Linux dalla *console* — 3

Bash scripting

Emiliano Vavassori

5 aprile 2017

FabLab Bergamo
Patronato San Vincenzo, Bergamo

Bergamo Linux Users Group

Sommario

Generalità sullo scripting

Generalità

Uno **script** è un file di testo contenente istruzioni che un *interprete* (ad esempio la shell) esegue nell'ordine in cui le stesse istruzioni sono state inserite.

`bash` può interpretare comandi *interni* o *esterni*: comandi propri della shell (`if`, `for`, `test`, ...) o comandi esterni (`ls`, `mkdir`, `grep`, `awk`, ...).

Cenni sull'esecuzione *batch*

```
$ bash -c '<comando 1>; <comando 2>; \
  <comando 3>'
```

Qualche trucco per eseguire più comandi...

```
$ <comando 1> && <comando 2> || <comando 3>
```

```
$ <comando molto lungo> &
```

```
$ fg
```

```
$ nohup <comando> >logfile
```

Anatomia di uno script

Estratto: **nomescript.sh**

```
#!/bin/bash
# Commenti

<comando 1>
<comando 2>
<comando 3>
<comando 4>

# Ulteriori comandi
```

Eseguire uno script

```
$ bash nomescript.sh
```

Eseguire uno script — Eseguibile?

```
$ chmod +x nomescript.sh
```

```
$ ./nomescript.sh
```

Qualche altro trucco sugli eseguibili...

Attenzione a chi do accesso in esecuzione!

```
$ chmod +x nomefile.sh || chmod u+x nomefile.sh
```

Percorso di sistema — $PATH

```
$ cp nomescript.sh $HOME/bin/; cd /tmp
```

```
$ nomescript.sh
```

Programmazione in bash

Definizione delle variabili

```
$ miavariabile="ciccio pasticcio"
```

Variabili d'ambiente:

```
$ export MIAVAR="valore"
```

Distruzione di una variabile

```
$ unset MIAVAR
```

Uso delle variabili

```
$ echo "$miavariabile" > ciccio.txt
```

```
$ echo "${miavariabile}" > ciccio.txt
```

Controllo di flusso — `if`

```
$ if [[ … ]]; then
  …
elif [[ … ]]; then
  …
else
  …
fi
```

Controllo di flusso — case

```
$ case $var in
  1)
    <comando 1>
    ;;
  2-5)
    <comando 2>
    ;;
  *)
    <comando default>
    ;;
esac
```

Ciclo `for`

```
$ for f in <seq>; do
  …
done
```

Sostituire i cicli con `find` e `xargs`

```
$ find . -iname \*.txt -exec chmod go-wx {} \;
```

```
$ find /tmp -iname \*.sh | xargs grep sudo
```

Farsi ritornare il risultato di un comando

```
$ var=` <comando 1> `
```

```
$ var=$( <comando 2> )
```

Input dall'utente

Caserecci:

```
$ echo -n "Dimmi il tuo nome: "; read var
```

Un po' più carini:

```
$ var=`whiptail --inputbox "Dimmi il tuo nome:" \
  0 0 3>&1 1>&2 2>&3`
```

Non sempre comodo...

Spesso e volentieri si preferisce chiedere input all'utente tramite gli *argomenti* dello script.

```
$ bash mioscript.sh argomento1 argomento2
```

Argomenti

`$0` Percorso e nome dello script

`$1` Primo argomento

`$2` Secondo argomento

...

`$@` Tutti gli argomenti

`$#` Il numero degli argomenti passati

Lavorare con i file

```
$ dirname $var
```

```
$ basename $var .ext
```