기초 웹 프로그래밍 과제 **[~9/23/월/18:00까지 업로드]**

HTTP 프로토콜에 관하여 다음의 내용을 조사하여 제출하세요.

(1) HTTP 프로토콜의 Request Method 종류와 설명

(2) HTTP 프로토콜의 Response의 Status의 종류와 설명

(3) HTTP/2의 특징은 무엇이며, HTTP/1.1과의 차이점은 무엇인가?

(4) HTTP/3의 특징은 무엇이며, 이전 버전들과의 차이점은 무엇인가?

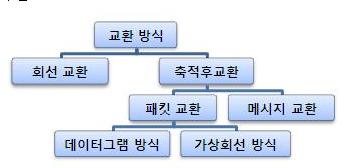
1. **HTTP (HyperText Transfer Protocol)**

먼저 HTTP에 대해 설명하자면 Hypertext Transfer Protocol의 준말로, 있는 그대로 해석하자면 ‘Hypertext를 전송하는 protocol’이라는 의미이다. 여기서 Hypertext(이하 하이퍼텍스트)란 1965년 Ted Nelson에 의해 정의된 개념으로 다른 텍스트들에 대해 링크를 포함하고 있는 텍스트를 의미한다. Protocol의 원래 의미는 외교 상에서의 의례 또는 의정서를 나타내는 말이지만 네트워크 상에서는 표준화된 통신규약으로서 네트워크의 기능을 효율적으로 발휘하기 위한 협정을 뜻한다. 한마디로 인터넷 상에서 지켜야 할 규칙을 protocol(이하 프로토콜)이라고 하는 것이다. 각 단어의 의미를 근거로 HTTP를 다시 설명하자면 하이퍼텍스트를 전송하기 위한 인터넷 상에서의 규칙인 것이다.

웹은 하이퍼텍스트에 접근하기 위한 시스템이지만 이것이 단순히 접근만을 의미한다고 보기 어렵다. 네트워크를 통해서 서로 주고받은 무수히 많은 하이퍼텍스트 형태의 정보들을 교환하여 의미 있는 활동을 하는 것이 중요한데 이것을 가능하게 하려면 일단 정보를 네트워크를 통해서 주고받을 수 있어야 한다.

1. 패킷(packet)

정보를 네트워크를 통해 전달하기 위해서는 그 정보 그대로 전달할 수 없기에 전달 가능한 단위로 잘라야 하는데 이렇게 데이터를 일정 크기로 자른 것을 packet(이하 패킷)이라고 하며 IP 데이터 그램 역시 같은 의미이다. 따라서 인터넷에서의 정보 송수신은 패킷을 교환하는 것과 같다고 볼 수 있다. 데이터를 교환하는 방식은 크게 회선 교환과 축적 후 교환으로 나눠지며 축적 후 교환은 또다시 패킷 교환과 메시지 교환으로, 여기서 패킷 교환은 데이터 그램 방식과 가상회선 방식으로 한 번 더 나눠진다.

 -> 연결 방식

2) 연결 방식

여기서 회선 교환이란 전화통신에 사용하던 전통적인 교환방식으로써 연결되는 전체 기간 동안 회선이 고정 할당되므로 회선 자체가 점유되는 방식을 뜻한다. 즉, 네트워크에서 양단간 연결 링크를 공유하는 것인데 이때 링크는 사용자에 의해 통화하는 동안 독점되며, 그것이 해제되어야 다른 사용자가 이용할 수 있다.

축적 교환 방식은 송수신 상호 간에 직접적인 접속경로를 만들지 않고 통신정보를 중간 노드 등의 기억매체를 일단 경유해서 중계루트를 선택하여 상대방에게 전송하는 교환방식이다. 축적 후 교환 방식은 입력되는 프레임을 출력측 어느 하나로 교환하는 방식 중에 하나로써, 들어오는 프레임 전부를 일단 버퍼에 담아두고 에러 검출과 같은 처리를 완전히 수행한 후에 비로소 전달되는 방식이다.

패킷 교환 방식은 기본 축적단위가 일련의 긴 메시지를 패킷 단위로 전달하는 것이고 메시지 교환 방식은 기본 축적단위가 메시지 전체를 하나로 전달하는 것이다. 데이터 그램 방식은 패킷 교환에서 각 패킷이 독립적으로 처리되어 목적지까지 도달하는 방식을 말한다. 여기서 데이터 그램이란 패킷 교환에서 각각 독립적으로 취급되는 각각의 패킷을 말한다. 데이터 그램 방식의 특징은 회선 교환과 달리, 사전에 연결 설정이 필요 없다는 것이다. 또한 각 패킷들을 독립적으로 처리하게 되며 순서와 무관하게 전달된다. 따라서 목적지가 같은 패킷이라도 항상 같은 경로를 따르지 않는다. 또한 통상적으로 양단간에 요청-응답의 형태를 갖는다.

데이터 그램 방식은 비 연결성 통신방식이기도 한데 어떠한 사전 호출이나 연결 설정행위가 없어도 두 호스트 시스템 사이에서 트래픽을 교환할 수 있는 방식을 의미하며 단순하게 목적지를 향해서 목적지 주소를 담은 패킷을 전송한다. 전송만을 하기에 패킷이 도중에 손실되는 경우도 있는데 이때 수신 측에서 문제를 검출하여 송신 측에 재전송을 요구한다. 또한 송수신 간에 논리적인 연결없이 불연속적으로 발생되는 데이터마다 다른 경로를 따르는 방식이기도 하다.

이와 반대되는 것이 연결지향성 통신방식이다. 두 개체 간 1개 이상의 메시지들이 연결 상태를 유지하며 데이터 교환이 가능한 서비스를 뜻한다. 데이터가 지속적이고 연속적인 흐름에 적합하다. 연결지향성 통신방식은 데이터 교환이 시작되기 전에 송수신측 간 논리적인 연결이 설정되어 있어야 한다. 이것은 마치 사용자 입장에서는 상대방과 연결이 끊길 때까지 사용하는 둘만의 전용선이 존재하는 것처럼 보인다. 이러한 방식을 가상 회선 방식이라 한다.

1. Routing(라우팅)

이처럼 패킷을 ‘전송’만하는 역할을 하는 네트워크 측에서는 패킷 내 목적지 주소로 패킷을 경로배정, 경로제어 혹은 routing(이하 라우팅)만 하면 그 임무를 다하는 셈이다. 라우팅이란 네트워크 상에서 주소를 이용하여 목적지까지 경로를 체계적으로 결정하는 경로선택 과정을 의미하며 네트워크를 효율적으로 이용하여 서로 연결된 노드 간에 경로 찾기를 뜻한다. 라우팅 방식의 주요 구분은 고정 라우팅과 정적 라우팅으로 나눈다면 미리 정해진 루트를 따라 경로를 선택하는 것을 기준으로 한 것이고, 동적 라우팅과 적응적 라우팅으로 나눈다면 네트워크의 상태에 따라 경로를 선택하는 자동 경로 설정을 기준으로 한 것이다.

<http://www.ktword.co.kr/word/abbr_view.php?nav=&m_temp1=422&id=484> (패킷)

1. TCP/IP

네트워크를 이용한 웹에서의 데이터 송수신은 비연결성 통신방식으로 이뤄지는데 단순하게 패킷에 담긴 목적지 주소를 향해서 전달되므로 순서에 대한 신뢰성과 수신자 측에서 정보를 받을 수 있다는 신뢰성에 문제가 생길 수 있다. 이러한 문제점을 보완하기 위해서 나온 것이 TCP/IP 개념이다.

여기서 TCP와 IP에 대해 간단하게 말하자면 다음과 같다. TCP(Transmission Control Protocol)은 IP(Internet Protocol)의 핵심 프로토콜 중 하나로, IP와 함께 주로 TCP/IP라고 쓰인다. TCP는 IP 위에서 동작하며 peer간의 데이터 전달을 안정적으로 순서대로, 에러가 없도록 보장해주는 역할을 한다. IP(Internet Protocol)은 송신 및 수신 간의 패킷 교환 프로토콜이며 비신뢰성(unreliability)과 비연결성(connectionless)이 특징이다. 비신뢰성은 흐름에 관여하지 않기 때문에 보낸 정보가 전송 과정에서 손실될 수도 있고 순서가 바뀔 수도 있고 같은 패킷이 여러 번 전송될 수도 있다. 따라서 패킷 전송의 신뢰성과 순서 보장에 대한 신뢰성을 위해서 TCP 프로토콜과 같은 IP의 상위 프로토콜을 IP와 함께 이용해야 한다. 현재 IPv4와 IPv6을 함께 사용하고 있다.

[https://ko.wikipedia.org/wiki/인터넷\_프로토콜](https://ko.wikipedia.org/wiki/%EC%9D%B8%ED%84%B0%EB%84%B7_%ED%94%84%EB%A1%9C%ED%86%A0%EC%BD%9C) ip

<https://ko.wikipedia.org/wiki/%EC%A0%84%EC%86%A1_%EC%A0%9C%EC%96%B4_%ED%94%84%EB%A1%9C%ED%86%A0%EC%BD%9C> tcp

1. **HTTP Request Method**

먼저 Internet에 대한 개념을 짚어 볼 필요가 있다. Internet(이하 인터넷)은 International Network의 준말로, 전 세계의 무수히 많은 컴퓨터 네트워크를 하나로 묶어서 이루어진 네트워크를 뜻하며 표준 인터넷 프로토콜을 이용한다. 이러한 인터넷 서비스의 종류에는 검색엔진, E-mail, FTP(File Transfer Protocol), WWW(World Wide Web) 등이 있다.

WWW(이하 웹)는 인터넷을 통해서 웹페이지에 접속할 수 있게 해주는 서비스이며 흔히 W3, 혹은 Web이라고 불린다. Tim Berners-Lee가 웹의 구조를 최초로 제안했으며 이후 웹 서버, 웹 브라우저, 웹 페이지에 대한 개념 역시 최초로 제안하면서 오늘날까지도 인터넷의 아버지라 불리게 되었다. 웹은 몇 가지 구성요소를 가지는데 HTTP, HTML, URL, Web Server, Web Browser 가 그것이다.

1. 구성요소-HTTP

<https://developer.mozilla.org/en-US/docs/Web/HTTP>

HTTP(HyperText Transfer Protocol)는 [application layer](https://en.wikipedia.org/wiki/Application_layer)(공유된 커뮤니케이션 프로토콜과 커뮤니케이션 네트워크에서 호스트들에 의해 사용되는 인터페이스 메소드를 분명하게 해주는 추상적 레이어)로 하이퍼미디어 문서들을 전송하기 위한 것이다. 다른 목적으로도 사용될 수 있으나 기본적으로는 웹 브라우저와 웹 서버들 사이에서의 커뮤니케이션을 위해 고안되었으며, client-server 모델을 따른다. 이 모델은 client가 request를 만들어서 서버 측에 보내면 다시 서버로부터 response가 올 때까지 기다리는 것이다. HTTP는 stateless protocol(접속이 유지되지 않는 프로토콜)이라서 단순히 정보를 전달하는 역할만 한다. 이때 정보 교환에 대한 신뢰성과 순서에 대한 신뢰성을 위해 TCP/IP 위에서 HTTP가 작동하게 된다.

1. 구성요소-HTML

<https://developer.mozilla.org/en-US/docs/Web/HTML>

HTML(HyperText Markup Language)은 가장 기본적인 웹 구성 블록으로 웹 콘텐츠의 의미와 구조를 정의한다. HTML은 웹 브라우저에 텍스트, 이미지, 콘텐츠 등등을 띄우기 위해서(client에게 보여주기 위해서) markup을 사용하는데 이는 tag<>라는 기호를 사용하여 나타낸다. 예를 들면 <title>, <head>,<body>,<header>,<footer> 등등이 있다.

1. 구성요소-Web server

<https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server>

Web Server(이하 웹 서버)는 하드웨어이거나 소프트웨어이거나 혹은 둘 다 의미하는 말이다. 하드웨어 측면에서 웹서버는 컴퓨터로 웹 서버 소프트웨어와 웹사이트들의 구성요소 파일들을 보관하고 있는 것을 가리킨다. 또한 인터넷과 연결되어 있어서 다른 장치들과 웹에서의 연결을 통해 물리적 데이터 교환을 한다. 소프트웨어 측면에서는 웹 사용자가 요청한 HTTP 메시지를 처리하는 것을 의미한다.

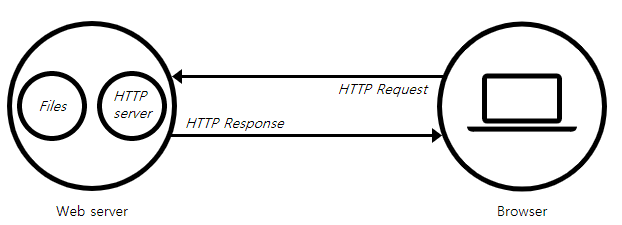
1. 구성요소-Web Browser

Web Browser(이하 웹 브라우저)는 웹 서버에서 이동하며 쌍방향으로 통신하고 HTML문서나 파일을 출력하는 그래픽 사용자 인터페이스 기반의 응용 소프트웨어이다. HTTP요청을 하거나 HTML Rendering 등등의 기능을 하며 예시로는 구글 크롬, 모질라 파이어폭스, 마이크로소프트 엣지 등등이 있다.

<HTTP 메시지 >

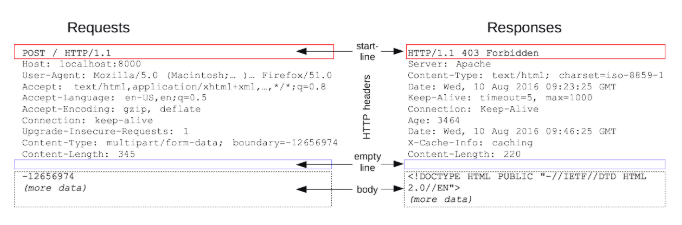
앞서 말한 바와 같이 HTTP는 client-server 모델을 따른다. client측에서 server로 request

를 보내면 server 측에서는 그에 해당되는 response를 보내준다.



* Client-server model

이때 HTTP 메시지는 HTTP Request, HTTP Response 메시지가 있는데 각각에 대해 살펴보면 다음과 같다. HTTP 메시지는 크게 start line, HTTP headers, empty body, body으로 나눠진다. HTTP Request 메시지는 웹서버에 메소드를 통한 어떤 동작을 요구하는 것이고 HTTP Response 메시지는 request의 결과를 client에게 돌려주는 역할을 한다.



1. HTTP Request

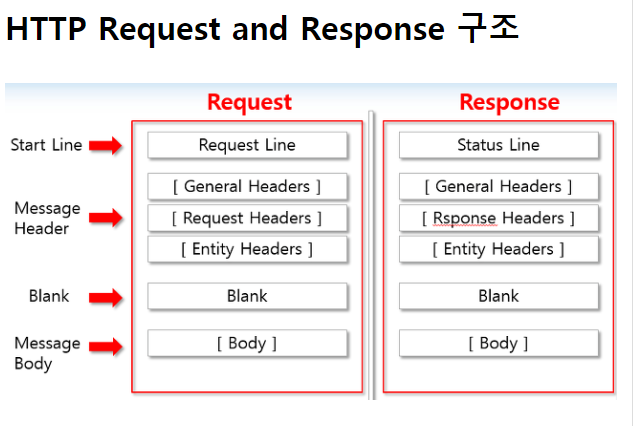
client측에서 웹 서버에 전송하는 메시지로 서버에 있는 파일을 요청할 때 사용된다.

HTTP Request는 크게 4가지 구조로 나눠지는데 request line, headers, empty, body가 그것이다. 먼저 request line을 보자. Method/URL/버전으로 구성되어 있다. 여기에서 쓰이는 method에 대해 정리하면 다음과 같다. //request method

<https://hyeonnii.tistory.com/80>

<https://feel5ny.github.io/2019/08/15/HTTP_003_01/>

<https://m.blog.naver.com/PostView.nhn?blogId=s2kiess&logNo=30182409370&proxyReferer=https%3A%2F%2Fwww.google.com%2F>



<각 구조와 메소드 쓰기,,사실상 여기서부터가 과제 본문인셈,,,ㅎ>

1. HTTP 메시지 구조 설명

크게

1. HTTP request 메시지 구조 설명 + Method
2. HTTP response status code 설명