



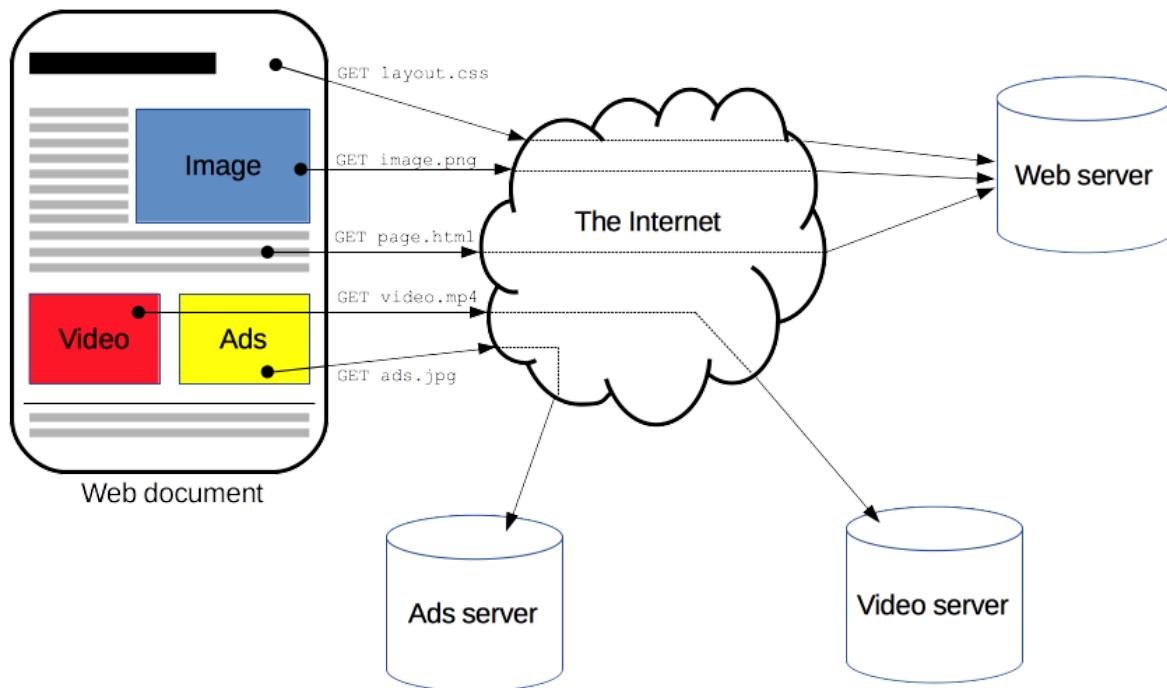
HTTP의 기초

CodeSquad Master
Hoyoung Jung

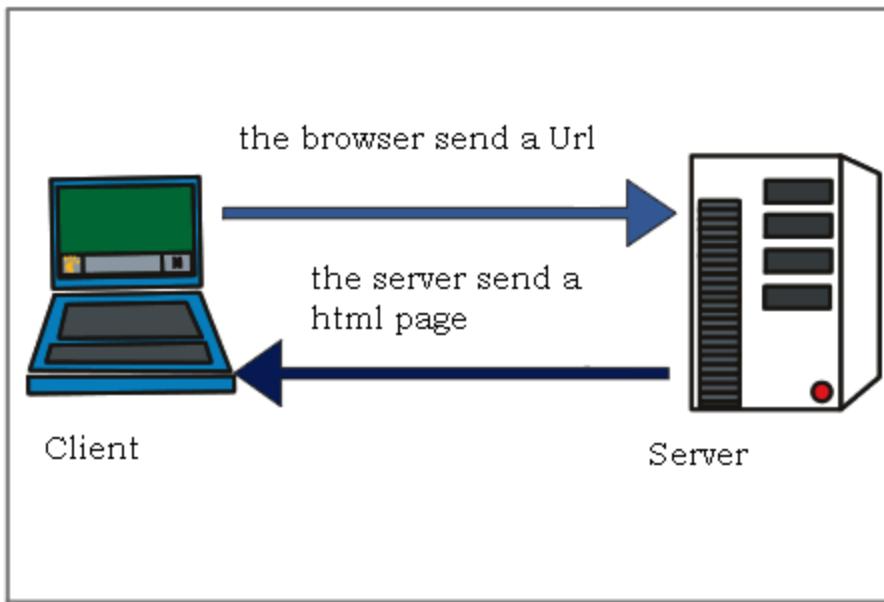
HTTP

HyperText Transfer Protocol, 초본문전송규약?

WWW 상에서 정보를 주고받을 수 있는 프로토콜이다. 주로 HTML 문서를 주고받는 데에 쓰인다. TCP와 UDP를 사용하며, 80번 포트를 사용한다.



Web Client와 Server



엄청난 추상화가 되어 있는 그림

클라이언트: 주로 웹 브라우저로 항상 요청만 한다.

서버: 클라이언트의 요청에 대해 적당한 문서를 제공해 준다.

프록시: 클라이언트와 서버 사이에 존재. 무언가 일을 해 준다.

- 캐싱, 필터링, 로드 밸런싱, 인증, 로깅 등의 다양한 기능을 수행

HTTP의 특징

- 간단
- 확장 가능
- 상태가 없다. = stateless
- 세션은 존재 = 쿠키를 이용해
- HTTP1 --> HTTP/1.1 --> HTTP/2로 발전

실습1: node.js 서버 실행해 보기

```
//server.js
var http = require('http');
var url = require('url');

http.createServer(function (request, response) {
    console.log("url: " + url.parse(request.url).pathname)
    response.writeHead(200, {'Content-Type': 'text/html'})
    response.end('<h1>Hello World</h1>\n');
}).listen(8000);

console.log('Server running at http://127.0.0.1:8000/');
```

```
$ node server.js
```

node.js가 없는 경우 && mac 이라면

```
$ git clone http://github.com/code-squad/white-common  
$ cd white-common/week3  
$ python -m SimpleHttpServer 8000
```

1. 웹 브라우저로 열어 보기

2. curl로 열어보기

```
$ curl 127.0.0.1:8000
```

telnet으로 열어 보기

```
$ telnet 127.0.0.1 8000
GET / HTTP/1.0
Host: localhost:8000
# 엔터 한번 더
```

```
$ telnet 127.0.0.1 8000
GET /squad/ HTTP/1.0
Host: localhost:8000
# 엔터 한번 더
```

HTTP Response example

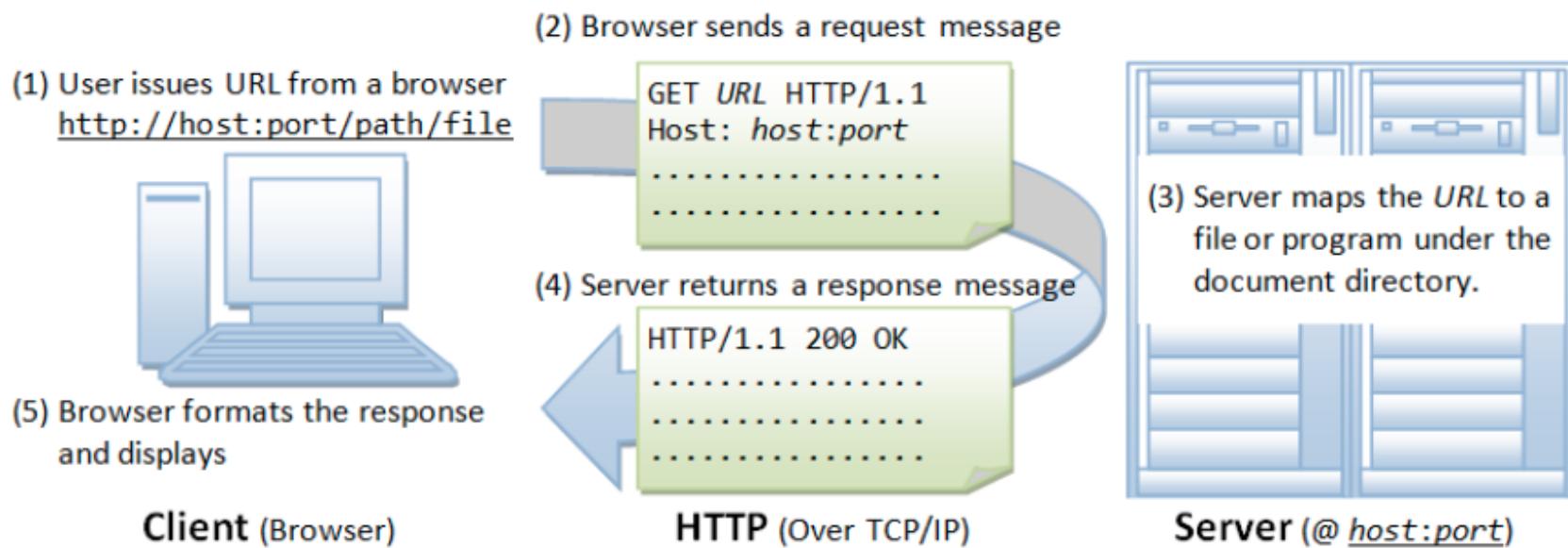
```
HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/2.7.10
Date: Thu, 12 Jan 2017 07:57:37 GMT
Content-type: text/html
Content-Length: 161
Last-Modified: Thu, 12 Jan 2017 07:48:02 GMT
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Simple HTML Page</title>
  </head>
  <body>
    <h1>2nd Page: Squad</h1>
  </body>
</html>
```

웹 브라우저로 다시 열어 봅시다.

개발자 도구의 네트워크 탭을 열고 새로 고침을 해 봅시다.

HTTP 동작방식2



- 실제로는 DNS 서버가 개입한다.

HTTP Request 메시지의 구조

```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35
bookId=12345&author=Tan+Ah+Teck
```

Request Line

Request Headers

A blank line separates header & body

Request Message Body

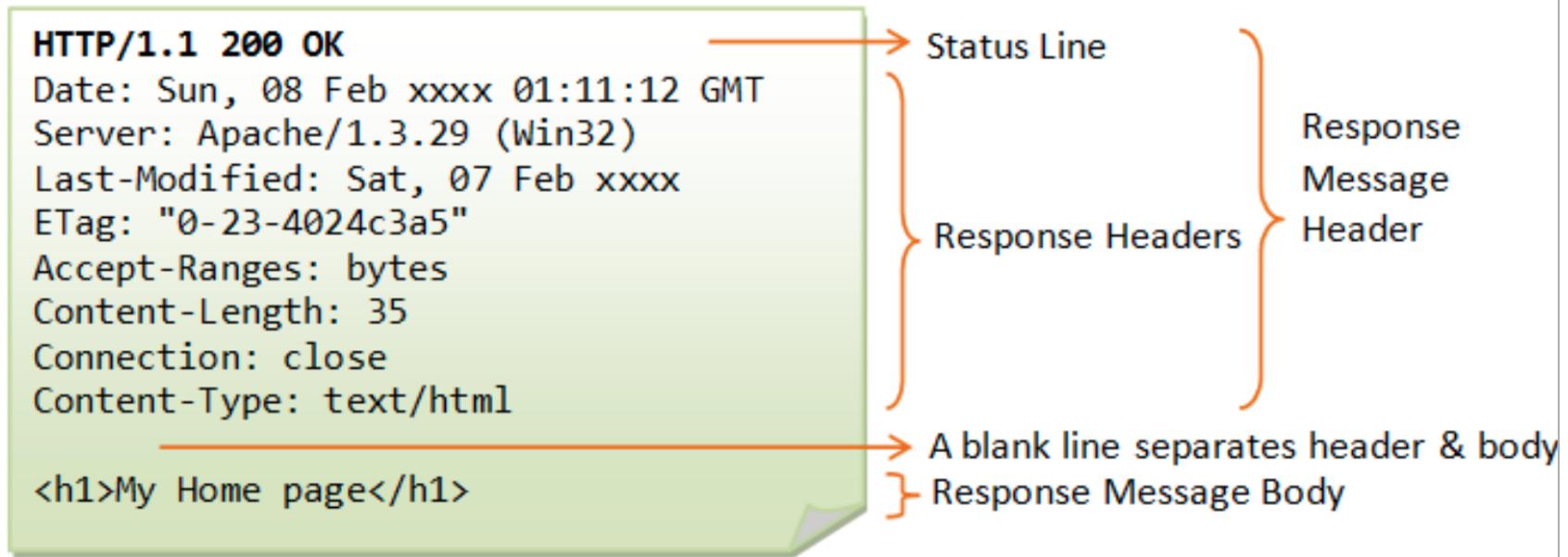
Request Message Header

The diagram illustrates the structure of an HTTP Request message. It shows the following components:

- Request Line:** The first line of the message, containing the method (GET), the resource path (/doc/test.html), and the protocol version (HTTP/1.1).
- Request Headers:** A group of key-value pairs that provide information about the request. These include Host, Accept, Accept-Language, Accept-Encoding, User-Agent, and Content-Length.
- Request Message Body:** The final part of the message, which contains the query parameters bookId=12345&author=Tan+Ah+Teck.
- Request Message Header:** A bracketed group that encompasses both the Request Headers and the Request Message Body.

A horizontal line separates the Request Headers from the Request Message Body. A blank line is also shown between the Request Headers and the Request Message Body.

HTTP Response 메시지의 구조



브라우저의 동작

- 최초에는 HTML 을 가져옴
- HTML에서 CSS, js, 이미지에 대한 링크 정보를 추출
- 추출한 정보의 URL을 이용 새로운 요청을 보냄
- 모든 웹 자원을 받아와서 렌더링 시작
- 1.1 은 파이프라인, 2.0은 병렬처리로 성능개선

웹의 구성요소

HTTP

HTML

URL

URL: Uniform Resource Locator

(URI = URL + URN)

리소스를 식별하는 주소

스킴:사용자이름:비번@호스트:포트/경로;패러미터?쿼리#프래그먼트

```
https://honux77:pw1234@github.com:443/
honux77/MMT?file=sum.py#30
```

- 쿼리: 편의상 =과 &를 사용한다.
- 프래그먼트: 클라이언트만 사용

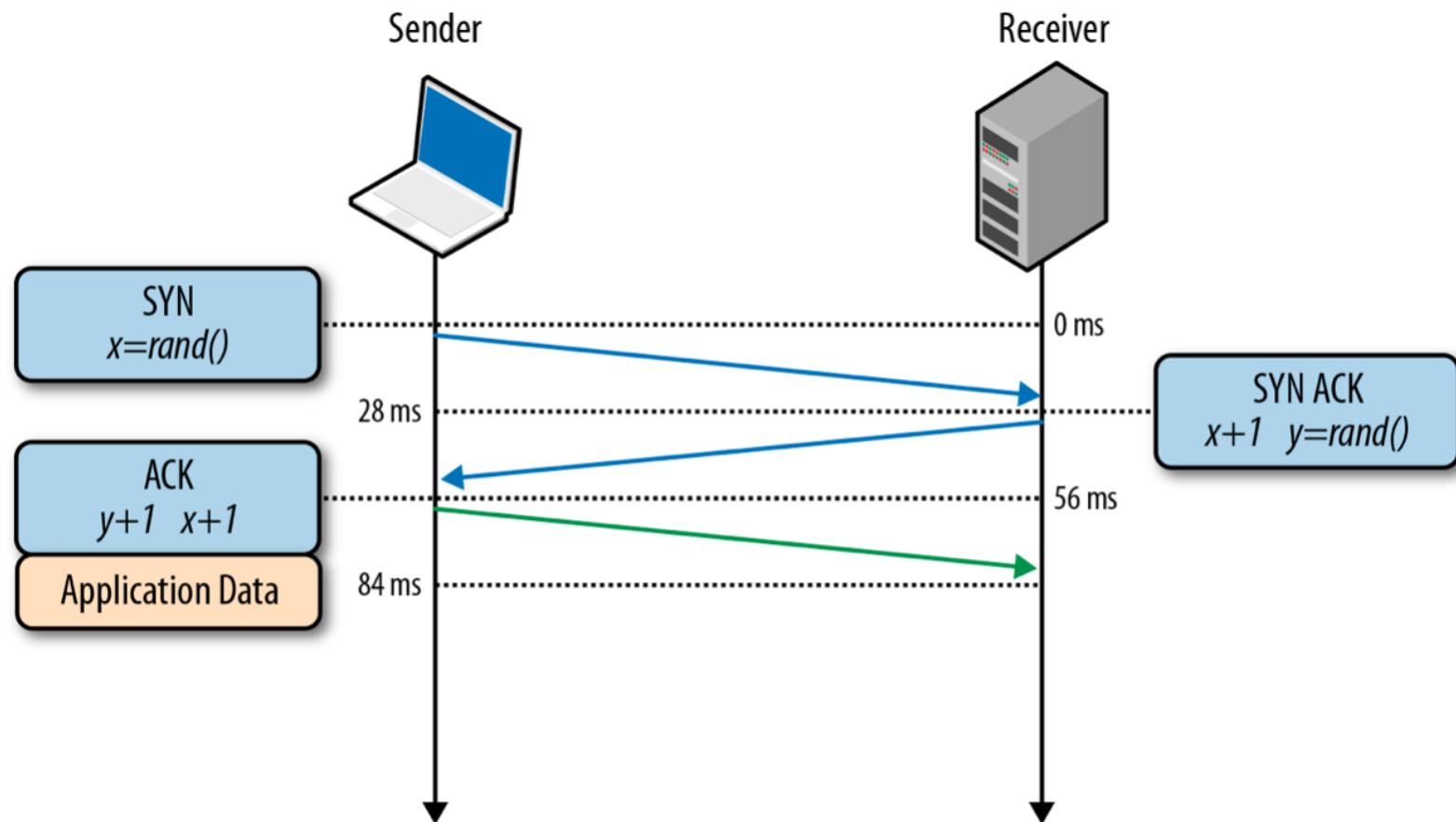
포맷

MIME (Multipurpose Internet Mail Extensions) 타입으로 포맷을 분류
원래는 전자 우편을 위한 표준

```
Content-Type: text/plain
text/html
text/css
image/jpeg
image/png
audio/mpeg
audio/ogg
audio/*
video/mp4
application/octet-stream
multipart/mixed
```

- https://developer.mozilla.org/ko/docs/Web/HTTP/Basics_of_HTTP/MIME_types

TCP의 3-way Handshake



HTTP method

SAFE METHODS	{ GET HEAD	HTTP/1.1 MUST IMPLEMENT THIS METHOD INSPECT RESOURCE HEADERS
MESSAGE WITH BODY	{ PUT POST PATCH TRACE OPTIONS DELETE	DEPOSIT DATA ON SERVER – INVERSE OF GET SEND INPUT DATA FOR PROCESSING PARTIALLY MODIFY A RESOURCE ECHO BACK RECEIVED MESSAGE SERVER CAPABILITIES DELETE A RESOURCE – NOT GUARANTEED

주로 많이 사용: GET, POST

GET VS POST

- GET은 URL 뒤에 쿼리스트링으로 패러미터를 전달
- POST는 요청 바디에 숨겨져 보내짐
- form을 이용해서 대용량 파일을 전송하면?
- 구글 주소창에서 검색을 하면?

추가로 알아야 하는 것들

- TCP / IP
- 소켓 통신
- 네트워크 기본 지식
- 그림으로 배우는 HTTP & Network Basic 정도는 꼭 보자!

HTTP 응답 코드

상태	응답	의미
200	OK	정상적인 처리
302	See Other	주로 리다이렉트 용도
404	Not Found	리소스가 없다
403	Forbidden	권한 없음
500	Internal Server Error	서버 내부 오류
502	Bad Gateway	중간 계층 오류
503	Service Unavailable	서비스 제공불가

로그인 상태란 어떤 상태일까?

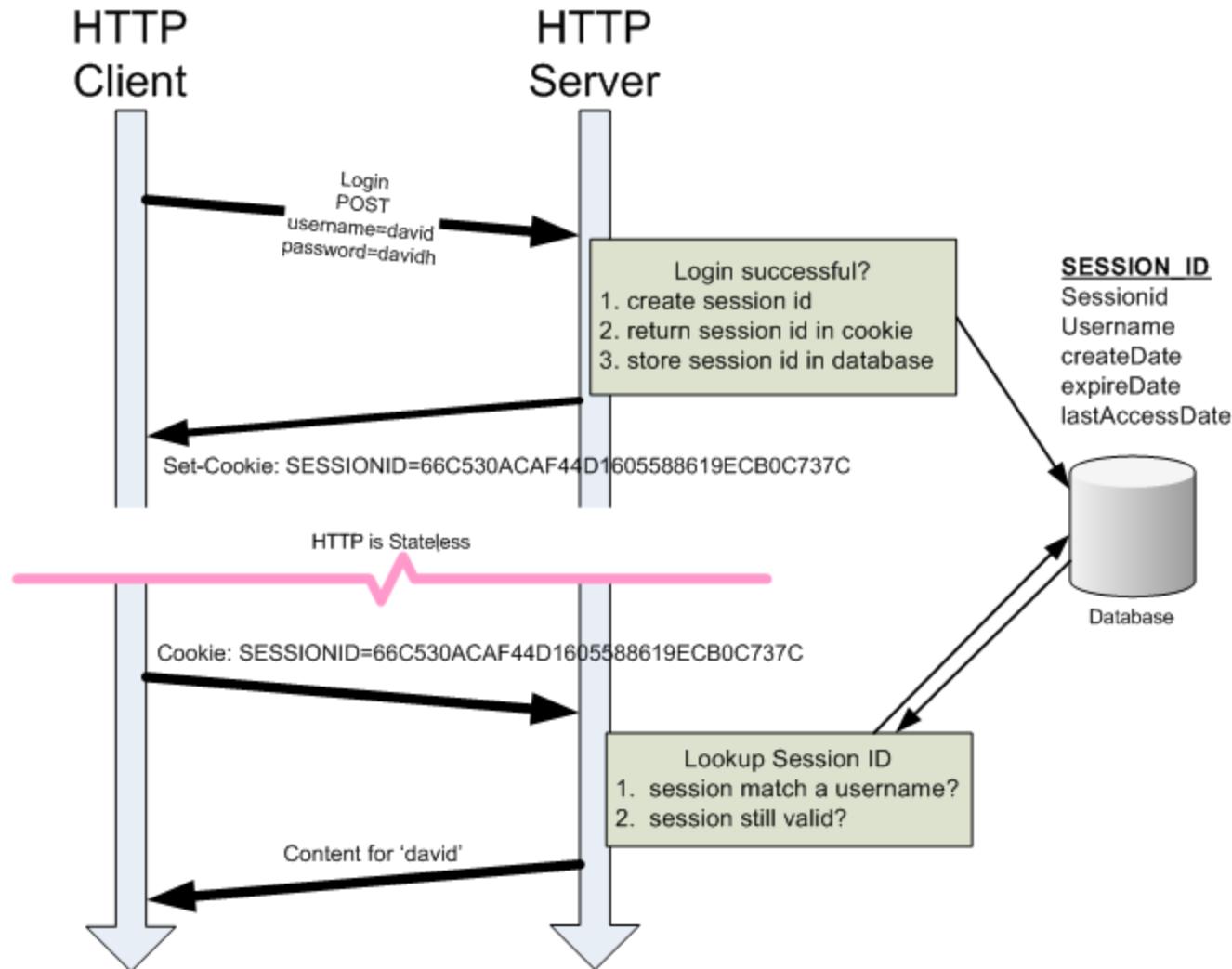
사용자 식별 - 인증을 위한 방법

- IP 추적
- HTTP Authentication
- URL에 식별자 포함
- Cookie

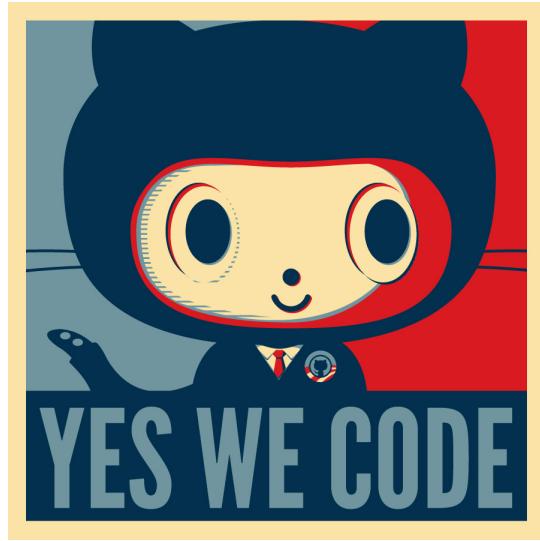
쿠키



session



Thanks 😊



더 공부하기

- www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html
- <https://developer.mozilla.org/ko/docs/Web/HTTP/Overview>
- <https://howdns.works/ep1/>
- <http://www.slideshare.net/devview/d2-campus-http>
- <http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9788931447897>

쉬어가는 페이지

CRLF - <http://ohgyun.com/554>