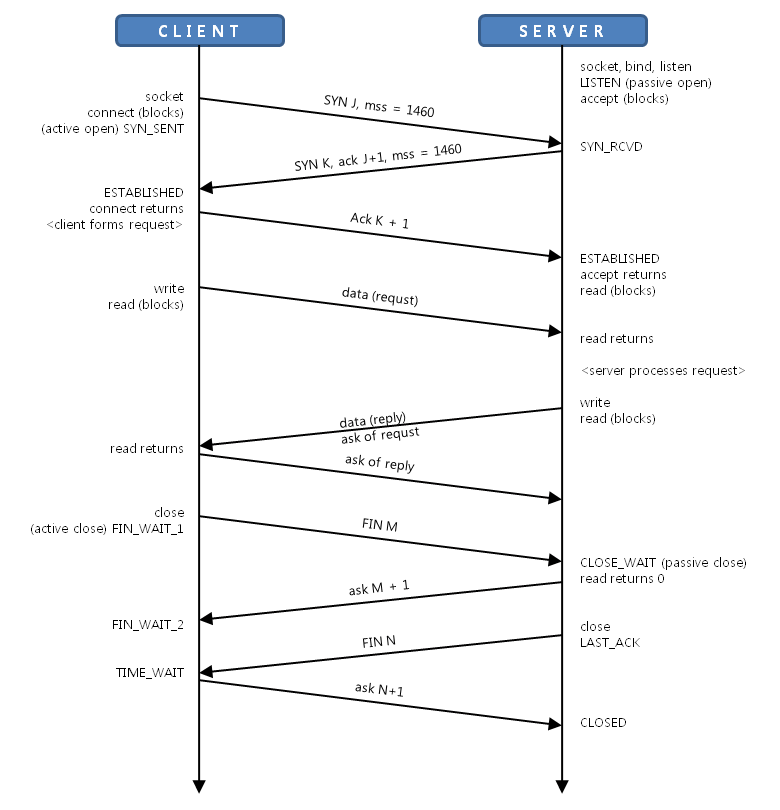
HTTP개관 1장

HTTP 인터넷 멀티미디어 배달부

http는 신뢰성 있는 데이터 전송 프로토콜을 사용하기 때문에 데이터가 지구 반대편에서 와도 전송중 손상되거나 꼬이지 않음을 보장한다(TCP)



웹클라이언트와 서버

웹 콘텐츠는 웹서버에 존재한다.

웹서버는 HTTP 프로토콜로 의사통신하기 때문에 보통 HTTP서버라고 한다. (웹서버)

클라이언트의 웹브라우저를 통해 → HTTP서버 에게 객체를 요청하고 → 클라이언트(웹브라우저)응답을 받아

화면에출력한다.

리소스

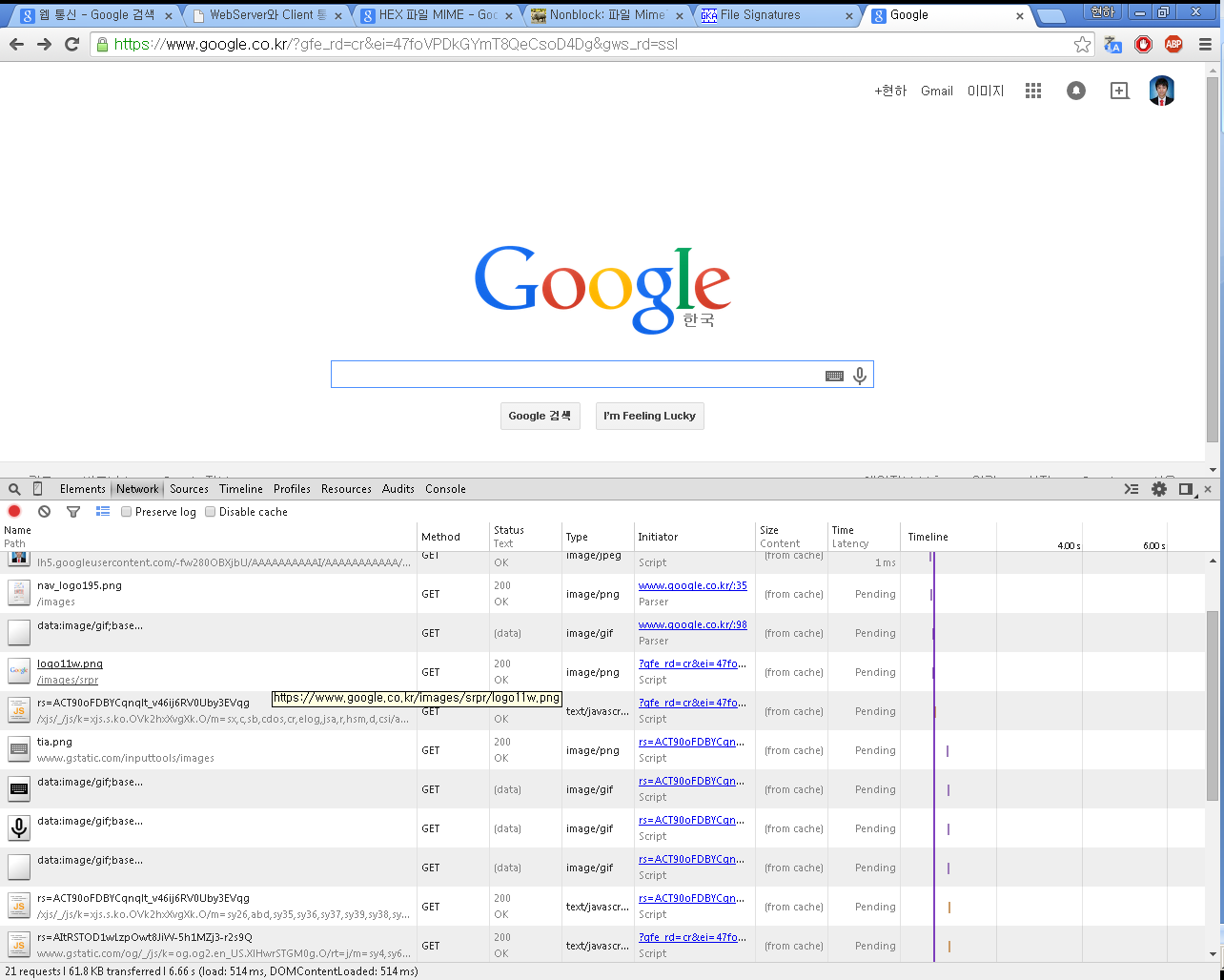
웹리소스는 굳이 정적인 리소스 일필요는 없다. 동적, 정적 리소스 다 포함된다.

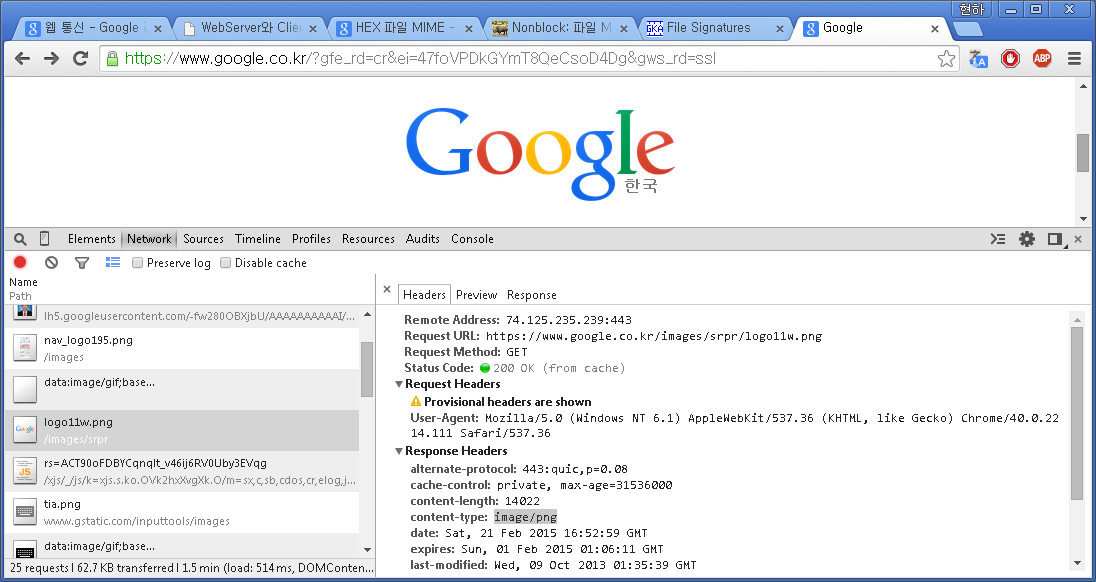
어떤 종류의 콘텐츠 소스도 리소스가 될수 있다.

미디어 타입

인터넷은 수천 가지 데이터 타입을 다루기 때문에 HTTP는 웹에서 전송되는 객체 각각에 신중하게

MIME(Multipurpose Internet Mail Extensions 다목적 인터넷 메일 확장)타입이라는 데이터 포맷 라벨을 붙인다.





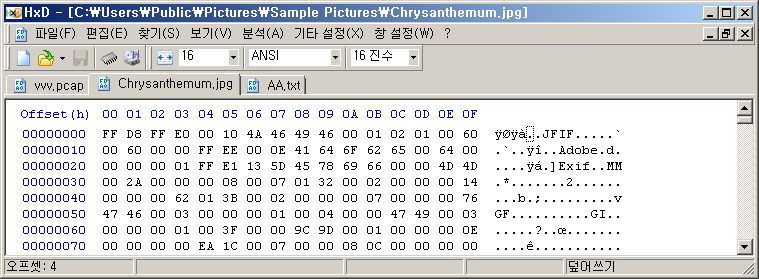
MMII타입 구분방법

**1. (웹서버 다운로드의 경우) HTTP 헤더의 컨텐트 타입 값을 검사**

**2. 파일 컨텐트의 바이트를 검사**

예를 들어 image/jpg 의 파일을 바이트스트림으로 읽어 표시하면 다음과 같은 헥스값을 볼 수 있다.

FF D8....



파일 시그너쳐

<http://www.garykessler.net/library/file_sigs.html>

자바로 MMIE알아내기

**public** **static** String getMIMEType(File file){

String contentType = **new** MimetypesFileTypeMap().getContentType(file);

**return** contentType;

}

**public** **static** String getMimeType(String fileUrl)

**throws** java.io.IOException, MalformedURLException

{

String type = **null**;

URL u = **new** URL(fileUrl);

URLConnection uc = **null**;

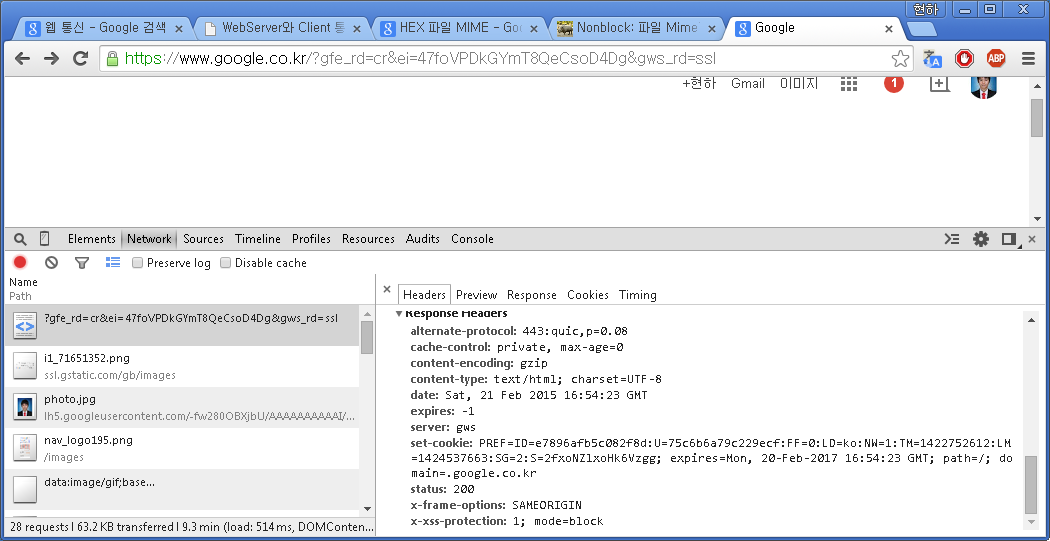
uc = u.openConnection();

type = uc.getContentType();

**return** type;

}

|  |  |
| --- | --- |
| 내용 | MMIE |
| HTML로 작성된 텍스트 | text/html |
| plain ASCII 텍스트 문서 | text/plain |
| Jpeg 이미지 | image/jpeg |
| Gif 이미지는 | image/gif |
| 애플 퀵타임파일 | video/quicktime |
| 파워포인트 (마소) | application/vnd.ms-powerpoint |



URI

클라이언트는 서버의 리소스를 요청해야 하기때문에 서버에 리소스 위치를 명시해야될 필요가 있다.

서버 리소스 이름은 통한 자원 식별자(uniform resource identifier) 혹은 URI 라고한다.

죠의 컴퓨터 가게의 웹 서버에 있는 이미지 리소스에 대한URI라면

|  |  |  |
| --- | --- | --- |
| http:// | [www.joes.com](http://www.joes.com) | /homeimg.jpg |
| http프로토컬 | 죠의 서버이동 | 리소스위치 |

이해하기 쉽게 동물로 표현하자면.

URI(동물) 가 좀더 상위 개념이라서 URL(강아지), URN(다람쥐) 등의 하위 개념을 포함한다.

따라서 위의 주소는 URL이면서 URI이다.

URL

통합 자원 지시자(uniform resource locator, URL)은 리소스가 정확히 어디에 있고 어떻게 접근할 수 있는지 분명히 알려준다.

|  |  |
| --- | --- |
| URL | 설명 |
| <http://www.or.com/index.html> | or사의 홈페이지메인 |
| http://www.or.com/logo.jpg | or사의 로고 |
| http://www.or.com/inventory-check.cgi?item=123 | or사의 123재고 체크 프로그램 URL |
| <ftp://jeo:tools4u@ftp.jo.com/locking-pliers.gif> | 비밀번호로 보호되는ftp 통해 이미지 접근 |

URN

URI두번째 종류는 유니폼 리소스 이름 (uniform resource name, URN)이다

URN은 콘테느를 이루는 한 리소스에 대해 그리소스의 위치에 영향받지 않는 유일무이한 이름 역할을 한다 이위치 독립적인 URN인 리소스를 여기저기 옮기더라도 문제 없이 동작한다. 리소스가 그 이름을 변하지 않게 유지하는한 여러 종류의 네트워크 접속 프로토콜로 접근해도 문제 없다.

트랜잭션

|  |  |
| --- | --- |
| http://cfile4.uf.tistory.com/image/18299C194AC8D609A05C1E | http://cfile2.uf.tistory.com/image/140A011249AB51AF56C634 |

HTTP메서드

메서드는 서버에게 어떤 동작이 취해져야 하는지 말해준다.

(웹페이지 가져오기, 게이트웨이 프로그램 실행하기, 파일삭제하기등)

|  |  |
| --- | --- |
| HTTP 메서드 | 설명 |
| GET | 서버에→클라이언트로 지정한 리소스를 보내라 |
| PUT | 클라이언트→서버로 보낸 데이터를 지정한 이름의 리소스를 저장하라 |
| DELETE | 지정한 리소스를 서버에서 삭제하라 |
| POST | 클라이언트 데이터를 → 서버 게이트웨이 애플리케이션으로 보내라 |
| HEAD | 지정한 리소스에 대한 응답에서, HTTP 헤더 부분만 보내라. |

1) **GET**: URL로 자원 또는 파일을 달라고(get) 요청함  
2) **POST**: Request 에 첨부한 몸체정보를 서버로 보내어, 요청한 URL 로 이 정보를 넘겨주라고 요청함.  
               용량이 큰 GET, 부가정보를 가진 GET  
3) HEAD : GET이 무엇을 리턴하든 간에 헤더 정보만 요청함.   
               이는 Response 의 몸체 정보가 없다는 것만 삐면 GET 과 동일함.  
               요청한 URL 로부터 요청한 정보는 빼고 헤더 정보만 가져오는 것임.  
4) TRACE : 요청한 메시지의 루프백(loopvack) 테스트를 요청함.   
                 서버로부터 받은 내용을 클라이언트 쪽에서 확인하기 위해 사용.  
                 테스트 목적 또는 문제 해결을 위해 사용함  
5) **PUT**: 동봉한 몸체 정보를 요청한 URL 로 올리기 위해 사용함.  
6) **DELETE**: 요청한 URL에 있는 자원이나 파일을 삭제하기 위해 사용함.  
7) OPTION : 요청한 URL이 응답할 수 있는 HTTP 메소드가 무엇인지 요청함.  
8) CONNECT : 터널링의 목적으로 연결을 요청함.

상태코드

|  |  |
| --- | --- |
| 상태코드 | 내용 |
| 1xx (조건부 응답) | 이 상태의 상태 코드는 상태-라인과 선택적 헤더(컴퓨터에서 출력될 때 각 페이지 맨 윗부분에 자동으로 붙는 부분)만을 포함하는 임시의 응답을 나타내고 빈 라인에 의해서 종결된다. HTTP/1.0이래로 어떤 1XX 상태 코드들도 정의 되지 않았다. 서버들은 1XX 응답을 실험적인 상태를 제외하고 HTTP/1.0 클라이언트(서버에 연결된 컴퓨터)로 보내면 안 된다. |
| 2xx (성공) | 이 클래스의 상태 코드는 클라이언트가 요청한 동작을 수신하여 이해했고 승낙했으며 성공적으로 처리했음을 가리킨다. |
| 3xx (리다이렉션 완료) | 클라이언트는 요청을 마치기 위해 추가 동작을 취해야 한다. |
| 4xx (요청 오류) | 클래스의 상태 코드는 클라이언트에 오류가 있음을 나타낸다. |
| 500(내부 서버 오류) | 서버에 오류가 발생하여 요청을 수행할 수 없다. |

<http://ko.wikipedia.org/wiki/HTTP_%EC%83%81%ED%83%9C_%EC%BD%94%EB%93%9C>

웹페이지는 여러 객체로 이루어질수있다.

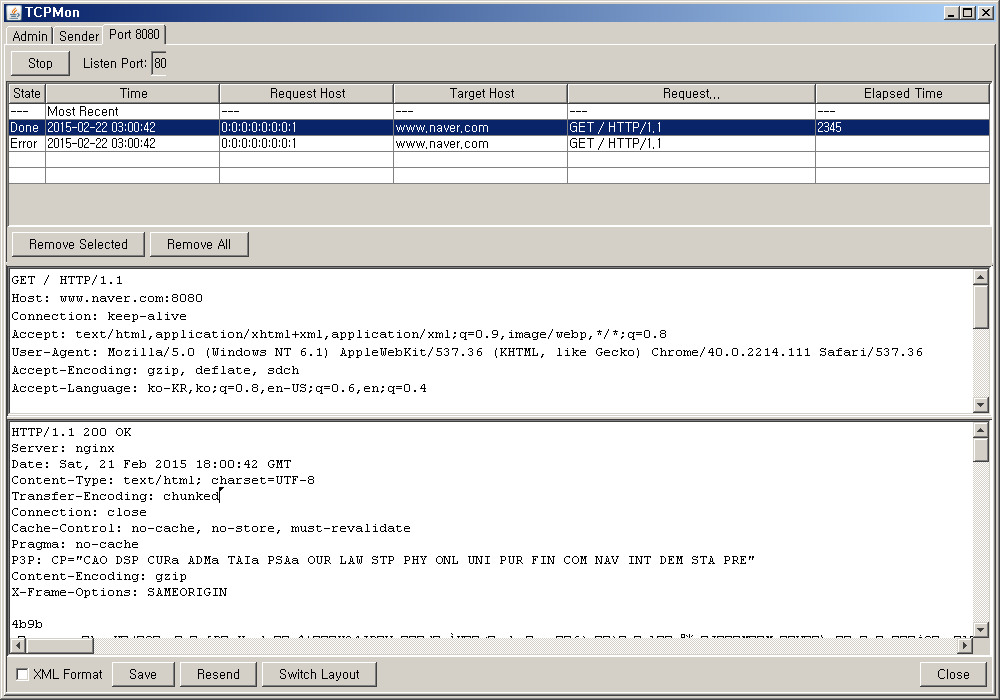
풍부한 웹페이지를 가져오기위해 HTMl 뼈대를 한번의 트랜잭션으로 가져온뒤 첨부된 이미지 그래픽조각등 가져오기위해 HTTP 트랜잭션들을 또다시 수행한다.(첨부된 리소스들에 대한 각각 별개의 HTTP의 트랜잭션이 필요)

메시지

HTTP메시지는 단순한 줄 단위의문자열이다 이진형식이 아닌 일반 텍스트이 기때문에 사람이 읽고 쓰기 쉽다.

클라이언트(HTTP요청)request → 서버(HTTP응답)respons → 클라이언트

TCMMon으로 확인해보자 ㅋㅋ <https://ws.apache.org/tcpmon/>



|  |  |
| --- | --- |
| 요청 | 응답 |
| GET / HTTP/1.1  Host: www.naver.com:8080  Connection: keep-alive  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.111 Safari/537.36  Accept-Encoding: gzip, deflate, sdch  Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4 | HTTP/1.1 200 OK  Server: nginx  Date: Sat, 21 Feb 2015 18:00:42 GMT  Content-Type: text/html; charset=UTF-8  Transfer-Encoding: chunked  Connection: close  Cache-Control: no-cache, no-store, must-revalidate  Pragma: no-cache  P3P: CP="CAO DSP CURa ADMa TAIa PSAa OUR LAW STP PHY ONL UNI PUR FIN COM NAV INT DEM STA PRE"  Content-Encoding: gzip  X-Frame-Options: SAMEORIGIN  ..data.. |
| 1. 시작줄: 무엇을해야하는지  2~\* 헤더: 0개이상의 헤더 필드가 이어진다. 구문분석으루이해 콜론으로구분  이름:값...  \*~본문 : 데이터.  끝맺음 : 캐리지리턴 | 1. 시작줄: 무슨일이 있어났는지  2~\* 헤더: 0개이상의 헤더 필드가 이어진다. 구문분석으루이해 콜론으로구분  이름:값...  \*~본문 : 데이터.  끝맺음 : 캐리지리턴 |

TCP커넥션

|  |  |
| --- | --- |
| http://word.tta.or.kr/image/terms/imgv2/OS_I.jpg | http://cfile26.uf.tistory.com/image/186F8B0C4BB1728080068E |

(OSI 는 Open Systems Interconnection Reference Model 의 약자로써 국제 표준화 기구(ISO) 에서 개발한 모델)

일단 TCP커넥션이 맺어지면 서버 컴퓨터간에 교환되는 메시지가 없어지거나 손상되거나 순서가 뒤바뀌어 수신되는을은 결코 없을것이다.

1. 웹브라우저는 서버의 URL에서 호스트명을추출한다

2. 웹브라우저는 서버의 호스트명을 IP변환한다

3. 웹브라우저는 URL에서 포트번호(있다면) 추출한다

4. 웹브라우저는 웹 서버와 커넥션을 맺는다

5. 웹브라우저는 서버에 HTTP 요청을보낸다

6. 서버는 웹브라우저에 HTTP응답을 돌려준다

7. 커넥션이 닫히면 웹브라우저는 문서를보여준다(클라이언트)

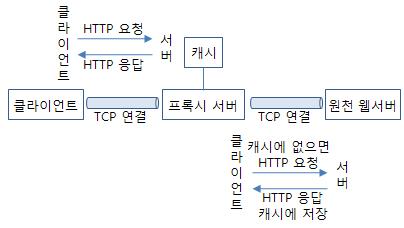
프로토콜 버전

|  |  |
| --- | --- |
| 버전 | 내용 |
| HTTP/0.9 | 1991년 프로토타입, 심각한결함있음, GET메서드만 지원  MIME타입이나 HTTP헤더등을 지원하지않았다. |
| HTTP/1.0 | 웹페이지와 상호작용하는 폼을 실현했음 |
| HTTP/1.0+ | keep-alive 커넥션 가상 호스팅 지원, 프락시연결 지원포함 많은 기능 지원 |
| HTTP/1.1 | 기존프로토컬의 나쁜점 제거등 성능향상이 되었다.(현재 버전) |
| HTTP/2.0 | 구글의 SPDY프로토컬을 기반으로 설계가 진행중인 프로토콜이다. |

웹의구성요소

1. 프락시는 클라이언트와 서버 사이에 위치하여 클라이언트의 모든 HTTP요청을 받아 서버에게 전달한다(대개 요청을 수정한 뒤어) 이 애플리케이션은 사용자를 위한 프락스로 동작하며 사용자 대신해서 서버에 접근한다.

2. 캐시는 프락시는 자신을 거쳐 가는 문서들 중 자주 찾는 것의 사본을 저장해두는 특별한 종류의 HTTP프락시 서버이다.

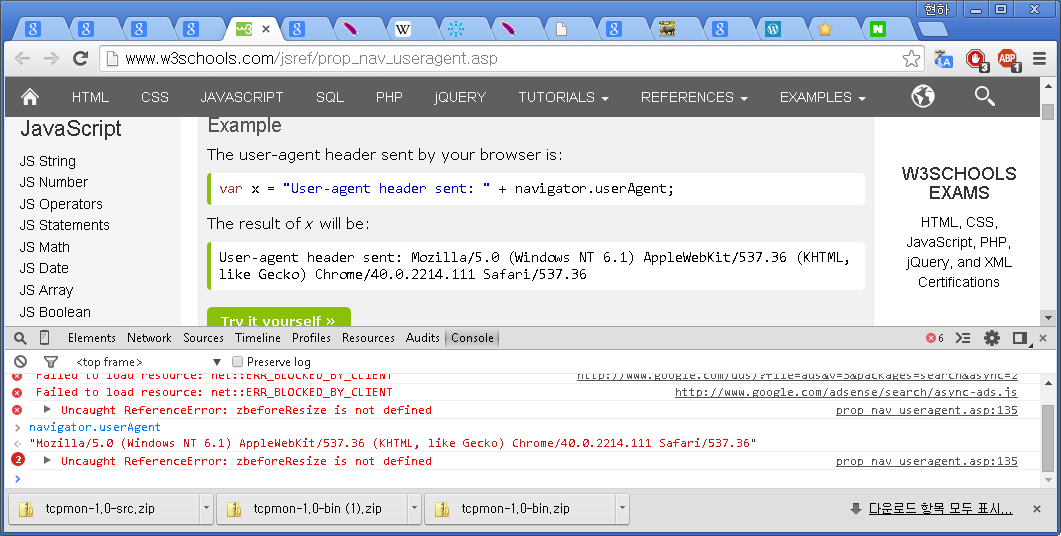


3. 게이트웨이는 다른 서버들의 중개자로 동작하는 특별한 서버이다 게이트웨이는 주로 HTTP트래픽을 다른 프로토콜로 변환하기 위해서 사용된다. 게이트웨이는 언제나 스스로가 리소스를 가지고 있는 진짜 서버인것처럼 요청을 다룬다.

4. 터널은 두커넥션 사이에서 날(raw) 데이터를 열어보지 않고 그대로 전달해주는 HTTP 애플리케이션이다 HTTP터널은 주로 HTTP데이터를 하나 이상 HTTP연결을 통해 그대로 전송해주기 위해 사용한다.HTTPS(SSL)사용

5. 에이전트은 사용자를 위해 HTTP요청을 만들어주는 클라이언트 프로그램이다. 웹요청을 만드는 애플리케이션은 뭐든 HTTP에이전트다.

예를들어 사람의 통제 없이 스스로 웹을 돌아다니며 HTTP 트랜잭션을 일으키고 콘텐츠를 받아오는 자동화된 사용자 에이전트도 있다(구글봇, 크롤링봇 등등)



URL과 리소스 2장

인터넷의 리소스 탐색하기

|  |  |  |
| --- | --- | --- |
| http:// | [www.joes.com](http://www.joes.com) | /homeimg.jpg |
| ftp:// | [www.joes.com](http://www.joes.com) | /homeimg.jpg |
| rtsp:// | [www.joes.com](http://www.joes.com) | /homeimg.jpg |
| 스킴:// | 서버위치 | 경로 |
| 스킴(어떻게)  http프로토컬 | 호스트(어디에) 서버위치  죠의 서버이동 | 리소스위치 경로 |

URL문법 p31

<스킴>://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>;<파라미터>?<질의>#<프래그먼트>

|  |  |  |
| --- | --- | --- |
| 컴포넌트 | 설명 | 기본값 |
| 스킴 | 리소스를 가져오면 어떤 프로토콜을 사용하여 서버에 접근해야 하는지 가리킨다 | 없음 |
| 사용자이름 | 몇몇 스킴은 리소스에 접근하기 위해 필요하다 | anonymous |
| 비밀번호 | 사용자의 비밀번호를 가르키며, 사용자 이름에 콜론: 으로 이어 기술한다 | <이메일주소> |
| 호스트 | 리소스를 호스팅하는 서버의 호스트명이나 IP | 없음 |
| 포트 | 리소스를 호스팅하는 서버가 열어놓은 포트번호 많은 스킴이 기본 포트를 가지고 있다 | 스킴에 따라다름  HTTP(80), HTTPS(443), FTP(21).. |
| 경로 | 이전 컴포턴트와 / 로 구분되어 있으며 서버 내 리소스가 서버 어디에 있는지 가르킨다. | 없음 |
| 파라미터 | 특정 스킴들에서 입력파라미터를 기술하는 용도로 사용한다. | 없음 |
| 질의 | 애플리케이션에 파라미터를 전달한대 쓰인다 | 없음 |
| 프래그먼트 | 리소스의 조각이나 일부분을 가르키는 이름이다. | 없음 |

인코딩 체계

안전한 문자 집합을 이용하는 경우 그 표현의 한계를 넘기위해 URL에 있는 안전하지 않는 문자들을 표현할 수 있느 인코딩 방식을 고안되었다. 문자를 퍼센티지기호(%)로 시작해, ASCII코드로 표현되는 두개의 16진수 숫자로 이루어진 이스케이프 문자로 바꾼다

|  |  |  |
| --- | --- | --- |
| 문자 | ASCII코드 | URL예 |
| ~ | 126(0x7E) | <http://www.gg.com/%7Ejeo.html> |
| 빈문자 | 32(0x20) | <http://www.gg.com/100%20jeo.html> |
| % | 37(0x25) | <http://www.gg.com/gg1004%25jeo.html> |

문자제한 p42

URL상에 허용되는 문자는 원칙상 [a-z1-9-\_.]뿐입니다. (규격이 그렇습니다)

그 외의 문자를 사용한다면 %xx로 인코딩해줘야 합니다. (역시 규격이 그렇습니다)

스킴의바다 p44

HTTP 메시지 3장

메시지의 흐름

HTTP 메시지는 HTTP 애플리케이션 간에 주고받은 데이터의 블록들이다.

클라이언트,서버,프락시 사이를 흐르는 데이터들은 인바운드,아웃바운드 업스트림,다운스크팀메시지의 방향을가진다.(인바운드:서버방향, 아웃바운드:에이전트방향)

서버

클라이언트

프락시1

프락시2

프락시3

인바운드(서버 방향), 요청(다운스트림흐름)

아웃바운드(에이젠트 방향), 응답(다운스트림흐름)

1. 원 서버로 인바운드 이동하고 클라이언트로 아웃바운드로 복귀하는 메시지

2. 모든 메시지는 다운스트림으로 흐른다.

메시지의 각부분

메시지는 시작줄, 헤더 블록, 본문 이렇게 3부분으로 이루어진다. 시작줄은 이것이 어떤 메시지인지 소술하며, 헤더 블록은 속성을, 본문은 데이터를 담고있다.(본문은 없을수도있다)

시작줄과 헤더는 그냥 줄 단위로 분리된 아스키 문자열이다 각줄은 캐리지 리턴(ASCII 13)과 개행문자(ASCII 10)으로 구성된 두 글자의 줄바꿈문자열으로 끝난다 CRLF

하지만 오래되거나 잘못 만들어진 HTTP애플리케이션들 중에서는 캐리지 리턴과 개행문자를 모두 항상 전송하지 않는것들도 있다.

헤더에 많은정보가 포함되어있는대 Content-Type줄은 본문이 무엇인지 말해준다(MMIE)

Content-Length줄은 본문의 크기를 말해준다

p52 ~ p55

|  |  |
| --- | --- |
| 요청 | 응답 |
| GET /board/humor/list.html?table=pds HTTP/1.1  Host: web.humoruniv.com:8080  Connection: keep-alive  Cache-Control: max-age=0  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.111 Safari/537.36  Accept-Encoding: gzip, deflate, sdch  Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4 | HTTP/1.1 200 OK  Date: Sat, 21 Feb 2015 19:30:49 GMT  Server: Apache  Keep-Alive: timeout=1, max=10000  Connection: Keep-Alive  Transfer-Encoding: chunked  Content-Type: text/html  data.. |
| <메서드> <요청URL> <버전>  <헤더>..  <엔터티 본문> | <버전> <상태코드> <사유 구절>  <헤더>  <엔터티 본문> |

**버전번호**

HTTP/x.y형식으로 요청과 응답 메시지 양쪽 모두에 기술된다 이것은 HTTP애플리케이션들이 자신이 따르는 프로토콜의 버전을 상대방에게 말해주기 위한 수단이된다.

예를들어 HTTP/1.0 애플리케이션이 버전번호가 HTTP/1.1 으로 응답을 받았을때 메시지라고 해석하는 경우가 있기때문이다.

응답의 프로토콜 버전이 HTTP/1.1 이라는 것은 사실 응답을 보낸 애플리케이션이 HTTP/1.1 이해할수 있다는것을 말한다.

**헤더를 여러줄로 나누기**

긴 헤더줄은 그들을 여러 줄로 쪼개서 더 읽기 좋게 만들수 있는데, 추가 줄앞에 는 최소 하나의 스페이스 혹은 탭문자가 와야한다.

GET /board/humor/list.html?table=pds HTTP/1.1

Host: web.humoruniv.com:8080

Connection: keep

-alive

메서드들..p60~67

상태코드..p67~76

헤더 76~84

일반헤더 : 클라이언트,서버 그리고 어딘가에 메시지를 보내는 애플리케이션들의 위해 다양한 목적으로사용

요청헤더 : 요청메시지를 위한 헤더

응답헤더 : 클라이언트에게 정보를 제공하기 위한 자신만의 헤더

엔터티 헤더 : 본문에 대한 헤더 말한다.

확장 헤더 : 개발자들에 의해 만들어졌지만 아직 승인안된것

HTTP 요청과 HTTP 응답은 헤더를 사용하여 HTTP 메시지 정보를 보냅니다. 헤더는 이름 뒤에 콜론, 공백, 값이 순서대로 나오는 일련의 줄입니다. 필드는 순서에 상관 없이 정렬할 수 있습니다. 일부 헤더 필드는 요청 헤더와 응답 헤더 모두에 사용되지만 그 외의 헤더 필드는 요청이나 응답 헤더 중 한 헤더에만 사용할 수 있습니다.

대부분 요청 헤더 필드를 사용하여 클라이언트는 허용되는 여러 옵션을 값 부분에 지정할 수 있으며 각 옵션의 기본 설정 순위를 정할 수도 있습니다. 여러 항목은 쉼표로 구분됩니다. 예를 들어, 클라이언트는 다음 압축 형식이 적용됨을 나타내는 "Content-Encoding: gzip, compress"가 포함된 요청 헤더를 보낼 수 있습니다. 서버에서 응답 본문에 gzip 인코딩을 사용하면 해당 응답 헤더에 "Content-Encoding: gzip"가 포함됩니다.

일부 필드는 단일 헤더에 두 번 이상 발생할 수 있습니다. 예를 들어, 헤더에 "Warning" 필드가 있을 수 있습니다.

다음은 HTTP 1.1 헤더 필드를 나열한 표입니다. 일부 헤더 필드는 MIME 필드입니다. MIME 필드는 IETF(Internet Engineering Task Force) 문서 RFC 2045에서 정의되지만 HTTP 1.1 프로토콜에도 사용됩니다. MIME 및 HTTP 1.1 사양에 대한 자세한 내용은 IETF 페이지를 참조하십시오.

## 일반 헤더 필드

일반 헤더 필드는 요청 메시지와 응답 메시지에 사용할 수 있습니다.

|  |  |
| --- | --- |
| **이름** | **예제 값** |
| Cache-Control | "max-age=10" |
| Connection | "close" |
| Date | "Tue, 11 Jul 2000 18:23:51 GMT" |
| Pragma | "no-cache" |
| Trailer | "Date" |
| Transfer-Encoding | "chunked" |
| Upgrade | "SHTTP/1.3" |
| Via | "HTTP/1.1 Proxy1, HTTP/1.1 Proxy2" |
| Warning | "112 Disconnected Operation" |

## 요청 헤더 필드

요청 헤더 필드는 요청 메시지에만 사용됩니다.

|  |  |
| --- | --- |
| **이름** | **예제 값** |
| Accept | "text/html, image/\*" |
| Accept-Charset | "iso8859-5" |
| Accept-Encoding | "gzip, compress" |
| Accept-Language | "en, fr" |
| Authorization | [credentials] |
| Content-Encoding | "gzip" |
| Expect | "100-continue" |
| From | "user@microsoft.com" |
| Host | "www.microsoft.com" |
| If-Match | "entity\_tag001" |
| If-Modified-Since | "Tue, 11 Jul 2000 18:23:51 GMT" |
| If-None-Match | "entity\_tag001" |
| If-Range | "entity\_tag001" or "Tue, 11 Jul 2000 18:23:51 GMT" |
| If-Unmodified-Since | "Tue, 11 Jul 2000 18:23:51 GMT" |
| Max-Forwards | "3" |
| Proxy-Authorization | [credentials] |
| Range | "bytes=100-599" |
| Referer | "http://www.microsoft.com/resources.asp" |
| TE | "trailers" |
| User-Agent | "Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)" |

## 응답 헤더 필드

응답 헤더 필드는 응답 메시지에만 사용됩니다.

|  |  |
| --- | --- |
| **이름** | **예제 값** |
| Accept-Ranges | "none" |
| Age | "2147483648(2^31)" |
| ETag | "b38b9-17dd-367c5dcd" |
| Last-Modified | "Tue, 11 Jul 2000 18:23:51 GMT" |
| Location | "http://localhost/redirecttarget.asp" |
| Proxy-Authenticate | [challenge] |
| Retry-After | "Tue, 11 Jul 2000 18:23:51 GMT" or "60" |
| Server | "Microsoft-IIS/5.0" |
| Vary | "Date" |
| WWW-Authenticate | [challenge] |

## 엔터티 헤더 필드

엔터티 헤더 필드는 요청 메시지나 응답 메시지에 사용할 수 있습니다. 엔터티 헤더 필드에는 사용된 인코딩 형식 등의 메시지의 엔터티 본문에 대한 정보가 있습니다.

|  |  |
| --- | --- |
| **이름** | **예제 값** |
| Allow | "GET, HEAD" |
| Content-Encoding | "gzip" |
| Content-Language | "en" |
| Content-Length | "8445" |
| Content-Location | "http://localhost/page.asp" |
| Content-MD5 | [md5-digest] |
| Content-Range | "bytes 2543-4532/7898" |
| Content-Type | "text/html" |
| Expires | "Tue, 11 Jul 2000 18:23:51 GMT" |
| Last-Modified | "Tue, 11 Jul 2000 18:23:51 GMT" |

## 요청 헤더 예제

다음은 HTTP 요청의 기본 예제입니다.

GET /articles/news/today.asp HTTP/1.1

Accept: \*/\*

Accept-Language: en-us

Connection: Keep-Alive

Host: localhost

Referer: http://localhost/links.asp

User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

Accept-Encoding: gzip, deflate

이 요청에는 메서드(GET), 리소스 경로(/articles/news/today.asp) 및 HTTP 버전(HTTP/1.1)을 포함하는 요청 표시줄이 있습니다. 이 요청에는 본문이 없으므로 요청 표시줄 다음에 나오는 모든 것은 모두 헤더의 일부입니다. 헤더 다음에는 빈 줄이 오며 헤더의 끝을 표시합니다.

## 응답 헤더 예제

웹 서버는 여러 방법으로 이전 요청에 응답할 수 있습니다. 파일을 액세스할 수 있고 사용자가 해당 파일을 볼 수 있는 권한이 있으면 응답은 다음과 유사합니다.

HTTP/1.1 200 OK

Server: Microsoft-IIS/5.0

Date: Thu, 13 Jul 2000 05:46:53 GMT

Content-Length: 2291

Content-Type: text/html

Set-Cookie: ASPSESSIONIDQQGGGNCG=LKLDFFKCINFLDMFHCBCBMFLJ; path=/

Cache-control: private

<HTML>

<BODY>

...

응답의 첫째 줄은 상태 표시줄입니다. 이 상태 표시줄에는 응답에 사용된 HTTP 버전, 상태 코드(200) 및 원인이 있습니다. 이 예제에는 다섯 개 필드로 이루어진 헤더가 있으며, 이 헤더 뒤에 빈 줄(캐리지 리턴 및 줄 바꿈)과 응답 본문의 첫 두 줄이 차례로 옵니다.

#### 참고 항목

위키피디아 참조

http://en.wikipedia.org/wiki/List\_of\_HTTP\_header\_fields

커넥션 관리 4장

TCP커넥션

HTTP통신은 지구상의 컴퓨터와 네트워크장비에 널리 쓰이고 있는 패킷 교환 네트워크 프로토콜의 계층화된 집합인 TCP/IP를 통해 이루어진다.

일단 커넥션이 맺어지면 클라이언트와 서버 컴퓨터 간에 주고 받는 메시지들은 손실 혹은 손상되거나 순서가 바뀌지 않고 안전하게 전달된다.

TCP 스트림은 세그먼트로 나뉘어 IP패킷을 통해 전송된다.

IP,TCP,HTTP로 구성된 프로토콜 스택에서 최상위 계측이다. HTTP에 보안 기능을 더한 HTTPS는 TLS혹은 SSL이라고 불리기도 하며 HTTP와 TCP사이에 있는 암호화 계측이다.

TCP커넥션 유지하기

TCP커넥션은 4가지 값으로 식별한다

<발신IP, 발신PORT, 수신IP, 수신PORT>