Metacaracteres

Material elaborado por:

L.I. Eduardo Iván Ortega Alarcón

eduardo.ortega@unam.mx

Departamento de Supercómputo, DGTIC, UNAM.

Metacaracteres

- Son caracteres reconocidos como especiales por el shell.
- También se les conoce como wildcards (por su nombre en inglés).
- El shell los interpreta de forma diferente, por lo que cada uno de ellos tiene un significado especial que es traducido antes de ejecutar el comando.

DGTIC

Separación de comandos

- Regularmente se utiliza la tecla <intro> para indicar que el comando ha terminado.
- El carácter punto y coma (;) se puede utilizar para indicar que un comando termina.
- Esto significa que se pueden escribir varios comandos en una misma línea para que se ejecuten uno tras otro.

DGTIC

Separación de comandos

```
[chaos@gnulinux ~]$ echo Hola
Hola
[chaos@gnulinux ~]$ echo Mundo
Mundo
[chaos@gnulinux ~]$ echo Hola; echo Mundo
Hola
Mundo
[chaos@gnulinux ~]$ _
```

DGTIC

Separación de palabras

- Una palabra es cualquier conjunto de caracteres.
- El caracter espacio es el que se utiliza para delimitar una palabra.
- El shell utiliza las palabras como argumentos para los comandos e instrucciones.
- Se ignoran espacios que están escritos de forma consecutiva.

DGTIC

Separación de palabras

```
[chaos@gnulinux ~]$ echo Hola Mundo
Hola Mundo
[chaos@gnulinux ~]$ echo Hola         Mundo
Hola Mundo
[chaos@gnulinux ~]$ _
```

DGTIC

Sustitución de caracteres

Metacaracter	Descripción
*	Representa cualquier cadena de cero o más caracteres.
?	Un carácter. Debe ser uno forzosamente, pero cualquiera.
[ ]	Funciona como el metacaracter ? Pero aquí se pueden delimitar los caracteres posibles, también pueden indicarse rangos de caracteres de acuerdo a la codificación ASCII. En otras palabras, significa un carácter, solo uno, de los que están en el conjunto dentro de los corchetes. Ejemplos: [aeiou] significa una vocal. [a-z] significa cualquier letra de la “a” a la “z”. [0-9CHAOS] significa cualquier número o una “C” o una “H” o una “A” o una “O” o una “S”
{ }	Metacaracter de expansión. Se utiliza para crear listas.
~	Representa la ruta absoluta al directorio hogar de quien ejecuta el proceso.
&	Utilizado para enviar un proceso a segundo plano en el momento en el que se ejecute.
!	Niega la salida generada por el comando que se ejecute.
( )	Para ejecutar comandos en un sub-shell.

DGTIC

Desactivación de metacaracteres

- Existen mecanismos para poder desactivar el significado de un metacaracter en shell.
- Esto significa que si se desactiva su significado, para el shell, los metacaracteres son caracteres literales.
- Por ejemplo, si se desactiva el significado de el caracter “espacio”, para el shell ya no significaría “final de palabra”.
- Se pueden desactivar los metacaracteres con el caracter \

DGTIC

Desactivación de metacaracteres

```
[chaos@gnulinux ~]$ echo Hola Mundo
Hola Mundo
[chaos@gnulinux ~]$ echo Hola \ \ \ \ \ \ \ \ Mundo
Hola         Mundo
[chaos@gnulinux ~]$ _
```

DGTIC

Comillas

Comilla	Descripción
‘ ‘	Comillas simples. Desactivan el significado de todos los metacaracteres.
“ “	Comillas dobles. Desactivan el significado de todos los metacaracteres, excepto: \ ` ` !X $

` `	Comillas invertidas.
	Ejecutan lo que esté dentro de las comillas como si fuera un comando y sustituye la salida de ese comando donde estén escritas las comillas invertidas. Ejemplo:
	`$ mv imagen.jpg foto`date +%d%m%y`.jpg`
	
	Primero que nada se ejecuta el comando date, obteniendo la fecha en formato ddmmaa. Lo cual se sustituiría por:
	
	`$ mv imagen.jpg foto251023.jpg`

DGTIC

Manejo del historial

- Comando history
- Muestra el historial de comandos ejecutados.
- La variable de entorno `HISTFILE` contiene la ruta al archivo del historial.
- Sintaxis:

```
$ history [n]
```

  - Donde n es un número entero positivo, para indicar que se muestren solo los últimos n comandos ejecutados.
  - Sin argumentos, muestra todo el historial registrado.
  - La opción `-c` se utiliza para eliminar el historial del archivo asociado.
  - La opción `-d n` se utiliza para eliminar la entrada n en el historial.

DGTIC

Manejo del historial, uso de !

Uso	Descripción
!!	Significa el comando anterior (completo).
!n	Comando n en el historial.
!cad	Ultimo comando que comienza con “cad”.
!:n	Palabra n del comando anterior, comienza desde 0.
!$	Última palabra del comando anterior.
!*	Todos los argumentos del comando anterior.

DGTIC

Variables

- Una variable es un identificador almacenado en memoria que tiene un nombre y un valor.

- Cuando se crea una variable en un shell, está disponible sólo para el mismo shell y no es heredada a los procesos hijos del shell.

- Una variable de ambiente sí es heredada a sus procesos hijos. Sin embargo, cuando un proceso hijo modifica una variable de ambiente, solo la modifica para sí mismo.

DGTIC

Variables

Uso	Descripción
`$ variable=Valor`	Crea la variable y le asigna un valor.
`$ unset variable`	Elimina la variable.
`$ export variable`	Hace que la variable sea de ambiente.
`$ export variable=Valor`	Crea la variable, le asigna un valor y la convierte en variable de ambiente.

DGTIC

Metacaracter $

- El metacaracter $ se utiliza para diversos fines en el shell.
- Las más comunes son:

Uso	Descripción
`$(comando)`	Funciona igual que las comillas invertidas. La diferencia es que pueden anidarse las instrucciones.
`$variable`	Sirve para poder acceder al contenido de una variable.
`${variable}`	Otra forma para acceder al contenido de una variable.

DGTIC

Metacaracter $

- Existen otros usos para las variables en algunos intérpretes (como bash):

Uso	Descripción
`${variable+existe}`	Utilizado para saber si una variable existe.
`${variable:-valor_predeterminado}`	Permite usar un valor predeterminado si la variable no está definida o está vacía.
`${variable:=valor_predeterminado}`	Asigna un valor predeterminado a la variable si está vacía o no definida.
`${#variable}`	Contiene la longitud de la cadena de la variable.

`${variable:inicio:fin}`	Permite extraer subcadenas del contenido de la variable. Ejemplos:
	
	`variable=”Hola a todos”`
	`echo ${variable:1:3}`
	`ola`
	
	`echo ${variable:3}`
	`a a todos`
	
	`echo ${variable:  -4}`
	`odos`
	
	`echo ${variable:  -4:2}`
	`od`

DGTIC

Metacaracter $

- Algunos usos extra

Uso	Descripción
`${variable/buscar/reemplazar}`	Busca lo escrito en `buscar` y reemplaza la primera ocurrencia con `reemplazar`. Puede funcionar para todas las ocurrencias con la sintaxis: `${variable//buscar/reemplazar}`
`${variable%sufijo}`	Elimina el sufijo del contenido de la variable.
`${variable#prefijo}`	Elimina el prefijo del contenido de la variable.
`${variable^^}`	Convierte el contenido a mayúsculas.
`${variable,,}`	Convierte el contenido a minúsculas.

DGTIC

Variables

- Comando set
  - Sin argumentos se utiliza para ver las variables y funciones existentes.
  - También utilizado para crear variables (cshell).

- Comando env
  - Utilizado para ejecutar un comando con modificaciones en el ambiente.
  - Sin argumentos se utiliza para ver las variables de ambiente.

DGTIC

Algunas variables de ambiente

- $HOME
- $LOGNAME
- $PWD
- $OLDPWD
- $UID
- $LANG
- $SHELL

```
[chaos@gnulinux ssh]$ echo $HOME
/home/chaos
[chaos@gnulinux ssh]$ echo $LOGNAME
chaos
[chaos@gnulinux ssh]$ echo $PWD
/etc/ssh
[chaos@gnulinux ssh]$ echo $OLDPWD
/home/chaos
[chaos@gnulinux ssh]$ echo $UID
1000
[chaos@gnulinux ssh]$ echo $LANG
en_US.UTF-8
[chaos@gnulinux ssh]$ echo $SHELL
/bin/bash
[chaos@gnulinux ssh]$ _
```

DGTIC

Variables especiales

Uso	Descripción
`$?`	Resultado del comando anterior. Si el valor es 0, el comando anterior se ejecutó sin errores. Si es diferente de 0, el comando se ejecutó con errores. El número depende de la definición de errores en la biblioteca de C errno.h
`$PS1`	Definición del prompt.
`$#`	Cantidad de argumentos del shell actual.
`$*`	Lista de todos los argumentos del shell actual.
`$n`	Argumento n del shell actual.

DGTIC

Comando shift

- Sintaxis:

  `$ shift [n]`

  - Recorre n veces los argumentos hacia la izquierda.

  - Si no se especifica n, recorre los argumentos una vez.

```
[chaos@gnulinux ~]$ cat ./recorre.sh
#!/bin/bash
echo "El valor de \$1 es $1"
echo "El valor de \$2 es $2"
echo "El valor de \$3 es $3"
echo "El valor de \$4 es $4"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "El valor de \$2 es $2"
echo "El valor de \$3 es $3"
echo "El valor de \$4 es $4"
echo "Realizando un shift de 2 posiciones..."
shift 2
echo "El valor de \$1 es $1"
echo "El valor de \$2 es $2"
echo "El valor de \$3 es $3"
echo "El valor de \$4 es $4"
[chaos@gnulinux ~]$
[chaos@gnulinux ~]$ ./recorre.sh A B C D
El valor de $1 es A
El valor de $2 es B
El valor de $3 es C
El valor de $4 es D
Realizando un shift...
El valor de $1 es B
El valor de $2 es C
El valor de $3 es D
El valor de $4 es
Realizando un shift de 2 posiciones...
El valor de $1 es D
El valor de $2 es
El valor de $3 es
El valor de $4 es
[chaos@gnulinux ~]$
```

DGTIC

Shift

```
[chaos@gnulinux ~]$ cat ./recorre2.sh
#!/bin/bash
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
echo "Realizando un shift..."
shift
echo "El valor de \$1 es $1"
[chaos@gnulinux ~]$
[chaos@gnulinux ~]$ ./recorre2.sh A B C D E F
El valor de $1 es A
Realizando un shift...
El valor de $1 es B
Realizando un shift...
El valor de $1 es C
Realizando un shift...
El valor de $1 es D
Realizando un shift...
El valor de $1 es E
Realizando un shift...
El valor de $1 es F
Realizando un shift...
El valor de $1 es
[chaos@gnulinux ~]$ _
```

DGTIC

Leer entrada estándar

- Comando read
- Sintaxis:

```
$ read [variable …]
```

  - Lee una línea de la entrada estándar y cada palabra la asigna a una variable.
  - Si hay más palabras que nombres de variables, se asigna lo sobrante a la última variable especificada.
  - Si hay menos palabras que nombres de variables, a las variables restantes se les asigna un valor vacío.

DGTIC

Leer entrada estándar

```
[chaos@gnulinux ~]$ read var1 var2 var3
A B C
[chaos@gnulinux ~]$ echo $var1;echo $var2; echo $var3
A
B
C
[chaos@gnulinux ~]$ read var1 var2 var3
A B C D E F G
[chaos@gnulinux ~]$ echo $var1;echo $var2; echo $var3
A
B
C D E F G
[chaos@gnulinux ~]$ read var1 var2 var3
A
[chaos@gnulinux ~]$ echo $var1;echo $var2; echo $var3
A


[chaos@gnulinux ~]$ _
```

DGTIC

Opciones de read

Opción	Descripción
`-u fd`	Lee del descriptor de archivos fd en lugar de la entrada estándar.
`-a arreglo`	Guarda las palabras en un arreglo con elementos secuenciales.
`-n nchars`	Lee hasta tener nchars caracteres leídos o un salto de línea.
`-N nchars`	Lee hasta tener exactamente nchars caracteres leídos.
`-s`	Silencioso. Hace que no se imprima mientras se escribe.

DGTIC