Talk About Terminale

Ovvero: ginnastica per le dita
e per la mente, usando il terminale
con un occhio a bash

Daniele Forsi

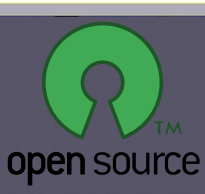
Gruppo Utenti Linux Livorno

https://linux.livorno.it/

Avvertenza

Nelle pagine seguenti vedremo intrecciarsi due aspetti:

- Il livello “fisico” del terminale che rappresenta l’interfaccia utente
  (tipicamente un emulatore, ma nulla vieta di usare un terminale hardware)
- Il livello “logico” del programma che usa il terminale come input/output
  (tipicamente getty+bash, ma nulla vieta di usare direttamente un’applicazione)



Elencare tutte le associazioni tasto-comando (binding)

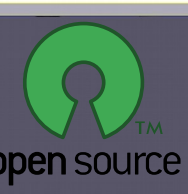
- `bind -p # è un comando interno di bash`

```
"\C-g": abort
"\C-x\C-g": abort
"\e\C-g": abort
"\C-j": accept-line
"\C-m": accept-line
# alias-expand-line (not bound)
# arrow-key-prefix (not bound)
```

☞ 482 righe

\c = Ctrl

\e = ESC



Muovere il cursore (1)

- Tasti singoli:
  freccia sx, freccia dx
  tasto “home”, tasto “end”

☞ \C = Ctrl

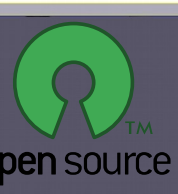
\e = ESC

- Combinazioni di tasti:

```
"\C-a": beginning-of-line
"\C-e": end-of-line
"\e[H": beginning-of-line
"\e[F": end-of-line
```

☞ home

☞ end



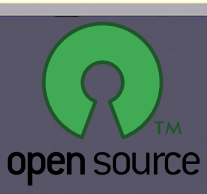
Muovere il cursore (2)

```
"\eb": backward-word
"\ef": forward-word
```

☞ Ctrl-freccia sx
Ctrl-freccia dx

```
"\e ": set-mark
"\C-@": set-mark
"\C-x\C-x": exchange-point-and-mark
```

```
"\C-]": character-search
"\e\C-]": character-search-backward
```

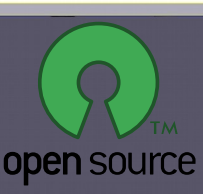



Modificare la riga corrente (1)

```
"\er": revert-line
"\C-_": undo

"\el": downcase-word
"\eu": upcase-word
"\ec": capitalize-word
"\ed": kill-word
"\C-k": kill-line
```

☞ tutti questi comandi agiscono da sotto il cursore verso destra



Modificare la riga corrente (2)

```
"\C-u": unix-line-discard
"\C-w": unix-word-rubout
```

☞ sempre tutta la riga

☞ da sotto il cursore verso sinistra

```
"\C-y": yank
"\e.": yank-last-arg
"\e_": yank-last-arg
```

☞ incollano quello che `\C-u` `\C-w` hanno eliminato; yank-last-arg scorre in ordine inverso tutte le eliminazioni

```
"\C-t": transpose-chars
"\et": transpose-words
```

☞ scambiano a sinistra del cursore, cioè con il carattere o la parola precedente

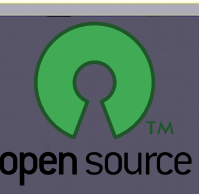
Modificare la riga corrente (3)

```
"\e!": complete-command
"\e/": complete-filename
"\e@": complete-hostname
"\e~": complete-username
"\e$": complete-variable
```

```
"\C-x!": possible-command-completions
"\C-x/": possible-filename-completions
"\C-x@": possible-hostname-completions
"\C-x~": possible-username-completions
"\C-x$": possible-variable-completions
```

```
"\C-q": quoted-insert
"\C-v": quoted-insert
"\e[2~": quoted-insert
```

☞ vi ricordate di home e end?



Muoversi nella cronologia

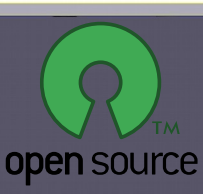
- `$ history # è un comando interno di bash`

```
"\e<": beginning-of-history
"\e>": end-of-history
"\C-s": forward-search-history
"\C-r": reverse-search-history
"\C-n": next-history
"\C-p": previous-history
```

freccia su
freccia giù

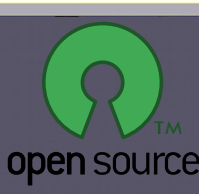
Cosa uso

- `\C-w \C-u \C-k \C-r Home End \C-FrecciaSx \C-FrecciaDx FrecciaSu FrecciaGiù`



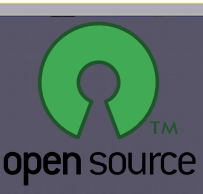
Fine prima parte

- Domande?



Facciamo lavorare bash

- Variabili
- Cicli for
- Alternative if-then-else
- ...altre cose inutili ;-)



Variabili

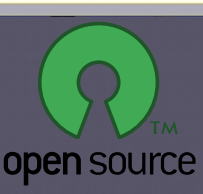
- Impostare: nome della variabile, seguito dal segno di uguale, seguito dal valore (senza spazi intorno all'uguale)

```
MIA_VARIABILE="virgolette se più parole"
MIA_VARIABILE= # imposta un valore vuoto
```

☞ Per convenzione: nomi TUTTI MAIUSCOLI con trattino basso

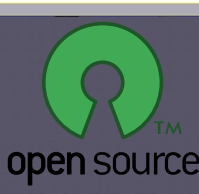
- Usare: segno del dollaro davanti al nome della variabile (senza spazi, usare le virgolette se necessario, es. se è nome di file)

```
echo $MIA_VARIABILE
```

Variabili

- Per eliminare (senza $):
  `unset MIA_VARIABILE`
- Per esportare agli script (senza $):
  `export MIA_VARIABILE`
- Per elencare:
  `set # (troppo output, usare more o less)`
- Curiosità. Modalità di modifica riga:
  `set -o emacs # (predefinita)`
  `set -o vi`

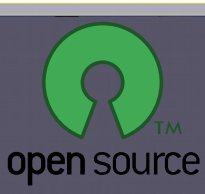


Variabili

- Matematica solo con numeri interi:
  `$((var_o_int + var_o_int))`
- Esempio:

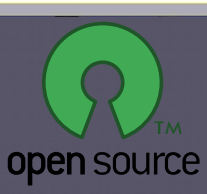
```
a=1
b=2
echo $((a + b))
```

- Per calcoli più complicati vedere
  `man bc`



Variabili utili

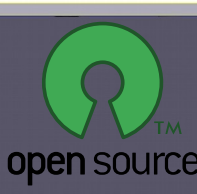
- $LANG
- $PATH
- $1 $2… $_
- $USER $HOME
- $PWD $OLDPWD
- $COLUMNS $LINES



Ciclo for

- Ciclo for:
  ```
  for name in word ...; do
  list ; done
  ```
- Esempi (costanti, nomi di file)
  ```
  for a in x y z; do
   echo $a; done

  for f in *; do
   echo "$f"; done
  ```

Ciclo for

- Esempio con sostituzione

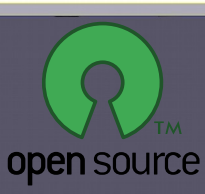
```
for f in aa/*.txt bb/*.txt; do
  cp "$f" "dest/${f/%txt/bak}";
done
```

- / ovunque prima occorrenza,
  // ovunque tutte le occorrenze,
  /# inizio stringa, /% fine stringa
- `man bash` Parameter Expansion, Compound Commands

Alternative if-then-else

- Istruzione If: `if list; then list; [ elif list; then list; ] ... [ else list; ] fi`
- Se il comando dopo “if” o “elif” esce con zero è “vero”, altrimenti è “falso” (vedere `/bin/true`, `/bin/false`)
- Esempi:

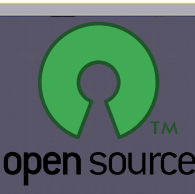
```
if true; then echo "sì"; else
echo "no"; fi
```

Alternative if-then-else

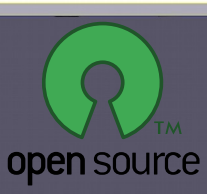
- Esempi (gli spazi dentro [ ] sono indispensabili, perché `/usr/bin/test` e `/usr/bin/[` hanno lo stesso scopo):

```
if [ -e miofile ]; then echo
esiste; fi
if [ -x miofile ]; then echo
"è eseguibile"; fi
```

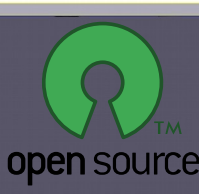
Alternative if-then-else

- `man bash` Compound Commands
- `man bash` Conditional Expressions
- `man [` `man test` Controlla valori e file



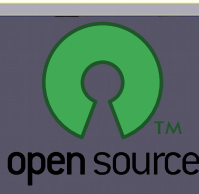
...altre cose inutili ;-)

- Vedere `man bash`



Fine seconda parte

- Domande?



Esercitazione pratica

- Configurazione “volante” del server
  ```
  # getty 38400,19200,9600 ttyUSB0
  ```
  (esce al logout)
- Oppure configurazione di systemd
  ```
  # systemctl start serial-
  getty@ttyUSB0.service
  ```


Esercitazione pratica

- Verificare la configurazione del terminale

```
$ echo $TERM
xterm-256color
$ stty --all
speed 38400 baud; line = 0;
-brkint ixoff -imaxbel iutf8
[...]
```


Esercitazione pratica

- Configurazione del terminale
  Ctrl-Select (Select è in alto a destra) poi F3 (Display Setup) e scegliere colonne e righe, poi F6 (Fkeys) configurare Canc con Ctrl-D
- F9 (Exit)
  `$ stty cols 132 rows 43`
  (notare che `/usr/bin/reset` fa perdere queste impostazioni a questo terminale ma `/bin/stty` le ricorda!)

Talk About Terminale

Fine.

Grazie per l'attenzione

Daniele Forsi

Gruppo Utenti Linux Livorno

https://linux.livorno.it/