KWEB Study Week1: Node.js Basic

KWEB 2학기 준회원 스터디

Today's Contents

1. Before Study
2. Module in Node.js
3. npm 사용하기
4. Module 만들기
5. View Template
6. 과제

M.G. BAE

J.H. BAEK

2017 Fall Semester

Before Study

- KWEB 2학기 준회원 스터디
  - 월 7시: 15 배민근 (baemingun@naver.com)
  - 화 7시: 16 백지훈 (bjh970913@gmail.com)

- 0주차인 지난주에 2학기 스터디에 사용할 Node.js 소개 및 설치를 완료하였습니다.

- 2학기에도 역시 OUT COUNT 사용하는 거 아시죠? 출석 & 과제 열심히 해주세요!
  - 파이팅 하세요 여러분! 모두 정회원에서 뵈어요 ㅎ

- 1주차의 목표는 Node.js의 Module 알아가기 입니다!

JSON

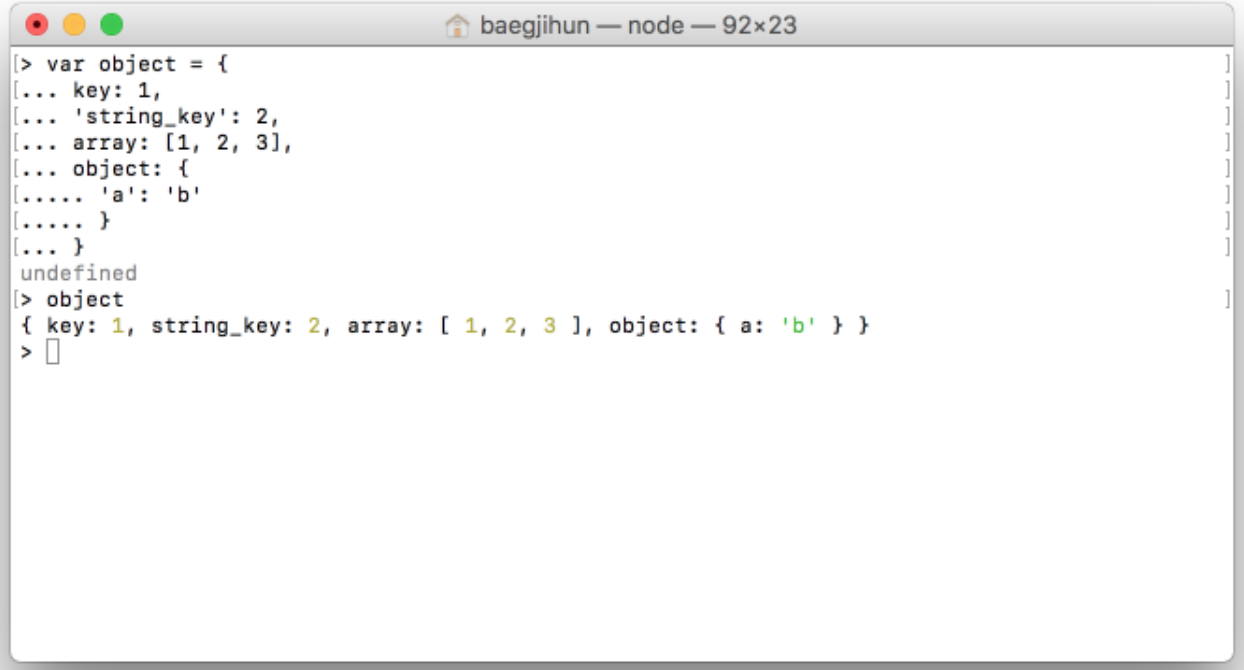
- JSON(JavaScript Object Notation): 일단 이름에서 보이듯이 자바스크립트의 객체를 표시하는 방법입니다. 앞으로 자주 볼 형태이기 때문에 먼저 소개합니다.

- JSON is a syntax for storing and exchanging data. JSON is text, written with JavaScript object notation.

- 속성-값 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷

- JSON이 XML을 대체해서 설정의 저장이나 데이터를 전송 등에 많이 사용됩니다.

JSON 예시



- JSON의 자료형
  - 수(Number): 숫자 아시죠?
  - 문자열(String): 일반적으로 코딩에서 사용하는 그 문자열입니다. 예시: “KWEB”
  - 참/거짓(Boolean): true 는 false
  - 배열(Array): 여러분이 생각하는 그 배열입니다. 대괄호로 나타내며 요소는 쉼표로 구분한다. ex) [1,2,3], [‘A’, ‘B’, ‘C’]
  - 객체(Object): JSON 내부에 JSON 객체가 정의가능합니다.

- 사실 0주차의 그 객체입니다.

Module in Node.js

- Module이란?
  - Node.js에서 모듈이란 관련된 코드들을 하나의 코드 단위로 캡슐화 하는 것으로 메서드와 속성을 미리 정의해 놓은 것을 말합니다

- 즉, 모듈을 사용하면 프로그램의 여러 부분에서 쓰이는 코드조각들을 한대 모아놓고 사용하고자 하는 곳에서 불러와 사용 할 수 있습니다.

- Node.js의 모듈들은 require를 통해 불러와 사용합니다.
  - ex) const fs = require('fs'), var path = require('path')

내장 모듈 사용하기

- Node.js 이미 다양한 내장 모듈들이 있습니다. 아래 사이트에 모든 정보가 있습니다.
  - Node.js의 내장 모듈 알아보기: https://nodejs.org/api/

- 우리는 오늘 그 중에 fs모듈과 path모듈을 사용할 것입니다. 이미 Node.js에 내장되어 있으므로 require('fs')와 require('path')를 이용해 불러오시면 됩니다.

- 먼저, fs 모듈과 path 모듈에 관해 알아본 후 실습을 진행하도록 하겠습니다.

모듈 설명 - fs

- fs 모듈: 파일을 읽고 쓰는 사용하는 모듈, fs는 File System 줄임말이다.
- fs 모듈 아래에는 엄청나게 많은 메소드가 있어서 2가지만 한번 짚어봅시다.
- readFileSync(file, encoding): 파일을 동기적으로 읽어옵니다.
- readFile(file, encoding, callback) - 파일을 비동기적으로 읽어옵니다.
- 비동기적으로 파일을 읽으면 이벤트 리스너를 등록하고 파일을 모두 읽은 뒤 후처리를 할 수 있습니다. 무슨 말인지 모르겠으니 다음주에 알아가는 걸로 하죠.

모듈 설명 - path

- path 모듈: 파일의 path를 다루는 모듈입니다.
  - join(): 여러 개의 이름들을 모두 합쳐 하나의 파일 path로 만들어주는 메소드
  - dirname(): 파일 path에서 디렉토리 이름을 반환해주는 메소드
  - basename(): 파일 path에서 파일의 확장자를 제외한 이름을 반환해주는 메소드
  - extname(): 파일 path에서 파일의 확장자를 반환해주는 메소드

```
const path = require('path');

var my_file = "C:\\project\\kweb\\tired.txt";
var dirname = path.dirname(my_file);
var basename = path.basename(my_file);
var extname = path.extname(my_file);

console.log('path.dirname = %s\npath.basename = %s\npath.extname = %s',dirname,basename,extname);
```

실습 - Node.js에 HTML 파일 띄우기

- index.js - 0주차 파일을 수정해봅시다.

```
const http = require('http');
const path = require('path');
const fs = require('fs');
// const process = require('process');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  // res.end('Hello World\n');

  const indexContent = fs.readFileSync(path.join(__dirname, 'index.html'));
  res.end(indexContent);
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});

```

- index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>KWEB</title>
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

실습 - Node.js에 HTML 파일 띄우기

- node index.js를 입력해서 실행시키면 이상해요. 오른쪽처럼 고칩시다.


```
const http = require('http');
const path = require('path');
const fs = require('fs');
// const process = require('process');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/html');
  // res.end('Hello World');

  const indexContent = fs.readFileSync(path.join(__dirname, 'index.html'));
  res.end(indexContent);
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

실습 - Node.js에 HTML 파일 띄우기

- Hello World가 적혀진 웹페이지가 보인다면 성공!

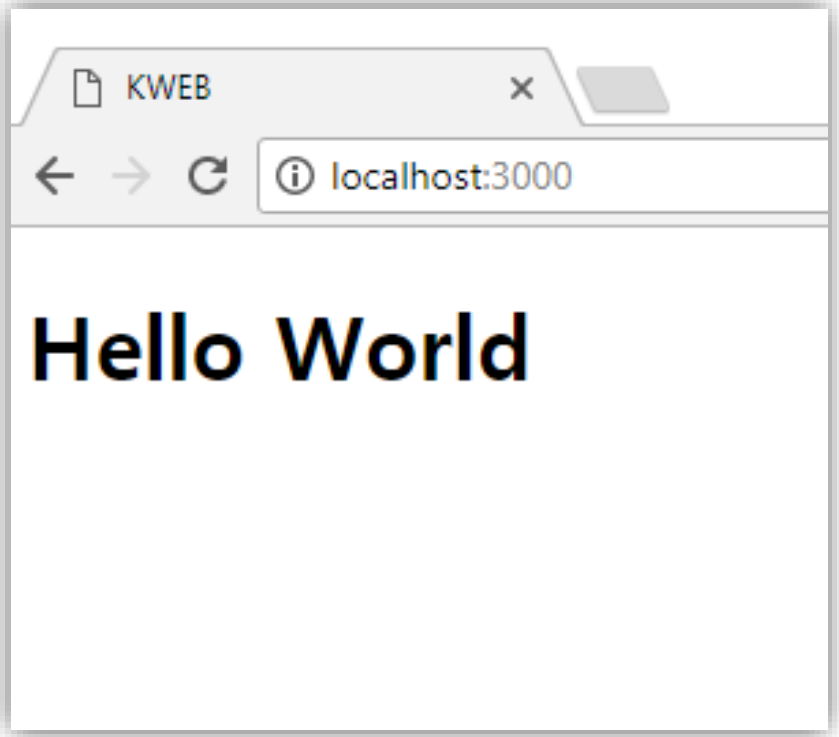
```
const http = require('http');
const path = require('path');
const fs = require('fs');
// const process = require('process');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/html');
  // res.end('Hello World');

  const indexContent = fs.readFileSync(path.join(__dirname, 'index.html'));
  res.end(indexContent);
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```



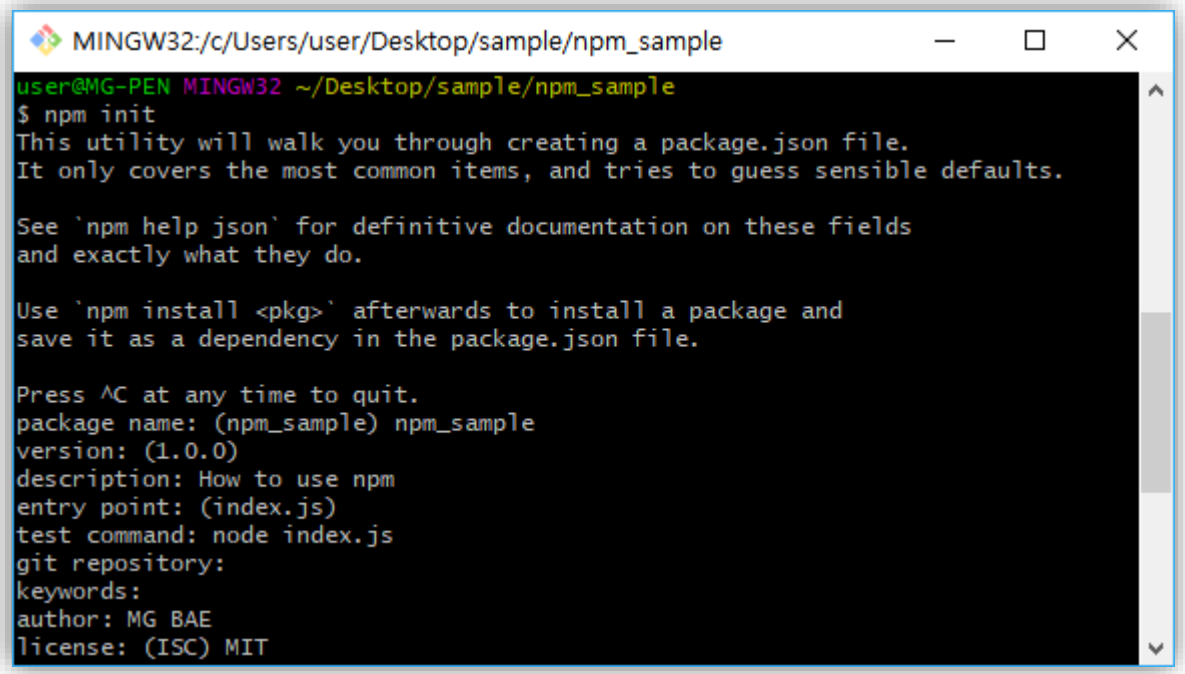
외장 모듈

- Node.js 는 npm에 힘입어 엄청나게 많은 외장 모듈을 보유하고 있습니다.
  - 아! 외장 모듈은 다른 사람이 만들어 둔 모듈을 외장 모듈이라고 해요! 세상은 넓고 능력자는 많..
- 외장 모듈은 npm install을 이용하여 나의 프로젝트 내부에 가져온 이후, 내장 모듈과 동일하게 require을 이용해 사용합니다.
- 외장 모듈 사용은 불러온 이후로는 내장 모듈과 같은 방법으로 사용하면 되므로 딱히 예제는 없습니다!
- But, 앞으로 진행될 스터디에서 꾸준히 사용될 예정이에요~

npm 사용하기

- 대신! 외장 모듈 사용할 때 설치할 때 사용하는 npm을 사용하는 방법을 알아야하겠죠.
- npm init, npm install 등 여러분들이 앞으로 Node.js를 사용하다 보면 consol창에 npm을 쓸 일이 제법 있을 것입니다.
- npm init를 한번 console창에 쳐봅시다.

실습 - npm init



- 왼쪽대로 진행해봅시다.
- Package name, Author 정도만 수정하시고 나머진 그대로 입력해주세요!
- 그러고나면 package.json 파일이 생성될겁니다.
- 이 파일로 npm 의존성 모듈의 버전 등 정보를 관리해줍니다.

package.json

```
{
  "name": "npm_sample",
  "version": "1.0.0",
  "description": "How to use npm",
  "main": "index.js",
  "scripts": {
    "test": "node index.js"
  },
  "author": "MG BAE",
  "license": "MIT"
}
```

- 아까 json 객체 배웠죠? package.json은 패키지의 정보를 담고 있습니다.
- 아까 실습에서 사용한 index.js와 index.html을 같은 폴더에 넣고 npm run test 를 입력해봅시다.
- 아까와 같은 결과를 얻으실 수 있을겁니다.
- 앞으로 실제 개발에서 볼 일이 많을겁니다.

How to use npm install

- 오늘 뒤에 ejs 설치하면서 사용할거지만 여기서 사용방법만 알아봅시다.

- npm install <모듈 이름>: 모듈을 현재 프로젝트에 설치합니다.
  - ex) npm install mysql, npm install ejs

- npm install <모듈 이름> --save: 모듈을 현재 프로젝트에 설치하고 설치 내용을 package.json에 저장해줍니다.
  - ex) npm install mysql --save, npm install ejs --save

- npm uninstall <모듈 이름>: 현재 프로젝트에서 모듈을 삭제해줍니다.

- package.json이 과연 달라지는지는 뒤에 ejs 설치하면서 알아봅시다.

모듈 만들기

- npm 사용해봤고! 외장 모듈 설치하는 방법도 알았습니다!
- 근데 가만히 생각해보면 외장 모듈이 존재한다는 이야기는 모듈을 제작할 수 있다는 이야기가 되죠?
- 그렇다면 당연히 여러분들도 모듈을 제작하실 수 있습니다.
- 간단한 계산기 모듈을 한 번 제작해봅시다!

모듈 만들기

- 그 전에 먼저 간단한 모듈 만드는 예제부터 보죠. 더하기 함수입니다!

- 이제 require을 통해 해당 모듈을 불러오면 오른쪽의 모듈에 정의되어있는 add 기능을 사용할 수 있습니다.

```
var calc = {};

calc.add = function(a,b) {
    return a + b;
};

module.exports = calc;
```

실습 - 계산기 모듈 제작

```
function add(a,b) {
    return a+b;
}

function sub(a,b) {
    return a-b;
}

function mul(a,b) {
    return a*b;
}

function div(a,b) {
    if (b==0) {
        return new Error('zero divider error.')
    }
    return a/b;
}

function getE() {
    return Math.E;
}

module.exports = {
    add: add,
    sub: sub,
    mul: mul,
    div: div,
    getE: getE
}
```

- 기본적으론 모듈 내부에서 정의된 변수나 함수는 해당 파일 내부에서만 유효합니다.
- But! 외부에서 사용할 변수나 함수는 module.exports를 통해 공개 할 수 있습니다.
- 앞의 예제랑 만드는 방법이 좀 다른 거 같지만 똑같은 겁니다.
- 왼쪽의 calc.js 파일을 만들어봅시다!
  - 이 파일은 모듈이며 사칙연산을 수행하는 함수와 E 상수를 반환 하는 함수를 외부로 노출 시키고 있습니다.

실습 - 계산기 모듈 사용하기

```
const http = require('http');
const fs = require('fs');
const path = require('path');
const calculator = require('./calc.js');
const ejs = require('ejs');

const hostname = '127.0.0.1';
const port = 3000;

var result = calculator.add(1,3);

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end(`1 + 3 = ${result}`);
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
  console.log(`add(1,3) = ${result}`);
});
```

- 왼쪽 파일을 만들고 파일의 위치에 calc이라는 폴더를 만들어 그 안에 calc.js를 넣어봅시다.
- require함수를 사용해 대상 모듈의 내용을 로드하며, 반환값은 모듈 내부에서 module.exports에 넣어 놓은 값과 같습니다.
- 사용은 왼쪽과 같이 Caluator라는 임의의 변수에 모듈을 대입하여 놓고 Calculator.add(1, 3)와 같은 방법으로 사용합니다.

View Template & View Engine

- View Template: 클라이언트에 응답을 보낼 때 사용하려고 미리 만들어 놓은 웹 문서의 원형
- View Engine: 뷰 템플릿을 사용해 결과 웹 문서를 자동으로 생성한 후 응답을 보내는 역할을 수행합니다.
- Node.js에는 몇 개의 뷰 엔진이 있으나 우리는 HTML과 가장 똑같은 ejs를 사용할 것입니다.
- (원래는 Express.js와 함께 등장하려고 했으나! 좀 편하게 지금 등장시켰어요, MVC 패턴 관련 공부할 때 좀 더 자세히 알아봅시다. 지금은 사용법만 익혀요!)

EJS

- 당연하게도 외장 모듈입니다.
- EJS는 기존의 HTML과 동일하게 웹 문서에서 사용하는 태그를 템플릿 파일로 집어넣을 수 있습니다.
- 이외에도 필요한 부분에만 변수를 삽입하거나 중간에 Javascript 코드도 사용가능합니다. (묘하게 PHP 생각이 나서 기분 나쁘기도..)
- 이러한 View Engine의 사용으로 손쉽게 서버의 응답 결과를 View로 넘겨줄 수 있습니다.

EJS 설치 - npm 사용

- npm install ejs --save로 실행시키면 저장된 package.json의 내용이 바뀝니다. 이후에 이 package.json을 이용하면 npm install만 해도 ejs가 설치됩니다.

- 한번 npm uninstall, npm install ejs, npm install ejs --save 등 명령어들을 사용해보세요~

- EJS는 실습으로 한번 알아봅시다. 익히 보던 HTML과 같은 형식이라 편하실 거에요!

실습 - 계산기 모듈 사용하기 + ejs

```
const http = require('http');
const fs = require('fs');
const path = require('path');
const calculator = require('./calc.js');
const ejs = require('ejs');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  fs.readFile(path.join(__dirname, 'ejs.ejs'), (err,data) => {
    if(err) {
        res.statusCode = 500;
        return;
    } else {
        const templateData = {
            E: calculator.getE(),
            '1 + 2': calculator.add(1,2),
            '3 - 1': calculator.sub(3,1),
            '3 * 14': calculator.mul(3,14),
            '14 / 2': calculator.div(14,2)
        }

        res.statusCode = 200;
        res.setHeader('Content-Type', 'text/html');
        res.end(ejs.render(data.toString(), {
            arr: [1,2,3],
            data: templateData
        }));
    }
  });
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

- 앞의 계산기 모듈을 ejs를 사용해 홈페이지로 결과를 띄워봅시다.
- 크게 바뀌는건 맨마지막 res.end가 ejs.render로 바뀝니다. <변수: 데이터>의 형식으로 ejs 템플릿에 값을 전달해줄 수 있습니다.
- 복잡한 것 같지만 Express.js를 배우고 나면 더 간단해집니다. 기대하세요 ^^
- 참고로 readFile은 비동기 함수입니다.

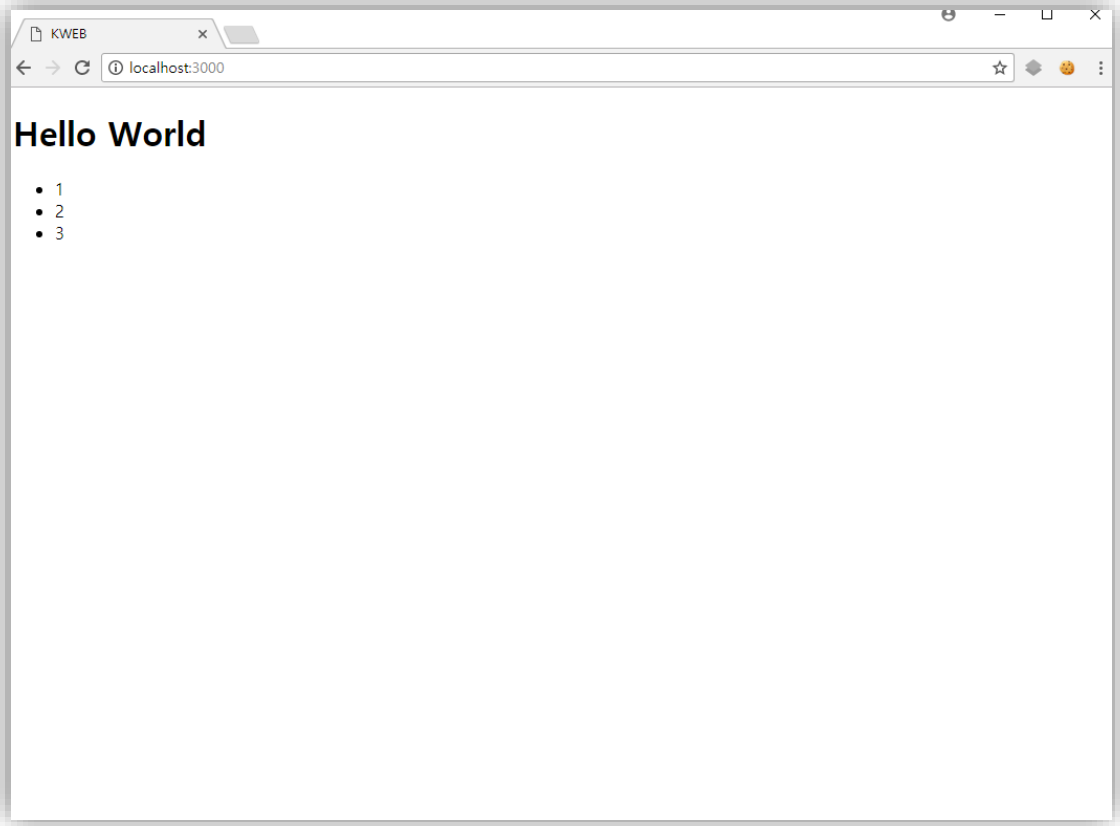
실습 - 계산기 모듈 사용하기 + ejs

- ejs.ejs

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>KWEB</title>
</head>
<body>
    <h1>Hello World</h1>

    <ul>
        <% for(var i = 0; i<arr.length; i++) {%>
        <li><%- arr[i] %></li>
        <% } %>
    </ul>
</body>
</html>
```

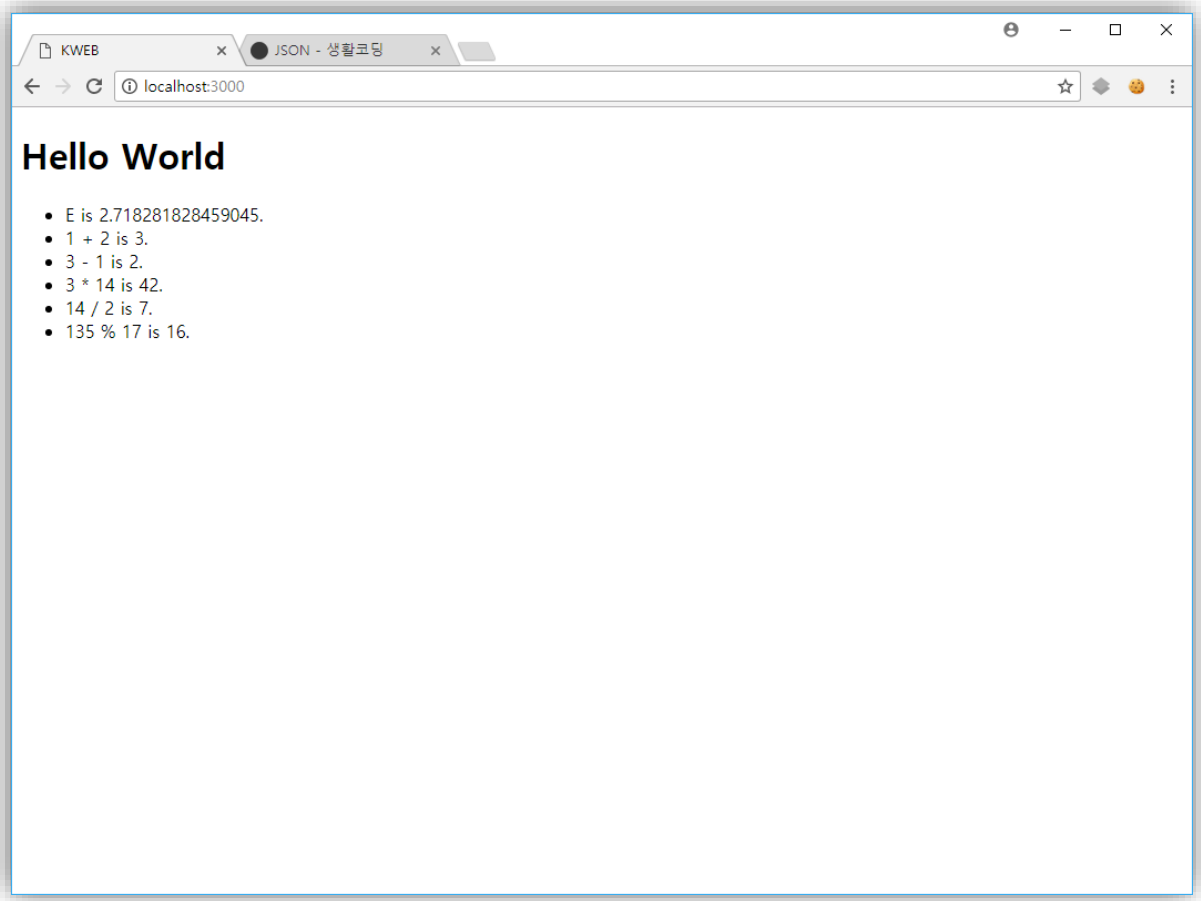
- 실행 결과



과제 (~ 9/25 17:00)

- 1주차 내용이 너무 어려워서 당황했죠.. 쥬륵.. 그치만 과제는 쉽게 출제했어요!
- 새로운 모듈 cal2.js를 만들고 그 안에는 나눗셈 연산에서 나머지를 구하는 즉 module 연산(%) 을 정의하고 앞의 index.js에서 받아와서 templateData 에 % 연산을 사용하는 예제를 추가 시키세요.
- 그러고 templateData를 다음 슬라이드와 같이 출력시키는 것이 과제입니다. 실습에선 ejs에 arr만 출력하고 data는 출력안했어요! 그래서 그걸 출력시키는게 과제입니다.
- 제출기한: 9월 25일 오후 5시 - 제출은 각 스터디장에게!

과제 예시



- index.js, calc.js, calc2.js, ejs.ejs 파일 4개 다 제출해주세요!
- [KWEB 1주차] 학번_이름 으로 제출해주시면 감사하겠습니다.
- 과제 제출 E-mail
  - 월 7시: 15 배민근 (baemingun@naver.com)
  - 화 7시: 16 백지훈 (bjh970913@gmail.com)

It’s all today!