

## which (locate a command)

This command shows you where in your executable path a program resides. For example:

```
$ which touch
/usr/bin/touch
```

Don't type in the dollar sign; I used it to indicate the command prompt. The output shows that `/usr/bin/touch` is executed whenever the command `→touch` is invoked. Now try `which` with the option `-a` ("all"). This will show all places in the path where an executable file is found:

```
$ which -a touch
/usr/bin/touch
/bin/touch
```

You can see that my system also has a file called `/bin/touch`, but `/usr/bin/touch` has priority over it, that's why it's listed first. In this case, there is no real difference between the two "competing" commands because `/usr/bin/touch` is a symbolic link to `/bin/touch` (see `→ln`). But it may happen that you're not sure which program is executed when you call a particular command, and that's when `which` comes in handy.

It's important to note that `which` can only identify executable files. Shell builtins such as `→cd` or `→alias` do not have their own executable file, so `which` will not give you any information about them. Another common stumbling block for newcomers is that root has its own executable path. It follows that you won't get any output when you call `which shutdown`, for example, because `→shutdown` is in root's path. You have to run `sudo which shutdown` in this case, which will output the following:

```
/sbin/shutdown
```

The directory `/sbin` is not part of a normal user's executable path because it contains programs for administrative work.