

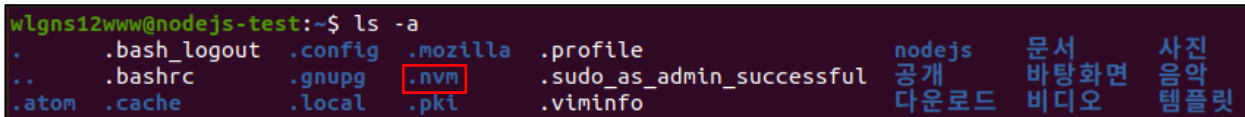
Node.js 설치 및 간단예제

Node.js 설치

- NVM(Node.js Version Manager)을 통해서 Node.js를 설치
- NVM은 이름과 같이 Node.js의 버전을 관리해주는 기능을 한다.
- 터미널에 아래와 같이 입력
: sudo curl -o- <https://raw.githubusercontent.com/creationix/nvm/v0.33.8/install.sh> | bash

- ✓ 만약 curl이 설치되어 있지 않다는 Error가 뜰 시에는 터미널에 아래와 같이 입력
: sudo apt install curl

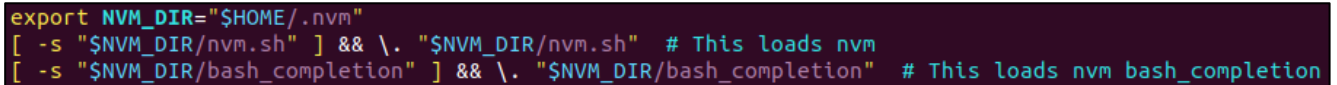
- NVM설치 완료확인 1)
사용자의 홈 디렉터리에 .nvm라는 숨김 디렉터리가 생성되어 있음



```
wlgns12www@nodejs-test:~$ ls -a
.      .bash_logout  .config  .mozilla  .profile
..     .bashrc      .gnupg   .nvm      .sudo_as_admin_successful
.atom  .cache       .local   .pkg      .viminfo
```

<그림> nvm이 정상적으로 설치된 모습1

- NVM설치 완료확인 2)
사용자의 홈 디렉터리의 .bashrc라는 파일의 마지막 부분에 추가내용이 입력되어 있음

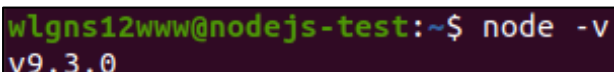


```
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
[ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion
```

<그림> nvm이 정상적으로 설치된 모습2

- .bashrc에 추가된 내용을 \$PATH 환경변수에 적용하기 위해 터미널에 아래와 같이 입력
: source ~/.bashrc
- 9.3.0 버전의 node.js 설치를 위해 터미널에 아래와 같이 입력
: nvm install 9.3.0

- node.js설치 완료확인 1)
node명령을 통해 node.js 버전을 확인가능
: node -v



```
wlgns12www@nodejs-test:~$ node -v
v9.3.0
```

<그림> 9.3.0 버전의 node.js가 설치된 모습

- node.js설치 완료확인 2)

\$PATH 환경변수에 nvm을 통해 node.js 9.3.0의 디렉터리 경로가 등록된 것을 확인가능

: echo \$PATH

```
wlgns12www@nodejs-test:~$ echo $PATH
/home/wlgns12www/.nvm/versions/node/v9.3.0/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

<그림> \$PATH 환경변수에 node.js의 디렉터리 경로가 등록된 모습

예제1) 터미널에 간단한 메시지 출력

1. vi hello.js
2. 아래 박스의 내용과 같이 hello.js에 입력 후 저장

```
"use strict";

console.log("Hello, node.js");
```

3. 입력된 hello.js를 node명령으로 실행
: node hello.js

✓ 실행화면

```
wlgns12www@nodejs-test:~/nodejs$ ls
hello.js
wlgns12www@nodejs-test:~/nodejs$ node hello.js
Hello, Universe!
```

<그림> hello.js가 정상적으로 실행된 화면

예제2) forEach와 화살표함수 사용해보기

1. vi message.js
2. 아래 박스의 내용과 같이 message.js에 입력 후 저장

```
"use strict";

let messages = [
    "You will make it!",
    "Just run with the code!"
];

messages.forEach(message => console.log(message));
```

3. 입력된 message.js를 node명령으로 실행
: node message.js

✓ 실행화면

```
wlgns12www@nodejs-test:~/nodejs$ ls
hello.js  message.js
wlgns12www@nodejs-test:~/nodejs$ node message.js
You will make it!
Just run with the code!
```

<그림> message.js가 정상적으로 실행된 화면

예제3) 문자열 보간 사용해보기

1. vi interpolation.js
2. 아래 박스의 내용과 같이 interpolation.js에 입력 후 저장

```
"use strict";

let num = [1, 2, 3];
let str = "node.js";
let a = 123.123;
let b = -123;

console.log("line %d : Hello, %s", num[0], str);
console.log("line %d : %f", num[1], a);
console.log("line %d : %d", num[2], b);
```

3. 입력된 interpolation.js를 node명령으로 실행

✓ 실행화면

```
wlgns12www@nodejs-test:~/nodejs$ ls
hello.js  interpolation.js  message.js
wlgns12www@nodejs-test:~/nodejs$ node interpolation.js
line 1 : Hello, node.js
line 2 : 123.123
line 3 : -123
```

<그림> interpolation.js가 정상적으로 실행된 화면

Tip 1) 엄격모드

- JavaScript 엄격모드
: node.js는 JavaScript엔진으로 제작되어 JavaScript의 엄격모드를 사용가능하다.
: 엄격모드란 웹 브라우저나 node.js에서 그냥 넘어갈 만한 JavaScript의 오류들을 캐치한다.
: 엄격모드로 실행하기 위해선 파일 내의 가장 위쪽에 "use strict"; 를 추가한다.
- 엄격모드로 발견할 수 있는 오류들
 1. 생성자 없이 만든 전역변수들 var, let, const같은 생성자 없이는 전역변수를 만들 수 없다.
 2. 할당될 수 없는 변수에 변수 할당
 3. 객체 리터럴에서 고유하지 않은 함수 매개변수 이름 또는 속성 이름 사용

Tip 2) var, let, const의 차이점

- var

: var로 생성된 변수는 한 블록에서만 사용가능한 [지역변수](#).

```
function counter()
{
    for(var i=0; i<10; i++)
        console.log("%d", i);
}
counter();
```

`console.log("%d", i); // ERROR발생이유 : i변수는 counter()함수에서 var로 생성되었으므로 함수내에서만 존재가능`

- let

: let로 생성된 변수는 파일 전체에서 사용가능한 [전역변수](#).

: let으로 생성된 변수는 [재할당과 재선언이 가능하다](#).

- const

: const로 생성된 변수는 파일 전체에서 사용가능한 [전역변수](#).

: const로 생성된 변수는 [재할당과 재선언이 불가능하다](#).

참고문헌

- NODEJS로 프로그래밍 시작하기 : 2020.01.31 : 조나단 웨슬러(지은이), 김성준(옮긴이) : (주)에어컨출판