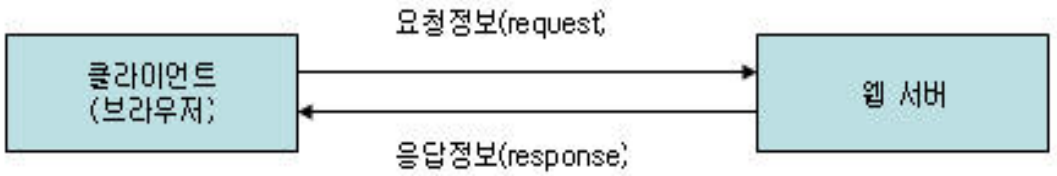
목차

4강
TCP 소켓 프로그래밍 III

Simple HTTP Server

1. 웹서버

- 간단한 웹 서버를 직접 **TCP** 소켓 프로그래밍을 이용해서 작성
- **HTTP** 프로토콜

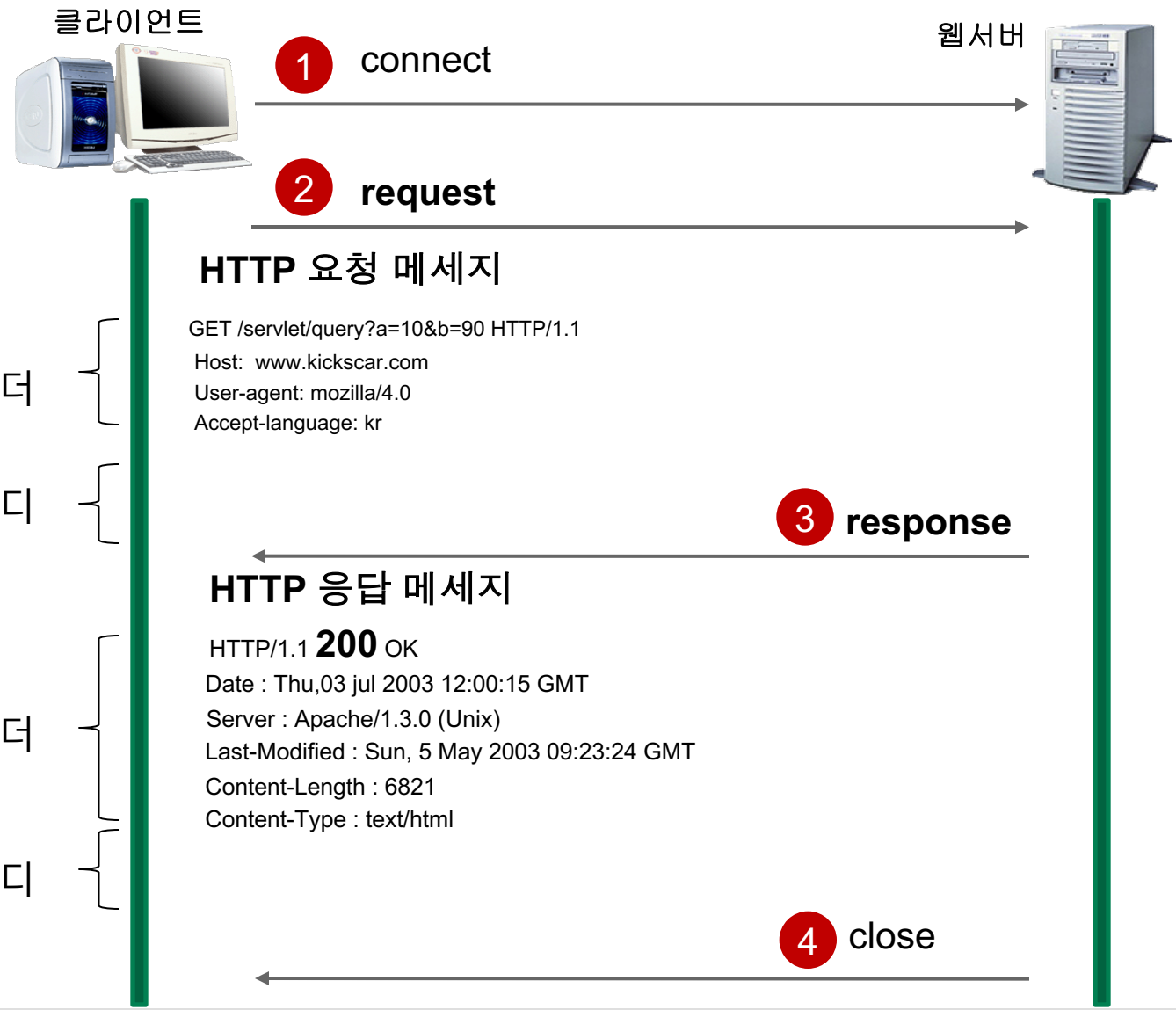


- 브라우저는 웹 서버에게 요청 정보를 보낸다.

- 웹 서버는 요청정보를 분석하여 응답정보를 브라우저에게 보낸다.

Sunnyvale

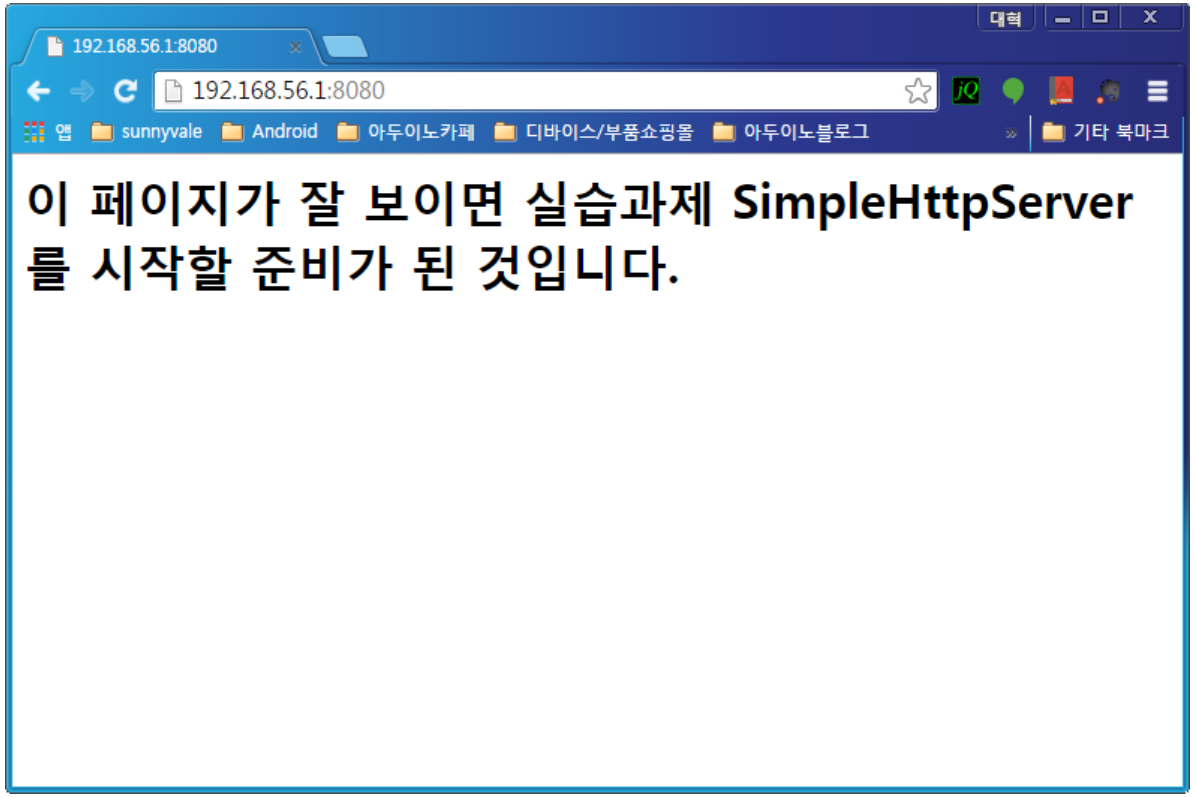
❑ HTTP (stateless)



Sunnyvale

3. 웹 서버 시작 및 테스트

- **SimpleHttpServer**는 브라우저의 요청을 받아 **RequestHandler**에 요청처리 작업을 위임한다.
- 브라우저의 모든 요청에 대한 처리는 **RequestHandler**의 **run()** 메소드가 담당한다.
- **SimpleHttpServer**를 실행한 후, 브라우저에서 **http://192.168.56.1:8080** 으로 접속하여 다음 화면을 확인 한다.



Sunnyvale

4. 요구사항1 – 요청 정보(Header + Body) 출력하기

브라우저가 보내는 요청정보 를 분석하기 위해 요청 정보만 콘솔에 출력하는 코드를 작성한다.

[HINT]

1. **InputStream**에 보조 스트림 **InputStreamReader, BufferedReader**를 사용하여 소켓의 바이트 데이터를 문자열 데이터로 그리고 라인단위로 받는다. **(O)**
2. 다음 코드를 참고해서 요청정보를 모두 출력한다.
3. 콘솔 출력은 **SimpleHttpServer.*consolLog*** 메소드를 사용한다.

```
while( true ) {

 ...

 if( line == null || "".equals( line ) ) {
     break;
 }

 ...

}
```


Sunnyvale

4. 요구사항1 – 요청 정보(Header + Body) 출력하기

[결과 화면 예시]

```
[HttpServer] connected from kickscar-PC:49348
[HttpServer] ============================== Request Information ==============================
[HttpServer] connected from kickscar-PC:49349
[HttpServer] ============================== Request Information ==============================
[HttpServer] connected from kickscar-PC:49350
[HttpServer] ============================== Request Information ==============================
[HttpServer] GET / HTTP/1.1
[HttpServer] Host: 192.168.56.1:8080
[HttpServer] Connection: keep-alive
[HttpServer] Cache-Control: max-age=0
[HttpServer] Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
[HttpServer] Upgrade-Insecure-Requests: 1
[HttpServer] User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.86 Safari/537.36
[HttpServer] Accept-Encoding: gzip, deflate, sdch
[HttpServer] Accept-Language: en-US,en;q=0.8,ko;q=0.6
[HttpServer] ================================================================================
[HttpServer] GET /favicon.ico HTTP/1.1
[HttpServer] Host: 192.168.56.1:8080
[HttpServer] Connection: keep-alive
[HttpServer] Pragma: no-cache
[HttpServer] Cache-Control: no-cache
[HttpServer] User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.86 Safari/537.36
[HttpServer] Accept: */*
[HttpServer] Referer: http://192.168.56.1:8080/
[HttpServer] Accept-Encoding: gzip, deflate, sdch
[HttpServer] Accept-Language: en-US,en;q=0.8,ko;q=0.6
[HttpServer] ================================================================================
[HttpServer] ================================================================================
[HttpServer] error:java.net.SocketException: Software caused connection abort: socket write error
```

각 자의 콘솔 로그를 보고 브라우저에서 보내는 요청 정보를 이해 한다.

Sunnyvale

5. 요구사항2 – HTML 문서 내용 응답

브라우저에서 **http://192.168.56.1:8080 /index.html** 로 접속했을 때 **webapp** 디렉토리의 **index.html** 를 읽어 클라이언트에 응답한다.

[HINT]

(1) http://192.168.56.1:8080 /index.html 로 접속했을 때 서버 콘솔 로그의 요청 헤더 정보를 확인하자

(2) 상용 웹 서버 또는 **WAS** 라면 헤더정보를 모두 파싱해서 적당한 객체에 저장하고 웹 애플리케이션에 넘겨 주거나 응답 시, 참고 정보로 사용 하겠지만 여기서는 **1**번째 라인만 사용한다.

GET **/index.html** **HTTP/1.1**

(요청명령) **(url)** **(프로토콜 버전)**

(3) 1번째 라인만 요청 명령, 요청 파일 이름 그리고 프로토콜 버전으로 분리해야 한다.

```
String[] tokens = line.split( " " );
```


Sunnyvale

5. 요구사항2 – HTML 문서 내용 응답

[HINT]

(4) 다음 메소드를 정의하여 분리된 토큰을 넘기도록 하고 메소드에서 **URL**를 처리하도록 한다.

```
private void responseStaticResource( OutputStream outputStream, String url, String protocol )
        throws IOException {
}
```

(5) 요청 **URL**에 해당하는 파일을 **webapp** 디렉토리에 읽어서 브라우저에 전달하면 된다.
Files.readAllBytes 를 사용해서 파일을 바이트단위로 읽는다.

```
File file = new File( "./webapp" + url );
Path path = file.toPath();
byte[] body = Files.readAllBytes( path );
```

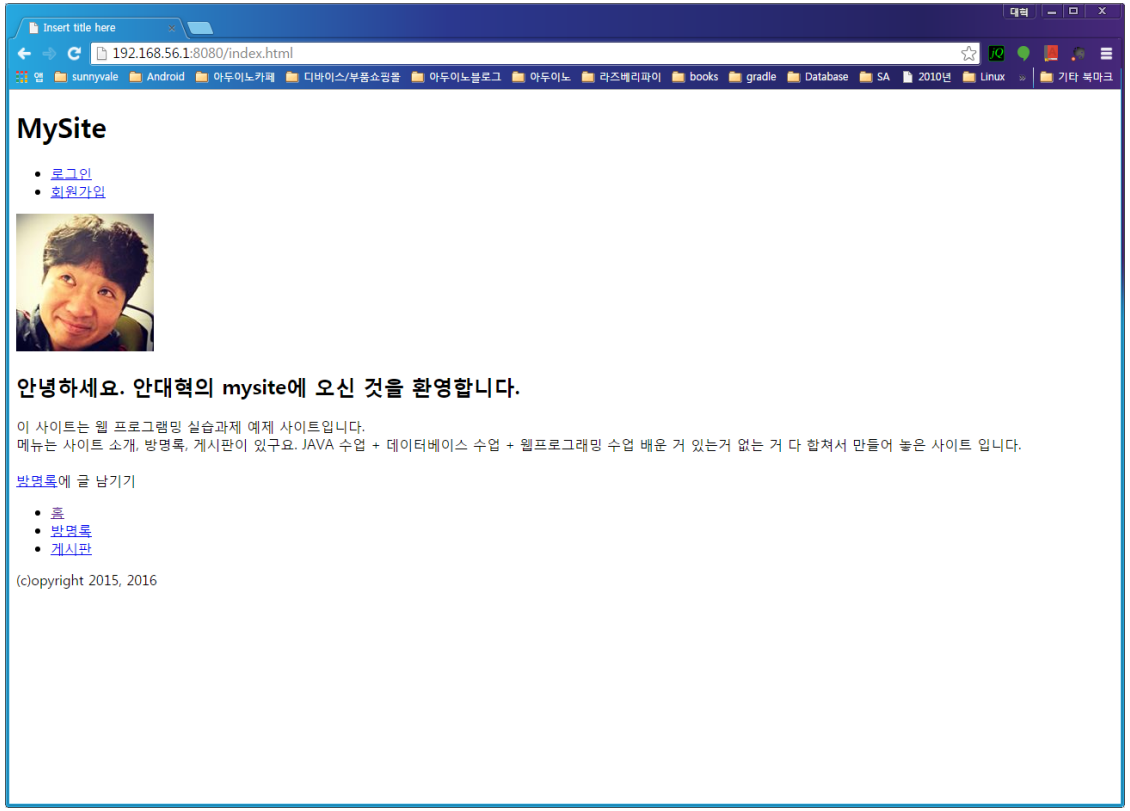
6) 응답은 예제 응답을 참고한다. **(protocol** 부분을 파라미터와 맞추어 준다 **)**

```
outputStream.write( "HTTP/1.1 200 OK\r\n".getBytes( "UTF-8" )
outputStream.write( "Content-Type:text/html\r\n".getBytes( "UTF-8" ) );
outputStream.write( "\r\n".getBytes() );
outputStream.write( body );
```


Sunnyvale

5. 요구사항2 – HTML 문서 내용 응답

[결과 화면 예시]



(1) consol log를 통해 어떤 **URL**(파일)들이 요청이 들어 왔는지 확인해 본다.

(2) HTML 파일뿐만 아니라 이미지 파일, **css** 파일, **ico** 파일 등의 요청에 정상적으로 응답하는가?

(3) 오류가 있는 **favicon.ico** 는 무엇인지 알아본다.

(4) css 파일 내용을 정상적으로 전달하는 데 왜 페이지에 **css**가 적용이 안되어 있는가?

Sunnyvale

6. 요구사항3 – MIME 타입 지정

응답하는 내용에 맞는 타입을 지정해서 브라우저가 인식할 수 있도록 해야 한다.

이를 **MIME Type**이라 한다. 응답하는 콘텐츠의 내용에 맞는 **mime** 타입을 지정한다.

[HINT]

(1) 다음 코드를 참고하면 요청 **URL**의 **MIME** 타입을 구할 수 있다.

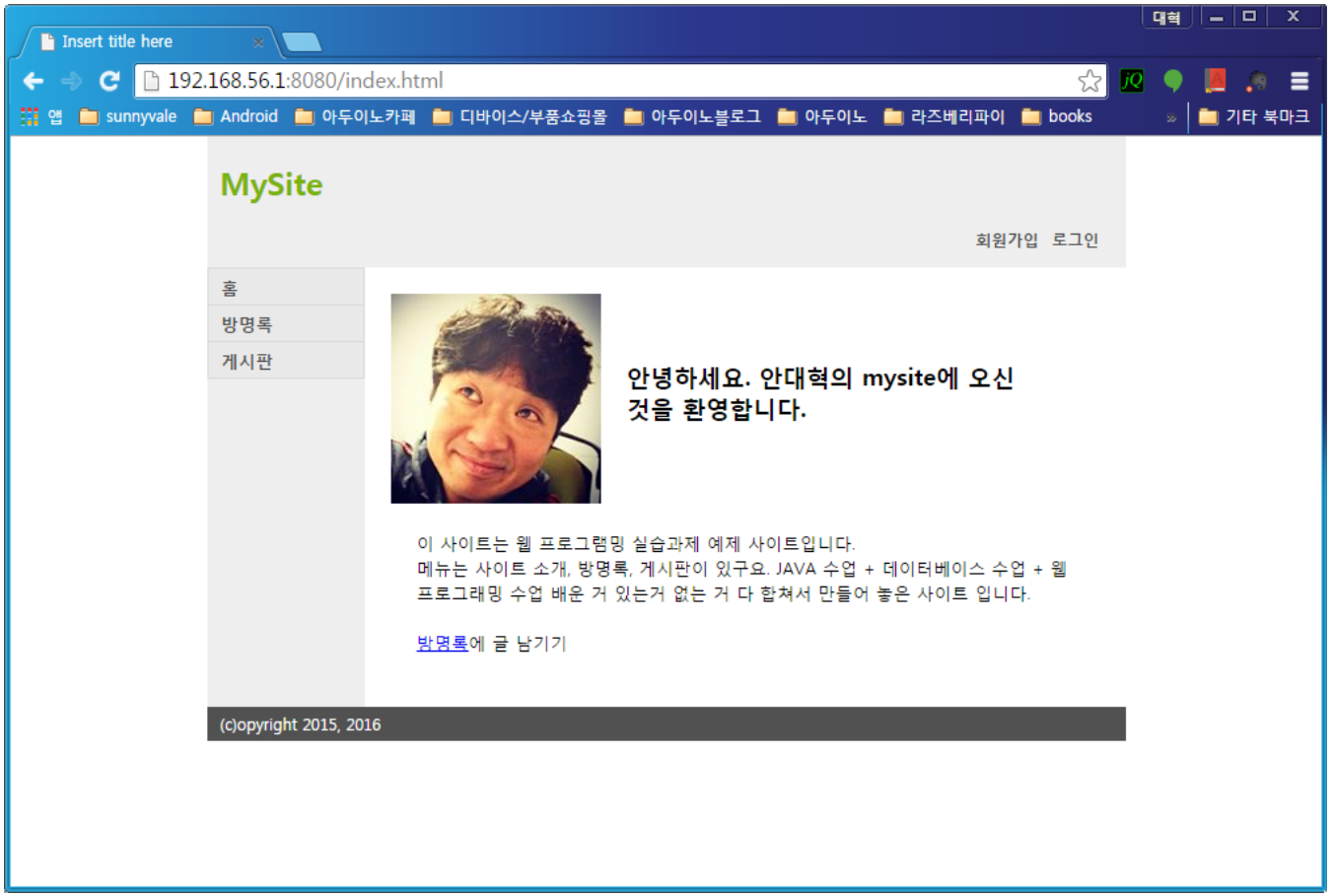
```
String mimeType = Files.probeContentType( path );
```

(2) 응답 헤더의 **Content-Type:text/html”** 에서 **text/html**을 해당 **mimeType** 으로 수정한다.

Sunnyvale

6. 요구사항3 – MIME 타입 지정

[결과 화면 예시]



css가 적용이 되어 있는 페이지가 보여야 한다.

Sunnyvale

7. 요구사항4 – 디폴트 URL 처리

왼쪽 메뉴의 홈을 눌러 보자. 페이지가 없거나 에러가 난다.

"/" URL에 대한 처리가 필요하다. URL 이 "/" 인 경우에는 "/index.html" 로 응답하도록 한다.

Sunnyvale

8. 요구사항5 – 404 에러 처리

상단의 로그인 메뉴를 눌러 보자. 로그인 페이지는 아직 준비되지 않았기 때문에 **404** 에러를 브라우저에

게 전송해야 한다. 브라우저가 요청한 페이지나 리소스가 존재하지 않을 경우, 응답코드 **404**의 에러 응답을 해주어야 한다.

[HINT]

```
 if ( file.exists() == false ) {
     response404Error( outputStream, protocol );
     return;
}
```

(2) response404Error 메소드 안의 응답은 다음과 같다. 응답 **body**는 **/webapp/error/404.html**

```
outputStream.write( ( protocol + " 404 File Not Found\r\n" ).getBytes() );
outputStream.write( "Content-Type:text/html\r\n".getBytes() );
outputStream.write( "\r\n".getBytes() );
outputStream.write( body )
```


Sunnyvale

8. 요구사항5 – 404 에러 처리

[결과화면 예시]

Sunnyvale

9. 요구사항6 – 400 에러 처리

400 에러는 **Bad Request** 즉, 잘못된 요청이다.

HTTP 명령에는 **GET, POST, PUT, DELETE** 등이 있는데, **Simple HTTP Server**는 **GET** 방식만 지원하고 나머지 요청은 **Bad Request**로 처리한다.

메인에서 회원가입 메뉴를 누르면 가입 폼이 나온다. 내용을 입력하고 회원가입 버튼을 누르면 **POST** 방식으로 데이터를 전달하게 된다. 콘솔 로그로 확인해 보자.

[HINT]

(1) tokens[0] 가 **"GET"** 인 경우만 **responseStaticResource** 를 호출 한다.

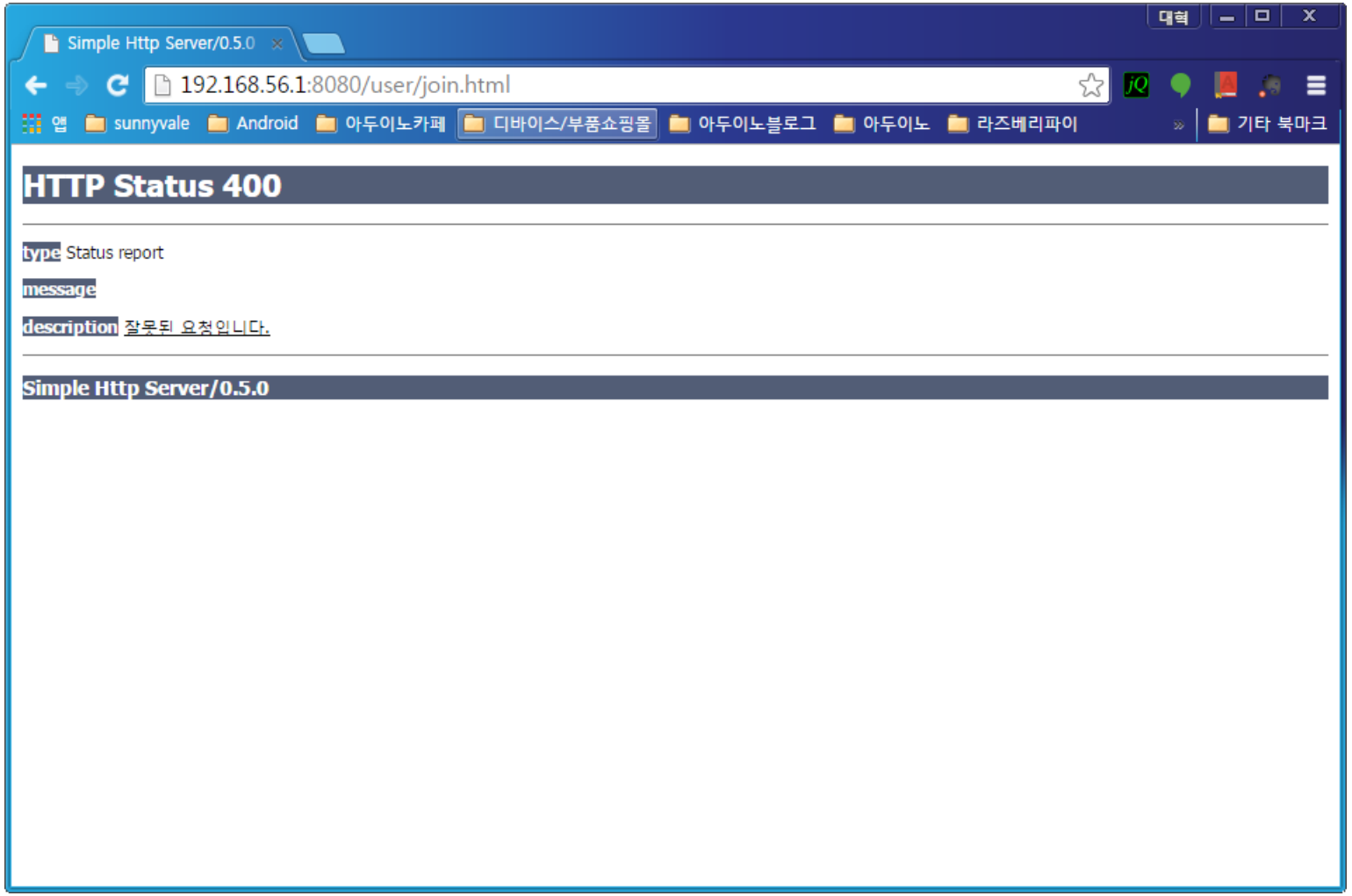
(2) 다른 명령어인 경우에는 **response400Error** 메소드를 호출한다. 메소드안의 응답내용은 다음과 같다. **Body**의 내용은 는 **/webapp/error/400.html** 내용을 읽어 보낸다.

```
outputStream.write( ( protocol + " 400 Bad Request\r\n" ).getBytes() );
outputStream.write( "Content-Type:text/html\r\n".getBytes() );
outputStream.write( "\r\n".getBytes() );
outputStream.write( body )
```


Sunnyvale

9. 요구사항6 – 400 에러 처리

[결과화면 예시]



Sunnyvale