

MEAN 스택 활용 웹 개발

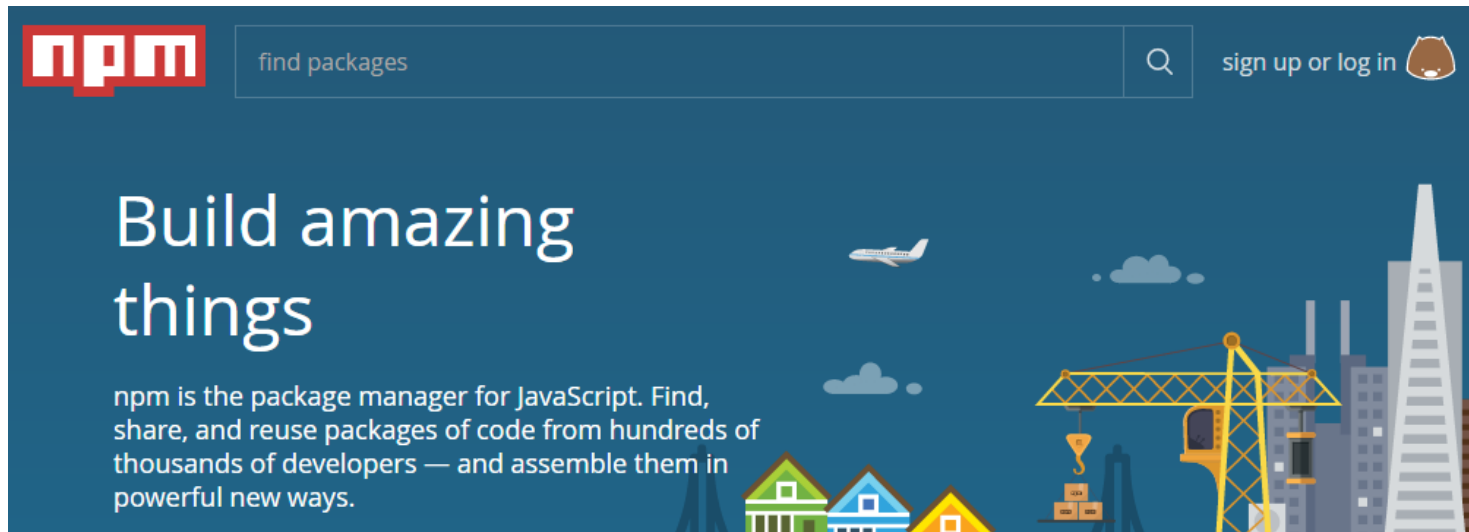
Node.js 모듈 관리



2016년도 2학기 2학년 방과후학교

Node.js 모듈 관리

- npm은 아이작 슐레터(Issac Z. Schlueter)라는 사람이 만든 node.js를 위한 패키지 매니저
- npm을 이용하여 많은 개발자가 만들어 공유하고 있는 확장 모듈의 검색과 설치, 업데이트 등을 쉽게 할 수 있다.
- <https://www.npmjs.org>



모듈 검색

- 모듈 설명
- 모듈 install 방법
- 모듈 사용 예제

★ **urlencode** public

encodeURIComponent with charset



npm install urlencode

1 dependency version 1.1.0
71 dependents updated a year ago
12,810 downloads in the last month

7 ★

Install

```
$ npm install urlencode
```

Usage

```
var urlencode = require('urlencode');

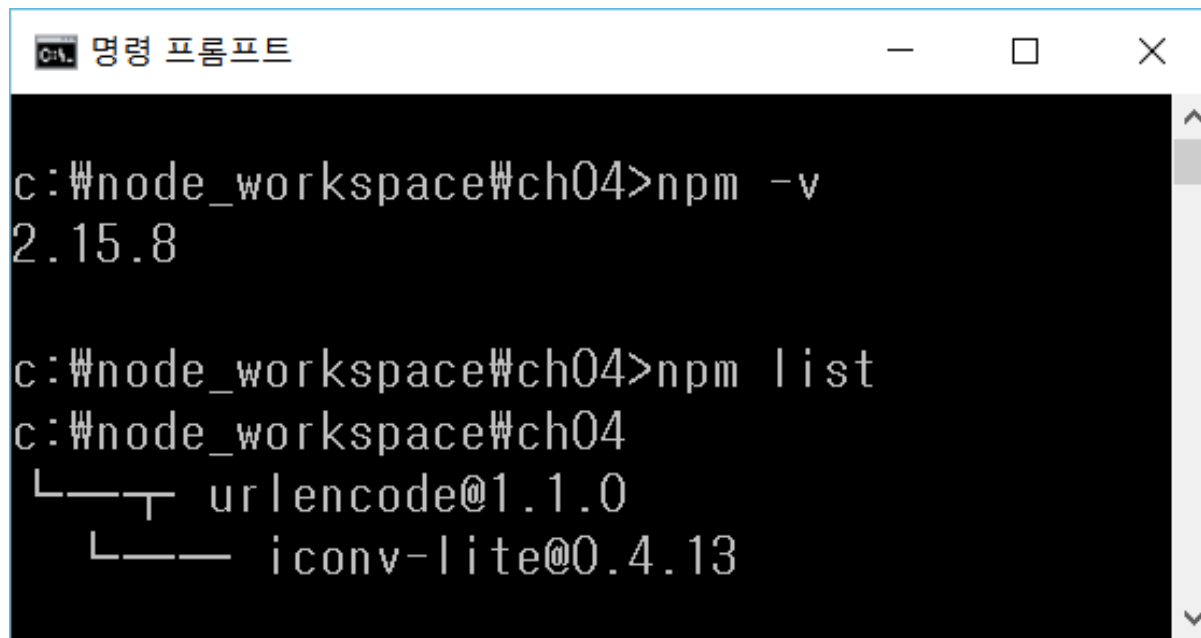
console.log(urlencode('苏千')); // default is utf8
console.log(urlencode('苏千', 'gbk')); // '%CB%D5%C7%A7'

// decode gbk
urlencode.decode('%CB%D5%C7%A7', 'gbk'); // '苏千'

// parse gbk querystring
urlencode.parse('nick=%CB%D5%C7%A7', {charset: 'gbk'}); // {nick: '苏千'}
```

패키지 매니저 - npm

- npm은 Node.js와 함께 설치되는 콘솔 기반의 프로그램이다.
- npm 사이트 뿐만 아니라 npm 명령을 이용해서 모듈을 검색하고 모듈의 정보를 볼 수 있다. 모듈을 설치하고 업데이트, 삭제하는 기능도 제공한다.



```
C:\node_workspace\ch04>npm -v
2.15.8

C:\node_workspace\ch04>npm list
C:\node_workspace\ch04
├── urlencode@1.1.0
└── iconv-lite@0.4.13
```

npm 주요 옵션

- `init` : 패키지 준비, `package.json` 파일 생성
- `install` : `package.json` 패키지에 정의된 모듈 전체 설치
- `install [module]` : 개별 모듈 설치
- `list` : 설치된 모듈 목록 보기
- `info` : 모듈 정보 보기
- `search` : 모듈 검색
- `update` : 모듈 업데이트
- `uninstall` : 모듈 삭제

npm 모듈 설치

■ 전역 설치

- ✓ 라이브러리 폴더(lib/node_modules)에 설치하고, 모든 프로젝트에서 공유한다.
- ✓ 설치 명령에 -g 옵션을 추가로 작성하고, 관리자 권한이 필요하다.
- ✓ 예) > [sudo] npm install -g nodemon

■ 지역 설치

- ✓ 설치하려는 폴더에 node_modules 폴더가 생성되고 설치된다.
- ✓ 지역 설치 Node.js 프로젝트마다 모듈을 설치해야 한다.
- ✓ 예) > npm install async

npm 모듈 설치

- 보통 Node.js 애플리케이션에서 사용하는 모듈은 지역 설치를 권장한다.
- 모듈을 전역으로 설치하면 모든 프로젝트에 영향을 주기 때문에, 특정 버전에 의존적인 프로젝트의 경우 문제가 생길 수 있다.
- 단, mocha와 같은 테스트용 모듈, nodemon 등의 공통 유틸과 같은 모듈은 전역 설치를 권장한다.

패키지 설정 파일

- Node.js 애플리케이션을 작성하는 프로젝트에는 다양한 모듈을 사용하고, 실행을 위한 환경정보를 package.json 파일을 이용해서 작성한다.
- npm은 package.json에 작성한 정보를 이용해서 모듈을 설치하거나, package에 작성한 스크립트를 실행시키기도 한다.
- package.json은 npm init 명령을 이용해서 생성한다.

패키지 설정 파일

■ package.json 파일 생성

```
npm
c:\node_workspace\ch05>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg> --save` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
name: (ch05) npm_test
version: (1.0.0)
description: npm test
entry point: (app.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to c:\node_workspace\ch05\package.json
{
  "name": "npm_test",
  "version": "1.0.0",
```

```
명령 프롬프트
{
  "name": "npm_test",
  "version": "1.0.0",
  "description": "npm test",
  "main": "app.js",
  "scripts": {
    "test": "echo %Error: no test specified% && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this ok? (yes)

c:\node_workspace\ch05>
```

패키지 설정 파일

- package.json 파일에 모듈 의존성 직접 추가

```
"dependencies": {  
  "async": "^1.5.0",  
  "jade": "^1.11.0"  
},  
"devDependencies": {  
  "mocha": "^2.3.4"  
},
```

- 모듈 의존성 자동 추가

✓ npm install async --save

패키지 설정 파일

```
C:\ 명령 프롬프트

c:\#node_workspace#ch05>npm install async --save
npm WARN package.json npm_test@1.0.0 No repository field.
npm WARN package.json npm_test@1.0.0 No README data
async@2.0.1 node_modules\async
└── lodash@4.14.0

c:\#node_workspace#ch05>npm install jade --save
npm WARN package.json npm_test@1.0.0 No repository field.
npm WARN package.json npm_test@1.0.0 No README data
npm WARN deprecated jade@1.11.0: Jade has been renamed to pug, please install the latest version of pug instead of jade
npm WARN deprecated transformers@2.1.0: Deprecated, use jstransformer
jade@1.11.0 node_modules\jade
├── commander@2.6.0
├── character-parser@1.2.1
└── void-elements@2.0.1
```

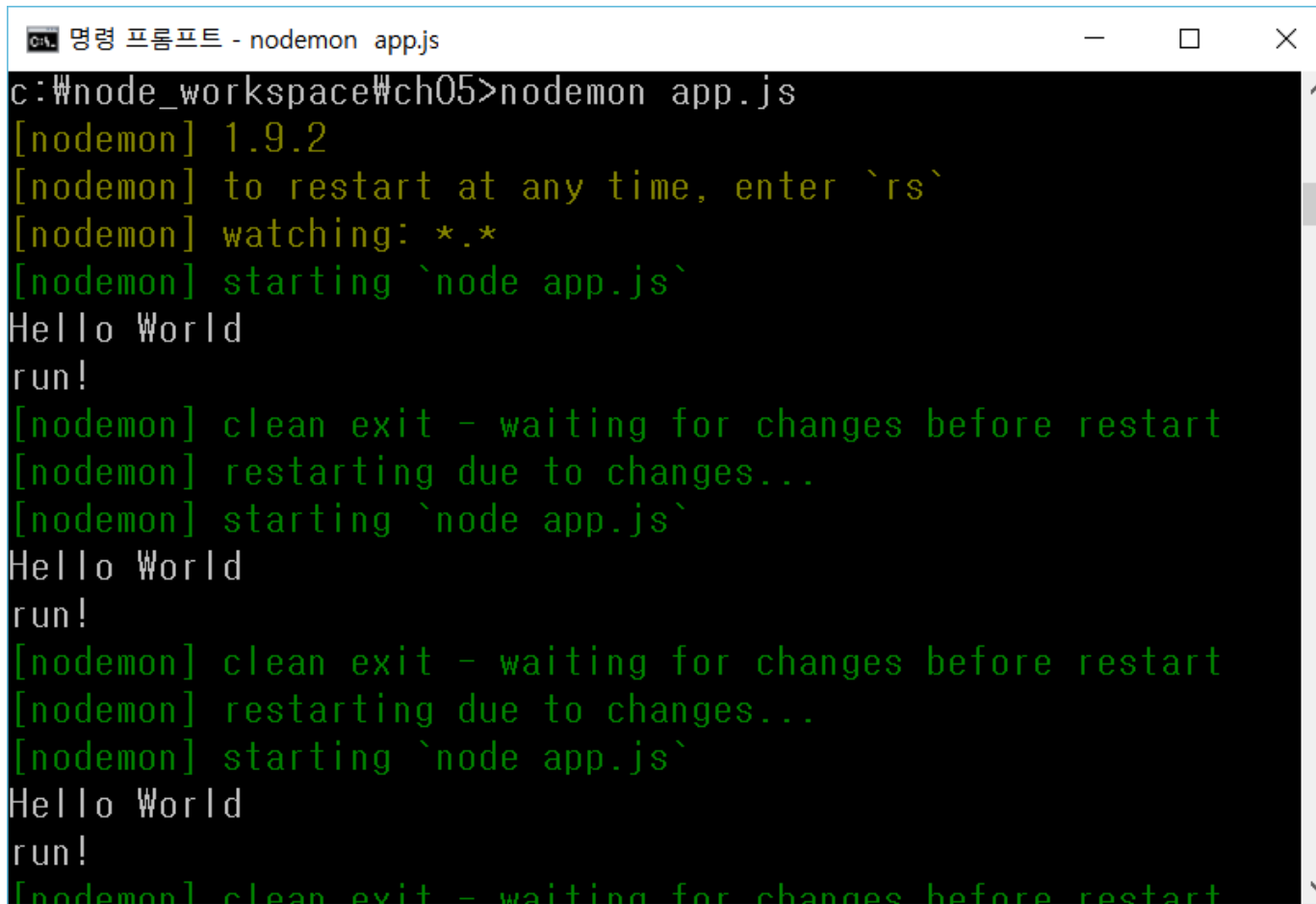
```
1 {
2   "name": "npm_test",
3   "version": "1.0.0",
4   "description": "npm test",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "",
10  "license": "ISC",
11  "dependencies": {
12    "async": "^2.0.1",
13    "jade": "^1.11.0"
14  }
15 }
```

확장 모듈 - nodemon

- nodemon 모듈은 소스 코드를 저장하면 자동으로 재시작시켜 주는 모듈이다.
- 사용하는 방법은 node 명령어 대신 nodemon 명령어를 이용해서 시작시킨다.
→ nodemon myApp.js
- 콘솔에 rs를 입력하면 소스가 변경되지 않아도 자동으로 재시작한다.

nodemon 설치 및 실행

- [sudo] npm install -g nodemon



```
명령 프롬프트 - nodemon app.js
c:\Wnode_workspace\ch05>nodemon app.js
[nodemon] 1.9.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node app.js`
Hello World
run!
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Hello World
run!
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Hello World
run!
[nodemon] clean exit - waiting for changes before restart
```

[실습] 패키지 관리

- package.json 파일 생성 후 async 모듈을 지역모듈로 설치해보자!!

```
1  {
2    "name": "12_basic_module",
3    "version": "1.0.0",
4    "description": "",
5    "main": "event.js",
6    "dependencies": {
7      "async": "^2.0.1",
8      "urlencode": "^1.1.0"
9    },
10   "devDependencies": {},
11   "scripts": {
12     "test": "echo \"Error: no test specified\" && exit 1"
13   },
14   "author": "",
15   "license": "ISC"
16 }
```