

Bash 빠른 참조

명령어, 스크립팅, 파이프, 리다이렉션, 작업 제어

기본

echo & 탐색

```
echo "Hello, World!" # print text
pwd                 # print working directory
cd /path/to/dir    # change directory
cd ..              # go up one level
cd ~               # go to home directory
cd -               # go to previous directory
```

파일 목록 & 생성

```
ls                 # list files
ls -la            # long format, show hidden
ls -lh            # human-readable sizes
mkdir mydir       # create directory
mkdir -p a/b/c    # create nested directories
```

복사, 이동 & 삭제

```
cp file.txt copy.txt # copy file
cp -r dir/ backup/   # copy directory recursively
mv old.txt new.txt   # rename / move
rm file.txt           # delete file
rm -r dir/            # delete directory recursively
rm -rf dir/          # force delete (no prompt)
```

변수 & 확장

변수

```
name="Alice"        # assign (no spaces!)
echo "$name"        # variable expansion
echo "${name}_file" # braces for clarity
readonly PI=3.14     # constant
unset name           # delete variable
```

특수 변수

```
$0      스크립트 이름
$1 $2 ... 위치 인수
 $#     인수 개수
 $@     모든 인수 (개별 단어)
 $*     모든 인수 (단일 문자열)
 $?     마지막 명령의 종료 상태
 $$     현재 프로세스 ID
 !$     마지막 백그라운드 프로세스 PID
```

명령 치환 & 산술

```
files=$(ls) # capture output
today=$(date +%Y-%m-%d) # command substitution
count=$((5 + 3)) # arithmetic: 8
echo $(10 / 3) # integer division: 3
echo $(10 % 3) # modulo: 1
```

문자열 연산

```
${#str}      문자열 길이
${str:0:5}   부분 문자열 (오프셋:길이)
${str/old/new} 첫 번째 매칭 교체
${str//old/new} 모든 매칭 교체
${str^^}     대문자 변환
${str,,}     소문자 변환
```

조건문

if / elif / else

```
if [[ "$name" == "Alice" ]]; then
  echo "Hi Alice"
elif [[ "$name" == "Bob" ]]; then
  echo "Hi Bob"
else
  echo "Who are you?"
fi
```

테스트 연산자

```
-eq -ne 정수 같음 / 다름
-lt -gt 정수 작음 / 큼
-le -ge 정수 작거나 같음 / 크거나 같음
== != 문자열 같음 / 다름
-z "$str" 문자열이 비어 있음
-n "$str" 문자열이 비어 있지 않음
-f file 파일이 존재하고 일반 파일
-d dir 디렉토리 존재
-e path 경로 존재 (모든 유형)
-r -w -x 읽기 가능 / 쓰기 가능 / 실행 가능
&& || 논리 AND / OR
```

반복문

for 반복문

```
for fruit in apple banana cherry; do
  echo "$fruit"
done

for f in *.txt; do
  echo "File: $f"
done
```

C 스타일 for 반복문

```
for ((i=0; i<5; i++)); do
  echo "$i"
done
```

while 반복문

```
count=0
while [[ $count -lt 5 ]]; do
  echo "$count"
  ((count++))
done
```

반복문 제어

```
break 반복문 종료
continue 다음 반복으로 건너뛸
```

함수

정의 & 호출

```
greet() {
  echo "Hello, $1!" # $1 = first arg
  return 0          # exit status
}
greet "Alice"      # Hello, Alice!
```

지역 변수 & 반환값

```
add() {
  local sum=$(( $1 + $2 ))
  echo "$sum" # "return" via stdout
}
result=$(add 3 5) # capture: 8
```

파이프 & 리다이렉션

파이프

```
ls -l | grep ".txt" # pipe output
cat log | sort | uniq # chain commands
cmd1 | tee out.txt # pipe + save to file
```

리다이렉션

```
cmd > file stdout 리다이렉트 (덮어쓰기)
cmd >> file stdout 리다이렉트 (추가)
cmd < file 파일에서 stdin 읽기
cmd 2> file stderr 리다이렉트
cmd 2>&1 stderr를 stdout으로 리다이렉트
cmd &> file stdout + stderr 리다이렉트
cmd << EOF 히어 다큐먼트 (인라인 입력)
/dev/null 출력 폐기: cmd > /dev/null
```

파일 작업

파일 보기

```
cat file.txt # print entire file
head -n 10 file.txt # first 10 lines
tail -n 10 file.txt # last 10 lines
tail -f log.txt # follow (live updates)
less file.txt # paginated viewer
```

계산 & 검색

```
wc -l file.txt # count lines
wc -w file.txt # count words
wc -c file.txt # count bytes
find . -name "*.txt" # find by name
find . -type d # find directories
find . -mtime -7 # modified in last 7 days
```

기타 파일 명령

```
touch file 파일 생성 / 타임스탬프 갱신
stat file 파일 메타데이터 (크기, 날짜)
file img.png 파일 유형 감지
diff a.txt b.txt 두 파일 비교
sort file.txt 줄 정렬
uniq 인접 중복 제거
cut -d: -f1 구분자로 필드 추출
tr 'a-z' 'A-Z' 문자 변환 / 교체
```

텍스트 처리

grep

```
grep "error" log.txt # search for pattern
grep -i "error" log.txt # case-insensitive
grep -r "TODO" src/ # recursive search
grep -n "func" file.go # show line numbers
grep -c "error" log.txt # count matches
grep -v "debug" log.txt # invert match
```

sed

```
sed 's/old/new/' file # replace first per line
sed 's/old/new/g' file # replace all
sed -i 's/old/new/g' file # edit in place
sed -n '5,10p' file # print lines 5-10
sed '/pattern/d' file # delete matching lines
```

Bash 빠른 참조

awk

```
awk '{print $1}' file # print first field
awk -F: '{print $1}' file # custom delimiter
awk '$3 > 100' file # filter by field value
awk '{sum+=$1} END{print sum}' file # sum column
```

퍼미션

chmod

```
chmod 755 script.sh # rwxr-xr-x
chmod +x script.sh # add execute
chmod -w file.txt # remove write
chmod u+x,g-w file # user +exec, group -write
```

퍼미션 참조

r (4)	읽기
w (2)	쓰기
x (1)	실행
u / g / o	소유자/그룹/기타
755	소유자: rwx, 그룹/기타: r-x
644	소유자: rw-, 그룹/기타: r--

소유권

```
chown user file.txt # change owner
chown user:group file.txt # change owner + group
chown -R user:group dir/ # recursive
```