Unix Shell Family Essential Metacharacters

Wild Card (globbing) metacharacters try to match files

?	any single character
*****	zero or more occurrances of any character
[*set−of−characters*]	any single character in the *set−of−characters*
[!*set−of−characters*]	any single character **not** in the *set−of−characters*

Quoting and Grouping

any−character	*any−character* as a literal (not meta)
'*any−characters***'**	*any−characters* as literal text
"*most−characters*"	*most−characters* as literal text, but:
	"*Double quotes*" don't suppress $*Variable*, `Embedding` or $(Nesting)
	Quoted expressions will be parsed as one **word**, i.e. as one token

Command Embedding and Nesting

`***shell−command*`	replaced by output of embedded *sub−command (old syntax)*
$(*shell−command*)	replaced by output of nested *sub−command (new syntax)*

Variables

name=*word*	creates variable *name* with value *word (local to shell*)
export *name*=*word*	creates environment variable *name* with value *word (passed to new processes*)
$*name* or **${***name***}**	replaced by value of variable *name* or nothing
	Expand variables inside double quotes to avoid surprises!

Command I/O Redirection

< *file−name*	standard input comes from *file−name*
> *file−name*	standard output goes to *file−name*, overwriting it
>> *file−name*	standard output goes to *file−name*, appending to it

Sequencing Metacharacters

*command***&**	runs *command* in the *background* for parallel processing
$command_1$ **&&** $command_2$	*and then* if $command_1$ succeeds, also runs $command_2$
$command_1$ **||** $command_2$	*or else* if $command_1$ fails, runs alternative $command_2$
{ *command***;** ... **}**	grouped sequence of commands, use **;** or *newline*
! *command*	succeeds when *command* fails and vice versa