REFERENCIA RÁPIDA DE BASH

Comandos, scripting, pipes, redirección, control de trabajos

Fundamentos

echo y Navegación

```
echo "Hello, World!"     # imprimir texto
pwd                      # mostrar directorio actual
cd /path/to/dir          # cambiar directorio
cd ..                    # subir un nivel
cd ~                     # ir al directorio home
cd -                     # ir al directorio anterior
```

Listar y Crear

```
ls               # listar archivos
ls -la           # formato largo, mostrar ocultos
ls -lh           # tamaños legibles
mkdir mydir      # crear directorio
mkdir -p a/b/c   # crear directorios anidados
```

Copiar, Mover y Eliminar

```
cp file.txt copy.txt      # copiar archivo
cp -r dir/ backup/        # copiar directorio recursivamente
mv old.txt new.txt        # renombrar / mover
rm file.txt               # eliminar archivo
rm -r dir/                # eliminar directorio recursivamente
rm -rf dir/               # eliminar forzado (sin confirmación)
```

Variables y Expansión

Variables

```
name="Alice"              # asignar (¡sin espacios!)
echo "$name"              # expansión de variable
echo "${name}_file"       # llaves para claridad
readonly PI=3.14          # constante
unset name                # eliminar variable
```

Variables Especiales

`$0`	Nombre del script
`$1 $2 ...`	Argumentos posicionales
`$#`	Número de argumentos
`$@`	Todos los argumentos (palabras separadas)
`$*`	Todos los argumentos (cadena única)
`$?`	Estado de salida del último comando
`$$`	ID del proceso actual
`$!`	PID del último proceso en segundo plano

Sustitución de Comandos y Aritmética

```
files=$(ls)               # capturar salida
today=$(date +%Y-%m-%d)   # sustitución de comando
count=$((5 + 3))          # aritmética: 8
echo $((10 / 3))          # división entera: 3
echo $((10 % 3))          # módulo: 1
```

Operaciones con Cadenas

`${#str}`	Longitud de cadena
`${str:0:5}`	Subcadena (desplazamiento:longitud)
`${str/old/new}`	Reemplazar primera coincidencia
`${str//old/new}`	Reemplazar todas las coincidencias
`${str^^}`	Mayúsculas
`${str,,}`	Minúsculas

Condicionales

if / elif / else

```
if [[ "$name" == "Alice" ]]; then
    echo "Hi Alice"
elif [[ "$name" == "Bob" ]]; then
    echo "Hi Bob"
else
    echo "Who are you?"
fi
```

Operadores de Prueba

`-eq  -ne`	Igual / distinto (enteros)
`-lt  -gt`	Menor / mayor que (enteros)
`-le  -ge`	Menor/mayor o igual (enteros)
`==  !=`	Igual / distinto (cadenas)
`-z "$str"`	Cadena vacía
`-n "$str"`	Cadena no vacía
`-f file`	Archivo existe y es regular
`-d dir`	Directorio existe
`-e path`	Ruta existe (cualquier tipo)
`-r  -w  -x`	Lectura / escritura / ejecución
`&&  ||`	AND / OR lógico

Bucles

Bucle for

```
for fruit in apple banana cherry; do
    echo "$fruit"
done

for f in *.txt; do
    echo "File: $f"
done
```

Bucle for Estilo C

```
for ((i=0; i<5; i++)); do
    echo "$i"
done
```

Bucle while

```
count=0
while [[ $count -lt 5 ]]; do
    echo "$count"
    ((count++))
done
```

Control de Bucles

`break`	Salir del bucle
`continue`	Saltar a la siguiente iteración

Funciones

Definir y Llamar

```
greet() {
    echo "Hello, $1!"   # $1 = primer arg
    return 0            # estado de salida
}
greet "Alice"           # Hello, Alice!
```

Variables Locales y Valores de Retorno

```
add() {
    local sum=$(($1 + $2))
    echo "$sum"         # "retornar" vía stdout
}
result=$(add 3 5)       # capturar: 8
```

Pipes y Redirección

Pipes

```
ls -l | grep ".txt"     # pipe de salida
cat log | sort | uniq   # encadenar comandos
cmd1 | tee out.txt      # pipe + guardar en archivo
```

Redirección

`cmd > file`	Redirigir stdout (sobrescribir)
`cmd >> file`	Redirigir stdout (añadir)
`cmd < file`	Redirigir stdin desde archivo
`cmd 2> file`	Redirigir stderr
`cmd 2>&1`	Redirigir stderr a stdout
`cmd &> file`	Redirigir stdout + stderr
`cmd << EOF`	Here document (entrada en línea)
`/dev/null`	Descartar salida: `cmd > /dev/null`

Operaciones con Archivos

Ver Archivos

```
cat file.txt              # mostrar archivo completo
head -n 10 file.txt       # primeras 10 líneas
tail -n 10 file.txt       # últimas 10 líneas
tail -f log.txt           # seguir (actualizaciones en vivo)
less file.txt             # visor paginado
```

Contar y Buscar

```
wc -l file.txt            # contar líneas
wc -w file.txt            # contar palabras
wc -c file.txt            # contar bytes
find . -name "*.txt"      # buscar por nombre
find . -type d            # buscar directorios
find . -mtime -7          # modificados en últimos 7 días
```

Otros Comandos de Archivos

`touch file`	Crear archivo / actualizar marca de tiempo
`stat file`	Metadatos del archivo (tamaño, fechas)
`file img.png`	Detectar tipo de archivo
`diff a.txt b.txt`	Comparar dos archivos
`sort file.txt`	Ordenar líneas
`uniq`	Eliminar duplicados adyacentes
`cut -d: -f1`	Extraer campos por delimitador
`tr 'a-z' 'A-Z'`	Traducir / reemplazar caracteres

Procesamiento de Texto

grep

```
grep "error" log.txt      # buscar patrón
grep -i "error" log.txt   # sin distinción mayúsculas
grep -r "TODO" src/       # búsqueda recursiva
grep -n "func" file.go    # mostrar números de línea
grep -c "error" log.txt   # contar coincidencias
grep -v "debug" log.txt   # invertir coincidencia
```

sed

```
sed 's/old/new/' file     # reemplazar primera por línea
sed 's/old/new/g' file    # reemplazar todas
sed -i 's/old/new/g' file # editar en sitio
sed -n '5,10p' file       # imprimir líneas 5-10
sed '/pattern/d' file     # eliminar líneas coincidentes
```

awk

```
awk '{print $1}' file     # imprimir primer campo
awk -F: '{print $1}' file # delimitador personalizado
awk '$3 > 100' file       # filtrar por valor de campo
awk '{sum+=$1} END{print sum}' file  # sumar columna
```

Permisos

chmod

```
chmod 755 script.sh    # rwxr-xr-x
chmod +x script.sh     # agregar ejecución
chmod -w file.txt      # quitar escritura
chmod u+x,g-w file     # usuario +ejec, grupo -escritura
```

Referencia de Permisos

`r (4)`	Lectura
`w (2)`	Escritura
`x (1)`	Ejecución
`u / g / o`	Usuario / Grupo / Otros
`755`	Propietario: rwx, Grupo/Otros: r-x
`644`	Propietario: rw-, Grupo/Otros: r--

Propietario

```
chown user file.txt         # cambiar propietario
chown user:group file.txt   # cambiar propietario + grupo
chown -R user:group dir/    # recursivo
```