

Visual Basic

1. ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΗΣ VISUAL BASIC	201
1.1. Το κεντρικό παράθυρο	202
1.2. Η φόρμα εργασίας	202
1.3. Η εργαλειοθήκη	204
1.4. Προσθήκη εργαλείων	205
1.5. Ο εξερευνητής εργασίας	206
1.6. Το παράθυρο ιδιοτήτων	207
1.7. Το παράθυρο μορφής φόρμας	208
2. ΑΡΧΕΣ ΣΥΓΧΡΟΝΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ	208
2.1. Εισαγωγή στον προγραμματισμό μιας εφαρμογής	208
2.2. Αντικειμενοστραφής προγραμματισμός	209
2.3. Προγραμματισμός οδηγούμενος από τα γεγονότα	210
3. Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ	212
3.1. Δομή μίας Visual Basic εφαρμογής	212
3.2. Τα τμήματα κώδικα	213
3.3. Σταθερές	214
3.4. Μεταβλητές	214
3.5. Ορισμός μεταβλητών	215
3.6. Δομές Απόφασης ή Επιλογής	216
3.7. Δομές Ανακύκλωσης	216
4. ΤΕΧΝΙΚΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ	217
4.1. Μενού επιλογών	217
4.2. Πλαίσια διαλόγου	218
4.3. Εκτυπώσεις	218
4.4. Επικοινωνία με το Clipboard	218
5. ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ	219
5.1. Εργαλείο δεδομένων	219
5.2. Ιδιότητες βάσεων δεδομένων	219
5.3. Τα εργαλεία εμφάνισης δεδομένων	220
6. ΓΡΑΦΙΚΑ	220
6.1. Το σύστημα συντεταγμένων	220
6.2. Μονάδες μέτρησης	221
6.3. Αλλαγή κλίμακας συστήματος συντεταγμένων	221
6.4. Χρώματα	221
6.5. Εργαλεία γραφικών	222
6.6. Μέθοδοι γραφικών	222
6.7. Ιδιότητες γραφικών	223
7. ΠΟΛΥΜΕΣΑ (MULTIMEDIA)	225
7.1. Multimedia εργαλεία της Visual Basic	225
7.2. Ηχος	226
7.3. Προσομοίωση κίνησης	226
7.4. Video	226

ΕΙΣΑΓΩΓΗ

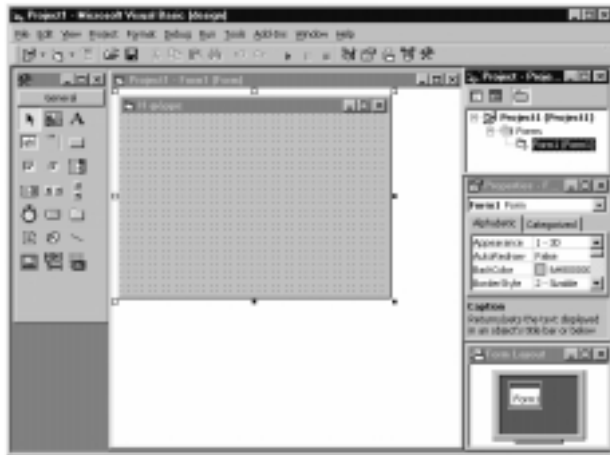
Η Visual Basic είναι το πρώτο ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών που δημιούργησε τις προϋποθέσεις για εύκολο και δημιουργικό προγραμματισμό στο περιβάλλον των Windows. Οι δυνατότητες και τα εργαλεία προγραμματισμού που προσφέρει, επιτρέπουν σε ένα προγραμματιστή να δημιουργήσει πολύ εύκολα και σε σύντομο χρονικό διάστημα, δυναμικές εφαρμογές ανάλογες με αυτές που χρησιμοποιούνται στο περιβάλλον των Windows. Οι εφαρμογές της Visual Basic χαρακτηρίζονται από τη φιλικότητα που διακρίνει το περιβάλλον εργασίας τους, αφού είναι δυνατό να περιέχουν όλα τα γνωστά στοιχεία που αποτελούν το γραφικό περιβάλλον επικοινωνίας των Windows. Όμως το στοιχείο που σίγουρα θα καταπλήξει έναν προγραμματιστή της Visual Basic είναι το εύχρηστο περιβάλλον προγραμματισμού της.

1. ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΗΣ VISUAL BASIC

Ας ξεκινήσουμε όμως τη γνωριμία μας με το περιβάλλον ανάπτυξης εφαρμογών της Visual Basic. Μόλις εκκινήσουμε τη Visual Basic στον υπολογιστή εμφανίζεται στην οθόνη το περιβάλλον ανάπτυξης εφαρμογών της Visual Basic (Σχήμα 1).

Η αρχική εικόνα της Visual Basic θυμίζει περισσότερο μία εφαρμογή των Windows παρά ένα προγραμματιστικό εργαλείο.

Αρχικά το περιβάλλον της Visual Basic φαίνεται να είναι αρκετά πολύπλοκο για ένα καινούριο χρήστη, όμως πολύ σύντομα θα εξοικειωθεί με αυτό και θα ανακαλύψει τα πλεονεκτήματα και την απλότητα του. Το περιβάλλον ανάπτυξης εφαρμογών της Visual Basic αποτελείται από τα παρακάτω στοιχεία :



Σχ.1. Το περιβάλλον ανάπτυξης εφαρμογών της Visual Basic.

- ➔ Το κεντρικό παράθυρο της Visual Basic.
- ➔ Τη φόρμα εργασίας (Form)
- ➔ Την εργαλειοθήκη (Toolbox).
- ➔ Τον εξερευνητή εργασίας (Project Explorer).
- ➔ Το παράθυρο ιδιοτήτων (Properties Window).
- ➔ Το παράθυρο μορφής φόρμας (Form Layout Window).

1.1. Το κεντρικό παράθυρο

Στο επάνω τμήμα της οθόνης εμφανίζεται το κεντρικό παράθυρο της Visual Basic (Σχήμα 2), το οποίο αποτελείται από τη γραμμή τίτλου (Title bar), το κεντρικό μενού επιλογών (Menu bar) και τη γραμμή εργαλείων (Toolbar).

Η αρχική θέση του κεντρικού παραθύρου, είναι στο πάνω τμήμα του περιβάλλοντος προγραμματι-



Σχ.2. Το κεντρικό παράθυρο της Visual Basic.

σμού της Visual Basic. Ωστόσο αν επιθυμούμε μπορούμε πολύ εύκολα να το μετακινήσουμε ή να αλλάξουμε τις διαστάσεις του όπως ένα οποιοδήποτε άλλο παράθυρο των Windows. Προσοχή όμως, μόλις κλείσουμε το κεντρικό παράθυρο τερματίζει αυτόματα και η Visual Basic.

Η **γραμμή τίτλου**, αποτελείται από τα βασικά στοιχεία ενός κλασικού παραθύρου των Windows. Επάνω της εμφανίζονται τα στοιχεία ταυτότητας του προγράμματος, το Πλαίσιο Μενού Ελέγχου (Control Menu Box) και τα πλήκτρα μεγιστοποίησης, ελαχιστοποίησης και κλεισίματος του παραθύρου (Minimize, Maximize buttons & Close buttons).

Το **κεντρικό μενού**, περιέχει όλες τις εντολές που χρειάζεται ο προγραμματιστής για τη δημιουργία μίας εφαρμογής. Για να ανοίξει ένα μενού επιλογών, πατάμε το αριστερό πλήκτρο του ποντικιού, όταν ο δρομέας του βρίσκεται επάνω στο όνομα του ή χρησιμοποιούμε από το πληκτρολόγιο κλειδιά πρόσβασης (Access Keys), πατάμε δηλαδή συνδυαστικά το αριστερό πλήκτρο Alt και το υπογραμμισμένο γράμμα του ονόματος της επιθυμητής επιλογής. Για να επιλέξουμε μια εντολή από ένα ήδη ανοιγμένο μενού επιλογών χρησιμοποιούμε επίσης το ποντίκι ή από το πληκτρολόγιο πατάμε το υπογραμμισμένο γράμμα του ονόματος της επιλογής. Στις εργασίες των μενού επιλογών που εκτελούνται συχνά, μπορούμε να έχουμε άμεση πρόσβαση χωρίς να ανοίξουμε ένα μενού επιλογών με τη χρήση πλήκτρων συντομίας (shortcut keys), τα οποία εμφανίζονται δίπλα από το όνομα της επιλογής.

Στη **γραμμή εργαλείων**, υπάρχουν οι πιο συχνά εκτελέσιμες επιλογές του κεντρικού μενού επιλογών σε μορφή εικονιδίων (Σχήμα 3). Τα εικονίδια είναι χωρισμένα σε ομάδες (groups) ανάλογα με τις εργασίες που εκτελούν.



Σχ.3. Η γραμμή εργαλείων.

Στο δεξί τμήμα της γραμμής εργαλείων (Toolbar) υπάρχουν δύο ενδεικτικά πλαίσια, το πλαίσιο θέσης και το πλαίσιο μεγέθους. Στο πλαίσιο θέσης εμφανίζονται κάθε φορά οι συντεταγμένες της θέσης του επιλεγμένου γραφικού αντικείμενου, ενώ στο πλαίσιο μεγέθους εμφανίζονται οι διαστάσεις του. Όταν μεταφέρουμε το ποντίκι επάνω σε κάποιο εικονίδιο της γραμμής εργαλείων, να εμφανίζονται επεξηγηματικά πλαίσια (Tooltips) που περιγράφουν τη λειτουργία του.

1.2. Η φόρμα εργασίας

Στο κέντρο της αρχικής οθόνης της Visual Basic εμφανίζεται ένα κενό παράθυρο το οποίο ονομάζεται φόρμα εργασίας (form). Η φόρμα εργασίας είναι το βασικότερο γραφικό αντικείμενο της Visual Basic. Κατά την εκτέλεση μιας εφαρμογής, κάθε φόρμα της Visual Basic αποτελεί ένα ανεξάρτητο παράθυρο επικοινωνίας της εφαρμογής με το χρήστη του συστήματος (Σχήμα 4).

Η φόρμα εργασίας αποτελεί το σημείο εκκίνησης του σχεδιασμού κάθε Visual Basic εφαρμογής. Ο σχεδιασμός της διεπαφής χρήστη (user interface) ξεκινά πάντοτε από τη δημιουργία μίας φόρμας εργασίας. Στη συνέχεια επάνω στη φόρμα θα "χτιστούν" σιγά-σιγά τα υπόλοιπα τμήματα της εφαρμογής. Αφού σχεδιάσουμε τη φόρμα θα τοποθετήσουμε επάνω σε αυτή τα υπόλοιπα γραφικά αντικείμενα της Visual Basic, μέσω των οποίων θα γίνεται η επικοινωνία του χρήστη με την εφαρμογή κατά την εκτέλεση της.



Σχ.4 Η φόρμα εργασίας.

Εξετάζοντας προσεχτικά μια φόρμα της Visual Basic, θα παρατηρήσουμε ότι περιέχει όλα τα γνωστά στοιχεία ενός παραθύρου, παρόμοιο με αυτά που χρησιμοποιούνται σε όλες τις κλασικές εφαρμογές των Windows. Το μέγεθος και τη θέση της φόρμας επάνω στην οθόνη, είναι δυνατό να τα μεταβάλλουμε με μεγάλη ευκολία κατά το σχεδιασμό της εφαρμογής. Αν μεταφέρουμε το ποντίκι επάνω σε κάποιο περίγραμμα της φόρμας, ο δείκτης του θα μετατραπεί σε διπλό βέλος. Κρατώντας πατημένο συνεχώς το αριστερό πλήκτρο του ποντικιού μπορούμε να το μετακινήσουμε και ταυτόχρονα παρατηρούμε ότι μεταφέρεται προς την ίδια κατεύθυνση και το περίγραμμα της φόρμας. Μόλις απελευθερώσουμε το πλήκτρο του ποντικιού η φόρμα θα αποκτήσει τις νέες διαστάσεις που της αποδώσαμε. Για να μεταφέρουμε τη φόρμα σε μια καινούρια θέση, πηγαίνουμε το ποντίκι επάνω στη γραμμή τίτλου της φόρμας και κρατώντας πατημένο συνεχώς το αριστερό του πλήκτρο το μετακινούμε και ταυτόχρονα μεταφέρεται η φόρμα εργασίας. Μόλις απελευθερώσουμε το πλήκτρο του ποντικιού η φόρμα καταλαμβάνει τη καινούρια θέση. Οι διαστάσεις και η θέση που θα αποδώσουμε στη φόρμα κατά το σχεδιασμό της, θα είναι αυτές με τις οποίες θα εμφανίζεται στην οθόνη του υπολογιστή και κατά την εκτέλεση της εφαρμογής.

Κατά τη διάρκεια του σχεδιασμού μιας φόρμας εμφανίζεται επάνω στην επιφάνειά της, ένα πλέγμα (Grid) το οποίο αποτελείται από μικρές μαύρες τελείες που ισαπέχουν μεταξύ τους. Οι τελείες αυτές δεν εμφανίζονται κατά την εκτέλεση της εφαρμογής, αλλά αποτελούν βοήθημα του προγραμματιστή για την τοποθέτηση και το σχεδιασμό των υπόλοιπων γραφικών αντικειμένων της Visual Basic επάνω στην επιφάνεια της φόρμας. Αν θέλουμε, κατά το σχεδιασμό της εφαρμογής μπορούμε να εξαφανίσουμε το πλέγμα από την επιφάνεια της φόρμας ή να αλλάξουμε την απόσταση των τελειών που το αποτελούν, θέτοντας την τιμή Show Grid=No και μετατρέποντας τις αριθμητικές τιμές στα πλαίσια Width και Height στην κατηγορία General της επιλογής Options στο μενού Tools της

Visual Basic. Ακόμη στην ίδια κατηγορία, αν ενεργοποιήσουμε την επιλογή Align Controls To Grid οι εξωτερικές άκρες κάθε εργαλείου θα στοιχίζονται στις γραμμές πλέγματος.

Μία εφαρμογή της Visual Basic μπορεί να περιέχει μία ή περισσότερες φόρμες εργασίας. Ο αριθμός των φορμών που θα σχεδιάσουμε, συνήθως είναι ανάλογος με το μέγεθος της εφαρμογής. Κάθε φόρμα μέσα σε μία εφαρμογή αποτελεί ένα ξεχωριστό παράθυρο και μπορεί να έχει διαφορετικό ρόλο. Μπορεί για παράδειγμα μια φόρμα να χρησιμοποιείται για εισαγωγή δεδομένων ή την εμφάνιση ενημερωτικών στοιχείων κ.ο.κ. Σε μια εφαρμογή είναι προτιμότερο, αν είναι δυνατό να αποφεύγουμε τη χρήση πολλών φορμών εργασίας, γιατί συνήθως ο μεγάλος αριθμός τους επηρεάζει αρνητικά τη λειτουργικότητα της εφαρμογής.

Το κυριότερο χαρακτηριστικό μίας φόρμας της Visual Basic, είναι ότι περιέχει τα δικά της εσωτερικά στοιχεία και γνωρίζει κάθε φορά πως πρέπει να συμπεριφερθεί στις απαιτήσεις του χρήστη. Για να γίνει κατανοητό αυτό μπορούμε να υποθέσουμε ότι η φόρμα που εμφανίζεται στην οθόνη εκκίνησης της Visual Basic αποτελεί από μόνη της την πρώτη μας εφαρμογή και μάλιστα χωρίς να χρειαστεί να γράψουμε καθόλου εντολές κώδικα. Αν θέλουμε να εκκινήσουμε την εκτέλεση της, επιλέγουμε από το μενού επιλογών Run της Visual Basic την επιλογή Start ή το πλήκτρο συντομίας F5 ή εναλλακτικά επιλέγουμε με το ποντίκι του υπολογιστή το δέκατο εικονίδιο της γραμμής εργαλείων.

Αμέσως η Visual Basic περνάει σε κατάσταση εκτέλεσης (run mode), όπως δηλώνει το λεκτικό [run] επάνω στη γραμμή τίτλου. Η φόρμα μας εμφανίζεται στην οθόνη στο σημείο που την είχαμε τοποθετήσει κατά το σχεδιασμό της και είναι δυνατό να εκτελέσουμε με αυτή όλες τις γνωστές εργασίες των παραθύρων των Windows. Μπορούμε δηλαδή, να μετακινήσουμε τη φόρμα και να μεταβάλλουμε το μέγεθος και τη θέση της επάνω στην οθόνη χρησιμοποιώντας το ποντίκι ή τα πλήκτρα μεγιστοποίησης και ελαχιστοποίησης και τέλος μπορούμε να κλείσουμε το παράθυρο μέσω των επιλογών του Πλαισίου Μενού Ελέγχου ή με το πλήκτρο κλεισίματος. Αν κλείσουμε τη φόρμα, επειδή αποτελεί το μοναδικό ανοικτό παράθυρο της εφαρμογής τερματίζει αυτόματα και η εκτέλεση της. Μπορούμε επίσης να σταματήσουμε την εκτέλεση της εφαρμογής και με την επιλογή End από το μενού επιλογών Run ή επιλέγοντας το δωδέκατο εικονίδιο της γραμμής εργαλείων της Visual Basic. Μετά το κλείσιμο της εφαρμογής επιστρέφουμε πάλι στην κατάσταση σχεδιασμού της Visual Basic.

Με το παραπάνω παράδειγμα έγινε αντιληπτό ότι η δημιουργία ενός παραθύρου με τη Visual Basic γίνεται πολύ απλά και γρήγορα, ενώ είναι σημαντικό ότι μια φόρμα κατά την εκτέλεση της εφαρμογής αποτελεί ένα πραγματικό αντικείμενο που αναγνωρίζει αυτόματα τις ενέργειες του χρήστη και αντιδρά ανάλογα, χωρίς να χρησιμοποιήσουμε ούτε μία εντολή κώδικα.

Ακόμη ο προγραμματιστής είναι δυνατό μέσω εντολών κώδικα να μεταβάλλει την εμφάνιση και να αναλάβει το χειρισμό μιας φόρμας. Μέσω εντολών κώδικα Basic έχουμε τη δυνατότητα μετατρέποντας την τιμή μιας ιδιότητας της φόρμας να αλλάξουμε τα χαρακτηριστικά της (π.χ. αλλαγή της ιδιότητας BackColor σημαίνει μεταβολή στο χρώμα φόντου), αλλά και με τη χρήση μεθόδων που υποστηρίζει η Visual να εκτελέσουμε μια ενέργεια (π.χ. με τη μέθοδο Move μπορούμε να μεταβάλουμε τη θέση της).

1.3. Η εργαλειοθήκη

Η εργαλειοθήκη (Toolbox) εμφανίζεται στο αριστερό άκρο της οθόνης και περιέχει τα βασικά εργαλεία σχεδιασμού της Visual Basic (Σχήμα 5). Τα εργαλεία σχεδιασμού βρίσκονται επάνω στην εργαλειοθήκη σε τρεις στήλες με μορφή εικονιδίου (icons) και ονομάζονται controls.



Σχ.5 Η εργαλειοθήκη.

Τα εργαλεία που περιέχει η εργαλειοθήκη αποτελούν μαζί με τις φόρμες εργασίας τα γραφικά αντικείμενα της Visual Basic. Με τα γραφικά αντικείμενα σχεδιάζουμε το τρόπο επικοινωνίας χρήστη-εφαρμογής (user interface). Η δημιουργία της διεπαφής χρήστη ολοκληρώνεται με την τοποθέτηση των κατάλληλων εργαλείων επάνω στη φόρμα εργασίας. Κατά την εκτέλεση της εφαρμογής μπορεί κάθε στιγμή ο χρήστης να επικοινωνεί μέσω των γραφικών αντικειμένων με την εφαρμογή και να κατευθύνει τη ροή εκτέλεσής της.

Ένας προγραμματιστής πριν ξεκινήσει το σχεδιασμό μίας εφαρμογής στη Visual Basic, πρέπει να γνωρίζει ακριβώς πως μπορεί να χρησιμοποιήσει κάθε εργαλείο που περιέχει η εργαλειοθήκη και ποια η λειτουργία καθενός από αυτά. Η σωστή επιλογή των κατά-

λληλων εργαλείων κατά τη δημιουργία μιας εφαρμογής εξασφαλίζει την επιτυχία της εκτέλεσής της.

Όπως είδαμε ήδη με τις φόρμες εργασίας, έτσι και κάθε εργαλείο της Visual Basic περιέχει τα δικά του χαρακτηριστικά και κυρίως έχει τη δική του εσωτερική συμπεριφορά. Ο χειρισμός των περισσότερων εργαλείων είναι ήδη γνωστός σε ένα πεπειραμένο χρήστη των Windows αφού θα τα έχει ήδη χρησιμοποιήσει σε κάποια άλλη εφαρμογή.

Η Visual Basic υποστηρίζει τρεις τύπους εργαλείων τα **εσωτερικά** (intrinsic controls) τα **ActiveX** και τα **ένθετα** (insertable objects). Τα εσωτερικά εργαλεία είναι τα βασικά εργαλεία, εμπεριέχονται στο εκτελέσιμο αρχείο της Visual Basic και εμφανίζονται πάντοτε στην εργαλειοθήκη. Τέτοια εργαλεία είναι τα πλήκτρα εντολής, οι εικόνες και οι ετικέτες. Στη συνέχεια θα κάνουμε μία περιγραφή των εσωτερικών εργαλείων που περιέχει η εργαλειοθήκη της Visual Basic.

- ✓ **Δείκτης (Pointer):** Ο δείκτης είναι το πρώτο εικονίδιο που εμφανίζεται στο πάνω αριστερό άκρο της εργαλειοθήκης και είναι το μοναδικό εργαλείο με το οποίο δεν μπορούμε να σχεδιάσουμε κάποιο γραφικό αντικείμενο. Ο δείκτης απεικονίζει το δρομέα (cursor) του ποντικιού και χρησιμοποιείται για την επιλογή και το χειρισμό των υπόλοιπων εργαλείων της Visual Basic. Κάθε φορά που ολοκληρώνουμε το σχεδιασμό ενός αντικειμένου με κάποιο από τα υπόλοιπα εργαλεία της εργαλειοθήκης αυτόματα επανεπιλέγεται ο δείκτης.
- ✓ **Πλαίσιο εικόνας (Picture box):** Είναι το πρώτο εργαλείο της εργαλειοθήκης και χρησιμοποιείται για την εμφάνιση εικόνων και γραφικών. Ένα πλαίσιο εικόνας είναι δυνατό να περιέχει αρχεία εικονιδίων (icons) με επέκταση ονόματος .ICO, αρχεία χαρτογράφησης (bitmaps) με επέκταση ονόματος .BMP ή τέλος μεταρχεία (metafiles) των Windows με επέκταση ονόματος .WMF ή .EMF.
- ✓ **Ετικέτα (Label):** Με τη χρήση των ετικετών μπορούμε να εμφανίσουμε κείμενο επάνω σε κάποιο σημείο της φόρμας. Το κείμενο που περιέχει μια ετικέτα δεν μπορεί να το μεταβάλλει ο χρήστης κατά την εκτέλεση της εφαρμογής. Συνήθως οι ετικέτες σε μία εφαρμογή χρησιμοποιούνται δίπλα από ένα άλλο γραφικό αντικείμενο της Visual Basic και περιγράφουν τα περιεχόμενα ή τη λειτουργία του.
- ✓ **Πλαίσιο κειμένου (Text box):** Χρησιμοποιείται κατά την εκτέλεση της εφαρμογής σαν πλαίσιο εισαγωγής, εμφάνισης και τροποποίησης κειμένου.
- ✓ **Πλαίσιο (Frame):** Το χρησιμοποιούμε αν θέλουμε να ομαδοποιήσουμε λειτουργικά ή διακοσμητικά διάφορα εργαλεία επάνω σε μία φόρμα. Για να ομαδοποιήσουμε ένα αριθμό εργαλείων πρέπει απαραίτητα να σχεδιάσουμε πρώτα το πλαίσιο και στη συνέχεια θα τοποθετήσουμε τα εργαλεία στο εσωτερικό του.

- ✓ **Πλήκτρο εντολής (Command button):** Απεικονίζει γραφικά ένα πλήκτρο, το οποίο όταν επιλεγεί από το χρήστη εκτελεί μία σειρά εντολών κώδικα που έχουμε συμπεριλάβει στην εφαρμογή μας.
- ✓ **Πλαίσιο ελέγχου (Check box):** Εμφανίζει ένα διακόπτη επιλογής στον οποίο ο χρήστης επιλέγει ανάμεσα σε δύο καταστάσεις (Αληθής-Ψευδής ή On-Off). Συνήθως σε μία φόρμα τοποθετούμε μία ομάδα πλαισίων ελέγχου για την εμφάνιση όλων των δυνατών τιμών μίας επιλογής. Κατά την εκτέλεση της εφαρμογής την ίδια χρονική στιγμή μπορούν να είναι επιλεγμένα δύο ή περισσότερα πλαίσια ελέγχου ταυτόχρονα.
- ✓ **Πλήκτρο επιλογής (Option button):** Δημιουργεί όπως και το πλαίσιο ελέγχου ένα διακόπτη επιλογής. Η λειτουργική διαφορά τους, είναι ότι κάθε φορά σε μία ομάδα πλήκτρων επιλογής μπορεί να είναι επιλεγμένο μόνο το ένα από αυτά, ενώ τα υπόλοιπα θα βρίσκονται σε κατάσταση μη επιλογής.
- ✓ **Πλαίσιο σύνθετης λίστας (Combo box):** Είναι ο συνδυασμός ενός πλαισίου εισαγωγής κειμένου και μίας λίστας επιλογών. Σε ένα πλαίσιο σύνθετης λίστας ο χρήστης έχει τη δυνατότητα να επιλέξει μία επιλογή της λίστας ή αν δεν τον ικανοποιεί καμία να πληκτρολογήσει μια νέα εισαγωγή στο πλαίσιο εισαγωγής κειμένου.
- ✓ **Λίστα (List):** Εμφανίζει μία λίστα επιλογών, στην οποία ο χρήστης έχει τη δυνατότητα να επιλέξει μία επιλογή μέσα από τα περιεχόμενα της. Η απλή λίστα δεσμεύει το χρήστη και δεν του επιτρέπει να εισάγει μια καινούργια επιλογή στα περιεχόμενα της.
- ✓ **Οριζόντια και κατακόρυφη γραμμή κύλισης (Horizontal and Vertical scroll bars):** Είναι δύο γραφικά εργαλεία χρήσιμα για γρήγορη μεταφορά μέσα σε ένα συνεχές πεδίο τιμών. Επιτρέπουν στο χρήστη μετακινώντας τα βέλη της γραμμής να μεταφερθεί γραφικά στην επιθυμητή τιμή και να παρουσιάσει τη θέση της ανάμεσα σε ένα πλήθος επιλογών.
- ✓ **Χρονομέτρης (Timer):** Εκτελεί εντολές κώδικα σε συγκεκριμένα χρονικά διαστήματα. Το εργαλείο του χρόνου δεν εμφανίζεται επάνω στη φόρμα κατά την εκτέλεση της εφαρμογής αλλά λειτουργεί πάντοτε στο παρασκήνιο. Για το λόγο αυτό δεν έχει σημασία σε ποιο σημείο της φόρμας θα το τοποθετήσουμε κατά το σχεδιασμό του.
- ✓ **Πλαίσιο λίστας επιλογής δίσκου (Drive list box):** Εμφανίζει σε μία λίστα όλους τους οδηγούς δίσκου του συστήματος και επιτρέπει στον χρήστη την ενεργοποίησή τους. Ο χρήστης δε χρειάζεται να ορίσει στο εργαλείο τους οδηγούς δίσκου του συστήματος, αλλά τους ανιχνεύει αυτόματα η Visual Basic.
- ✓ **Πλαίσιο λίστας επιλογής καταλόγων (Directory list box):** Εμφανίζει σε ιεραρχική δομή καταλόγους προγραμμάτων.
- ✓ **Πλαίσιο λίστας επιλογής αρχείων (File list box):** Χρησιμοποιείται για τη εμφάνιση λίστας αρχείων.
- ✓ **Γεωμετρικό Σχήμα (Shape):** Επιτρέπει κατά το σχεδιασμό της εφαρμογής τη τοποθέτηση ενός γεωμετρικού σχήματος επάνω στη φόρμα. Το εργαλείο αυτό δίνει τη δυνατότητα στο χρήστη να επιλέξει από ένα πλήθος γεωμετρικών σχημάτων όπως τετράγωνο, ορθογώνιο, κύκλο, κ.α.
- ✓ **Γραμμή (Line):** Επιτρέπει κατά το σχεδιασμό της εφαρμογής τη τοποθέτηση μίας ευθείας γραμμής επάνω στη φόρμα. Ο χρήστης μπορεί να επιλέξει την εμφάνιση της γραμμής μέσα από μία ποικιλία σχεδίων.
- ✓ **Εικόνα (Image):** Εμφανίζει εικόνες και γραφικά όπως και το πλαίσιο εικόνας, αξιοποιώντας όμως καλύτερα τη μνήμη και της πηγής του συστήματος. Το μειονέκτημα των εικόνων είναι ότι υποστηρίζουν λιγότερες ιδιότητες, γεγονότα και μεθόδους από τα πλαίσια εικόνας.
- ✓ **Εργαλείο Δεδομένων (Data):** Επιτρέπει τη πρόσβαση μίας εφαρμογής της Visual Basic σε μια βάση δεδομένων. Η πρόσβαση στα δεδομένα της βάσης γίνεται μέσω άλλων εργαλείων της Visual Basic που έχουν δυνατότητα σύνδεσης με τα πεδία της (bound controls).
- ✓ **Εργαλείο πλέγματος (Grid):** Η εμφάνισή του θυμίζει ένα φύλλο εργασίας του Excel. Δημιουργεί πίνακες μέσα στους οποίους μπορούμε να εμφανίσουμε ή να εισάγουμε πληροφορίες και να τις επεξεργαστούμε κατά την εκτέλεση της εφαρμογής.
- ✓ **Εργαλείο OLE (OLE Container):** Χρησιμοποιείται για την πραγματική διασύνδεση ή ενσωμάτωση σε μία Visual Basic εφαρμογή, αντικειμένων που έχουν δημιουργηθεί σε άλλες εφαρμογές μέσω του μηχανισμού OLE των Windows.

1.4. Προσθήκη εργαλείων

Η εργαλειοθήκη της Visual Basic εκτός από τα παραπάνω εσωτερικά εργαλεία όπως αναφέραμε μπορεί να εμπλουτιστεί και με επιπλέον εργαλεία σχεδιασμού τα ActiveX εργαλεία ή με ένθετα αντικείμενα (insertable objects) που έχουν δημιουργηθεί σε κάποια άλλη εφαρμογή.

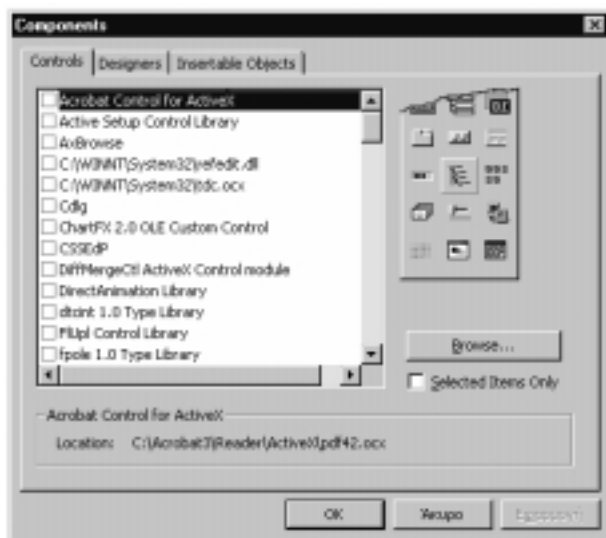
Η Visual Basic υποστηρίζει δύο τύπους ActiveX εργαλείων με επέκταση ονόματος .OCX. ή .VBX. Τα OCX ActiveX εργαλεία εκμεταλλεύονται την σύγχρονη OLE (Object Linking and Embedding) τεχνολογία και χρησιμοποιούνται στην 32-μπιτ αλλά και στη 16-μπιτ έκδοση της Visual Basic. Τα VBX εργαλεία είναι εργαλεία παλαιότερης τεχνολογίας, χρησιμοποιούνται μόνο στη 16-μπιτ έκδοση της Visual Basic και παραμένουν κυρίως για συμβατότητα με προηγούμενες εκδόσεις της.

Όταν ανοίξουμε μια εργασία που έχει δημιουργηθεί σε παλιότερη έκδοση της Visual Basic και περιέχει κάποιο VBX εργαλείο, η Visual Basic εξ' ορισμού αναβαθμίζει αυτόματα το συγκεκριμένο εργαλείο με την OCX έκδοση του αν φυσικά υπάρχει.

Τα ένθετα αντικείμενα, όπως αναφέραμε είναι αντικείμενα που έχουν κατασκευαστεί σε άλλη εφαρμογή (π.χ. ένα λογιστικό φύλλο του Excel) και μπορούμε να τα προσθέσουμε στην εργαλειοθήκη της Visual Basic. Τα εργαλεία αυτής της κατηγορίας αφού είναι δυνατό να προστεθούν στην εργαλειοθήκη μπορούμε να τα προσομοιώσουμε με τα ActiveX εργαλεία. Ορισμένα από αυτά τα εργαλεία επιπλέον υποστηρίζουν την τεχνολογία OLE Automation και έχουμε τη δυνατότητα να επέμβουμε προγραμματιστικά στα αντικείμενα άλλης εφαρμογής μέσα από μια Visual Basic εφαρμογή.

Από τη στιγμή που θα προσθέσουμε ένα ActiveX εργαλείο ή ένα ένθετο αντικείμενο στην εργαλειοθήκη της Visual Basic μπορούμε να το χειριστούμε σαν ένα εσωτερικό εργαλείο της γλώσσας.

Για να προσθέσουμε ένα ActiveX εργαλείο ή ένα ένθετο αντικείμενο στην εργαλειοθήκη χρησιμοποιούμε την επιλογή Components του μενού επιλογών Project της Visual Basic. Αμέσως θα εμφανιστεί στη οθόνη μας το πλαίσιο διαλόγου Components (σχήμα 6).



Σχ.6 Το πλαίσιο διαλόγου Components.

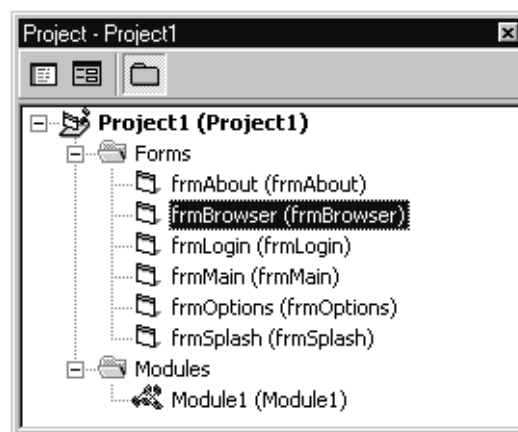
Στη λίστα των διαθέσιμων εργαλείων, επιλέγουμε το πλαίσιο ελέγχου που βρίσκεται αριστερά από το όνομά του εργαλείου που θέλουμε να προσθέσουμε και πατάμε το πλήκτρο εντολής OK. Το εργαλείο θα προστεθεί στην εργαλειοθήκη και μπορούμε να το χρησιμοποιήσουμε όπως ακριβώς κάνουμε με κάποιο από τα βασικά εργαλεία της Visual Basic.

Για να αφαιρέσουμε ένα ActiveX εργαλείο ή ένα ένθετο αντικείμενο πρέπει αρχικά να αφαιρέσουμε από

την εργασία όλα τα αντίγραφα του (instances) και τις εντολές κώδικα που αναφέρονται σε αυτά. Στη συνέχεια εμφανίζουμε πάλι το πλαίσιο διαλόγου Components, αποεπιλέγουμε το πλαίσιο ελέγχου που βρίσκεται αριστερά από το όνομά του και πατάμε το πλήκτρο εντολής OK. Προσοχή, αν δεν έχουμε αφαιρέσει από την εργασία μας τα αντίγραφα του αντικείμενου και τις αναφορές κώδικα που γίνονται σε αυτό, θα εμφανιστεί μήνυμα λάθους.

1.5. Ο εξερευνητής εργασίας

Κάθε εφαρμογή που δημιουργείται στο περιβάλλον της Visual Basic είναι δυνατό να περιέχει διαφορετικούς τύπους αρχείων. Το σύνολο όλων των αρχείων που αποτελούν μία εφαρμογή στη Visual Basic, κατά το σχεδιασμό της περιέχονται σε μια εργασία (Project). Ο εξερευνητής εργασίας (Σχήμα 7) περιέχει κάθε φορά τα ονόματα όλων των αρχείων της τρέχουσας εργασίας.



Σχ.7 Ο εξερευνητής εργασίας.

Αν δεν εμφανίζεται το παράθυρο εργασίας στην οθόνη, πρέπει να επιλέξουμε την επιλογή Project Explorer του μενού View.

Τα επιμέρους αρχεία που αποτελούν μία εργασία της Visual Basic αποθηκεύονται ξεχωριστά στο δίσκο του συστήματος. Η λίστα όλων των αρχείων που αποτελούν την εργασία καθώς και οι πληροφορίες για τη διεύθυνση τους επάνω στο δίσκο του συστήματος αποθηκεύονται στο **αρχείο εργασίας** (Project file). Ακόμη ένα αρχείο εργασίας είναι δυνατό να περιέχει τις αναφορές σύνδεσης (references) και πληροφορίες ή ειδικές ρυθμίσεις που τυχόν έχουμε κάνει για τη συγκεκριμένη εργασία, τα αντικείμενα που περιέχει, το περιβάλλον και το εκτελέσιμο αρχείο της.

Οι βασικοί τύποι αρχείων που συνήθως περιέχει μια εργασία και συνεπώς το παράθυρο εργασίας της Visual Basic είναι:

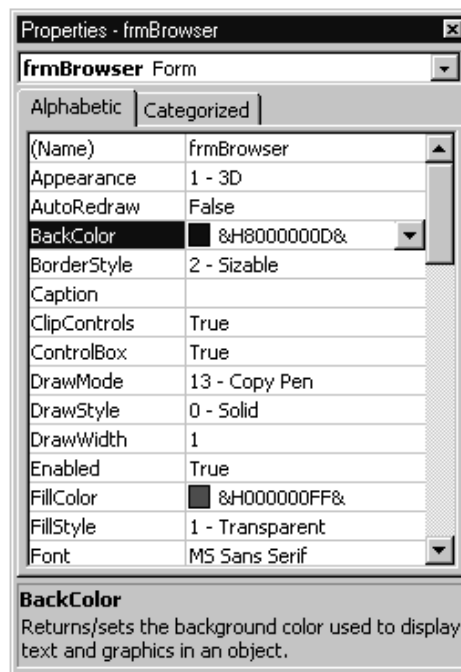
- ✓ **Αρχεία φόρμας (Form modules):** Περιλαμβάνουν τα οπτικά στοιχεία και τις ιδιότητες μίας φόρμας και των εργαλείων που περιέχει, καθώς και τις μεταβλητές, τις σταθερές και τις ρουτίνες κώδικα που συνοδεύουν τη φόρμα. Κάθε αρχείο φόρμας που δημιουργείται σε μία εφαρμογή της Visual Basic αποθηκεύεται ανεξάρτητα στο σκληρό δίσκο με επέκταση ονόματος .FRM.
- ✓ **Αρχεία βασικής προγραμματιστικής μονάδες (Standard modules):** Είναι αρχεία κώδικα, τα οποία περιέχουν τις ρουτίνες, τις συναρτήσεις και τις δηλώσεις και τους ορισμούς των γενικών μεταβλητών, των δεδομένων και των τύπων δεδομένων κάθε προγράμματος της Visual Basic. Τα αρχεία βασικής προγραμματιστικής μονάδας έχουν επέκταση ονόματος .BAS.
- ✓ **Αρχεία προγραμματιστικής μονάδας κλάσης (Class modules):** Είναι επίσης αρχεία κώδικα και τα χρησιμοποιούμε για να ορίσουμε καινούργια αντικείμενα. Τα αρχεία κλάσης είναι παρόμοια με τα αρχεία φόρμας, αλλά δεν περιέχουν οπτικά στοιχεία. Ένα αρχείο κλάσης είναι δυνατό να περιέχει ορισμούς ιδιοτήτων και μεθόδων του αντικειμένου που ορίζουμε, όχι όμως κάποια γεγονότα. Τα αρχεία κλάσης έχουν επέκταση ονόματος .CLS.
- ✓ **Αρχείο πηγής (Resource file):** Στη Visual Basic 4.0 έχουμε τη δυνατότητα όταν γράφουμε κώδικα να χρησιμοποιήσουμε δεδομένα από κάποιο αρχείο Πηγής. Τα αρχεία Πηγής είναι ανεξάρτητα αρχεία με επέκταση ονόματος .RES και περιέχουν διάφορους τύπους δεδομένων (π.χ. αρχεία bitmap ή τμήματα κειμένου), τα οποία μπορούμε να χρησιμοποιήσουμε χωρίς να χρειάζεται συνεχή επεξεργασία στον κώδικα τους. Με το τρόπο αυτό αυξάνονται οι δυνατότητες εκτέλεσης της εφαρμογής, γιατί φορτώνουμε κατά απαίτηση στη μνήμη του συστήματος τα δεδομένα που πραγματικά χρειαζόμαστε και όχι αναγκαστικά όλα όσα περιέχει μια φόρμα ή μια προγραμματιστική μονάδα.

1.6. Το παράθυρο ιδιοτήτων

Το παράθυρο ιδιοτήτων (Properties window), εμφανίζεται στην αρχική οθόνη της Visual Basic (Σχήμα 8) και περιέχει κάθε φορά μία λίστα με τις ιδιότητες του επιλεγμένου αντικειμένου (εργαλείο, φόρμα, ή κλάση), προγραμματιστικής μονάδας ή μενού επιλογών. Οι ιδιότητες είναι τα δεδομένα κάθε αντικειμένου και καθορίζουν τα χαρακτηριστικά της εμφάνισης του (π.χ. τη θέση ή το χρώμα) και τη συμπεριφορά του (π.χ. πως θα αντιδράσει στην εισαγωγή στοιχείων του χρήστη ή αν είναι ενεργό).

Οι τιμές των περισσότερων ιδιοτήτων ενός γραφικού αντικειμένου μπορούν να μεταβληθούν κατά το σχεδιασμό της εφαρμογής μέσω του παραθύρου ιδιοτήτων. Το παράθυρο ιδιοτήτων συνήθως βρίσκεται στο δεξί τμήμα της αρχικής οθόνης της Visual Basic και

μπορούμε να το μετακινήσουμε ή να μεταβάλλουμε τις διαστάσεις του με τη χρήση του ποντικιού όπως ένα οποιοδήποτε άλλο παράθυρο των Windows.



Σχ.8 Το παράθυρο ιδιοτήτων.

Αν το παράθυρο ιδιοτήτων δεν εμφανίζεται στην οθόνη, επιλέγουμε από το μενού View την επιλογή Properties Window ή πατάμε το πλήκτρο συντομίας F4 ή εναλλακτικά επιλέγουμε το αντίστοιχο εικονίδιο της γραμμής εργαλείων. Ακόμη το παράθυρο ιδιοτήτων μπορούμε να το εμφανίσουμε από το μενού συντομίας, που εμφανίζεται όταν πατήσουμε το δεξί πλήκτρο του ποντικιού ενώ βρισκόμαστε επάνω σε κάποιο γραφικό αντικείμενο της Visual Basic.

Στο επάνω τμήμα του παραθύρου ιδιοτήτων εμφανίζονται τα κλασικά στοιχεία ενός παραθύρου των Windows και στη συνέχεια υπάρχει το πλαίσιο των αντικειμένων (Object box) μέσα στο οποίο εμφανίζεται κάθε φορά το όνομα του ενεργού αντικειμένου της εργασίας. Πατώντας το βέλος που εικονίζεται στο δεξί τμήμα του πλαισίου αντικειμένων, εμφανίζεται μια λίστα που περιέχει την ενεργή φόρμα και όλα τα αντικείμενά της. Επιλέγοντας από τη λίστα ένα όνομα, το αντίστοιχο αντικείμενο έρχεται στο προσκήνιο και μετατρέπεται αυτόματα σε ενεργό αντικείμενο της εργασίας.

Στο κάτω τμήμα του παραθύρου εμφανίζονται δύο καρτέλες από τις οποίες επιλέγουμε τον τρόπο εμφάνισης των ιδιοτήτων, αλφαβητικά (Alphabetic) ή κατηγοριοποιημένες (Categorized). Αμέσως μετά, υπάρχει η λίστα ιδιοτήτων (Properties list), η οποία αποτελείται από δύο στήλες και περιέχει όλες τις ιδιότητες του ενεργού αντικειμένου και τις επιλεγμένες τιμές τους. Για

να μεταβάλουμε την τιμή μιας ιδιότητας, επιλέγουμε την ιδιότητα στη λίστα ιδιοτήτων και πληκτρολογούμε τη νέα τιμή στη πλαίσιο της δεξιάς στήλης που ονομάζεται πλαίσιο τιμών (Settings box). Σε ορισμένες ιδιότητες υπάρχουν προκαθορισμένες τιμές, οι οποίες εμφανίζονται με μορφή λίστας και μπορούμε να τις εμφανίσουμε και να επιλέξουμε κάποια τιμή αναδιπλώνοντας τα περιεχόμενά της λίστας, με το βελάκι που εμφανίζεται δεξιά από το πλαίσιο τιμών.

Στο σημείο αυτό πρέπει να σημειώσουμε ότι ορισμένα αντικείμενα της Visual Basic περιέχουν και ιδιότητες που δεν είναι διαθέσιμες κατά τη διάρκεια του σχεδιασμού αλλά παίρνουν τιμές μόνο μέσω εντολών κώδικα κατά τη διάρκεια της εκτέλεσης της εφαρμογής.

1.7. Το παράθυρο μορφής φόρμας

Το παράθυρο μορφής φόρμας (form layout window) μας επιτρέπει κατά το σχεδιασμό της εφαρμογής να ρυθμίσουμε οπτικά τη θέση της φόρμας (Σχήμα 9).

Στο παράθυρο μορφής φόρμας, κατά το χρόνο σχεδιασμού εμφανίζονται όλες οι φόρμες που είναι ορατές στο περιβάλλον εργασίας στην σωστή θέση και με τις αντίστοιχες διαστάσεις.



Σχ.9 Το παράθυρο μορφής φόρμας.

2. ΑΡΧΕΣ ΣΥΓΧΡΟΝΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Ως τώρα παρουσιάσαμε τα στοιχεία που αποτελούν το περιβάλλον ανάπτυξης εφαρμογών της Visual Basic. Το βασικότερο συμπέρασμα, είναι ότι στο περιβάλλον της Visual Basic τα εργαλεία προγραμματισμού διαφοροποιούνται από τα μέχρι σήμερα γνωστά εργαλεία προγραμματισμού του DOS και ακολουθούν τη φιλοσοφία του γραφικού περιβάλλοντος των Windows.

Στο περιβάλλον ανάπτυξης εφαρμογών της Visual Basic, βλέπουμε ότι εγκαταλείπεται ο ανιαρός τρόπος χειρισμού των εργαλείων μιας γλώσσας προγραμματισμού μέσω εντολών κώδικα. Χαρακτηριστικά μπορούμε να αναφέρουμε ότι δεν υπάρχει ένας απλός επεξεργαστής κειμένου μέσα στον οποίο αναπτύσσουμε την εφαρμογή, αλλά ένα πλήρες γραφικό περιβάλλον ανάπτυξης, μέσα στο οποίο χρησιμοποιούμε πρακτικά και εύκολα τα εργαλεία που προσφέρονται για το σχεδιασμό μιας εφαρμογής.

2.1. Εισαγωγή στον προγραμματισμό μιας εφαρμογής

Οι περισσότεροι από εμάς γνωρίζουμε ότι η φιλοσοφία ανάπτυξης μίας εφαρμογής διαφοροποιείται ανάλογα με το περιβάλλον μέσα στο οποίο πρόκειται να εργαστεί. Στο περιβάλλον των Windows ο προγραμματιστής πρέπει να έχει πάντοτε υπόψη, ότι πρόκειται να εργαστεί μέσα σε ένα γραφικό περιβάλλον. Άρα θα σχεδιάσει την εφαρμογή του βλέποντας την πάντοτε από την άποψη του χρήστη. Επίσης εκμεταλλευόμενος τα πλεονεκτήματα σχεδιασμού που προσφέρει το γραφικό περιβάλλον εργασίας των Windows πρέπει να δημιουργήσει ένα φιλικό και λειτουργικό τρόπο επικοινωνίας χρήστη-εφαρμογής.

Ο σχεδιασμός της εφαρμογής ξεκινά πάντοτε από τη δημιουργία του τρόπου επικοινωνίας χρήστη-εφαρμογής. Ο λόγος είναι προφανής. Σε μία εφαρμογή που εργάζεται στο περιβάλλον των Windows και συνεπώς και της Visual Basic, η ροή εκτέλεσής της καθορίζεται κάθε φορά από τις αντιδράσεις του χρήστη και του συστήματος. Ο χρήστης κάθε χρονική στιγμή είναι ο κυρίαρχος της εφαρμογής και μπορεί να επεμβαίνει και να καθοδηγεί τη ροή εκτέλεσής της, στέλνοντας σε αυτή ένα μήνυμα (message) ή προκαλώντας ένα γεγονός (event), μέσω των γραφικών αντικειμένων που αποτελούν το μέσο επικοινωνίας εφαρμογής-χρήστη. Επίσης στη ροή εκτέλεσης της εφαρμογής είναι πιθανό να επιδράσει και το ίδιο το σύστημα, αφού έχει τη δυνατότητα να προκαλέσει ένα εσωτερικό μήνυμα και να μεταβάλλει τις συνθήκες μέσα στο περιβάλλον εργασίας. Αφού η ροή εκτέλεσης μίας εφαρμογής εξαρτάται από την επικοινωνία του χρήστη και του συστήματος με τα αντικείμενα της εφαρμογής, εύκολα γίνεται κατανοητό ότι δε μπορούμε να ξεκινήσουμε το γράψιμο του κώδικα αν δε ξέρουμε πρώτα ποιες θα είναι οι πιθανές παρεμβάσεις τους κατά την εκτέλεση της εφαρμογής.

Σύμφωνα με τα παραπάνω μπορούμε να περιγράψουμε την ανάπτυξη μίας εφαρμογής με τη Visual Basic σαν διαδικασία τριών βημάτων:

- Σχεδιασμός του τρόπου επικοινωνίας χρήστη-εφαρμογής.
- Καθορισμός της αρχικής συμπεριφοράς των αντικειμένων της εφαρμογής μέσω των ιδιοτήτων που τα χαρακτηρίζουν.
- Δημιουργία και εκσφαλμάτωση των ρουτινών κώδικα.

Με τα δύο πρώτα βήματα δημιουργούμε τον τρόπο επικοινωνίας χρήστη-εφαρμογής και εισάγουμε τα αρχικά στοιχεία της εφαρμογής. Με το τρίτο βήμα γράφουμε τον κώδικα, ο οποίος θα καθοδηγεί την εφαρμογή και θα εκτελεί ακριβώς τις εργασίες που εμείς επιθυμούμε, μετά από κάποια αντίδραση του χρήστη ή του συστήματος. Για τη δημιουργία του κώδικα μίας εφαρμογής με τη Visual Basic συνδυάζονται οι τεχνικές του δομημένου (structural) αλλά και του τμη-

ματικού προγραμματισμού (modular). Ο δομημένος προγραμματισμός είναι λίγο πολύ γνωστός στους περισσότερους προγραμματιστές, αφού χρησιμοποιείται στην ανάπτυξη εφαρμογών στο περιβάλλον του DOS, με τις γλώσσες προγραμματισμού τρίτης γενιάς. Ο τμηματικός προγραμματισμός επιτυγχάνεται με τη χρήση δύο τεχνικών προγραμματισμού, του αντικειμενοστραφή (object-oriented) και του οδηγούμενου από τα γεγονότα (event-driven) προγραμματισμού. Οι δύο αυτές τεχνικές καθορίζουν και τον τρόπο με τον οποίο σχεδιάζεται μία εφαρμογή στο περιβάλλον της Visual Basic.

2.2. Αντικειμενοστραφής προγραμματισμός

Σε ένα πραγματικά αντικειμενοστραφές περιβάλλον, η εφαρμογή του αντικειμενοστραφή προγραμματισμού (Object-Oriented programming) γίνεται με τον κερματισμό της εφαρμογής σε αντικείμενα, καθένα από τα οποία αποτελεί ξεχωριστή οντότητα. Κάθε αντικείμενο περιέχει το δικό του κώδικα και δεδομένα και έχει ανεξάρτητη συμπεριφορά μέσα στο περιβάλλον εργασίας. Ο σωστός χειρισμός και η κατάλληλη συνεργασία των αντικειμένων του προγράμματος, θα επιφέρουν την επιτυχή ολοκλήρωση της εφαρμογής.

Η Visual Basic για να απλοποιήσει τον προγραμματισμό στο περιβάλλον των Windows περιέχει όπως είδαμε τα δικά της εσωτερικά αντικείμενα, μέσω των οποίων επιτυγχάνεται η εφαρμογή του αντικειμενοστραφή προγραμματισμού στο περιβάλλον της. Ας περιγράψουμε όμως τον κόσμο των αντικειμένων της Visual Basic. Στην καθημερινή μας ζωή όταν βλέπουμε ένα φυτό το αντιμετωπίζουμε σαν ένα απλό αντικείμενο. Αν όμως το προσέξουμε καλύτερα θα παρατηρήσουμε ότι το φυτό αποτελείται από περισσότερα αντικείμενα όπως τα φύλλα του, τη ρίζα του, τα λουλούδια του κ.ά. Στη συνέχεια μπορούμε να παρατηρήσουμε ότι το κάθε επιμέρους αντικείμενο περιέχει τα δικά του δεδομένα, δηλαδή τις δικές του ιδιότητες (π.χ. τα φύλλα έχουν πράσινο χρώμα) και τη δική του συμπεριφορά (π.χ. τα λουλούδια ανθίζουν). Ακόμη κάθε αντικείμενο έχει το δικό του τρόπο επικοινωνίας με το περιβάλλον (π.χ. τα φύλλα αναπνέουν μέσα από τους πόρους τους).

Με τον ίδιο τρόπο μπορούμε να περιγράψουμε μία εφαρμογή της Visual Basic. Εξετάζοντας προσεκτικά τη διαχείριση των προκαθορισμένων γραφικών αντικειμένων στο περιβάλλον της εφαρμογής θα παρατηρήσουμε ότι γίνεται περίπου με τον ίδιο τρόπο που διαχειριζόμαστε τα αντικείμενα του φυσικού περιβάλλοντος.

Πρώτα από όλα η ίδια η εφαρμογή μπορεί να χαρακτηριστεί σαν ένα ανεξάρτητο αντικείμενο. Ένα αντικείμενο περιέχει τα δικά του δεδομένα και το τρόπο επικοινωνίας για να επικοινωνεί με το περιβάλλον του. Τα στοιχεία αυτά είναι χαρακτηριστικά κάθε εφαρμο-

γής της Visual Basic. Στη συνέχεια μπορούμε να παρατηρήσουμε ότι η εφαρμογή αποτελείται από πολλά γραφικά αντικείμενα όπως φόρμες, πλήκτρα επιλογής κ.α. Αναλύοντας τα επιμέρους στοιχεία της εφαρμογής θα δούμε ότι και αυτά με τη σειρά τους μπορούν να χαρακτηριστούν αυτόνομα αντικείμενα, τα οποία περιέχουν τις δικές τους ιδιότητες και μεθόδους που καθορίζουν την εμφάνιση και τη συμπεριφορά τους μέσα στο περιβάλλον εργασίας και φυσικά το δικό τους μέσο επικοινωνίας, με το οποίο αναγνωρίζουν τα μηνύματα του χρήστη και του συστήματος.

Η Visual Basic περιέχει αρκετά ενσωματωμένα αντικείμενα. Κάθε αντικείμενο της Visual Basic, προέρχεται από μια κλάση. Μια κλάση, αντιπροσωπεύεται από ένα εργαλείο της εργαλειοθήκης ή δημιουργείται μέσω κώδικα. Όλα τα αντικείμενα, δημιουργούνται σαν ακριβή αντίγραφα ή στιγμιότυπα (instances) των κλάσεων τους. Όταν τα αντικείμενα αποκτήσουν ανεξάρτητη υπόσταση, μπορούμε να διαφοροποιήσουμε κάποια από τα χαρακτηριστικά τους. Έτσι τις περισσότερες φορές σε μια εφαρμογή συναντάμε αντικείμενα που προέρχονται από την ίδια κλάση αντικειμένων (π.χ. εργαλεία πλαισίου κειμένου), τα οποία έχουν διαφορετική εμφάνιση.

Κάθε αντικείμενο υποστηρίζει τις δικές του ιδιότητες (properties) και μεθόδους (methods). Ο προγραμματιστής χρειάζεται να διαχειριστεί αυτά τα χαρακτηριστικά για να συνδέσει τα αντικείμενα μεταξύ τους και να επιτύχει την αρμονική συνεργασία τους μέσα στην εφαρμογή. Οι ιδιότητες προσδιορίζουν την εμφάνιση των αντικειμένων της Visual Basic. Κάθε αντικείμενο περιέχει τις δικές του ιδιότητες. Για παράδειγμα μια φόρμα (Form) χαρακτηρίζεται από τις ιδιότητες AutoRedraw, BackColor, BorderStyle, Caption, Font, Name κ.α. Για να ορίσουμε τα χαρακτηριστικά ενός αντικειμένου πρέπει πρώτα φυσικά να δημιουργήσουμε το ίδιο. Στη συνέχεια αντιστοιχούμε σε αυτό τις ιδιότητες που θέλουμε να το περιγράψουν. Αν για παράδειγμα επιλέξουμε το εργαλείο πλήκτρο εντολής (command button) και το τοποθετήσουμε επάνω στη φόρμα, στη συνέχεια μπορούμε να ορίσουμε τον τίτλο του, τη θέση του μέσα στη φόρμα, τις διαστάσεις του κλπ.

Η τιμή μιας ιδιότητας ορίζεται κατά το σχεδιασμό της εφαρμογής μέσω του παραθύρου ιδιοτήτων (Properties window) της Visual Basic. Η μετατροπή της τιμής της όμως κατά την εκτέλεση της εφαρμογής μπορεί να γίνει μόνο μέσω εντολών κώδικα. Για να αναφερθούμε σε μια ιδιότητα ενός γραφικού αντικειμένου πρέπει πρώτα να αναφερθούμε στο ίδιο το αντικείμενο, γιατί όπως είναι φανερό δύο διαφορετικά αντικείμενα είναι δυνατό να περιέχουν την ίδια ιδιότητα. Για παράδειγμα, τα εργαλεία φόρμα και πλήκτρο εντολής υποστηρίζουν και τα δύο την ιδιότητα Caption (Ετικεταλίδι). Αν θέλουμε να αναφερθούμε στην ιδιότητα Caption της φόρμας χρησιμοποιούμε την εντολή:

```
Form1.Caption = "Πρώτη Φόρμα"
```

Όταν θέλουμε να αναφερθούμε στην ιδιότητα ενός αντικειμένου που βρίσκεται σε διαφορετική φόρμα εργασίας, πρέπει να αναφέρουμε και το όνομα της φόρμας με την παρακάτω σύνταξη:

Όνομα_φόρμας! Όνομα_αντικείμενου. Ιδιότητα

Ένα αντικείμενο εκτός από ιδιότητες υποστηρίζει και μεθόδους (methods). Οι μέθοδοι είναι στην πραγματικότητα ενέργειες που μπορούν να επιδράσουν στα αντικείμενα της εφαρμογής, μέσω μηνυμάτων που στέλνουν σε αυτά. Για να στείλει ο χρήστης ένα μήνυμα σε ένα αντικείμενο προσδιορίζει το όνομα του και τη μέθοδο μέσω της οποίας θα το στείλει. Για παράδειγμα αν θέλουμε να εισάγουμε σε ένα εργαλείο Λίστας (List), μια καινούρια επιλογή μέσω της μεθόδου AddItem χρησιμοποιούμε την εντολή:

List1.AddItem "Πρώτη επιλογή"

Μέσα στο περιβάλλον της Visual Basic υπάρχει η δυνατότητα να ανταλλάσσουν μηνύματα και δυο διαφορετικά αντικείμενα μιας εφαρμογής. Η ανταλλαγή των μηνυμάτων ανάμεσα στα δύο αντικείμενα επιτυγχάνεται επίσης μέσω των μεθόδων της Visual Basic. Για να αντιγράψουμε τα περιεχόμενα ενός Πλαισίου κειμένου (Text box) στο ειδικό αντικείμενο Clipboard χρησιμοποιούμε τη μέθοδο SetText:

Clipboard.SetText Text1.Text

Τέλος, μηνύματα μπορούν να ανταλλάσσουν και δυο διαφορετικές εφαρμογές μέσα στο περιβάλλον των Windows. Η εφαρμογή κατά την εκτέλεση της μπορεί να ανταλλάσσει μηνύματα με άλλες εφαρμογές μέσω του συστήματος επικοινωνίας API των Windows. Αν χρειαστεί για παράδειγμα μια εφαρμογή της Visual Basic, τη συνεργασία μίας άλλης εφαρμογής για την εκτέλεση μίας εργασίας, στέλνει ένα μήνυμα στην αντίστοιχη εφαρμογή, αυτή με τη σειρά της αναλαμβάνει τον έλεγχο του συστήματος και μόλις ολοκληρώσει την εργασία, απαντάει με τα αποτελέσματα της. Για να στείλουμε το περιεχόμενο ενός πλαισίου κειμένου, σε μια άλλη εφαρμογή των Windows (π.χ. Word) δημιουργούμε μια σύνδεση (DDE Link) ανάμεσα στις δύο εφαρμογές και χρησιμοποιούμε τη μέθοδο LinkSend:

Picture1.LinkSend

Η δημιουργία της διεπαφής χρήστη Visual Basic, χαρακτηρίζεται από τα αντικείμενα που το αποτελούν. Κατά το σχεδιασμό του θα πρέπει ο προγραμματιστής να έχει τη σκέψη του κυρίως στα αντικείμενα που θα συμπεριλάβει σε αυτό και στον τρόπο που θα μπορεί να μεταβάλλει τη συμπεριφορά τους.

Τα αντικείμενα που υποστηρίζει η Visual Basic είναι:

- ✓ Οι **φόρμες** αποτελούν τα παράθυρα επικοινωνίας της εφαρμογής με το χρήστη. Ένα παράθυρο μπορεί για παράδειγμα να είναι μία οθόνη εμφάνισης

στοιχείων ή ένα πλαίσιο διαλόγου του χρήστη με την εφαρμογή.

- ✓ Τα **εργαλεία** (controls), αποτελούν μαζί με τις φόρμες τα "υλικά" σχεδιασμού της εφαρμογής. Μια εφαρμογή της Visual Basic είναι δυνατό να περιέχει και άλλα αντικείμενα:
- ✓ Τα **ειδικά αντικείμενα** της Visual Basic δεν έχουν οπτική εμφάνιση αλλά χρησιμοποιούνται για την εκτέλεση ειδικών εργασιών. Τέτοια αντικείμενα είναι το Clipboard, ο Debugger, ο Εκτυπωτής και η Οθόνη. Τα αντικείμενα αυτά Η Visual Basic διαχειρίζεται τα ειδικά αντικείμενα μέσω των μεθόδων που υποστηρίζουν. Τα ειδικά αντικείμενα μπορεί να υποστηρίζουν μεθόδους, αλλά δεν έχουν δικές τους ιδιότητες. Για παράδειγμα έχουμε τη δυνατότητα να αντιγράψουμε δεδομένα στο αντικείμενο Clipboard, αλλά δεν μπορούμε να μετατρέψουμε τα χαρακτηριστικά του, γιατί δεν έχει οπτική υπόσταση.
- ✓ Τα **αντικείμενα δεδομένων** (Database objects). Κάθε βάση δεδομένων υποστηρίζει ένα αριθμό ειδικών αντικειμένων με το χειρισμό των οποίων μπορούμε να έχουμε πρόσβαση στα δεδομένα της.
- ✓ **Κλάσεις** (Classes). Η Visual Basic ως την τρίτη έκδοσή της δε μπορούσε να χαρακτηριστεί σαν μία πλήρη αντικειμενοστραφή γλώσσα προγραμματισμού, γιατί ο προγραμματιστής δεν ήταν δυνατό να δημιουργήσει στο περιβάλλον της τα δικά του αντικείμενα, ούτε μπορούσε να αντιστοιχίσει καινούριες ιδιότητες και μεθόδους στα ήδη υπάρχοντα. Από την τέταρτη έκδοση της Visual Basic, έχουμε τη δυνατότητα σε μια εφαρμογή να δημιουργήσουμε δικά μας αντικείμενα προσαρμοσμένα σε συγκεκριμένες ανάγκες, τα οποία θα περιέχουν τις δικές τους ιδιότητες και μεθόδους. Ο ορισμός ενός τέτοιου αντικειμένου, καλείται κλάση και πραγματοποιείται μέσα από αρχεία κώδικα.

Μία αντικειμενοστραφή εφαρμογή επιτρέπει στον προγραμματιστή της να χρησιμοποιεί αρκετές φορές τα ίδια αντικείμενα. Με την τεχνική αυτή γλιτώνει αρκετό χρόνο και απαλλάσσεται από την επανάληψη του κώδικα. Με την Visual Basic έχουμε τη δυνατότητα να επαναχρησιμοποιήσουμε όλα τα αντικείμενα της, ακόμη και αλλάζοντας μερικές μόνο από τις ιδιότητες τους.

2.3. Προγραμματισμός οδηγούμενος από τα γεγονότα

Στην καθημερινή μας ζωή, κάθε χρονική στιγμή είναι πιθανό να συμβεί ένα γεγονός το οποίο μπορεί να μεταβάλλει την κατάσταση των αντικειμένων που μας περιβάλλουν. Για παράδειγμα, το άνοιγμα ενός παραθύρου θα επιτρέψει την είσοδο του αέρα μέσα στο δωμάτιο και θα μετακινηθούν αρκετά αντικείμενα.

Με τον ίδιο τρόπο σε ένα αντικειμενοστραφές περιβάλλον όπως το περιβάλλον της Visual Basic, ένα γεγονός μπορεί να προκαλέσει τη μεταβολή της κατάστασης των γραφικών αντικειμένων και να επιδράσει στη ροή εκτέλεσης της εφαρμογής. Κατά την εκτέλεση της εφαρμογής ένα γεγονός μπορεί να προκληθεί από το χρήστη, όπως το πάτημα ενός πλήκτρου από το πληκτρολόγιο προκαλεί το KeyPress γεγονός, το σύστημα, όπως ένα χρονικό μήνυμα του εσωτερικού ρολογιού προκαλεί το Timer γεγονός, ή έμμεσα από το κώδικα όπως η εντολή με την οποία “φορτώνουμε” μια φόρμα προκαλεί το γεγονός Load.

Ο τρόπος για να καθοδηγούμε τη ροή εκτέλεσης μίας εφαρμογής της Visual Basic, είναι να εφαρμόσουμε τις αρχές του οδηγούμενου από τα γεγονότα προγραμματισμού. Ουσιαστικά πρέπει να γράψουμε τον κώδικα που θα κάνει τα γραφικά αντικείμενα να αντιδρούν στα γεγονότα, που πιθανών να συμβούν μέσα στο περιβάλλον εργασίας τους.

Ας δούμε όμως, πως εφαρμόζεται η τεχνική του οδηγούμενου από τα γεγονότα προγραμματισμού σε μία εφαρμογή της Visual Basic. Σε κάθε γραφικό αντικείμενο της Visual Basic μπορούμε να συμπεριλάβουμε εντολές κώδικα, οι οποίες θα εκτελούνται μετά από την επίδραση ενός γεγονότος. Ο κώδικας για κάθε γεγονός που πιθανών πρόκειται να συμβεί βρίσκεται σε ανεξάρτητες ρουτίνες, οι οποίες ονομάζονται ρουτίνες γεγονότων (event procedures).

Κάθε ρουτίνα γεγονότος χαρακτηρίζεται από το αντικείμενο και το γεγονός που θα την ενεργοποιήσει. Για παράδειγμα για να γράψουμε κώδικα για την περίπτωση που θέλουμε να εκτελεστεί μία εργασία όταν “φορτώνεται” μία φόρμα της εφαρμογής δημιουργούμε μία ρουτίνα η οποία θα τρέχει όταν αναγνωρίσει το γεγονός “φόρτωμα” (Load) της φόρμας. Η σύνταξη της ρουτίνας είναι ως εξής:

Sub Form_Load().

Το μεγάλο πλεονέκτημα της Visual Basic είναι ότι τα γεγονότα που συμβαίνουν μέσα στο περιβάλλον της, αναγνωρίζονται αυτόματα από τα γραφικά αντικείμενα που αποτελούν το μέσο επικοινωνίας της εφαρμογής και δε χρειάζεται να γράψουμε επιπλέον κώδικα για το σκοπό αυτό. Τα γραφικά αντικείμενα της Visual Basic δεν αντιδρούν σε κάθε γεγονός που προκαλείται στο περιβάλλον εργασίας τους, παρά μόνο σε αυτά που τα αφορούν, αφού κάθε γραφικό αντικείμενο της Visual Basic υποστηρίζει ένα προκαθορισμένο αριθμό γεγονότων (Events).

Το εργαλείο Πλαίσιο Ελέγχου (Check box) για παράδειγμα υποστηρίζει τα γεγονότα:

Click	KeyDown	KeyUp	KeyPress
DragOver	DragDrop	GotFocus	LostFocus

Κατά την ανάπτυξη μίας εφαρμογής μόλις σχεδιάσουμε ένα γραφικό αντικείμενο, η Visual Basic δημιουρ-

γεί αυτόματα στο παράθυρο κώδικα (code window) τα πρότυπα των ρουτινών γεγονότος για κάθε γεγονός που υποστηρίζει. Ο προγραμματιστής στη συνέχεια, επιλέγει τα γεγονότα που θέλει να συμπεριλάβει στην εφαρμογή και συμπληρώνει τις γραμμές κώδικα. Σε μία εφαρμογή δεν είναι απαραίτητο για κάθε γραφικό αντικείμενο να συμπεριλάβουμε κώδικα σε όλες τις ρουτίνες γεγονότων, αλλά επιλέγουμε και γράφουμε κώδικα μόνο για τα γεγονότα τα οποία μας ενδιαφέρουν και θέλουμε να αντιδρά η εφαρμογή.

Ας δούμε όμως πως αντιδρούν τα γραφικά αντικείμενα της Visual Basic όταν συμβεί ένα γεγονός στο περιβάλλον τους. Κατά την εκτέλεση της εφαρμογής τα γραφικά αντικείμενα που περιέχει βρίσκονται σε κατάσταση αναμονής. Μόλις συμβεί στο περιβάλλον ένα γεγονός, αναγνωρίζεται αυτόματα από το γραφικό αντικείμενο που το υποστηρίζει και η Visual Basic ελέγχει αν έχουμε γράψει κώδικα στην αντίστοιχη ρουτίνα γεγονότος. Αν υπάρχουν γραμμένες κάποιες εντολές κώδικα προχωράει στην εκτέλεση τους, διαφορετικά αγνοεί το γεγονός. Μετά την ολοκλήρωση της εκτέλεσης του κώδικα, τα γραφικά αντικείμενα επιστρέφουν πάλι σε κατάσταση αναμονής και η εφαρμογή περιμένει το επόμενο γεγονός να συμβεί.

Ακόμη πρέπει να τονίσουμε, ότι ο προγραμματιστής μπορεί να προκαλεί γεγονότα μέσω εντολών κώδικα. Για παράδειγμα όταν “φορτώνεται” μια φόρμα, μπορούμε να συμπεριλάβουμε στην αντίστοιχη ρουτίνα γεγονότος, μια εντολή κώδικα με την οποία θα μετατρέψουμε την τιμή της ιδιότητας Text και θα προκαλείται η αλλαγή της κατάστασης ενός πλαισίου κειμένου. Τα γεγονότα που προκαλούνται μέσα από κώδικα καλούνται trigger events.

Ένα εξωτερικό μήνυμα, μπορεί να προκαλέσει δύο ή περισσότερα διαφορετικά γεγονότα στην εφαρμογή. Για παράδειγμα όταν συμβεί το γεγονός Dblick (διπλό πάτημα) του αριστερού πλήκτρου του ποντικιού, συγχρόνως συμβαίνουν και τα γεγονότα Click, MouseDown και MouseUp. Έτσι η Visual Basic θα ανιχνεύσει για εντολές κώδικα σε περισσότερες από μία ρουτίνες γεγονότων.

Θα μπορούσαμε λοιπόν να παρουσιάσουμε μία εφαρμογή που ακολουθεί την τεχνική του οδηγούμενου από τα γεγονότα προγραμματισμού, με τα παρακάτω βήματα:

1. Η εκτέλεση της εφαρμογής ξεκινά και “φορτώνεται” η πρώτη φόρμα της.
2. Τα γραφικά αντικείμενα της εφαρμογής βρίσκονται σε κατάσταση αναμονής.
3. Όταν ένα γραφικό αντικείμενο αναγνωρίσει ένα γεγονός που το αφορά, καλεί την ανάλογη ρουτίνα γεγονότος.
4. Αν η ρουτίνα γεγονότος περιέχει κώδικα εκτελείται, διαφορετικά η εφαρμογή αγνοεί το γεγονός.
5. Τα αντικείμενα της εφαρμογής, περνούν πάλι σε κατάσταση αναμονής.

Ένα σημείο, που πρέπει να προσέξουμε ιδιαίτερα στην εφαρμογή του οδηγούμενου από τα γεγονότα προγραμματισμού, είναι η αποφυγή της εκτέλεσης μιας ενέργειας μέσα από μια ρουτίνα γεγονότος που θα προκαλεί το ίδιο γεγονός. Αυτό το γεγονός ονομάζεται cascading event και θα έχει ως συνέπεια τη συνεχή εκτέλεση της ίδιας ρουτίνας, μέχρι να προκληθεί πρόβλημα μνήμης στη Visual Basic (Out of stack space error). Για παράδειγμα, όταν συμπεριλάβουμε στην ρουτίνα γεγονότος αλλαγή κατάστασης ενός πλαισίου κειμένου, μια εντολή που θα μετατρέπει την ιδιότητά του Text, θα έχουμε επανάληψη του ίδιου γεγονότος και θα καλείται συνεχώς η ίδια ρουτίνα.

3. Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Αν συγκρίνουμε μία παραδοσιακή Basic που εργάζεται στο περιβάλλον του DOS με τη Visual Basic, η βασικότερη διαφορά στην οποία πρέπει να σταθούμε βρίσκεται στη φιλοσοφία προγραμματισμού. Όπως είδαμε, ο σχεδιασμός μίας εφαρμογής με τη Visual Basic στηρίζεται στην τεχνική του αντικειμενοστραφή και του οδηγούμενου από τα γεγονότα προγραμματισμού. Ο λόγος είναι ότι η εφαρμογή πρόκειται να εργαστεί μέσα σε ένα αντικειμενοστραφές περιβάλλον, όπως των Windows όπου η ροή εκτέλεσης κάθε εφαρμογής προσδιορίζεται από τα γεγονότα που συμβαίνουν κάθε χρονική στιγμή και τα μηνύματα που δέχεται ο τρόπος διασύνδεσης και επικοινωνίας της, μέσα στο περιβάλλον εργασίας. Αντιθέτως μία εφαρμογή που έχει δημιουργηθεί σε μια κλασική Basic και εργάζεται στο περιβάλλον του DOS, ακολουθεί πάντοτε ένα ιεραρχικά δομημένο σχεδιασμό και η ροή εκτέλεσης της εξαρτάται αποκλειστικά και μόνο από τις κλήσεις των ρουτινών κώδικα του προγράμματος.

Όμως τα δύο περιβάλλοντα προγραμματισμού, παρά τις διαφορές που παρουσιάζουν στη φιλοσοφία σχεδιασμού μιας εφαρμογής, όπως προδίδει το όνομα τους έχουν και πολλά κοινά χαρακτηριστικά. Το πιο σημαντικό κοινό σημείο τους, είναι ο κώδικας που δημιουργεί ένας προγραμματιστής για τη κατασκευή ενός προγράμματος. Η φιλοσοφία δημιουργίας των ρουτινών ενός προγράμματος παραμένει σταθερή, αφού η ανάπτυξη του κώδικα της Visual Basic ακολουθεί επίσης τις τεχνικές του δομημένου προγραμματισμού. Οι εντολές κώδικα που χρησιμοποιεί η Visual Basic, είναι σχεδόν όλες ίδιες με τις εντολές μίας παραδοσιακής Basic (π.χ QuickBasic) και φυσικά γνώριμες στους περισσότερους προγραμματιστές. Ακόμη η διαχείριση και ο ορισμός των μεταβλητών και των δομών δεδομένων που δημιουργούμε μέσα σε ένα πρόγραμμα, γίνεται με παρόμοιο τρόπο και στα δύο περιβάλλοντα προγραμματισμού. Τέλος αρκετές τεχνικές που εφαρμόζονται κατά την ανάπτυξη των εφαρμογών είναι ίδιες, όπως για παράδειγμα η διαχείριση των αρχείων και οι δομές ελέγχου του προγράμματος.

Στη συνέχεια του κεφαλαίου θα παρουσιάσουμε τη δομή που πρέπει να έχει μια εφαρμογή που σχεδιάζουμε με τη Visual Basic και θα αναφέρουμε τα βασικά σημεία του κώδικα, που πρέπει να έχει σαν εφόδια ένας προγραμματιστής για τη δημιουργία ενός προγράμματος.

3.1. Δομή μίας Visual Basic εφαρμογής

Οι προγραμματιστές που εργάζονται στο περιβάλλον της Visual Basic πρέπει να σχεδιάζουν την εφαρμογή τους, εφαρμόζοντας τις κατάλληλες τεχνικές που θα της δίνουν τη δυνατότητα να εργάζεται μέσα σε ένα παράθυρο των Windows. Κατά την ανάπτυξη μιας εφαρμογής με τη Visual Basic τα πιο σημαντικά στοιχεία επεξεργασίας είναι, η επικοινωνία της με το χρήστη και το περιβάλλον, ο χειρισμός και η αποτελεσματική επεξεργασία των στοιχείων εισόδου της (δεδομένα) και τέλος η μορφοποίηση των στοιχείων εξόδου.

Ένας προγραμματιστής για να επιτύχει την επίλυση ενός προβλήματος με τη Visual Basic, πρέπει να δημιουργήσει μέσα σε μια εφαρμογή τα παρακάτω τμήματα:

- ✓ Το τρόπο επικοινωνίας της εφαρμογής με το περιβάλλον εργασίας και το χρήστη
- ✓ Το χειρισμό των στοιχείων εισόδου (Data Input)
- ✓ Τα τμήματα κώδικα (ρουτίνες προγράμματος), που θα επεξεργαστούν τα στοιχεία εισόδου και θα δημιουργήσουν τα αποτελέσματα της εφαρμογής
- ✓ Τη μορφοποίηση και εξαγωγή των αποτελεσμάτων (Data Output)

Το τμήμα που είναι καινούριο για ένα προγραμματιστή, είναι η διασύνδεση της εφαρμογής με το σύστημα και το χρήστη. Όπως είδαμε, με τη Visual Basic η εργασία αυτή έχει απλοποιηθεί και στη πραγματικότητα θυμίζει τη χρήση ενός σχεδιαστικού προγράμματος. Η δημιουργία της διεπαφής χρήστη δεν ξεκινάει από το μηδέν, αλλά ο προγραμματιστής χρησιμοποιεί τα προκαθορισμένα γραφικά εργαλεία της (controls) και σχεδιάζει πολύ εύκολα και γρήγορα το περιβάλλον εργασίας της εφαρμογής.

Τα τρία τελευταία στοιχεία της εφαρμογής, είναι ήδη γνωστά στους περισσότερους προγραμματιστές από το περιβάλλον του DOS και η δημιουργία τους γίνεται μέσω εντολών κώδικα της Visual Basic. Το μοναδικό σημείο που πρέπει να προσέξει ιδιαίτερα ένας προγραμματιστής της Visual Basic κατά τη δημιουργία του κώδικα του προγράμματος, είναι ότι χρειάζεται να χωρίσει την εφαρμογή σε διακεκριμένα τμήματα, για να μπορεί να εκμεταλλευτεί τις αντιδράσεις των γραφικών αντικειμένων της εφαρμογής στα γεγονότα και τα μηνύματα που δέχεται κάθε χρονική στιγμή το περιβάλλον επικοινωνίας της.

3.2. Τα τμήματα κώδικα

Όπως περιγράψαμε, μία εφαρμογή της Visual Basic αποτελείται από το τρόπο επικοινωνίας του χρήστη με την εφαρμογή και το κώδικα που περιέχει. Τα γραφικά αντικείμενα διασύνδεσης που περιέχει η εφαρμογή, είναι τα στοιχεία που προσδιορίζουν τα αρχικά χαρακτηριστικά του περιβάλλοντος επικοινωνίας της. Η ουσία όμως, κάθε εφαρμογής στο περιβάλλον της Visual Basic είναι πάντοτε ο κώδικας που περιέχει. Οι εντολές κώδικα που περιέχουν οι ρουτίνες μιας εφαρμογής, καθορίζουν τη ροή εκτέλεσής της και προσδιορίζουν ακριβώς τις εργασίες που θα εκτελεί.

Σε μία εφαρμογή της Visual Basic μπορούμε να χρησιμοποιήσουμε μόνο μία φόρμα και όλες οι εντολές κώδικα να αφορούν εργασίες της συγκεκριμένης φόρμας. Σχεδιάζοντας όμως, μία μεγαλύτερη εφαρμογή, για να αποφύγουμε τη δημιουργία ενός πολύπλοκου και δύσχειρου περιβάλλοντος επικοινωνίας χρήστη-εφαρμογής, σίγουρα θα προσθέσουμε και άλλες φόρμες εργασίας και συνεπώς θα χρειαστούμε περισσότερες εντολές κώδικα για το χειρισμό τους. Επίσης, όσο μεγαλώνει η εφαρμογή εμφανίζονται σημεία, που πρέπει να διαχειριστούν κοινά τμήματα κώδικα και παρουσιάζεται η ανάγκη για γενικές μεταβλητές που θα ανταλλάσσουν τα δεδομένα τους μέσα σε διαφορετικές ρουτίνες της εφαρμογής.

Για το σκοπό αυτό η Visual Basic μας δίνει τη δυνατότητα να δημιουργήσουμε μέσα σε ένα πρόγραμμα ανεξάρτητα τμήματα κώδικα, τα οποία ονομάζονται προγραμματιστικές μονάδες (modules). Ο διαχωρισμός των προγραμματιστικών μονάδων γίνεται με την εμβέλεια και τη χρήση τους μέσα σε μια εφαρμογή. Σε ένα πρόγραμμα είναι δυνατό να υπάρχουν βασικές προγραμματιστικές μονάδες (standard modules) και προγραμματιστικές μονάδες φόρμας (form modules) και κλάσης (class modules). Οι προγραμματιστικές μονάδες φόρμας, στη πραγματικότητα είναι προγραμματιστικές μονάδες κλάσης, αλλά περιέχουν και οπτικά στοιχεία, όπως τις γραφικές περιγραφές του παραθύρου φόρμας και των αντικειμένων που υπάρχουν στην επιφάνειά του. Μια προγραμματιστική μονάδα είναι δυνατό να περιέχει ένα τμήμα δηλώσεων μεταβλητών και ρουτίνες κώδικα.

3.2.1. Τμήματα Δηλώσεων μεταβλητών

Κάθε προγραμματιστική μονάδα (βασική, φόρμας ή κλάσης), περιέχει ένα τμήμα δηλώσεων μεταβλητών (Declarations section). Στο τμήμα αυτό μπορούμε να συμπεριλάβουμε γενικές δηλώσεις μεταβλητών (Declarations). Οι δηλώσεις μεταβλητών είναι μη εκτελέσιμες εντολές κώδικα και τις χρησιμοποιούμε για να αποδώσουμε κάποιο όνομα και να ορίσουμε τα χαρακτηριστικά (π.χ. τύπο δεδομένων) στις εξωτερικές ρουτίνες, στις σταθερές, στις δομές δεδομένων και στα ορίσματα κλήσης DLL αρχείων του προγράμματος.

3.2.2. Ρουτίνες κώδικα

Το μεγάλο πλεονέκτημα του κώδικα της Visual Basic, είναι ότι μπορούμε να διασπάσουμε ένα πρόγραμμα, σε μικρότερα λογικά τμήματα που καλούνται ρουτίνες (procedures). Οι ρουτίνες περιέχουν εντολές κώδικα, που εκτελούν συγκεκριμένες εργασίες οι οποίες επαναλαμβάνονται μέσα σε μια εφαρμογή, όπως αριθμητικούς υπολογισμούς ή την διαχείριση των δεδομένων εισόδου.

Οι ρουτίνες, μας επιτρέπουν να απλοποιήσουμε τις προγραμματιστικές εργασίες, αφού είναι ευκολότερη η τμηματική δημιουργία και πιο αξιόπιστη η αποσφαλμάτωση του προγράμματος. Επίσης είναι δυνατό να τις χρησιμοποιήσουμε αυτούσιες ή με μικρές αλλαγές και σε άλλα προγράμματα με παρόμοιες εργασίες.

Σε ένα πρόγραμμα της Visual Basic, είναι δυνατό να συμπεριλάβουμε τους παρακάτω τύπους ρουτινών :

➤ Sub ρουτίνες

Οι Sub ρουτίνες, είναι τμήματα κώδικα που εκτελούν συγκεκριμένη εργασία μέσα σε ένα πρόγραμμα. Μια Sub ρουτίνα δεν επιστρέφει κάποια τιμή στο σημείο κλήσης της και ο κώδικάς της ξεκινά πάντοτε με την εντολή Sub και τελειώνει με την εντολή End Sub. Σε μια εφαρμογή είναι δυνατό να έχουμε γενικές ρουτίνες (general procedures) και ρουτίνες γεγονότων (event procedures).

Οι γενικές ρουτίνες, δεν έχουν σχέση με τα γεγονότα που συμβαίνουν στο περιβάλλον εργασίας της εφαρμογής, αλλά αποτελούν ανεξάρτητα τμήματα του κώδικά της. Στις γενικές ρουτίνες συμπεριλαμβάνουμε εργασίες οι οποίες είναι δυνατό να εκτελούνται από διαφορετικά σημεία του προγράμματος. Με τη χρήση των γενικών ρουτινών έχουμε τη δυνατότητα να τοποθετούμε τις κοινές εντολές κώδικα σε ένα σημείο, αποφεύγοντας την επανάληψή τους σε διαφορετικά σημεία του προγράμματος.

Οι ρουτίνες γεγονότων, περιέχουν εντολές κώδικα οι οποίες εκτελούνται αυτόματα, μόλις ένα γραφικό αντικείμενο της διεπαφής χρήστη αναγνωρίσει ένα γεγονός που το αφορά μέσα στο περιβάλλον εργασίας. Οι ρουτίνες γεγονότων δεν αποτελούν ανεξάρτητο τμήμα κώδικα, αλλά ανήκουν απαραίτητα στο κώδικα που περιέχει μία προγραμματιστική μονάδα φόρμας της εφαρμογής.

➤ Ρουτίνες συνάρτησης

Η Visual Basic μας παρέχει ένα αριθμό έτοιμων ρουτινών συνάρτησης (Function procedures), που μπορούμε να χρησιμοποιήσουμε μέσα σε ένα πρόγραμμά μας, όπως τις συναρτήσεις Sqr ή Chr. Ένα από τα πλεονέκτημα της όμως, είναι ότι μας δίνει τη δυνατότητα να χρησιμοποιήσουμε επιπλέον ρουτίνες συνάρτησης που θα προέρχονται από άλλες εφαρμογές ή θα τις δημιουργήσουμε μόνοι μας σύμφωνα με τις ανάγκες των προγραμμάτων μας.

⇒ Ρουτίνες ιδιοτήτων

Οι ρουτίνες ιδιοτήτων (property procedures), χρησιμοποιούνται για τη δημιουργία και το χειρισμό νέων εξειδικευμένων ιδιοτήτων. Όταν δημιουργούμε μια ιδιότητα, αυτή είναι χαρακτηριστικό της προγραμματιστικής μονάδας που περιέχει τη ρουτίνα δημιουργίας της. Με τις ρουτίνες ιδιοτήτων μπορούμε να δημιουργήσουμε ιδιότητες, για κάθε τύπο προγραμματιστικής μονάδας. Στις ιδιότητες αυτές, ο χρήστης θα έχει δικαίωμα μόνο ανάγνωσης (read-only) και έχουμε τη δυνατότητα να ορίσουμε εντολές κώδικα που θα εκτελούνται αυτόματα μόλις οριστεί η τιμή της ιδιότητας.

3.3. Σταθερές

Σε πολλά σημεία ενός προγράμματος της Visual Basic, είναι δυνατό να συναντάμε μια σταθερή τιμή η οποία θα επαναλαμβάνεται συνεχώς. Αρκετές φορές οι τιμές αυτές είναι πολύπλοκες για να απομνημονευτούν ή δε δείχνουν αντιπροσωπευτικά τη σημασία τους μέσα στην εφαρμογή. Για το σκοπό αυτό η Visual Basic επιτρέπει στον προγραμματιστή τη χρήση συμβολικών Σταθερών (Constants) μέσα σε ένα πρόγραμμα της.

Μια συμβολική σταθερά πρέπει να έχει πάντοτε ένα αντιπροσωπευτικό όνομα για να είναι εύκολα αναγνώσιμος ο κώδικας του προγράμματος. Η τιμή μιας συμβολικής σταθεράς δε μπορεί να αλλάξει κατά την εκτέλεση ενός προγράμματος.

Μέσα σε ένα πρόγραμμα της Visual Basic, μπορούμε να χρησιμοποιήσουμε έτοιμες συμβολικές σταθερές του συστήματος που παρέχονται από τη Visual Basic ή άλλες εφαρμογές και εργαλεία. Οι συμβολικές σταθερές συστήματος, βρίσκονται στις βιβλιοθήκες αντικειμένων του παράθυρου Επισκόπησης Αντικειμένων (Object Browser) και μπορούμε εφόσον είναι ενεργό το παράθυρο κώδικα να τις επικολλήσουμε και να τις χρησιμοποιήσουμε με το πλήκτρο Paste, σε οποιοδήποτε τμήμα του προγράμματός μας.

Ακόμη η Visual Basic μας δίνει τη δυνατότητα να δημιουργήσουμε τις δικές μας συμβολικές σταθερές, σύμφωνα με τις ανάγκες του προγράμματός μας. Αν θέλουμε να ορίσουμε μόνοι μας μία συμβολική σταθερά, πρέπει να χρησιμοποιήσουμε την εντολή Const:

[Private|Public]Const όνομα_σταθεράς[As τύπος_δεδομένων]=έκφραση

Στον τύπο ορισμού μιας συμβολικής σταθεράς η τιμή της έκφρασης μπορεί να είναι αριθμητική, αλφαριθμητική ή τύπου Ημερομηνίας:

Const Pi = 3.1415

Const Dealer = "Αποστόλου"

Const HireDate = #1/3/92#

Η εμβέλεια μιας συμβολικής σταθεράς μέσα σε ένα πρόγραμμα της Visual Basic καθορίζεται από τον τύπο της εντολής και το τμήμα κώδικα στο οποίο δηλώθηκε:

- ✓ Για να ορίσουμε μια τοπική συμβολική σταθερά, απλά χρησιμοποιούμε την εντολή Const μέσα σε μια τοπική ρουτίνα.
- ✓ Για να ορίσουμε μια συμβολική σταθερά επιπέδου προγραμματιστικής μονάδας, ορίζουμε τη σταθερά με την εντολή Const στο τμήμα δηλώσεων (Declarations section) της προγραμματιστικής μονάδας.
- ✓ Τέλος για να ορίσουμε μια καθολική συμβολική σταθερά, στην τιμή της οποίας θα έχουν πρόσβαση όλα τα τμήματα κώδικα της εφαρμογής, χρησιμοποιούμε την εντολή Const με το δηλωτικό Public και ορίζουμε τη σταθερά στο τμήμα δηλώσεων (Declarations section) μιας βασικής προγραμματιστικής μονάδας (standard module):

Public Const Pi=3.14

Οι σταθερές που εμφανίζονται στο παράθυρο Επισκόπησης Αντικειμένων, δε χρειάζεται να οριστούν πάλι στο πρόγραμμά μας με την εντολή Const.

3.4. Μεταβλητές

Οι μεταβλητές είναι θέσεις μνήμης του συστήματος στις οποίες αποθηκεύονται προσωρινά δεδομένα κατά την εκτέλεση μίας εφαρμογής. Στη Visual Basic κάθε μεταβλητή χαρακτηρίζεται από το όνομά της και τον τύπο δεδομένων που υποστηρίζει. Το όνομα μίας μεταβλητής πρέπει να ξεκινά πάντοτε με αλφαριθμητικό στοιχείο, δεν πρέπει να είναι κάποια από της δεσμευμένες λέξεις της Visual Basic (π.χ Sub ή End) και το μήκος του δεν πρέπει να ξεπερνά τους 255 χαρακτήρες. Σε μια εφαρμογή της Visual Basic είναι δυνατό να έχουμε Αλφαριθμητικές μεταβλητές (String) με περιεχόμενα γράμματα και αριθμούς, Αριθμητικές μεταβλητές (Numeric) με περιεχόμενα αριθμητικές τιμές, Λογικές μεταβλητές (Logical) οι οποίες μπορούν να περιέχουν τις τιμές Αληθής (True) ή Ψευδής (False), μεταβλητές Ημερομηνίας (Date) οι οποίες μπορούν να περιέχουν δεδομένα ημερομηνίας ή ώρας και τέλος μεταβλητές Αντικειμένων (Object) οι οποίες αναφέρονται σε αντικείμενα της εφαρμογής.

Όταν δηλώνουμε μια μεταβλητή μέσα σε ένα πρόγραμμα της Visual Basic έχουμε τη δυνατότητα να ορίσουμε τον τύπο των δεδομένων (Πίνακας 1), που πρόκειται να αποθηκεύσει, όταν θα εκτελείται η εφαρμογή. Με τον τρόπο αυτό μπορούμε να επιτύχουμε καλύτερη διαχείριση της μνήμης του συστήματος και συνεπώς τις βέλτιστες επιδόσεις για τις εφαρμογές που δημιουργούμε.

Τύπος μεταβλητής	Αρ.bytes	Εύρος τιμών
Ακέραιος (Integer)	2	-32768 έως +32767
Μεγάλος Ακέραιος (Long)	4	-2,147,483,648 έως 2,147,483,647
Απλής Ακρίβειας (Single)	4	(αρνητικές τιμές) -3.402823E38 έως -1.401298E-45 (θετικές τιμές) 1.401298E-45 έως 3.402823E38
Διπλής Ακρίβειας (Double)	8	(αρνητικές τιμές) -1.79769313486231E308 έως -4.94065645851247E-324 (θετικές τιμές) 4.94065645851247E-324 έως 1.79769313486231E308
Νομίσματος (Currency)	8	-9.22337203685477.5808 έως 9.22337203685477.5807
Αλφαριθμητική (String)	1byte/ χαρακτ.	0 έως 65000 χαρακτήρες 0 έως 2E32 σε 32-bit συστήματα
Ψηφιολέξη (Byte)	1	0 έως 255
Λογική (Boolean)	2	True ή False
Ημερομηνίας (Date)	8	January 1, 100 to December 31, 9999
Αντικειμένου (Object)	4	Οποιαδήποτε αναφορά αντικειμένου

Πίνακας 1: Οι τύποι μεταβλητών της Visual Basic.

Εκτός από τους παραπάνω τύπους μεταβλητών, η Visual Basic υποστηρίζει ακόμη και ένα ειδικό τύπο μεταβλητής τη Variant. Σε μία μεταβλητή τύπου Variant, μπορούμε να αποθηκεύσουμε διαφορετικούς τύπους δεδομένων.

Το μεγαλύτερο πλεονέκτημα μιας μεταβλητής τύπου Variant είναι ότι, μας επιτρέπει να κάνουμε πράξεις με δεδομένα διαφορετικού τύπου (format), χωρίς καμία μετατροπή. Μια Variant μεταβλητή δεσμεύει 16 bytes για αριθμητικές και 1 byte ανά χαρακτήρα για αλφαριθμητικές τιμές. Η Visual Basic όταν συναντά δεδομένα διαφορετικού τύπου, κάνει αυτόματη μετατροπή και εκτελεί την πράξη. Φυσικά όταν κάνουμε πράξεις με μία Variant μεταβλητή πρέπει να είμαστε ιδιαίτερα προσεκτικοί, γιατί οι πράξεις ορίζονται διαφορετικά για κάθε τύπο δεδομένων που χρησιμοποιούμε. Για παράδειγμα ο χαρακτήρας της πρόσθεσης (+), σε αλφαριθμητικές μεταβλητές σημαίνει ένωση των μεταβλητών, ενώ σε αριθμητικές σημαίνει το άθροισμά τους.

Αν μέσα σε ένα πρόγραμμα δεν ορίσουμε τον τύπο μίας μεταβλητής, η Visual Basic την ορίζει αυτόματα σαν τύπου Variant. Επειδή όμως ο συγκεκριμένος τύπος μεταβλητής δεσμεύει αρκετή μνήμη του συστήματος, είναι προτιμότερο αν θέλουμε να έχουμε ευέλικτες και γρήγορες εφαρμογές, να ορίζουμε τον τύπο κάθε μεταβλητής ανάλογα με τα δεδομένα που πρόκειται να αποθηκεύει. Για παράδειγμα, αν είμαστε σίγουροι ότι μια μεταβλητή σε όλη τη διάρκεια εκτέλεσης της εφαρμογής θα περιέχει μικρές αριθμητικές τιμές, είναι σκόπιμο να την ορίσουμε σαν τύπου Integer οπότε θα δεσμεύει κάθε φορά μόνο δύο bytes μνήμης.

Αν θέλουμε υποχρεωτικά να ορίζουμε κάθε μεταβλητή και τον τύπο δεδομένων της, πριν τη χρησιμοποιήσουμε σε μια προγραμματιστική μονάδα της Visual Basic πρέπει να συμπεριλάβουμε στο κώδικά της την εντολή Option Explicit. Η εντολή Option Explicit, πρέπει να τοποθετηθεί στο τμήμα δηλώσεων της προγραμματιστικής μονάδας, πριν από οποιαδήποτε δηλωτική εντολή μεταβλητής ή συμβολικής σταθεράς. Όταν έχουμε συμπεριλάβει στο κώδικα της προγραμματιστικής μονάδας την εντολή Option Explicit, αν προσπαθήσουμε να χρησιμοποιήσουμε μια μεταβλητή χωρίς πρώτα να την έχουμε δηλώσει θα εμφανιστεί μήνυμα λάθους. Με τη τεχνική αυτή, ανιχνεύουμε τυχόν λάθη στη πληκτρολόγηση των ονομάτων των μεταβλητών κατά τη συγγραφή του κώδικα.

3.5. Ορισμός μεταβλητών

Σε μία εφαρμογή της Visual Basic έχουμε τη δυνατότητα να δημιουργήσουμε μεταβλητές με διαφορετική εμβέλεια (scope). Η εμβέλεια μιας μεταβλητής, ορίζει ποια σημεία της εφαρμογής θα έχουν τη δυνατότητα πρόσβασης στα περιεχόμενα της. Ο ορισμός μιας μεταβλητής στη Visual Basic εξαρτάται κάθε φορά από την εμβέλεια που πρόκειται να έχει μέσα στην εφαρμογή που θα χρησιμοποιηθεί αλλά και από τον τύπο των δεδομένων που πρόκειται να αποθηκευτούν σε αυτή.

Στη Visual Basic η δήλωση μιας μεταβλητής γίνεται με δηλωτικές εντολές της μορφής:

Εντολή Όνομα_μεταβλητής As τύπος_δεδομένων.

Η δηλωτική Εντολή που χρησιμοποιούμε κάθε φορά, εξαρτάται από το σημείο κώδικα στο οποίο θα δηλώσουμε τη μεταβλητή και καθορίζει την εμβέλεια της μέσα στην εφαρμογή. Ο τύπος δεδομένων μπορεί είναι κάποια από τις τιμές του Πίνακα 1. της προηγούμενης παραγράφου ή η τιμή Variant.

Οι μεταβλητές της Visual Basic χωρίζονται ανάλογα με την εμβέλεια τους μέσα στην εφαρμογή σε μεταβλητές τοπικής εμβέλειας, προγραμματιστικής μονάδας και καθολικού επιπέδου.

➡ Μεταβλητές τοπικής εμβέλειας

Μία μεταβλητή τοπικής εμβέλειας (local variable), περιέχει δεδομένα τα οποία είναι ορατά μόνο μέσα

από τη διαδικασία (ρουτίνα ή συνάρτηση) στην οποία έχει δηλωθεί. Η διάρκεια της τοπικής μεταβλητής εξαρτάται από το χρονικό διάστημα το οποίο είναι ανοικτή η διαδικασία. Η ζωή της τοπικής μεταβλητής ξεκινά όταν αρχίζει η εκτέλεση της διαδικασίας και τελειώνει μόλις ολοκληρωθεί η εκτέλεση. Σε δύο διαφορετικές διαδικασίες έχουμε τη δυνατότητα να χρησιμοποιούμε διαφορετικές τοπικές μεταβλητές με το ίδιο όνομα. Αυτό δεν επηρεάζει καθόλου την εφαρμογή αφού κάθε φορά που ολοκληρώνουμε την εκτέλεση μίας διαδικασίας η τιμή των τοπικών της μεταβλητών μηδενίζεται και η θέση μνήμης που καταλαμβάνουν στο σύστημα απελευθερώνεται.

Αν θέλουμε να ορίσουμε μέσα σε μία διαδικασία μία μεταβλητή σαν μεταβλητή τοπικής εμβέλειας χρησιμοποιούμε τη δήλωση Dim. Με μία δήλωση Dim μπορούμε να ορίσουμε μια ή περισσότερες μεταβλητές:

```
Dim Total As Long
```

```
Dim Test As Integer, Name As String
```

Η τιμή μιας τοπικής μεταβλητής όπως ήδη αναφέραμε, χάνεται μόλις κλείσουμε τη διαδικασία μέσα στην οποία την ορίσαμε. Αν θέλουμε να διατηρήσουμε την τιμή μίας τοπικής μεταβλητής μεταξύ διαδοχικών κλήσεων μιας ρουτίνας, πρέπει να την ορίσουμε με τον ειδικό τύπο Static ως εξής:

```
Static Flag As Integer
```

Οι μεταβλητές τύπου Static έχουν το πλεονέκτημα ότι διατηρούν την τιμή τους σε όλη τη διάρκεια εκτέλεσης της εφαρμογής, ακόμη και αν κλείσει η διαδικασία που τις χρησιμοποιεί.

⇒ Μεταβλητές προγραμματιστικής μονάδας

Μία προγραμματιστική μονάδα περιγράψαμε, αποτελεί ανεξάρτητο τμήμα κώδικα του μια κώδικα μιας εφαρμογής. Κάθε μεταβλητή που δηλώνουμε μέσα στο τμήμα δηλώσεων μίας προγραμματιστικής μονάδας (module-level variable), περιέχει δεδομένα που είναι προσπελάσιμα μόνο από τα τμήματα κώδικα της συγκεκριμένης προγραμματιστικής μονάδας. Η διάρκεια ζωής των μεταβλητών της προγραμματιστικής μονάδας διαρκεί για όλο το διάστημα της εκτέλεσης της εφαρμογής, ακόμη και αν σταματήσει να εκτελείται ο κώδικας της συγκεκριμένης προγραμματιστικής μονάδας ή αν κατεβάσουμε τη φόρμα από τη μνήμη του συστήματος.

Για να ορίσουμε μία μεταβλητή επιπέδου προγραμματιστικής μονάδας, χρησιμοποιούμε μια από τις δηλωτικές εντολές Dim ή Private. Η δήλωση της μεταβλητής όμως, πρέπει να γίνει απαραίτητα στο τμήμα δηλώσεων (Declarations section) της προγραμματιστικής μονάδας :

```
Private Counter As Integer
```

ή

```
Dim Counter As Integer
```

Για τον ορισμό των μεταβλητών επιπέδου προγραμματιστικής μονάδας, προτείνεται να χρησιμοποιούμε την δηλωτική εντολή Private, για να είναι ευδιάκριτος ο διαχωρισμός των μεταβλητών προγραμματιστικής μονάδας από τις μεταβλητές καθολικής εμβέλειας που θα γνωρίσουμε στη συνέχεια.

⇒ Μεταβλητές καθολικής εμβέλειας

Αποτελούν μέσα σε ένα πρόγραμμα της Visual Basic τις μεταβλητές με τη μεγαλύτερη εμβέλεια. Στα δεδομένα μίας καθολικής μεταβλητής (public variable), έχουν πρόσβαση όλα τα τμήματα κώδικα που περιέχει η εφαρμογή.

Η δήλωση μίας μεταβλητής καθολικής εμβέλειας γίνεται στο τμήμα δηλώσεων (Declarations section), οποιασδήποτε προγραμματιστικής μονάδας (επιπέδου κώδικα ή φόρμας) με τη δηλωτική εντολή Public:

```
Public LastName As String
```

3.6. Δομές Απόφασης ή Επιλογής

Τις δομές απόφασης (Decision structures), ή δομές επιλογής μπορούμε να τις περιγράψουμε σαν τις μεθόδους λήψης αποφάσεων μέσα στον κώδικα της Visual Basic. Σε μία εφαρμογή της Visual Basic μία ρουτίνα μπορεί να αποτελείται από διαφορετικές εντολές κώδικα. Ο προγραμματιστής έχει τη δυνατότητα να συμπεριλάβει μία συνθήκη μέσα στην ρουτίνα, η οποία κάθε φορά θα καθορίζει ανάλογα με τα αποτελέσματα της ποιες από τις εντολές κώδικα θα εκτελούνται και ποιες θα αγνοούνται. Μία συνθήκη συνήθως είναι μία σύγκριση αριθμητικών ή αλφαριθμητικών τιμών, αλλά μπορεί να είναι και μια επαλήθευση των λογικών τιμών Αληθής ή Ψευδής.

Η Visual Basic υποστηρίζει τους παρακάτω τύπους δομών απόφασης :

- ✓ **If...Then**
- ✓ **If...Then...Else**
- ✓ **Select Case**

Η σύνταξη των δομών απόφασης της Visual Basic είναι ίδια με την σύνταξη των αντίστοιχων δομών απόφασης της QuickBasic.

3.7. Δομές Ανακύκλωσης

Οι δομές ανακύκλωσης (Loops), επιτρέπουν μέσα από την ικανοποίηση μίας συνθήκης την επαναληπτική εκτέλεση μίας ομάδας εντολών κώδικα. Η Visual Basic υποστηρίζει τους παρακάτω τύπους δομών ανακύκλωσης:

- ✓ **For...Next**
- ✓ **For Each...Next**
- ✓ **Do...Loop**

Η σύνταξη των δομών ανακύκλωσης της Visual Basic είναι ίδια με την σύνταξη των αντίστοιχων δομών ανακύκλωσης της QuickBasic.

4. ΤΕΧΝΙΚΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Όπως αναφέραμε, η ανάπτυξη μιας εφαρμογής στο περιβάλλον προγραμματισμού της Visual Basic γίνεται με μεγάλη ευκολία και ταχύτητα. Η ευκολία προγραμματισμού της εφαρμογής, στηρίζεται στην απλότητα σχεδίασης του γραφικού μέσου διασύνδεσης χρήστη-εφαρμογής (graphical user interface) και στον τρόπο δημιουργίας του κώδικα των προγραμμάτων με τη χρήση των γνωστών εντολών της Basic του DOS. Αρκετοί από εσάς σκέφτηκαν ίσως, ότι αφού η δημιουργία μιας εφαρμογής με τη Visual Basic γίνεται τόσο απλά και γρήγορα, άρα το περιβάλλον της δεν θα προσφέρεται για την ανάπτυξη επαγγελματικών εφαρμογών.

Η Visual Basic όμως θα σας διαψεύσει πολύ σύντομα. Μια εφαρμογή που δημιουργείται μέσα στο γραφικό περιβάλλον προγραμματισμού της, έχει τη δυνατότητα υποστήριξης ανεπτυγμένων τεχνικών προγραμματισμού, όπως χρήση μενού επιλογών, διαχείριση λαθών (error handling), υποστήριξη γραφικών (graphics), επικοινωνία με άλλες εφαρμογές (μηχανισμοί DDE και OLE), κ.α. Αλλά το γεγονός που σίγουρα θα σας εντυπωσιάσει είναι ότι όσο μεγάλη και πολύπλοκη και αν είναι μια εφαρμογή της Visual Basic, το περιβάλλον εργασίας της διατηρεί τη λειτουργικότητά του και τη φιλικότητα επικοινωνίας με το χρήστη.

4.1. Μενού επιλογών

Όλοι μας θα έχουμε σίγουρα χρησιμοποιήσει μέσα από τις εφαρμογές μας κάποιο μενού επιλογών (menu). Μέσα από τις επιλογές ενός μενού, έχουμε τη δυνατότητα κατά την εκτέλεση μιας εφαρμογής, να παρέχουμε στους χρήστες μια εύκολα προσπελάσιμη λίστα εργασιών. Σε μια εφαρμογή της Visual Basic, μπορούμε να δημιουργήσουμε πολύ εύκολα μενού επιλογών και να εισάγουμε σε κάθε επιλογή του εντολές κώδικα για την εκτέλεση συγκεκριμένων εργασιών.

Σε κάθε φόρμα που περιέχει μια εφαρμογή της Visual Basic, μπορούμε να δημιουργήσουμε ένα μενού επιλογών. Τα μενού επιλογών που υποστηρίζει η Visual Basic, συνδυάζουν όλα τα πλεονεκτήματα των μενού επιλογών που έχουμε συναντήσει σε άλλες Windows-based εφαρμογές, όπως δυνατότητα υποστήριξης υπομενού επιλογών, πλήκτρων άμεσης πρόσβασης (access keys), πλήκτρων συντομίας (shortcut keys), κ.α.

Το εργαλείο που χρησιμοποιούμε για να σχεδιάσουμε ένα μενού επιλογών, είναι ο Επεξεργαστής Μενού (Menu Editor) της Visual Basic (Σχήμα 10). Για να εμφανίσουμε τον Επεξεργαστή Μενού, επιλέγουμε από το

μενού Tools την επιλογή Menu Editor ή το πλήκτρο Menu Editor της γραμμής εργαλείων.



Σχ.10 Ο Επεξεργαστής Μενού της Visual Basic.

Μέσα από τον Επεξεργαστή Μενού δημιουργούμε τις επιλογές ενός μενού, οι οποίες ονομάζονται αντικείμενα μενού (menu items). Κάθε επιλογή που εισάγουμε σε ένα μενού της Visual Basic, αποτελεί ανεξάρτητο αντικείμενο της εφαρμογής και χαρακτηρίζεται όπως και τα υπόλοιπα αντικείμενα της Visual Basic, από ένα προκαθορισμένο αριθμό ιδιοτήτων και επιπλέον το γεγονός click. Τις τιμές των ιδιοτήτων ενός αντικειμένου μενού, μπορούμε να τις μετατρέψουμε κατά το σχεδιασμό της εφαρμογής μέσω του Επεξεργαστή Μενού ή μέσω του παραθύρου ιδιοτήτων. Οι εργασίες που πρέπει να εκτελούν οι επιλογές ενός μενού, ορίζονται μέσα από εντολές κώδικα που θα συμπεριλάβουμε στις ρουτίνες γεγονότων click κάθε αντικειμένου μενού.

Ο Επεξεργαστής Μενού αποτελείται από δύο τμήματα. Στο επάνω τμήμα του, παρατηρούμε ότι εμφανίζεται μια ομάδα ιδιοτήτων, που ονομάζονται ιδιότητες εργαλείου μενού (menu control properties). Μέσω των ιδιοτήτων κάθε εργαλείου, καθορίζουμε τα χαρακτηριστικά εμφάνισης και την αρχική συμπεριφορά των επιλογών του μενού. Οι δύο πιο βασικές ιδιότητες μιας επιλογής ενός μενού, είναι η Name που δηλώνει το όνομα και η Caption που περιγράφει την επικεφαλίδα της επιλογής. Η ιδιότητα Name είναι το όνομα με το οποίο διαχειριζόμαστε το αντικείμενο μενού μέσω εντολών κώδικα της Visual Basic και η ιδιότητα Caption είναι το λεκτικό με το οποίο θα εμφανίζεται η επιλογή επάνω στο μενού κατά την εκτέλεση της εφαρμογής.

Το κάτω τμήμα του Επεξεργαστή Μενού, καλείται πλαίσιο λίστας εργαλείων μενού (menu control list box)

και κατά το σχεδιασμό ενός μενού περιέχει όλες τις επιλογές που έχουμε ήδη δημιουργήσει και το επίπεδό τους.

4.2. Πλαίσια διαλόγου

Κατά την εκτέλεση μιας εφαρμογής χρειάζεται πολλές φορές ο χρήστης να ανταλλάσσει μηνύματα με την εφαρμογή. Για να επιτύχουμε την επικοινωνία του χρήστη με την εφαρμογή, έχουμε τη δυνατότητα να δημιουργήσουμε με τη Visual Basic πλαίσια διαλόγου (dialog boxes). Τα πλαίσια διαλόγου μπορούμε να τα χρησιμοποιήσουμε για να εισάγει ο χρήστης κάποια στοιχεία στην εφαρμογή, αλλά και προς την αντίθετη κατεύθυνση για την εμφάνιση μηνυμάτων της εφαρμογής προς το χρήστη.

Σε μια εφαρμογή της Visual Basic, υπάρχουν τρεις τρόποι να προσθέσουμε πλαίσια διαλόγου:

- ⇒ με τη δημιουργία καινούριων πλαισίων σύμφωνα με τις ανάγκες μιας εφαρμογής, τοποθετώντας εργαλεία επάνω σε μια φόρμα ή εισάγοντας και μορφοποιώντας στην εφαρμογή κάποιο υπάρχον πλαίσιο διαλόγου.
- ⇒ με τη χρήση συγκεκριμένων πλαισίων διαλόγου (standard dialog boxes), μέσω του εργαλείου Common dialog ActiveX control της Visual Basic, όπως το πλαίσιο αποθήκευσης αρχείων.
- ⇒ με τη χρήση προκαθορισμένων πλαισίων διαλόγου μέσω των συναρτήσεων Input() και MsgBox().

Από τους παραπάνω τύπους πλαισίων διαλόγου, καινούριο στοιχείο αποτελούν τα προκαθορισμένα πλαίσια της Visual Basic. Η Visual Basic υποστηρίζει δύο συναρτήσεις τη **MsgBox()** και την **InputBox()** για τη δημιουργία προκαθορισμένων πλαισίων διαλόγου. Χρησιμοποιώντας τις δύο παραπάνω συναρτήσεις δε χρειάζεται να σχεδιάσουμε από την αρχή πλαίσια διαλόγου αλλά μπορούμε να εμφανίσουμε με τις δικές μας παραμέτρους τα εσωτερικά πλαίσια της Visual Basic. Ωστόσο με τη χρήση των δύο αυτών συναρτήσεων, δεν έχουμε ιδιαίτερες δυνατότητες μορφοποίησης της εμφάνισης των πλαισίων διαλόγου.

Τα πλαίσια διαλόγου της Visual Basic χωρίζονται σε δύο κατηγορίες:

- ⇒ Τα υποχρεωτικά (modal), τα οποία παραμένουν τα ενεργά αντικείμενα της εφαρμογής μέχρι να τα κλείσει ο χρήστης. Με την τεχνική αυτή είναι σίγουρο ότι ο χρήστης θα δει το μήνυμα του πλαισίου, αφού είναι υποχρεωμένος πρώτα να το κλείσει για να προχωρήσει στη συνέχεια της εφαρμογής.
- ⇒ Τα μη υποχρεωτικά (modeless), τα οποία δεν είναι απαραίτητο να κλείσει ο χρήστης για να συνεχίσει την εκτέλεση της εφαρμογής. Όταν εμφανίζεται ένα μη υποχρεωτικό πλαίσιο διαλόγου, μπορούμε να φέρουμε στο προσκήνιο της εφαρμογής κάποια άλλη φόρμα. Τα μη υποχρεωτικά πλαίσια διαλόγου δεν εμφανίζονται συχνά μέσα σε εφαρμογές

και χρησιμοποιούνται κυρίως για την εμφάνιση βοηθητικών μηνυμάτων προς το χρήστη και όχι για την εκτέλεση κρίσιμων εργασιών της εφαρμογής.

4.3. Εκτυπώσεις

Μια εφαρμογή της Visual Basic μας δίνει τη δυνατότητα να εκμεταλλευτούμε όλα τα πλεονεκτήματα εκτύπωσης που προσφέρει το περιβάλλον γραφικών των Windows, όπως συνδυασμένη εκτύπωση γραφικών και κειμένου, χρήση TrueType γραμματοσειρών, επιλογή εκτυπωτή, κ.α.

Δυστυχώς όμως, η εργασία των εκτυπώσεων μέσα από το περιβάλλον των Windows δεν είναι ιδιαίτερα φιλική, αφού το αποτέλεσμα μιας εκτύπωσης εξαρτάται από πολλές παραμέτρους. Σε μια διαδικασία εκτύπωσης είναι δυνατό να μην επιτύχουμε καλή ποιότητα, όχι λόγω προβλημάτων της εφαρμογής αλλά από την επίδραση εξωτερικών παραγόντων όπως η χρήση διαφορετικού οδηγού εκτυπωτή (printer driver) ή ενός εκτυπωτή με περιορισμένες δυνατότητες. Τέλος η ποιότητα εκτύπωσης μέσα από μια εφαρμογή της Visual Basic, είναι συνάρτηση και της μεθόδου που εκτελεί την εκτύπωση.

Για να εκτυπώσουμε δεδομένα, μέσα από μια εφαρμογή της Visual Basic σε ένα εκτυπωτή, χρησιμοποιούμε τρεις τεχνικές:

- ✓ Δημιουργούμε την αναφορά εκτύπωσης επάνω σε μια φόρμα εργασίας και την τυπώνουμε με τη μέθοδο PrintForm.
- ✓ Στέλνουμε τα στοιχεία εκτύπωσης απευθείας στον εκτυπωτή, θέτοντας τον εξορισμού εκτυπωτή σαν στοιχείο του αντικειμένου συλλογής εκτυπωτών (Printers Collection).
- ✓ Στέλνουμε τα στοιχεία εκτύπωσης στο ειδικό αντικείμενο Printer της Visual Basic με τη μέθοδο Print και τα εκτυπώνουμε με τις μεθόδους NewPage και EndDoc που υποστηρίζει.

4.4. Επικοινωνία με το Clipboard

Μια εφαρμογή της Visual Basic, έχει τη δυνατότητα εκμετάλλευσης των εργαλείων του λειτουργικού συστήματος, όπως το Clipboard των Windows. Το Clipboard είναι ένα προσωρινό μέσο αποθήκευσης, που χρησιμοποιείται για τη μεταφορά γραφικών και κειμένου μεταξύ εφαρμογών. Στο χώρο μνήμης του Clipboard, έχουν πρόσβαση όλες οι εφαρμογές των Windows.

Οι περισσότεροι χρήστες των Windows έχουν χρησιμοποιήσει το χώρο μνήμης του Clipboard, ακόμη και αν δε το έχουν ανοίξει ποτέ. Κάθε φορά που αντιγράφουν ένα κομμάτι κειμένου σε κάποιο από τους γνωστούς επεξεργαστές κειμένου των Windows, τα δεδομένα περνάνε απαραίτητα μέσα από το χώρο μνήμης του Clipboard.

Για να ανταλλάξουμε δεδομένα μεταξύ μιας εφαρμογής και του Clipboard, χρησιμοποιούμε το ειδικό αντικείμενο Clipboard που υποστηρίζει η Visual Basic. Το ειδικό αντικείμενο Clipboard, χρησιμοποιείται για το χειρισμό κειμένου και γραφικών μέσα στο χώρο μνήμης του Clipboard. Η ανταλλαγή των δεδομένων, γίνεται μέσω των μεθόδων SetText και GetText όταν πρόκειται για χαρακτήρες κειμένου ή των μεθόδων SetData και GetData όταν πρόκειται για γραφικά.

Στη συνέχεια, θα παρουσιάσουμε το τρόπο με τον οποίο αντιγράφουμε ένα τμήμα κειμένου από ένα πλαίσιο κειμένου μιας εφαρμογής προς το Clipboard και το τρόπο με τον οποίο διαβάζουμε τα περιεχόμενά του. Πριν όμως, αντιγράφουμε ένα τμήμα κειμένου από το πλαίσιο κειμένου, πρέπει πρώτα να το επιλέξουμε. Ο πιο απλός τρόπος για να επιλέξουμε ένα τμήμα κειμένου, είναι να τοποθετήσουμε το δρομέα μπροστά από το πρώτο χαρακτήρα, να κρατήσουμε το αριστερό πλήκτρο του ποντικιού συνεχώς πατημένο και να μεταφερθούμε στο τελευταίο χαρακτήρα του τμήματος κειμένου. Το επιλεγμένο κείμενο, μπορούμε να το χειριστούμε μέσω της ιδιότητας SelText του πλαισίου κειμένου. Μια ακόμη χρήσιμη ιδιότητα των πλαισίων κειμένου για το χειρισμό κειμένου, είναι η SelLength που δηλώνει το μήκος του επιλεγμένου κειμένου. Όταν δεν έχουμε επιλέξει κανένα τμήμα του κειμένου, η τιμή της SelLength είναι ίση με το μηδέν.

Για να αντιγράφουμε το επιλεγμένο κείμενο από το πλαίσιο κειμένου TxtEntry στο Clipboard, πρέπει να χρησιμοποιήσουμε τη μέθοδο SetText του ειδικού αντικειμένου Clipboard:

```
Clipboard.SetText TxtEd.it.SelText
```

Η επαναφορά του επιλεγμένου κειμένου από το Clipboard στην εφαρμογή, μπορεί να γίνει μέσω της μεθόδου GetText του ειδικού αντικειμένου Clipboard:

```
.TxtEdit.SelText = Clipboard.GetText
```

Όταν χρησιμοποιούμε γραφικά αντί για κείμενο οι τεχνικές ανταλλαγής δεδομένων μεταξύ του Clipboard και της εφαρμογής, είναι ίδιες. Το μόνο που αλλάζει είναι οι μέθοδοι (GetData, SetData) που θα χρησιμοποιήσουμε.

Πριν αντιγράψουμε το επιλεγμένο κείμενο, έχουμε τη δυνατότητα να καθαρίσουμε τα περιεχόμενα του Clipboard με την εντολή :

```
Clipboard.Clear
```

Τα περιεχόμενα του Clipboard, πρέπει να τα καθαρίζουμε μετά από την αντιγραφή μεγάλου όγκου δεδομένων, για να μη δεσμεύουν χώρο στη μνήμη του συστήματος.

5. ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Στις εφαρμογές της Visual Basic μπορούν να χρησιμοποιηθούν έτοιμα δεδομένα που προέρχονται από

συστήματα διαχείρισης βάσεων δεδομένων. Η Visual Basic παρέχει δυνατότητα πρόσβασης σε αρχεία βάσεων δεδομένων διαφορετικών μορφοποιήσεων. Επιτυγχάνεται πρόσβαση σε αρχεία δεδομένων των Access, Excel, Btrieve, dBase, FoxPro, Paradox κ.ά. Όλες αυτές οι βάσεις δεδομένων, από τις οποίες μπορεί να αντληθεί στοιχεία η Visual Basic, ονομάζονται εξωτερικές βάσεις δεδομένων.

5.1. Εργαλείο δεδομένων

Για να μπορέσουμε να επιτύχουμε πρόσβαση της εφαρμογής μας σε μία βάση δεδομένων, θα πρέπει να χρησιμοποιήσουμε το εργαλείο δεδομένων (data control). (σχήμα 11).



Σχ. 11 Το εργαλείο δεδομένων.

Χρησιμοποιώντας το εργαλείο δεδομένων μπορούμε να συνδέσουμε μια εφαρμογή της Visual Basic με μία βάση δεδομένων και να ανακτήσουμε ένα σύνολο εγγραφών (recordset).

Με τον όρο recordset object αναφέρεται ένα αντικείμενο που αυτόματα δημιουργείται όταν ανοίγουμε μία βάση δεδομένων. Το recordset αποτελεί ένα σύνολο από εγγραφές.

5.2. Ιδιότητες βάσεων δεδομένων

5.2.1. Η ιδιότητα Connect

Η ιδιότητα Connect προσδιορίζει τον τύπο της βάσης δεδομένων που χρησιμοποιείται και σε ορισμένες περιπτώσεις κάποιες επιπρόσθετες παραμέτρους (πχ. password).

5.2.2. Η ιδιότητα DatabaseName

Η ιδιότητα DatabaseName χρησιμοποιείται για τον προσδιορισμό της βάσης δεδομένων που θα χρησιμοποιηθεί. Οι βάσεις δεδομένων που μπορούν να χρησιμοποιηθούν σε μία εφαρμογή της Visual Basic μπορούν να βρίσκονται σε τοπικό ή σε remote επίπεδο, δηλαδή στο server ενός δικτύου. Ακόμα η χρήση μιας βάσης δεδομένων μπορεί να γίνεται ταυτόχρονα από πολλούς χρήστες.

5.2.3. Η ιδιότητα Exclusive

Από τη στιγμή που θα επιτύχουμε πρόσβαση σε μία βάση δεδομένων, μπορούμε να απαιτήσουμε τον αποκλειστικό έλεγχο της, ορίζοντας την τιμή της ιδιότητας Exclusive σαν True. Σε κάποιες περιπτώσεις μία τέτοια απαίτηση μπορεί πραγματικά να είναι απαραίτητη. Χρησιμοποιώντας αποκλειστικά μία βάση δεδο-

μένων, επιτυγχάνεται γρηγορότερη πρόσβαση στα δεδομένα, άρα και ταχύτερη εκτέλεση της εφαρμογής.

5.2.4. Η ιδιότητα Options

Ορίζοντας τιμές στην ιδιότητα Options, μπορούμε να επιτύχουμε έλεγχο σε ότι αφορά στην πρόσβαση στους πίνακες της βάσης δεδομένων. Μέσω αυτών των επιλογών μπορούμε να ορίσουμε απαγόρευση πρόσβασης σε κάποιους πίνακες οι οποίοι περιέχουν τις εγγραφές εκείνες που θέλουμε να χαρακτηρίσουμε σαν μη προσβάσιμες.

5.2.5. Η ιδιότητα ReadOnly

Αν μέσω μίας εφαρμογής της Visual Basic θέλουμε να ανοίξουμε μία βάση δεδομένων, την οποία δεν πρόκειται να τροποποιήσουμε σε ότι αφορά την δομή ή τα δεδομένα της, τότε μπορούμε να θέσουμε στην ιδιότητα ReadOnly την τιμή True.

5.2.6. Η ιδιότητα RecordSource

Η ιδιότητα RecordSource προσδιορίζει πού θα βρεθούν τα δεδομένα της βάσης δεδομένων που θα χρησιμοποιηθεί στην εφαρμογή της Visual Basic. Η ιδιότητα αυτή δεν έχει νόημα παρά μόνο αφού ανοιχτεί η βάση δεδομένων.

5.3. Τα εργαλεία εμφάνισης δεδομένων

Σαν εργαλεία εμφάνισης (Bound Controls) χαρακτηρίζονται όλα εκείνα τα εργαλεία με τα οποία δημιουργούνται τα αντίστοιχα αντικείμενα, μέσω των οποίων επιτυγχάνεται η εμφάνιση των πληροφοριών που βρίσκονται σε μια βάση δεδομένων που έχει συνδεθεί με την εφαρμογή μας. Όταν ένα αντικείμενο εμφάνισης συνδέεται με το αντικείμενο δεδομένων, τότε η Visual Basic αποθέτει στο συγκεκριμένο αντικείμενο εμφάνισης, τις τιμές ενός πεδίου από την επιλεγμένη βάση δεδομένων. Οι τιμές των πεδίων μιας εγγραφής μπορούν να αλλάξουν μέσω της εφαρμογής της Visual Basic, οι δε αλλαγές που πραγματοποιούνται στη συγκεκριμένη εγγραφή, σώζονται αυτόματα μόλις ο έλεγχος του προγράμματος περάσει σε μία άλλη εγγραφή. Η Visual Basic υποστηρίζει αρκετά εργαλεία (εικόνα (image) ετικέτα (label) πλαίσιο εικόνας (picture box) πλαίσιο κειμένου (text box) πλαίσιο ελέγχου (check box) λίστα εμφάνισης δεδομένων (DBList) συνδυασμένη λίστα εμφάνισης δεδομένων (DBCombo) κλπ) τα οποία μπορούν δημιουργήσουν τα αντίστοιχα αντικείμενα τα οποία θα συνδεθούν με το αντικείμενο δεδομένων και θα χρησιμοποιηθούν για την εμφάνιση των πληροφοριών της βάσης δεδομένων.

Τα αντικείμενα εμφάνισης παρουσιάζουν δύο χαρακτηριστικές ιδιότητες :

- DataField: προσδιορίζει το όνομα του πεδίου οι τιμές του οποίου εμφανίζονται στο συγκεκριμένο αντικείμενο εμφάνισης.

- DataSource: προσδιορίζει το όνομα του αντικείμενου δεδομένων με το οποίο συνδέεται το συγκεκριμένο αντικείμενο εμφάνισης.

6. ΓΡΑΦΙΚΑ

Η Visual Basic διαθέτει δύο διαφορετικούς τρόπους για τη δημιουργία γραφικών σε μία εφαρμογή. Μπορούμε να χρησιμοποιήσουμε είτε τα εργαλεία γραφικών είτε μεθόδους γραφικών. Σχεδιάζουμε σημεία, γραμμές, πλαίσια, κύκλους και άλλα γεωμετρικά σχήματα και τροποποιούμε τη μορφή και τη θέση εμφάνισής τους επάνω στην φόρμα.

6.1. Το σύστημα συντεταγμένων

Κάθε διαδικασία με γραφικά χρησιμοποιεί ένα σύστημα συντεταγμένων που αφορά την περιοχή σχεδίασης. Χρησιμοποιώντας το σύστημα συντεταγμένων μπορούμε να προσδιορίσουμε σημεία στην οθόνη, σε μία φόρμα, σ' ένα πλαίσιο εικόνας. Η μορφή που χρησιμοποιείται για να οριστεί ένα σημείο είναι η (x,y). Το x είναι η τιμή της προβολής του σημείου στον οριζόντιο άξονα των x και το y η αντίστοιχη τιμή της προβολής του στον κατακόρυφο άξονα των y.

Το σύστημα συντεταγμένων της Visual Basic υπόκειται σε κάποιους κανόνες:

- Η αλλαγή μεγέθους ή η μετακίνηση ενός αντικείμενου καθορίζεται από το σύστημα συντεταγμένων που χρησιμοποιεί η οντότητα που περιέχει αυτό το αντικείμενο. Για παράδειγμα εάν σχεδιάζουμε ένα πλαίσιο εικόνας πάνω σε μία φόρμα, το σύστημα συντεταγμένων της φόρμας, είναι αυτό που θα καθορίζει οτιδήποτε αφορά τη μετακίνηση ή τις αλλαγές του μεγέθους του πλαισίου εικόνας.
- Όλες οι μέθοδοι γραφικών χρησιμοποιούν το σύστημα συντεταγμένων της φόρμας ή του αντικείμενου στο οποίο αναφέρονται. Αν για παράδειγμα χρησιμοποιούμε μία μέθοδο σχεδίασης για ένα πλαίσιο εικόνας, το σύστημα συντεταγμένων που χρησιμοποιείται είναι αυτό του πλαισίου εικόνας.
- Οι εντολές που χρησιμοποιούνται για τον επαναπροσδιορισμό του μεγέθους ή τη μετακίνηση μιας φόρμας, εκφράζουν πάντοτε τη θέση και το μέγεθος της φόρμας σε twips.
- Η επάνω αριστερή γωνία της οθόνης έχει τιμές συντεταγμένων (0,0). Επίσης η επάνω αριστερή γωνία κάθε φόρμας, πλαισίου εικόνας ή πλαισίου, έχει - στο εξ ορισμού σύστημα συντεταγμένων - τιμές (0,0).
- Οι μονάδες μέτρησης που χρησιμοποιούνται για τον προσδιορισμό των σημείων κατά μήκος των δύο αξόνων ονομάζονται κλίμακα. Η Visual Basic μπορεί να υποστηρίξει στο σύστημα συντεταγμένων διαφορετική κλίμακα για κάθε άξονα. Επίσης μπορεί ν' αλλαχτεί τόσο η διεύθυνση και τα σημεία

εκκίνησης των αξόνων, όσο και η κλίμακα του συστήματος συντεταγμένων.

6.2. Μονάδες μέτρησης

Εξ ορισμού στην Visual Basic όλες οι εντολές σχεδίασης, μετακίνησης και καθορισμού μεγέθους γραφικών, χρησιμοποιούν σαν μονάδα μέτρησης το twip. Ένα twip είναι το 1/20 του ενός σημείου εκτύπωσης ή διαφορετικά 1440 twips ισούνται με μία ίντσα. Αυτές οι αντιστοιχίες αναφέρονται στο μέγεθος του αντικειμένου κατά την εκτύπωση. Σε ότι αφορά την εμφάνιση του αντικειμένου στην οθόνη, οι φυσικές αποστάσεις ποικίλουν ανάλογα με το μέγεθος της οθόνης.

6.3. Αλλαγή κλίμακας συστήματος συντεταγμένων

Ο χρήστης σε ότι αφορά το σύστημα των συντεταγμένων έχει τρεις επιλογές:

- ✓ να χρησιμοποιήσει την εξ ορισμού κλίμακα μέτρησης, δηλαδή τα twips
- ✓ να χρησιμοποιήσει μία από τις υπόλοιπες έτοιμες κλίμακες που παρέχει η Visual Basic
- ✓ να δημιουργήσει μία δική του κλίμακα.

Ένας τρόπος, ο απλούστερος, ορισμού του συστήματος συντεταγμένων είναι να αποδοθεί μία τιμή στην ιδιότητα ScaleMode. Οι δυνατές τιμές της ιδιότητας, η οποία αναφέρεται στα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής, φαίνονται στον πίνακα 2.

Τιμή	Περιγραφή
0 - User	Δηλώνει πως μία ή περισσότερες από τις τιμές των ιδιοτήτων ScaleHeight, ScaleWidth, ScaleLeft και ScaleTop ορίζονται από το χρήστη.
1 - Twip	Η εξ ορισμού κλίμακα, όπου 567 twips = 1 cm.
2 - Point	Όπου 1 inch = 72 σημεία (points).
3 - Pixel	Όπου σαν pixel ορίζεται η μικρότερη μονάδα ανάλυσης της οθόνης.
4 - Character	Όπου 1 χαρακτήρας (character) αναπαρίσταται μέσα σε ορθογώνιο πλαίσιο, οριζόντιας πλευράς 120 twips και κατακόρυφης πλευράς 240 twips
5 - Inch	Ίντσα, όπου 1 inch = 2,53 cm.
6 - Milimeter	Χιλιοστό του μέτρου.
7 - Centimeter	Εκατοστό του μέτρου.

Πίνακας 2: Οι δυνατές τιμές της ιδιότητας ScaleMode.

Όταν αποδίδουμε μία από τις δυνατές τιμές (πλην της 0) στην ιδιότητα ScaleMode, η Visual Basic επαναπροσδιορίζει τις τιμές των ιδιοτήτων ScaleWidth και ScaleHeight, αποδίδοντάς τους τιμές που να αντιστοιχούν στη νέα κλίμακα συντεταγμένων. Ταυτόχρονα μηδενίζονται οι τιμές των ιδιοτήτων ScaleTop και ScaleLeft. Επίσης αλλάζουν οι τιμές των ιδιοτήτων CurrentX και CurrentY, έτσι ώστε να εκφράζουν τις συντεταγμένες του τρέχοντος σημείου στο καινούριο σύστημα συντεταγμένων.

Ένας άλλος τρόπος ορισμού του συστήματος συντεταγμένων είναι χρησιμοποιώντας τις συσχετιζόμενες ιδιότητες ScaleHeight, ScaleWidth, ScaleLeft και ScaleTop. Αποδίδοντας τιμή σε τουλάχιστον μία από αυτές, μπορούμε να δημιουργήσουμε ένα δικό μας σύστημα συντεταγμένων για το αντικείμενο στο οποίο αναφερόμαστε. Θέτοντας όμως τιμή σε οποιαδήποτε από τις παραπάνω ιδιότητες, αυτόματα τίθεται η τιμή 0 στην ιδιότητα ScaleMode.

Στην κλίμακα που δημιουργεί ο χρήστης, μπορεί να αποδώσει στην επάνω αριστερή γωνία τόσο της φόρμας όσο και του αντικειμένου που περιέχεται σ' αυτήν, οποιοδήποτε ζεύγος τιμών θέλει, και όχι υποχρεωτικά το ζεύγος τιμών (0,0).

Τέλος σημειώνουμε πως με τη χρήση της μεθόδου Scale μπορούμε επίσης να αποδώσουμε τιμές στις ιδιότητες ScaleHeight, ScaleWidth, ScaleLeft και ScaleTop, πράγμα που σημαίνει, όπως ήδη εξηγήθηκε, τον ορισμό μιας καινούριας κλίμακας συντεταγμένων.

6.4. Χρώματα

Η Visual Basic υποστηρίζει 256 χρώματα, με την προϋπόθεση βέβαια πως το σύστημα απεικόνισης του υπολογιστή μπορεί να τα αποδώσει. Η ανάγκη για ταυτόχρονη απεικόνιση 256 διαφορετικών χρωμάτων, εμφανίζεται κύρια σε Multimedia εφαρμογές και στις εφαρμογές εκείνες που χειρίζονται εικόνες με μεγάλες απαιτήσεις στη χρωματική απόδοση.

Μπορούμε να χρησιμοποιήσουμε 256 χρώματα σε μεθόδους γραφικών που επιδρούν σε φόρμες, αντικείμενα πλαισίου εικόνας και αντικείμενα εικόνας. Θα πρέπει όμως να σημειωθεί πως για αρχεία εικόνας τύπου Metafile, η Visual Basic δεν υποστηρίζει 256 χρώματα, αλλά μόνο τα 16 της QuickBasic.

Κάθε χρώμα στη Visual Basic αντιπροσωπεύεται από ένα long integer. Η γλώσσα διαθέτει δύο διαφορετικούς τρόπους για τον προσδιορισμό των απαιτούμενων χρωμάτων μιας εφαρμογής:

- ✓ Με χρήση της συνάρτησης RGB.
- ✓ Με χρήση της συνάρτησης QBColor που παρέχει τη δυνατότητα επιλογής ενός από τα 16 χρώματα που υποστηρίζει η QuickBasic.

6.4.1. Η συνάρτηση RGB

Η συνάρτηση RGB επιστρέφει σαν τιμή ένα long integer ο οποίος αντιπροσωπεύει την τιμή ενός χρώματος. Η τιμή αυτή κατ' ουσία αντιστοιχεί στην σχετική ένταση συμμετοχής των χρωμάτων κόκκινο, πράσινο και μπλε, ώστε να δημιουργηθεί το επιθυμητό χρώμα ή χρωματική απόχρωση. Ο τρόπος σύνταξης της συνάρτησης περιλαμβάνει τη χρήση τριών ακεραίων - μετά τη δήλωση του ονόματός της - οι οποίοι μπορούν να παίρνουν τιμές από 0 μέχρι 255 και καθένας από αυτούς αντιπροσωπεύει κατά σειρά, τη συμμετοχή των τριών βασικών χρωμάτων κόκκινου, πράσινου και μπλε.

```
Form1.BackColor = RGB(0,255,0)
```

```
Form2.ForeColor = RGB(0,0,128)
```

Κάθε χρώμα ή χρωματική απόχρωση μπορεί να παρασταθεί σαν συνδυασμός διαφορετικών τιμών συμμετοχής των τριών αυτών βασικών χρωμάτων, η παράθεση των τιμών των οποίων πρέπει να γίνεται αποκλειστικά με την προαναφερθείσα σειρά. Στον πίνακα 3 απεικονίζεται η δεκαεξαδική μορφή μερικών συνηθισμένων χρωμάτων και η τιμή της συμμετοχής των τριών βασικών χρωμάτων κόκκινου, πράσινου και μπλε στη δημιουργία καθενός από αυτά.

Χρώμα	Τιμή RGB	Τιμή κόκκινου	Τιμή πράσινου	Τιμή μπλε
Μαύρο	&H00	0	0	0
Μπλε	&HFF0000	0	0	255
Πράσιν	&HFF00	0	255	0
Κόκκινο	&HFF	255	0	0
Κυανό	&HFFFF00	0	255	255
Κίτρινο	&HFFFF	255	255	0
Μοβ	&HFF00FF	255	0	255
Άσπρο	&HFFFFFF	255	255	255

Πίνακας 3: Τιμές RGB μερικών συνηθισμένων χρωμάτων.

6.4.2. Η συνάρτηση QBColor

Η συνάρτηση QBColor δέχεται έναν ακεραίο αριθμό από 0 έως 15, ο οποίος αντιστοιχεί σ' ένα από τα 16 χρώματα που χρησιμοποιούνται από την QuickBasic (πίνακας 4) και επιστρέφει μία τιμή η οποία αντιστοιχίζει το επιλεγμένο χρώμα της QuickBasic, στο αντίστοιχό του στο χρωματικό σύστημα RGB που χρησιμοποιεί η Visual Basic. Για παράδειγμα:

```
TestForm.BackColor = QBColor(4)
```

Με την παραπάνω εντολή, αποδίδεται στην ιδιότητα BackColor της φόρμας TestForm το κόκκινο χρώμα, όπως ορίζεται στην QuickBasic με τη σταθερά 4.

Αριθμός - Χρώμα	Αριθμός - Χρώμα
0 - Μαύρο	8 - Γκρι
1 - Μπλε	9 - Ανοιχτό Μπλε
2 - Πράσινο	10 - Ανοιχτό Πράσινο
3 - Κυανό	11 - Ανοιχτό Κυανό
4 - Κόκκινο	12 - Ανοιχτό Κόκκινο
5 - Μοβ	13 - Ανοιχτό Μοβ
6 - Κίτρινο	14 - Ανοιχτό Κίτρινο
7 - Άσπρο	15 - Έντονο Άσπρο

Πίνακας 4: Τα 16 χρώματα της συνάρτησης QBColor και οι τιμές τους.

6.5. Εργαλεία γραφικών

Η Visual Basic διαθέτει τρία εργαλεία που μπορούν να χρησιμοποιηθούν για τη δημιουργία γραφικών εφέ σε εφαρμογές. Τα εργαλεία αυτά είναι το εργαλείο εικόνα (Image control), το εργαλείο γραμμή (Line control) και το εργαλείο γεωμετρικό σχήμα (Shape control). Τα τρία αυτά εργαλεία γραφικών αποδεικνύονται πολύ χρήσιμα για τη δημιουργία γραφικών κατά τη σχεδίαση της εφαρμογής.

Ένα πλεονέκτημά τους είναι ότι απαιτούν την εγγραφή πολύ λιγότερου κώδικα απ' ό,τι απαιτείται αν χρησιμοποιηθούν οι μέθοδοι γραφικών. Για παράδειγμα, μπορούμε να σχεδιάσουμε έναν κύκλο χρησιμοποιώντας είτε το εργαλείο γεωμετρικού σχήματος, είτε τη μέθοδο Circle. Η μέθοδος Circle βέβαια απαιτεί την εγγραφή κώδικα, ενώ με τη χρησιμοποίηση του εργαλείου γεωμετρικό σχήμα δεν έχουμε παρά να "ζωγραφίσουμε" τον ζητούμενο κύκλο ορίζοντας την κατάλληλη τιμή στην ιδιότητα Shape του αντικειμένου.

Βέβαια από την άλλη πλευρά παρουσιάζουν και κάποια μειονεκτήματα:

- ✓ Τα αντικείμενα γραφικών δεν μπορούν να αναδυθούν μέσα από άλλα αντικείμενα, εκτός κι αν βρίσκονται μέσα σε ένα τρίτο που έχει αυτή την ικανότητα.
- ✓ Δεν μπορούν να συμπεριλάβουν μέσα τους άλλα αντικείμενα.
- ✓ Δεν μπορούν να υποστηρίξουν διαδικασίες εστίασης πάνω στο γραφικό κατά τη διάρκεια της εκτέλεσης της εφαρμογής.

6.6. Μέθοδοι γραφικών

Η Visual Basic παρέχει συμπληρωματικά προς τα εργαλεία γραφικών, μία σειρά από μεθόδους γραφικών οι οποίες είναι σχεδιασμένες ειδικά για τη δημιουργία γραφικών στις εφαρμογές της. Οι μέθοδοι γραφικών μπορούν να προσφέρουν μερικά ενδιαφέροντα οπτικά εφέ, που δεν μπορούν να δημιουργηθούν με τη χρήση των εργαλείων γραφικών. Γενικά, οι μέθοδοι

γραφικών δεν ενδείκνυνται για τις περιπτώσεις εκείνες, που θέλουμε να δημιουργήσουμε πολύ απλά σχέδια στην παρουσίαση της εφαρμογής μας. Στις περιπτώσεις δημιουργίας γραφικών με μεθόδους γραφικών, θα πρέπει να εκτελέσουμε την εφαρμογή, για να διαπιστώσουμε το οπτικό αποτέλεσμα.

Οι μέθοδοι γραφικών μπορούν να εφαρμοστούν στα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής και είναι οι ακόλουθοι:

- ➔ Cls, η οποία καθαρίζει όλα τα γραφικά που δημιουργούνται κατά τη διάρκεια εκτέλεσης της εφαρμογής σε μία φόρμα ή σ' ένα πλαίσιο εικόνας.
- ➔ PSet, η οποία προσδίδει το καθοριζόμενο χρώμα σ' ένα σημείο του αντικειμένου.
- ➔ Point, η οποία επιστρέφει, σε κλίμακα RGB, την τιμή του χρώματος ενός συγκεκριμένου σημείου μιας φόρμας ή ενός πλαισίου εικόνας.
- ➔ Line, η οποία χρησιμοποιείται για τη σχεδίαση γραμμών και ορθογωνίων σχημάτων.
- ➔ Circle, η οποία χρησιμοποιείται για τη σχεδίαση καμπυλόγραμμων σχημάτων (κύκλου, έλλειψης ή τόξου).
- ➔ Print, η οποία αναφέρεται στο αντικείμενο Debug μπορεί να θεωρηθεί σαν μέθοδος γραφικών. Η μέθοδος αυτή μπορεί να εμφανίσει στο παράθυρο Debug κείμενο που έχει αποδοθεί σαν τιμή σε μία μεταβλητή.
- ➔ PaintPicture, η οποία σχεδιάζει τα περιεχόμενα ενός αρχείου γραφικών μορφοποίησης .ico, .wmf, .bmp και .dib. Εφαρμόζεται στα αντικείμενα φόρμα, πλαίσιο εικόνας.

6.7. Ιδιότητες γραφικών

Οι φόρμες και αρκετά άλλα αντικείμενα, χαρακτηρίζονται από μία σειρά από ιδιότητες γραφικών, οι οποίες και οι κατηγορίες των οποίων παρουσιάζονται στον παρακάτω πίνακα.

Κατηγορία	Ιδιότητες
Τρεχουσών συντεταγμένων	CurrentX, CurrentY
Επεξεργασίας εμφάνισης	AutoRedraw, ClipControls
Τεχνικών σχεδίασης	DrawMode, DrawStyle, DrawWidth, BorderStyle, BorderWidth
Τεχνικών γεμίσματος	FillColor, FillStyle
Χρωματισμού	BackColor, ForeColor, BorderColor, FillColor
Ορισμού κλίμακας	ScaleLeft, ScaleTop, ScaleHeight, ScaleWidth, ScaleMode

Πίνακας 5: Ιδιότητες γραφικών ανά κατηγορία.

6.7.1. Ιδιότητες τρεχουσών συντεταγμένων

Οι ιδιότητες CurrentX και CurrentY επιστρέφουν ή θέτουν τιμή στην οριζόντια και κατακόρυφη συντεταγμένη αντίστοιχα, του σημείου από το οποίο θα αρχίσει να εφαρμόζεται στη συνέχεια μία μέθοδος γραφικών ή μία διαδικασία εκτύπωσης. Οι ιδιότητες αυτές συναντώνται στα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής. Δεν είναι όμως διαθέσιμες στη φάση της σχεδίασης, παρά μόνο στη φάση εκτέλεσης της εφαρμογής.

Η σύνταξη της εντολής είναι :

όνομα αντικειμένου. CurrentX [= x]

όνομα αντικειμένου. CurrentY [= y]

όπου x και y είναι αριθμοί που προσδιορίζουν τις συντεταγμένες θέσης. Οι αριθμοί αυτοί αντιστοιχούν σε twips, ή στην οποιαδήποτε άλλη κλίμακα μέτρησης χρησιμοποιείται.

6.7.2. Ιδιότητες επεξεργασίας εμφάνισης

Η ιδιότητα AutoRedraw (Αυτόματη Επανασχεδίαση) παίρνει τιμές boolean και χαρακτηρίζει κάθε αντικείμενο φόρμα και πλαίσιο εικόνας. Η AutoRedraw χρησιμοποιείται για την αυτόματη επανασχεδίαση επί της οθόνης γραφικών, τα οποία έχουν δημιουργηθεί με χρήση μεθόδων γραφικών, όταν στη φάση εκτέλεσης της εφαρμογής

✓ ολόκληρα γραφικά ή μέρος τους, αποκαλύπτονται μετά από μετακίνηση άλλων αντικειμένων που τα κάλυπταν

✓ γραφικά αλλάζουν το μέγεθος τους.

Η AutoRedraw έχει εξ ορισμού τιμή False. Αυτό σημαίνει πως κάθε γραφικό που σχηματίστηκε με τη χρήση μεθόδων γραφικών και εμφανίζεται πάνω στη φόρμα, θα χαθεί αν καλυφτεί προσωρινά από ένα άλλο αντικείμενο. Επίσης το ίδιο αποτέλεσμα απώλειας των γραφικών θα συμβεί αν μικρύνει τη φόρμα που τα περιλαμβάνει και στη συνέχεια την επανέλθει στις προηγούμενες διαστάσεις της. Όταν όμως η τιμή της AutoRedraw είναι True, τότε η Visual Basic αναλαμβάνει τη διαχείριση της οθόνης και την επανεμφάνιση των γραφικών όποτε χρειάζεται.

6.7.3. Ιδιότητες τεχνικών σχεδίασης

Η ιδιότητα DrawWidth (Πλάτος Σχεδίασης) προσδιορίζει το πλάτος της γραμμής σχεδίασης για τη μεθόδους γραφικών Circle, Cls, Line, PaintPicture, Point, Print και PSet. Εφαρμόζεται στα αντικείμενα φόρμα, πλαίσιο εικόνας, εκτυπωτής και OLE. Όταν η τιμή της DrawWidth είναι μεγαλύτερη από 1, τότε οι τιμές 1 έως 4 της ιδιότητας DrawStyle, δεν επιφέρουν το αποτέλεσμα που περιγράφουν.

Η ιδιότητα BorderWidth (Πλάτος Περιγράμματος) προσδιορίζει το πάχος του περιγράμματος των αντικειμένων γραμμή και γεωμετρικό σχήμα..

Η ιδιότητα `DrawStyle` (Είδος Σχεδίασης) εφαρμόζεται επί των αντικειμένων φόρμα, πλαίσιο εικόνας και εκτυπωτής και προσδιορίζει το τρόπο εμφάνισης των σχεδιαζόμενων γραμμών. Σημειώνεται πως αν η ιδιότητα `DrawWidth` πάρει τιμές μεγαλύτερες του 1, τότε οι επιλογές 1-4 της ιδιότητας `DrawStyle` παρέχουν το ίδιο αποτέλεσμα.

Η ιδιότητα `BorderStyle` (Είδος Περιγράμματος) έχει διαφορετικό σκοπό και χρήση στα ποικίλα αντικείμενα που απαντάται. Στα αντικείμενα γραμμή και γεωμετρικό σχήμα, η `BorderStyle` έχει την ίδια χρήση με αυτή που έχει η ιδιότητα `DrawStyle`, δηλαδή περιγράφει το είδος της σχεδιαζόμενης γραμμής. Στα αντικείμενα `DBList` και `DBCombo`, η `BorderStyle` προσδιορίζει κατά πόσο το αντικείμενο θα έχει απλό περίγραμμα (`border`), και αν ναι, αν θα εμφανίζονται διάφορα στοιχεία ενός παραθύρου, όπως τίτλος, πλήκτρα μεγιστοποίησης και ελαχιστοποίησης και αν το παράθυρο θα είναι σταθερού ή μεταβλητού μεγέθους. Στα αντικείμενα φόρμα, πλαίσιο εικόνας, εικόνα, πλαίσιο κειμένου, ετικέτα, πλέγμα και `OLE` η `BorderStyle` επιστρέφει ή θέτει τιμή που προσδιορίζει κατά πόσο το αντικείμενο θα έχει απλό περίγραμμα ή δεν θα έχει καθόλου.

Η ιδιότητα `DrawMode` (Τρόπος Σχεδίασης) προσδιορίζει την εμφάνιση των γραφικών που προέρχονται από τη χρησιμοποίηση μεθόδων γραφικών στα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής, καθώς επίσης και την εμφάνιση των αντικειμένων γραμμή και γεωμετρικό σχήμα. Μπορούμε να θεωρήσουμε ότι για τη σχεδίαση κάθε γραφικού μπορεί να χρησιμοποιηθεί ένας μεγάλος αριθμός από πένες σχεδίασης. Η χρήση κάθε μιας από αυτές επιφέρει και διαφορετικά αποτελέσματα στη σχεδίαση. Η τιμή της ιδιότητας `DrawMode`, προσδιορίζει ποιο είναι το οπτικό αποτέλεσμα αν συμβεί ένα γραφικό να δημιουργείται πάνω από ένα άλλο στη φάση της εκτέλεσης της εφαρμογής.

6.7.4. Ιδιότητες τεχνικών γεμίσματος

Η ιδιότητα `FillStyle` (Είδος Γεμίσματος) προσδιορίζει το εσωτερικό σχέδιο ενός αντικειμένου γεωμετρικό σχήμα ή ενός γραφικού που δημιουργείται με τη χρήση μεθόδων γραφικών πάνω σε ένα από τα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής.

Η ιδιότητα `FillColor` (Χρώμα Γεμίσματος) προσδιορίζει το εσωτερικό χρώμα ενός αντικειμένου γεωμετρικό σχήμα ή ενός γραφικού που δημιουργείται με τη χρήση των μεθόδων γραφικών `Line` και `Circle` πάνω σε ένα από τα αντικείμενα φόρμα, πλαίσιο εικόνας και εκτυπωτής. Η εξ ορισμού τιμή της ιδιότητας `FillColor` είναι 0 (Μαύρο), η δε επιλογή των χρωμάτων γίνεται με τη βοήθεια της παλέτας χρωμάτων. Όταν η ιδιότητα `FillStyle` έχει τιμή Διαφανής, η τιμή της ιδιότητας `FillColor` αγνοείται.

6.7.5. Ιδιότητες χρωματισμού

Οι ιδιότητες `BackColor` (Χρώμα Παρασκήνιου) και `ForeColor` (Χρώμα Προσκήνιου) ορίζουν το χρώμα του παρασκήνιου και του προσκήνιου αντίστοιχα για το επιλεγμένο αντικείμενο. Οι δύο αυτές ιδιότητες αναφέρονται σε πάνω από 20 αντικείμενα, μεταξύ των οποίων η φόρμα, η ετικέτα, το πλαίσιο εικόνας, το πλαίσιο κειμένου κλπ.

Η ιδιότητα `BorderColor` (Χρώμα Περιγράμματος), αποδίδει χρώμα στο περίγραμμα ενός αντικειμένου. Η ιδιότητα αυτή χαρακτηρίζει τα αντικείμενα γεωμετρικό σχήμα και γραμμή.

Η απόδοση τιμής στις τρεις αυτές ιδιότητες μπορεί να γίνει στη φάση σχεδίασης της εφαρμογής μέσω του παράθυρου των ιδιοτήτων. Στη φάση της εκτέλεσης της εφαρμογής η απόδοση των τιμών γίνεται με τον εξής τρόπο σύνταξης :

```
Form1.BackColor = color
```

```
Text1.ForeColor = color
```

```
Shape1.BorderColor = color
```

όπου `color` είναι μία τιμή ή μία σταθερά που προσδιορίζει το χρώμα που αποδίδουμε στα εν λόγω αντικείμενα. Αν πρόκειται για τιμή, αυτή εκφράζεται είτε χρησιμοποιώντας τις χρωματικές συναρτήσεις `RGB` και `QBColor`, είτε χρησιμοποιώντας τη δεκαεξαδική έκφραση των χρωματικών αποχρώσεων της παλέτας χρωμάτων.

6.7.6. Ιδιότητες ορισμού κλίμακας

Οι ιδιότητες `ScaleWidth` (Πλάτος Κλίμακας) και `ScaleHeight` (Ύψος Κλίμακας) επιστρέφουν ή θέτουν τιμή στην οριζόντια και κατακόρυφη αντίστοιχα εσωτερική διάσταση ενός αντικειμένου όταν χρησιμοποιούνται σε αυτό μέθοδοι γραφικών. Λέγοντας εσωτερική διάσταση του αντικειμένου, εννοούμε ότι δεν συμπεριλαμβάνεται το πάχος της γραμμής πλαισίου. Οι ιδιότητες `ScaleWidth` και `ScaleHeight` εφαρμόζονται στα αντικείμενα φόρμα και πλαίσιο εικόνας και στο ειδικό αντικείμενο εκτυπωτής.

Οι ιδιότητες `ScaleWidth` και `ScaleHeight` διαφέρουν από τις ιδιότητες `Width` και `Height`, αφού οι δεύτερες αναφέρονται στις εξωτερικές διαστάσεις του αντικειμένου, συμπεριλαμβανομένου δηλαδή και του πάχους της γραμμής του πλαισίου του.

Οι ιδιότητες `ScaleLeft` (Αριστερό Κλίμακας) και `ScaleTop` (Κορυφή Κλίμακας) αποδίδουν τιμές στις, οριζόντια και κατακόρυφη αντίστοιχα, συντεταγμένες της επάνω αριστερής γωνίας ενός αντικειμένου όταν χρησιμοποιούνται μέθοδοι γραφικών. Οι ιδιότητες αυτές χαρακτηρίζουν τα αντικείμενα φόρμα, πλαίσιο εικόνας και το ειδικό αντικείμενο εκτυπωτής.

Οι ιδιότητες `ScaleLeft` και `ScaleTop` δεν είναι ταυτόσημες με τις ιδιότητες `Left` και `Top` οι οποίες απαντώνται σε περισσότερα από 20 αντικείμενα. Η ιδιότητα

Left δηλώνει την απόσταση μεταξύ της αριστερής εσωτερικής ακμής ενός αντικειμένου και της αριστερής ακμής του αντικειμένου που το περιέχει, συννηθέστερα της φόρμας. Αντίστοιχα η ιδιότητα Top δηλώνει την απόσταση μεταξύ της επάνω εσωτερικής ακμής του αντικειμένου και της επάνω ακμής του αντικειμένου που το περιέχει. Για τη φόρμα οι τιμές των ιδιοτήτων Left και Top εκφράζονται πάντα σε twips, ενώ για τα άλλα αντικείμενα εκφράζονται σύμφωνα με το σύστημα συντεταγμένων που ισχύει για το αντικείμενο στο οποίο περιέχονται.

7. ΠΟΛΥΜΕΣΑ (MULTIMEDIA)

Η Visual Basic μπορεί να συμπεριλάβει στις εφαρμογές της τόσο δεδομένα κειμένου, όσο και δεδομένα ήχου, εικόνας, προσομοίωσης κίνησης και video. Κύρια χάρη στις διαδικασίες OLE και DDE που υποστηρίζει, η Visual Basic μπορεί να επιλεγεί σαν εργαλείο συγγραφής (authoring tool), Interactive Multimedia και Hypermedia εφαρμογών. Η χρησιμοποίηση Multimedia δεδομένων στις εφαρμογές, συνεπάγεται μεγαλύτερες απαιτήσεις σε υλικό, τόσο σε ταχύτητα επεξεργασίας, όσο και σε μνήμη RAM, αλλά και σε αποθηκευτικές δυνατότητες. Τα αρχεία κυρίως video (πχ. .avi) και προσομοίωσης κίνησης (πχ. .flc), αλλά και αυτά ήχου (πχ. .wav) και στη συνέχεια εικόνας (πχ. .bmp) είναι πολύ μεγάλου μεγέθους. Κατά συνέπεια η χρησιμοποίηση τους χωρίς να έχουν εξασφαλιστεί οι απαιτούμενοι υλικοί πόροι του συστήματος, θα επιφέρει αρνητικά αποτελέσματα στην εφαρμογή. Ένα video που συμπεριλάβαμε σε μία εφαρμογή, αλλά στη φάση της εκτέλεσης εμφανίζεται με μία τρεμώδη και σπαστή κίνηση, σίγουρα δεν μπορεί να καταχωρηθεί στα θετικά στοιχεία της εφαρμογής.

7.1. Multimedia εργαλεία της Visual Basic

Τα εργαλεία (controls) που παρέχει η Visual Basic προς χρήση για δημιουργία Multimedia εφαρμογών είναι αυτά τα οποία μπορούν να δεχτούν δεδομένα κειμένου, εικόνας, ήχου, animation και video. Αυτά τα εργαλεία είναι :

⇒ Εργαλείο πλαίσιο εικόνας (PictureBox) : Το εργαλείο αυτό, μέσω της ιδιότητάς του Picture, μπορεί να χρησιμοποιηθεί για την εμφάνιση εικόνων διαφορετικής μορφοποίησης (πχ. bmp, dib, wmf, ico κλπ). Αν η εικόνα είναι μεγαλύτερη από το αντικείμενο πλαίσιο εικόνας, τότε ένα τμήμα μόνο της εικόνας εμφανίζεται. Αν η εικόνα είναι μικρότερη από το αντικείμενο πλαίσιο εικόνας, τότε προβάλλεται ολόκληρη, ενώ μένει κενός ο υπόλοιπος χώρος του πλαισίου εικόνας. Το αντικείμενο πλαίσιο εικόνας επαναπροσδιορίζει τις διαστάσεις του σύμφωνα με τις διαστάσεις της εικόνας, όταν στην ιδιότητά του AutoSize έχει τεθεί τιμή True.

⇒ Εργαλείο εικόνα (Image) : Η χρήση του εργαλείου εικόνα είναι παρόμοια με αυτή του εργαλείου πλαίσιο εικόνας. Εμφανίζει της ίδιας μορφοποίησης αρχεία εικόνας και οι διαστάσεις του προσαρμόζονται στις διαστάσεις της εικόνας όταν στην ιδιότητα Stretch τεθεί τιμή True.

⇒ Εργαλείο ετικέτα (Label): Από την άποψη πως όταν στην ιδιότητα BackStyle αποδώσουμε τιμή Transparent, το κείμενο που έχει γραφτεί στο αντικείμενο ετικέτα, μπορεί να προβληθεί πάνω από κάποιο άλλο αντικείμενο, δημιουργώντας την εντύπωση των τίτλων ή των υπότιτλων πάνω από γραφικά ή video, το εργαλείο ετικέτα μπορεί να θεωρηθεί σαν ένα Multimedia εργαλείο.

⇒ Εργαλείο πολυμέσα (MMControl) : Το εργαλείο αυτό παρέχει το βασικό interface για το χειρισμό όλων των Multimedia συσκευών (κασετόφωνο, CD-player, video κλπ). Θέτοντας στις αντίστοιχες ιδιότητες τις κατάλληλες τιμές, μπορούμε να επιλέξουμε εκείνα τα πλήκτρα που θέλουμε να είναι ορατά στη φάση της εκτέλεσης και να ανταποκρίνονται σε λειτουργίες.

⇒ Εργαλείο OLE : Σαν multimedia εργαλείο μπορεί βέβαια να καταχωρηθεί και το εργαλείο OLE, ανεξάρτητα από το γεγονός πως η χρήση του δεν στοχεύει αποκλειστικά στην πρόσβαση σε multimedia τύπου εφαρμογές. Το εργαλείο OLE χρησιμοποιείται ευρέως για την δημιουργία απλών και εύκολων multimedia εφαρμογών σε περιβάλλον Visual Basic.

Όπως όλα τα αντικείμενα της Visual Basic, έτσι και το αντικείμενο OLE χαρακτηρίζεται από μία σειρά από ιδιότητες. Από αυτές στη συνέχεια παρουσιάζεται αναλυτικά η ιδιαίτερα σημαντική ιδιότητα Action, η οποία είναι διαθέσιμη μόνο κατά τη φάση της εκτέλεσης της εφαρμογής.

Κατά τη διάρκεια εκτέλεσης μιας εφαρμογής μπορούμε με ένα αντικείμενο OLE να πραγματοποιήσουμε μία σειρά από διαφορετικές ενέργειες, ανάλογα με την τιμή που αποδίδουμε στη ιδιότητα Action. Οι σημαντικότερες τιμές που μπορεί να πάρει η ιδιότητα Action είναι οι παρακάτω :

✓ 7 Αυτή η τιμή ενεργοποιεί το αντικείμενο OLE. Το τι ακριβώς θα συμβεί όταν το αντικείμενο θα ενεργοποιηθεί εξαρτάται από τη τιμή της ιδιότητάς του Verb. Κάθε τύπος αντικείμενου μπορεί να υποστηρίζει το δικό του σύνολο από verbs, που ουσιαστικά δεν είναι τίποτε άλλο από το σύνολο των ενεργειών που μπορεί να πραγματοποιήσει το αντικείμενο. Για να δούμε την λίστα αυτών των ενεργειών, θα πρέπει στην ιδιότητα AutoVerbMenu να έχουμε θέσει τιμή True και στη φάση της εκτέλεσης της εφαρμογής να πατήσουμε το δεξί πλήκτρο του ποντικιού όταν βρισκόμαστε πάνω από αντικείμενο OLE. Αν λοιπόν στην ιδιότητα Verb έχουμε αποδώσει τιμή 2, αυτό σημαίνει πως όταν ενεργοποιήσουμε το αντικείμενο μέσω κώδικα (ΌνομαOLE.Action = 7), τότε αυτό

που θα συμβεί είναι η δεύτερη ενέργεια που παρατίθεται στη λίστα των ενεργειών. Η εξ ορισμού τιμή της ιδιότητας Verb είναι 0, πράγμα που σημαίνει πως αυτό το οποίο συμβαίνει όταν το αντικείμενο ενεργοποιείται, είναι η πιο συνηθισμένη ενέργεια, η οποία βέβαια εξαρτάται από το είδος της διασυνδεδεμένης εφαρμογής (π.χ. edit για κείμενο, play για ήχο και video).

- ✓ 9 Αυτή η τιμή σταματάει τη σύνδεση, εφ' όσον πρόκειται για ένα ενσωματωμένο (embedded) αντικείμενο OLE, με την διασυνδεδεμένη εξωτερική εφαρμογή την οποία και κλείνει, ενώ δεν επιφέρει κανένα ουσιαστικό αποτέλεσμα στην περίπτωση του συνδεδεμένου (linked).

7.2. Ήχος

Παρ' ότι η Visual Basic προσφέρει έναν πραγματικά μεγάλο αριθμό συναρτήσεων, που προσθέτουν στην εφαρμογή πολλά multimedia χαρακτηριστικά και καθιστούν ευκολότερο τον τρόπο επικοινωνίας με τον χρήστη, εντούτοις φαίνεται να υστερεί σε επίπεδο διαχείρισης ήχου. Η QuickBasic με τις συναρτήσεις της Sound () και Play (), παρέχει τη δυνατότητα για τη δημιουργία ενδιαφερόντων ηχητικών εφέ. Αντίθετα, η Visual Basic φαίνεται να παρέχει πολύ λιγότερες δυνατότητες σε ότι αφορά τον ήχο, αφού δεν έχει να παρουσιάσει παρά μία μόνο εντολή, την Beep ().

Στην πραγματικότητα όμως εφαρμογές της Visual Basic μπορούν να εμπλουτιστούν με κάθε είδους ηχητικά εφέ. Υπάρχουν δύο τρόποι για την προσθήκη ηχητικών στοιχείων στις εφαρμογές. Ο πρώτος είναι με τη χρήση DLLs και ο δεύτερος με τη δημιουργία OLE αντικειμένων. Ο δεύτερος είναι οπωσδήποτε απλούστερος και κατά συνέπεια το εύρος δράσης του χρήστη σε ηχητικό επίπεδο είναι περιορισμένο.

7.3. Προσομοίωση κίνησης (animation)

Ανάμεσα στα πολλά ενδιαφέροντα χαρακτηριστικά που μπορεί να παρουσιάσει η Visual Basic στις εφαρ-

μογές της, είναι η δυνατότητα δημιουργίας προσομοίωσης κίνησης (animation). Με τον όρο προσομοίωση κίνησης, εννοούμε την προβολή μιας σειράς από εικόνες, που προκαλούν στο χρήστη την αίσθηση της κίνησης. Υπάρχουν τρεις βασικοί τρόποι για τη δημιουργία προσομοίωσης κίνησης : frame-based, object και palette animation.

- ⇒ Προσομοίωση κίνησης βασισμένη σε πλαίσια (frame-based animation), είναι εκείνη η τεχνική που στηρίζεται στην γρήγορη, συνεχή προβολή εικόνων που παρουσιάζουν μικρές διαφορές μεταξύ τους, δημιουργώντας κατά αυτό το τρόπο την αίσθηση της κίνησης.
- ⇒ Προσομοίωση κίνησης με αντικείμενα (object animation), είναι εκείνη η τεχνική που χρησιμοποιούνται διαφορετικά, ανεξάρτητα μεταξύ τους, γραφικά αντικείμενα για τη δημιουργία της παρουσίας. Αυτό το είδος προσομοίωσης κίνησης στηρίζεται αυστηρά στο χρόνο (time-based).
- ⇒ Προσομοίωση κίνησης παλέτας (palette animation) είναι εκείνη η τεχνική που στηρίζεται στην αλλαγή των χρωμάτων μιας εικόνας, δημιουργώντας περισσότερο την αίσθηση της χρονικής κίνησης. Για παράδειγμα, αλλάζουν τα χρώματα μιας εικόνας, δίνοντας την εντύπωση πως πέρασε η μέρα και νύχτωσε.

Με τη Visual Basic, μπορούμε να δημιουργήσουμε και να χειριστούμε και τα τρία είδη προσομοίωσης κίνησης. Η ίδια η Visual Basic διαθέτει μία σειρά από εικόνες και εικονίδια, πολλά από τα οποία εμφανίζονται ανά ζευγάρια, και τα οποία μπορούν να χρησιμοποιηθούν σε απλές εφαρμογές προσομοίωσης κίνησης.

7.4. Video

Η Visual Basic μπορεί να συμπεριλάβει video στις εφαρμογές της, εμπλουτίζοντας τες ακόμη περισσότερο. Η χρήση video μπορεί να γίνει με τη χρησιμοποίηση του αντικειμένου OLE ή με το ActiveX εργαλείο MCI.