

# w3ii.com (/it/index.html)

Gli ultimi tutorial di sviluppo web



(/index.html)

ESERCITAZIONI ▼

Italiano

**Amo viaggiare così.  
Sul divano 22H.**

Nuova Premium  
Economy Class:  
per chi vuole  
di più

→ Per saperne di più



Lufthansa

## Batch Script Guida veloce

«Precedente (batch\_script\_

Prossimo capitolo " (batch\_script

## Script batch - Panoramica

Script batch è incorporato per automatizzare sequenze di comandi, che sono di natura ripetitiva. Scripting è un modo con cui si può alleviare questa necessità automatizzando queste sequenze di comandi in modo da rendere la propria vita al guscio più facile e più produttivo. Nella maggior parte delle organizzazioni, script batch è incorporato in un modo o l'altro per automatizzare roba.

Alcune delle caratteristiche di Batch Script sono -

- Può leggere gli ingressi da parte degli utenti in modo che possa essere ulteriormente elaborati.
- Ha strutture di controllo, come per se, mentre, passare per una migliore automatizzare e di scripting.
- Supporta funzionalità avanzate come funzioni e array.
- Supporta le espressioni regolari.
- Può includere altri codici di programmazione come Perl.

Alcuni degli usi più comuni di Batch Script sono -

- Impostazione dei server per scopi diversi.

- Automatizzando le attività di pulizia, come l'eliminazione di file indesiderati o file di log.
- Automatizzare l'implementazione di applicazioni da un ambiente all'altro.
- Installazione di programmi su diverse macchine in una sola volta.

script batch sono memorizzate in semplici file di testo contenenti linee con comandi che vengono eseguiti in sequenza, uno dopo l'altro. Questi file hanno l'estensione BAT speciale o CMD. I file di questo tipo sono riconosciuti ed eseguiti attraverso un'interfaccia (talvolta chiamato shell) fornito da un file system chiamato l'interprete di comandi. Sui sistemi Windows, questo interprete è conosciuto come cmd.exe.

L'esecuzione di un file batch è una semplice questione di semplicemente cliccando su di esso. I file batch possono essere eseguiti anche in un prompt dei comandi o alla linea di partenza-Run. In tal caso, il nome completo del percorso deve essere utilizzato a meno che il percorso del file è in un ambiente percorso. Di seguito è un semplice esempio di un batch di script. Questo lotto di script quando viene eseguito elimina tutti i file nella directory corrente.

```
:: Deletes All files in the Current Directory With Prompts and Warnings  
::(Hidden, System, and Read-Only Files are Not Affected)  
:: @ECHO OFF  
DEL . DR
```

## Script batch - Ambiente

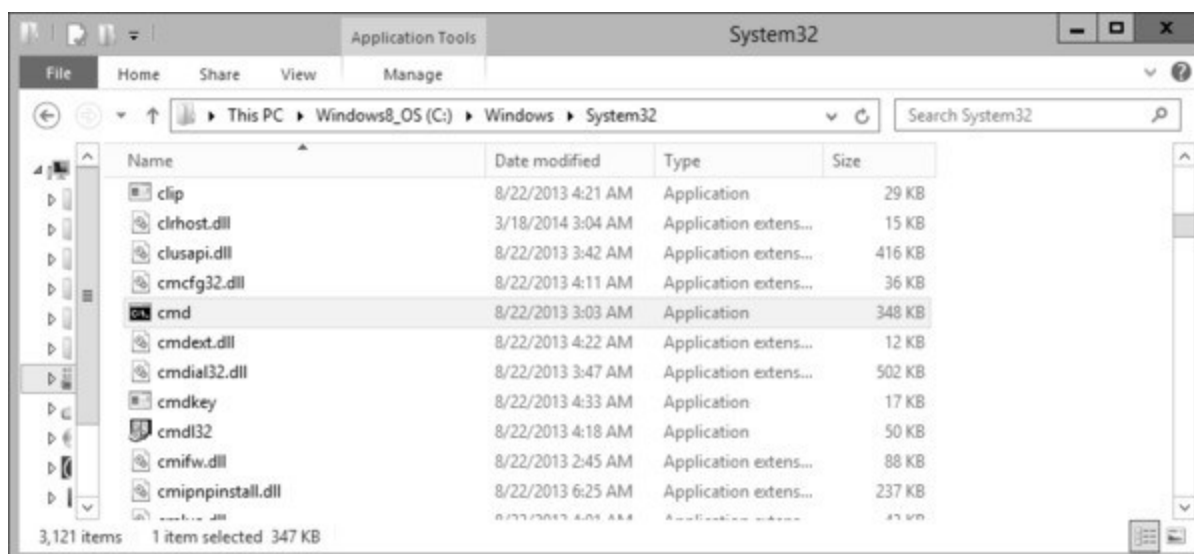
Questo capitolo illustra l'ambiente connessi alla batch Script.

### La scrittura ed esecuzione

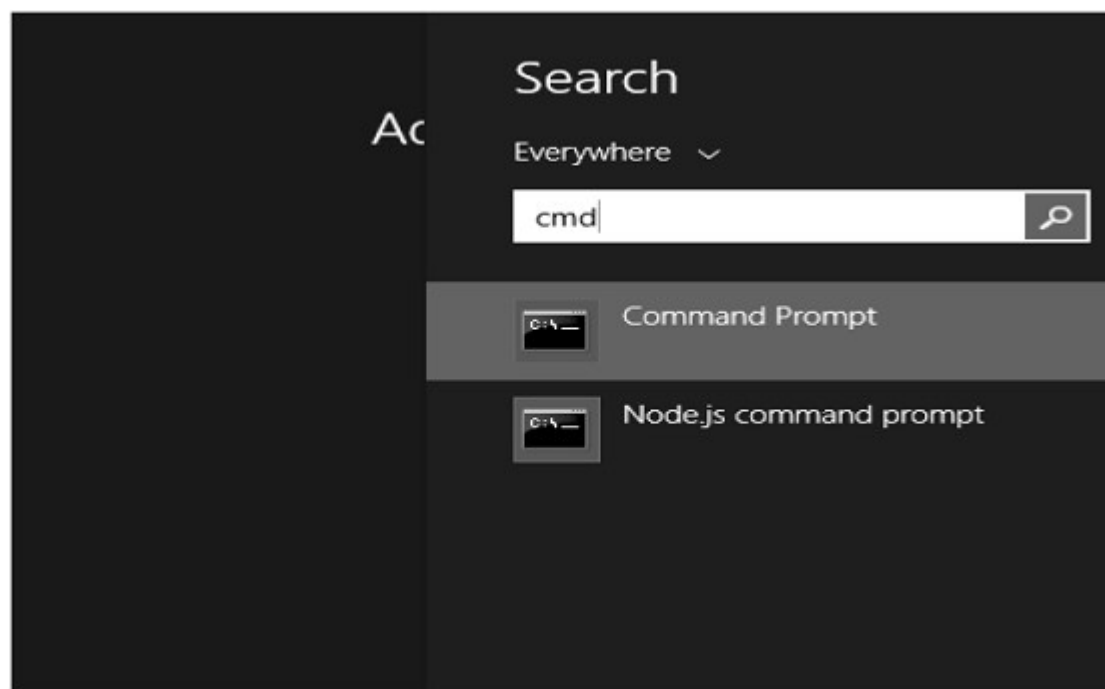
In genere, per creare un file batch, viene utilizzato il blocchetto per appunti. Questo è lo strumento più semplice per la creazione di file batch. La prossima è l'ambiente di esecuzione per gli script batch. Sui sistemi Windows, questo viene fatto tramite il prompt dei comandi o cmd.exe. Tutti i file batch vengono eseguiti in questo ambiente.

Di seguito sono riportati i diversi modi per avviare cmd.exe -

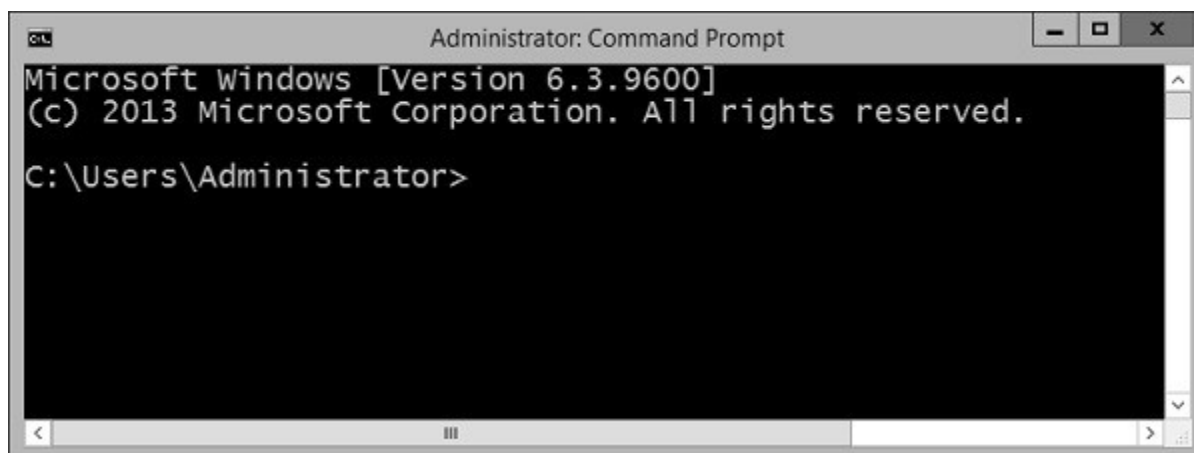
**Metodo 1 - Vai alla C: \ Windows \ System32 e fare doppio clic sul file cmd.**



**Metodo 2 - Tramite il comando di marcia - La seguente fotografia mostra per trovare il prompt dei comandi (cmd.exe) sul server Windows 2012.**



Una volta che il cmd.exe viene lanciato, vi si presenterà con la seguente schermata. Questo sarà il vostro ambiente per l'esecuzione di vostri script batch.



## variabili ambientali

Al fine di eseguire i file batch dal prompt dei comandi, vi sia bisogno di andare alla posizione a cui è memorizzato il file batch o in alternativa si può inserire il percorso del file nella variabile d'ambiente PATH. Così partendo dal presupposto che il file batch viene memorizzato nel percorso `C:\Application\bin`, si avrebbe bisogno di seguire queste istruzioni per il percorso di inclusione variabile.

OS	Produzione
finestre	Aggiungere la stringa; <code>C:\Application\bin</code> alla fine del sistema PATH variabile.

## Script batch - Comandi

In questo capitolo, vedremo alcuni dei comandi batch di uso frequente.

S.No	Comandi e descrizione
1	VER (batch_script_ver.html) Questo comando batch mostra la versione di MS-DOS che si sta utilizzando.
2	ASSOC (batch_script_assoc.html) Si tratta di un comando batch che associa un interno con un tipo di file (FTYPE) associazioni, esposizioni esistenti, o elimina un'associazione.
3	CD (batch_script_cd.html) Questo comando batch aiuta a fare le modifiche in una directory diversa, o visualizza la directory corrente.
4	CLS (batch_script_cls.html) Questo comando batch pulisce lo schermo.
5	COPIA (batch_script_copy.html) Questo comando automatico è utilizzato per copiare file da una posizione all'altra.
6	DEL (batch_script_del.html) Questo comando batch elimina i file e le directory non.

7	DIR (batch_script_dir.html) Questo comando batch elenca il contenuto di una directory.
8	DATA (batch_script_date.html) Questo comando help batch per trovare la data di sistema.
9	ECO (batch_script_echo.html) Questo comando batch visualizza i messaggi, o si comando eco acceso o spento.
10	USCITA (batch_script_exit.html) Questo comando batch esce dalla console DOS.
11	MD (batch_script_md.html) Questo comando batch crea una nuova directory nella posizione corrente.
12	MOSSA (batch_script_move.html) Questo comando batch sposta file o directory tra directory.
13	IL PERCORSO (batch_script_path.html) Questo lotto comando visualizza o imposta la variabile del percorso.
14	PAUSA (batch_script_pause.html) Questo comando batch richiede all'utente e attende una riga di input da inserire.
15	RICHIESTA (batch_script_prompt.html) Questo comando batch può essere utilizzato per modificare o ripristinare il prompt cmd.exe.
16	RD (batch_script_rd.html) Questo comando batch elimina le directory, ma le directory devono essere vuote prima che possano essere rimossi.
17	REN (batch_script_ren.html) Rinomina file e directory
18	REM (batch_script_rem.html) Questo comando batch viene utilizzato per osservazioni in file batch, impedendo il contenuto della osservazione venga eseguita.

19	INIZIO (batch_script_start.html) Questo comando batch avvia un programma in una nuova finestra, o si apre un documento.
20	TEMPO (batch_script_time.html) Questo lotto set di comandi o visualizza l'ora.
21	DIGITARE (batch_script_type.html) Questo comando batch stampa il contenuto di uno o più file per l'output.
22	VOL (batch_script_vol.html) Questo comando batch visualizza le etichette di volume.
23	ATTRIB (batch_script_attrib.html) Visualizza o imposta gli attributi dei file nella directory curret
24	CHKDSK (batch_script_chkdsk.html) Questo comando batch controlla il disco per eventuali problemi.
25	SCELTA (batch_script_choice.html) Questo comando batch fornisce un elenco di opzioni per l'utente.
26	CMD (batch_script_cmd.html) Questo comando batch richiama un'altra istanza di prompt dei comandi.
27	COMP (batch_script_comp.html) Questo comando batch confronta 2 file in base alla dimensione del file.
28	CONVERTIRE (batch_script_convert.html) Questo comando batch converte un volume da FAT16 o file system FAT32 al file system NTFS.
29	DRIVERQUERY (batch_script_driverquery.html) Questo comando batch mostra tutti i driver dei dispositivi installati e le loro proprietà.
30	ESPANDERE (batch_script_expand.html) Questo comando batch estrae i file da compressi file CAB CAB.

31	<b>TROVA</b> (batch_script_find.html) Questo comando ricerca in batch per una stringa in file o di input, output di linee corrispondenti.
32	<b>FORMATO</b> (batch_script_format.html) Questo comando batch formatta un disco da usare file di sistema di Windows, supportato, ad esempio FAT, FAT32 o NTFS, sovrascrivendo così il precedente contenuto del disco.
33	<b>AIUTO</b> (batch_script_help.html) Questo comando batch mostra l'elenco dei comandi di Windows-forniti.
34	<b>IPCONFIG</b> (batch_script_ipconfig.html) Questo comando batch mostra Configurazione IP di Windows. Mostra configurazione connessione e il nome di tale connessione.
35	<b>ETICHETTA</b> (batch_script_label.html) Questo comando batch aggiunge, imposta o rimuove un'etichetta del disco.
36	<b>DI PIÙ</b> (batch_script_more.html) Questo comando batch visualizza il contenuto di un file o di file, uno schermo alla volta.
37	<b>NETTO</b> (batch_script_net.html) Fornisce vari servizi di rete, a seconda del comando utilizzato.
38	<b>PING</b> (batch_script_ping.html) Questo comando batch invia ICMP / IP pacchetti "Echo" in rete per l'indirizzo indicato.
39	<b>FERMARE</b> (batch_script_shutdown.html) Questo comando batch spegne un computer, o si disconnette l'utente corrente.
40	<b>ORDINARE</b> (batch_script_sort.html) Questo comando batch prende l'input da un file di origine e ordina il suo contenuto in ordine alfabetico, dalla A alla Z o dalla Z alla A. viene stampato l'output sulla console.
41	<b>SUBST</b> (batch_script_subst.html) Questo comando batch assegna una lettera di unità in una cartella locale, mostra le assegnazioni attuali, o rimuove un incarico.

# Script batch - i file

In questo capitolo, impareremo come creare, salvare, eseguire e modificare i file batch.

## Creazione di file batch

I file batch sono normalmente creati in blocco note. Quindi il modo più semplice è quello di aprire il Blocco note e inserire i comandi necessari per lo script. Per questo esercizio, aprire il blocco note e inserire le seguenti dichiarazioni.

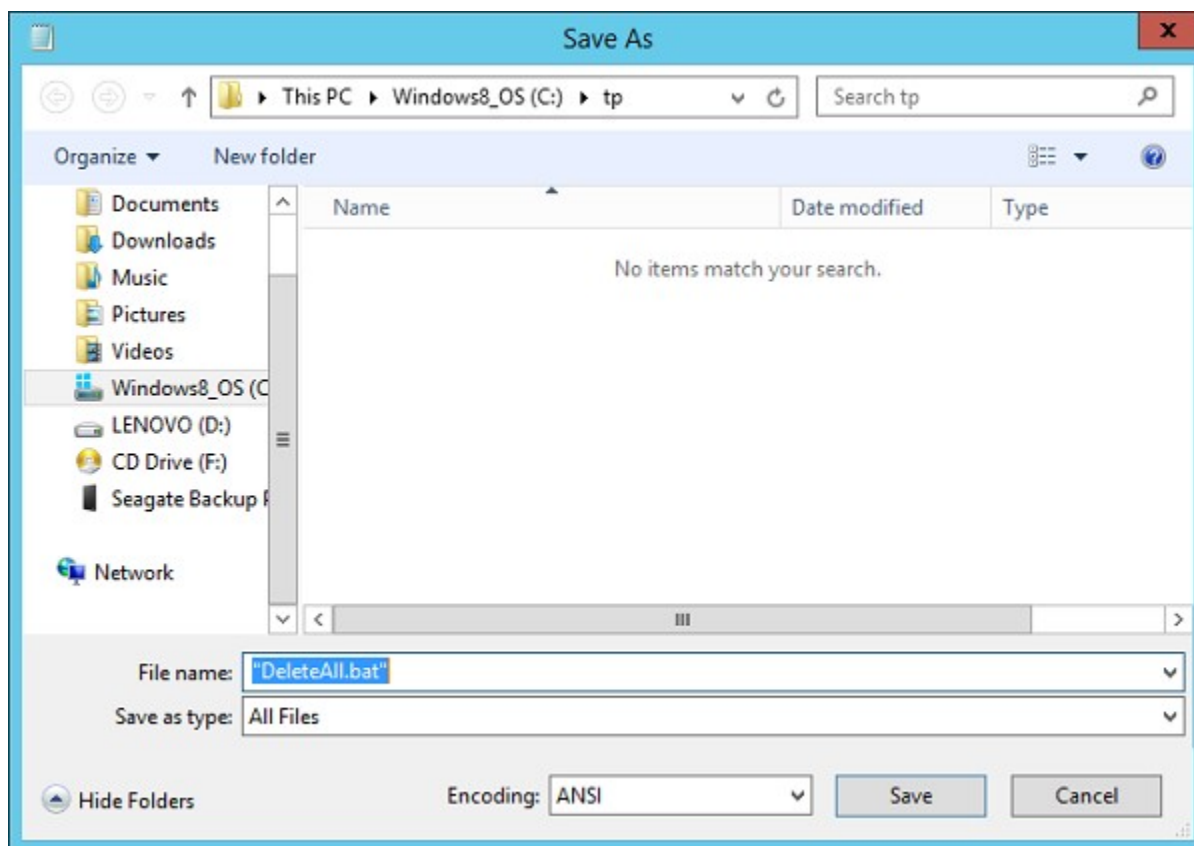
```
:: Deletes All files in the Current Directory With Prompts and Warnings  
::(Hidden, System, and Read-Only Files are Not Affected)  
::  
@ECHO OFF  
DEL .  
DR
```

## Il salvataggio dei file batch

Una volta creato il file batch, il passo successivo è quello di salvare il file batch. I file batch hanno l'estensione di una bat o .cmd. Alcune regole generali da tenere a mente quando si nominano i file batch -

- Cercate di evitare gli spazi per la denominazione file batch, si crea a volte problemi quando vengono chiamati da altri script.
- Non li nome dopo file batch comuni che sono disponibili nel sistema, come ping.cmd.





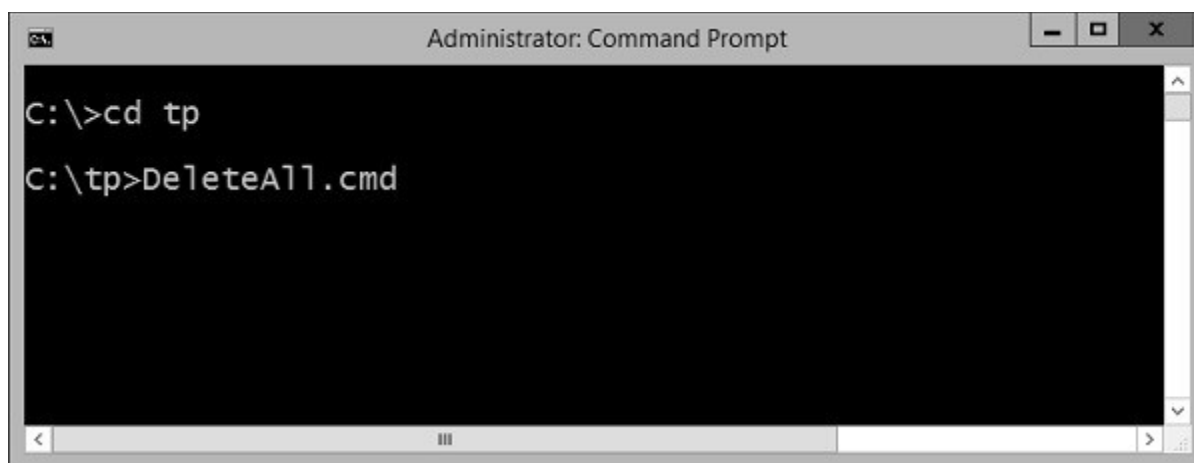
L'immagine qui sopra mostra come salvare il file batch. Quando si salva il file batch alcuni punti da tenere a mente.

- Ricordate di mettere il .bat o .cmd alla fine del nome del file.
- Scegliere l'opzione "Salva come" come "Tutti i file".
- Mettere l'intero nome di file tra virgolette "".

## File di Esecuzione batch

Di seguito sono riportati i passaggi per eseguire un file batch -

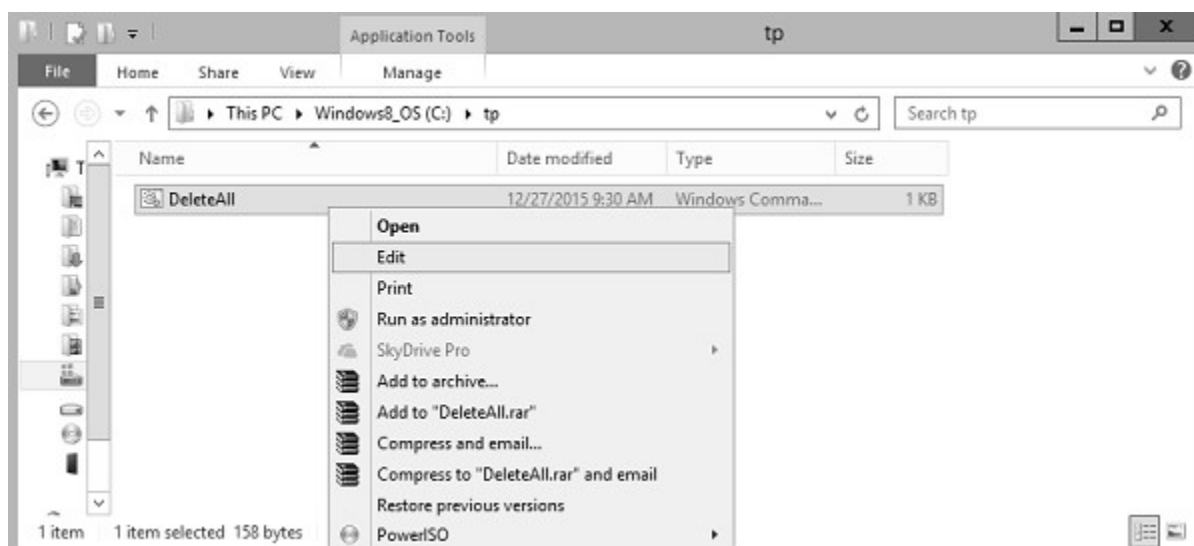
- **Fase 1 - Aprire il prompt dei comandi (cmd.exe).**
- **Fase 2 - Vai alla posizione in cui è memorizzato il file bat o .cmd.**
- **Fase 3 - Scrivere il nome del file, come mostrato nell'immagine seguente e premere il tasto Invio per eseguire il file batch.**



## I file batch Modifica

Di seguito sono riportati i passaggi per la modifica di un file batch esistente.

- **Fase 1 - Aprire Esplora risorse.**
- **Fase 2 - Vai alla posizione in cui è memorizzato il file bat o .cmd.**
- **Fase 3 - Fare clic destro sul file e scegliere l'opzione "Modifica" dal menu contestuale.** Il file si apre in Blocco note per ulteriori modifiche.



## Script batch - Sintassi

Normalmente, la prima linea in un file batch, spesso costituito da il seguente comando.

### ECHO comando

```
@echo off
```

Per impostazione predefinita, un file batch verrà visualizzato il suo comando durante l'esecuzione. Lo scopo di questo primo comando è disattivare questo display. Il comando "echo off" spegne il display per l'intero script, ad eccezione del comando "echo off" stesso. La "a" segno "& commat;" davanti rende il comando applica a se stesso pure.

## Documentazione

Molto spesso i file batch contiene anche le linee che iniziano con il comando "Rem". Questo è un modo di inserire commenti e documentazione. Il computer ignora qualsiasi cosa su una linea che segue Rem. Per i file batch con l'aumentare della quantità di complessità, questo è spesso una buona idea avere commenti.

## Prima Batch programma script

Costruiamo il nostro semplice programma script primo lotto. Aprire il Blocco note e digitare le seguenti righe di codice. Salvare il file come "List.cmd".

Il codice esegue le seguenti -

- Utilizza l'eco fuori comando per accertarsi che i comandi non vengono visualizzati quando viene eseguito il codice.
- Il comando Rem viene utilizzato per aggiungere un commento per dire che cosa esattamente questo file batch fa.
- Il comando dir viene usato per prendere il contenuto della posizione C: \ Program Files.
- Il comando '>' viene utilizzata per reindirizzare l'output al file C: \ lists.txt.
- Infine, il comando echo è usato per indicare all'utente che l'operazione è completata.

```
@echo off
Rem This is for listing down all the files in the directory Program files
dir "C:\Program Files" > C:\lists.txt
echo "The program has completed"
```

Quando viene eseguito il comando precedente, i nomi dei file in C: \ Program Files verranno inviati al file C: \ Lists.txt e nel Prompt dei comandi il messaggio "ha completato il programma" sarà visualizzato.

# Script batch - Variabili

Ci sono due tipi di variabili in file batch. Uno è per i parametri che possono essere passati quando il file batch viene chiamato e l'altro viene fatto tramite il comando set.

## Argomenti della riga di comando

script batch supportano il concetto di argomenti della riga di comando in cui gli argomenti possono essere passati al file batch quando viene richiamato. Gli argomenti possono essere chiamati dai file batch attraverso le variabili % 1,% 2,% 3, e così via.

L'esempio seguente mostra un file batch che accetta 3 argomenti della riga di comando e l'eco di loro è alla schermata riga di comando.

```
@echo off  
echo %1  
echo %2  
echo %3
```

Se lo script batch precedente viene memorizzato in un file chiamato test.bat e siamo stati per eseguire il batch come

```
Test.bat 1 2 3
```

Di seguito è uno screenshot di come questo possa guardare nel prompt dei comandi quando viene eseguito il file batch.



Il comando precedente produce il seguente output.

```
1  
2  
3
```

Se dovessimo eseguire il batch come

```
Example 1 2 3 4
```

L'uscita rimarrebbe lo stesso come sopra. Tuttavia, il quarto parametro viene ignorato.

## set di comandi

L'altro modo in cui le variabili possono essere inizializzate è tramite il comando 'set'. In seguito è la sintassi del comando set.

### Sintassi

```
set /A variable-name = value
```

dove,

- **-nome della variabile è il nome della variabile che si desidera impostare.**
- **il valore è il valore che deve essere impostato contro la variabile.**
- **/ A - Questo interruttore viene utilizzato se il valore deve essere numerico in natura.**

L'esempio seguente mostra un modo semplice il comando set può essere utilizzato.

### Esempio

```
@echo off  
set message = Hello World  
echo %message%
```

- Nel frammento di codice di cui sopra, una variabile chiamata messaggio è definita e impostata con il valore di "Ciao Mondo".
- Per visualizzare il valore della variabile, si noti che la variabile deve essere racchiuso nel segno%.

## Produzione

Il comando precedente produce il seguente output.

```
Hello World
```

## Lavorare con valori numerici

In script batch, è anche possibile definire una variabile per contenere un valore numerico. Questo può essere fatto utilizzando l'interruttore A /.

Il codice seguente mostra un modo semplice in cui i valori numerici possono essere impostati con l'opzione / a.

```
@echo off
SET /A a = 5
SET /A b = 10
SET /A c = %a% + %b%
echo %c%
```

- Stiamo prima impostazione del valore di 2 variabili, A e B a 5 e 10, rispettivamente.
- Stiamo aggiungendo quei valori e memorizzare nella variabile c.
- Infine, mostriamo il valore della variabile c.

L'uscita di questo programma sarebbe 15.

Tutti gli operatori aritmetici funzionano nei file batch. L'esempio seguente mostra operatori aritmetici possono essere utilizzati nei file batch.

```
@echo off
SET /A a = 5
SET /A b = 10
SET /A c = %a% + %b%
echo %c%
SET /A c = %a% - %b%
echo %c%
SET /A c = %b% / %a%
echo %c%
SET /A c = %b% * %a%
echo %c%
```

Il comando precedente produce il seguente output.

```
15  
-5  
2  
20
```

## Local vs variabili globali

In qualsiasi linguaggio di programmazione, vi è la possibilità di contrassegnare le variabili come aventi una sorta di applicazione, ossia la sezione di codice su cui possono accedere. Normalmente, variabile con una portata globale si può accedere ovunque da un programma mentre le variabili con ambito locali hanno un confine definito in cui è possibile accedere.

scripting DOS ha anche una definizione per le variabili livello locale e globale con ambito. Per impostazione predefinita, le variabili sono globali per l'intera sessione di prompt dei comandi. Chiama il comando SETLOCAL per rendere variabili locali alla portata del vostro script. Dopo aver chiamato SETLOCAL, qualsiasi assegnazione di variabile ritornano su chiamando ENDLOCAL, chiamando EXIT, o quando l'esecuzione raggiunge la fine del file (EOF) nello script. L'esempio seguente mostra la differenza quando le variabili locali e globali sono impostati nello script.

### Esempio

```
@echo off  
set globalvar = 5  
SETLOCAL  
set var = 13145  
set /A var = %var% + 5  
ENDLOCAL  
echo %var%  
echo %globalvar%
```

Poche cose fondamentali da notare sul programma di cui sopra.

- Il 'globalvar' è definito con una portata globale ed è disponibile in tutto l'intero script.
- La variabile 'var' è definito in un ambito locale, in quanto è racchiuso tra un blocco e 'SETLOCAL' 'ENDLOCAL'. Quindi, questa variabile sarà distrutto non appena l'affermazione 'ENDLOCAL' viene eseguito.

## Produzione

Il comando precedente produce il seguente output.

```
13150  
5
```

Si noterà che il comando `echo% var%` non produrrà nulla, perché dopo l'istruzione `ENDLOCAL`, la variabile 'var' non esisterà più.

## Lavorare con le variabili d'ambiente

Se si dispone di variabili che sarebbero stati utilizzati in tutto i file batch, quindi è sempre preferibile utilizzare le variabili di ambiente. Una volta che la variabile d'ambiente è definita, si può accedere tramite il segno%. L'esempio seguente mostra come vedere la `JAVA_HOME` definita su un sistema. La variabile `JAVA_HOME` è un componente chiave che viene normalmente utilizzato da una vasta gamma di applicazioni.

```
@echo off  
echo %JAVA_HOME%
```

L'uscita dovrebbe mostrare la directory `JAVA_HOME` che dipenderebbe da sistema a sistema. È un esempio di una uscita.

```
C:\Atlassian\Bitbucket\4.0.1\jre
```

## Script batch - Commenti

È sempre una buona pratica per aggiungere commenti o documentazione per gli script che vengono creati. Questo è necessario per la manutenzione degli script per capire cosa realmente fa lo script.

Ad esempio, si consideri il seguente frammento di codice che non ha forma di commenti. Se una persona media che non ha sviluppato il seguente script cerca di capire lo script, ci vorrebbe un sacco di tempo per quella persona per capire cosa realmente fa lo script.



```
ECHO OFF
IF NOT "%OS%"=="Windows_NT" GOTO Syntax
ECHO.%* | FIND "?" >NUL
IF NOT ERRORLEVEL 1 GOTO Syntax
IF NOT [%2]==[] GOTO Syntax
SETLOCAL
SET WSS=
IF NOT [%1]==[] FOR /F "tokens = 1 delims = \ " %%A IN ('ECHO.%~1') DO SET W
FOR /F "tokens = 1 delims = \ " %%a IN ('NET VIEW ^| FIND /I "\\%WSS%") DO
"tokens = 1 delims = " %%A IN ('NBTSTAT -a %%a ^| FIND /I /V "%a" ^| FIND "
DO ECHO.%%a %%A
ENDLOCAL
GOTO:EOF
ECHO Display logged on users and their workstations.
ECHO Usage: ACTUSR [ filter ]
IF "%OS%"=="Windows_NT" ECHO Where: filter is the first part
of the computer name^(s^) to be displayed
```

## Commenti utilizzando l'istruzione Rem

Ci sono due modi per creare commenti in script batch; uno è tramite il comando REM. Qualsiasi testo che segue l'istruzione Rem sarà trattata come commenti e non verrà eseguito. In seguito è la sintassi generale di questa affermazione.

### Sintassi

```
Rem Remarks
```

dove «Osservazioni» sono le osservazioni che deve essere aggiunto.

L'esempio seguente mostra un modo semplice il comando **REM può essere utilizzato**.

### Esempio

```
@echo off
Rem This program just displays Hello World
set message = Hello World
echo %message%
```

## Produzione

Il comando precedente produce il seguente output. Si noter  che non verr  eseguito il linea con la dichiarazione Rem.

```
Hello World
```

## Commenti utilizzando il :: Statement

L'altro modo per creare i commenti di script batch   tramite il comando ::. Qualsiasi testo che segue l':: dichiarazione sar  trattata come commenti e non verr  eseguito. In seguito   la sintassi generale di questa affermazione.

### Sintassi

```
:: Remarks
```

dove «Osservazioni»   il commento che deve essere aggiunto.

L'esempio seguente mostra un modo semplice il comando REM pu  essere utilizzato.

### Esempio

```
@echo off  
:: This program just displays Hello World  
set message = Hello World  
echo %message%
```

## Produzione

Il comando precedente produce il seguente output. Si noter  che non verr  eseguita la linea con la dichiarazione ::.

```
Hello World
```

**Nota - Se si dispone di un numero eccessivo di righe di Rem, potrebbe rallentare il codice, perch  alla fine di ogni riga di codice nel file batch deve ancora essere eseguite.**

Vediamo l'esempio della grande sceneggiatura che abbiamo visto all'inizio di questo argomento e vedere come appare quando la documentazione viene aggiunto ad esso.

```
::=====
:: The below example is used to find computer and logged on users
::
::=====
ECHO OFF
:: Windows version check
IF NOT "%OS%"=="Windows_NT" GOTO Syntax
ECHO.%* | FIND "?" >NUL
:: Command line parameter check
IF NOT ERRORLEVEL 1 GOTO Syntax
IF NOT [%2]==[] GOTO Syntax
:: Keep variable local
SETLOCAL
:: Initialize variable
SET WSS=
:: Parse command line parameter
IF NOT [%1]==[] FOR /F "tokens = 1 delims = \ " %%A IN ('ECHO.%~1') DO SET W
:: Use NET VIEW and NBTSTAT to find computers and logged on users
FOR /F "tokens = 1 delims = \ " %%a IN ('NET VIEW ^| FIND /I "\\%WSS%") DO
"tokens = 1 delims = " %%A IN ('NBTSTAT -a %%a ^| FIND /I /V "%a" ^| FIND
"<03>"') DO ECHO.%%a %%A
:: Done
ENDLOCAL
GOTO:EOF
:Syntax
ECHO Display logged on users and their workstations.
ECHO Usage: ACTUSR [ filter ]
IF "%OS%"=="Windows_NT" ECHO Where: filter is the first part of the
computer name^(s^) to be displayed
```

Ora è possibile vedere che il codice è diventato più comprensibile per gli utenti che non hanno sviluppato il codice e quindi è più gestibile.

## Script batch - Strings

In DOS, una stringa è un insieme ordinato di caratteri, come "Ciao, mondo!".

S.No	Stringhe e descrizione
1	creare String (batch_script_create_string.html) Una stringa può essere creato in DOS nel modo seguente.

# Script batch - Array

Gli array non sono specificamente definito come un tipo di script batch, ma possono essere implementate. Le seguenti cose devono essere notato quando gli array siano implementati in batch Script.

- Ciascun elemento della matrice deve essere definita con il comando set.
- L' 'per' ciclo sarebbe necessario per scorrere i valori della matrice.

## Creazione di un array

Un array viene creato utilizzando il seguente comando set.

```
set a[0] = 1
```

Dove 0 è l'indice della matrice e 1 è il valore assegnato al primo elemento della matrice.

Un altro modo per implementare array consiste nel definire una lista di valori e scorrere l'elenco di valori. L'esempio seguente mostra come questo può essere implementato.

## Esempio

```
@echo off
set list = 1 2 3 4
(for %%a in (%list%) do (
    echo %%a
))
```

## Produzione

Il comando precedente produce il seguente output.

```
1
2
3
4
```

## Accesso Array

È possibile recuperare un valore dalla matrice utilizzando la sintassi pedice, passando l'indice del valore che si desidera recuperare tra parentesi quadre subito dopo il nome della matrice.

## Esempio

```
@echo off
set a[0] = 1
echo %a[0]%
```

In questo esempio, l'indice inizia da 0 che significa che il primo elemento può essere raggiunto usando come indice 0, il secondo elemento può essere raggiunto usando come indice 1 e così via. Controlliamo il seguente esempio per creare, inizializzare e accesso array -

```
@echo off
set a[0] = 1
set a[1] = 2
set a[2] = 3
echo The first element of the array is %a[0]%
echo The second element of the array is %a[1]%
echo The third element of the array is %a[2]%
```

Il comando precedente produce il seguente output.

```
The first element of the array is 1
The second element of the array is 2
The third element of the array is 3
```

## Modifica di un array

Per aggiungere un elemento alla fine dell'array, è possibile utilizzare l'elemento set insieme con l'ultimo indice dell'elemento dell'array.

## Esempio

```
@echo off
set a[0] = 1
set a[1] = 2
set a[2] = 3
Rem Adding an element at the end of an array
Set a[3] = 4
echo The last element of the array is %a[3]%
```

Il comando precedente produce il seguente output.

```
The last element of the array is 4
```

È possibile modificare un elemento esistente di un array assegnando un nuovo valore ad un determinato indice, come mostrato nel seguente esempio -

```
@echo off
set a[0] = 1
set a[1] = 2
set a[2] = 3
Rem Setting the new value for the second element of the array
Set a[1] = 5
echo The new value of the second element of the array is %a[1]%
```

Il comando precedente produce il seguente output.

```
The new value of the second element of the array is 5
```

## L'iterazione di un array

Iterazione di un array è ottenuta utilizzando la 'per' anello e passando attraverso ogni elemento della matrice. L'esempio seguente mostra un modo semplice che una matrice può essere implementata.

```
@echo off
setlocal enabledelayedexpansion
set topic[0] = comments
set topic[1] = variables
set topic[2] = Arrays
set topic[3] = Decision making
set topic[4] = Time and date
set topic[5] = Operators

for /l %%n in (0,1,5) do (
    echo !topic[%%n]!
)
```

A seguito di cose devono essere notato sul programma di cui sopra -

- Ciascun elemento della matrice deve essere specificamente definita utilizzando il comando set.
- L' 'per' ciclo con il parametro / L per muoversi attraverso le gamme viene utilizzato per scorrere la matrice.

## Produzione

Il comando precedente produce il seguente output.

```
Comments
variables
Arrays
Decision making
Time and date
Operators
```

## Lunghezza di un array

La lunghezza di un array avviene iterando sulla lista di valori nella matrice poiché non vi è alcuna funzione diretta per determinare il numero di elementi in una matrice.

```
@echo off
set Arr[0] = 1
set Arr[1] = 2
set Arr[2] = 3
set Arr[3] = 4
set "x = 0"
:SymLoop

if defined Arr[%x%] (
    call echo %%Arr[%x%]%%
    set /a "x+=1"
    GOTO :SymLoop
)
echo "The length of the array is" %x%
```

## Produzione

Uscita Il comando precedente produce il seguente output.

```
The length of the array is 4
```

## Creazione di strutture in array

Le strutture possono essere attuate anche in file batch usando un po' di una codifica in più per l'attuazione. L'esempio seguente mostra come questo può essere realizzato.

### Esempio



```
@echo off
set len = 3
set obj[0].Name = Joe
set obj[0].ID = 1
set obj[1].Name = Mark
set obj[1].ID = 2
set obj[2].Name = Mohan
set obj[2].ID = 3
set i = 0
:loop

if %i% equ %len% goto :eof
set cur.Name =
set cur.ID =

for /f "usebackq delims ==. tokens = 1-3" %%j in (`set obj[%i%]`) do (
    set cur.%%k = %%l
)
echo Name = %cur.Name%
echo Value = %cur.ID%
set /a i = %i%+1
goto loop
```

Le seguenti cose fondamentali devono essere notato per il codice di cui sopra.

- Ogni variabile definita utilizzando il comando set ha 2 valori associati a ogni indice dell'array.
- **La variabile è impostata a 0 in modo che possiamo scorrere la struttura sarà la lunghezza della matrice, che è 3.**
- Abbiamo sempre controllare per la condizione che il valore di i è uguale al valore di **len e se non, ciclo attraverso il codice.**
- Siamo in grado di accedere a ciascun elemento della struttura utilizzando il obj [%i%] notazione.

## Produzione

Il comando precedente produce il seguente output.

```
Name = Joe
Value = 1
Name = Mark
Value = 2
Name = Mohan
Value = 3
```

## Script batch - Decision Making

Strutture decisionali richiedono che il programmatore di specificare una o più condizioni da valutare o verificate da programma, insieme con una o più istruzioni da eseguire se la condizione è determinata per essere **vero, e, facoltativamente, altre dichiarazioni da eseguire se il condizionale è determinata a essere falsa.**

S.No	Stringhe e descrizione
1	Se Statement (batch_script_if_statement.html) La prima istruzione decisionale è il 'se' dichiarazione.
2	Se / else (batch_script_if_else_statement.html) Il prossimo processo decisionale affermazione è il caso / else. In seguito è la forma generale di questa affermazione.
3	Istruzioni IF nidificate (batch_script_nested_if_statements.html) A volte, vi è una necessità di avere più 'se' istruzione incorporato dentro l'altro. In seguito è la forma generale di questa affermazione.

## Script batch - Operatori

Un operatore è un simbolo che dice al compilatore di eseguire specifiche manipolazioni matematiche o logiche.

In script batch, i seguenti tipi di operatori sono possibili.

- Gli operatori aritmetici
- operatori relazionali
- operatori logici
- operatori di assegnazione
- operatori bit per bit

## Operatori aritmetici

linguaggio di script batch supporta i normali operatori aritmetici come qualsiasi lingua. Di seguito sono riportati gli operatori aritmetici disponibili.

Visualizza Esempio ([batch\\_script\\_arithmetic\\_operators.html](batch_script_arithmetic_operators.html))

Operatore	Descrizione	Esempio
&più;	Aggiunta di due operandi	1 & costi; 2 darà 3
-	Sottrae secondo operando dal primo	2 - 1 darà 1
*	Moltiplicazione di entrambi gli operandi	2 * 2 darà 4
/	Divisione del numeratore per il denominatore	3/2 darà 1.5
%	operatore modulo e la restante parte dopo un / divisione float integer	3% 2 darà 1

## Operatori relazionali

operatori relazionali consentono di confronto di oggetti. Di seguito sono riportati gli operatori relazionali disponibili.

Visualizza Esempio ([batch\\_script\\_relational\\_operators.html](batch_script_relational_operators.html))

Operatore	Descrizione	Esempio
<b>EQU</b>	Verifica l'uguaglianza tra due oggetti	2 EQU 2 darà vero
<b>NEQ</b>	Test la differenza tra due oggetti	3 NEQ 2 darà vero
<b>LSS</b>	Verifica se l'oggetto sinistra è minore l'operando di destra	2 LSS 3 darà vero
<b>LEQ</b>	Verifica se l'oggetto sinistra è minore o uguale al secondo operando	2 LEQ 3 darà vero
<b>GTR</b>	Verifica se l'oggetto sinistro è maggiore dell'operando destra	3 GTR 2 darà vero
<b>GEQ</b>	Verifica se l'oggetto sinistra è maggiore o uguale al secondo operando	3 GEQ 2 darà vero

# Operatori logici

Gli operatori logici vengono utilizzati per valutare le espressioni booleane. Di seguito sono riportati gli operatori logici disponibili.

Il linguaggio batch è dotato di una serie completa di operatori logici booleani come AND, OR, XOR, ma solo per i numeri binari. Né ci sono valori per VERO o FALSO. L'unico operatore logico disponibile per condizioni è l'operatore NOT.

Visualizza Esempio ([batch\\_script\\_logical\\_operators.html](batch_script_logical_operators.html))

Operatore	Descrizione
<b>E</b>	Questo è l'operatore logico "e"
<b>O</b>	Questo è l'operatore logico "o"
<b>NON</b>	Questo è l'operatore logico "non"

# Operatori di assegnazione

Batch linguaggio di script fornisce anche operatori di assegnazione. Di seguito sono riportati gli operatori di assegnazione disponibili.

Visualizza Esempio ([batch\\_script\\_assignment\\_operators.html](batch_script_assignment_operators.html))

Operatore	Descrizione	Esempio
<b>&amp; Costi; =</b>	Questo aggiunge operando di destra l'operando sinistro e assegna il risultato operando sinistro	Set / A a = 5 un & costi; 3 = Uscita sarà 8
<b>- =</b>	Questo sottrae l'operando di destra dalla operando a sinistra e assegna il risultato l'operando sinistro	Set / A a = 5 a - = 3 Uscita sarà 2
<b>* =</b>	Questo moltiplica l'operando di destra con l'operando di sinistra e assegna il risultato l'operando sinistro	Set / A a = 5 a * = 3 Uscita sarà 15

/ =	Questo divide l'operando sinistro con l'operando di destra e assegna il risultato l'operando sinistro	Set / A a = 6 a / = 3 Uscita sarà 2
% =	Questo richiede modulo con due operandi e assegna il risultato l'operando sinistro	Set / A a = 5 un% = 3 Uscita sarà 2

## operatori bit per bit

operatori bit per bit sono possibili anche in script batch. Di seguito sono riportati gli operatori disponibili.

Visualizza Esempio ([batch\\_script\\_bitwise\\_operators.html](http://www.w3ii.com/it/batch_script/batch_script_bitwise_operators.html))

Operatore	Descrizione
&	Questo è il bit "e" operatore
	Questo è il bit "o" operatore
^	Questo è il bit "xor" o esclusivi o gestore

In seguito è la tabella di verità in mostra questi operatori.

p	q	p & q	p   q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

## Script batch - data e ora

La data e l'ora in DOS Scripting sono disponibili due comandi di base per recuperare la data e l'ora del sistema.

# DATA

Questo comando ottiene la data di sistema.

## Sintassi

```
DATE
```

## Esempio

```
@echo off  
echo %DATE%
```

## Produzione

La data corrente viene visualizzata nel prompt dei comandi. Per esempio,

```
Mon 12/28/2015
```

# TEMPO

Questo comando imposta o visualizza il tempo.

## Sintassi

```
TIME
```

## Esempio

```
@echo off  
echo %TIME%
```

## Produzione

verrà visualizzata l'ora attuale del sistema. Per esempio,

```
22:06:52.87
```

Di seguito sono alcune implementazioni che possono essere utilizzati per ottenere la data e l'ora in diversi formati.

## Data in formato anno-mese-giorno

### Esempio

```
@echo off
echo/Today is: %year%-%month%-%day%
goto :EOF
setlocal ENABLEEXTENSIONS
set t = 2&if "%date%z" LSS "A" set t = 1

for /f "skip = 1 tokens = 2-4 delims = (-)" %%a in ('echo/^|date') do (
    for /f "tokens = %t%-4 delims = .-/ " %%d in ('date/t') do (
        set %%a = %%d&set %%b = %%e&set %%c = %%f))
endlocal&set %1 = %yy%&set %2 = %mm%&set %3 = %dd%&goto :EOF
```

### Produzione

Il comando precedente produce il seguente output.

```
Today is: 2015-12-30
```

## Batch Script - Input / Output

Ci sono tre "file" universali per l'input da tastiera, la stampa di testo sugli errori di schermo e di stampa sullo schermo. Il file "Standard In", noto come **stdin**, **contiene l'input per il programma / script**. Il file "standard Out", noto come **stdout**, **viene utilizzato per scrivere l'uscita per la visualizzazione sullo schermo**. Infine, il file "standard Err", noto come **stderr**, **contiene eventuali messaggi di errore per la visualizzazione sullo schermo**.

Ciascuno di questi tre file standard, altrimenti noto come i flussi standard, vengono indicati con i numeri 0, 1, e 2 Stdin è file 0, stdout è il file 1, e stderr è il file 2.

## Il reindirizzamento di output (stdout e stderr)

Una pratica comune nei file batch sta mandando l'output di un programma per un file di

log. Il > operatore invia, o reindirizza, stdout o stderr in un altro file. L'esempio seguente mostra come questo può essere fatto.

```
Dir C:\ > list.txt
```

Nell'esempio precedente, la **stdout del comando Dir C: \ viene reindirizzato al file list.txt.**

Se si aggiunge il numero 2 al filtro di reindirizzamento, allora sarebbe reindirizzare il **stderr al file lists.txt.**

```
Dir C:\ 2> list.txt
```

Si può anche combinare i flussi di **stdout e stderr utilizzando il numero di file e la 'e' prefisso.** Di seguito è riportato un esempio.

```
DIR C:\ > lists.txt 2>&1
```

## Soppressione di uscita Programma

Il file pseudo NUL viene utilizzato per eliminare qualsiasi uscita da un programma. Il seguente esempio mostra che l'uscita del comando DIR scarta inviare l'uscita a NUL.

```
Dir C:\ > NUL
```

## stdin

Per lavorare con stdin, è necessario utilizzare una soluzione per raggiungere questo obiettivo. Questo può essere fatto reindirizzando proprio stdin prompt del comando, chiamato CON.

L'esempio seguente mostra come è possibile reindirizzare l'output in un file chiamato lists.txt. Dopo aver eseguito il seguente comando, il prompt dei comandi avrà tutti gli input inseriti dall'utente fino a che non riceve un carattere EOF. Successivamente, invia tutti gli input al file lists.txt.

```
TYPE CON > lists.txt
```



# Script batch - Codice di ritorno

Per impostazione predefinita, quando l'esecuzione della riga di comando è stata completata o esso deve restituire zero quando l'esecuzione ha esito positivo o diverso da zero quando l'esecuzione fallisce. Quando uno script batch restituisce un valore diverso da zero dopo l'esecuzione fallisce, il valore non-zero indica qual è il numero di errore. Ci sarà quindi utilizzare il numero di errore per determinare ciò che l'errore è di circa e risolvere di conseguenza.

Di seguito sono riportati il codice di uscita comune e la loro descrizione.

S.No.	Codice di errore e descrizione
1	<b>0</b> Programma completato con successo.
2	<b>1</b> Funzione non corretta. Indica che l'azione ha tentato di eseguire il comando non riconosciuta in Windows prompt dei comandi cmd.exe.
3	<b>2</b> Il sistema non trova il file specificato. Indica che il file non può essere trovato in posizione specificata.
4	<b>3</b> Il sistema non può trovare il percorso specificato. Indica che il percorso specificato non può essere trovato.
5	<b>5</b> L'accesso è negato. Indica che l'utente non ha alcun diritto di accesso alla risorsa specificata.
6	<b>9009</b> <b>0x2331</b> Programma non è riconosciuto come comando interno o esterno, un programma eseguibile o un file batch. Indica che il comando, il nome dell'applicazione o il percorso è stato scritto male durante la configurazione di azione.

7	<b>221225495</b> <b>0xC0000017</b> <b>-1.073,741801 millions</b> Non abbastanza memoria virtuale disponibile. Essa indica che Windows ha esaurito la memoria.
8	<b>3221225786</b> <b>0xC000013A</b> <b>-1.073,74151 milioni</b> L'applicazione risolto come conseguenza di un CTRL + C. Indica che l'applicazione è stata terminata da input da tastiera dell'utente CTRL + C o CTRL + Pausa o chiudere la finestra del prompt dei comandi.
9	<b>3221225794</b> <b>0xc0000142</b> <b>-1.073,741502 millions</b> L'applicazione non é stata inizializzata correttamente. Indica che l'applicazione è stata lanciata su un desktop a cui l'utente corrente non ha diritti di accesso. Un'altra possibile causa è che o gdi32.dll o user32.dll non è riuscita a inizializzare.

## Livello Errore

L'% ERRORLEVEL variabile d'ambiente% contiene il codice di ritorno dell'ultimo programma eseguito o uno script.

Per impostazione predefinita, il modo per controllare la ERRORLEVEL è tramite il codice seguente.

## Sintassi

```
IF %ERRORLEVEL% NEQ 0 (  
    DO_Something  
)
```

E 'comune usare il comando EXIT / B% ERRORLEVEL% alla fine del file batch per restituire

i codici di errore dal file batch.

EXIT / B alla fine del file batch interrompere l'esecuzione di un file batch.

Utilizzare EXIT / B <exitcodes> alla fine del file batch per tornare personalizzato tornare codici.

Ambiente variabile % ERRORLEVEL% contiene l'ultima ERRORLEVEL nel file batch, che è l'ultima codici di errore dell'ultimo comando eseguito. Nel file batch, è sempre una buona pratica di utilizzare variabili di ambiente invece di valori costanti, dal momento che la stessa variabile ottenere ampliato per valori diversi su diversi computer.

Diamo un'occhiata a un rapido esempio su come verificare la presenza di codici di errore da un file batch.

## Esempio

Supponiamo che abbiamo un file batch denominato Find.cmd che ha il seguente codice. Nel codice, abbiamo chiaramente detto che se non troviamo il file chiamato lists.txt allora dovremmo impostare il errorlevel a 7. Allo stesso modo, se vediamo che il userprofile variabile non è definita allora dovremmo impostare il codice di errorlevel 9.

```
if not exist c:\lists.txt exit 7
if not defined userprofile exit 9
exit 0
```

Supponiamo abbiamo un altro file chiamato App.cmd che chiama Find.cmd prima. Ora, se il Find.cmd restituisce un errore in cui si imposta il errorlevel per maggiore di 0 allora sarebbe uscire dal programma. Nel seguente file batch, dopo aver chiamato il Find.cnd trovare, in realtà controlla se l'errorlevel è maggiore di 0.

```
Call Find.cmd

if errorlevel gtr 0 exit
echo "Successful completion"
```

## Produzione

Nel programma di cui sopra, possiamo avere i seguenti scenari come l'uscita -

- Se il file C: \ lists.txt non esiste, allora nulla sarà visualizzata nell'output della console.
- Se la userprofile variabile non esiste, allora nulla sarà visualizzata nell'output della console.

- Se entrambe le condizioni sopra passa quindi la stringa "Il completamento" verrà visualizzato nel prompt dei comandi.

## Loops

Nel capitolo decisionale, abbiamo visto affermazioni che sono stati eseguiti uno dopo l'altro in un modo sequenziale. Inoltre, le implementazioni possono anche essere fatto in batch Script per modificare il flusso di controllo in logica di un programma. Essi vengono poi classificati in flusso di istruzioni di controllo.

S.No	Loops e descrizione
1	Mentre Dichiarazione Attuazione ( <a href="#">batch_script_while_statement_implementation.html</a> ) Non vi è alcuna dichiarazione, mentre diretta disponibile in script batch, ma possiamo fare un'implementazione di questo ciclo molto facilmente utilizzando l'istruzione if e le etichette.
2	Per Statement - Elenco Implementazioni ( <a href="#">batch_script_for_statement_implementations.html</a> ) Il "FOR" costruire offerte loop funzionalità per i file batch. Di seguito è riportato il costruito comune del 'per' dichiarazione per lavorare con un elenco di valori.
3	Scorrendo Ranges ( <a href="#">batch_script_looping_through_ranges.html</a> ) L' 'per' dichiarazione ha anche la capacità di muoversi attraverso un intervallo di valori. In seguito è la forma generale della dichiarazione.
4	Classic per l'implementazione Loop ( <a href="#">batch_script_classic_loop_implementationn.html</a> ) Di seguito è riportato il classico 'per' affermazione che è disponibile nella maggior parte dei linguaggi di programmazione.

## Looping attraverso Argomenti della riga di comando

L' 'per' istruzione può essere utilizzato anche per controllare gli argomenti della riga di comando. L'esempio seguente mostra come la 'per la' dichiarazione può essere utilizzato per scorrere gli argomenti della riga di comando.

### Esempio

```
@ECHO OFF
:Loop

IF "%1" == "" GOTO completed
FOR %%F IN (%1) DO echo %%F
SHIFT
GOTO Loop
:completed
```

## Produzione

Supponiamo che il nostro codice di cui sopra è memorizzato in un file chiamato Test.bat. Il comando precedente produce il seguente output se il file batch passa gli argomenti della riga di comando di 1,2 e 3 come Test.bat 1 2 3.

```
1
2
3
```

S.No	Loops e descrizione
1	Istruzione break Attuazione ( <a href="http://batch_script_break_statement_implementation.html">batch_script_break_statement_implementation.html</a> ) L'istruzione break viene utilizzato per modificare il flusso di controllo all'interno di un ciclo all'interno di qualsiasi linguaggio di programmazione. L'istruzione break è normalmente usato in costrutti di loop e viene utilizzato per provocare la cessazione immediata del ciclo più interno.

## Script batch - Funzioni

Una funzione è un insieme di istruzioni organizzati insieme per eseguire un compito specifico. In script batch, un approccio simile è adottato per le dichiarazioni logiche di gruppo insieme per formare una funzione.

Come come qualsiasi altra lingua, funzioni in batch script segue la stessa procedura -

- **Dichiarazione di funzione - Si dice al compilatore sul nome di una funzione, tipo di ritorno e parametri.**
- **Funzione Definizione - fornisce il corpo reale della funzione.**

# definizione di funzione

In script batch, una funzione è definita utilizzando l'istruzione etichetta. Quando una funzione è stato recentemente definito, potrebbero essere necessari uno o più valori come input 'parametri' alla funzione, processo le funzioni del corpo principale, e passare di nuovo i valori alle funzioni di uscita "tipi restituiti.

Ogni funzione ha un nome di funzione, che descrive il compito che la funzione esegue. Per utilizzare una funzione, è "chiamata" quella funzione con il suo nome e passa i suoi valori di input (noto come argomenti) che corrisponde ai tipi di parametri della funzione.

In seguito è la sintassi di una funzione semplice.

```
:function_name  
Do_something  
EXIT /B 0
```

- Il function\_name è il nome dato alla funzione che dovrebbe avere un qualche significato per corrispondere a ciò che realmente fa la funzione.
- L'istruzione EXIT viene utilizzato per garantire che la funzione esce correttamente.

È un esempio di una semplice funzione.

## Esempio

```
:Display  
SET /A index = 2  
echo The value of index is %index%  
EXIT /B 0
```

S.No	Funzioni e descrizione
1	Chiamata di una funzione (batch_script_calling_function.html) Una funzione viene chiamata in script batch utilizzando il comando di chiamata.
2	Funzioni con parametri (batch_script_functions_with_parameters.html) Le funzioni possono lavorare con i parametri semplicemente passando quando viene effettuata una chiamata alla funzione.
3	Funzioni con Valori restituiti (batch_script_functions_with_return_values.html) Le funzioni possono lavorare con i valori di ritorno semplicemente passando i nomi delle variabili

# Script batch - Processo

In questo capitolo, discuteremo i vari processi di batch Script.

## Visualizzazione della lista dei processi in esecuzione

In script batch, il comando TASKLIST può essere usato per ottenere l'elenco dei processi attualmente in esecuzione all'interno di un sistema.

### Sintassi

```
TASKLIST [/S system [/U username [/P [password]]]] [/M [module] | /SVC | /V]  
[/FO format] [/NH]
```

**In seguito sono la descrizione delle opzioni che possono essere presentati al comando TASKLIST.**

S.No.	Opzioni e descrizione
1.	<b>/ S sistema</b> Specifica il sistema remoto a cui connettersi
2.	<b>/ U</b> <b>[dominio \] utente</b> Specifica il contesto utente in cui eseguire il comando.
3.	<b>/ P [password]</b> Specifica la password per il contesto utente indicato. La password viene richiesta se omissa.
4.	<b>/ M [modulo]</b> Elenca tutte le attività che attualmente adottano il nome dato exe / dll. Se non viene specificato vengono visualizzati tutti i moduli caricati specificate il nome del modulo.
5.	<b>/ SVC</b> servizi Visualizza ospitati in ogni processo.
6.	<b>/ V</b> Consente di visualizzare informazioni sulle attività verbose.

7.	<b>/ Filtro FI</b> Consente di visualizzare un insieme di attività che corrispondono a un dato criterio specificato dal filtro.
8.	<b>Formato / FO</b> Specifica il formato di output. Valori validi: "tavolo", "LIST", "CSV".
9.	<b>/ NH</b> Specifica che il "Intestazione colonna" non deve mostrare in uscita. Valido solo per "TABLE" e formati "CSV".

## Esempi

TASKLIST

Il comando precedente ottenere l'elenco di tutti i processi in esecuzione sul sistema locale. In seguito è un'istantanea della produzione che viene resa quando il comando precedente viene eseguito così com'è. Come si può vedere dalla seguente output, non solo si ottiene i vari processi in esecuzione sul sistema, si ottiene anche l'utilizzo della memoria di ogni processo.



Image Name	PID	Session Name	Session#	Mem Usage
=====	=====	=====	=====	=====
System Idle Process	0	Services	0	4 K
System	4	Services	0	272 K
smss.exe	344	Services	0	1,040 K
csrss.exe	528	Services	0	3,892 K
csrss.exe	612	Console	1	41,788 K
wininit.exe	620	Services	0	3,528 K
winlogon.exe	648	Console	1	5,884 K
services.exe	712	Services	0	6,224 K
lsass.exe	720	Services	0	9,712 K
svchost.exe	788	Services	0	10,048 K
svchost.exe	832	Services	0	7,696 K
dwm.exe	916	Console	1	117,440 K
nvsvc.exe	932	Services	0	6,692 K
nvxdsync.exe	968	Console	1	16,328 K
nvsvc.exe	976	Console	1	12,756 K
svchost.exe	1012	Services	0	21,648 K
svchost.exe	236	Services	0	33,864 K
svchost.exe	480	Services	0	11,152 K
svchost.exe	1028	Services	0	11,104 K
svchost.exe	1048	Services	0	16,108 K
wlanext.exe	1220	Services	0	12,560 K
conhost.exe	1228	Services	0	2,588 K
svchost.exe	1276	Services	0	13,888 K
svchost.exe	1420	Services	0	13,488 K
spoolsv.exe	1556	Services	0	9,340 K

```
tasklist > process.txt
```

Il comando sopra prende l'output visualizzato da tasklist e lo salva nel file process.txt.

```
tasklist /fi "memusage gt 40000"
```

Il comando precedente recuperare solo quei processi la cui memoria è maggiore di 40 MB. Di seguito è riportato un esempio di output che può essere reso.

Image Name	PID	Session Name	Session#	Mem Usage
=====	=====	=====	=====	=====
dwm.exe	916	Console	1	127,912 K
explorer.exe	2904	Console	1	125,868 K
ServerManager.exe	1836	Console	1	59,796 K
WINWORD.EXE	2456	Console	1	144,504 K
chrome.exe	4892	Console	1	123,232 K
chrome.exe	4976	Console	1	69,412 K
chrome.exe	1724	Console	1	76,416 K
chrome.exe	3992	Console	1	56,156 K
chrome.exe	1168	Console	1	233,628 K
chrome.exe	816	Console	1	66,808 K

## Uccidere un particolare processo

Consente a un utente che esegue Microsoft Windows XP Professional, Windows 2003, o più tardi per uccidere un compito da una riga di comando di Windows dal processo ID (PID) o il nome dell'immagine. Il comando utilizzato per questo scopo è il comando TASKKILL.

### Sintassi

```
TASKKILL [/S system [/U username [/P [password]]]] { [/FI filter]
[/PID processid | /IM imagename] } [/T] [/F]
```

**In seguito sono la descrizione delle opzioni che possono essere presentati al comando TASKKILL.**

S.No.	Opzioni e descrizione
1.	<b>/ S sistema</b> Specifica il sistema remoto a cui connettersi
2.	<b>/ U</b> <b>[dominio \] utente</b> Specifica il contesto utente in cui eseguire il comando.
3.	<b>/ P [password]</b> Specifica la password per il contesto utente indicato. La password viene richiesta se omissa.

4.	<b>/ FI</b> <b>FilterName</b> Si applica un filtro per selezionare un insieme di compiti. Permette "*" per essere utilizzato. ex. nomeimmagine eq acme * Vedi sotto filtri Per ulteriori informazioni ed esempi.
5.	<b>/ PID</b> <b>idprocesso</b> Specifica il PID del processo da terminare. Utilizzare TaskList per ottenere il PID.
6.	<b>/SONO</b> <b>ImageName</b> Specifica il nome immagine del processo da terminare. Jolly '*' può essere utilizzato per specificare tutti i compiti o nomi delle immagini.
7.	<b>/ T</b> Termina il processo specificato e tutti i processi figlio che sono stati avviati da esso.
8.	<b>/ F</b> Specifica di interrompere forzatamente il processo (es).

## Esempi

```
taskkill /f /im notepad.exe
```

Il comando precedente uccide il compito blocco note aperto, se aperto.

```
taskkill /pid 9214
```

Il comando precedente uccide un processo che ha un processo di 9214.

## Avvio di un nuovo processo

scripting DOS ha anche la disponibilità ad avviare un nuovo processo del tutto. Ciò si

ottiene mediante il comando di START.

## Sintassi

```
START "title" [/D path] [options] "command" [parameters]
```

dove

- **Titolo** - Testo per la barra del titolo della finestra CMD (richiesto.)
- **percorso** - directory di partenza.
- **di comando** - Il comando, file batch o un programma eseguibile per l'esecuzione.
- **Parametri** - I parametri passati al comando.

In seguito sono la descrizione delle opzioni che possono essere presentati al comando di START.

S.No.	Opzioni e descrizione
1.	<b>/ MIN</b> finestra di avvio ridotto a icona
2.	<b>/ MAX</b> finestra di avvio ingrandita.
3.	<b>/BASSO</b> Utilizzare classe di priorità IDLE.
4.	<b>/NORMALE</b> Utilizzare classe di priorità normale.
5.	<b>/SUPERIORE AL NORMALE</b> Utilizzare classe di priorità ABOVENORMAL.
6.	<b>/AL DI SOTTO DELLA NORMA</b> Utilizzare BELOWNORMAL classe di priorità.
7.	<b>/ALTO</b> Utilizzare classe di priorità ALTA.

- |    |  |
|----|--|
| 8. | <b>/TEMPO REALE</b><br>Utilizzare REALTIME classe di priorità. |
|----|--|

## Esempi

```
START "Test Batch Script" /Min test.bat
```

Il comando precedente eseguire lo script batch test.bat in una nuova finestra. Le finestre inizieranno in modalità minimizzata e hanno anche il titolo di "Test Batch Script".

```
START "" "C:\Program Files\Microsoft Office\Winword.exe" "D:\test\TESTA.txt"
```

Il comando precedente effettivamente eseguito Microsoft Word in un altro processo e quindi aprire il file TESTA.txt in MS Word.

## Script batch - Alias

Alias significa creare scorciatoie o parole chiave per i comandi esistenti. Supponiamo che se volessimo eseguire il comando di sotto del quale non è altro che la directory di comando messa in vendita con l'opzione / w per non mostrare tutti i dettagli necessari in un elenco di directory.

```
Dir /w
```

Supponiamo che se dovessimo creare un collegamento a questo comando come segue.

```
dw = dir /w
```

Quando vogliamo eseguire il comando **dir / w**, possiamo semplicemente digitare la **parola dw**. La parola 'DW' è diventata un alias al comando DIR / w.

## Creazione di un alias

Alias vengono gestiti utilizzando il comando **DOSKEY**.

## Sintassi

```
DOSKEY [options] [macroname = [text]]
```

dove

- **nomemacro** - Un nome breve per la macro.
- **il testo** - I comandi che si desidera ricordare.

In seguito sono la descrizione delle opzioni che possono essere presentati al comando DOSKEY.

S.No.	Opzioni e descrizione
1.	<b>/ REINSTALL</b> Installa una nuova copia di Doskey
2.	<b>/ LISTSIZE = dimensione</b> Imposta dimensione del buffer dei comandi.
3.	<b>/ MACRO</b> Consente di visualizzare tutte le macro Doskey.
4.	<b>/ MACRO: ALL</b> Consente di visualizzare tutte le macro Doskey per tutti i file eseguibili che hanno macro di Doskey.
5.	<b>/ MACRO: nomeexe</b> Consente di visualizzare tutte le macro Doskey per un dato eseguibile.
6.	<b>/STORIA</b> Consente di visualizzare tutti i comandi immagazzinati nella memoria.
7.	<b>/INSERIRE</b> Specifica che il nuovo testo digitato viene inserito nel vecchio testo.
8.	<b>/ OVERSTRIKE</b> Specifica che il nuovo testo sovrascrive vecchio testo.
9.	<b>/ EXENAME = nomeexe</b> Specifica l'eseguibile.

10.	<b>/ MACROFILE = filename</b> Specifica un file di macro da installare.
11.	<b>nomemacro</b> Specifica un nome per una macro si crea.
12.	<b>testo</b> Specifica i comandi che si desidera registrare.

## Esempio

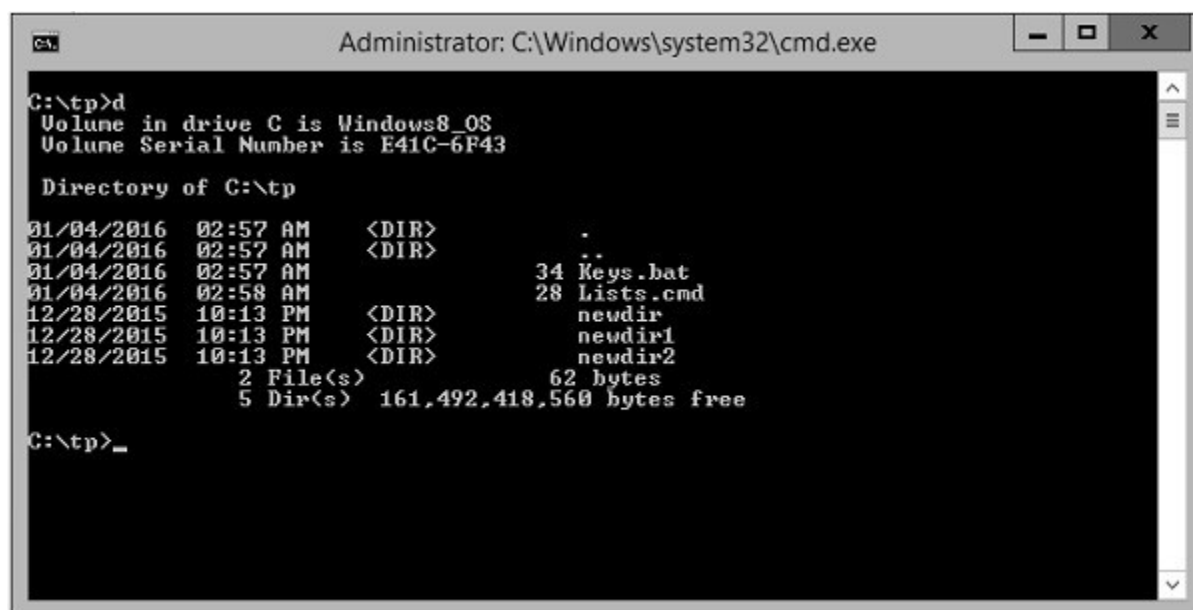
Creare un nuovo file chiamato keys.bat e digitare i seguenti comandi nel file. I comandi di seguito crea due alias, uno se per il comando cd, che va automaticamente alla directory chiamata di prova. E l'altro è per il comando dir.

```
@echo off
doskey cd = cd/test
doskey d = dir
```

Una volta che si esegue il comando, sarà in grado di eseguire questi alias nel prompt dei comandi.

## Produzione

La seguente schermata mostra che, dopo il file batch sopra creato viene eseguito, si può liberamente immettere il comando 'd' e vi darà l'elenco di directory che significa che il tuo alias è stato creato.



```
Administrator: C:\Windows\system32\cmd.exe

C:\>tp>d
Volume in drive C is Windows8_OS
Volume Serial Number is E41C-6F43

Directory of C:\tp

01/04/2016  02:57 AM    <DIR>          .
01/04/2016  02:57 AM    <DIR>          ..
01/04/2016  02:57 AM                34 Keys.bat
01/04/2016  02:58 AM                28 Lists.cmd
12/28/2015  10:13 PM    <DIR>          newdir
12/28/2015  10:13 PM    <DIR>          newdir1
12/28/2015  10:13 PM    <DIR>          newdir2
                2 File(s)                62 bytes
                5 Dir(s)  161,492,418,560 bytes free

C:\>tp>_
```

## Eliminazione di un alias

Un alias o macro possono essere eliminati impostando il valore della macro NULL.

### Esempio

```
@echo off
doskey cd = cd/test
doskey d = dir
d =
```

Nell'esempio di cui sopra, siamo prima impostazione della macro D a D = dir. Dopo di che stiamo impostando a NULL. Perché abbiamo impostare il valore di d NULL, la macro D sarà cancellato.

## Sostituzione di un alias

Un alias o macro possono essere sostituiti impostando il valore della macro al nuovo valore desiderato.

### Esempio

```
@echo off
doskey cd = cd/test
doskey d = dir

d = dir /w
```

Nell'esempio di cui sopra, siamo prima impostazione della macro D a D = dir. Dopo di che stiamo impostando a dir / w. Dal momento che abbiamo impostato il valore di D per un nuovo valore, l'alias 'd' ora assumere il nuovo valore.

## Script batch - Dispositivi

Windows ora ha una nuova libreria che può essere utilizzato in batch Script per lavorare con dispositivi collegati al sistema. Questo è noto come la console dispositivo - devcon.exe.

gli sviluppatori di driver di Windows e tester possono utilizzare DevCon per verificare che un driver è installato e configurato correttamente, inclusi i file INF corretti, stack di driver,



file del driver, e pacchetto driver. È inoltre possibile utilizzare i comandi DevCon (abilitare, disabilitare l'installazione, avvio, arresto e continuano) negli script per testare il driver.

**DevCon è uno strumento da riga di comando che esegue funzioni di gestione dei dispositivi su computer locali e computer remoti.**

Driver di visualizzazione e informazioni dispositivo DevCon in grado di visualizzare le seguenti proprietà di driver e dispositivi su computer locali e computer remoti (che eseguono Windows XP e versioni precedenti) -

- ID hardware, ID compatibili, e ID istanza dispositivo. Questi identificatori sono descritte in dettaglio nelle stringhe di identificazione del dispositivo.
- classi di configurazione del dispositivo.
- I dispositivi in una classe di installazione dispositivi.
- file INF e file di driver di dispositivo.
- Dettagli di pacchetti di driver.
- risorse hardware.
- Stato del dispositivo.
- stack di driver previsto.
- pacchetti driver di terze parti nell'archivio driver.
- Ricerca di dispositivi DevCon può cercare i dispositivi installati e disinstallati in un computer locale o remoto di ID hardware, dispositivo esempio ID, o classe di installazione dispositivi.
- Cambiare le impostazioni del dispositivo DevCon può modificare lo stato o la configurazione di periferiche Plug and Play (PnP) sul computer locale nei seguenti modi -
  - Attivare un dispositivo.
  - Disabilitare un dispositivo.
  - Aggiornare i driver (interattiva e non interattiva).
  - Installare un dispositivo (creare un devnode e installare il software).
  - Rimuovere un dispositivo dalla struttura dei dispositivi e cancellare il suo stack di periferica.
  - Ripetere la scansione per dispositivi Plug and Play.
  - Aggiungere, cancellare e riordinare gli ID hardware dei dispositivi di root-enumerati.
  - Modificare i driver di filtro superiore e inferiore per una classe di installazione

dispositivi.

- Aggiungere ed eliminare i pacchetti driver di terze parti dal driver.

DevCon (devcon.exe) è incluso quando si installa il WDK, Visual Studio e Windows SDK per applicazioni desktop. Kit devcon.exe è disponibile nelle seguenti posizioni una volta installato.

```
%WindowsSdkDir%\tools\x64\devcon.exe  
%WindowsSdkDir%\tools\x86\devcon.exe  
%WindowsSdkDir%\tools\arm\devcon.exe
```

## Sintassi

```
devcon [/m:\\computer] [/r] command [arguments]
```

dove

- **/ m: \\ computer - Esegue il comando sul computer remoto specificato.**  
Sono richiesti i backslash.
- **/ r - riavvio condizionale.** Riavvia il sistema dopo aver completato l'operazione solo se è necessario un riavvio per fare un cambiamento efficace.
- **di comando - Specifica un comando DevCon.**
- Per elencare e visualizzare le informazioni sui dispositivi sul computer, utilizzare i seguenti comandi -
  - HwIDs DevCon
  - Classi DevCon
  - DevCon listclass
  - DriverFiles DevCon
  - DriverNodes DevCon
  - Risorse DevCon
  - DevCon Stack
  - DevCon Stato
  - DevCon Dp\_enum
- Per la ricerca di informazioni sui dispositivi sul computer, utilizzare i seguenti comandi -

- DevCon Ricerca
- DevCon FindAll
- Per manipolare il dispositivo o cambiare la sua configurazione, utilizzare i seguenti comandi -
  - DevCon Abilita
  - DevCon Disabilita
  - DevCon Aggiornamento
  - DevCon UpdateNI
  - DevCon Installare
  - DevCon Rimuovere
  - DevCon Rescan
  - DevCon Restart
  - DevCon Reboot
  - DevCon SetHwID
  - DevCon ClassFilter
  - DevCon Dp\_add
  - DevCon Dp\_delete

## Esempi

Di seguito sono riportati alcuni esempi su come viene utilizzato il comando DevCon.

```
List all driver files
```

Il comando seguente utilizza l'operazione DriverFiles DevCon per elencare i nomi dei file di driver che i dispositivi per l'uso del sistema. Il comando utilizza il carattere jolly (\*) per indicare tutti i dispositivi del sistema. Poiché l'output è ampia, il comando utilizza il carattere di reindirizzamento (>) per reindirizzare l'output in un file di riferimento, driverfiles.txt.

```
devcon driverfiles * > driverfiles.txt
```

Il comando seguente utilizza l'operazione di stato DevCon per trovare lo stato di tutti i dispositivi sul computer locale. E 'quindi salva lo stato nel file status.txt per la registrazione o la revisione successiva. Il comando utilizza il carattere jolly (\*) per

rappresentare tutti i dispositivi e il carattere di reindirizzamento (>) per reindirizzare l'output al file status.txt.

```
devcon status * > status.txt
```

Il seguente comando consente a tutti i dispositivi di stampa sul computer specificando la classe di installazione della stampante in una DevCon comando di abilitazione. Il comando include il parametro /r, che riavvia il sistema se è necessario effettuare l'effettiva abilitazione.

```
devcon /r enable = Printer
```

Il comando seguente utilizza il DevCon Installa operazione per installare un dispositivo tastiera sul computer locale. Il comando include il percorso completo del file INF per la periferica (Keyboard.inf) e un ID hardware (\* PNP030b).

```
devcon /r install c:\windows\inf\keyboard.inf *PNP030b
```

Il seguente comando esegue la scansione del computer per i nuovi dispositivi.

```
devcon scan
```

Il seguente comando una nuova scansione del computer per i nuovi dispositivi.

```
devcon rescan
```

## Script batch - Registro

Il Registro è uno degli elementi chiave su un sistema Windows. Esso contiene molte informazioni su vari aspetti del sistema operativo. Quasi tutte le applicazioni installate su un sistema Windows interagiscono con il Registro di sistema in una forma o l'altra.

Il Registro contiene due elementi di base: . Chiavi ei valori **chiavi di registro sono oggetti contenitore simile alle cartelle valori del Registro di sistema sono oggetti non-contenitore simile a file..** Le chiavi possono contenere valori o ulteriori chiavi. Le chiavi si fa riferimento con una sintassi simile a nomi di percorso di Windows ', utilizzando backslash per indicare i livelli di gerarchia.

Questo capitolo esamina varie funzioni come l'interrogazione di valori, l'aggiunta,

l'eliminazione e la modifica dei valori dal Registro di sistema.

S.No	Tipi di Registro e descrizione
1	La lettura dal Registro di sistema (batch_script_reading_registry.html) La lettura dal Registro di sistema avviene tramite il comando REG QUERY.
2	L'aggiunta al Registro (batch_script_adding_registry.html) L'aggiunta al Registro di sistema avviene tramite il comando REG ADD.
3	L'eliminazione dal Registro di sistema (batch_script_deleting_registry.html) L'eliminazione dal registro avviene tramite il comando REG DEL.
4	Copia di chiavi di registro (batch_script_copying_registry_keys.html) Copia dal Registro di sistema viene eseguita tramite il comando REG COPY.
5	Confrontando chiavi di registro (batch_script_comparing_registry_keys.html) Confrontando le chiavi di registro viene fatto tramite il comando REG COPY.

## Script batch - Rete

script batch ha la possibilità di lavorare con le impostazioni di rete. Il comando NET viene utilizzato per aggiornare, correggere, o visualizzare le impostazioni di rete o di rete.

Questo capitolo esamina le diverse opzioni disponibili per il comando net.

S.No	NET Comandi e descrizione
1	CONTI NET (batch_script_net_accounts.html) Guarda le attuali restrizioni di password e di accesso per il computer.
2	NET CONFIG (batch_script_net_config.html) Consente di visualizzare le impostazioni del server o gruppo di lavoro corrente.
3	COMPUTER NET (batch_script_net_computer.html) Aggiunge o rimuove un computer collegato al controller di dominio Windows.
4	NET USER (batch_script_net_user.html) Questo comando può essere utilizzato per le seguenti Guarda i dettagli di un particolare account utente.

5	NET STOP / START (batch_script_net_stop_start.html) Questo comando viene utilizzato per arrestare e avviare un servizio particolare.
6	STATISTICHE NET (batch_script_net_statistics.html) statistiche di rete visualizzazione della workstation o server.
7	NET USE (batch_script_net_use.html) Si connette o disconnette il computer da una risorsa condivisa o visualizza le informazioni sulle connessioni.

## Script batch - Stampa

La stampa può anche essere controllato da nell'ambito di un gruppo di script tramite il comando NET PRINT.

### Sintassi

```
PRINT [/D:device] [[drive:][path]filename[...]]
```

Dove / D: dispositivo - Specifica una periferica di stampa.

### Esempio

```
print c:\example.txt /c /d:lpt1
```

Il comando precedente stamperà il file example.txt alla porta parallela LPT1.

## Printer Control riga di comando

Come di Windows 2000, molti, ma non tutti, le impostazioni della stampante possono essere configurate dalla riga di comando di Windows utilizzando printui.dll e RUNDLL32.EXE

### Sintassi

RUNDLL32.EXE PRINTUI.DLL,PrintUIEntry [ Qualora alcune delle opzioni disponi

- / dl - Elimina stampante locale.
- / dn - Eliminare la connessione della stampante di rete.
- / dd - Elimina driver della stampante.
- / E - Preferenze di stampa schermo.
- / f [file] - In entrambi i file INF o file di output.
- / F [file] - posizione di un file INF che il file INF specificato con /
- / ia - Installare driver di stampa utilizzando il file inf.
- / id - Installare driver della stampante utilizzando Installazione guid
- / se - installare la stampante utilizzando il file inf.
- / II - Installare stampante tramite guidata stampante con un file inf.
- / IL - Installare stampante utilizzando Installazione guidata stampante
- / a - Aggiungere la connessione della stampante di rete.
- / ip - Installare stampante tramite procedura guidata di installazione
- / k - Stampa pagina di prova per stampante specificata, non può essere
- / l [percorso] - Stampante percorso di origine del driver.
- / m [modello] - Stampante nome del modello del driver.
- / n [nome] - nome della stampante.
- / o - stampante schermo vista della coda.
- / p - proprietà della stampante di visualizzazione.
- / SS - impostazioni della stampante Conservare in un file.
- / Sr - Ripristinare le impostazioni della stampante da un file.
- / y - Imposta stampante come predefinita.
- / Xg - Get impostazioni della stampante.
- / Xs - Impostare le impostazioni della stampante.

## Verifica se una stampante esiste

Ci possono essere casi in cui si potrebbe essere collegato a una stampante

L'esistenza di una stampante può essere valutata con l'aiuto del RUNDLL32.E

## Esempio

```
SET PrinterName = Test Printer
SET file = %TEMP%\Prt.txt
RUNDLL32.EXE PRINTUI.DLL,PrintUIEntry /Xg /n "%PrinterName%" /f "%file%"

IF EXIST "%file%" (
    ECHO %PrinterName% printer exists
) ELSE (
    ECHO %PrinterName% printer does NOT exists
)
```

Il comando sopra farà il seguente -

- Sarà prima impostare il nome della stampante e impostare un nome di file
- I comandi RUNDLL32.EXE printui.dll saranno utilizzati per verificare se

## Script batch - Debugging

Molto spesso di quanto non si può incorrere in problemi durante l'esecuzione

### Messaggio di errore

Per scoprire la fonte del messaggio, attenersi alla seguente procedura -

**Fase 1** - REM il & commat; ECHO linea OFF, cioè REM & commat; ECHO OFF o ::

**Fase 2** - Eseguire il file batch con i parametri della riga di comando neces

```
test.bat > batch.log 2>&1
```

**Fase 3** - Cerca il batch.log file per i messaggi di errore

**Fase 4** - Controllare la riga precedente per qualsiasi comando imprevisti o

**Passo 5** - Correggere l'errore e ripetere questo processo fino a quando tutt

### Righe di comando complesse

Un'altra fonte comune di errori vengono erroneamente comandi reindirizzato,



Per verificare la validità di questi comandi complessi, attenersi alla seguente procedura:

**Fase 1 - Inserire "linee di controllo di comando" appena prima di una linea di comando.**

Di seguito è riportato un esempio in cui è inserito il comando echo per seguire la procedura.

```
TYPE %Temp%.\apipaorg.reg
ECHO.===== TYPE %Temp%.\apipaorg.reg
| FIND
"[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\TCPIP\Parameters\...]
```

**Fase 2 - Seguire la procedura di trovare fonti messaggio di errore descritti nel file di log.**

**Fase 3 - Prestare particolare attenzione alla produzione delle linee di comando.**

## Le subroutine

Le subroutine che generano messaggi di errore rappresentano una "sfida" in quanto non sono facilmente identificabili. Per aiutare a scoprire ciò che provoca la chiamata non corretta alla subroutine, si può utilizzare la seguente procedura:

**Fase 1 - Aggiungere e ripristinare la variabile contatore all'inizio dello script.**

```
SET Counter = 0
```

**Fase 2 - incrementa il contatore ogni volta che la subroutine viene chiamata.**

```
SET /A Counter += 1
```

**Fase 3 - Inserire un'altra linea subito dopo l'incremento del contatore, che incrementa il contatore.**

**Fase 4 - Seguire la procedura di trovare fonti messaggio di errore descritti nel file di log.**

## versioni di Windows

Se avete intenzione di distribuire i file batch ad altri computer che possono essere di diverse versioni di Windows, è importante controllare le varie versioni del sistema operativo. L'esempio seguente mostra come controllare per le varie versioni del sistema operativo.

```
@ECHO OFF
:: Check for Windows NT 4 and later

IF NOT "%OS%"=="Windows_NT" GOTO DontRun
:: Check for Windows NT 4
VER | FIND "Windows NT" >NUL && GOTO DontRun
:: Check for Windows 2000
VER | FIND "Windows 2000" >NUL && GOTO DontRun
:: Place actual code here . . .
:: End of actual code . . .
EXIT

:DontRun
ECHO Sorry, this batch file was written for Windows XP and later versions
```

## Script batch - Registrazione

Accesso in è possibile in script batch utilizzando il comando di reindirizz

## Sintassi

```
test.bat > testlog.txt 2> testerrors.txt
```

## Esempio

Creare un file denominato test.bat e digitare il seguente comando nel file.

```
net statistics /Server
```

Il comando precedente ha un errore perché l'opzione per il comando net stat

## Produzione

Se il comando con il file test.bat sopra viene eseguito come

```
test.bat > testlog.txt 2> testerrors.txt
```

E si apre il file testerrors.txt, verrà visualizzato il seguente errore.

```
The option /SERVER is unknown.
```

La sintassi di questo comando è -

```
NET STATISTICS  
[WORKSTATION | SERVER]
```

Ulteriori informazioni sono disponibili digitando NET HELPMSG 3506.

Se si apre il file chiamato testlog.txt, che vi mostrerà una registrazione

```
C:\tp>net statistics /Server
```

«Precedente (batch\_scrip

Prossimo capitolo " (batch\_sc



## COLOR PICKER



(../colors/colors\_picker.html)

## PER SAPERNE DI PIÙ:

- color Converter (../colors/colors\_converter.html)
- Google Maps (../howto/howto\_google\_maps.html)
- pulsanti animati (../howto/howto\_css\_animate\_buttons.html)
- modali Scatole (../howto/howto\_css\_modals.html)
- Immagini modali (../howto/howto\_css\_modal\_images.html)
- Tooltips (../howto/howto\_css\_tooltip.html)

caricatori ([../howto/howto\\_css\\_loader.html](#))  
JS animazioni ([../howto/howto\\_js\\_animate.html](#))  
Progress Bar ([../howto/howto\\_js\\_progressbar.html](#))  
menu a discesa ([../howto/howto\\_js\\_dropdown.html](#))  
presentazione ([../howto/howto\\_js\\_slideshow.html](#))  
navigazione laterale ([../howto/howto\\_js\\_sidenav.html](#))  
HTML include ([../howto/howto\\_html\\_include.html](#))  
Tavolozze di colori ([../w3css/w3css\\_color\\_palettes.html](#))

---



SEGNALA ERRORE  
STAMPA PAGINA ()  
DI ()

---

## Top 10 Tutorial

HTML lezione ([../html/default.html](#))  
CSS lezione ([../css/default.html](#))  
JavaScript lezione ([../js/default.html](#))  
W3.CSS lezione ([../w3css/default.html](#))  
Bootstrap lezione ([../bootstrap/default.html](#))  
SQL lezione ([../sql/default.html](#))

[PHP lezione \(../php/default.html\)](#)  
[jQuery lezione \(../jquery/default.html\)](#)  
[Angular lezione \(../angular/default.html\)](#)  
[XML lezione \(../xml/default.html\)](#)

## Top 10 Riferimenti

[HTML Riferimento \(../tags/default.html\)](#)  
[CSS Riferimento \(../cssref/default.html\)](#)  
[JavaScript Riferimento \(../jsref/default.html\)](#)  
[W3.CSS Riferimento \(../w3css/w3css\\_references.html\)](#)  
[Statistiche browser \(../browsers/default.html\)](#)  
[PHP Riferimento \(../php/php\\_ref\\_array.html\)](#)  
[HTML Colors \(../colors/colors\\_names.html\)](#)  
[HTML Set di caratteri \(../Charsets/default.html\)](#)  
[jQuery Riferimento \(../jquery/jquery\\_ref\\_selectors.html\)](#)  
[AngularJS Riferimento \(../angular/angular\\_ref\\_directives.html\)](#)

## Top 10 Esempi

[HTML Esempi \(../html/html\\_examples.html\)](#)  
[CSS Esempi \(../css/css\\_examples.html\)](#)  
[JavaScript Esempi \(../js/js\\_examples.html\)](#)  
[W3.CSS Esempi \(../w3css/w3css\\_examples.html\)](#)  
[HTML DOM Esempi \(../js/js\\_dom\\_examples.html\)](#)  
[PHP Esempi \(../php/php\\_examples.html\)](#)  
[jQuery Esempi \(../jquery/jquery\\_examples.html\)](#)  
[ASP Esempi \(../asp/asp\\_examples.html\)](#)  
[XML Esempi \(../xml/xml\\_examples.html\)](#)  
[SVG Esempi \(../svg/svg\\_examples.html\)](#)

---