~~학습 1.~~

~~- 주제 : Http, Https 학습문서 만들기~~

~~- 학습상세 :~~

~~\* 각 프로토콜의 설명 ㅇㅇ~~

~~\* URI & URL ㅇㅇ~~

\* Method(post, get 등등)

\* 프로토콜 헤더 포멧

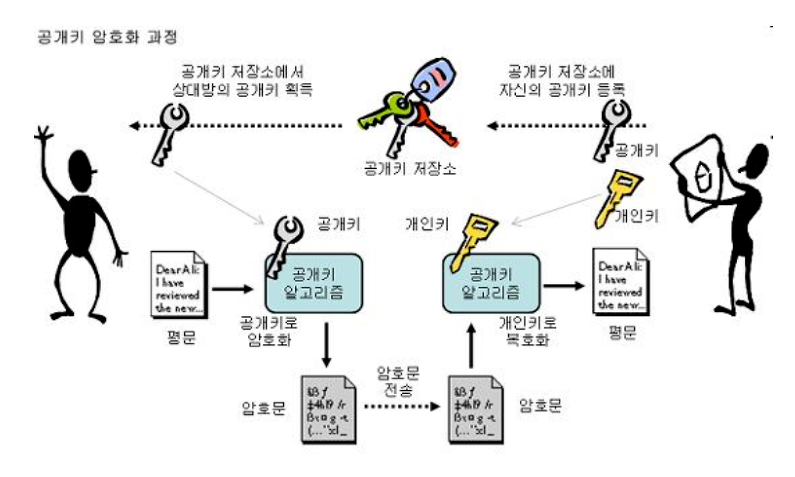
\* 응답코드

\* 쿠키와 세션

- 참고사항 : 구글에서 정보를 가져오는것은 좋으나, 문서를 다음에 다시 읽었을때 가독성 좋도록 편집해주세요.

|  |  |  |
| --- | --- | --- |
|  | HTTP(HyperText Tranfer Protocol) | HTTPS(HyperText Tranfer Protocol Secure Socket) |
| 개념 | -웹 브라우저(Client)와 웹서버(Server)간에 메시지 교환 프로토콜 | -같은 통신 프로토콜, HTTP와 거의 흡사 |
| 교환 방식 | 단순 텍스트 |  |
| 특징 | -만약 누군가 주고받는 네트워크 패킷을 가로채 본다면 메시지 내용이 그대로 보이게 되므로 보안 취약 | -HTTP의 취약점인 보안 취약을 보완하기 위해 주고받는 모든 메시지를 암호화  -암호화 방식 : 대칭/비대칭  -공유된 세션키(대칭키)를 모르는 상황에서 암호를 풀기는 불가능  -SSL(Secure Sockey Layer)를 이용하여 정보를 암호화.  -HTTP 메시지(Text)를 암호화하는 것  -\*공개키 암호화 방식 |
|  |  |  |

\*공개키 암호화 과정



<https://jeong-pro.tistory.com/89>

|  |  |  |
| --- | --- | --- |
|  | **URL(Uniform Resource Locator)** | **URI(Uniform Resource Identifier)** |
| 개념 | -네트워크 상에서 자원이 어디 있는지를 알려주기 위한 규약이다. 즉, 컴퓨터 네트워크와 검색 매커니즘에서의 위치를 지정하는 웹 리소스에 대한 참조이다. | -인터넷에 있는 자원을 나타내는 유일한 주소  -URI의 존재는 인터넷에서 요구되는 기본조건으로서 인터넷 프로토콜에 항상 붙어 다님. |
| Ex) | - <http://ko.wikipedia.org/>  라는 주소는 서버도 나타내기 때문에 URL이면서 URI이다.  - <http://sunychoi.github.io/java/2015/04/12/java-bigdecimal.html> 라는 주소는  sunychoi.github.io 호스트 주소 하위에 java/2015/04/12 디렉토리아래 java-bigdecimal.html 이라는 자원의 위치를 가리키고 있으므로 URL이면서 URI이다.  - <https://www.google.co.kr/search?q=uri> 라는 주소는 https 프로토콜을 가지고있고 호스트 이름을 가지고 있다. 하지만 그 뒤에  /search?q=uri와 같은 문자열이 붙어 있음 q=rui 는 query string 인 q의 값에 따라 여러가지 결과값을 가져올 수 있다.  URL : <https://www.google.co.kr/search> 여기까지 이고  q=uri라는 식별자가 필요하므로, https://www.google.co.kr/search?q=uri 이 주소는 URI이지만 URL은 아니다. | |
| 특징 | -URL은 웹 주소 뿐만 아니라 컴퓨터 네트워크상의 자원을 모두 나타낼 수 있다.  -주소에 접근하려면 해당 URL에 맞는 프로토콜을 알아야한다.  -FTP 프로토콜인 경우 FTP 클라이언트를 이용, HTTP인 경우 웹 브라우저를 이용해야함, 텔넷인 경우 텔넷 프로그램을 이용하여 접속해야함. | -URL,URN의 상위 개념  -위의 ex로 이해 |
|  |  |  |

**POST메서드 GET메서드**

:웹 서비스 개발에 주로 사용하는 메서드이다. 사용자가 URL을 브라우저 주소창에 작성하고 엔터를 누르면 원하는 웹페이지가 나온다. 사용자는 웹페이지를 보기위해 단순한 일을 한 것 이지만, 특정 웹페이지를 사용자 웹브라우저에게 보여주기 위해서는 내부적인 처리들이 있다.

-요청 : 클라이언트가 서버에게 웹페이지를 보여달라고 말하는 것.

-응답 : 서버가 클라이언트에게 요청받은 것에 대한 대답으로, 웹페이지 내용을 표현하기 위해 html문서로 주는 것을 응답이라 부른다.

**HTTP 패킷**

: 클라이언트가 서버로 요청을 했을 때, 보내는 데이터를 HTTP 패킷이라 표현함.

HTTP 프로토콜을 쓰므로, 앞에 HTTP가 붙고 인터넷을 통해 보내는 데이터를 패킷이라 표현하므로, HTTP패킷 이라 부른다. HTTP패킷의 구조는 크게 헤더와 바디로 나뉘어진다.

헤더에는 7?8?가지 HTTP 메서드 방식 중 무엇을 썻는지, 클라이언트의 정보, 브라우저 정보, 접속할 URL 등등 과 같은 클라이언트 정보를 담는다.

바디는 보통 비어있다. 하지만, 특정 데이터를 담아서 서버에게 요청을 보낼 수 있다.

\*8가지 HTTP 메서드 : 1)GET 2)POST 3)HEAD 4)OPTION 5)PUT 6)DELETE 7)TRACE 8)CONNECT

**GET방식 vs POST 방식**

: 두 방식 모두, 서버에 요청을 하는 메서드이다. 클라이언트가 서버에 요청을 할 때, 제공해야 하는 자원이 있다고 하자. Ex)로그인을 하는 경우 -> 아이디와 패스워드는 클라이언트가 작성한 후, 그 정보를 서버에 요청하여 클라이언트가 작성한 아이디와 패스워드가 올바른 것인지 검사를한다.

|  |  |  |
| --- | --- | --- |
|  | **POST** | **GET** |
| 개념 | WWW에서 사용되는 HTTP가 지원하는 요청 방식이다. | GET방식은 URL에 데이터를 붙여서 보냄 |
| 특징 | -요청 URI(URL)에 폼 입력을 처리하기 위해 구성한 서버측 스크립트(ASP,PHP,JSP) 혹은 CGI 프로그램으로 구성되고 Form Action과 함께 전송됨 이때 헤더 정보에 포함되지 않고 데이터 부분에 요청 정보가 들어가게 된다.  -키 값의 쌍은 ‘&’ 문자로 구분되며 각 키는 ‘=’ 문자의 값과 구분된다. 키와 값들은 둘 다 공백을 ‘+’ 문자로 대체하여 영어숫자가 아닌 그 밖의 모든 문자는 퍼센트 인코딩 처리한다.  -데이터 전송을 기반으로 한 요청 메서드  - BODY에다가 데이터를 넣어서 보낸다.  따라서 헤더필드 중 body를 설명하는 content-type이라는 헤더 필드가 들어가고 어떤 타입인지 명시한다. | -URI(URL)가 가진 정보를 검색하기 위해 서버 측에 요청하는 형태 |
| Ex) | Name: Gareth Wylie  Age: 24  Formula: a + b == 13%!  위의 키 값 쌍은  Name=Gareth+Wylie&Age=24&Formula=a+%2B+b+%3D%3D+13%25%21  다음과 같이 인코딩된다.  다음과 같이 클라이언틔 데이터를 URL 뒤에 붙여서 보낸다.  [컨텐츠 타입]   **application/x-www-form-urlencoded**  **(default)**   **text/plain**   **multipart/form-data** | [[www.example.com?id=mommoo&pass=1234](http://www.example.com?id=mommoo&pass=1234)] (1)‘?’를 통해서 URL의 끝을 알리면서 데이터 표현의 시작점을 알린다. 데이터는 key와 value 쌍으로 넣어야한다.  Key : id,pass  Value : mommoo,1234  (2)중간의 ‘&’마크는 구분자 이다. 2개이상의 key-value쌍 데이터를 보보낼 때는 &마크로 구분  (3)URL에 붙이므로,HTTP패킷의 헤더에 포함되여 서버에 요청 따라서, BODY에 특별한 내용을 넣을 것이 없으므로 BODY가 빈상태로 보내진다.따라서 헤더에 바디를 설명하는 Content-Type이라는 헤더필드는 들어가지 않는다.  (4)URL형태로 표현되므로, 특정 페이지를 다른사람 에게 접속하게 할 수 있다.  (5)간단한 데이터를 넣도록 설계 되어 데이터를 보내는 양의 한계가 있다. |

\*퍼스트 인코딩 : URL에 문자를 표현하는 문자 인코딩 방법. 이 방법에 따르면 알파벳이나 숫자 등 몇몇 문자를 제외한 값은 옥텟 단위로 묶어서,16진수 값으로 인코딩한다.ex) %EC%9C%84%ED%82%A4%EB%B0%B1%EA%B3%BC

**HEAD 메서드 방식**

**;** HEAD 요청 방식은 GET과 유사한 방식이나 웹 서버에서 헤더 정보이외에는 어떠한 정보도 보내지 않는다. 웹 서버의 다운 여부 점검(Health Check) 이나 웹 서버 정보 (버전 등) 등을 얻기 위해 사용.