HTTP (Hypertext Transfer Protocol) 정리 및 상세 설명

HTTP는 클라이언트와 서버 간의 데이터 통신을 위해 사용하는 애플리케이션 계층 프로토콜입니다. 웹 브라우저(클라이언트)는 HTTP 요청을 서버에 보내고, 서버는 해당 요청에 대한 응답을 반환하는 방식으로 작동합니다. 이 과정에서 다양한 요청 메서드와 상태 코드를 사용하며, 무상태 프로토콜이라는 특성을 가집니다.

1. HTTP의 정의 및 특성

HTTP는 하이퍼텍스트 문서를 교환하기 위해 개발되었으며, 현재는 텍스트, 이미지, 비디오 등 다양한 형식의 데이터를 전송하는 데 사용됩니다. HTTP는 TCP/IP 프로토콜을 기반으로 작동하며, 클라이언트-서버 모델을 따릅니다.

- **클라이언트-서버 모델**: 클라이언트(주로 웹 브라우저)가 서버로 요청을 보내고, 서버는 요청에 대한 응답을 클라이언트에게 반환합니다.
- **애플리케이션 계층**: HTTP는 애플리케이션 계층에서 동작하며, 데이터 전송의 구조를 정의합니다.

2. HTTP의 무상태(stateless) 특성

HTTP는 무상태 프로토콜입니다. 각 요청은 독립적으로 처리되며, 서버는 이전 요청에 대한 정보를 유지하지 않습니다. 이는 서버 측에서 요청의 처리 효율성을 높이지만, 상태를 유지할 필요가 있을 때는 별도의 방법을 사용해야 합니다. 예를 들어, 사용자가 로그인한 상태를 유지하기 위해 세션, 쿠키, 또는 JWT(JSON Web Token)와 같은 기술을 활용합니다.

- **장점**: 서버는 상태 정보를 유지할 필요가 없기 때문에 확장성이 좋고, 서버 부하가 줄어듭니다.
- **단점**: 연속적인 사용자 경험을 위해 상태 정보를 클라이언트 측 또는 다른 방식으로 관리해야 합니다.

3. HTTP 요청 메서드

HTTP 메서드는 클라이언트가 서버에 어떤 작업을 요청하는지 정의하는 중요한 요소입니다.

메서드	설명	주된 사용 예시
GET	서버로부터 데이터를 요청. 쿼리 문자열로 데이터 전송.	리소스 조회, 검색
POST	데이터를 서버로 전송. 요청 본문(body)에 데이터를 포함하여 전송.	폼 제출, 데이터 전송
PUT	리소스를 생성하거나 수정. 존재하지 않으면 새로 생성하고, 있으면 수정.	데이터 전체 업데이트
PATCH	리소스의 일부만 수정.	부분 업데이트
DELETE	서버에서 리소스를 삭제.	리소스 삭제
HEAD	GET과 유사하지만 본문 없이 응답 헤더만 반환.	리소스 존재 확인
OPTIONS	서버가 지원하는 메서드 확인.	지원하는 메서드 목록 요청

이 외에도 **HEAD**(GET과 동일하지만 응답 본문은 포함하지 않음), **OPTIONS**(서버에서 허용하는 메서드 조회), **TRACE**(요청 메시지의 경로를 따라가며 진단을 수행) 등의 메서드가 존재합니다.

4. HTTP 메시지 구조

HTTP 통신은 요청(request)과 응답(response)으로 이루어집니다.

- **요청 메시지**: 클라이언트에서 서버로 보내는 메시지로, 메서드, URL, 버전, 헤더, 본문으로 구성됩니다.
  - **메서드**: 요청의 종류를 나타냅니다. (GET, POST 등)
  - **URL**: 요청하려는 리소스의 경로입니다.
  - **헤더**: 클라이언트와 서버 간의 추가 정보를 전달합니다.
  - **본문**: (선택적) POST, PUT 등의 메서드로 데이터를 전송할 때 포함됩니다.
- **응답 메시지**: 서버에서 클라이언트로 보내는 메시지로, 상태 코드, 헤더, 본문으로 구성됩니다.
  - **상태 코드**: 요청에 대한 서버의 처리 결과를 나타냅니다.
  - **헤더**: 응답에 대한 추가 정보를 전달합니다.
  - **본문**: 요청된 리소스나 처리 결과에 대한 데이터가 포함됩니다.

5. HTTP 상태 코드

상태 코드는 서버가 클라이언트의 요청을 어떻게 처리했는지를 나타내는 3자리 숫자입니다.

- **1xx: 정보 (Informational)**
  요청이 수신되어 처리 중임을 나타냅니다. (예: `100 Continue`)
- **2xx: 성공 (Success)**
  요청이 성공적으로 처리되었음을 의미합니다.
  - `200 OK`: 요청이 성공적으로 처리됨.
  - `201 Created`: 요청이 성공적으로 처리되었으며 새로운 리소스가 생성됨.
- **3xx: 리다이렉션 (Redirection)**
  클라이언트가 요청한 리소스를 다른 위치로 이동시키도록 안내합니다.
  - `301 Moved Permanently`: 요청한 리소스가 영구적으로 이동됨.
  - `302 Found`: 요청한 리소스가 일시적으로 다른 위치에 있음.
- **4xx: 클라이언트 오류 (Client Error)**
  클라이언트 측에서 문제가 발생했음을 나타냅니다.
  - `400 Bad Request`: 잘못된 요청 형식.
  - `401 Unauthorized`: 인증이 필요함.
  - `403 Forbidden`: 접근 권한 없음.
  - `404 Not Found`: 요청한 리소스를 찾을 수 없음.
- **5xx: 서버 오류 (Server Error)**
  서버 측에서 문제가 발생했음을 나타냅니다.
  - `500 Internal Server Error`: 서버에서 알 수 없는 오류 발생.
  - `502 Bad Gateway`: 게이트웨이 서버가 잘못된 응답을 받음.
  - `503 Service Unavailable`: 서버가 일시적으로 사용 불가.

6. HTTP/1.0, HTTP/1.1, HTTP/2 비교

- **HTTP/1.0**: 초기 버전으로, 각 요청마다 새로운 TCP 연결을 설정하는 방식입니다. 성능 상의 제약이 있어 이후 버전으로 대체되었습니다.
- **HTTP/1.1**: 연결 재사용(persistent connection)이 가능해져 성능이 개선되었습니다. 또한, 청크 전송 인코딩(chunked transfer encoding)과 같은 기능도 추가되어 대용량 데이터를 전송할 때 유용합니다.
- **HTTP/2**: 성능 향상을 위해 다중화(Multiplexing)를 도입하여, 하나의 연결로 여러 요청을 동시에 처리할 수 있습니다. 이외에도 헤더 압축, 서버 푸시 등의 기능이 추가되어 웹 성능을 더욱 개선했습니다.

7. HTTPS (HTTP Secure)

HTTPS는 HTTP에 보안 계층을 추가한 프로토콜입니다. SSL(Secure Sockets Layer) 또는 TLS(Transport Layer Security)를 사용하여 데이터가 암호화된 상태로 전송되므로, 중간에서 데이터를 가로채더라도 내용을 해석할 수 없습니다. 이를 통해 데이터의 기밀성과 무결성을 보장합니다.
주요 기능:

- **기밀성**: 데이터를 암호화하여 보호.
- **무결성**: 전송 중 데이터가 변조되지 않도록 보호.
- **인증**: 인증서를 통해 서버와 클라이언트 간의 신뢰성을 보장.

HTTPS는 특히 금융, 개인 정보 보호가 중요한 서비스에서 필수적이며, 오늘날 웹에서는 기본적으로 HTTPS를 사용하도록 권장됩니다.

이 정리에서는 HTTP의 작동 방식, 주요 개념, 그리고 각 버전의 차이점에 대해 다루었습니다.