

Resumen Bash

Orden	Descripción
<code>man, info, help</code>	
<code>ls directorio</code>	lista los contenidos de un directorio
<code>-l</code>	Muestra los contenidos en formato largo
<code>cd directorio</code>	Cambia de directorio de trabajo: <code>.</code> directorio actual, <code>..</code> directorio padre, <code>~</code> home
<code>pwd</code>	Camino absoluto del directorio actual
<code>mkdir directorio</code>	Crea un directorio a partir del nombre dado como argumento
<code>rmdir directorio</code>	Borra el directorio (si está vacío)
<code>cat archivo(s)</code>	Muestra el contenido de un archivo o varios, concatena archivos, copia un archivo, crea un archivo de texto o muestra los caracteres invisibles de control
<code>cp archivo1 archivo2</code>	Copia el archivo1 en archivo2. Si archivo2 no existe, se crea
<code>mv fuente destino</code>	Renombra archivos (el archivo fuente puede ser archivo o directorio, al igual que el destino) y puede mover de lugar un archivo u otro
<code>file archivo(s)</code>	Muestra el tipo de archivo dado como argumento
<code>more archivo(s)</code> <code>page archivo(s)</code> <code>pg archivo(s)</code>	Visualiza el archivo
<code>rm directorio_archivos</code>	Borra archivos y directorios con contenido
<code>touch archivo(s)</code>	Si existen los archivos dados como argumentos se modifican su fecha y hora. Si no, se crean con la fecha actual del sistema
<code>clear</code>	Borra el contenido del terminal actual
<code>tail archivo(s)</code>	Muestra la parte final del contenido de un archivo dado como argumento. Por defecto muestra 10 líneas
<code>head archivo(s)</code>	Muestra la parte inicial del contenido de un archivo dado como argumento. Por defecto muestra 10 líneas
<code>sort archivo(s)</code>	Ordena, según un criterio elegido, el contenido de los archivos dados como argumentos
<code>chmod</code>	Modificación de permisos: [usuario] [+/-] [permiso] <ul style="list-style-type: none"> Grupo de usuarios: <ul style="list-style-type: none"> o u: propietario o g: grupo o o: resto de usuarios o a: todos los grupos de usuarios Tipo de permiso: <ul style="list-style-type: none"> o r: lectura o w: escritura o x: ejecución

Metacaracteres de archivo

Metacarácter / Descripción

?	Cualquier carácter simple en la posición indicada	*	Cualquier secuencia de cero o más caracteres
[]	Designan un carácter o rango de caracteres que representan un carácter simple a través de guiones	{ }	

Metacaracteres de redirección

Metacarácter	Descripción
< nombre	Redirecciona la entrada de una orden para que la obtenga de un archivo <i>nombre</i>
> nombre	Redirecciona la salida de una orden para que la escriba en el archivo <i>nombre</i> . Si dicho archivo ya existe, lo sobrescribe
&> nombre	La salida estándar se combina con la salida de error estándar y ambas se escriben en el archivo <i>nombre</i>
>> nombre	Funciona igual que ">" pero añade la salida estándar al final del contenido de <i>nombre</i>
&>> nombre	Funciona igual que "&>" pero añade las dos salidas combinadas al final de <i>nombre</i>
2> nombre	Redirige la salida de error estándar a un archivo (sólo funciona a shells de "bash")
	Crea un cauce entre dos órdenes . La salida de una de ellas se usa como entrada de la otra
&	Crea un cauce entre dos órdenes utilizando las dos salidas (estándar y error) de una de ellas como entrada de la otra

Metacaracteres sintácticos

Metacarácter	Descripción
;	Separador entre órdenes que se ejecutan secuencialmente
()	Se usan para aislar órdenes separadas por ";" o por " ". Las órdenes dentro de los paréntesis son tratadas como una única orden
&&	Separador entre órdenes, el que sigue a "&&" se ejecuta sólo si la orden precedente ha tenido éxito (no ha habido errores)
	Separador entre órdenes, el que sigue a " " se ejecuta sólo si la orden precedente falla (ha habido errores)

Variables

Tipos de variables

- **Variables de entorno o globales:** son comunes a todos los shells. Para visualizarlas, puede usarse `env` o `printenv`.
- **Variables locales:** sólo son visibles desde el Shell desde donde se definen. Para visualizarlas, puede usar la orden `set`.

Ayuda: `help variables`

Contenido de las variables

- **Cadenas:** su valor es una secuencia de caracteres.
Ej.: NOMBRE=Pacocurro
- **Números:** se podrán usar en operaciones aritméticas.
Ej.: VAR=2
- **Constantes:** su valor no puede ser alterado.
Ej.:
- **Vectores o arrays:** conjunto de elementos a los que se puede acceder mediante un índice.
Ej.: colores=(rojo azul verde)
Si queremos leer el primer valor:
echo \${colores[0]}

Borrar una variable: unset

Variables especiales

Metacarácter	Descripción
\$BASH	Ruta de acceso completa usada para ejecutar la instancia actual de bash
\$HOME	Almacena el directorio raíz del usuario
\$PATH	Guarda el camino de búsqueda de las órdenes, formado por una lista de todos los directorios en los que queremos buscar una orden
\$?	Código de retorno de la última orden ejecutada, bien sea una instrucción o un guion

Variables con atributos

Si queremos usar una variable con ciertos atributos, utilizaremos declare (use help declare). Podemos declarar que es numérica con -i y ver los atributos con -p:

```
$ declare -i IVA=18
$ declare -p IVA
declare -i IVA="18"
```

Exportar variables

Para emplear variables fuera de un mismo shell, usaremos export.

Comillas y sustitución de órdenes

En el shell bash podemos hacer uso de la *sustitución de órdenes*, que permite la ejecución de una orden, con o sin argumentos, de forma que su salida se trata como si fuese el valor de una variable. Se puede hacer de las siguientes formas:

`$(orden argumentos)` ``orden argumentos``

Es importante hacer distinción entre las comillas simples y dobles, que varían la protección de los caracteres que se encuentran en su interior:

- **Comillas dobles:**
- **Comillas simples:**

Asignación de resultados de órdenes a variables

```
VAR=`orden`
```

Alias

Se crean con `alias` y se eliminan con `unalias`.

```
alias dir='ls -l'
```

La orden empotrada `printf`

Es del tipo:

```
printf formato argumentos
```

Donde *formato* describe cómo se imprimirán los elementos de *argumentos*, mediante códigos de escape y especificaciones de formato.

Códigos de escape

Secuencia de escape	Acción
<code>\b</code>	Espacio atrás
<code>\n</code>	Nueva línea
<code>\t</code>	Tabulador
<code>\'</code>	Carácter comilla simple
<code>\\</code>	Barra invertida
<code>\0n</code>	n es el número en octal que representa un carácter ASCII de 8 bits

Especificaciones de formato

Código de formato	Representa
<code>%d</code>	Número con signo
<code>%f</code>	Número en coma flotante sin notación exponencial <i>En este caso, la parte entera se interpreta como la anchura de la columna y la decimal como el número mínimo de dígitos</i>
<code>%q</code>	Entrecomilla una cadena
<code>%s</code>	Muestra una cadena sin entrecomillar
<code>%x</code>	Muestra un número en hexadecimal
<code>%o</code>	Muestra un número en octal

Órdenes de búsqueda: `find` y `grep`, `egrep`, `fgrep`

Orden `find`

```
find lista-de-directorios expresiones
```

Las *expresiones* describen el criterio de búsqueda. Use `man find` para encontrarlas.

Orden `grep`

```
grep opciones patrón archivos
```

Algunas de las opciones:

Opción	Descripción
-x	Localizan líneas que coincidan totalmente, desde el principio hasta el final, con el patrón especificado
-v	Selecciona todas las líneas que no contengan el patrón especificado
-c	Produce solamente un recuento de las líneas coincidentes
-i	Ignora las distinciones entre mayúsculas y minúsculas
-n	Añade el número de línea en el archivo frente a las coincidencias
-l	Selecciona sólo los nombres de aquellos archivos que coincidan con el patrón de búsqueda
-e	Especial para el uso de múltiples patrones e incluso si el patrón empieza por -
-q	"Silencia" las coincidencias
-w	Encuentra literalmente lo que se quiera buscar

Variantes de grep: egrep y fgrep

- fgrep acepta sólo una cadena simple de búsqueda en vez de una expresión regular.
- egrep permite un conjunto más complejo de expresiones regulares.

Algunas observaciones

- Listar directorios: `ls -d`
- Mostrar archivos poco a poco: `less`
- El `less` y el `cat` pueden usar expresiones de búsqueda para mostrar varios archivos.
- `&>` y `&>>` hacen lo mismo que `>` y `>>` a menos que haya error, en cuyo caso lo copia.
- **Comando cut:** `cut -d "[delimitador]" -f [sección] [archivo]`
 - o La sección pueden ser varias. Ej.: `-f 1,2`
- A grep se le puede pasar cualquier secuencia de caracteres de un comando. Un ejemplo de esto es: `cut -d ":" -f 1 /etc/passwd | grep mianfg; echo $?`
- **Equivalentes para conocer el estado de error de una orden:**
 - o `[orden]; echo $?`
 - o `[orden] && echo 1 || echo 0`
- **Orden if:**
 - o Si se quiere comparar con un valor numérico: se usa `=` en lugar de `==`
 - o Cuando se usa `(())` no puede usarse `-eq` `-ne` `-le` `-lt` `-ge` `-gt` `=`
 - o Para comparar cadenas de caracteres con espacios, si la variable es necesario poner `"[variable]":`
 - Da error: `if [$valor == "hola amigos"]`
 - No da error: `if ["$valor" == "hola amigos"]`
- El test puede usarse como comparativa
- **Búsqueda**
 - o `find` sirve para buscar a nivel de características generales de los archivos, como, por ejemplo, nombre o condiciones de acceso, pero sin entrar en su contenido; por el contrario, `grep` y `egrep` examinan la cadena que se le dé mediante la entrada estándar o el contenido de los archivos que se le pongan como argumento.