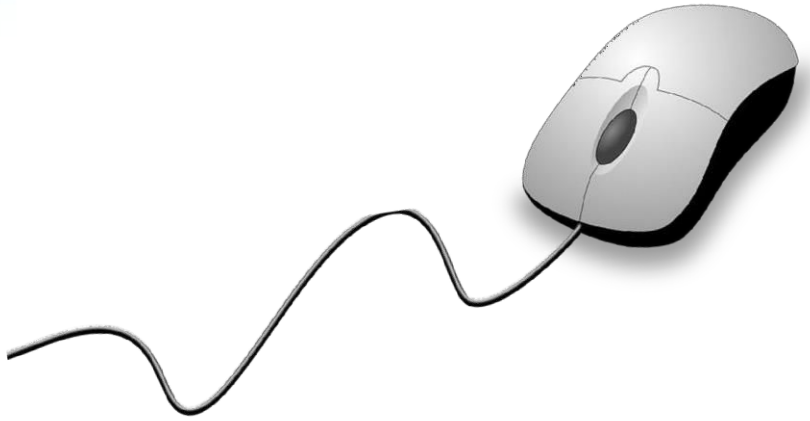
공개SW솔루션 설치 & 활용 가이드

시스템SW > SW공학도구



제대로 배워보자

How to Use Open Source Software

Open Source Software Installation & Application Guide

오픈소스 소프트웨어 통합지원센터
Open Source Software Support Center

CONTENTS

1. 개요

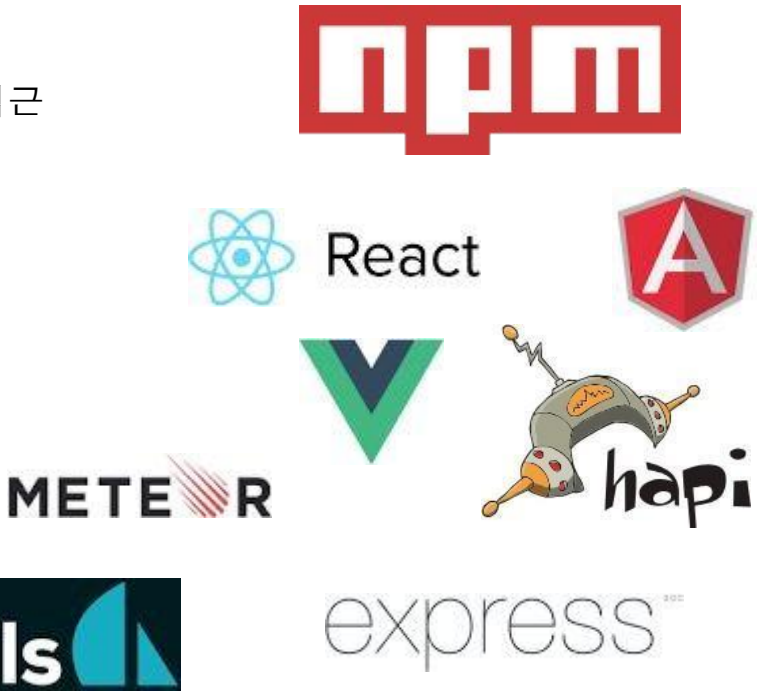
소개	• Chrome V8 JavaScript 엔진으로 빌드된 JavaScript 런타임 • 이벤트 기반, 논 블로킹 I/O 모델을 사용하여 가볍고 효율적임 • Node.js 패키지 생태계인 npm은 세계에서 가장 큰 오픈 소스 라이브러리 환경		
주요기능	• JavaScript 서버, 모바일, 단말 로직 개발 가능		
대분류	• 시스템 SW	**소분류**	• SW공학도구
라이선스 형태	• MIT License	**사전설치 솔루션**	• N/A
운영체제	• Linux, macOS, Microsoft Windows, SmartOS, FreeBSD, IBM AIX	**버전**	• 19.3.0(2022-12 최신) • 18.12.1(권장 버전)
특징	• 한가지 언어(JavaScript)로 server-side, client-side 모두 개발 가능 • Single Thread기반의 비동기 IO 처리 방식으로 작업 속도 빠름		
보안취약점	• 취약점 ID : CVE-2022-3602 • 심각도 : High • 취약점 설명 : x.509 인증서 확인에서의 이름 제약조건 검사 중 버퍼 오버플로우가 발생할 수 있으며, 이를 이용해 공격자가 악의적인 이메일 주소를 만들어 공격자가 제어하는 4바이트를 오버플로우 시킬 수 있음. 이를 통해 DoS(서비스 거부)및 잠재적으로는 원격 코드 실행이 발생할 수 있음 • 대응방안 : 18.x, 19.x 버전의 이 문제를 해결한 새로운 패치 적용 • 참고 경로 : https://nodejs.org/en/blog/vulnerability/november-2022-security-releases/		
개발회사/커뮤니티	• Node.js Foundation		
공식 홈페이지	• https://nodejs.org		

1. 개요

- Node.js는 2009년에 Ryan Dahl이 만든 server-side 개발도구
- 파일 업로드가 진행되는 동안 다른 작업을 수행 할 수 없는 서버의 문제를 해결하기 위해 고안
- 이벤트 기반의 비동기 방식 모델을 사용함으로써 데이터를 주고 받는데 효율적이며 그러한 이유로 데이터 송수신이 빈번한 실시간 어플리케이션 제작 적합

2. 기능요약

- WEB 기술 스택 중 가장 유명한 JavaScript를 사용함으로써 언어 장벽 낮음
- 서버와 클라이언트 사이드 모두에 runtime 환경 제공
- 비동기와 이벤트 기반 처리 방식
- 빠른 실행속도
- 단일 스레드 / 높은 확장성
- Node.js의 패키지 생태환경(npm)을 통한 간편한 패키지 접근
- 활용 가능 분야
  - 디스크 또는 네트워크 입출력이 잦은 응용프로그램
  - 데이터 스트리밍 응용 프로그램
  - 데이터 집약적인 실시간 응용프로그램(DIRT)
  - JSON APIs 기반의 응용프로그램



<node.js위한 다양한 web framework 및 관련 항목>

3. 실행환경

- 지원 형태(Support type)
  - **Tier 1**: Node.js 코어 팀과 광범위한 커뮤니티에 의해 전 범위 테스트와 유지보수 지원
  - **Tier 2**: 전 범위 테스트를 지원하지만 유지보수가 제한되며 빈번하게 플랫폼의 공급업체가 지원하는 경우 있음
  - **Experimental** : 안정적으로 컴파일 할 수 없거나 'test suite'을 통과하지 못할 수 있음
    적어도 한 명이 적극적으로 유지보수를 지원하며 팀은 신뢰할 수 있는 질 좋은 지원을 광범위하게 제공하려 노력함

Operating System	Architectures	Versions	Support Type	Notes
GNU/Linux	x64	kernel >= 4.18[1], glibc >= 2.28	Tier 1	e.g. Ubuntu 20.04, Debian 10, RHEL 8
GNU/Linux	x64	kernel >= 3.10, musl >= 1.1.19	Experimental	e.g. Alpine 3.8
GNU/Linux	x86	kernel >= 3.10, glibc >= 2.17	Experimental	Downgraded as of Node.js 10
GNU/Linux	arm64	kernel >= 4.18[1], glibc >= 2.28	Tier 1	e.g. Ubuntu 20.04, Debian 10, RHEL 8
GNU/Linux	armv7	kernel >= 4.18[1], glibc >= 2.28	Tier 1	e.g. Ubuntu 20.04, Debian 10
GNU/Linux	armv6	kernel >= 4.14, glibc >= 2.24	Experimental	Downgraded as of Node.js 12
GNU/Linux	ppc64le >=power8	kernel >= 4.18[1], glibc >= 2.28	Tier 2	e.g. Ubuntu 20.04, RHEL 8
GNU/Linux	s390x	kernel >= 4.18[1], glibc >= 2.28	Tier 2	e.g. RHEL 8
Windows	x64, x86 (WoW64)	>= Windows 10/Server 2016	Tier 1	[2] [3]

출처 : https://github.com/nodejs/node/blob/master/BUILDING.md#fn2

4. 설치 및 실행

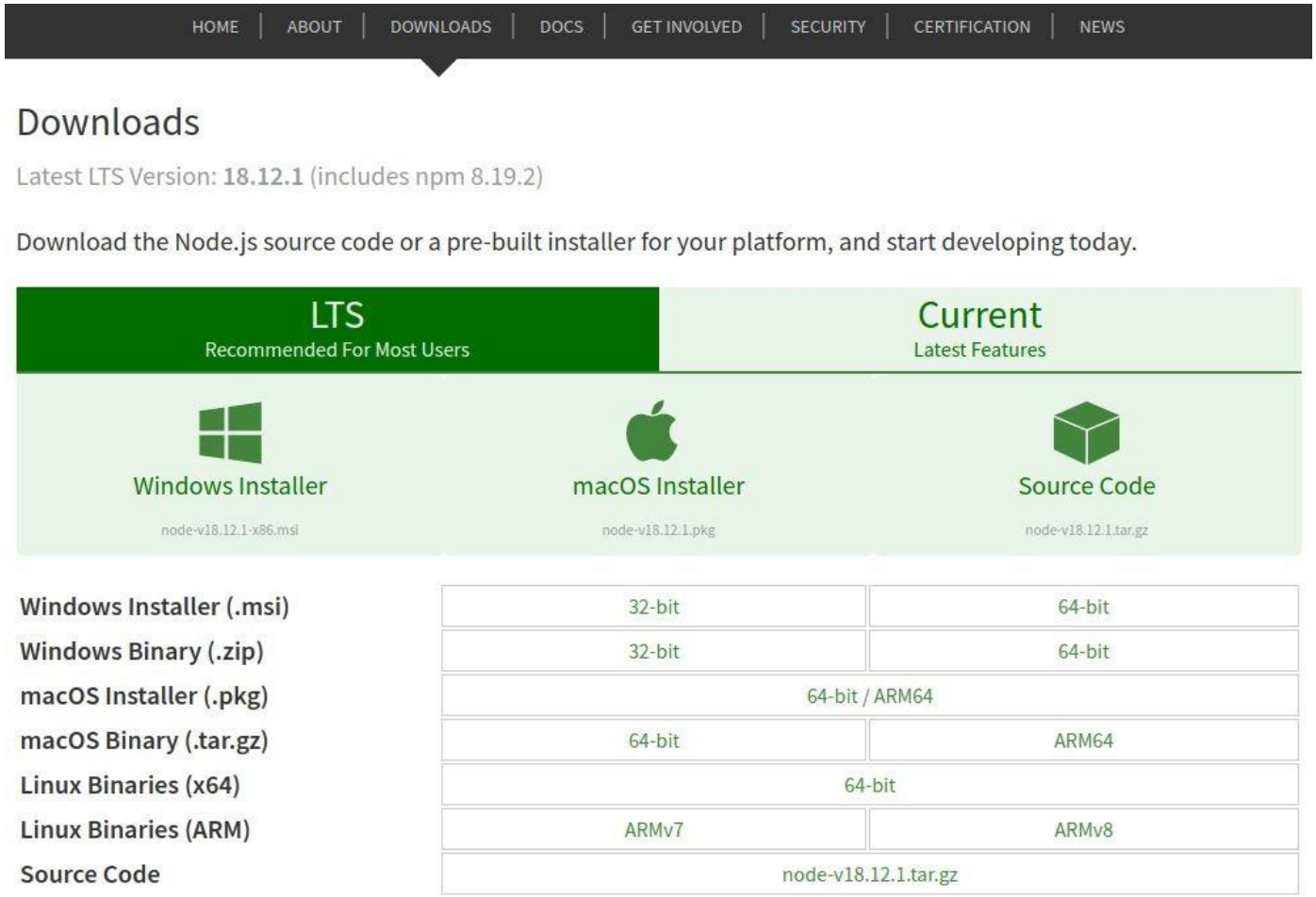
세부 목차

1. Window
2. Linux(Ubuntu)

4. 설치 및 실행

4.1 Windows 설치(1/3)

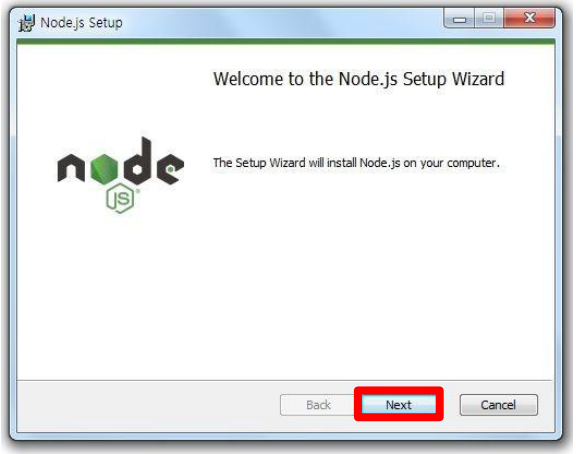
- Node.js의 홈페이지(https://nodejs.org/en/download/)에 접속하여 사용자의 컴퓨터 환경(32bit or 64bit)에 적합한 버전 선택

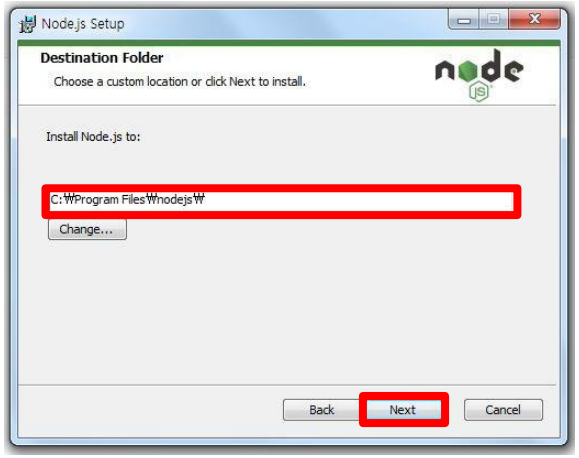


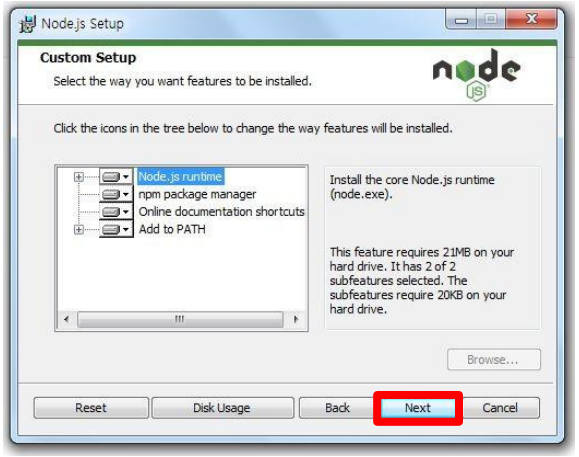
4. 설치 및 실행

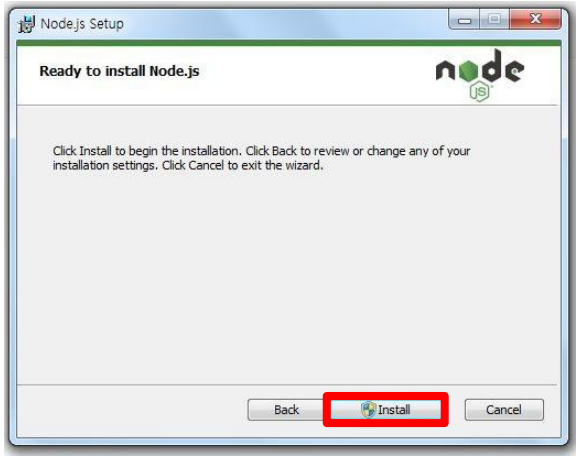
4.1 Windows 설치(2/3)

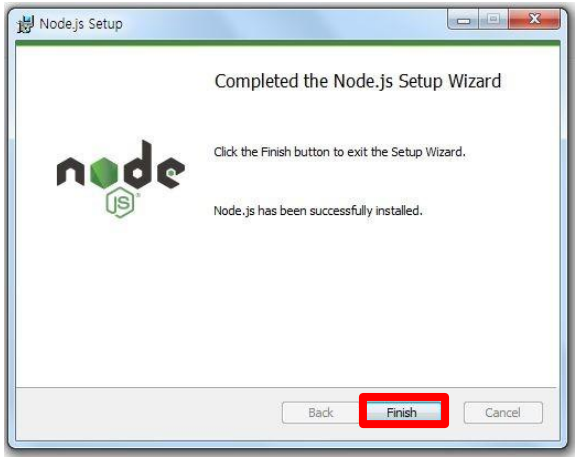
- 경로를 확인 후 'Next' 및 'Install' 버튼을 클릭하여 설치 완료







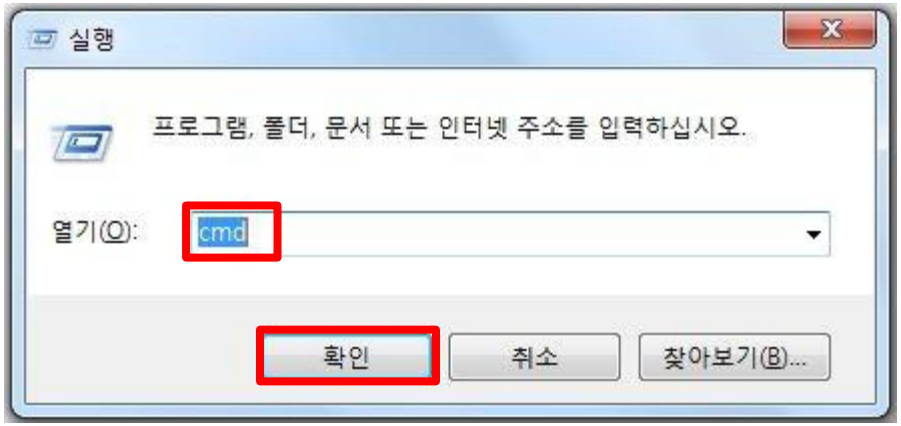


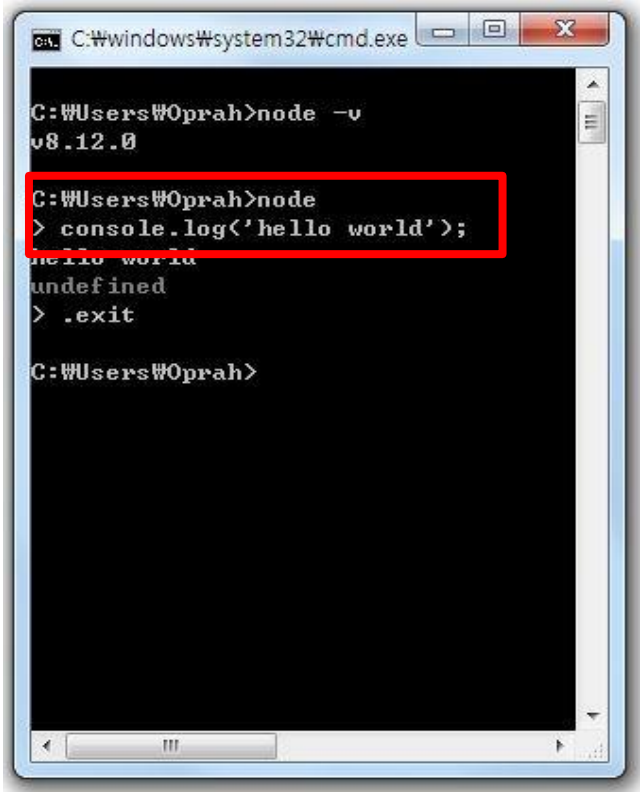


4. 설치 및 실행

4.1 Windows 설치(3/3)

- cmd.exe 실행
- 버전 정보를 확인하는 명령어 'node –v' 를 이용하여 node.js가 정상 설치 확인
- node 명령어를 이용하여 node.js 실행
- node.js의 console.log함수를 이용하여 'hello world' 출력
- .exit를 입력하여 node.js 에서 빠져 나옴

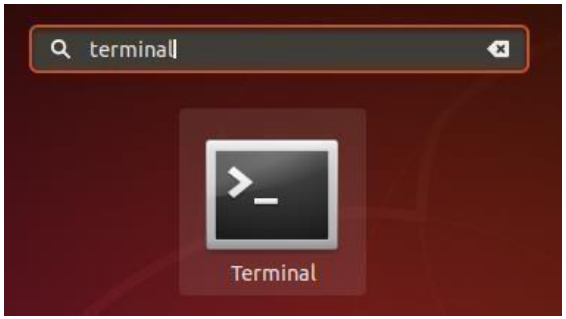




4. 설치 및 실행

4.2 Linux(Ubuntu) (1/2)

- 터미널 실행



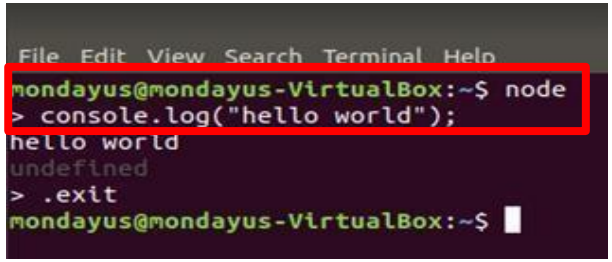
- 'sudo apt-get install node.js'를 입력하여 node.js 설치

```
File  Edit  View  Search  Terminal  Help
mondayus@mondayus-VirtualBox:~$ sudo apt-get install node.js
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'node-jstimezonedetect' for regex 'node.js'
Note, selecting 'node-json-buffer' for regex 'node.js'
Note, selecting 'node-json-localizer' for regex 'node.js'
Note, selecting 'node-json-stable-stringify' for regex 'node.js'
Note, selecting 'node-json-schema-traverse' for regex 'node.js'
Note, selecting 'node-jszip-utils' for regex 'node.js'
Note, selecting 'node-jsv' for regex 'node.js'
Note, selecting 'node-jsonstream' for regex 'node.js'
Note, selecting 'node-json-schema' for regex 'node.js'
Note, selecting 'node-js-cookie' for regex 'node.js'
Note, selecting 'node-jsesc' for regex 'node.js'
Note, selecting 'node-json-stringify-safe' for regex 'node.js'
```

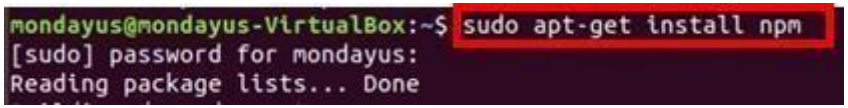
4. 설치 및 실행

4.2 Linux(Ubuntu) (2/2)

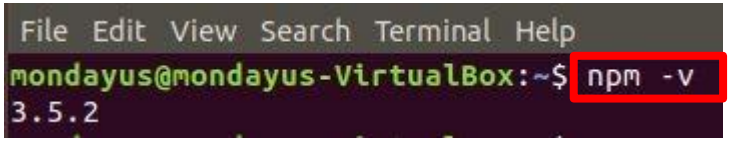
- node 명령어를 이용하여 node.js 실행
- node.js의 console.log함수를 이용하여 'hello world' 출력
- 정상적으로 log가 출력 확인
- .exit를 입력하여 node.js 에서 빠져 나옴



- window's installer와는 달리 linux는 node.js를 설치 하는 동안 npm이 함께 설치 되지 않음
- 'sudo apt-get install npm'을 이용하여 npm 설치



- npm -v 나 npm 명령어를 이용하여 npm이 정상 설치 확인



5. 기능소개

세부 목차

5. 기능소개

5.1.1 모듈 가져오기 (require)

- 프로그램에 필요한 모듈 임포트
  - 수 천개의 다양한 모듈을 간단한 명령어를 통해 npm에서 검색, 설치, 삭제 가능

```
Oprah@Oprah-PC MINGW64 /
$ npm search express
NAME                      | DESCRIPTION
| VER
expres
| 4.1
path-
| 2.4
morga
| 1.9
```

```
Oprah@Oprah-PC MINGW64 ~/Documents/nodeJs
$ npm install express
npm WARN saveError ENOENT: no such file or directory, open 'C:
\Users\Oprah\packa
npm WARN enoent EN
ers\Oprah\package.
npm WARN Oprah No
npm WARN Oprah No
npm WARN Oprah No
npm WARN Oprah No

+ express@4.16.4
updated 1 package
found 0 vulnerabi
```

```
Oprah@Oprah-PC MINGW64 ~/Documents/nodeJs
$ npm uninstall express
npm WARN saveError ENOENT: no such file or directory, open 'C:
\Users\Oprah\package.json'
npm WARN enoent ENOENT: no such file or directory, open 'C:\Us
ers\Oprah\package.json'
npm WARN Oprah No description
npm WARN Oprah No repository field.
npm WARN Oprah No README data
npm WARN Oprah No license field.

removed 1 package and audited 152 packages in 2.236s
found 0 vulnerabilities
```

  - 'require'를 사용하여 다음과 같이 응용프로그램 개발에 필요한 모듈을 외부에서 가져올 수 있음
  - 기본 모듈이 아닐 경우 위와 같이 npm을 통해 install 후 require 가능

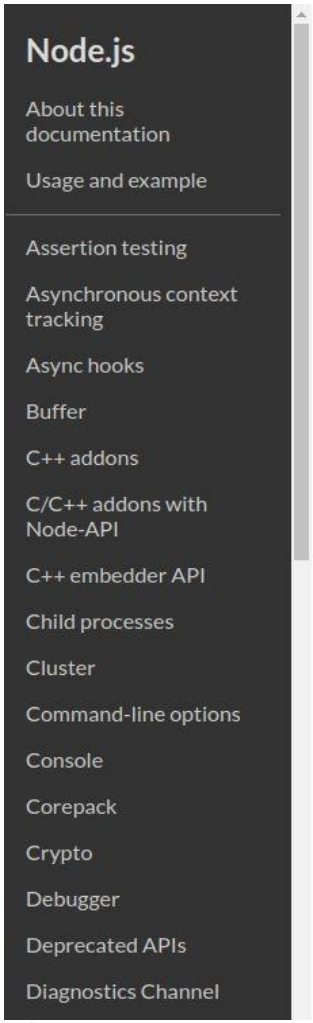
JS main.js

```
1   //'express' 모듈 임포트
2   var expr = require('express');
```

5. 기능소개

5.1.2 기본 모듈 소개

- node.js에서 제공하는 모든 기본 모듈은 API 문서를 통해 확인(http://nodejs.org/api/)



Node.js v19.3.0 documentation

► Other versions | ► Options

- About this documentation
- Usage and example

- Assertion testing
- Asynchronous context tracking
- Async hooks
- Buffer
- C++ addons
- C/C++ addons with Node-API
- C++ embedder API
- Child processes
- Cluster
- Command-line options
- Console
- Corepack
- Crypto
- Debugger
- Deprecated APIs
- Diagnostics Channel
- DNS
- Domain
- Errors
- Events

5. 기능소개

5.1.2 기본 모듈 소개 (File System)

- fs 모듈은 파일 시스템을 다루는데 사용하는 API 제공
- 사용 방법 : var fs = require('fs');
  - 파일 읽기

```
1   var fs = require('fs');
2   var file = 'c:\\Users\\Oprah\\Documents\\nodeJs\\first_server\\test.js';
3
4   //[3] 파일 읽기
5   fs.readFile(file, 'utf8', function(err, data) {
6       console.log("[3] 파일 읽기 : ");
7       console.log("====================");
8       console.log(data);
9       console.log("====================");
10  });
```

```
C:\Users\Oprah\Documents\nodeJs\first_server>node apiTest.js
[3] 파일 읽기 :
====================
console.log("테스트 파일을 작성한다.")
var sum = 1+1;
console.log("1+1 =", sum);내용을 추가합니다.
내용을 추가합니다.
====================
```

  - 파일 쓰기

```
1   var fs = require('fs');
2   var file = 'c:\\Users\\Oprah\\Documents\\nodeJs\\first_server\\test.js';
3
4   //[4] 파일 쓰기 (writeFile / appendFile)
5   var text = '\n내용을 추가합니다.';
6   fs.appendFile(file, text, 'utf8', function(error){
7       console.log('[4] 파일쓰기 완료');
8   });
```

```
C:\Users\Oprah\Documents\nodeJs\first_server>node apiTest.js
[4] 파일쓰기 완료

C:\Users\Oprah\Documents\nodeJs\first_server>
```

5. 기능소개

5.1.2 기본 모듈 소개 (OS)

- os 모듈은 여러 가지 운영 체제 관련 유틸리티 메소드 제공
- 사용 방법 : var os = require('os');

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

console.log("os.arch : ",os.arch()); //운영 체제 CPU 아키텍처를 식별하는 문자열
console.log("os.constants : ",os.constants);  // 오류 코드, 프로세스 신호 등에 일반적으로 사용되는 운영 체제 관련 상수 객체를 반환
console.log("os.cpus : ",os.cpus());//각 논리 CPU 코어에 대한 정보가 들어있는 객체 배열을 반환
console.log("os.endianness : ",os.endianness()); // CPU의 엔디안을 식별하는 문자열을 반환
console.log("os.freemem : ",os.freemem()); //사용 가능한 시스템 메모리의 양을 정수로 반환
console.log("os.getPriority([pid]) : ",os.getPriority(2844)); //pid로 지정된 프로세스의 스케줄링 우선 순위를 리턴.
console.log("os.homedir : ",os.homedir()); //현재 사용자의 홈 디렉토리를 문자열로 반환
console.log("os.hostname : ",os.hostname()); //영 체제의 호스트 이름을 문자열로 반환
console.log("os.networkInterfaces : ",os.networkInterfaces()); //네트워크 주소가 할당 된 네트워크 인터페이스 만 포함하는 객체를 반환
console.log("os.platform : ",os.platform()); //컴파일 시간 동안 설정된 운영 체제 플랫폼을 식별하는 문자열을 반환
console.log("os.release : ",os.release());//운영 체제의 임시 파일에 대한 기본 디렉토리를 지정하는 문자열을 반환
console.log("os.totalmem : ",os.totalmem());//총 시스템 메모리 양을 정수로 반환
console.log("os.type : ",os.type());//uname이 반환 한 운영 체제 이름을 나타내는 문자열을 반환
console.log("os.uptime : ",os.uptime());//시스템 가동 시간을 초 단위로 반환
```

5. 기능소개

5.1.2 기본 모듈 소개 (Console)

- 콘솔 모듈은 웹 브라우저에서 제공하는 JavaScript 콘솔 메커니즘과 유사한, 간단한 디버깅 콘솔을 함
- Console.log (), console.error () 및 console.warn ()과 같은 메소드를 사용하여 모든 Node.js 스트림에 쓸 수 있는 Console 클래스
- process.stdout 및 process.stderr에 기록하도록 구성된 global console 인스턴스이며 global console은 require ('console')를 호출하지 않고 사용
- global console 예제

```
1  //hello world  출력
2  console.log('hello world');
3  console.log('hello %s', 'world');
4  // [Error: Whoops, something bad happened] 출력
5  console.error(new Error('Whoops, something bad happened'));
6  //Danger Will Robinson! Danger! 출력
7  const name = 'Will Robinson';
8  console.warn(`Danger ${name}! Danger!`);
```

- Console class 예제

```
1   const out = getStreamSomehow();
2   const err = getStreamSomehow();
3   const myConsole = new console.Console(out, err);
4
5   //hello world  출력
6   myConsole.log('hello world');
7   myConsole.log('hello %s', 'world');
8   // [Error: Whoops, something bad happened] 출력
9   myConsole.error(new Error('Whoops, something bad happened'));
10  //Danger Will Robinson! Danger! 출력
11  const name = 'Will Robinson';
12  myConsole.warn(`Danger ${name}! Danger!`);
```

5. 기능소개

5.1.2 기본 모듈 소개 (http)

- http 모듈을 이용해 웹 서버 구현이 가능하며, 그러나 일반적인 HTTP applications을 다루기에는 낮은 수준의 모듈이어서 보통은 express나 sails와 같은 web framework 이용
- 사용 방법 : var http = require('http');

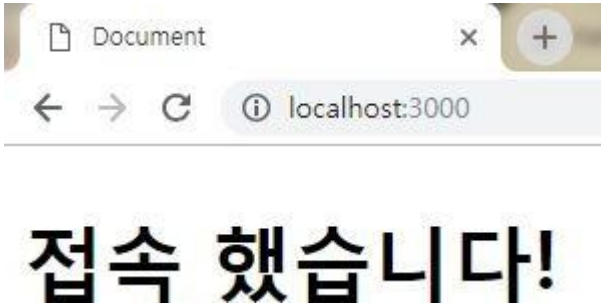
server.js

```
var http=require("http"); //모듈 호출
var fs=require("fs");

//서버 객체 생성
var server=http.createServer();

//접속 감지 이벤트
server.on("connection", function(){
    console.log("client 접속");
});

//클라이언트측에 index.html 출력
server.on("request", function(request, response){
    fs.readFile("index.html", function(error, data){
        if(error){
            console.log(error);
        }else{
            response.end(data);
        }
    });
});

server.on("close", function(){
    console.log("접속 정료");
});

//서버 가동
server.listen(3000, function(){
    console.log("웹 서버가 3000포트에서 가동중...");
});
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <h1>접속 했습니다!</h1>
</body>
</html>
```

```
[Running] node "c:\Users\Oprah\nodeJs_20181103\test_server\server.js"
웹 서버가 3000포트에서 가동중...
client 접속
```



5. 기능소개

5.1.2 기본 모듈 소개 (URL)

- 서버에 들어온 client 요청 url을 파싱하여 서버 내에 존재하는 리소스 접근 원활
- 사용 방법 : var http = require('url');

```
3   var url = require('url');
4
5   //서버 객체 생성
6   var server = http.createServer();
7
8   //클라이언트측에 index.html 출력
9   server.on("request", function (request, response) {
10
11      //요청 url 출력
12      console.log(request.url);
13      //uri 추출
14      var parse = url.parse(request.url);
15      var path = parse.pathname;
16      console.log('path=%s', path);
17
18      //index.html 파일 위치
19      var file = __dirname + '\\index.html';
20      fs.readFile("index.html", function (err, data) {
21          if (err) {
22              console.log(err);
23              //페이지 찾을 수 없음 404
24              response.writeHead(404, { 'Content-Type': 'text/html' });
25          } else {
26              //페이지 찾음 200
27              response.writeHead(200, { 'Content-Type': 'text/html' });
28              //해당 uri에 접근할 경우 client에 메시지 전달.
29              if (path == "/blog") {
30                  fs.appendFile(file,'blog 페이지입니다', 'utf8',function(err,dt){});
31                  response.write(data);
32              } else if (path == '/image') {
33                  fs.appendFile(file,'image 페이지입니다', 'utf8',function(err,dt){});
34                  response.write(data);
35              }
36          }
37          response.end();
38      });
39  });
```

```
[Running] node "c:\Users\Oprah\nodeJ
웹 서버가 3000포트에서 가동중...
/blog
path=/blog
/favicon.ico
path=/favicon.ico
```

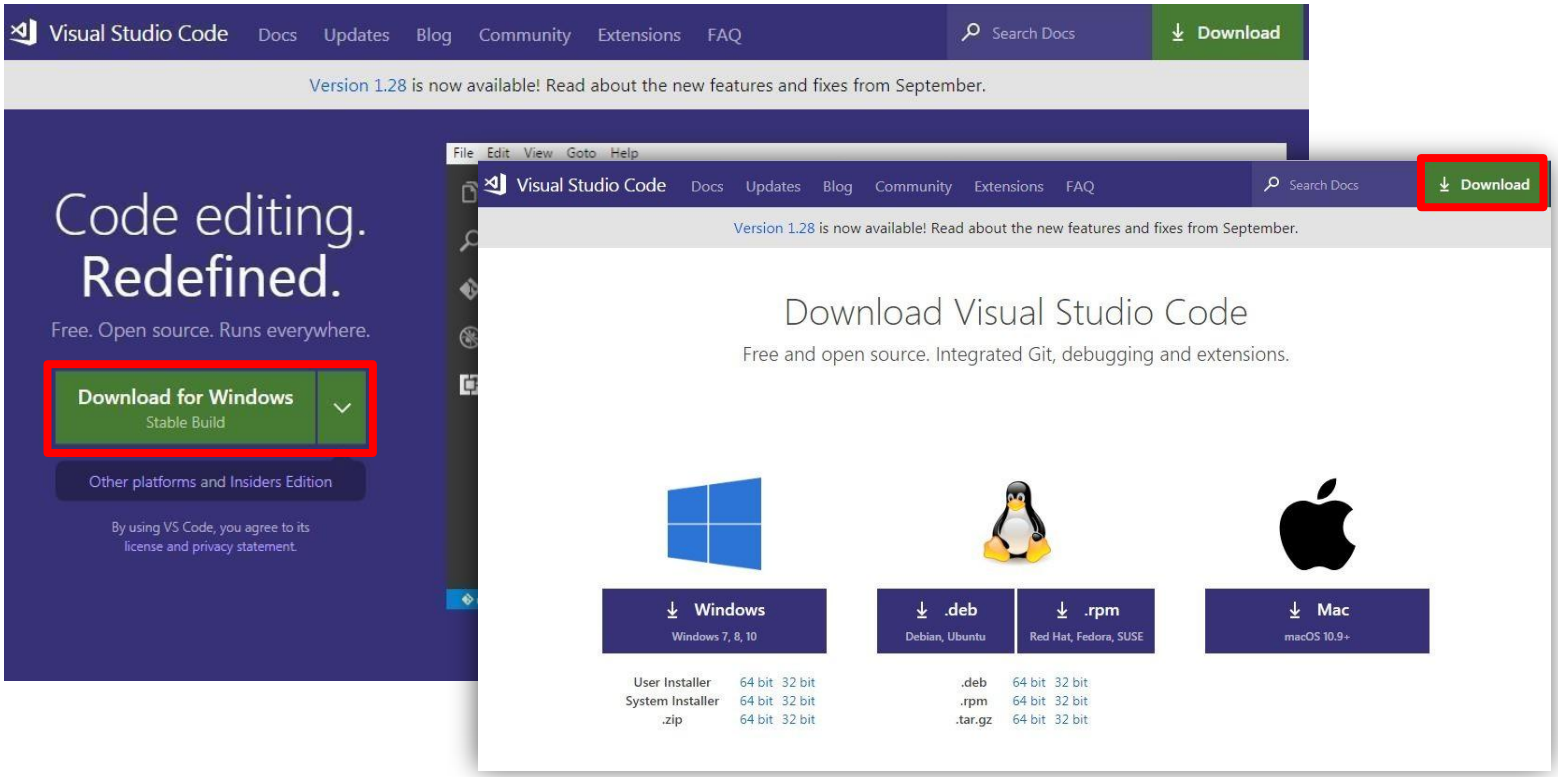
6. 활용예제

세부 목차

6. 활용예제

6.1.1 에디터 소개(1/3)

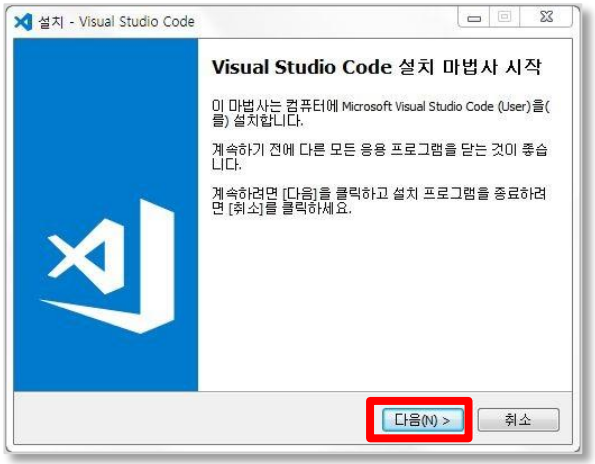
- Visual Studio Code
  - Microsoft사에서 만든 오픈 소스 기반 편집 도구
  - Node.js를 기본적으로 지원하며, 무료 사용
  - 사이트 메인 or 'https://code.visualstudio.com/Download' 에서 설치파일을 받을 수 있음

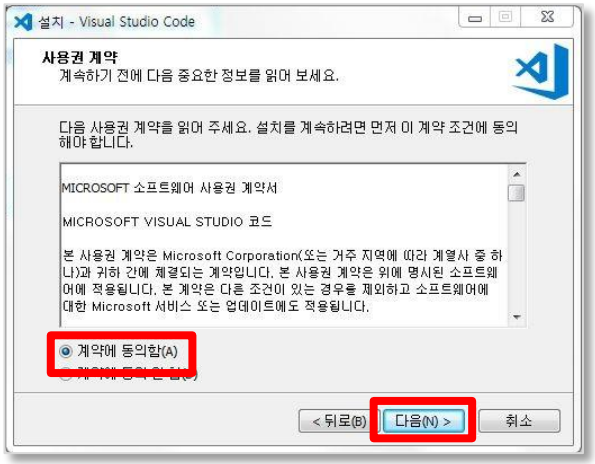


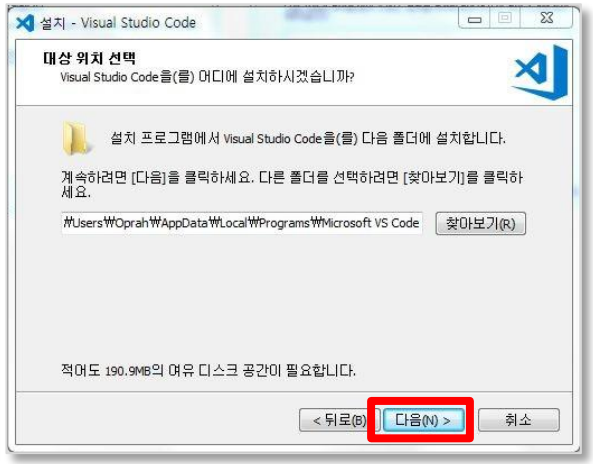
6. 활용예제

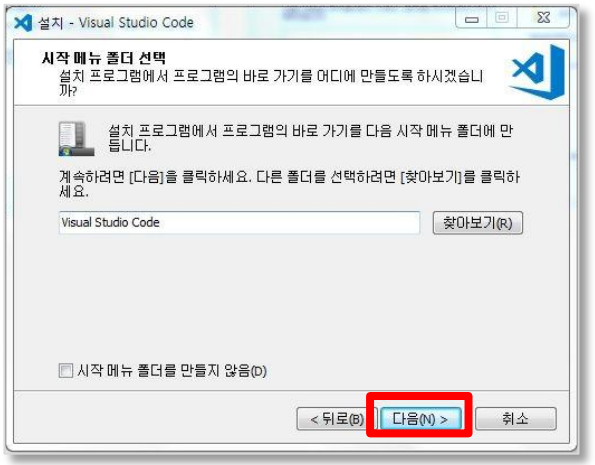
6.1.1 에디터 소개(2/3)

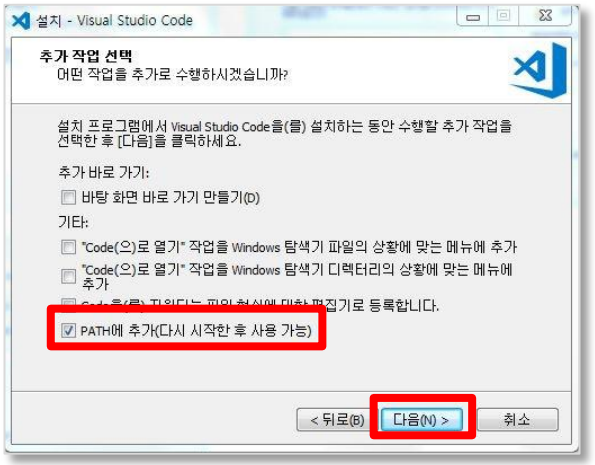
- Visual Studio Code

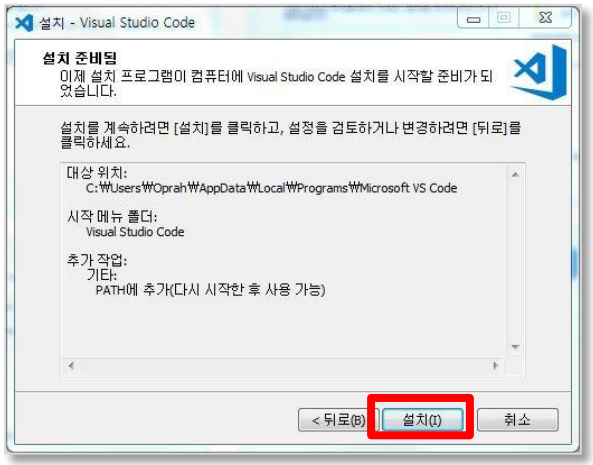








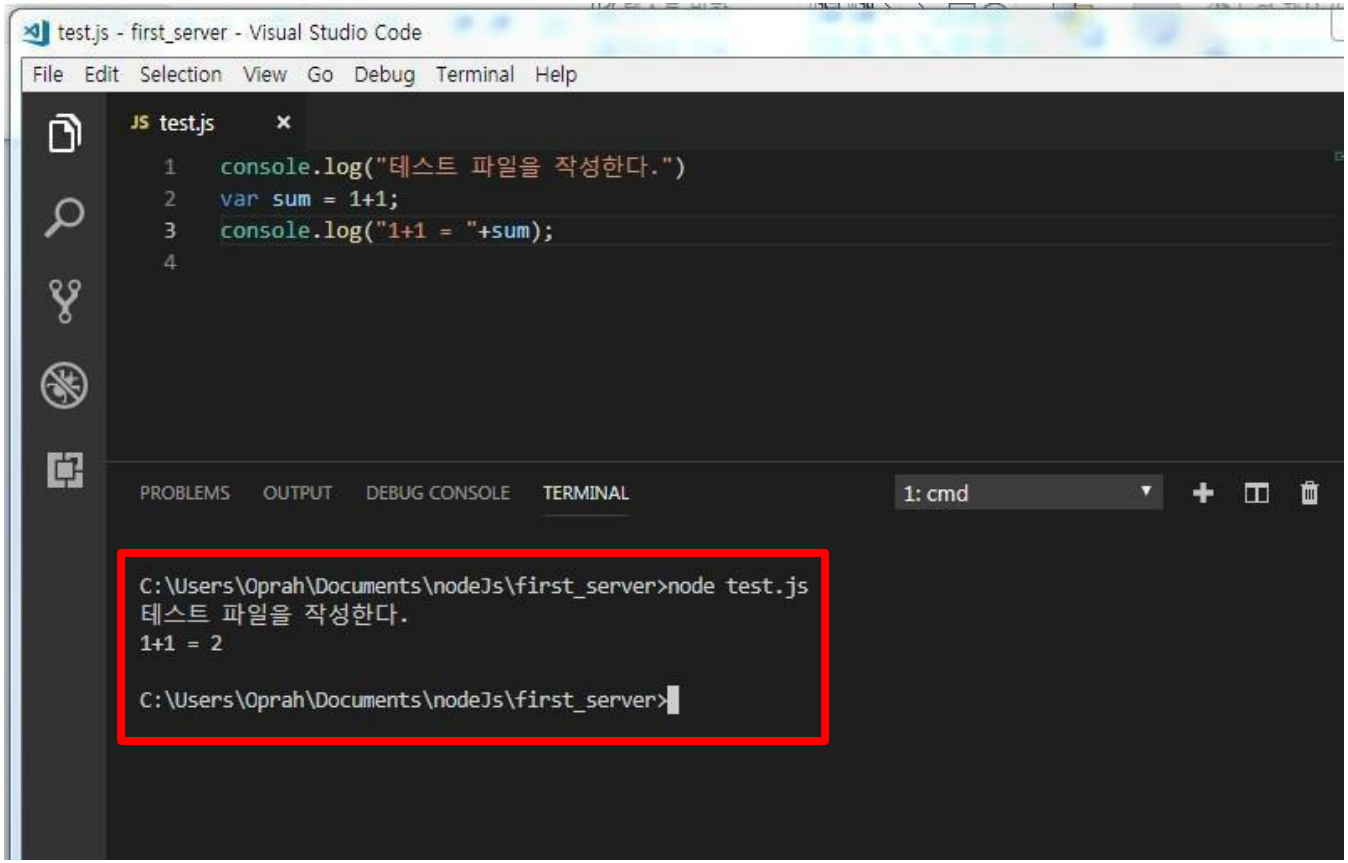




6. 활용예제

6.1.1 에디터 소개(3/3)

- Visual Studio Code
  - 프로그램 내부에서 TERMINAL을 통해 cmd동작을 수행할 수 있으므로 코드 작성화면에서 코딩 후에 TERMINAL에서 바로 수행



6. 활용예제

6.1.2 express를 활용한 사이트 띄우기

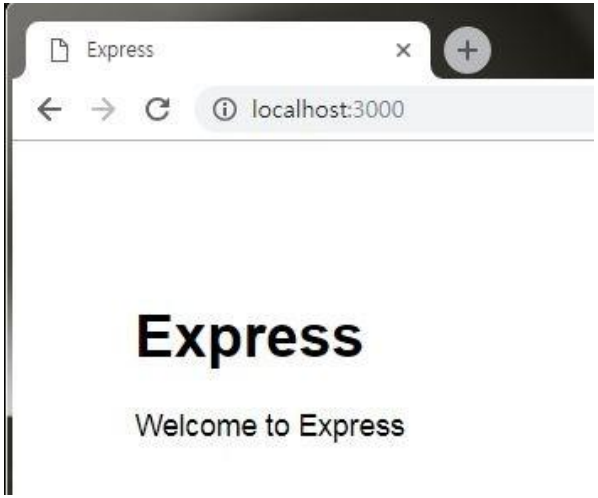
- express를 활용하여 사이트를 만들기 위해 우선 다음 명령어를 이용하여 express 설치
  - npm install -g express-generator
- stylus를 사용하는 express 사이트 기틀을 'express_ex'디렉터리 하위 생성
  - express -c stylus express_ex
- express_ex 하위에, 다음과 같이 app.js, bin, package.json 등 기본적인 사이트 형성에 필요한 디렉터리와 파일이 생성 확인

```
C:\Users\Oprah\nodeJs_20181103\express_ex>dir
 C 드라이브의 볼륨에는 이름이 없습니다.
 볼륨 일련 번호: 1446-5A82

 C:\Users\Oprah\nodeJs_20181103\express_ex 디렉터리

2018-11-13  오후 11:08    <DIR>          .
2018-11-13  오후 11:08    <DIR>          ..
2018-11-13  오후 11:08             1,167 app.js
2018-11-13  오후 11:08    <DIR>          bin
2018-11-13  오후 11:08               322 package.json
2018-11-13  오후 11:08    <DIR>          public
2018-11-13  오후 11:08    <DIR>          routes
2018-11-13  오후 11:08    <DIR>          views
               2개 파일               1,489 바이트
               6개 디렉터리  104,010,170,368 바이트 남음
```

6. 활용예제

6.1.3 사이트 띄워보기(1/7)

- 응용프로그램 부팅을 위해 express_ex 디렉터리 내에서 다음 명령 실행
  - **npm install**
- 다음 명령어를 이용하여 응용프로그램 부팅
  - window : **set DEBUG=express_ex:* & npm start**
  - XOS or LINUX : **DEBUG=express_ex:* npm start**
- localhost:3000 에 접근했을 때, 다음과 같은 화면이 보이면, 정상적으로 설치 및 실행 확인



6. 활용예제

6.1.3 사이트 띄워보기(2/7)

- 일반적으로 Node.js 사이트를 개발 시 변경사항을 적용 할 때마다 응용 프로그램을 다시 시작해야 하지만, nodemon을 이용하면 변경사항이 감지 될 때마다 응용프로그램을 자동으로 로드 하므로 응용프로그램을 재시작 할 필요가 없고, 다음을 이용하여 nodemon 설치
  - npm install -g nodemon

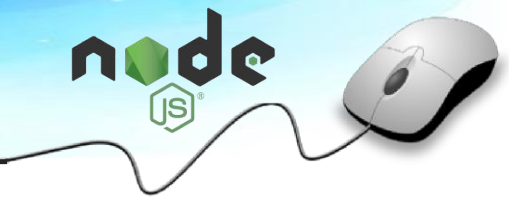
```
C:\Users\Oprah\nodeJs_20181103\express_ex>npm install -g nodemon
[        ...........] - extract:nodemon: verb lock using C:\Users\Opra
```

- 다음 명령어를 이용하여 서버를 시작하고, localhost:3000에 접속하면 결과 변함 없음
  - set DEBUG=express_ex:* & nodemon start

```
C:\Users\Oprah\nodeJs_20181103\express_ex>set DEBUG=express_ex:* & nodemon start
[nodemon] 1.18.6
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node ./bin/www start`
```

6. 활용예제

6.1.3 사이트 띄워보기(3/7)



- localhost:3000의 기본 화면을 변경하기 위해 /views/layout.jade 에 있는 기본 소스 대신 다음 소스 입력

```
doctype
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
    link(rel='stylesheet', href='/stylesheets/chunkfive-fontface.css')
  body
    header
      nav
        ul
          li
            a(href="/") Home
          li
            a(href="/about") About
          li
            a(href="/contact") Contact
    section#wrapper
      block content
      footer
        section.css-table
          section.four-column
            section.cell
              p jade란? <br /> jade는 node.js용으로 만들어진 view 템플릿 엔진 <br /> 홈페이지:ht
            section.cell
              p stylus란? <br />CSS를 효율적이고 동적으로 표현할 수있는 언어<br /> 홈페이지 : ht

```

6. 활용예제

6.1.3 사이트 띄워보기(4/7)

- 링크 이동을 위해 /views/layout.jade와 동일 위치에 layout.jade를 복사한 후 이름을 각각 about.jade, contact.jade로 변경하고 소스를 다음과 같이 수정

```
doctype
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
    link(rel='stylesheet', href='/stylesheets/chunkfive-fontface.css')
  body
    header
      nav
        ul
          li
            a(href="/") Home
          li
            a(href="/about") About
          li
            a(href="/contact") Contact
    section#wrapper
        block content
        footer
          section.css-table
            section.four-column
              section.cell
                p About 페이지이다.

```

6. 활용예제

6.1.3 사이트 띄워보기(5/7)

- css적용을 위해 /stylesheets/style.styl 에 있는 기본 소스 대신 다음 소스 입력

```
body
  font 62.5%/1.5  Helvetica, Arial, "Lucida Grande", "Lucida Sans", Tahoma, Verdana, sans-serif
  text-align center
  background #6a9fca

#wrapper
  width 920px
  text-align left
  margin-left auto
  margin-right auto
  background #fff
  padding 20px
  border-bottom-radius(15px)
```

- 링크 이동을 위해 /routes/index.js 에 있는 기본 소스 대신 다음 소스 입력

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

router.get('/about', function(req, res, next) {
  res.render('about', { title: 'About' });
});

router.get('/contact', function(req, res, next) {
  res.render('contact', { title: 'Contact' });
});

module.exports = router;

```

6. 활용예제

6.1.3 사이트 띄워보기(6/7)

- nodemon에 의해 변경한 소스는 자동 반영 되므로 접속되어 있던 localhost:3000 페이지에서 F5를 눌러 페이지 리프레시 후 메인 화면이 아래와 같이 변경 확인

6. 활용예제

6.1.3 사이트 띄워보기(7/7)

- localhost:3000/about 으로 접근하거나 About링크를 클릭하면 다음 페이지 이동

- localhost:3000/contact 으로 접근하거나 Contact링크를 클릭하면 다음 페이지 이동

7. FAQ

Q **Node.js를 웹 서버라고 말할 수있나요?**

A Node.js는 Java의 JRE와 같은 런타임 환경입니다. Node.js는 웹 이외의 응용 프로그램에서 점점 더 많이 사용되고 있습니다. Node.js에는 웹 서버를 쉽게 만들 수 있는 매우 높은 수준의 네이티브 함수(http.CreateServer (...))가 있으나, 이를 추가하지 않는 한은 웹 서버 기능이 없습니다. 따라서 Node.js 자체는 웹 서버가 아닙니다.

Q **Node.js를 사용하여 어떤 종류의 응용 프로그램을 만들 수 있나요?**

A 사실, Node.js로 무엇이든 만들 수 있지만 그것이 항상 현명한 선택은 아닙니다. Node.js가 능가하는 곳은 여러 I / O 작업과 실시간 시스템을 다루는 곳입니다. 이벤트 기반의 특성으로 인해 다른 언어나 프레임워크보다 훨씬 효율적이고 빠르게 처리 할 수 있습니다.

7. FAQ

Q **어떤 경우에 Node.js를 사용하면 안되나요?**

A Node.js는 CPU가 많은 소프트웨어와 잘 맞지 않습니다. 장기 실행 계산은 들어오는 요청을 차단하므로 성능이 저하 될 수 있습니다.

Q **Node.js를 기계 학습에 사용할 수 있나요?**

A Node.js는 무엇이든 사용할 수 있지만 기계 학습을위한 최상의 도구는 아닙니다. 단일 스레드 특성으로 인해 노드 응용 프로그램은 특히 기계 학습과 같은 과중한 작업 계산에 특히 좋지 않습니다. 경우에 따라 Node.js는 기본 회귀, 분류 또는 피쳐 추출과 같은 부분에서 나쁘지 않은 성과를 보일 수는 있습니다. 그러나 기계 학습관련 응용프로그램을 개발하려면 Tensorflow 또는 Theano와 함께 Python을 고려해 보는 것이 좋습니다.

8. 용어정리

용어	설 명
Chrome V8 JavaScript	V8 엔진(Chrome V8)은 웹 브라우저를 만드는 데 기반을 제공하는 오픈소스 응용 프로그램 프레임워크, 구글 크롬 브라우저와 안드로이드 브라우저에 탑재, V8로 줄여 불리기도 하며, 현재 라스백이 책임 프로그래머 ECMAScript(ECMA – 262) 3rd Edition 규격의 C++로 작성되었으며, 독립적으로 실행이 가능, 또한 C++로 작성된 응용 프로그램의 일부로 작동됨
콜백	프로그래밍에서 콜백(callback)은 다른 코드의 인수로서 넘겨주는 실행 가능한 코드, 콜백을 넘겨받는 코드는 이 콜백을 필요에 따라 즉시 실행 및 나중 실행 가능
test suite	소프트웨어 개발에서 일반적으로 '검증 스위트'로 불리는 테스트 스위트는 소프트웨어 프로그램이 특정 동작 집합을 갖고 있음을 보여주기 위해 테스트하는데 사용되는 테스트 사례 모음, 테스트 슈트는 종종 테스트 케이스의 각 컬렉션에 대한 상세한 지침이나 목표 및 테스트 중에 사용될 시스템 구성에 대한 정보를 포함 , 테스트 케이스 그룹은 전제 조건 또는 단계 및 다음 테스트에 대한 설명 포함
스트리밍 데이터	스트리밍 데이터는 수천 개의 데이터 소스에서 연속적으로 생성되는 데이터로서, 보통 데이터 레코드를 작은 크기(KB 단위)로 동시에 전송, 스트리밍 데이터에는 모바일이나 웹 애플리케이션을 사용하는 고객이 생성하는 로그 파일 , 전자 상거래 구매 , 게임 내 플레이어 활동, 소셜 네트워크의 정보, 주식 거래소, 지리공간 서비스, 연결된 디바이스의 텔레메트리, 데이터 센터의 계측 등 다양한 데이터가 포함

Open Source Software Installation & Application Guide