Bash Shell and scripting

CONCEPTS

Bash expands a line first (variables, globs, substitutions), then runs it. Quotes decide what gets expanded.

Read line split into words
↓ expand
Expansions braces, $var, $(cmd), globs
↓ redirect
Redirections stdin, stdout, stderr
↓ run
Command exit status in $?

Always quote "$var": unquoted values are split at spaces and globbed.

REDIRECTION & PIPES

cmd > *<file>*
Stdout to file, overwrite

cmd >> *<file>*
Stdout appended to file

cmd < *<file>*
Read stdin from file

cmd1 | cmd2
Pipe stdout into next command

cmd 2> *<file>*
Stderr to file

cmd > *<file>* 2>&1
Stdout and stderr to file

cmd > /dev/null 2>&1
Discard all output

QUOTING & EXPANSION

"$var"
Expand, keep as one word

'$var'
Literal text, no expansion

$(cmd)
Command substitution

***.txt**
Glob: any characters

{a,b,c}
Brace expansion list

$((a + b))
Arithmetic expansion

VARIABLES

name=*<value>*
Set a variable (no spaces)

export name=*<value>*
Pass to child processes

$1 $2 ... $@
Script arguments

$?
Exit status of last command

CONTROL FLOW

if cmd; then ...; fi
Branch on exit status

[[-f *<file>*]]
Test: regular file exists

[["$a" == "$b"]]
Test: strings equal

for x in a b c; do ...; done
Loop over words

for f in *.txt; do ...; done
Loop over files

cmd1 || cmd2
Run second only on failure

COMMAND LINE (READLINE)

Ctrl + R
Search history backwards

Ctrl + A / Ctrl + E
Start / end of line

Ctrl + C
Interrupt running command

Ctrl + L
Clear the screen

Tab
Complete command or path

JOBS & PROCESSES

cmd &
Run in the background

SAFE SCRIPTS

#!/usr/bin/env bash
Shebang line

set -euo pipefail
Stop on errors and unset vars