1. **HTTP 통신**

HTTP는 HTML 문서와 같은 리소스를 가져올 수 있도록 하는 프로토콜입니다. 웹에서 이루어지는 모든 데이터 교환의 기초이며 수신자 측에 의해 요청이 초기화되는 클라이언트-서버 프로토콜입니다.

클라이언트와 서버는 메시지 교환에 의해 통신하며 보통 브라우저인 클라이언트에 의해 전송되는 메시지를 요청(requests)이며 서버에서 응답으로 전송되는 메시지를 응답(responses)라고 부른다. 연결상태를 유지하지 않는 비연결성 프로토콜이다.

HTTP request의 구조는 request line, header, body로 이루어져 있습니다.

- Request line

1) HTTP 메서드(GET, POST 등)을 사용해 서버가 수행해야 할 동작을 나타냄  
2) 요청 타겟(URL, 프로토콜, 포트, 도메인)을 나타내며 요청 컨텍스트에   
 의해 지어진다.  
3) HTTP 버전이 들어갑니다. 응답 메시지에 사용할 HTTP 버전을 알려주는  
 역할을 합니다

- Header

1) 요청에 들어가는 HTTP헤더는 HTTP 헤더 기본 구조를 따른다.

2) 대소문자 구분이 없는 문자열 다음 콜론이 붙으며 그 뒤 값은 헤더에 따라 달라진다.

- Body

1) request의 마지막 부분에 들어갑니다. 모든 요청에 들어가지 않고 Get, Head, DELETE, OPTIONS 처럼 리소스를 가지고 오는 요청은 본문을 필요로 하지 않는다.

HTTP Response의 구조는 Status Line, Header Lines, Body로 이루어져 있다.

-Status Line

1) 프로토콜 버전을 나타냅니다.

2) 상태코드, 요청 성공 여부를 나타냅니다. (200, 404 등 코드)

3) 상태 텍스트, 짧고 간결하게 상태 코드에 대한 설명을 글로 나타냅니다.

- Header

1) 응답에 들어가는 HTTP 헤더는 요청과 마찬가지로 HTTP 헤더의 기본 구조를 따릅니다.

2) 대소문자 구분 없는 문자열 다음 콜론이 붙으며 그뒤 값은 헤더에 따라 나눠진다.

- Body

1) response의 마지막 부분에 들어갑니다. 모든 응답에 들어가지 않고 코드를 가진 응답에는 본문이 없습니다.

2) response body 는 이미 길이가 알려진 단일 파일로 구성된 두 content-type, content-length 를 가진 단일-리소스 본문, 길이를 모르는 단일 파일로 구성된 단일-리소스 본문, 서로 다른 정보를 담고 있는 멀티파트로 이루어진 다중-리소스 본문으로 3가지로 나눠집니다.

1. **웹 브라우저에서 URL을 입력하면 서버에서 응답하는 과정**
2. 웹 브라우저에서 URL을 입력한다.
3. 웹 브라우저가 도메인의 IP주소를 조회하여 캐시를 찾고 DNS를 검색한다.
4. 웹 브라우저가 찾은 IP주소를 기반으로 서버와의 TCP 연결을 시작한다.
5. 웹 브라우저가 HTTP 요청을 서버로 전송한다.
6. 웹 서버가 요청을 처리하고 응답을 다시 웹 브라우저로 전송한다.
7. 웹 브라우저가 전송 받은 컨텐츠를 렌더링한다.
8. **REST API**

REST는 “Representational State Transfer”의 약자이며 자원 이름으로 구분하여 해당 자원의 상태를 주고 받는 모든 것을 의미합니다.

자원은 해당 소프트웨어가 관리하는 모든 것(문서, 그림, 데이터 등)으로 상태 정보를 데이터가 요청되어지는 시점에서 자원의 상태를 전달합니다.

**REST의 구성요소**

1. 자원: URI

- 모든 자원에 고유한 ID 가 존재하고 이 자원은 서버에 존재한다.

- 자원을 구별하는 ID 는 HTTP URI 이다.

- Client는 URI를 이용해 자원을 지정하고 해당 자원의 상태에 대한 조

직을 서버에 요청한다.

1. 행위 : HTTP 메서드

- HTTP 프로토콜의 메서드를 사용한다.

- HTTP 프로토콜은 GET, POST, PUT, DELETE 같은 메서드를 제공한다.

1. 표현(Representation of Resource)

* Client 가 자원의 상태에 대한 조작을 요청하면 서버는 이에 적절한 응답을 보낸다.
* REST에서 하나의 자원은 JSON, XML, TEXT, RSS 등 여러 형태의 응답으로 나타낼 수 있다.
* JSON 이나 XML을 통해 데이터를 주고받는 것이 일반적이다.

**REST API**

API는 어플리케이션이나 디바이스가 서로 연결하여 통신할 수 있는 방법을 정의하는 규칙 세트입니다. REST API 는 아키텍쳐 스타일의 디자인 원칙을 준수하는 API 입니다.

**REST API 설계 기본 규칙**

참고 리소스 원형

DOCUMENT : 객체 인스턴스나 데이터베이스 레코드와 유사한 개념

COLLECTION : 서버에서 관리하는 디렉터리라는 리소스

STORE: 클라이언트에서 관리하는 리소스 저장소

1. URI는 정보의 자원을 표현해야 한다.
2. 동사보단 명사, 대문자보단 소문자 사용.

도큐먼트 이름으론 단수명사

컬렉션 이름으론 복수명사

스토어 이름으론 복수명사를 사용해야 한다.

EX) GET /MEMBER/1 -> GET /members/1

1. 자원에 대한 행위는 HTTP Method로 표현한다.
2. URI에 HTTP 메서드가 들어가면 안된다.
3. URI에 행위에 대한 동사 표현이 들어가면 안된다.
4. 경로 부분 중 변하는 부분은 유일한 값으로 대체한다.

**REST API 설계 규칙**

1. 슬래시 구분자는 계층 관계를 나타내는데 사용한다.
2. URI 마지막 문자로 슬래시를 포함하지 않는다.
3. 하이픈은 URI 가독성을 높이는데 사용한다.
4. 밑줄은 URI에 사용하지 않는다.
5. URI 경로에는 소문자가 적합하다.
6. 파일 확장자는 URI에 포함하지 않는다.
7. 리소스 간에 연관관계가 있는 경우 다른 리소스명을 사용

**RESTful의 개념**

Restful이란 일반적으로 REST라는 아키텍처를 구현하는 웹 서비스를 나타내기 위해 사용되는 용어이다. RESTful의 목적은 이해하기 쉽고 사용하기 쉬운 REST API를 만들어 근본적인 목적이 성능 향상에 있는 것이 아니라 일관적인 컨벤션을 통한 API의 이해도 및 호환성을 높이는 것이 주 동기이다.