Compiled by Fernando Fenandez on 2/9/25

Z-Shell (zsh) MacOs Cheat Sheet

Configuration Files

~/.zshrc - Main user configuration file
~/.zshenv - Environment variables
~/.zprofile - Login shell configuration
~/.zlogin - Executed after .zprofile
~/.zlogout - Executed when logging out

Process Control

Ctrl Z – suspend running process and return to terminal
jobs – list all pending processes
fg %1 – resume process
bg %1 – resume process in background and stay on prompt
command & - starts command in background and stay on the prompt
command1 | command2 – output from command1 to command2
command > file – overwrite file with output
command >> file – append file with output
ps -ax – list running processes
ps aux – list running processes with more details (CPU/mem)
top – list processes in real-time
kill pid - kill process by id
killall name - kill process by name

Command History

history - command history
!! - repeat last command
!n - execute nth command from history
!-n - execute previous nth command
!command - repeat last command starting with command
^old^new - replace old with new in last command and execute
!!:s/old/new - replace old with new in last command
!:p - print command without executing it

Navigation and Editing

Ctrl A - move to beginning of line
Ctrl E - move to end of line
Option ⬅️- move back 1 word
Option ➡️- move forward 1 word
Ctrl K - cut text from cursor to end of line
Ctrl W - cut word before cursor
Ctrl R - search through command history
Command K - clear screen
Command D - split terminal
Shift Command D - close split
Command T - new terminal tab

Scripting - Variables

$var - returns value
${!var_name} - returns value of variable whose name is stored in another variable

${#str} - length of str
${str:1:3} - substring from pos 2 to 4 (zero indexed)

((var += 5)) - adds to var
var=$(expr 15 - 2) - calculation

typeset -i var - declare integer
typeset -A map - declare map
typeset -a arr - declare array
typeset -F var - declare float

${var:-default} - return var or default value if unset

${val1/val2/val3} - replaces val2 with val3
${val1//val2/val3} - replaces all val2 with val3

Scripting - IF

```
if CONDITION1; then
  …
elif CONDITION2; then
  …
else
  …
fi
```

Scripting - Variables (arrays)

arr=(element1 element2 element3)
${arr[2]} - 2nd element
${arr[-1]} - last element
${arr[@]} - all elements
${#arr[@]} - element count
${arr:offset:length} - slice array
${(O|o)arr} - sort array (O=desc)
${(u)arr} - deduplicate array
${(j:,:)arr} - join array with comma

arr+=(element) - append element
unset arr[2] - removes 2nd elem
arr=([key1]=value1 [key2]=value2) - map

arr=($(command)) - stores output of command into array

Scripting - Conditions

Condition can be:
  a command (exit status 0 = true)
  [expression] (file/string/number)
  test expression (same as above)
  ((expression)) (numeric)

if [["$name" == a*]]; then
# name starts with a

if ((x > y)); then
if [["$str1" == "abc"]]; then

if [-f "test.txt"]; then
# file exists

if [-d "abc"]; then
# abc is a directory

-r -w -x
(readable/writable/executable)

if[-s "test.txt"]; then
# file size > 0

-n "$variable"
# variable is set

-z "$variable"
# variable is empty

Scripting - LOOPS

```
for i in {1..5}; do
  …
done
for i in {10..1..-1} do… - reverse
loop with step

for arg in "$@"; do … - loop over
arguments

for var in "${array[@]}"; do …

for ctry in "${(@k)cities}"; do
   echo "$ctry: ${cities[$ctry]}"
done

for file in *.txt; do …
for file in **/*.txt; do … - recursive

case $num in
   1) echo "You chose one." ;;
   2) echo "You chose two." ;;
   *) echo "Invalid choice. ;;
esac

while (( i > 0 )); do
   …
done
```

Splitting Files

```
split -l [number_lines] -b [size]
input_file prefix_result_files

csplit -k -f prefix -n nbr input_file
/pattern/ {9999999}
# standard for MacOs

# after brew install coreutils
csplit input_file '/pattern/' '{*}'
--elide-empty-files
--suffix-format='suffix%03d'
# 0 = pad left with zeroes / 3 =
length / d = decimal
```

Scripting - Parameters

```
$n - returns nth parameters
$# - number of parameters
$@ - all positional parameters
$* - all positional parameters as a
single word
$$ - PID of the shell
$? - exit status of last command
$! - PID of last background job
$_ - last argument of previous
command
$IFS - Internal Field Separator
$PWD - current working directory
$OLDPWD - previous working
directory
$RANDOM - random number
between 0 and 32767
$SECONDS - seconds since shell
started
```

Finding and Processing Files

```
find path -type f -name "filemask"
-exec command {} \;

find path -type f -name "filemask" |
while read file; do
    echo Processing file $file
done

# find files by size
find path -size +50M
```

Modifying Files

```
sed -i '' 's/old_text/new_text/g'
filename

# deletes line N from file
sed -i '' 'Nd' filename

# add 10 to 2nd col (cols
separated by space/tab)
awk '{$2 += 10; print}' input.txt >
output.txt

# use comma as separator
awk 'BEGIN {FS = ","}; { $2 += 10;
print}' test.txt
awk -F',' '{$2 += 10; print}' input.txt

# only print if 3rd col > 50
awk '$3 > 50 { print $1, $3 }'
data.txt
```

Scripting - String Manipulation

```
${param%suffix} - remove suffix
${param#prefix} - remove prefix
${param/patt/str} - replace first
pattern match with string
${param//patt/str} - replace first
pattern match with string

${var:pos:len} - substring from
position with length
${var:(-pos):len} - substring from
the right
${var::-1} - exclude last character
${var:(-2)} - get last 2 characters

${#var} - length of string

${var:l} - lowercase string
${var:u} - uppercase string
```

Searching Files

```
# search all files recursively
grep -r "pattern" path

# case insensitive + numbered
matches
grep -i -n "pattern" path

grep -E "regex" path
```

Useful Snippets

```
# Flush DNS Cache
sudo dscacheutil -flushcache;
sudo killall -HUP mDNSResponder

# Reset Dock
killall Dock

# Restart
sudo shutdown -r now
# Shutdown
sudo shutdown -h now

# Clipboard
cat file | pbcopy
pbpaste | target
```