
# Metodi di elaborazione dei file con VBA

L'elaborazione dei file è la capacità di memorizzare i valori di un documento nel computer in modo da poter recuperare tali valori successivamente, oppure è la possibilità di salvare i valori da un'applicazione e recuperare tali valori quando è necessario. Prima di eseguire l'elaborazione dei file, la prima azione che è necessario eseguire consiste nel creare un file e per sostenere tale processo, VBA fornisce una procedura denominata Open che presenta la seguente sintassi

```
Open pathname For Output [Access access] [lock] As [#]filenumber [Len=reclength]
```

La dichiarazione **Open** mostra molti argomenti, alcuni sono necessari e altri no, le espressioni obbligatorie sono: La dichiarazione Open, l'espressione For Output e la dichiarazione As #, oltre all'argomento, pathname (percorso) che è rappresentato da una stringa che può essere il nome del file, inoltre Il file può avere un'estensione o meno. Ecco un esempio:

```
Open "Prova.txt"
```

Se si specifica solo il nome del file, sarebbe considerato nella stessa cartella in cui è la cartella di lavoro corrente (la cartella di lavoro che è stata aperta quando è stata eseguita questa istruzione). Se si desidera, è possibile fornire un percorso completo per il file, che include l'unità, il nome della cartella, fino al nome del file, con o senza estensione. Oltre al nome del file o il suo percorso, la modalità è un fattore necessario in quanto indica l'azione reale che si desidera eseguire, come la creazione di un nuovo file o solo l'apertura di uno esistente. Questo fattore può essere rappresentato da una delle seguenti parole chiave

- Output: verrà creato il file e pronto a ricevere i valori (normali)
- Binary: verrà creato il file e pronto a ricevere i valori in formato binario (come combinazioni di 1 e 0)
- Append: Se il file esiste già, verrà aperto e nuovi valori possono essere aggiunti alla fine

Ecco un esempio di creazione di un file:

```
Sub apri ()
   Open "Prova.txt" For Output As #1
End Sub
```

Il tipo di accesso al file è facoltativo, questo argomento specifica quali tipi di azioni saranno eseguite sul file, come ad esempio la scrittura dei valori o solo la lettura dei valori esistenti. Questo fattore può avere uno dei seguenti valori:

- Scrittura: Dopo che è stato creato un nuovo file, verranno scritti i nuovi valori
- Lettura e Scrittura: Quando un nuovo file è stato creato o un file esistente è stato aperto, i valori possono essere letti o scritti

Se si decide di specificare il tipo di accesso, si deve precedere il suo valore con la parola chiave **Access** mentre il fattore di **Lock** (Blocco) è facoltativo, e indica come il processore dovrebbe comportarsi mentre si utilizza il file. I suoi possibili valori sono:


- Shared (In comune): Altre applicazioni (effettivamente richiamati dal processo) possono accedere al file mentre l'applicazione corrente lo sta utilizzando
- Blocca Scrittura: Non lasciare che altre applicazioni (processi) possano accedere a questo file, mentre l'applicazione corrente (processo) sta scrivendo su di esso
- Blocca Lettura e Scrittura: Non permettere ad altre applicazioni (processi) di accedere a questo file, mentre l'applicazione corrente (processo) lo sta utilizzando

Sul lato destro dell'espressione **#**, si deve digitare un numero, per l'argomento *filenumber*, compreso tra 1 e 511 e se si lavora su un file solo, si consiglia di utilizzare il numero 1, mentre se si sta lavorando su più file, è necessario utilizzare un numero incrementale. Se non si è riuscito a tenere traccia del numero o esiste la possibilità di confondersi, per conoscere il numero successivo è possibile utilizzare, la funzione FreeFile (), che restituisce il numero successivo disponibile nella sequenza. L'argomento reclength è facoltativo, se il file è stato aperto, questo fattore specifica la lunghezza del record che è stato letto.

### Chiusura di un file

Quando si crea e si inizia ad usare un file, o dopo aver aperto un file e mentre lo si utilizza, si impegnano risorse di sistema che potrebbero essere significative, ed è necessario quando si è terminato di utilizzare il file, liberare la memoria che si stava usando e rilasciare le risorse impegnate. Per eseguire questa operazione VBA fornisce una procedura denominata **Close**, la cui sintassi è:

```
Close [filenumberlist]
```

Dove l'argomento *filenumberlist* è lo stesso argomento *filenumber* che è stato precedentemente utilizzato per creare o aprire il file. Ecco un esempio di chiusura di un file:

```
Sub Prova ()
   Open "Prova.txt" For Output As #1
   Close #1
End Sub
```

### Scrittura in un file

Dopo aver creato un file, si deve scrivere dei valori al suo interno, a sostegno di questa azione, VBA fornisce due procedure, una si chiama **Print** e la sua sintassi è:

```
Print #filenumber, [outputlist]
```

La sintassi della dichiarazione Print richiede due argomenti, ma è necessario solo *filenumber* che corrisponde all'argomento *filenumber* che si è usato per creare il file. Filenumber è seguito da una virgola e poi dall'argomento outputlist che può essere costituito da 0 a 1 o più valori. Questo argomento è facoltativo, se non si vuole assegnare un valore al file, si può lasciare vuoto. Se invece volete assegnare un valore, si deve digitare una virgola dopo filenumber e seguire queste regole:


- Se si desidera assegnare un valore con spazi vuoti, utilizzare la funzione **Spc ()** e passare un numero intero, tra parentesi che rappresenta il numero di spazi vuoti. Per esempio **Spc (4)** includerebbe 4 spazi vuoti. Questo argomento è facoltativo, il che significa che è possibile ometterlo
- Invece di un determinato numero di spazi vuoti, è possibile lasciare che il sistema operativo si occupi di questa operazione e per fare questo, si deve richiamare la funzione **Tab ()** come parte del fattore di outputlist. La funzione Tab () specifica il numero di colonne da includere prima del valore e può essere utile se si desidera l'allineamento dei valori che si scriveranno nel file. Questo argomento è facoltativo, il che significa che è possibile ometterlo
- Per scrivere una stringa deve essere racchiusa tra virgolette
- Per scrivere un numero, sia un numero intero, o decimale, è sufficiente inserire il numero normalmente
- Per scrivere un valore booleano, si deve inserire come **True** o **False**
- Per scrivere un valore di data o ora, si deve digitarlo tra **#** e **#** e seguire le regole di data o ore della lingua del sistema operativo installato
- Per scrivere un valore nullo, digitare **Null**

Ecco un esempio di scrittura di alcuni valori:

```
Sub Prova ()
   Open "Prova.txt" For Output As #1
   Print #1, "Gino"
   Print #1, "Gigetto"
   Print #1, True
   Print #1, #17/08/2014#
   Close #1
End Sub
```

Invece di scrivere un valore per riga, è possibile scrivere più di un valore con una sola dichiarazione. Per fare questo, si deve separarli con un punto e virgola o uno spazio vuoto. Ecco un esempio.

```
Sub Prova ()
   Open "Prova.txt" For Output As #1
      'I valori sono separate dal punto e virgola
   Print #1, "Gino"; "Gigetto"
     ' I valori sono separate da uno spazio vuoto
   Print #1, True #17/08/2014#
   Close #1
End Sub
```

### La scrittura di un file

Oltre alla procedura Print, VBA fornisce anche una procedura denominata **Write** che può essere utilizzata per scrivere uno o più valori in un file. La sintassi è la stessa di quella dell'argomento Print:

Write #filenumber, [outputlist]

Dove l'argomento *filenumber* è necessario e deve essere specificato al momento della creazione del file, mentre l'argomento outputlist è facoltativo e se si vuole omettere, si deve digitare una virgola dopo la dichiarazione filenumber e terminare la dichiarazione di scrittura.


In questo caso, una riga vuota verrà scritta nel file. Per scrivere i valori nel file, si devono seguire queste regole.

- Per assegnare un valore con spazi vuoti, si deve richiamare la funzione **Spc ()** e passare un numero che rappresenta il numero di spazi vuoti. Questo fattore è facoltativo, il che significa che è possibile ometterlo
- Per assegnare il valore di un determinato numero di colonne, si deve richiamare la funzione **Tab ()** e passare il numero di colonne come argomento. Questo fattore è facoltativo, il che significa che è possibile ometterlo
- Per scrivere una stringa, si deve includerla tra virgolette
- Per scrivere un numero, si può inserirlo normalmente
- Per scrivere un valore booleano, si deve inserirlo TRUE o FALSE
- Per scrivere un valore nullo, inserire NULL
- Per scrivere un valore di data o ora, si deve inserirlo tra # e #

Ecco un esempio di scrittura di alcuni valori:

```
Sub Prova ()
    Open "Prova.txt" For Output As #1
    Write #1, "Gino"
    Write #1, "Dott."
    Write #1, "Ginetto"
    Write #1, #17/08/2014#
    Write #1, 12.30
    Write #1, True
    Close #1
End Sub
```

È anche possibile scrivere i valori sulla stessa linea, basta separarli con uno spazio vuoto, una virgola o un punto e virgola. Ecco un esempio:

```
Sub Prova ()
    Open "Prova.txt" For Output As #1
     ` I valori sono separate da un punto e virgola
    Write #1, "Gino"; "Dott."; "Ginetto"
    ` I valori sono separate da uno spazio vuoto
    Write #1, #17/08/2014#, 12.30
    Write #1, True
    Close #1
End Sub
```

### Apertura di un file

A volte è necessario aprire un file esistente per poter leggere o scrivere altri dati, in questo caso VBA fornisce una procedura denominata **Open**, la cui sintassi è.

```
Open pathname For Input [Access access] [lock] As [#]filenumber [Len=reclength]
```

La sintassi della procedura Open mostra molti argomenti, alcuni sono necessari e altri no. Sono da ritenersi obbligatori i seguenti argomenti: La dichiarazione Open, l'espressione As # e il percorso che può essere il nome del file, inoltre il file può avere un'estensione o meno. Ecco un esempio:

```
Open "Prova.txt"
```


Se si specifica solo il nome del file, si cerca il file nella stessa cartella in cui è alloggiata la cartella di lavoro corrente. Se si desidera, è possibile fornire un percorso completo, ciò include l'unità, la cartella (opzionale), fino al nome del file, con o senza estensione. Oltre al nome del file o il suo percorso, è necessario fornire il tipo di accesso che si desidera fare. Per aprire un file, le modalità possono essere:

- **Binary**: Il file verrà aperto e il suo valore viene letto come valore binario
- **Append**: Il file verrà aperto e i nuovi dati verranno aggiunti alla fine dei dati già presenti
- **Input**: Il file verrà aperto in sola lettura
- **Random**: Il file sarà aperto per l'accesso casuale

Ecco un esempio di apertura di un file:

```
Sub Prova ()
   Open "Prova.txt" For Input As #1
End Sub
```

La modalità di accesso è facoltativa e può avere uno dei seguenti valori:

- Lettura: Dopo che il file è stato aperto, i valori presenti al suo interno verranno letti
- Lettura e Scrittura: Se il file è stato creato o aperto, i valori possono essere letti e/o scritti

Se si decide di specificare la modalità di accesso si deve precedere il suo valore con la parola chiave di accesso

### La lettura da un file

Dopo l'apertura di un file, è possibile leggere i valori al suo interno, e prima di leggere il valore, si dovrebbe dichiarare una o più variabili che riceverebbero i valori da leggere. Si deve ricordare che i benefici utilizzando una variabile sono di riservare uno spazio di memoria in cui è possibile memorizzare un valore. Allo stesso modo, quando si legge un valore da un file, si otterrebbe il valore dal file e quindi memorizzare tale valore nella memoria del computer. Una variabile renderebbe più facile per fare riferimento a tale valore in caso di necessità.

Per sostenere la possibilità di aprire un file, il VBA fornisce due procedure. Se i valori sono stati memorizzati usando la dichiarazione Print, si possono leggere i valori, utilizzando la dichiarazione **Input** o **Line Input**. La sintassi della procedura è

Input #filenumber, varlist

La dichiarazione richiede due argomenti, ma il secondo può essere realizzato in varie parti. L'argomento filenumber è lo stesso che si è usato per aprire il file ed è seguito da una virgola. L'argomento *varlist* può essere di 1 o più parti. Per leggere un solo valore, dopo la virgola di filenumber, si deve digitare il nome della variabile che riceverà il valore. Ecco un esempio:

```
Sub Prova ()
   Dim primo As String
   Open "Prova.txt" For Input As #1
   Input #1, primo
   Close #1
End Sub
```


Allo stesso modo, è possibile leggere ogni valore su una riga separata. Uno dei migliori usi di usare la dichiarazione Input è la capacità di leggere molti valori utilizzando una singola istruzione. Per effettuare questa operazione, si devono digitare le variabili sulla stessa, separandole con virgole. Ecco un esempio:

```
Sub Prova ()
   Dim primo As String
   Dim secondo As String
   Dim tempo As Boolean
   Open "Prova.txt" For Input As #1
   Input #1, primo, secondo, tempo
   Close #1
End Sub
```

Se si dispone di un file che contiene molte linee, per leggere una riga alla volta, è possibile utilizzare la parola chiave Line Input. La sua sintassi è:

Line Input #filenumber, varname

Questa affermazione necessita di due argomenti, ed entrambi sono necessari. L'argomento filenumber è il numero che si sarebbe usato per aprire il file. Quando l'istruzione Line Input viene richiamata chiamato, legge una riga di testo fino a raggiungere la fine del file. Uno dei limiti di Line Input è la difficoltà a leggere qualcosa di diverso testo perché potrebbe non essere in grado di determinare dove termina la linea.

Nel riesaminare la possibilità di scrivere i valori in un file, abbiamo visto che l'istruzione Print scrive un valore booleano come Vero o Falso, se si utilizza l'istruzione Input per leggere tale valore, l'interprete potrebbe non essere in grado di leggere il valore. Abbiamo visto che in alternativa alla dichiarazione Print si può usare la dichiarazione Write, inoltre abbiamo visto che, tra le differenze tra Input e Write, quest'ultimo scrive i valori booleani con il simbolo #, questo rende possibile per l'interprete di leggere facilmente tale valore. Per queste ragioni, nella maggior parte dei casi, può essere più utile usare la scrittura quando si creano valori stringa diversi in un file.