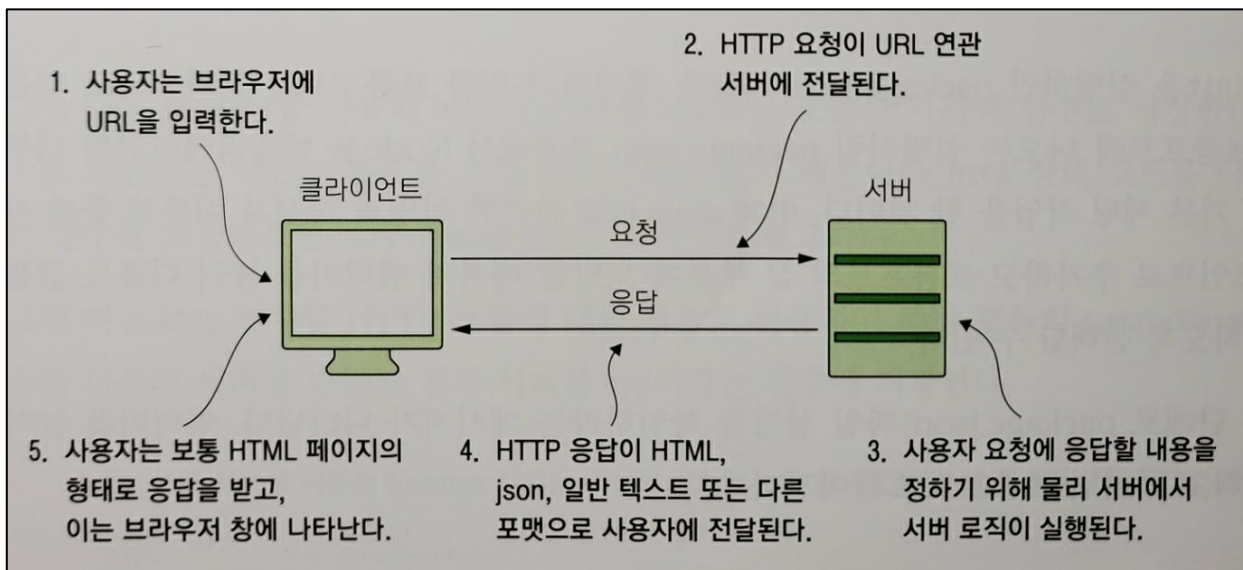


웹 서버 만들기

웹 서버란?

- 웹 서버란 데이터 읽기 및 처리를 통해 인터넷상의 요청에 대한 응답을 위해 설계된 소프트웨어이다.
- 서버와 클라이언트가 통신하는 방법 중 하나는 HTTP요청을 사용하는 것이다. 만들어진 요청이 어떤 요청인지, 예를 들어 사용자가 새로운 웹 페이지를 읽어 들이는지, 또는 지금 보고 있는 페이지의 업데이트인지 나타낸다.
- HTTP 요청 메소드는 크게 두 가지가 있으며 다음과 같다.
 - GET : 클라이언트가 서버로부터 정보를 요청.
보통 서버는 브라우저에서 볼 수 있는 콘텐츠로 응답.
 - POST : 클라이언트가 서버로 데이터를 전송.
서버는 데이터처리 후 HTML 페이지로 응답 및 다른 페이지로 이동.
- 간단한 HTTP 요청-응답 사이클



- Node.js를 설치할 때 주요 모듈 중 하나인 HTTP모듈은 자동으로 설치되어 있다.
`require("http")`로 HTTP모듈을 불러올 수 있다.

예제1) HTTP기반 웹 서버 만들기

1. 웹 서버의 루트 디렉터리 생성
: `mkdir` (루트 디렉터리 명)
2. 루트 디렉터리로 이동 후 애플리케이션 초기화
: `cd` (루트 디렉터리 명)
: `npm init`

3. HTTP 상태 코드 패키지 다운로드

: npm install http-status-codes --save

4. 루트 디렉터리에 main.js 생성

: vi main.js

5. 아래 박스의 내용과 같이 main.js에 입력 후 저장 (주석은 입력하지 않아도 무관)

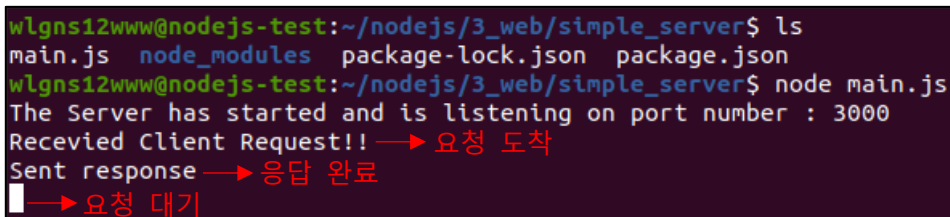
```
"use strict";

const port = 3000; // 서버 포트 번호
const http = require("http"); // http 모듈 불러오기
const httpStatus = require("http-status-codes"); // http-status-codes 모듈 불러오기
const app = http.createServer( function(request, response){ // 응답하는 콜백 함수 작성
    console.log("Received Client Request!!"); // 요청이 왔을 시 터미널에 메시지 출력
    response.writeHead(httpStatus.OK, { "Content-Type" : "text/html" }); // HTTP 응답 헤더 작성
    response.write( "<h1> Hello, Client! </h1>"); // HTTP 응답 작성 및 전송
    response.end(); // 응답 종료 및 커넥션 끊김
    console.log("Sent response"); // 응답 종료 후 터미널에 메시지 출력
}); // 콜백 함수 끝 지점

app.listen(port); // 지정된 포트 번호로 웹 서버 열기
console.log("The server has started and is listening on port number : %d", port);
```

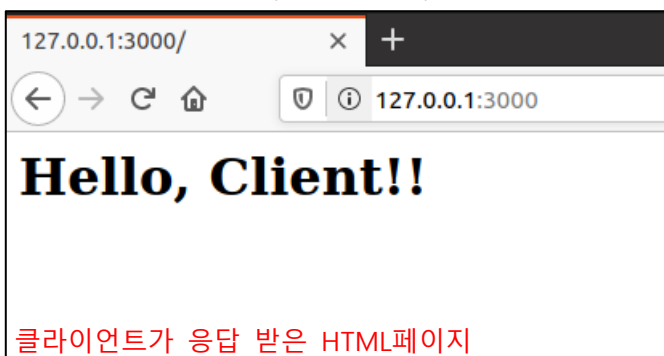
6. main.js를 node 명령으로 실행

✓ 서버 실행 화면 (터미널)



```
wlgns12www@nodejs-test:~/nodejs/3_web/simple_server$ ls
main.js  node_modules  package-lock.json  package.json
wlgns12www@nodejs-test:~/nodejs/3_web/simple_server$ node main.js
The Server has started and is listening on port number : 3000
Received Client Request!! → 요청 도착
Sent response → 응답 완료
█ → 요청 대기
```

✓ 클라이언트 실행 화면 (웹 브라우저)



참고문헌

- 조나단 웨슬러, 김성준 (2020.01.31). NODEJS로 프로그래밍 시작하기. 80-85.