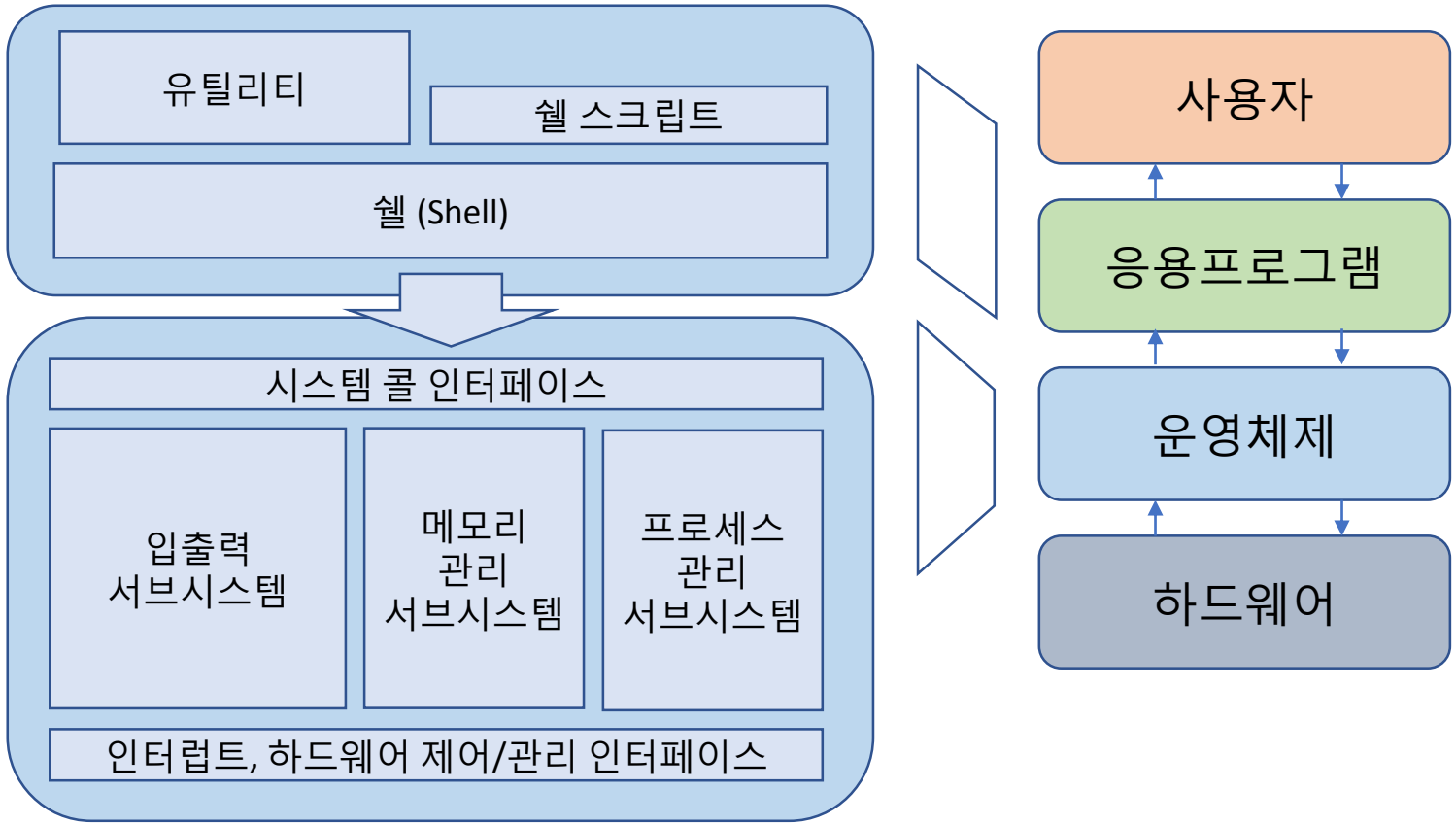
Chapter 02. 리눅스 쉘과 CLI 명령어

BASH 쉘과 친숙해지기

리눅스 쉘(shell) 이란?

사용자 명령어 및 프로그램을 실행할 수 있는 공간 (사용자 인터페이스)



리눅스 쉘(shell) 의 종류

두개의 메인 타입
- Bourne shell - 특징 $ 프롬프트(prompt)
- C shell - 특징 % 프롬프트

Bourne Shell 의 변천사
- Bourne shell -> sh
- Korn shell -> ksh
- **Bourne Again shell -> bash**
- POSIX shell -> sh

C-type shell 의 변천사
- C shell -> csh
- TENEX/TOPS C shell -> tcsh

```
user1@user1-VirtualBox:~$ cat /etc/passwd | grep user --color=none
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
user1:x:1000:1000:user1,,,:/home/user1:/bin/bash
user2:x:1001:1001:UserName2:/home/user2:/bin/bash
user1@user1-VirtualBox:~$
```

```
user1@user1-VirtualBox:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/dash
/bin/bash
/bin/rbash
/bin/csh
user1@user1-VirtualBox:~$
```

리눅스 쉘(shell) - 프롬프트(prompt)

사용자와 인터렉티브(interactive) 한 입력을 주고 받을 수 있는 명령 대기 표시자

우분투 기본 프롬프트:
[username@hostname]:<directory> $

```
user1@user1-VirtualBox:~$ whoami
user1
user1@user1-VirtualBox:~$ hostname
user1-VirtualBox
user1@user1-VirtualBox:~$ pwd
/home/user1
user1@user1-VirtualBox:~$
```

환경변수 PS1 에 기록됨 (PS1 = Prompt Statement One)

```
user1@user1-VirtualBox:~$ echo $PS1
\[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033
[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$
user1@user1-VirtualBox:~$
```

\u : username
\h : hostname
\w : current directory

```
user1@user1-VirtualBox:~$ DEFAULT=$PS1
user1@user1-VirtualBox:~$ PS1="\u@\h:\w$ "
user1@user1-VirtualBox:~$ cd dir1
user1@user1-VirtualBox:~/dir1$ cd ..
user1@user1-VirtualBox:~$ PS1=$DEFAULT
user1@user1-VirtualBox:~$
```

\d : date (Mon May 5)
\D{format} : date 포멧정의
\e : ASCII escape char (033)
\h : hostname .이전
\H : hostname
\t : time (24-hr HH:MM:SS)
\T : time (12-hr HH:MM:SS)
\@ : time (12-hr am/pm format)
\u : username
… 그 외 다수 생략

리눅스 쉘(shell) - 색상(color)

사용자와 인터렉티브(interactive) 한 입력을 주고 받을 수 있도록 도움을 주는 색상값

ANSI 표준 (ANSI : American National Standard Institute, 미국 국가표준 협회)
ASCII-CODE 등 정의

ANSI color : ANSI escape code 중 하나로 ISO/IEC-9529 표준 (UNIX, Linux, DOS, Windows)

사용방법 :

- ESC + [코드값;코드값;… + 글자
  - ESC 는 고정값 \e 또는 \033
  - 코드값 : 진함, 흐림, 이탤릭, 밑줄, 깜빡임, … , 글자색, 배경색, 밝기, … 등
    - 칼러코드
      - 검정색(Black) : 0
      - 빨간색(Red) : 1
      - 초록색(Green) : 2
      - 노란색(Yellow) : 3
      - 파란색(Blue) : 4
      - 자주색(Magenta) : 5
      - 하늘색(Cyan) : 6
      - 하얀색(White) : 7

리눅스 쉘(shell) - 색상(color)

https://en.wikipedia.org/wiki/ANSI_escape_code

사용 예시:

```
user1@user1-VirtualBox:~$ echo -e "\e[31mHello 빨강"
Hello 빨강
user1@user1-VirtualBox:~$ echo -e "\e[32mHello 초록"
Hello 초록
user1@user1-VirtualBox:~$ echo -e "\e[33mHello 노랑"
Hello 노랑
user1@user1-VirtualBox:~$ echo -e "\e[34mHello 파랑"
Hello 파랑
user1@user1-VirtualBox:~$ echo -e "\e[41mHello 빨강배경"
Hello 빨강배경
user1@user1-VirtualBox:~$ echo -e "\e[44mHello 파랑배경"
Hello 파랑배경
user1@user1-VirtualBox:~$ echo -e "\e[33;42mHello 노란글씨 초록배경"
Hello 노란글씨 초록배경
user1@user1-VirtualBox:~$ echo -e "\e[33;42mHello 노란글씨 초록배경 초기화\e[0m"
Hello 노란글씨 초록배경 초기화
user1@user1-VirtualBox:~$
```

환경변수 PS1 에 기록됨 (PS1 = Prompt Statement One)

```
user1@user1-VirtualBox:~$ echo $PS1
\[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033
[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$
user1@user1-VirtualBox:~$
```

Name	FG Code	BG Code	Ubuntu[nb 7]
Black	30	40	1,1,1
Red	31	41	222,56,43
Green	32	42	57,181,74
Yellow	33	43	255,199,6
Blue	34	44	0,111,184
Magenta	35	45	118,38,113
Cyan	36	46	44,181,233
White	37	47	204,204,204
Bright Black	90	100	128,128,128
Bright Red	91	101	255,0,0
Bright Green	92	102	0,255,0
Bright Yellow	93	103	255,255,0
Bright Blue	94	104	0,0,255
Bright Magenta	95	105	255,0,255
Bright Cyan	96	106	0,255,255
Bright White	97	107	255,255,255

기본 명령어 - 출력 (echo)

echo [OPTION]... [STRING]...
화면에 글자를 출력

옵션 :

- -n : 뉴라인 제외
- -e : Escape 코드 지원
- -E : Escape 모드 미지원 (기본값)

```
user1@user1-VirtualBox:~$ echo "Hello"
Hello
user1@user1-VirtualBox:~$ echo "\e[33mHello"
\e[33mHello
user1@user1-VirtualBox:~$ echo -e "\e[33mHello"
Hello
user1@user1-VirtualBox:~$ echo -n "Hello\nWorld"
Hello\nWorlduser1@user1-VirtualBox:~$ echo -en "Hello\nWorld"
Hello
Worlduser1@user1-VirtualBox:~$ echo -e "Hello\nWorld"
Hello
World
```

기본 명령어 - 재지향(리다이렉션) (>, >>, 2>, 2>&)

결과물을 다른 장치로 보냄 (output, append, error, merge)

사용 예시 :

- echo “Hello” > hello.txt : 파일로 출력
- echo “Hello another” > hello.txt : 기존 파일을 덮어씀

- echo “Hello again” >> hello.txt : 기존 파일에 누적

- ls > file.txt : 출력 결과물을 파일로 출력 (단, stdout만)
- aaa > file.txt : 아무런 내용도 기록되지 않음

- aaa 2> file.txt : 실패한 결과물을 파일로 출력

출력 장치의 유형 :

- stdout : 표준출력 : 장치번호 1
- stderr : 에러출력 : 장치번호 2

- stdin : 입력장치 : 장치번호 0

복합 사용 예시 :

- ls /tmp/* > result.txt 2>&1
  출력 결과물의 성공값(표준출력) 을 result.txt 로 보내고 에러값(에러출력) 을 1번(표준출력)과 같은 곳으로 보내라
- ls /tmp/* &> result.txt
  (상동) 줄여쓰는 표현법

```
user1@user1-VirtualBox:~$ aaa > file.txt
'aaa' 명령을 찾을 수 없습니다. 비슷한 명령:
 'aha' 명령은 패키지 'aha'(universe)에 있습니다.
 'aa' 명령은 패키지 'astronomical-almanac'(universe)에 있습니다.
 'ara' 명령은 패키지 'ara'(universe)에 있습니다.
 'jaaa' 명령은 패키지 'jaaa'(universe)에 있습니다.
aaa: 명령을 찾을 수 없습니다
user1@user1-VirtualBox:~$ cat file.txt
user1@user1-VirtualBox:~$ aaa 2> file.txt
user1@user1-VirtualBox:~$ cat file.txt
'aaa' 명령을 찾을 수 없습니다. 비슷한 명령:
 'aha' 명령은 패키지 'aha'(universe)에 있습니다.
 'jaaa' 명령은 패키지 'jaaa'(universe)에 있습니다.
 'aa' 명령은 패키지 'astronomical-almanac'(universe)에 있습니다.
 'ara' 명령은 패키지 'ara'(universe)에 있습니다.
aaa: 명령을 찾을 수 없습니다
```

기본 명령어 - 재지향(리다이렉션) (<, <<)

입력값 리다이렉션 (표준 입력 - stdin) 및 delimiter

사용 예시 :

- echo “Hello” > hello.txt : 파일로 출력
- echo < hello.txt : 입력값을 받고 싶으나 동작하지 않음, 왜?? stdin 입력 지원여부…

- cat < hello.txt

- cat (enter, 종료시 Ctrl-D)

Delimiter 사용 예시 :

- cat << end
  표준입력으로부터 end 값이 들어올때까지 입력

- cat << end > hello.txt
  표준입력으로부터 end 값이 들어올때까지의 입력 결과를 파일로 출력

```
user1@user1-VirtualBox:~$ cat
asdf
asdf
user1@user1-VirtualBox:~$ cat << end
> hello
> world
> end
hello
world
user1@user1-VirtualBox:~$ cat << end > hello.txt
> hello
> world
> end
user1@user1-VirtualBox:~$ cat hello.txt
hello
world
user1@user1-VirtualBox:~$
```

기본 명령어 - 파이프 (|)

출력값 프로세스간 전달

사용 예시 :

- ls -l | grep hello : 출력값 내에서 검색
- ls -l | wc -l : 출력값 내에서 줄 개수 확인
- ls -l | grep hello | wc -l : 다중 파이프 활용
- cat hello.txt | more : 출력값 내에서 페이징 처리

표준 출력을 입력으로 받는 유틸리티 :

- 필터링 : less, more, grep
- 변경 : awk, sed
- 유틸리티 : sort, wc

```
user1@user1-VirtualBox:~$ ls -l
합계 76
drwxrwxr-x 4 user1 user1 4096  4월 25 00:39 dir1
drwxrwxr-x 3 user1 user1 4096  4월 25 00:40 dir2
drwxrwxrwt 2 user1 user1 4096  5월  4 00:10 dir3
-rw-r--r-- 1 user1 user1 8980  4월  5 18:38 examples.desktop
-rw-rw-r-- 1 user1 user1  356  5월  4 11:38 file.txt
-rw-rw-r-- 2 user1 user2   12  5월  4 12:03 hello.txt
-rw-rw-r-- 2 user1 user2   12  5월  4 12:03 hellolink
lrwxrwxrwx 1 user1 user1    9  5월  3 23:41 hellosymlink -> hello.txt
-rw-rw-r-- 1 user1 user1  586  5월  4 12:48 result.txt
drwxrwxr-x 2 user1 user1 4096  5월  3 22:46 testdir
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 공개
drwxr-xr-x 2 user1 user1 4096  4월 18 01:18 다운로드
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 문서
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 바탕화면
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 비디오
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 사진
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 음악
drwxr-xr-x 2 user1 user1 4096  4월  5 18:54 템플릿
user1@user1-VirtualBox:~$ ls -l | grep hello.txt
-rw-rw-r-- 2 user1 user2   12  5월  4 12:03 hello.txt
lrwxrwxrwx 1 user1 user1    9  5월  3 23:41 hellosymlink -> hello.txt
user1@user1-VirtualBox:~$
```

명령어 히스토리 (history)

쉘에서 입력한 명령어들의 기록 (몇 개나? 최근 1000개, 총 2000개)

저장 개수 :

echo $HISTSIZE : 버퍼링
echo $HISTFILESIZE : 파일에 기록 (쉘 종료시)

사용 예시 :

- history [OPTION]
  - 10 : 최근 10개의 히스토리 보기
  - -c : 히스토리 버퍼 삭제(clear)

쉘명령어 :

- !15 : 15번째 라인 다시 실행
- !! : 바로 이전 명령어 다시 실행

```
user1@user1-VirtualBox:~$ history
    1  ifconfig
    2  ssh localhost
    3  ps aux | grep sshd
    4  clear
    5  sudo service sshd restart
    6  clear
    7  sudo apt-get search sshd
    8  clear
    9  sudo apt-get install sshd
   10  sshd
   11  clear
   12  sudo apt install openssh-server
   13  clear
   14  sudo service sshd start
   15  sudo service sshd status
   16  ifconfig
   17  ifconfig -a
   18  cat /boot
   19  cat /boot/grub/grub.cfg
   20  cat /etc/default/grub
   21  clear
   22  ls -al
```

환경변수 - PATH

명령어 실행 (어디에?)
배포판에 따라 현재 디렉토리를 1순위로 실행하는 배포판도 있으나, 우분투는 그렇지 않음.

echo $PATH
export PATH=$PATH:<추가할디렉토리>

1. PATH 디렉토리 확인
2. 실행권한 확인
3. 명령어를 해당 사용자ID 로 실행

2.1. SetUID 확인
3.1. 해당 명령어의 소유주 권한으로 명령어 실행

바이너리 실행파일은, PATH 의 순차적으로 검색이 된다.

```
user1@user1-VirtualBox:~$ echo $PATH
/home/user1/bin:/home/user1/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:
/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
user1@user1-VirtualBox:~$
```

환경변수 - PATH - which

which [FILENAME]
내가 실행하는 바이너리가 어디에서 실행되는가?

사용 예시 :

- which ls
- which python

```
user1@user1-VirtualBox:~$ echo $PATH
/home/user1/bin:/home/user1/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:
/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
user1@user1-VirtualBox:~$ which ls
/bin/ls
user1@user1-VirtualBox:~$ which python
/usr/bin/python
```

환경변수 - printenv, env

printenv
다양한 환경변수 확인

주요 환경변수

환경 변수	기능	환경 변수	기능
PATH	실행 파일의 경로 모음	HOME	사용자의 홈 디렉토리
PWD	현재 워킹 디렉토리	LANG	현재 언어 셋 (인코딩)
USER	사용자 아이디	LANGUAGE	현재 언어
COLUMNS	터미널의 가로 크기	LINES	터미널의 세로 크기
LS_COLORS	ls 명령어의 출력 칼러코딩	PS1	쉘 프롬푸트

확인방법 및 변경방법
- echo $환경변수
- 환경변수 = 값 (해당 터미널에서만)
- export 환경변수 = 값 (전체 터미널에서)

환경변수 - LANGUAGE / LANG

언어(LANGUAGE) 및 언어셋(LANG) 활용

환경변수 활용방법 :
- echo $LANGUAGE
- echo $LANG

언어 (한시적으로) 변경
- LANGUAGE=en COMMAND [ARGS]…

언어셋 (한시적으로) 변경
- LANG=c COMMAND [ARGS]…

영구적으로 변경 시에는
- export LANGUAGE=en

```
user1@user1-VirtualBox:~$ echo $LANGUAGE
ko
user1@user1-VirtualBox:~$ aaa
'aaa' 명령을 찾을 수 없습니다. 비슷한 명령:
 'aa' 명령은 패키지 'astronomical-almanac'(universe)에 있습니다.
 'aha' 명령은 패키지 'aha'(universe)에 있습니다.
 'ara' 명령은 패키지 'ara'(universe)에 있습니다.
 'jaaa' 명령은 패키지 'jaaa'(universe)에 있습니다.
aaa: 명령을 찾을 수 없습니다
user1@user1-VirtualBox:~$ LANGUAGE=en aaa
No command 'aaa' found, did you mean:
 Command 'aa' from package 'astronomical-almanac' (universe)
 Command 'ara' from package 'ara' (universe)
 Command 'jaaa' from package 'jaaa' (universe)
 Command 'aha' from package 'aha' (universe)
aaa: command not found
user1@user1-VirtualBox:~$
```

```
user1@user1-VirtualBox:~$ cd dir1
user1@user1-VirtualBox:~/dir1$ ls -l
합계 8
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub1
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub2
user1@user1-VirtualBox:~/dir1$ LANGUAGE=en ls -l
total 8
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub1
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub2
user1@user1-VirtualBox:~/dir1$
```

환경변수 - LANG - locale

언어와 언어셋(캐릭터셋), 그리고 다양한 지역 설정값을 확인 (=로케일)

현재 로케일(locale) 정보 확인
- locale
- localectl (status)

설정 가능한 모든 로케일(locale) 정보 확인
- locale -a

누락된 로케일(locale) 을 새로 활성화 하려면?
- /etc/locale-gen (주석처리 확인)
- locale-gen (로케일 재빌드)

```
user1@user1-VirtualBox:~$ date
2020. 05. 05. (화) 10:45:59 KST
user1@user1-VirtualBox:~$ LC_TIME=en_US date
Tue May  5 10:46:04 KST 2020
user1@user1-VirtualBox:~$ LC_TIME=en_ZM date
Tue  5 May 10:46:09 KST 2020
```

```
user1@user1-VirtualBox:~$ locale
LANG=ko_KR.UTF-8
LANGUAGE=ko
LC_CTYPE="ko_KR.UTF-8"
LC_NUMERIC="ko_KR.UTF-8"
LC_TIME="ko_KR.UTF-8"
LC_COLLATE="ko_KR.UTF-8"
LC_MONETARY="ko_KR.UTF-8"
LC_MESSAGES="ko_KR.UTF-8"
LC_PAPER="ko_KR.UTF-8"
LC_NAME="ko_KR.UTF-8"
LC_ADDRESS="ko_KR.UTF-8"
LC_TELEPHONE="ko_KR.UTF-8"
LC_MEASUREMENT="ko_KR.UTF-8"
LC_IDENTIFICATION="ko_KR.UTF-8"
LC_ALL=
```

```
user1@user1-VirtualBox:~$ locale -a
C
C.UTF-8
POSIX
en_AG
en_AG.utf8
en_AU.utf8
en_BW.utf8
en_CA.utf8
en_DK.utf8
en_GB.utf8
en_HK.utf8
en_IE.utf8
en_IN
en_IN.utf8
en_NG
en_NG.utf8
en_NZ.utf8
en_PH.utf8
en_SG.utf8
en_US.utf8
en_ZA.utf8
en_ZM
en_ZM.utf8
en_ZW.utf8
ko_KR.utf8
```

단축명령어 (alias)

bash 쉘의 장점 - 축약어 기능 (alias)

자주 쓰는 긴 명령어를 짧게 요약
- ls 명령어는 이미 “ls --color=auto” 의 축약어
- ll 명령어는 이미 “ls -alF“ 의 축약어

그렇다면? 내가 원하는 축약어는?
- alias **..**=“cd ..”
- alias **...**=“cd ../..”
- alias en=“LANGUAGE=en”

```
user1@user1-VirtualBox:~$ cd dir1
user1@user1-VirtualBox:~/dir1$ ..
user1@user1-VirtualBox:~$ cd dir1/sub1
user1@user1-VirtualBox:~/dir1/sub1$ ...
user1@user1-VirtualBox:~$ en ls -l dir1
total 8
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub1
drwxrwxr-x 2 user1 user1 4096  4월 25 00:39 sub2
user1@user1-VirtualBox:~$
```

```
user1@user1-VirtualBox:~$ alias
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo
 error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[;&|]\s*alert$//'\'
')"'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
user1@user1-VirtualBox:~$
```

쉘 부팅(시작) 시퀀스 (.profile, .bashrc 등)

```
user1@user1-VirtualBox:~$ ls -al .profile
-rw-r--r-- 1 user1 user1 655  4월  5 18:38 .profile
user1@user1-VirtualBox:~$ ls -al .bash*
-rw------- 1 user1 user1 18465  5월  4 12:05 .bash_history
-rw-r--r-- 1 user1 user1   220  4월  5 18:38 .bash_logout
-rw-r--r-- 1 user1 user1  3771  4월  5 18:38 .bashrc
```

```
user1@user1-VirtualBox:~$ sudo su - user2
START /etc/profile
START /etc/bash.bashrc
END /etc/bash.bashrc
END /etc/profile
START .profile
START .bashrc
END .bashrc
END .profile
```

BASH 의 interactive shell 시작 시퀀스

- /etc/profile 수행 (공통 수행 - 환경 설정들)
  - /etc/profile.d/*.sh (공통 수행)
  - /etc/bash.bashrc (공통 수행 - 시스템 alias 등)
- ~/.profile 수행 (사용자별 디렉토리 - 시작 프로그램 등)
  - ~/.bashrc 수행 (사용자별 디렉토리 - alias 등)
    - ~/.bash_aliases (이 파일이 추가적으로 있다면 수행 - 기본은 없음)

BASH 의 종료 시퀀스

- ~/.bash_logout

참고: 사용자 계정이 만들어질 때

- 원본파일 /etc/skel/* 내용이 사용자 home 에 복사 됨

쉘 스크립트

bash 쉘의 공간에서 나만의 프로그래밍을 작성

쉘 스크립트 예시1: hello.txt 파일을 읽어서 화면에 출력

while read line; do echo $line; done < hello.txt

```
user1@user1-VirtualBox:~$ while read line; do echo $line; done < hello.txt
hello
world
user1@user1-VirtualBox:~$
```

쉘 스크립트 예시2 : (/etc/passwd 파일을 입력값으로 읽어서, 해당 컬럼을 파싱)

while IFS=: read -r F1 F2 F3 F4 F5 F6 F7
do
echo "사용자 $F1 는 $F7 쉘을 사용하고 $F6 홈디렉토리를 사용한다."
done < /etc/passwd

참고:
IFS = Internal Field Separator
기본값 <space><tab><newline>

```
user1@user1-VirtualBox:~$ while IFS=: read -r F1 F2 F3 F4 F5 F6 F7
> do
> echo "사용자 $F1 는 $F7쉘을 사용하고 $F6 홈디렉토리를 사용한다."
> done < /etc/passwd
사용자 root 는 /bin/bash쉘을 사용하고 /root 홈디렉토리를 사용한다.
사용자 daemon 는 /usr/sbin/nologin쉘을 사용하고 /usr/sbin 홈디렉토리를 사용한다.
사용자 bin 는 /usr/sbin/nologin쉘을 사용하고 /bin 홈디렉토리를 사용한다.
사용자 sys 는 /usr/sbin/nologin쉘을 사용하고 /dev 홈디렉토리를 사용한다.
```

쉘 프로그래밍 - 실행방법 및 shebang

스크립트 작성 및 /bin/sh 또는 /bin/bash 를 통해 실행
- /bin/sh test1.sh

실행 퍼미션을 통한 직접 실행
- chmod +x test1.sh
- ./test1.sh

이때, 해당 쉘 스크립트의 속성을 첫 줄에 정의 (she(#)bang(!) 또는 shabang, hashbang)
- #!/bin/bash
- #!/usr/bin/perl
- #!/usr/bin/python

shebang 무시하고 직접 실행
- bash -m test1.sh

```
user1@user1-VirtualBox:~$ cat test1.sh
#!/bin/sh
echo "Hello, world"
user1@user1-VirtualBox:~$ cat test2.sh
#!/bin/more
hello
world
user1@user1-VirtualBox:~$ ./test1.sh
Hello, world
user1@user1-VirtualBox:~$ ./test2.sh
#!/bin/more
hello
world
user1@user1-VirtualBox:~$
```

쉘 프로그래밍 - 리얼 프로그래밍 및 입력값, 출력값, 실행결과 등

리얼 프로그램을 위한 각종 명령어 :

- for / do / while / test / read / print
- function 등등…

- 프로그래밍 강의 참조

입출력 인자값 :

- $0 - 스크립트/명령어 이름
- $# - 전달된 파라미터 수
- $$ - 프로세스 번호
- $? - 실행 결과

- $1, $2, … 입력 인자
- $* 입력인자 모두