### [[리눅스 쉘(Shell) 스크립트]](/241)

[SW개발](/category/SW개발) 2012/08/17 07:53

**1. 쉘(Shell)스크립트란 ?**

* 쉘을 통하여 프로그램을 만든 것
* DOS의 Batch 파일 작성과 비슷하지만 더 막강한 기능들을 가지고 있음
* 기존의 명령어들을 이용하여 만들어지는 스크립트 파일의 문법은 다른 프로그래밍 언어보다 매우 간단하며 쉽게 응용이 가능

**2. 쉘 스크립트의 시작**

* 쉘 프로그램을 만들기 위해서 vi에디터나 gedit 등을 이용
* 스크립트 파일의 가장 위의 첫 라인은 "#! /bin/bash" 로 시작
* bash로 작성되었다는 것을 의미하며, bash을 기준으로 실행된다는 것을 의미
* Bourne Shell인 경우 "#! /bin/sh"

**3. 쉘 스크립트의 실행**

아래와 같이 echo 를 이용하여 화면에 "bash script test!!" 를 출력 할 수 있도록 스크립트 작성

아래와 같이 작성한 스크립트 파일에 실행 권한이 있는지 확인하고, 없으면 실행 권한을 추가하고 실행을 하면 작성한 메시지가 Display되는 것을 볼 수 있다.

**4. 변수 사용**

* Shell에 있어 변수의 사용은 매우 간단히 정의
* 변수명=값
* 주의 할 것은 '=' 사이의 좌우 공백이 없어야 함
* 대소문자 구별
* 숫자형 변수와 문자형 변수의 구별이 없음
* a=10
* b="헬로우 리눅스"
* 변수의 내용을 사용할 때는 변수명 앞에 '$'를 붙인다.

**5. 숫자 계산을 위한 연산자**

* expr : 숫자 계산
* + : 더하기
* - : 빼기
* \* : 곱하기('\\*'로 사용)
* /: 나누기
* %: 나누기를 한 수 나머지 값
* 주의
  + expr로 지정하는 경우 역작은따옴표( ` )를 사용 : 작은 따옴표( ' )가 아님 !
  + 연산자 \*와 괄호 앞에는 역슬래스(\)를 같이 사용
  + 연산자와 숫자, 변수, 기호 사이에는 space가 존재

**Ex1)**

**결과)**

**Ex2)**

retult=`expr \( 1 + 2 - 1 \) \\* 2`

**6. 사용자의 입력 받기**

read를 사용하여 사용자의 입력을 변수에 받을 수 있다.

**Ex)**

Echo에 –n을 붙이면 줄넘김을 하지 않음

**결과)**

**7. 조건문 (if, elif, else)**

if로 시작해서 fi로 끝난다**.**

**8. 숫자 비교**

[$A -gt $B] :        A가 B보가 크다

[$A -lt $B] :        A가 B보다 작다.

[$A -ge $B] :        A가 B보다 크거나 같다.

[$A -le $B] :        A가 B보다 작거나 같다.

[$A -eq $B] :        A와 B가 같다.

[$A -ne $B] :        A와 B가 다르다.

**Ex)**

**9. 문자 비교**

["string1" = "string2"] :    두 문자열이 같은 경우(== 도 가능)

["string1" ! "string2"] :    두 문자열이 다른 경우

[-z "string"] :            문자열의 길이가 0인 경우

[-n "string"] :            문자열의 길이가 0이 아닌경우

**10. 기타 조건**

[-e "path"] :        path의 파일이나 디렉토리가 있는 경우

**11. 관계 연산자**

-a : and

-o : or

**Ex)**

**12. for 문**

* 쉘 프로그램에서의 for 문은 일반적인 다른 프로그래밍 언어에서 사용되는 for 문의 형태와 다름
* in 다음에 나오는 변수값을 순차적으로 변수에 대입하면서 반복 실행
* 변수의 개수만큼 루프를 돌면서 do 다음의 문장을 실행하고 done으로 마무리

for 변수 in 변수값1 변수값2 변수값3 변수값4 …

do

    문장

done

**Ex1) 숫자를 하나씩 출력**

**Ex2) ls 의 결과를 하나씩 출력**

**13. while 문**

* 조건문이 항상 참이라면, do와 done 사이의 문장이 무한 루프
* ':' 기호와 1은 항상 참을 나타냄,
* break, continue 사용 가능

while [ 조건문 ]

do

    문장…

done

**Ex)**

**14. until 문**

* while 문과 같지만 반대로 동작
* 거짓이면 루프를 돌고 참이면 루프를 종료
* break, continue 사용 가능

until [ 조건문 ]

do

    문장…

done

**15. case 문**

변수 값을 받아들여서, 이 값에 맞는 문장을 실행

case 변수 in

패턴1)

    문장

    끝문장;;

패턴2)

    문장;;

     끝문장;;

\*)

    문장

     끝문장;;

esac

**Ex)**

**결과)**

**16. 함수**

* 함수는 불러주기 전에 선언되어야 함
* C에서처럼 함수를 미리 선언하는 방법은 없음

**16. 종료**

* exit : shell을 종료

**17. Bash 주요 환경변수**

|  |  |
| --- | --- |
| **Environment variable** | **의          미** |
|
| ENV | .bashrc 파일에서 설정하며 함수와 별명 등을 설정하는 환경 파일의 이름을 설정 |
| EUID | Shell이 시작할 때 현재 사용자의 유효 ID를 확장한다. |
| HISTFILE | 명령어 히스토리를 저장할 파일이름. 기본값은 ~/.bash\_history이다. |
| HOME | 사용자의 홈 디렉토리의 값을 가지고 있으며 특정 디렉토리를 지정하지 않고 cd 명령을 사용할 때 이용한다. |
| LANG | LC\_로 시작하는 변수들에 포함되지 않는 로케일을 설정 |
| PATH | 명령어의 검색 경로, 콜론으로 구분한 디렉토리 목록 |
| PPID | 부모프로세스의 PID |
| PWD | 현재 작업 디렉토리 |
| SHELL | 현재 사용되어지고 있는 SHELL |
| UID | 현재 사용자의 UID |
| OLDPWD | 이전 작업 디렉토리 (이전 한개만 저장) |

#### '[SW개발](/category/SW개발)' 카테고리의 다른 글

|  |  |
| --- | --- |
| [**[3GPP2]**](/254)**(0)** | 2012/09/03 |
| [**[QXDM] [로그 저장하고 불러오기]**](/248)**(2)** | 2012/08/24 |
| [**[리눅스 쉘(Shell) 스크립트]**](/241)**(0)** | 2012/08/17 |
| [**[Linux와 Shell]**](/238)**(0)** | 2012/08/14 |
| [**[VI 글자 색상 바꾸기]**](/237)**(2)** | 2012/08/13 |
| [**[Android]서비스의 라이프 사이클**](/235)**(0)** | 2012/08/10 |

#### 설정

[TRACKBACK : 0](#tb) , [COMMENT : 0](#rp)

#### 트랙백

http://gongdoo.tistory.com/trackback/241

#### 댓글