



응용 SW 기초 활용 기술 part 1

# 리눅스 쉘 스크립트



한국기술교육대학교  
온라인평생교육원



- 셸(Shell)
- 셸 스크립트(Shell Script)



- 셸의 개념을 설명하고, 셸의 기능을 활용할 수 있다.
- 셸 스크립트 문법을 설명하고, 셸 스크립트를 작성하여 실행할 수 있다.

# 셸(Shell)

## 1 셸의 기본

### 1) 셸의 개요

#### 1 명령어 해석기 제공

- 사용자의 명령어를 입력받아, 커널을 통해 명령을 실행시킴

#### 2 환경변수 및 지역변수 제공

#### 3 셸 스크립트 제공

- 리눅스 운영에 필요한 프로그램 작성

#### 4 다양한 셸을 제공(선택 가능)

- 셸에 따라서 사용하는 명령어 문법 및 셸 스크립트 문법이 다르기 때문에 사용자가 편리한 셸을 선택해서 사용할 수 있음
- 사용자가 셸을 선택하기 위해서는 패스워드 파일을 수정하면 됨
  - /etc/passwd

# 셸(Shell)

## 1 셸의 기본

### 2) 셸의 종류

#### Bourne셸 계열

- 비에이셸(bash)
- 본셸(bsh)
- 콘셸(ksh)

#### C셸 계열

- 씨셸(csh)
- 티씨셸(tcsh)

#### 비에이셸

- 리눅스 설치 시, 기본적으로 설정되는 셸
- 배쉬셸이라고도 부름
- 콘셸과 씨셸의 장점을 모아 만든 셸



## 셸(Shell)

### 1 셸의 기본

#### 3) bash 기능

기능	설명
별명 기능	<ul style="list-style-type: none"> <li>■ 명령어 단축 기능</li> <li>■ # alias ls= 'ls -l'</li> <li>■ # alias c='clear'</li> </ul>
history 기능	<ul style="list-style-type: none"> <li>■ 사용한 명령어 저장</li> <li>■ # history</li> <li>■ # ↑↓</li> </ul>
자동 이름 완성 기능	■ 탭(Tab)키를 이용한 파일이름 자동완성
프롬프트 제어 기능	<ul style="list-style-type: none"> <li>■ 환경변수에 셸 프롬프트 정보 변경</li> <li>■ # PS1='\$LOGNAME@ \$PWD #'</li> </ul>
명령 편집 기능	■ 프롬프트상에서 명령어 수정 기능

## 셸(Shell)

### 2 환경변수와 지역변수

#### 1) 환경변수 개요

##### 환경변수

리눅스 운영체제가 제공하는 **시스템 설정값**

환경변수	설명
HOME	로그인한 사용자의 홈디렉토리
PWD	현재 작업 디렉토리
LOGNAME, USER, USERNAME	로그인한 사용자 이름
PS1	셸 프롬프트
HOSTNAME	호스트의 이름
SHELL	로그인하여 사용 중인 셸의 종류



## 셸(Shell)

### 2 환경변수와 지역변수

#### 2) 환경변수 관련 명령어

##### 전체 환경변수 보기

- # printenv
- # env

##### 특정 환경변수 보기

- # echo \$환경변수

##### 환경변수 설정

- # 환경변수='값'
- # export 환경변수='값'



- 부모 셸 : 로그인하여 사용하는 셸
- 자식 셸 : 부모 셸에서 파생되는 셸

## 셸(Shell)

### 2 환경변수와 지역변수

#### 3) 지역변수

지역변수

임시로 만들어 사용하는 변수

##### 지역변수 보기

■ # echo \$환경변수

##### 지역변수 설정

■ # 환경변수='값'

■ # export 환경변수='값'



## 셸 스크립트(Shell Script)

### 1 셸 스크립트 작성과 실행

#### 1) 셸 스크립트 개요

1 C언어와 유사

2 변수, 반복문, 제어문 사용 가능

3 스크립트 언어이기 때문에 컴파일이 필요 없음

셸 스크립트 프로그램은 명령어들을 나열해 놓는 **배치(batch) 파일의 형태임**

## 셸 스크립트(Shell Script)

### 1 셸 스크립트 작성과 실행

#### 2) 셸 스크립트 작성

셸 스크립트 작성 도구    vi 편집기, gedit

셸 스크립트 파일 확장자    \*.sh

셸을 사용하겠다는 의미  
첫 칸 #은 설명문으로 취급하며, 프  
로그램 실행과 상관 없이 작성 가능

셸 스크립트 프로그램 작성

정상적으로 종료된다는 의미로, 종  
료코드 0을 운영체제에 반환

#### 셸 스크립트 파일의 기본 형태

```
#!/bin/sh
```

첫 줄

```
exit 0
```

본문

마지막 줄

## 셸 스크립트(Shell Script)

### 1 셸 스크립트 작성과 실행

#### 3) 셸 스크립트 실행

##### sh 명령 이용

```
# sh info.sh
```

##### 실행권한 부여

```
# chmod ugo+x info.sh  
# ./info.sh
```

## 셸 스크립트(Shell Script)

### 2 셸 스크립트 문법

#### 1) 변수

##### 1 변수 선언 없음

- 처음 값이 할당되면 변수 생성

##### 2 변수의 모든 값은 문자열로 취급함

##### 3 변수명은 대소문자를 구분해야 함

##### 4 대입 연산자(=) 좌우에 공백이 없어야 함

##### 5 대입하는 문자열은 " 또는 " "로 묶음

- " 또는 " "로 묶지 않는 경우에는 공백이 없어야 함

##### 6 변수값의 출력

- 변수명 앞에 \$를 붙임

##### 7 변수값의 입력

- read 이용

## 셸 스크립트(Shell Script)

### 2 셸 스크립트 문법

#### 2) 숫자 계산

사칙 연산이 필요한 경우 : 'expr' 키워드 사용

'expr' 키워드가 들어가는 연산식은 ``로 묶음

() 또는 \* 기호 사용 시, \와 함께 사용

## 셸 스크립트(Shell Script)

### 2 셸 스크립트 문법

#### 3) 기타 문법

**파라미터(Parameter)**    셸 스크립트를 실행할 경우 인자로 지정된 값을 저장하는 변수



파라미터(Parameter) 변수이름 : \$0, \$1, \$2, ...

문자열을 명령으로 인식하여 실행 : **eval**

반복 및 조건문, 비교, 산술, 논리 연산자, 파일 조건문, 사용자 정의 함수 등

- if, if~else, case~esac, for~in, while, until, break, continue, exit, return, printf, set, \$(명령어), shift

## 셸 스크립트(Shell Script)

### 3 셸 초기화 파일 설정

#### 1) 셸 초기화 파일 개요

셸 초기화 파일

시스템 사용 설정 파일

리눅스가 부팅될 때,  
모든 사용자에게  
적용되도록 실행되는  
셸 스크립트 파일임

모든 사용자를 위한 파일

- 1차 초기화 파일 : `/etc/profile`
- 2차 초기화 파일 : `/etc/bashrc`

홈디렉토리에 존재하는  
숨김파일이며,  
각 사용자의 로그인 시 적  
용됨

각 사용자를 위한 파일

- 1차 초기화 파일 : `$HOME/.bash_profile`
- 2차 초기화 파일 : `$HOME/.bashrc`

## 셸 스크립트(Shell Script)

### 3 셸 초기화 파일 설정

#### 2) 셸 초기화 파일 역할

##### 1차 초기화 파일

- \$HOME/.bash\_profile
- 2차 초기화 파일 실행
- 환경변수 설정

##### 2차 초기화 파일

- \$HOME/.bashrc
- 명령어 단축 별칭 설정
- 기타 셸 설정



사용자가 편리하도록  
초기화 파일을 수정하여 사용



## 셸(Shell)

---

### + 셸의 개요

- 명령어 해석기 제공
- 환경변수 및 지역변수 제공
- 셸 스크립트 제공
- 다양한 셸을 제공하며, 선택이 가능 : /etc/passwd

### + 셸의 종류

- 본셸 계열, C셸 계열

### + bash 셸의 기능

- History 기능, 자동이름 완성 기능, 프롬프트 제어 기능, 명령 편집 기능

### + 환경변수의 대표적인 종류

- HOME, PWD, LOGNAME, USER, USERNAME, PS1, HOSTNAME, SHELL

## 셸 스크립트( Shell Script )

---

### + 셸 스크립트의 개요

- C언어와 유사함
- 변수, 반복문, 제어문 사용 가능함
- 컴파일 필요 없음
- 명령어 배치(batch) 파일

### + 셸 스크립트 작성

- vi 편집기 또는 gedit 사용
- 셸 스크립트 파일 확장자 : \*.sh

### + 셸 스크립트 파일 구조

- 첫 줄 : `#!/bin/sh`
- 마지막 줄 : `exit 0`

### + 셸 스크립트 실행

- sh 명령 이용 : `# sh info.sh`
- 실행권한 부여 : `# chmod ugo+x info.sh, # ./info.sh`

### + 셸 스크립트 문법(변수)

- 변수 선언 없음 : 처음 값이 할당되면 변수 생성
- 변수의 모든 값은 문자열로 취급
- 대소문자 구분
- 대입 연산자(=) 좌우에 공백이 없어야 함
- 대입하는 문자열은 " 또는 " "로 묶음 ( ' 또는 ' "로 묶지 않는 경우는 공백이 없어야 함)
- 변수값의 출력 : 변수명 앞에 \$를 붙임.
- 변수값의 입력 : read 이용

## 셸 스크립트( Shell Script )

### + 셸 스크립트 문법(숫자 계산)

- 사칙 연산이 필요한 경우 'expr' 키워드 사용
- 연산식에는 역 따옴표( ` )로 묶음
- 역슬래쉬(\)와 함께 사용 : 괄호(()), 곱셈(\*) 기호

### + 파라미터(Parameter) 변수이름

- \$0, \$1, \$2, ...

### + 문자열을 명령으로 인식하여 실행

- eval

### + 반복 및 조건문, 비교, 산술, 논리 연산자, 파일 조건문, 사용자 정의 함수

- if, if~else, case~esac, for~in, while, until, break, continue, exit, return, printf, set, \$(명령어), shift

### + 셸 초기화 파일

#### 모든 사용자를 위한 파일

- 1차 초기화 파일 : /etc/profile
- 2차 초기화 파일 : /etc/bashrc

#### 각 사용자를 위한 파일

- 1차 초기화 파일 :  
\$HOME/.bash\_profile
- 2차 초기화 파일 : \$HOME/.bashrc



### + 마운트 관련 명령어

기능	설명
별명 기능	<ul style="list-style-type: none"> <li>■ 명령어 단축 기능</li> <li>■ # alias ls='ls -l'</li> <li>■ # alias c='clear'</li> </ul>
History 기능	<ul style="list-style-type: none"> <li>■ 사용한 명령어 저장</li> <li>■ # history</li> <li>■ # ↑↓</li> </ul>
자동 이름 완성 기능	<ul style="list-style-type: none"> <li>■ 탭(tab)키를 이용한 파일이름 자동완성</li> </ul>
프롬프트 제어 기능	<ul style="list-style-type: none"> <li>■ 환경변수에 쉘 프롬프트 정보 변경</li> <li>■ # PS1='\$LOGNAME@ \$PWD #'</li> </ul>
명령 편집 기능	<ul style="list-style-type: none"> <li>■ 프롬프트상에서 명령어 수정 기능</li> </ul>

명령	설명
mount	마운트 정보 보기
mount 장치명 디렉토리명	마운트하기
umount 장치명	마운트 해제하기