

[Linux] 명령어 정리 (Ubuntu 사용)

devyang97 · 2020년 6월 6일

linux



디렉토리 관련 명령어

디렉토리 목록 확인

```
$ ls
$ ls -al : 김춰진 파일은 .이 붙는다. a 옵션을 사용하면 해당 파일을 확인할 수 있다.
$ ls -l
```

새 디렉토리 생성

```
$ mkdir [디렉토리명]
$ mkdir -p [디렉토리명/디렉토리명/디렉토리명...]: 여러 디렉토리 생성
```

디렉토리 이동

```
$ cd [디렉토리명]
$ cd .. : 부모 디렉토리로 이동
```

tip) 디렉토리명이 너무 길 때, 조금만 쓰고 tab키 누르면 자동완성

디렉토리 삭제

```
$ rm -r [디렉토리명]
: -r (remove directories and their contents recursively; 해당 디렉토리 아래 있는 내용들도 삭제한다.)
```

파일 관련 명령어

비어있는 파일 생성

```
$ touch [파일명]
```

파일 삭제

```
$ rm [파일명]
```

파일 복사

```
$ cp [파일위치 및 파일이름] [목적지 파일위치 및 파일 이름]
```

파일 이동 (파일 이름을 바꿀 때에도 사용)

```
$ mv [파일위치 및 파일이름] [목적지 파일위치 및 파일 이름]
$ mv [원래 파일 이름] [바꾸고 싶은 파일 이름]
```

파일 만들고 편집 (nano 에디터)

```
$ nano [파일명] : 새 파일 작성 or 존재하는 파일 수정
```

파일 내용 보기

```
$ cat [파일명]
```

파일 위치 검색

```
$ locate *.log : (확장자가 .log인 파일 찾기)
: 디렉토리를 뒤지는게 아니라 데이터베이스(mlocate)를 뒤져서 찾는다.
```

```
$ find . *.log : 디렉토리 뒤짐
```

```
$ whereis ls
```

```
$ whereis rm
```

: 실행파일 위치 찾기

디렉토리 관련 명령어
디렉토리 목록 확인
새 디렉토리 생성
디렉토리 이동
디렉토리 삭제
파일 관련 명령어
비어있는 파일 생성
파일 삭제
파일 복사
파일 이동 (파일 이름을 바꿀 때
에도 사용)
파일 만들고 편집 (nano 에디
터)
파일 내용 보기
파일 위치 검색
현재 위치 확인
명령어 내보내기
명령어 도움말 확인
패키지 매니저 (Package
Manager)
패키지 목록 업데이트
패키지 찾기
패키지 설치
포킬체스터
백그라운드 프로세스 목록까지
확인
프로세스 강제 종료
그 외 프로세스 목록 확인 명령
어
백그라운드 실행
백그라운드로 돌아가는 프로그
램 확인
foreground로 돌아가기
백그라운드 프로그램 죽이기
daemon: 항상 실행되고 있다.
테스트를 위해 apache2 웹서버
설치
apache2 켜기
잘 켜진지 확인
apache2 끄기
꺼져있는지 확인
cron (정기적으로 실행시키고 싶
은 작업이 있을 때)
단축키 만들기
.bashrc 파일
다중 사용자
sudo (super user do)
사용자 주가
sudo 사용할 수 있도록 변경
비밀번호 설정
로그인 (su 사용)
permission (권한)
권한 변경 (chmod)
execute
directory
Numerical permissions
'=' 사용
IP 주소 확인
웹서버
웹서버 설치 (apache2)
웹서버 시작, 중지, 재시작
일을 통해 웹서버 접속 (불에서
웹브라우징) : elinks 사용
웹서버의 설정
웹서버의 로그 확인
SSH
ssh 심화수업 - 비밀번호 입력없
이 로그인하기
포트 (port)
도메인과 DNS (Domain Name
System)
서브도메인
rsync
등장방법 예시
컴퓨터와 컴퓨터 사이에 전송
암호화
대칭적인 방식
비대칭적인 방식
RSA 등장방식

현재 위치 확인

◀ more

명령창 내용 삭제

```
$ clear
```

명령어 도움말 확인

```
$ [명령어] --help  
$ man [명령어] : /[찾고 싶은 단어] 사용해서 단어 찾기 가능, 그 상태로 n을 누르면 다음 단어 찾기
```

패키지 매니저 (Package Manager)

- 기본으로 내장되어 있는 패키지(프로그램)가 아닌 새로운 패키지(프로그램)를 설치하려고 할 때 도와주는 소프트웨어 (안드로이드의 구글플레이, iOS의 앱스토어 같은 ..)
- apt, yum 등이 있음.

패키지 목록 업데이트

```
$ apt-get update
```

- 패키지 매니저를 통해 설치할 수 있는 패키지 목록들을 업데이트한다. (설치하기 전에)
- 패키지가 설치되는게 아니라 패키지 목록들이 업데이트 되는 것.

패키지 찾기

```
$ apt-cache search [패키지명]
```

- 저장된 패키지 목록 중에 해당 패키지 찾기..?
- 관련된 패키지 목록까지 나온다.

패키지 설치

```
$ apt-get install [패키지명]
```

패키지 업그레이드

```
$ apt-get upgrade  
$ apt-get upgrade [패키지명]
```

패키지 삭제

```
$ apt-get remove [패키지명]
```

패키지를 설치하는 순서 (항상 이 순서를 따르는게 좋음.)

- 패키지 목록 업데이트 (apt-get update)
- 패키지 설치 (apt-get install)

다운로드

파일을 다운로드 (wget 사용)

```
$ wget -O [저장할 파일명] [다운로드 url]
```

소스코드 다운로드 (git 사용)

- git 설치

```
$ apt-get install git
```
- 소스코드 다운

```
$ git clone [소스코드 url] [디렉토리명]
```

: 명시한 디렉토리에 소스코드 다운받는다.

why CLI?

1. 순차적 실행 (using semicolon)

ex) \$ mkdir why; cd why;

- 언제 유용한가?
- 하나하나의 명령들이 1000시간이 걸린다고 할 때, 명령어 하나하나씩 치는 것과 여러개를 한번에 치는 것은 차이가 있음.
(여러 개 명령어를 한 번에 치면 알아서 마지막 결과만 나타남.)

2. 파일라인

- 명령어의 연결
- 어떤 프로세스의 출력을 다른 프로세스의 입력으로 ...
- 원하는 정보가 포함되어 있는 행을 출력
- ex)

```
$ grep linux linux.txt · 'linux'가 포함된 행 출력
```

```
$ ls --help | grep sort : ls --help에서 'sort'가 포함된 행을 출력 (ls --help의 출력을 grep의 입력으로)
$ ls --help | grep sort | grep file : ls --help에서 'sort'와 'file'이 포함된 행을 출력
```

IO Redirection

```
1. output
$ ls -l > result.txt : result.txt 확인해보면 ls -l 출력을 담겨있음.

2. error (에러 결과를 저장하려면? '2>' 를 사용한다. (standard error를 가리킴))
ex) 해당 디렉토리에 rename2.txt가 없을 때
$ rm rename2.txt 2> error.log : error.log에 에러 내용이 담겨있음.

3. input
$ cat hello.txt : cat의 command-line arguments로써 역할
$ cat < hello.txt : hello.txt 내용을 standard input으로..
$ head -n1 < linux.txt > one.txt : linux.txt 내용을 input, one.txt에 출력물을 저장
```

디렉토리 구조

- /bin: 사용자가 사용하는 명령어 모음
- /sbin: 관리자가 사용하는 명령어 모음
- /etc: 프로그램 설정을 관리하는 디렉토리
- /etc/init.d: daemon의 목적을 가진 프로그램들 있음.
- /var: 내용이 바뀔 수 있는 파일들 모음
- /tmp: 임시파일들, 컴퓨터가 깨지면 날아간다.
- /home: 사용자들의 파일들이 저장되는 디렉토리
- /lib: /bin과 /sbin에 있는 프로그램들이 사용하는 라이브러리 모음
- /usr: 유저가 다운받은 프로그램들 저장..

프로세스 관련 명령어

프로세스 목록 확인

```
$ ps
```

백그라운드 프로세스 목록까지 확인

```
$ ps aux
$ ps aux | grep apache2 : 'ps aux'의 출력에서 apache2가 포함된 행 출력
```

프로세스 강제 종료

```
$ kill [pid]
```

그 외 프로세스 목록 확인 명령어

```
$ top
$ htop (top과 비슷하지만, 시각적인 면에서 더 좋음)
```

백그라운드 실행

ex) nano 에디터 실행하다가 ctrl + z -> 백그라운드에서 돌아가도록 등록

백그라운드로 돌아가는 프로그램 확인

```
$ jobs
: +가 불어있는 프로그램 -> fg 실행했을 때, 실행되는 프로그램
: -가 불어있는 프로그램 -> fg 실행했을 때, + 다음으로 실행될 프로그램
```

foreground로 돌아가기

```
$ fg
$ fg %번호 : 해당 번호 프로그램으로 돌아간다.
```

백그라운드 프로그램 죽이기

```
$ kill
$ kill -9 %번호 : 강제종료
```

daemon: 항상 실행되고 있다.

테스트를 위해 apache2 웹서버 설치

```
$ sudo apt-get install apache2
```

설치 후 /etc/init.d에 가보면 apache2가 있다.

apache2 켜기

```
$ sudo service apache2 start
```

잘 커진지 확인

```
$ ps aux | grep apache2
```

apache2 끄기

```
$ sudo service apache2 stop
```

꺼져있는지 확인

```
$ ps aux | grep apache2 : 목록에 나타나지 않음.
```

cron (정기적으로 실행시키고 싶은 작업이 있을 때)

```
$ crontab -e : 처음 실행했을 땐, 에디터 골라야 함
```

```
m h dom mon dow command
```

- m: 분 주기
ex) 10: 매 시간 10분에 한 번
ex) */1: 1분에 한 번
- h: 시간 주기
ex) *: 시간에 상관 없이
ex) 1: 매 1시에
- dom: day of month
ex) 24: 매 달 24일에 한 번
- mon: month
- dow: day of week (요일)
- command: 주기적으로 실행할 명령어

테스트

1. command 자리에 `date >> date.log` 입력 (>>는 해당 파일 아래로 쭉 출력결과가 쌓임)
2. `$ tail -f date.log` (꼬리에 새로운 텍스트가 추가 될 때, 실시간으로 표시해줌.)

에러가 난 경우도 로그로 남기는게 좋다.

```
*/1 * * * * date >> date.log 2>&1
: 표준 출력을 date.log에 넘기고 표준 에러가 발생하면 표준 출력으로 리다이렉션
: date.log에 에러도 함께 저장됨
```

단축키 만들기

```
$ alias l='ls -al'
-> l을 입력하면 ls -al의 출력이 나옴
```

.bashrc 파일

bash가 시작되었을 때 실행됨.

다중 사용자

```
$ id : 현재 접속한 계정의 정보 확인
$ who : 이 컴퓨터에 누가 접속해있는가
```

~\$로 끝나면 일반 사용자
~#로 끝나면 root 사용자

sudo (super user do)

- 리눅스는 다중 사용자 운영체제
- 사용자마다 어떤 일에 대한 권한이 있음.
- 중요한 일에 대한 권한은 super user(root)만 가지고 있음.
- 따라서 super user의 권한이 필요할 때 sudo를 사용한다.
- 직접 root계정에 접속해서 명령어를 실행 시킬 수도 있지만, 안전을 위해 일반 사용자로 로그인 후 sudo를 사용하도록 하자.

사용자 추가

```
$ sudo useradd -m [사용자명]
```

-> /home/egoing 사용자에게 sudo 권한을 부여하는 명령어.
-> 그러나, 이렇게 만들면 해당 계정에서 sudo 명령어를 사용할 수 없음.

sudo 사용할 수 있도록 변경

```
$ sudo usermod -a -G sudo [사용자명]
```

비밀번호 설정

```
$ sudo passwd [사용자명]
```

로그인 (su 사용)

```
$ su - [사용자명]
```

permission (권한)

```
$ ls -l  
-> -rw-rw-r-- 1 egoing egoing 0 Dec 4 23:19 perm.txt
```

(1) type	(2) access mode (r: read, w: write, x: execute)	(3) owner	(4) group		
- file: -, directory: d rw-	[owner 권한] [group의 권한] [other의 권한 (owner && !group)]	rw-	r--	egoin g	egoin g

권한 변경 (chmod)

u: owner
g: group
o: other (not owner, not group)
a: all

```
$ chmod o+r perm.txt  
: perm.txt 파일의 other read 권한 삭제
```

```
$ chmod o+r perm.txt  
: perm.txt 파일의 other read 권한 허가
```

execute

```
$ nano hi-machine.sh
```

```
#!/bin/bash  
echo 'hi hi hi hi'
```

파일 실행 시도 1

```
$ ./hi-machine.sh -> 허가 거부됨
```

파일 실행 시도 2

```
$ /bin/bash hi-machine.sh -> 실행됨
```

owner 실행 권한 추가하기

```
$ chmod u+x hi-machine.sh -> 해당 파일 초록색으로 표시됨
```

파일 실행 시도 3

```
$ ./hi-machine.sh -> 실행됨
```

directory

- r 모드: 해당 디렉토리에 소속된 파일이나 디렉토리를 열람할 수 있는지?
- w 모드: 해당 디렉토리 내부를 수정 할 수 있는지? (파일 생성, 수정, 삭제, 파일이름 변경 등)
- x 모드: 해당 디렉토리로 들어갈 수 있는지?
- chmod 명령어의 -R 옵션: 재귀적으로 해당 디렉토리 안에 있는 것들의 mode를 모두 바꾼다.

Numerical permissions

숫자	의미	영문으로 표현
0	none	---
1	execute only	--x
2	write only	-w-
3	write and execute	-wx
4	read only	r--
5	read and execute	r-x
6	read and write	rw-
7	read, write and execute	rwx

ex) \$ chmod 111 perm.txt -> --x--x--x로 변경됨.

'=' 사용

추가/삭제 개념이 아니라 아예 바꿔는 것

```
$ chmod a=r perm.txt -> owner, group, other read만 가능  
$ chmod a= perm.txt -> owner, group, other 아무 권한 없음.
```

IP 주소 확인

```
$ ip addr : 컴퓨터 IP 확인 (private ip)  
$ curl ipinfo.io/ip : ipinfo.io/ip 입장에서 접속된 ip (public ip)
```

웹서버

웹서버 설치 (apache2)

```
$ sudo apt-get update  
$ sudo apt-get install apache2
```

웹서버 시작, 중지, 재시작

```
$ sudo service apache2 start  
$ sudo service apache2 stop  
$ sudo service apache2 restart
```

쉘을 통해 웹서버 접속 (쉘에서 웹브라우징) : elinks 사용

```
$ sudo apt-get install elinks  
$ elinks http://10.0.2.15/ (컴퓨터 IP 입력) -> local 서버 접속
```

웹서버의 설정

/etc/apache2 에 아파치 설정파일 있음. (apache2.conf)
웹서버가 어떤 storage에서 파일을 찾을 것인가는 설정파일에서 확인.
default는 /var/www/html 인데, 설정 바꿔서 변경시킬 수 있음.
이때 /var/www/html를 document root라고 부른다. (웹페이지를 찾는 최상위 디렉토리)

웹서버의 로그 확인

/etc/apache2/sites-enabled/000-default.conf 가보면 로그는 어디에 기록할지 명시해둔 부분이 있다. (/var/log/apache2)

access.log: 누군가가 웹서버에 접속할 때마다 기록이 남겨짐.
error.log: 여러 로그 확인 가능

실시간으로 확인하려면?

```
$ tail -f /var/log/apache2/access.log
```

SSH

- 다른 컴퓨터를 원격제어 할 때 사용
- 클라이언트를 원격접속 시도하는 컴퓨터, 서버를 원격접속 당하는 컴퓨터라고 한다면, 클라이언트에는 ssh client 프로그램이 있어야하고 서버에는 ssh server 프로그램이 있어야 한다.
- 오늘날 대부분의 유닉스 계열 시스템에서는 ssh 서버가 설치되어있기 때문에 설치할 필요가 거의 없다.
- 원격 접속 시도

```
$ ssh 계정명@접속아이피  
(처음 접속하면 진짜 접속할 건지 물어보는데 yes 하면 됨.)
```

- 포트번호와 함께 접속 시도

```
$ ssh -p 포트번호 계정명@접속아이피
```

ssh 심화수업 - 비밀번호 입력없이 로그인하기

접속을 시도하는 컴퓨터: 내 컴퓨터

접속 대상이 되는 컴퓨터: 원격 컴퓨터

원격 컴퓨터에 내 컴퓨터의 공개키를 저장해두면, 원격 컴퓨터에 로그인 없이 (비밀번호 입력없이/인증 없이) 로그인 가능

어떤 컴퓨터가 인증된 컴퓨터?

원격 컴퓨터의 ~/.ssh/authorized_keys 파일에 들어있는 키 목록을 보고 판단한다.
(authorized_keys 파일은 있을 수도 있고 없을 수도 있음.)

원격 컴퓨터에 내 컴퓨터를 인증시키는 방법

- ssh 공개키 비밀키 만들기 [내 컴퓨터에서 실행]

```
$ ssh-keygen  
Q. Enter file in which to save the key? : (저장되는 위치) 엔터  
Q. Enter passphrase : 비밀번호 쓰려면 입력하고 아니면 그냥 엔터
```

~/.ssh 들어가보면 id_rsa(ssh private key), id_rsa.pub(ssh public key)가 있음. (id_rsa는 절대로 비공개)

2. authorized_keys 파일 끝에다가 내 컴퓨터의 id_rsa.pub 내용을 붙여넣는다. (ssh-copy-id 명령어를 사용하여 안전하게 붙여넣자.) [내 컴퓨터에서 실행]
-> \$ ssh-copy-id (원격 컴퓨터 계정)(원격 컴퓨터 주소)
-> 원격 컴퓨터의 비밀번호 입력
3. 잘 붙여넣어졌는지 확인 [원격 컴퓨터에서 실행]
\$ cat ~/.ssh/authorized_keys
4. 로그인 없이 로그인 시도 [내 컴퓨터에서 실행]
\$ ssh (원격 컴퓨터 계정)(원격 컴퓨터 주소)
-> 비밀번호 입력 없이 로그인 가능

포트 (port)

default port (기본적으로 쓰도록 약속)

ex) 웹서버는 80, ssh 서버는 22

- 각 서버들은 해당 포트에서 대기중이다. (listen)
- 약 65000개의 포트가 있는데, 그 중에 1024개의 well-known 포트가 있다. (잘 알려진 포트)

도메인과 DNS (Domain Name System)

1. 사용자가 google.com 이라고 브라우저 주소창에 입력하면
2. DNS 서버가 google.com 이름을 가진 IP를 찾아서 사용자 컴퓨터에 알려준다.
3. 사용자 컴퓨터는 그 IP 정보로 google.com에 접속
 - /etc/hosts 파일: 아이피-도메인 명시해둔 것?
 - ex) hosts파일에 127.0.0.1 google.com 추가하고 브라우저에 google.com 이라고 입력했을 때, 127.0.0.1 웹서버와 매칭됨.
 - /etc/hosts에서 먼저 찾아보고, 찾아봤을 때 없으면, DNS 서버에서 찾고..
 - 만약 /etc/hosts에 있으면 DNS 서버는 관련없음.
 - /etc/resolv.conf -> nameserver 적혀있음.
 - \$ host [도메인] -> 호스트 정보 나옴 (어떤 아이피로 연결되어 있는지)

서브도메인

- 하나의 도메인 구입으로 여러 도메인 사용할 수 있음.
- ex) admin.egoing.ga, blog.egoing.ga, news.egoing.ga
- 서로 다른 아이피를 가리키게
- 하나의 도메인으로 여러 사이트를 운영!

rsync

동작방법 예시

```
$ mkdir src  
$ mkdir dest  
  
src 디렉토리안에 파일 생성  
$ touch test{1..10}; test1~test10까지 생성됨.  
  
$ rsync -a src/ dest -> src 아래 있는 파일들이 dest 폴더안으로 복사된다.  
(a 옵션 : 아카이브 모드로 동작. 파일 뿐만 아니라 디렉토리도 복사되고, 권한과 같은 파일/  
디렉토리 속성이 바뀌면 반영되고, 변경사항들만 전송)  
  
dest 폴더 가서 test10 삭제  
$ rm test10  
  
$ rsync -av(더 자세하게 출력) src/ dest 실행 -> dest안에 있던 test10 다시 생김.  
  
src 폴더 가서 추가 파일 생성 후, rsync 해보면 src에서 만든 파일 만이 dest 안에 생성된다.
```

컴퓨터와 컴퓨터 사이에 전송

```
$ rsync -azP ~/rsync/src/ k8805@대상아이피:~/rsync/dest  
az: 압축한다.  
P: 전송되는 상황을 progress bar로 보여준다.
```

암호화

대칭적인 방식

어떤 정보를 암호화/복호화 할 때 키를 사용하는데, 암호화 했을 때와 복호화 했을 때 같은 키를 사용한다면 대칭적인방식

비대칭적인 방식

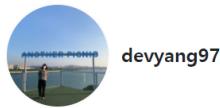
- ssh-keygen으로 만든 private key, public key를 사용
- 암호화 할 때에는 private key, 복호화 할 때에는 public key를 사용

- private key, public key는 짹꿍이다.
- 대표 주자: RSA

RSA 동작방식

1. ssh client로 ssh server에 접속하면
2. ssh server는 랜덤한 키를 생성하여 client 컴퓨터에게 준다.
3. ssh client는 id_rsa 파일(private key)이 있는지 찾는다. 있으면 그 파일을 이용해서 전달 받은 랜덤키를 암호화 시킨다.
4. 암호화된 결과를 ssh server에게 전송
5. ssh server는 authorized_keys에 저장된 공개키를 이용해서 전달받은 암호화된 파일을 복호화 한다.
6. 복호화된 결과가 처음에 전달한 랜덤키와 같으면 인증 성공

-
- 생활코딩 수업을 기반으로 작성하였습니다.
 - <https://opentutorials.org/course/2598>



◀ 이전 포스트
[Git] 명령어 정리

다음 포스트 ▶
[CentOS 8] 1. VirtualBox 설치

4개의 댓글

댓글을 작성하세요

댓글 작성



2021년 6월 23일

미쳤당 간결하고 필요한것만 있네요 !! 달달 외울 개용 !! ㅎㅎㅎㅎㅎ

▣ 습기기



2021년 6월 23일

화이팅!!

▣ 답글 달기

답글 작성하기



2021년 8월 24일

감사합니다 :)

▣ 1개의 답글

관심 있을 만한 포스트



리눅스 쉘 기본 명령어(Basic Shell Co...
리눅스 쉘 기본 명령어에 대해 알아보겠습니다.
다. 옵션은 대부분 제외하였습니다. cd 가장





[GitHub] Git 명령어 모음

```
_ # INDEX _ _ Git 명령어 _ 명령어 | 명령어 |
내용 설명 | [-----] | [-----] | $ git init |
.git 하위 디렉토리 생성 | $ git add 파일명...
```

2020년 8월 28일 - 1개의 댓글

by delilah

19



Docker 도커 - #1 기본 명령어 모음

도커란 간단하게 말해 서버환경에서의 다양한 프로그램, 실행환경을 컨테이너라는 고립된 환경에서 실행할 수 있게 해주는 컨테이너...

2019년 9월 24일 - 3개의 댓글

by wlsdud2194

8



TIL 27 | Git찮아도 알아둬야 할 Git명령어

스럽고 외양간 고지.. 아니 파일 읽고 Git 명령어 공부하기...

2021년 10월 17일 - 3개의 댓글

by 4_21ee

64

Docker로 프로젝트 배포하기

Docker!!

2020년 3월 20일 - 1개의 댓글

by devmin



[MySQL] 설치와 초기 설정 for macOS

MAC에서 MySQL설치방법은 MySQL Community Server dmg파일을 다운받아 설치하는 방법과 Homebrew로 MySQL 패키지 ...

2020년 6월 6일 - 0개의 댓글

by devmin

10

by inyong_pang

9



[Git] Git 사용법 및 터미널 명령어 정리

pwd Print working directory: 현재 작업 위치 알려줌. ls list files: 현재의 directory의 모든 파일들을 보여줌. cd .. 상위 디렉토리로 이...

2020년 10월 8일 - 0개의 댓글

by grinding_hannah

12

[ft_server] 총 정리 : 도커 설치부터 ...

과제 시작부터 마무리까지 기록한 내용들.

2020년 10월 1일 - 3개의 댓글

by hidaehyunlee

5

우분투 개발 환경 설정

설치 프로그램의 업데이트가 필요하다면 위 명령어를 입력한다.Git을 설치하는 명령어다.Git을 설치하고 user@email과 user.name을...

2020년 5월 12일 - 1개의 댓글

by gwak2837

AWS EC2에 SpringBoot 배포하기

본 글에서는 AWS EC2 인스턴스 서버에 SpringBoot를 배포하는 과정에 대해서 정리하고자 한다(Gradle 프로젝트 기준). ./gradlew...

2021년 2월 1일 - 0개의 댓글

by banjoknim

3

SVN(Subversion) - 개념 및 명령어

SVN(SubVersion)은 여러분이서 작업하는 프로젝트의 버전관리나 각자 만든 소스의 통합과 같은 문제를 해결하기 위해 저작자를 만들어 그곳에 소스를 저장해 소스 출처이나 여러 문제를 해결하기 위한 형성관리/소스 관리 둘이다. 형성관리: 소스의 변화를 들입없이 관리...

2020년 11월 5일 - 0개의 댓글

by gillog

0

CentOS7 Apache 설치

yum을 이용하여 apache를 설치한다.apache 버전을 확인하여 설치가 제대로 되었는지 확인 위해 봤다. apache 실행CentOS7부터 기준에...

2020년 6월 6일 - 0개의 댓글

by recordboy

0



Powered by
GraphCDN