Unzip Multiple Files from Linux Command Line

chrisjean.com/unzip-multiple-files-from-linux-command-line/

Here's a quick tip that will help you work with multiple zip files on the command line.

If you are in a folder and have three zip files in it (`a.zip`, `b.zip`, `c.zip`) that you want to unzip, "no problem," you think. "I can take care of that with one command." And you quickly run the following:

```
[gaarai@tenshi ~]$ unzip *.zip
```

However, rather than having the unzip program nicely unzip each file one after another, you receive the following:

```
[gaarai@tenshi ~]$ unzip *.zip
Archive:  a.zip
caution: filename not matched:
b.zip
caution: filename not matched:
c.zip
[gaarai@tenshi ~]$
```

I'm sure that this is not what you were expecting. I know I certainly wasn't expecting this when I first tried it. However, this problem can help us understand more of how the command line works.

The Problem

Whenever you use a wildcard (`*`), the shell itself will expand that and pass the results to the program rather than the program handling the expansion itself. That means that our previous command was actually expanded to the following before being executed:

```
[gaarai@tenshi ~]$ unzip a.zip b.zip
c.zip
```

Again, this may not look odd since other programs (such as `mkdir`, `chmod`, etc) can take one or more arguments and repeat the process for each. However, `unzip` is different and actually has use for the additional arguments. If you specify additional arguments after the zip file name, `unzip` will try to only extract those specific files from the archive rather than all the files.

The previous command told `unzip` that it should extract `b.zip` and `c.zip` from inside the `a.zip` archive. Since those files don't exist inside `a.zip`, we get the unexpected error output seen earlier.

You might think that you will have to manually unzip each archive one at a time, but you'd be wrong.

The Solution

Just because the shell expands out wildcard characters automatically doesn't mean that programs can't as well. The simple solution to this problem is to quote the argument to prevent the shell from interpreting it:

```
[gaarai@tenshi ~]$ unzip '*.zip'
Archive:  c.zip
   creating: c/
  inflating: c/test.txt

Archive:  a.zip
   creating: a/
  inflating: a/test.txt

Archive:  b.zip
   creating: b/
  inflating: b/test.txt

3 archives were successfully
processed.
[gaarai@tenshi ~]$
```

That's all there is to it. I've seen solutions from coding loops in bash script to running find combined with `xargs`, but none of that is necessary. Simply let the `unzip` command itself take care of things for you.