

Form과 Method, 헤더

허강준

충남대학교 정보보호동아리 ARGOS

2021. 04. 10



Review: HTTP 패킷의 구조

■ 패킷 = 헤더 + 본문(Body)

```
GET https://www.argos.or.kr/ HTTP/1.1
Host: www.argos.or.kr
Connection: keep-alive
Cache-Control: max-age=0
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) App
Accept: text/html,application/xhtml+xml,application/xml;q
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: ko.en:g=0.9.en-US:g=0.8
Cookie: PHPSESSID=
                                             ; mobile=fals
```

■ 요청 패킷 (GET)

```
HTTP/1.1 200 OK
Server: nginx/1.4.6 (Ubuntu)
Date: Tue, 23 Mar 2021 05:05:38 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.5.9-1ubuntu4.29
Expires: Mon. 26 Jul 1997 05:00:00 GMT
Last-Modified: Tue, 23 Mar 2021 05:05:38 GMT
Cache-Control: no-store, no-cache, must-revalidate
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 43873
<!DOCTYPE html>
<html lang="ko">
<head>
<!-- META -->
<meta charset="utf-8">
<meta name="Generator" content="XpressEngine">
```

<meta http-equiv="X-UA-Compatible" content="IE=edge">

■ 응답 패킷

Form & HTTP Method

HTML Form 태그

Form & HTTP Method

- Form: 서식
- 회원가입, 로그인, 게시글 작성, 검색창, 등등...
- 사용자와 서버를 연결해주는 UI(User Interface)를 구성함
- 입력창, 체크박스, 버튼 등등...







HTML Form 태그

Form & HTTP Method

```
<form action="target address" method="method">
 <!-- 여기에 input 등의 태그를 집어넣기 -->
</form>
```

- action: 폼 데이터가 전송될 위치(주소)
- method: 폼 데이터를 전송하는 방법(Method) GET, POST, PUT, DELETE 등.. (뒤에 자세히 설명)

HTML Form Input 태그

```
<form action="target address" method="method">
 <!-- 여기에 input 등의 태그를 집어넣기 -->
</form>
```

- type: Input 태그의 종류
- name: 서버로 전송될 값의 "이름"
- value: 서버로 전송될 값

HTML Form Input: text & password

input type=text placeholder로 설명을 추가 안녕하세요! input type=password password도 설명 추가 가능

- 키보드로 값을 입력받는 폼 요소
- type=password 사용시 입력 내용을 숨길 수 있음
- placeholder 속성 사용시 비어있는 칸에 설명 추가 가능

HTML Form Input: checkbox & radio

input type=checkbox

☑ 피자 ☑ 치킨 □ 햄버거

input type=radio

◉ 남자 ○ 여자

- 여러개 선택 vs 한개만 선택
- radio의 경우 같은 name 끼리는 하나만 선택 가능
- checkbox와 같이 여러개 선택하는 경우 name에 [] 추가하여 배열로 넘기도록 해야함
- 체크된 항목의 value만 서버로 전송됨 (중요)



HTML Form Input: select

select

Form & HTTP Method



- 여러 항목중에서 하나 선택
- <input> 이 아닌 <select> 태그를 사용
- 선택 사항은 <select>, </select> 사이에 <option> 태그로 정의
- 미리 선택되게 하고 싶으면 해당 <option> 태그에 selected 설정

HTML Form Input: select

```
<select name="favorite">
 <option value="">선택해주세요!</option>
 <option value="cse" selected>컴퓨터융합학부</option>
 <option value="chem">화학공학과</option>
 <option value="phys">물리학과</option>
 <option value="krlit">국어국문학과</option>
</select>
```

HTML Form Input: textarea

textarea

Form & HTTP Method 000000000

> 여러중의 값을 입력하는 textarea EH□

- text 나 password와 비슷하지만 여러줄 입력 가능
- <textarea> 태그 사용
- 값은 <textarea>와 </textarea> 사이에 입력

HTML Form Input: submit

input type=submit



- text 나 password와 비슷하지만 여러줄 입력 가능
- <textarea> 태그 사용
- 값은 <textarea>와 </textarea> 사이에 입력

Method? GET? POST? ???

- Method: 특정 자원(리소스) 에 대한 요청을 처리하는 방법 (Method)
- GET: 자원에 대한 "표현" 요청
- POST: 특정 자원에 대한 "전송" 요청
- PUT: 특정 자원에 대한 "변경" 요청
- DELETE: 특정 자원에 대한 "제거" 요청

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



HTTP GET

- 자원에 대한 "표현" 요청
- 주소에 데이터를 삽입함 http://example.com/index.php?n=example
- 여러 데이터의 경우 & 를 이용하여 연결 http://example.com/index.php?n=example&n2=example2
- 주로 검색창이나 글 읽기 등에 사용
- 구현이 간편하나 데이터가 주소에 그대로 표현되므로 보안이 중요하면 사용 금지



HTTP GET in PHP

- \$_GET 은 GET으로 전달받은 데이터를 보관하는 "배열"
- \$_GET["name"] 식으로 전달된 데이터를 사용할 수 있음

HTTP GET in PHP



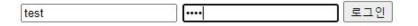
HTTP POST

- \$_POST 는 GET으로 전달받은 데이터를 보관하는 "배열"
- 마찬가지로 \$_POST["name"] 식으로 전달된 데이터를 사용할 수 있음

HTTP POST in PHP

Form & HTTP Method

로그인(POST)



localhost:60000/sample_form_proc_post.php

array(2) { ["id"]=> string(4) "test" ["password"]=> string(4) "1234" }

Request: GET vs POST

Form & HTTP Method

```
GET http://localhost:60000/sample_form_proc_get.php?id=test&password=1234 HTTP/1.1
Host: localhost:60000
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: User-Agent: Moc: 110.0; Win64; x64) Applewebkit/537.36 (KHTML, lik
Accept: text/html,application/xhtml+xml,application/xml;q=0.9;image/webp,image/apng,
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: 71
Sec-Fetch-Dest: document
Referer: http://localhost:60000/sample_form.ohp
Accept=Encoding: gzip, deflate, br
Accept-Language: ko,en;q=0.9,en-US;q=0.8
```

```
POST http://localhost:60000/sample_form_proc_post.php HTTP/1.1
Host: localhost:60000
Connection: keep-alive
Content-Length: 21
Cache-Control: max-age=0
Origin: http://localhost:60000
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,i
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:60000/sample_form.php
Accept-Encoding: gzip, deflate, br
Accept-Language: ko,en;q=0.9,en-US;q=0.8
```

id=test&password=1234

Header in Packet

```
POST http://localhost:60000/sample_form_proc_post.php HTTP/1.1
Host: localhost:60000
Connection: keep-alive
Content-Length: 21
Cache-Control: max-age=0
Origin: http://localhost:60000
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
Accept: text/html.application/xhtml+xml.application/xml:g=0.9.image/webp.i
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: 21
Sec-Fetch-Dest: document
Referer: http://localhost:60000/sample form.php
Accept-Encoding: gzip, deflate, br
Accept-Language: ko.en; g=0.9.en-US; g=0.8
id=test&password=1234
```

Header in Packet

```
HTTP/1.1 200 OK
Server: nginx/1.4.6 (Ubuntu)
Date: Tue, 23 Mar 2021 05:05:38 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.5.9-1ubuntu4.29
Expires: Mon. 26 Jul 1997 05:00:00 GMT
Last-Modified: Tue, 23 Mar 2021 05:05:38 GMT
Cache-Control: no-store, no-cache, must-revalidate
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 43873
<!DOCTYPE html>
<html lang="ko">
<head>
<!-- META -->
<meta charset="utf-8">
<meta name="Generator" content="XpressEngine">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

Header 다시보기

- HTTP 패킷은 헤더 + 본문(바디)로 이루어짐
- 통신을 요청하는 클라이언트나 서버, 혹은 전송되는 데이터에 대한 메타데이터
- 헤더와 바디는 헤더 이후의 빈 줄 하나로 구분됨
- 바디는 있을수도 없을 수도 있음

Request: User-Agent

- 현재 접속중인 브라우저의 종류를 표현하는 헤더
- IE인지, 크롬인지, 사파리인지, 파이어폭스인지, PC인지 모바일인지에 따라 그 내용이 다름
- 구현에 따라 전송하지 않도록 할 수 있음
- 사용자의 환경에 대한 가장 자세한 정보이므로 여러모로 활용 가능

Request/Response: Content-Type

- 전송되는 데이터의 종류를 표현하는 헤더
- 요청/응답 양쪽에서 전송/활용할 수 있음
- 값은 MIME이라는 프로토콜을 따름

Request/Response: Content-Type - MIME

- Multipurpose Internet Mail Extensions
- 음악, 사진, 영상 등등의 데이터를 간결하게 표현하기 위한 형식
- 표준이므로 웬만한 브라우저 등에서 호환 가능
- https://www.iana.org/assignments/media-types/media-types.xhtml

Request/Response: Content-Type - MIME

- 0|□|⊼|: image/bmp, image/jpeg, image/png ...
- 음악: audio/mpeg, audio/ogg, ...
- 영상: video/mp4, ...

X-Powered-By, Server

- 프로그램을 구현하는데 사용한 언어, 프레임워크. 서버 데몬의 종류를 표현하는 헤더
- 프로그램 구현에 거의 사용될 필요가 없고, 전송해봐야 사용자에게 별다른 영향을 주지 않음
- 오히려 Banner Grabbing에 활용될 수 있으므로 설정을 통해 제거하는게 바람직

PHP에서 디버깅

- var_dump 함수는 인자로 넘어온 값의 내부 표현을 출력해줌
- 단순 변수, 배열, 객체 등등의 타입, 값을 출력

```
sample_form_proc_post.php
              var_dump($_POST);
                       localhost:60000/sample_form_proc_post.php
array(2) { ["id"]=> string(4) "test" ["password"]=> string(4) "1234" }
```

실습: PHP Calculator

- 폼 POST 요청을 사용하여 간단한 계산기를 구현해보세요.
- 파일명은 calculator.php와 calculator_proc.php를 사용할 것
- 덧셈, 뺄셈, 나눗셈, 곱셈이 가능해야합니다!
- 기한: 다음주 활동시간까지
- 선택사항1: CSS 등을 사용해서 예쁘게 만들어보세요!
- 선택사항2: 굳이 calculator_proc.php를 사용할 필요가 있을까요?

질문?