

[Home Page](#) [Informazioni](#) [Aiuto](#) 

## Utilizzare il registro di Windows

(prima parte) 

[http://www.vbsimple.net/howto/ht\\_044.htm](http://www.vbsimple.net/howto/ht_044.htm)

Difficoltà:  5 / 5

### **Premessa:**

Il codice trattato in questo articolo è stato riscritto per renderlo più intuitivo possibile e per utilizzarlo all'interno di un modulo di classe, aumentando e semplificando quindi la sua riusabilità. È stato studiato intorno a *Windows 95/98/ME* e tralascia alcuni aspetti fondamentali necessari per l'utilizzo in *Windows NT/2000*.

Quasi certamente in futuro il codice verrà nuovamente modificato per renderlo compatibile anche con gli altri sistemi ed implementare così tutte quelle caratteristiche non ancora presenti in questa seconda revisione.

Il giudizio di complessità massimo non deve spaventare perché in gran parte dovuto alla lunghezza e completezza del modulo di classe sviluppato ed in parte alla necessaria *poca dettagliatezza* che altrimenti avrebbe portato via molto più spazio del dovuto.

Un problema comunissimo per tantissimi programmi è il salvataggio ed il caricamento delle opzioni del programma sviluppato. Non è molto sensato, infatti, obbligare ogni volta l'utente a regolarsi i parametri del programma secondo le sue necessità; è molto più agevole far sì che all'uscita del programma tali parametri ed impostazioni vengano salvate da qualche parte e riprese al successivo ricaricamento del programma.

Esistono due soluzioni fondamentali per il salvataggio ed il ripristino delle impostazioni: la prima consiste nell'utilizzare i classici ed antiquati files di impostazioni con **estensione INI**, mentre la seconda sfrutta il **registro di Windows** (chiamato talvolta semplicemente *Registry*), un particolare file di Windows contenente molte impostazioni di quasi tutti i moderni programmi per Windows.

All'interno del registro sono contenute tantissime informazioni e talvolta è necessario accedervi per recuperare tali informazioni in mancanza di una funzione **API** che li ricavi direttamente. Si raccomanda quindi la manipolazione del registry soltanto come ultima possibilità per il recupero dei dati, preferendo quindi, l'utilizzo delle funzioni API specializzate.

In questo lungo tutorial svilupperemo una classe  per la lettura, la modifica, l'aggiunta e l'eliminazione di chiavi e valori del registro. Sono state aggiunte in seguito due funzioni per l'esportazione e l'importazione di parti del registro su file compatibili con Regedit. Si suppone una minima conoscenza della struttura del Registry.

Tralasciamo le spiegazioni e vediamo subito il codice denso di API:

```

1. Option Explicit
2. Option Base 0
3.
4. Private Const STANDARD_RIGHTS_ALL As Long = &H1F0000
5. Private Const STANDARD_RIGHTS_READ As Long = &H20000
6. Private Const STANDARD_RIGHTS_WRITE As Long = &H20000
7. Private Const SYNCHRONIZE As Long = &H100000
8. Private Const KEY_QUERY_VALUE As Long = &H1
9. Private Const KEY_SET_VALUE As Long = &H2
10. Private Const KEY_CREATE_SUB_KEY As Long = &H4
11. Private Const KEY_ENUMERATE_SUB_KEYS As Long = &H8
12. Private Const KEY_NOTIFY As Long = &H10
13. Private Const KEY_CREATE_LINK As Long = &H20
14.
15. Private Const REG_OPTION_NON_VOLATILE As Long = 0
16.

```

Si apre il sipario con la presentazione di una serie di costanti  API utilizzate più avanti: sono suddivise in due tipologie: quelle relative alla sicurezza e quella indicativa del tipo di dati da trattare ovvero **non volatili**, i normali dati del registro opposti a quelli volatili la cui esistenza si spegne al riavvio del computer. Vedremo le altre costanti nelle [enumerazioni](#)  successive.

```

17. Private Declare Function RegOpenKeyEx Lib "advapi32.dll" Alias
    "RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal ulOptions As
    Long, ByVal samDesired As Long, phkResult As Long) As Long
18. Private Declare Function RegCreateKeyEx Lib "advapi32.dll" Alias
    "RegCreateKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal Reserved As
    Long, ByVal lpClass As String, ByVal dwOptions As Long, ByVal samDesired As Long,
    ByVal lpSecurityAttributes As Long, phkResult As Long, lpdwDisposition As Long) As
    Long
19. Private Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As
    Long
20. Private Declare Function RegFlushKey Lib "advapi32.dll" (ByVal hKey As Long) As
    Long
21. Private Declare Function RegDeleteKey Lib "advapi32.dll" Alias
    "RegDeleteKeyA" (ByVal hKey As Long, ByVal lpSubKey As String) As Long
22. Private Declare Function RegEnumKeyEx Lib "advapi32.dll" Alias
    "RegEnumKeyExA" (ByVal hKey As Long, ByVal dwIndex As Long, ByVal lpName As String,
    lpcbName As Long, lpReserved As Long, ByVal lpClass As String, lpcbClass As Long,
    ByVal lpftLastWriteTime As Long) As Long
23. Private Declare Function RegQueryInfoKey Lib "advapi32.dll" Alias
    "RegQueryInfoKeyA" (ByVal hKey As Long, ByVal lpClass As String, lpcbClass As Long,
    lpReserved As Long, lpcSubKeys As Long, lpcbMaxSubKeyLen As Long, lpcbMaxClassLen
    As Long, lpcValues As Long, lpcbMaxValueNameLen As Long, lpcbMaxValueLen As Long,
    lpcbSecurityDescriptor As Long, lpftLastWriteTime As Long) As Long

```

Segue un primo elenco di dichiarazioni di funzioni API relative alle chiavi del registro. Non potendoci soffermare su ognuna accenneremo soltanto qualcosa sul loro funzionamento generale:

- **RegOpenKeyEx**  
Effettua l'apertura di una chiave o di una sua sottochiave.
- **RegCreateKeyEx**  
Crea ed apre una nuova sottochiave della chiave indicata.
- **RegCloseKey**  
Chiude una chiave precedentemente aperta liberando l'[handle](#) assegnato.
- **RegFlushKey**

Aggiorna i dati di una chiave e forzando il sistema operativo a scrivere tutte le modifiche non ancora scritte sul disco.

- **RegDeleteKey**  
Elimina una chiave e tutte le sue sottochiavi ed i relativi valori (solo su Windows 9x).
- **RegEnumKeyEx**  
Recupera i nomi di una o più sottochiavi della chiave indicata.
- **RegQueryInfoKey**  
Ricava informazioni su una determinata chiave come il numero di sottochiavi o di valori presenti.

```

24. Private Declare Function RegQueryValueEx Lib "advapi32.dll" Alias
    "RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String, ByVal
    lpReserved As Long, lpType As Long, lpData As Any, lpcbData As Long) As Long
25. Private Declare Function RegSetValueEx Lib "advapi32.dll" Alias
    "RegSetValueExA" (ByVal hKey As Long, ByVal lpValueName As String, ByVal Reserved
    As Long, ByVal dwType As Long, lpData As Any, ByVal cbData As Long) As Long
26. Private Declare Function RegEnumValue Lib "advapi32.dll" Alias
    "RegEnumValueA" (ByVal hKey As Long, ByVal dwIndex As Long, ByVal lpValueName As
    String, lpcbValueName As Long, ByVal lpReserved As Long, lpType As Long, lpData As
    Byte, lpcbData As Long) As Long
27. Private Declare Function RegDeleteValue Lib "advapi32.dll" Alias
    "RegDeleteValueA" (ByVal hKey As Long, ByVal lpValueName As String) As Long
28.

```

Dopo le funzioni relative alle chiavi segue l'elenco delle funzioni relative ai valori contenuti nelle chiavi stesse:

- **RegQueryValueEx**  
Interroga un valore di cui si conosce il nome ottenendo in risposta il suo contenuto ed il suo tipo di dati.
- **RegSetValueEx**  
Modifica il contenuto di un valore esistente e crea un nuovo valore nel caso che esso non dovesse esistere, con il tipo di dati specificato.
- **RegEnumValue**  
Interroga una chiave aperta per richiedere il nome di un valore in essa contenuti e contestualmente consente di recuperare anche il loro contenuto ed il tipo di dati.
- **RegDeleteValue**  
Effettua semplicemente l'eliminazione del valore specificato. Se il valore da eliminare è quello predefinito della chiave ne sarà cancellato soltanto il suo contenuto.

Si sottolinea il fatto che tutte le funzioni che utilizzano stringhe si riferiscono alla versione [ANSI](#) piuttosto che alla versione [Unicode](#) come indicato dalla A finale al nome della funzione dopo la parola chiave "Alias"; la versione Unicode avrebbe il suffisso W.

Pertanto ogni qual volta viene passata una stringa sotto forma di [array](#) di bytes è necessario

fare la conversione di formato tramite la funzione *StrConv*.

Seguono le [enumerazioni](#) scritte per semplificare l'utilizzo delle varie costanti API e poter pertanto usufruire del sistema [Intellisense](#).

```

29. Public Enum ChiaviPrincipali
30.     CHIAVE_APERTA = 0
31.     HKEY_CLASSES_ROOT = &H80000000
32.     HKEY_CURRENT_CONFIG = &H80000005
33.     HKEY_CURRENT_USER = &H80000001
34.     HKEY_DYN_DATA = &H80000006
35.     HKEY_LOCAL_MACHINE = &H80000002
36.     HKEY_PERF_ROOT = HKEY_LOCAL_MACHINE
37.     HKEY_PERFORMANCE_DATA = &H80000004
38.     HKEY_USERS = &H80000003
39. End Enum
40.

```

L'enumerazione *ChiaviPrincipali* verrà utilizzata da molte funzioni della classe per determinare quale chiave utilizzare: quella aperta dalla nostra [istanza](#) (riga 30) oppure una delle chiavi di sistema (righe 31-38) sempre aperte. In funzione della struttura della classe il programma potrà anche fornire una chiave arbitraria, ovvero un handle ad una chiave già aperta.

```

41. Public Enum ChiaviSecurity
42.     KEY_UNKNOWN = 0
43.     KEY_READ = ((STANDARD_RIGHTS_READ Or KEY_QUERY_VALUE Or KEY_ENUMERATE_SUB_KEYS Or
KEY_NOTIFY) And (Not SYNCHRONIZE))
44.     KEY_WRITE = ((STANDARD_RIGHTS_WRITE Or KEY_SET_VALUE Or KEY_CREATE_SUB_KEY) And
(Not SYNCHRONIZE))
45.     KEY_ALL_ACCESS = ((STANDARD_RIGHTS_ALL Or KEY_QUERY_VALUE Or KEY_SET_VALUE Or
KEY_CREATE_SUB_KEY Or KEY_ENUMERATE_SUB_KEYS Or KEY_NOTIFY Or KEY_CREATE_LINK) And
(Not SYNCHRONIZE))
46. End Enum
47.

```

L'enumerazione *ChiaviSecurity* trova il suo vero significato in Windows NT/2000 ed indica 4 permessi standard: da sconosciuto ad accesso massimo. In questa versione della classe studiata appositamente per Windows 95/98/ME ha poco senso di esistere e può tranquillamente essere ignorata.

```

48. Public Enum TipoValoriRegistro
49.     REG_NONE = 0
50.     REG_SZ = 1
51.     REG_EXPAND_SZ = 2
52.     REG_BINARY = 3
53.     REG_DWORD = 4
54.     REG_DWORD_LITTLE_ENDIAN = 4
55.     REG_DWORD_BIG_ENDIAN = 5
56.     REG_LINK = 6
57.     REG_MULTI_SZ = 7
58.     REG_RESOURCE_LIST = 8
59. End Enum
60.

```

L'ultima enumerazione è *TipoValoriRegistro* e rappresenta tutti i possibili tipi di dati presenti nel registro. Nel 90% dei casi si utilizzano i tipi stringa (**REG\_SZ**), binario (**REG\_BINARY**) e valore Double Word, un numero intero a 32 bit e senza segno (**REG\_DWORD**). Sono questi infatti i tipi di dati che è possibile inserire nel Registry utilizzando il programma Regedit. Questa classe non implementa i tipi di dati

**REG\_LINK, REG\_RESOURCE\_LIST e REG\_DWORD\_BIG\_ENDIAN.**

```
61. Private lngKeyValue As Long
62. Private lngKeySecurity As Long
63.
```

Gli unici due membri privati interni sono due valori di tipo Long: il primo identifica l'[handle](#) alla chiave aperta (vedi l'enumerazione *ChiaviPrincipali*) e verrà utilizzato per la maggior parte delle operazioni. L'altro valore rappresenta invece un indice di sicurezza come indicato dall'enumerazione *ChiaviSecurity*.

```
64. Private Sub Class_Initialize()
65.     lngKeyValue = 0
66. End Sub
67.
68. Private Sub Class_Terminate()
69.     Call RegCloseKey(lngKeyValue)
70. End Sub
71.
```

Giusto per sicurezza, all'istanza della classe la variabile **lngKeyValue** viene posta uguale a 0 per indicare che nessuna chiave è stata ancora aperta. Alla [deallocazione](#) dell'istanza viene comunque effettuata la chiusura della chiave **lngKeyValue**. È importante ricordarsi di chiudere le chiavi quando esse non sono più necessarie e, come vedremo in seguito, non chiudere mai le chiavi delle quali non è stata effettuata la diretta apertura.

[Segue parte 2 >>](#)

[Fibia FBI](#)

1 Aprile 2002

Corretto il 20 Settembre 2002



[Torna all'indice degli HowTo](#)

---