숭실대학교 Soongsil University

Web Programming 기초 III

숭실대학교 스마트시스템소프트웨어 학과
Department of Smart Systems Software

Contents

1. JSON
2. Cookies
3. Concurrency
4. HTTPS
5. HTTP/2
6. HTTP 학습용 사이트

JSON

- JavaScript Object Notation의 약어로, network등을 통해 client(web browser)/server(web server) 사이에 전송되는 데이터를 표현하는 국제 표준 방식임
  - JSON은 JavaScript에서 처음 사용되었으나, C, C++, Java, Python등 거의 모든 language에서 지원
  - Filename extension : .json
  - JSON media type(content-type) : application/json
  - JSON 데이터와 language의 object data type의 conversion이 매우 용이하여, client에서 server로 data를 보낼 때 object -> json 형태로 바꿔 보내고, server에서는 json 형태를 object type 형태로 변환하여 프로그램내에서 쉽게 사용할 수 있음
- 과거에는 client/server 사이의 데이터 전송할 때 XML 데이터 포맷을 많이 사용하였으나, 현재는 JSON 형태가 거의 산업 표준으로 자리 잡았음

- JSON과 XML 데이터 형태 예제

XML

```
<empinfo>
   <employees>
      <employee>
         <name>James Kirk</name>
         <age>40></age>
      </employee>
      <employee>
         <name>Jean-Luc Picard</name>
         <age>45</age>
      </employee>
      <employee>
         <name>Wesley Crusher</name>
         <age>27</age>
      </employee>
   </employees>
</empinfo>
```

JSON

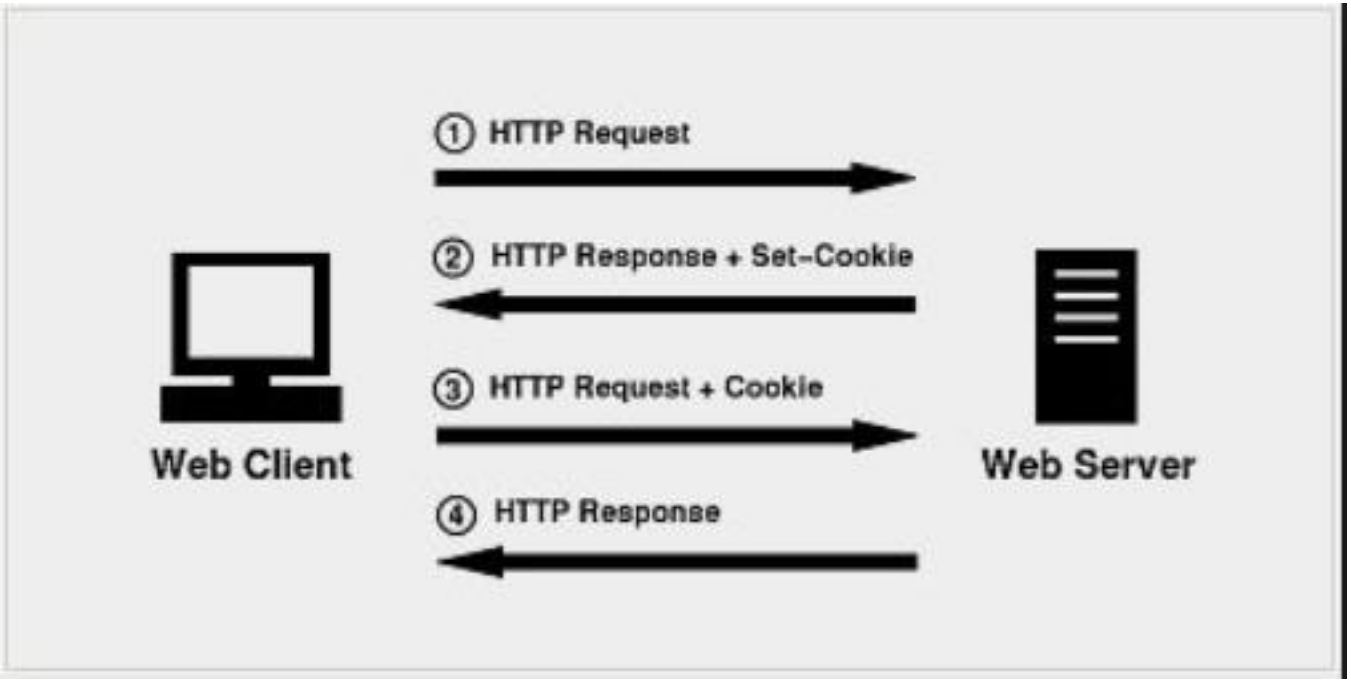
```
{  "empinfo" :
      {
            "employees" :  [
            {
                  "name" : "James Kirk",
                  "age" : 40,
            },
            {
                  "name" : "Jean-Luc Picard",
                  "age" : 45,
            },
            {
                  "name" : "Wesley Crusher",
                  "age" : 27,
            }
                              ]
      }
}
```

Cookies

- 서버와 클라이언트 간에 교환되는 작은 데이터
  - Name과 value의 String 쌍
- 서버는 클라이언트로 쿠키를 보내고, 이후 클라이언트는 매번 요청에 이 값을 전송한다
- Cookies 사용 목적
  - 세션관리(Session Management)
    - 서버에 저장해야 할 개인 로그인(id, password) 정보, 장바구니 정보, 게임 스코어 등의 정보 관리
  - 개인화(Personalization) : 사용자 선호, 테마 등의 setting
  - Tracking : 사용자 행동을 기록하고 분석하는 용도
- Web browser를 종료하면 쿠키도 사라진다. 그러나 쿠키를 저장하여 브라우저를 새로 open하더라도 사용하게 할 수 있다

Cookies

- Web browser와 web server사이에 cookie를 생성 및 사용 프로세스 flow



Cookies

- Server response header에서 cookie 설정

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: yummy_cookie=choco
Set-Cookie: tasty_cookie=strawberry

[page content]
```

- Web browser에서 server로 cookie 전달하고, server에서 cookie 사용

```
1 GET /sample_page.html HTTP/1.1
2 Host: www.example.org
3 Cookie: yummy_cookie=choco; tasty_cookie=strawberry
```

Cookies

- Server response header에서 set-cookie 속성
  - Expires : 최대 생존시간 (Date로 표기), 이 필드가 없으면 session cookie로 인식하고 client가 종료될 때 cookie 소멸함
  - Max-age : 초 단위로 지정 (음수이면 즉시 만료)
  - Domain : domain지정하면 하위 domain까지 cookie 전달됨
  - Path : URL 경로 (예: path=/docs, /docs/Web 등으로 표기)

```
Set-Cookie: <cookie-name>=<cookie-value>
Set-Cookie: <cookie-name>=<cookie-value>; Expires=<date>
Set-Cookie: <cookie-name>=<cookie-value>; Max-Age=<non-zero-digit>
Set-Cookie: <cookie-name>=<cookie-value>; Domain=<domain-value>
Set-Cookie: <cookie-name>=<cookie-value>; Path=<path-value>
Set-Cookie: <cookie-name>=<cookie-value>; Secure
Set-Cookie: <cookie-name>=<cookie-value>; HttpOnly

Set-Cookie: <cookie-name>=<cookie-value>; SameSite=Strict
Set-Cookie: <cookie-name>=<cookie-value>; SameSite=Lax

// Multiple directives are also possible, for example:
Set-Cookie: <cookie-name>=<cookie-value>; Domain=<domain-value>; Secure; HttpOnly
```

Concurrency

- Web server application process가 여러 개의 thread를 fork하여 각 thread가 web browser의 request를 처리하여 같은 시간대에 web server가 여러 개의 web browser request를 동시에 처리하는 것을 말함
  - Bookmark Server 예제 프로그램에서 다음과 같이 변경하면 concurrency 지원함

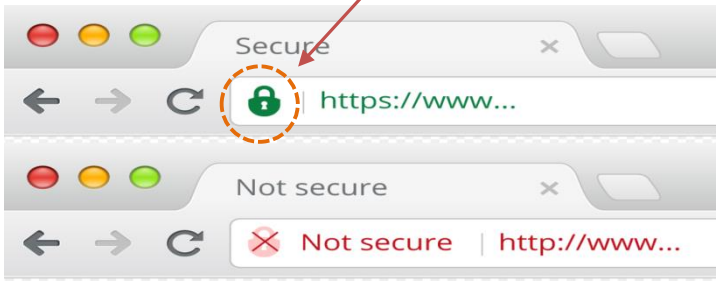
```
import threading
from socketserver import ThreadingMixIn

// This is an HTTPServer that supports thread-based concurrency
// http.server.HTTPServer standard library : concurrency 지원하지 않음
// ThreadingMixIn의 역할 : concurrency 지원하기 위한 helper class
class ThreadHTTPServer (ThreadingMixIn, http.server.HTTPServer):

if __name__ == '__main__':
    port = int(os.environ.get('PORT', 8000))
    server_address = ('', port)
    httpd = ThreadHTTPServer(server_address, Shortener)
    httpd.serve_forever()
```

- HTTPS = HTTP + encrypted connection
  - Encryption은 TLS(Transport Layer Security) 프로토콜을 따름
  - Privacy : Web browser와 web server간 data를 암호화하여 주고 받아, 특정 web browser와 web server만 data 해독 가능함
  - Web server authentication : web browser가 https://securesites.com로 접속할 때, web browser가 받는 data 결과값은 web server인 securesites.com로부터 받은 것이라고 인증하는 것
  - Data integrity : web server가 보낸 데이터가 통신 중에 modify 되거나 replace되지 않았다고 guarantee하는 것

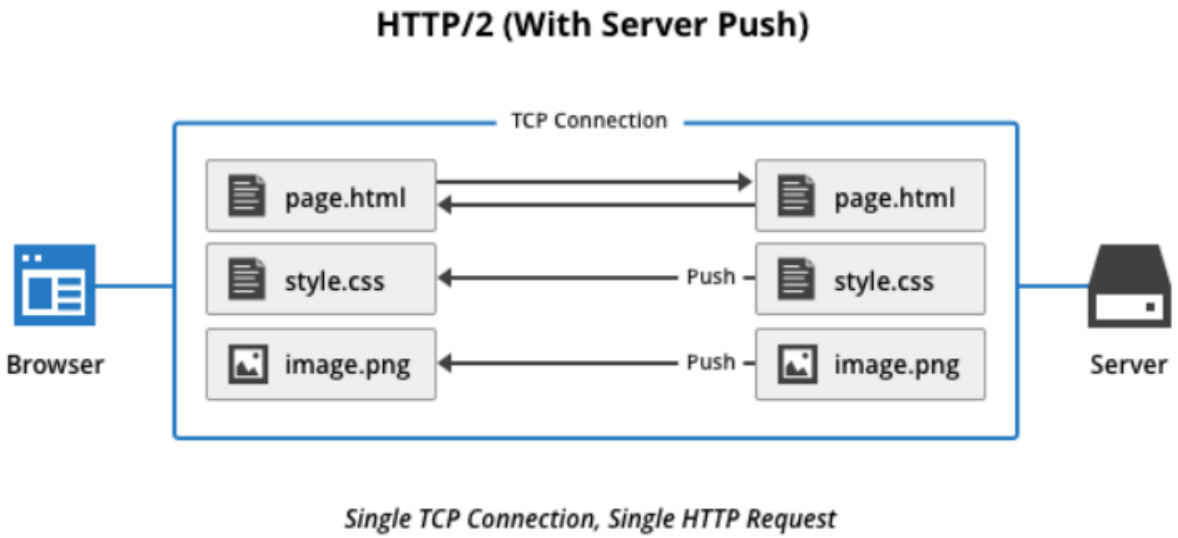
- TLS은 SSL(Secure Socket Layer) 3.0 기반으로 만들어졌음
- Web browser의 https setup 정보
  - Web 주소의 시작은 https 로 시작함
  - Browser 주소 왼쪽의 lock icon : browser마다 틀릴 수 있음. chrome은 오른쪽의 icon과 같이 제공함



- 보안
- 정보 또는 안전하지 않음
- 안전하지 않음 또는 위험

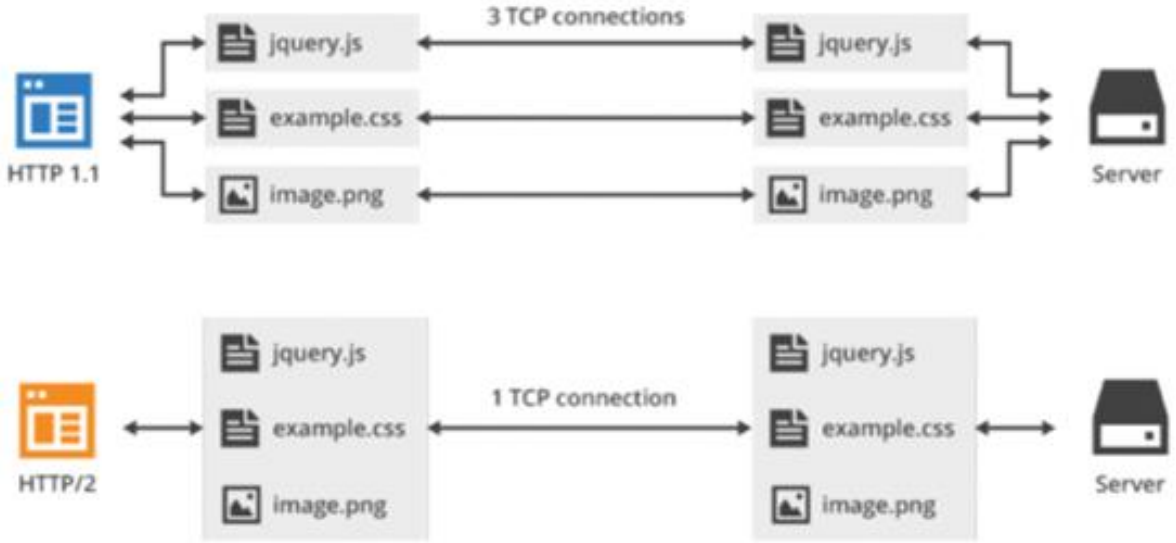
  - Web server가 신뢰성이 있는 사이트인지 여부는 인증기관에서 제공한 TLS certificate를 통해서 알 수 있음
    - Chrome인 경우 lock icon을 우측 click하여 정보 확인 가능

- HTTP/2는 http/1.1의 새로운 버전임
  - Web browser와 web server간 데이터 전송할 때 http/1.1은 application 레벨에서 이루어지던 것을 http/2에서는 transport layer에서 최적화하여 어플리케이션 수행 속도 개선함
  - http/2에서는 web browser가 html file을 server에 요청하면, html file안에 있는 css, javascript, image 파일들도 자동으로 동시에 web browser에게 보내줌 (server push 기능 제공)



HTTP/2

- http/1.1에서는 web browser가 server에 html file을 요청하면, html file이 server로 부터 browser에 먼저 온 후, html file안에 있는 css, javascript file은 web browser에서 다시 server에 요청하여 별도로 가져옴.



- 2017년 기준 python framework에서 fully support되지 못함

- Http/2 demo site
  - Gophertiles demo : https://http2.golang.org/gophertiles
  - Browser http/2 지원 여부 : https://caniuse.com
  - http://www.http2demo.io/
  - https://http2.akamai.com/demo

HTTP 학습용 사이트

- Mozilla Developer Network's HTTP index page : https://developer.mozilla.org/en-US/docs/Web/HTTP
- HTTP/1.1 standard documents : https://tools.ietf.org/html/rfc7230
- HTTP/2 standard documents : https://http2.github.io/
- SSL/TLS certificate 무료 제공 및 사용방법 가이드 제공site : https://letsencrypt.org/