

Chapter 15. File globbing

The shell is also responsible for **file globbing** (or dynamic filename generation). This chapter will explain **file globbing**.

15.1. * asterisk

The asterisk ***** is interpreted by the shell as a sign to generate filenames, matching the asterisk to any combination of characters (even none). When no path is given, the shell will use filenames in the current directory. See the man page of **glob(7)** for more information. (This is part of LPI topic 1.103.3.)

```
[paul@RHELv4u3 gen]$ ls
file1 file2 file3 File4 File55 FileA fileab Fileab FileAB fileabc
[paul@RHELv4u3 gen]$ ls File*
File4 File55 FileA Fileab FileAB
[paul@RHELv4u3 gen]$ ls file*
file1 file2 file3 fileab fileabc
[paul@RHELv4u3 gen]$ ls *ile55
File55
[paul@RHELv4u3 gen]$ ls F*ile55
File55
[paul@RHELv4u3 gen]$ ls F*55
File55
```

15.2. ? question mark

Similar to the asterisk, the question mark **?** is interpreted by the shell as a sign to generate filenames, matching the question mark with exactly one character.

```
[paul@RHELv4u3 gen]$ ls
file1 file2 file3 File4 File55 FileA fileab Fileab FileAB fileabc
[paul@RHELv4u3 gen]$ ls File?
File4 FileA
[paul@RHELv4u3 gen]$ ls Fil?4
File4
[paul@RHELv4u3 gen]$ ls Fil??
File4 FileA
[paul@RHELv4u3 gen]$ ls File??
File55 Fileab FileAB
```

15.3. [] square brackets

The square bracket `[]` is interpreted by the shell as a sign to generate filenames, matching any of the characters between `[` and the first subsequent `]`. The order in this list between the brackets is not important. Each pair of brackets is replaced by exactly one character.

```
[paul@RHELv4u3 gen]$ ls
file1 file2 file3 File4 File55 FileA fileab Fileab FileAB fileabc
[paul@RHELv4u3 gen]$ ls File[5A]
FileA
[paul@RHELv4u3 gen]$ ls File[A5]
FileA
[paul@RHELv4u3 gen]$ ls File[A5][5b]
File55
[paul@RHELv4u3 gen]$ ls File[a5][5b]
File55 Fileab
[paul@RHELv4u3 gen]$ ls File[a5][5b][abcdefghijklm]
ls: File[a5][5b][abcdefghijklm]: No such file or directory
[paul@RHELv4u3 gen]$ ls file[a5][5b][abcdefghijklm]
fileabc
```

You can also exclude characters from a list between square brackets with the exclamation mark `!`. And you are allowed to make combinations of these **wild cards**.

```
[paul@RHELv4u3 gen]$ ls
file1 file2 file3 File4 File55 FileA fileab Fileab FileAB fileabc
[paul@RHELv4u3 gen]$ ls file[a5][!Z]
fileab
[paul@RHELv4u3 gen]$ ls file[!5]*
file1 file2 file3 fileab fileabc
[paul@RHELv4u3 gen]$ ls file[!5]?
fileab
[paul@RHELv4u3 gen]$
```

15.4. a-z and 0-9 ranges

The bash shell will also understand ranges of characters between brackets.

```
[paul@RHELv4u3 gen]$ ls
file1  file3  File55  fileab  FileAB  fileabc
file2  File4  FileA   Fileab  fileab2
[paul@RHELv4u3 gen]$ ls file[a-z]*
fileab  fileab2  fileabc
[paul@RHELv4u3 gen]$ ls file[0-9]
file1  file2  file3
[paul@RHELv4u3 gen]$ ls file[a-z][a-z][0-9]*
fileab2
```

15.5. \$LANG and square brackets

But, don't forget the influence of the **LANG** variable. Some languages include lower case letters in an upper case range (and vice versa).

```
paul@RHELv4u4:~/test$ ls [A-Z]ile?
file1  file2  file3  File4
paul@RHELv4u4:~/test$ ls [a-z]ile?
file1  file2  file3  File4
paul@RHELv4u4:~/test$ echo $LANG
en_US.UTF-8
paul@RHELv4u4:~/test$ LANG=C
paul@RHELv4u4:~/test$ echo $LANG
C
paul@RHELv4u4:~/test$ ls [a-z]ile?
file1  file2  file3
paul@RHELv4u4:~/test$ ls [A-Z]ile?
File4
```

If **\$LC_ALL** is set, then this will also need to be reset to prevent file globbing.

15.6. preventing file globbing

The screenshot below should be no surprise. The **echo *** will echo a ***** when in an empty directory. And it will echo the names of all files when the directory is not empty.

```
paul@ubul010:~$ mkdir test42
paul@ubul010:~$ cd test42
paul@ubul010:~/test42$ echo *
*
paul@ubul010:~/test42$ touch file42 file33
paul@ubul010:~/test42$ echo *
file33 file42
```

Globbing can be prevented using quotes or by escaping the special characters, as shown in this screenshot.

```
paul@ubul010:~/test42$ echo *  
file33 file42  
paul@ubul010:~/test42$ echo \  
*  
paul@ubul010:~/test42$ echo '*'  
*  
paul@ubul010:~/test42$ echo "*"
.
```