

Ηλίας Χατζηθεοδωρίδης

Ανάπτυξη Εφαρμογών Πληροφορικής

Σημειώσεις για το μάθημα του 3ου Εξαμήνου

Κεφάλαιο 1^ο

Εισαγωγή στην Visual Basic

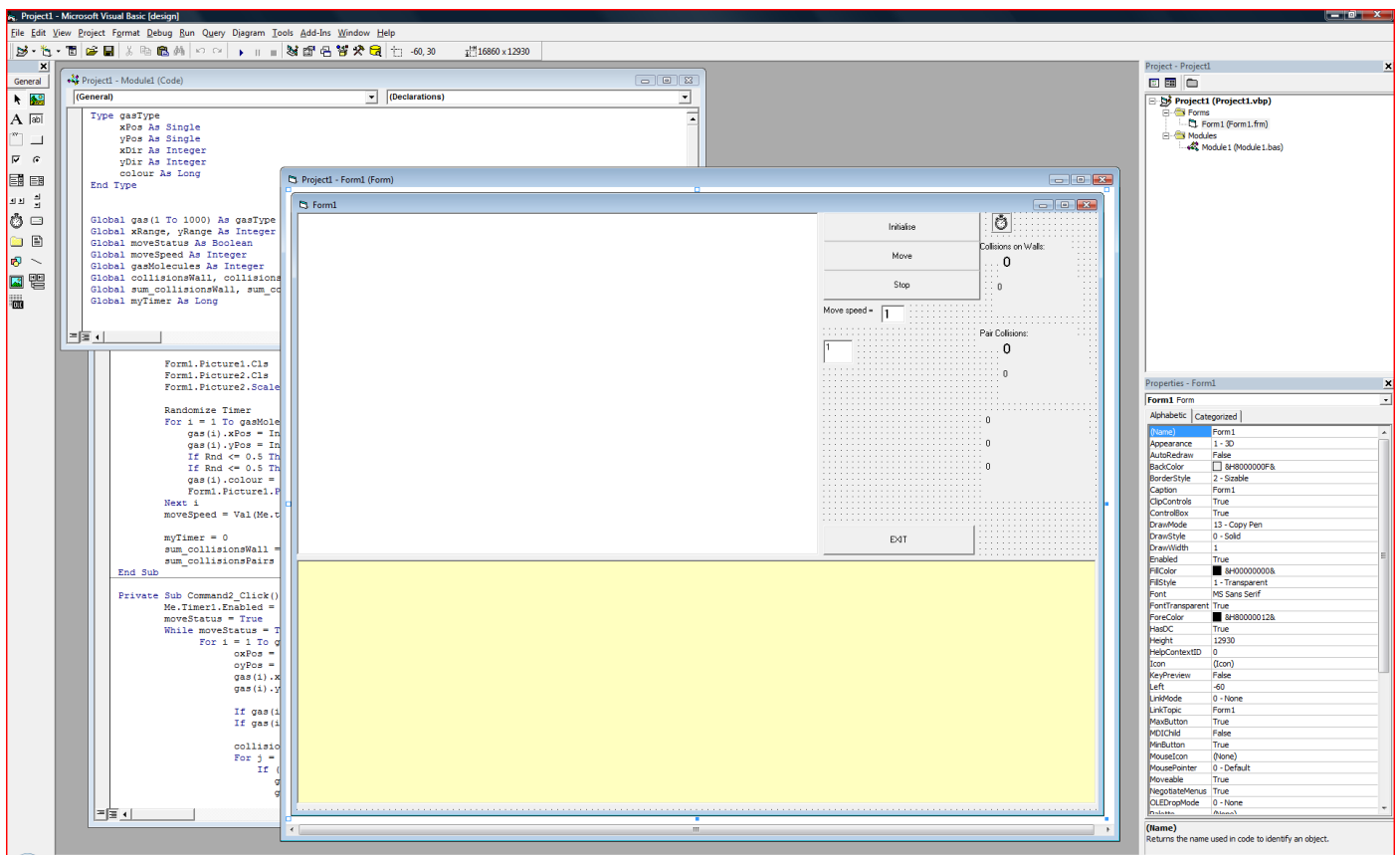
Προηγούμενες γνώσεις

Στην προηγούμενη σειρά μαθημάτων μάθατε πώς να χειρίζεστε το περιβάλλον της γλώσσας, βασικές εντολές για τον καθορισμό των μεταβλητών, συνθηκών, εντολών εκτύπωσης καθώς και κάποια βασικά στοιχεία για την σχεδίαση γραφικών. Σε αυτό το κεφάλαιο θα γίνει μια ανασκόπηση κάποιων από αυτές τις εντολές με την μορφή παραδείγματος.

Προσομοίωση αερίου σε κλειστό χώρο

Το παράδειγμα με το οποίο θα ξεκινήσουμε δεν προσπαθεί να προσομοιάσει αέρια σε κλειστό χώρο χρησιμοποιώντας πλήρως φυσικούς κανόνες αλλά να βάλει τις βάσεις της γλώσσας και του περιβάλλοντος πάνω στις οποίες όποιος ενδιαφέρεται θα στήσει το πραγματικό πρόβλημα.






Στο παρακάτω σχήμα βλέπετε το περιβάλλον της γλώσσας. Αριστερά είναι τα εργαλεία διάφορων αντικειμένων (object tools), στην μέση βλέπετε τα παράθυρα προγραμματισμού καθώς και μορφοποίησης του παραθύρου της εφαρμογής σας ή αλλιώς φόρμας (form), δεξιά επάνω τα διάφορα αρχεία που συνδυαζόμενα συνθέτουν την εφαρμογή σας (Project structure, project files), ενώ κάτω δεξιά μια λίστα ονομάτων που αποτελεί τις ιδιότητες του αντικειμένου που έχετε επιλέξει τελευταίο με το ποντίκι (οι μεταβλητές του).



Το περιβάλλον της Visual Basic είναι αντικειμενοστραφές (object oriented) ωστόσο η γλώσσα αυτή καθαυτή δεν περιέχει όλα τα στοιχεία που περιέχουν καθαρά αντικειμενοστραφείς γλώσσες όπως ή C++. Βέβαια, για το μεγαλύτερο ποσοστό εφαρμογών οι δυνατότητες που σας δίνονται είναι περισσότερο από αρκετές.

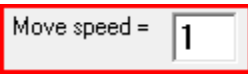
Τα πρώτα αντικείμενα που θα γνωρίσετε είναι τα διάφορα εργαλεία που σας δίνει η γλώσσα με την μορφή εικονιδίων και μπορείτε να τα βάλετε πάνω σε μια φόρμα πολλές φορές το καθένα αλλά με διαφορετικά ονόματα. Υπάρχει η δυνατότητα βέβαια να δημιουργήσετε ένα ομαδοποιημένο σύνολο αντικειμένων με το ίδιο όνομα αλλά με διαφορετικούς δείκτες (Array of objects). Στο επόμενο σχήμα βλέπετε τα έτοιμα αντικείμενα που είναι στην διάθεσή σας:




Στην εφαρμογή μας θα χρησιμοποιήσουμε το αντικείμενο επιφάνειας γραφικών και εικόνων (Picture ) , το εργαλείο δημιουργίας κουμπιών (Button, ) , το εργαλείο προβολής κειμένου (Label, ) , το εργαλείο επεξεργασίας κειμένου (TextBox, ) και τέλος, το εργαλείο του χρονομετρητή (Timer, ) .

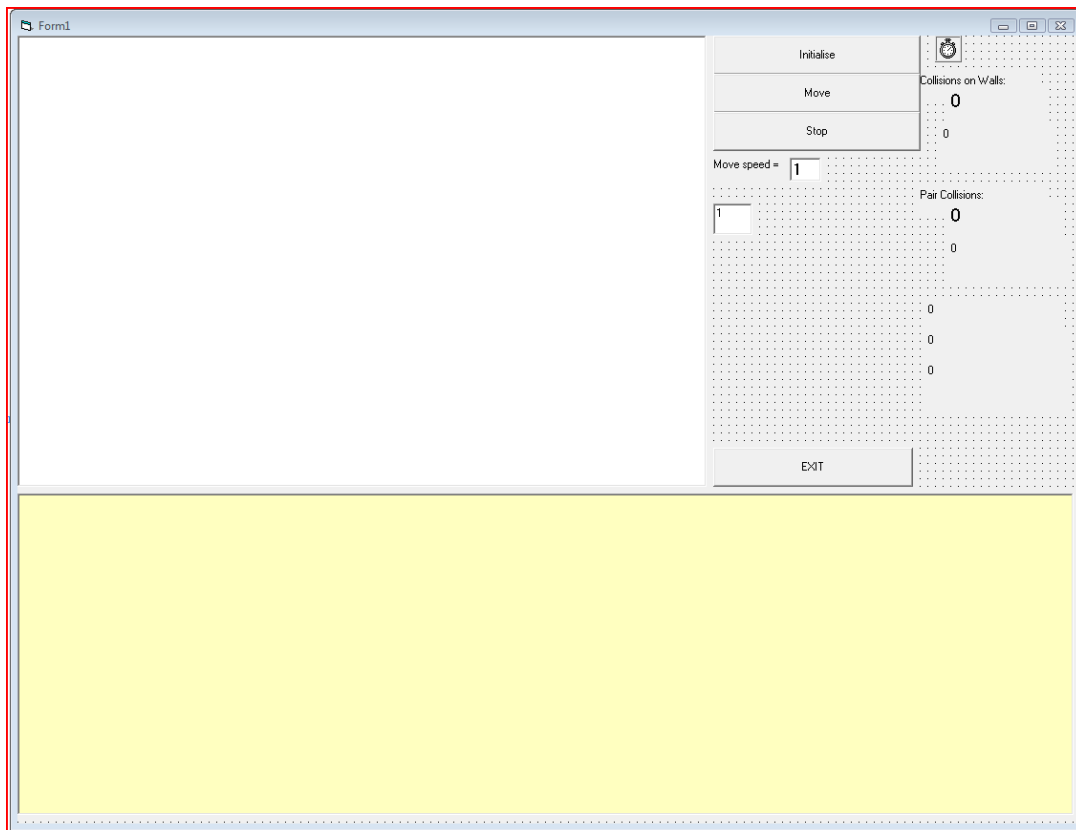
Μεταφέρουμε αυτά τα εργαλεία κάνοντας κλικ στο καθένα από αυτά και μετά με το ποντίκι ανοίγουμε περιοχή στο μέγεθος που θέλουμε πάνω στην φόρμα. Κάποια εργαλεία μπορούν να καταλαμβάνουν οποιαδήποτε επιφάνεια, όπως το Picture, άλλα όμως έχουν προκαθορισμένο μέγεθος, όπως ο Timer.

Ακολουθεί μια πρόχειρη σχεδίαση των απαραίτητων εργαλείων για την εργασία που θέλουμε. Η άσπρη μεγάλη περιοχή είναι η επιφάνεια όπου τα μόρια του αερίου θα κινούνται σε πραγματικό χρόνο, θα συγκρούονται με τα τοιχώματα ή μεταξύ τους και θα αλλάζουν διεύθυνση. Η κίτρινη περιοχή θα παρουσιάζει με γραφικό τρόπο τις συγκρούσεις αυτές ανά δευτερόλεπτο. Το γράφημα αυτό θα αλλάζει με τον χρόνο αλλά και με την ταχύτητα

κίνησης των μορίων που ορίζουμε εμείς μεταβάλλοντας το . Αυτό μπορεί να θεωρηθεί ότι είναι απόλυτα ανάλογο με την αύξηση της θερμοκρασίας, είναι σαν να ζεσταίνουμε το αέριο δηλαδή.

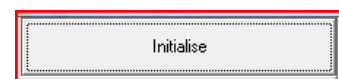
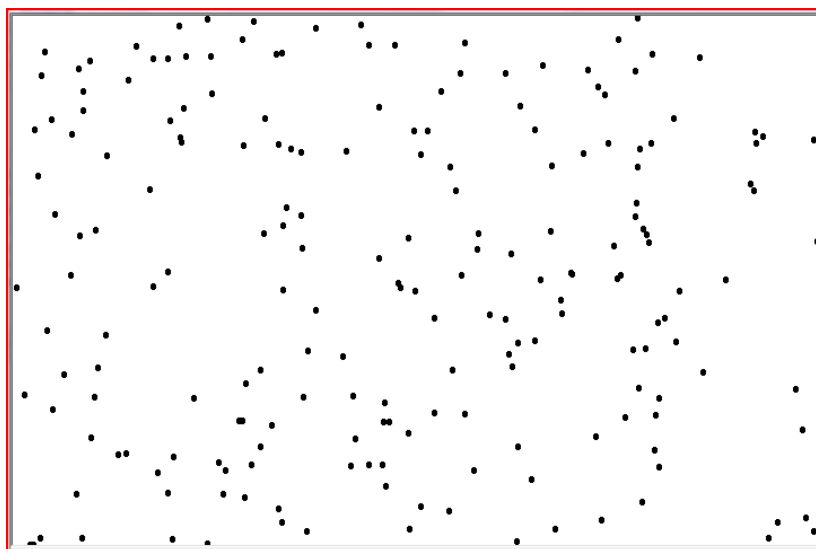
Δεξιά, μια σειρά από Label objects μας πληροφορούν για την μεταβολή των παραμέτρων που μετράμε, π.χ συγκρούσεις ανά δευτερόλεπτο.

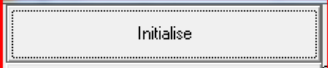
Το ρολόι  προσθέτει ουσιαστικά την λειτουργία μέτρησης του πραγματικού χρόνου και χρησιμοποιεί την τεχνολογία των events (συμβάντων) που μπορούμε εμείς με την σειρά μας να εκμεταλλευτούμε ώστε να κάνουμε κάποιες διεργασίες ή υπολογισμούς σε τακτά χρονικά διαστήματα πραγματικού χρόνου.



Το παραπάνω περιβάλλον μπορείτε εσείς σαν άσκηση να το διαμορφώσετε ώστε να είναι πιο εμφανές και λειτουργικό. Εδώ εμείς θα δούμε τώρα την λειτουργία του προγράμματος και τον κώδικα που εκτελείται.

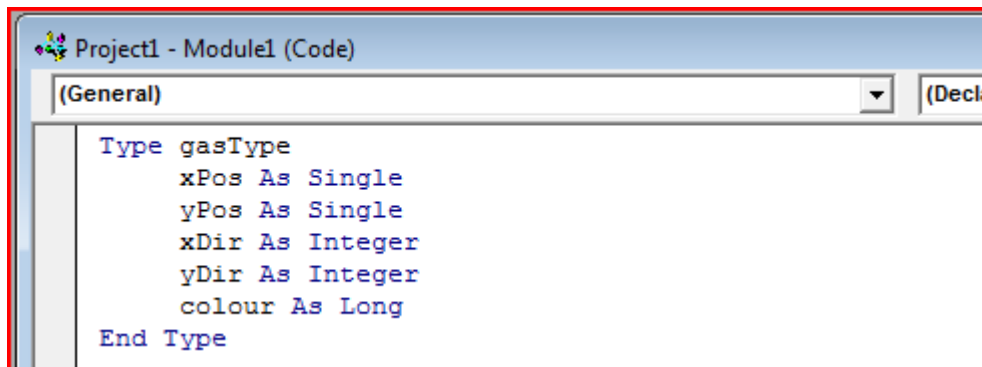
Στο επόμενο σχήμα βλέπουμε την άσπρη περιοχή, δηλαδή το κουτί με το αέριο, γεμισμένο με μόρια αερίου ενώ έχουμε παγώσει τον χρόνο οπότε δεν υπάρχει κίνηση.



Οι τελείες αντιστοιχούν στα μόρια του αερίου. Πατήσαμε το κουμπί  για να δημιουργηθεί αυτό. Ωστόσο, πρέπει να εξηγήσουμε το πώς έγινε.

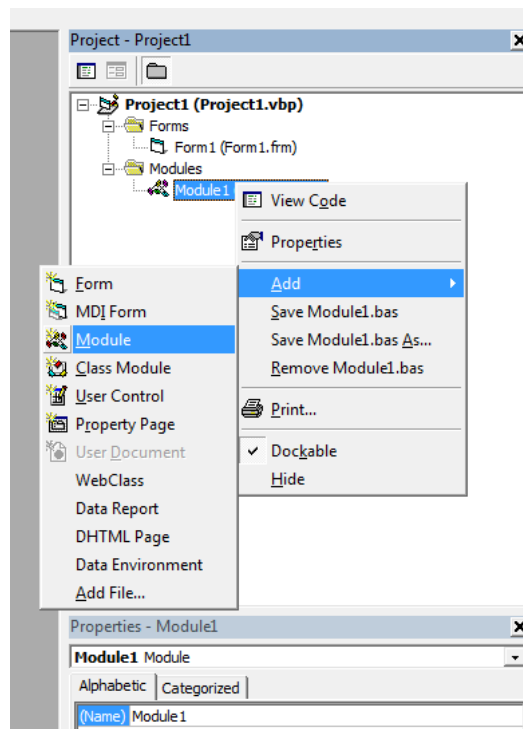
Κάθε μόριο αέρα θεωρούμε ότι περιέχει κάποια πληροφορία που εκφράζεται με κάποιες παραμέτρους. Για παράδειγμα, η θέση του αερίου στον δισδιάστατο εδώ χώρο πρέπει να εκφραστεί μέσα από κάποιες μεταβλητές που να περιέχουν αριθμούς. Η ταχύτητα κίνησης επίσης στις δύο διαστάσεις απαιτεί το ίδιο. Εμείς έχουμε βάλει και μια παράμετρο χρώματος κουκίδων που εάν αργότερα θελήσουμε μπορούμε να την δούμε σαν μόρια

διαφορετικών αερίων, καθένα με διαφορετικό χρώμα. Τις παραμέτρους αυτές τις εκφράζουμε συνολικά με την δομή **Type ... End Type** όπως φαίνεται στο επόμενο σχήμα. Η μορφή αυτή σύνταξης έχει στοιχεία δομημένου καθορισμού των μεταβλητών αλλά δεν είναι πλήρως αντικειμενοστραφείς.

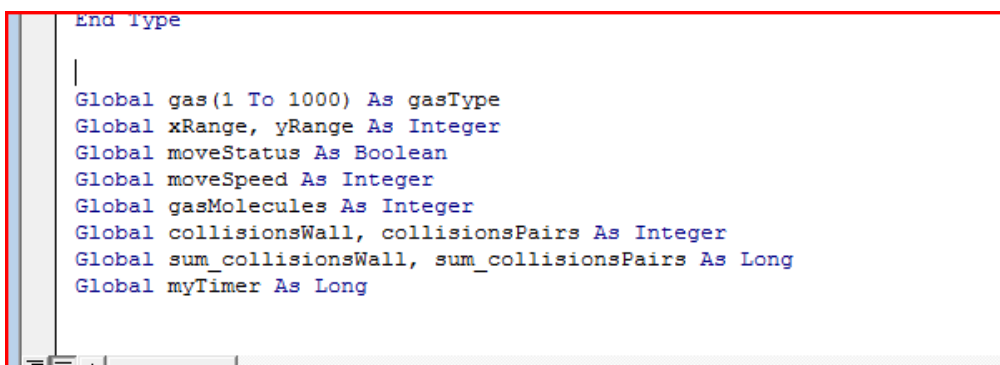


```
Project1 - Module1 (Code)
(General)
Type gasType
  xPos As Single
  yPos As Single
  xDir As Integer
  yDir As Integer
  colour As Long
End Type
```

Συνήθως, τους ορισμούς αυτούς τους δίνουμε μέσα σε περιοχές κώδικα που το περιβάλλον της γλώσσας τα ονομάζει modules. Στο επόμενο σχήμα φαίνεται πως δημιουργούμε ένα νέο module. Ακόμη και ένα module είναι αντικείμενο για το περιβάλλον της γλώσσας ενισχύοντας την άποψη ότι το περιβάλλον είναι πλήρως αντικειμενοστραφές.



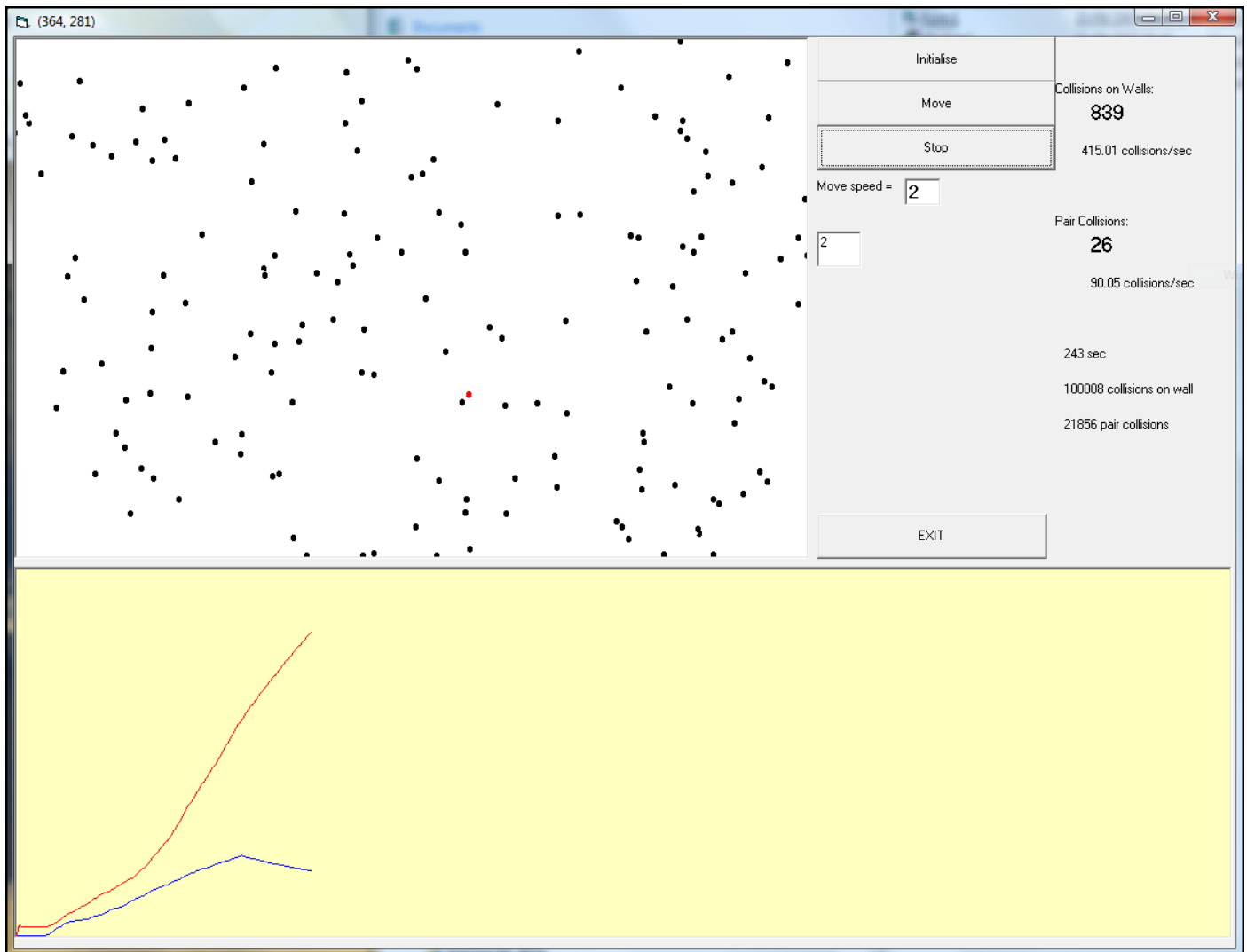
Έπειτα, μέσα στο module αυτό γράφουμε επιπλέον κώδικα που καθορίζει άλλες μεταβλητές του περιβάλλοντός μας (βλέπε επόμενο σχήμα). Για παράδειγμα



```
End Type

Global gas(1 To 1000) As gasType
Global xRange, yRange As Integer
Global moveStatus As Boolean
Global moveSpeed As Integer
Global gasMolecules As Integer
Global collisionsWall, collisionsPairs As Integer
Global sum_collisionsWall, sum_collisionsPairs As Long
Global myTimer As Long
```

Το επόμενο σχήμα δείχνει την οθόνη του προγράμματος αφού αυτό έχει τρέξει για κάποιο χρόνο και μετά το παγώσαμε. Οι καμπύλες στο κάτω μέρος δείχνουν την συχνότητα κρούσεων με τα τοιχώματα (κόκκινη) καθώς επίσης και τη συχνότητα κρούσεων μεταξύ των αερίων (μπλε). Η αύξηση οφείλεται στην αύξηση της ταχύτητας των ατόμων που έγινε σταδιακά από τον χρήστη μέσω στο χρονικό διάστημα αυτό. Εκφράζοντας καλύτερα τις εξισώσεις αερίων θα μπορούσαμε να πάρουμε ανάλογες καμπύλες που να δείχνουν την μεταβολή πίεσης που ασκεί το αέριο στα τοιχώματα με αύξηση της θερμοκρασίας (αύξηση δηλαδή της ταχύτητας κίνησης των ατόμων).



Ακολουθεί ο κώδικας που απαιτείται για να αρχικοποιήσει τις οθόνες και τις τιμές –ακόμη και τις αρχικές θέσεις των ατόμων του αερίου. Από αυτόν τον κώδικα το πρώτο μέρος βάζει σε μεταβλητές το μέγεθος (συντεταγμένες) του χώρου (επιφάνειας εδώ) που μπορούν να κινούνται τα άτομα (σε pixels ή άλλες μονάδες).

```

init positions
xRange = Form1.Picture1.ScaleWidth
yRange = Form1.Picture1.ScaleHeight
    
```

Έπειτα βεβαίως προσδιορίζει τον μέγιστο αριθμό ατόμων/μορίων που θα περιέχονται μέσα στον χώρο (επιφάνεια), έπειτα καθαρίζει το γραφικό περιβάλλον και ορίζει το μέγεθος της κατώτερης επιφάνειας με την μέθοδο Scale.

```

gasMolecules = 200

Form1.Picture1.Cls
Form1.Picture2.Cls
Form1.Picture2.Scale (0, 500)-(1000, 0)
    
```

```

Project1 - Form1 (Code)
Command1 Click
Private Sub Command1_Click()
    'init positions
    xRange = Form1.Picture1.ScaleWidth
    yRange = Form1.Picture1.ScaleHeight

    gasMolecules = 200

    Form1.Picture1.Cls
    Form1.Picture2.Cls
    Form1.Picture2.Scale (0, 500)-(1000, 0)

    Randomize Timer
    For i = 1 To gasMolecules
        gas(i).xPos = Int(Rnd * xRange)
        gas(i).yPos = Int(Rnd * yRange)
        If Rnd <= 0.5 Then gas(i).xDir = -1 Else gas(i).xDir = 1
        If Rnd <= 0.5 Then gas(i).yDir = -1 Else gas(i).yDir = 1
        gas(i).colour = RGB(0, 0, 0)
        Form1.Picture1.PSet (gas(i).xPos, gas(i).yPos), gas(i).colour
    Next i
    moveSpeed = Val(Me.txtMoveSpeed)

    myTimer = 0
    sum_collisionsWall = 0
    sum_collisionsPairs = 0
End Sub

```

Τελικά ο κώδικας που ακολουθεί βάζει αρχικές συντεταγμένες στα μόρια/άτομα αερίου και τα ζωγραφίζει στην οθόνη. Καθορίζει επίσης και την ταχύτητα, που την παίρνει από ένα TextBox της φόρμας μας ενώ αρχικοποιεί τον χρόνο αλλά και το σύνολο συγκρούσεων με τα τοιχώματα αλλά και μεταξύ τους.

```

Private Sub Command2_Click()
    '-----start animation
    Me.Timer1.Enabled = True
    moveStatus = True
    While moveStatus = True
        For i = 1 To gasMolecules
            oxPos = gas(i).xPos
            oyPos = gas(i).yPos
            gas(i).xPos = gas(i).xPos + gas(i).xDir * moveSpeed
            gas(i).yPos = gas(i).yPos + gas(i).yDir * moveSpeed

            If gas(i).xPos >= xRange Or gas(i).xPos <= 0 Then gas(i).xDir = -gas(i).xDir: collisionsWall = collisionsWall + 1
            If gas(i).yPos >= yRange Or gas(i).yPos <= 0 Then gas(i).yDir = -gas(i).yDir: collisionsWall = collisionsWall + 1

            collision = False
            For j = 1 To i - 1
                If (gas(i).xPos - gas(j).xPos) ^ 2 + (gas(i).yPos - gas(j).yPos) ^ 2 < moveSpeed ^ 2 Then
                    gas(i).xDir = -gas(i).xDir
                    gas(j).xDir = -gas(j).xDir
                    collision = True
                    collisionsPairs = collisionsPairs + 1
                Exit For
            End If
        Next j
        If collision = False Then
            For j = i + 1 To gasMolecules
                If (gas(i).xPos - gas(j).xPos) ^ 2 + (gas(i).yPos - gas(j).yPos) ^ 2 < moveSpeed ^ 2 Then
                    gas(i).xDir = -gas(i).xDir
                    gas(j).xDir = -gas(j).xDir
                    collisionsPairs = collisionsPairs + 1
                Exit For
            End If
        Next j
        End If

        Form1.Picture1.PSet (oxPos, oyPos), RGB(255, 255, 255)
        Form1.Picture1.PSet (gas(i).xPos, gas(i).yPos), gas(i).colour
    Next i
    DoEvents
Wend
End Sub

```

Στην προηγούμενη εικόνα έχουμε τον κώδικα που τοποθετήσαμε στο κουμπί Move της φόρμας μας. Εδώ ο κώδικας αυτός ελέγχει την κίνηση αλλά και τις συγκρούσεις των μορίων/ατόμων και φυσικά περιλαμβάνει τις κατάλληλες εντολές για να αλλάζει την διεύθυνση κίνησης. Μελετήστε τον κώδικα αυτό προσεκτικά για να καταλάβετε την λειτουργία του. Δακτυλογραφήστε τον μόνοι σας ώστε να θυμώσατε καλύτερα το κάθε μέρος του –είναι σημαντικό.

Ακολουθεί τώρα όλος ο υπόλοιπος κώδικας που περιλαμβάνεται στα άλλα κουμπιά, όπως για παράδειγμα η αλλαγή ταχύτητας των μορίων (αλλαγή θερμοκρασίας).

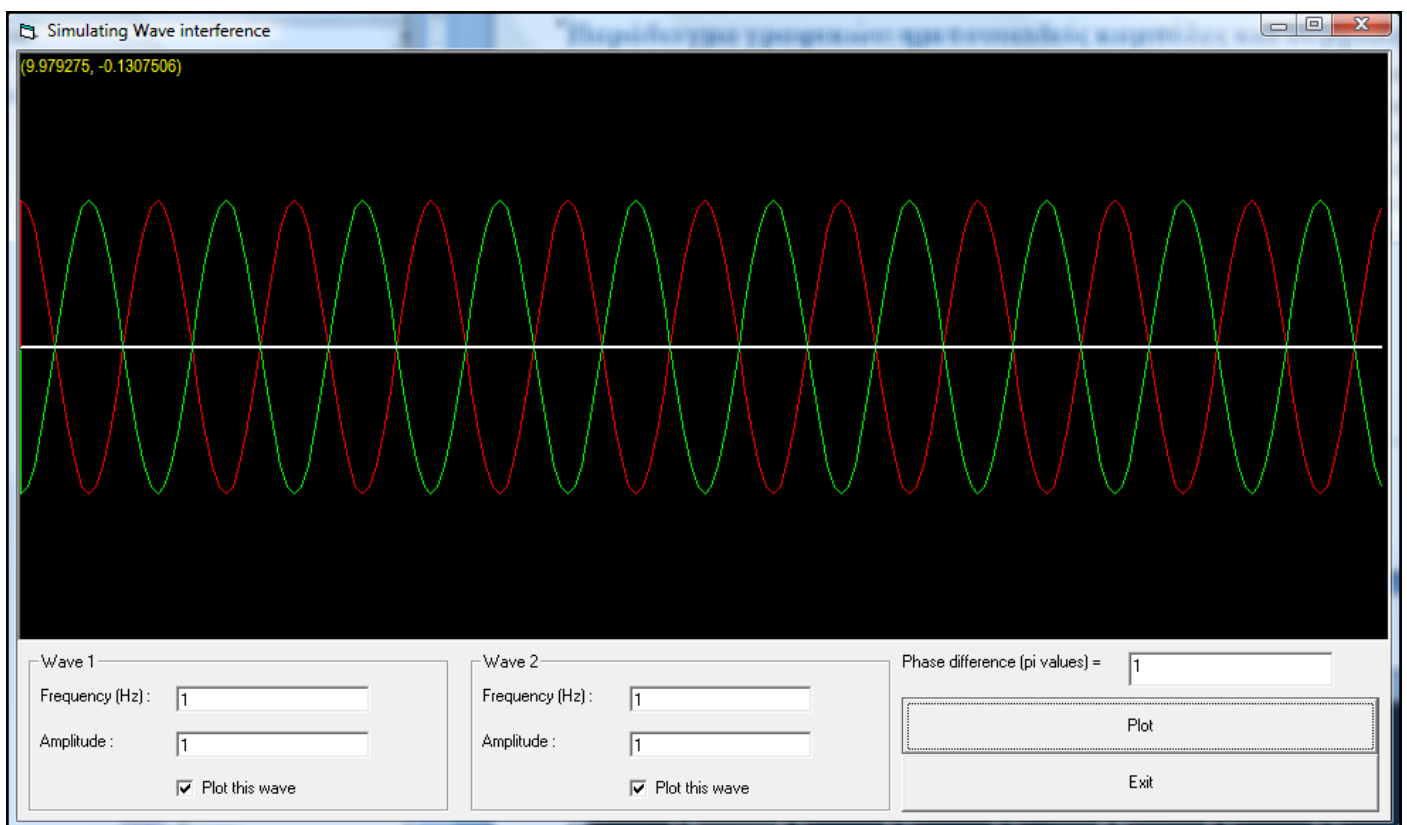
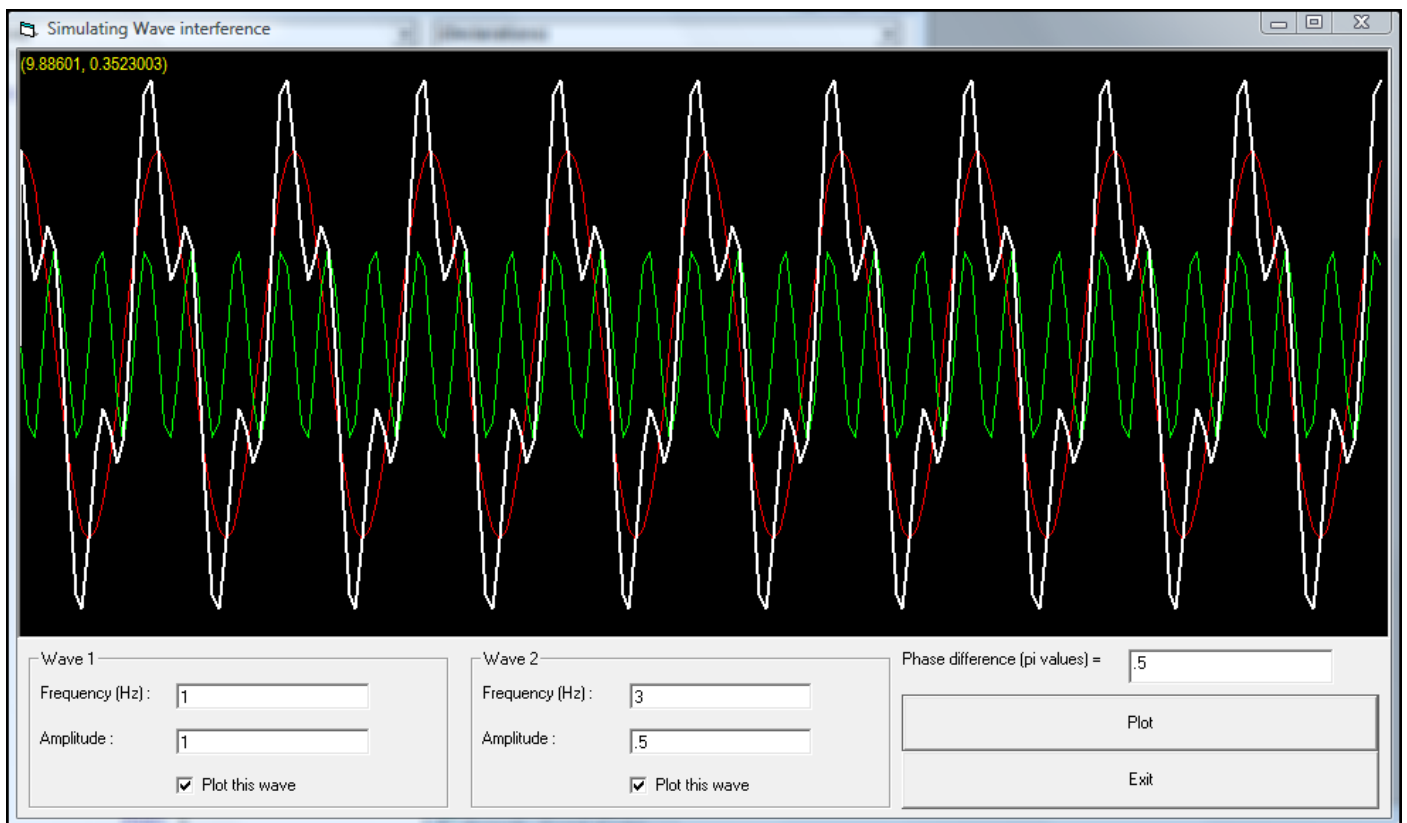
```
Private Sub Command3_Click()  
    '----- pause animation  
    moveStatus = False  
    Me.Timer1.Enabled = False  
End Sub  
  
Private Sub Command4_Click()  
    '----- stop and exit programme  
End  
End Sub  
  
Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)  
    Form1.Caption = "(" & X & ", " & Y & ")"  
End Sub  
  
Private Sub Text1_Change()  
    r = Val(Me.Text1)  
    gas(r).colour = RGB(255, 0, 0)  
End Sub  
  
Private Sub Timer1_Timer()  
    '----- timer control of statistics and statistical graphs  
    myTimer = myTimer + 1  
    Me.lblMyTimer = myTimer & " sec"  
    Me.lblsw = sum_collisionsWall & " collisions on wall"  
    Me.lblsp = sum_collisionsPairs & " pair collisions"  
    sum_collisionsWall = sum_collisionsWall + collisionsWall  
    sum_collisionsPairs = sum_collisionsPairs + collisionsPairs  
    Me.lblcWalls = collisionsWall: collisionsWall = 0  
    Me.lblcPairs = collisionsPairs: collisionsPairs = 0  
  
    Me.Picture2.Line (myTimer - 1, Val(Me.lblsumWalls))- (myTimer, sum_collisionsWall / myTimer), RGB(255, 0, 0)  
    Me.Picture2.Line (myTimer - 1, Val(Me.lblsumPairs))- (myTimer, sum_collisionsPairs / myTimer), RGB(0, 0, 255)  
  
    Me.lblsumWalls = Format(sum_collisionsWall / myTimer, "#0.00") & " collisions/sec"  
    Me.lblsumPairs = Format(sum_collisionsPairs / myTimer, "#0.00") & " collisions/sec"  
End Sub  
  
Private Sub txtMoveSpeed_Change()  
    moveSpeed = Val(Me.txtMoveSpeed)  
End Sub
```

Όπως είπαμε στην αρχή υπάρχει και ένα αντικείμενο Timer και εδώ φαίνεται και ο κώδικας που περιέχει. Κάθε δευτερόλεπτο κάνει μια στατιστική των κινήσεων και της όλης κατάστασης του αερίου και την γράφει σαν διάγραμμα στο κάτω γράφημα της φόρμας.

Μπορείτε να κάνετε οποιαδήποτε μετατροπή στον κώδικα αυτό ώστε να παρουσιάζει και επιστημονικό ενδιαφέρον. Εφαρμόστε τις εξισώσεις των αερίων σωστά και θα έχετε μια καλή προσομοίωση αυτών που μπορεί να σας φανεί χρήσιμη στο μέλλον.

Παράδειγμα γραφικών: ημιτονοειδείς καμπύλες και συμβολή

Στο παράδειγμα που ακολουθεί θα μελετήσουμε το φαινόμενο της συμβολής δύο ημιτονοειδών κυμάτων. Όπως φαίνεται και στην φόρμα που ακολουθεί, δημιουργούμε δύο κύματα, το κόκκινο και το πράσινο με τις παραμέτρους που απαιτούνται, όπως είναι η συχνότητα του κύματος και η ένταση αυτού. Επίσης, έχουμε και ένα πεδίο όπου μπορούμε να αλλάξουμε την διαφορά φάσης του ενός σε σχέση με το άλλο. Έτσι μπορούμε να δημιουργήσουμε κύματα σε φάση και κύματα με διαφορά φάσης. Τέλος με άσπρη γραμμή είναι η καμπύλη του αθροίσματος των δύο κυμάτων, δηλαδή το συνολικό κύμα. Στο επόμενο σχήμα βλέπετε κύματα με διαφορά φάσης αλλά και διαφορετικές συχνότητες και σχετικές εντάσεις. Στο μεθεπόμενο σχήμα βλέπετε δύο κύματα σε αντίθετη φάση και με ίδια ένταση. Το αποτέλεσμα είναι φυσικά ένα κύμα με μηδενική ένταση, δηλαδή ένα κύμα σε κατάσβεση. Εδώ καταλαβαίνετε πλέον τι σημαίνει η κατάσβεση που μαθαίναμε στην ορυκτολογία και πετρολογία αλλά επίσης και τι σημαίνουν τα χρώματα πόλωσης.



Ο κώδικας για το παραπάνω είναι ιδιαίτερα απλός. Όταν πατάμε το κουμπί Plot

```
Project1 - Form1 (Code)
Command1 Click
Private Sub Command1_Click()
    Call plotWaves
End Sub

Private Sub Command2_Click()
    End
End Sub

Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Label13.Caption = "(" & X & ", " & Y & ")"
End Sub
```

```
Project1 - Module1 (Code)
(plotWaves)
Public Sub plotWaves()
    w1A0 = Val(Form1.Text2)
    w1f = Val(Form1.Text1)
    w2A0 = Val(Form1.Text4)
    w2f = Val(Form1.Text3)
    phaseDiff = Val(Form1.Text5) * 3.1415926

    If Form1.Check1.Value + Form1.Check2.Value = 0 Then Exit Sub

    If Form1.Check1.Value = 1 Then sA = w1A0 + w1A0 * 0.1
    If Form1.Check2.Value = 1 Then sA = w2A0 + w2A0 * 0.1
    If Form1.Check1.Value + Form1.Check2.Value = 2 Then sA = w1A0 + w2A0

    Form1.Picture1.Scale (0, sA)-(10, -sA)

    Form1.Picture1.Cls

    Form1.Picture1.DrawWidth = 1

    If Form1.Check1.Value = 1 Then
        Form1.Picture1.PSet (0, 0)
        For i = 0 To 10 Step 0.05
            a1 = w1A0 * Cos(6.283185 * w1f * i)
            Form1.Picture1.Line -(i, a1), RGB(255, 0, 0)
        Next i
    End If

    If Form1.Check2.Value = 1 Then
        Form1.Picture1.PSet (0, 0)
        For i = 0 To 10 Step 0.05
            a2 = w2A0 * Cos(6.283185 * w2f * i + phaseDiff)
            Form1.Picture1.Line -(i, a2), RGB(0, 255, 0)
        Next i
    End If

    If Form1.Check1.Value + Form1.Check2.Value = 2 Then
        Form1.Picture1.DrawWidth = 2
        Form1.Picture1.PSet (0, 0)
        For i = 0 To 10 Step 0.05
            a1 = w1A0 * Cos(6.283185 * w1f * i)
            a2 = w2A0 * Cos(6.283185 * w2f * i + phaseDiff)
            a = a1 + a2
            Form1.Picture1.Line -(i, a), RGB(255, 255, 255)
        Next i
    End If

End Sub
```

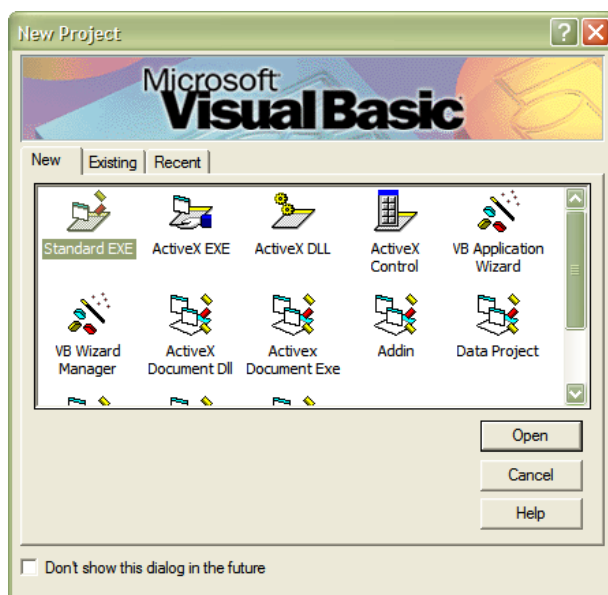
Κεφάλαιο 2^ο

Βάσεις δεδομένων

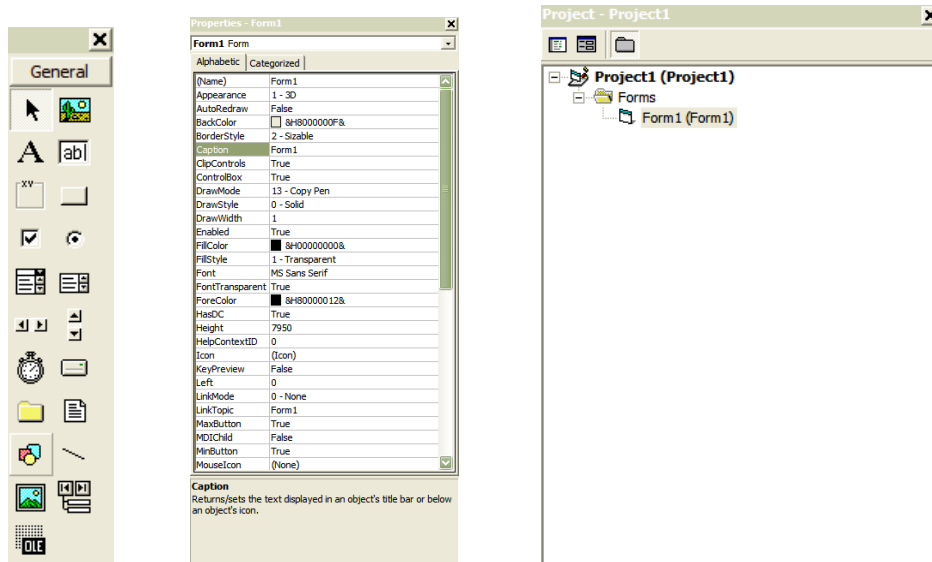
Εισαγωγή





Το κεφάλαιο αυτό περιλαμβάνει οδηγίες για το πώς θα φτιάξετε μια *βάση δεδομένων* χρησιμοποιώντας την *Visual Basic*. Κατά την διάρκεια του μαθήματος, θα χρησιμοποιήσαμε τις εντολές **FindFirst** και **FindNext** που συνδυάζονται με πιο ισχυρές εντολές για το ψάξιμο που έχουν την σύνταξη της γλώσσας *SQL*, αντίθετα με βιβλία που περιορίζονται στις εντολές **Seek** και **Index**.

Ξεκινάμε με την δημιουργία ενός νέου project, και επιλέγουμε το είδος της εφαρμογής να είναι Standard EXE, το οποίο δημιουργεί πλήρης εφαρμογές σε περιβάλλον Windows που βασίζονται σε φόρμες (παράθυρα).




Σας υπενθυμίζουμε τα βασικά μέρη του περιβάλλοντος της Visual Basic. Αυτά είναι η μπάρα με τα εργαλεία όπως φαίνεται αριστερά στο σχήμα που ακολουθεί, η περιοχή με τις ιδιότητες των αντικειμένων (objects) στην μέση του σχήματος και τέλος η περιοχή με τα μέρη και την δομή του προγράμματος (σε απλά προγράμματα με μια φόρμα θα φαίνεται μόνο το όνομα του project και το όνομα της φόρμας).



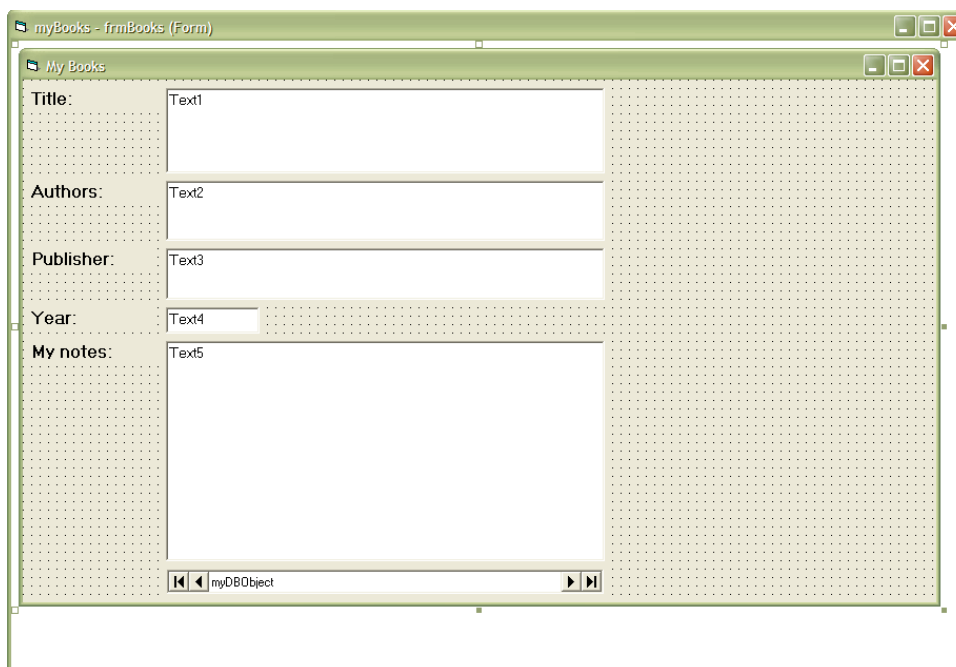
Έπειτα γεμίζουμε την φόρμα μας με τα απαραίτητα εργαλεία που είναι συνήθως ετικέτες (labels, ) , πεδία (text boxes, ) , το εργαλείο σύνδεσης με βάσεις δεδομένων (Data, ) και πιθανότατα κουμπιά ελέγχου (Command buttons, ) . Αυτό γνωρίζετε ήδη πώς να το κάνετε από τα μαθήματα προηγούμενου εξαμήνου.

Ένωση με την βάση δεδομένων: το αντικείμενο Data

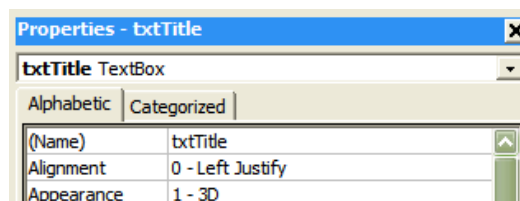
Στα παρακάτω όπου λέμε την λέξη **myDBObject** θα εννοούμε το αντικείμενο *Data* που συμβολίζεται με το εικονίδιο  που το πήρατε από την εργαλειοθήκη αντικειμένων της Visual Basic και το συνδέσαμε με *πίνακα* ενός *συνόλου εγγραφών* (*RecordSet*) μιας βάσης δεδομένων (*Database*). Όταν το εργαλείο αυτό τοποθετηθεί στην φόρμα, ενώ ήδη του έχουμε δώσει το όνομα που είπαμε παραπάνω, θα έχει την παρακάτω μορφή:



Έτσι, η εφαρμογή που κάνουμε κατά την διάρκεια του μαθήματος θα έχει μία φόρμα με τα αντικείμενα όπως φαίνονται στο επόμενο σχήμα.

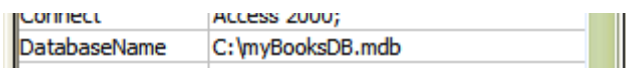


Πρέπει με την Access να έχετε δημιουργήσει μια βάση δεδομένων που περιέχει έναν πίνακα με πεδία ανάλογα των πεδίων που φαίνονται στο προηγούμενο σχήμα. Ονομάζουμε βεβαίως και τα αντικείμενα στην φόρμα μας με λογικά ονόματα, το καθένα ξεχωριστά αλλάζοντας τις ιδιότητες τους. Για παράδειγμα, το πεδίο που φαίνεται να ονομάζεται Text1 του δίνουμε ένα προφανές όνομα, π.χ. txtTitle όπως φαίνεται στο επόμενο σχήμα.

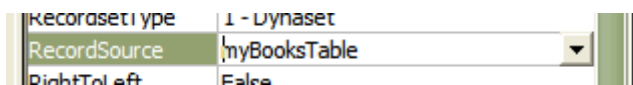


Βεβαίως, πρέπει να κάνουμε και την σύνδεσή του με την βάση δεδομένων (c:/myBooksDB) και τον πίνακά της (myBooksTable), καθώς και να ορίσουμε από ποιο πεδίο θα παίρνει τιμές, το οποίο μπορεί να το έχουμε ήδη ονομάσει μέσα στην Access σαν Title. Για να γίνουν όλα αυτά πρέπει πρώτα να συνδέσουμε το αντικείμενο της

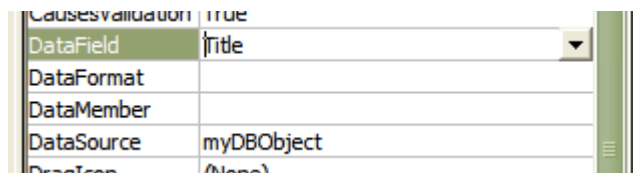
βάσης δεδομένων myDBObject με την βάση δεδομένων αλλάζοντας την παρακάτω ιδιότητα δίνοντάς της το όνομα αρχείου της βάσης που δημιουργήσαμε με την Access.



Επίσης αλλάζουμε και την ιδιότητα όπως παρακάτω ώστε να περιέχει το όνομα του πίνακα της βάσης.

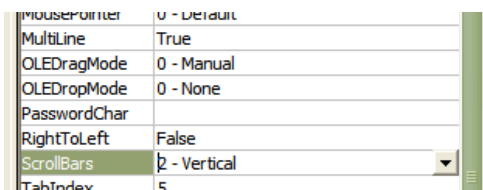


Επιπλέον κάνουμε και την σύνδεση του πεδίου της φόρμας με το αντικείμενο της βάσης δεδομένων αλλάζοντας τις ιδιότητές του στην φόρμα όπως φαίνονται στο παρακάτω σχήμα.



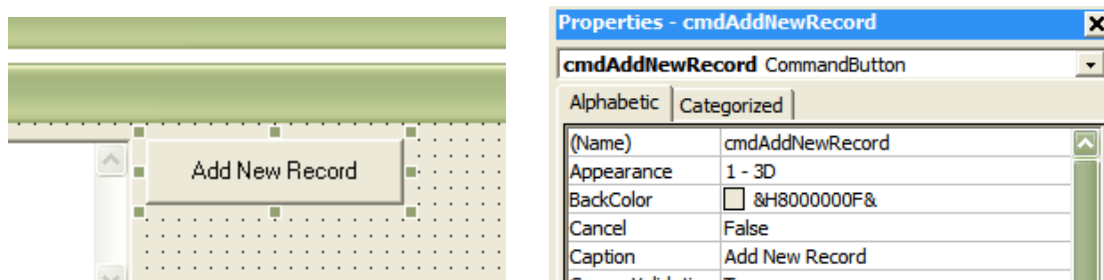
Αλλάζουμε δηλαδή τις ιδιότητες DataSource στο όνομα του αντικειμένου (δηλαδή στο myDBObject) που κάνει την σύνδεση με το αρχείο της βάσης, και την ιδιότητα DataField στο όνομα του αντίστοιχου πεδίου. Την ίδια εργασία κάνουμε για κάθε πεδίο της φόρμας, αντίστοιχα με το πεδίο που του ανήκει. Η Visual Basic διευκολύνει μια και με pull-down menu μας δίνει τις δυνατές επιλογές, αρκεί βέβαια η αρχική σύνδεση με την βάση δεδομένων να έχει γίνει σωστά.

Μπορούμε επιπλέον να αλλάξουμε και άλλες ιδιότητες του αντικειμένου txtTitle ώστε να είναι πιο χρήσιμο, π.χ. να μπορούμε να γράψουμε περισσότερες από μία γραμμές με αυτόματη ροή του κειμένου στην επιφάνειά του.



Για να γίνει αυτό αλλάζουμε την ιδιότητα MultiLine στην τιμή True ενώ μπορούμε να προσθέσουμε και κάθετη μπάρα ολίσθησης αλλάζοντας την ιδιότητα ScrollBars δίνοντάς της την τιμή 2 – Vertical. Με το ίδιο τρόπο μπορούμε να αλλάξουμε πολλές ιδιότητες ώστε να κάνουμε το πρόγραμμα πιο λειτουργικό αλλά και εμφανίσιμο.

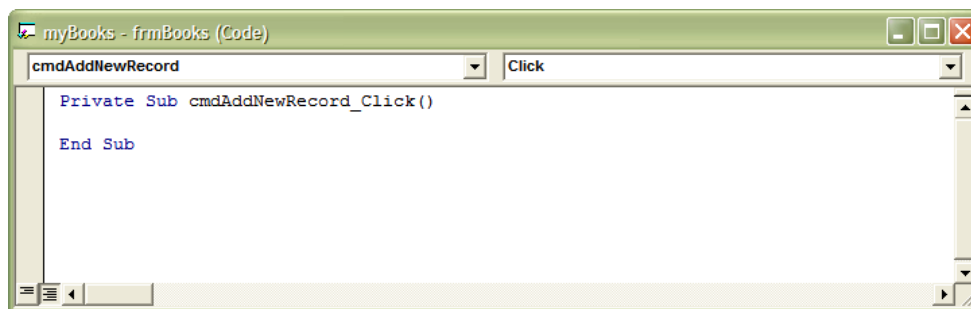
Επίσης απαιτείται και η εισαγωγή στην φόρμα ενός κουμπιού που να μας επιτρέπει να προσθέτουμε καινούργιες εγγραφές. Βάλτε λοιπόν ένα κουμπί CommandButton στην φόρμα και μετά αλλάξτε του την ιδιότητα Name σε ένα όνομα που να είναι προφανές, π.χ. cmdAddNewRecord, καθώς και την ιδιότητα Caption σε ένα σύντομο κείμενο που να εξηγεί την δουλειά του κουμπιού αυτού, π.χ. όπως φαίνεται παρακάτω.



Προσοχή! Στο Name δεν επιτρέπονται χαρακτήρες του κενού, στο Caption όμως μπορείτε να γράψετε ότι θέλετε.

Πρόσθεση νέας εγγραφής

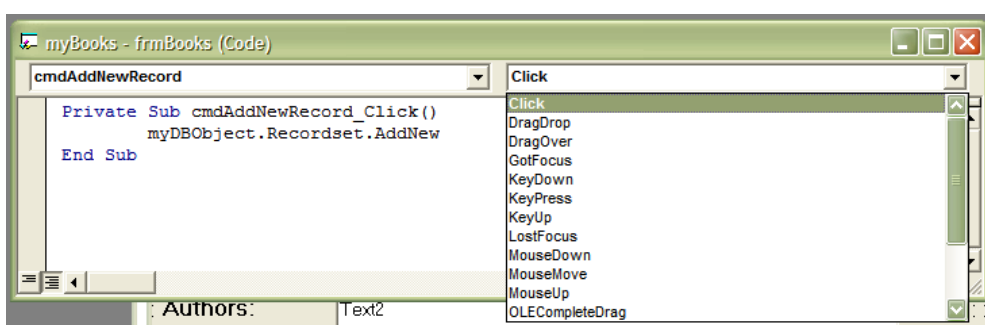
Πρέπει επιπλέον να προγραμματίσουμε το κουμπί αυτό ώστε να δίνει την εντολή για νέα εγγραφή. Πατήστε με το ποντίκι με διπλό κλικ πάνω στο κουμπί. Θα εμφανιστεί ένα παράθυρο όπως το παρακάτω.




Ανάμεσα στις γραμμές “Private Sub ...” και “End Sub” γράφετε τον κώδικα, οτιδήποτε πρέπει δηλαδή να συμβαίνει όταν πατάτε αυτό το κουμπί. Στην περίπτωσή μας αυτό που πρέπει να γίνεται είναι η ενεργοποίηση μια άδειας εγγραφής. Ο κώδικας που το κάνει αυτό είναι ο επόμενος:

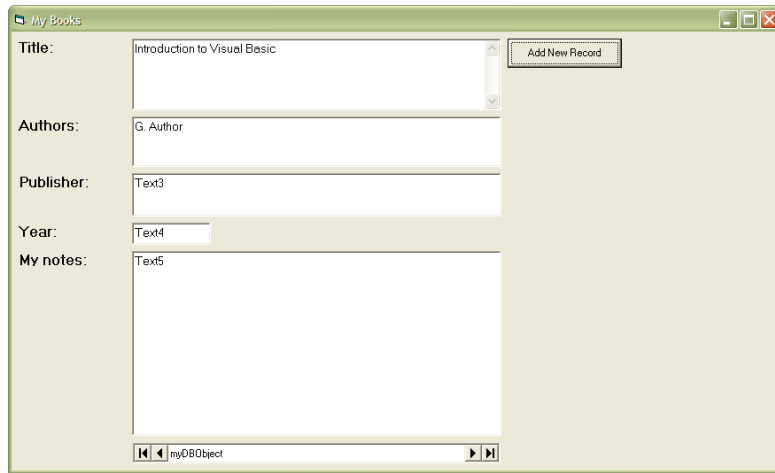
myDBObject.Recordset.AddNew

Το πρώτο μέρος του ονόματος είναι το αντικείμενο σύνδεσης με την βάση δεδομένων, το δεύτερο μέρος του ονόματος μετά την πρώτη τελεία δείχνει στο σύνολο εγγραφών της βάσης και το τρίτο μέρος σημαίνει ότι πρέπει στο σύνολο εγγραφών να προσθέσει μία νέα εγγραφή. Το AddNew είναι μία «μέθοδος» του Recordset, όπως και το Click είναι μέθοδος του κουμπιού cmdAddNewRecord.



Όσα είπαμε φαίνονται στο προηγούμενο σχήμα με επιπλέον μια πληροφορία: το μενού που ανοίγει εάν πατήσουμε το βέλος δεξιά του Click περιέχει όλες τις δυνατές μεθόδους του κουμπιού, δηλαδή περιοχές όπου εσείς μπορείτε να γράψετε κώδικα ώστε το πρόγραμμά σας να κάνει διάφορες λειτουργίες. Θα μπορούσατε π.χ. με το πάτημα του κουμπιού να αλλάζει το χρώμα του και όταν το αφήνετε να επανέρχεται στο αρχικό του χρώμα.





Αφού τελικά γίνουν όλες οι συνδέσεις των πεδίων και ο προγραμματισμός του κουμπιού μπορούμε πλέον να προσθέσουμε τίτλους βιβλίων αφού τρέξουμε βεβαίως το πρόγραμμα πατώντας το κουμπί  που φαίνεται στο μενού του περιβάλλοντος της γλώσσας. Η φόρμα όταν το πρόγραμμα τρέχει φαίνεται στο επόμενο σχήμα. Εδώ έχουμε επιπλέον πατήσει το κουμπί, αυτό άδειασε τα πεδία (έβαλε κενές τιμές) και τα έκανε έτοιμα για να εισάγουμε κείμενο (στην επόμενη φόρμα τον τίτλο και τον συγγραφέα μόνο).



Επαναλαμβάνοντας την λειτουργία μπορούμε να προσθέσουμε επιπλέον εγγραφές, με άλλα λόγια επιπλέον βιβλία. Για να καταλάβουμε τι εννοούμε δίνουμε το επόμενο σχήμα. Οι κολώνες αυτού περιέχουν τα πεδία της βάσης δεδομένων ενώ οι γραμμές (οριζόντια) τις εγγραφές, ή αλλιώς τα βιβλία, δηλαδή ότι εισάγαμε με την Visual Basic. Είναι προφανές ότι εισάγαμε τρία βιβλία (τρεις εγγραφές). Το παρακάτω σχήμα το πήραμε από την Access, ανοίγοντας την βάση δεδομένων και τον πίνακα που περιέχει.

ID	Title	Authors
1	Introduction to Visual Basic	G. Author
2	Advanced Programming with Visual Basic	F. Writer
4	Visual Basic as my first programming language	D. Simple & R. Easy

Μετακίνηση μεταξύ εγγραφών

Πίσω στην Visual Basic ξανά! Μπορούμε να μετακινούμαστε από εγγραφή σε εγγραφή ώστε να τις δούμε όλες πατώντας τα βέλη  και  στο αντικείμενο myDBObject της φόρμας μας. Στην πρώτη εγγραφή πάμε πατώντας το βέλος  και στη τελευταία με το . Όλα βρίσκονται στην φόρμα μας πάνω στο



Αναζητήσεις στην βάση

Χρησιμοποιούμε μια βάση δεδομένων όχι απλά να κρατάμε εγγραφές αλλά και να ψάχνουμε γρήγορα μέσα σε αυτές. Εδώ θα χρησιμοποιήσουμε μια σειρά εντολών που μπορούν να κάνουν πολύπλοκες αναζητήσεις.

Έτσι, χρησιμοποιούμε τις:

myDBObject.RecordSet.FindFirst SQL_String

για να βρούμε την πρώτη εγγραφή (*Record* ή *RecordSet*) που ικανοποιεί την συνθήκη που περιγράφετε στον αλφαριθμητική μεταβλητή SQL_String.

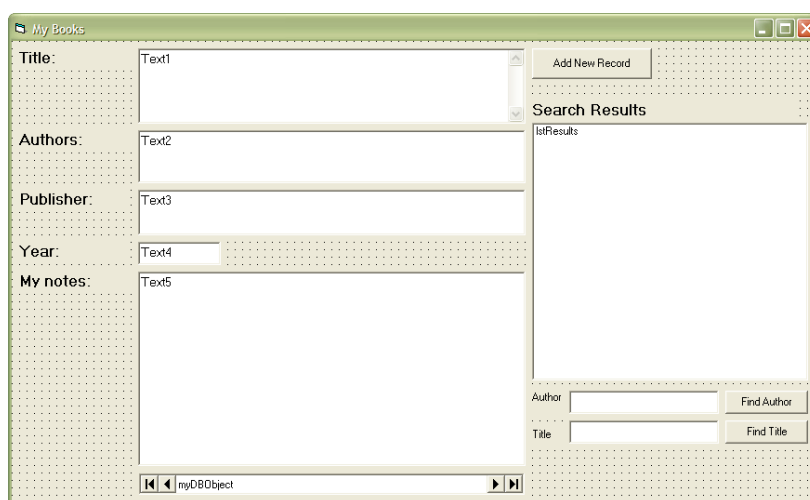
Βάζουμε την εντολή συνήθως εκτός και πριν μιας επανάληψης (*Loop*), όπως για παράδειγμα **While...Wend**.

myDBObject.RecordSet.FindNext SQL_String

για να βρούμε τις επόμενες εγγραφές (*Record* ή *RecordSet*) που ικανοποιούν τις συνθήκες που περιγράφετε στον αλφαριθμητική μεταβλητή *SQL_String* η οποία θα πρέπει να παραμείνει με το ίδιο περιεχόμενο ώστε η αναζήτηση να είναι σωστή.

Βάζουμε την εντολή συνήθως εντός της επανάληψης (*Loop*), όπως για παράδειγμα του ***While...Wend***.

Τα παραπάνω φαίνονται πως λειτουργούν εάν προσθέσουμε στην φόρμα μας μερικά ακόμη αντικείμενα, δηλαδή επιπλέον πεδία *TextBox*, μια λίστα και μερικά κουμπιά. Έτσι η φόρμα μας πλέον γίνεται όπως στο σχήμα:



Πατώντας το κουμπί θα ψάχνει στους συγγραφείς και πατώντας το θα ψάχνει στους τίτλους βιβλίων. Φυσικά θα ψάχνει για το κείμενο λέξη-κλειδί που θα βάζουμε κάθε φορά στα πεδία αριστερά των κουμπιών. Εάν είναι άδεια θα βρίσκει τα πάντα ή τίποτα, ανάλογα την συνθήκη που θα έχουμε στο *SQL_String*.

Λίγα λόγια για τις συνθήκες SQL

Η σύνταξη της *SQL_String* αλφαριθμητικής μεταβλητής είναι συγκεκριμένη και έχει γενικά την παρακάτω σύνταξη:

<Όνομα_πεδίου> <κενός χαρακτήρας> <Συνθήκη> <κενός χαρακτήρας> <Τιμή που αναζητούμε>

Όπου:

<Όνομα_πεδίου>: αντιστοιχεί σε όνομα πεδίου του πίνακα όπως αυτό το έχουμε ορίσει μέσα στην βάση δεδομένων (π.χ. εμείς το κάναμε μέσα από την Access).

<Συνθήκη>: αντιστοιχεί σε τελεστές σύγκρισης, όπως

- = για πλήρη ταύτιση
- > για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να είναι μεγαλύτερη
- < για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να είναι μικρότερη
- <> για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να είναι διαφορετική
- >= για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να είναι μεγαλύτερη ή ίση
- <= για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να είναι μικρότερη ή ίση
- Like** για να εκφράσουμε ότι η τιμή που αναζητούμε πρέπει να μοιάζει ή καλύτερα περιέχετε στη βάση

<Τιμή που αναζητούμε>: αντιστοιχεί στην τιμή που θέλουμε να βρούμε, δηλαδή:

Αριθμητική, οπότε την γράφουμε απευθείας

Αλφαριθμητική, η οποία όμως πρέπει να μπει μέσα σε απλά εισαγωγικά, δηλαδή *'κείμενο'*

Παράδειγμα 1 :

```
SQL_String = "Author = 'John' "  
myDBObject.RecordSet.FindNext SQL_String
```

Μέσα σε διπλά εισαγωγικά γράφουμε το **Author = 'John'**, όπου το **Author** είναι το *όνομα πεδίου* των εγγραφών του πίνακα που περιέχει τα ονόματα των συγγραφέων (Author) π.χ. των βιβλίων μας. Το = σημαίνει ότι θέλουμε πλήρη ταύτιση του αναζητούμενου ονόματος (ακριβώς ίσο δηλαδή) και το **'John'** μέσα σε απλά/μονά εισαγωγικά είναι το τι ζητάμε, δηλαδή εγγραφές που να περιέχουν συγγραφείς με το όνομα **John**.

Παράδειγμα 2 :

```
SQL_String = "Year >= 2002 "  
myDBObject.RecordSet.FindNext SQL_String
```

Μέσα σε διπλά εισαγωγικά γράφουμε το **Year = 2005**, όπου το **Year** είναι το *όνομα πεδίου* των εγγραφών του πίνακα που περιέχει τα έτη εκδόσεων των βιβλίων (Year). Το **>=** σημαίνει ότι θέλουμε να βρούμε τα βιβλία που έχουν ημερομηνία έκδοσης από το 2002 και μετά. Ο αριθμός δεν χρειάζεται να είναι μέσα σε απλά εισαγωγικά.

Παράδειγμα 3 :

```
SQL_String = "Title Like '*Visual Basic*' "  
myDBObject.RecordSet.FindNext SQL_String
```

Μέσα σε διπλά εισαγωγικά γράφουμε το **Title Like '*Visual Basic*'**, όπου το **Title** είναι το *όνομα πεδίου* των εγγραφών του πίνακα που περιέχει τους τίτλους των βιβλίων (Title). Το **Like** σημαίνει ότι θέλουμε να βρούμε τα βιβλία που έχουν κάπου μέσα στον τίτλο τους το όνομα Visual Basic, δηλαδή βιβλία της Visual Basic. Αυτό πρέπει να είναι μέσα σε απλά εισαγωγικά, δηλαδή **'*Visual Basic*'**. Τα αστεράκια σημαίνουν ότι το Visual Basic μπορεί να περιέχεται οπουδήποτε μέσα στον τίτλο. Αν γράφαμε **'Visual Basic'** θα βρίσκαμε τις εγγραφές με το Visual Basic στο τέλος του τίτλου, ενώ αν γράφαμε **'Visual Basic*'** θα βρίσκαμε τις εγγραφές με το Visual Basic πάντα στην αρχή του τίτλου.

Τελεστές

Επίσης μπορούμε να συνδέουμε τις παραπάνω συνθήκες με τελεστές συνθήκης όπως οι AND, OR, XOR κτλ.

Παράδειγμα:

```
SQL_String = "Title Like '*Visual Basic*' AND Year >= 2002"  
myDBObject.RecordSet.FindNext SQL_String
```

Με το παραπάνω βρίσκουμε όλες τις εγγραφές βιβλίων που έχουν οπουδήποτε στον τίτλο το Visual Basic και έχουν ταυτόχρονα εκδοθεί κατά και μετά το έτος 2002.

Ένα σύνθετο παράδειγμα

Κατά την διάρκεια του μαθήματος κάνουμε ένα παράδειγμα που σας βοηθά να φτιάχνετε τέτοιες πολύπλοκες συνθήκες, επιτόπου με επιλογές πάνω στην φόρμα. Αυτό συνήθως απαιτεί την χρήση δομών αποφάσεων όπως η **IF...THEN...ELSE...ENDIF** και με άλλα αντικείμενα όπως τα **OptionButton** και **CheckBox**.

Προχωρούμε να δούμε αρχικά μια απλή περίπτωση που βρίσκει μία μόνο εγγραφή. Το αποτέλεσμα της αναζήτησης θα φαίνεται στην λίστα της φόρμας, την οποία έχουμε ονομάσει **lstResults**. Με τον κώδικα που φαίνεται στο επόμενο σχήμα βρίσκουμε μόνο την πρώτη εγγραφή της οποίας τον τίτλο βάζει στην λίστα όπως φαίνεται και στο μεθεπόμενο σχήμα.

```
myBooks - frmBooks (Code)
cmdFindAuthor Click
Private Sub cmdFindAuthor_Click()
    SQL_String = "Authors = '" + frmBooks.txtFindAuthor + "'"
    frmBooks.lstResults.Clear
    frmBooks.myDBObject.Recordset.FindFirst SQL_String
    frmBooks.lstResults.AddItem frmBooks.myDBObject.Recordset!Title
End Sub
```

Με την γραμμή εντολής `SQL_String = "Authors = '" + frmBooks.txtFindAuthor + "'"` σχηματίζουμε την εντολή αναζήτησης της SQL. Εάν στο πεδίο αναζήτησης του συγγραφέα βάλουμε το όνομα "F. Writer", τότε μετά το πάτημα του κουμπιού για την αναζήτηση η αλφαριθμητική μεταβλητή `SQL_String` θα περιέχει την τιμή:

`SQL_String = "Authors = ' F. Writer'"`

Προσοχή στα εισαγωγικά. Τα διπλά εισαγωγικά (") ανήκουν στην Visual Basic ενώ τα απλά/μονά εισαγωγικά (') στην SQL, όπως είπαμε και προηγουμένως.

Έπειτα το πρόγραμμα καθαρίζει την λίστα αποτελεσμάτων από οτιδήποτε μπορεί ήδη να περιέχει, έπειτα εκτελείτε ή εντολή αναζήτησης και το πρώτο βιβλίο (εγγραφή) που βρίσκει με μοναδικό συγγραφέα τον **F. Writer** το προσθέτει στην λίστα στην τελευταία εντολή σαν τίτλο. Ωστόσο το πρόγραμμα δεν είναι πλήρες μια και εάν ο συγγραφέας δεν υπάρχει θα σας βάλει στην λίστα την πρώτη εγγραφή αλλά αυτό δεν θα έπρεπε να συμβαίνει. Για να αποφύγουμε αυτό μετατρέπουμε τον κώδικα όπως στο επόμενο σχήμα.

```
myBooks - frmBooks (Code)
cmdFindAuthor Click
Private Sub cmdFindAuthor_Click()
    SQL_String = "Authors = '" + frmBooks.txtFindAuthor + "'"
    frmBooks.lstResults.Clear
    frmBooks.myDBObject.Recordset.FindFirst SQL_String
    If frmBooks.myDBObject.Recordset.NoMatch = False Then
        frmBooks.lstResults.AddItem frmBooks.myDBObject.Recordset!Title
    Else
        frmBooks.lstResults.AddItem "No book found!"
    End If
End Sub
```

Με μία δομή αποφάσεων `IF...THEN...ELSE...ENDIF` και με την ιδιότητα `NoMatch` η οποία παίρνει την τιμή `FALSE` όταν βρει κάποια εγγραφή ή `TRUE` όταν δεν βρει βγάζει και το αντίστοιχο αποτέλεσμα. Δηλαδή για συγγραφέα που δεν υπάρχει η δομή απόφασης θα γράψει στην λίστα το κείμενο "No book found!".

Όταν ο **F. Writer** όμως έχει γράψει περισσότερα του ενός βιβλία και τα έχουμε στην βάση μας τότε πως θα τα δούμε; Αυτό είναι δυνατόν μετατρέποντας τον κώδικα σύμφωνα με το επόμενο σχήμα.

```
Private Sub cmdFindAuthor_Click()  
    SQL_String = "Authors = '" + frmBooks.txtFindAuthor + "'" +  
    frmBooks.lstResults.Clear  
    frmBooks.myDBObject.Recordset.FindFirst SQL_String  
  
    While frmBooks.myDBObject.Recordset.NoMatch = False  
        frmBooks.lstResults.AddItem frmBooks.myDBObject.Recordset!Title  
        frmBooks.myDBObject.Recordset.FindNext SQL_String  
    Wend  
  
End Sub
```

Εδώ έχουμε προσθέσει την δομή επανάληψης **While...Wend** και την μέθοδο **FindNext** που κάνουν το πρόγραμμα να σαρώνει όλη την βάση και να βρίσκει όλες τις περιπτώσεις που ικανοποιούν την συνθήκη SQL. Σε αυτή την περίπτωση εάν δεν βρεθεί κάτι η λίστα παραμένει άδεια οπότε δεν απαιτείται κάτι επιπλέον εκτός εάν οπωσδήποτε θέλουμε να φαίνεται ότι δεν βρέθηκε τίποτα οπότε προσθέτουμε αμέσως μετά το **Wend** σε νέα γραμμή τον επιπλέον κώδικα που φαίνεται στο επόμενο σχήμα.

```
Wend  
If frmBooks.lstResults.ListCount = 0 Then frmBooks.lstResults.AddItem "No book found!"  
End Sub
```

Εδώ το **ListCount** μετρά πόσες γραμμές εγγραφών έχουν γίνει στην λίστα. Εάν δεν έχει γίνει καμία σημαίνει ότι δεν βρέθηκε κανένα βιβλίο οπότε μπορεί απλά να ενημερώσει ανάλογα.

Για να μην χρειάζεται να βάζουμε ολόκληρο το όνομα ενός συγγραφέα, όπως ακριβώς αυτό είναι γραμμένο στην βάση δεδομένων, αλλά μέρος αυτού και μόνο τότε χρησιμοποιούμε τον τελεστή σύγκρισης της SQL που λέγεται **Like**. Αυτός σε συνδυασμό με αστεράκια (*) βρίσκει όλες τις εγγραφές που περιέχουν στο πεδίο που γίνεται η αναζήτηση το κείμενο που αναζητούμε. Το πώς αλλάζει η γραμμή εντολής όπου καθορίζουμε την μεταβλητή **SQL_String** φαίνεται στο επόμενο σχήμα. Όλα τα άλλα παραμένουν ίδια.

```
SQL_String = "Authors Like '%" + frmBooks.txtFindAuthor + "%'"
```

Εδώ καλύψαμε τα βασικότερα στοιχεία που μας επιτρέπουν να γράψουμε προγράμματα βάσεων δεδομένων στην Visual Basic. Ο συνδυασμός των παραπάνω επιτρέπει την δημιουργία πολύπλοκων προγραμμάτων με απόλυτο έλεγχο πάνω στην βάση δεδομένων. Σαν άσκηση χρησιμοποιήστε όσα ξέρετε από την γλώσσα και όσα μάθαμε εδώ για να κάνετε μια πληρέστερη εφαρμογή βιβλιοθήκης με πολύπλοκες αναζητήσεις.

Σημειώσεις:

- Είναι σημαντική βοήθεια να βάζετε πρώτα την SQL μέσα σε μια αλφαριθμητική μεταβλητή και μετά να εφαρμόζεται τις μεθόδους **FindFirst** και **FindNext** σε αυτή την μεταβλητή.
- Για μετακινήσεις στην βάση δεδομένων είδαμε ότι μπορούμε να χρησιμοποιούμε τις εντολές:

MoveFirst για μετακίνηση στην πρώτη εγγραφή

MoveLast για μετακίνηση στην τελευταία εγγραφή

MoveNext για μετακίνηση στην επόμενη εγγραφή από αυτή που είμαστε

MovePrevious για μετακίνηση στην προηγούμενη εγγραφή από αυτή που είμαστε

- Να θυμόμαστε την μέθοδο **NoMatch** γιατί σας είναι πολύ χρήσιμη για να τερματίζετε τις δομές επανάληψης και αποφάσεων, όπως την **While...Wend**, αλλά και να δίνεται τα κατάλληλα μηνύματα στον χρήστη.
- Όταν προσθέτετε μια νέα εγγραφή με την μέθοδο **AddNew**, η νέα εγγραφή σώζεται μόνο όταν μετακινηθείτε σε άλλη υπάρχουσα, προηγούμενη εγγραφή. Εάν, για παράδειγμα, για κάποιο λόγο ο

υπολογιστής σβήσει, ή κλείσετε το πρόγραμμα πριν μετακινηθείτε σε νέα εγγραφή, θα χάσετε την τελευταία εγγραφή.

- Εάν θέλετε να σβήσετε μια εγγραφή, δημιουργήστε ένα κουμπί διαγραφής και βάλτε για κώδικα σε αυτό (μέθοδο **Click**) την μέθοδο **Delete**, π.χ. **frmBooks.myDBObject.Recordset.Delete**
- Σημαντικό είναι να χρησιμοποιήσετε το HELP της Visual Basic για επιπλέον εντολές και για περισσότερες πληροφορίες στην σύνταξη των εντολών.

ΠΡΟΣΟΧΗ!!!

Είναι σημαντικό να δοκιμάσετε τα παραπάνω στον υπολογιστή.
Είναι ο μόνος τρόπος να τα καταλάβετε και να τα θυμόσαστε.

Κεφάλαιο 3^ο

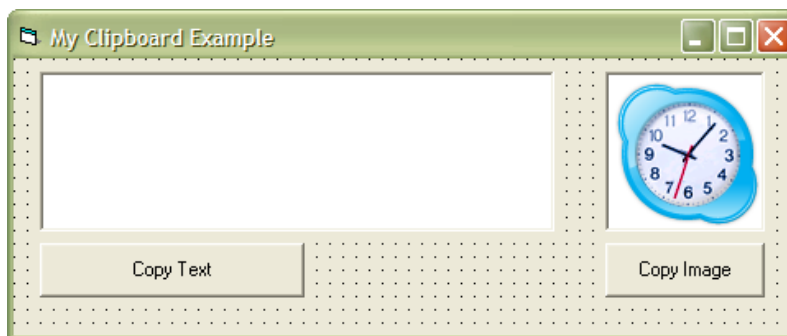
Χρήση του πρόχειρου για μεταφορά δεδομένων σε διαφορετικές εφαρμογές

Εισαγωγή

Γνωρίζεται πολύ καλά όλοι σας ότι είναι πολύ εύκολο να μεταφέρουμε μεταξύ προγραμμάτων Windows οποιαδήποτε πληροφορία σε μορφή κειμένου ή εικόνας. Πολλαπλές φορές χρησιμοποιείτε το *Copy/Cut & Paste* ή το *Αντιγραφή/Αποκοπή & Επικόλληση* όπως είναι στα Ελληνικά. Εδώ θα μάθετε πόσο απλά γίνεται αυτό ώστε να το χρησιμοποιείτε και εσείς. Το αντικείμενο που θα χρησιμοποιήσουμε το καταλαβαίνει η γλώσσα με το όνομα Clipboard.

Αντικείμενο Clipboard

Δημιουργήστε μια φόρμα με τα παρακάτω αντικείμενα, ένα κειμένου και ένα εικόνας και περάστε κάποια τυχαία εικόνα σε αυτό.

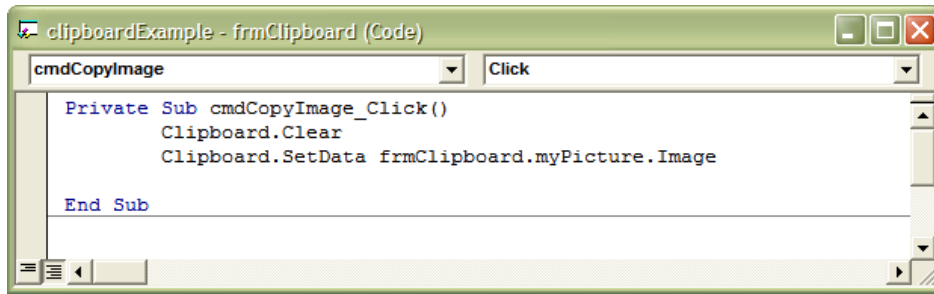


Στο κουμπί "Copy Text" βάλτε τον κώδικα που ακολουθεί.

```
clipboardExample - frmClipboard (Code)
cmdCopyText Click
Private Sub cmdCopyText_Click()
    Clipboard.Clear
    Clipboard.SetText frmClipboard.txtText.Text
End Sub
```

Η μέθοδος **Clear** καθαρίζει το περιεχόμενο του προχείρου ώστε να μπορεί να δεχτεί νέα δεδομένα και είναι απαραίτητη κάθε φορά. Η μέθοδος **SetText** τοποθετεί στο πρόχειρο κείμενο και μόνο, π.χ. το περιεχόμενο του txtText που είναι αντικείμενο της φόρμας που ονομάσαμε frmClipboard.

Αντίστοιχα, για την περίπτωση εικόνας χρησιμοποιούμε την εντολή **SetData** όπως φαίνεται στο επόμενο σχήμα.



Όταν από άλλη εφαρμογή θέλουμε να μεταφέρουμε κείμενο τότε μπορούμε να χρησιμοποιήσουμε την μέθοδο **GetText** ή την **GetData** όταν πρόκειται για εικόνες.

Παράδειγμα

Όταν θέλουμε στο αντικείμενο της φόρμας μας με όνομα `txtText` να βάλουμε κείμενο που περιέχεται στο πρόχειρο τότε γράφουμε τον κώδικα: **`frmClipboard.txtText.Text = Clipboard.GetText`**.

Κεφάλαιο 4^ο

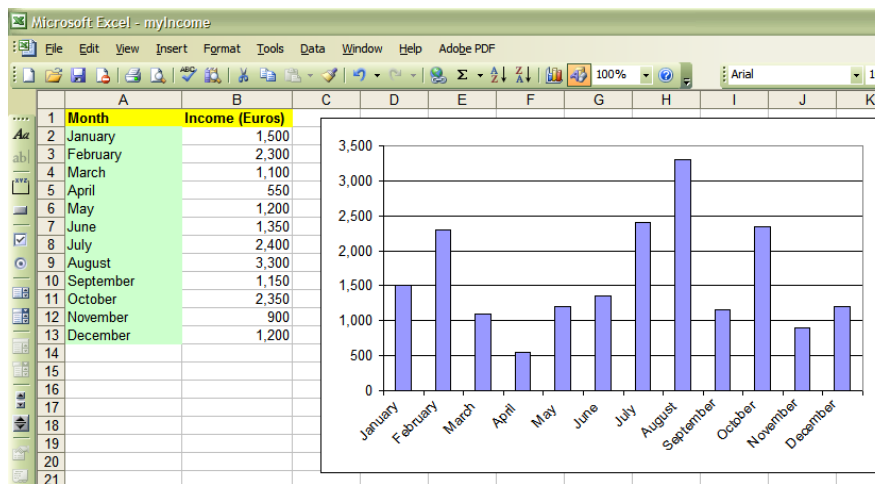
EXCEL σε σύνδεση με πρόγραμμα της Visual Basic


Εισαγωγή

Τα προγράμματα που γράφουμε σε Visual Basic μπορούν να συνδεθούν με οποιοδήποτε πρόγραμμα Windows και να πάρει δεδομένα ή να μεταφέρει δεδομένα. Μπορεί επίσης να αυτοματοποιήσει πλήρως μια άλλη εφαρμογή, π.χ. ένα λογιστικό φύλλο όπως το EXCEL, και να χρησιμοποιήσει τις δυνατότητές του. Εδώ θα δούμε πως γίνεται αυτό.

Εκτέλεση

Δημιουργήστε στο EXCEL το παρακάτω φύλλο εργασίας.

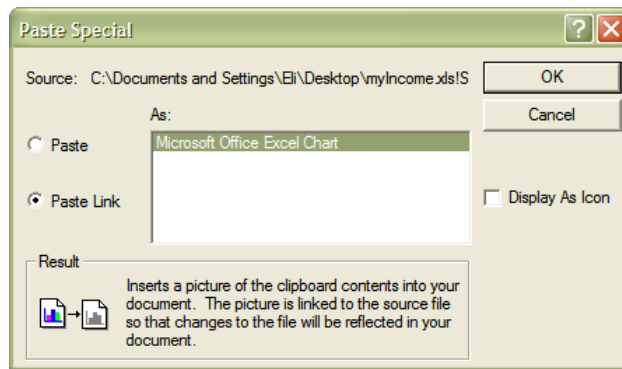


Δημιουργήστε και μία φόρμα στην Visual Basic με δύο αντικείμενα OLE με την χρήση του εργαλείου . Κατά την δημιουργία θα σας ζητηθεί ένα αντικείμενο από μία λίστα το οποίο θα χαρακτηρίζει το OLE. Η λίστα αυτή περιέχει αντικείμενα που έχουν εγκατασταθεί από διάφορες εφαρμογές στον υπολογιστή σας και ο αριθμός και το είδος τους θα είναι διαφορετικός για διαφορετικούς υπολογιστές. Εσείς πατήστε Cancel ώστε να δημιουργήσετε ένα άδικο αντικείμενο OLE. Δημιουργήστε τέσσερα άδεια αντικείμενα.

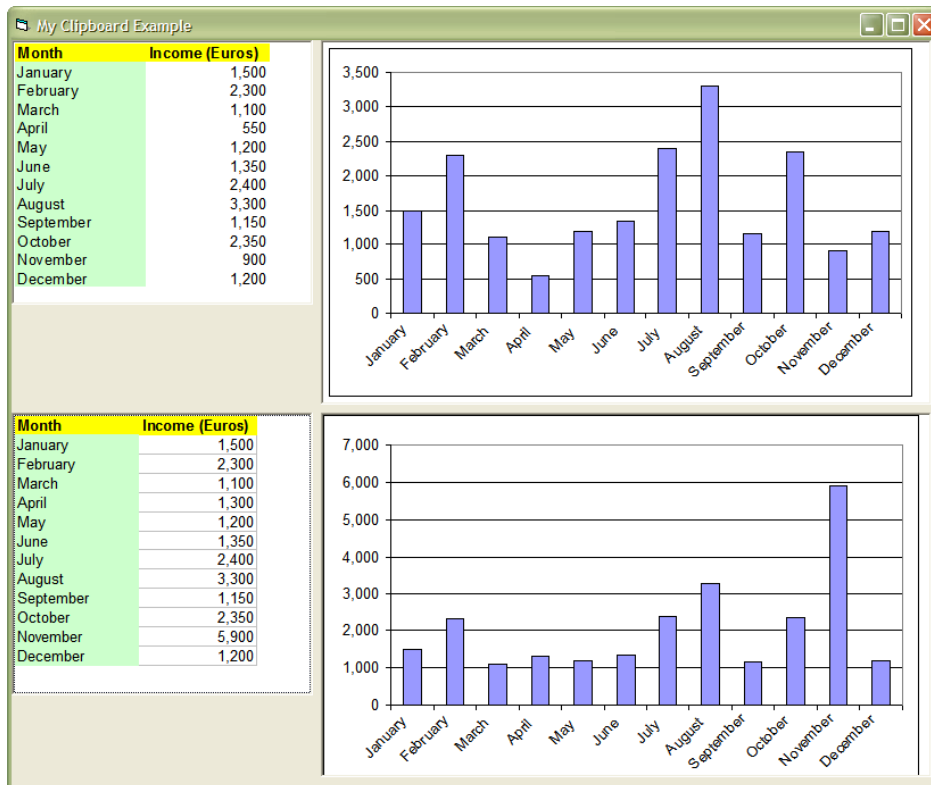
Πηγαίνετε στο λογιστικό φύλλο και επιλέξτε την περιοχή με τις τιμές. Έπειτα πηγαίνετε στο επάνω αριστερά αντικείμενο OLE, επιλέξτε το και με το δεξί πλήκτρο του ποντικιού ανοίξτε το μενού όπου επιλέγετε Paste (ή Επικόλληση). Κάντε το ίδιο επιλέγοντας μόνο το γράφημα του λογιστικού φύλλου και βάλτε το στο δεξί επάνω αντικείμενο OLE.

Τρέξτε την εφαρμογή σας και δοκιμάστε να αλλάξετε κάποια τιμή στο λογιστικό φύλλο. Να βλέπετε παράλληλα και τα δύο παράθυρα εάν είναι δυνατόν. Παρατηρήσατε κάποια αλλαγή στις τιμές ή στο γράφημα του προγράμματός σας της Visual Basic; Η απάντηση θα πρέπει να είναι όχι και αυτά τα δύο OLE κρατάνε μόνο τις τιμές που κάνατε την επικόλληση ή Paste.

Εάν επαναλάβετε το ίδιο αλλά χρησιμοποιήσετε την «Ειδική Επικόλληση» ή “Paste Special” τότε θα δείτε ένα παράθυρο όπως αυτό του σχήματος που ακολουθεί.

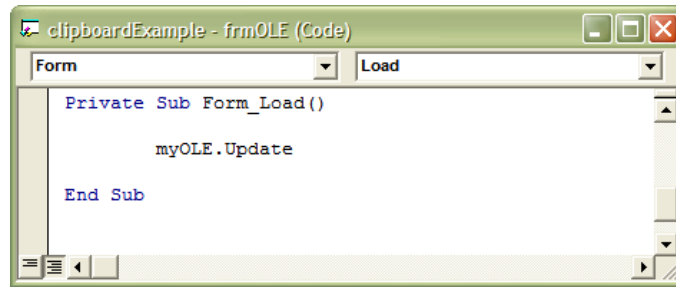


Όταν δείτε αυτό το παράθυρο σιγουρέψατε ότι επιλέξατε το “Paste Link”, μετά πατήστε το κουμπί “OK”. Κάντε το ίδιο και για το γράφημα. Τρέξτε το πρόγραμμά σας και θα δείτε το παρακάτω παράθυρο.



Αλλάξτε τώρα μία τιμή στο λογιστικό σας φύλλο, θα δείτε ότι και το γράφημα αλλάζει. Παρατηρήστε τις αλλαγές στο πρόγραμμά σας της Visual Basic που τρέχει. Θα παρατηρήσετε ότι τα επάνω αντικείμενα OLE διατηρούν τις αρχικές τιμές που τους δώσατε ενώ τα κάτω έχουν πάρει τις νέες τιμές. Αυτή η διαφορά οφείλεται στο ότι τα κάτω αντικείμενα OLE είναι συνδεδεμένα με την επιλογή “Paste Link” που πάντα θα αναλαμβάνει και την ενημέρωση αυτών.

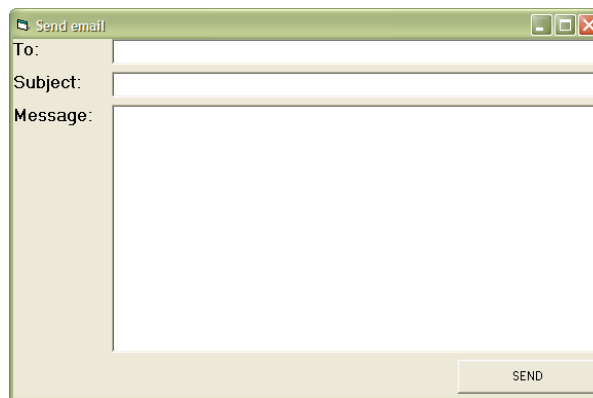
Όταν το αρχείο του EXCEL είναι ανοικτό ταυτόχρονα με την εφαρμογή σας τότε η νέες τιμές ανανεώνονται αμέσως στην εφαρμογή σας. Εάν η εφαρμογή σας δεν τρέχει και κάποια στιγμή έγινε αλλαγή των τιμών αυτό δεν φαίνεται στην εφαρμογή σας. Ακόμη και εάν τρέξετε την εφαρμογή σας (χωρίς και πάλι το EXCEL να είναι ανοικτό) οι νέες τιμές δεν θα εμφανιστούν. Οπότε, εάν κάθε φορά που τρέχετε την εφαρμογή σας θέλετε να έχει τις τελευταίες τιμές τότε θα πρέπει να εφαρμόσετε την μέθοδο Update. Εάν το αντικείμενο OLE έχει π.χ. το όνομα myOLE τότε η εντολή που πρέπει να γράψετε φαίνεται στο επόμενο σχήμα.



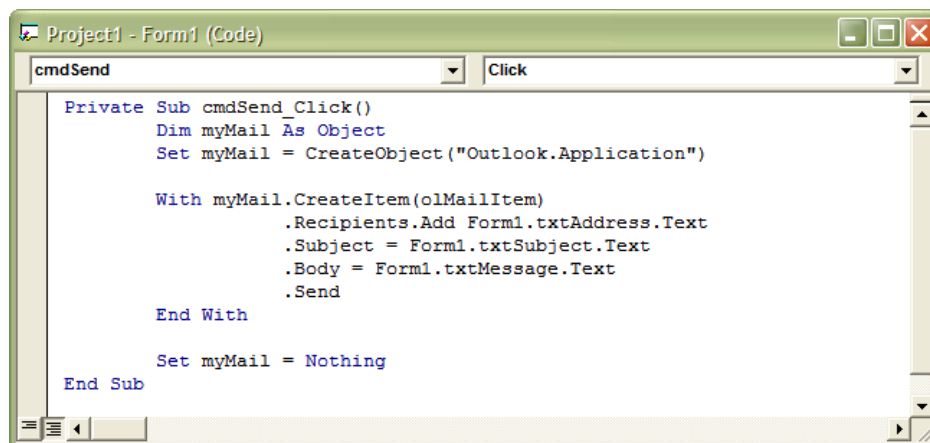
Ελέγχοντας εφαρμογές από την Visual Basic

Μια επιπλέον λειτουργία είναι ο αυτοματισμός εφαρμογών μέσα από την γλώσσα Visual Basic. Έχετε την δυνατότητα να ανοίγετε εφαρμογές, να τις κλείνετε, να δημιουργείτε νέα φύλλα εργασίας, να παίρνετε τιμές και να αλλάζετε τιμές, ακόμη και να κάνετε υπολογισμούς με λειτουργίες του EXCEL.

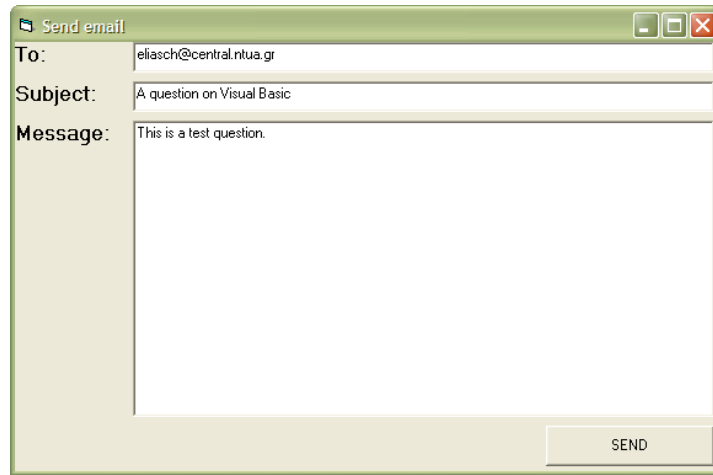
Το παράδειγμα που θα δώσουμε εδώ αφορά στον έλεγχο του Outlook από την Visual Basic. Αυτό που θα γίνει είναι να δημιουργήσετε μια φόρμα που να περιέχει όλα τα πεδία που απαιτούνται για την αποστολή ενός email όπως στο επόμενο σχήμα.



Στο κουμπί αποστολής (SEND) θα βάλετε τον παρακάτω κώδικα.



Οπότε μπορείτε να γεμίσετε τα πεδία με λογικές τιμές και να στείλετε email σε οποιονδήποτε μέσω της εφαρμογής σας. Μια γεμάτη φόρμα μπορεί να είναι η ακόλουθη:



Οπότε, πατώντας τώρα το κουμπί SEND θα στείλει σε εμένα το μήνυμα με τις πληροφορίες που αναγράφονται σαν Subject και Message.

Επίλογος

Καταλαβαίνετε πλέον ότι οι δυνατότητες είναι απεριόριστες όταν γνωρίζετε μια τουλάχιστον γλώσσα προγραμματισμού. Με τα παραπάνω αλλά και με την βοήθεια των αρχείων HELP του περιβάλλοντος της γλώσσας μπορείτε να μάθετε περισσότερα και να εξελιχθείτε στον προγραμματισμό. Χρησιμοποιήστε το κουμπί "F1" αφού επιλέξετε ένα αντικείμενο πάνω στην φόρμα ή μία εντολή όταν τοποθετήσετε τον κέρσορα πάνω της την ώρα που γράφετε το πρόγραμμα και θα ανοίξει ένα παράθυρο με σχετική βοήθεια. Διαβάζετε με λεπτομέρεια την βοήθεια, είναι ο μόνος τρόπος να καταλάβετε την γλώσσα στις λεπτομέρειές της. Η γνώση αυτή θα μείνει μόνιμα σε εσάς εάν δοκιμάζετε τις ιδέες σας με προγράμματα δικά σας.

Οι σημειώσεις αυτές καλύπτουν μεγάλο μέρος του μαθήματος, ωστόσο δεν αναπτύσσουν το θέμα στην λεπτομέρειά του. Κάτι τέτοιο θα ήταν αδύνατο για οποιοδήποτε βιβλίο πόσο μάλλον για αυτό το βοήθημα. Πιστεύω όμως ότι σας δόθηκε ο τρόπος χειρισμού του περιβάλλοντος της Visual Basic και ο τρόπος σκέψης ώστε να επεκταθείτε ευκολότερα χρησιμοποιώντας και τις γνώσεις για την γλώσσα που ήδη έχετε από προηγούμενο εξάμηνο.