

# Programming in shell 1

## File searching in a directory hierarchy.

Jan Trdlička



Czech Technical University in Prague, Faculty of Information Technology  
Department of Computer Systems

## 1 find

- Tests: `-name`, `-refex`, `-type`, `-size`, `-perm`,...
- Actions: `-print`, `-ls`, `-exec -ok`
- Operators: `\( \)`, `\!`, `-a`, `-o`

`find [options] [starting_point] [expression]`

- The `find` utility searches the directory tree rooted at given starting-point by evaluating the given expression from left to right.
- **Expression**
  - A kind of query specification describing how we match files and what we do with the files that were matched.
  - An expression is composed of a sequence of
    - **Tests** return a true or false value, usually on the basis of some property of a file we are considering.
    - **Actions** have side effects (such as printing something on the standard output or executing some command).
    - **Operators** join together the other items within the expression.
    - `-maxdepth n` ... descend at most  $n$  levels of directories below the starting-points.
    - `-mindepth n` ... do not apply any tests or actions at levels less than  $n$ .

# find – tests

- **-name** *pattern* ... true if *pattern* matches the basename.
  - Pattern can consists the following meta-characters
    - **\*** ... matching zero or more characters.
    - **?** ... matching exactly one character.
    - **[ ]** ... matching one character in the set or in the range.
    - **[^ ]** ... matching one character not in the set or in the range.
- **-regex** *pattern* ... true if *pattern* matches the whole path.
  - Emacs Regular Expressions are supported by default .
  - To get help use command: `find -regextype xxx`
- **-type** **[d,f,l,b,c]** ... true if the type of the file is d, f, l, b, c.
  - **d** ... directory.
  - **f** ... regular file.
  - **l** ... symbolic link.
  - **b** ... block (buffered) special file.
  - **c** ... character (unbuffered) special file.
- **-inum** *n* ... true if the file has inode number *n*.

# Examples: find – tests

- Print all items from your home directory (recursively).

```
find ~ 2>/dev/null
```

- Print all items from your home directory (recursively) with suffix ".txt".

```
find ~ -name "*.txt" 2>/dev/null  
find ~ -regextype posix-extended -regex "^.*\.txt$" 2>/dev/null
```

- Print only regular files from your home directory (recursively) with suffix ".txt".

```
find ~ -type f -name "*.txt" 2>/dev/null  
find ~ -type f -regex "^.*\.txt$" 2>/dev/null
```

- Print all names assigned to i-node representing /usr/bin/yppasswd.

```
NUM="$(ls -i /usr/bin/yppasswd | cut -d' ' -f1)"  
find /usr/bin -inum "$NUM"
```

# find – tests

- Numeric arguments can be specified as
  - $+n$  ... for greater than  $n$ ,
  - $-n$  ... for less than  $n$ ,
  - $n$  ... for exactly  $n$ .
- `-size [+] $n$ [ $cwbkMG$ ]` ... true if the file has size of  $n$  units.
  - `b` ... for 512-bytes blocks.
  - `c` ... for bytes.
  - `w` ... for two-byte words.
  - `k` ... for Kilobytes (units of 1024 bytes).
  - `M` ... for Megabytes (units of 1048576 bytes).
  - `G` ... for Gigabytes (units of 1073741824 bytes).
- `-mtime [+] $n$`  ... true if the file's data was modified  $n * 24$  hours ago.
- `-atime [+] $n$`  ... true if the file's data was accessed  $n * 24$  hours ago.
- `-ctime [+] $n$`  ... true if the file's attributes were modified  $n * 24$  hours ago.
- `-newer file` ... true if file was modified more recently than *file*.

# Examples: find – tests

- Print only nonempty regular files from your home directory (recursively) with suffix ".txt".

```
find ~ -type f -name "*.txt" -size +0 2>/dev/null
```

- Print only regular files from your home directory (recursively) that were modified during last 23 days.

```
find ~ -type f -mtime -24 2>/dev/null
```

```
stat -c "%y %n" $(find ~ -type f -mtime -24 2>/dev/null) | \
    sort -n                                # verification
```

- Print only regular files from your home directory (recursively) that were modified from 8:00 4.12.2018.

```
touch -t "201812040800" /tmp/stamp 2>/dev/null
find ~ -newer /tmp/stamp
stat -c "%y %n" $(find ~ -newer /tmp/stamp 2>/dev/null)
rm /tmp/stamp
```

```
stat -c "%y %n" $(find ~ -newermt "20181204 8:00" 2>/dev/null)
```

# find – tests

- `-user name` ... true if file is owned by user *name*  
(numeric group ID allowed).
- `-group name` ... true if file belongs to group *name*  
(numeric group ID allowed).
- `-perm mode` ... true if file's permission bits are exactly *mode*  
(octal or symbolic).
- `-perm -mode` ... true if all of the permission bits *mode* are set for  
the file.
- `-perm /mode` ... true if any of the permission bits *mode* are set for  
the file.
- `-readable` ... matches files which are readable.
- `-executable` ... matches files which are executable and directories  
which are searchable.
- `-writable` ... matches files which are writable.



# Examples: find – tests

- Print all items from your home directory (recursively) that have exactly the permissions "rwxr-xr--".

```
find ~ -perm 754 -ls  
find ~ -perm u=rwx,g=rx,o=r -ls
```

- Print all files from directory /usr/bin (recursively) that have the permission set-user-ID.

```
find /usr/bin -type f -perm -4000 -ls  
find /usr/bin -type f -perm -u=s -ls
```

- Print all files from directory /usr/bin (recursively) that have the permission set-group-ID.

```
find /usr/bin -type f -perm -2000 -ls  
find /usr/bin -type f -perm -g=s -ls
```

- Print all files from directory /usr/bin (recursively) that have the permissions set-user-ID or set-group-ID.

```
find /usr/bin -type f -perm /6000 -ls  
find /usr/bin -type f -perm /u=s,g=s -ls
```

# find – actions

- `-print` ... always true, prints the full file name on the standard output, followed by a newline.
- `-printf format` ... always true, prints attributes of file according *format*
- `-ls` ... always true, prints current pathname together with statistics (in `ls -lsld` format).
- `-exec cmd {} \;` ... executes command *cmd*; true if 0 status is returned.
- `-exec cmd {} +` ... use only if the last argument is `{}`; the command line is built by appending each selected file name at the end and *cmd* is invoked fewer times.
- `-ok cmd \;` ... like `-exec`, but it requires confirmation.

# Examples: find – tests

- Print attributes of items from your home directory (recursively).

```
find ~ -ls 2>/dev/null
```

- Copy all regular files from the directory /etc (recursively) to the directory /tmp/Files-1.

```
mkdir /tmp/Files-1  
find ~ -type f -name "*.txt" -ok cp {} /tmp/Files-1 \;  
find /etc -type f -exec cp {} /tmp/Files-1 \; 2>/dev/null
```

- Add permission write for user to all file in /tmp/Files-1 and create copy /tmp/Files-2 of /tmp/Files-1.

```
chmod -R u+w /tmp/Files-1  
cp -r /tmp/Files-1 /tmp/Files-2
```

- Remove all files of size greater than 100 B from /tmp/Files-1 and from /tmp/Files-2 and measure the time of command execution by command time.

```
time find /tmp/Files-1 -type f -size +100c -exec rm {} \;  
time find /tmp/Files-2 -type f -size +100c -exec rm {} +
```

# find – operators

- `\( \)` ... grouped items within the expression.
- `\!` ... negated expression.
- `-a` ... join items within the expression by logical and (default).
- `-o` ... join items within the expression by logical or.

- **Examples**

- Find all regular files with permissions set-user-ID and -set-group-ID in /usr/bin.

```
find /usr/bin -type f -perm -4000 -perm -2000 -ls
```

- Find all regular files with permissions set-user-ID or -set-group-ID in /usr/bin.

```
find /usr/bin -type f \( -perm -4000 -o -perm -2000 \) -ls
```