

컴퓨터 네트워크 프로젝트 #1 보고서

소프트웨어학부
2015038440 이승현

1) 서버 구성

- 개요 : 서버에 파일 요청을 보내면 해당 파일을 브라우저에 띄워주고 다음 요청을 기다린다.
- 개발 환경 : 우분투 16.04
- 테스트 환경 : 구글 크롬

- main 함수 : 실행 파일 인자로 포트 번호를 받는다. 해당 포트 번호로 소켓(socket)을 바인딩한다. 즉 서버를 생성한다. Accept 함수로 request 를 기다린다. request 가 들어와 accept 가 완료되면 새로운 소켓(newsocket)에 그 값을 저장한다. Newsocket에 저장된 request message 를 파싱해서 request method, file name, content type 을 저장한다. 이 때 content type 은 따로 만들어둔 content type 함수를 이용한다. 추가로 파일의 용량은 따로 만들어둔 file size 함수를 이용한다.

이 정보를 이용해서 response message header line 을 작성한다. Header line 의 구성은 다음과 같다.

```
"HTTP/1.1 200 OK
Server : LSH Server
Content-Type: "content-type"
Content-Length: "content-length"
```

“

Header line 을 작성해서 보낸 뒤에는 read 와 write 를 반복적으로 사용해 파일의 데이터를 버퍼에 담아서 보낸다. 한 번의 요청에 대한 응답이 끝나면 사용했던 버퍼들과 파일을 초기화하고 다음 요청을 기다린다. 프로그램이 종료되기 전에 사용했던 소켓들을 모두 닫는다.

- content type 함수 : 요청받은 파일의 이름을 인자로 받는다. Strtok 함수를 이용해 파일의 확장자를 알아내고 확장자 값에 따라 해당 MIME 타입을 리턴한다.

- file size 함수 : main 함수에서 open 한 파일을 인자로 받는다. fseek 를 이용해 파일의 용량을 알아내고 리턴한다.

2) 어려웠던 점

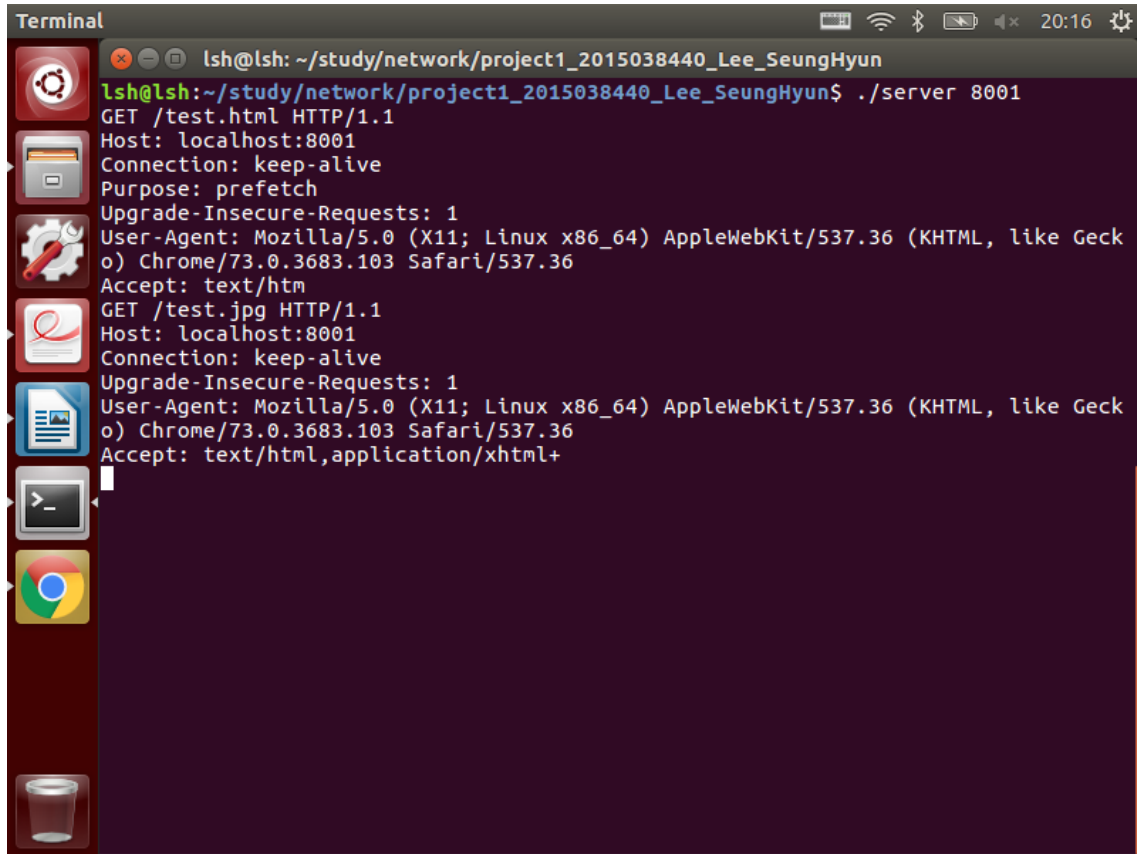
- 처음 프로젝트를 시작했을 때 뼈대 코드를 이해하는 것부터 시간이 많이 소모되었다. 또한 코드에 필요한 함수들에 대한 이해도가 부족했었다. 때문에 함수를 잘못 사용하는 일이 빈번했다. 이 문제점은 복습과 검색을 통해서 해결할 수 있었다. 결국 모든 문제는 이 첫 문제에서 비롯된 것이다.

두 번째 문제는 html 파일을 전송하는 것까지는 성공했지만 바이너리 파일을 전송하는 데 실패한 것이다. 파일 확장자에 대한 고려를 하지 않았고 버퍼를 반복적으로 사용하는 데 오류가 있었다. 콘텐츠 타입을 알아내는 함수와 버퍼를 확실하게 비워줌으로써 해결했다.

세 번째 문제는 사용했던 버퍼나 파일, 변수들을 확실하게 비워주지 않아서 발생했다. 한 번의 요청에 대한 응답을 마치고 나서 그 다음 요청을 보내면 오류가 나는 것이었다. 이는 사용했던 버퍼, 파일들을 제대로 초기화하지 않았기에 발생한 문제였다. 코드를 처음부터 끝까지 계속 읽어보면서 찾아낼 수 있었다.

종합하자면 네트워크 구성 뿐만 아니라 c 언어 함수에 대한 기초적인 지식이 없었기에 어려웠었던 프로젝트였다. 다른 학생들과 많은 토의를 통해서 내가 놓친 부분을 보완할 수 있었다.

3) 결과



```
Terminal
lsh@lsh: ~/study/network/project1_2015038440_Lee_SeungHyun
lsh@lsh:~/study/network/project1_2015038440_Lee_SeungHyun$ ./server 8001
GET /test.html HTTP/1.1
Host: localhost:8001
Connection: keep-alive
Purpose: prefetch
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
GET /test.jpg HTTP/1.1
Host: localhost:8001
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
```

그림 1) 브라우저에 request 시 console 에 HTTP request 메시지 출력

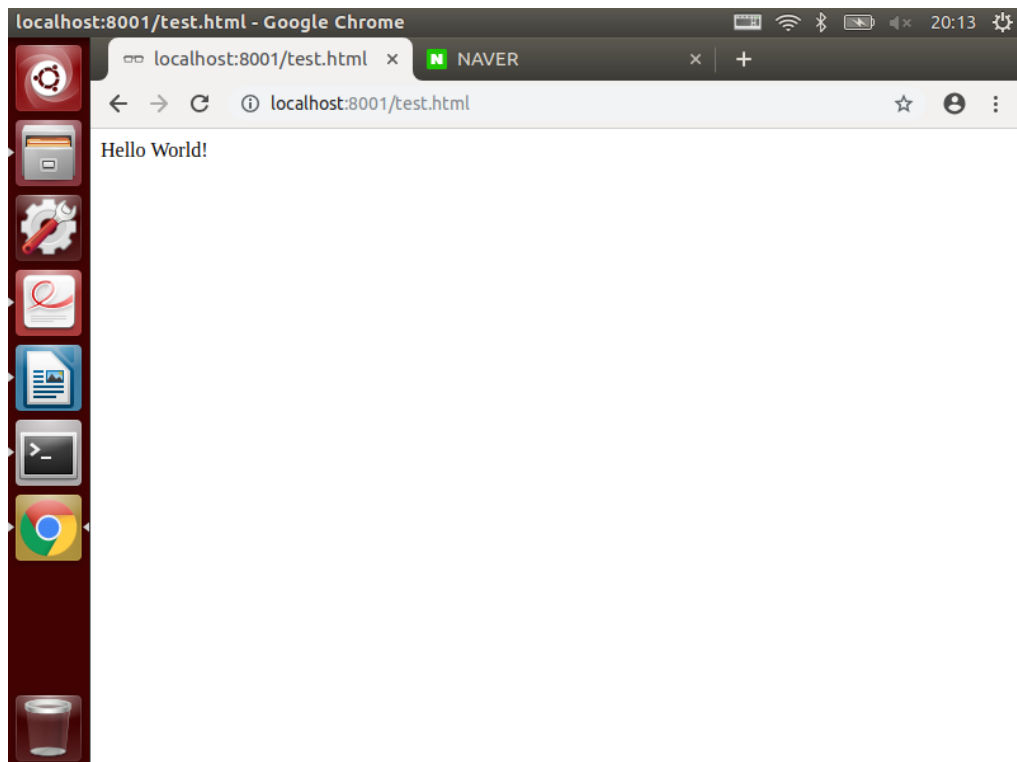


그림 2) test.html 파일 브라우저 출력

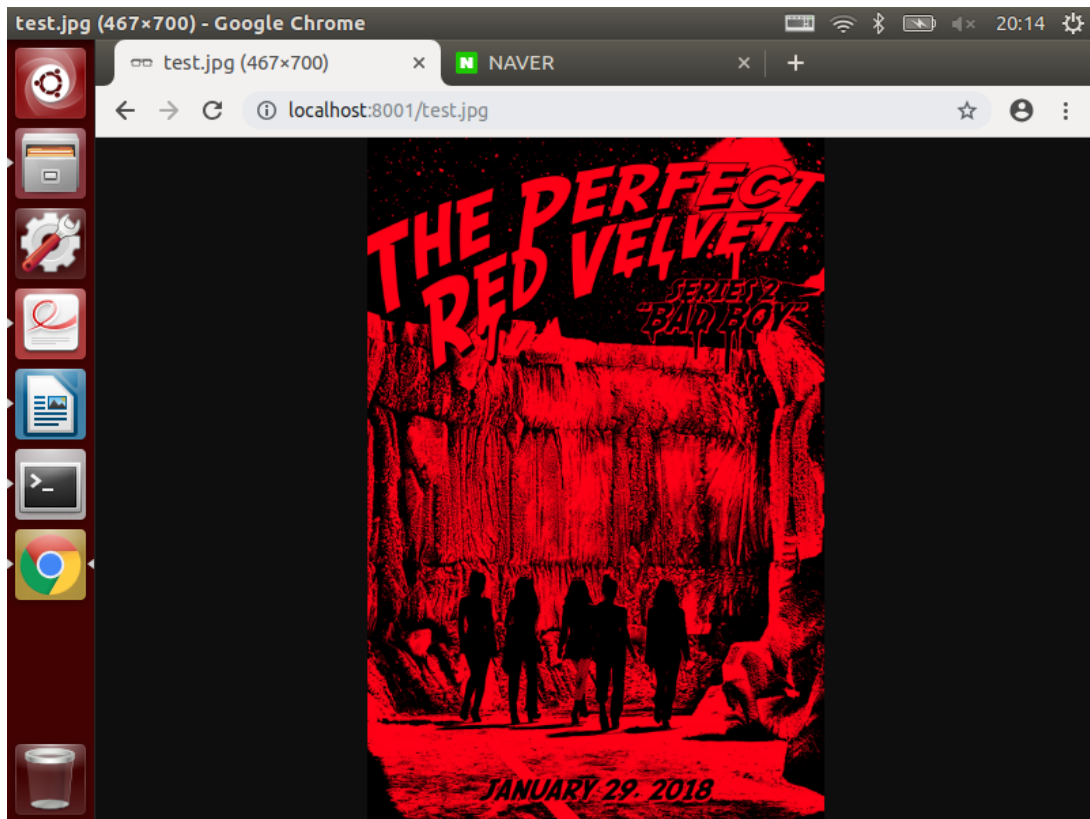


그림 3) test.jpg 파일 브라우저 출력

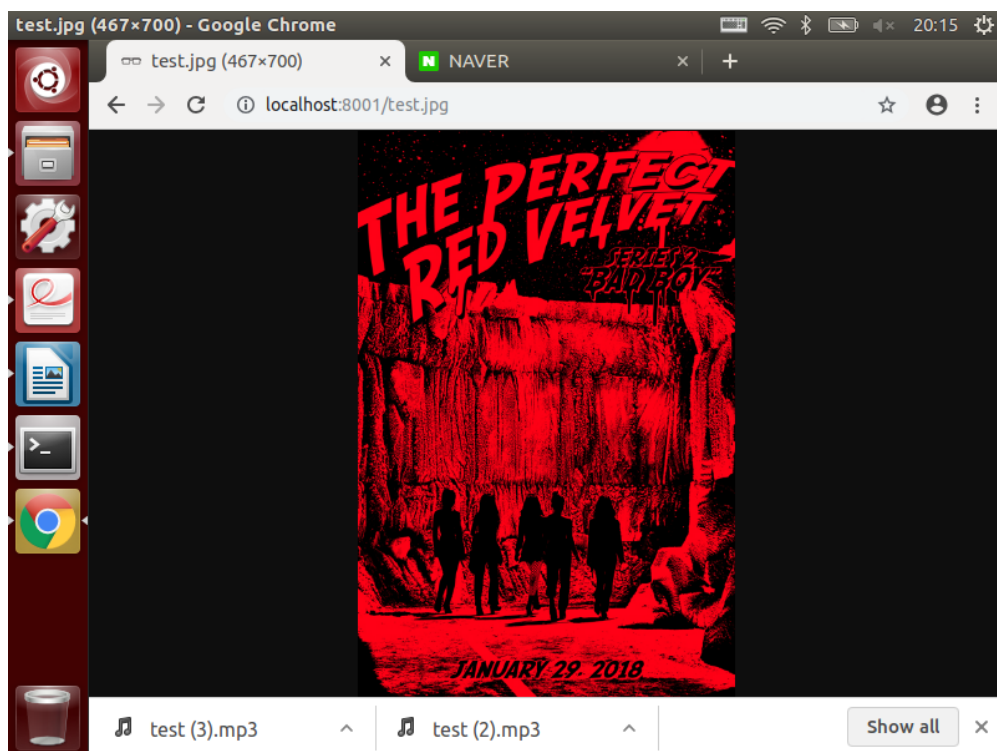


그림 4) test.mp3 파일 요청 시 다운로드