유틸리티

sample

you raise me up

When I am down, and, oh, my soul, so weary
When troubles come, and my heart burdened be
Then, I am still and wait here in the silence
Until you come and sit awhile with me

You raise me up, so I can stand on mountains
You raise me up to walk on stormy seas
I am strong when I am on your shoulders
You raise me up to more than I can be

There is no life - no life without its hunger
Each restless heart beats so imperfectly
But when you come and I am filled with wonder,
Sometimes, I think I glimpse eternity.

필터링: grep 명령어

- $ grep 패턴 파일
  - 대상 파일들을 읽어서, 해당 패턴을 검색하고, 패턴을 포함하는 줄을 출력

- $ grep with you.txt //with문자열 포함

  Until you come and sit awhile with me
  There is no life - no life without its hunger;
  But when you come and I am filled with wonder,

- $ grep -w with you.txt // with단어 포함

  Until you come and sit awhile with me
  But when you come and I am filled with wonder,

- $ grep -n with you.txt //줄번호 삽입

  4:Until you come and sit awhile with me
  11:There is no life - no life without its hunger;
  13:But when you come and I am filled with wonder,

grep 명령어

- $ grep -i when you.txt // 대소문자 구분안함

```
When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
I am strong, when I am on your shoulders
But when you come and I am filled with wonder,
```

- $ grep -v raise you.txt // 지정 단어 제외

```
When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
Then, I am still and wait here in the silence
Until you come and sit awhile with me
I am strong, when I am on your shoulders
There is no life - no life without its hunger;
Each restless heart beats so imperfectly;
But when you come and I am filled with wonder,
Sometimes, I think I glimpse eternity
```

정규표현식(regular expression)

- 정규표현식을 이용한 패턴 기술
  - . : 임의의 한 문자
  - * : 0개 이상 예) a*b는 b, ab, aab, aaab, ...
  - + : 1개 이상
  - [ ] : [과] 사이의 문자 중 하나
  - [^...] : [^ 과] 사이의 문자를 제외한 나머지 문자 중 하나
  - ^, $: 각각 줄의 시작과 끝을 의미한다.

- $ grep -w 'st.*d' you.txt
  You raise me up, so I can stand on mountains

- $ grep -w 'w.*t' you.txt
  Then, I am still and wait here in the silence
  There is no life - no life without its hunger;

정규표현식(regular expression)

- $ grep '^root' /etc/passwd
- $ grep 'bash$' /etc/passwd
- $ ls -l | grep -v ^d
- $ grep [aA]pple test.txt
- $ grep '^[ab]' test.txt

파일 찾기: find 명령어

- find 명령어
  - 옵션의 검색 조건에 따라 대상 디렉터리 밑에서 해당되는 파일을 모두 찾아 출력
  - $ find 디렉터리 [-옵션]

검색 조건 및 처리 방법	설명
-name 파일	파일 이름으로 찾는다.
-atime +n	접근 시간이 n일 이전인 파일을 찾는다.
-atime -n	접근 시간이 n일 이내인 파일을 찾는다.
-mtime +n	n일 이전에 변경된 파일을 찾는다.
-mtime -n	n일 이내에 변경된 파일을 찾는다.
-perm nnn	파일 권한이 nnn인 파일을 찾는다.
-type x	파일 타입이 x인 파일들을 찾는다.
-size n	사이즈가 n이상인 파일들을 찾는다.
-links n	링크된 개수가 n인 파일들을 찾는다.
-user 사용자이름	사용자이름으로 찾는다.
-group 그룹이름	그룹이름으로 찾는다.
-print	찾은 파일의 절대 경로명을 화면에 출력한다.
-exec cmd {};	찾은 파일들에 대해 cmd 명령어를 실행한다.

find 명령어 예

- $ find /usr -name '*.c' -print /usr 밑에 .c 파일들을 찾아 경로명 출력
- $ find . -name ping -ls 이름이 ping인 파일을 찾아 ls 명령어 수행
- $ find . -type d -print 디렉터리(d) 파일을 찾아 경로명을 출력
- $ find . -perm 700 -ls 사용권한 700인 파일 찾아 ls 명령어 수행
- $ find . -size +1024 -print 1024 bytes 이상인 파일을 찾아 출력
- $ find . -name core -size +2048 -ls
  파일 이름이 core이고 크기가 2048 bytes 이상인 파일을 찾아 ls 명령어 수행
- $ find . -user chang -print 파일 소유자가 chang인 파일을 찾아 출력
- $ find . -atime +30 -print 30일 이전에 접근되었던 파일을 찾아 출력
- $ find . -mtime -7 -print 7일 이내에 수정된 적이 있는 파일들 출력
- $ find . -name core -exec rm -i {} \;
  이름이 core인 파일에 대해 rm 명령어 수행
- $ find . -name '*.c' -atime +30 -exec ls -l {} \;
  30일 이전에 접근된 파일 중 *.c를 찾아 ls -l 명령어 수행

5.3 디스크 및 아카이브

df 유틸리티

- $ df 파일시스템*
  - 파일시스템에 대한 정보(사용중/사용 가능한 디스크 공간)

- $ df

```
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/sda3 49594228 7352576 39681696 16% /
/dev/sda5 86495548 66226168 15875608 81% /home
/dev/sdb1 141122196 103870536 30083060 78% /home1
/dev/sda1 101086 11455 84412 12% /boot
tmpfs 1037652 0 1037652 0% /dev/shm
```

아카이브

- 아카이브
  - 백업 또는 다른 장소로의 이동을 위해 여러 파일들을 하나로 묶어놓은 묶음
  - 아카이브를 만들거나 푸는데 tar(tape archive) 명령어 사용
- tar 명령어
  - 옵션: c(create), v(verbose), x(extract), t(table of contents), f(file)
  - $ tar -cvf 타르파일 파일+

    여러 파일들을 하나의 타르파일로 묶으며 보통 확장자로 .tar 사용
  - $ tar -xvf 타르파일

    하나의 타르파일을 풀어서 원래 파일들을 복원
  - $ tar -tvf 타르파일

    타르파일의 내용을 확인

아카이브: 예

- 현재 디렉터리에 있는 모든 파일을 다른 곳으로 옮기기

```
$ tar -cvf src.tar *
```

  - src.tar를 다른 곳으로 이동

```
$ tar -tvf src.tar
$ tar -xvf src.tar
```

압축

- compress/ uncompress 명령어 //리눅스 지원안함

  $ compress 파일

  $ uncompress 파일.Z*

- gzip 명령어 // 개별적으로 압축됨

  $ gzip 파일

  $ gzip -d 파일.gz

- 사용 방법
  - 파일들을 하나의 타르파일로 묶은 후 compress/gzip을 사용해 압축
  - 파일 복원: 압축을 해제한 후, 타르파일을 풀어서 원래 파일들을 복원

압축 예

- 사용예

```
$ tar -cvf src.tar *
$ gzip src.tar
```

  - 이 파일을 원하는 곳으로 이동

```
$ gzip -d src.tar.gz
$ tar -xvf src.tar
```

Shell

쉘(Shell)이란 무엇인가?

- 쉘의 역할
  - 쉘은 사용자와 운영체제 사이에 창구 역할을 하는 소프트웨어
  - 명령어 처리기(command processor)
  - 사용자로부터 명령어를 입력 받아 이를 처리한다

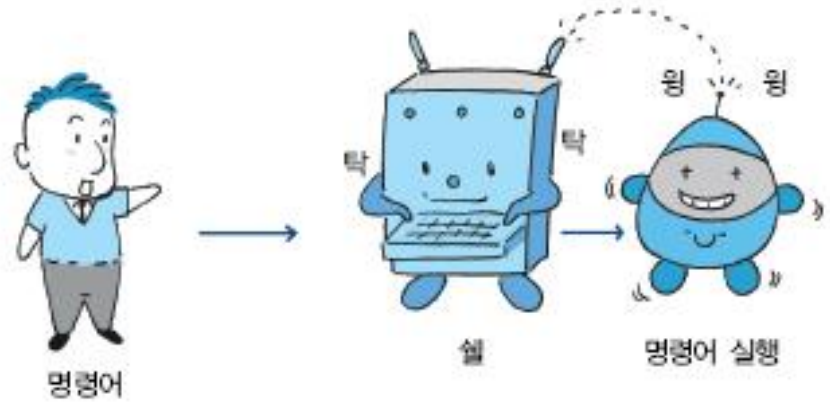


그림 6.1 쉘의 역할

쉘의 종류

- 유닉스/리눅스에서 사용 가능한 쉘의 종류

쉘의 종류	쉘 실행 파일
본 쉘	/bin/sh
콘 쉘	/bin/ksh
C 쉘	/bin/csh
Bash	/bin/bash
tcsh	/bin/tcsh

쉘의 종류

- 본 쉘(Bourne shell)
  - 벨연구소의 스티븐 본(Stephen Bourne)에 의해 개발됨
  - 유닉스에서 기본 쉘로 사용됨
- 콘 쉘(Korn shell)
  - 1980년대에는 역시 벨연구소에서 본 쉘을 확장해서 만듬.
- Bash(Bourne again shell)
  - GNU에서 본 쉘을 확장하여 개발한 쉘
  - 리눅스 및 맥 OS X에서 기본 쉘로 사용되면서 널리 보급됨
  - Bash 명령어의 구문은 본 쉘 명령어 구문을 확장함
- C 쉘(C shell)
  - 버클리대학의 빌 조이(Bill Joy)
  - 쉘의 핵심 기능 위에 C 언어의 특징을 많이 포함함
  - BSD 계열의 유닉스에서 많이 사용됨
  - 최근에 이를 개선한 tcsh이 개발됨어 되어 사용됨

로그인 쉘(login shell)

- 로그인 하면 자동으로 실행되는 쉘
- 보통 시스템관리자가 계정을 만들 때 로그인 쉘 지정

  /etc/passwd

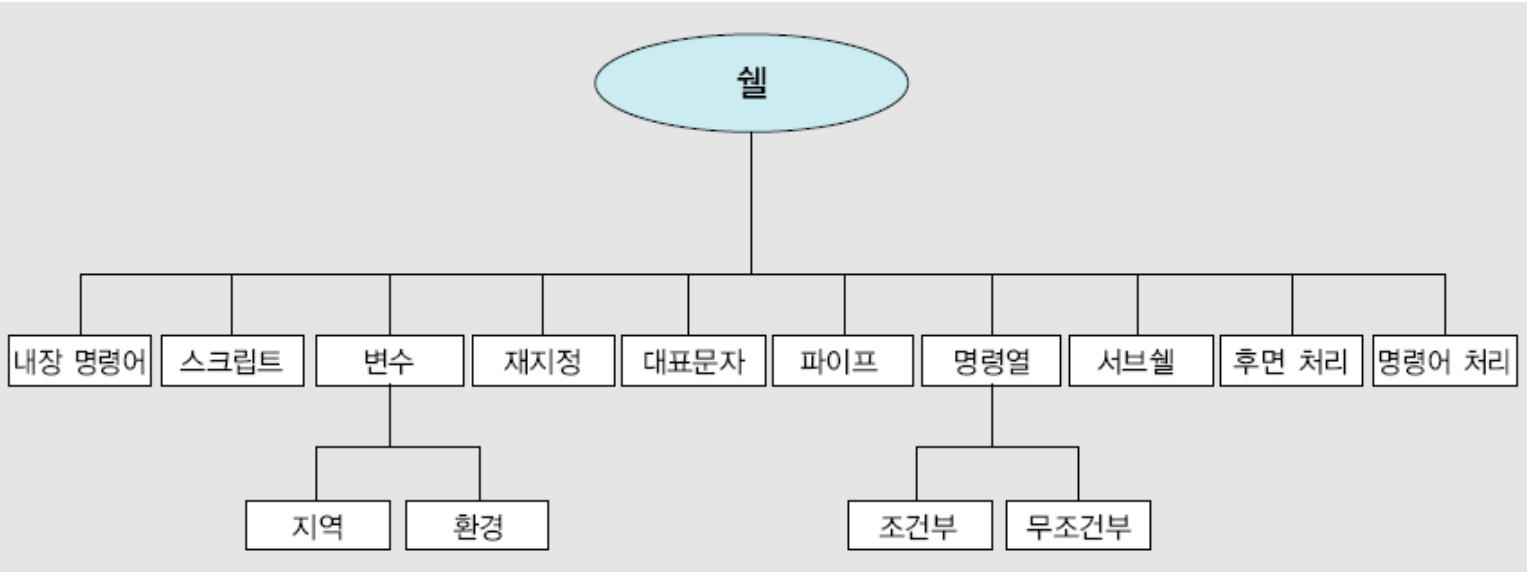
  ...

  chang:x:109:101:Byeong-Mo Chang:/user/faculty/chang:/bin/csh

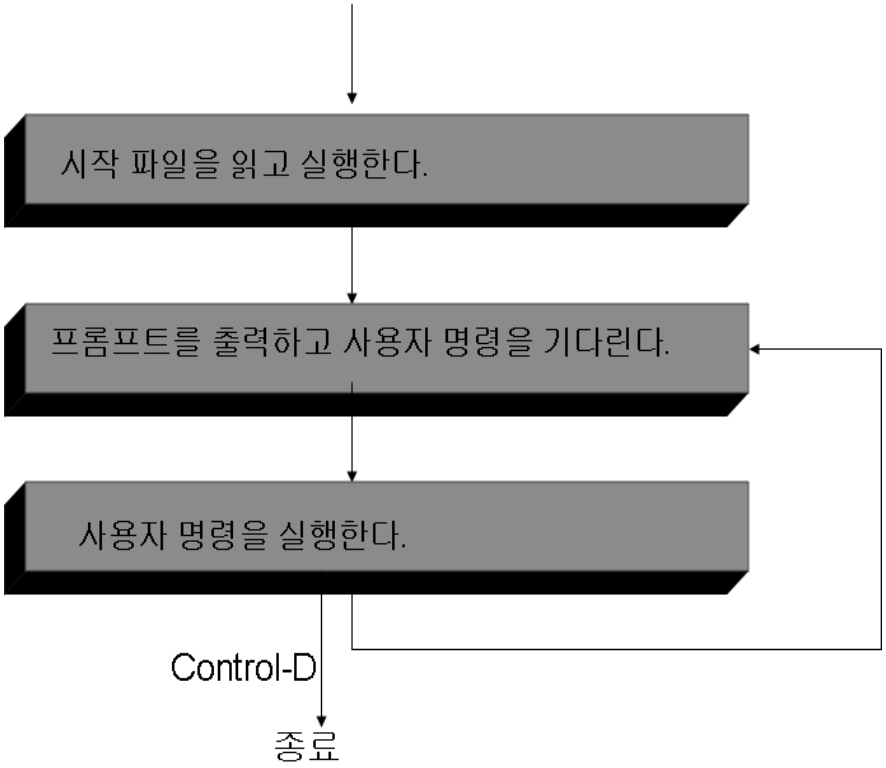
쉘의 기본 기능

쉘의 기본 기능

- 명령어 처리
  - 사용자가 입력한 명령을 해석하고 적절한 프로그램을 실행
- 시작 파일
  - 로그인할 때 실행되어 사용자 별로 맞춤형 사용 환경 설정
- 스크립트
  - 쉘 자체 내의 프로그래밍 기능



쉘의 실행 절차


시작 파일을 읽고 실행한다.
프롬프트를 출력하고 사용자 명령을 기다린다.
사용자 명령을 실행한다.
Control-D
종료

시작 파일(start-up file)

- 쉘마다 시작될 때 자동으로 실행되는 고유의 시작 파일
- 주로 사용자 환경을 설정하는 역할을 하며
- 환경설정을 위해서 환경변수에 적절한 값을 설정한다.

시작 파일(start-up file)

- 본 쉘

  /etc/profile

  ~/.profile

- bash

  /etc/profile : system wide environment and setup program for login setup

  /etc/bashrc : system wide functions and alias

  ~/.bash_profile : login shell에서 호출(일반적으로 내부에서.bashrc를 호출)

  ~/.bashrc : .bash_profile에서 호출

- C 쉘

  /etc/.login

  ~/.login

  ~/.cshrc

시작 파일 예

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH
```

- 시작 파일 바로 적용 방법(또는 쉘 시작시 자동 실행)

  $ . .bash_profile

  $ source .bash_profile

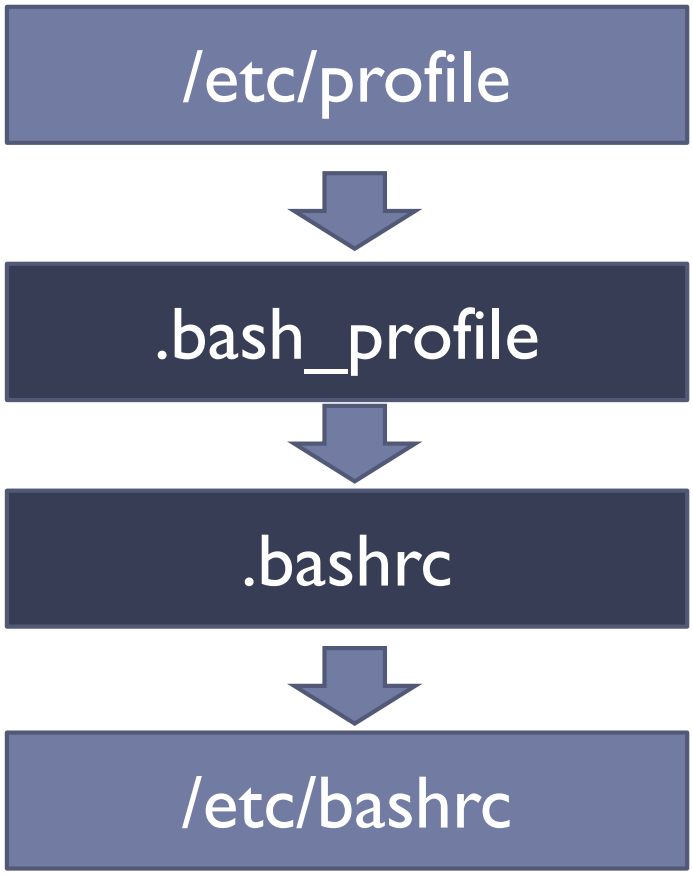
Bash

Bash 쉘 소개

Bash(Borune-again shell)

- 리눅스, 맥 OS X 등의 운영 체제의 기본 쉘
- Bash 문법은 본 쉘의 문법을 대부분 수용하면서 확장
- 시작 파일(start-up file)
  - /etc/profile
    전체 사용자에게 적용되는 환경 설정, 시작 프로그램 지정
  - /etc/bashrc
    전체 사용자에게 적용되는 별명과 함수들을 정의
  - ~/.bash_profile
    각 사용자를 위한 환경을 설정, 시작 프로그램 지정
  - ~/.bashrc
    각 사용자를 위한 별명과 함수들을 정의

로그인 후 Bash 시작 과정



system -wide

system -wide

시작 파일 예

```
# .bash_profile
# 사용자의 환경변수 설정 및 시작 프로그램
if [ -f ~/.bashrc ]
then
.  ~/.bashrc
fi

PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME="root"
export USERNAME BASH_ENV PATH
```

```
# .bashrc
# 사용자의 별명 설정
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
alias ll='ls -al --color=yes'
# 시스템 시작 파일 실행
if [ -f /etc/bashrc ]
then
.  /etc/bashrc
fi
```

별명 및 히스토리 기능

별명

- alias 명령어
  - 스트링이 나타내는 기존 명령에 대해 새로운 단어를 별명으로 정의

```
$ alias 단어=스트링
$ alias dir=ls -a|
$ dir

$ alias h=history
$ h
```

- 현재까지 정의된 별명들을 확인

```
$ alias          # 별명 리스트
dir ls -al
h history
```

- 이미 정의된 별명 해제

```
$ unalias 단어
```

히스토리

- 입력된 명령들을 기억하는 기능

  $ history [-rh] [번호]

```
$ history
1 ls
2 who
3 env
4 vi test.sh
5 chmod +x test.sh
6 test.sh
7 ls
8 date
9 history
...
```

재실행

형태	의미
!!	바로 전 명령 재실행
!n	이벤트 번호가 n인 명령 재실행
!시작스트링	시작스트링으로 시작하는 최후 명령 재실행

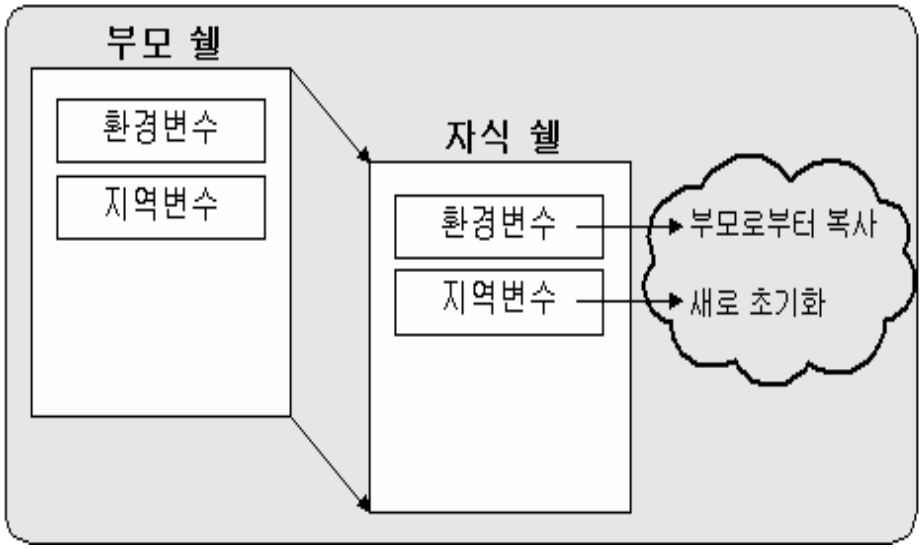
- 예

```
$ !!          # 바로 전 명령 재실행
$ !20         # 20번 이벤트 재실행
$ !gcc        # gcc로 시작하는 최근 명령 재실행
```

지역변수와 환경변수

환경변수와 지역변수

- 쉘 변수
  - 환경변수와 지역변수 두 종류로 나눌 수 있다.
  - 환경 변수는 값이 자식 프로세스에게 상속되며 지역변수는 그렇지 않다.



환경변수와 지역변수 예

```
$ country=대한민국 city=서울
$ export country
$ echo $country $city
대한민국 서울

$ sh
$ echo $country $city
대한민국
$ exit

$ echo $country $city
대한민국 서울
```

사전 정의 환경변수(predefined environment variable)

- 그 의미가 미리 정해진 환경변수들

이름	의미
$USER	사용자 이름
$TERM	터미널 타입
$PATH	명령어를 검색할 디렉터리들의 리스트
$HOME	홈 디렉터리
$SHELL	로그인 쉘의 경로명
$MAIL	메일 박스의 경로명
$HOSTNAME	호스트 이름

사전 정의 환경변수(predefined environment variable)

- 환경변수 $PS1
  - Prompt 모양에 관한 변수
    - 예) 기본값 PS1="[₩u@₩h ₩W]₩$ "
    - 기타 정의
      - ₩t : 시간
      - ₩d : 날짜
      - ₩s : shell 이름
      - ₩w : 작업 디렉토리
      - ₩# : 명령 번호
      - ₩! 등
  - 수정하려면
    - .bashrc 파일에서 export PS1=“형태정의” 를 삽입한 후에 source .bashrc를 실행하여 즉시 적용