find 빠른 참조

이름, 타입, 크기, 시간, 권한, 동작으로 파일 검색

기본 검색

find 실행

```
find .                              # list all files recursively
find /var/log                       # search from specific path
find . -name "*.txt"               # find by name
find / -name "config" 2>/dev/null  # suppress permission errors
find dir1 dir2 -name "*.py"        # search multiple directories
```

문법 개요

find [path...] [tests] [actions]	일반 형식 — 경로, 조건, 동작 순
path	시작 디렉터리 (기본: 현재 디렉터리)
test	파일 필터 조건 (**-name**, **-type** 등)
action	일치 항목에 할 작업 (**-print**, **-exec** 등)
Default action	동작 미지정 시 **-print**

이름 패턴

이름 일치

```
find . -name "*.log"               # case-sensitive glob
find . -iname "readme*"            # case-insensitive glob
find . -name "*.py" -o -name "*.js"  # OR: Python or JS files
find . -path "*/src/*.ts"          # match against full path
find . -regex '.*\.\(py\|js\)'    # POSIX regex on full path
```

이름 조건

-name pattern	파일명이 셸 글로브와 일치 (대소문자 구분)
-iname pattern	파일명이 글로브와 일치 (대소문자 무시)
-path pattern	전체 경로가 셸 글로브와 일치
-ipath pattern	전체 경로가 글로브와 일치 (대소문자 무시)
-regex pattern	전체 경로가 정규식과 일치
-iregex pattern	전체 경로가 정규식과 일치 (대소문자 무시)

타입 필터

타입으로 필터

```
find . -type f                     # regular files only
find . -type d                     # directories only
find . -type l                     # symbolic links
find . -type f -name "*.sh"        # combine type + name
```

파일 타입

-type f	일반 파일
-type d	디렉터리
-type l	심볼릭 링크
-type b	블록 장치
-type c	문자 장치
-type p	명명된 파이프 (FIFO)
-type s	소켓
-empty	빈 파일 또는 디렉터리

크기 & 시간

크기 & 시간 예시

```
find . -size +100M                 # larger than 100 MB
find . -size -1k                   # smaller than 1 KB
find . -mtime -7                   # modified in last 7 days
find . -mmin -30                   # modified in last 30 minutes
find . -newer reference.txt        # newer than reference file
```

크기 & 시간 조건

-size +/-Nc	바이트(c), KB(k), MB(M), GB(G) 단위 크기
-mtime +/-N	N*24시간 전 수정 (+이전, -이후)
-atime +/-N	N*24시간 전 접근
-ctime +/-N	N*24시간 전 상태 변경
-mmin +/-N	N분 전 수정
-newer file	파일보다 최근에 수정됨
-newermt date	날짜 문자열 이후 수정됨 (GNU)

권한

권한 예시

```
find . -perm 644                   # exact permissions: rw-r--r--
find . -perm -u+x                  # user has execute bit set
find . -perm /o+w                  # others have write (any match)
find . -user root                  # owned by root
find . -group www-data -type f     # owned by group
```

권한 조건

-perm mode	정확한 권한 일치
-perm -mode	지정한 비트가 모두 설정됨
-perm /mode	지정한 비트 중 하나라도 설정됨
-user name	사용자 소유 (이름 또는 UID)
-group name	그룹 소유 (이름 또는 GID)
-nouser	/etc/passwd에 일치하는 사용자 없음
-nogroup	/etc/group에 일치하는 그룹 없음

동작

동작 예시

```
find . -name "*.log" -print        # print paths (default)
find . -name "*.tmp" -delete       # delete matching files
find . -type f -ls                 # detailed listing
find . -name "*.txt" -print0       # null-delimited output
find . -type f -printf "%p %s\n"   # custom format (GNU)
```

동작 참조

-print	경로 출력 (개행 구분)
-print0	경로 출력 (null 구분, xargs 안전)
-ls	파일 세부 정보 출력 (**ls -dils** 형식)
-delete	일치하는 파일 삭제 (**-depth** 포함)
-printf format	커스텀 출력 형식 (GNU): **%p** 경로, **%s** 크기, **%t** 시간
-fprint file	경로를 파일에 기록
-quit	첫 번째 일치 후 종료

조건 결합

논리 연산자

```
find . -name "*.py" -type f         # implicit AND
find . -name "*.py" -a -size +10k   # explicit AND
find . -name "*.py" -o -name "*.js" # OR
find . ! -name "*.log"              # NOT
find . \( -name "*.py" -o -name "*.js" \) -type f
```

연산자 참조

expr1 expr2 / expr1 -a expr2	AND — 둘 다 참이어야 함 (기본)
expr1 -o expr2	OR — 하나라도 참이면 됨
! expr / -not expr	NOT — 표현식 부정
\(expr \)	표현식 그룹화 (셸에서 괄호 이스케이프)
Evaluation order	좌에서 우로; **-a**가 **-o**보다 강하게 묶임

Exec & 삭제

Exec 예시

```
find . -name "*.sh" -exec chmod +x {} \;
find . -name "*.log" -exec rm {} +
find . -type f -exec grep -l "TODO" {} +
find . -name "*.bak" -ok rm {} \;   # prompt before each
find . -name "*.tmp" -print0 | xargs -0 rm
```

Exec 참조

-exec cmd {} \;	파일당 한 번 cmd 실행 (**{}** = 파일 경로)
-exec cmd {} +	여러 파일을 한 번에 cmd 실행 (더 빠름)
-ok cmd {} \;	**-exec**와 같지만 확인 요청
-execdir cmd {} \;	파일의 디렉터리에서 cmd 실행
xargs -0	**-print0**과 함께 사용, 안전한 일괄 처리
-delete	파일 삭제; 가장 깊은 것부터 처리

깊이 & 가지치기

깊이 & 가지치기 예시

```
find . -maxdepth 1 -type f          # current dir only
find . -mindepth 2 -name "*.py"     # skip top-level
find . -name ".git" -prune -o -print # skip .git dirs
find . -depth -name "*.tmp" -delete  # process children first
```

깊이 옵션

-maxdepth N	최대 N 레벨까지 탐색 (0 = 시작 경로만)
-mindepth N	N 레벨 미만에서는 조건 적용 안 함
-depth	디렉터리 자체보다 내용을 먼저 처리
-prune	일치한 디렉터리 하위 탐색 안 함
-mount / -xdev	파일시스템 경계를 넘지 않음
-follow / -L	심볼릭 링크 따라가기

일반 패턴

한 줄 명령

```
find . -name "*.pyc" -delete             # clean Python bytecode
find . -type f -size 0 -delete            # remove empty files
find . -mtime +30 -name "*.log" -delete   # purge old logs
find . -type f -name "*.md" | wc -l       # count Markdown files
find . -type d -empty -delete             # remove empty dirs
```

레시피

Find largest files	**find . -type f -printf '%s %p\n' | sort -rn | head**
Find duplicates by name	**find . -type f | awk -F/ '{print $NF}' | sort | uniq -d**
Rename extension	**find . -name '*.txt' -exec rename 's/.txt/.md/' {} +**
Find broken symlinks	**find . -xtype l**
Archive recent files	**find . -mtime -7 -print0 | tar czf recent.tar.gz --null -T -**
Search code files	**find . -name '*.py' -exec grep -l 'pattern' {} +**