# Unix file permissions tips

## How do you change file/directory permissions

Permissions for files and directories can be changed using the **chmod** command. This allows the individual r/w/x permissions to be set/changed for the owner, group and other users. For the purpose of simplicity name below refers to both file and directory names.

Chmod has the following syntax:

```
chmod <permissions> <name>
```

chmod has two ways to represent the permissions for owner, group and other, Numeric and alphabetic.

### Numeric permissions

This method uses a 3 digit octal number (e.g. 644, 755), with each digit representing the permission to set for owner (1st digit), group (2nd) and other (3rd). Each of the permissions is assigned a number and the value of the digit is the addition of the numeric value for the permission.

The permissions have the following values:

r = 4, w = 2, x = 1, - = 0

To work out the number to use to set the correct permissions see the following table:

--- = 0 (no permissions)
--x = 1 x(1) only
-w- = 2 w(2) only
-wx = 3 w(2) + x(1)
r-- = 4 r(4) only
r-x = 5 r(4) + x(1)
rw- = 6 r(4) + w(2)
rwx = 7 r(4) + w(2) + x(1)

Here are some examples to explain how the numbers are used:

**chmod 100** - This sets owner --x (1), group --- (0), other --- (0) = --x------

**chmod 660** - This sets owner rw- (6), group rw- (6), other --- (0) = rw-rw----

**chmod 700** - This sets owner rwx (7), group --- (0), other --- (0) = rwx------

**chmod 754** - This sets owner rwx (7), group r-x (5), other r-- (4) = rwx-r-xr-

**Alphabetic Permissions**

It is also possible to use letters instead of numbers to change/set the permissions as follows:

```
chmod [u g o a][+ - =][r w x X] file
```

- u = user (owner), g = group, o = other, a = all (short for ugo). Any combination of ugo can be specified (u, ug, ugo, uo, go etc). If none are specified the default a is assumed.

- + = add permission, - = remove permission, = only specified permission will be set (replacing existing permissions). Only one can be used.

- r = read, w = write, x = execute/access, X = execute only if file is a directory or already has execute permission for some user. Any combination can be used.

**chmod u+x file** - this adds (+) x permission for the owner only (rw-rw-rw- becomes rwxrw-rw-).

**chmod u-w file** - this removes (-) w permission for the owner only (rwxr-xrwx becomes r-xr-xrwx).

**chmod g+w file** - this adds w permission to group (rwx---rwx becomes rwx-w-rwx).

**chmod o=x file** - this sets x permission only to other (rwxrwxrw- becomes rwxrwx--x).

**chmod a+rx file** - this sets r and x permission for owner, group and other (--------- becomes r-xr-xr-x).

You can also set different permissions for owner, group and other in the same command:

**chmod u+x,g-x,o+r file** - this adds x permission to owner, removes x permission for group and adds r permission to other (rw-rwx--- becomes rwxrw-r--).

An additional option to the chmod command is -R. This causes the chmod command to also recursively search all directories within the current directory and changes the permission of any matching files. This is useful when you have a lot of files in multiple sub-directories that you want the same permissions to be set without having to perform the chmod command in each one separately.

**chmod -R o+x *.php** - this will perform a recursive search of all files in all directories starting from the current directory changing any file that matches *.php to add owner execute permission. (e.g. rw-rw-rw- becomes rwxrw-rw-)

**chmod -R o+X *** - this will perform the same recursive search as above but all files and directories will have owner execute permission set.