1. The category of permission you want to change. There are three: the owner's permission (which **chmod** calls "user," abbreviated `u`), the group's permission (`g`), or others' permission (`o`). To change more than one category, string the letters together, such as `go` for "group and others," or simply use `a` to mean "all" (same as `ugo`).
2. Whether you want to add (`+`) the permission, delete (`-`) it, or specify it exactly (`=`).
3. What permissions you want to affect: read (`r`), write (`w`), or execute (`x`). To change more than one permission, string the letters together—for example, `rw` for "read and write."

Some examples should make this clearer! In the following command lines, you can replace *dirname* or *filename* with the pathname (absolute or relative) of the directory or file. An easy way to change permissions on the working directory is by using its relative pathname, . (dot), as in "**chmod a-w** .". You can combine two permission changes in the same **chmod** command by separating them with a comma (`,`), as shown in the final example.

- To protect a file from accidental editing, delete everyone's write permission with the command "**chmod a-w** *filename*". On the other hand, if you own an unwritable file that you want to edit, but you don't want to change other peoples' write permissions, you can add "user" (owner) write permission with "**chmod u+w** *filename*".
- To keep yourself from accidentally removing files (or adding or renaming files) in an important directory of yours, delete your own write permission with the command "**chmod u-w** *dirname*". If other users have that permission, too, you could delete everyone's write permission with "**chmod a-w** *dirname*".
- If you want you and your group to be able to read and write all the files in your working directory—but those files have various permissions now, so adding and deleting the permissions individually would be a pain—this is a good place to use the `=` operator to set the exact permissions you want. Use the filename wildcard `*`, which means "everything in this directory" (explained in the section "File and Directory Wildcards" of Chapter 4) and type: "**chmod ug=rw** *".

  If your working directory had any subdirectories, though, that command would be wrong because it takes away execute permission from the subdirectories, so the subdirectories couldn't be accessed