Home Page
Informazioni
Aiuto

Informazioni aggiuntive sulle DLL e funzioni API

http://www.vbsimple.net/info/info_09.htm

Le DLL - Dynamic Library Linkage (Librerie a collegamento dinamico) - sono una tipologia di file ampiamente utilizzata all'interno dei sistemi Windows, basti guardare il contenuto della cartella ***System*** e ***System32*** di Windows.

Vengono in genere utilizzate per riporre funzioni che verranno utilizzate da altri programmi tramite l'API (Application Programming Interface); ecco spiegato il collegamento dinamico. Infatti il contenuto di una DLL rimane separato dal file di un programma; essa viene collegata (linkata) al programma che ne fa uso in fase di esecuzione.
Si suddividono in tre grosse categorie:

1. DLL Standard
2. DLL ActiveX e Estensioni Visual Basic (VBX).
   Vedi Informazioni aggiuntive sugli ActiveX.
3. Driver normali (DRV) e di periferica virtuale (VXD)

Le seconde sono caratterizzate da classi, facilmente istanziabili e riutilizzabili all'interno di progetti Visual Basic. Non è possibile né creare né utilizzare VBX all'interno di Visual Basic 5, mentre era possibile farlo con le versioni precedenti.
Le ultime invece sono librerie con funzioni di basso livello o simulazione di periferica.

Le DLL Standard sono in genere scritte in linguaggio C e sono caratterizzate dall'esportazione delle loro funzioni. Ciò vuol dire che una funzione, se dichiarata esportabile, potrà essere richiamata attraverso l'API.

Purtroppo, essendo scritte in un linguaggio diverso e più raffinato di Visual Basic è necessario conoscere i prototipi della funzione, ovvero le dichiarazioni della funzione. La chiamata di una funzione esterna tramite parametri errati, nella tipologia, contenuto e numero può provocare un General Protection Fault oppure un blocco del computer.
In genere i prototipi di una DLL sono contenuti in un file header (intestazione) con estensione H distribuito assieme al kit di programmazione (SDK - Source Development Kit) di tale libreria.

Con l'edizione Professional o Enterprise di Visual Basic 5 è distribuito un programmino di nome API Viewer (APILOAD.EXE) che contiene i prototipi di molte funzioni API utilizzabili con Visual Basic. Esse riguardano le librerie tipiche di Windows 95/98/ME, presenti nella stragrande maggioranza dei computer.
Un altro, ottimo, visualizzatore di dichiarazioni API è API-Guide disponibile gratuitamente in rete all'indirizzo http://www.allapi.net.
Inoltre, all'interno del CD di Visual Basic 5, è fornita anche una serie di programmi dedicati alla lettura, investigazione e ricerca delle dichiarazioni di DLL standard.

Essendo, la maggior parte delle DLL, scritte in C, dove le stringhe sono trattate come

 Copyright Visual Basic Simple 2002

puntatori di caratteri, quando si accede a funzioni che manipolano stringhe, a volte è necessario fornire un numero intero lungo (*Long*) al posto della stringa. Per far questo è necessario utilizzare le funzioni di conversione da variabile a puntatore (*VarPtr*, *StrPtr*). In ogni caso le stinghe passate ad una funzione API dovrebbero terminare con un carattere NULL, poiché le funzioni C manipolano i vettori (matrici) di caratteri e, secondo gli schemi della programmazione in linguaggio C, tali vettori devono terminare con un NULL.

Altresì è importante, quando passiamo ad una funzione una variabile stringa che funga da buffer, inizializzarla, ovvero prepararla mediante l'istruzione *Space$* o *String$*, in modo da concedere alla funzione C uno spazio di memoria già allocato. Ove è richiesta la dimensione del buffer è importante calcolare bene lo spazio dichiarato. Ogni stringa in C è lunga tanti bytes per quanti sono i caratteri che compongono la stringa nel formato ANSI, mentre sono il doppio in formato UNICODE; in ogni caso c'è sempre un carattere NULL finale da tenere in considerazione nel calcolo dell'ampiezza del buffer.
Quasi sempre, sono previste due versioni della funzione che tratta stringhe: una per ANSI ed una per UNICODE. Tipicamente il nome della funzione per ANSI termina con un A finale, mentre il nome di quella per UNICODE termina con W.
Ad esempio, all'interno della libreria KERNEL32.DLL abbiamo una funzione di nome *CompareStringA* ed un'altra di nome *CompareStringW*; naturalmente la prima gestisce stringhe ANSI e la seconda quelle UNICODE.

Molte funzioni che richiedono il passaggio di un puntatore ad una stringa, sebbene siano dichiarate con *ByVal* possono modificare il contenuto stesso della stringa. Questo avviene perchè la chiamata mediante *ByVal* effettua una copia del puntatore, ma non dell'area di memoria puntata; questo fa sì che anche la copia del puntatore indichi la medesima area di memoria. Qualche ulteriore informazione è presente nella pagina dedicata alle funzioni.

Altre volte, nella dichiarazione della funzione API, è presente uno o più parametri di nome **Reserved**. Ciò indica che la funzione richiede necessariamente un puntatore (una variabile o un numero costante trasformato in puntatore). Il valore di tale puntatore può essere variato dalla funzione chiamata, ed in linea generale è consigliabile inizializzare il puntatore a 0, prima di passarlo alla funzione C.

Mediante l'utilizzo di alcune funzioni API è possibile alterare il comportamento di alcuni oggetti di un programma. Un'operazione tipica - ma molto rischiosa - è il subclassing di un form, ovvero l'intercettazione di messaggi inviati ad una finestra. Queste operazioni si effettuano mediante la creazione e comunicazione al sistema di una funzione di *callback* o di una *Window Procedure*, che, collegata alla ricezione dei messaggi alla finestra, ne anticipi l'esecuzione delle funzioni. Questo genere di operazione è detta anche *hooking* (agganciamento), da hook (uncino), poiché le Window Procedure o funzioni di callback vengono agganciate alle funzioni che danno vita alle finestre, ovvero a quelle funzioni che ricevono i messaggi.

Nota che quando parliamo di finestre nell'ambiente Windows, non intendiamo soltanto i forms, ma anche tutti i controlli Thunder ed i controlli aggiuntivi. Essi infatti, costituiscono delle window, ognuna con il suo handle univoco.

Prima di effettuare operazioni di questo genere è fondamentale salvare il progetto, poiché

 Copyright Visual Basic Simple 2002

una semplice variabile non inizializzata nel modo corretto può trasformarsi in un GPF e quindi in un rischio di blocco dell'ambiente di Visual Basic. Inoltre, in operazioni di subclassing è ***assolutamente sconsigliata*** l'interruzione del programma mediante le funzioni dell'IDE o attraverso l'istruzione END. Ciò provocherebbe un'instabilità del sistema che si può facilmente tramutare in un obbligo di riavvio della macchina.

Morale della favola? **Prima di richiamare una funzione API salvare sempre il progetto**.

Fibia FBI
18 Dicembre 2000

 Copyright Visual Basic Simple 2002