X-Cache:		HIT				
		▼ Request Headers		Raw		
Accept:		text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7				
Accept-Encoding:		gzip, deflate				
Accept-Language:		ko-KR;q=0.9,en-US;q=0.8,en;q=0.7				
Connection:		keep-alive				
Host:		example.com				
If-Modified-Since:		Thu, 17 Oct 2019 07:18:26 GMT				
If-None-Match:		"3147526947+gzip"				
Upgrade-Insecure-Requests:		1				
User-Agent:		Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36				

클라이언트와 서버가 서로 HTTP 요청 메시지와 HTTP 응답 메시지를 주고받음.
그래서 HTTP 요청과 응답 메시지는 메시지 형태가 다르다.

HTTP의 특성 - 미디어 독립적 프로토콜

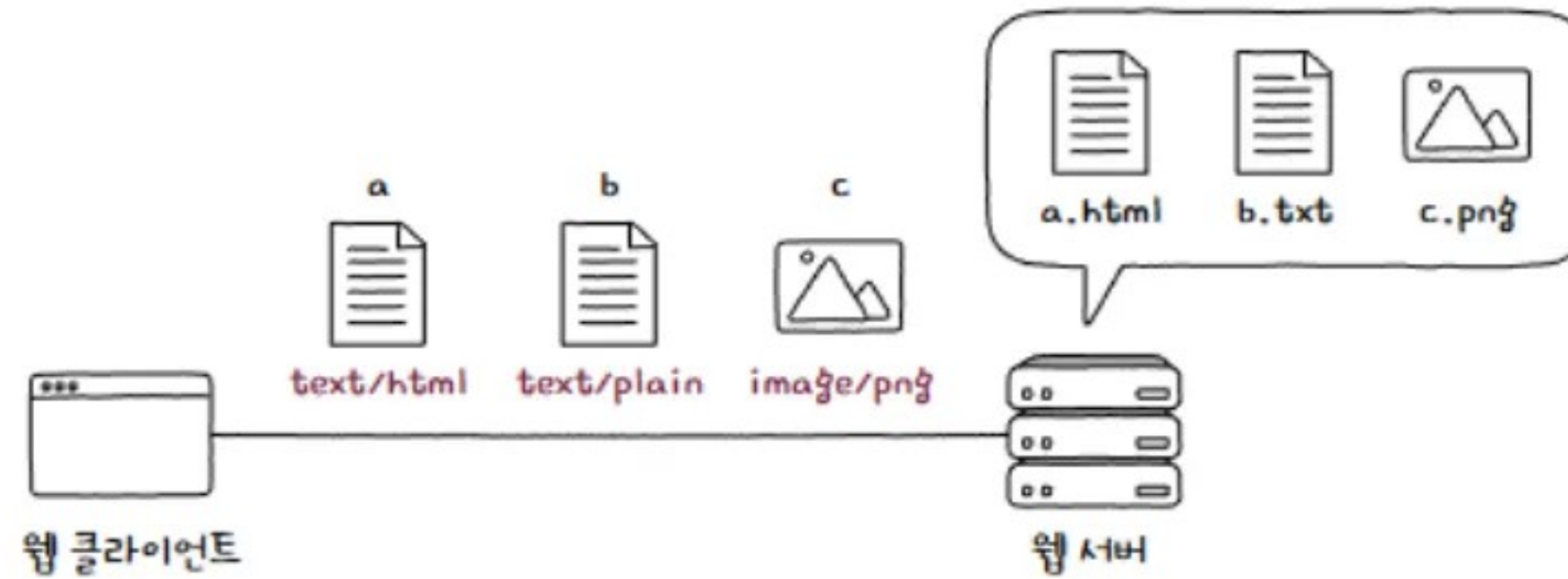


HTTP는 자원을 주고받을 수단의 역할만 수행한다.

HTTP에서 메시지로 주고받는 자원의 종류가 미디어 타입.

HTTP는 미디어 타입에 제한을 두지 않고 독립적으로 동작

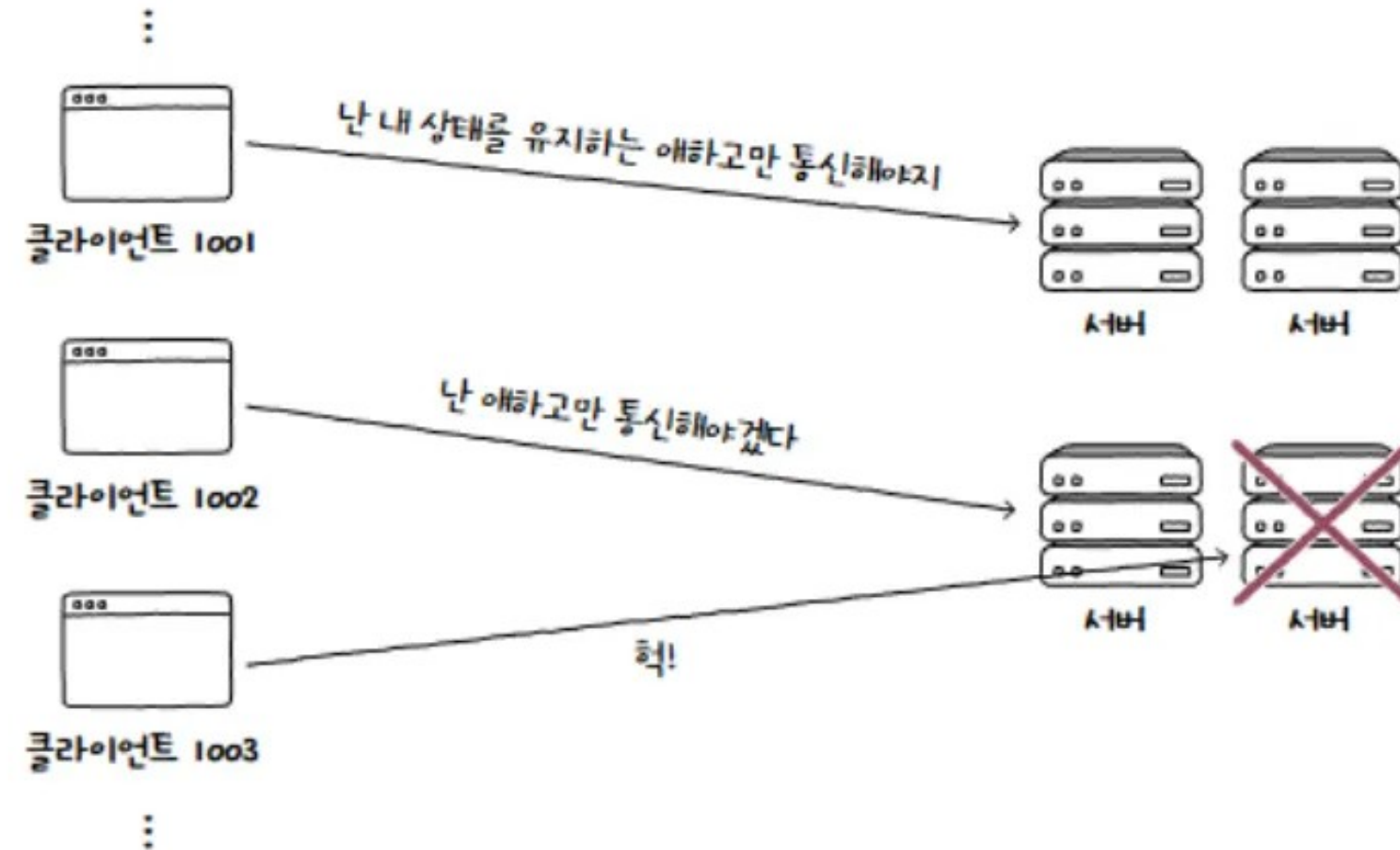
HTTP의 특성 - 스테이트리스 프로토콜



상태를 유지하지 않는다.

같은 요청을 여러 번 보내도 다른 요청으로 간주한다.

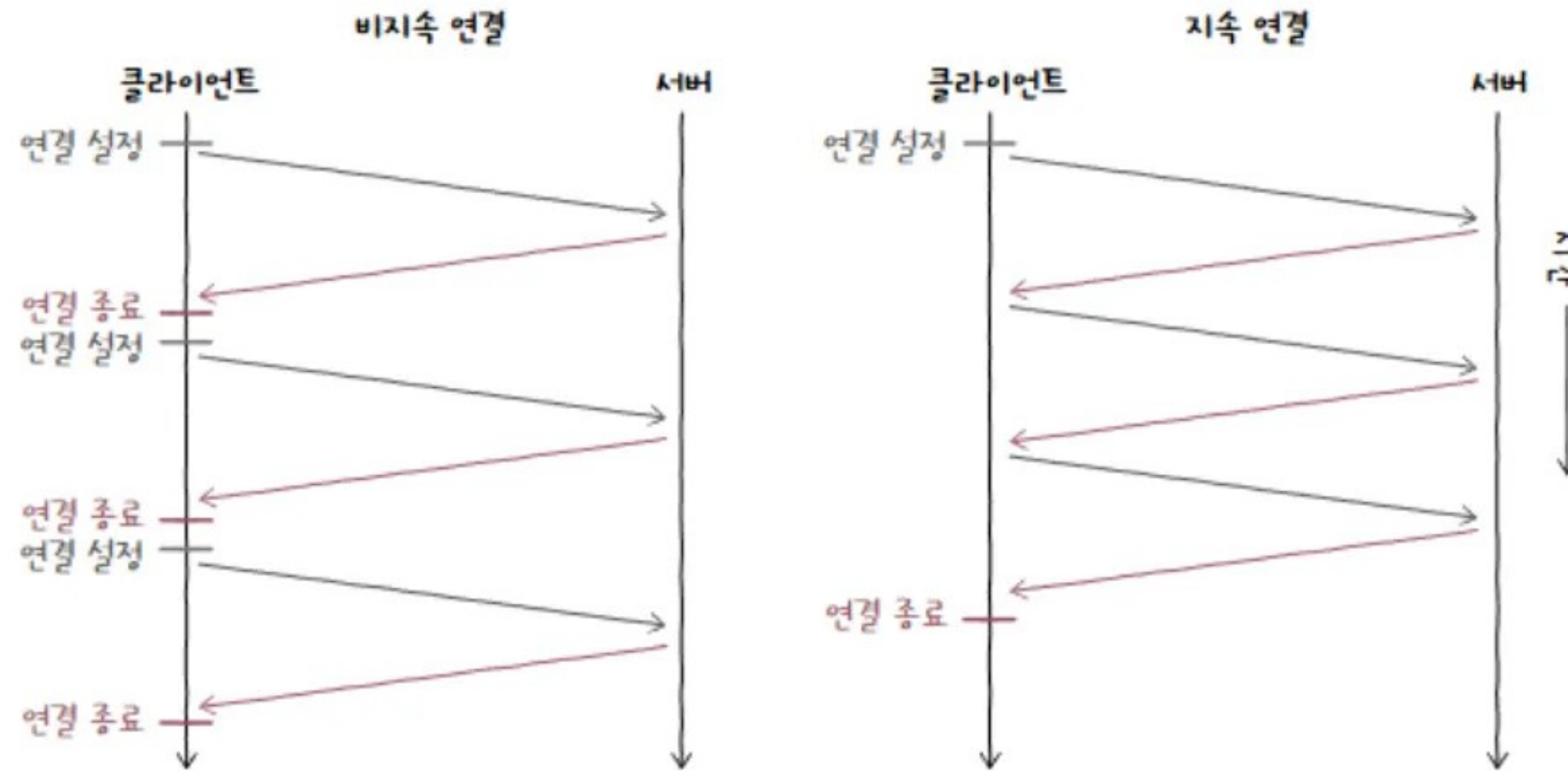
HTTP의 특성 - 스테이트리스 프로토콜



상태를 유지한다면?

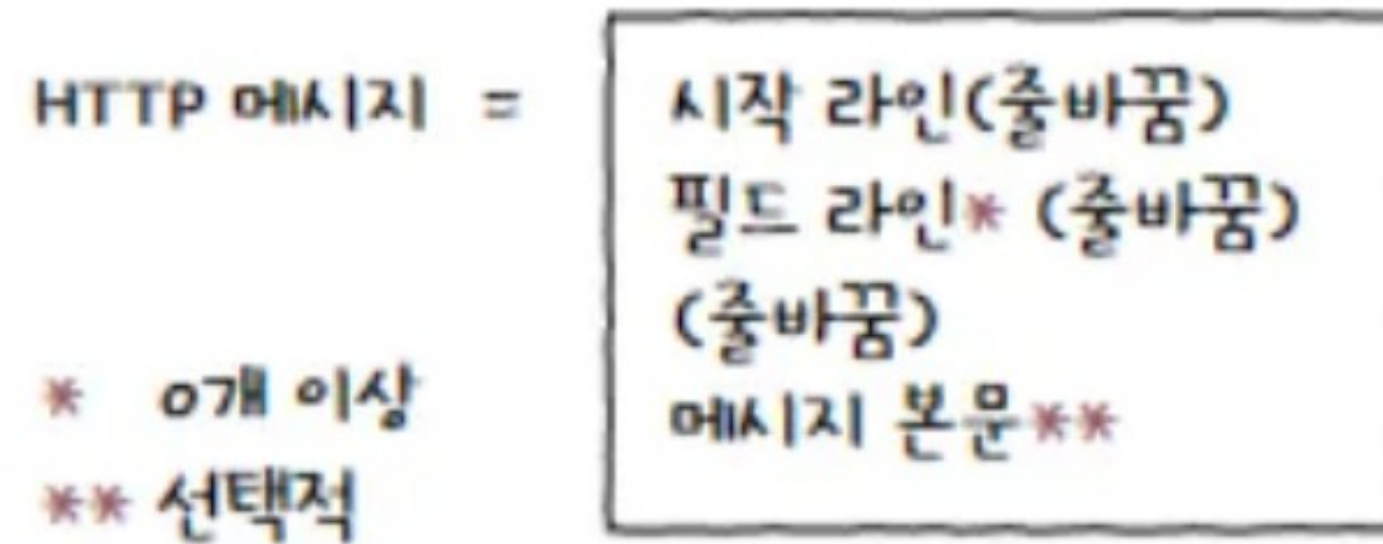
클라이언트는 자신의 상태를 기억하는 서버에 종속
해당 서버에 문제가 생기면 HTTP 통신 내역 삭제됨.

HTTP의 특성 - 지속 연결 프로토콜



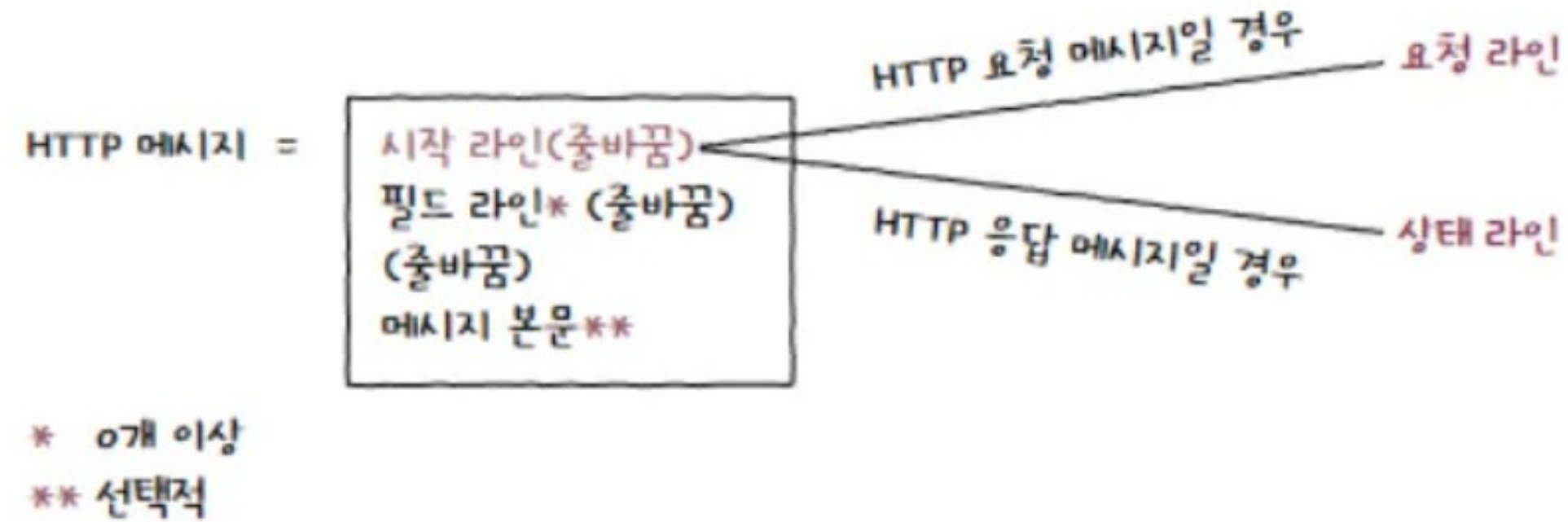
HTTP 1.1 이상에서 제공하는 지속 연결

HTTP 메시지 구조(HTTP 1.1 위주)



1. 호스트 A에서 프레임을 수신
2. 프레임 내 '송신지 MAC 주소' 정보로 MAC 주소 학습.
3. 아직 수신지 호스트 C가 어느 포트에 연결된지는 모른다.

HTTP 메시지 구조 - 시작 라인



HTTP 요청인 경우 시작 라인은 요청 라인
응답 메시지일 경우 시작 라인은 상태 라인

HTTP 메시지 구조 - 시작 라인(요청 라인)

요청 라인 = 메서드 (공백) 요청 대상 (공백) HTTP 버전 (줄바꿈)

메서드: GET, POST, DELETE, ...

요청 대상: URI 경로(/hello?q=world)

HTTP 버전: HTTP/1.1

HTTP 메시지 구조 - 시작 라인(응답 라인)

상태 라인 = HTTP 버전 (공백) 상태 코드 (공백) 이유 구문* (줄바꿈)

* 선택적

상태 코드와 이유 구문

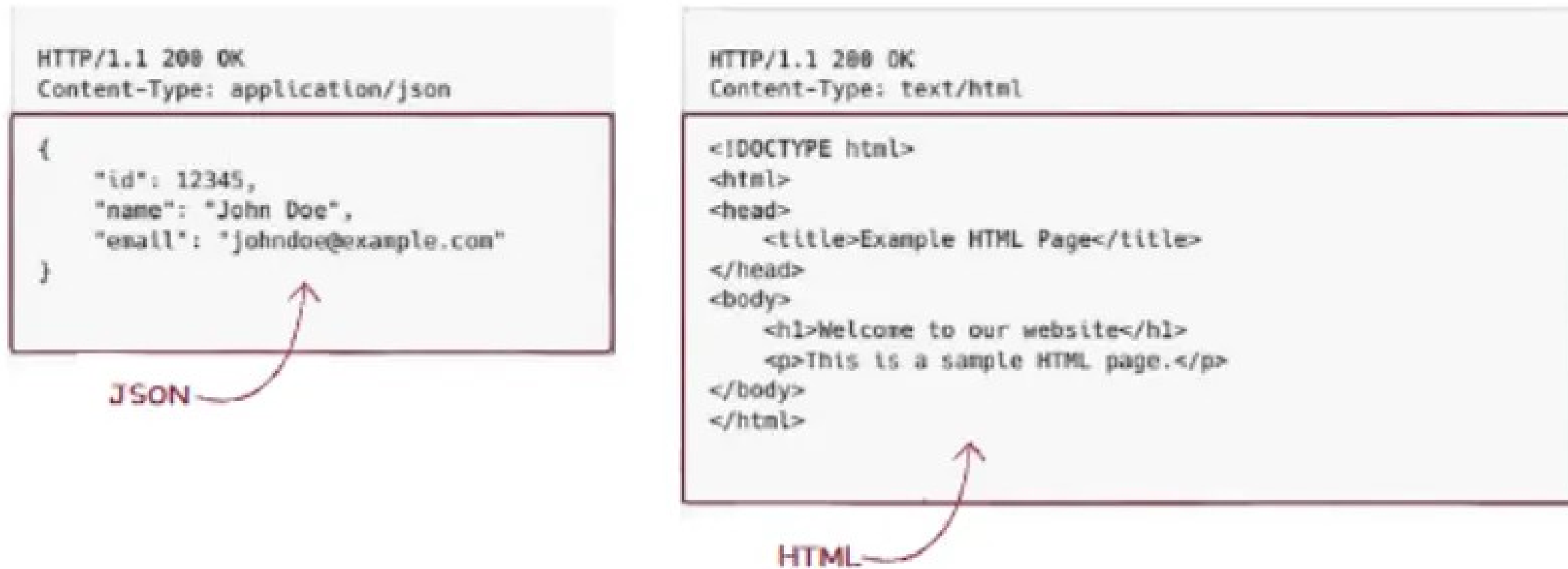
HTTP/1.1 200 OK : 200이 상태 코드, OK가 이유 구문

HTTP 메시지 구조 - 필드 라인

```
GET /example-page HTTP/1.1  
Host: www.example.com  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101  
Firefox/118.0  
Accept: text/html
```

헤더 이름과 하나 이상의 헤더 값으로 구성

HTTP 메시지 구조 - 메시지 본문



HTTP 요청, 응답 메시지에서 본문이 필요할 경우

HTTP 메서드부터 박두칠박두칠박두칠박두칠박두칠박두칠박두칠
박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두
칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박
두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠
박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두
칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박
두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠
박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두
칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠박두칠 형
이..