Shell Bash

Università di Modena e Reggio Emilia
Prof. Nicola Bicocchi (nicola.bicocchi@unimore.it)

Utilità

Builtins

- Bash, come interprete dei comandi, esegue file binari presenti nel sistema

```
$ ls
$ which ls
/bin/ls
$ which which
/usr/bin/which
```

Builtins

Esistono particolari comandi, detti bultins, che non provengono dall'esecuzione di un binario esterno ma sono implementati direttamente all'interno della shell. Nel loro caso, *$ which comando* non ritorna un percorso perchè il binario non esiste! Ad esempio:

```
$ cd
$ history
$ logout
```

https://www.gnu.org/software/bash/manual/html_node/Bash-Builtins.html

history

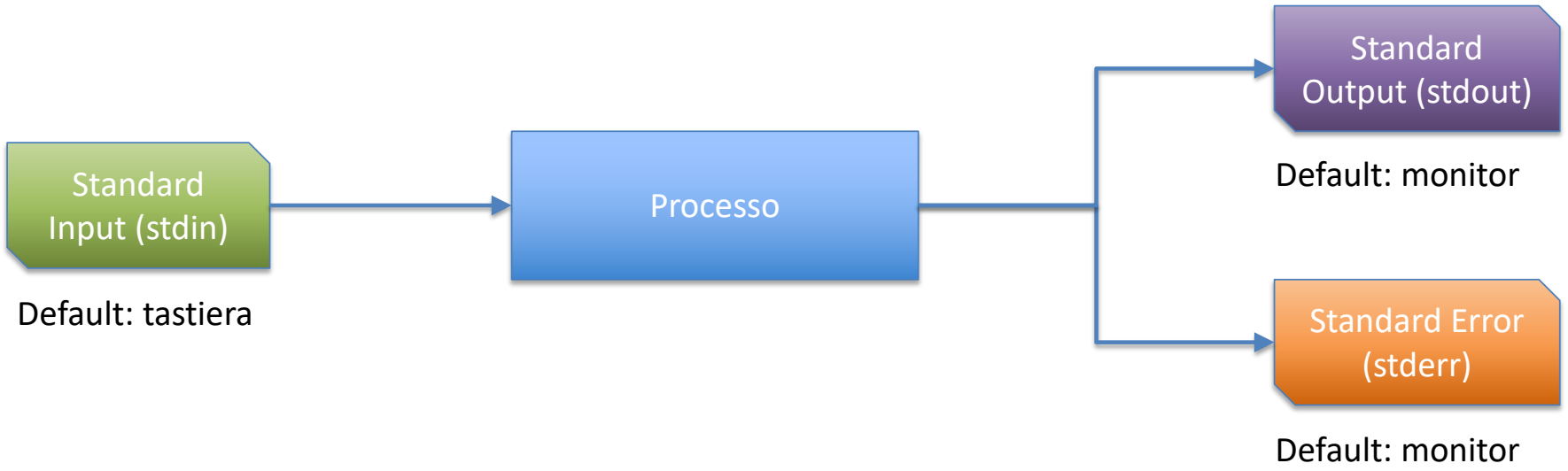
```
$ history
1 uname -a
2 clear
3 exit
4 ls

$ !!     (esegue ultimo comando)
$ !2     (esegue comando #2)
```

Freccia su-giù, ctrl-r, tab

- I tasti freccia (su e giù) consentono di spostarsi all'interno della lista dei comandi precedenti (lo stesso elenco mostrato dal commando history)
- Ctrl-r consente di inserire una stringa e selezionare tutti i comandi precedenti che la contengono. Ogni pressione della combinazione ctrl-r accede ai comandi successivi della stessa selezione
- Tab auto-completa i nomi di file. Una doppia pressione rapida mostra l'elenco di tutte le possibilità.

Flussi dati


Standard
Input (stdin)
Default: tastiera
Processo
Standard
Output (stdout)
Default: monitor
Standard Error
(stderr)
Default: monitor

Esempi filtri Unix

- cat [opzioni] [file…]
  - legge da file o stdin, scrive su stdout
- grep [opzioni] testo [file...]
  - Legge da file o stdin, scrive su stdout le linee che contengono <testo>
- cut [opzioni] [file…]
  - Legge da file o stdin, scrive su stdout un subset delle colonne del file
- head [opzioni] [file…]
  - Legge da file o stdin, scrive su stdout un subset delle righe (prime n)
- tail [opzioni] [file…]
  - Legge da file o stdin, scrive su stdout un subset delle righe (ultime n)
- sort [opzioni] [<file>...]
  - Legge da file o stdin, scrive su stdout line ordinate
- tee [opzioni] file
  - Legge da file, sdoppia il flusso in ingress su stdout e <file>

Ridirezione

- E’ possibile ridirigere input e/o output di un comando facendo sì che stdin/stdout/stderr siano sostituiti da file in modo trasparente al comando
- Ridirezione dell’input
  - `comando < filein`
- Ridirezione dell’output
  - `comando > fileout (sovrascrive file_output)`
  - `comando >> fileout (aggiunge alla fine di fileout)`

Ridirezione

```
$ cat /etc/passwd
```

cat apre il file /etc/passwd e stampa il contenuto su stdout

```
$ cat < /etc/passwd
```

cat legge da stdin, ma il flusso proviene da /etc/passwd

```
$ sort < f > f2
```

sort legge da stdin, ma il flusso proviene da f

sort scrive su stdout, ma il flusso è ridiretto su f2

```
$ head f2 > f3
```

head legge da f2

head scrive su stdout, ma il flusso è ridiretto su f3

Separazione stdout-stderr

$ grep nicola /etc/passwd

- Seleziona tutte le righe che contengono la stringa *nicola* all'interno del file */etc/passwd* e le stampa sul terminale
  - Se il file viene trovato, stampa il risultato su stdout
  - Se il file non viene trovato, stampa un errore su stderr
- E' possibile separare i due flussi menzionandoli in modo esplicito con i loro valori numerici (0 = stdin, 1=stdout, 2=stderr)

$ grep nicola /etc/passwd 1>/dev/null (scarta stdout, mostra solo stderr)

$ grep nicola /etc/passwd 2>/dev/null (scarta stderr, mostra solo stdout)

Separazione stdout-stderr

- Posso ridirigere un flusso all'interno di un altro flusso? Si!

$ grep nicola /etc/passwd 1>/dev/null 2>&1

$ grep nicola /etc/passwd >/dev/null 2>&1

(Stesso significato, scrittura meno chiara)

- Il flusso 2 viene ridiretto all'interno del flusso 1. Il carattere & chiarisce che non si tratta di un file di nome 1, ma del flusso 1 (stdout).

Pipes

Combinare comandi

- E’ possible combinare comandi utilizzando il filesystem come strumento di mediazione

```
$ sort f > f2; head f2
```

  - sort legge f, lo ordina, stampa su stdout (ridiretto su f2). head legge le prime line di f2. Il carattere ; è utilizzato per combinare comandi sulla stessa linea.
- Approccio estremamente inefficiente. La memoria secondaria (disco rigido) è molto meno performante della memoria primaria (ram)

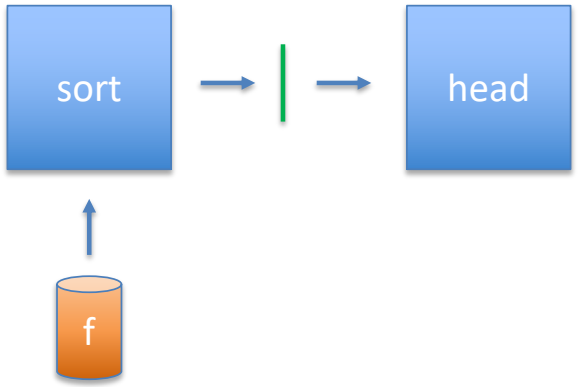
Combinare comandi

- L'output di un comando può esser diretto a diventare l'input di un altro comando (usando una pipe).
- Pipe come costrutto parallelo (l'output del primo comando viene reso disponibile al secondo e consumato appena possibile, in assenza di file temporanei)
- Si realizza con il carattere speciale '|'

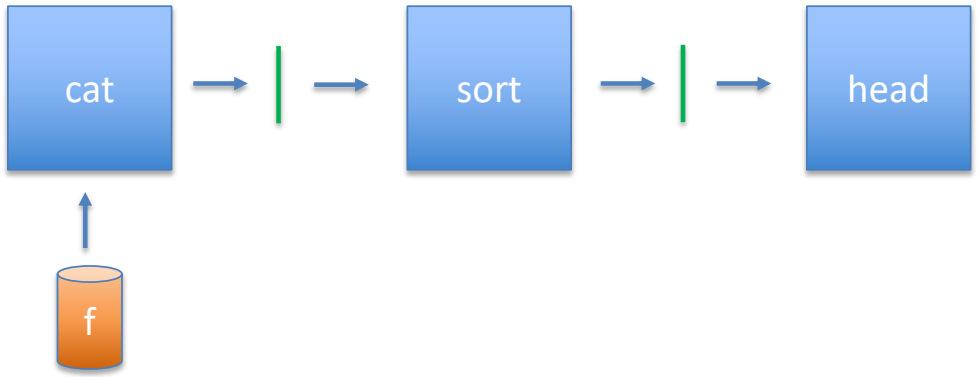
```
$ sort f | head
```

Combinare comandi

```
$ sort f | head
```




```
$ cat f | sort | head
```



Esempi

```
$ who | wc -l
```

Conta gli utenti collegati al sistema

```
$ ls -l | grep ^d | cut -d ‘ ‘ -f1 | sort
```

Stampa il contenuto della cartella corrente, seleziona le righe che iniziano per d, seleziona solo la prima colonna, ed ordina il risultato

Metacaratteri

Metacaratteri

- La shell riconosce caratteri speciali (wild card)
  - * una qualunque stringa di zero o più caratteri in un nome di file
  - ? un qualunque carattere in un nome di file
  - [abc] un qualunque carattere, in un nome di file, compreso tra quelli nell'insieme. Anche range di valori: [a-g]. Per esempio ls [q-s]* stampa tutti i file con nomi che iniziano con un carattere compreso tra q e s
  - \ segnala di non interpretare il carattere successivo come speciale

Metacaratteri

```
$ ls [a-p,1-7]*[c,f,d]?
```

Elenca i file i cui nomi hanno come iniziale un carattere compreso tra ‘a’ e ‘p’ oppure tra ‘1’ e ‘7’, e il cui penultimo carattere sia ‘c’, ‘f’, o ‘d’

```
$ ls *\**
```

Elenca i file che contengono, in qualunque posizione, il carattere ‘*’