Node js

리터럴 – 데이터를 표현하는 방식

Var a = 1 – 정수를 표현하는 방식

Var a = “1” – 문자를 표현하는 방식.

javascript에서 ‘내용’ + 변수 +’내용 \n’ 등의 표현은 번거롭기 때문에

template literal 을 사용한다

~칸의 ` 을 사용

`내용 ${변수}

내용‘ 으로 같은 결과를 얻는다. ${1+1}등도 가능

http – 서버와의 통신 방법, 프로토콜

opentutorials.org – 인터넷에 연결되어 있는 컴퓨터, host,domain

3000 – 포트

Main – 경로

?id=HTML&page=12 – 웹서버에 전달할 수 있는 정보 ?로 시작하며 값과 값은 &로 연결 값의 이름과 값은 =로 연결



Fs, http 같은건 모듈임 비슷한기능끼리 묶어둔 것.

Require(‘url’) - nodejs에게 url이라는 모듈을 사용할 것임을 말하는 것.

Fs = 파일 시스템

fs.readFile('/etc/passwd', (err, data) => {

if (err) throw err;

console.log(data);

});

최신기능?

fs.readFile('sample.txt', ‘utf8’ function(err, data) {

console.log(data);

});

구버전 같은 효과

이렇게 함으로서 다른 파일의 내용을 불러올 수 있게 된다.

Nodejs 파일은 수정될 때 마다 재시작 해야 하지만.

Nodejs 에서 불러온 파일이 수정된 경우 새로고침만 해도 바로 적용된다.

콘솔에서의 입력

Var args = process.argv;

args라는 배열에 콘솔에서 입력값을 문자로 받아온다.

콘솔에서 node name.js 123 이라고 입력하면

process.argv 가 ‘123’을 입력으로 받아

args[2]에 저장한다.

Response.writeHead(200) – 전송이 성공적이다.

Response.writeHead(404) – 파일을 찾을 수 없다.

Var \_url=request.url;

Url.parse(\_url,ture)는 url로부터 여러가지 정보를 가져온다.

Var pathname= Url.parse(\_url,ture).pathname;

Pathname==’/’라면 가장 최상위의 루트를 의미하며

Pathname은 / 이후의 쿼리 스트링은 포함하지 않는다.

따라서 쿼리스트링을 통해 페이지를 나누는 경우 pathname이 루트인지 아닌지에 따라 유효한 페이지인지 아닌지를 구분할 수 있다.

루트의 메인 페이지의 경우 /로만 구분되었기 때문에 하위 페이지와는 구분이 안되어있다.

따라서 queryData.id 가 undifined라면 메인페이지 임을 알 수 있으므로 하위페이지의 처리와 동일하게 조건문을 통해 구분하여 메인페이지를 구성할 수 있다.