HTTP Method(Get과 Post방식) Network Hacking 2016. 2. 5. 14:05

https://blog.naver.com/2000sky2/220619302087

+HTTP Method(GET&POST)+

정의 : HTTP 프로토콜은 TCP와 UDP 프로토콜을 기반으로 하여 웹에서 사용하는 프로토콜로서 클라이언트와 서버 사이에 이루어지는 요청과 응답 데이터를 전송하는 방식을 말한다.
즉, HTTP method는 서버에 요청을 보내는 방법이라 볼 수 있다.

HTTP 메소드 종류

HTTP Method	전송 형태	설명
GET	GET [request-uri]?query_string HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n	GET 요청 방식은 URI(URL)가 가진 정보를 검색하기 위해 서버 측에 요청하는 형태이다.
HTTP Method	전송 형태	설명
POST	POST [request-uri]? query_string HTTP/1.1₩r₩n HOST:[Hostname] 혹은 [IP] ₩r₩n Content-Lenght:[Lenght in Bytes] ₩r₩n ₩r₩n [query-string] 혹은 [데이터]	POST 요청 방식은 요청 URI(URL)에 폼 입력을 처리하기 위해 구성한 서버 측 스크립트(ASP, PHP, JSP 등) 혹은 CGI 프로그램으로 구성되고 Form Action과 함께 전송되는데 이때, 헤더 정보에 포함되지 않고 데이터 부분에 요청 정보가 들어가게 된다.
HTTP Method	전송 형태	설명
HEAD	HEAD [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n	HEAD 요청 방식은 GET과 유사한 방식이나 웹 서버에서 헤더 정보 이외에는 어떤 데이터도 보내지 않는다. 웹 서버의 다운 여부 점검(Health Check)이나 웹 서버 정보(버전 등) 등을 얻기 위해 사용될 수 있다.
HTTP Method	전송 형태	설명
OPTIONS	OPTIONS [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n	해당 메소드를 통해 시스템에서 지원되는 메소드 종류를 확인할 수 있다.
HTTP Method	전송 형태	설명
PUT	PUT [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n Content-Length:[Length in Bytes] ₩r₩n Content-Type:[Content Type] ₩r₩n ₩r₩n [데이터]	POST와 유사한 전송 구조를 가지기 때문에 헤더 이외에 메시지(데이터)가 함께 전송된다. 원격지 서버에 지정한 콘텐츠를 저장하기 위해 사용되며 홈페이지 변조에 많이 악용되고 있다.
HTTP Method	전송 형태	설명
DELETE	DELETE [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n ₩r₩n	원격지 웹 서버에 파일을 삭제하기 위해 사용되며 PUT과는 반대 개념의 메소드이다.
HTTP Method	전송 형태	설명

TRACE	TRACE [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n ₩r₩n	원격지 서버에 Loopback(루프백) 메시지를 호출하기 위해 사용된다.
HTTP Method	전송 형태	설명
CONNECT	CONNECT [request-uri] HTTP/1.1₩r₩n Host:[Hostname] 혹은 [IP] ₩r₩n ₩r₩n	웹 서버에 Proxy 기능을 요청할 때 사용된다.

-메소드를 보냈을 때 돌아오는 응답 코드에 대해 알아 보겠다.

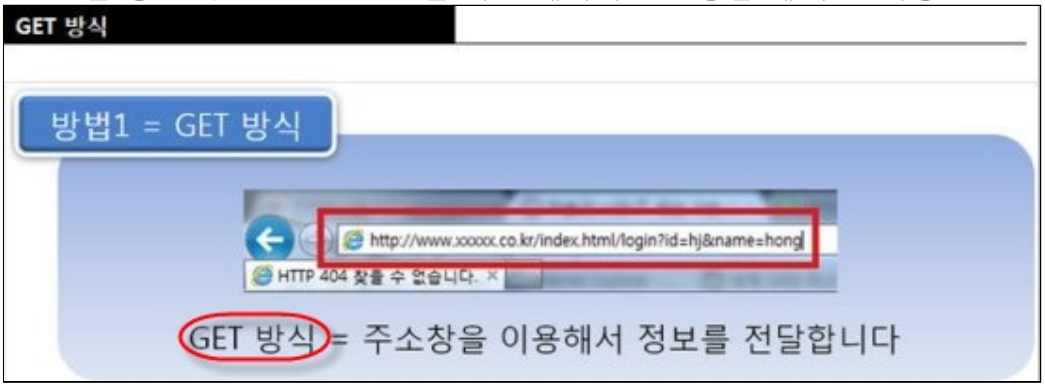
HTTP 응답 코드(상태코드) 종류

응답 코드	설명
100	Continue (클라이언트로 부터 일부 요청을 받았으며 나머지 정보를 계속 요청함)
101	Switching protocols(클라이언트는 대체 프로토콜의 사용을 요청하기 위해 Upgrade 헤더를 사용했고 서버는 이를 받아들임)
200	OK(요청이 성공적으로 수행되었음)
201	Created (PUT 메소드에 의해 원격지 서버에 파일 생성됨)
202	Accepted(웹 서버가 명령 수신함)
203	Non-authoritative information (서버가 클라이언트 요구 중 일부만 전송)
204	No content (사용자 요구 처리하였으나 전송할 데이터가 없음)
301	Moved permanently (요구한 데이터를 변경된 타 URL에 요청함)
302	Found (요청한 자원이 일시적으로 다른 URL 사용)
304	Not modified (컴퓨터 로컬의 캐시 정보를 이용함, 대개 gif 등은 웹 서버에 요청하지 않음)
400	Bad request (사용자의 잘못된 요청을 처리할 수 없음)
401	Unauthorized (인증이 필요한 페이지를 요청한 경우)
402	Payment required(예약됨)
403	Forbidden (접근 금지, 디렉토리 리스팅 요청 및 관리자 페이지 접근 등을 차단)
404	Not found (요청한 페이지 없음)
405	Method not allowed (허용되지 않는 http method 사용함)
407	Proxy authentication required (Proxy 인증 요구됨)
408	Request timeout (요청 시간 초과)
410	Gone (영구적으로 사용 금지)
412	Precondition failed (전체 조건 실패)
414	Request-URI too long (요청 URL 길이가 긴 경우임)
500	Internal server error (내부 서버 오류)
501	Not implemented (웹 서버가 처리할 수 없음)
503	Service unavailable (서비스 제공 불가)
504	Gateway timeout (게이트웨이 시간 초과)
505	HTTP version not supported (해당 http 버전 지원되지 않음)

+GET방식과 POST방식의 차이+

*데이터 전송 방식에는 크게 GET 방식과 Post 방식이 있다.
*페이지에서 페이지로 정보를 전송할 때 쓰이는 방법이다.

<GET 방식>
: URL을 통한 QUERY STRING을 다른 페이지로 전송할 때 주로 사용



- 모든 파라미터는 URL을 통해 전달된다.

- 구분자 ?뒤에 오는 값이 parameter값.(파라미터가 여러 개 있다면 &로 구분)

- 사용자의 눈에 직접적으로 표시되기 때문에, 로그인 비밀번호 등의 정보를 GET방식으로 전달 하면 문제가 발생 할 가능성이 있다. (최소의 보안유지도 없음)

- URL길이가 정해져 있기 때문에 많은 양의 정보를 전달 할 수 없다. URL 형식에 맞지 않는 파라미터 이름, 값은 인코딩되어서 전달해야 한다. (초과 데이터는 절단)

<POST 방식>
: HTML의 FORM 구성 요소(혹은 메시지 본문)를 이용해 데이터 전송할 때 주로 사용

- 전달하고자 하는 정보가 HTTP BODY(메시지 본문)에 포함되어 전달된다.

- HTTP BODY에 포함되므로, 웹 브라우저 사용자의 눈에 직접적으로 표시 되지 않는다.

- Post 방식은 Get방식에 비해 상대적으로 처리 속도가 늦다.

- 클라이언트측에서 데이터를 인코딩 → 서버측에서 디코딩

- 메세지 본문에 포함되어 전달 되므로 메시지 길이의 제한이 없지만 TIMEOUT이 존재.

- TIMEOUT이란 최대 요청 받는 시간을 의미하며 Client 에선 페이지를 요청하고 기다리는 시간을 나타낸다.

- 빨간 네모상자 안의 보면 구성요소: 32/60이란 부분이 보인다.

- 구성요소 부분이 30초(default TIMEOUT)이상 아무런 응답이 없을 경우엔 timeout이 일어나게 되며 에러페이지를 보여준다.

- 즉, TIMEOUT은 요청시간이 만료했을 경우에 발생하는 오류. (서버와 클라이언트 둘 다 존재 한다.)

따라서 POST방식은 GET방식과 달리 구조를 내부에 숨기기 때문에 민감한 데이터를 POST을 통해 보내는게 훨씬 안전하다.

<GET과 POST의 사용 시점>

+언제 GET을 쓰고 언제 POST를 써야 될까?

- 웹 표준에서도 그러 하듯이 현업의 개발에서는 "원래의 목적에 맞게 기술을 사용하고 있는가?"에는 대해서는 크게 관심이 없고 "어떤 기술이든 기능을 구현할 수 있는가?"에만 초점을 맞추는 것이 전반적으로 깔려있기 때문에 이런 부분에 대해서는 관심을 가지는 개발자는 그렇게 많지 않다.

- 일반적으로 id를 넘겨서 게시판의 리스트를 가져온다고 당연히 GET을 쓸 것이고 글을 작성한다고 하면 POST를 작성하는 것이 일반적이다. 전달되고자 하는 정보가 많을 때는 고민 없이 POST를 쓰게 되지만 전달되는 정보가 많지 않은 경우에는 GET을 써도 되고 POST도 써도 된다. (상당히 명백한 차이 같아 보이지만 실제로 개발을 하다 보면 고민되는 경우가 생긴다.)

*GET은 가져오는 것이고 POST는 수행하는 것.

1. GET은 Select적인 성향을 가지고 있다.

→GET은 서버에서 어떤 데이터를 가져와서 보여준다거나 하는 용도이지 서버의 값이나 상태 등을 바꾸지 않는다.
Ex)게시판의 리스트, 글을 보는 기능 같은 것들이 이에 해당
(방문자의 로그를 남긴다거나 글 읽는 횟수 올려준다거나 하는 것은 예외!)

2. POST는 서버의 값이나 상태를 바꾸기 위해서 사용

→글쓰기를 하면 글의 내용이 DB에 저장이 되고 수정을 하면 DB값이 수정된다.
이러한 경우에 POST를 사용한다.

+결론+

HTTP Method 중에서 데이터를 전달하는 방식에는 GET과 POST가 있다.
둘의 차이점은 이렇다.

GET&POST