



Convenzioni per la programmazione

1. Convenzioni sui nomi.

1.1 Obiettivi

Per aiutare i programmatori (specialmente in progetti di gruppo) standardizza e decodifica la struttura e la logica di un'applicazione.

Per essere precisi, completi, leggibili, semplici da ricordare e non ambigui.

1.2 Convenzioni

1.2.1 Option Explicit

"Option Explicit" deve sempre essere usato per forzare la dichiarazione delle variabili ed è un buon ausilio al loro commento.

Il tempo perso cercando di trovare i bugs causati da digitazioni (aUserNameTmp vs. sUserNameTmp vs. sUserNameTemp) supera notevolmente il tempo necessario per una Dim.

1.2.2 Denominazione dei Controlli

Normalmente si adotta un sistema di denominazione composto da un prefisso che indichi il tipo di oggetto e un nome vero e proprio che descriva l'oggetto stesso. Fra il prefisso e il nome non può esserci lo spazio vuoto (non è ammesso dal VBA, così come non sono ammessi i segni di punteggiatura): per questo, per identificare velocemente le due parti, si scrive la prima lettera del nome vero e proprio con la lettera maiuscola. Facciamo qualche semplice esempio: una casella di testo in cui un utente deve inserire il suo cognome si dovrebbe chiamare txtCognome; un form in cui un utente deve inserire i suoi dati anagrafici non potrà che chiamarsi frmAnagrafica. Semplice no? È bene abituarsi fin dall'inizio a nominare correttamente gli oggetti, tutto poi vi verrà più facile. Qui di seguito troverete un elenco dei principali prefissi in uso. Tenete presente che i prefissi sono tratti dai nomi inglesi dei diversi controlli e oggetti. Per i form e i moduli i prefissi sono rispettivamente frm e mod. Per i diversi tipi di controlli, invece, abbiamo:

- txt per le caselle di testo;
- lbl per le etichette;
- cbo per le caselle combinate;
- lst per le liste;
- cmd per i pulsanti di comando;
- opt per i pulsanti di opzione;
- chk per le caselle di controllo;
- img per le immagini.

2. Commenti

Tutte le procedure devono iniziare con una breve descrizione delle caratteristiche funzionali.

Questa descrizione NON deve dire come è implementata la routine, poiché questo può variare nel tempo portando a perdite di tempo nel rivedere i commenti e soprattutto a commenti erronei

Il codice stesso e ogni eventuale suo commento descriverà l'implementazione.

I parametri devono essere descritti se (a) sono di dubbia interpretazione (questo non dovrebbe mai accadere) (b) richiedono dei valori specifici per il funzionamento della funzione (c) il valore viene modificato all'interno della routine (questo dovrebbe accadere solo se il parametro modificato è un valore di ritorno richiesto: nessuna funzione deve prendersi la libertà di modificare il valore dei parametri che non devono restituire un valore)

Ogni variabile di dubbia utilità deve essere commentata.

I nomi di variabili, funzioni, controlli e quant'altro deve essere chiaro a sufficienza.

3. Formattazione del Codice

I blocchi di indentazione (tab) devono essere di quattro spazi.

Il commento alla funzione dovrebbe essere indentato di una tabulazione.

Il primo statement della funzione dovrebbe partire indentato di un tab, e un tab per ogni linea di codice all'interno del blocco.

```
Function iFindUser (rasUserList() as String, rsTargetUser as String) as Integer
    ' Search UserList and if found, return index of first occurrence of TargetUser,
    ' else return -1
    Dim i as Integer 'loop counter
    Dim bFound as Integer 'target found flag
    iFindUser = -1
    i = 0
    While i <= Ubound(rasUserList) and Not bFound
        If rasUserList(i) = rsTargetUser Then
            bFound = True
            iFindUser = i
        End If
    Wend
End Function
```

4. Operatori

Usare SEMPRE "&" quando si concatenano stringhe e "+" lavorando con valori numerici.

L'uso del solo "+" può portare a problemi lavorando con due variabili (specie con i variant).

Per esempio:

```
vntVar1 = "10.01"
```

```
vntVar2 = 11
```

```
vntResult = vntVar1 + vntVar2 'vntResult = 21.01
```

```
vntResult = vntVar1 & vntVar2 'vntResult = 10.0111
```