

# Comanda il tuo computer

# 1

---

*Uno dei primi e più bei giochi per computer nel quale veniva costruito un mondo da esplorare è stato Colossal Cave Adventure ([http://it.wikipedia.org/wiki/Colossal\\_Cave\\_Adventure](http://it.wikipedia.org/wiki/Colossal_Cave_Adventure)), creato nel 1976.*

In questo capitolo imparerai cos'è la shell, la finestra dove puoi digitare i comandi da impartire al computer. Aggiungerai queste conoscenze alla tua toolbox:

- come aprire una shell e digitarvi dei comandi
- com'è fatto un file system: file e cartelle
- come spostarsi tra le cartelle del file system

Era un gioco d'avventura basato solo sul testo, senza immagini. Per dare le istruzioni bastava digitare delle frasi semplici, per esempio “vai a nord”, “prendi l'ascia” o “uccidi il troll”, e il gioco faceva quello che gli si chiedeva, compreso strangolare il troll a mani nude.

Anche oggi i giochi utilizzano dei comandi di testo, Minecraft compreso. Ti sarà senz'altro capitato di digitare dei comandi nella finestra della chat usando un carattere /.

In questo capitolo imparerai a impartire dei comandi al tuo computer in un modo molto simile, usando la *riga di comando* per costruire dei plug-in e lavorare con i file.

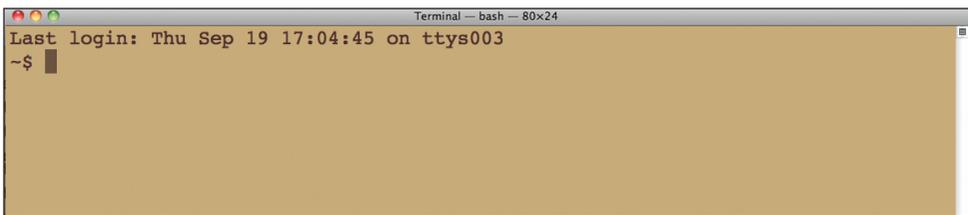
La riga di comando è uno strumento potente che ti permette di lavorare sul tuo computer così come su computer lontani, per esempio sul cloud. Parleremo meglio di come farlo nell'Appendice D, “Come installare un server sul cloud”.

Puoi anche usare un elaboratore a riga di comando per scrivere i programmi; l'elaboratore contiene un proprio linguaggio di programmazione, separato da Java. Per darti una mano, ho preparato uno *script* (cioè un elenco di comandi eseguibili) che ti aiuterà a costruire e installare i plug-in. Lo vedremo tra poco.

Se sai già usare la riga di comando, puoi passare direttamente alla fine del capitolo.

## Usare la riga di comando

Sul mio computer, la riga di comando ha l'aspetto che vedi in questa figura. Sul tuo computer potrebbe avere dei colori e dei caratteri diversi, e comunque puoi sempre impostarla secondo i tuoi gusti. Io preferisco le lettere nere su uno sfondo marroncino.



Ci sono vari modi per accedere alla riga di comando, che sono leggermente diversi a seconda del sistema operativo con cui lavori.

Windows ha una riga di comando molto essenziale, che si apre con il comando `cmd.exe`. Se la tua versione di Windows ha un comando *Start*, puoi selezionare *Start > Esegui* e poi digitare `cmd.exe`, oppure puoi usare la casella di ricerca che si apre per trovare ed eseguire `cmd.exe`. Il mio consiglio è però quello di non usare `cmd.exe` da solo; a questo proposito, leggi le istruzioni nel riquadro “Installare BusyBox in Windows” più avanti nel capitolo.

Su OS X, puoi aprire la riga di comando selezionando *Applicazioni > Utility > Terminale*.

Se usi Linux, è molto probabile che tu sappia già come accedere alla riga di comando. Per completezza, comunque, hai tre possibilità: aprire una shell, avviare la console o fare clic con il tasto destro mouse sul Desktop e aprire il terminale.

(Fortunatamente, tutte queste fastidiose differenze tra Windows, OS X e Linux spariscono una volta che inizi a scrivere codice Java; questo linguaggio di programmazione, infatti, si comporta nello stesso modo su tutte le piattaforme.)

Una volta aperta l'applicazione della riga di comando, puoi iniziare a digitare i comandi nell'elaboratore, chiamato *shell*. I comandi che utilizzerai sono pochi e semplici; te li spiegherò man mano.

Ogni shell visualizza un breve messaggio, chiamato *prompt*, che indica che è pronta per accettare i comandi che immetti. Tuttavia, invece di un messaggio chiaro come “Digita i tuoi comandi, maestro. Sono pronto”, i prompt che vedrai sono un po' più misteriosi.

Windows mostra in genere qualcosa come C:\>. I sistemi Linux e OS X potrebbero mostrare \$ o %. Inoltre i prompt potrebbero includere altre informazioni, come il tuo nome, quello del computer o quello di una cartella (*directory*). Poiché ogni sistema è diverso, negli esempi del libro ho scelto il più semplice e ti mostrerò il prompt dei comandi come \$. Indipendentemente dall'aspetto del prompt sul tuo computer, quando vedi \$, è lì che devi immettere i comandi.

NON DIGITARE IL SEGNO DEL DOLLARO (\$). È solo un simbolo che uso per indicarti dove devi digitare, ma non devi mai includerlo in quello che scrivi:

**\$ digita qui, ma non inserire il segno del dollaro**

Quando hai finito di scrivere un comando e vuoi che il computer lo esegua, premi il tasto Invio. Questa era la parte facile. Adesso devi imparare qualche comando di base!

Per ottenere un elenco di file e cartelle, si usa il comando `ls`:

**\$ ls**

Per passare a una cartella diversa da quella corrente, si usa invece il comando `cd`. Per esempio, per passare alla cartella *Desktop*, digita

**\$ cd Desktop**

e verrai trasportato a *Desktop* (o a qualsiasi altra cartella che hai indicato). Quando la shell si apre per la prima volta, ti trovi in una specie di cartella predefinita, di partenza, chiamata *home* (“casa”). Per vedere quali file contiene, digita il comando `ls` (che sta per *list files*, ovvero “elenca file”) e premi Invio.

Nei sistemi Windows che usano `cmd.exe`, devi invece digitare `dir`. Mentre il resto del mondo usa il carattere / nei nomi delle cartelle, Windows usa una barra che va nella direzione opposta (\), chiamata *backslash*.

Queste differenze spiegano perché in Windows è consigliabile usare il bash della shell standard POSIX, come ti spiego tra poco nel riquadro “Installare BusyBox in Windows”.

Vedrai apparire un elenco di file. Digita `cd Desktop`, e ti troverai nella cartella *Desktop*. Digita il comando `ls`, e vedrai un elenco dei file di *Desktop*. Visto? Stai lavorando con la riga di comando! Andiamo avanti.

## Spostarsi tra le cartelle

Quando vuoi vedere i file sul disco fisso del tuo computer, in genere usi dei programmi “grafici”, come Esplora risorse in Windows o il Finder su OS X. L’idea è quella di mostrarti i file, le cartelle e le applicazioni/programmi sul computer in modo che tu possa spostarti tra di essi e lavorare.

Qui faremo la stessa cosa, ma in modo più potente e senza immagini. Se sai già di cosa parlo, puoi saltare direttamente alla fine del capitolo. Altrimenti continua a leggere.

### Installare BusyBox in Windows

Gli ambienti OS X e Linux sono basati su Unix, che ha un ambiente a riga di comando e una shell molto ricchi. Per Unix esiste uno standard, chiamato POSIX, che include i comandi e le caratteristiche del linguaggio. La shell dello standard POSIX è in realtà un linguaggio di programmazione vero e proprio, che permette di scrivere degli script per compiere delle operazioni di base sul computer.

Windows non è però così sofisticato. L’elaboratore dei comandi predefinito non fa granché, i comandi hanno nomi diversi e i nomi di cartelle e file sono specificati in modo differente. Per ottenere un po’ di omogeneità, consiglio sempre agli utenti Windows di scaricare BusyBox. Questo software installa una shell più professionale e alcuni comandi comuni compatibili con POSIX, gli stessi usati da OS X, Linux e dal resto mondo. Per installare la shell e i comandi di base per Windows, scarica sul Desktop il file <ftp://ftp.tigress.co.uk/public/gpl/6.0.0/busybox/busybox.exe>.

Una volta scaricato il file, rinominalo da `busybox.exe` a `sh.exe`. Poi apri una finestra `cmd.exe` e digita quanto segue:

```
C:\> cd Desktop
C:\> C:\Windows\system32\cmd.exe /c sh.exe -l
```

Vedrai apparire una nuova shell con un prompt con il segno di dollaro. È da qui che lanceremo i comandi e compiremo le nostre azioni. Nota un dettaglio importante: `sh.exe` ha un flag `-l` che dice alla shell di agire come una “shell di login” nella quale puoi digitare i comandi.

Per comodità puoi tenere un file batch sul Desktop da dove lanciare la shell. Per farlo, crea un file di testo e salvalo sul Desktop. Chiama il file `shell.bat` e digita questa riga di testo al suo interno:

```
C:\Windows\system32\cmd.exe /c sh.exe -l
```

Salva il file. Ora puoi fare doppio clic su `shell.bat` e ottenere una shell compatibile con POSIX. Fatto questo, puoi accedere a tutti i comandi che utilizzeremo (come `ls`, `mv`, `cp` e `pwd`) e specificare i nomi delle cartelle usando il carattere `/` invece del `\` di Windows. In questo modo, tutto funzionerà nella stessa maniera in Windows, OS X e Linux, e potrai anche usare gli script di comando che troverai nel libro per creare e installare i plug-in.

L'home page di BusyBox all'indirizzo <http://intgat.tigress.co.uk/rmy/busybox/index.html> contiene ulteriori informazioni su BusyBox per Windows.

L'insieme dei file e delle cartelle sul tuo computer è chiamato *file system*. In qualsiasi momento, la finestra del Finder o di Esplora risorse sul Desktop o della shell dei comandi mostra la cartella corrente.

Sul Desktop puoi avere aperte anche più finestre, una per ciascuna cartella sul disco.

Tutte le finestre di shell aperte funzionano nello stesso modo: a ogni finestra corrisponde una *cartella corrente*. Alcuni sistemi sono impostati in modo da mostrarti la cartella corrente al prompt.

Tuttavia, puoi sempre capire in quale cartella ti trovi digitando il comando `pwd` (che sta per *print working directory*, cioè “mostra cartella di lavoro”):

```
$ pwd
/Users/andy/Desktop
```

Prova così: apri una nuova shell e, prima di fare qualsiasi altra cosa, digita `pwd` al prompt (qui è mostrato come `$`; sul tuo computer potrebbe apparire diversamente):

```
$ pwd
```

Questo comando ti indica la *cartella home*, che è il punto da cui partirà ciascuna delle tue shell.

In ogni shell, tutti i comandi che lanci vengono eseguiti secondo questa idea di cartella corrente: molti dei programmi che usi partono da qui per eseguire, aprire e salvare i file.

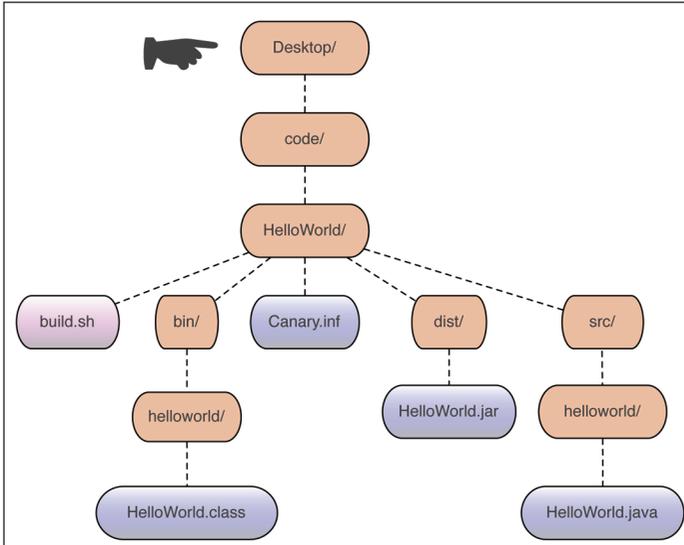
Se non l'hai ancora fatto, scarica sul Desktop i file del libro (<http://media.pragprog.com/titles/ahmine2/code/ahmine2-code.zip>) e decomprimi l'archivio (usa `unzip` dalla riga di comando oppure, in Windows, usa WinZip o 7-Zip). Il file verrà decompresso in una cartella chiamata *code*, che contiene tutti gli esempi del libro.

Sotto a *code* si trova un gruppo di cartelle di plug-in, una per ciascun plug-in del libro. Inizieremo dando un'occhiata ai file di *HelloWorld*. Questa cartella contiene altri file e sottocartelle.

Le figure che seguono ti mostrano come spostarti tra le cartelle.  
Partendo da qui:

**\$ cd Desktop**

ti trovi qui:

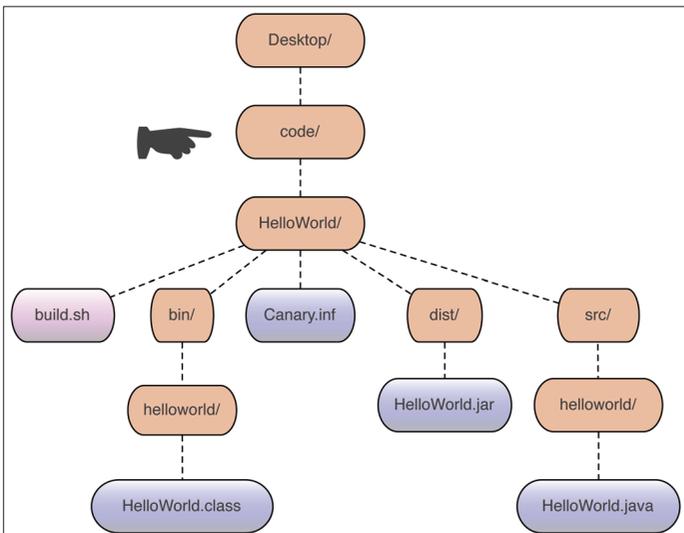


(Se non funziona, prova con `cd ~/Desktop`, oppure vai alla fine del capitolo per qualche suggerimento.)

Ora scendi nella cartella `code` digitando

**\$ cd code**

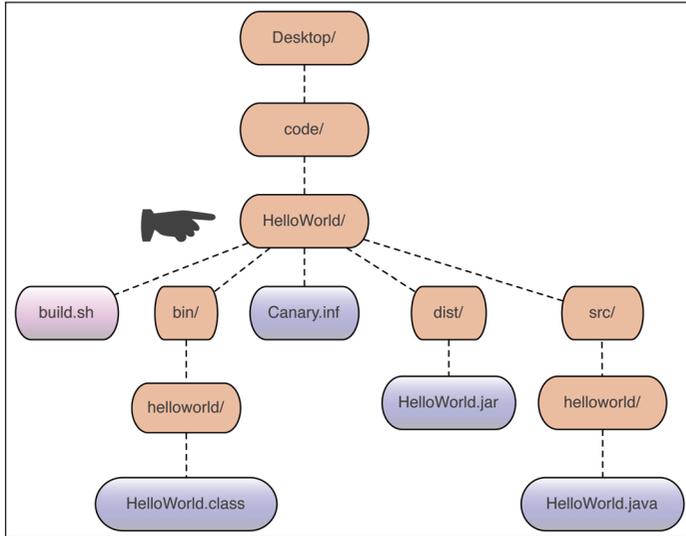
Ti troverai qui:



Entra in *HelloWorld*:

```
$ cd HelloWorld
```

... e ti troverai qui:



Ora elenca i file che si trovano in questa cartella. Vedrai questo:

```
$ ls
```

```
Canary.inf bin/ build.sh dist/ src/
```

Il mio sistema (OS X) è configurato in modo da mostrare le cartelle con uno slash (una barra, /) alla fine. (Se per te non è così, prova a digitare `ls -F`.) In questo esempio la mia cartella corrente contiene due file e altre tre cartelle. I tipi di file sono spesso identificabili dall'ultima parte del nome, il *suffixo*. Qui c'è uno script di shell con un suffisso `.sh` e un file di configurazione con un suffisso `.inf`. La cartella `src/` contiene una sottocartella `helloworld`; qui si trova un file sorgente Java con un suffisso `.java` (imparerai di più sui tipi di file man mano che proseguiremo).

Per entrare nella cartella `src`, digita `cd src`:

```
$ cd src
```

```
$ ls
```

```
helloworld
```

Scendi ancora più in profondità, in `helloworld`:

```
$ cd helloworld
```

```
$ ls
```

```
HelloWorld.java
```

Nella cartella `src/helloworld` si trova il file `HelloWorld.java`, il cuore del nostro primo plug-in.

Sei arrivato fino a *HelloWorld/src/helloworld*. Ma come fare per tornare a *HelloWorld*? Per retrocedere di un livello digita questo:

```
$ cd ..
```

Per tornare indietro di due livelli digita questo:

```
$ cd ../..
```

Hai visto i due punti? Un punto (.) indica la cartella corrente; con *cd*, non serve a molto, ma con altri comandi sì, soprattutto quando si devono copiare dei file.

Ora immagina di dover accedere a una cartella che non si trova in quella corrente o in quella immediatamente sopra. Come fare? Supponi di trovarti in un punto completamente diverso, per esempio in */home/mincraft*, e di dover arrivare a */Users/andy/Desktop/code/HelloWorld*.

Dovrai scrivere qualcosa del genere:

```
$ cd /Users/andy/Desktop/code/HelloWorld
```

Lo slash è quello che fa la differenza. In precedenza, quando hai digitato *cd src*, il comando *cd* ha cercato *src* sotto la cartella corrente. Ma se avessi digitato *cd /src*, avresti cercato una cartella chiamata *src* sotto una cartella chiamata */*, la cosiddetta *root*.

La cartella *root* è quella che si trova in cima a tutto il file system. È a capo del codice, del Desktop, di tutto. Sotto alla *root* si trovano la cartella *home* e il Desktop. Nel mio caso il percorso è */Users/andy/Desktop*. Volendo, potrei arrivare in questo punto digitando questa sequenza di comandi:

```
$ cd /
$ cd Users
$ cd andy
$ cd Desktop
```

Ma come vedremo tra poco esiste un metodo molto più rapido. Parlando di scorciatoie, infatti, non serve neanche specificare il nome per intero.

Sulla maggior parte dei sistemi, esiste una scorciatoia da tastiera che permette di abbreviare la digitazione dei nomi più lunghi: è il tasto Tab, . Se digiti le prime lettere di un nome lungo e poi premi Tab, il nome si completerà automaticamente. Supponi di essere nella tua cartella *code*:

```
$ ls
```

Adventure	CreeperCow	LocationSnapshot	SquidBomb
ArrayAddMoreBlocks	EZPlugin	MySimple	SquidBombConfig
ArrayOfBlocks	FireBow	NameCow	Stuck
BackCmd	FlyingCreeper	NamedSigns	install
BackCmdSave	HashPlay	PlayerStuff	mkplugin.sh
BuildAHouse	HashPlayClamp	PortingGuide.txt	numbers
CakeTower	HelloWorld	Simple	runtime
CanaryLinks.txt	LavaVision	Simple2	
CowShooter	ListPlay	SkyCmd	

Se digiti `cd Ad` e poi premi Tab (o abbastanza lettere perché il nome sia distinguibile), il nome si completerà automaticamente

```
$ cd Adventure/
```

e ti basterà premere Invio. Nel caso di nomi brevi questo trucco non serve granché, ma se hai una cartella dal nome lunghissimo, come *RumpelstiltskinReincarnationSpellPlugin*, digitare `Ru` e poi premere Tab diventerà molto utile!

### Copia e incolla

Può accadere che tu debba copiare del testo e incollarlo nella riga di comando. Per esempio, potresti voler copiare una riga da questo libro e poi incollarla.

Copiare e incollare dalla riga di comando può essere leggermente diverso che farlo in un'applicazione come Word o in un browser web. Per iniziare, devi comunque fare clic sul testo con il mouse e trascinare per selezionarlo.

In Linux, nella maggior parte delle applicazioni, puoi premere i tasti `Ctrl+C` per copiare e `Ctrl+V` per incollare. Se si trovi nella riga di comando, potresti dover aggiungere anche il tasto Maiusc; quindi, per copiare, premi `Ctrl+Maiusc+C`.

Su OS X, usa `CMD+C` per copiare e `CMD+V` per incollare.

La finestra della riga di comando di Windows è leggermente diversa. Per prima cosa devi abilitare la *Modalità modifica rapida*. Fai clic con il tasto destro del mouse sulla barra in alto nella finestra della shell e seleziona *Proprietà*. Nella scheda *Opzioni*, nella sezione *Opzioni di modifica*, spunta la casella *Modalità modifica rapida* per attivarla.

Per utilizzare questa modalità, dopo aver selezionato il testo, premi Invio per copiare e poi fai clic con il tasto destro del mouse o premi `Ctrl+V` per incollare.

Questo espediente vale solo per la riga di comando. Altrove nel sistema (il tuo editor di testo e così via), usa sempre `Ctrl+C` e `Ctrl+V`, come al solito.

### Prova da solo

Creiamo ora qualche cartella e file attraverso la riga di comando. Imparerai a creare la tua copia di un plug-in, con le cartelle e tutto il resto.

Parti dalla cartella *Desktop* (per essere certo di essere lì, digita `pwd`) e crea una nuova cartella chiamata *myplugins* usando il comando `mkdir`:

```
$ cd Desktop
$ pwd
/Users/andy/Desktop
$ mkdir myplugins
```

Elenca i file contenuti in *Desktop* (usando `ls`), e vedrai tutti i file racchiusi in questa cartella, oltre che una nuova cartella, *myplugins*. Entra in *myplugins*:

```
$ cd myplugins
```

Usa `pwd` per avere una conferma che ti trovi in *myplugins*.

Se esegui qui il comando `ls` ma non ti appare niente, è solo perché non abbiamo ancora creato dei file. Rimediamo realizzando la struttura della cartella, che sarà la

stessa di quella del plug-in *HelloWorld*. Inizia creando una cartella che si chiamerà come il plug-in:

```
$ mkdir HelloWorld
```

Ora esegui `cd` per scendere in profondità in *HelloWorld*:

```
$ cd HelloWorld
```

Adesso puoi creare tutte le cartelle che ti servono: *src*, *src/helloworld*, *bin* e *dist*:

```
$ mkdir src
$ mkdir src/helloworld
$ mkdir bin
$ mkdir dist
```

Usa `ls` per assicurarti che ci siano:

```
$ ls
bin/ dist/ src/
```

```
$ ls src
helloworld/
```

Qui ti servono tre file, che puoi copiare dal codice di esempio del libro. Puoi trascinarli e rilasciarli utilizzando la consueta finestra grafica, oppure puoi usare il comando per copiare, `cp`:

```
$ cp ~/Desktop/code/HelloWorld/build.sh .
```

Il carattere della tilde (`~`) sta per “la mia cartella *home*.” E ricorda: devi usare il punto! La riga di comando completa significa “copia questo file nella cartella corrente”.

Ti serve anche il file che ti indico qui sotto, quindi copialo:

```
$ cp ~/Desktop/code/HelloWorld/Canary.inf .
```

Hai così creato le cartelle e i file di supporto necessari per un plug-in.

### I file sul Desktop

In questo libro faremo tutto il lavoro sul Desktop perché è il punto più facile dove trovare i file, e vale tanto in Windows quanto su OS X e Linux.

Nella vita quotidiana, è improbabile che tu voglia riempirlo con tutti i nuovi progetti su cui stai lavorando, ma finché non avrai acquisito dimestichezza su come muoverti e impostare gli ambienti, rimani sul Desktop.

### Se non funziona

Una situazione che potrebbe crearti problemi è quella in cui il nome della tua cartella *home* contiene degli spazi. Per esempio, se lavori in Windows e ti chiami “Mario Rossi”, digitando un comando usando la tilde

```
$ cp ~/Desktop/code/HelloWorld/build.sh .
```

otterrai qualcosa del genere:

```
$ cp C:/Users/Mario Rossi/Desktop/code/HelloWorld/build.sh .
```

Il computer interpreta la riga come se dicesse “copia C:/Users/Mario e Rossi/Desktop/code/HelloWorld/build.sh nella cartella corrente”, e otterrai un errore del tipo “nessun file o cartella” (“no such file or directory”).

Le soluzioni sono due. Puoi usare un percorso relativo digitando `..` per le cartelle genitore, così da salire di “due livelli” ed entrare in *code*:

```
$ cp ../../code/HelloWorld/build.sh .
```

Oppure puoi digitare il tutto direttamente racchiudendo il nome del file tra virgolette:

```
$ cp "C:/Users/Mario Rossi/Desktop/code/HelloWorld/build.sh" .
```

Un altro problema che potresti incontrare è quando non sei nella cartella in cui credevi di trovarti. Se hai un dubbio, puoi sempre ricorrere al comando `pwd` per visualizzare la cartella di lavoro corrente:

```
$ pwd
/Users/andy/Desktop
```

Qui sono nella cartella *Desktop*, il punto da cui inizierà la maggior parte del nostro lavoro.

## Partiamo dalla cartella Desktop

Sui quasi tutti i sistemi è possibile digitare `cd Desktop` per accedere alla cartella *Desktop*. Se non funziona, potresti dover digitare `cd ~/Desktop` – con la tilde – oppure potresti dover scrivere tutto per esteso, come in `cd /Users/andy/Desktop`.

Qualunque sia il metodo utilizzato, quando ti dico di iniziare da *Desktop* o semplicemente di digitare `cd Desktop`, dovrai farlo sempre, a prescindere da dove ti trovi in quel momento, come:

```
$ cd Desktop
```

dalla tua cartella *home*, innanzitutto:

```
$ cd
$ cd Desktop
```

usando una tilde per la cartella *home*:

```
$ cd ~/Desktop
```

digitando il nome per esteso della tua cartella *home*:

```
$ cd /Users/mario/Desktop
```

o racchiudendo il tutto tra parentesi se il nome contiene degli spazi:

```
$ cd /Users/"Mario Rossi"/Desktop
```

Quello che vedrai sarà sempre

```
$ cd Desktop
```

## E ora divertiamoci

Nel codice del libro che hai scaricato, nella cartella *Desktop/code*, trovi una sottocartella speciale chiamata *Adventure*. Usando la riga di comando, digita `cd` e dai un'occhiata ai file e alle cartelle che contiene.

Puoi utilizzare `ls` per elencare le cartelle come abbiamo fatto qui. Per vedere rapidamente i file di testo (quelli con il suffisso `.txt`), puoi usare il comando `cat`.

Parti dalla cartella *Desktop* (in uno dei modi sopra descritti):

```
$ cd Desktop
```

```
$ cd code
```

```
$ cd Adventure
```

```
$ cat README.txt
```

```
These are some files to make exploring the file system a little more fun.
```

Esegui il comando `ls`, guarda l'elenco che appare ed esplora un po' le sottocartelle. Scopri i tesori – e i pericoli – che contengono.

## Comandi più comuni

Ecco alcuni dei comandi più comuni che utilizzerai nella shell.

---

<code>java</code>	Esegue le classi e gli archivi Java (JAR) come un programma.
<code>javac</code>	Compila il codice sorgente Java nei file <code>.class</code> .
<code>cd</code>	Cambia la cartella.
<code>pwd</code>	Mostra la cartella di lavoro corrente.
<code>ls</code>	Elenca i file nella cartella corrente.
<code>cat</code>	Visualizza il contenuto di un file.
<code>echo</code>	Visualizza un testo; con il prefisso <code>\$</code> , mostra anche le variabili d'ambiente.
<code>mkdir</code>	Crea una nuova cartella.
<code>cp</code>	Copia un file.
<code>mv</code>	Sposta un file.
<code>rm</code>	Cancella definitivamente un file. Usalo con molta attenzione; non è uguale al Cestino e non hai la possibilità di fare <i>Annulla</i> .
<code>chmod</code>	Modifica i permessi dei file (compresi quelli di lettura, scrittura ed esecuzione).
<code>.</code>	(Un punto) Indica la cartella corrente.
<code>..</code>	(Due punti) Indica la cartella genitore.
<code>~</code>	(Tilde) Indica la tua cartella <i>home</i> .

---

## Per continuare

Adesso devi imparare a digitare nel codice sorgente Java. Ti servirà Java e un'applicazione che metta insieme il tutto. Scaricheremo questi due elementi nel prossimo capitolo, e poi passeremo a creare dei plug-in.

## La tua toolbox



### Adesso sai come:

- Usare la shell a riga di comando.