

Node.js

npm - 김근형 강사

npm(Node Package Manager)

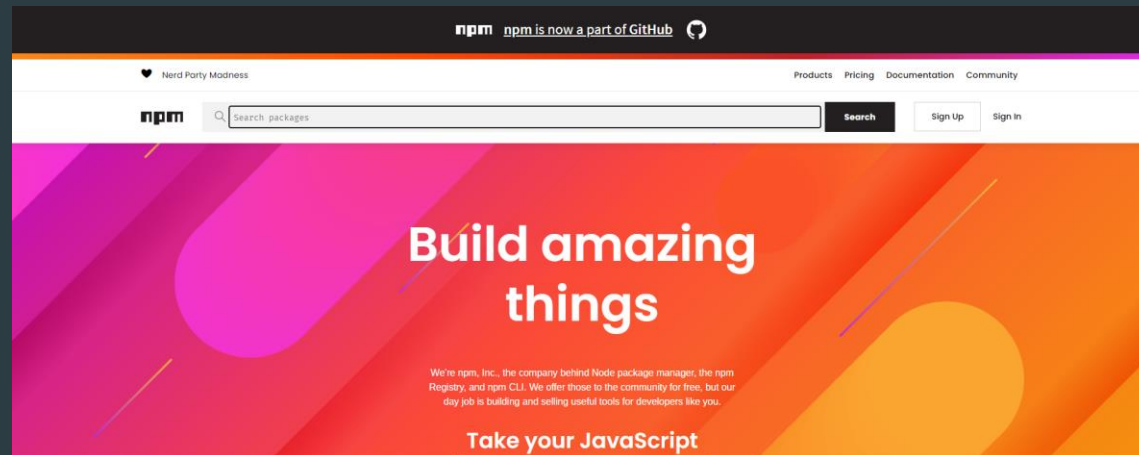
► npm



npm(Node Package Manager)

▶ npm

- ▶ 자바스크립트 패키지 매니저.
- ▶ Node.js에서 사용할 수 있는 모듈들을 패키지화하여 모아둔 저장소 역할과 패키지 설치 및 관리를 위한 CLI(Command line interface)를 제공한다.
- ▶ 자신이 작성한 패키지를 공개할 수도 있고 필요한 패키지를 검색하여 재사용할 수도 있다.
- ▶ npmjs.com 사이트에서 모듈을 검색해 사용이 가능하다.



npm(Node Package Manager)

▶ npm 명령

명령	설명
npm install <package>	패키지를 설치한다 지역설치가 아닌 전역설치를 하고 싶다면 옵션 -g 를 준다. npm install 명령어에 --save 옵션을 사용하면 패키지 설치와 함께 package.json 의 dependencies 에 설치된 패키지 이름과 버전이 기록된다. --save-dev 는 개발 패키지 설치 옵션이다.
npm init	package.json 을 생성한다. npm 은 package.json 파일을 통해서 프로젝트 정보와 패키지의 의존성(dependency)을 관리한다.
npm uninstall <package-name>	패키지를 제거한다. -g 옵션을 주면 전역 패키지를 제거한다.
npm update <package-name>	패키지를 업데이트 한다.
npm ls -g --depth=0	전역 설치 패키지를 확인한다.
npm start	package.json scripts 프로퍼티의 start 실행
npm root	패키지 설치 폴더 확인
npm -v	npm 버전 확인
npm help <command>	npm 사용법 확인

npm(Node Package Manager)

▶ npm을 이용한 외부모듈 설치 예제

```
PS C:\Users\User> npm install upper-case
npm WARN ENOENT: no such file or directory, open 'C:\Users\User\package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN ENOENT: no such file or directory, open 'C:\Users\User\package.json'
npm WARN User No description
npm WARN User No repository field.
npm WARN User No README data
npm WARN User No license field.

+ upper-case@2.0.1
added 2 packages from 2 contributors and audited 2 packages in 0.548s
found 0 vulnerabilities
```

```
var http = require('http');
var uc = require('upper-case');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write(uc.upperCase("Hello World!"));
  res.end();
}).listen(2000);
```

HELLO WORLD!

package.json

▶ package.json

- ▶ package.json은 프로젝트 정보와 의존성(dependencies)을 관리하는 문서다.
- ▶ 이미 작성된 package.json 문서는 어느 곳에서도 동일한 개발 환경을 구축할 수 있게 해준다.
- ▶ JSON 포맷으로 작성해야 하며, 다음과 같은 옵션들이 추가될 수 있다.

옵션	설명
name	URL이나 Command Line의 일부로 사용될 소문자로 표기된 214자 이내의 프로젝트(패키지) 이름으로, 간결하고 직관적인 이름으로 설정하되 다른 모듈과 동일한 이름을 피해야 한다.
version	SemVer(The semantic versioner for npm)로 분석 가능한 형태의 버전을 지정한다.
description	프로젝트(패키지)의 설명을 지정한다. (npm search 사용 시 도움이 된다.)
keywords	프로젝트(패키지)의 키워드를 배열로 지정한다. (npm search 사용 시 도움이 된다.)
homepage	프로젝트 홈페이지로 연결되는 URL을 지정한다.
bugs	패키지에 문제가 있을 때 보고될 이슈 트래커(추적시스템) 및 이메일 주소 등에 대한 URL을 지정한다.
license	패키지 사용을 허용하는 방법과 제한 사항을 알 수 있도록 라이선스를 지정한다.
author	제작자의 이름을 지정한다.

package.json

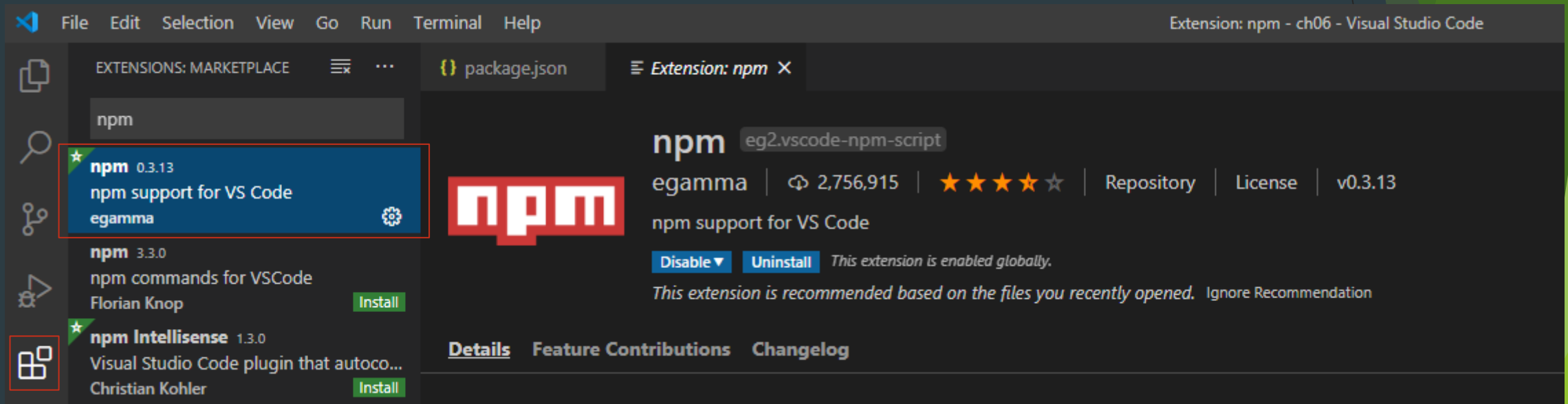
▶ package.json

- ▶ JSON 포맷으로 작성해야 하며, 다음과 같은 옵션들이 추가될 수 있다.

옵션	설명
files	패키지가 의존성으로 설치될 때 같이 포함될 파일들의 배열. 생략하면 자동 제외로 설정된 파일을 제외한 모든 파일이 포함된다
main	프로그램의 기본 진입 점(entry point)을 지정한다. 패키지의 이름이 <code>jquery</code> 이고, 사용자가 <code>require('jquery')</code> 를 사용하면 진입 점의 메인 모듈에서 <code>exports object</code> 가 반환(return)된다.
repository	코드가 존재하는 장소를 지정한다. <code>GitHub</code> 를 사용하면 <code>npm docs</code> 명령을 사용하여 찾을 수 있다.
script	패키지 라이프 사이클에서 여러 번 실행되는 스크립트 명령을 포함한다.
dependencies	패키지의 배포 시 포함될 의존성 모듈을 지정한다.
devDependencies	패키지의 개발 시 사용될 의존성 모듈을 지정한다.(배포 시 포함되지 않는다)
peerDependencies	패키지의 호환성 모듈을 지정한다.(<code>npm@3</code> 이후로 배포 시 포함되지 않는다.)
bundledDependencies	패키지를 게시할 때 번들로 묶을 패키지 이름을 배열로 지정한다. <code>npm</code> 패키지를 로컬에서 보존해야 하거나 단일 파일 다운로드를 통해 사용할 수 있는 경우 <code>npm pack</code> 을 실행하여 패키지를 <code><name>-<version>.tgz</code> 형태의 <code>tarball</code> 파일)로 묶을 수 있다.
optionalDependencies	<code>npm</code> 을 찾을 수 없거나 설치에 실패한 경우 계속 진행하려면 <code>optionalDependencies</code> 객체에 넣을 수 있다. <code>dependencies</code> 동일하게 배포 시 포함될 의존성 모듈을 지정하지만, 빌드 실패로 인해 설치 과정이 중단되지 않는다.
engines	패키지가 작동하는 <code>Node</code> 버전을 지정한다.
private	개인 저장소의 우연한 발행을 방지하기 위해 <code>npm</code> 에서 패키지의 공개 여부를 지정한다.

npm 프로젝트 환경 세팅(Visual Studio Code)

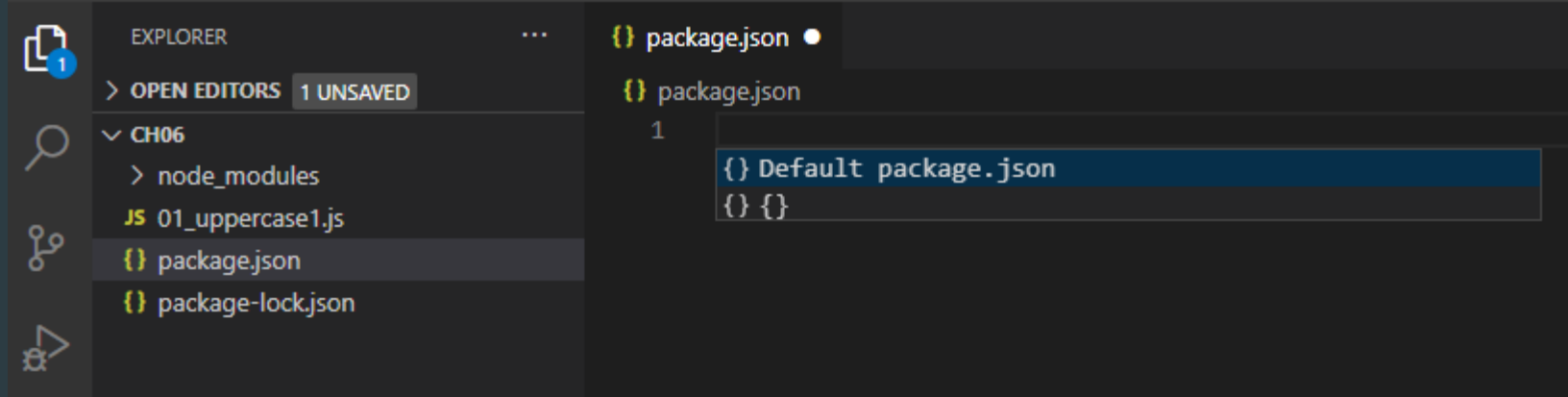
▶ Visual Studio Package 설치



npm 프로젝트 환경 세팅(Visual Studio Code)

▶ package.json 생성

- ▶ package.json을 생성한 후에 `ctrl+space` 를 누르게 되면 자동 완성 기능이 활성화 된다.




npm 프로젝트 환경 세팅(Visual Studio Code)

▶ dependencies 설정

- ▶ package.json의 dependencies에 원하는 패키지를 넣고 ctrl+shift+p 를 통해 커맨드 창을 불러온다
- ▶ 해당 창에서 npm: install dependencies 실행

```
{  
  "name": "test",  
  "description": "ch06 package",  
  "dependencies": {  
    "lower-case": "*"   
  }  
}
```



```
>npm  
npm: Install Dependencies recently used
```

npm 프로젝트 환경 세팅(Visual Studio Code)

- ▶ dependencies 설정
 - ▶ package-lock.json 이 생성되면 성공

```
{ package-lockjson > ...
1  {
2    "name": "test",
3    "requires": true,
4    "lockfileVersion": 1,
5    "dependencies": {
6      "lower-case": {
7        "version": "2.0.1",
8        "resolved": "https://registry.npmjs.org/lower-case/-/lower-case-2.0.1.tgz",
9        "integrity": "sha512-LiWgFDLLb1dwbfQZsSglpRj+1ctGnayXz3Uv0/W08n558JycT5fg6zkNcnW0G68Nn0aEldTFeEfmjCfmqry/rQ==",
10       "requires": {
11         "tslib": "^1.10.0"
12       }
13     },
14     "tslib": {
15       "version": "1.13.0",
16       "resolved": "https://registry.npmjs.org/tslib/-/tslib-1.13.0.tgz",
17       "integrity": "sha512-i/6DQjL8Xf3be4K/E6Wgpekn5Qas11usyw++dAA35Ue5orEn65VIx0A+YvNN19HV3qv70T7CNwj0DHZrLwvd1Q=="
18     }
19   }
20 }
21
```

npm 프로젝트 환경 세팅(Visual Studio Code)

▶ 확인

```
var http = require('http');
var lc = require('lower-case');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write(lc.lowerCase("Hello World!"));
  res.end();
}).listen(2000);
```

hello world!