ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΠΑΤΡΩΝ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΥ ΣΧΕΔΙΑΣΜΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΠΤΥΞΗ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΥΛΙΚΟΥ ΓΙΑ ΤΗ VISUAL BASIC

ΤΩΝ ΦΟΙΤΗΤΩΝ:

ΡΩΜΑΝΟΥ ΣΟΦΙΑ
ΚΟΜΗΝΟΥ ΣΟΦΙΑ
ΟΙΚΟΝΟΜΙΔΗΣ ΓΕΩΡΓΙΟΣ

ΕΠΟΠΤΕΥΩΝ ΚΑΘΗΓΗΤΗΣ: ΜΠΑΚΑΛΗΣ ΑΡΗΣ

ΠΑΤΡΑ 2012

ΠΡΟΛΟΓΟΣ

Η πτυχιακή εργασία που κρατάτε στα χέρια σας, με τίτλο «Ανάπτυξη εκπαιδευτικού υλικού για τη Visual Basic» έχει αναπτυχθεί στα πλαίσια του προγράμματος σπουδών του Τμήματος Επιχειρηματικού Σχεδιασμού και Πληροφοριακών Συστημάτων, της Σχολής Διοίκησης και Οικονομίας του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Πατρών. Αποτελείται από το έντυπό αλλά και από το συνοδευτικό CD ROM στο οποίο εμπεριέχεται το εκπαιδευτικό υλικό σε ηλεκτρονική μορφή και συνιστάται η χρήση του.

Αναπτύχθηκε, σχεδιάστηκε, συντάχθηκε, βελτιώθηκε και υλοποιήθηκε από τους: Κομηνού Σοφία, Ρωμανού Σοφία και Οικονομίδη Γεώργιο, φοιτητές του Τμήματος Επιχειρηματικού Σχεδιασμού και Πληροφοριακών Συστημάτων, υπό τη συνεχή υποστήριξη, του καθηγητή Δρ. Άρη Μπακάλη.

Στόχο έχει να προσφέρει επίκαιρη, ευκατανόητη και έγκυρη γνώση στον εκπαιδευόμενο, ενώ συνάμα να προσφέρει τεκμηρίωση της ύλης της, της διάρθρωσής της και των προδιαγραφών της σε πιθανό εκπαιδευτή που επιθυμεί να τη χρησιμοποιήσει ως βοήθημα, με στόχο την όσο το δυνατόν μεγαλύτερη διευκόλυνση της όλης εκπαιδευτικής διαδικασίας.

ΠΕΡΙΛΗΨΗ

Η εργασία εμπεριέχει επεξήγηση του πώς προσεγγίστηκε το έργο, επεξήγηση αποφάσεων που λήφθηκαν καθώς και την αναφορά κάθε επιστημονικής γνώσης ή μεθόδου που χρησιμοποιήθηκε. Αναφέρονται οι προδιαγραφές, ενώ στη συνέχεια, υπάρχει για κάθε ένα κεφάλαιο του εκπαιδευτικού υλικού κι ένα κεφάλαιο στο έντυπο. Μέσα στα κεφάλαια του εντύπου εκτός από το εκπαιδευτικό υλικό, υπάρχει επιπλέον στο τέλος, μια συνοπτική τεκμηρίωση της προσέγγισής του κεφαλαίου. Το εκπαιδευτικό υλικό αποτελείται από θεωρητική κατάρτιση βασικών γνώσεων, πρακτική υλοποίηση αυτών των γνώσεων βήμα βήμα με τη βοήθεια εικόνων, ενώ σε περιπτώσεις που θεωρήθηκε αναγκαία κάποια επιπλέον εκπόνηση, συμπεριλήφθηκαν ασκήσεις με τις λύσεις τους. Η μορφή του είναι σε HTML, είναι έτοιμο για χρήση τοπικά σε έναν ηλεκτρονικό υπολογιστή ή και στο διαδίκτυο.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Κάθε Πτυχιακή Εργασία, κατά τον Δ.Γ. Τσαούση, αποτελεί ως επιστημονική εργασία, ατομική συμβολή στην πρόοδο της επιστήμης. Αυτό σημαίνει πως πρέπει να χαρακτηρίζεται από πρωτοτυπία. Στο βιβλίο του Τσαούση «Λεξικό της κοινωνιολογίας της εκπαίδευσης» συναντάμε τον παρακάτω ορισμό: «Εκπαίδευση είναι μορφή εξειδικευμένης κοινωνικοποίησης που ασκείται κατά κανόνα από φορείς που βρίσκονται έξω από την οικογένεια και έχει ως περιεχόμενό της την μετάδοση γνώσεων, δεξιοτήτων και μορφών ενέργειας ή συμπεριφοράς (εντός ειδικών ιδρυμάτων).

Οι σκοποί της εκπαίδευσης διαφέρουν από χώρα σε χώρα και από εποχή σε εποχή. Η σύγχρονη εκπαίδευση δεν έχει ξεκαθαρισμένους σκοπούς λόγω της γρήγορης ανάπτυξης, παρόλα αυτά γίνονται γενικώς αποδεκτοί οι παρακάτω σκοποί:

- Ο σχηματισμός του χαρακτήρα, που θα βοηθήσει τους νέους στις σχέσεις τους με τους συνανθρώπους τους.
- Η ανάπτυξη της ευφυΐας.
- Η μετάδοση και ίσως η βελτίωση της εθνικής κουλτούρας.
- Ο εφοδιασμός των νέων με γνώσεις και δεξιότητες ανάλογα με τις ικανότητές τους, ώστε να κερδίσουν τα απαραίτητα για τη ζωή και να συμβάλλουν στην περαιτέρω ανάπτυξη της εθνικής οικονομίας.
- Η προσπάθεια να γίνουν οι νέοι ικανοί να προσαρμόζονται στις μεταβαλλόμενες κοινωνικές και οικονομικές συνθήκες.

Ήδη κατά την ανάπτυξη του εκπαιδευτικού υλικού, κατανοήσαμε πόσο δύσκολο τελικά είναι να εκπαιδεύσει κάποιος τον συνάνθρωπό του και πλέον κατανοούμε και τη δυσκολία του έργου όλων των καθηγητών και δασκάλων μας μέχρι σήμερα. Έπρεπε λοιπόν να κρατηθεί κατά νου αφ’ ενός το τι είναι εκπαίδευση, η σημαντικότητά της και η εργασία αυτή, έπρεπε και να πρωτοτυπήσει.

Μεγάλη πρόκληση. Αποφασίστηκε λοιπόν να επιστρατευτούν όσες γνώσεις μας παρείχε το Τμήμα Επιχειρηματικού Σχεδιασμού και Πληροφοριακών Συστημάτων και να αντιμετωπίστεί το εγχείρημα ως επαγγελματικό project.

Το πρώτο πράγμα που έπρεπε να γίνει, ήταν να κατανοηθεί τη θέση μας στην «αγορά». Ποια είναι τα δυνατά μας σημεία ως άνθρωποι και ως ομάδα; Ποιες είναι οι αδυναμίες μας; Ποιες είναι οι ευκαιρίες που μας ανοίγονται και ποιες «απειλές» πρέπει να αντιμετωπίσουμε; Όλα αυτά απαντήθηκαν με μία προσαρμοσμένη στις ανάγκες μας SWOT ανάλυση.

Τα δυνατά μας σημεία ήταν σαφή. Είχαμε την τεχνογνωσία της HTML, η οποία μπορεί να δομήσει υλικό το οποίο να είναι συμβατό με μια σωρεία διαφορετικών ηλεκτρονικών συσκευών, με διαφορετικά λειτουργικά συστήματα, χωρίς κανένα πρόβλημα. Είχαμε πόρους από πρόσβαση στο internet και βιβλιοθήκες ακαδημαϊκών ιδρυμάτων, ενώ μπορούσαμε να επικοινωνούμε με e-mail. Μέλος της ομάδας κατείχε βασικές γνώσεις επεξεργασίας εικόνων, γνωρίζαμε όλοι τεχνογνωσίες Project Management ενώ στην ομάδα είχαμε ένα μέλος με μία μικρή αλλά πολύ χρήσιμη εμπειρία στον προγραμματισμό Visual Basic. Επίσης στο Τμήμα μας, είχαμε παρουσιάσει ανά καιρούς, διάφορες εργασίες που μας είχαν ανατεθεί, με αποτέλεσμα, να έχουμε μία ελαφριά γνώση του πως πρέπει να μεταφέρεται η πληροφορία αποδοτικά.

Οι ευκαιρίες που βλέπαμε μπροστά μας ήταν αρκετά σαφείς κι αυτές. Το βοήθημα μπορούσε να αναπτυχθεί χρησιμοποιώντας την τεχνολογία υπέρ μας, ώστε να ανταποκρίνεται στις σύγχρονες τάσεις του e-learning. Η Ελληνική βιβλιογραφία είναι αρκετά πλήρης, αλλά τόσο για λόγους κόστους όσο και πρακτικότητας, δεν είναι υπεραναλυτική,

μιας και ανάλυση βήμα βήμα με εικόνες, αυξάνει τον όγκο ενός βιβλίου, οπότε ίσως και να αποτρέψει τον τελείως αρχάριο, ιδιαίτερα από θέμα κόστους κτήσης. Το Visual Studio είναι μια τάση του μέλλοντος. Επίσης είχαμε ένα ίδρυμα Δευτεροβάθμιας Εκπαίδευσης και 4 καθηγητές πρόθυμους και διαθέσιμους να μας βοηθήσουν με ανατροφοδότηση πληροφοριών. Επίσης είχαμε πρόθυμους για Beta testers πρωτοετείς φοιτητές του Πανεπιστημίου Πειραιά.

Οι αδυναμίες μας ήταν αρχικά ο περιορισμένος χρόνος και η περιορισμένη οικονομική μας δυνατότητα, σε συνδυασμό με την απόσταση μεταξύ μας και δεν είχαμε και κοινά ωράρια Ένα μέλος της ομάδας μας, υπήρξε πολλά χρόνια μαθητής, ένα μέλος δεν έχει πολλή πείρα με τον προγραμματισμό και κανένα μέλος δεν έχει ουσιαστικά διδακτική πείρα.

Οι «απειλές» που εντοπίσαμε ήταν λίγες αλλά ουσιώδεις. Υπάρχει ήδη βιβλιογραφία στα Αγγλικά, σχετικά απλή και αναλυτική. Υπάρχει στα Αγγλικά εκπαιδευτικό υλικό στον ηλεκτρονικό ιστότοπο του MSDN της Microsoft. Σε συνδυασμό με τους σύγχρονους μεταγλωττιστές ιστοσελίδων, κάποιος, με πιο περίπλοκο τρόπο βέβαια, μπορεί να έχει πρόσβαση σε σεβαστό υλικό. Τέλος, σε κάποια άλλα Εκπαιδευτικά Ιδρύματα της Ελλάδος, υπάρχουν παρόμοια άρθρα και σημειώσεις, γεγονός που μας ανεβάζει πολύ ψηλά τον πήχη.

Συνδυάζοντας όλα τα παραπάνω, προσπαθώντας να εξαλείψουμε τις αδυναμίες μας ή να τις μετατρέψουμε σε δυνάμεις, προσπαθώντας να εξαλείψουμε τις «απειλές» μετατρέποντάς τες σε ευκαιρίες, και αναλογιζόμενοι των δυνατών σημείων μας και ευκαιριών, καταλήξαμε στο συμπέρασμα πως το εκπαιδευτικό υλικό για να συμβάλει, έστω και λίγο, στην πρόοδο της επιστήμης θα πρέπει:

1) Να δώσει μεγάλη βαρύτητα στην απλότητα, κατόπιν στην πληρότητα και τέλος στην αισθητική.
2) Συνεπακόλουθα να μη θεωρηθεί καμία βασική γνώση ως δεδομένη, όπως γίνεται σε άλλα βοηθήματα, εκτός πια της βασικής χρήσης ενός ηλεκτρονικού υπολογιστή και βασικών γνώσεων υποχρεωτικής εκπαίδευσης.
3) Το έργο θα έπρεπε να είναι απλό, τόσο στο εκπαιδευτικό υλικό του, όσο και στη χρήση του. Έπρεπε να ληφθεί υπ' όψιν η ευκολία πρόσβασης στην πληροφορία από αρχάριους χρήστες. Αρχικά προσεγγίστηκε σενάριο ανάπτυξης σε PHP, με διαδραστικές δυνατότητες, προσθήκη video, αλλά όλα αυτά, ναι μεν θα το έκαναν πιο όμορφο και ευχάριστο, αλλά δυστυχώς θα δημιουργούσαν πιθανά προβλήματα συμβατότητας με φυλλομετρητές και λειτουργικά συστήματα, και μία σειρά από προβλήματα που θα βάραιναν τον τελικό χρήστη. Δε θα είχε λοιπόν νόημα η απλότητα του περιεχομένου, αν δεν ήταν και απλό στη χρήση. Γι' αυτό περιοριστήκαμε σε εικόνα και κείμενο μόνο.
4) Η απλότητα απειλείται όμως και από τον όγκο δεδομένων αφ' ενός κι αφ' ετέρου δεν έπρεπε να γίνει το βοήθημα κουραστικό. Οπότε, ως αντικρουόμενος στόχος, έπρεπε να θυσιαστεί μέρος της πληρότητας. Αυτό θα έπρεπε όμως να αντιμετωπιστεί κάπως. Μια παλιά Κινέζικη ρήση λέει: «Δώσε τρία ψάρια σε έναν άνθρωπο και τον τάισες για μία μέρα. Μάθε τον να ψαρεύει και τον ταΐζεις όλη του τη ζωή». Εμπνευσμένοι από τη ρήση, δομήθηκε το βοήθημα με τέτοιο τρόπο, και επιλέχθηκαν να αναλυθούν μέρη της γλώσσας προγραμματισμού, έτσι ώστε ότι παραλειφθεί, να είναι παρεμφερές και να κινείται στην ίδια λογική χρήσης των ήδη επεξηγημένων μερών της. Με αυτό τον τρόπο ο μαθητευόμενος θα μπορέσει έχοντας τα εφόδια, μόνος του, να εμβαθύνει όσο θέλει, έχοντας όμως με εύκολο κι απλό τρόπο εμπεδώσει θεμελιώδεις γνώσεις. Όλα με φιλική στο χρήστη γλώσσα, οικεία κι όχι απόμακρη, ώστε να παραμένει πάντοτε, ευχάριστη εμπειρία.
5) Το έργο πρέπει να είναι καινοτόμο και πρωτότυπο. Πρωτότυπα απλά παραδείγματα, δίνουν μια αρχική πινελιά. Το ανταγωνιστικό μας πλεονέκτημα, δεν αρκείται μόνο σε αυτό. Δομήσαμε τα κεφάλαια με τέτοιο τρόπο, έτσι ώστε το ένα να είναι θεμέλιο του

επομένου, με αποτέλεσμα, να μη χρειάζεται ο αναγνώστης να ανατρέχει σε έναν λαβύρινθο υποδείξεων, αλλά τακτοποιημένα και γραμμικά να ακολουθεί μια λογική συνοχή, ενώ, παράλληλα, γνώση σε διάφορα εκτενή σε όγκο θέματα, δίνεται σταδιακά και εμπλουτίζεται με την πάροδο των κεφαλαίων, κάτι που προτιμούμε να αναφέρουμε εδώ, παρά να επαναλαμβανόμαστε στην τεκμηρίωση κάθε κεφαλαίου.

6) Το έργο πέρασε από γνωμάτευση εκπαιδευτικών και μαθητών από τους οποίους συλλέχθηκαν πολύτιμες πληροφορίες για τη βελτιστοποίησή του. Επίσης, η πολυετής μαθητική εμπειρία του ενός μέλους εκ των τριών, βοήθησε αρκετά, ενώ το αδύναμο στον προγραμματισμό μέλος, συνέβαλλε στην εκτίμηση του κατά πόσο ήταν κατανοητά τα κείμενα, καθώς αυτή η αδυναμία εξαλειφόταν σε πραγματικό χρόνο όσο προχώραγε το έργο.

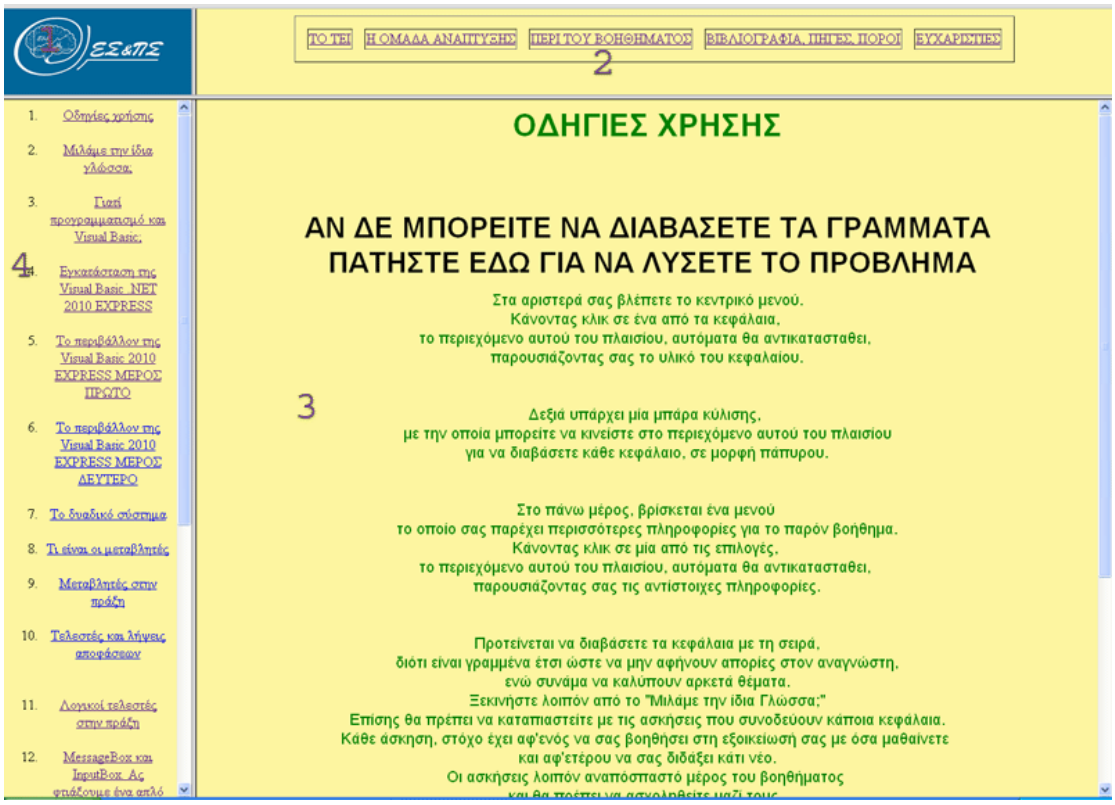
Γνωρίζοντας, λοιπόν, το πώς έπρεπε να κινηθούμε και τι να παράξουμε, δομήθηκε η οργάνωση. Ανατέθηκαν διάφορες εργασίες στα μέλη και δομήθηκε ένα χρονοδιάγραμμα για να παρακολουθείται και να ελέγχεται η εξέλιξη του έργου. Το χρονοδιάγραμμα άλλαξε πολλές φορές κατά την πορεία του έργου ριζικά, ενώ τροποποιήθηκε αμέτρητες φορές. Οι εργασίες που συμπεριλάμβανε το αρχικό χρονοδιάγραμμα ήταν:

1) Τακτικές συσκέψεις των μελών τουλάχιστο μία φορά το μήνα
2) Εγκατάσταση απαραίτητου λογισμικού στους ηλεκτρονικούς μας υπολογιστές
3) Συλλογή πληροφοριών από βιβλία και το διαδίκτυο
4) Σχεδιασμός του μέσου επικοινωνίας (HTML περιβάλλον διεπαφής)
5) Σχεδιασμός διάρθρωση και υλοποίηση των κεφαλαίων (η πιο χρονοβόρα διαδικασία που χωρίζεται σε δύο διαδικασίες για κάθε κεφάλαιο, μία την δημιουργία κι επεξεργασία εικόνων και δεύτερη τη σύνταξη του κειμένου)
6) Συνένωση κεφαλαίων
7) Έλεγχος συνοχής
8) Beta Testing κι έλεγχος ποιότητας από φοιτητές κι εμάς
9) Test αποδοτικότητας σε πραγματικό περιβάλλον διδασκαλίας και συλλογή πληροφοριών για βελτιστοποίηση
10) Μελέτη των πληροφοριών
11) Συμπεράσματα
12) Τροποποίηση και βελτιστοποίηση
13) Τελική μορφοποίηση
14) Ορθογραφικός συντακτικός και νοηματικός έλεγχος
15) Πρακτικός έλεγχος του κώδικα που παρατίθεται εντός του εκπαιδευτικού υλικού
16) Τελική μορφοποίηση
17) Σύνταξη σε επεξεργαστή κειμένου
18) Τελική γνωμάτευση από τον επιβλέποντα καθηγητή
19) Τελική βελτιστοποίηση
20) Εκτύπωση και παράδοση

Επιπλέον, πρέπει να αναφερθούν κάποια ιδιαίτερα γενικά χαρακτηριστικά και προδιαγραφές του εκπαιδευτικού υλικού:

- Το βοήθημα έχει ως φόντο ένα χρώμα ήπιο για τα μάτια έναντι του προκαθορισμένου λευκού, το οποίο κουράζει.
- Η γραμματοσειρά είναι ευανάγνωστη και πρακτική ενώ κεντράροντας το κείμενο, το μάτι δε αναγκάζεται να κάνει άσκοπα μεγάλες κινήσεις για να διαβάσει.

- Ο κώδικας HTML είναι γραμμένος στο χέρι, αποφεύγοντας κατάλοιπα, που έτοιμες λύσεις θα άφηναν στον κώδικα. Συνοδεύεται με ελαφριά τεκμηρίωση με τη μορφή σχολίων εντός των αρχείων.
- Οι εικόνες θα μπορούσαν να είναι τύπου bitmap, υψηλής ποιότητας, αλλά το μέγεθος αυτής της μορφής θα καθιστούσε το βοήθημα μη κατάλληλο για διαδικτυακή χρήση. Έτσι, μικρύναμε τα αρχεία σε μέγεθος, αλλάζοντας τον τύπο τους, με ποιοτική απόκλιση ανεκτή και τους αλλάξαμε μέγεθος, ώστε να προσαρμοστούν ομαλά στο βοήθημα.
- Υιοθετήθηκε ένα υβρίδιο αρχιτεκτονικής του πρότυπου ανάγνωσης ιστοσελίδας Stanford Poynter Project, με την List and Tabs αρχιτεκτονική και ακολουθώντας τον κανόνα των τριών κλικ, δομήθηκε ένα ελπίζουμε εύχρηστο και πρακτικό περιβάλλον διεπαφής. Πιο συγκεκριμένα το περιβάλλον διεπαφής στο βοήθημα χωρίζεται σε 4 μέρη, το πρώτο (1) εκ των οποίων είναι ανενεργό.



Το ανενεργό μέρος είναι πάνω αριστερά (1), κι έχει λογότυπο του Τμήματος της Σχολής μας. Δίπλα υπάρχει ένα μενού με tabs (2), το οποίο παρέχει άμεσα επιπλέον πληροφορίες, αλλάζοντας ένα πολύ μεγάλο χώρο του φυλλομετρητή, ακριβώς από κάτω (3) , στον οποίο επίσης εμφανίζονται και τα περιεχόμενα των εκάστοτε κεφαλαίων. Αριστερά από εκείνο το χώρο και κάτω από το λογότυπο, βρίσκεται το μενού (4) , από το οποίο ο χρήστης μπορεί να επιλέξει κεφάλαιο, ή να διαβάσει πάλι τις οδηγίες χρήσης, που αυτόματα εμφανίζονται ως πρώτη πληροφορία στο περιβάλλον εμφάνισης περιεχομένου.

- Οι οδηγίες χρήσης εμπεριέχουν για κάθε περίπτωση άμεσα μία εικόνα με κείμενο στα Ελληνικά, έτσι ώστε αν υπάρχει πρόβλημα με τις γραμματοσειρές, κάνοντας κλικ πάνω στην εικόνα με το κείμενο, το βοήθημα να δίνει οδηγίες για την επίλυση του προβλήματος.
- Αν και τα περιθώρια (borders) δε φαντάζουν πολύ όμορφα, τα αφήσαμε για τη βέλτιστη προσαρμογή του βοηθήματος, στην ανάλυση οθόνης της αρεσκείας του χρήστη.

- Το βοήθημα συνοδεύεται δε, από τεκμηρίωση αρχειακής διάρθρωσης και δέντρου αρχείων, για τη διευκόλυνση αναβάθμισης και συντήρησής του.

ΤΕΚΜΗΡΙΩΣΗ ΕΙΚΟΝΩΝ ΚΑΙ ΑΡΧΕΙΑΚΗΣ ΔΙΑΡΘΡΩΣΗΣ ΤΟΥΣ

Ο Φάκελος Pictures περιέχει όλες τις εικόνες και τους υποφακέλους με εικόνες. Ο Φάκελος CHAPTERS περιέχει τους Φακέλους της μορφής Chapter_X. Κάθε κεφάλαιο έχει τις εικόνες του σε ξεχωριστό φάκελο της μορφής Chapter_X, όπου X ο αριθμός του κεφαλαίου.
Μορφή ονομασίας φωτογραφιών ανά κεφάλαιο: XXXyy
XXX Από 001 ως 999 είναι το κεφάλαιο.
yy Από 00 ως 99 είναι η σειρά εμφάνισης των εικόνων.
Ο Φάκελος ANSWERS περιέχει τις απαντήσεις σε ασκήσεις δύο κεφαλαίων, σε δύο φακέλους, C_3_A και C_15_A .
Μορφή ονομασίας φωτογραφιών λύσεων ασκήσεων : C_I_A_yy
Chapter I Answers
I = Αριθμός Κεφαλαίου
yy Από 00 ως 99 είναι η σειρά εμφάνισης των εικόνων.
Ο Φάκελος Background περιέχει εικόνες για φόντο.
Ο Φάκελος Troubleshooting περιέχει συγκεντρωτικά, εικόνες για οποιοδήποτε θέμα αντιμετώπισης προβλήματος, σε δύο φακέλους, τους Font και Sp3orDotNET.
Ο Φάκελος Font περιέχει εικόνες για την επίλυση πιθανού προβλήματος με τη γραμματοσειρά.
Ο Φάκελος Sp3orDotNET περιέχει εικόνες για την επίλυση πιθανού προβλήματος με τα service Pack ή το .NET Framework
κατά την εγκατάσταση της Visual Basic 2010 Express.

<u>Δέντρο αρχείων και φακέλων</u>

1] Κεντρικός Φάκελος
Περιεχόμενα:
Όλα τα HTML αρχεία.
Αρχεία για Autorun για την περίπτωση που επιθυμείται να τρέξει αυτόματα από CD ROM.
1.1] Φάκελος PICTURES.
Το text αρχείο τεκμηρίωσης.
Το text αρχείο File Tree
Το πρότυπο αρχείο Κεφαλαίου ChapterXX.html

1.1] Φάκελος PICTURES
Περιεχόμενα:
1.1.1] Φάκελος ANSWERS
1.1.2] Φάκελος BACKGROUND
1.1.3] Φάκελος CHAPTERS
1.1.4] Φάκελος TROUBLESHOOTING
Τα αρχεία:
bpis1.jpg
tei0.jpg
tei1.jpg
tei2.jpg
tei3.jpg
team.gif

1.1.1] Φάκελος ANSWERS
Περιεχόμενα:
1.1.1.1] Φάκελος C_3_A
1.1.1.2] Φάκελος C_15_A

1.1.2] Φάκελος BACKGROUND
Περιέχει τις δύο εικόνες φόντου.
BOHRA.jpg
logo.gif

1.1.4] Φάκελος TROUBLESHOOTING
Περιεχόμενα:
1.1.4.1] Φάκελος Font
1.1.4.2] Φάκελος Sp3orDotNET

1.1.3] Φάκελος CHAPTERS
Περιεχόμενα:
Όλους τους φακέλους της μορφής Chapter_X
Με X από 1 ως 16

1.1.1.1] Φάκελος C_3_A
Περιέχει όλες τις εικόνες λύσης της άσκησης του Κεφαλαίου 3

1.1.1.2 Φάκελος C_15_A
Περιέχει όλες τις εικόνες λύσης της άσκησης του Κεφαλαίου 15

1.1.4.1] Φάκελος Font
Περιέχει όλες τις εικόνες για την επίλυση του πιθανού προβλήματος με τη γραμματοσειρά.

1.1.4.2] Φάκελος Sp3orDotNET
Περιέχει όλες τις εικόνες για την επίλυση του πιθανού προβλήματος
κατά την εγκατάσταση της Visual Basic 2010 Express όσον αφορά
μη εγκατεστημένα Service Packs ή και .NET Frameworks.

1.1.3.X Φάκελοι Chapter_X
Περιέχει όλες τις εικόνες του Κεφαλαίου X

Σύνολο:
406 ΕΙΚΟΝΕΣ
35 ΑΡΧΕΙΑ HTML
3 ΑΡΧΕΙΑ TEXT
2 ΕΙΔΙΚΑ ΑΡΧΕΙΑ για το Autorun σε περίπτωση που διανέμεται σε ηλεκτρονική μορφή, όχι διαδικτυακά.

Συνολικό μέγεθος:
24MB περίπου
Αρχικό στόχο είχαμε το βοήθημα να μη ξεπερνάει τα 30MB σε μέγεθος, γεγονός που καταφέραμε

ΚΕΦΑΛΑΙΟ 1

ΜΙΛΑΜΕ ΤΗΝ ΙΔΙΑ ΓΛΩΣΣΑ;

Θα αναρωτιέστε τώρα, τι ανούσια ερώτηση είναι αυτή. Σήμερα, που η τεχνολογικές εξελίξεις και οι επιστήμες είναι σε ακμή, είναι φανερό πως κάθε κλάδος, έχει τους δικούς του όρους και τη δική του διάλεκτο. Ας υποθέσουμε ότι έχετε έναν αδελφό που δουλεύει σε κατάστημα πώλησης κατοικίδιων ζώων, και επίσης ας υποθέσουμε πως είναι άσχετος με την πληροφορική. Εσείς θέλετε να πάει να σας αγοράσει ένα ποντίκι για τον ηλεκτρονικό σας υπολογιστή. Εσείς ξέρετε τι είναι το ποντίκι για έναν ηλεκτρονικό υπολογιστή. Εκείνος όχι. Αν του ζητήσετε λοιπόν να σας αγοράσει ένα δε θα πρέπει να εκπλαγείτε, αν σας φέρει ένα χάμστερ.

Οπότε όταν απευθυνόμαστε σε ένα ευρύ κοινό, με ειδικούς όρους, θα πρέπει να λάβουμε υπ' όψιν, την πιθανότητα κάποιοι, με το δίκιο τους, να μη μας καταλαβαίνουν. Ωστόσο υποθέτουμε πως γνωρίζετε τι είναι το ποντίκι, τι είναι το πληκτρολόγιο, και γενικά υποθέτουμε πως γνωρίζετε τα βασικά όσον αφορά τη χρήση ενός ηλεκτρονικού υπολογιστή. Ο προγραμματισμός είναι ένα ειδικό θέμα και θα ήταν λάθος να υποθέσουμε πως όλοι οι άνθρωποι που θέλουν να μάθουν προγραμματισμό καταλαβαίνουν και γνωρίζουν εξ αρχής την ορολογία του. Γι' αυτό το λόγο, το παρόν κεφάλαιο στόχο έχει να σας εξηγήσει αυτή την ορολογία, έτσι ώστε τα επόμενα κεφάλαια να σας γίνουν σαφή και διασκεδαστικά. Επειδή δε θέλουμε αυτό το κεφάλαιο να μοιάζει με γλωσσάριο, ούτε να γίνει κουραστικό, θα παρουσιάσουμε μόνο λίγους βασικούς όρους, που πρέπει να γνωρίζετε. Οτιδήποτε άλλο, θα επεξηγείται εντός του κεφαλαίου που θα πρωτοεμφανίζεται.

Πρόγραμμα:

Σύμφωνα με τον γενικό ορισμό που έδωσε ο John Von Neumann το 1945, το πρόγραμμα αποτελείται από μια συνεχή αλληλουχία εντολών τις οποίες ο υπολογιστής καλείται να εκτελέσει μία προς μία για να παραχθεί το επιθυμητό αποτέλεσμα. Τι παρατηρούμε λοιπόν; Κατά τον John Von Neumann το πρόγραμμα είναι μία σειρά εντολών. Στόχο έχουν την επίλυση ενός προβλήματος. Υπάρχουν προβλήματα, που είναι πολύ περίπλοκα. Μπορεί ένα πρόβλημα να αποτελείται από πολλά μικρότερα προβλήματα. Το πρόγραμμα λοιπόν μπορεί να λύνει ένα ή και περισσότερα προβλήματα, για να παραχθεί το επιθυμητό αποτέλεσμα. Ένα πρόγραμμά ή μία συλλογή προγραμμάτων, λέγεται και λογισμικό. Βασικό δομικό στοιχείο ενός προγράμματος, είναι ο αλγόριθμος. Το πρόγραμμα ουσιαστικά είναι ένας μεγάλος αλγόριθμος, ο οποίος μπορεί να αποτελείται από μικρότερους αλγόριθμους. Τι είναι όμως ο αλγόριθμος;

Αλγόριθμος:

Ως αλγόριθμος ορίζεται μια πεπερασμένη σειρά ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος. Δηλαδή; Ας τα πάρουμε τα πράγματα από την αρχή. Πεπερασμένη, σειρά ενεργειών. Άρα μιλάμε για ενέργειες που γίνονται με μία σειρά. Πόσες είναι αυτές οι ενέργειες που κάνουμε; Πεπερασμένες λέει. Όσες χρειάζεται λοιπόν, φτάνει να μην είναι ατελείωτες. Μπορεί να είναι πέντε, δέκα, εκατό χιλιάδες, αλλά ποτέ άπειρες, διότι δε θα επιλυόταν ποτέ το πρόβλημα. Οι ενέργειες αυτές λέει είναι αυστηρά καθορισμένες. Δηλαδή; Απλό. Δεν υπάρχει καμία ασάφεια στη διατύπωσή τους. Για παράδειγμα μια εντολή "Βάψε μπλε, με το πινέλο-ρολό τον τοίχο" μας λέει με αρκετή σαφήνεια τι να κάνουμε. Αν είχαμε μπροστά μας κόκκινη, κίτρινη και πράσινη μπογιά, αλλά και μια συλλογή από διαφορετικά πινέλα, τότε μια εντολή "Βάψε τον τοίχο" θα περιείχε ασάφειες. Τι χρώμα; Με ποιό πινέλο; Ασάφειες σε αλγόριθμο δεν υπάρχουν λοιπόν. Είναι, λέει μετά, αυτές οι ενέργειες, εκτελέσιμες σε πεπερασμένο χρόνο. Τι

εννοεί; Αν οι ενέργειες δεν τελείωναν ούτε μετά το τέλος του Κόσμου, τότε ποιο το νόημα; Θέλουμε να βρούμε λύση και μάλιστα το συντομότερο δυνατόν. Δεν έχουμε μπροστά μας όλη την αιωνιότητα. Οπότε δεν πρέπει να απαιτείται άπειρος χρόνος για την εκτέλεση μιας ενέργειας. Οι ενέργειες κάποτε αρχίζουν και κάποτε τελειώνουν. Αυτό εννοεί.

Σύνταξη:
Σύνταξη είναι θα μπορούσαμε να πούμε το σύνολο των κανόνων για τον τρόπο με τον οποίο συνδυάζονται οι λέξεις για τον σχηματισμό προτάσεων σε μια γλώσσα. Κάθε γλώσσα έχει και το συντακτικό της. Σωστά; Η Ελληνική έχει διαφορετική σύνταξη από την Αγγλική. Έτσι και κάθε γλώσσα προγραμματισμού έχει διαφορετική σύνταξη. Υπάρχουν ειδικές λέξεις-κλειδιά, συγκεκριμένοι κανόνες στίξης κ.τ.λ. Κάθε γλώσσα προγραμματισμού έχει διαφορετικό συντακτικό. Όταν έναν αλγόριθμο τον γράφουμε χρησιμοποιώντας μια γλώσσα προγραμματισμού, τότε το κείμενο αυτό το ονομάζουμε κώδικα. Υπάρχει άραγε κάποιος τρόπος, να διατυπώνω έναν αλγόριθμο, και να μπορούν μετά, γνώστες διαφορετικών γλωσσών προγραμματισμού, να τον μετατρέπουν σε κώδικα για τη γλώσσα προγραμματισμού που ξέρουν; Υπάρχει.

Ψευδοκώδικας:
Ο ψευδοκώδικας είναι εργαλείο που χρησιμοποιείται από προγραμματιστές, κυρίως στα αρχικά στάδια της σχεδίασης και κατασκευής ενός προγράμματος. Με τον ψευδοκώδικα, τα βήματα μιας λύσης ή η μορφή ενός αλγόριθμου περιγράφονται με σύντομες και περιεκτικές προτάσεις που όμως υπακούουν σε μια τυποποίηση που πλησιάζει την τυποποίηση μιας γλώσσας προγραμματισμού.

Τελεστής:
Αυτή η έννοια στον προγραμματισμό, είναι ουσιαστικά απλή να εξηγηθεί. Ωστόσο αν κάποιος κάνει το λάθος να ανατρέξει στο μαθηματικό επιστημονικό ορισμό του τελεστή τότε θα διαβάσει κάτι σαν το παρακάτω: Ο τελεστής στα μαθηματικά ορίζεται γενικά ως μία συνάρτηση που δρα πάνω σε κάποια άλλη συνάρτηση, μετασχηματίζοντάς την κατά έναν καθορισμένο τρόπο. Μπορεί να θεωρηθεί γενίκευση της έννοιας της συνάρτησης, καθώς οι συναρτήσεις δρουν συνήθως πάνω σε μεμονωμένα "αντικείμενα", ενώ ένας τελεστής μπορεί να δράσει πάνω στη "μορφή" μιας συνάρτησης ως σύνολο και να δώσει μια άλλη συνάρτηση. Ένας τελεστής παριστάνεται συνήθως με ένα σύμβολο, το οποίο τίθεται μπροστά από μια συνάρτηση (με τη γενική έννοια) και την αλλάζει σε κάποια άλλη συνάρτηση των ίδιων μεταβλητών. Σας φαίνεται απλό; Ούτε εμάς!
Ας δούμε πιο απλά το ζήτημα. Λέμε στα μαθηματικά 1 + 3 = 4. Αυτό το <<συν>> είναι ένας τελεστής. Κοινώς έχετε κατά νου πως ένα σύμβολο που υποδηλώνει μια πράξη που γίνεται, μεταξύ δύο τελεστέων, λέγεται τελεστής. Και τι είναι οι τελεστέοι;
Στο παράδειγμα 1 + 3 = 4 τελεστέοι του + είναι το (1) και το (3). Επίσης υπάρχει και ο τελεστής εξίσωσης, το = <<ίσον>>. Οι τελεστέοι του, είναι η μαθηματική παράσταση (1+3) και το (4) και λέμε πως τα εξισώνει. Ωστόσο, θα μάθουμε στη συνέχεια και τους λογικούς τελεστές, που είναι κάπως διαφορετικοί από τους μαθηματικούς.

GUI:
Αρχικά από το Graphical User Interface. Σημαίνει Γραφικό Περιβάλλον Χρήστη. Είναι ένα σύνολο γραφικών στοιχείων, τα οποία εμφανίζονται στην οθόνη κάποιας ψηφιακής συσκευής, όπως για παράδειγμα του ηλεκτρονικού υπολογιστή και χρησιμοποιούνται για την αλληλεπίδραση του χρήστη με τη συσκευή αυτή. Παρέχουν στον τελευταίο, μέσω γραφικών, ενδείξεις και εργαλεία προκειμένου αυτός να φέρει εις πέρας κάποιες επιθυμητές λειτουργίες. Το βέλος στην οθόνη, που το κινείτε με το ποντίκι, καθώς κι η επιφάνεια

εργασίας, είναι μέρος του GUI του λειτουργικού συστήματος που χρησιμοποιείτε (Linux,Windows,OsX ή άλλο). Για τον λόγο αυτό δέχεται το GUI και είσοδο από τον χρήστη και τα διάφορα μέρη του αντιδρούν ανάλογα στα συμβάντα που αυτός προκαλεί με τη βοήθεια κάποιας συσκευής εισόδου (π.χ. πληκτρολόγιο, ποντίκι).

Εκτελέσιμο αρχείο:
Είναι ένα αρχείο αυτόνομο, το οποίο συνήθως με διπλό κλικ εκτελεί κώδικα που περιέχει, διότι πολύ απλά, είναι πρόγραμμα. Δε χρειάζεται η γλώσσα προγραμματισμού, με την οποία το γράψαμε, να είναι εγκατεστημένη στον ηλεκτρονικό υπολογιστή για να τρέξει το πρόγραμμα αυτό. Είναι εκτελέσιμο, δηλαδή βρίσκεται σε μια μορφή που μόνο του εκτελείται.

COMPILE:
Είναι η μεταγλώττιση ενός προγράμματός, από κώδικα σε γλώσσα μηχανής. Ο compiler είναι το πρόγραμμα που εκτελεί αυτή την μεταγλώττιση. Διότι η μηχανή, καταλαβαίνει αλληλουχίες από "0" και "1". Όλες οι εντολές λοιπόν του κώδικα, για να γίνουν κατανοητές στη μηχανή, μετατρέπονται σε γλώσσα μηχανής. Έτσι δημιουργούνται τα εκτελέσιμα αρχεία.

ΔΙΕΡΜΗΝΕΥΤΗΣ (INTERPRETER):
Αν θέλουμε να δοκιμάσουμε κατά πόσο δουλεύει σωστά το πρόγραμμά μας, την ώρα που το γράφουμε για παράδειγμα στη Visual Basic, αν δεν υπήρχε ο διερμηνευτής, θα έπρεπε κάθε φορά να το κάνουμε compile για να το τρέξουμε κατόπιν στη μορφή γλώσσας μηχανής. Ωστόσο, αυτό δεν είναι καθόλου βολικό. Πώς θα εντοπίζαμε τα λάθη μας σε ένα πρόγραμμα; Ο Διερμηνευτής, είναι ένα πρόγραμμα, το οποίο μετατρέπει τον κώδικά μας σε γλώσσα μηχανής, εντολή προς εντολή, και εκτελεί τις εντολές αυτές, μία, μία! Έτσι, μπορούμε να έχουμε πρόσβαση εμείς στον κώδικά μας και να παίρνουμε τις πληροφορίες που χρειαζόμαστε για το πώς τρέχει το πρόγραμμά μας, σα να ήταν ήδη μεταγλωττισμένο σε γλώσσα μηχανής.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Το κεφάλαιο αυτό στόχο έχει την εισαγωγή του μαθητευόμενου σε κάποιες βασικές έννοιες. Πρόγραμμα, λογισμικό, αλγόριθμος, σύνταξη, κώδικας, ψευδοκώδικας τελεστής, τελεστέος γραφικό περιβάλλον χρήστη, εκτελέσιμο αρχείο, μεταφραστής, διερμηνευτής και μεταγλώττιση είναι οι όροι που επεξηγούνται, έτσι ώστε ο μαθητευόμενος να μπορεί να κατανοήσει στη συνέχεια τα κεφάλαια που θα ακολουθήσουν. Αυτό γίνεται για να μην πάρει μεγάλη έκταση το συγκεκριμένο κεφάλαιο και γίνει κουραστικό. Ενημερώνεται δε, ότι νέοι όροι θα αναλύονται και θα επεξηγούνται στο κεφάλαιο που θα πρωτοεμφανίζονται. Επεξηγείται το γιατί αυτό το κεφάλαιο είναι απαραίτητο, διότι ο μαθητευόμενος αντιμετωπίζει το υλικό προς μάθηση πολύ πιο θετικά, όταν ξέρει γιατί πρέπει να το μάθει.

ΚΕΦΑΛΑΙΟ 2
ΓΙΑΤΙ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ; ΓΙΑΤΙ VISUAL BASIC;

Όλοι αναρωτιούνται όταν πρόκειται να μάθουν κάτι, ποιος είναι ο λόγος που το μαθαίνουν. Ένας μαθητής Δημοτικού διδάσκεται Μαθηματικά. "Και γιατί παρακαλώ διδάσκομαι τα μαθηματικά;", ρωτάει. Δεκάδες οι απαντήσεις που μπορούν να του δοθούν. Πολλές φορές όμως, κανείς δε του τις δίνει. Κι όμως! Αυτές οι απαντήσεις είναι απαραίτητες! Αν ξέρεις γιατί κάνεις κάτι, το κάνεις με διαφορετικό ενδιαφέρον. Οπότε οφείλουμε να σας δώσουμε κι εμείς απάντηση σε αυτό το "γιατί". Γιατί λοιπόν να μάθει κάποιος προγραμματισμό ηλεκτρονικών υπολογιστών; Αν κάποιος γίνει προγραμματιστής και δουλειά του είναι να φτιάχνει προγράμματα, τότε το όφελος είναι προφανές. Αποκτάει χρήματα, δημιουργώντας κάτι χρήσιμο.

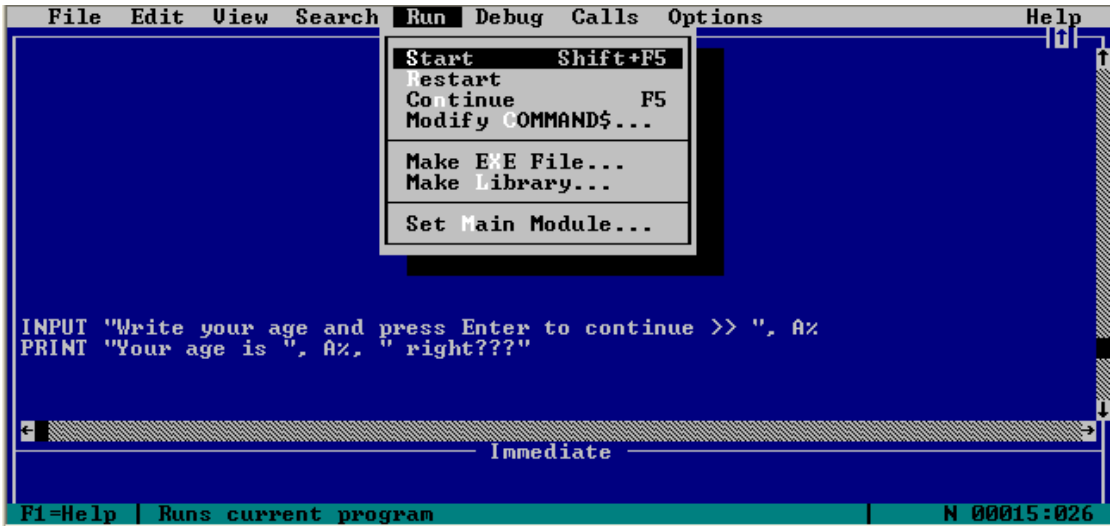
Από την άλλη, σχεδόν οτιδήποτε θελήσει κάποιος, μπορεί να το βρει έτοιμο στο διαδίκτυο, δωρεάν ή επί πληρωμή. Αν δε βρείτε ένα δωρεάν πρόγραμμα που να ικανοποιεί κάποια ανάγκη σας, μπορεί να μη σας νοιάζει και πολύ να πληρώσετε κάποιο αντίτιμο, για να αποκτήσετε μια εμπορική έκδοση λογισμικού που θα σας διευκολύνει. Εντάξει, αν προγραμματίζατε θα εξοικονομούσατε κάποια χρήματα, αλλά δε φαντάζει και πολύ σπουδαίο όφελος. Αποφύγατε να κάτσετε να κουραστείτε να μάθετε προγραμματισμό, αποφύγατε τον κόπο να φτιάξετε το πρόγραμμα. Δώσατε ένα χρηματικό ποσό και βρήκατε τη λύση. Σωστά; Πριν λίγο αναφέραμε ότι σχεδόν οτιδήποτε θελήσει κάποιος, μπορεί να το βρει έτοιμο στο διαδίκτυο. Σχεδόν οτιδήποτε. Σχεδόν! Εκεί κρύβεται το μέγιστο όφελος του να μάθετε προγραμματισμό.

Πολλοί άνθρωποι στον κόσμο έχουν κοινές ανάγκες με εσάς. Πολλοί άνθρωποι θέλουν να γράψουν κείμενο στον ηλεκτρονικό τους υπολογιστή, να το αποθηκεύσουν, να το επεξεργαστούν κ.ο.κ. Έτσι υπάρχουν εμπορικοί, αλλά και δωρεάν επεξεργαστές κειμένου. Για την ακρίβεια υπάρχουν ολόκληρες σουίτες γραφείου, δηλαδή συλλογές προγραμμάτων για διευκόλυνση εργασιών γραφείου και όχι μόνο. Ωστόσο, ας υποθέσουμε πως μόνο εσείς, έχετε μια ιδιαίτερη ανάγκη, που κανείς άλλος δεν έχει. Για παράδειγμα ας υποθέσουμε πως στην εργασία σας, πρέπει να παίρνετε δέκα αριθμούς και να κάνετε με αυτούς, γύρω στις εικοσιπέντε πράξεις. Αυτή η διαδικασία, ας υποθέσουμε πως είναι επαναλαμβανόμενη. Ας υποθέσουμε ότι κάνετε αυτό για οκτώ ώρες και πως σας παίρνει μισή ώρα να ολοκληρώσετε τη διαδικασία μία φορά. Κάνετε τη διαδικασία αυτή 16 φορές το οχτάωρο δηλαδή! Έχετε το άγχος να μην κάνετε λάθος, και ας υποθέσουμε τώρα πως πρέπει να απαντάτε και στα τηλέφωνα που χτυπάνε συνεχώς, έχετε βαρεθεί να κάνετε συνεχώς τα ίδια και τα ίδια....Μπορείτε να φανταστείτε, σε αυτό το υποθετικό παράδειγμα, πόσο ευκολότερη θα ήταν η ζωή σας, αν είχατε ένα προγραμματάκι, που σε δύο λεπτά θα περνούσατε τα δεδομένα σας και με ένα κλικ θα σας έδινε το αποτέλεσμα;

Θα είχατε άνεση, καλύτερες συνθήκες εργασίας, μεγαλύτερη ποιότητα ζωής.
Όμως μόνο εσείς έχετε αυτή την ανάγκη! Αν κάποια εταιρία παραγωγής λογισμικού δεν πουλήσει το πρόγραμμα αυτό σε κανέναν άλλο, διότι μόνο εσείς το έχετε ανάγκη, τότε γιατί να το φτιάξει; Η εταιρία θέλει μαζικές πωλήσεις. Μόνο εσείς έχετε αυτή την ανάγκη!
Και εξατομικευμένες ανάγκες, στην παραγωγή λογισμικού κοστίζουν πολύ ακριβά.
Αντί, λοιπόν, να σας κοστίσουν πανάκριβα οι λύσεις στις ιδιαίτερες ανάγκες σας,
ή να μείνουν ακάλυπτες οι ανάγκες αυτές, μπορείτε να φτιάξετε εσείς τη λύση.
Από την αρχαιότητα ο άνθρωπος προόδευσε, βρίσκοντας λύσεις. Μια πέτρα για να σκάβει μετατράπηκε περίτεχνα σε κοπίδι και μετά σε τσεκούρι. Το τσεκούρι έκοψε δέντρα, τα δέντρα έγιναν σπίτια. Ο άνθρωπος πάντα έφτιαχνε κάτι για να κάνει τη ζωή του ευκολότερη.

Κι όσο έκανε τη ζωή του πιο εύκολη, τόσο πιο πολύ προόδευε. Σήμερα, μετά από τόσα χιλιάδες χρόνια, έχουμε φτάσει πια να έχουμε στην υπηρεσία μας περίπλοκες μηχανές, τους ηλεκτρονικούς υπολογιστές, που ανάλογα με το πως τις χειριζόμαστε, μας δίνουν λύσεις σε διάφορα προβλήματα. Ο προγραμματισμός, αν τον μάθετε και τον εφαρμόσετε σωστά, θα δώσει προσαρμοσμένη στις ανάγκες σας χρησιμότητα, στον ηλεκτρονικό σας υπολογιστή. Αυτό θα είναι και το μεγαλύτερο όφελος που θα πάρετε. Θα μπορείτε να φτιάχνετε λύσεις, για εσάς και για τους γύρω σας.

Γιατί Visual BASIC όμως; Εδώ πρέπει να αναφέρουμε πως η Visual Basic 2010, είναι μια αντικειμοστραφής γλώσσα προγραμματισμού, δηλαδή είναι μια γλώσσα προγραμματισμού, στην οποία ένα πρόγραμμα, αποτελείται από αντικείμενα. Οι πρόγονοί της Visual Basic, είχαν περιβάλλον εργασίας που ήταν βασισμένο σε επεξεργασία κειμένου (κώδικα), κι ενώ ήταν πολύ δύσχρηστο περιβάλλον συγκρινόμενο με τα σημερινά, κάποιες εκδοχές της, όπως η QBASIC (για τα PC) είχαν κερδίσει πολύ έδαφος, λόγω της απλότητας και της ευκολίας της BASIC. Η παρακάτω εικόνα μας δείχνει το περιβάλλον της QBasic V4.5, η οποία ήταν μια πολύ παλιά έκδοση για PC.



Η ονομασία "BASIC" γράφεται ουσιαστικά B.A.S.I.C. και σημαίνει Beginner's All-purpose Symbolic Instruction Code, δηλαδή, Συμβολικός Κώδικας Εντολών Γενικής Χρήσης για Αρχάριους. Γεννήθηκε στο Dartmouth το 1964 επηρεασμένη από τις FORTRAN και ALGOL60 από τους John Kemeny και Thomas Kurtz. Στόχος του project ήταν η δημιουργία μιας γλώσσας προγραμματισμού που να είναι εύκολη στη μάθηση για αρχάριους, να είναι γενικού σκοπού, να δύναται να προστεθούν εξελιγμένα χαρακτηριστικά για προχωρημένους χρήστες,δίχως να θυσιάζεται η απλότητα για τον αρχάριο χρήστη, να είναι διαδραστική, (Διαδραστικότητα είναι η δυνατότητα ενός μέσου να δέχεται αμφίδρομη επικοινωνία, δηλαδή να στέλνει και να λαμβάνει ταυτόχρονα πληροφορίες) να εμφανίζει ξεκάθαρα και φιλικά προς τον χρήστη μηνύματα σφαλμάτων, να είναι γρήγορη, αν τα προγράμματα είναι μικρά και να μην απαιτούνται γνώσεις του υλικού του ηλεκτρονικου υπολογιστή. Η γλώσσα διαδόθηκε πολύ γρήγορα. Σε αυτό μάλιστα συνέβαλλαν οι William Gates και Paul Allen, επειδή το 1976 έφτιαξαν για τον μικρουπολογιστή MITS Altair 8800
έναν διερμηνευτή BASIC τον Altair BASIC.

GATES

ALLEN

Ο Altair ήταν ένα κιτ "φτιάξτο μόνος σου" που συναρμολογούσες και είχες έναν ηλεκτρονικό υπολογιστή! Το κιτ ήταν ουσιαστικά όλα τα εξαρτήματα του ηλεκτρονικού υπολογιστή μεμονωμένα, έτσι ώστε ο αγοραστής να τον συναρμολογήσει μόνος. Κάτι σα μοντελισμός δηλαδή. Οι σχεδιαστές του, δεν το περίμεναν, αλλά είχε απίστευτη εμπορική επιτυχία για την εποχή του!

Ο Altair ήταν ο πρώτος μικροϋπολογιστής που διατέθηκε μαζικά. Βασίστηκε στο μικροεπεξεργαστή Intel 8080 που έτρεχε στα 2 MHz! Κι αν το όνομα William Gates δε σας θυμίζει κάτι, σας αναφέρουμε πως ολόκληρο το όνομα του κυρίου αυτού είναι William Henry "Bill" Gates III. Με το διερμηνευτή αυτό, ο Gates και ο Allen θεμελίωσαν την εταιρία Microsoft Inc. Πιο κάτω βλέπετε τους Allen και Gates να περιστοιχίζονται από διάφορους ηλεκτρονικούς υπολογιστές εποχής.

Όσο για την περεταίρω διάδοση της BASIC, τις δεκαετίες του 80 κι 90, την εποχή που δέσποζαν οι Home Computers, όπως Amstrad, Spectrum, Commodore 64, Commodore Amiga, Atari ST κ.τ.λ. πολλοί από αυτούς συνοδεύονταν με δισκέττα που να περιέχει μια BASIC για το μηχάνημα, είτε είχαν τη BASIC μέσα τους σε chip (Built in). Χαρακτηριστικό παράδειγμα Built in ήταν οι Spectrum και Amstrad.

Παρακάτω βλέπετε έναν Amstrad που μόλις έχει τεθεί σε λειτουργία. Παρατηρείτε την άμεσα φορτωμένη BASIC 1.0;

Παρακάτω βλέπετε τον υπολογιστή Commodore Amiga 500.Η Amiga όπως την αποκαλούσαν, ήταν η "βασίλισσα" των Home Computers! Το λειτουργικό σύστημα που φαίνεται στη φωτογραφία, λεγόταν Workbench και το είχε αναπτύξει η Microsoft.

Από την Amiga δεν ήταν δυνατόν να λείπει η BASIC. Παρακάτω βλέπετε 4 προγράμματα που τρέχουν ταυτόχρονα, τρέχοντας μόνο ένα αρχείο BASIC!

Η φιλικότητα προς τον χρήστη και η απλότητά της BASIC, ήταν τα βασικά συστατικά της επιτυχίας της. Σήμερα η Visual Basic 2010 της Microsft, διατηρεί σε σπουδαίο βαθμό την απλότητα της BASIC, ενώ τη συνδυάζει με αντικειμενοστραφές αλλά και ολοκληρωμένο παραθυρικό γραφικό περιβάλλον ανάπτυξης. Είναι η πλέον κατάλληλη γλώσσα λοιπόν κατά την άποψή μας, για να εισάγει έναν αρχάριο, στο περιβάλλον αυτό. Γιατί όμως την έκδοση VISUAL BASIC .NET 2010 και όχι την Visual Basic 6 ή κάποια παλιότερη; Είναι αλήθεια πως η Visual Basic 6 υπήρξε μια πολύ φιλική προς το χρήστη γλώσσα. Ωστόσο, η VISUAL BASIC .NET 2010 πέρα των έξτρα χαρακτηριστικών που την κάνουν σαφώς ισχυρότερη είναι ότι πιο νέο στον τομέα της Visual Basic. (Έτος έναρξης συντάξεως του παρόντος είναι το 2010), οπότε επιλέξαμε αυτή για να προσφέρουμε επίκαιρη γνώση.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό επεξηγούμε τα οφέλη του να μάθει κάποιος προγραμματισμό, ώστε να δώσουμε επιπλέον ενδιαφέρον και κίνητρο στο μαθητευόμενο, να ασχοληθεί με το αντικείμενο. Επίσης παραθέτουμε τους λόγους να μάθουν συγκεκριμένα τη Visual Basic και τα οφέλη που θα αποκομίσουν από τη γνώση αυτή, καθώς και μια σύντομη ιστορική αναδρομή της γλώσσας.

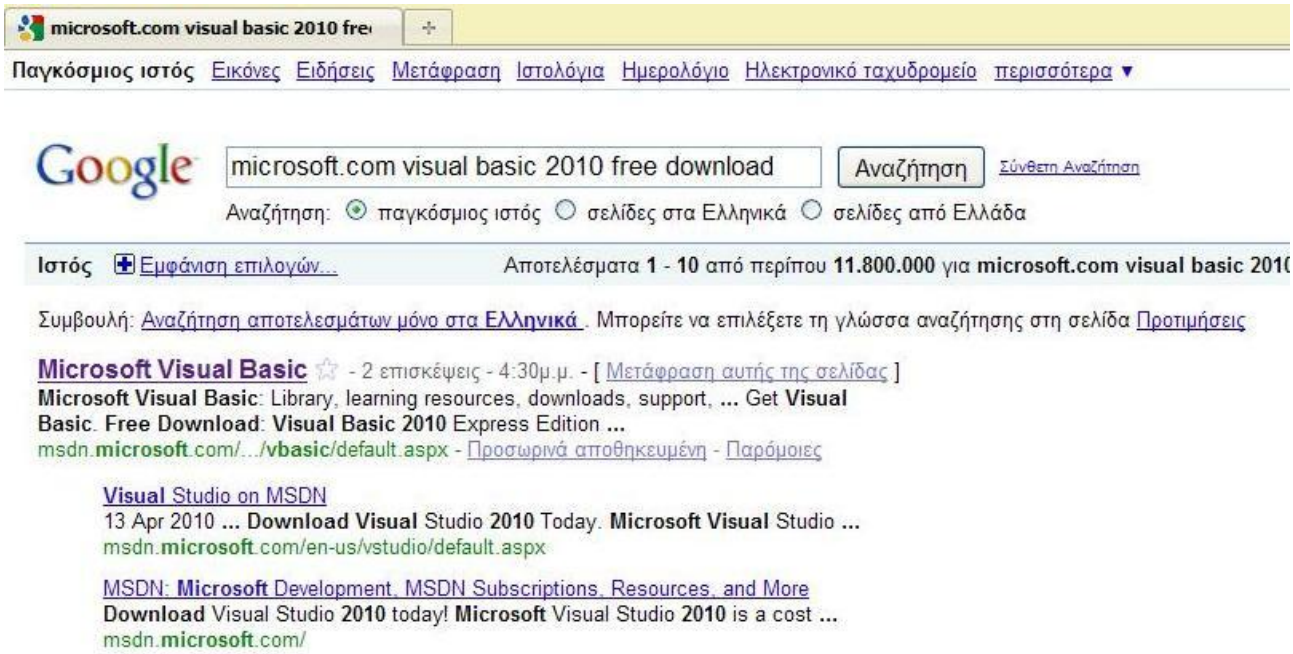
ΚΕΦΑΛΑΙΟ 3
ΕΓΚΑΤΑΣΤΑΣΗ ΤΗΣ VISUAL BASIC .NET 2010 EXPRESS

ΠΡΟΤΕΙΝΕΤΑΙ ΝΑ ΔΙΑΒΑΣΕΤΕ ΟΛΟΚΛΗΡΗ ΤΟ ΚΕΦΑΛΑΙΟ ΜΙΑ ΦΟΡΑ ΚΑΙ ΜΕΤΑ ΝΑ ΑΡΧΙΣΕΤΕ ΤΗΝ ΕΓΚΑΤΑΣΤΑΣΗ ΒΗΜΑ ΒΗΜΑ

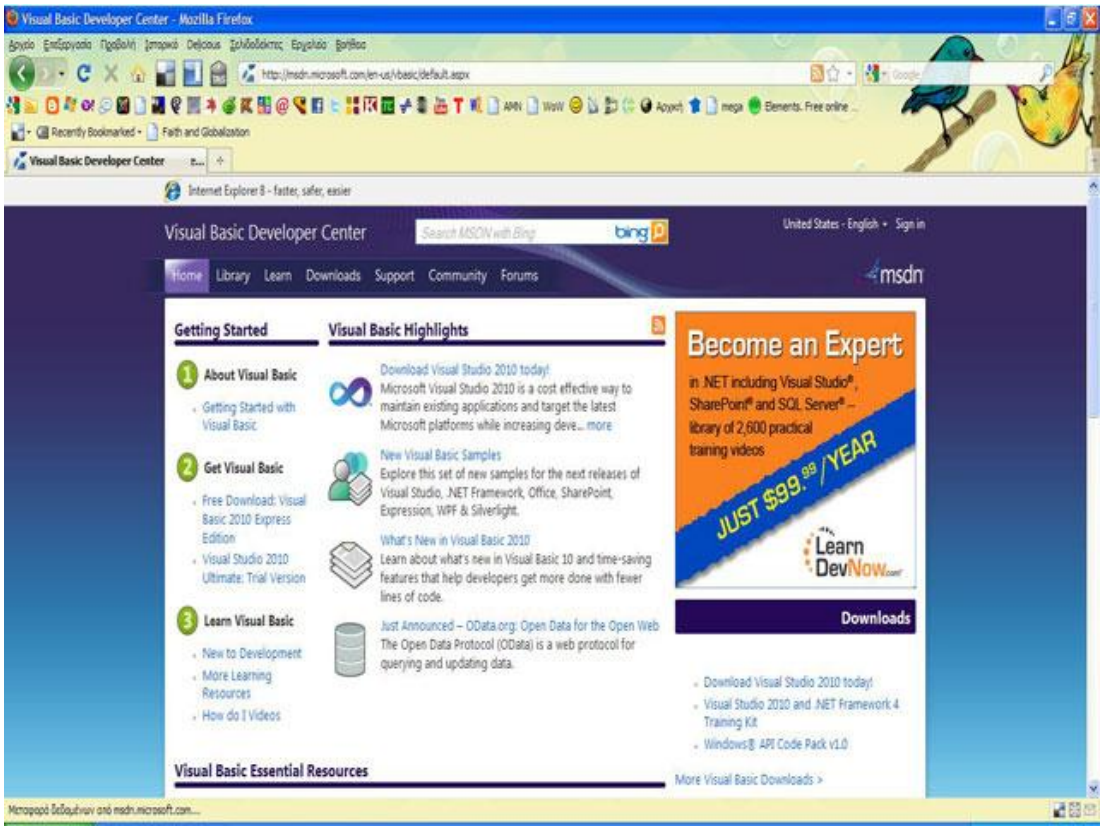
Η εταιρία Microsoft προσφέρει μια έκδοση της VISUAL BASIC .NET 2010 δωρεάν. Για διδακτικούς σκοπούς. Φυσικά πολλές λειτουργίες της εμπορικής έκδοσης, δεν είναι διαθέσιμες στη δωρεάν. Ωστόσο αυτό δε θα σας περιορίσει καθόλου στο να τη μάθετε. Θα μπορείτε να γράψετε τα δικά σας προγράμματα και να τα τρέχετε μέσω του περιβάλλοντος της, αλλά δε θα μπορείτε να τα μετατρέψετε σε αυτόνομα εκτελέσιμα αρχεία.

Ο πιο εύκολος τρόπος να την εγκαταστήσετε, είναι να την κατεβάσετε από την ιστοσελίδα της Microsoft. Πιο κάτω θα δείτε μια τυπική εγκατάσταση, βήμα βήμα. Το πιο πιθανό είναι στο μέλλον να υπάρξουν διαφορές στην εγκατάσταση. Ωστόσο, αν δείτε αυτή την εγκατάσταση, δεν πιστεύουμε πως θα δυσκολευτείτε με κάποια αλλαγή. Άλλωστε αλλάζουν οι εγκαταστάσεις, με γνώμονα την απλότητα.

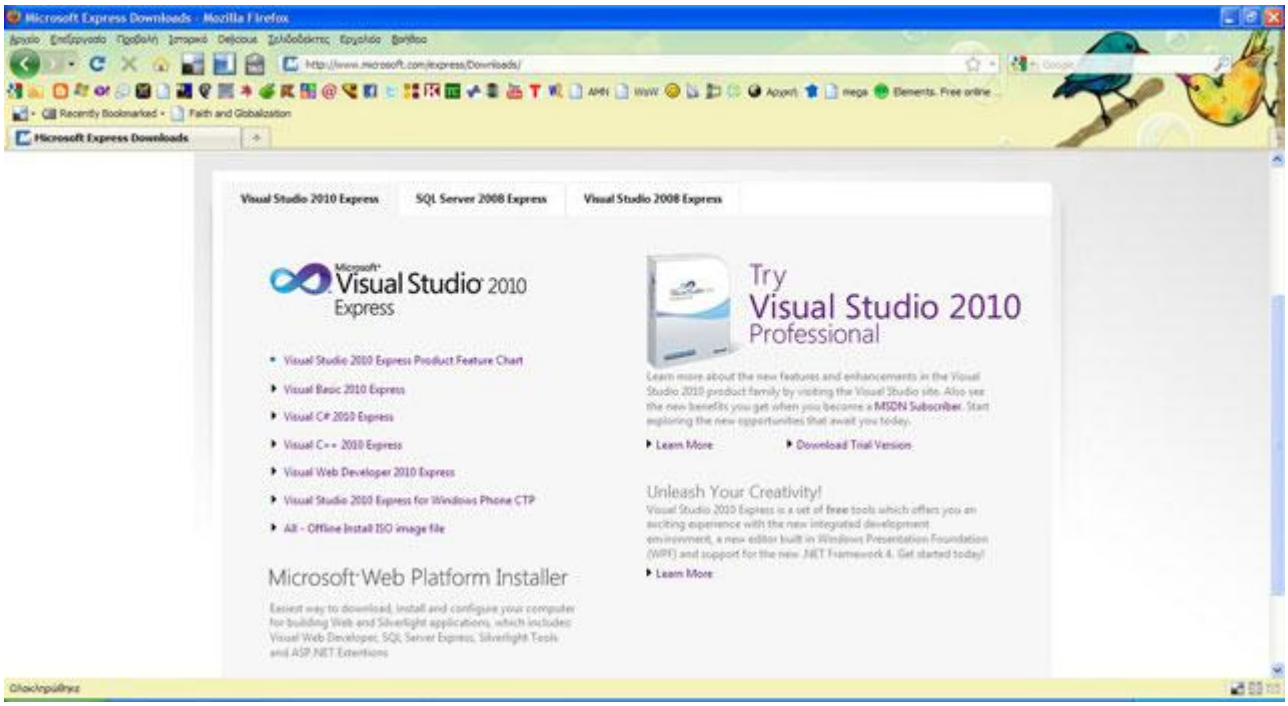
Στη μηχανή αναζήτησης της αρεσκείας σας, κάντε αναζήτηση, όπως βλέπετε στην παρακάτω εικόνα. Εμείς προτιμήσαμε το Google για την αναζήτηση. Έχετε υπ'όψιν σας, ότι το internet είναι ένας χώρος δυναμικός. Οι ιστοσελίδες μπορεί να αλλάζουν τελείως μορφή, ενώ είναι δυνατόν η συγκεκριμένη δωρεάν διανομή να πάψει να διανέμεται ή να αντικατασταθεί με μία πιο νέα έκδοση. Ο οδηγός αυτός, στόχο έχει να σας δείξει μια τυπική διαδικασία εγκατάστασης της Visual Basic .NET Express, ώστε να σας διευκολύνει. Πιστεύουμε πως αν έχετε τη γενική εικόνα μιας εγκατάστασης, μικροαλλαγές και σε αυτό τον τομέα, δε θα σας δυσκολέψουν ιδιαίτερα.



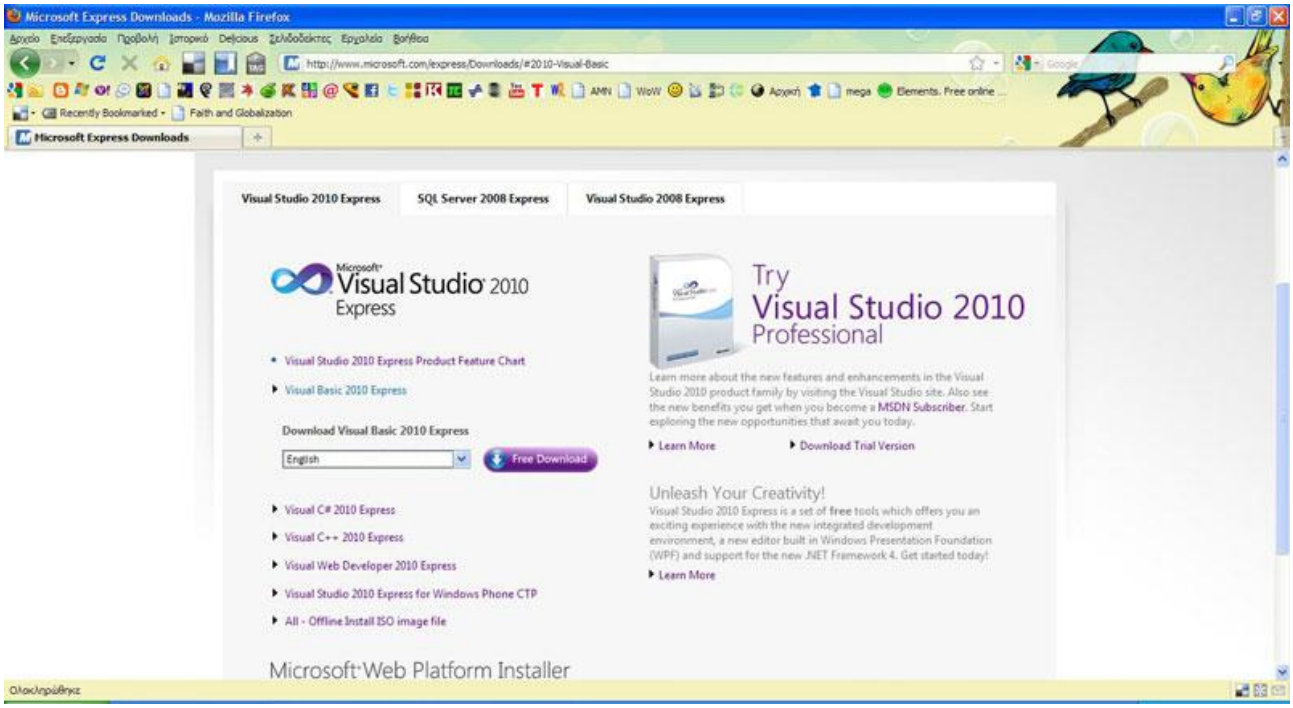
Η πρώτη επιλογή στο παράδειγμά μας, είναι και αυτή που θα μας πάει κατ' ευθείαν στη Visual Basic .NET 2010! Πατήσαμε κλίκ στο σύνδεσμο και μας πήγε στη σχετική ιστοσελίδα που βλέπετε παρακάτω. Παρατηρείτε τι λέει το βήμα 2 στην ιστοσελίδα;



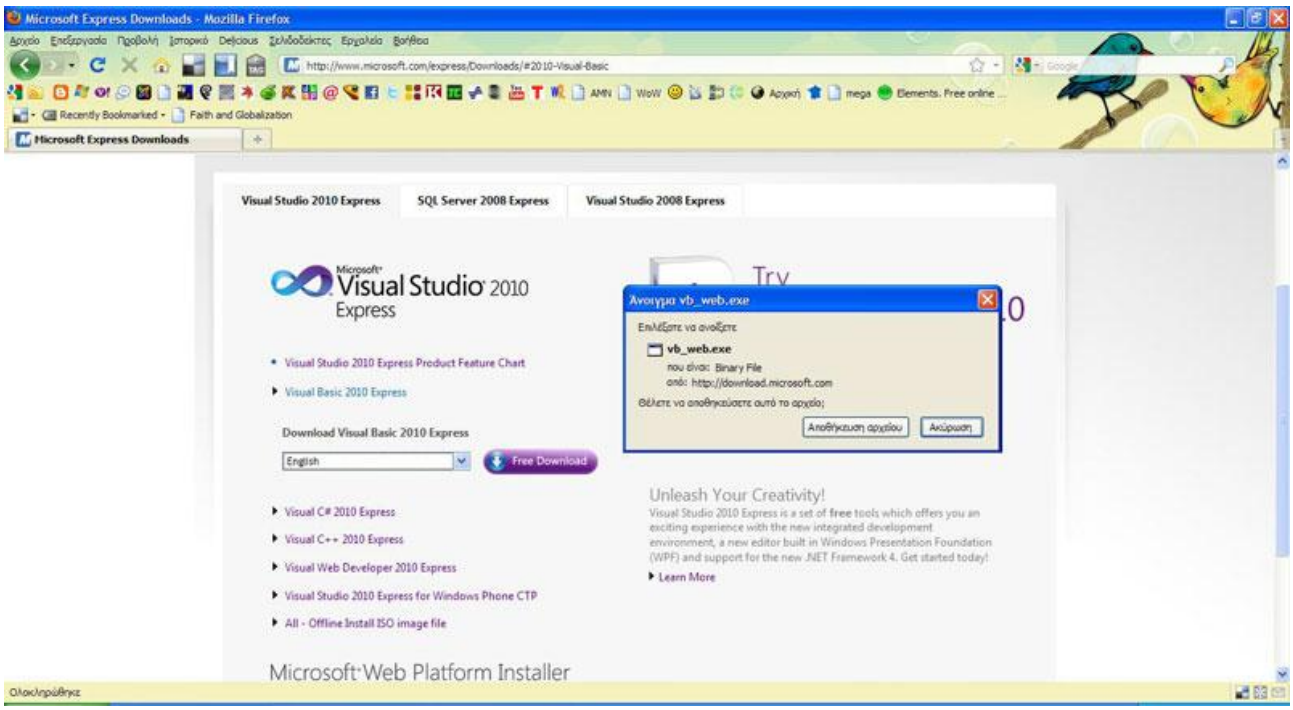
Κάναμε κλικ εκεί και μας πήγε στη σελίδα που μπορούμε να κατεβάσουμε διάφορα πακέτα λογισμικού. Η δεύτερη επιλογή όπως φαίνεται στην παρακάτω εικόνα είναι και η επιλογή με την οποία, κατεβάσαμε τη Visual Basic .NET 2010 EXPRESS.



Όπως βλέπετε στην παρακάτω εικόνα, επιλέγοντάς την, κι επιλέγοντας ως γλώσσα την Αγγλική εμφανίζεται ένα κουμπί,το οποίο γράφει "Free Download". Το πατήσαμε για να παραλάβουμε το λογισμικό.

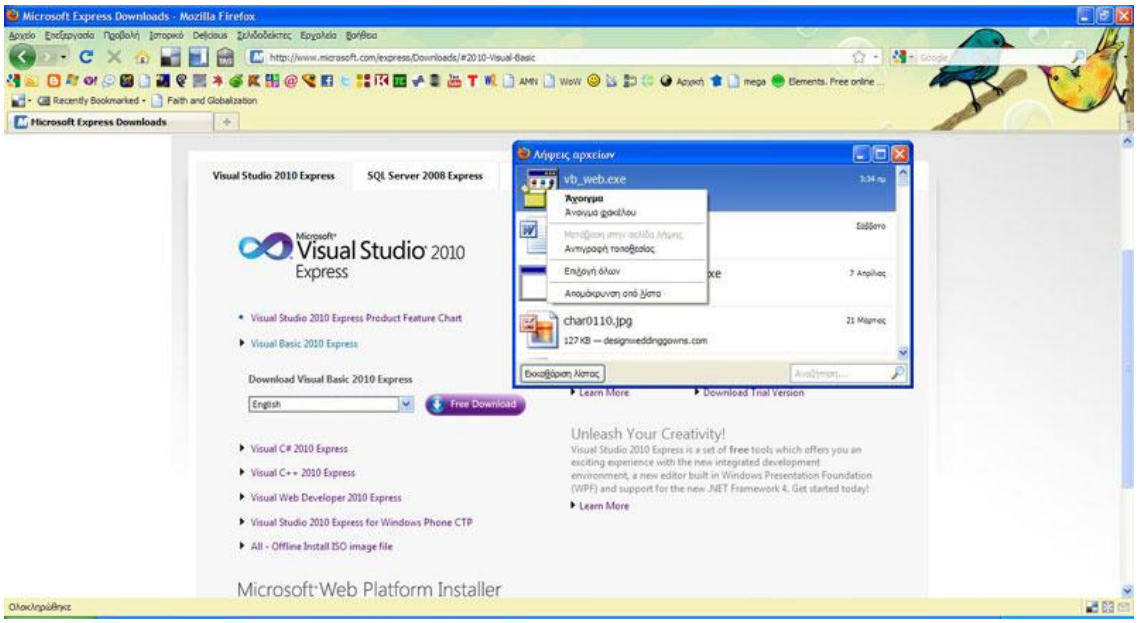


Εμφανίζεται τότε, ένα μικρότερο παράθυρο με δύο επιλογές. Επιλέγουμε "Αποθήκευση αρχείου".

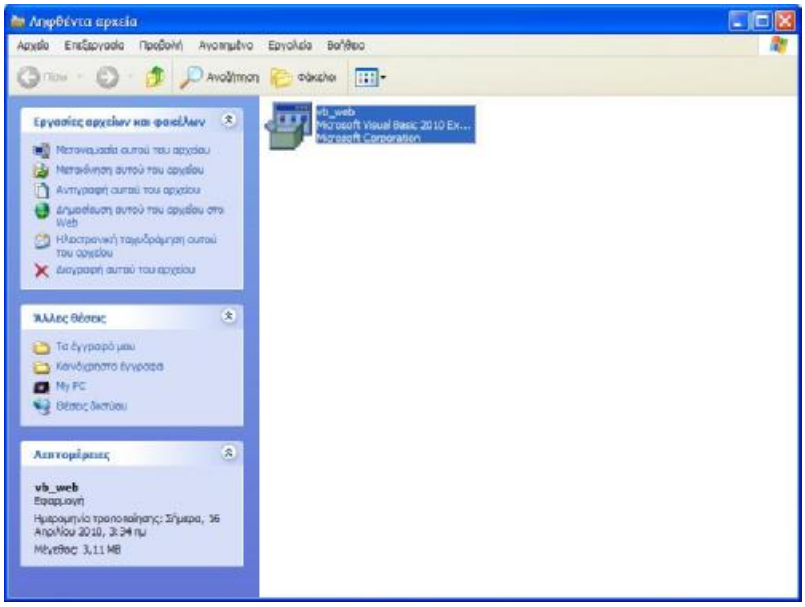


Αφού κατέβει, το εκτελούμε κάνοντας δεξί κλικ πάνω του κι επιλέγοντας "Ανοιγμα" από το μενού. Αυτός ο τρόπος, είναι ο ευκολότερος, μιας και το κάνουμε, όπως βλέπετε στην παρακάτω εικόνα από το παράθυρο "Λήψεις αρχείων" του Firefox. Εναλλακτικά, θα πρέπει

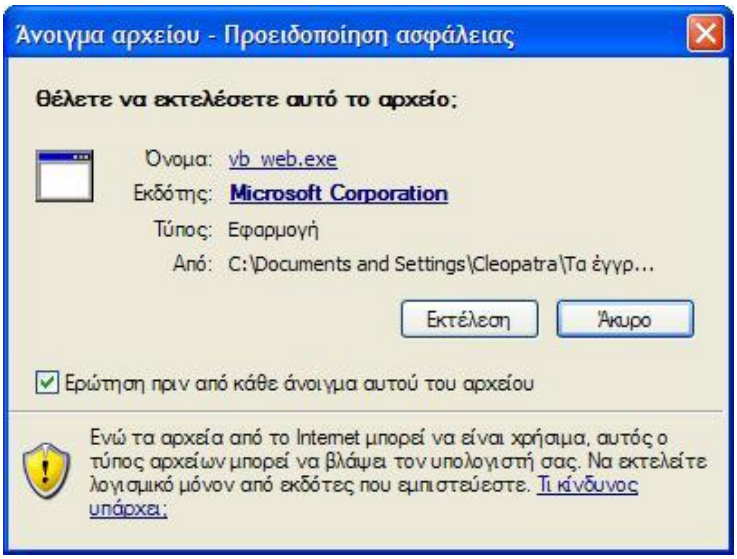
να πάμε στο φάκελο που τοποθετούνται τα αρχεία που κατεβάζουμε από το Intenet, και να κάνουμε διπλό κλίκ στο αρχείο "vb_web.exe" που κατεβάσαμε, για να εκτελεστεί. Αυτό γίνεται με δύο τρόπους. Είτε θα πάμε εκεί μέσω του Windows, ανοίγοντας το φάκελο που βρίσκεται...



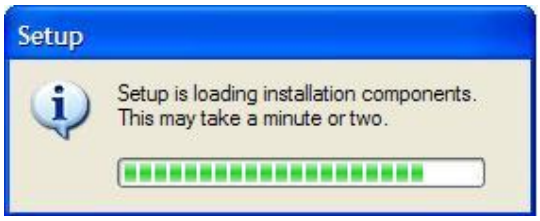
είτε πάλι από το παράθυρο του Firefox "Λήψεις αρχείων" δεξί κλικ πάνω στο αρχείο κι επιλέγουμε πολύ απλά "Άνοιγμα φακέλου". Ανοίγει ο φάκελος, όπως βλέπετε στην επόμενη εικόνα, βρίσκουμε εκεί το αρχείο και το τρέχουμε κάνοντας πάνω του διπλό κλικ.



Κι εδώ βλέπετε... ένα από τα πιθανά μηνύματα ασφαλείας που μπορεί να συναντήσετε κατά την εγκατάσταση! Γενικώς εμείς, εμπιστευτήκαμε το αρχείο εγκατάστασης και πατήσαμε "Εκτέλεση" Άλλωστε, από την επίσημη ιστοσελίδα το κατεβάσαμε. Δύσκολο να υπάρχει κάποιος κίνδυνος. ΠΡΟΣΕΧΤΕ ΟΜΩΣ ΑΝ ΚΑΤΕΒΑΣΕΤΕ ΑΠΟ ΑΛΛΟΥ ΤΟ ΑΡΧΕΙΟ! Κυκλοφορούν virus κι άλλοι κίνδυνοι.

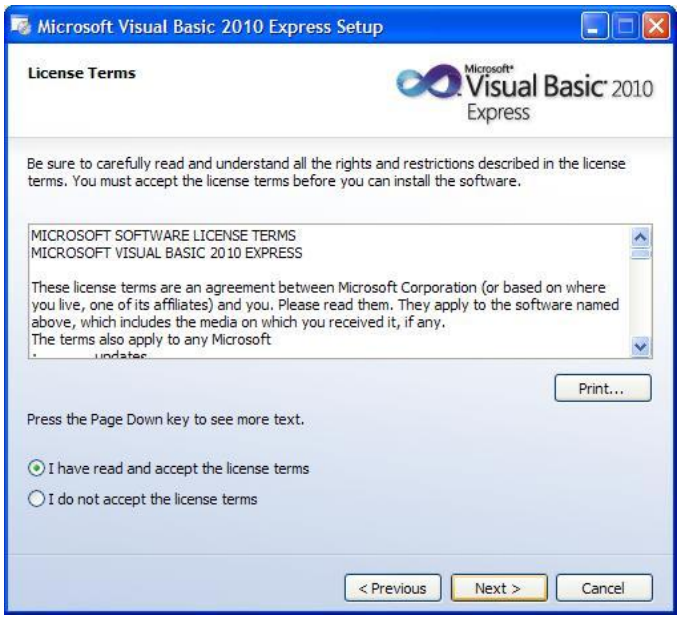


Ξεκινάει η εγκατάσταση. Εμφανίζεται το παρακάτω παράθυρο και πατάμε "Next".

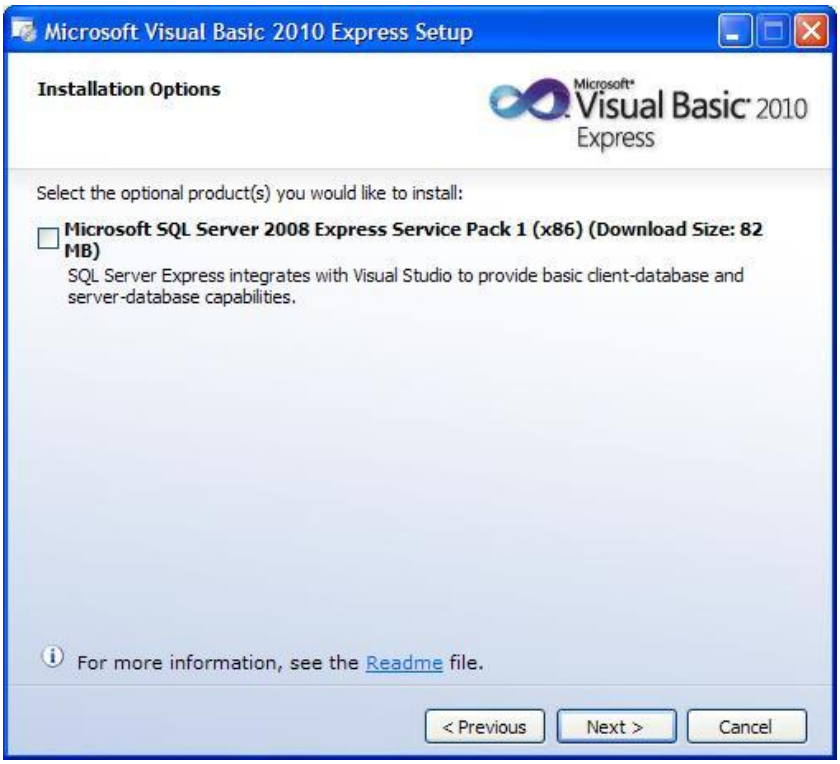


Εμείς επειδή θέλαμε να τελειώσει η εγκατάσταση το συντομότερο δυνατόν, δεν επιλέξαμε το τετραγωνάκι που μας επιτρέπει να στείλουμε πληροφορίες, για την εμπειρία μας, όσον αφορά την εγκατάσταση.

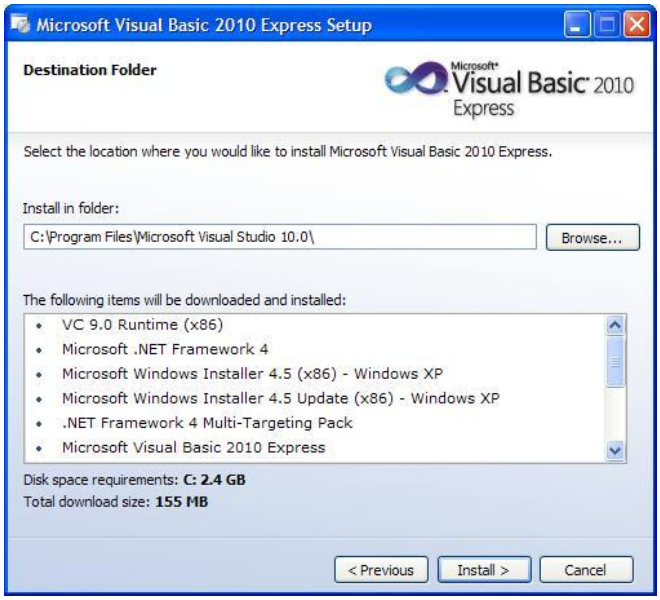
Διαβάσαμε την άδεια χρήσης και επιλέξαμε να την αποδεχτούμε επιλέγοντας: "I have read and accept the license term" και πατήσαμε "Next"



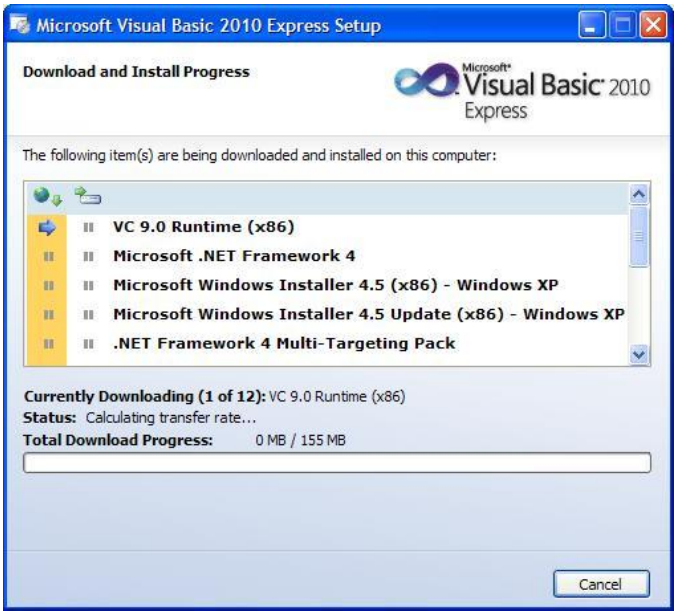
Κι εδώ έχετε την επιλογή να εγκαταστήσετε ή όχι, τον Microsoft SQL Server 2008. Εμείς δεν τον εγκαταστήσαμε, για να μη καταναλώσουε χώρο στο σκληρό δίσκο μας, αλλά κι επειδή η εγκατάσταση δε θα γίνει από τον υπολογιστή μας, αλλά μέσω Internet, οπότε δεν επιθυμούσαμε να επιβαρύνουμε επιπλέον την εγκατάσταση. Αλλωστε αυτο που θέλουμε είναι μονο τη γλώσσα προγραμματισμού.



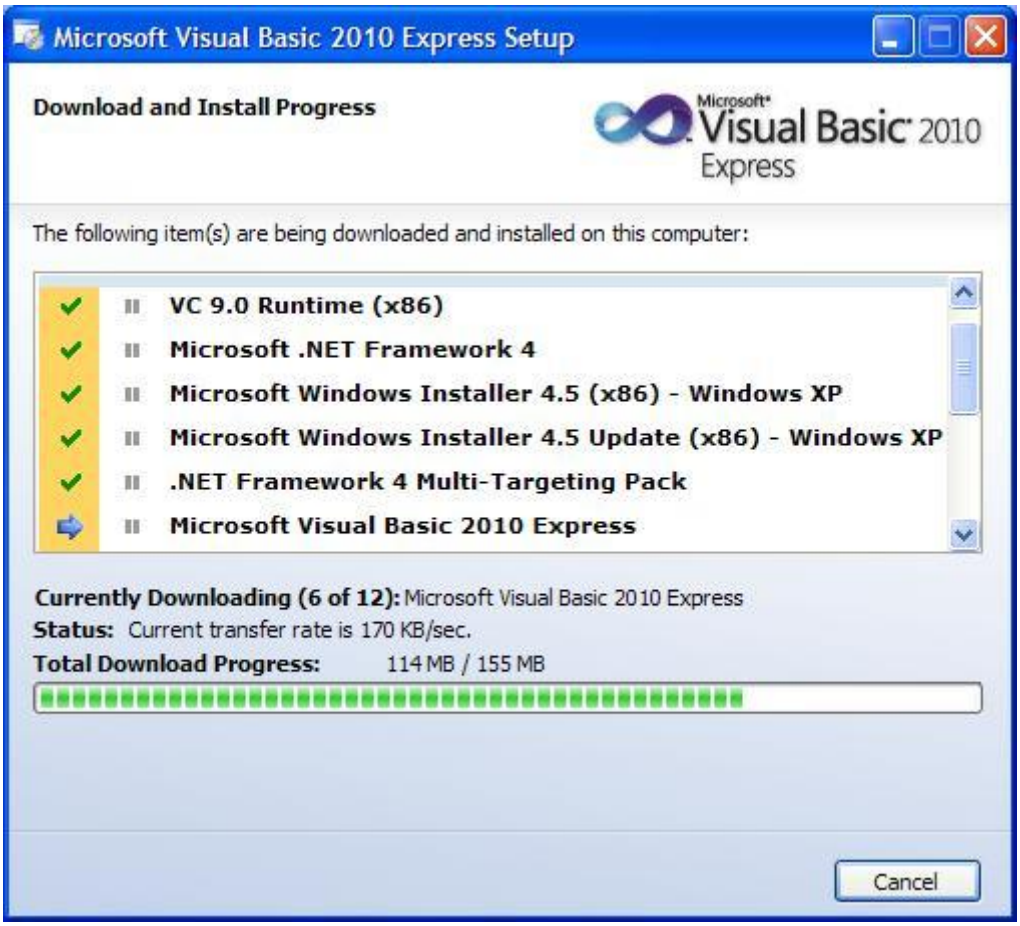
Όπως βλέπετε παρακάτω, στο παράθυρο που εμφανίζεται, εμφανίζονται δύο πληροφορίες. Θα κατεβάσετε από το Internet 155 MB αρχεία εγκατάστασης και θα χρειαστείτε 2.4 GB ελεύθερο χώρο στο σκληρό δίσκο. Πατήστε "Install".



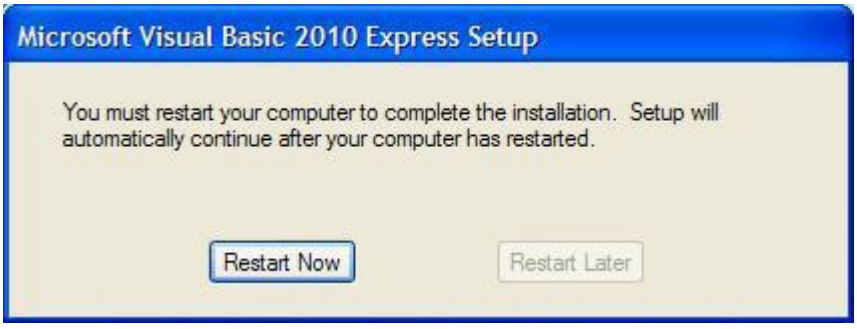
Αρχίζει μια μεγάλη εγκατάσταση. Θα αργήσει λίγο και ίσως να χρειαστεί να κάνετε αρκετές επανεκκινήσεις. Αν σε οποιαδήποτε περίπτωση, κάτι πάει στραβά,τρέξτε ξανά το αρχείο "vb_web.exe" και η εγκατάσταση θα συνεχιστεί κανονικά, σα να μην υπήρξε ποτέ το πρόβλημα. Αν μετά από επανεκκίνηση ΠΕΡΙΜΕΝΕΤΕ 5 λεπτά και δεν συνεχιστεί η εγκατάσταση, τρέξτε ξανά το αρχείο "vb_web.exe" και δεν θα έχετε κανένα πρόβλημα.



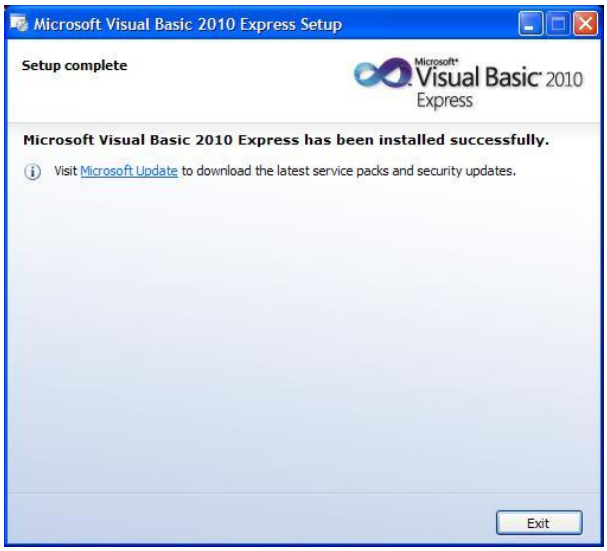
Έτσι, θα μοιάζει η εγκατάσταση, όταν θα φτάνει στο τέλος της. Αφού τα κατεβάσει, θα τα εγκαταστήσει αυτόματα, ή θα τα εγκαθιστα ένα ένα αφού κατεβαίνουν.



Αυτό το μήνυμα θα βγαίνει αν χρειάζονται επανεκκινήσεις.
Επιλέγετε "Restart Now".



Όταν τελειώσει η εγκατάσταση, θα εμφανιστεί το παρακάτω παράθυρο. Συγχαρητήρια.
Πατήστε "EXIT" και τελειώσατε την εγκατάσταση!

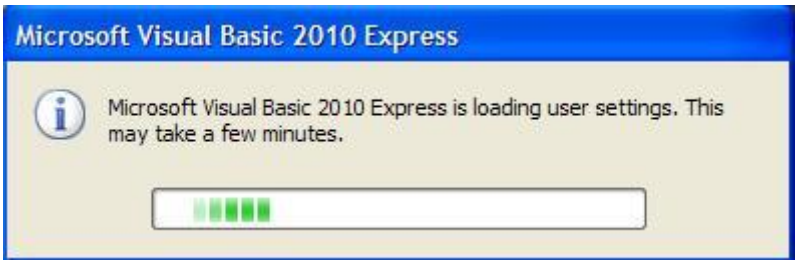


Καιρός να τρέξετε τη Visual Basic .NET 2010!Πηγαίνετε στο κουμπί "Έναρξη"....

..."Όλα τα προγράμματα..."

...κι επιλέγετε από το μενού όπως βλέπετε παρακάτω, τη Visual Basic 2010 Express.

Εμφανίζεται το παρακάτω μήνυμα και μετά φορτώνει η Visual Basic .NET 2010 EXPRESS



Τη Visual Basic .NET 2010 μπορείτε να τη χρησιμοποιήσετε για 30 μονο μέρες ΕΚΤΟΣ κι αν δηλώστετε στη Microsoft πως τη χρησιμοποιείτε. Στο Menu επιλέγετε το Help, και από εκεί επιλέγετε Register product. Συμπληρώνετε τα στοιχεία σας, στο e-mail σας έρχεται ένας κωδικός, τον περνάτε στο πρόγραμμα και είστε έτοιμοι.

Αν συναντήσατε πρόβλημα λόγω έλλειψης Service Pack ή .NET Framework πατήστε εδώ!

ΚΑΙ ΠΑΛΙ ΣΥΓΧΑΡΗΤΗΡΙΑ!

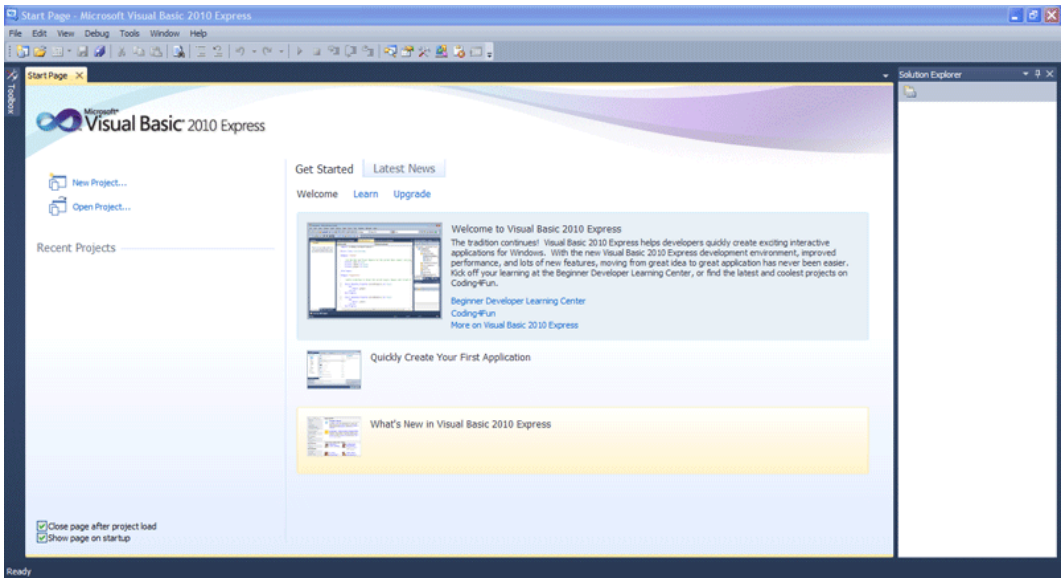
ΕΙΣΑΣΤΕ ΕΤΟΙΜΟΙ ΝΑ ΠΡΟΧΩΡΗΣΕΤΕ ΣΤΟ ΕΠΟΜΕΝΟ ΚΕΦΑΛΑΙΟ!

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

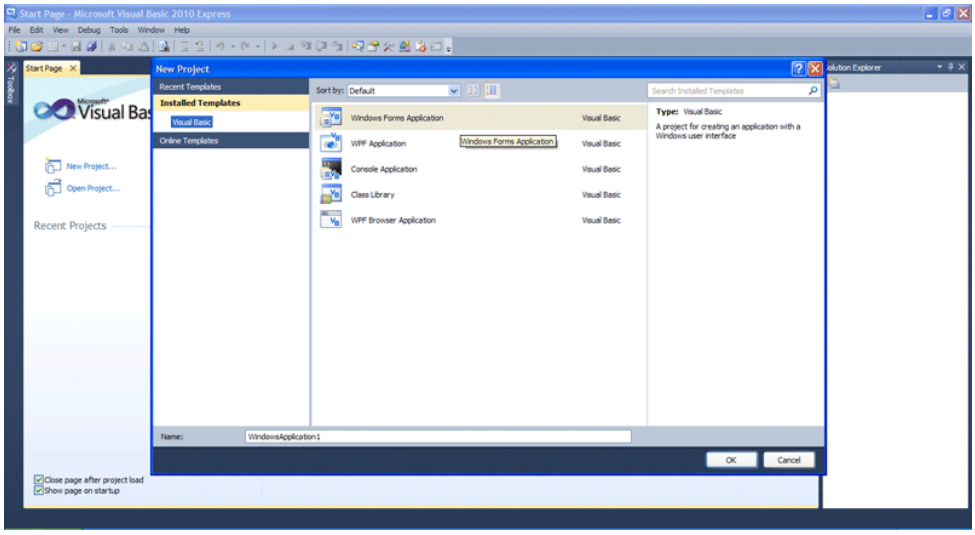
Το κεφάλαιο αυτό στόχο έχει την εκτενή υποστήριξη του μαθητευόμενου, βήμα βήμα στην εγκατάσταση του λογισμικού του προγραμματιστικού περιβάλλοντος που θα χρησιμοποιήσουμε. Στόχο έχει να καλύψει κάθε πιθανό πρόβλημα που μπορεί να παρουσιαστεί και παράλληλα δίνονται και υπερσυνδέσμους λύσεις στα ιδιαίτερα προβλήματα που μπορεί να αντιμετωπίσει ένας χρήστης αν του λείπει κάποιο service pack ή .Net Framework.

ΚΕΦΑΛΑΙΟ 4
ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΤΗΣ VISUAL BASIC .NET 2010 EXPRESS
ΜΕΡΟΣ ΠΡΩΤΟ

Με το που εκτελέσετε τη Visual Basic 2010 Express, θα σας εμφανίσει το παρακάτω μενού. Παρατηρήστε πάνω αριστερά τις δύο επιλογές που σας δίνει. New Project κι Open Project. Το πρώτο το επιλέγουμε όταν θέλουμε να δημιουργήσουμε ένα νέο πρόγραμμα, το δεύτερο αν θέλουμε να ανοίξουμε ένα που ήδη έχουμε ξεκινήσει να γράφουμε στο παρελθόν. Εσείς επιλέξτε το New Project.



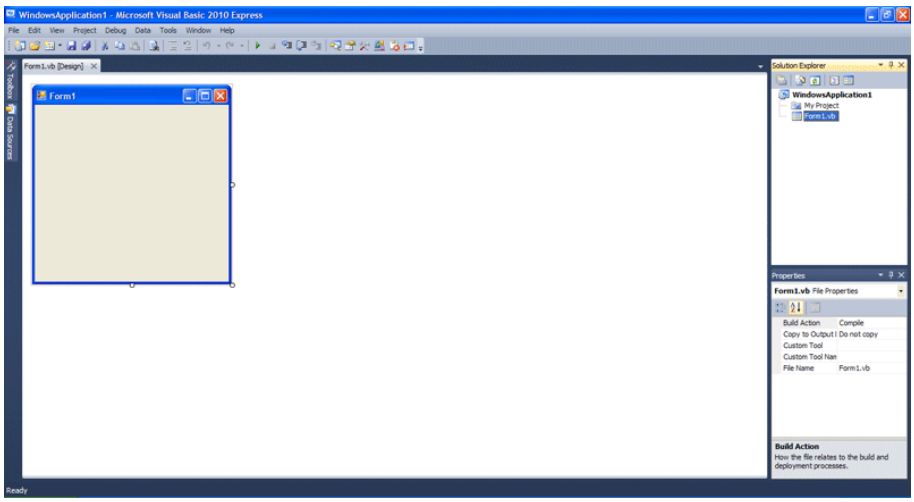
Αναδύεται ένα νέο παράθυρο, στο οποίο πρέπει να επιλέξετε τι είδους έργο θέλετε να δημιουργήσετε. Επιλέξτε Windows Forms Application. Θα φτιάξουμε ουσιαστικά παραθυρική εφαρμογή. Τις υπόλοιπες επιλογές μπορείτε να τις μελετήσετε στο μέλλον, αν θέλετε να ασχοληθείτε αρκετά και σε βάθος με αυτή τη γλώσσα. Στα πρώτα βήματά σας όμως, μάλλον πρέπει να ξεκινήσετε με απλούστερα πράγματα. Οπότε, εδώ επιλέγουμε το Windows Forms Application, για να γνωρίσουμε το περιβάλλον που θα εργασθούμε.



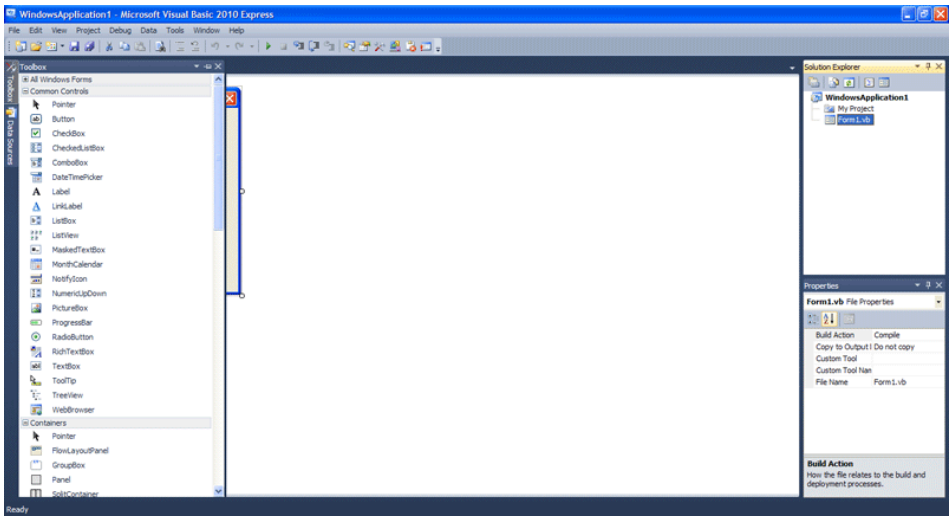
Το περιβάλλον της Visual Basic 2010 Express είναι αρκετά συμμαζεμένο και άνετο αν εξοικειωθείτε μαζί του. Ο μεγάλος λευκός καμβάς, που περιέχει μία φόρμα με όνομα Form1, είναι το περιβάλλον γραφικής σχεδίασης του προγράμματός μας. Εκεί φτιάχνουμε ουσιαστικά το τι θα βλέπει ο χρήστης, όταν τρέχει το πρόγραμμά μας. Η φόρμα είναι ένα βασικό αντικείμενο, στο οποίο τοποθετούνται πάνω άλλα αντικείμενα. Φανταστείτε τη σαν τα θεμέλια του γραφικού περιβάλλοντος του προγράμματός σας. Κοιτάξτε δεξιά τώρα. Βλέπουμε δύο μενού, που μας βοηθούν σε διάφορες διεργασίες, όπως για παράδειγμα να αλλάξουμε τις αρχικές ιδιότητες ενός αντικειμένου.

Έχουμε ήδη αναφέρει δύο φορές, μία πολύ σπουδαία έννοια, το αντικείμενο. Πρέπει απαραιτήτως να εξηγήσουμε τι είναι ένα αντικείμενο λοιπόν. Άλλωστε με αντικειμενοστρεφή προγραμματισμό ασχολούμαστε. Αντικείμενο είναι μία οντότητα, με συγκεκριμένες ιδιότητες, η οποία μπορεί να κάνει συγκεκριμένες πράξεις. Τις πράξεις τις αποκαλούμε μεθόδους. Για να γίνουμε πιο κατανοητοί, η φόρμα με όνομα Form1 είναι ένα αντικείμενο. Μερικές από τις ιδιότητές της, είναι το όνομά της και οι διαστάσεις της (δηλαδή το μέγεθός της σε μήκος και πλάτος). Η φόρμα ανοίγει (εμφανίζεται) και κλείνει (εξαφανίζεται). Αυτές οι δύο πράξεις, είναι μέθοδοί της.

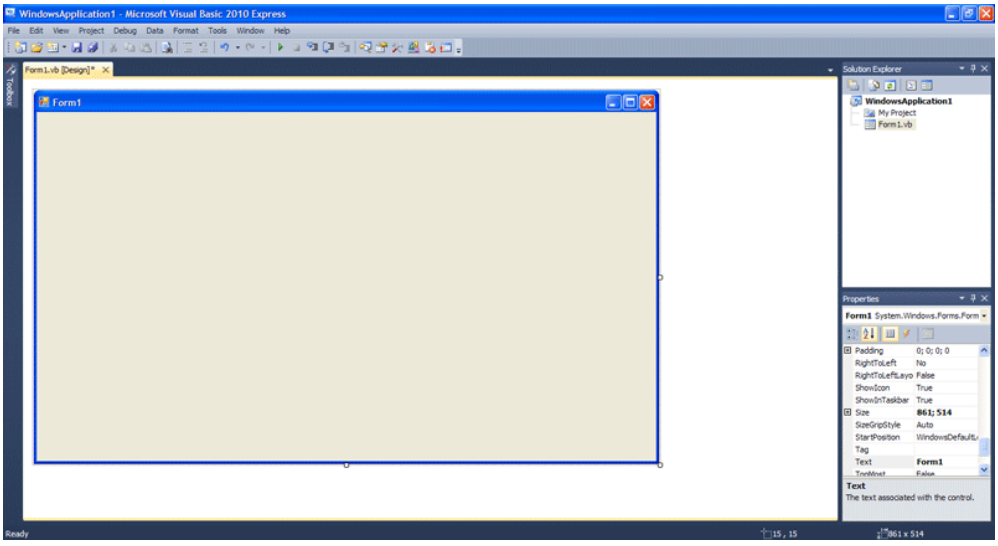
Ας αφήσομε για λίγο όμως την έννοια αυτή, θα την καταλάβετε ευκολότερα στην πράξη, στη συνέχεια.



Πάνω αριστερά, παρατηρούμε δύο κάθετα κουμπιά. Το Data Source και το Toolbox. Πατήστε το Toolbox. Αναδύεται ένα παράθυρο με πάρα πολλές επιλογές. Μη σας ανησυχεί. Προς το παρόν δείτε τι περιέχει.

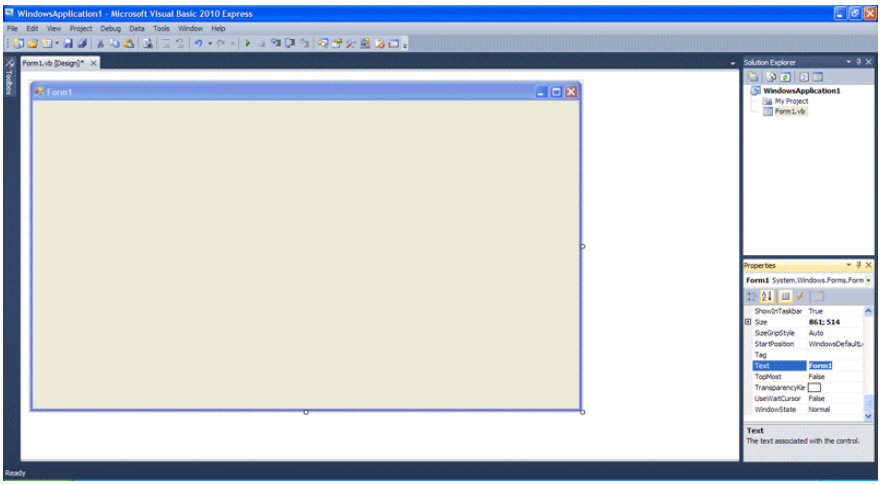


Αφήστε το Toolbox και κοιτάξτε την φόρμα Form1 που αυτόματα δημιούργησε η Visial Basic. Παρατηρήστε στη μέση του κάτω μέρους, στη μέση του δεξιού μέρους, αλλά και στην κάτω δεξιά γωνία, τα λευκά μικρά τετράγωνα. Πατήστε το αριστερό πλήκτρο του ποντικιού, έχοντας τον κέρσορα στο τετράγωνο της κάτω δεξιάς γωνίας και κρατώντας το πατημένο, σύρτε το ποντίκι προς τα κάτω και δεξιά. Έτσι αυξάνετε το μέγεθος της φόρμας, σε μήκος και πλάτος. Στην επάνω εικόνα φαίνονται, τα δεξιά λευκά τετράγωνα. Στη κάτω εικόνα φαίνονται και τα τρία, μικρά, λευκά τετράγωνα, που μας βοήθησαν να μεγεθύνουμε τη φόρμα.

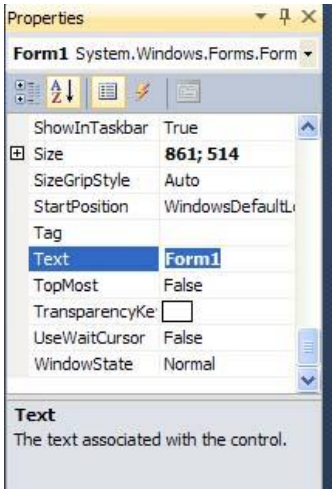


Έχοντας κάνει κλικ πάνω στη φόρμα, αλλάζει το δεξί, κάτω μενού σε (Properties). Επειδή έχετε κάνει κλικ στο αντικείμενο Form1, αυτή τη στιγμή δείχνει όλες τις ιδιότητες του αντικειμένου αυτού και σας δίνει τη δυνατότητα, να τους δώσετε αρχικές τιμές. Μπορείτε να αλλάξετε το μέγεθός της, το χρώμα της και διάφορες άλλες ιδιότητές της. Βρείτε την ιδιότητα Text. Θα έχει τιμή Form1. Η ιδιότητα αυτή στη φόρμα, της δίνει τον τίτλο της, στο πάνω αριστερό μπλε μέρος της. Προσοχή όμως! Δεν είναι το όνομά της. Το όνομα του αντικειμένου είναι η ιδιότητα Name και παραμένει Form1 εκτός κι αν του αλλάξουμε τιμή. Ψάξτε στο μενού αυτό, να βρείτε την ιδιότητα (Name). Αν και όλες οι ιδιότητες είναι στοιχισμένες στο μενού αλφαβητικά, η Name βρίσκεται σε παρένθεση, πάνω πάνω στο μενού, για να τη βρίσκετε εύκολα. Μην αλλάξετε την τιμή Form1 που έχει η Name. Απλώς κατανοήστε ότι ο τίτλος και το όνομα, είναι διαφορετικές μεταξύ τους ιδιότητες.
Σημείωση: Στα ονόματα (Name) των αντικειμένων ΔΕ μπορεί να χρησιμοποιηθεί ο

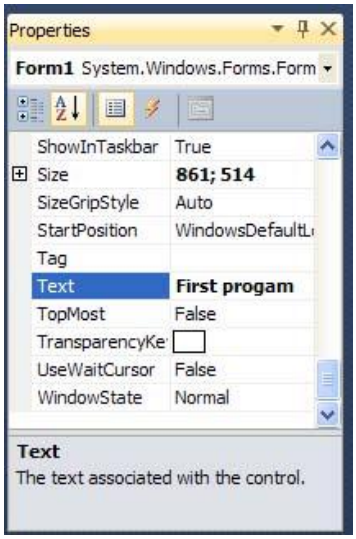
χαρακτήρας του κενού! Οπότε το όνομα "Form 1" δεν είναι έγκυρο ενώ τα "Form1" και "Form_1" είναι εντάξει.



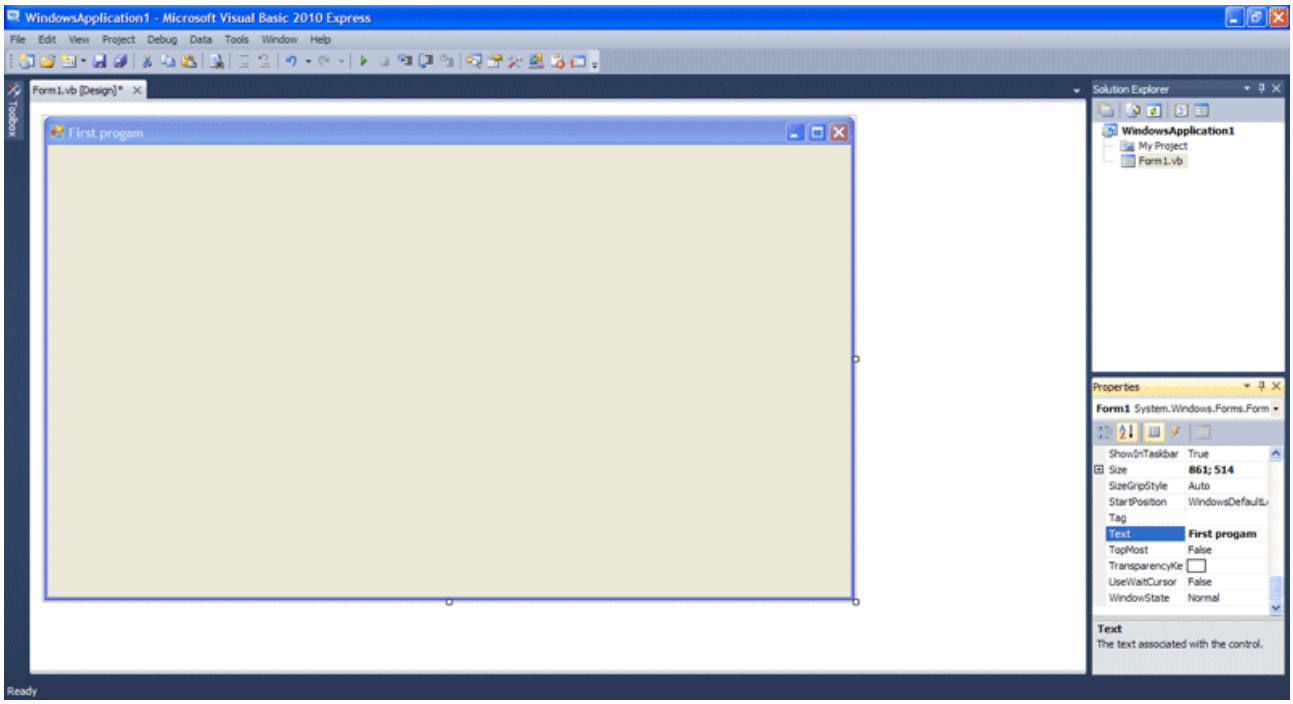
Ας αλλάξουμε τίτλο στη φόρμα. Σβήστε το Form1 από την ιδιότητα Text κι αντικαταστήστε το με τη φράση "First Program".



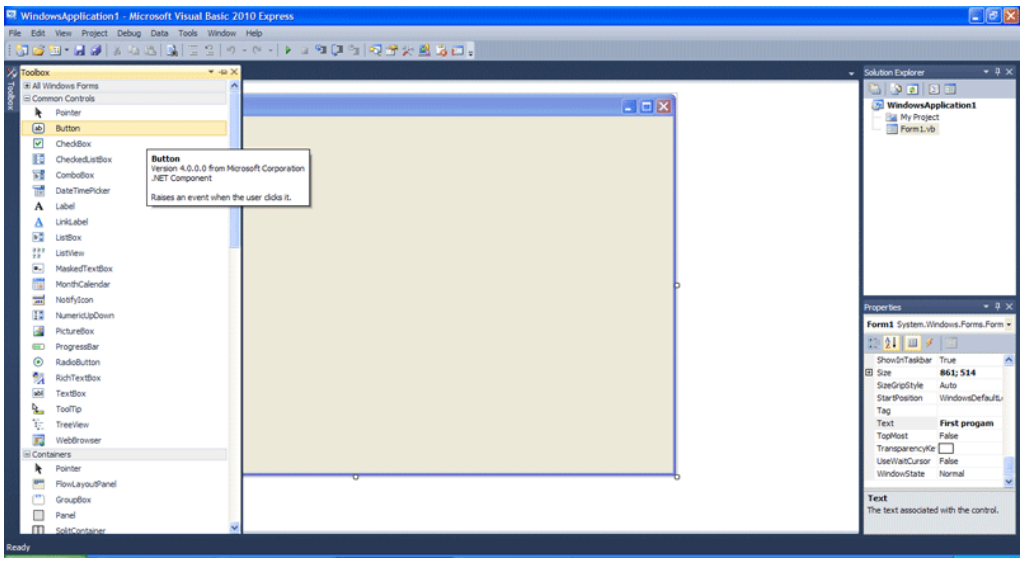
Έτσι πρέπει να φαίνεται το μενού μετά τις αλλαγές.



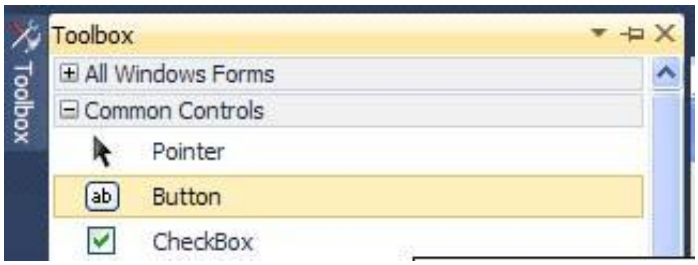
Παρατηρήστε τη φόρμα πάλι. Ο τίτλος της, πάνω αριστερά άλλαξε. Με αυτό το μενού λοιπόν, μπορούμε να ορίσουμε τις αρχικές τιμές που θα έχεί κάθε αντικείμενο. Αρχικές τιμές είναι οι τιμές που έχουν οι ιδιότητες των αντικειμένων με το που ξεκινάει το πρόγραμμα. Είναι ΠΑΝΤΑ καλό να δίνονται αρχικές τιμές σε ένα πρόγραμμα, στις ιδιότητες των αντικειμένων και στις μεταβλητές που θα χρησιμοποιήσετε (θα μιλήσουμε για μεταβλητές σε μεταγενέστερο κεφάλαιο). Το να δίνεις αρχικές τιμές, λέγεται αρχικοποίηση στον προγραμματισμό, initialization στα αγγλικά. Πολλά προγράμματα με το που ξεκινάνε, το πρώτο που κάνουν, είναι αρχικοποίηση και καλώς το κάνουν. Αν δε γίνει αυτό, ίσως προξενηθούν ανεπιθύμητα προβλήματα.



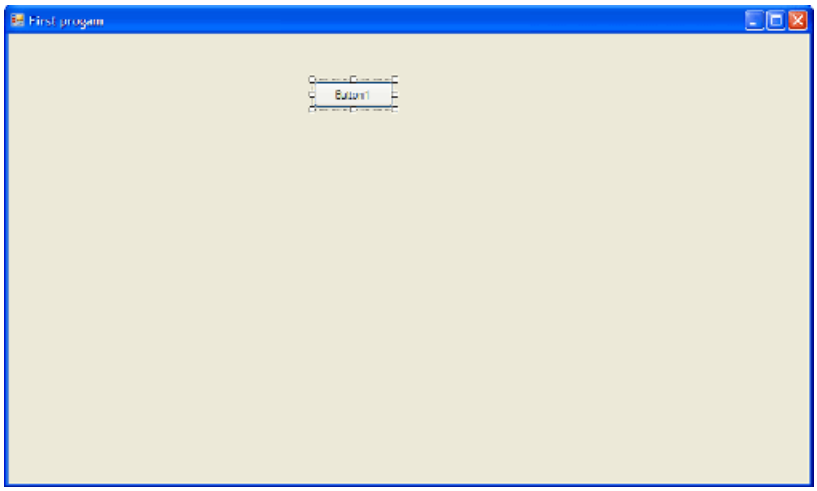
Ας φτιάξουμε ακόμα ένα αντικείμενο λοιπόν, το οποίο μάλιστα τοποθετείται μέσα στη φόρμα. Πηγαίνετε πάλι στο Toolbox, βρείτε κι επιλέξτε το αντικείμενο Button, πατώντας το αριστερό πλήκτρο του ποντικιού και κρατώντας το πατημένο, σύρετε το πάνω στη φόρμα. Αφήστε το κουμπί του ποντικιού. Μόλις φτιάξατε το αντικείμενο με όνομα και τίτλο Button1.



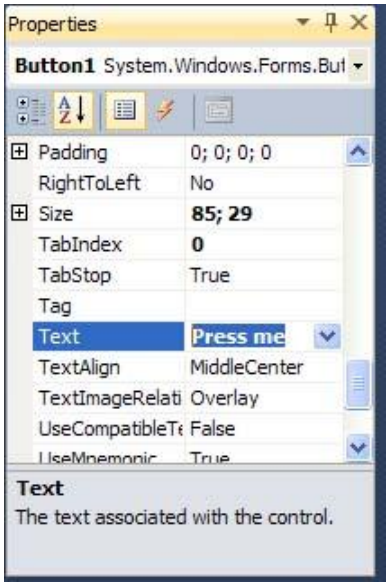
Το αντικείμενο Button βρίσκεται στο υπομενού Common controls του Toolbox.



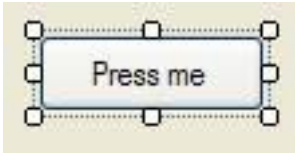
Κάπως έτσι θα μοιάζει η φόρμα σας μετά από αυτό.
Τι είναι όμως το κουμπί που μόλις φτιάξαμε;



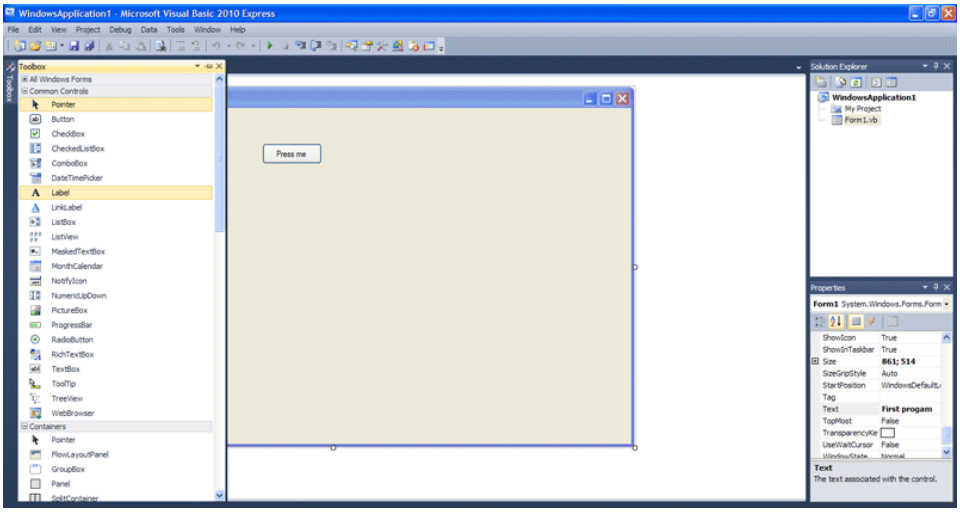
Είναι ένα αντικείμενο, ένα κουμπί που ο χρήστης θα μπορεί να το πατήσει κάνοντας κλικ πάνω του. Το κουμπί εκτελεί τον κώδικα που περιέχει κάθε φορά που το πατάμε. Ακόμα όμως δεν του έχουμε βάλει κώδικα. Για την ώρα, ας γράψουμε το μήνυμα "Press Me" αντί του "Button1" στον τίτλό του. Κάντε κλικ μία φορά πάνω στο Button1 και πηγαίνετε πάλι στο μενού κάτω δεξιά. (Properties) Βρείτε την ιδιότητα Text και αλλάξτε την τιμή της, από Button1 σε "Press me".



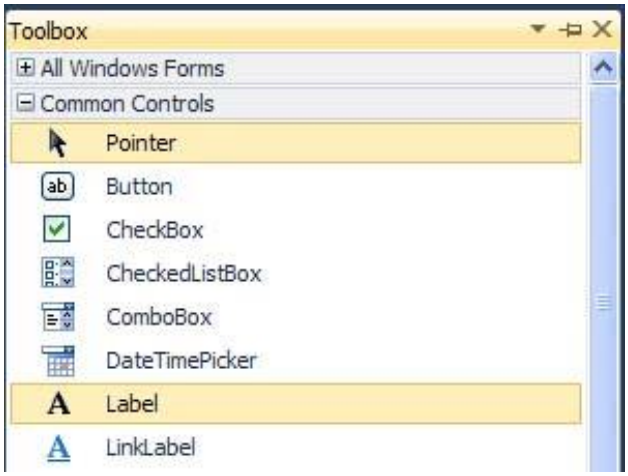
Το κουμπί άλλαξε τίτλο.



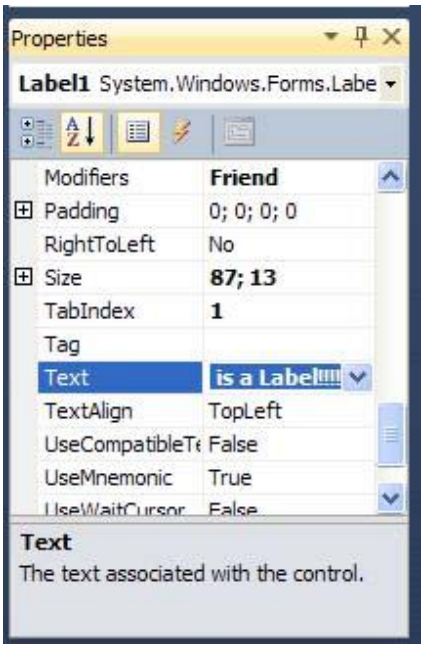
Ξαναπάτε στο Toolbox και βρείτε κι επιλέξτε μια Label. Είναι και αυτή ένα αντκείμενο. Επιλέξτε τη και βάλτε τη στη φόρμα, ακριβώς όπως κάνατε με το κουμπί, αλλά σε διαφορετικό μέρος της φόρμας για να φαίνεται. Κοινώς μη τη βάλετε πάνω στο κουμπί!



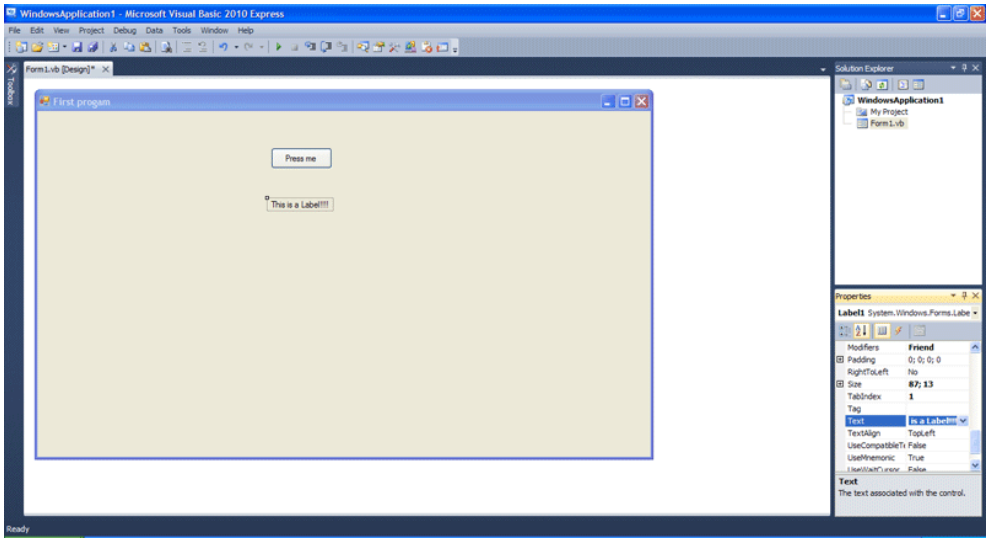
Δε θα δυσκολευτείτε να το βρείτε μιας και βρίσκεται κι αυτό μέσα στο υπομενού Common controls λίγο πιο κάτω από το Button.



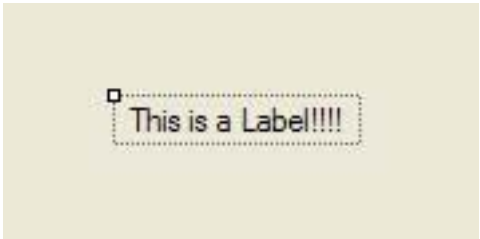
Κάντε κλικ πάνω της και πηγαίνετε στο μενού ιδιοτήτων (Properties). Στο Text βάλτε το κείμενο "This is a label!!!" Οκ, τελειώσαμε και με την αρχική τιμή της Label. Με το που ξεκινήσει το πρόγραμμα, η ετικέτα θα γράφει "This is a label!!!" θα αναρωτιέται τι κάνει βέβαια. Είναι μία ετικέτα. Απλώς εμφανίζει ένα κείμενο, και συνήθως χρησιμεύει στο να επεξηγεί κάτι που βρίσκεται στη φόρμα. Ωστόσο, εδώ θα τη χρησιμοποιήσουμε, για να σας βάλουμε μέσα στο κλίμα του προγραμματισμού με Visual Basic, όσο πιο απλά γίνεται.



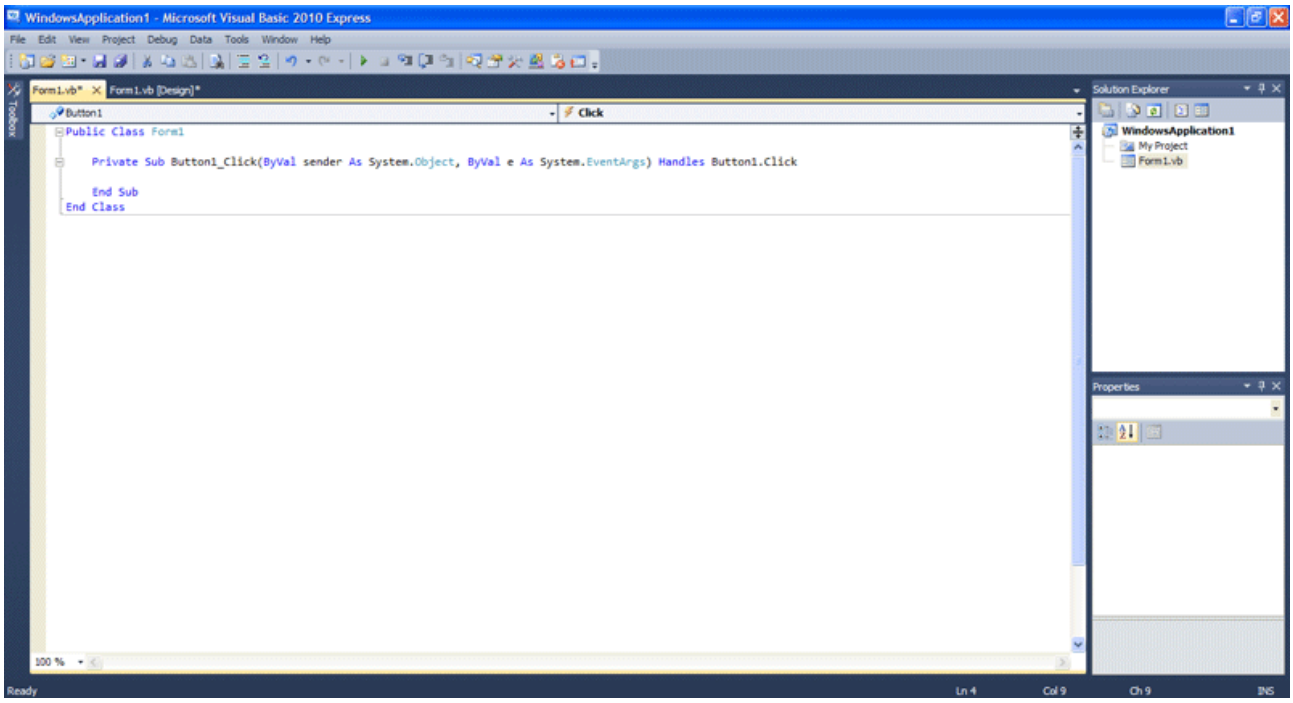
Κάπως έτσι πρέπει να είναι η φόρμα...



...και έτσι πρέπει να φαίνεται η Label1.



Τι κάνει όμως το Button1; Δεν ξεχάσαμε να σας πούμε. Το αφήσαμε για το τέλος.
Διότι θα είναι το πρώτο αντικείμενο, που θα εκτελέσει δικό σας κώδικα!
Κάντε ΔΙΠΛΟ κλικ πάνω στο Button1 (που γράφει πάνω του "Press me").
Εμφανίζεται το παρακάτω περιβάλλον.



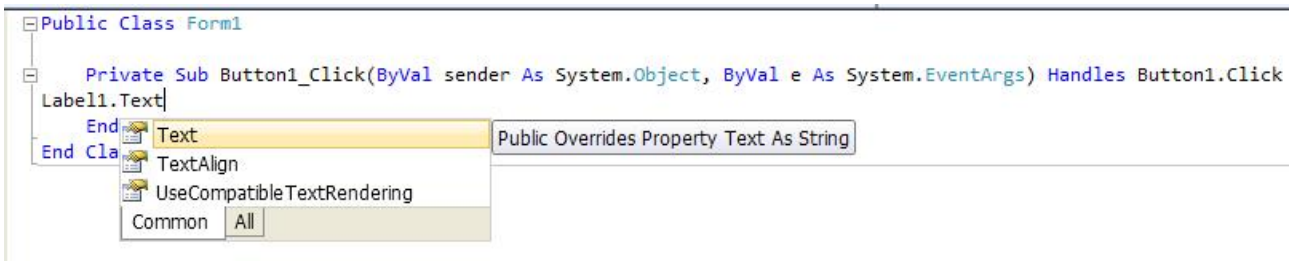
Μη σας τρομάζει. Τα μόνα που χρειάζεται να καταλάβετε από τον παρακάτω κώδικα θα σας τα εξηγήσουμε αμέσως. Κατ' αρχήν, όπως παρατηρείτε, κανένας δεν έγραψε αυτό τον κώδικα. Δημιουργήθηκε αυτόματα. Το "Public Class Form1" το οποίο κλείνει με το "End Class" δημιουργήθηκε, όταν δημιουργήσαμε το project διότι αυτόματα η Visual Basic έφτιαξε τη φόρμα με όνομα Form1. Μεταξύ των "Public Class Form1" και "End Class" εμπεριέχεται και κάτι ακόμα. Είναι το "Private Sub Button1_Click(ByVal e As System.EventArgs) Handles Button1.Click" που κλείνει με το "End Sub". Αυτό είναι η αυτόματη συγγραφή κώδικα για το Button1. Όταν δημιουργήσαμε το Button1, και του κάναμε διπλό κλικ, επειδή αυτή είναι πολύ συνηθισμένη μέθοδος, αυτόματα φτιάχτηκε αυτός ο κώδικας για να δώσουμε χρήση στο αντικείμενο. Το τοποθετήσαμε μέσα στη φόρμα Form1, γι' αυτό και βρίσκεται μέσα στην Class Form1.
Τι πρέπει να ξέρουμε εδώ;
Είπαμε πως αρχίζει ο κώδικας των αντικειμένων και πως τελειώνει. Στη Visual Basic λοιπόν, ο κώδικας ενός αντικειμένου, ξεκινάει (ανοίγει) με συγκεκριμένο τρόπο και επίσης τελειώνει (κλείνει), με συγκεκριμένο τρόπο. Ενδάμεσα στο άνοιγμα και το κλείσιμο, βρίσκεται ο κώδικας που θα εκτελεστεί. Οπότε το "Private Sub Button1_Click(ByVal e As System.EventArgs) Handles Button1.Click" αυτό που μας λέει είναι ότι το Button1 όταν του κάνουμε click (Button1_Click), θα εκτελέσει τον κώδικα που βρίσκεται μέσα του. Δεν έχουμε βάλει όμως τίποτα μέσα του, οπότε αν του κάνουμε κλικ δε θα κάνει απολύτως τίποτα!

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

    End Sub
End Class
```

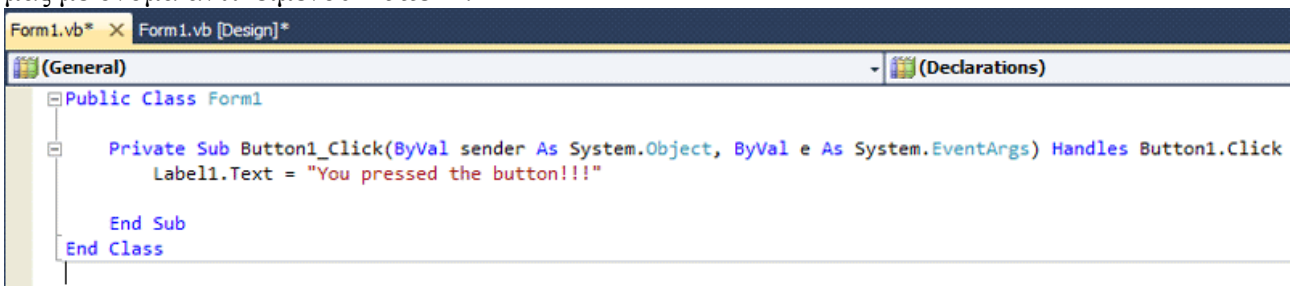
Καιρός να γράψετε την πρώτη σας γραμμή κώδικα σε Visual Basic! Πηγαίνετε τον κέρσορα μεταξύ ανοίγματος και κλεισίματος της Private Sub Button1_Click.
Γράψτε το παρακάτω:

Label1.Text = "You pressed the button!!!"

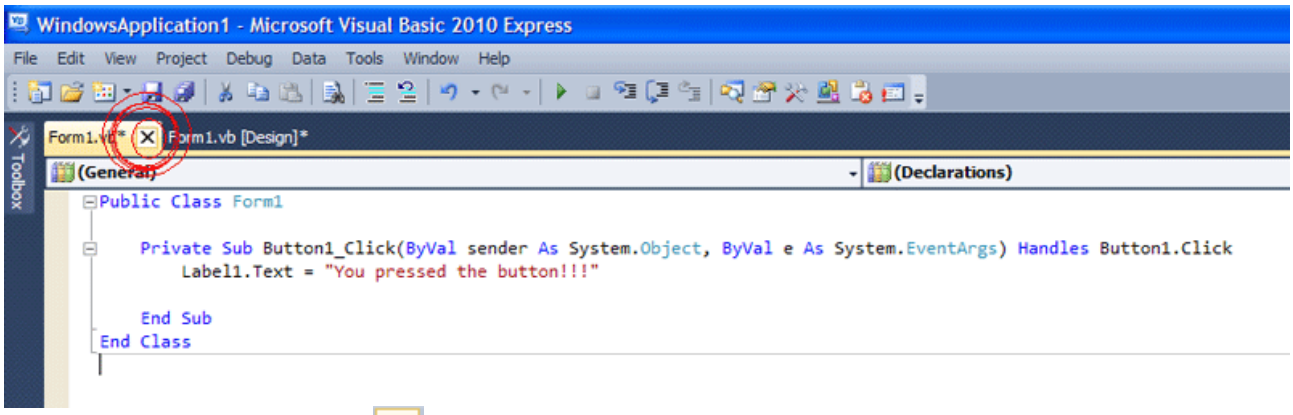
Παρατηρήσατε ότι το περιβάλλον της Visual Basic σας εμφάνισε διάφορες επιλογές καθώς το γράφατε. Η Visual Basic σας βοηθάει πολύ να γράφετε κώδικα. Όταν γράφετε Text..... η Visual Basic καταλαβαίνει στο περίπου τί μπορεί να θέλετε να γράψετε. Οπότε σας βγάζει επιλογές. Με τα βέλη (πάνω, κάτω) στο πληκτρολόγιο, ή με το ποντίκι, επιλέγετε την επιλογή που θέλετε και η Visual Basic τη γράφει με το που πατήσετε το space ή την τελεία (ανάλογα τι θα ακολουθεί στη σύνταξη). Εδώ, αν γράφετε "Label1", με το που πατήσετε την τελεία, θα σας εμφανίσει το "Text". Πατώντας spacebar θα το γράψει αυτόματα! Ωστόσο, όπως και στη φυσική γλώσσα, έτσι κι εδώ, όσο πιο πολύ κώδικα πληκτρολογείς μόνος, τόσο πιο καλός γίνεσαι. Αν όμως δε θυμάστε πώς ακριβώς λέγεται για παράδειγμα μια ιδιότητα, αυτή η δυνατότητα που έχει η Visual Basic σας λύνει τα χέρια, μιας και σας εμφανίζει ό,τι μπορεί να χρειάζεστε.



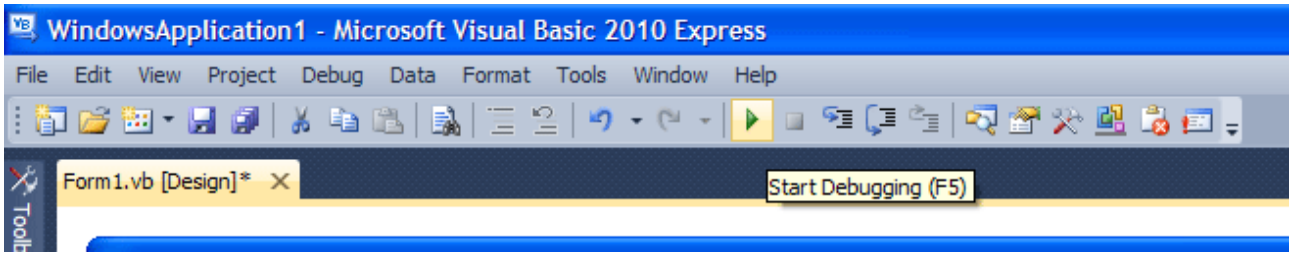
Τι κάνει λοιπόν αυτό που μόλις γράψαμε; Αυτή η εντολή λέει στον υπολογιστή: Το κείμενο της ετικέτας Label1 να γίνει "You pressed the button!!!" Πιο συγκεκριμένα η εντολή αυτή ακολουθεί μια σύνταξη βάσει του τύπου: όνομα_αντικεμένου.ιδιότητα_αντικειμένου = τιμή
Σημείωση: Οι κάτω παύλες δεν είναι συντακτικό. Χρησιμοποιούνται για να τονίσουμε, ότι τα ονόματα αυτά, (αντικειμένου, ιδιότητας) ΔΕ μπορούν να περιέχουν τον κενό χαρακτήρα. Με άλλα λόγια, ΔΕ γίνεται να είναι δύο λέξεις. Με την κάτω πάυλα όμως, παραμένουν μία λέξη. Είναι καλό τέχνασμα, αν θέλετε να δίνετε πιο περίπλοκα ονόματα στίς μεταβλητές σας. Τα ονόματα αυτά μπορεί να περιέχουν αριθμούς και γράμματα, αλλά και την κάτω παύλα "_" όμως δε μπορεί να περιέχεται κενό. Δηλαδή το "object one" είναι λάθος όνομα. Το "object_one" είναι σωστό. Όταν λοιπόν θέλουμε να πούμε:
Η ιδιότητα Α του αντικειμένου objectB, θα γράψουμε:
ObjectB.A
Άρα πιο ορθά, μπορούμε να πούμε, ότι η εντολή Label1.Text = "You pressed the button!!!" λέει στον υπολογιστή: Η ιδιότητα Text του αντικειμένου Label1, θα πάρει την τιμή "You pressed the button!!!"
Πότε θα εκτελεστεί;
Υπενθυμίζουμε: Private Sub Button1_Click! Θα εκτελεστεί όταν κάνουμε κλικ στο κουμπί μας με όνομα αντικειμένου Button1.



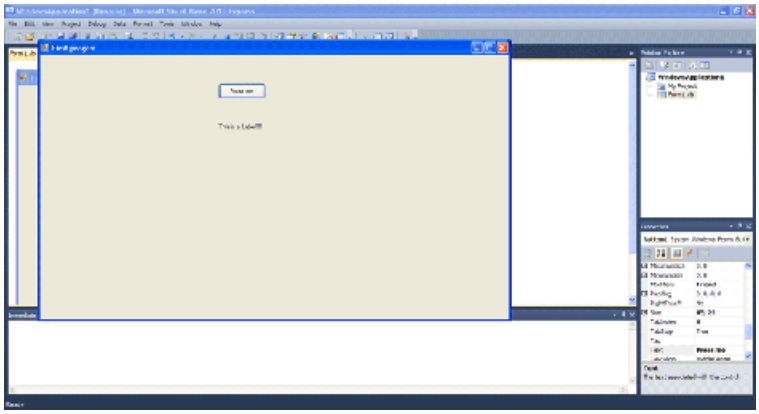
Μην ασχοληθείτε και μην αναρωτηθείτε, για τον υπόλοιπο κώδικα και παραμέτρους που βλέπετε. Τελειώσαμε εδώ. Πατήστε το "X" που έχουμε μαρκάρει με κόκκινους κύκλους στην επόμενη εικόνα, για να επιστρέψετε στη σχεδίαση του γραφικού περιβάλλοντος.



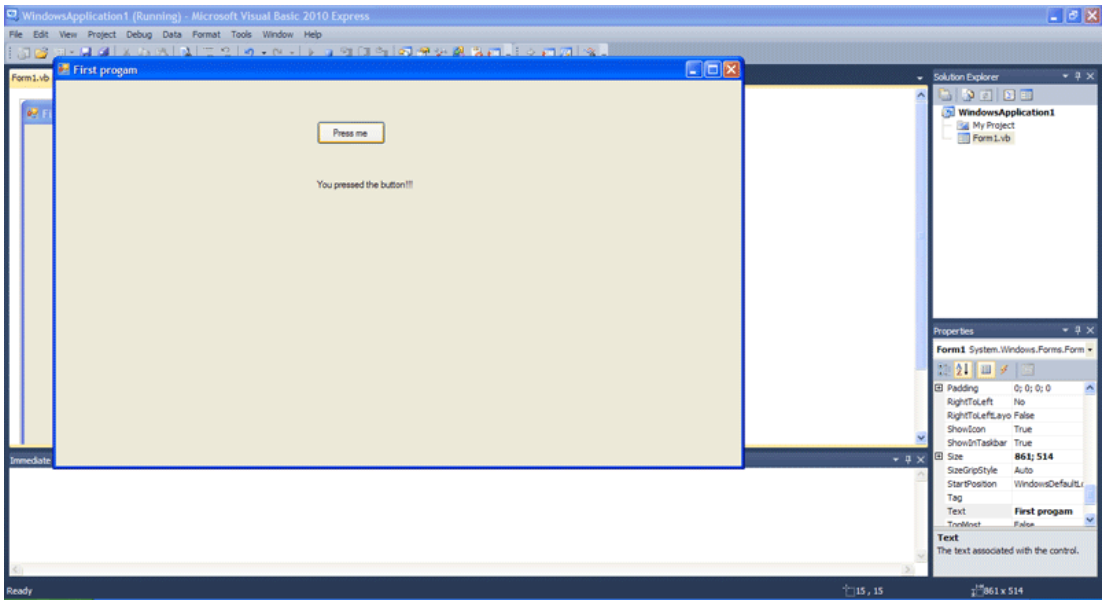
Πατήστε το πράσινο κουμπί  για να εκτελέσετε το πρόγραμμα!



Θα σας εμφανιστεί το πρόγραμμα μπροστά σας, στο δικό του παράθυρο, το οποίο είναι η φόρμα που σχεδιάσατε!
Διαβάστε την ετικέτα στο πρόγραμμά σας! Πατήστε το κουμπί που έχει το πρόγραμμά σας!



Βλέπετε; Άλλαξε το κείμενο στην ετικέτα! Τι κάναμε δηλαδή; Με ένα αντικείμενο, εκτελώντας μια πράξη πάνω του, το κλικ, αλλάξαμε την τιμή της ιδιότητας ενός ΑΛΛΟΥ αντικειμένου! Τώρα αρχίζετε να έχετε μια μικρή εικόνα για το τι θα κάνουμε και για το πως λειτουργεί η Visual Basic!
Κλείστε το πρόγραμμα που φτιάξατε πατώντας το κλασσικό κόκκινο "Χ" στο πάνω δεξί μέρος της φόρμας του.



Ένα τελευταίο σχόλιο. Παρατηρήσατε ότι το Button1 και η Form1 είχαν κώδικα, ενώ η Label1 όχι; Αυτό συμβαίνει γιατί η ετικέτα δεν εκτελεί κάτι. Δεν κάνει κάποια πράξη. Δε "φορτώνει", (δηλαδή ανοίγει κι εμφανίζεται), ούτε και κλείνει, όπως η Form1, ούτε της κάνεις click για να κάνει κάτι, όπως το Button1. Ωστόσο δεν παύει να είναι ένα αντικείμενο με δικές του ιδιότητες. Απλά δεν εκτελείται κώδικας σε αυτή. Μπορεί να εκτελεστεί κώδικας όμως, για την περίπτωση που, για παράδειγμα, το βελάκι του ποντικιού, βρεθεί στην επιφάνειά της. MouseEnter αντί Click θα ήταν το γεγονός κατά το οποίο θα εκτελείτο κώδικασε αυτή την περίπτωση.
Συγχαρητήρια. Μόλις φτιάξατε ένα απλό υποτυπώδες πρόγραμμα σε Visual Basic. Μπορεί να μην ήταν κάτι ιδιαίτερο, αλλά είναι το πρώτο σας πρόγραμμα. Σας προτείνουμε να κρατήσετε σημειώσεις και να μελετήσετε αυτό το κεφάλαιο. Ξεκουραστείτε και ΜΕΤΑ προχωρήστε στο επόμενο. Ήδη μάθατε σε αυτό το κεφάλαιο πολλά, αλλά βασικά κι απαραίτητα πράγματα.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Σε αυτό το κεφάλαιο παρουσιάστηκαν βασικές γνώσεις χρήσης του περιβάλλοντος της Visual Basic Express 2010, σε συνδυασμό με την προτροπή δημιουργίας ενός απλού προγράμματος, διότι είναι πολύ πιο εύκολο να αφομοιωθεί η γνώση, αν παράλληλα ο μαθητευόμενος κάνει πρακτική πάνω σε ότι μαθαίνει και επίσης δύσκολα γίνεται το μάθημα βαρετό.

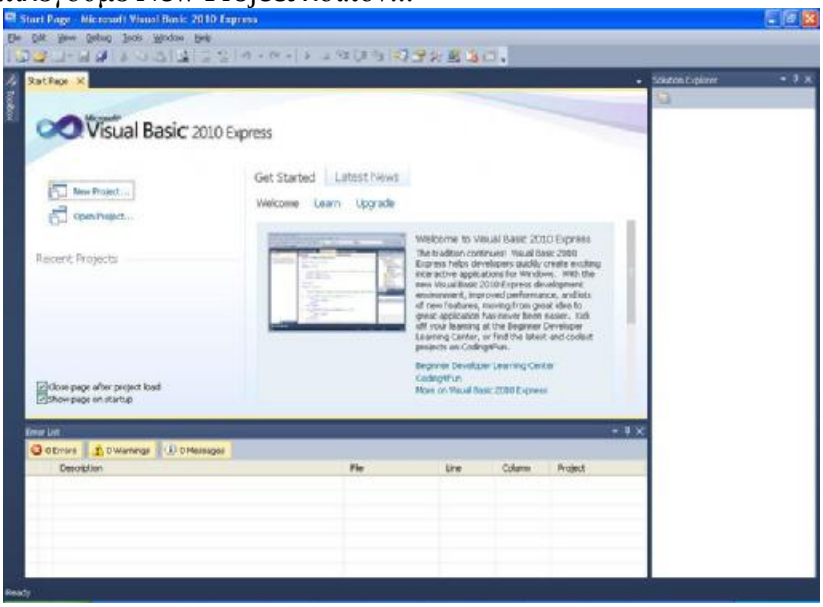
ΛΥΣΗ ΑΣΚΗΣΗΣ ΚΕΦΑΛΑΙΟΥ 4

Φτιάξτε ένα πρόγραμμα τύπου Windows Form Applicaton με 2 κουμπιά και 2 ετικέτες.
Το πρώτο κουμπί , το Button1 δηλαδή, πρέπει να κάνει το εξής:
Το text της Label1 να παίρνει τιμή "Γεια χαρά πάτησες το κουμπί 1"
Το text της Label2 να παίρνει τιμή "Το κουμπί 1 αλλάζει τιμή και στις δύο Labels!!"
Τα εισαγωγικά στην εκφώνηση εμπεριέχουν τις τιμές και δεν είναι μέρος τους.
Κοινώς δε θέλουμε να εμφανίζονται εισαγωγικά στην οθόνη.
Το Button2 θα πρέπει να αδειάζει το περιεχόμενο των Label1.text και Label2.text δηλαδή να καθαρίζει το περιεχόμενό τους. Κοινώς να μην εμφανίζεται κάτι στις Label1 και Label2

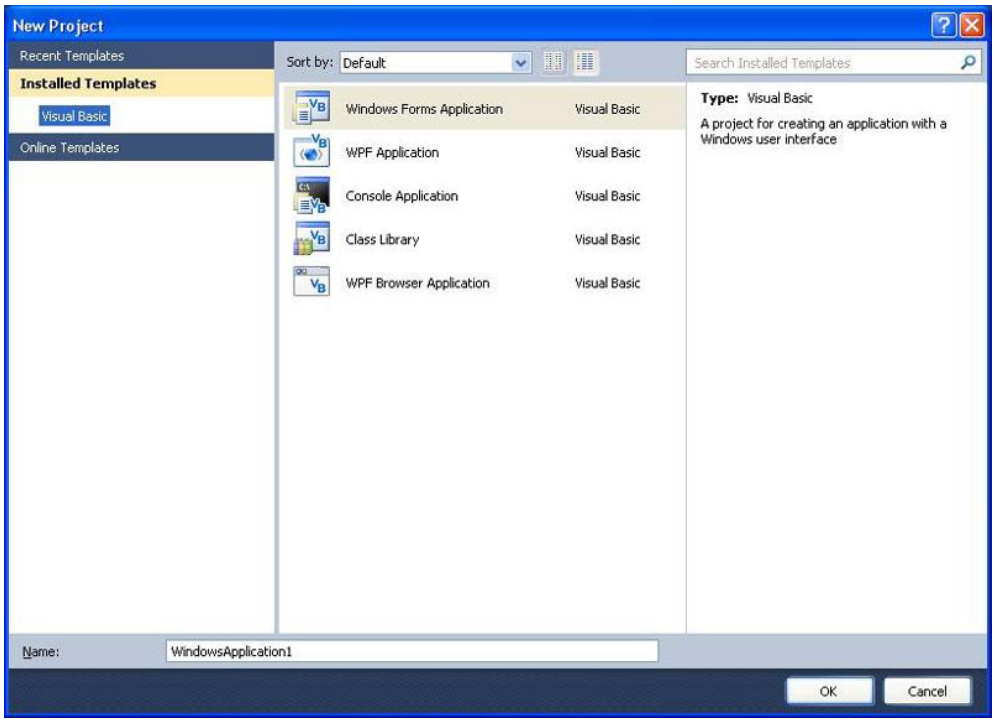
ΠΡΟΣΠΑΘΗΣΤΕ ΝΑ ΛΥΣΕΤΕ ΤΗΝ ΑΣΚΗΣΗ ΠΡΙΝ ΔΕΙΤΕ ΤΗ ΛΥΣΗ.

ΛΥΣΗ

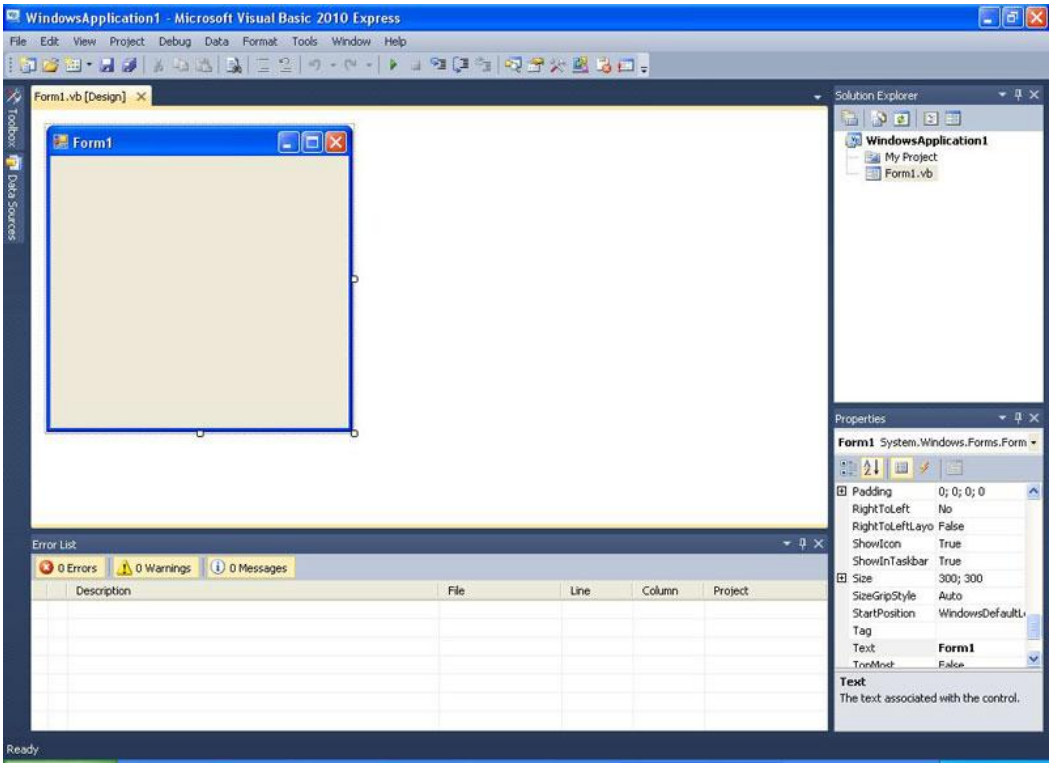
Η άσκηση αποτελείται από δύο προκλήσεις, που στόχο έχουν να σας κάνουν να σκεφτείτε με προγραμματιστικό τρόπο, μια λύση σε ένα πρόβλημα. Το πρόβλημα εδώ είναι σαφώς η εκφώνηση της άσκησης. Θα μπορούσαμε εντός του κεφαλαίου να σας δίναμε τη λύση και για τα δύο αυτά ερωτήματα, αλλά προτιμήσαμε να σας τα δώσουμε ως άσκηση, πρώτον διότι πρέπει να αρχίσετε να σκέφτεστε, όπως ένας προγραμματιστής πρέπει να σκέφτεται, δεύτερον διότι αν τη λύσετε χωρίς να δείτε τη λύση, θα λάβετε μεγάλη ικανοποίηση και τρίτον διότι αν τελικά δεν καταφέρετε να τη λύσετε μόνοι σας, και δείτε τη λύση μετά από αρκετό κόπο, θα είναι δύσκολο να ξεχάσετε τόσο τον τρόπο υλοποίησης αυτής της άσκησης, όσο και τον τρόπο σκέψης που θέλουμε να αναπτύξετε. Ποιες είναι λοιπόν οι δύο προκλήσεις της άσκησης; Τι μου λέει η εκφώνηση; "Φτιάξτε ένα πρόγραμμα τύπου Windows Form Applicaton". Επιλέγουμε New Project λοιπόν...



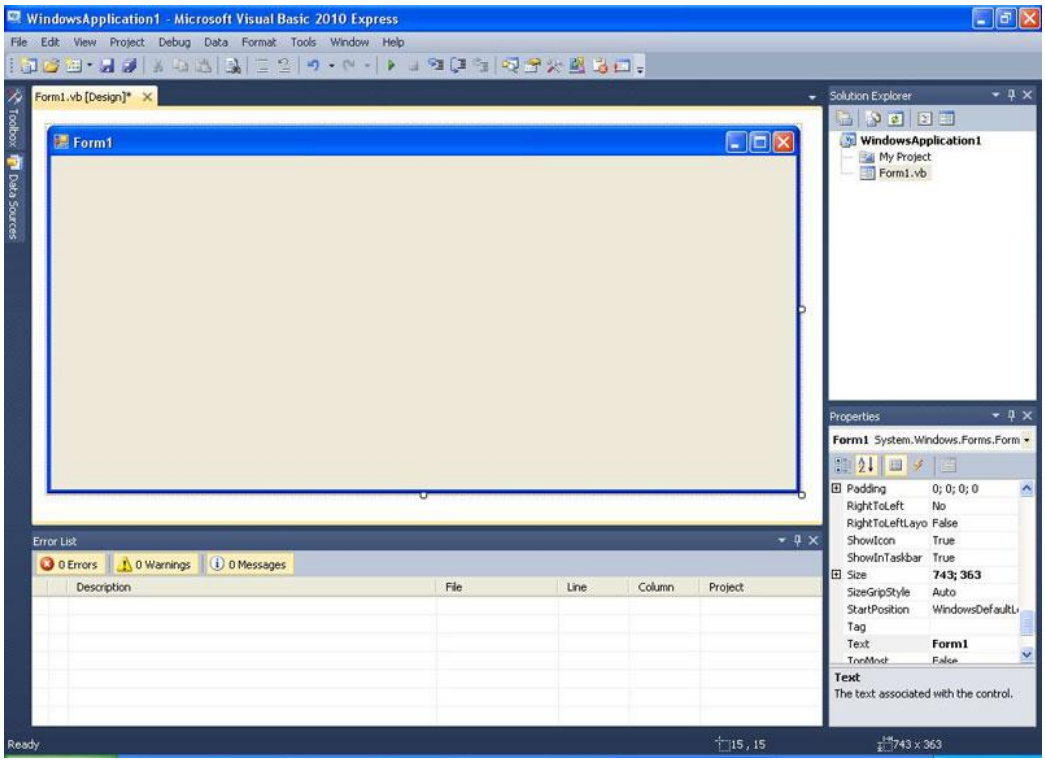
... κι επιλέγουμε Windows Forms Applicaton.



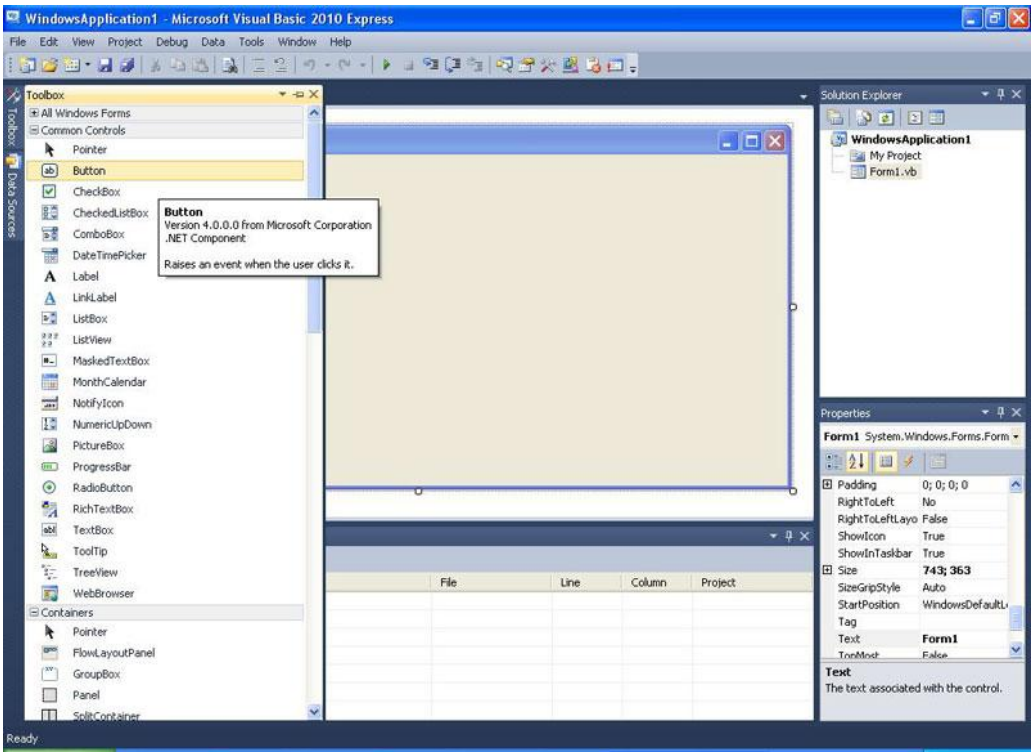
Ως εδώ καλά. Δεν κάναμε κάτι καινούργιο.
Πιάνουμε τη φόρμα και τη μεγαλώνουμε όσο θέλουμε.



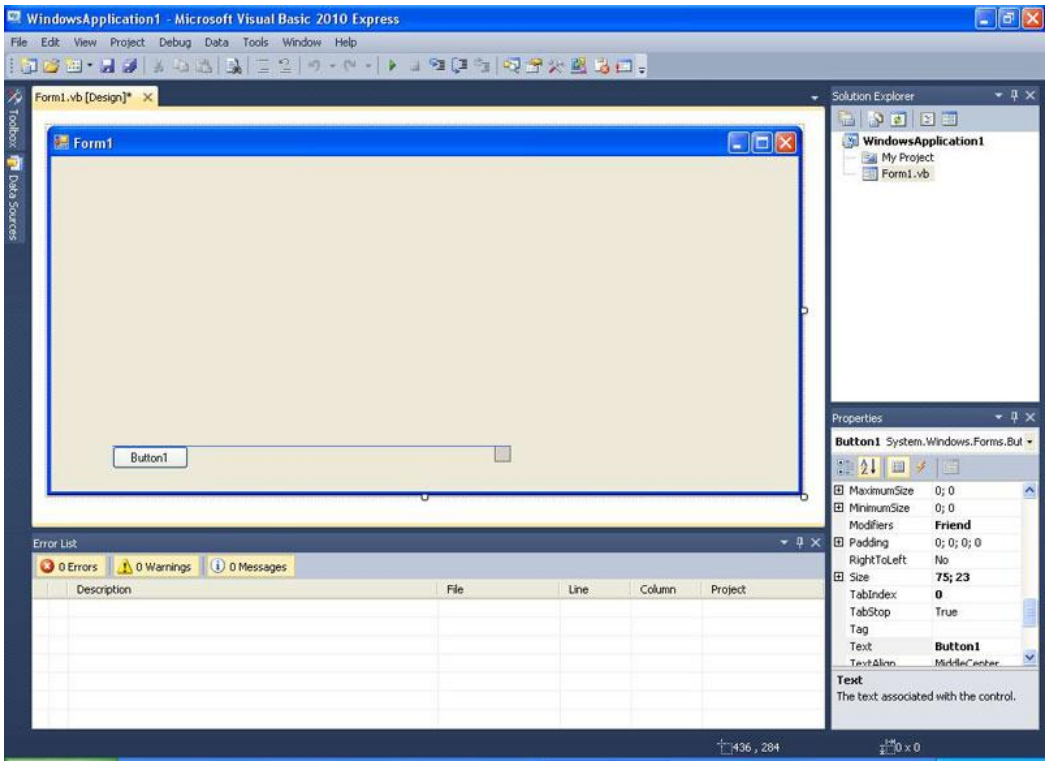
Εμείς την κάναμε σχεδόν διπλάσια.



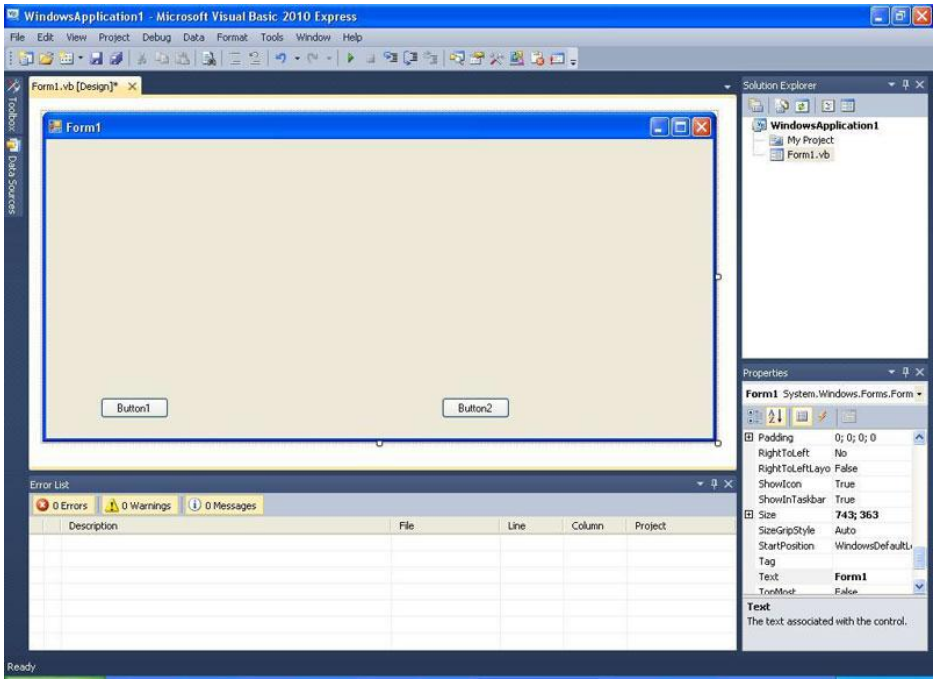
Επιλέγουμε ένα Button από το Toolbox και το τοποθετούμε στη φόρμα.



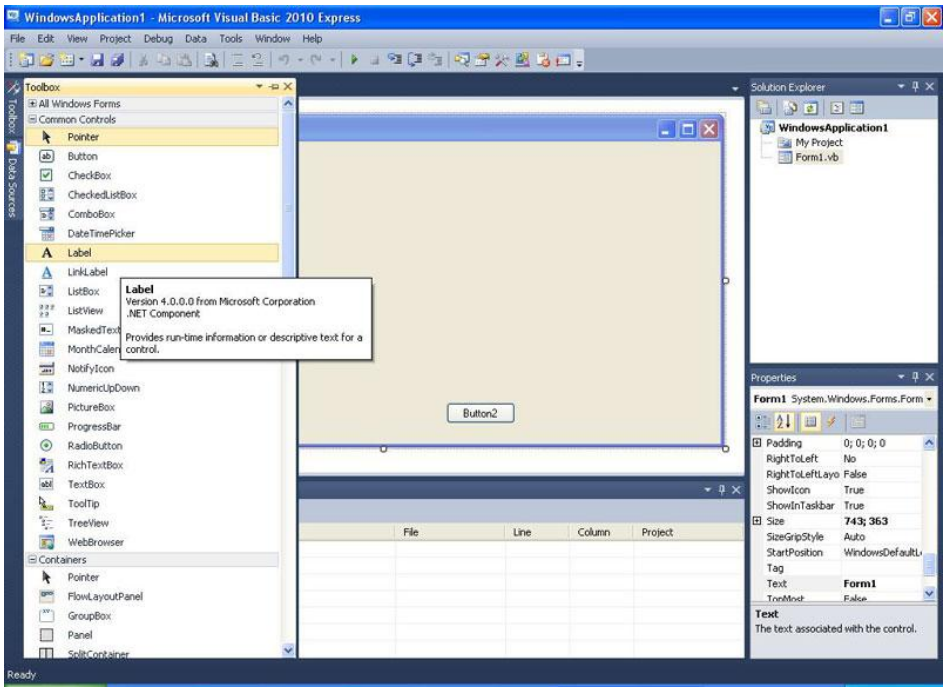
Ομοίως με Drag n drop τοποθετούμε και το δεύτερο button. Παρατηρείτε πόσο σας βοηθάει το περιβάλλον της visual Basic 2010 Express στη στοίχιση;



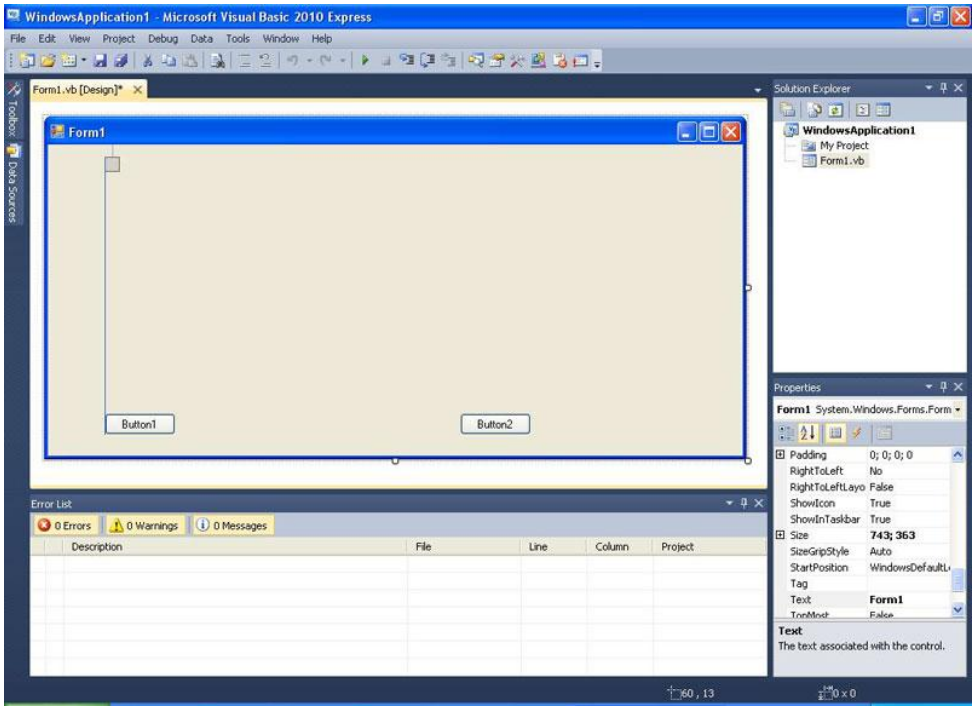
Έτοιμα τα κουμπιά.



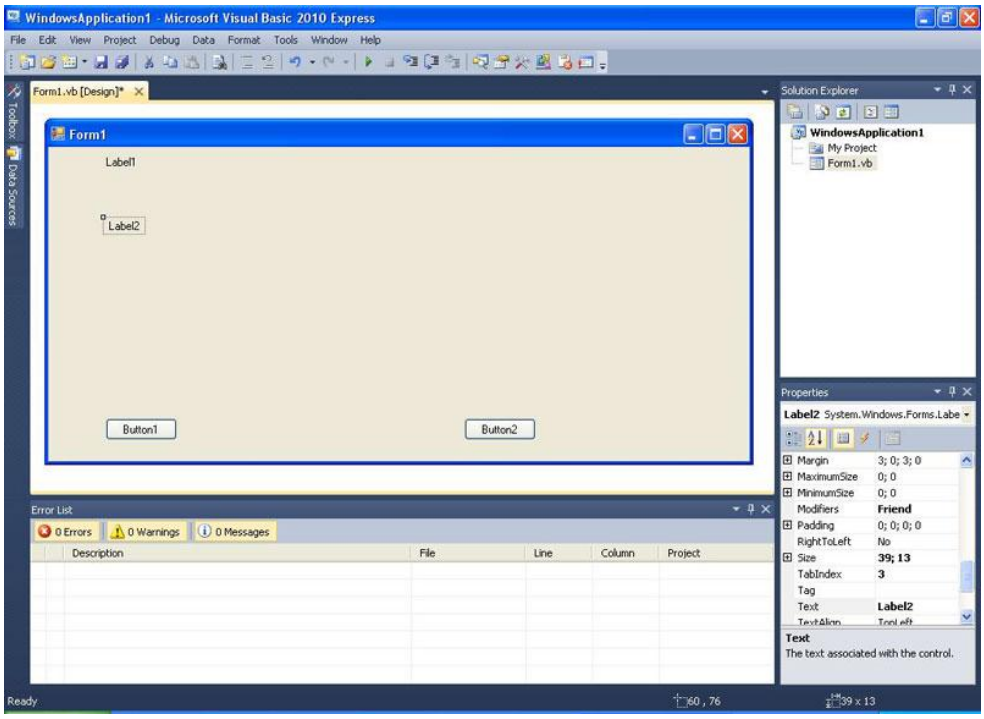
Ομοίως επιλέγουμε Label...



... τη στοιχίζουμε...

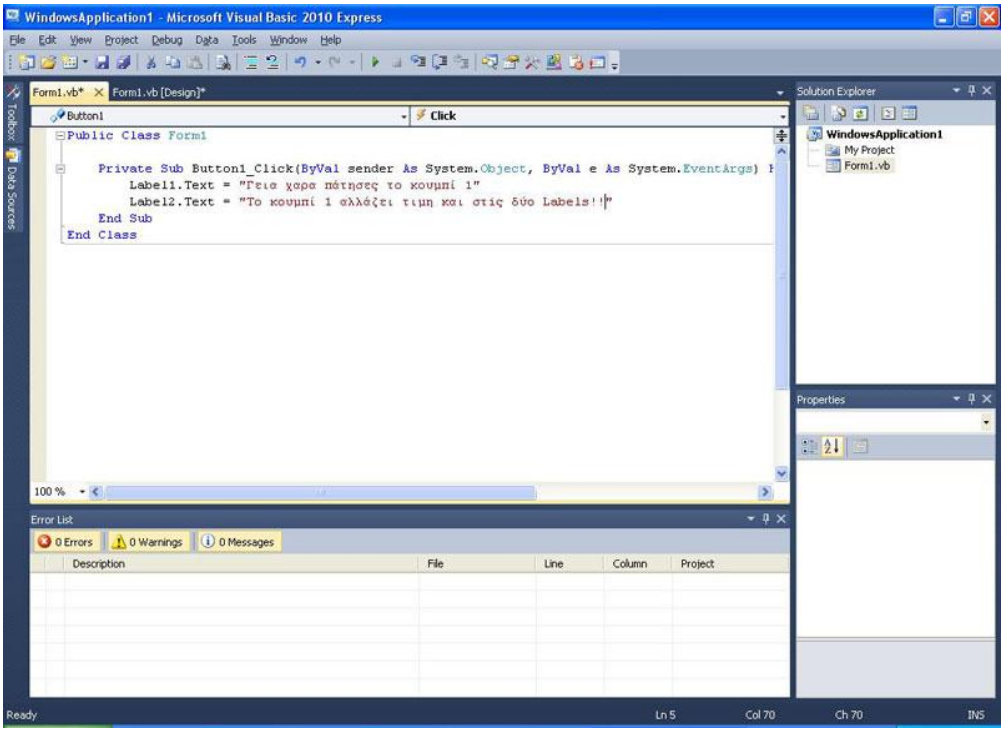


... και προσθέτουμε και μια δεύτερη.



Διπλό κλικ στο Button1. Κι εδώ είναι η πρώτη πρόκληση. Όταν κάνουμε κλικ στο Button1 θέλουμε να αλλάξουν οι τίτλοι και των δύο Label. Είναι απλό το τι κάνουμε. Γράφουμε μια γραμμή κώδικα για το text της Label1 και μια ακόμα για το text της Label2. Ορίστε οι δύο γραμμές κώδικα για το Private Sub Button1_Click

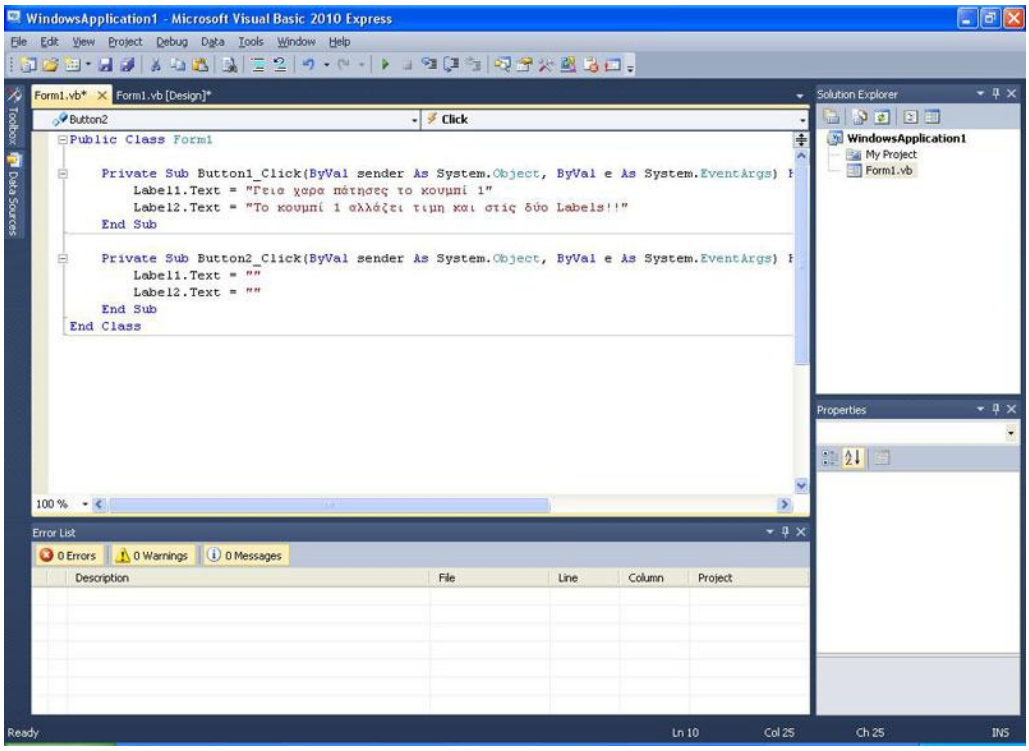
```
Private Sub Button1_Click
Label1.Text = "Γεια χαρά πάτησες το κουμπί 1"
Label2.Text = "Το κουμπί 1 αλλάζει τιμή και στις δύο Labels!!"
End Sub
```



Με την ίδια λογική που χρησιμοποιήσαμε παραπάνω, δηλαδή με ένα συμβάν κλικ να αλλάξουν δύο ιδιότητες, και μάλιστα διαφορετικών αντικειμένων, ομοίως λύνεται και η δεύτερη πρόκληση, αν και θέλει λίγο φαντασία. Αν θέλουμε να αδειάσουμε το περιεχόμενο του κειμένου μιας Label μπορούμε μεταξύ των εισαγωγικών να μη βάλουμε τίποτα απολύτως. Ορίστε οι δύο γραμμές κώδικα για το Private Sub Button2_Click

```
Private Sub Button2_Click
Label.Text = ""
Labe2Text = ""
End Sub
```

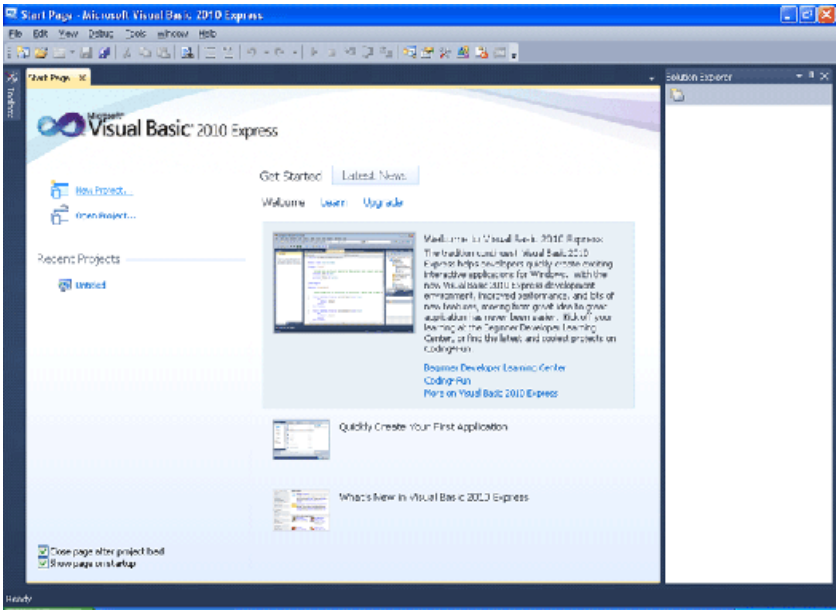
Ουσιαστικά, αν κάποιος βρει το πρώτο μέρος της άσκησης, δηλαδή σκεφτεί πως με ένα κλικ θα εκτελεστούν δύο εντολές, το μόνο που πρέπει να φανταστεί εδώ επιπλέον, είναι να ορίσει το κείμενο ως τίποτα.



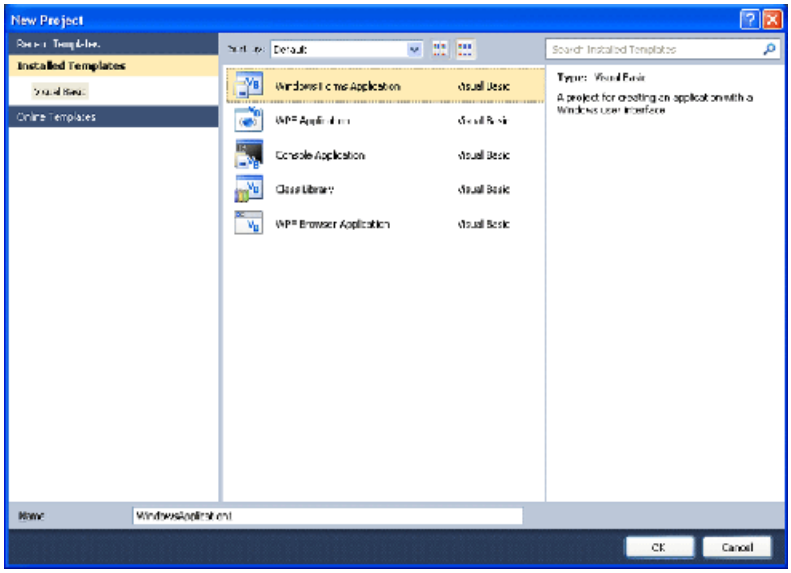
Στόχος μας ήταν λοιπόν, να κατανοήσετε τον τρόπο που πρέπει να σκέφτεστε όταν γράφετε ένα πρόγραμμα. Πολλή φαντασία, δημιουργικότητα και πρακτικό μυαλό. Αν δε γνωρίζετε προγραμματισμό, κι αυτό το βοήθημα είναι η πρώτη σας επαφή με το χώρο και βρήκατε τη λύση μόνοι σας, συγχαρητήρια. Ωστόσο αν δεν τα καταφέρατε, μην απογοητεύεστε. Μάθατε τη λογική εκ των υστέρων. Και στις δύο περιπτώσεις κατέχετε την ίδια γνώση.

ΚΕΦΑΛΑΙΟ 5
ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΤΗΣ VISUAL BASIC .NET 2010 EXPRESS
ΜΕΡΟΣ ΔΕΥΤΕΡΟ

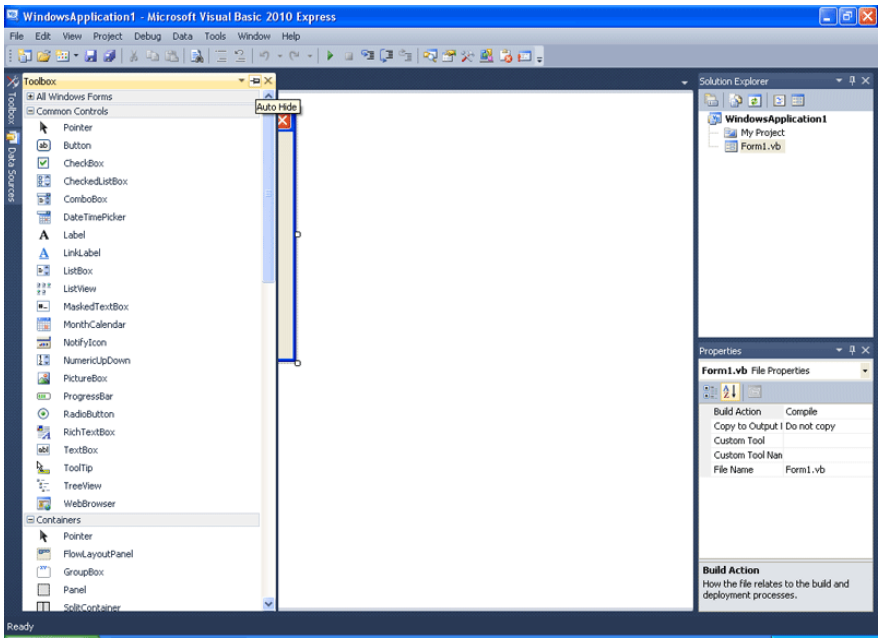
Αφού έχουμε εξοικειωθεί με τα βασικά του περιβάλλοντος εργασίας της Visual Basic Express, είναι καιρός να δούμε κάποια επιπλέον χρήσιμα πράγματα που μπορούμε να κάνουμε. Δημιουργήστε ένα νέο project.



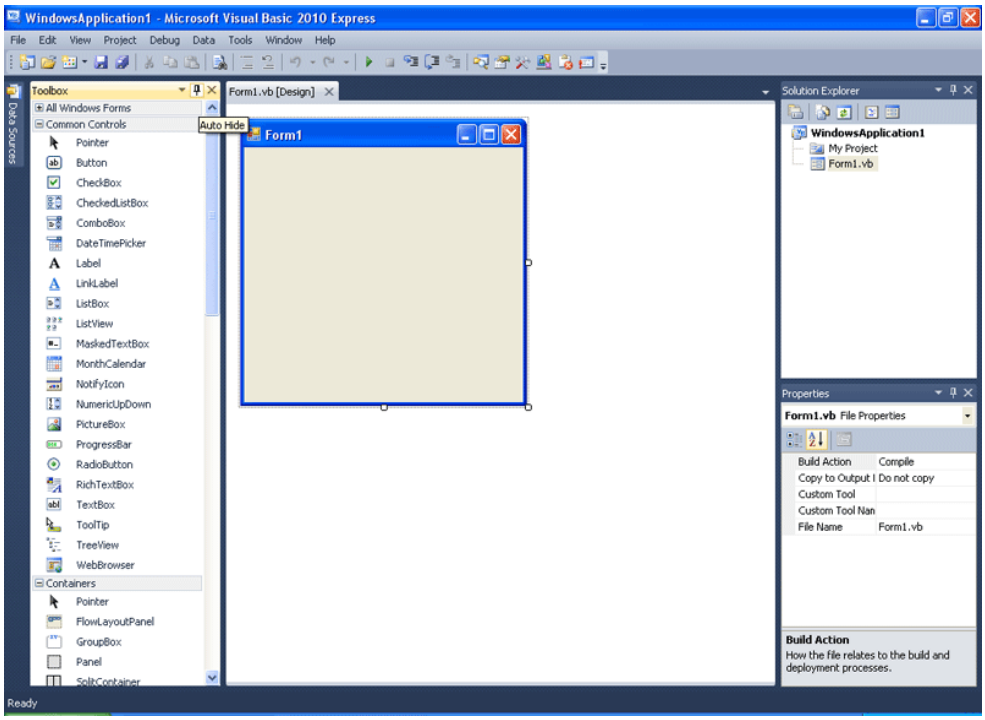
Θα είναι πάλι τύπου Windows Forms Application.



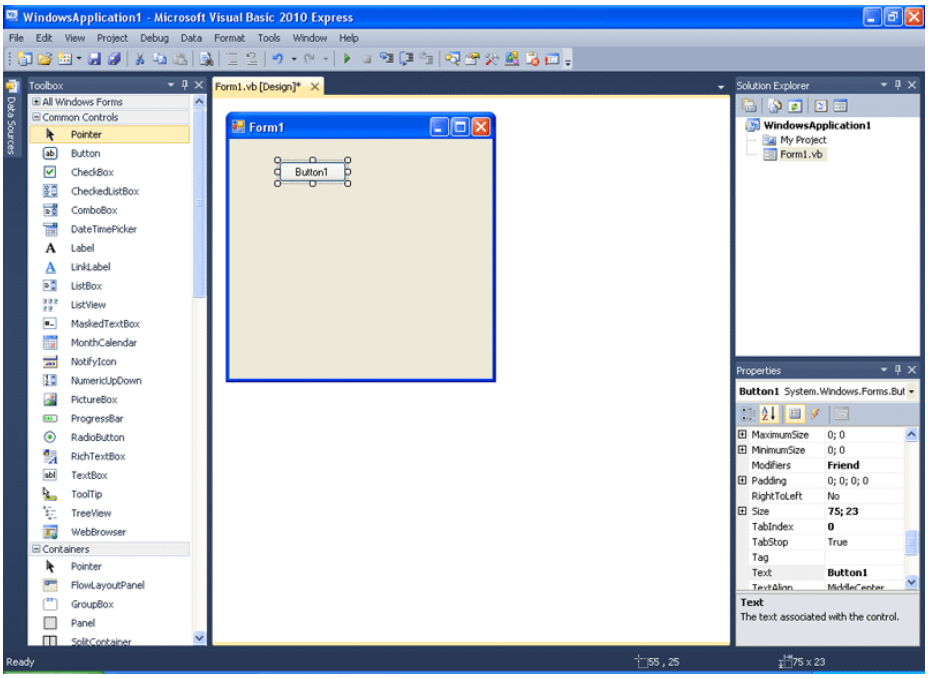
Πηγαίνετε στο Toolbox, όπως βλέπετε στην παρακάτω εικόνα και πατήστε το κουμπί καρφίτσα. Είναι το λεγόμενο κουμπί Auto Hide. Αυτό το κουμπί "καρφιτσώνει" το Toolbox, ώστε να μην κρύβεται όταν δεν το χρησιμοποιείτε. Κοινώς φαίνεται μόνιμα όταν είναι πατημένο.



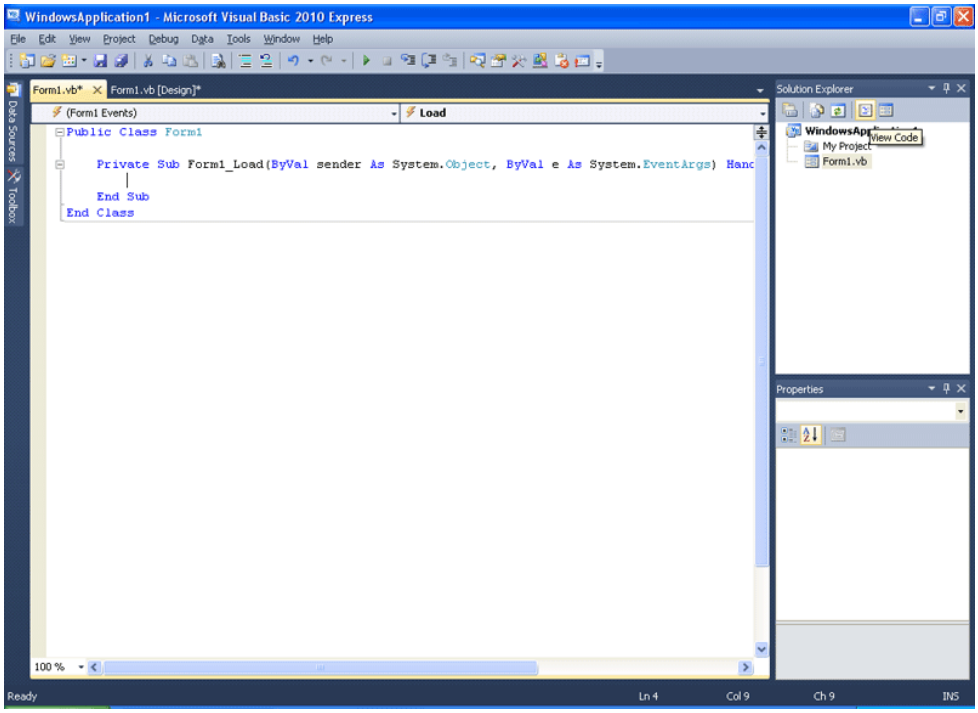
Για να ξανακρύβεται όπως πριν, θα πρέπει να το ξαναπατήσετε.



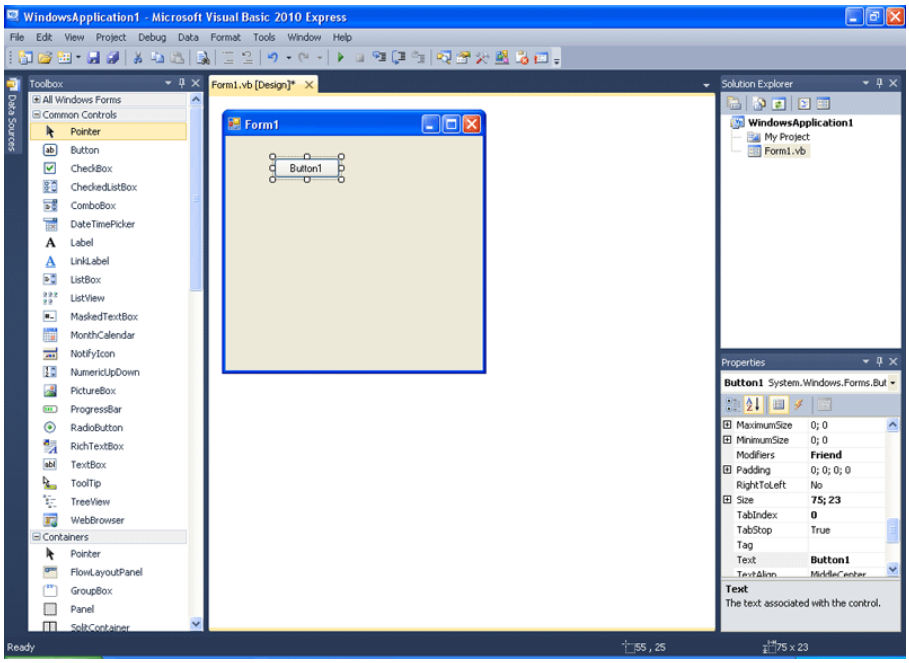
Φτιάξτε ένα Button. Κάντε διπλό κλικ πάνω του για να εμφανιστεί το περιβάλλον που γράφουμε κώδικα.



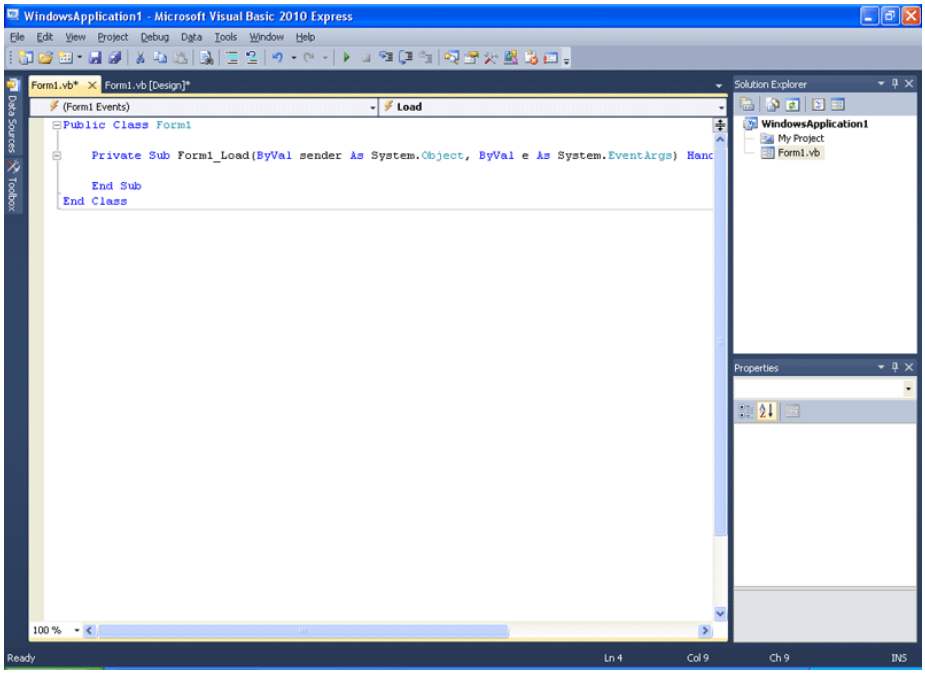
Παρατηρήστε πάνω από τον κώδικα, τις δύο επιλογές. Η μία είναι η Form1.vb* η οποία μάλιστα είναι σε ένα μπεζ πλαίσιο. Δίπλα της υπάρχει η Form1.vb[Design]*. Αν κάνετε κλικ πάνω της, θα κρυφτεί το περιβάλλον που γράφουμε κώδικα και θα εμφανιστεί το περιβάλλον που σχεδιάζουμε τη φόρμα.



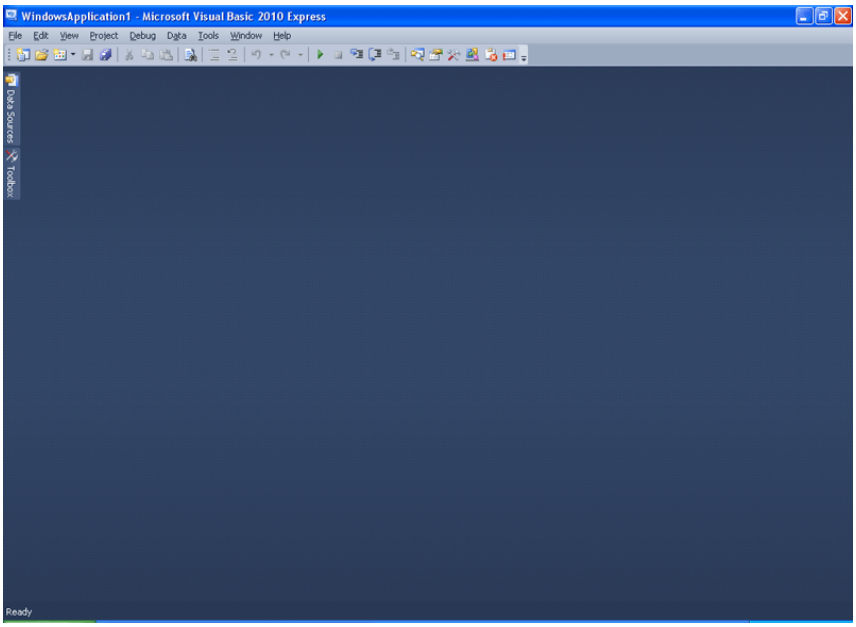
Βλέπετε; Αυτός ο τρόπος μετάβασης, είναι ο σύνηθες. Μπορείτε να πατάτε το "X" για να σβήνετε το Form1.vb* αν τελειώνετε με τη συγγραφή κώδικα. Μέχρι τότε, μπορείτε να μεταβαίνετε από το ένα περιβάλλον στο άλλο απλά κι εύκολα.



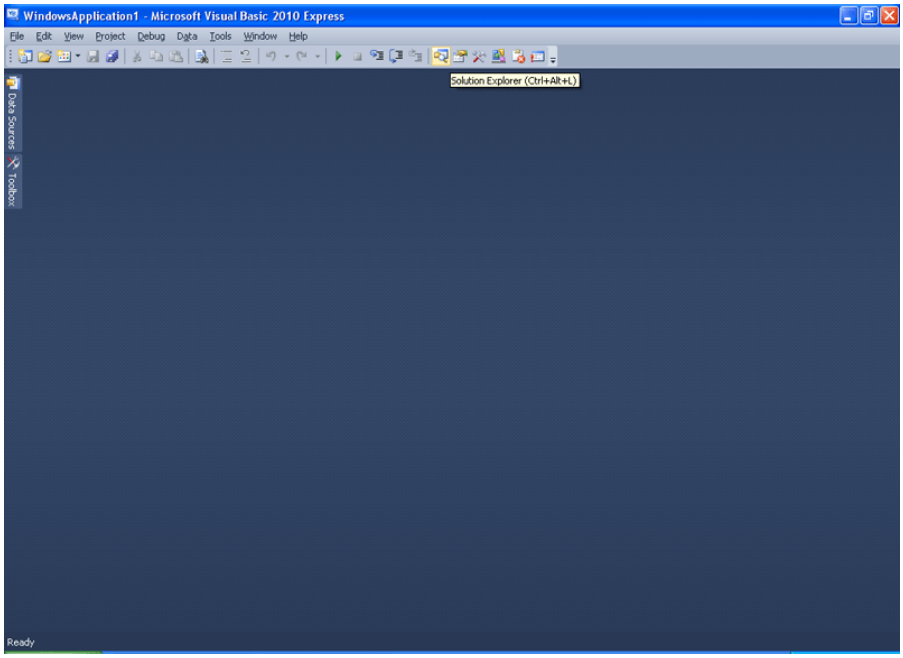
Ξαναμεταβείτε στο περιβάλλον κώδικα. Επίσης από το Solution Explorer μπορείτε να δείτε τον κώδικα πατώντας το εικονίδιο που φαίνεται στην παρακάτω εικόνα. Πατήστε "X" στο Form1.vb* για να κλείσει. Πατήστε "X" στο Properties για να κλείσει. Πατήστε "X" στο Solution Explorer για να κλείσει.



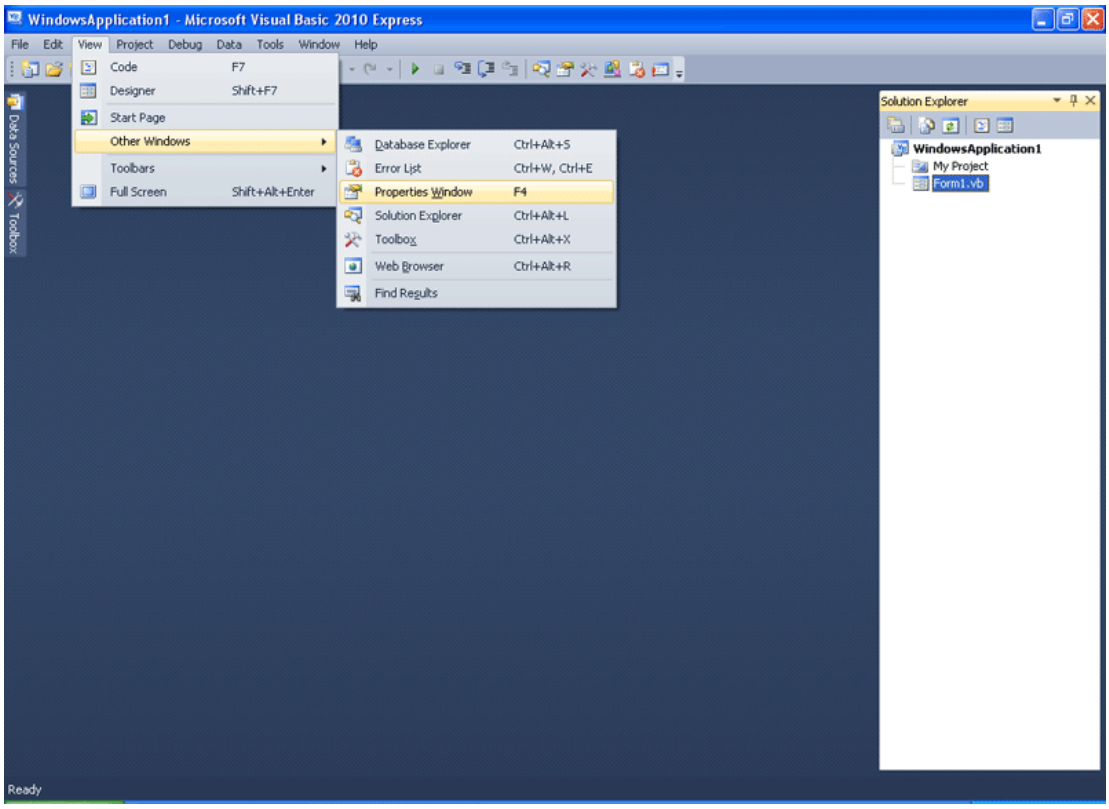
Το περιβάλλον σας πρέπει να είναι κάπως έτσι.



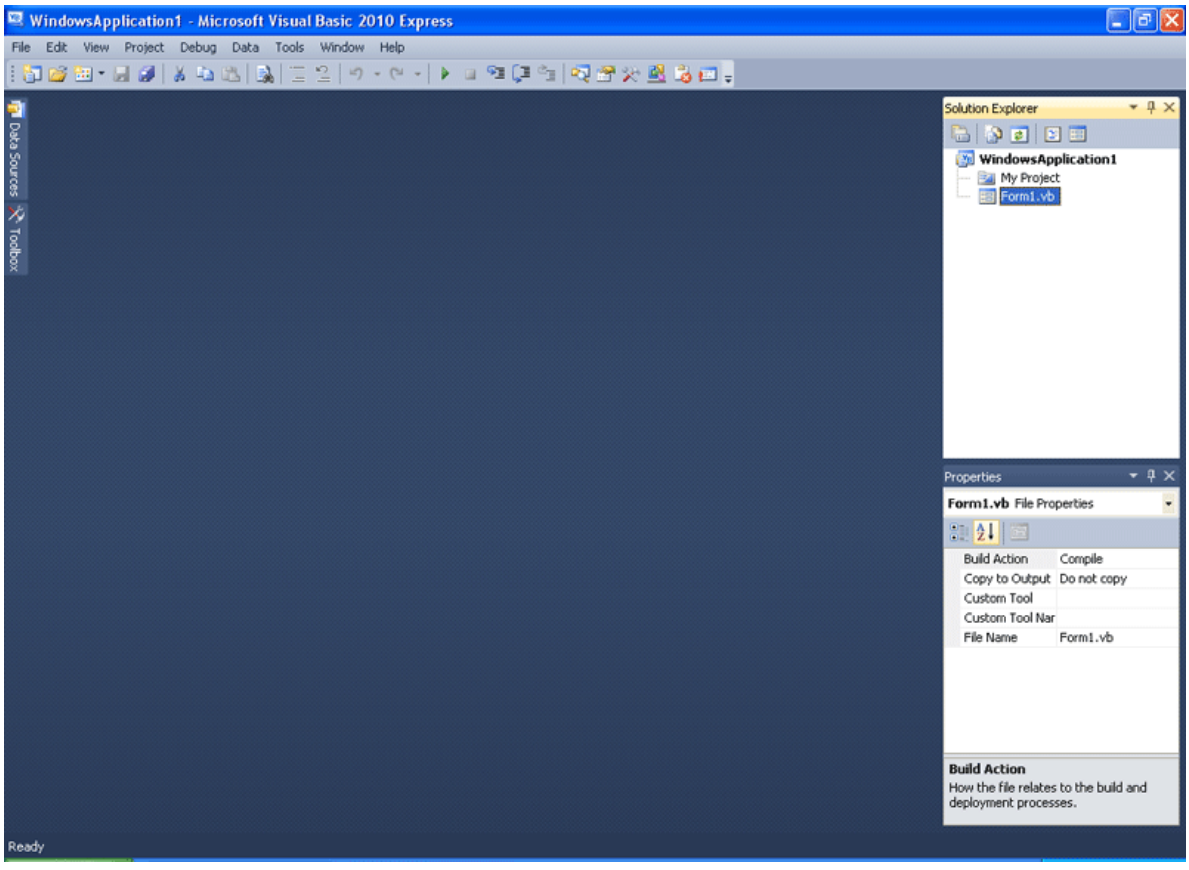
Σε περίπτωση που κατά λάθος ή εσκεμμένα, σβήσατε κάποια από αυτά τα παράθυρα, μπορείτε από το menu εργαλείων να τα ξανανοίξετε. Εδώ ανοίγει το Solution Explorer.



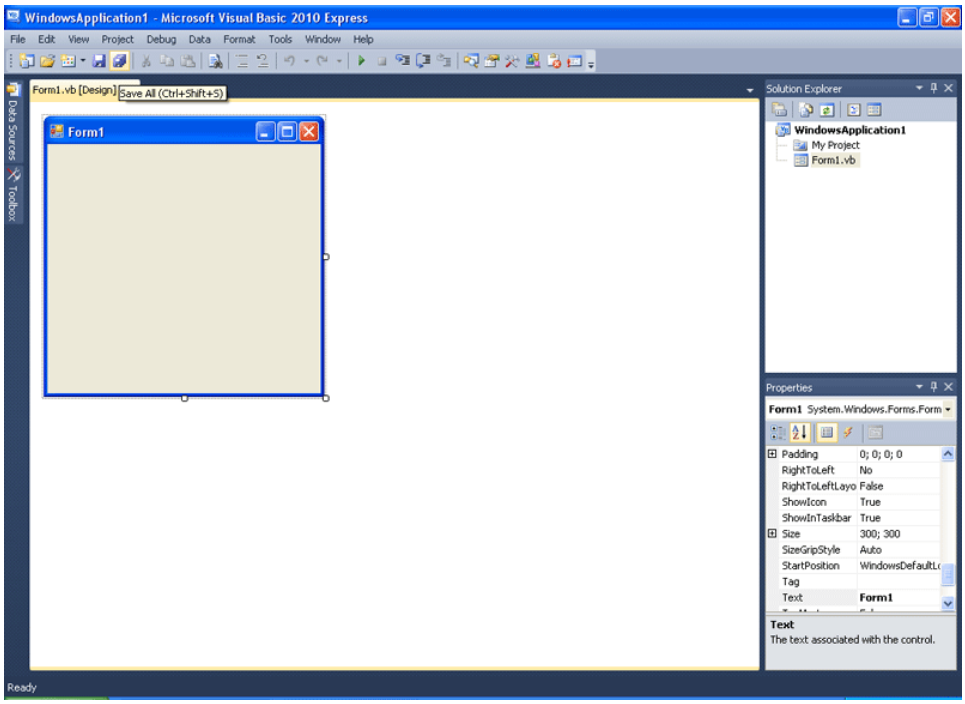
Εναλλακτικά ανοίγουν και από το menu View.



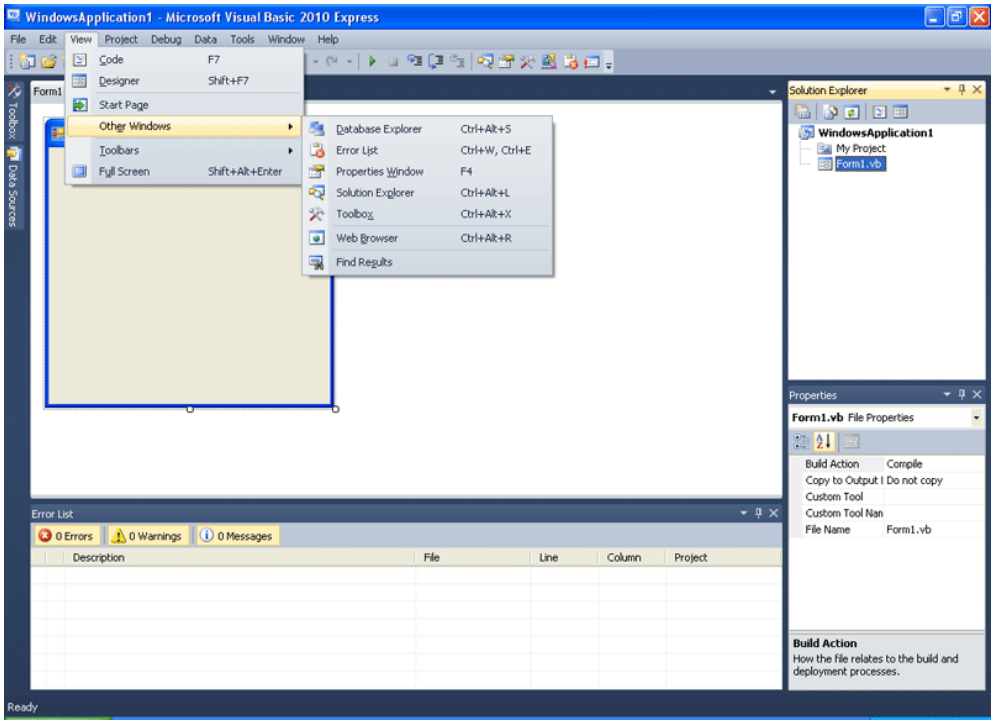
Από το Solution Explorer, ανοίγουν πάλι τα Form1.vb[Design]*



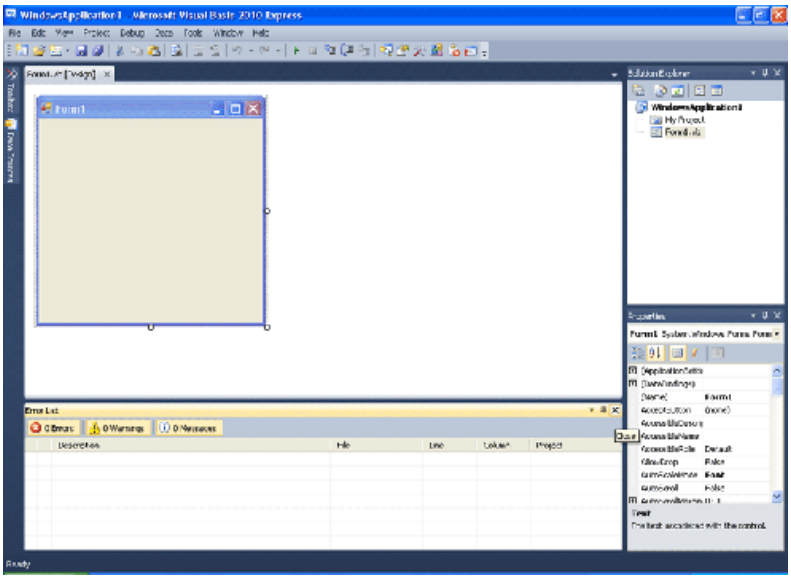
Με το κουμπί Save All αποθηκεύετε όλο το project και όλα τα συστατικά του. Να κάνετε συχνά save ώστε να μη χάνετε την εργασία και τον κόπο σας από κάποιο ατυχές γεγονός. Φανταστείτε το ενδεχόμενο, να έχετε γράψει ένα μεγάλο μέρος κώδικα, να είχε πρόβλημα και μετά από δύο ώρες κόπο, να το έχετε διορθώσει. Έστω ότι γίνεται διακοπή ηλεκτρικού ρεύματος. Χάνεται όλος αυτός ο κόπος άμεσα, σε μια στιγμή, επειδή πολύ απλά δεν είχατε κάνει ένα κλικ. Είναι κρίμα, οπότε έχετε κατά νου ότι πρέπει πολύ συχνά να κάνετε save.



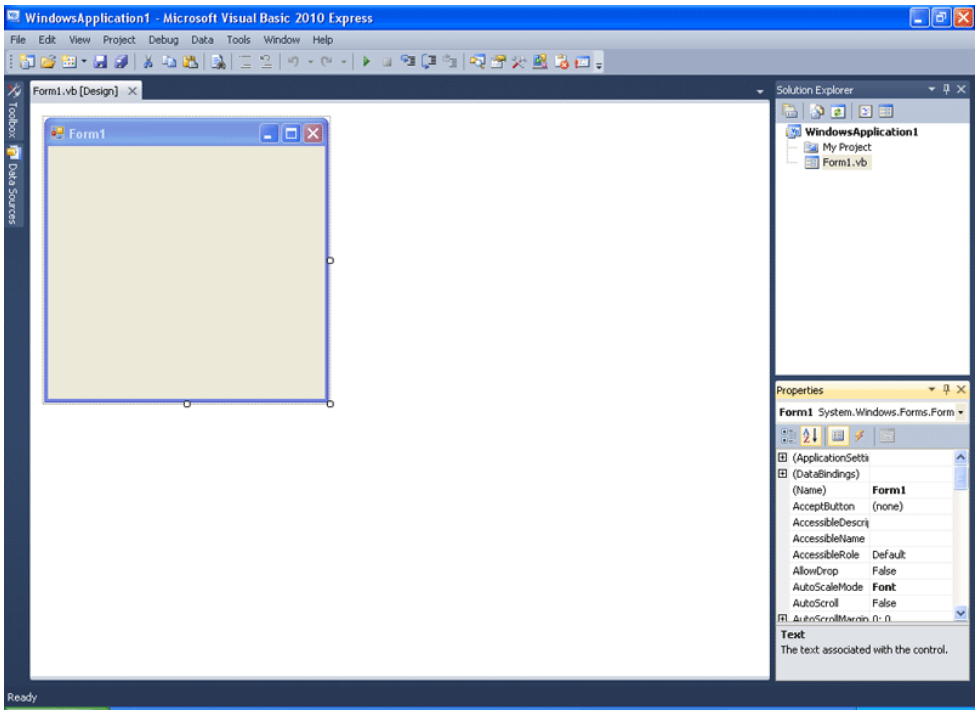
Ξαναπηγαίνετε στο View Menu. Επιλέξτε την Error List. Εμφανίζεται στο κάτω μέρος. Στην εικόνα την ανοίξαμε ήδη. Εδώ θα εμφανιστούν πιθανά προβλήματα του προγράμματός σας, όταν το δοκιμάσετε, να δείτε αν τρέχει σωστά.



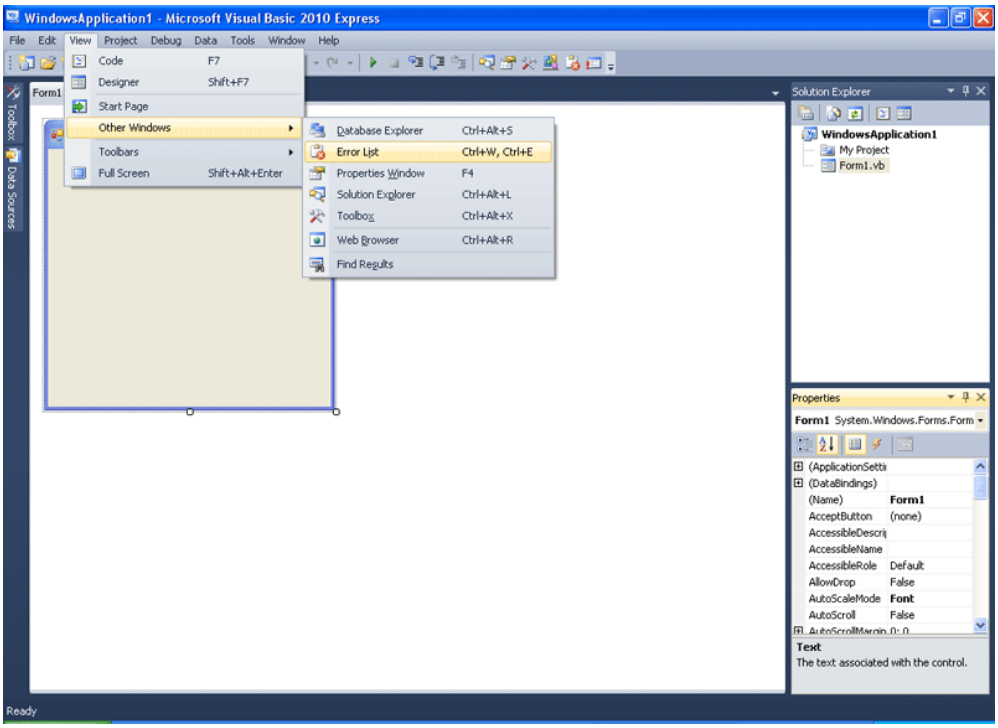
Σβήστε την πάλι πατώντας το "X" στην πάνω δεξιά της γωνία.



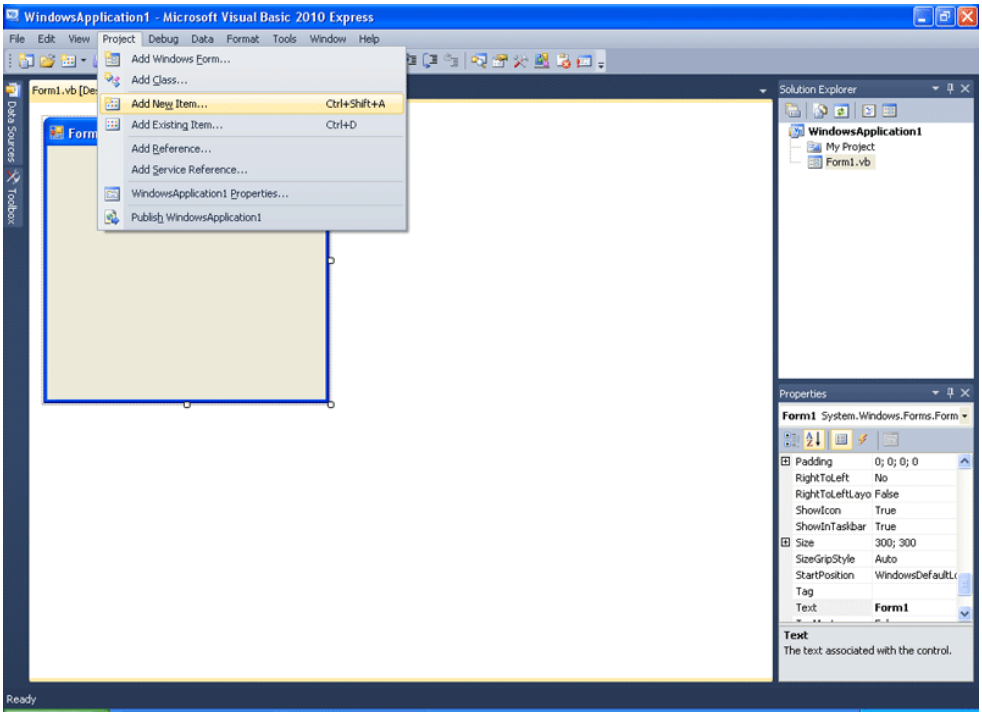
Το περιβάλλον επανήλθε στην αρχική του μορφή.



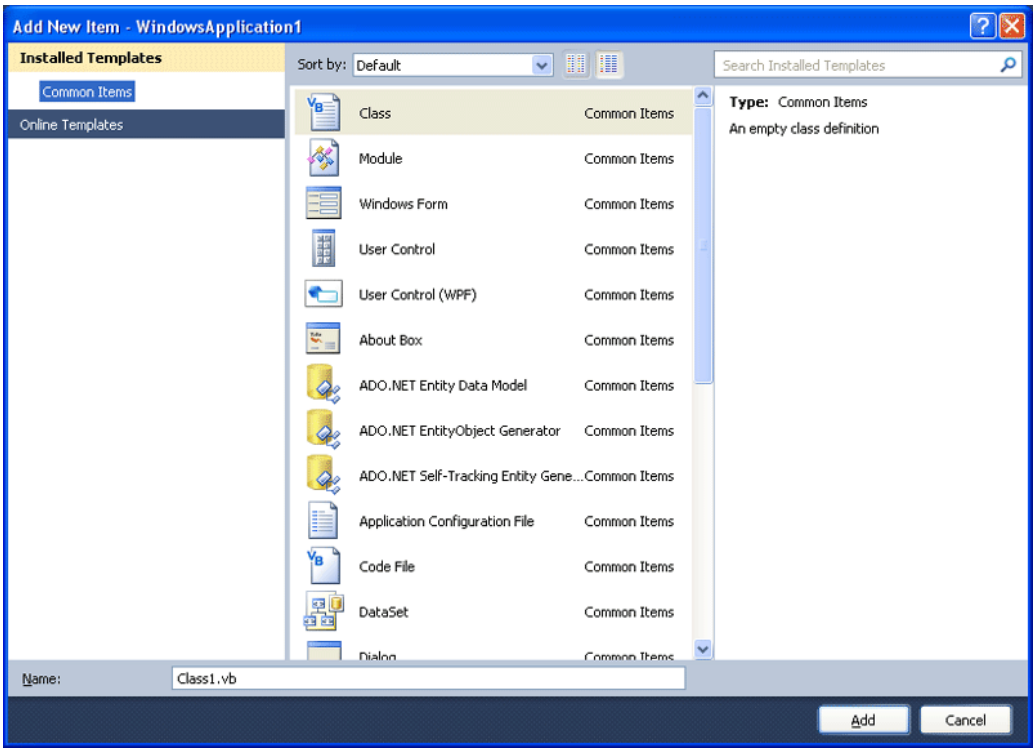
Το View Menu λοιπόν μας βοηθά να επαναφέρουμε παράθυρα που κλείσαμε. Παρατηρήστε και τους συνδυασμούς πλήκτρων που μπορείτε να πατάτε, αντί να πηγαίνετε από το menu. Γενικά το περιβάλλον της Visual Basic . NET Express είναι πολύ πρακτικό.



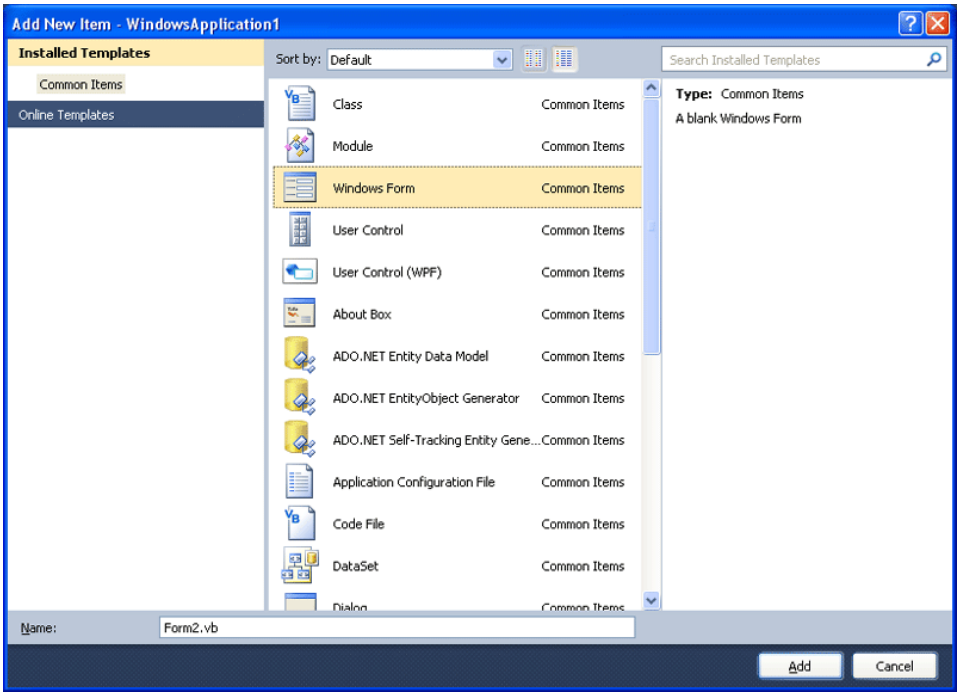
Πηγαίνετε τώρα στο menu Project κι επιλέξτε Add New Item. Αυτό μπορείτε να το κάνετε και από το μενού με τα εικονίδια, αλλά προς το παρόν, ακολουθήστε αυτό τον τρόπο.



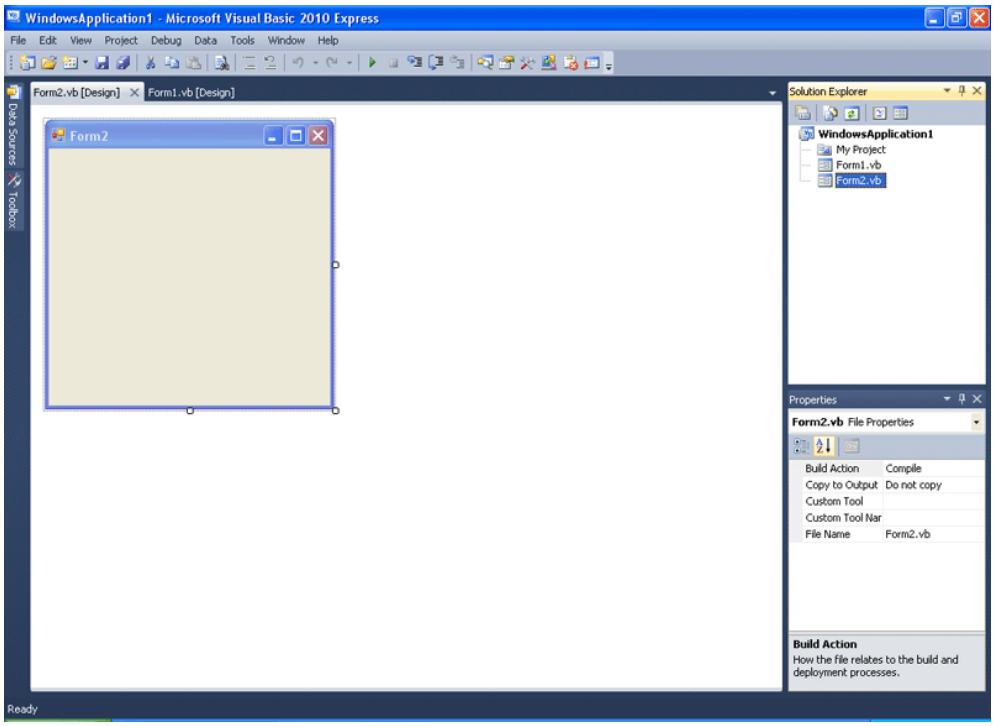
Εμφανίζεται ένα παράθυρο με επιλογές. Εδώ, πάνω πάνω, βλέπετε δύο πολύ σημαντικά συστατικά. Την Class και το Module. Δε θα ασχοληθούμε τώρα με αυτά. Μιας κι ασχολούμαστε όμως με το περιβάλλον...



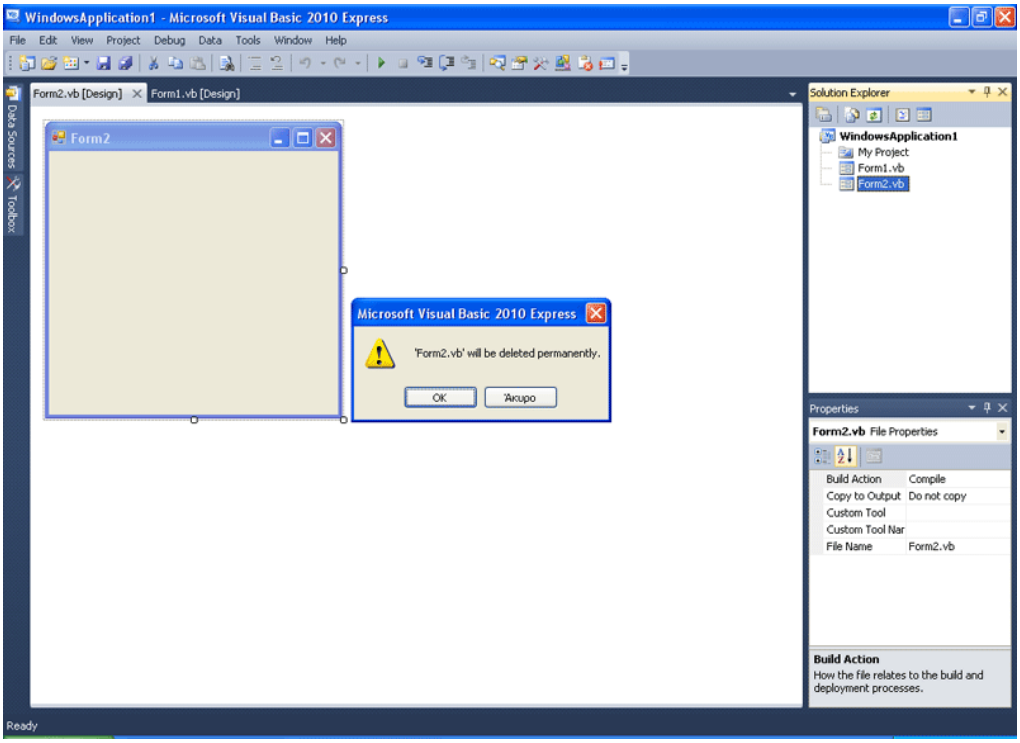
... επιλέξτε Windows Form.



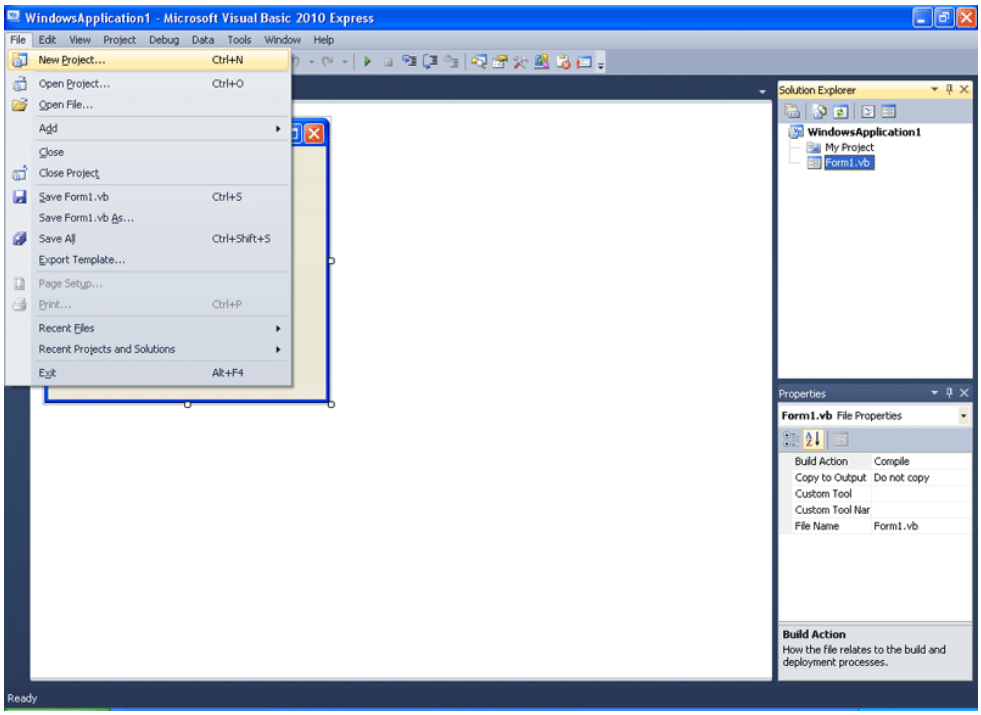
Πλέον έχετε δύο Form [Design]* περιβάλλοντα, ένα για κάθε φόρμα. Αυτό φαίνεται και στο Solution Explorer. Παρεμπιπτόντως, λέγεται περιηγητής λύσης, διότι κάθε πρόγραμμα - project που φτιάχνουμε, το θεωρούμε ως μία λύση σε κάποιο πρόβλημά μας.



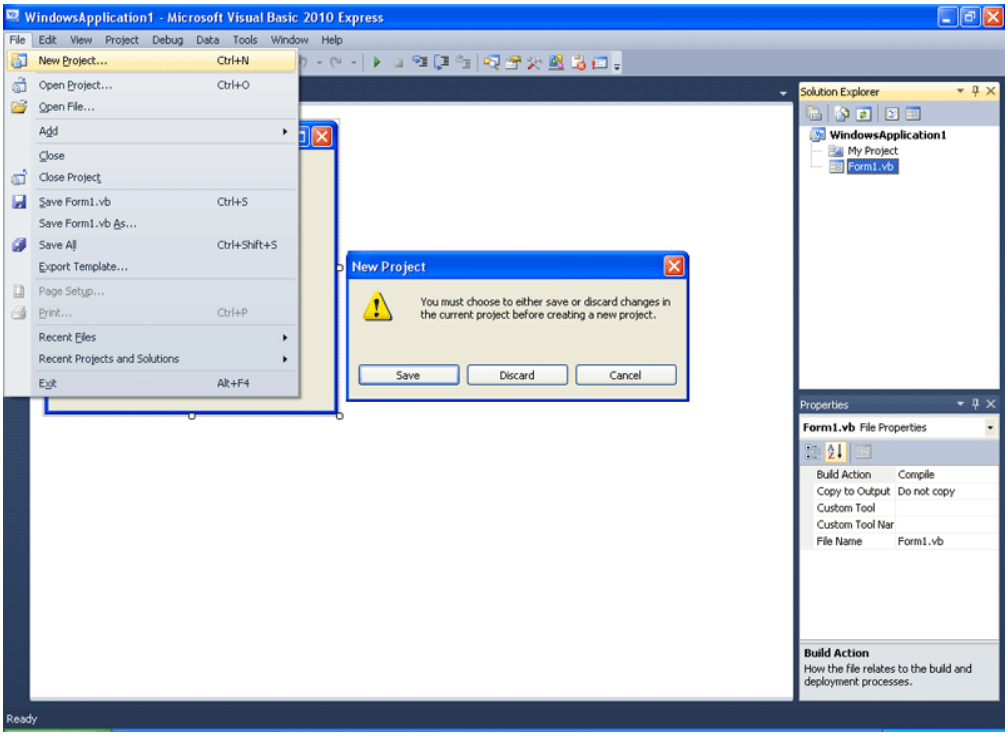
Πηγαίνετε στο Solution Explorer και επιλέξτε τη Form2. Πατήστε το πλήκτρο DEL στο πληκτρολόγιο. Στο μήνυμα επιλέγετε OK. Έτσι σβήνετε ένα αντικείμενο.



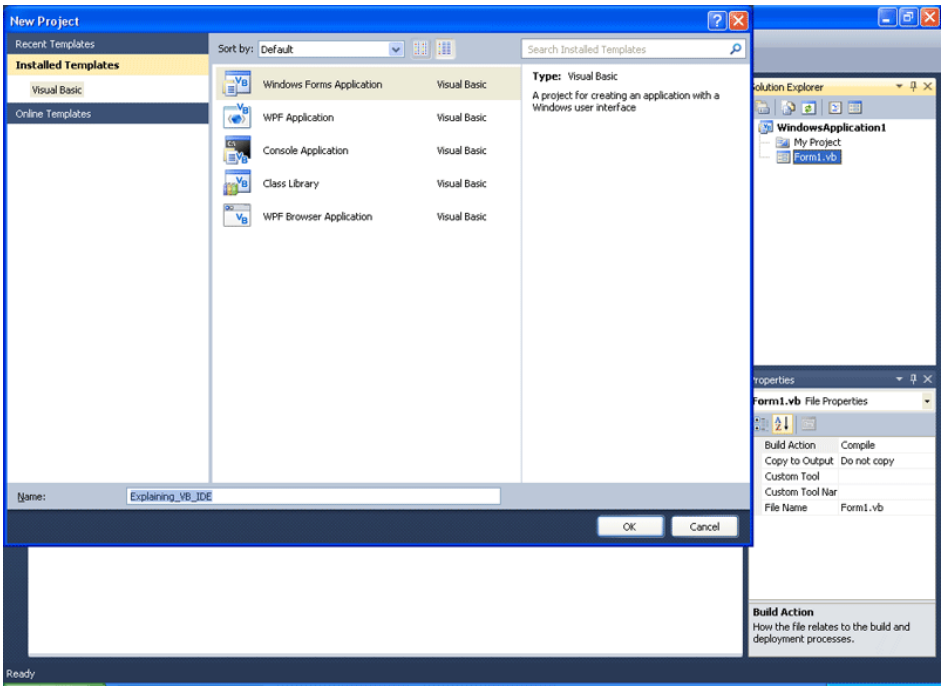
Αν πάτε να δημιουργήσετε ένα νέο project χωρίς να έχετε κάνει save...



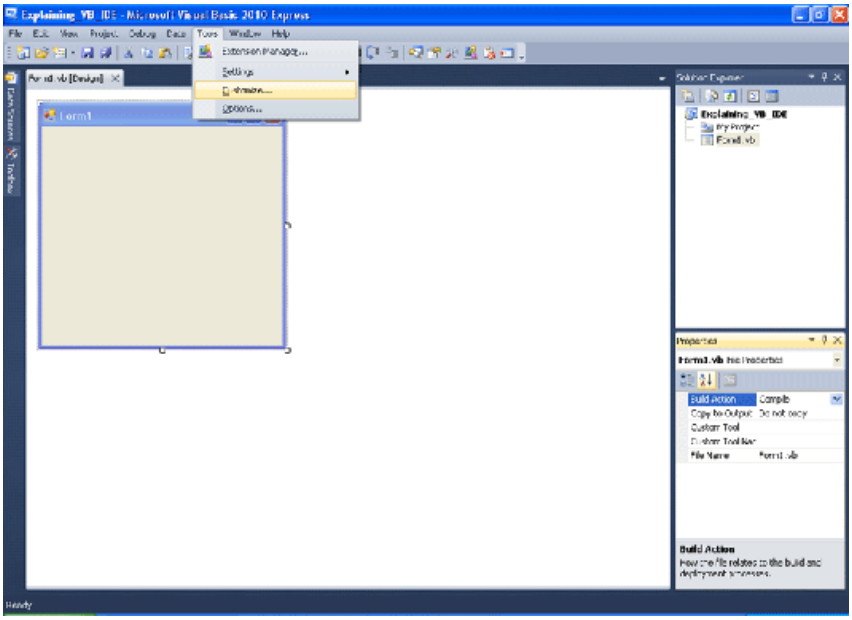
..σας εμφανίζει η γλώσσα κατάλληλο μήνυμα με επιλογές. Εδώ κάντε Discard γιατί έτσι κι αλλιώς δεν κάναμε και κάτι σπουδαίο. Ωστόσο σε άλλη περίπτωση, όταν φτιάχνετε κάποιο πρόγραμμά, είναι καλό που σας ενημερώνει ότι πρέπει να κάνετε save.



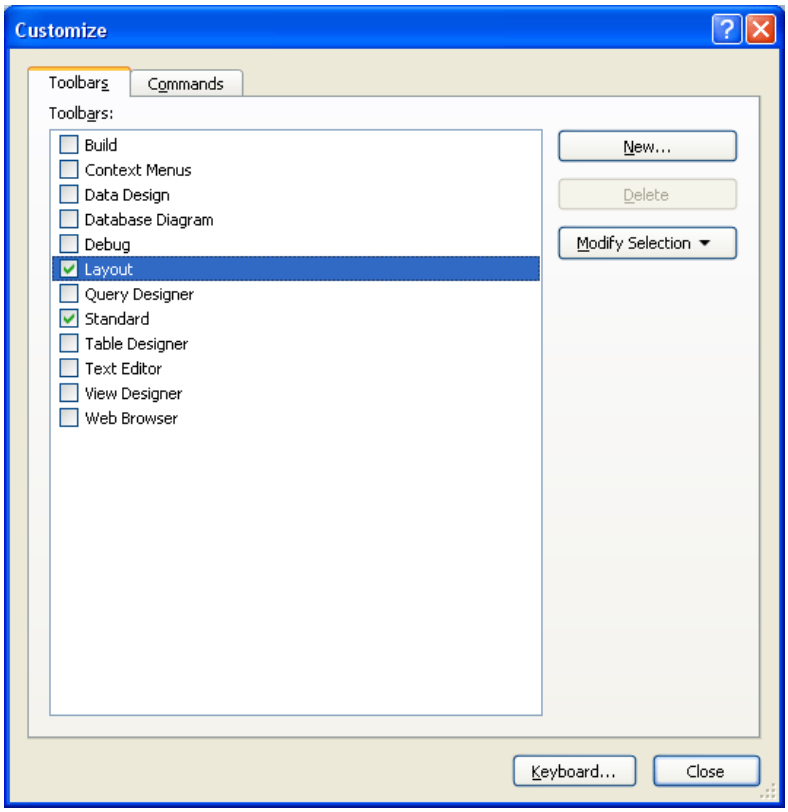
Επιλέγετε πάλι Windows Forms application και είστε πάλι έτοιμοι να εργαστείτε σε ένα νέο project.



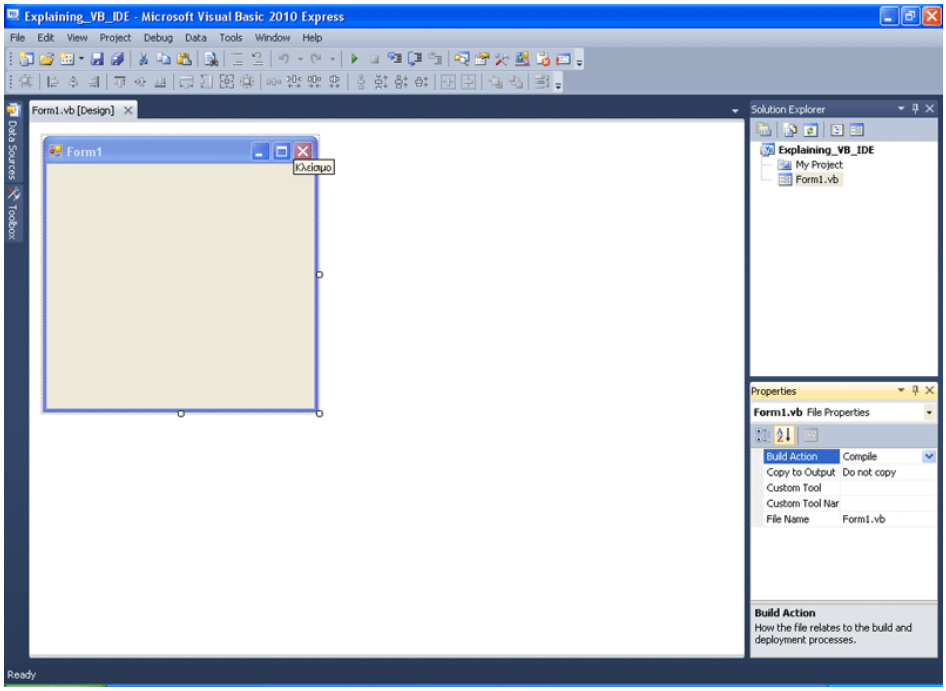
Στο menu Tools Customize... μπορείτε να επιλέξετε να εμφανίζονται περισσότερα μενού με εικονίδια κι εργαλεία.



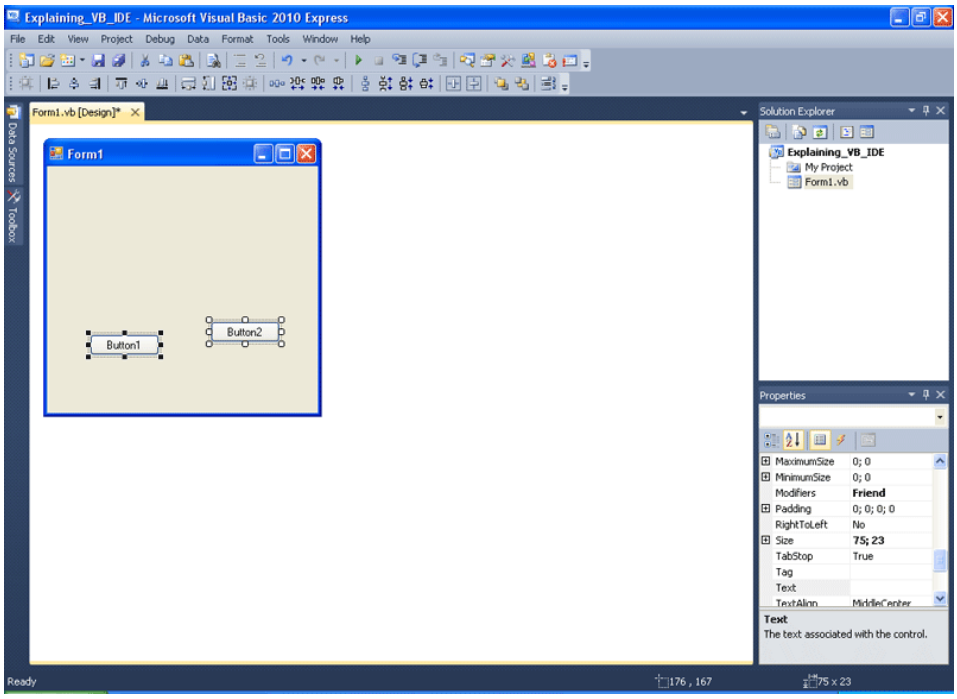
Επιλέγετε τι θέλετε να υπάρχει σαν μπάρα εργαλείων, ανάλογα με τις ανάγκες σας. Επιλέξτε επιπλέον το Layout και πατήστε το κόκκινο "X" για να κλείσει το παράθυρο.



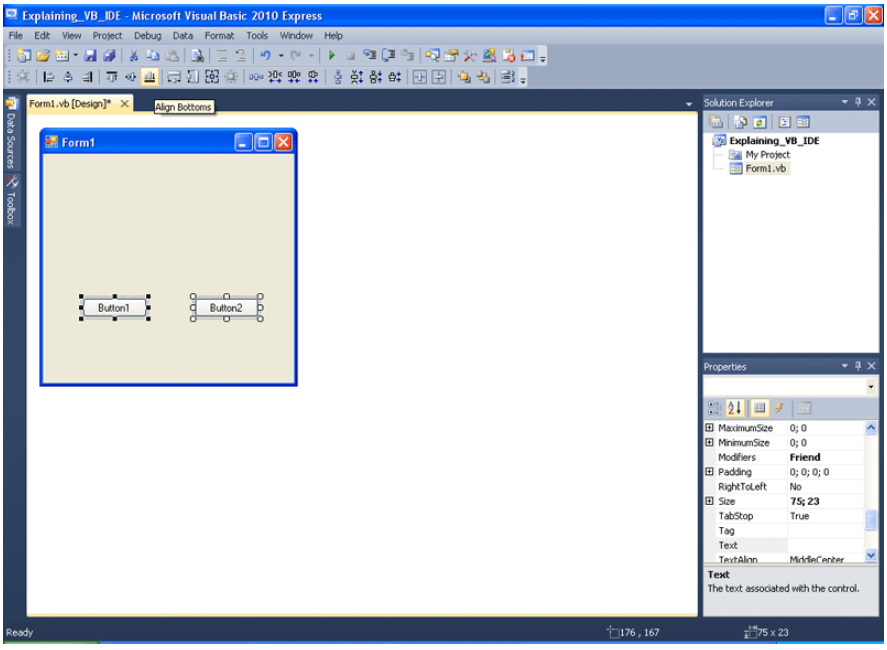
Κάντε κλικ στη Form1.



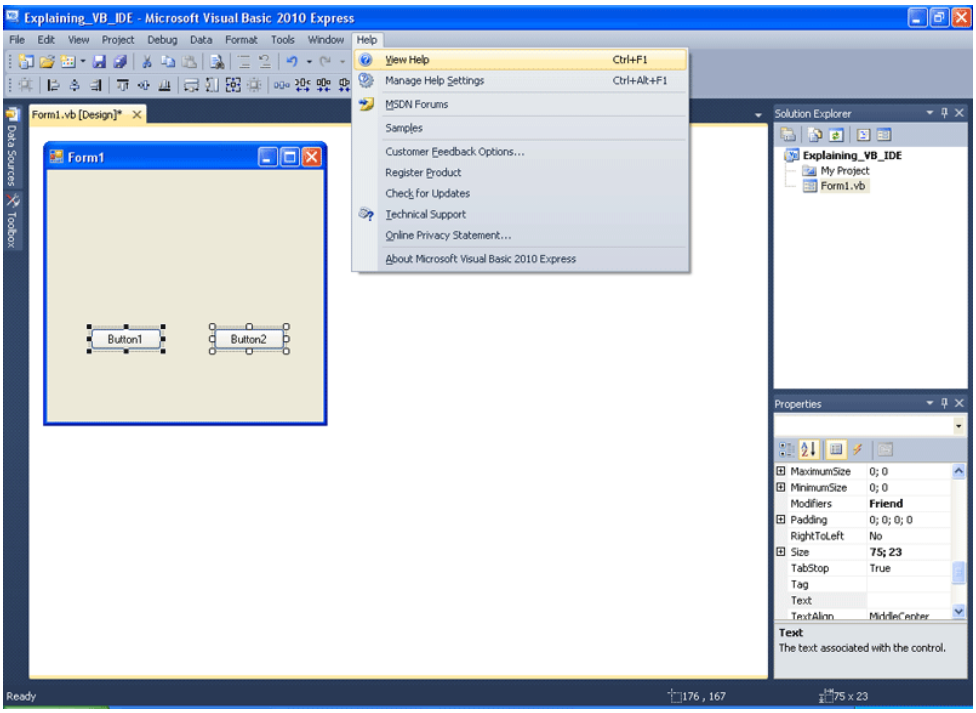
Προσθέστε δύο buttons. Η μπάρα Layout έχει ήδη ενεργά κάποια στοιχεία της. Κάντε αριστερό κλικ πάνω στη φόρμα, και κρατώντας πατημένο το αριστερό κουμπί του ποντικιού, σύρετε το ώστε να δημιουργήσετε ένα τεράγωνο που θα εμπεριέχει τα δύο Buttons. Με αυτό τον τρόπο τα επιλέγετε.



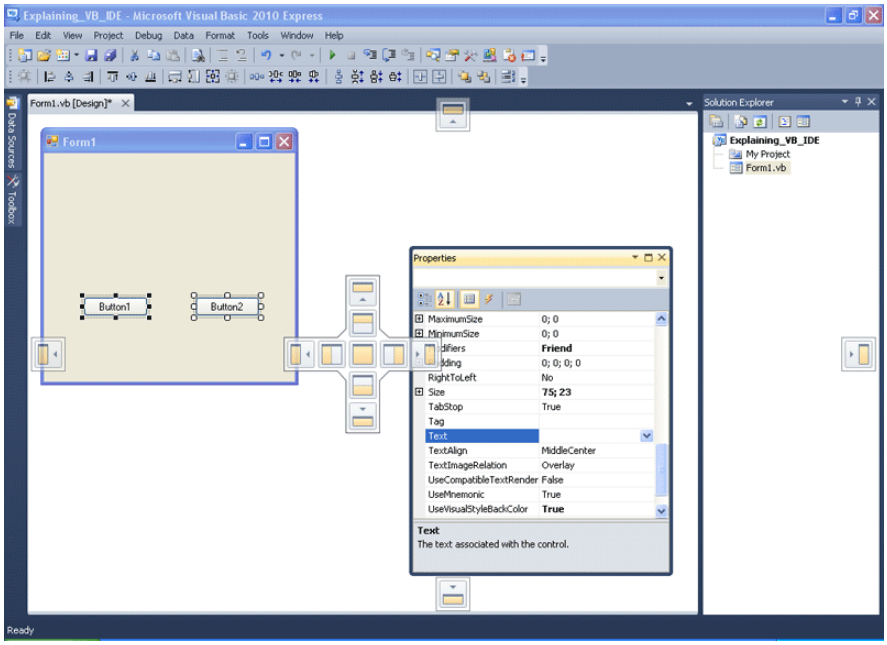
Βρείτε το Align Button και πατήστε το. Αυτό στοιχίζει τα επιλεγμένα στοιχεία.



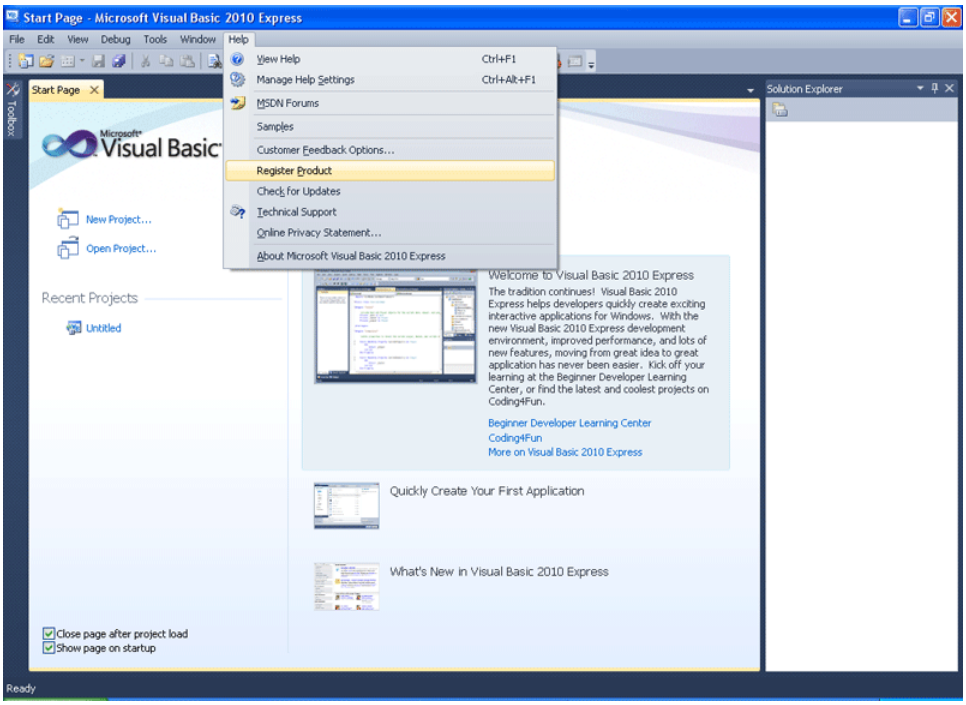
Στο menu Help το View Help μπορεί να σας δώσει αρκετές πληροφορίες για κάθε σχεδόν θέμα, που αφορά τη Visual Basic .NET Express.



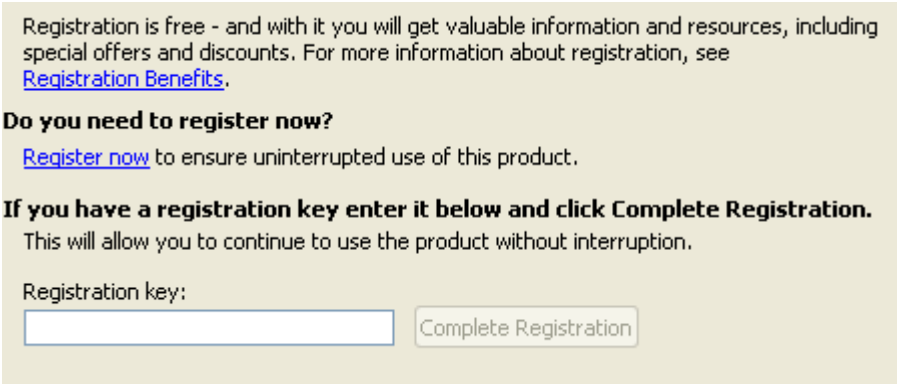
Κάθε παράθυρο του περιβάλλοντος, μπορεί να μετακινηθεί σε άλλο σημείο, κρατώντας πατημένο το αριστερό πλήκτρο του ποντικιού και σέρνοντάς το, κοινώς κάνοντας Drag n Drop.



Τέλος θα σας δείξουμε πως να δηλώσετε το αντίγραφο της Visual Basic .NET 2010 Express στη Microsoft. Αυτό πρέπει να γίνει εντός 30 ημερών, αλλιώς κλειδώνει. Ωστόσο με τη δήλωση θα πάρετε ένα δωρεάν κλειδί για να τη χρησιμοποιείτε για απεριόριστο χρόνο πάντα υπό τη συμφωνία και τους όρους που συνοδεύουν το λογισμικό. Πηγαίνετε στο menu Help κι επιλέγετε Register Product.

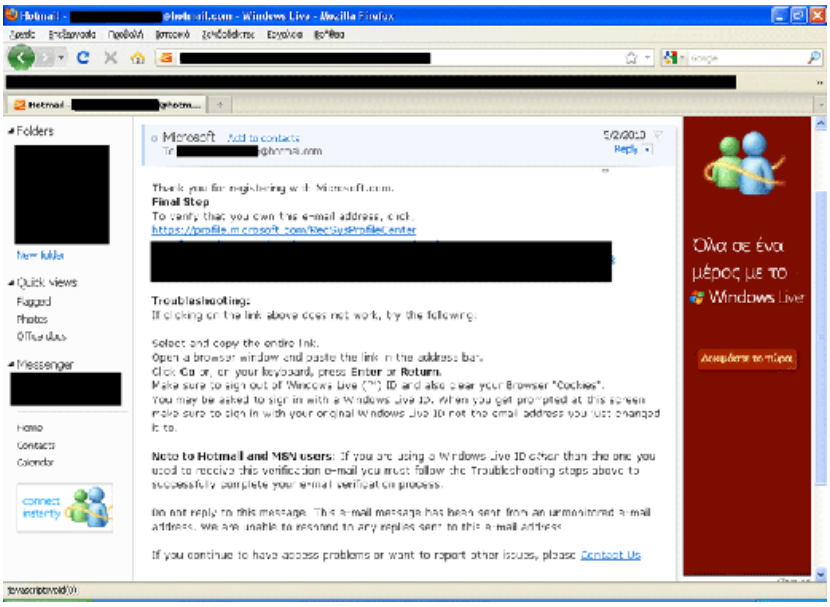


Επιλέγετε Register Now...

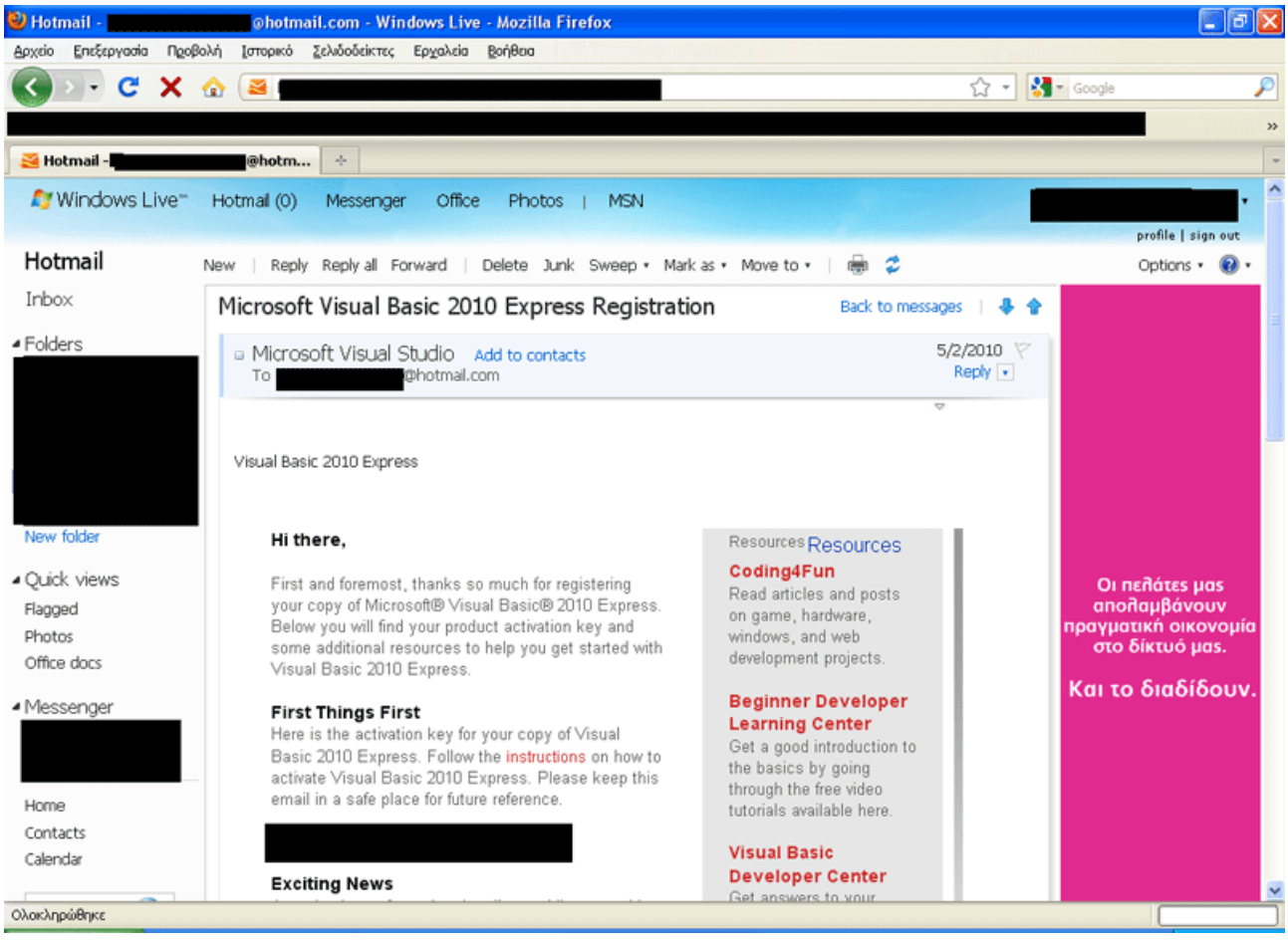


Φτιάχνετε αν δεν έχετε, ένα Windows Live id, εμείς είχαμε hotmail και δεν είχαμε πρόβλημα, και μέσω αυτού τη δηλώνετε.

Θα σας έρθει ένα μήνυμα, που θα σας ζητά πιθανόν να κάνετε κλικ σε ένα σύνδεσμο, για να σας στείλουν το κλειδί. Βεβαιωθείτε ότι ήρθε από τη Microsoft και ότι πρόκειται γι' αυτό που ζητήσατε, και πατήστε το σύνδεσμο.



Το κλειδί σας δίνεται. Γράψτε το και φυλάξτε το σε ασφαλές μέρος.



Πλέον έχετε μια στοιχειώδη εξοικείωση με το περιβάλλον που θα εργασθούμε. Το επόμενο βήμα, είναι να εμβαθύνουμε λίγο έτσι ώστε να προχωρήσουμε σε πιο πολύπλοκα προγράμματα. Ωστόσο πρώτα πρέπει να σας δώσουμε κάποιες βάσεις. Αυτό ακριβώς θα κάνουμε στο επόμενο κεφάλαιο.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό διδάσκονται επιπλέον λειτουργίες του γραφικού περιβάλλοντος ανάπτυξης, ενώ δίνονται οδηγίες για το πώς μπορεί να δηλωθεί η Visual Basic Express στη Microsoft, ώστε να μην έχει περιορισμένο χρόνο χρήσης.

ΚΕΦΑΛΑΙΟ 6

ΤΟ ΔΥΑΔΙΚΟ ΣΥΣΤΗΜΑ

ΥΠΑΡΧΟΥΝ 10 ΕΙΔΗ ΑΝΘΡΩΠΩΝ! ΑΥΤΟΙ ΠΟΥ ΞΕΡΟΥΝ ΤΟ ΔΥΑΔΙΚΟ ΣΥΣΤΗΜΑ ΚΙ ΑΥΤΟΙ ΠΟΥ ΔΕΝ ΤΟ ΞΕΡΟΥΝ!

Αυτή η φράση, αν δεν ξέρετε δυαδικό σύστημα, ίσως να σας κάνει να αναρωτιέστε, ποια είναι τα άλλα 8 είδη ανθρώπων. Ωστόσο, αφότου διαβάσετε το κεφάλαιο, το νόημά της θα αλλάξει ριζικά. Μία από τις δυσκολότερες προκλήσεις, όσον αφορά τη διδασκαλία πληροφορικής, είναι η επεξήγηση του δυαδικού συστήματος, με τρόπο απλό και κατανοητό.

Ας πάρουμε λοιπόν ένα παράδειγμα απλό, ώστε να προσεγγίσουμε το θέμα βήμα βήμα. Η Ελληνική γλώσσα έχει 24 γράμματα. Με αυτά έχουν δημιουργηθεί χιλιάδες λέξεις, οι οποίες σωστά δομημένες δημιουργούν προτάσεις. Ας πάρουμε λέξεις που αναπαριστούν κάτι συγκεκριμένο. Δηλαδή για παράδειγμα: μολύβι, πληκτρολόγια, μητέρα, γάτα, σκύλος, βιβλίο κλπ...Φανταστείτε τώρα η γλώσσα μας, να μην είχε 24 γράμματα αλλά μόνο 2!!!

Θα μας πείτε τώρα, μα πώς είναι δυνατόν; Και θα έχετε δίκιο. Θα ήταν απαράδεκτο κι ίσως εφιαλτικό η γλώσσα μας να είχε μόνο δύο γράμματα, αλλά φανταστείτε το. Έστω ότι τα γράμματα ήταν το Α και το Β. Πώς θα αναπαραστούσαμε τις λέξεις τότε; Έστω Α σημαίνει "μητέρα". Έστω Β σημαίνει "πατέρας" Έστω ΑΒ σημαίνει "κόρη". Έστω ΒΑ σημαίνει "γιος". Έστω ΑΑ σημαίνει "νερό". Έστω ΒΒ σημαίνει "γη".Έστω ΑΑΑ σημαίνει "θάλασσα". Έστω ΑΒΒ σημαίνει "νησί"....Παρατηρείτε τι κάνουμε; Χρησιμοποιούμε συνδυασμούς αυτών των δύο γραμμάτων για να αναπαραστήσουμε τα πάντα. Αν είχαμε λοιπόν μόνο δύο γράμματα για να αναπαραστήσουμε τα πάντα, ίσως μια λέξη να ήταν:

ΑΒΒΑΒΑΒΑΒΑΒΒΒΑΑΒΑΒΑΒΑΑΑΒΑΑΒΑΑΑΑΒ

Μπορείτε να μας φανταστείτε να μιλάμε, λέγοντας κάθε φορά, για κάθε λέξη, κάτι σαν το παραπάνω; Αν ναι, πολύ πιθανό είναι να γελάτε αυτή τη στιγμή. Έτσι όμως θα ήταν τα πράγματα, αν κάτι μας περιόριζε στο να έχουμε μόνο δύο γράμματα στη γλώσσα μας. Τέτοιος περιορισμός, ευτυχώς δεν υφίσταται στη φυσική γλώσσα που χρησιμοποιούν οι άνθρωποι, αλλά ισχύει όμως στα ηλεκτρονικά κυκλώματα και άρα και στους ηλεκτρονικούς υπολογιστές.

Τα δεδομένα και οι εντολές σε έναν ηλεκτρονικό υπολογιστή, είναι ουσιαστικά, ψηφιακοί "τετραγωνικοί" παλμοί. Στην παρακάτω εικόνα, ένας ιδεώδης ψηφιακός παλμός, αλλαγής τάσης (volt), είναι οι διακεκομμένες γραμμές. Μια πιο ρεαλιστική αναπαράσταση του τι συμβαίνει στην πραγματικότητα, μας δείχνει η μπλε συνεχής γραμμή.

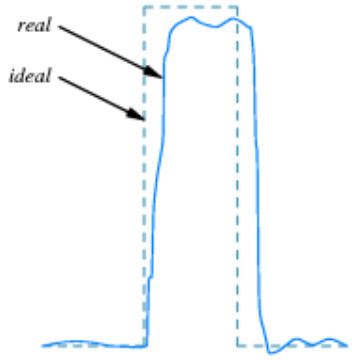


Figure 5.4 An illustration of the voltage emitted by a real device as it transmits a bit. In practice, voltages are often worse than this example.

Η αρχιτεκτονική των ηλεκτρονικών υπολογιστών λοιπόν, τους επιτρέπει να αντιλαμβάνονται μόνο δύο τάσεις. Δηλαδή κάτι σαν το Α και Β. Αν το Α το κάνουμε 1 και το Β το κάνουμε 0 έχουμε το δυαδικό σύστημα.

Στην παρακάτω εικόνα, βλέπετε ένα πρότυπο μετάδοσης δεδομένων, το πρότυπο RS-232. Έχει δύο τάσεις: -15 Volt για αναπαράσταση του 1 και +15 Volt για αναπαράσταση του 0. Έτσι μπορούν να αναπαρασταθούν δεδομένα με χρήση δύο ψηφίων, του 1 και του 0. Εδώ πρέπει να σας αναφέρουμε, ότι για χάρη απλούστευσης, λένε πως 0= δεν περνάει ρεύμα και 1=περνάει ρεύμα. Αυτό δεν είναι αληθές. Πάντα περνάει ρεύμα από τα κυκλώματα του ηλεκτρονικού υπολογιστή. Απλώς έχουμε δύο διαφορετικές τάσεις. Μία τάση για 1 και μία άλλη για 0.

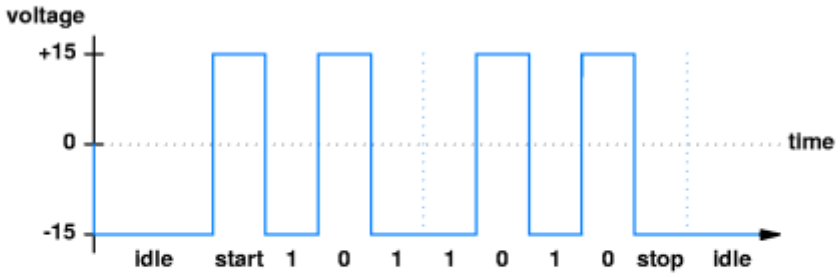


Figure 5.2 The voltage on a wire as a character is transmitted using RS-232†. A start bit notifies the receiver that a character is starting, and each bit transmission lasts the same length of time.

Τι είναι όμως το δυαδικό σύστημα;
Πρόκειται για ένα αριθμητικό σύστημα με βάση το 2. Εδώ κάπου λοιπόν, θα πρέπει να σας εξηγήσουμε τι εννοούμε.

Το δυαδικό σύστημα λέμε πως είναι ένα αριθμητικό σύστημα. Αριθμητικό σύστημα είναι ένα σύστημα από σύμβολα, που τα αποκαλούμε και ψηφία, με το οποίο αναπαριστούμε ποσοτικές έννοιες. Οι άνθρωποι χρησιμοποιούμε συνήθως το δεκαδικό σύστημα. Το πλήθος των ψηφίων που χρησιμοποιούμε στο δεκαδικό σύστημα, είναι δέκα. Έτσι με τα δέκα ψηφία 0,1,2,3,4,5,6,7,8,9 αναπαριστούμε ποσοτικά μεγέθη. Λέμε 50 άνθρωποι. 100 κιλά. 45 βήματα. 94 χρόνια. Ας πάρουμε έναν αριθμό του δεκαδικού συστήματος στην τύχη. Έστω για το παράδειγμά μας, παίρνουμε τον 3408. Το 3408 αναπαριστά 3 χιλιάδες και 4 εκατοντάδες και 0 δεκάδες και 8 μονάδες. Σωστά; Οι χιλιάδες, οι εκατοντάδες, οι δεκάδες και οι μονάδες όμως, μπορούν να εκφραστούν ως δυνάμεις του 10.
1000 = 103, 100 = 102, 10 = 101 και 1 = 100 σωστά;
Οπότε έχουμε:
(3 * 103) + (4 * 102) + (0 * 101) +(8 * 100) = 3000 + 400 + 0 + 8 = 3408
Υπενθυμίζουμε πως α0=1 και πως όταν εκφράζουμε μια δύναμη αν λέμε πως βάση της είναι το α. Λέμε δηλαδή πως έχουμε ως βάση το 10 στο δεκαδικό σύστημα.
Στην πρόσθεση, όταν φτάνουμε στο τελευταίο ψηφίο του συστήματος, δηλαδή το 9 εδώ στο δεκαδικό, αν προσθέσουμε 1, μηδενίζουμε το παρόν ψηφίο και προσθέτουμε 1 στο αμέσως πιο αριστερό του.
9+1 = 10
89+1 = 90

Ομοίως εργαζόμαστε και με το δυαδικό σύστημα.
Στο δυαδικό σύστημα η βάση μας είναι το 2. Όπότε κάθε αριθμός συμβολίζεται με δυνάμεις του 2. Οπότε στο δυαδικό σύστημα δεν έχουμε μονάδες, δεκάδες κι εκατοντάδες.
ΕΧΟΥΜΕ:
$2^0 = 1$ ΜΟΝΑΔΕΣ
$2^1 = 2$ ΔΥΑΔΕΣ
$2^2 = 4$ ΤΕΤΡΑΔΕΣ
$2^3 = 8$ ΟΚΤΑΔΕΣ
$2^4 = 16$ ΔΕΚΑΕΞΑΔΕΣ
$2^5 = 32$ ΤΡΙΑΝΤΑΔΥΑΔΕΣ
$2^6 = 64$ ΕΞΗΝΤΑΤΕΤΡΑΔΕΣ
$2^7 = 128$ ΕΚΑΤΟΝΕΙΚΟΣΙΟΚΤΑΔΕΣ
...και τα λοιπά.

Τα ψηφία που έχουμε είναι δύο. Το 0 και το 1. Στο δεκαδικό δε φτάναμε ποτέ τη βάση (το 10), θυμάστε; Στο 89 αν προσθέταμε 1 μονάδα, μηδένιζαν οι μονάδες και προσθέταμε μία δεκάδα κι έτσι είχαμε 90. 89 +1 δε γινόταν με άλλα λόγια 810... αλλά 90.
Οπότε στο δυαδικό σύστημα 1 + 1 = 10. Οι μονάδες μηδενίζουν (φτάσαμε τη βάση που είναι το 2) και οι δυάδες (το αμέσως πιο αριστερό ψηφίο) αυξάνεται κατά 1.
Αριστερά αν δεν υπάρχουν άλλα ψηφία, εννοούνται όσα μηδενικά θέλουμε, όπως και στο δεκαδικό. Δηλαδή όπως στο δεκαδικό 00000122 = 122 έτσι και στο δυαδικό 00000010 = 10
Ο παρακάτω πίνακας είναι ένας πίνακας που χρησιμοποιούμε, για να μετατρέπουμε δεκαδικούς αριθμούς σε δυαδικούς και αντίστροφα. Η κάτω γραμμή έχει τις δυνάμεις του 2. Η μεσαία έχει την ισοδυναμία της εκάστοτε δύναμης. Η πάνω γραμμή είναι κενή για να γεμίσει με ψηφία 1 ή 0.

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Έστω τώρα πως έχω το δυαδικό αριθμό 10101011 και θέλω να τον μετατρέψω σε δεκαδικό. Τον τοποθετώ στον πίνακα, όπως φαίνεται στην παρακάτω εικόνα. Αμέσως φαίνεται πως έχω να προσθέσω μία εκατονεικοσιοκτάδα, μία τρανταδυάδα, μία οκτάδα, μία δυάδα και μία μονάδα. Οπότε το 10101011 σε δυαδικό σύστημα είναι ίσο με 161 στο δεκαδικό σύστημα. Η παρακάτω εικόνα μας δείχνει συγκεντρωμένη την όλη διαδικασία μετατροπής δυαδικού σε δεκαδικό αριθμό.

1	0	1	0	1	0	1	1
128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

(1 x 128) + (0 x 64) + (1 x 32) + (0 x 16) + (1 x 8) + (0 x 4) + (1 x 2) +(1 x 1) =

128 + 0 + 32 + 0 + 8 + 0 + 2 + 1 = 161

ΑΡΑ ΤΟ ΔΥΑΔΙΚΟ 10101011 ΙΣΟΥΤΑΙ ΜΕ ΤΟ ΔΕΚΑΔΙΚΟ 161

Τι γίνεται όμως αν θέλω έναν δεκαδικό να τον μετατρέψω σε δυαδικό; Παίρνω πάλι τον πίνακα. Έστω θέλω να μετατρέψω σε δυαδικό τον 120. Ξεκινάω από τα αριστερά προς τα δεξιά. Ποια είναι η πιο αριστερή δύναμη; Είναι το 27 οπότε το 128!
Ρωτάω τον εαυτό μου. Το 128 στο 120 χωράει; Όχι. Οπότε συμπληρώνω από πάνω του 0! Γενικά, για δική σας διευκόλυνση, να θυμάστε ότι 0 σημαίνει και "ψευδές" ή "όχι". Αντιθέτως 1 σημαίνει και "αληθές" ή "ναι". Κινούμαι μια θέση δεξιά. Το 64 στο 120 χωράει; Ναι. Οπότε συμπληρώνω από πάνω του 1 και αφαιρώ από το 120 το 64. Υπόλοιπο 56! Πάμε ομοίως μια θέση πιο δεξιά. Το 32 στο 56 χωράει; Ναι. Οπότε συμπληρώνω από πάνω του 1 και αφαιρώ από το 56 το 32. Υπόλοιπο 24! Πάμε ομοίως μια θέση πιο δεξιά. Το 16 στο 24 χωράει; Ναι. Οπότε συμπληρώνω από πάνω του 1 και αφαιρώ από το 24 το 16. Υπόλοιπο 8! Πάμε ομοίως μια θέση πιο δεξιά. Το 8 στο 8 χωράει; Ναι. Οπότε συμπληρώνω από πάνω του 1 και αφαιρώ από το 8 το 8. Υπόλοιπο 0! Πάμε ομοίως μια θέση πιο δεξιά. Το 4 στο0 χωράει; Όχι. Οπότε συμπληρώνω από πάνω του 0. Πάμε ομοίως μια θέση πιο δεξιά. Το 2 στο0 χωράει; Όχι. Οπότε συμπληρώνω από πάνω του 0. Πάμε ομοίως μια θέση πιο δεξιά. Το 1 στο0 χωράει; Όχι. Οπότε συμπληρώνω από πάνω του 0. Τελείωσα και το υπόλοιπό μου είναι μηδέν. Έκανα λοιπόν σωστά τη μετατροπή. Οτιδήποτε ως υπόλοιπο εκτός από μηδέν, σημαίνει πως κάπου εχω κάνει λάθος. Στην παρακάτω εικόνα βλέπετε συγκεντρωτικά τη διαδικασία.

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

ΕΣΤΩ ΕΧΩ ΤΟ 120 ΚΑΙ ΘΕΛΩ ΝΑ ΤΟ ΚΑΝΩ ΔΥΑΔΙΚΟ.

ΑΠΟ ΑΡΙΣΤΕΡΑ ΠΡΟΣ ΤΑ ΔΕΞΙΑ

ΤΟ 128 ΣΤΟ 120 ΧΩΡΑΕΙ; ΟΧΙ. ΜΗΔΕΝ ΠΑΡΑΜΕΝΕΙ 120
ΤΟ 64 ΣΤΟ 120 ΧΩΡΑΕΙ; ΝΑΙ! ΕΝΑ 120 – 64 = 56
ΤΟ 32 ΣΤΟ 56 ΧΩΡΑΕΙ; ΝΑΙ! ΕΝΑ 56 – 32 = 24
ΤΟ 16 ΣΤΟ 24 ΧΩΡΑΕΙ; ΝΑΙ! ΕΝΑ 24 - 16 = 8
ΤΟ 8 ΣΤΟ 8 ΧΩΡΑΕΙ; ΝΑΙ! ΕΝΑ 8 - 8 = 0
ΤΟ 4 ΣΤΟ 0 ΧΩΡΑΕΙ; ΟΧΙ. ΜΗΔΕΝ ΠΑΡΑΜΕΝΕΙ 0
ΤΟ 2 ΣΤΟ 0 ΧΩΡΑΕΙ; ΟΧΙ. ΜΗΔΕΝ ΠΑΡΑΜΕΝΕΙ 0
ΤΟ 1 ΣΤΟ 0 ΧΩΡΑΕΙ; ΟΧΙ. ΜΗΔΕΝ ΠΑΡΑΜΕΝΕΙ 0

ΥΠΟΛΟΙΠΟ = 0 ΣΩΣΤΑ!

ΜΗΔΕΝ, ΕΝΑ, ΕΝΑ, ΕΝΑ, ΕΝΑ ΜΗΔΕΝ, ΜΗΔΕΝ, ΜΗΔΕΝ

ΑΡΑ 120 ΔΕΚΑΔΙΚΟ ΕΙΝΑΙ ΙΣΟ ΜΕ 01111000 ΔΥΑΔΙΚΟ

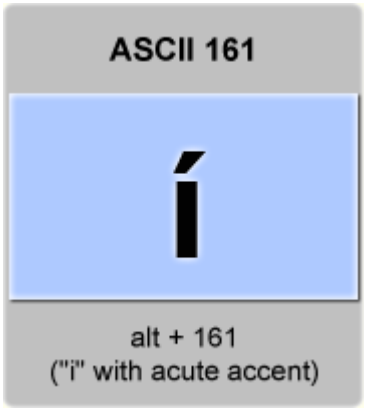
Η επαλήθευση είναι απλή. Πολλαπλασιάστε τα μηδέν και ένα με τους αντίστοιχους αριθμούς και προσθέστε τα γινόμενα. Στην παρακάτω εικόνα, το κάναμε παραλείποντας γινόμενα με το μηδέν, μιας και δεν έχει νόημα να προσθέσουμε το τίποτα. Όντως βγαίνει 120!

0	1	1	1	1	0	0	0
128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

64 + 32 + 16 + 8 =120

Γιατί σας τα είπαμε όλα αυτά όμως; Ο ηλεκτρονικός υπολογιστής, αντιλαμβάνεται τα δεδομένα και τις εντολές, με τη μορφή ομάδων από 0 και 1. Ένα ψηφίο 0 ή ένα ψηφίο 1, αποκαλείται bit (από το Binary digit) που στα Αγγλικά σημαίνει ψηφιακό στοιχείο. Δε θέλουμε να εμβαθύνουμε σε δυαδικής μορφής εντολές εδώ. Θέλουμε όμως να σας δείξουμε τον τρόπο που ο ηλεκτρονικός υπολογιστής αντιλαμβάνεται, αποθηκεύει, μεταφέρει κι επεξεργάζεται τα διάφορα δεδομένα που του δίνουμε. Κοινώς, όταν πατάτε στο πληκτρολόγιο το γράμμα Α, πώς αντιλαμβάνεται ο υπολογιστής ότι είναι το Α και το εμφανίζει στην οθόνη; Αυτό είναι πολύ σημαντικό να το κατανοήσετε, διότι αποκτώντας αυτές τις γνώσεις, θα μπορέσουμε να σας εξηγήσουμε στη συνέχεια μερικά βασικά πράγματα για τις μεταβλητές, ώστε να μπορέσουμε κατόπιν να προχωρήσουμε στο να φτιάξουμε ένα πιο ενδιαφέρον και χρήσιμο πρόγραμμα. Είπαμε τι είναι bit. Φανταστείτε τώρα ότι ο υπολογιστής αντιλαμβάνεται ομάδες των 8bit. Ο πίνακας που χρησιμοποιήσαμε έχει 8 θέσεις. Ο αριθμός 10101011 λοιπόν, καταλαμβάνει χώρο της τάξεως των 8 bit. Να θυμάστε λοιπόν τώρα ότι 8 bit = 1 byte. Με 8 bit, μπορούμε να έχουμε 256 διαφορετικούς δυαδικούς αριθμούς.
128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255.
Συν ένα χαρακτήρα το 0. 256 χαρακτήρες!

Οι κατασκευαστές ηλεκτρονικών υπολογιστών αλλά και οι προγραμματιστές, συμφώνησαν στο να δημιουργήσουν κάποια πρότυπα αντιστοίχισης. Ένα από αυτά λοιπόν τα πρότυπα, είναι το: American Standard Code for Information Interchange (ASCII) το οποίο αντιστοιχίζει χαρακτήρες όπως αριθμούς, γράμματα, σύμβολα, με αριθμούς από το 0 ως το 255. Όταν λοιπόν θέλουμε να εμφανιστεί για παράδειγμα το "ι" το Ελληνικό με τόνο από επάνω σε ASCII, αυτό πού δίνουμε στον ηλεκτρονικό υπολογιστή, είναι το δυαδικό ισοδύναμο του αριθμού 161, στον οποίο αντιστοιχεί το γράμμα "ί".



Οπότε ένα σύμβολο απαιτεί 1 byte για να αποθηκευτεί. Ανάλογα με τον τύπο δεδομένων που θα επεξεργαστούμε, στον προγραμματισμό χρησιμοποιούμε δοχεία, που θα γεμίσουν με bytes αυτών των δεδομένων. Αυτά τα δοχεία λέγονται μεταβλητές και αναλύονται στο επόμενο κεφάλαιο.
Πριν όμως τελειώσουμε με αυτό το κεφάλαιο, να υπενθυμίσουμε κάτι. Στο δεκαδικό σύστημα 000145 = 145 σωστά; Όσα μηδενικά και να βάλουμε στην αρχή, μπορούμε να τα παραλείψουμε. Ομοίως και στο δυαδικό. 00000010 = 10 (=2 δεκαδικό)

Ξαναδιαβάστε τη φράση τώρα...

ΥΠΑΡΧΟΥΝ 10 ΕΙΔΗ ΑΝΘΡΩΠΩΝ!
ΑΥΤΟΙ ΠΟΥ ΞΕΡΟΥΝ ΤΟ ΔΥΑΔΙΚΟ ΣΥΣΤΗΜΑ ΚΙ ΑΥΤΟΙ ΠΟΥ ΔΕΝ ΤΟ ΞΕΡΟΥΝ!

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό επιχειρούμε να διδάξουμε στο μαθητευόμενο τι είναι το δυαδικό σύστημα, καθώς και να τον καταστήσουμε ικανό να κάνει και μετατροπές εύκολα και σχετικά γρήγορα. Στη συνέχεια τον εισάγουμε στα μεγέθη bit και byte και του επεξηγούμε πώς όλα αυτά εφαρμόζονται σε έναν ηλεκτρονικό υπολογιστή, με μία αντιστοίχιση σε ASCII κώδικα. Το κεφάλαιο ξεκινάει με μία φράση – λογοπαίγνιο, την όποια στο τέλος του κεφαλαίου ο μαθητευόμενος θα κατανοήσει, λόγω της γνώσης που απέκτησε, γεγονός που στοχεύει στην ευχαρίστησή του.

ΚΕΦΑΛΑΙΟ 7
ΤΙ ΕΙΝΑΙ ΟΙ ΜΕΤΑΒΛΗΤΕΣ;

Φανταστείτε πως οι μεταβλητές είναι σαν δοχεία. Έχουν διαφορετικά μεγέθη και είναι σχεδιασμένες για να χωρέσουν συγκεκριμένα δεδομένα και πληροφορίες. Εδώ θα ασχοληθούμε με τρία βασικά είδη μεταβλητών, για να κατανοήσουμε το τι είναι μεταβλητή. Οι τρεις τύποι που θα ασχοληθούμε είναι οι: INTEGER, DOUBLE και STRING.

Οι μεταβλητές INTEGER, είναι δοχεία που χωράνε ακεραίους αριθμούς. Μπορούν αν γίνουν πράξεις με άλλους αριθμούς. Το μέγεθός τους είναι 4 bytes (32bit).
Παίρνουν τιμές από το -2,147,483,648 ως το 2,147,483,647

Οι μεταβλητές DOUBLE είναι δοχεία που χωράνε δεκαδικούς αριθμούς, με υποδιαστολή. Μπορούν αν γίνουν πράξεις με άλλους αριθμούς. Το μέγεθός τους είναι 8 bytes (64bit). Παίρνουν τιμές από -1.79769313486231E308 ως -4.94065645841247E-324 για αρνητικές τιμές. Από 4.94065645841247E-324 ως 1.79769313486232E308 για θετικές τιμές.

Οι μεταβλητές STRING είναι δοχεία που χωράνε κείμενο. Όταν λέμε κείμενο, εννοούμε γράμματα, σύμβολα, αριθμούς. Οι αριθμοί μέσα σε κείμενο, δε δύναται να πάρουν μέρος σε μαθηματικές πράξεις. Το μέγεθός τους είναι 10 bytes + (2 επί το μήκος της συμβολοσειράς). Μήκος της συμβολοσειράς είναι το πλήθος των χαρακτήρων στο κείμενο, (συμπεριλαμβανομένων των κενών). Παίρνει τιμές από κανέναν ως περίπου 2 ΔΙΣ χαρακτήρες προτυποποίησης Unicode. Η προτυποποίηση Unicode είναι μια ακόμα προτυποποίηση, σαν το πρότυπο ASCII.

Με μία πρώτη ματιά, παρατηρούμε ότι η STRING μεταβλητή είναι η μεγαλύτερη. Η αμέσως πιο μικρή ειναι η DOUBLE. Η μικρότερη από τις τρεις είναι η INTEGER. Οπότε κάποια δοχεία είναι μεγαλύτερα από άλλα δοχεία. Γιατί όμως; Είναι απλό. Ένας ακέραιος έχει κάποια ψηφία. Τα ίδια ψηφία, συν ένα σύμβολο, την υποδιαστολή, συν δεκαδικά ψηφία, έχει ο δεκαδικός. Η συμβολοσειρά, είναι ολόκληρο κείμενο.

Για να το κάνουμε τώρα πιο κατανοητό. Υποθέστε πως έχουμε ένα μήλο, ένα ποτήρι νερού, μία κανάτα και μία κατσαρόλα. Το μήλο έστω πως είναι ένας δεκαδικός αριθμός. Το ποτήρι είναι η μεταβλητή INTEGER. Η κανάτα είναι η μεταβλητή DOUBLE. Η κατσαρόλα είναι η μεταβλητή STRING.

Το μήλο είπαμε είναι δεκαδικός, οπότε χωράει σε μια μεταβλητή DOUBLE, δηλαδή την κανάτα.

Ωστόσο ΔΕ ΧΩΡΑΕΙ σε μια μεταβλητή INTEGER, που συμβολίζουμε με το ποτήρι, διότι η INTEGER δεν έχει χώρο για το δεκαδικό μέρος ή την υποδιαστολή.

Όμως κάθε ακέραιος, μπορεί να εκφραστεί σαν δεκαδικός με υποδιαστολή και μηδενικά στο δεκαδικό μέρος. Δηλαδή ο αριθμός 4564 είναι ο ίδιος με τον 4564,00000000. Σωστά;
Οπότε όλοι οι ακέραιοι INTEGER, χωράνε στους DOUBLE. Όπως βλέπετε, ολόκληρο το ποτήρι (INTEGER), χωράει στην κανάτα (DOUBLE).

Ομοίως η κανάτα (DOUBLE), άρα και το ποτήρι (INTEGER), χωράνε μέσα στην κατσαρόλα (STRING)!

Όντως η μεταβλητή τύπου συμβολοσειρά (STRING), μπορεί να χωρέσει περίπου 2 ΔΙΣ σύμβολα!

Ωστόσο, είναι αδύνατον να χωρέσει η κατσαρόλα στην κανάτα!!! Οπότε μια συμβολοσειρά, δε χωράει σε μία μεταβλητή τύπου (DOUBLE) ή (INTEGER)!!!

Μπορεί να φαίνεται πολύ αστείο όπως το παρουσιάζουμε, αλλά είναι πολύ σημαντικό. Πρέπει να ξέρετε που, πως και ποιες μεταβλητές, πρέπει να χρησιμοποιείτε. Θα καταλάβετε γιατί, στα επόμενα κεφάλαια. Αυτοί οι τρεις τύποι δεν είναι οι μόνοι τύποι μεταβλητών.

Παρακάτω σας παραθέτουμε τους υπόλοιπους τύπους δοχείων.

Boolean
Μέγεθος: 2 bytes
Τιμές: True ή False

Byte
Μέγεθος:1 byte
Τιμές: 0 ως 255

Date
Μέγεθος: 8 bytes
Τιμές:January 1, 1 Ελάχιστη δυνατή χρονολογία μηχανήματος ως December 31, 9999

Char
Μέγεθος: 2 bytes
Τιμές: Ένας κωδικός χαρακτήρα από το 0 ως το 65,535

Decimal
Μέγεθος: 12 bytes
Τιμές:
+/-79,228,162,514,264,337,593,543,950,335 δίχως υποδιαστολή
7.9228162514264337593543950335 με 28 δεκαδικά ψηφία.
Μικρότερη μη μηδενική τιμή: +/-0.0000000000000000000000000001

Long (long integer)
Μέγεθος: 8 bytes
Τιμές: -9,223,372,036,854,775,808 ως 9,223,372,036,854,775,807

Object
Μέγεθος: 4 bytes

Short
Μέγεθος: 2 bytes
Τιμές: Από -32,768 ως 32,767

Single (single precision floating point)
Μέγεθος: 4 bytes
Τιμές:
Από -3.402823E38 ως -1.401298E-45 για αρνητικές τιμές.
Από 1.401298E-45 ως 3.402823E38 για θετικές τιμές.

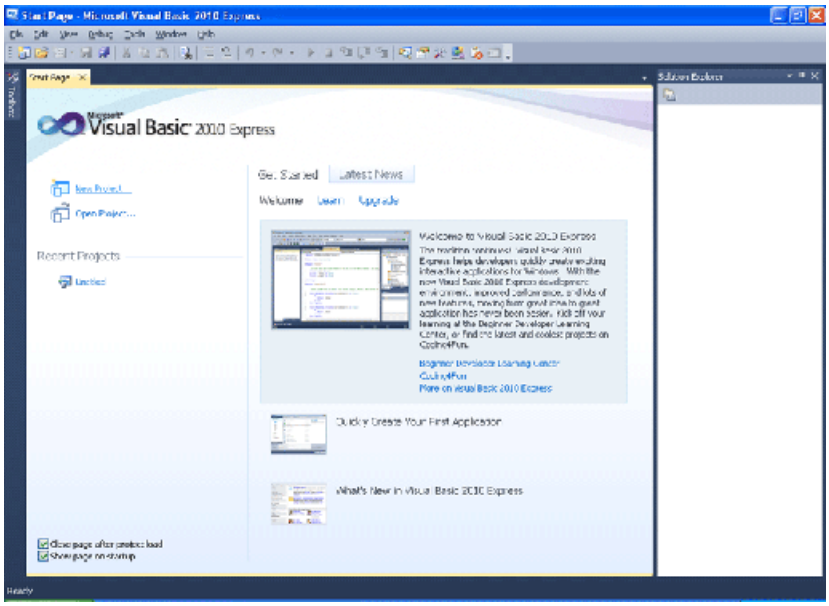
User-Defined Type (structure)
Δηλαδή δομές δεδομένων ορισμένες από το χρήστη (προγραμματιστή).
Μέγεθος: Ανάλογα τη δομή που ορίζει ο προγραμματιστής.
Ουσιαστικά δομή μπορεί να είναι ένας συνδυασμός των παραπάνω πρωτογενών μεταβλητών.
Στα τελευταία κεφάλαια, θα ασχοληθούμε με μια πολύ πρακτική δομή δεδομένων, τον πίνακα.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό, έχοντας ο μαθητευόμενος διδαχτεί δυαδικό σύστημα και έννοιες όπως bit και byte, τον εισάγουμε στην έννοια της μεταβλητής. Οι μεταβλητές στην πληροφορική, να διδαχθούν είναι δύσκολο. Ωστόσο αν μπορούσαμε να εφαρμόσουμε, με αναλογική μέθοδο, παράδειγμα του τι είναι μεταβλητές, παρομοιάζοντάς τες με αντικείμενα του φυσικού κόσμου, ίσως γινόταν ευκολότερο. Αυτή ήταν η προσέγγισή μας, οπότε χρησιμοποιήσαμε ένα μήλο, ένα ποτήρι, μια κανάτα και μια κατσαρόλα έτσι ώστε εκτός του να διασκεδαστικό, το παράδειγμα να μπορεί να μεταφέρει με σαφήνεια και καθαρότητα τις πληροφορίες που θέλαμε να μεταφέρουμε. Επεξηγούμε λοιπόν βασικούς τύπους μεταβλητών, ενώ παραθέτουμε και τους υπόλοιπους τύπους στο τέλος. Τη γενική εικόνα ο μαθητευόμενος την εκλαμβάνει και την καταλαβαίνει. Αν θέλει να δει τι είναι οι υπόλοιπες μεταβλητές από μόνος του δε θα δυσκολευτεί να κατανοήσει τους υπόλοιπους τύπους, οπότε δεν τον κουράζουμε και τους παραθέτουμε συνοπτικά.

ΚΕΦΑΛΑΙΟ 8
ΜΕΤΑΒΛΗΤΕΣ ΣΤΗΝ ΠΡΑΞΗ

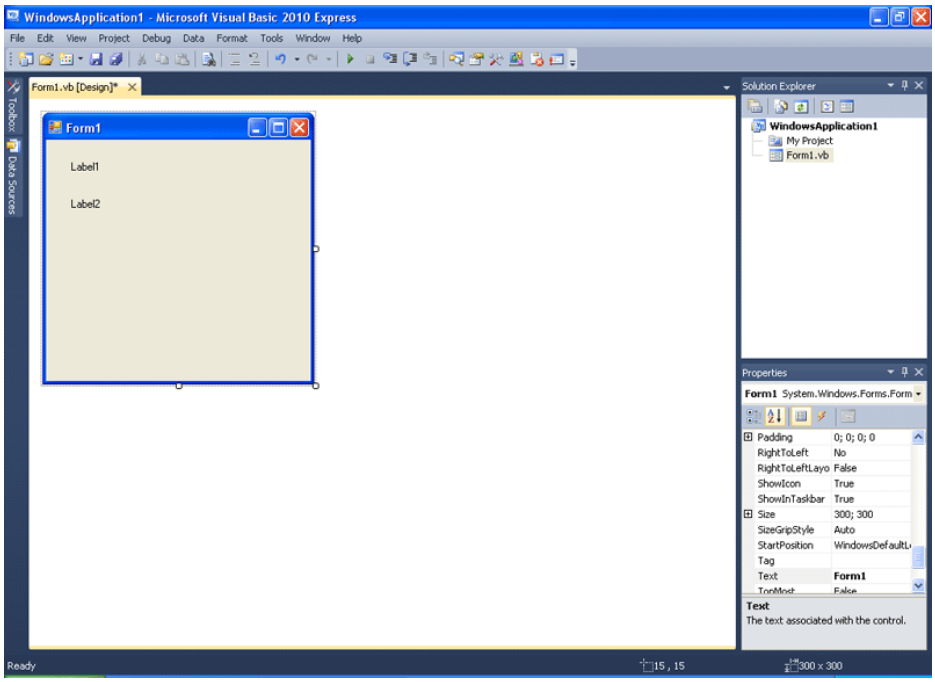
Μάθαμε λοιπόν τι είναι οι μεταβλητές. Μάθαμε επίσης πως υπάρχουν πολλοί τύποι μεταβλητών. Καιρός να φτιάξουμε βήμα βήμα, ένα απλό πρόγραμμα που χειρίζεται μεταβλητές. Ανοίξτε τη Visual Basic και επιλέξτε New Poject και κατόπιν, Windows Forms Application.



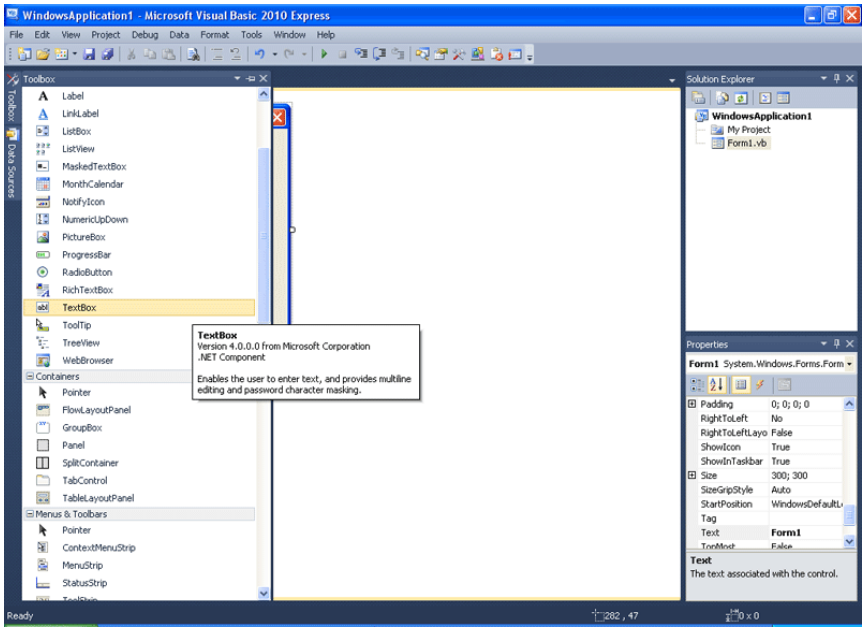
Προτού φτιάξουμε ένα πρόγραμμα, κανονικά θα πρέπει να ξέρουμε τι ακριβώς θέλουμε να κάνει, να το σχεδιάσουμε, να το μελετήσουμε κι έπειτα να αρχίσουμε να το υλοποιούμε. Ωστόσο, επιλέξαμε αυτό το θέμα να το αναλύσουμε προς το τέλος, πρώτον, για να έχετε ήδη μια γενική εικόνα της γλώσσας, και δεύτερον για να κατανοήσετε τη λογική ευκολότερα. Αργότερα, σε μεταγενέστερο κεφάλαιο, θα ασχοληθούμε με έναν πιο ορθό κι επαγγελματικό τρόπο να φτιάχνουμε προγράμματα. Εδώ όμως, πρώτον επειδή τα προγράμματα που φτιάχνουμε είναι απλά και δεύτερον επειδή εστιαζόμαστε στο να κατανοήσουμε άλλα θέματα, δεν αναλύουμε ούτε σχεδιάζουμε αλγορίθμους ή περιβάλλοντα. Ωστόσο πρέπει σιγά σιγά, να σας δίνουμε και μια ιδέα του πως να σκέφτεστε και του τι θα κάνουμε, από την αρχή της υλοποίησης.

Εδώ λοιπόν, στόχος μας είναι να φτιάξουμε ένα απλό πρόγραμμα, το οποίο θα αθροίζει δύο αριθμούς που ο χρήστης θα δίνει μέσα σε δύο πλαίσια. Το άθροισμα θα γίνεται όταν πατάει ο χρήστης το ανάλογο κουμπί και θα εμφανίζεται το αποτέλεσμα σε ένα άλλο πλαίσιο. Αυτό το πρόγραμμα θα το φτιάξουμε με δύο τρόπους. Στόχος είναι να καταλάβετε τη λειτουργία των μεταβλητών. Τώρα που ξέρουμε τι θέλουμε λοιπόν να κάνουμε, πάμε να το φτιάξουμε.

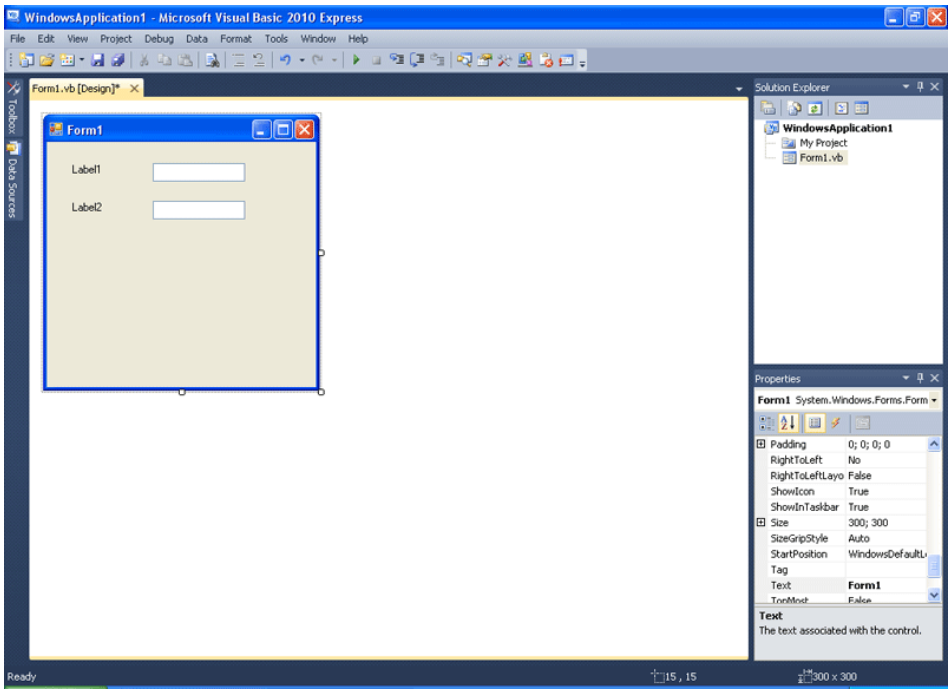
Στην αρχική φόρμα, φτιάξτε δύο Labels όπως δείχνει η παρακάτω εικόνα. Αυτές θα δίνουν στη συνέχεια στο χρήστη να καταλάβει τι πρέπει να κάνει.



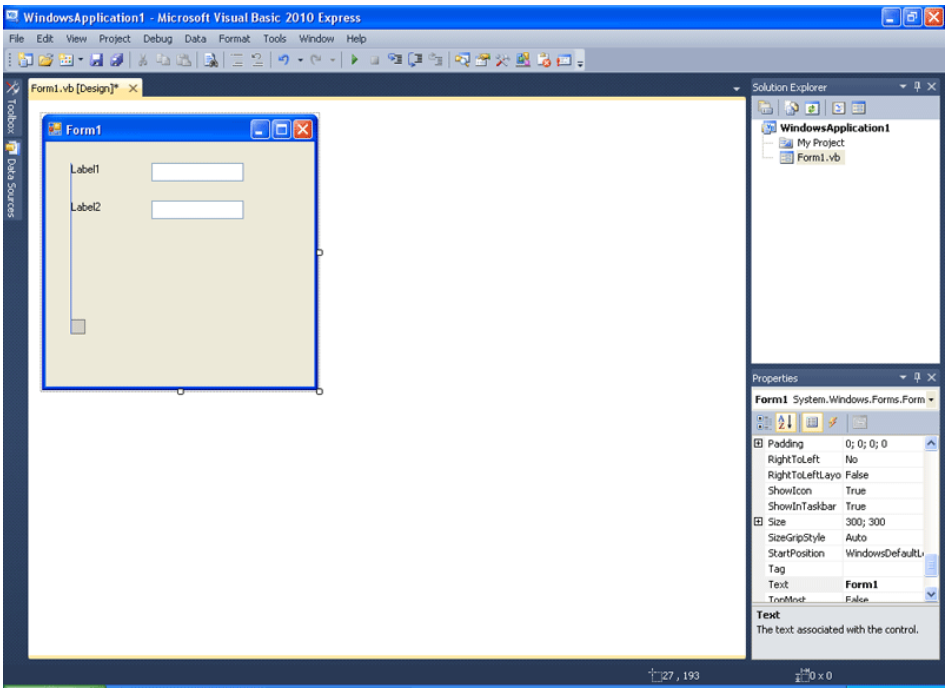
Πηγαίνετε στο toolbox και επιλέξτε το textbox όπως δείχνει η παρακάτω εικόνα.



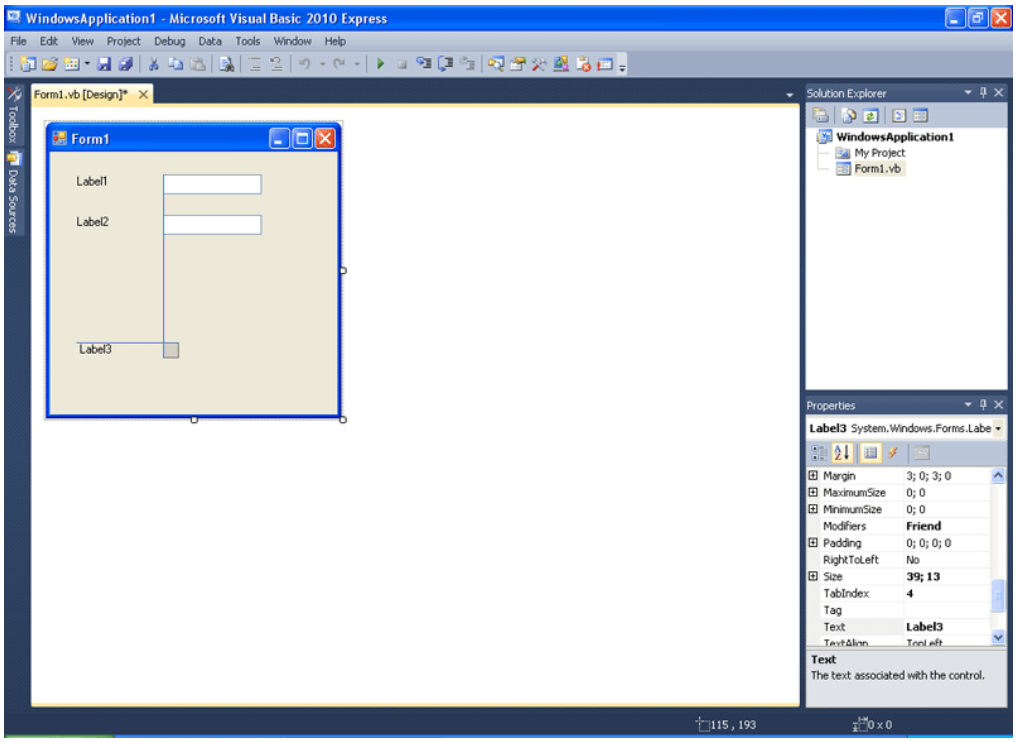
Τοποθετήστε δύο textbox δίπλα από τις labels. Αυτά θα είναι τα πλαίσια που θα βάζει ο χρήστης τους αριθμούς.



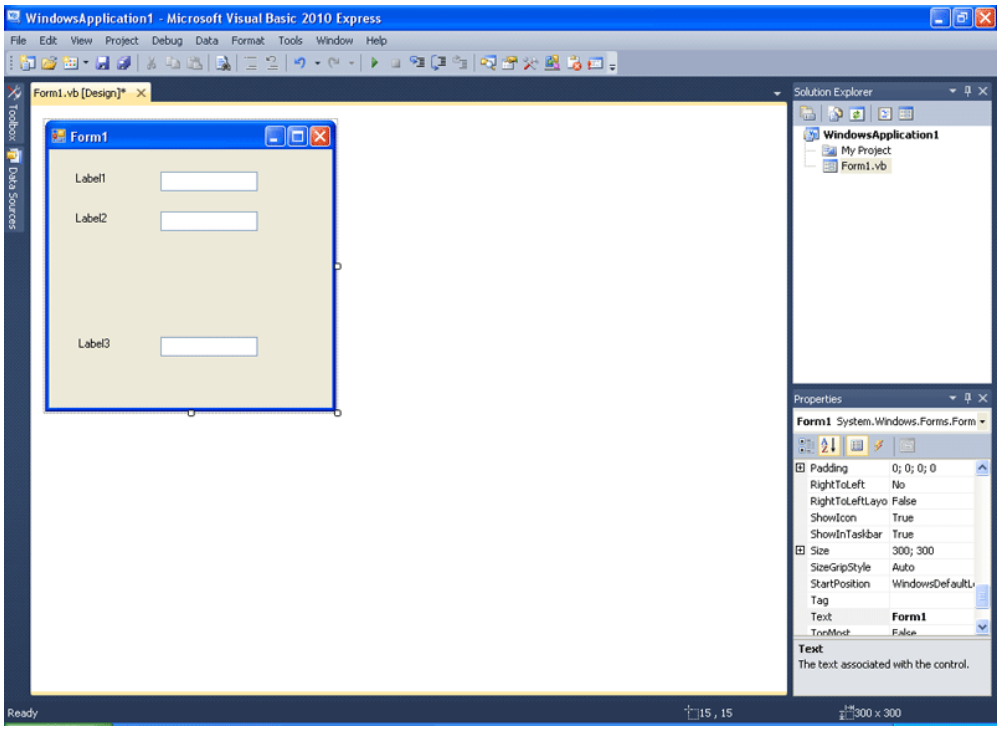
Επιλέξτε ακόμα μια label και καθώς έχετε πατημένο το κουμπί για να τη σύρετε στη φόρμα στοιχίστε τη όπως δείχνει η παρακάτω εικόνα, με τις άλλες δύο. Με αυτή θα εξηγούμε, πως εδώ, θα εμφανίζεται το αποτέλεσμα.



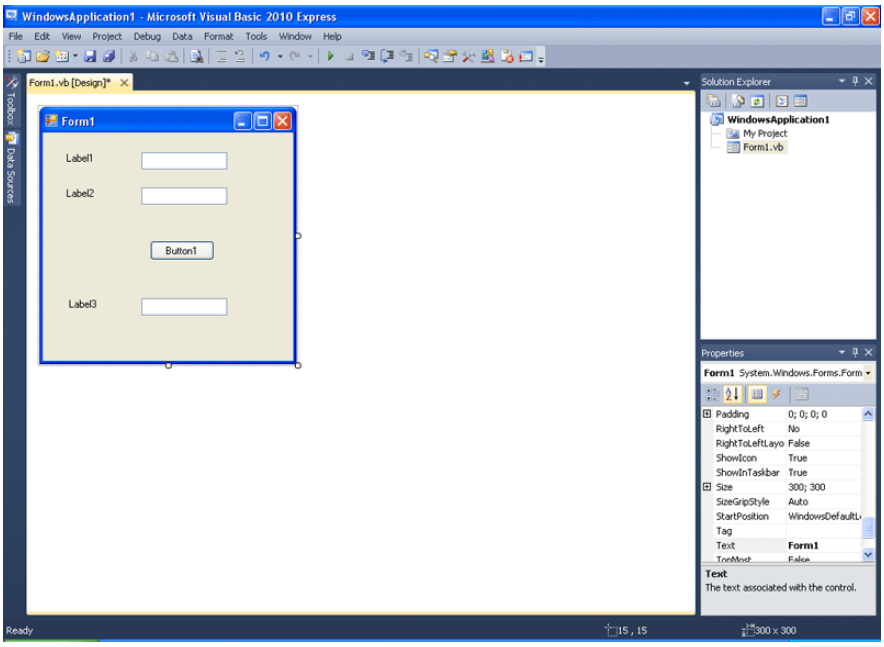
Με τον ίδιο τρόπο βάλτε κι ένα ακόμα textbox, αυτή τη φορά, δίπλα στη Label3. Εδώ μέσα θα εμφανίζεται το αποτέλεσμα.



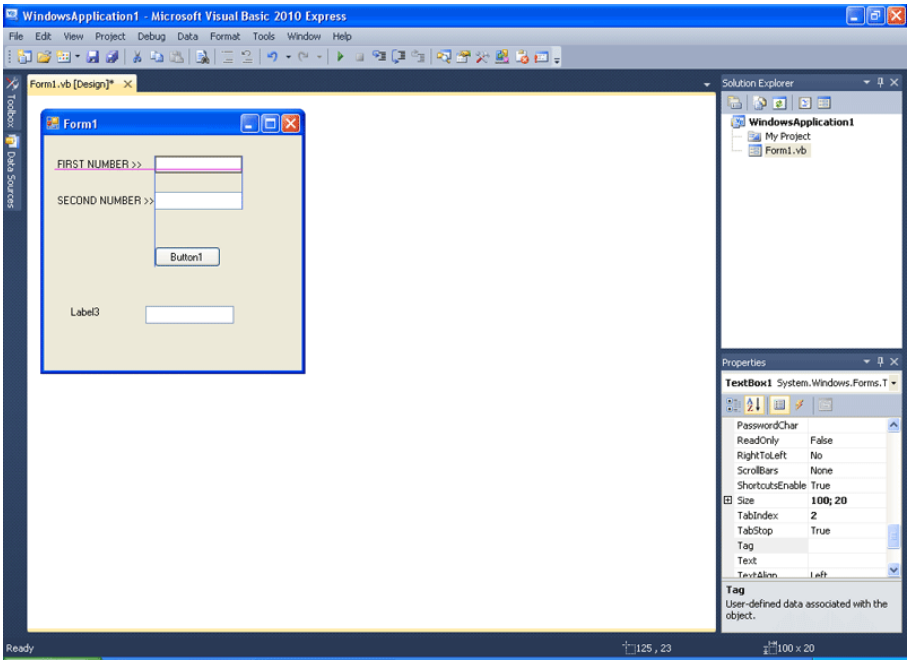
Το περιβάλλον του προγράμματος πρέπει να μοιάζει με την παρακάτω φωτογραφία.



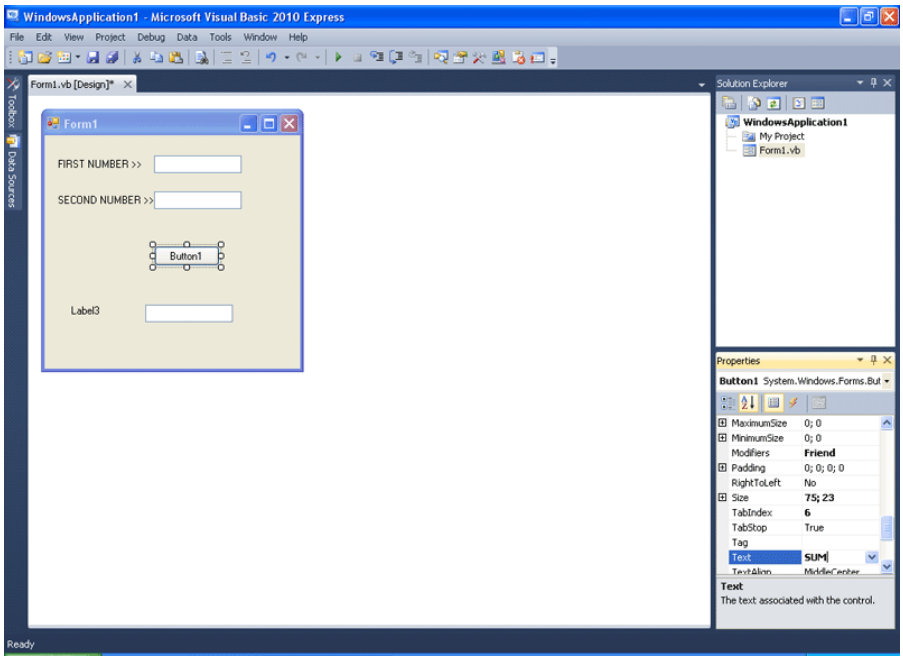
Προσθέστε ένα κουμπί. Καλά το μαντέψατε, με αυτό θα γίνεται το άθροισμα.



Στοιχίστε το στη φόρμα.

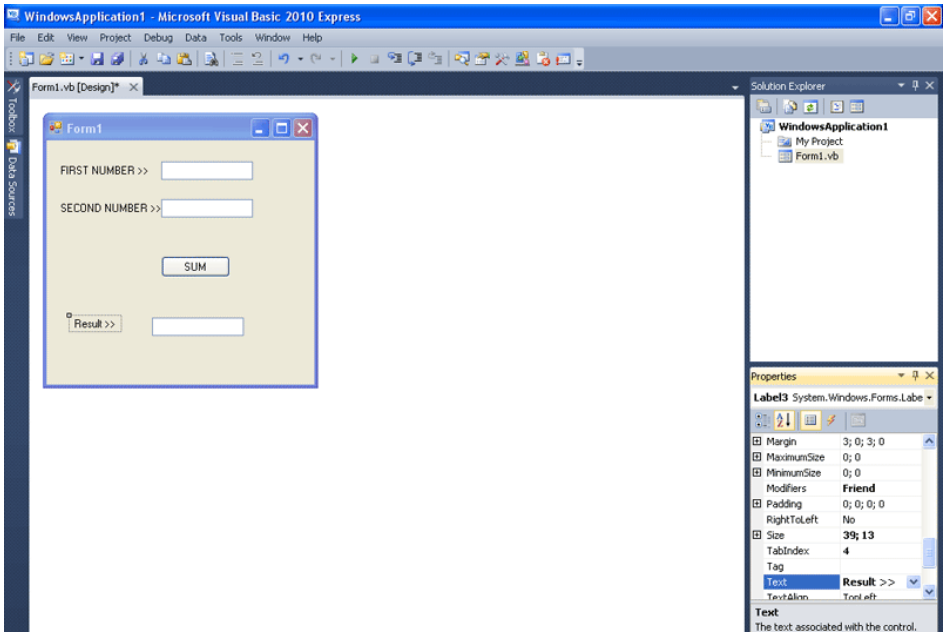


Πηγαίνετε κάτω δεξιά στις ιδιότητές του και επιλέξτε την ιδιότητα text. Αλλάξτε την τιμή της σε " SUM". SUM στα Αγγλικά σημαίνει πρόσθεση.

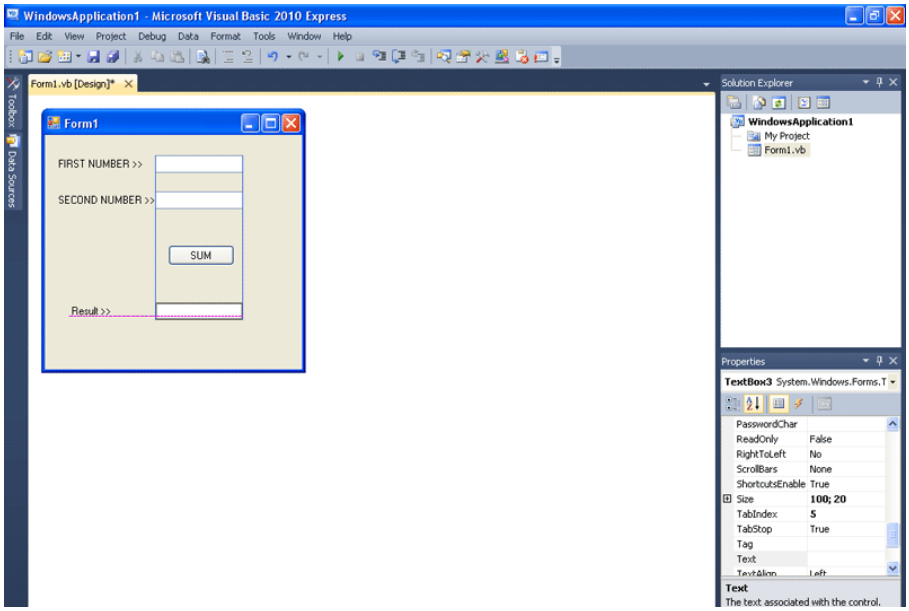


Τις ιδιότητες text των Labels αλλάξτε τις αντίστοιχα ως εξής:
Label 1 "FIRST NUMBER >>"
Label 2 "SECOND NUMBER >>"
Label 3 "Result >>"
Στα Αγγλικά αυτά σημαίνουν αντίστοιχα:
ΠΡΩΤΟΣ ΑΡΙΘΜΟΣ >>
ΔΕΥΤΕΡΟΣ ΑΡΙΘΜΟΣ >>
ΑΠΟΤΕΛΕΣΜΑ >>

Εμείς στο παράδειγμα το κάναμε στα Αγγλικά.
Αν θέλετε όμως, μπορείτε να κάνετε το πρόγραμμά σας στα Ελληνικά.

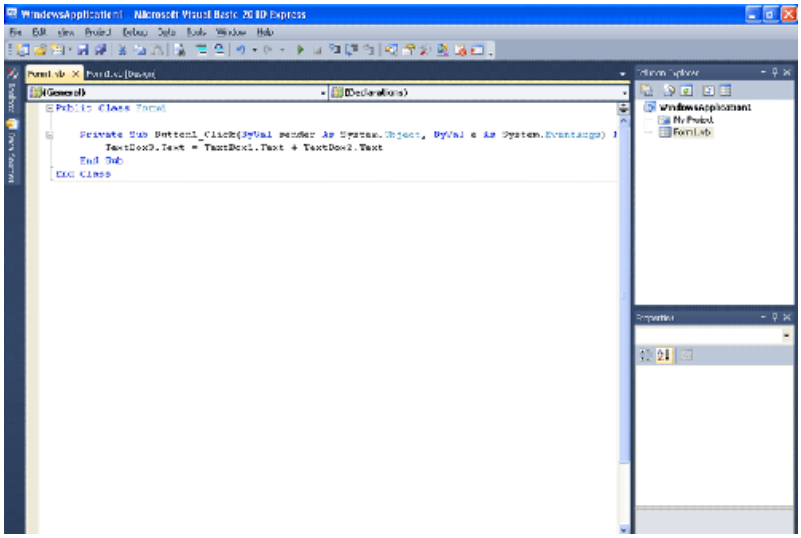


Εδώ αποφασίζουμε για την τελική στοίχιση των αντικειμένων. Ξαναλέμε πως αν είχαμε αναλύσει και σχεδιάσει από πριν το πρόγραμμα, όπως καταλαβαίνετε, θα είχαμε δώσει αυτή τη μορφή στη φόρμα εξ αρχής διότι θα ξέραμε τι πρέπει να κάνουμε. Αν αυτό εξοικονομεί χρόνο σε ένα τόσο απλό πρόγραμμα, όπως αυτό που φτιάχνουμε, φανταστείτε πόσο χρήσιμο είναι σε πιο πολύπλοκα προγράμματα. Γι αυτό το λόγο, προς το τέλος του βοηθήματος αυτού, θα καταπιαστούμε ακριβώς με αυτή την επαγγελματική προσέγγιση ανάπτυξης προγραμμάτων. Για την ώρα θα συνεχίσουμε με την υλοποίηση του προγράμματος.

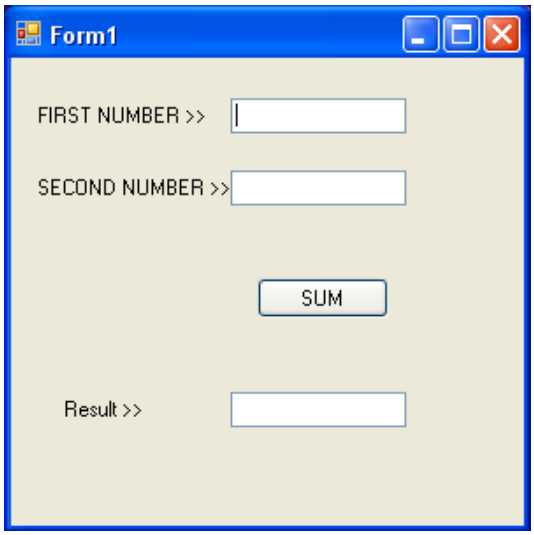


Κάντε διπλό κλίκ στο κουμπί για να εμφαινιστεί περιβάλλον συγγραφής κώδικα γι' αυτό. Γράψτε το παρακάτω μέσα στο Private Sub Button1_Click:

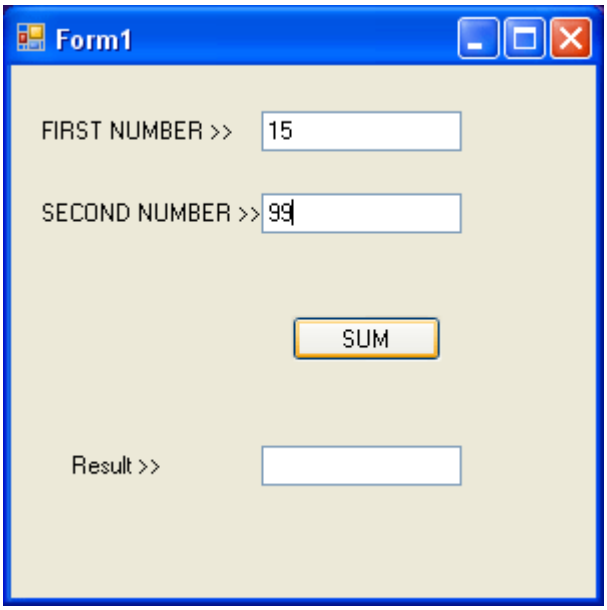
Textbox3.Text = Textbox1.Text + Textbox2.Text



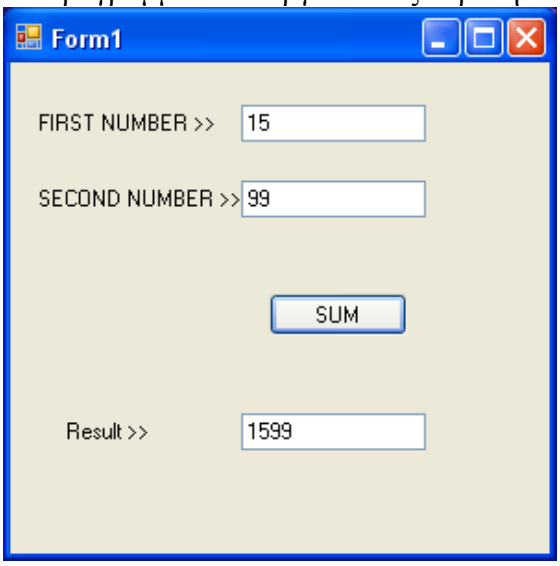
Βγείτε από το περιβάλλον εργασίας και τρέξτε το πρόγραμμα.



Βάλτε του να προσθέσει το 15 και το 99. Θα πρέπει να μας δίνει αποτέλεσμα 114.



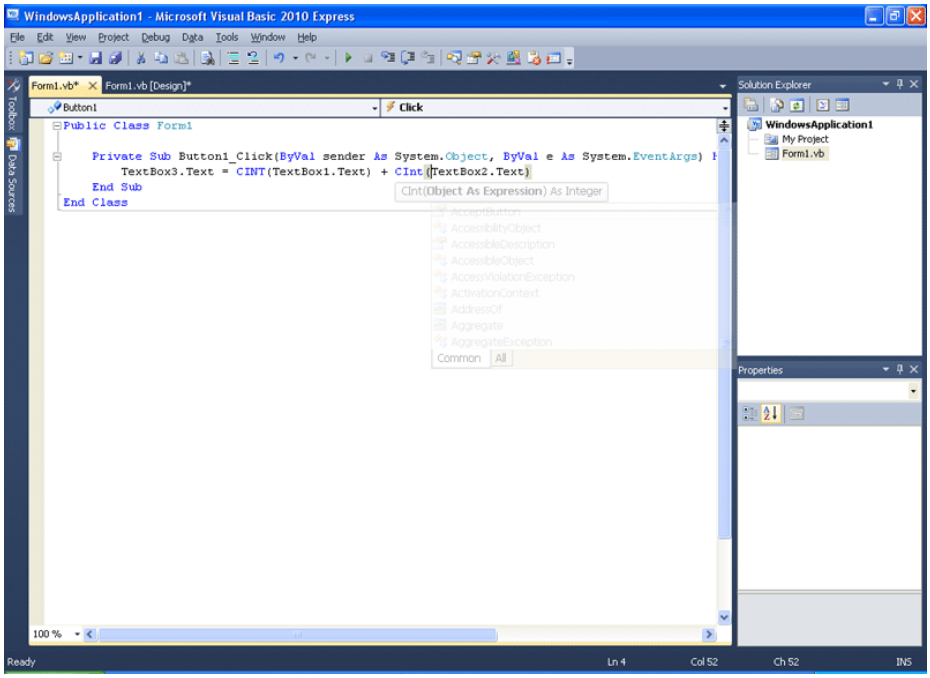
Πατήστε το κουμπί. Το πρόγραμμα λειτουργεί λάθος. Κρατήστε την ψυχραιμία σας.



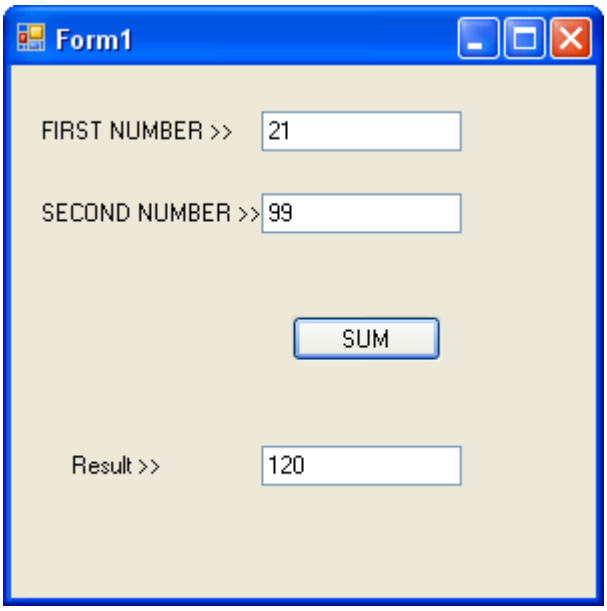
Αντί για 114 μας έβγαλε 1599 που είναι σαφώς λάθος. Γιατί συνέβη αυτό; Λοιπόν, η ιδιότητα text των textbox ουσιαστικά, αυτόματα, αποθηκεύεται σε μία μεταβλητή που η Visual Basic δημιουργεί. Τι τύπου είναι η μεταβλητή που αποθηκεύεται το text του textbox; Text = κείμενο στα Αγγλικά. Οπότε ο τύπος είναι String!!! Οπότε σωστά το πρόγραμμα, εκτέλεσε πρόσθεση δύο κειμένων το ένα μετά το άλλο. Θυμάστε που σας είπαμε πως κείμενο ακόμα κι αν είναι αριθμοί δε μπορεί να παίρνει μέρος σε πράξεις; Αυτό συμβαίνει εδώ ακριβώς. Η πρόσθεση κειμένου, είναι συγκόλληση δύο κειμένων και τίποτα παραπάνω. Είναι αυτό, το επιθυμητό αποτέλεσμα όμως; Όχι. Το σφάλμα είναι δικό μας.
Τι έπρεπε να κάνουμε;

Εδώ θα σας μάθουμε ένα τέχνασμα, που θα διορθώσει λίγο το παραπάνω πρόβλημα. Θα καταλάβετε στη συνέχεια γιατί λέμε "λίγο". Όπως βλέπετε στην παραπάνω εικόνα, τροποποιήστε τον κώδικα, ώστε η εντολή να πάρει την παρακάτω μορφή:

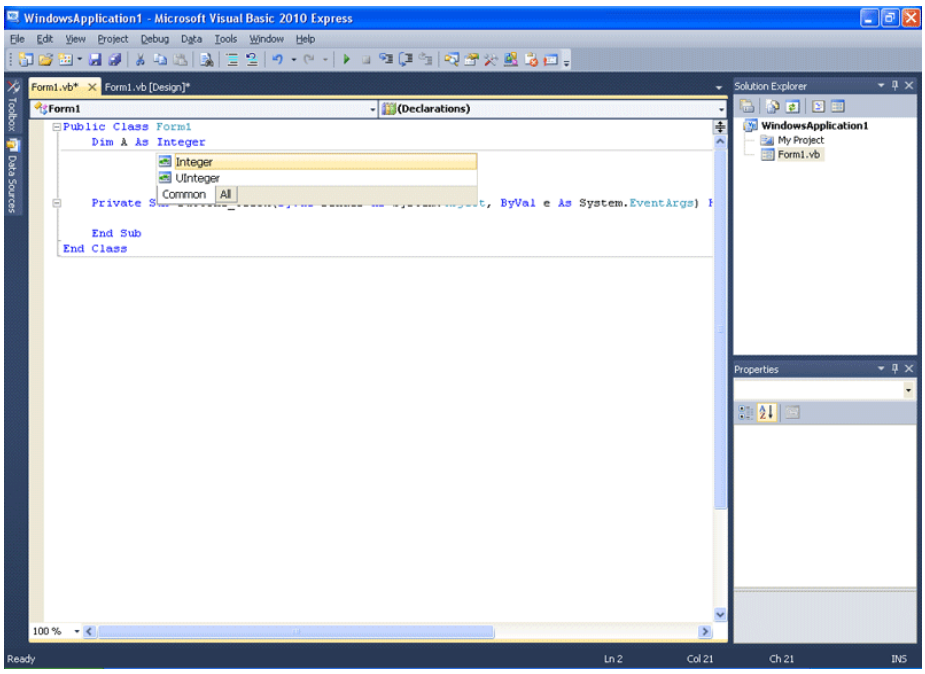
```
Textbox3.Text = CInt(Textbox1.Text) + CInt(Textbox2.Text)
```



Τι ακριβώς κάνουμε εδώ; Τα Textbox1.Text και Textbox2.Text είναι μεταβλητές τύπου String είπαμε. Το CINT (Convert to INTeger) μετατρέπει το κείμενο σε ακέραιο αριθμό. Οπότε η Textbox3.Text θα πάρει αυτή τη φορά ως τιμή το άθροισμα των ακεραίων αριθμών Textbox1.Text και Textbox2.Text. Ξανατρέξτε το πρόγραμμα, βάλτε του να αθροίσει τους αριθμούς 21 κι 99 αυτή τη φορά. Πατήστε το κουμπί. Βγάζει το επιθυμητό αποτέλεσμα, 120.



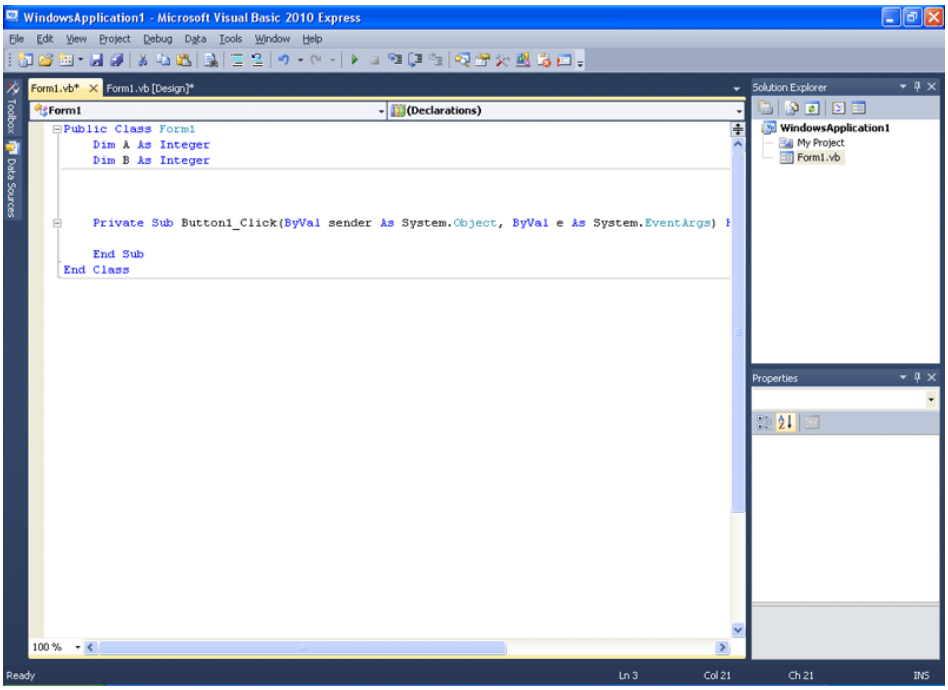
Τώρα θα σας δείξουμε έναν άλλο τρόπο να κάνετε το ίδιο ακριβώς πράγμα. Υπάρχει μια διαφορά. Στον πρώτο τρόπο θελήσαμε να σας δείξουμε, πως για το text ενός textbox δε χρειάζεται να φτιάξετε μεταβλητή, μιας και η Visual Basic ήδη έχει μία γι' αυτό. Με τον παρακάτω τρόπο, που θα σας δείξουμε, φτιάχνουμε δύο μεταβλητές, με ονομασίες A και B, οι οποίες θα πάρουν το περιεχόμενο των Textbox1.text και Textbox2.text αντίστοιχα. Αυτό είναι λάθος μιας και δεσμεύουμε δίχως λόγο περισσότερο χώρο στη μνήμη. Αλλά θέλουμε να σας δείξουμε πως να ορίζετε μεταβλητές και πως να κάνετε πράξεις μεταξύ τους. Διότι το Textbox1.text έχει μεν έτοιμη μεταβλητή, αλλά στη συνέχεια θα χρειαστεί, σε άλλα προγράμματα να ορίσετε μεταβλητές, που δεν φτιάχνονται αυτόματα. Πηγαίνετε στο περιβάλλον συγγραφής κώδικα και σβήστε ότι γράψαμε στο Private Sub Command1_click. Έπειτα πηγαίνετε στο Public Class Form1 και γράψτε: Dim A As Integer



Ομοίως γράψτε από κάτω:
Dim B As Integer
Τι κάναμε μόλις τώρα; Ορίσαμε δύο μεταβλητές, για την κλάση Form1, χρησιμοποιώντας την εντολή Dim. Για το τι είναι κλάση θα μιλήσουμε αργότερα. Μέσα στο αντικείμενο Form1 υπάρχουν κι άλλα αντικείμενα. Ένα από αυτά είναι το Button1. Το Button1 μπορεί να αντιληφθεί και να χειριστεί τις μεταβλητές αυτές, επειδή ανήκει στη Form1. Υπάρχουν κάποιοι κανόνες ως προς τα ονόματα μεταβλητών. Πρέπει πάντοτε:
Να ξεκινάνε με λατινικό γράμμα. Μπορούν να αποτελούνται από συνδυασμό των παρακάτω:
λατινικά γράμματα
αριθμούς
κάτω παύλα.

Μπορεί να έχει μήκος μέχρι 256 χαρακτήρες.

Δε μπορούμε να χρησιμοποιήσουμε δεσμευμένες λέξεις της Visual Basic όπως για παράδειγμα "Sub" "If" "End" "Function" και τα λοιπά. Κοινώς, λέξεις που είναι εντολές, ορίσματα, μέθοδοι, αντικείμενα, κλπ δε γίνεται να χρησιμοποιηθούν ως είναι, για ονόματα μεταβλητών.

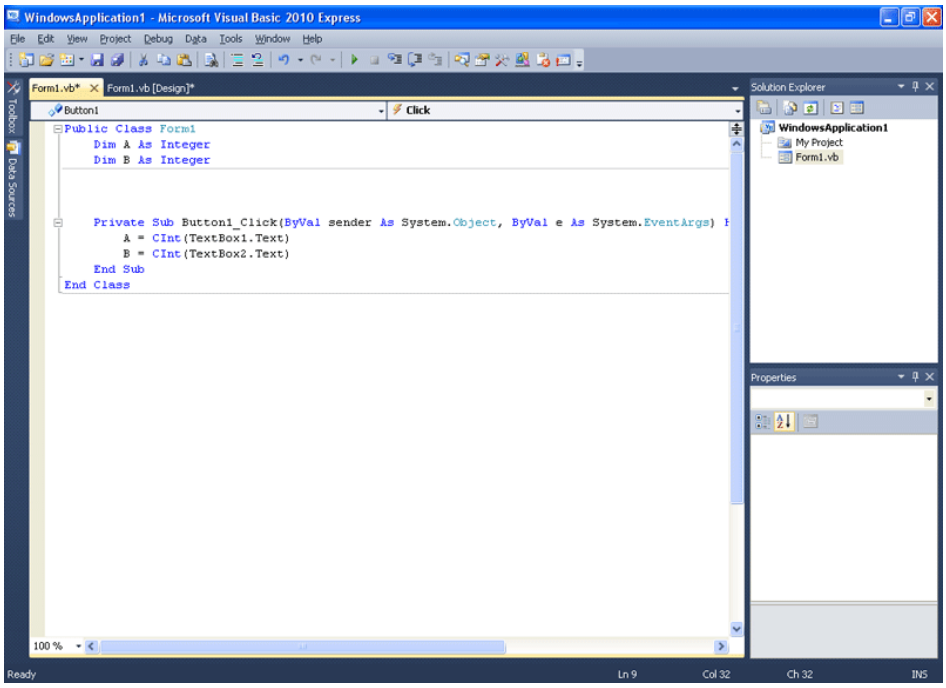


Πηγαίνετε τώρα στο Private Sub Button1_Click και γράψτε μέσα του τα παρακάτω:

A = CInt(Textbox1.Text)
B = CInt(Textbox2.Text)

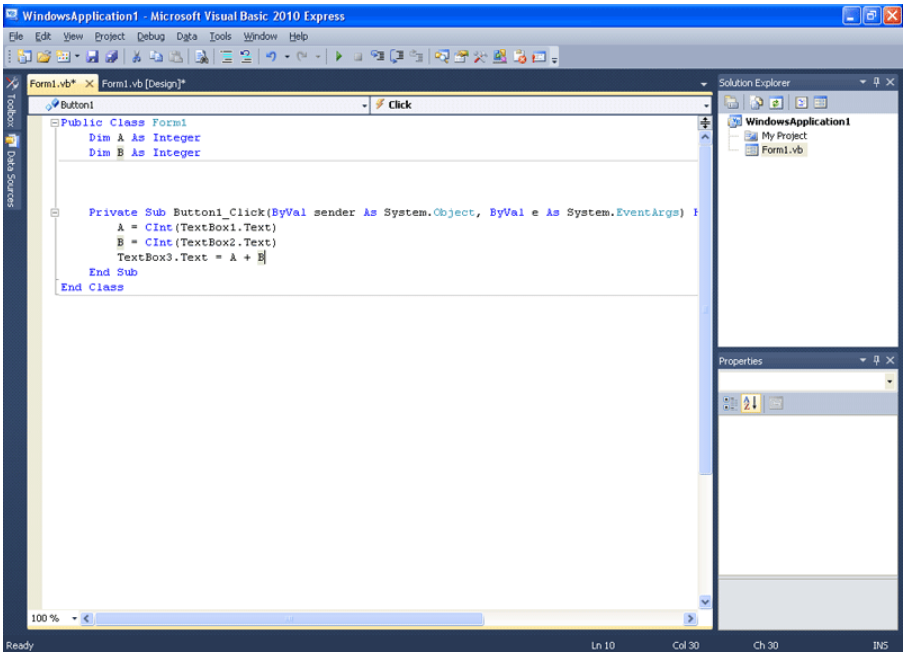
Τι κάναμε μόλις τώρα; Έχουμε ορίσει από πριν δύο μεταβλητές τύπου Integer. Την A και τη B.

Δώσαμε στη μεταβλητή Α την τιμή του Textbox1.Text αφού τη μετατρέψαμε σε ακέραιο.
Δώσαμε στη μεταβλητή Β την τιμή του Textbox2.Text αφού τη μετατρέψαμε σε ακέραιο.

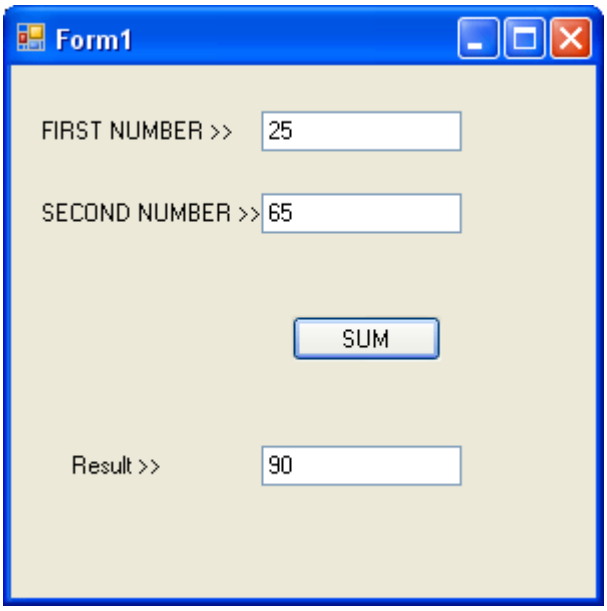


Γράφουμε λοιπόν από κάτω:
Textbox3.Text = A + B

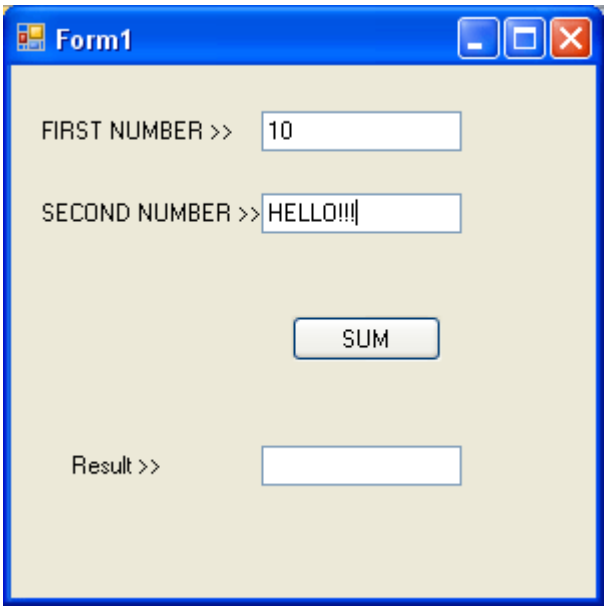
Τι κάναμε τώρα; Δώσαμε στο Textbox3.text (τύπου String) την τιμή του αθροίσματος των ακεραίων μεταβλητών Α και Β που είναι τύπου Integer. Integer χωράει σε String, θυμάστε το παράδειγμα του προηγούμενου κεφαλαίου;
Το ποτήρι χωράει στην κατσαρόλα κοινώς. Το πρόγραμμα αυτό, κάνει το ίδιο πράγμα με το προηγούμενο, απλώς φαίνεται πιο καθαρά η χρήση των μεταβλητών.



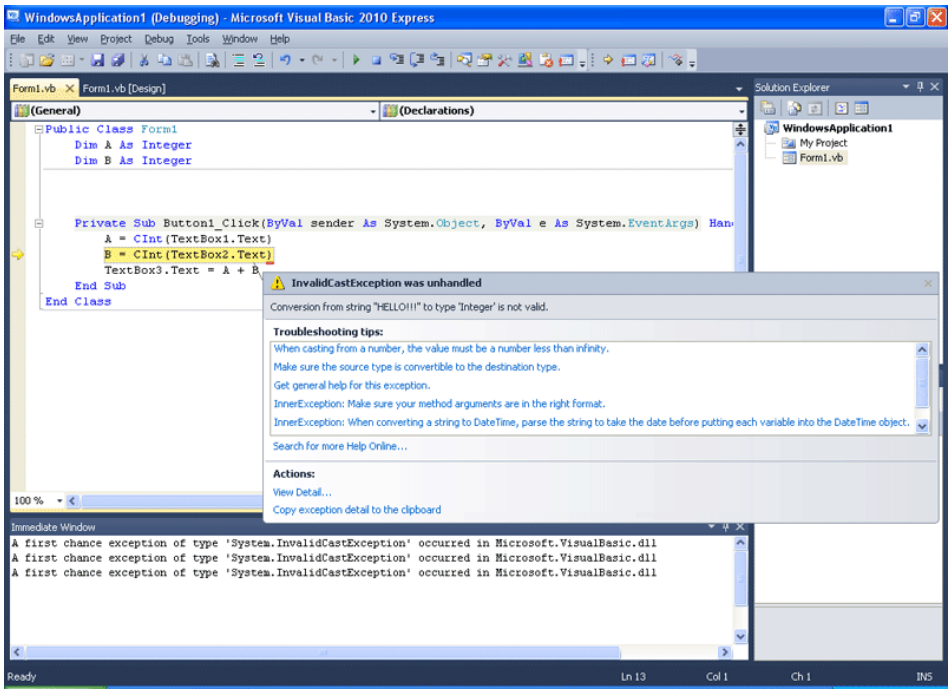
Τρέξτε το και βάλτε το να προσθέσει το 25 και το 65. Μας βγάζει 90. Σωστό.



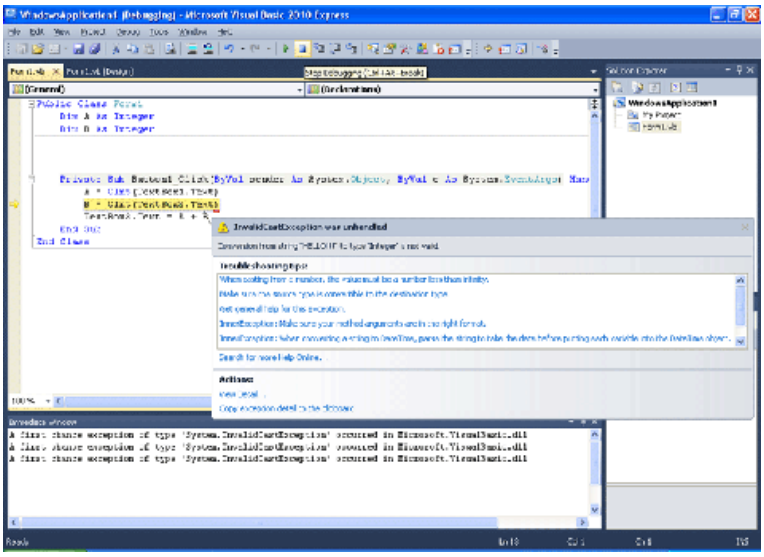
Ωστόσο δε μπορούμε να εμποδίσουμε (ακόμα) το χρήστη να κάνει κάτι ανεπιθύμητο στο πρόγραμμα. Και να είστε σίγουροι, πως αν κάτι μπορεί να πάει στραβά σε ένα πρόγραμμα, θα πάει!!! Στο ένα textbox βάλτε το 10 και στο δεύτερο γράψτε Hello!!!



Πατήστε το κουμπί. ΜΠΑΜ!



Το πρόγραμμα διακόπτεται αυτομάτως από τον compiler της Visual Basic και μας προειδοποιεί πως κάτι δεν πάει καλά. Τι λέει;
"Conversion from String "Hello!!!" to type 'Integer' is not valid."
Η μετατροπή της συμβολοσειράς "Hello!!!" σε τύπο ακέραιο δεν υφίσταται. Κοινώς το τέχνασμα με την CInt(), λειτουργεί μόνο αν η συμβολοσειρά αποτελείται από αριθμούς και μόνο. Πατήστε το κουμπί Stop debugging όπως δείχνει η παρακάτω εικόνα. Όταν γράψαμε "Hello!!!" και η Visual Basic πήγε να το μετατρέψει σε ακέραιο...



... συνέβη αυτό στο πρόγραμμα! Και ο compiler το σταμάτησε λέγοντάς μας ποιο είναι το πρόβλημα. Για να είμαστε πιο σωστοί μάλιστα, αυτό που συνέβη, ήταν ότι πήγαμε να βάλουμε μια κατσαρόλα μέσα σε ένα ποτήρι. Ακόμα χειρότερα δηλαδή!

Υπάρχουν τρόποι για να αποφύγουμε το παραπάνω δυσάρεστο συμβάν. Ωστόσο χρειάζονται λίγες παραπάνω γνώσεις. Γνώσεις που στο επόμενο κεφάλαιο θα αποκτήσετε!

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό, οδηγούμε το μαθητευόμενο στο να φτιάξει ένα πρόγραμμα το οποίο να χρησιμοποιεί μεταβλητές. Παράλληλα, του διδάσκουμε μερικές πρακτικές λειτουργίες του γραφικού περιβάλλοντος ανάπτυξης, ενώ του δείχνουμε τι μπορεί να πάει στραβά, αν ένας τύπος δεδομένων πάει να αποθηκευτεί σε μια μεταβλητή διαφορετικού τύπου.

ΚΕΦΑΛΑΙΟ 9
ΤΕΛΕΣΤΕΣ ΚΑΙ ΛΗΨΕΙΣ ΑΠΟΦΑΣΕΩΝ

Ήδη έχουμε μιλήσει για το τι είναι τελεστής από το κεφάλαιο "Μιλάμε την ίδια Γλώσσα;". Εδώ θα μιλήσουμε συγκεντρωτικά για σχεδόν όλους τους τελεστές που μπορεί να σας χρησιμεύσουν στη Visual Basic. Είναι απαραίτητη η γνώση τους σε κάθε προγραμματιστή. Αντί όμως να σας κάνουμε μια απλή παρουσίαση για το τι κάνουν όλοι, θα σας εισάγουμε, λίγο πιο βαθιά, στη λογική των πιο συνηθισμένων. Παράλληλα, θα σας δείξουμε μια πολύ βασική δομή λήψεως αποφάσεων, την εντολή IF. Τέλος, θα σας δείξουμε πως μπορείτε να δίνετε στο χρήστη μηνύματα σφάλματος στην πράξη, με την εντολή MsgBox(). Την MsgBox() ωστόσο, μπορούμε να τη χρησιμοποιήσουμε και για να στείλουμε στο χρήστη απλώς ένα μήνυμα, γεγονός που θα δείξουμε και στο επόμενο κεφάλαιο.

Αυτό το κεφάλαιο, περιέχει πολύ σποδαία θέματα. Συνιστάται να κάνετε μια επανάληψη σε όσα ήδη μάθατε, προτού ξεκινήσετε την ανάγνωσή του. Προτείνεται επίσης, να το μελετήσετε ολόκληρο κι όχι αποσπασματικά. Οπότε μελετήστε το κάποια στιγμή που θα έχετε αρκετό χρόνο στη διάθεσή σας. Απαιτεί πολλή προσοχή, διότι εδώ αρχίζετε να μπαίνετε πιο βαθιά στην προγραμματιστική λογική.

ΤΕΛΕΣΤΗΣ ΑΠΟΔΟΣΗΣ ΤΙΜΗΣ:
Για να αποδώσουμε μια τιμή σε μια μεταβλητή, χρησιμοποιούμε το ίσον. Για παράδειγμα αν Α και Β είναι μεταβλητές τύπου Integer, λέμε:
A = 10
Η Α παίρνει την τιμή 10 δηλαδή.
B = A
Η Β παίρνει την τιμή της Α, δηλαδή το 10 που έχει ήδη από την προηγούμενη εντολή.
B = A + A
Η Β παίρνει ως ΝΕΑ τιμή, το Α+Α που είναι 10+10 άρα 20.
B = B + 10
Η Β παίρνει ως νέα τιμή, ότι έχει ήδη μέσα της η ίδια (Β), το 20 δηλαδή, ΣΥΝ 10. Οπότε γίνεται 30.
A = 30
Η Α χάνει την προηγούμενη τιμή και παίρνει ως νέα τιμή της το 30.
B = B - A
Η Β παίρνει ως νέα τιμή, ότι έχει ήδη μέσα της, μείον ότι έχει η Α.
30 - 30 = 0 λοιπόν.
Καταλήγουμε λοιπόν στο ότι η Α περιέχει την τιμή 30 και η Β την τιμή 0.

ΜΑΘΗΜΑΤΙΚΟΙ ΤΕΛΕΣΤΕΣ:
Το + είναι ο τελεστής αθροίσματος. Σε αριθμητικές μεταβλητές, λειτουργεί όπως το μαθηματικό + ενώ θέλει προσοχή το γεγονός ότι μεταξύ μεταβλητών τύπου String απλώς ενώνει τα κείμενα γιατί λειτουργεί ως τελεστής συνένωσης. Τελεστής συνένωσης είναι και ο & Το - αντίστοιχα είναι ο τελεστής αφαίρεσης όπως γνωρίζουμε στα μαθηματικά. Δηλαδή η εντολή A = A - B σημαίνει πως η μεταβλητή Α θα αφαιρέσει το περιεχόμενο αριθμό της Β, από ότι έχει μέσα της ήδη η Α.

Το * είναι ο τελεστής πολλαπλασιασμού οπότε A = A * B είναι η εντολή που πολλαπλασιάζει οτιδήποτε περιέχεται μέσα στην Α, με οτιδήποτε περιέχεται μέσα στη Β και το γινόμενο, το τοποθετεί ως νέα τιμή στην Α.

Έστω τα παρακάτω:
A = 10
B = 2
A = A * B
Η A θα πάρει την τιμή 20!
Το ^ είναι ο τελεστής ύψωσης σε δύναμη. Δηλαδή όταν λέμε A = A ^ 2
εννοούμε ότι η A παίρνει ως νέα τιμή, την ήδη υπάρχουσα υψωμένη στο τετράγωνο.
A = 2
A = A ^ 3
Η A θα γίνει 8 μίας και 2 στον κύβο ισούται με 8!
Το / είναι ο τελεστής της διαίρεσης. Κοινώς A = A / B σημαίνει πως το A διαιρείται με το B.
Έστώ ότι η A είναι τύπου Double (πρέπει μιας και διαίρεση μπορεί να βγάλει δεκαδικό αποτέλεσμα).
A = 10
B = 3
A = A / B
Η A θα πάρει την τιμή 3.333333...
Θέλει πολλή προσοχή το B να μην παίρνει ποτέ την τιμή 0, μιας και τέτοια διαίρεση δεν ορίζεται.
Το \ είναι ένας ειδικός μαθηματικός τελεστής. Δίνει τιμή το ακέραιο πηλίκο μιας διαίρεσης.
A = 10 \ 3
Η A παίρνει την τιμή 3, διότι τρεις φορές το 3 μας κάνει 9 και περισσεύει 1.
Η A δε θα πάρει την τιμή 3.3333... αλλά την τιμή 3!
Ο \ αδιαφορεί για το υπόλοιπο και κρατά μόνο το πόσες ακέραιες φορές χωράει το B στο A δηλαδή.
Το Mod είναι ένας ειδικός μαθηματικός τελεστής που κάνει το αντίθετο του \.
Δηλαδή μας δίνει το υπόλοιπο από μια διαίρεση αδιαφορώντας για το πόσες φορές χώρεσε το B στο A
A = 10 Mod 3
Η A θα πάρει την τιμή 1.
Διότι το 3 χωράει τρεις φορές στο 10, οπότε 10 - 9 = 1 υπόλοιπο.

ΣΥΝΘΕΤΟΙ ΤΕΛΕΣΤΕΣ ΑΠΟΔΟΣΗΣ ΤΙΜΗΣ:
Παρατηρήσατε στα προηγούμενα παραδείγματα, ότι για να ΠΡΟΣΘΕΣΟΥΜΕ μια τιμή σε μια μεταβλητή, αντί να την αντικαταστήσουμε με μια άλλη, έπρεπε στο δεύτερο μέρος της πράξης να βάζουμε και τον εαυτό της. Με άλλα λόγια, αν η μεταβλητή A είχε τιμή 10 και θέλαμε να της προσθέσουμε 5 λέγαμε A = A +5
Αντί αυτού, μπορούμε να χρησιμοποιήσουμε τον σύνθετο τελεστή += Δηλαδή αν πούμε
A += 5
εννοούμε A = A + 5
Ομοίως
A -= 5
εννοούμε A = A - 5
και...
A *= 5
εννοούμε A = A * 5
A /= 5
εννοούμε A = A / 5

και τα λοιπά.

ΤΕΛΕΣΤΕΣ - ΣΥΝΑΡΤΗΣΕΙΣ ΜΕΤΑΤΡΟΠΗΣ ΤΥΠΟΥ ΔΕΔΟΜΕΝΩΝ:
Θυμάστε το τέχνασμα με το οποίο το κείμενο το κάναμε ακέραιο αριθμό, αλλά έπρεπε το κείμενο να αποτελείται μόνο απο αριθμούς; Ήταν ουσιαστικά μια συνάρτηση μετατροπής μορφής δεδομένων, σε τύπο ακεραίου (Integer). Η σύνταξή της, είναι η εξής:
CInt(expression)
όπου expression, βάζουμε μια μαθηματική έκφραση ή απλώς το όνομα μιας μεταβλητής. Δείτε το παρακάτω παράδειγμα:

```
Dim A As Double
Dim B As Integer
A = 2345.5678
B = CInt(A)
```

Η B παίρνει τη στρογγυλοποιημένη τιμή 2346. Όπως βλέπετε, ουσιαστικά η χρήση της εντολής, τροποποιεί τον δεκαδικό αριθμό σε ακέραιο, αλλά αυτή η στρογγυλοποίηση, δημιουργεί μεγάλες ανακρίβειες. Θυμάστε που σας δίναμε το παράδειγμα ότι όπως το μήλο δε χωράει σε ένα ποτήρι έτσι και και η τιμή μιας Double δε χωράει σε μια μεταβλητή Integer;

Ουσιαστικά εδώ, με τη στρογγυλοποίηση, είναι σαν κάποιος να ζούληξε το μήλο κι όσο χώρεσε να περάσει στο ποτήρι πέρασε, ενώ το υπόλοιπο έμεινε απ' έξω. Χώρεσε μέρος του μήλου, αλλά δεν είναι πλέον μήλο. Είναι πετσοκομμένο μήλο. Διότι 2345.5678 δεν είναι το ίδιο σε καμία περίπτωση με το 2346. Θέλει λοιπόν πολλή προσοχή αν είναι να χρησιμοποιήσετε τέτοιες εντολές. Είναι καλύτερο να ξέρετε να χρησιμοποιείτε το σωστό τύπο μεταβλητών παρά να κάνετε "αλχημίες". Και θα αναρωτιέστε τώρα, πώς τότε να χειριστώ τα δεδομένα σε ένα textbox; Η απάντηση θα σας δοθεί πολύ σύντομα!

Ομοίως πάντως για μετατροπή σε Double, String ,Boolean, Byte, Char, Date, Decimal, Long, Short, υπάρχουν αντίστοιχες εντολές:
CDbl(expression)
CStr(expression)
CBool(expression)
CByte(expression)
CChar(expression)
CDate(expression)
CDec(expression)
CLng(expression)
CShort(expression)

Μας μένουν οι τελεστές σύγκρισης και οι λογικοί τελεστές. Για να προχωρήσουμε όμως στην επεξήγηση των τελεστών σύγκρισης, αλλά και των λογικών τελεστών, θα πρέπει πρώτα να μάθουμε την εντολή, με την οποία ένα πρόγραμμα παίρνει αποφάσεις. Αυτή η εντολή είναι η εντολή IF, που στα Ελληνικά σημαίνει "Εάν". Αυτό το μαθαίνουμε πρώτα, γιατι θα γίνει πιο εύκολη για εσάς, η εκμάθηση αυτών των τελεστών, με τη χρήση της IF. Προτού προχωρήσουμε λοιπόν στην επεξήγησή της, θα πρέπει να σας δώσουμε να καταλάβετε τη λογική της IF με απλό τρόπο.

Έστω ότι εργάζεστε τα πρωινά, σε μια επιχείρηση. Έστω ότι θέλετε πάση θυσία να είστε τυπικοί στην εργασία σας, οπότε δε θέλετε να λείψετε ή να αργήσετε ποτέ. Έστω τώρα ότι ξυπνάτε το πρωί στις 5 τα ξημερώματα και πρέπει να πάρετε μια απόφαση. Να σηκωθώ από το κρεβάτι ή όχι; Από τι εξαρτάται; Από το αν είναι αργία ή όχι φυσικά.
Η απόφασή σας λοιπόν, παίρνει την αλγοριθμική μορφή:
ΑΝ (είναι αργία) ΤΟΤΕ
κοιμήσου κι άλλο
ΑΛΛΙΩΣ
ξύπνα κι ετοιμάσου να πας στη δουλειά
ΤΕΛΟΣ ΑΝ

Τι λέμε εδώ δηλαδή; Αν η συνθήκη που βρίσκεται στην παρένθεση είναι αληθής (1), εκτελείται η πράξη μετά το "ΤΟΤΕ" Αν η συνθήκη που βρίσκεται στην παρένθεση είναι ψευδής (0), εκτελείται η πράξη μετά το "ΑΛΛΙΩΣ" Στον προγραμματισμό, αληθές είναι το 1 και ψευδές το 0 σε μια boolean έκφραση. Κοινώς, μια συνθήκη είναι είτε αληθής είτε ψευδής! Δε γίνεται και τα δύο ταυτόχρονα.

Σε αυτό το παράδειγμα που μόλις είδαμε, ελέγχαμε μόνο μια παράμετρο (συνθήκη σε παρένθεση) αν είναι αληθής ή όχι, δηλαδή το αν είναι αργία ή όχι. Στο επόμενο παράδειγμα, θα προχωρήσουμε σε μια πιο προχωρημένη λογική λήψης απόφασης, βασισμένη σε έλεγχο τριών συνθηκών. Αν είναι Σάββατο, αν είναι Κυριακή, αν είναι αργία λόγω γιορτής.

ΑΝ (είναι Σάββατο) ΤΟΤΕ κοιμήσου κι άλλο
ΑΛΛΙΩΣ ΑΝ (είναι Κυριακή) ΤΟΤΕ κοιμήσου κι άλλο
ΑΛΛΙΩΣ ΑΝ (είναι αργία λόγω γιορτής) ΤΟΤΕ κοιμήσου κι άλλο
ΑΛΛΙΩΣ ξύπνα κι ετοιμάσου να πας στη δουλειά
ΤΕΛΟΣ ΑΝ Τι λέμε εδώ; Ελέγχεις αν είναι Σάββατο κι αν όντως είναι, κοιμήσου, αν όχι ελέγχεις αν είναι Κυριακή. Αν είναι Κυριακή κοιμήσου, αν δεν είναι, ελέγχεις αν είναι αργία λόγω γιορτής. Αν είναι κοιμήσου, αν όχι, ξύπνα κι ετοιμάσου να πας στη δουλειά (δε

τη γλιτώνεις σήμερα). Μα είναι αυτό προγραμματισμός; Κι όμως, είναι ο ακρογωνιαίος λίθος στη λήψη αποφάσεων από προγράμματα. Κοινώς στα Ελληνικά, έχουμε κάποιες εντολές κλειδιά. Κάποια τουβλάκια δηλαδή, που δομούν την εντολή "ΑΝ" Αυτά είναι:
ΑΝ
ΤΟΤΕ
ΑΛΛΙΩΣ ΑΝ
ΑΛΛΙΩΣ
ΤΕΛΟΣ ΑΝ

Ομοίως κι αντίστοιχα στα Αγγλικά, έχουμε:
IF
THEN
ELSE IF
ELSE
END IF

Ας υποθέσουμε ότι θέλουμε να κάνουμε έναν έλεγχο και ανάλογα με το αποτέλεσμά του ελέγχου αυτού, το πρόγραμμα να κάνει κάτι διαφορετικό. Ας πούμε λοιπόν, πως θέλουμε να φτιάξουμε ένα πρόγραμμα, στο οποίο ο χρήστης μπορεί να βάζει έναν αριθμό, θετικό, αρνητικό ή μηδέν. Έπειτα θα πατάει ένα κουμπί και κάτι θα συμβαίνει, ανάλογα με το τι αριθμό έβαλε ο χρήστης. Αν έβαλε θετικό, το φόντο της φόρμας θα γίνεται κόκκινο. Αν έβαλε αρνητικό, το φόντο της φόρμας θα γίνεται μπλε. Αν τέλος έβαλε μηδέν, το φόντο θα γίνεται γκρίζο. Να δούμε στα Ελληνικά πρώτα την εντολή IF;

ΑΝ (ο αριθμός που έβαλε ο χρήστης είναι μεγαλύτερος του μηδενός) ΤΟΤΕ
κάνε το φόντο της φόρμας κόκκινο
ΑΛΛΙΩΣ ΑΝ (ο αριθμός που έβαλε ο χρήστης είναι μικρότερος του μηδενός) ΤΟΤΕ
κάνε το φόντο της φόρμας μπλε
ΑΛΛΙΩΣ κάνε το φόντο της φόρμας γκρίζο
ΤΕΛΟΣ ΑΝ
Εδώ τι λέμε δηλαδή; Αν είναι μεγαλύτερος ο αριθμός από το μηδέν κάνε το φόντο κόκκινο. Αν όχι, τότε κάνε έναν επιπλέον έλεγχο, γιατί μας μένουν δύο επιλογές. Είτε θα είναι μηδέν, είτε αρνητικός. Οπότε αν είναι μικρότερος ο αριθμός από το μηδέν, κάνε το φόντο μπλε. Αν δεν είναι τότε τι μας μένει; Μόνο το να είναι ίσο με μηδέν. Οπότε μετά το τελευταίο "ΑΛΛΙΩΣ", αν τα δύο παραπάνω είναι ψευδή, σημαίνει πως ο αριθμός ήταν μηδέν. Και γράφουμε απλά, κάνε γκρίζο το φόντο.

Παρατηρήσατε κάτι όμως; Είπαμε φράσεις όπως: "Μεγαλύτερο του", "Μικρότερο του", "Ίσο με". Πώς αυτά εκφράζονται προγραμματιστικά; Μα με τους τελεστές σύγκρισης!

ΤΕΛΕΣΤΕΣ ΣΥΓΚΡΙΣΗΣ:

Χρησιμοποιούνται σε εκφράσεις που θέλουμε να ελέγξουμε αν είναι αληθείς ή ψευδείς.
Ένα καλό παράδειγμα είναι μια εντολή IF που περιέχει τουλάχιστο μία έκφραση ελέγχου αληθείας.
Τελεστές σύγκρισης:
=
A = B
Έλεγχος ισότητας. Αληθής φράση όταν το Α ισούται με το Β

Εδώ απαιτείται προσοχή! Σε άλλες γλώσσες προγραμματισμού ο τελεστής απόδοσης τιμής, είναι :=
και όχι =
για να μη γίνεται σύγχυση των δύο αυτών διαφορετικών συμβόλων. Στη Visual Basic, αυτό δεν ισχύει.

Με άλλα λόγια ο τελεστής απόδοσης τιμής, αλλά και ο τελεστής σύγκρισης, είναι το ίσον. Αυτό κάνει πολλές φορές όσους μαθαίνουν Visual Basic να μπερδεύονται. Να έχετε στα υπ' όψιν σας το παρακάτω λοιπόν.

Όταν δίνετε μια εντολή
A = 10 σημαίνει πως η μεταβλητή Α παίρνει την τιμή 10.

Όταν όμως το ίσον βρίσκεται στη συνθήκη μιας εντολής (IF για παράδειγμα) τότε χρησιμοποιείται ως τελεστής σύγκρισης.
Σύνταξη:
IF (ΣΥΝΘΗΚΗ) THEN Εντολές αν αληθεύει η συνθήκη
ELSE Εντολές αν δεν αληθεύει η συνθήκη
ENDIF

Κοινώς δείτε το παρακάτω παράδειγμα:
```
If A = 5 THEN
Label1.Text="It's 5!"
Else
Label1.Text="It's not 5!"
End IF
```
Εδώ, όπως θα δείτε παρακάτω, η εντολή είναι η IF και το A = 5 είναι μια συνθήκη.
Κοινώς η IF μελετάει αν η μεταβλητή ισούται με 5.
Αν ναι, εκτελείται η εντολή
Label1.Text="It's 5!"
Αλλιώς εκτελείται η εντολή
Label1.Text="It's not 5!"

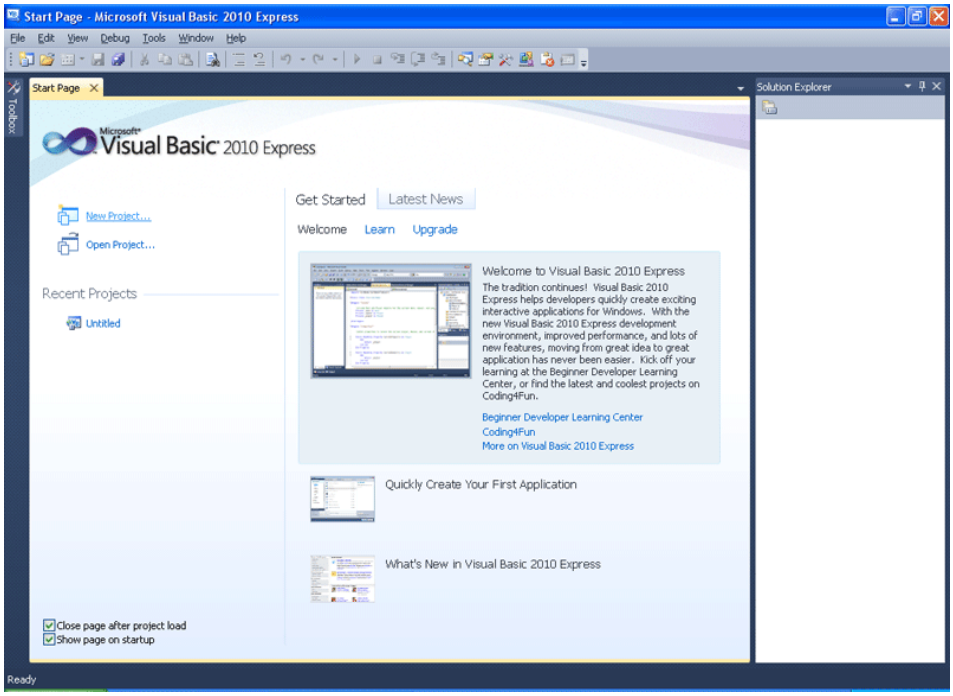
Το A = 5 της IF με άλλα λόγια ΔΕΝ έδωσε τιμή 5 στην Α,
αλλά έλεγξε απλώς, αν η Α έχει την τιμή 5.

Ομοίως έχουμε λοιπόν κι άλλους τελεστές σύγκρισης.
>
A > B
Έλεγχος σύγκρισης μεγέθους. Αληθής φράση όταν το Α είνι μεγαλύτερο του Β.
<
A < B
Έλεγχος σύγκρισης μεγέθους. Αληθής φράση όταν το Α είναι μικρότερο του Β.
<>
A <> B
Έλεγχος διαφορετικότητας. Αληθεύει όταν Α και Β διαφορετικά, μεταξύ τους.
>=
A >= B
Έλεγχος σύγκρισης μεγέθους. Αληθής φράση όταν το Α είναι μεγαλύτερο ή ίσο του Β.
<=
A <= B
Έλεγχος σύγκρισης μεγέθους. Αληθής φράση όταν το Α είναι μικρότερο ή ίσο του Β.

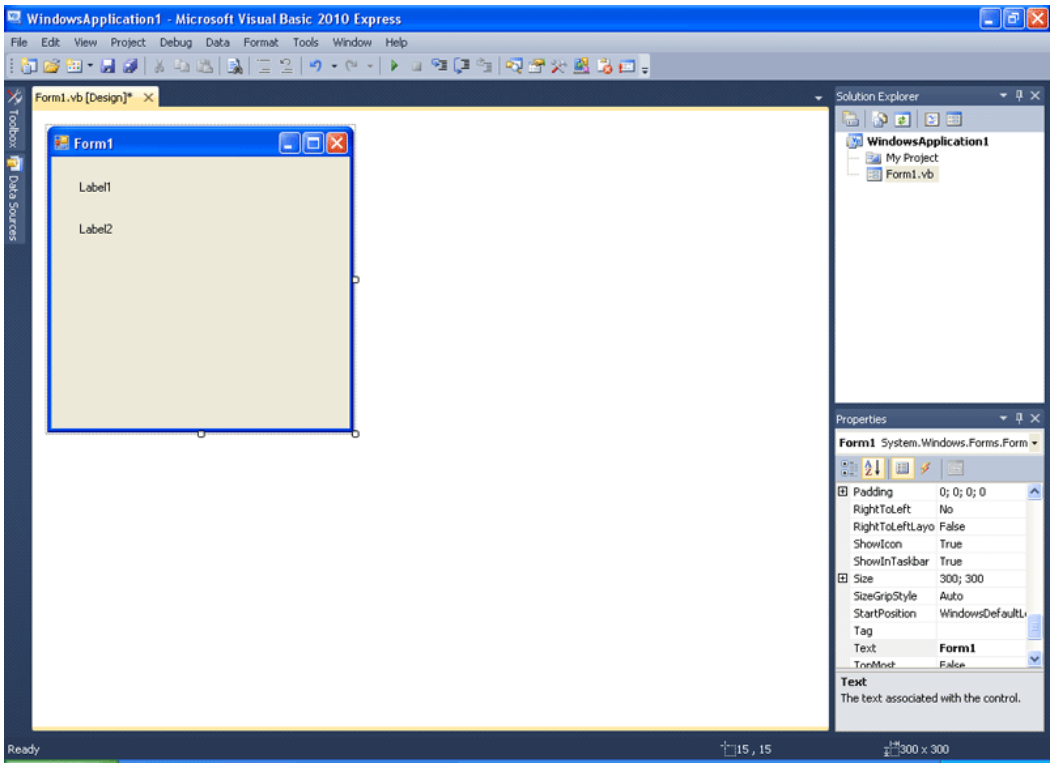
Ας ξαναγράψουμε τώρα την ΑΝ βάσει αυτών που μάθαμε:

ΑΝ (ο αριθμός που έβαλε ο χρήστης > 0) ΤΟΤΕ
κάνε το φόντο της φόρμας κόκκινο
ΑΛΛΙΩΣ ΑΝ (ο αριθμός που έβαλε ο χρήστης < 0) ΤΟΤΕ
κάνε το φόντο της φόρμας μπλε
ΑΛΛΙΩΣ
κάνε το φόντο της φόρμας γκρίζο
ΤΕΛΟΣ ΑΝ

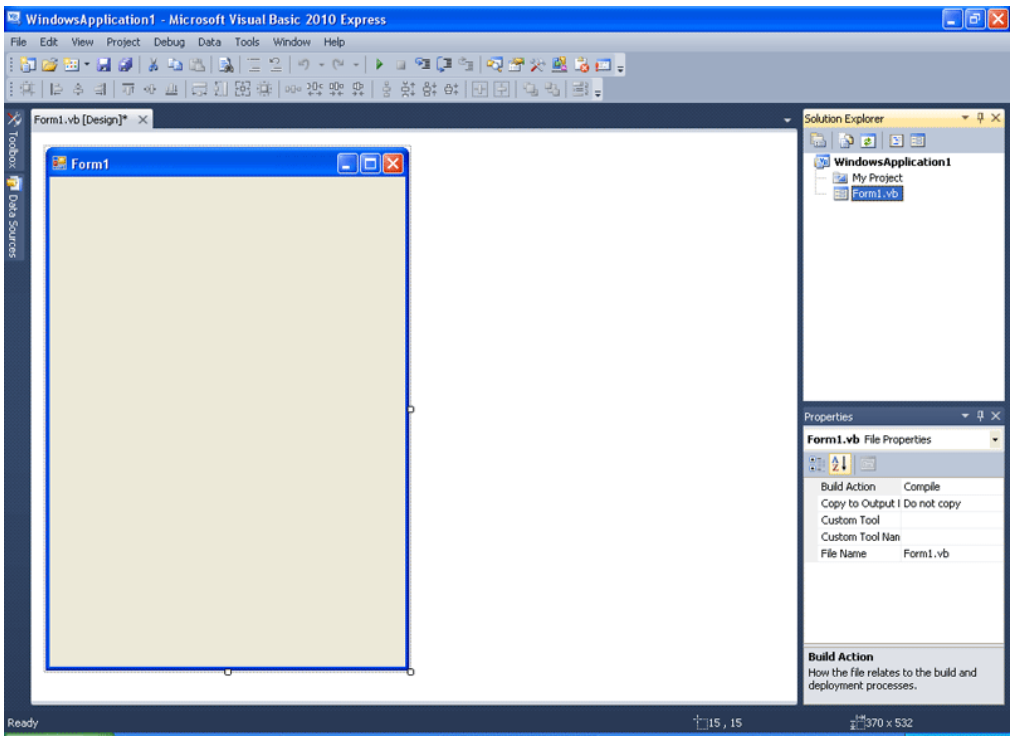
Βλέπετε; Σιγά σιγά τα κομμάτια συνδέονται και καταλήγουμε σε μια τελική μορφή.
Εδώ όμως σταματάμε τα λόγια. Καιρός να φτιάξουμε αυτό το υποθετικό πρόγραμμα, σαφώς εξηγώντας τον κώδικά του. Αρχίζουμε κλασσικά, επιλέγοντας New Project.



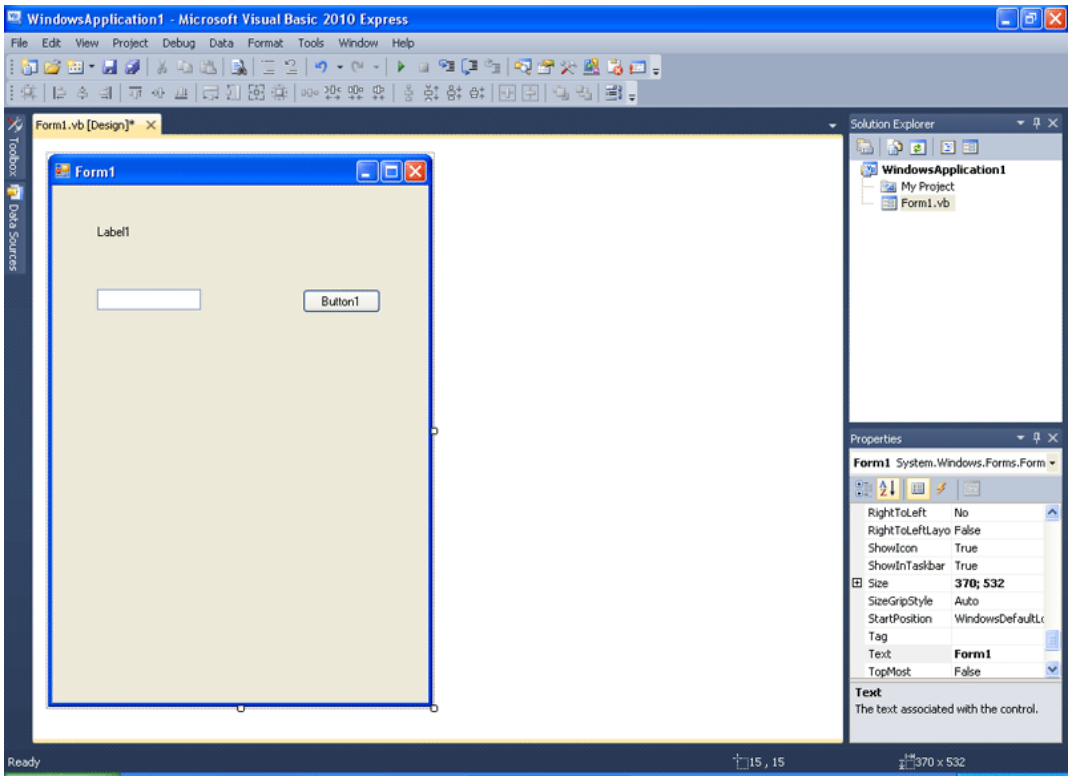
Επιλέγουμε Windows Forms Application



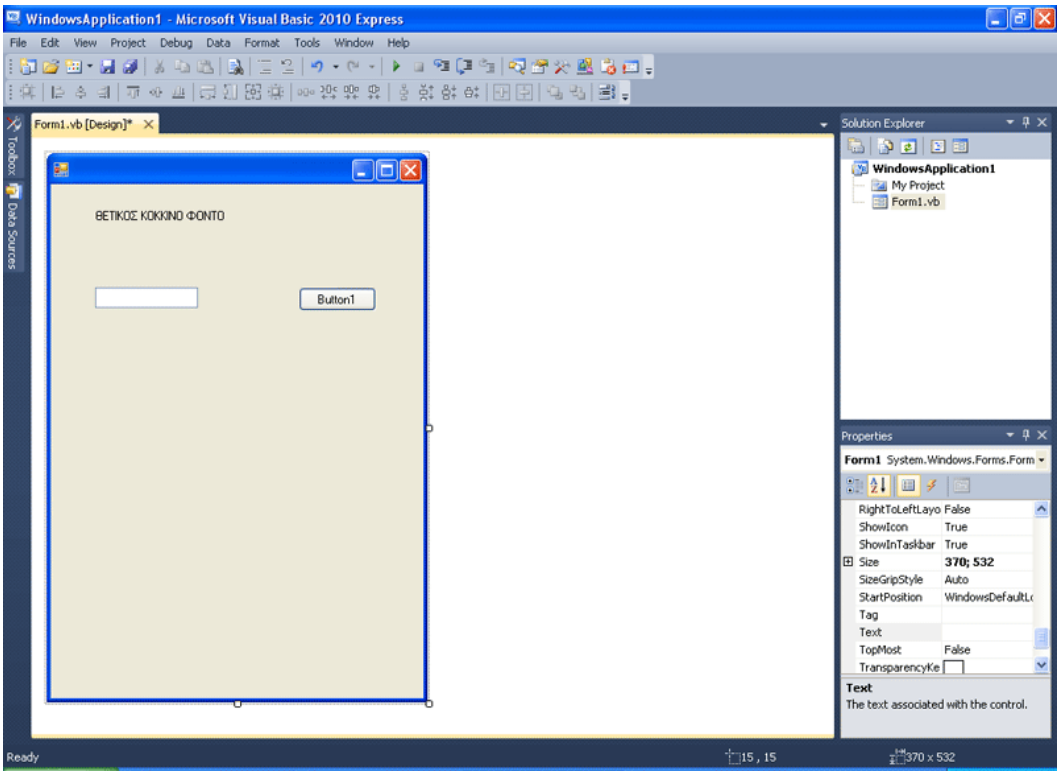
Κι έτσι έχουμε μια φόρμα.
Τη μεγαλώνουμε λίγο (δε ξέρουμε το τελικό μέγεθος εξ αρχής)



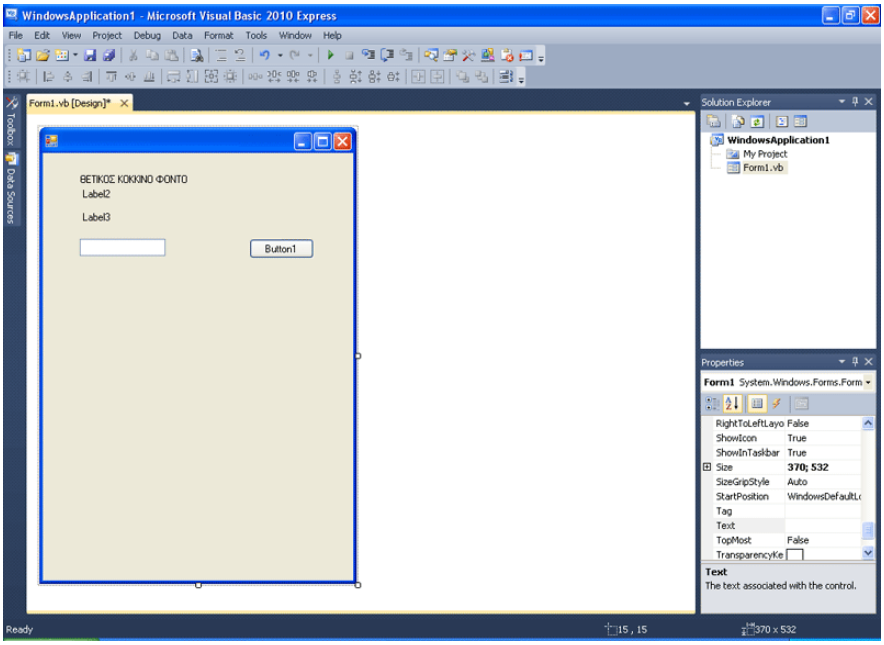
Βάζουμε μία Label, ένα Textbox κι ένα Button.



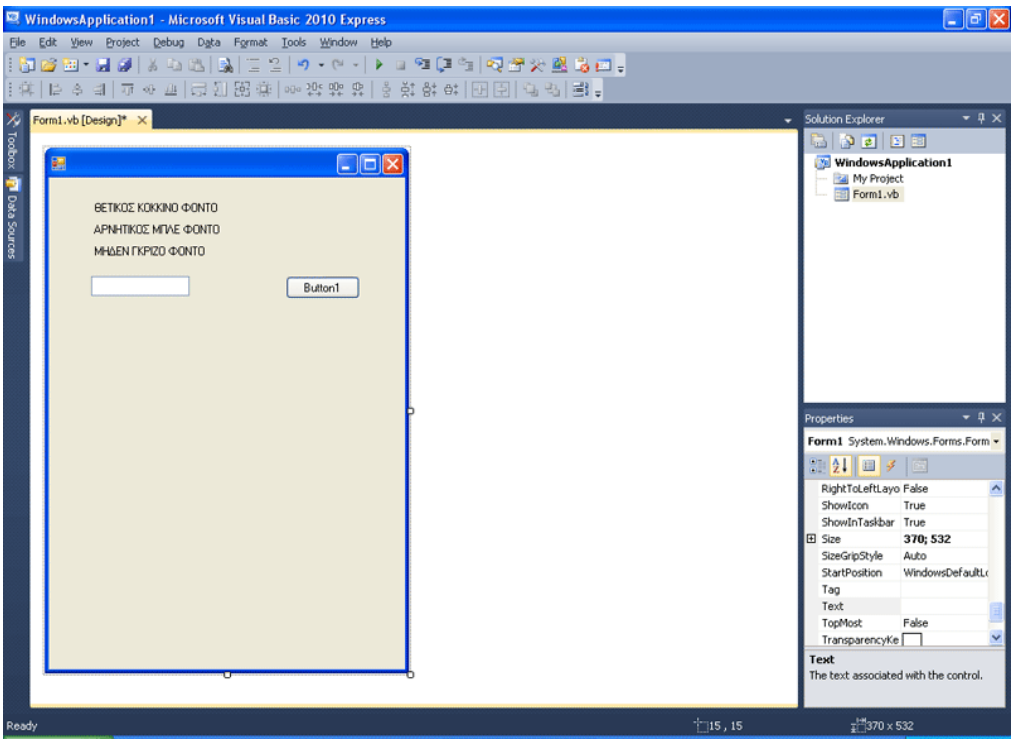
Το κείμενο της Label θα είναι "ΘΕΤΙΚΟΣ ΚΟΚΚΙΝΟ ΦΟΝΤΟ"



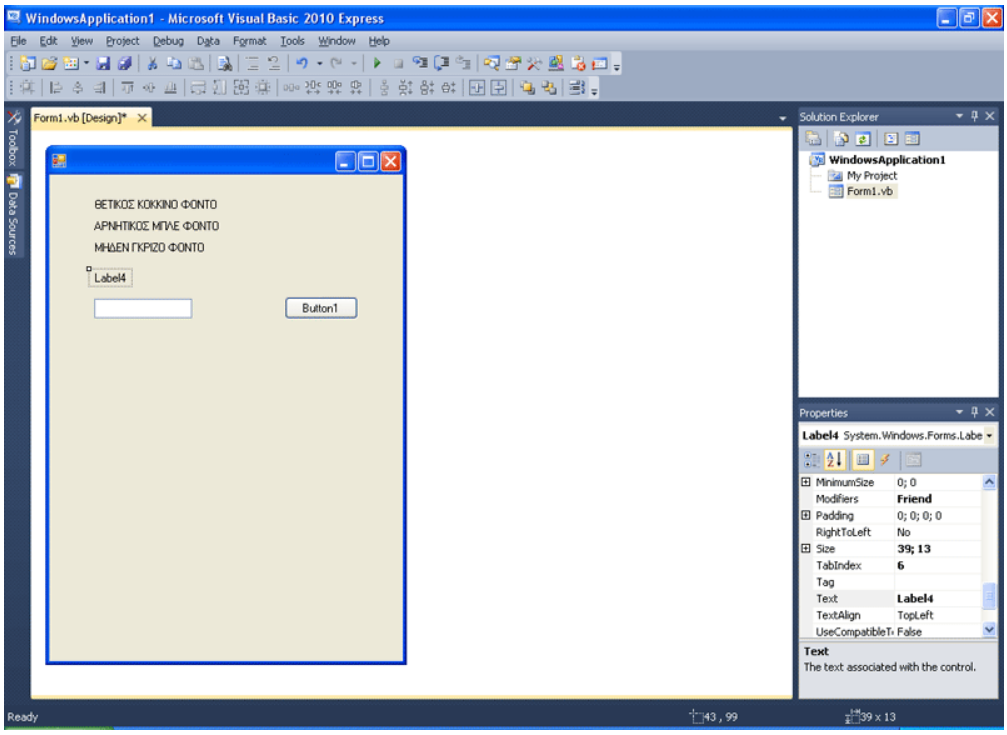
Βάζουμε από κάτω από τη Label1 δύο ακόμα Labels.



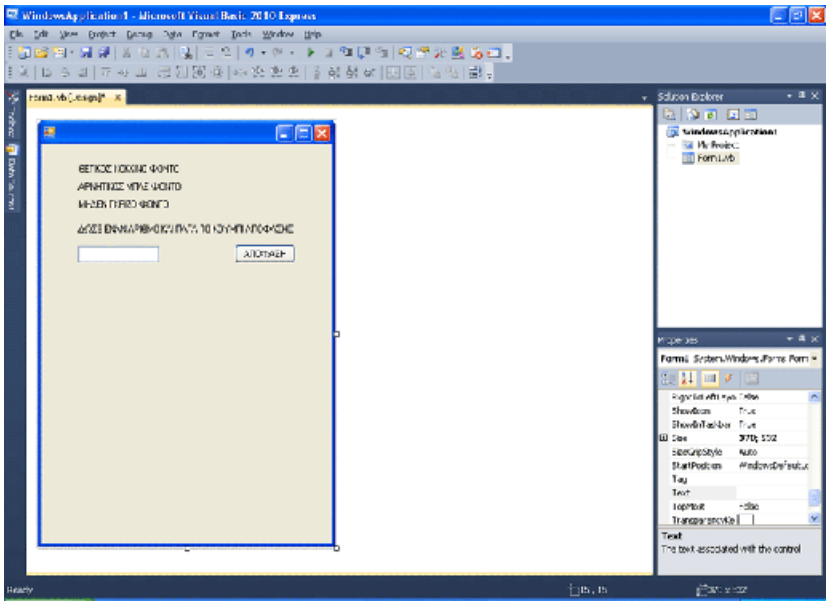
Το κείμενο της Label2 είναι "ΑΡΝΗΤΙΚΟΣ ΜΠΛΕ ΦΟΝΤΟ"
ενώ της Label3 είναι "ΜΗΔΕΝ ΓΚΡΙΖΟ ΦΟΝΤΟ"



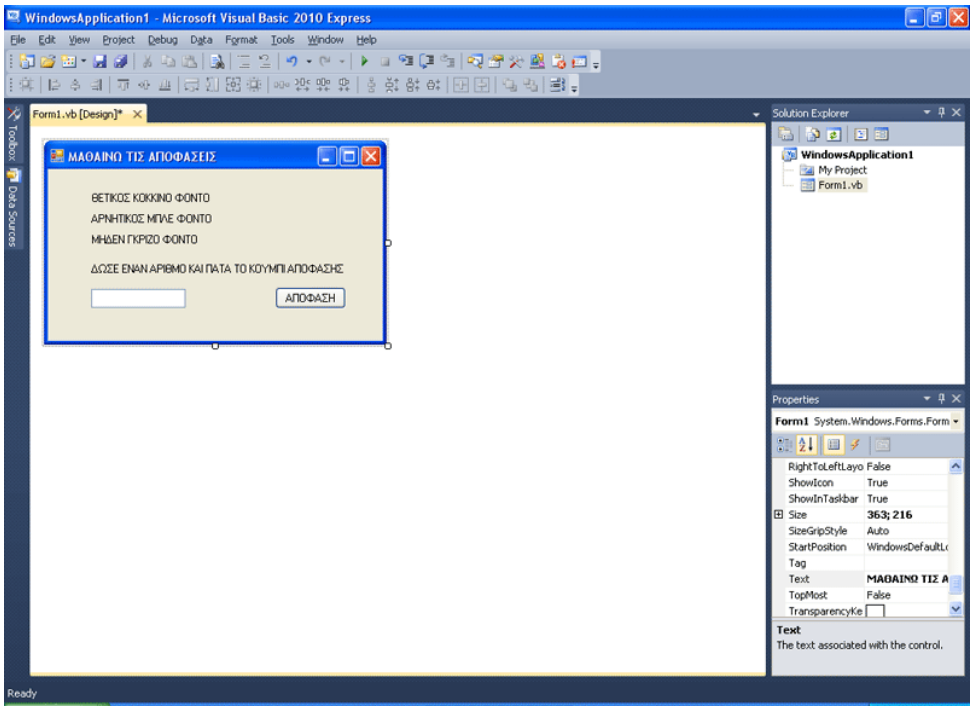
Από κάτω βάζουμε ακόμα μία Label. Αν χρειαστεί, μετακινούμε τα υπόλοιπα αντικείμενα.



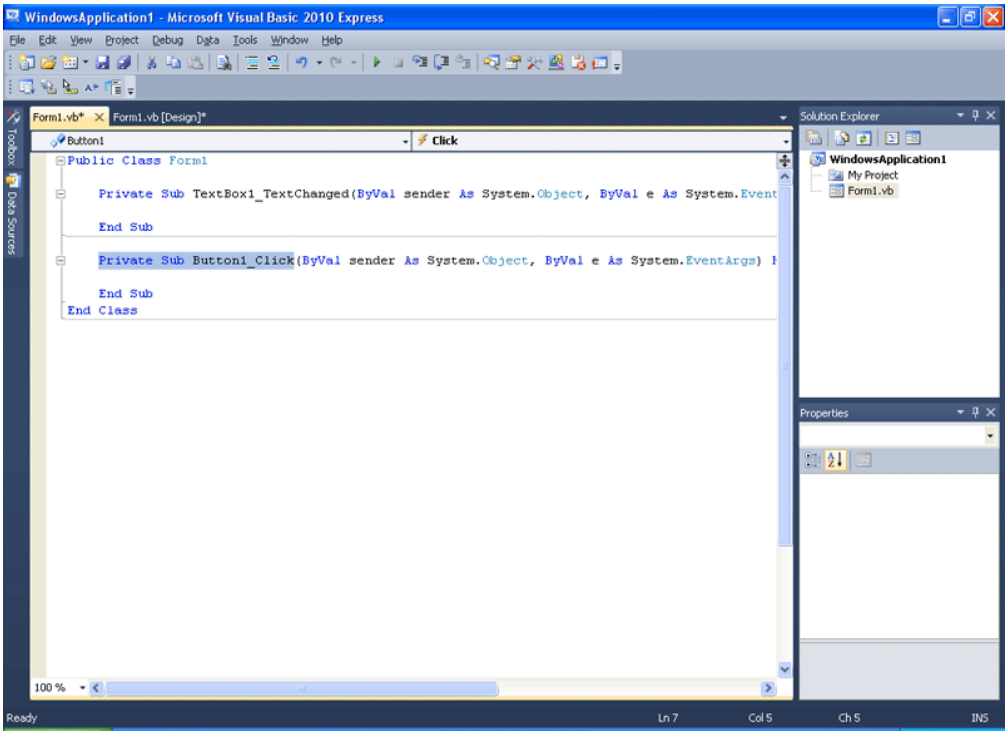
Το κείμενό της είναι "ΔΩΣΕ ΕΝΑΝ ΑΡΙΘΜΟ ΚΑΙ ΠΑΤΑ ΤΟ ΚΟΥΜΠΙ ΑΠΟΦΑΣΗΣ" Στην ιδιότητα text του κουμπιού, γράφουμε "ΑΠΟΦΑΣΗ".



Μικραίνουμε τη φόρμα, ώστε να γίνει πρακτική και όμορφη. Την ιδιότητα text της φόρμας την αλλάζουμε από Form1 σε "ΜΑΘΑΙΝΩ ΤΙΣ ΑΠΟΦΑΣΕΙΣ" Παρατηρήστε για άλλη μια φορά, πόση ώρα μας πήρε, να φτιάξουμε το περιβάλλον, επειδή δεν έχουμε προτυποποιήσει το πρόγραμμα από πριν. Φανταστείτε τώρα ένα πρόγραμμα να χρησιμοποιεί περισσότερα αντικείμενα ελέγχου, να αποτελείται από περισσότερες από μία φόρμες, και τα λοιπά. Αργότερα που θα αναφερθούμε στην προεργασία πριν την υλοποίηση του προγράμματος, αν θυμηθείτε πόση ώρα μας πήρε να φτιάξουμε κάτι τόσο απλό, θα καταλάβετε τη σπουδαιότητα του θέματος που θα αναλύουμε και τη χρησιμότητα των συμβουλών που θα σας δώσουμε.



Για την ώρα, κάνουμε διπλό κλικ στο κουμπί Button1 "ΑΠΟΦΑΣΗ", για να μπούμε στη σύνταξη κώδικα. Είμαστε έτοιμοι να γράψουμε τον πρώτο κώδικα λήψεως απόφασης!

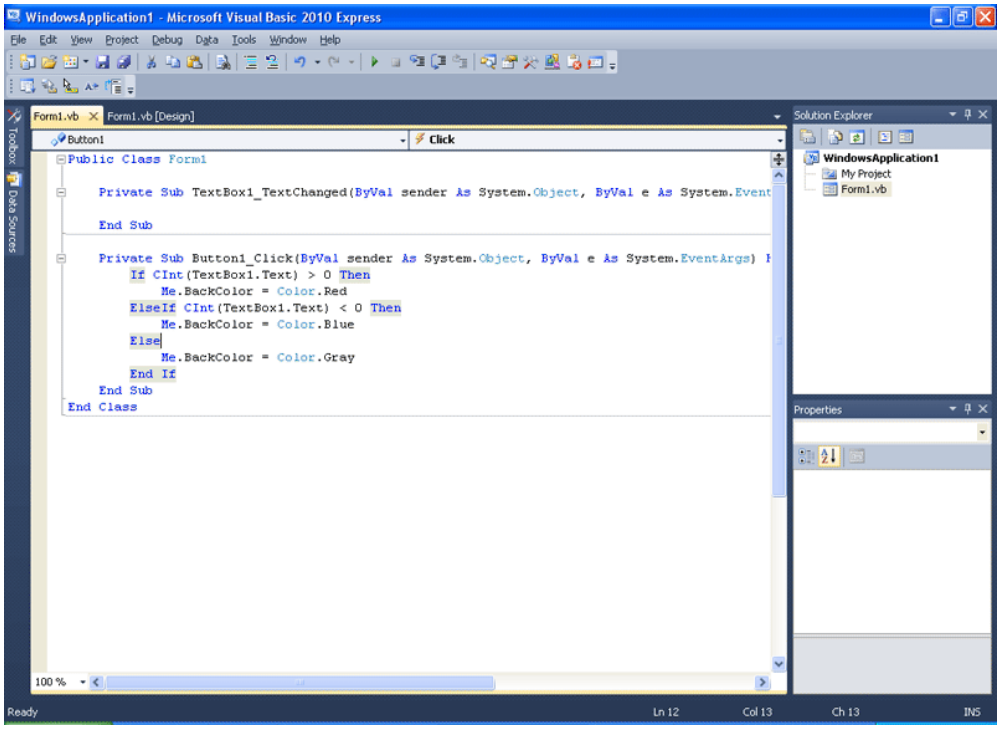


Έτσι λοιπόν, μεταξύ Private Sub Button1_Click(...) και End Sub
όπως βλέπουμε στην εικόνα γράφουμε το παρακάτω και μετά θα σας το εξηγήσουμε.

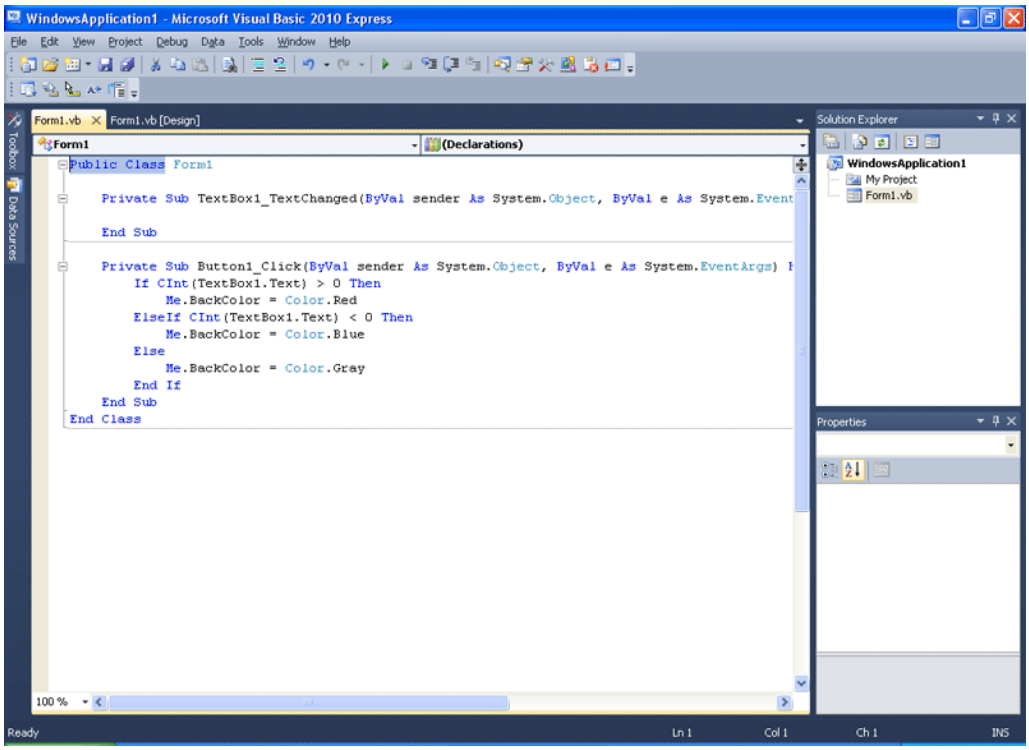
```
If  CInt(TextBox1.Text) > 0 Then
   Me.BackColor = Color.Red
ElseIf  CInt(TextBox1.Text) < 0 Then
   Me.BackColor = Color.Blue
Else
   Me.BackColor = Color.Gray
End If
```

Επεξηγούμε τώρα:
If CInt(TextBox1.Text) > 0 Then
Αν αφού κανείς ακέραιο το κείμενο του Textbox1 σου βγει αριθμός μεγαλύτερο του μηδενός, τότε...
Me.BackColor = Color.Red
Το φόντο μου, παίρνει χρώμα κόκκινο
ElseIf CInt(TextBox1.Text) < 0 Then
Αλλιώς αν αφού κάνεις ακέραιο το κείμενο του Textbox1 σου βγει αριθμός μικρότερο του μηδενός, τότε
Me.BackColor = Color.Blue
Το φόντο μου παίρνει χρώμα μπλε
Else
Αλλιώς
Me.BackColor = Color.Gray
Το φόντο μου παίρνει χρώμα γκρίζο.
End If
Τέλος ΑΝ

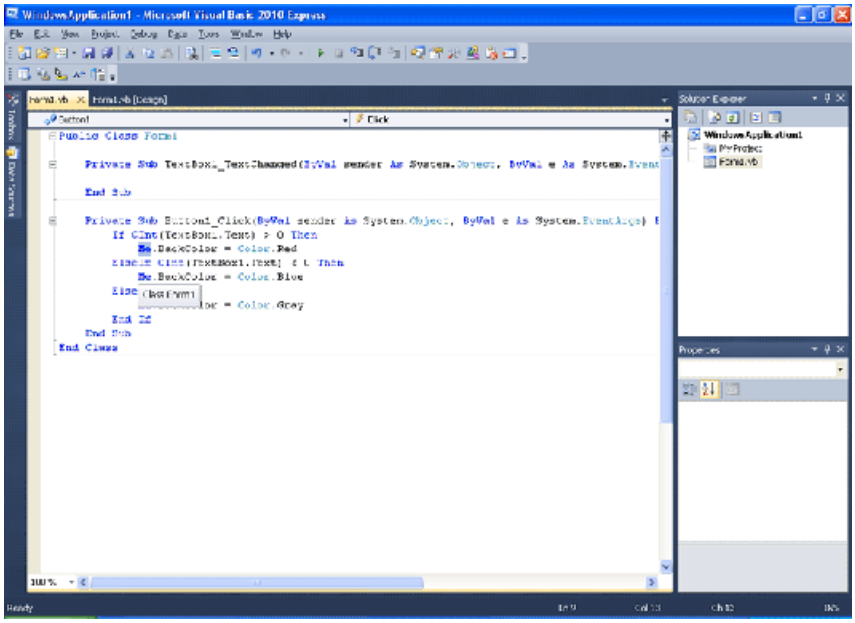
Τι ακριβώς συμβαίνει με αυτό το "me" όμως. Το φόντο μου; Ποιανού το φόντο;



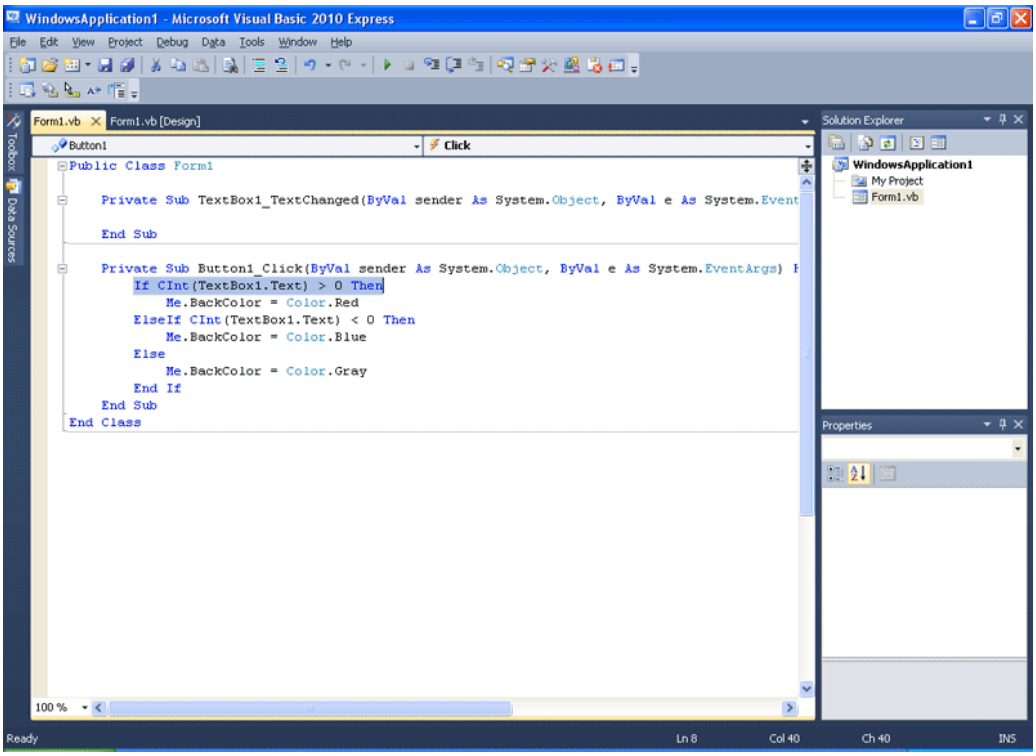
Δείτε πάνω πάνω στον κώδικα, πως ξεκινάνε όλα. Όλα συμβαίνουν μέσα στην Κλάση Form1.



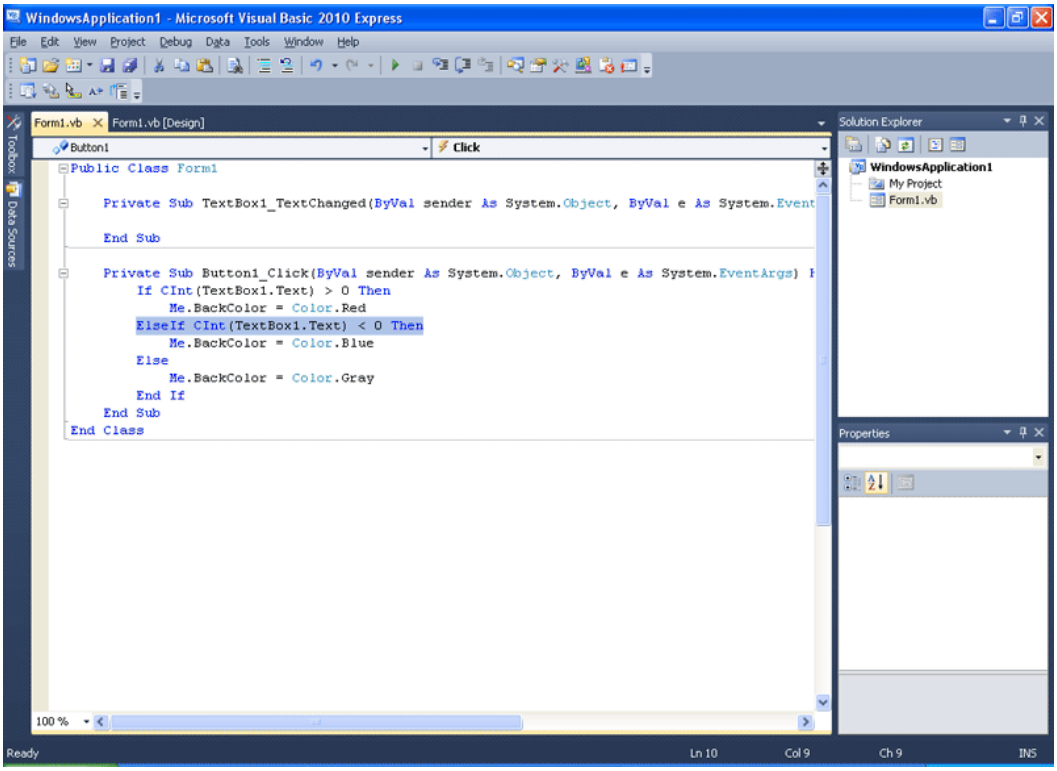
Αν πατήσετε πάνω στο Me η Visual Basic θα σας εμφανίσει ένα πολύ μικρό ταμπελάκι, που γράφει Class Form1. Το Me λοιπόν αναφέρεται στη φόρμα μέσω του εαυτού της, διότι ο κώδικας που γράφουμε βρίσκεται μέσα της.



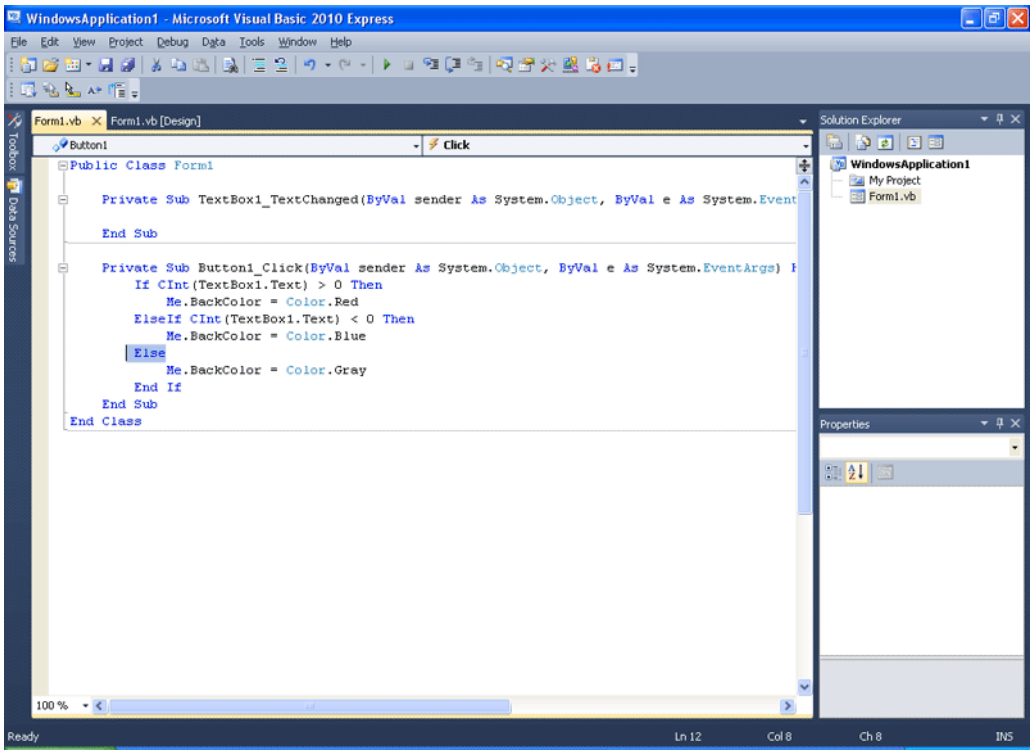
Παίρνουμε λοιπόν το ενδεχόμενο αν είναι θετικός ο αριθμός.



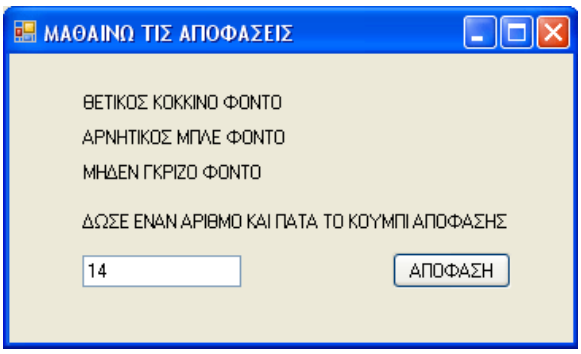
Παίρνουμε κι ενδεχόμενο αρνητικού.



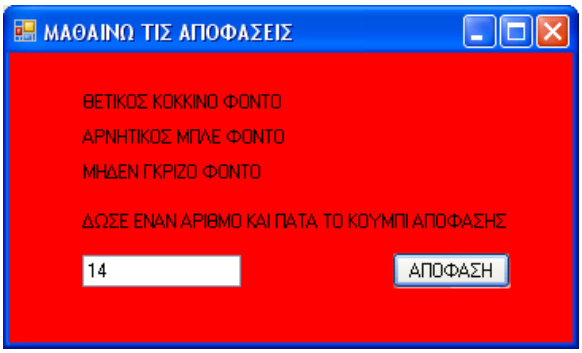
Στο Else μας μένει μόνο η περίπτωση του μηδενός.



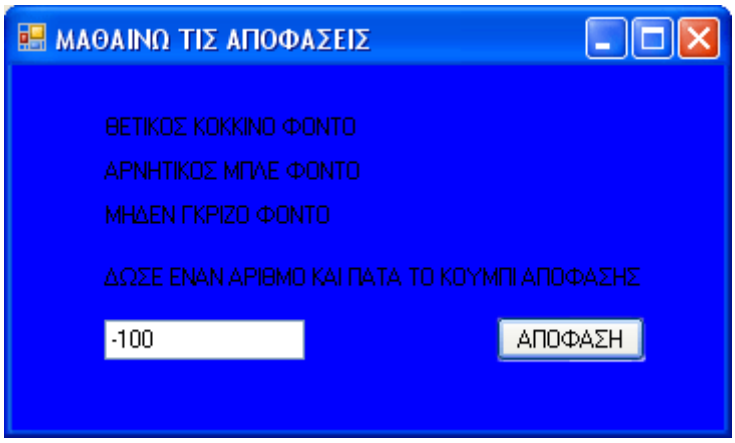
Ας το τρέξουμε βήμα βήμα, να το ελέγξουμε.
Τρέξτε το και βάλτε τον αριθμό 14 στο Textbox και πατήστε το κουμπί "ΑΠΟΦΑΣΗ".



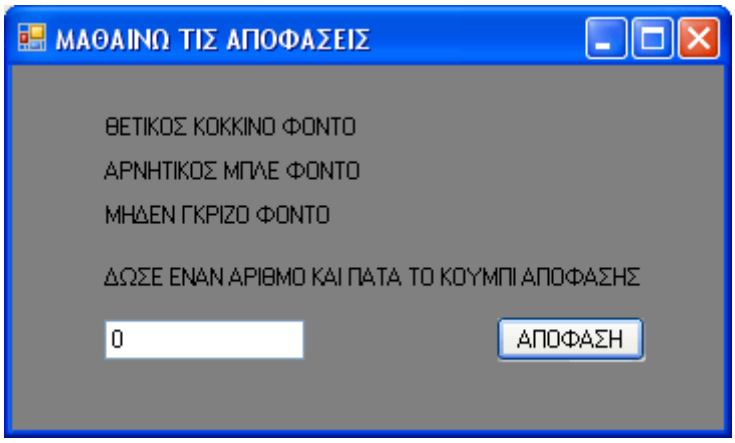
Κόκκινο φόντο στη φόρμα όπως θέλαμε!



Βάλτε στο Textbox "-100". Πατώντας το κουμπί "ΑΠΟΦΑΣΗ" το φόντο γίνεται μπλε. Παρατηρήστε ότι το πρόσημο, αν και δεν είναι αριθμός, δεν παρουσίασε σφάλμα.

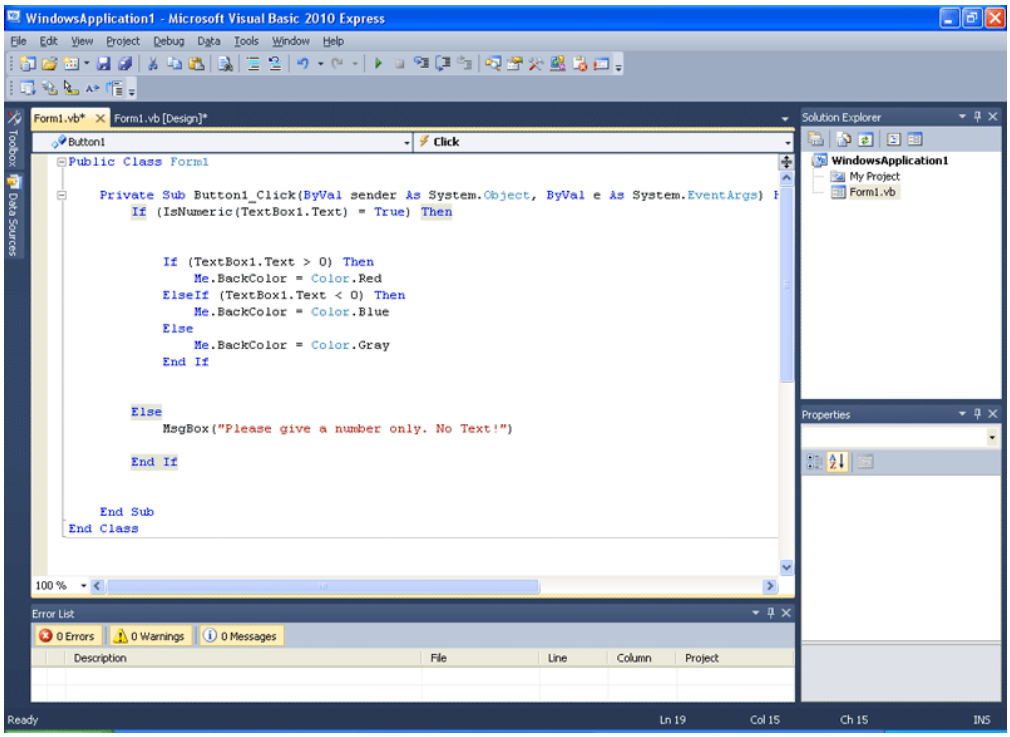


Βάλτε "0" τώρα, κι όταν πατήσετε το κουμπί "ΑΠΟΦΑΣΗ", το φόντο θα γίνει γκρίζο.



Λειτουργεί σωστά το πρόγραμμα! Είναι όμως καλό; Όχι! Όχι ακόμα δηλαδή! Αν βάλετε κείμενο στο Textbox, ο debugger θα σας βγάλει πρόβλημα. Το πρόγραμμα κοινώς θα κολλήσει. Θυμηθείτε το "Hello!!!" που βάλαμε στο προηγούμενο κεφάλαιο στο πρόγραμμα. Ο χρήστης μπορεί να βάζει κείμενο. Μπορούμε να αποτρέψουμε το πρόβλημα αυτό; Πλέον ναι, με έναν έλεγχο και μια λήψη απόφασης φυσικά! Όπως θα δείτε στην παρακάτω εικόνα, ο κώδικας αλλάζει!

```
Private Sub Button1_Click(...)
  If  (IsNumber(TextBox1.Text) = True) Then
    If  CInt(TextBox1.Text) > 0 Then
      Me.BackColor = Color.Red
    ElseIf  CInt(TextBox1.Text) < 0 Then
      Me.BackColor = Color.Blue
    Else
      Me.BackColor = Color.Gray
    End If
  Else
   MsgBox("Please give a number only. No text!")
  End If
End Sub
```



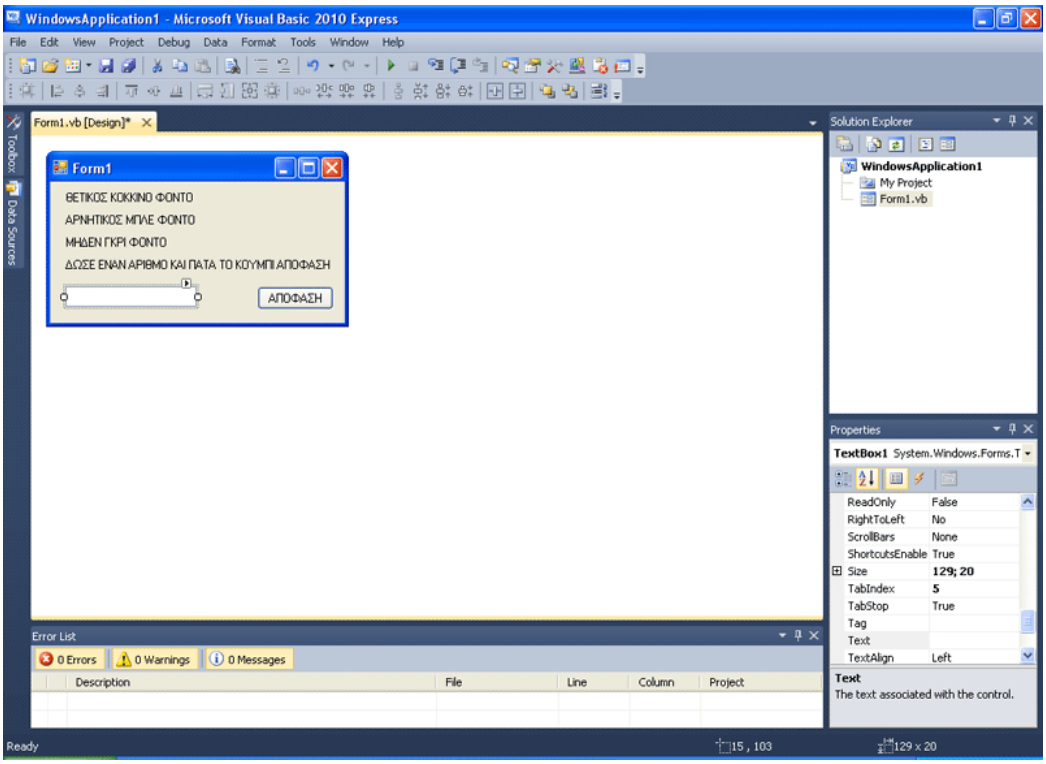
Τι σημαίνει ο κώδικάς αυτός; Τι κάναμε ακριβώς; Επιγραμματικά για αρχή:
Λέμε στο πρόγραμμα να κάνει έναν έλεγχο πριν κάνει οτιδήποτε άλλο.
ΑΝ το Textbox1 έχει μέσα του αριθμό με ή χωρίς πρόσημο, με ή χωρίς μία υποδιαστολή,
ΤΟΤΕ κάνε τους ελέγχους μεγαλύτερο ή μικρότερο του μηδενός
ΑΛΛΙΩΣ Εμφάνισε μήνυμα με μία οδηγία, ότι μόνο αριθμός επιτρέπεται. Η οδηγία μπορεί να είναι και στα Ελληνικά.
Πιο συγκεκριμένα:
If (IsNumber(TextBox1.Text) = True) Then

Η συνάρτηση IsNumber() είναι αληθής (δηλαδή 1) μόνο αν το περιεχόμενο της παρένθεσης, είναι αριθμός. Ο αριθμός αυτός μπορεί να έχει ένα πρόσημο - μπροστά του ενώ επίσης μπορεί να περιέχει και μία υποδιαστολή (τελεία). Γιατί τη χρησιμοποιούμε εδώ όμως; Κατά το παράδειγμα που δώσαμε για τις μεταβλητές, σε προηγούμενο κεφάλαιο, το κείμενο (string) μπαίνει σε κατσαρόλα. Διότι ως κατσαρόλα, περιγράψαμε τη μεταβλητή string. Τώρα η συνάρτηση IsNumber() τι κάνει; Βλέπει αν αυτό που περιέχει η κατσαρόλα, μπορεί να χωρέσει σε κανάτα ουσιαστικά. Υπενθυμίζουμε πως, ως ένα μήλο συμβολίζουμε έναν αριθμό, με δεκαδικό μέρος. Μπορεί η κατσαρόλα να περιέχει ένα μήλο; Σαφώς! Κι εδώ ελέγχουμε αν χωράει ή όχι σε μια κανάτα. Είναι αριθμός; "Is it a number?" στα Αγγλικά. Αυτό ελέγχει η συνάρτηση IsNubmer() λοιπόν. Προχωράμε λοιπόν με τον υπόλοιπο κώδικα.

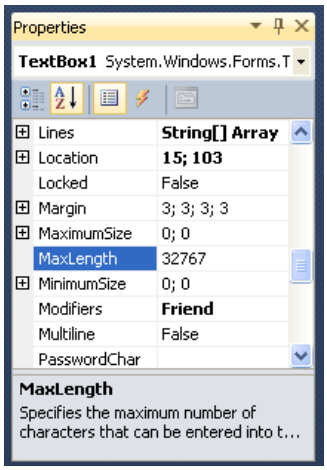
Αν είναι αληθής λοιπόν η IsNumber() τότε θα εκτελέσει το παρακάτω,
το οποίο είναι ακριβώς ο αρχικός κώδικας που γράψαμε.
If CInt(TextBox1.Text) > 0 Then
Me.BackColor = Color.Red
ElseIf CInt(TextBox1.Text) < 0 Then
Me.BackColor = Color.Blue
Else
Me.BackColor = Color.Gray
End If

Else
ΑΛΛΙΩΣ (αν δηλαδή είναι κείμενο, αυτό που περιέχει το Textbox και όχι μόνο αριθμός)
MsgBox("Please give a number only. No text!")
Εμφάνισε ένα μικρό παράθυρο-κουτί μηνύματος με το μήνυμα:
"Please give a number only. No text!"
End If
Τέλος ΑΝ

Η MsgBox() είναι μία συνάρτηση που εμφανίζει κουτιά με μηνύματα.



Τώρα αν τρέξετε το πρόγραμμα, αν βάλετε κείμενο, δε θα κολλήσει. Θα σας ζητήσει απλώς να το αλλάξετε με έναν αριθμό. Είναι σταθερό 100% το πρόγραμμά μας; Σχεδόν! Υπάρχει και ένας άλλος κίνδυνος. Επιλέξτε το TextBox1 κάνοντας κλικ πάνω του. Πηγαίνετε στις ιδιότητες στου TextBox1 κάτω αριστερά. Βρείτε το MaxLength. Ο χρήστης δε μπορεί να βάλει κείμενο, αλλά μπορεί να βάλει αριθμό μήκους 32767 ψηφίων! Αυτό δεν ακούγεται και τόσο καλό. Αλλάξτε το MaxLength σε 10. Τώρα ο χρήστης μπορεί να βάλει μόνο 10 χαρακτήρες στο Textbox. ΤΩΡΑ το πρόγραμμα είναι σταθερό σαν βράχος! Τρέξτε το και δοκιμάστε το.



Εναλλακτικά θα μπορούσατε να βάλετε μια ακόμα If που να ελέγχει αν ο αριθμός είναι μεγαλύτερος από έναν καθορισμένο αριθμό ή μικρότερος από έναν άλλο. Εναλλακτικά επίσης, θα μπορούσατε να βάλετε μια If που θα έλεγχε το μήκος αυτού που πληκτρολόγησε ο χρήστης και αν ξεπερνούσε τους 10 χαρακτήρες, να του έβγαζε κουτί μηνύματος δίνοντάς του την οδηγία να μην ξεπερνά τους 10 χαρακτήρες ότι γράψει. Ωστόσο, αυτό θα αύξανε την πολυπλοκότητα στον κώδικα όπως καταλαβαίνετε. Διότι ήδη έχουμε μια If μέσα σε μία άλλη If. Αν βάλουμε και τρίτη δίχως λόγο, κάνουμε άδικα τη ζωή μας δύσκολη, επιβαρύνουμε άδικα τον υπολογιστή, κάνουμε άδικα πιο πολύπλοκο τον αλγόριθμο. Είναι καλύτερα να βάλουμε τον περιορισμό, από τις ιδιότητες εξ αρχής και να γλιτώσουμε τόσο κόπο; Σαφώς! Γι' αυτό υπάρχουν οι τόσες ιδιότητες. Για να μας κάνουν τη ζωή μας ευκολότερη.

Οι IF εντολές που είδαμε ως τώρα, ήταν της μορφής:
If (expression1) THEN
.
.
.
Εντολές που θα εκτελεστούν αν η expression1 είναι ΑΛΗΘΗΣ
.
.
.
ElseIf (expression2) THEN
.
.
.
Εντολές που θα εκτελεστούν αν η expression2 είναι ΑΛΗΘΗΣ
.
.
.
Else
.
.
.
Εντολες που θα εκτελεστούν αν η expression1 και η expression2 είναι ψευδείς.
.
.
.
EndIf

Τι γίνεται όμως αν θέλω να ελέγξω αν ισχύουν, ή όχι, ΤΑΥΤΟΧΡΟΝΑ δύο expression;
Πώς μπορεί να γίνει αυτό; Την απάντηση σε αυτό το ερώτημα και όχι μόνο, δίνουν οι λογικοί τελεστές.

ΛΟΓΙΚΟΙ ΤΕΛΕΣΤΕΣ
Υπενθυμίζουμε, πως τελεστής είναι με απλά λόγια, ένα σύμβολο, που υποδηλώνει μια πράξη που θα γίνει μεταξύ δύο τελεστέων. Τελεστέοι εδώ είναι συνθήκες στην ουσία!
Το αποτέλεσμα ενός λογικού τελεστή μπορεί να είναι μόνο ένα από τα παρακάτω δύο: ΑΛΗΘΕΣ, ΨΕΥΔΕΣ. Συνηθίζουμε το αληθές να το συμβολίζουμε με 1 και το ψευδές με 0, όπως ήδη αναφέραμε. Τι εννοούμε με αυτό;
Έστω θέλω να κάνω έλεγχο αν είμαι άνθρωπος. Η τελεστέα συνθήκη δηλαδή, είναι: "Είμαι άνθρωπος". Είμαι; Ναι. Οπότε η έκφραση "Είμαι άνθρωπος" είναι αληθής. Συνεπώς (Είμαι άνθρωπος) = 1.
Ας δούμε τι κάνει ο πρώτος λογικός τελεστής.
Ο τελεστής άρνησης, NOT (δηλαδή ΟΧΙ) έχει έναν μόνο τελεστέο και δρα πάνω στον τελεστέο αυτό, έτσι ώστε να δίνει ως τιμή, την αντίθετη από την τιμή του τελεστέου. Με άλλα λόγια NOT σε Αληθή έκφραση, δίνει αποτέλεσμα "Ψευδές". Η NOT σε Ψευδή έκφραση, δίνει αποτέλεσμα "Αληθές".

(ΟΧΙ (Είμαι άνθρωπος)) = 0 λοιπόν, αν αναφέρεται σε εμένα που είμαι. Γιατί;
Διότι (ΟΧΙ (Είμαι άνθρωπος)), ισούται με (ΟΧΙ (1))
ΟΧΙ 1 = 0
ΟΧΙ ΑΛΗΘΕΣ = ΨΕΥΔΕΣ!
Κατανοητό;
Αν η ίδια συνθήκη είχε ως αντικείμενο ένα σκύλο αντί για εμένα, τότε θα ήταν ψευδής! "Είμαι άνθρωπος" για το σκύλο είναι ψευδής, άρα 0. Αν της εφαρμοστεί ο τελεστής NOT, θα γίνει:
NOT(ΕΙΜΑΙ ΑΝΘΡΩΠΟΣ)
οπότε NOT(0) = 1!

Συνοψίζουμε λοιπόν:
Όταν αποδίδουμε τον τελεστή NOT σε True (1) συνθήκες, το αποτέλεσμα είναι False (0).
Επίσης όταν αποδίδουμε τον τελεστή NOT σε False συνθήκες, το αποτέλεσμα είναι True .
Όταν μια συνθήκη μπορεί να γίνει True ή False λοιπόν, η NOT αποδίδει το αντίθετο!
Έστω "Α" λοιπόν μια συνθήκη που δίνεται σαν είσοδος στον τελεστή NOT. Αν είναι αληθής έχει τιμή 1 κι αν είναι ψευδής έχει τιμή 0.
Παρακάτω φαίνεται ο πίνακας αληθείας του τελεστή NOT. Πίνακας αληθείας, είναι ένας πίνακας που μας διευκολύνει να δούμε τι αποτέλεσμα δίνει ένας τελεστής ανάλογα τις τιμές των εισόδων του.

Είσοδος	Έξοδος
Α	ΟΧΙ Α
0	1
1	0

Μπορεί να φάνηκε πως επαναλαμβανόμαστε, αλλά ουσιαστικά αυτό που κάνουμε, είναι να σας δείχνουμε με διάφορους τρόπους τη λογική του τελεστή NOT. Είναι πολύ σημαντικό να γίνει κατανοητή η λογική των τελεστών αυτών.

Ο επόμενος τελεστής που θα δούμε είναι ο AND (ΚΑΙ στα Ελληνικά). Αυτός ο τελεστής έχει δύο εισόδους. Με άλλα λόγια, μελετάει ταυτόχρονα δύο συνθήκες, έστω Α και Β.
Για να πάρει τιμή True ο τελεστής AND, θα πρέπει ΚΑΙ ΟΙ ΔΥΟ συνθήκες να είναι True.
Παρακάτω θα δείτε και τον πίνακα αληθείας.
Ένα παράδειγμα:
Έστω θέλω να κάνω έναν έλεγχο, για το αν η Σοφία είναι ενήλικη γυναίκα. Αυτό θα αληθεύει αν και μόνο αν είναι πάνω από 18 χρονών και το φύλο της είναι γυναίκα.
Έστω Α η συνθήκη "Ηλικία >= 18"
Έστω Β η συνθήκη "Φύλο=Γυναίκα"
Οπότε (Είναι Ενήλικη Γυναίκα) = [(Ηλικία >= 18) AND (Φύλο=Γυναίκα)]
Συνεπώς (Είναι Ενήλικη Γυναίκα) = [A AND B]
Εδώ Α =1 και Β =1.
Οπότε A AND B = True.
Οπότε για τη Σοφία (Είναι Ενήλικη Γυναίκα) = True.
Για το Μιχάλη που είναι 20 ετών, τι βγάζει η σύνθετη συνθήκη:
[(Ηλικία >= 18) AND (Φύλο=Γυναίκα)]
[(1) AND (0)] = 0
Τι είπαμε; Θα πρέπει ΚΑΙ ΟΙ ΔΥΟ συνθήκες να είναι True για να δώσει η AND αποτέλεσμα True (1)
Ο Μιχάλης λοιπόν ΔΕΝ ΕΙΝΑΙ ενήλικη γυναίκα.
Πότε είπαμε είναι ενήλικη γυναίκα κάποιο άτομο; Όταν αληθεύουν ΚΑΙ ΤΑ ΔΥΟ!
Παρακάτω βλέπετε τον πίνακα αληθείας του τελεστή AND.

Είσοδοι		Έξοδος
A	B	A ΚΑΙ B
0	0	0
0	1	0
1	0	0
1	1	1

Ο τελεστής OR (" Ή "διαζευκτικό στα Ελληνικά) έχει δύο εισόδους και δίνει στην έξοδό του True, αν ΤΟΥΛΑΧΙΣΤΟ μία από τις συνθήκες είναι αληθής. Δηλαδή είτε η Α, είτε η Β, είτε και οι δύο αν είναι True, δίνει αποτέλεσμα True. Παρακάτω βλέπετε τον πίνακα αληθείας του.

Είσοδοι		Έξοδος
A	B	A Η' B
0	0	0
0	1	1
1	0	1
1	1	1

Ο τελεστής XOR έχει δύο εισόδους και δίνει στην έξοδό του True, αν ΜΙΑ ΚΑΙ ΜΟΝΟ ΜΙΑ από τις συνθήκες είναι αληθής.

Είσοδοι		Έξοδος
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Δύο ιδιαίτεροι και γρηγορότεροι, κατά περίπτωση, λογικοί τελεστές:

Ο τελεστής AndAlso στην παρακάτω παράσταση (A AndAlso B) λειτουργεί σαν AND, αλλά με τη διαφορά ότι, αν το A βρεθεί False δίνει κατ' ευθείαν τιμή False δίχως να υπολογίσει το B.

Αντίστοιχα ο τελεστής OrElse λειτουργεί σαν OR με τη διαφορά ότι αν για παράδειγμα στην παράσταση (A OrElse B) βρει το A True, δίνει κατ ευθείαν True τιμή χωρίς να υπολογίσει το B.

Πλέον γνωρίζετε αρκετά για να συντάξετε κώδικα πιο προχωρημένων προγραμμάτων. Στο επόμενο κεφάλαιο, θα ασχοληθούμε με τους λογικούς τελεστές στην πράξη, για να γίνει πιο κατανοητή η χρήση τους.

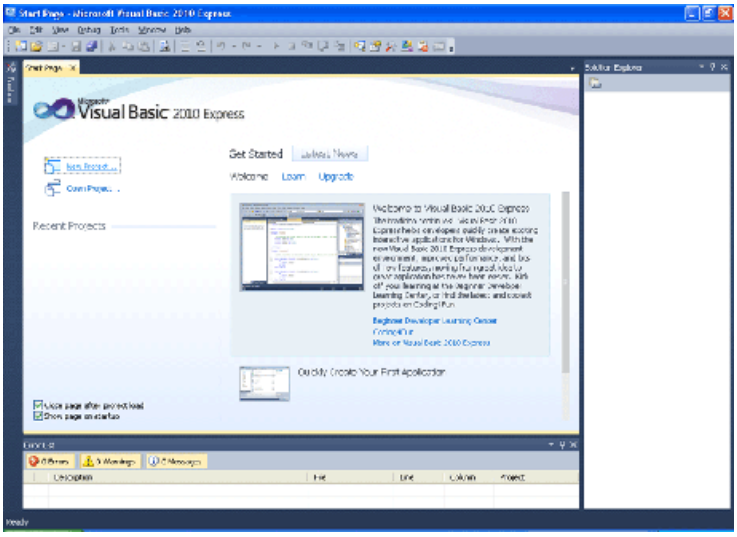
ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό επεξηγούνται οι τελεστές θεωρητικά, και παράλληλα δημιουργείται ένα πρόγραμμα, για επεξήγηση στην πράξη. Παράλληλα διδάσκεται και η εντολή IF μιας κι έχει άμεση πρακτική εφαρμογή με τους λογικούς τελεστές και τις λογικές τιμές. Επίσης αναφέρεται η σπουδαιότητα επαγγελματικής προσέγγισης υλοποίησης ενός προγράμματος, για εξοικονόμηση χρόνου και αποφυγή άδικου κόπου. Οι λογικοί τελεστές και οι έννοιες των λογικών τιμών, επεξηγήθηκαν στο τέλος, διότι θα ακολουθήσει ένα κεφάλαιο με πρακτική χρήση τους.

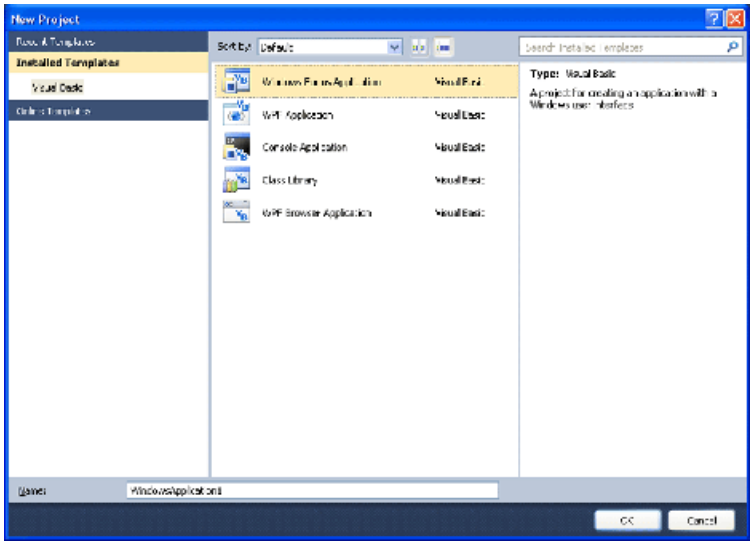
ΚΕΦΑΛΑΙΟ 10
ΛΟΓΙΚΟΙ ΤΕΛΕΣΤΕΣ ΣΤΗΝ ΠΡΑΞΗ

Η εντολή IF με τους λογικούς τελεστές AND και OR: Ας προσπαθήσουμε να πάρουμε ένα υποθετικό μεν σενάριο, αλλά να φτιάξουμε ένα πρόγραμμα προσεγγίζοντας ρεαλιστικά το ζητούμενο. Σε ένα υποθετικό εκπαιδευτικό ίδρυμα δευτεροβάθμιας εκπαίδευσης, στο οποίο διδάσκονται μόνο 4 μαθήματα, Μαθηματικά, Πληροφορική, Φυσική, Χημεία ο υπεύθυνος καθηγητής ενός τμήματος, έχει μπροστά του τις καρτέλες με τις τελικές βαθμολογίες του κάθε μαθητή του τμήματος. Οι βαθμολογίες αυτές είναι ακέραιοι αριθμοί. Επιθυμεί να του φτιάξουμε ένα πρόγραμμα, στο οποίο να εισάγει σε μια φόρμα τις βαθμολογίες ενός μαθητή και πατώντας ένα κουμπί, το πρόγραμμα να τον ενημερώνει αν ο μαθητής προβιβάζεται ή όχι. Στο συγκεκριμένο εκπαιδευτικό ίδρυμα, η βαθμολογία κυμαίνεται από άριστα 20 ως ελάχιστη βαθμολογία το 1. Ο μαθητής προβιβάζεται σύμφωνα με τον κανονισμό του ιδρύματος, αν γράψει σε όλα τα μαθήματα πάνω από 8 και έχει μέσο όρο πάνω από 10.

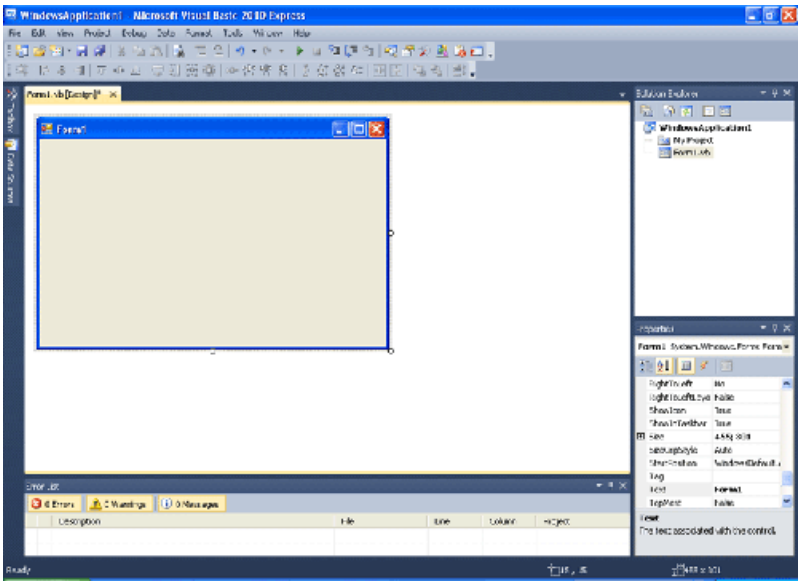
Ξεκινάμε λοιπόν ως συνήθως. New project...



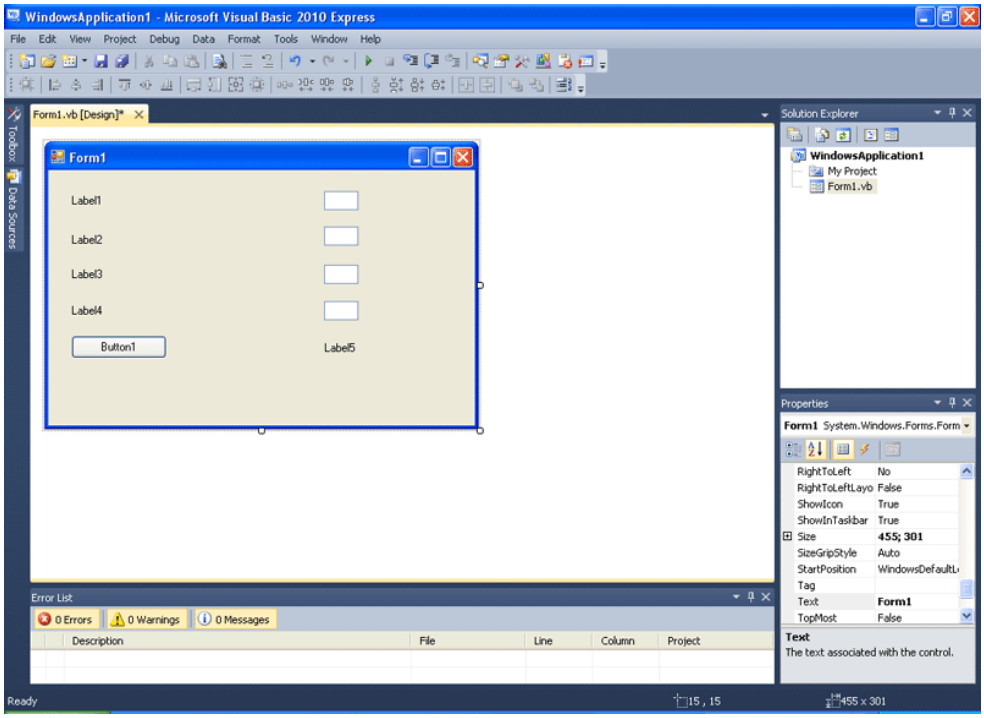
... επιλέγουμε Windows Forms Application...



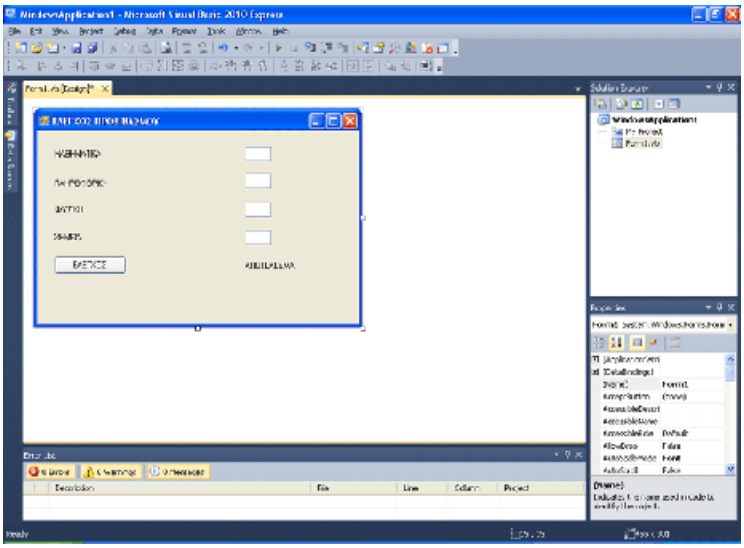
κι εμφανίζεται η φόρμα μας. Τη μεγαλώνουμε λίγο.



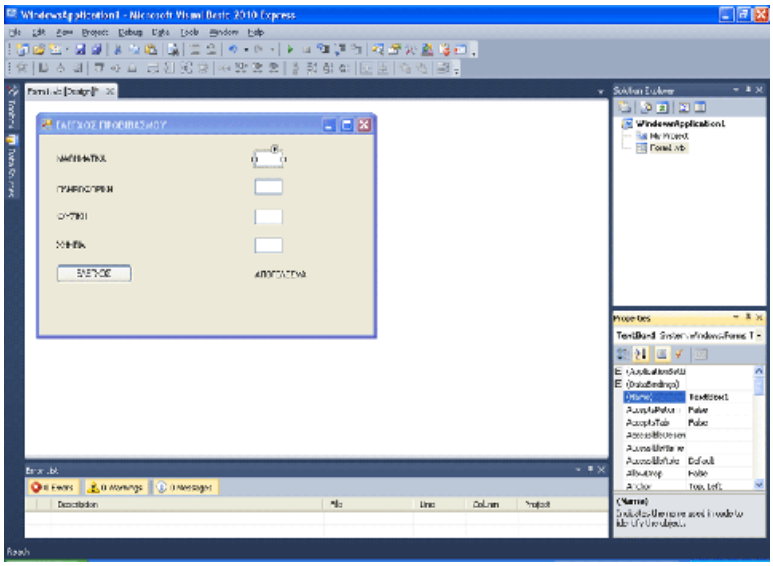
Τοποθετούμε με τον τρόπο που βλέπετε στην παρακάτω εικόνα, πέντε Labels, τέσσερα Textbox κι ένα Button.



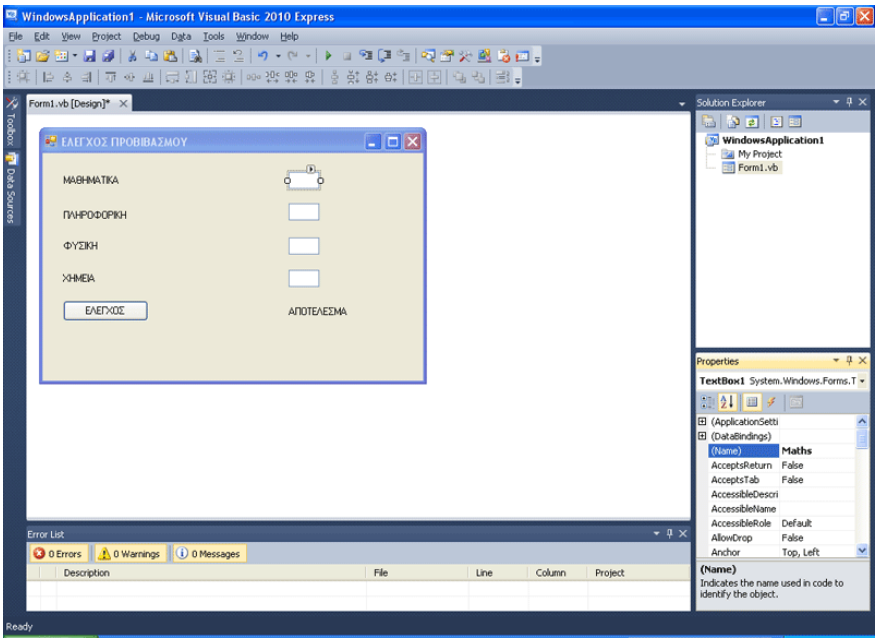
Δίνουμε τίτλο στη φόρμα "ΕΛΕΓΧΟΣ ΠΡΟΒΙΒΑΣΜΟΥ" δίνουμε τιμές στα Text των Labels
"ΜΑΘΗΜΑΤΙΚΑ"
"ΠΛΗΡΟΦΟΡΙΚΗ"
"ΦΥΣΙΚΗ"
"ΧΗΜΕΙΑ"
"ΑΠΟΤΕΛΕΣΜΑ"
όπως βλέπετε στην παρακάτω εικόνα.
Στο κουμπί δίνουμε τιμή στο Text "ΕΛΕΓΧΟΣ"



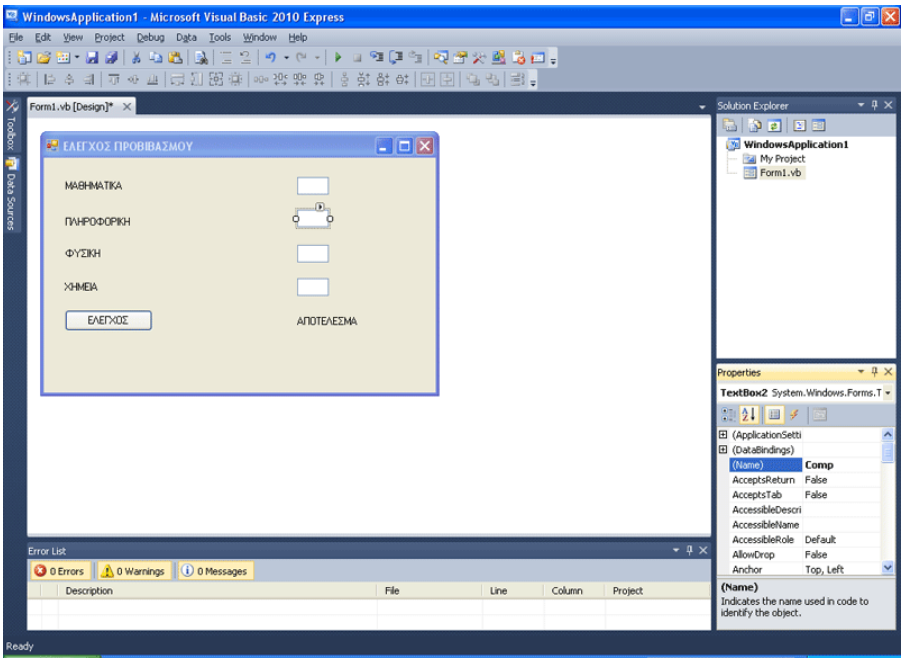
Δώστε τώρα πολλή προσοχή. Κάντε κλικ στο textbox1, όπως δείχνει η φωτογραφία παρακάτω και βρείτε την ιδιότητα (Name), κάτω δεξιά, στο Properties. Πλέον θα πρέπει να μην αφήνουμε τα ονόματα των αντικειμένων ως TextBox1, TextBox2 και τα λοιπά. Θα δείτε στη συνέχεια γιατί σας το λέμε αυτό.



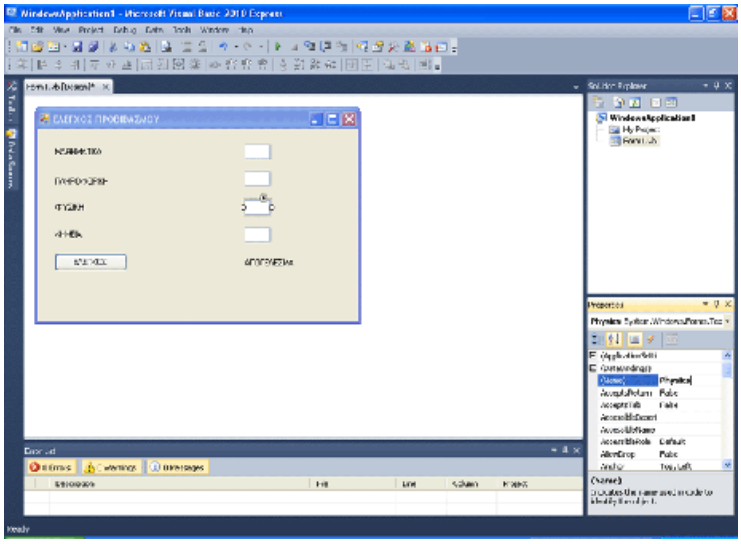
Μετονομάζουμε το TextBox1 σε Maths. Προσοχή! Εδώ πρέπει να χρησιμοποιείτε λατινικούς χαρακτήρες. Κοινώς καλή τακτική είναι να μη χρησιμοποιούμε Ελληνικά. Μπορούμε, αλλά για διάφορους λόγους ήθισται η χρήση λατινικών. Μερικοί από τους λόγους σχετίζονται με την εύκολη εξσφαλμάτωση και τη πιθανότητα συντήρησης στο μέλλον από άλλον προγραμματιστή.



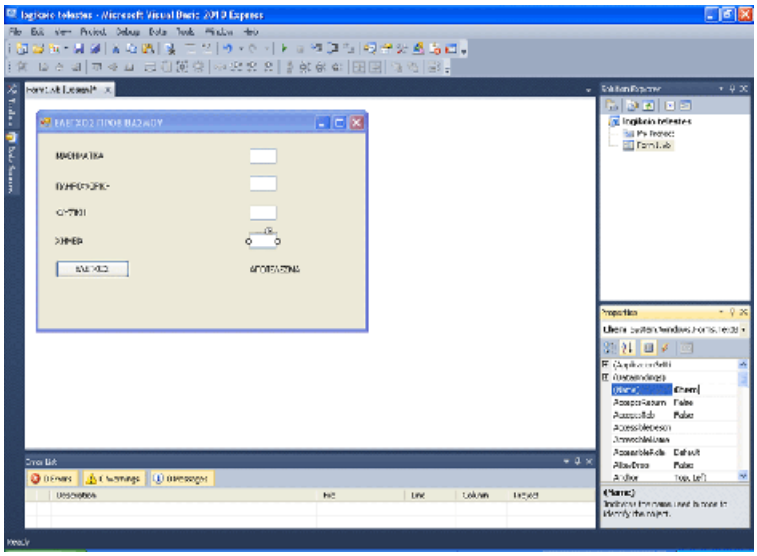
Μετονομάζουμε το TextBox2 σε Comp, με τον ίδιο τρόπο.



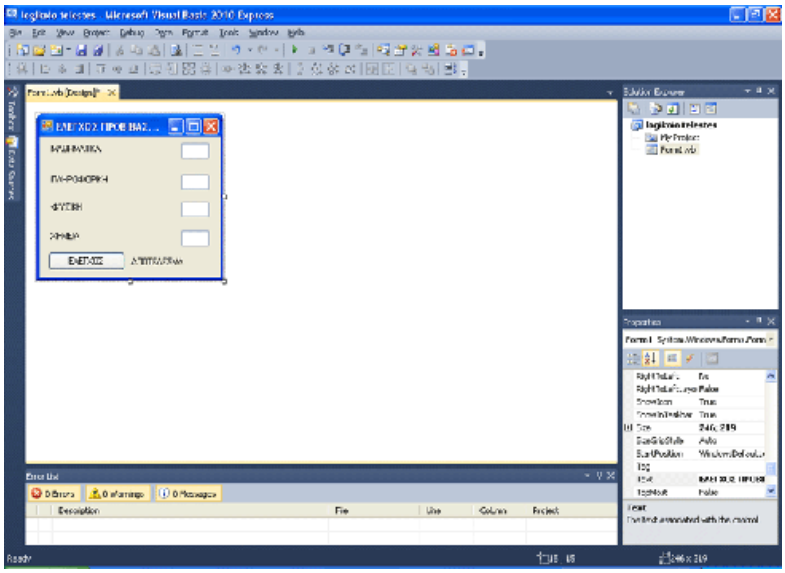
Το TextBox3 σε Physics...



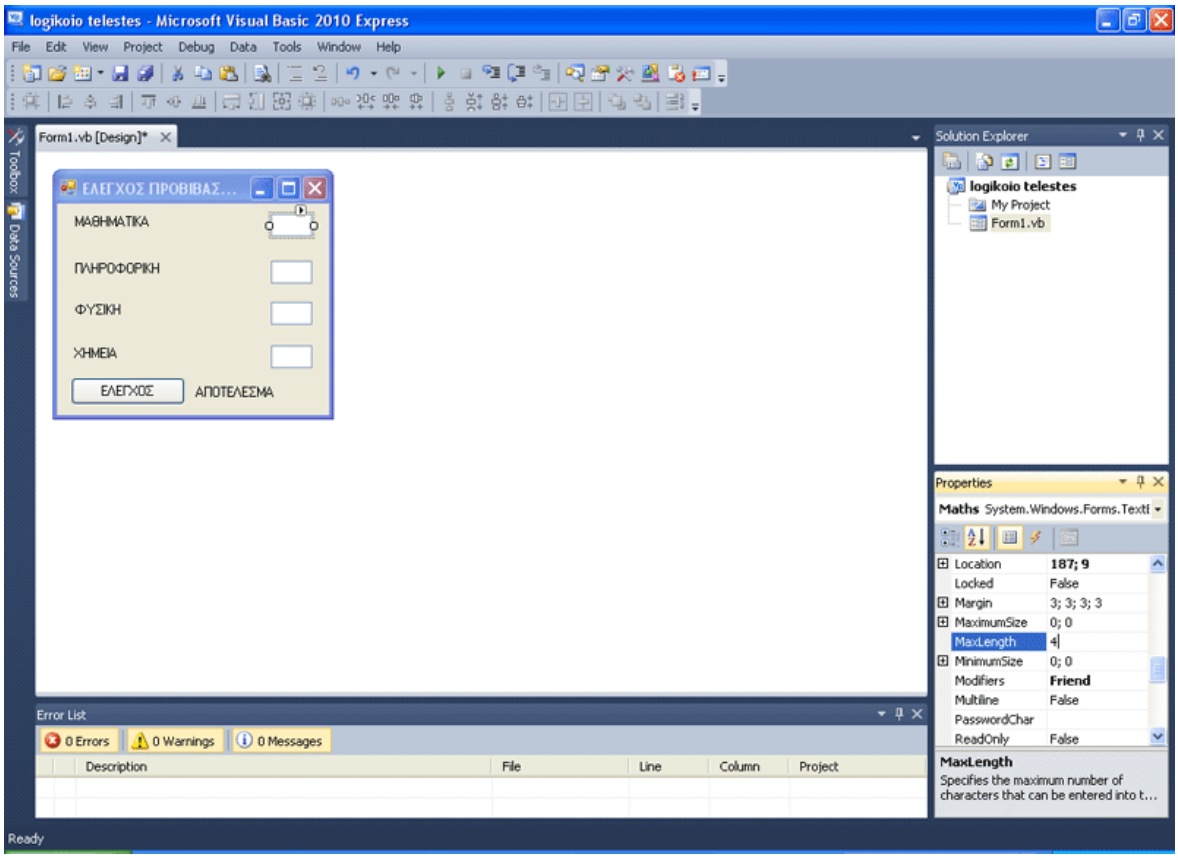
...και το TextBox4 σε Cehm.



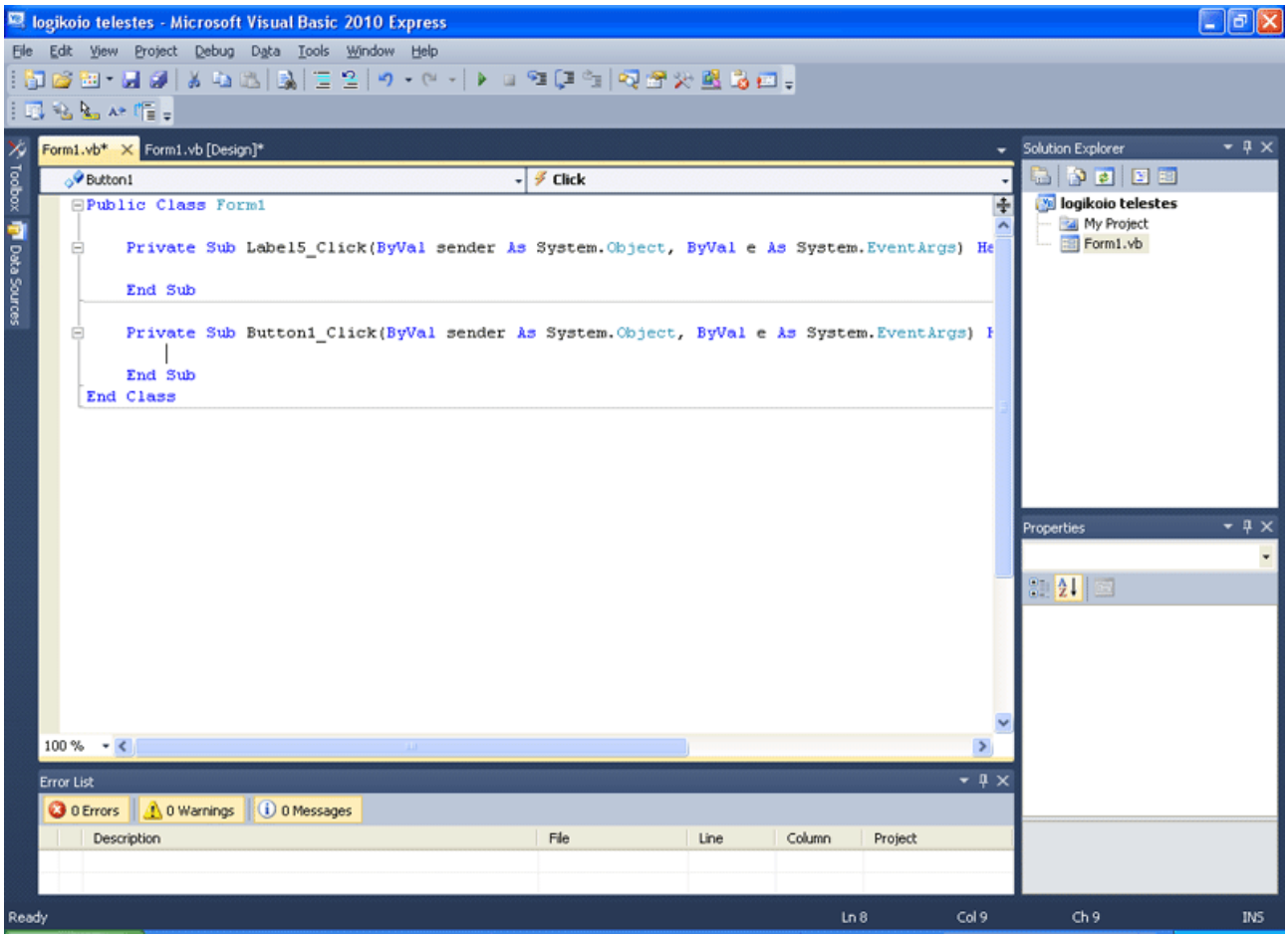
Φτιάχνουμε τη φόρμα πιο όμορφη.



Κι εδώ αρχίζει η προγραμματιστική λογική. Επιλέξτε το Maths και βρείτε την ιδιότητα MaxLength. Κάντε τη 4. Γιατί το κάνουμε αυτό; 4 ψηφία είναι αρκετά για να εισάγουμε ένα τελικό βαθμό. Για την ακρίβεια ακόμα και δύο ψηφία θα ήταν αρκετά, αφού πρόκειται για ακεραίους από το 1 ως το 20. Είτε 4 είτε 2 βάλετε λοιπόν, είναι το ίδιο. Κάντε το και στα Comp, Physics και Chem. Με αυτό τον τρόπο αποφεύγεται η δυνατότητα του χρήστη να εισάγει τεράστια νούμερα. Κάνουμε τώρα διπλό κλικ στο κουμπί Button1. Το όνομα του Button1 δεν το αλλάξαμε γιατί εδώ είναι μόνο ένα.



Στο περιβάλλον σύνταξης κώδικα, μπείτε στο κατάλληλο Sub για το γεγονός Button1_Click όπως δείχνει η παρακάτω εικόνα.



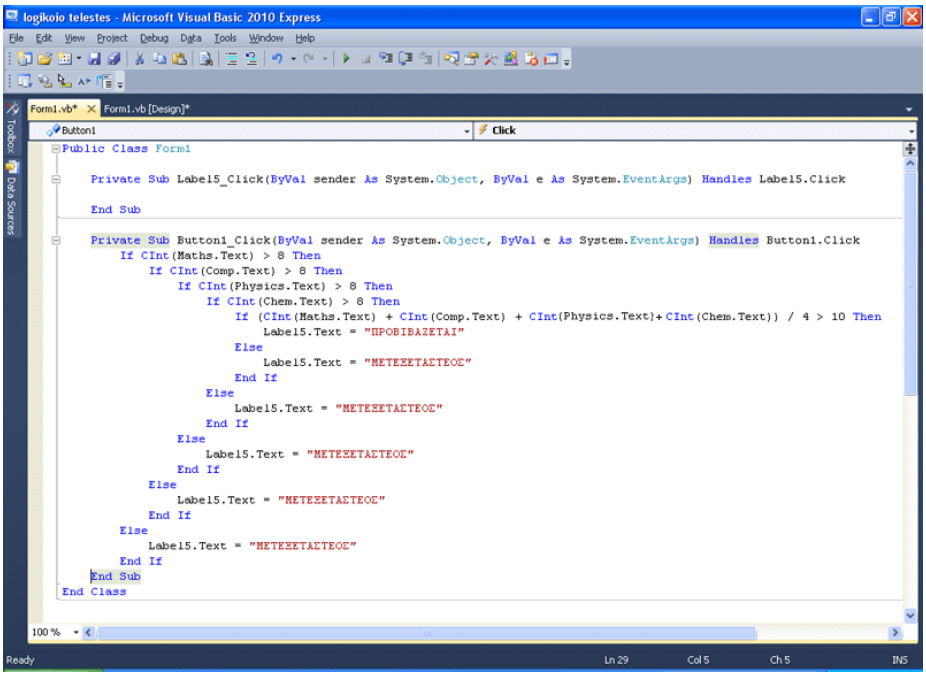
Αν δε χρησιμοποιούσαμε λογικούς τελεστές, το πρώτο βήμα που έπρεπε να κάνουμε, θα ήταν να δούμε την καρδιά του ζητούμενου. Τι θέλουμε να κάνει το κουμπί; Αν όλοι οι βαθμοί είναι πάνω από 8 κι αν ο μέσος όρος τους, δηλαδή το άθροισμά τους δια το 4, είναι μεγαλύτερο του 10, τότε θα γράφει "ΠΡΟΒΙΒΑΖΕΤΑΙ", αλλιώς θα γράφει μετεξεταστέος.

Δείτε τον παρακάτω κώδικα δίχως λογικούς τελεστές, αλλά μην επιχειρήσετε να τον γράψετε ακόμα:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
 If CInt(Maths.Text) > 8 Then
   If CInt(Comp.Text) > 8 Then
     If CInt(Physics.Text) > 8 Then
       If CInt(Chem.Text) > 8 Then
        If (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Maths.Text) + CInt(Chem.Text)) / 4 >
10 Then
            Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
         Else
            Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
          End If
        Else
         Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
         End If
      Else
       Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
       End If
   Else
    Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
    End If
Else
Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
End If
End Sub
```

Λειτουργεί σωστά! Είδατε όμως πόσες φορές έπρεπε να γράψουμε:

Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"; Είναι λίγο βαρετό έτσι δεν είναι; Φανταστείτε τώρα πως όλες αυτές οι If, που είναι ενθυλακωμένες η μία μέσα στην άλλη, εκτελούνται στο βέλτιστο σενάριο άλλων If. Διότι θα πρέπει, να μη μπορεί να βάλει κείμενο ο χρήστης, εκεί που ζητάμε αριθμούς σωστά; Θα πρέπει επίσης να ελέγχεται αν ο κάθε αριθμός είναι μεγαλύτερος από 20. Είναι περιορισμός του ζητούμενου. Δεν υπάρχει μεγαλύτερος βαθμός του 20. Επίσης θα πρέπει να ελέγχεται αν ο κάθε αριθμός θα είναι μικρότερος του 1. Δε μπαίνει βαθμός μικρότερος του 1, άρα δεύτερος περιορισμός. Φαντάζεστε το μέγεθος και το χάος που πρόκειται να προκύψει, αν χρησιμοποιήσουμε την If όπως μέχρι τώρα την ξέρουμε;



Ωστόσο έχουμε καλά νέα. Οι λογικοί τελεστές απλουστεύουν κατά πολύ τον κώδικα και σαφώς τον μικραίνουν. Δείτε τον παρακάτω κώδικα που είναι ισοδύναμος με τον προηγούμενο.

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
If CInt(Maths.Text) > 8 And CInt(Comp.Text) > 8 And CInt(Physics.Text) > 8 And
CInt(Chem.Text) > 8 And (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Phys.Text) +
CInt(Chem.Text)) / 4 > 10 Then
  Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
Else
   Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
End If
End Sub
```

Τι λέει ο παραπάνω κώδικας δηλαδή;
ΑΝ ο βαθμος Μαθηματικών που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Πληροφορικής που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Φυσικής που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Χημείας που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
(το άθροισμα των βαθμών)/4 (ο μέσος όρος δηλαδή) είναι μεγαλύτερος του 10
ΤΟΤΕ
Εμφάνισε πως προβιβάζεται
ΑΛΛΙΩΣ
Εμφάνισε πως είναι μετεξεταστέος.
ΤΕΛΟΣ ΑΝ

Ας θυμηθούμε δύο πράγματα.
Η εντολή If εκτελεί οτιδήποτε είναι έπειτα του "ΤΟΤΕ" αν η συνθήκη της είναι αληθής σωστά; Αν είναι ψευδής εκτελείται το κομμάτι κώδικα μετά το "ΑΛΛΙΩΣ". Εδώ ποια συνθήκη μελετάμε αν είναι αληθής;

Τη σύνθετη:
ο βαθμός Μαθηματικών που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Πληροφορικής που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Φυσικής που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
ο βαθμός Χημείας που βάζουμε είναι μεγαλύτερος του 8
ΚΑΙ
(το άθροισμα των βαθμών)/4 > 10

Όλο αυτό είναι ΜΙΑ συνθήκη, η συνθήκη της If, που αποτελείται από συνενωμένες υποσυνθήκες με το λογικό τελεστή AND. Συνεπώς πότε είναι αληθής; ΟΤΑΝ ΟΛΑ ΙΣΧΥΟΥΝ! Διότι ο τελεστής "ΚΑΙ", ο AND δηλαδή, τι λέμε ότι κάνει; Λέμε πως δίνει απάντηση True, μόνο αν ΚΑΙ οι δύο συνθήκες είναι True. Εδώ έχουμε τέσσερις AND που συνδέουν 5 συνθήκες! Οπότε ΚΑΙ οι 5 πρέπει να είναι αληθείς, για να είναι ολόκληρη η έκφραση αληθής.

Με πιο απλά λόγια: Αν ένα πρόγραμμα πρέπει να κάνει την πράξη Χ, όταν και μόνο όταν ισχύουν τα Α,Β και C, κάνουμε το παρακάτω:

```
IF (A) AND (B) AND (C) THEN
κάνε την πράξη Χ
ELSE
κάνε κάτι άλλο...
End If.
```

Κοιτάξτε γιατί:
(A) AND (B) AND (C) =
(A) AND [(B) AND (C)]

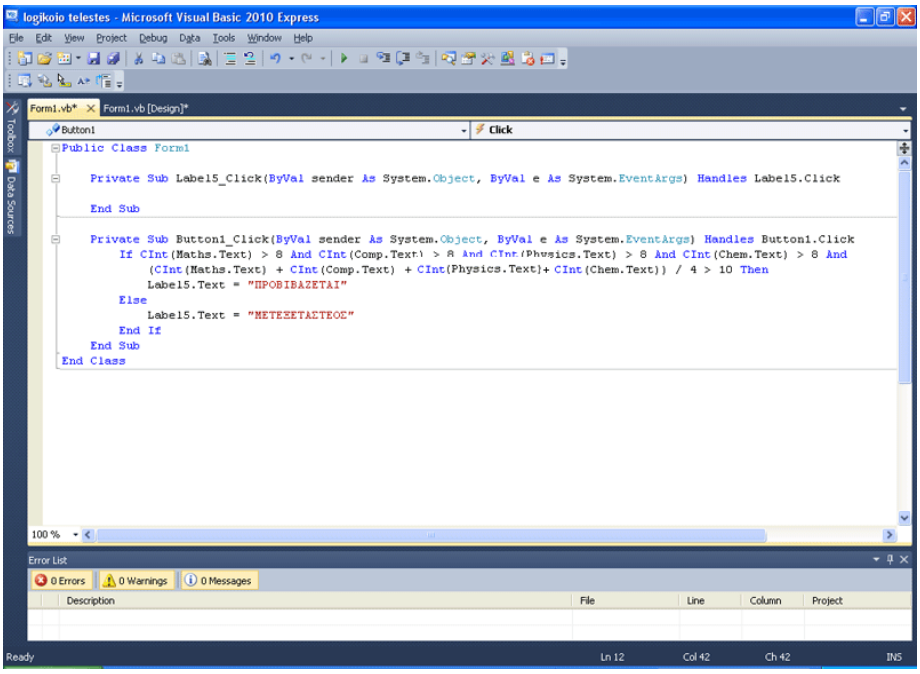
Έστω (D) = [(B) AND (C)]

Είναι (A) AND (B) AND (C) =
(A) AND [(B) AND (C)] =
(A) AND (D)

Πρέπει λοιπόν να ισχύει το Α ΚΑΙ το D
Το D πότε ισχύει; Όταν ισχύει το [(B) AND (C)]

Οπότε το D ισχύει αν ισχύουν τα (Β) και (C).

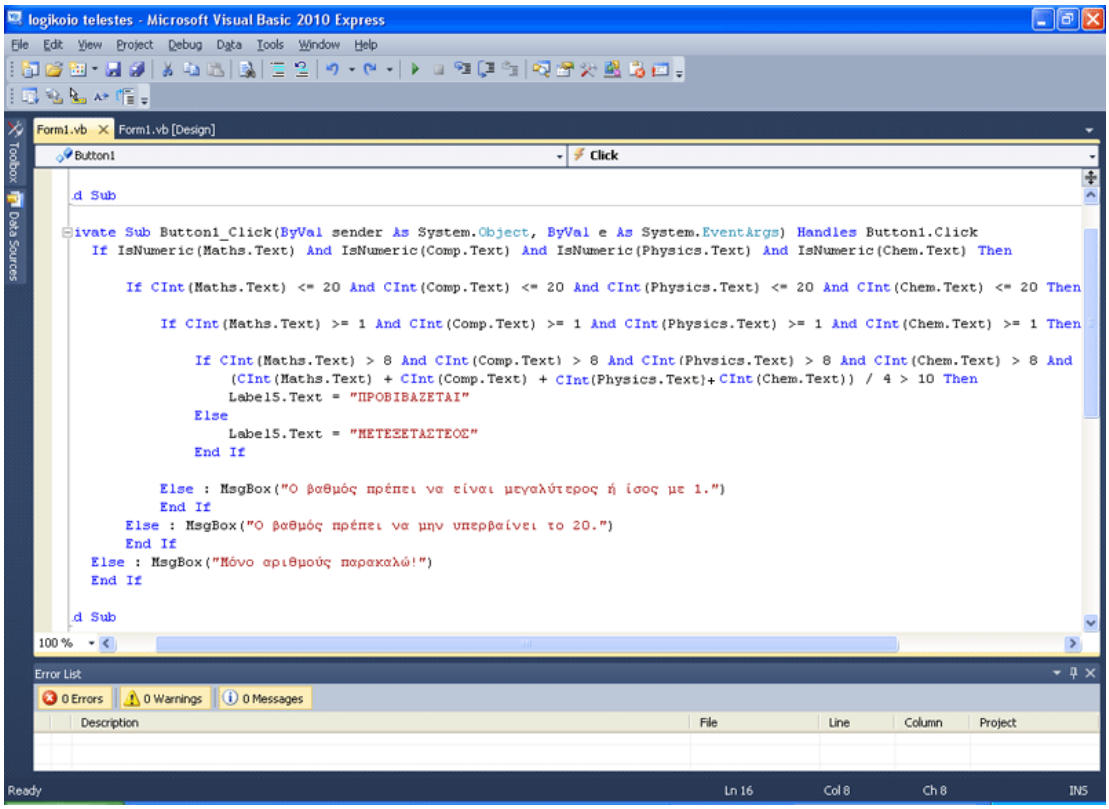
Οπότε στη [(A) AND (B) AND (C)] πρέπει να ισχύουν τα πάντα για να έχει τιμή True



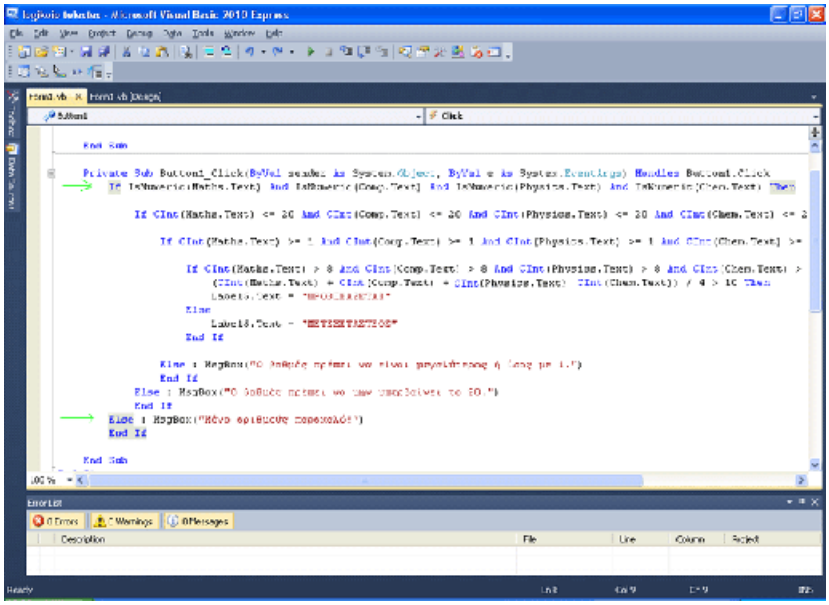
Θα σας παραθέσουμε τώρα ΟΛΟ τον τελικό κώδικα, με το πρόγραμμα σταθερό και αξιόπιστο. Μην ανησυχείτε, θα τον αναλύσουμε βήμα βήμα.

```
 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
 If IsNumeric(Maths.Text) And IsNumeric(Comp.Text) And IsNumeric(Physics.Text)
 And IsNumeric(Chem.Text) Then
  If CInt(Maths.Text) <= 20 And CInt(Comp.Text) <= 20 And CInt(Physics.Text) <= 20
  And CInt(Chem.Text) <= 20 Then
   If CInt(Maths.Text) >= 1 And CInt(Comp.Text) >= 1 And CInt(Physics.Text) >= 1
   And CInt(Chem.Text) >= 1 Then

    If CInt(Maths.Text) > 8 And CInt(Comp.Text) > 8
    And CInt(Physics.Text) > 8 And CInt(Chem.Text) > 8
    And  (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Physics.Text) + CInt(Chem.Text)) / 4
> 10 Then
     Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
     Else
     Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
     End If
    Else : MsgBox("Ο βαθμός πρέπει να είναι μεγαλύτερος ή ίσος με 1.")
    End If
   Else : MsgBox("Ο βαθμός πρέπει να μην υπερβαίνει το 20.")
   End If
 Else : MsgBox("Μόνο αριθμούς παρακαλώ!")
 End If
 End Sub
```

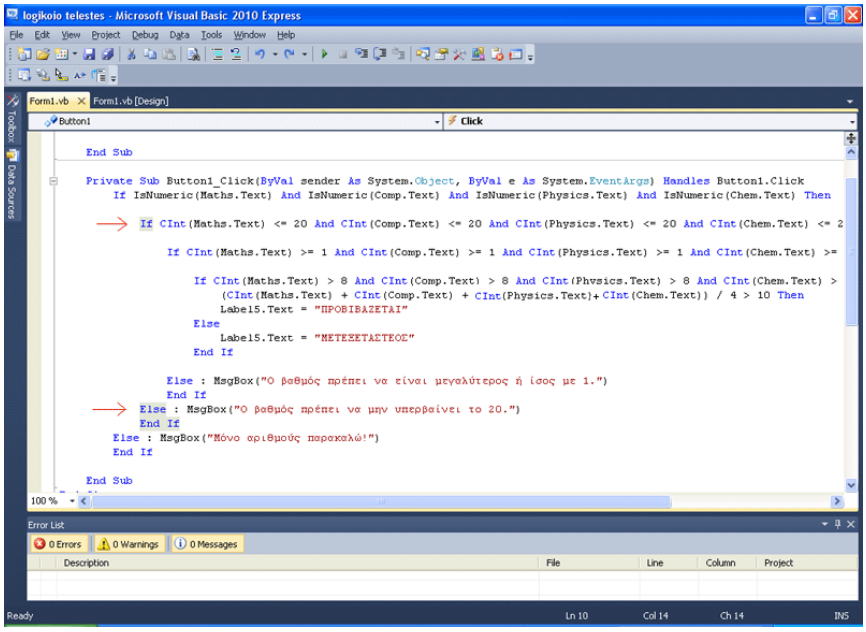
Προσέξατε κάτι; Πριν χρησιμοποιήσουμε τους λογικούς τελεστές, χρησιμοποιήσαμε πέντε If για να κάνουμε ΜΟΝΟ τη βασική λειτουργία. Ο παραπάνω κώδικας, όχι μόνο έχει 4 If μονάχα, αλλά και κάνει τους απαραίτητους ελέγχους ώστε το πρόγραμμα να είναι αξιόπιστο! Θα σας εξηγήσουμε τώρα βήμα βήμα τι κάνει ο κώδικας.



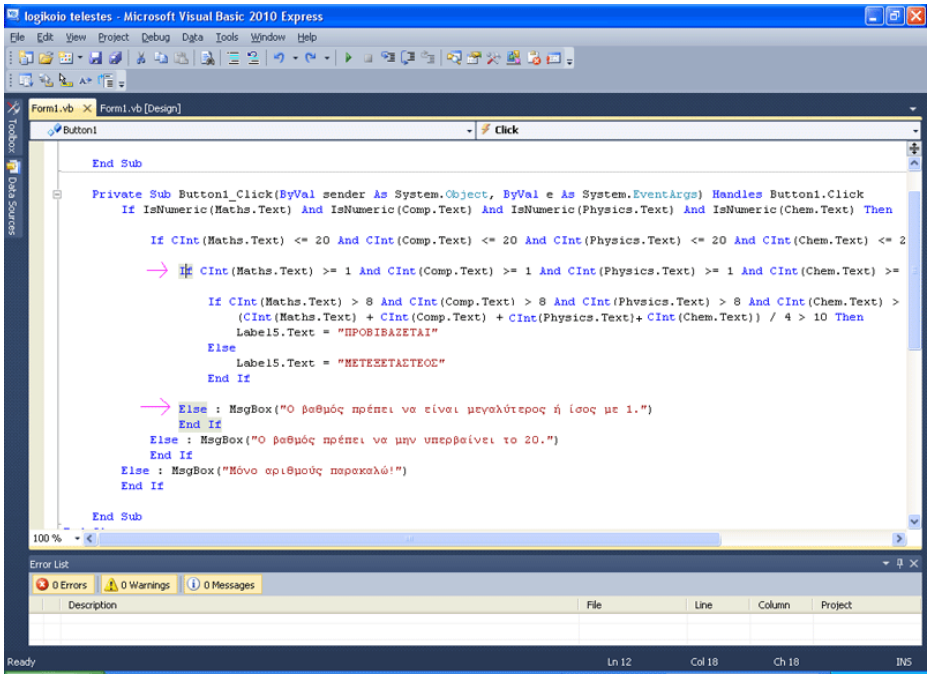
Δείτε την παρακάτω εικόνα. Τα πράσινα βελάκια σας βοηθούν να παρατηρήστε την πρώτη δομή If. Η Visual Basic έχει γκριζάρει κι εκείνη το IF, το THEN και το ELSE καθώς και το End IF. Η πρώτη If λοιπόν ελέγχει αν ΟΛΑ τα Textbox μας, περιέχουν αριθμό. Αν ναι, προχωράει στην επόμενη If. Αν όχι εμφανίζει μήνυμα που ενημερώνει το χρήστη για το σφάλμα του.



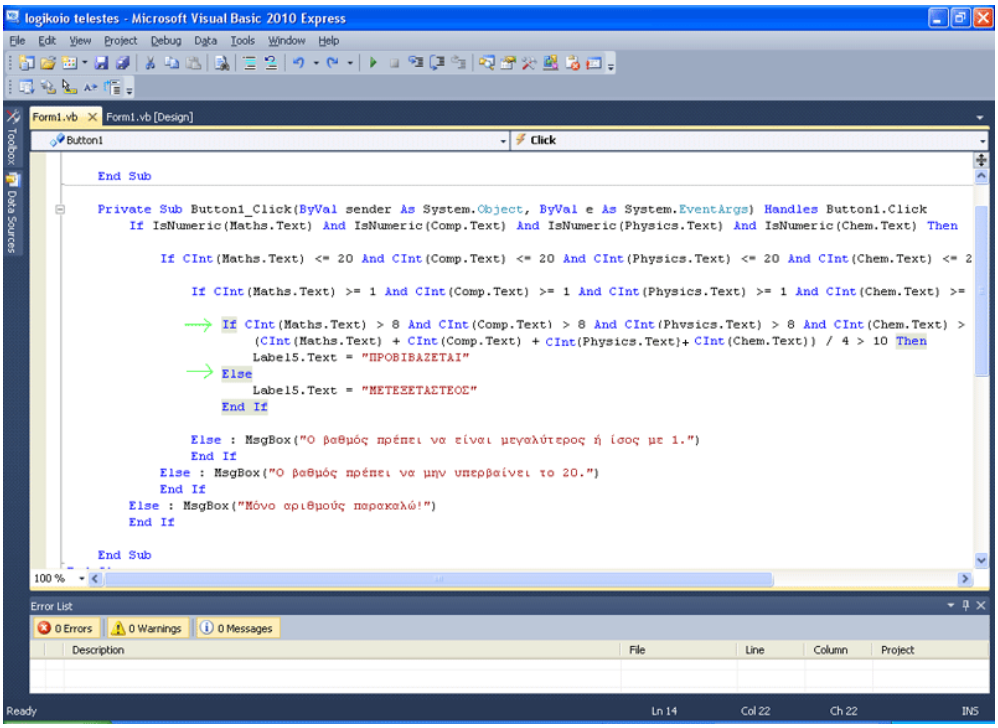
Η δεύτερη If με τα κόκκινα βελάκια, όπως δείχνει η παρακάτω εικόνα, μελετάει αν οι βαθμοί που βάλαμε είναι μικρότεροι ή ίσοι του 20. Αν ναι προχωράει στην επόμενη If. Αν όχι εμφανίζει μήνυμα που ενημερώνει το χρήστη για το σφάλμα του.



Η τρίτη If που δείχνει η παρακάτω εικόνα, μελετάει αν όλοι οι βαθμοί που βάλαμε είναι μεγαλύτεροι της μονάδας. Αν ναι προχωράει στην επόμενη If. Αν όχι εμφανίζει μήνυμα που ενημερώνει το χρήστη για το σφάλμα του.



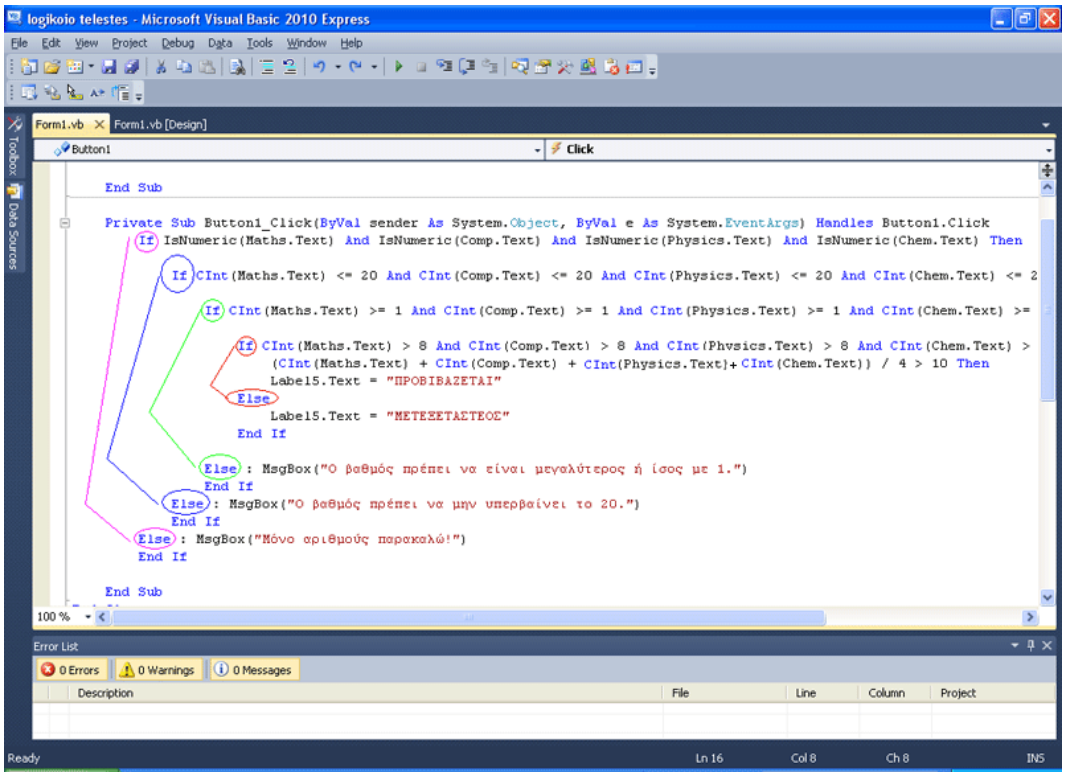
Η τελευταία If που εκτελείται, είναι ουσιαστικά το ζητούμενο, το οποίο ήδη συντάξαμε προηγουμένως. Μας δίνει το αποτέλεσμα αν κάποιος προβιβάστηκε ή όχι.



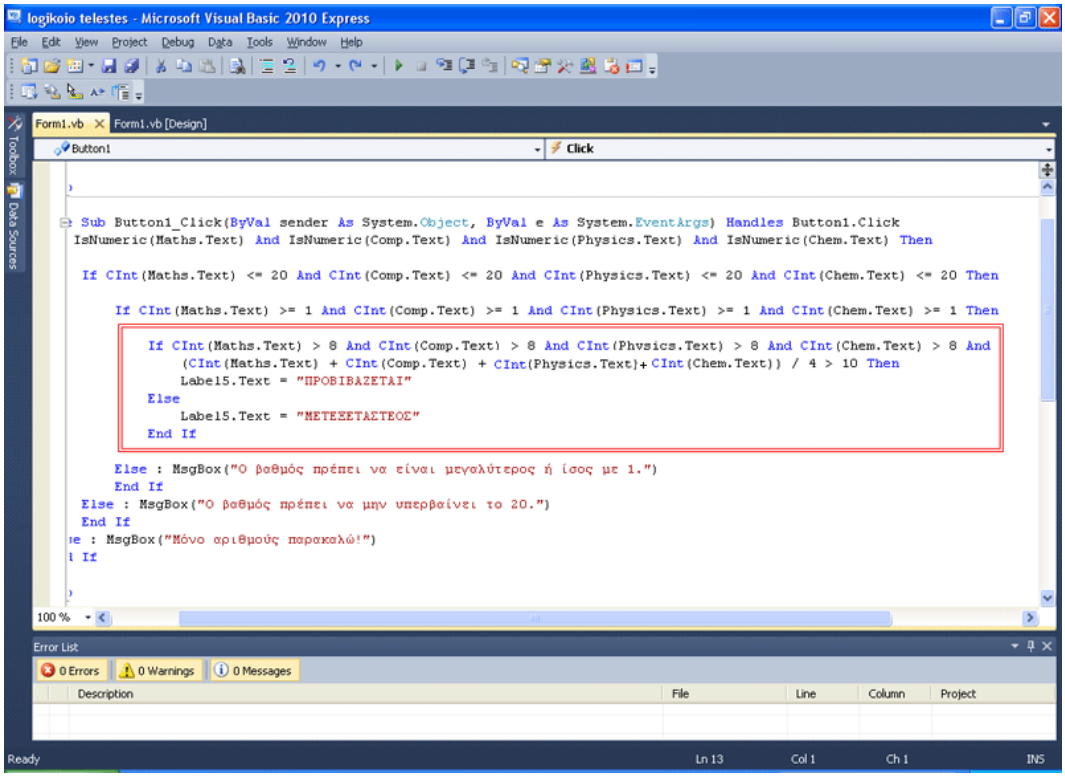
Η παρακάτω εικόνα δείχνει καλύτερα τον τρόπο που ενθυλακώνονται οι If και τη λογική σύνταξή τους.

```
Public Class Form1

    Private Sub Label5_Click(ByVal sender As System.Object, ByVal e As Syst

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As Sys
        If IsNumeric(Maths.Text) And IsNumeric(Comp.Text) And IsNumeric(Phy

            If CInt(Maths.Text) <= 20 And CInt(Comp.Text) <= 20 And CInt(Ph

                If CInt(Maths.Text) >= 1 And CInt(Comp.Text) >= 1 And CInt(

                    If CInt(Maths.Text) < 8 Or CInt(Comp.Text) < 8 Or CInt(
                        (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Maths.Te
                        Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
                    Else
                        Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"

                    End If

                Else : MsgBox("Ο βαθμός πρέπει να είναι μεγαλύτερος ή ίσος
                End If
            Else : MsgBox("Ο βαθμός πρέπει να μην υπερβαίνει το 20.")
            End If
        Else : MsgBox("Μόνο αριθμούς παρακαλώ!")
        End If

    End Sub
End Class
```

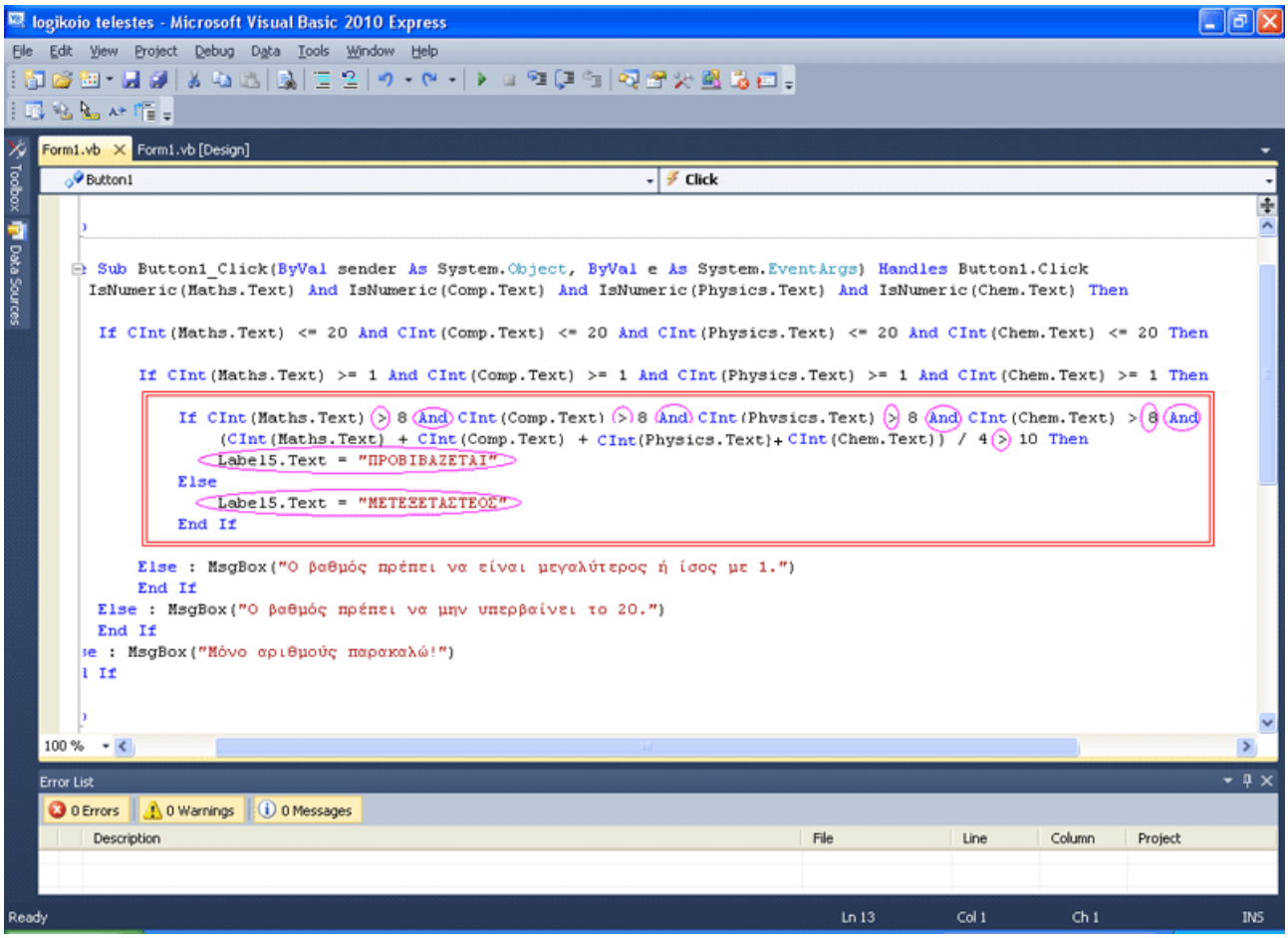
Ομοίως, δείτε το τώρα στη Visual Basic. Παρατηρούμε δηλαδή, πως όταν έχουμε πολλές If τη μία μέσα στην άλλη, εκτελούνται από έξω προς τα μέσα.



Θα θέλαμε να σας δείξουμε τώρα τη χρησιμότητα της OR. Όμως όχι μόνο θα σας δείξουμε τη χρησιμότητά της, αλλά και μια περίεργη σχέση που έχει με την AND.



Για να γίνει απλό το θέμα, θα ασχοληθούμε μόνο με τον πυρήνα του προγράμματος, τον οποίο έχουμε πλαισιώσει σε κόκκινο τετράγωνο στην παρακάτω εικόνα.



Παρατηρήστε τις ανισότητες, το λογικό τελεστή AND, αλλά και τι βρίσκεται μετά το Then και μετά το Else.

```
If CInt(Maths.Text) > 8 And CInt(Comp.Text) > 8 And CInt(Physics.Text) > 8 And CInt(Chem.Text) > 8 And
    (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Physics.Text)+ CInt(Chem.Text)) / 4 > 10 Then
   Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
Else
   Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
End If
```

Αν όπου > βάλουμε το <=
όπου < βάλουμε το >=
όπου = βάλουμε το <>
και γενικά αν βάλουμε τον άκρως αντίθετο τελεστή σύγκρισης,
αν βάλουμε OR όπου έχουμε AND και αντιμεταθέσουμε τις εργασίες μετά το Then, με τις εργασίες μετά το Else, τότε μας βγαίνει λογικά ισοδύναμη και όμοια συνθήκη If! Κοινώς το ίδιο πρόγραμμα μπορούσε να υλοποιηθεί με διαφορετική προσέγγιση; Ακριβώς!

```
If CInt(Maths.Text) <= 8 OR CInt(Comp.Text) <= 8
      OR CInt(Physics.Text) <= 8 OR CInt(Chem.Text) <= 8
      OR  (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Physics.Text) + CInt(Chem.Text)) / 4 <= 10 Then
       Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
       Else
        Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
       End If
```

Με χρήση των λογικών τελεστών AND, απαιτούμε μέσω των AND να ισχύουν όλες οι συνθήκες.

```
If CInt(Maths.Text) > 8 And CInt(Comp.Text) > 8 And CInt(Physics.Text) > 8 And CInt(Chem.Text) > 8 And
    (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Physics.Text)+ CInt(Chem.Text)) / 4 > 10 Then
   Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
Else
   Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
End If
```

Η OR, αν τουλάχιστο μία συνθήκη είναι αληθής δίνει τιμή True.
Δηλαδή η συνθήκη C = [(A) OR (B)] παίρνει τιμή True,
είτε άν το Α είναι True,
είτε αν το Β είναι True,
είτε άν το Α είναι True και το Β είναι True!

Οπότε, τι λέμε; Αν έστω κι ένα πράγμα, που δεν θέλω, ισχύει. Τότε κάνε κάτι γι'αυτό (μένει μετεξεταστέος στο παράδειγμα) Αλλιώς προχώρα.

```
If CInt(Maths.Text) <= 8 OR CInt(Comp.Text) <= 8
      OR CInt(Physics.Text) <= 8 OR CInt(Chem.Text) <= 8
      OR  (CInt(Maths.Text) + CInt(Comp.Text) + CInt(Physics.Text) + CInt(Chem.Text)) / 4 <= 10 Then
       Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
       Else
        Label5.Text = "ΠΡΟΒΙΒΑΖΕΤΑΙ"
       End If
```

Κοινώς οι παραπάνω δύο συνθήκες,
η μία με πολλές AND
και η άλλη με πολλές OR
κάνουν ΑΚΡΙΒΩΣ ΤΟ ΙΔΙΟ ΠΡΑΓΜΑ!

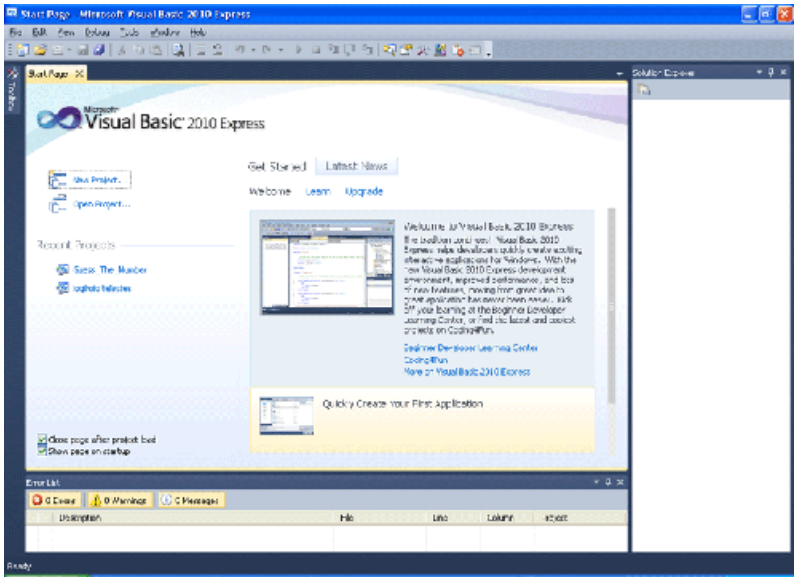
Το πώς θα υλοποιηθεί το πρόγραμμα, είναι ξεκάθαρα δική σας επιλογή λοιπόν. Και οι δύο προσεγγίσεις είναι σωστές. Ωστόσο πολλά μέρη ενός προβλήματος υλοποιούνται πολλές φορές εκ φύσεως, μόνο με OR ή μόνο με AND.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

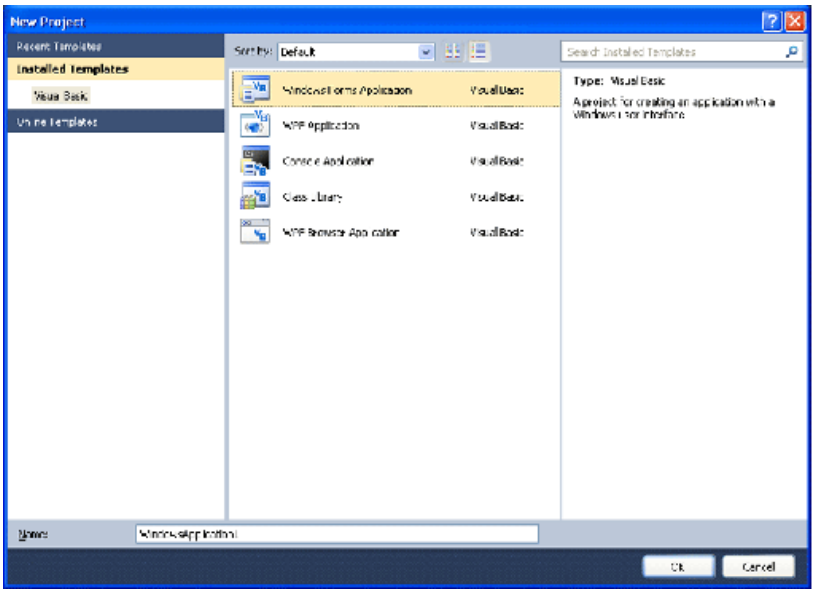
Το κεφάλαιο αυτό, γνωρίζοντας ο χρήστης την IF, έχει στόχο να επεξηγήσει πρακτικά τους λογικούς τελεστές, τη λογική πίσω από αυτούς και τη χρησιμότητά τους. Επίσης εφαρμόζεται το θεώρημα De Morgan έτσι ώστε να δειχθεί στο μαθητευόμενο η ιδιαίτερη σχέση που έχουν ο AND με τον OR τελεστή.

ΚΕΦΑΛΑΙΟ 11
MessageBOX & InputBox:
ΑΣ ΦΤΙΑΞΟΥΜΕ ΕΝΑ ΑΠΛΟ ΠΑΙΧΝΙΔΙ!

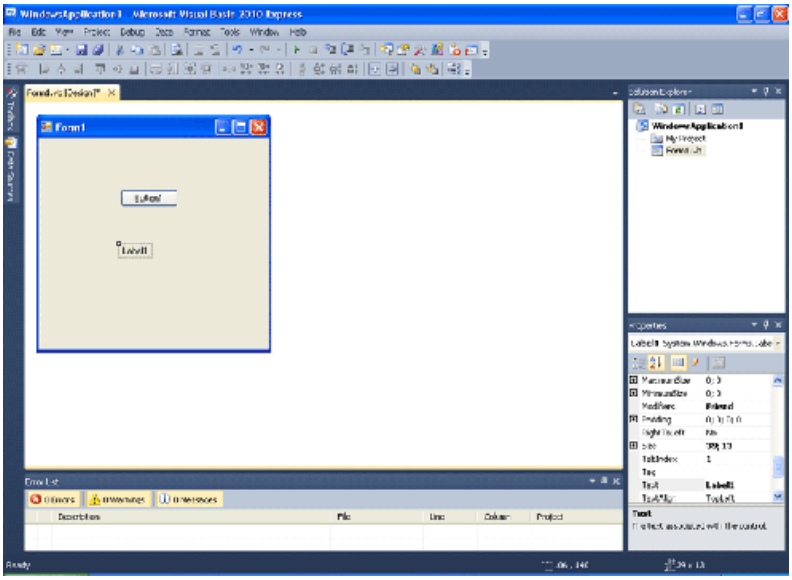
Πριν φτιάξουμε το παιχνίδι, θα σας μιλήσουμε για την εντολή InputBox().
Επιλέγουμε λοιπόν ξανά New Project...



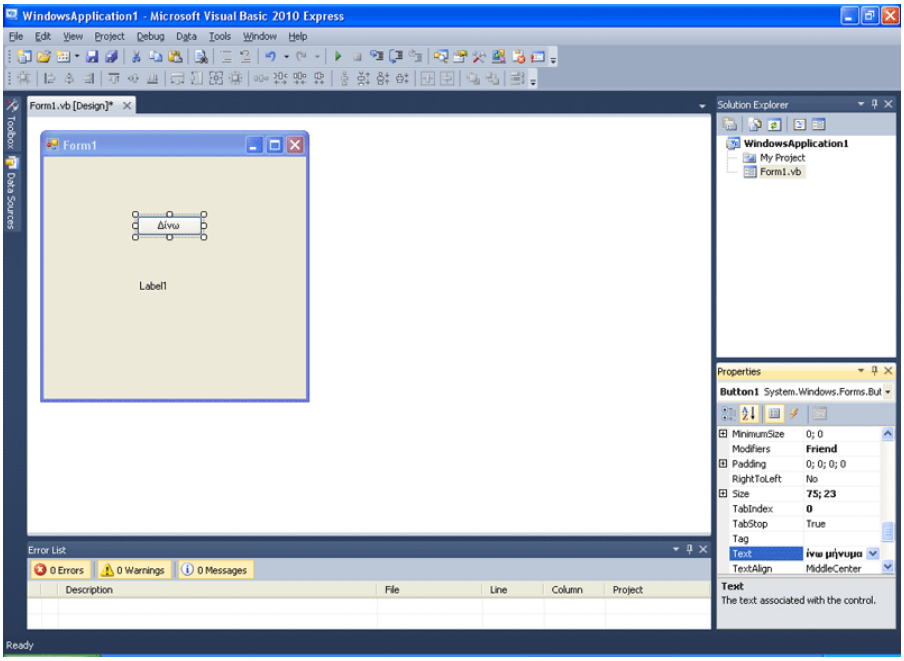
...Windows Forms Application...



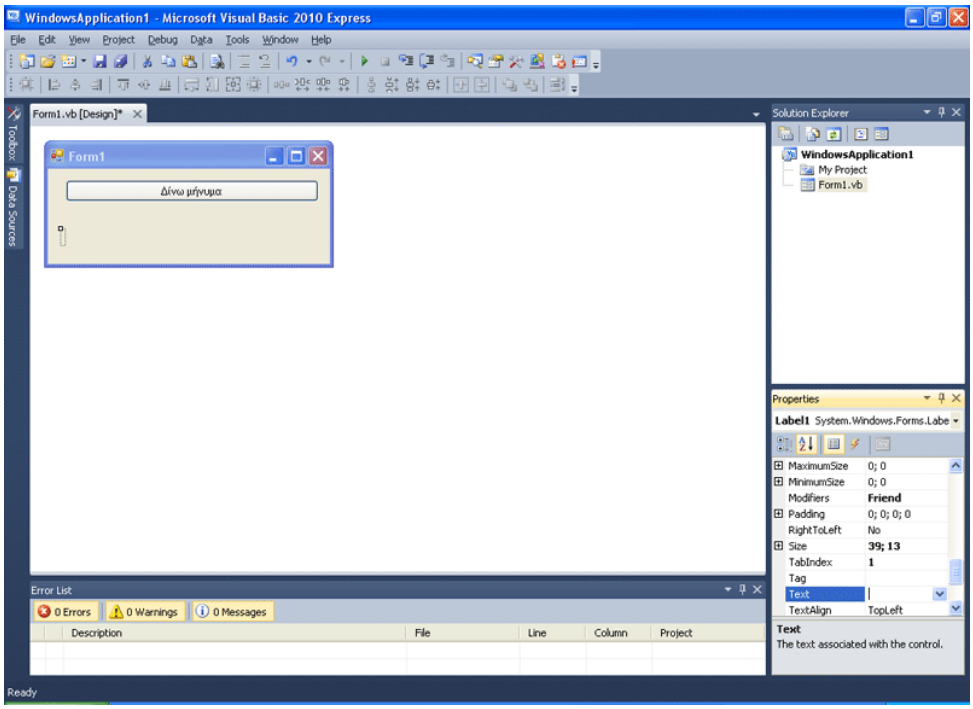
..και βάζουμε ένα Button και μία Label στη φόρμα μας.



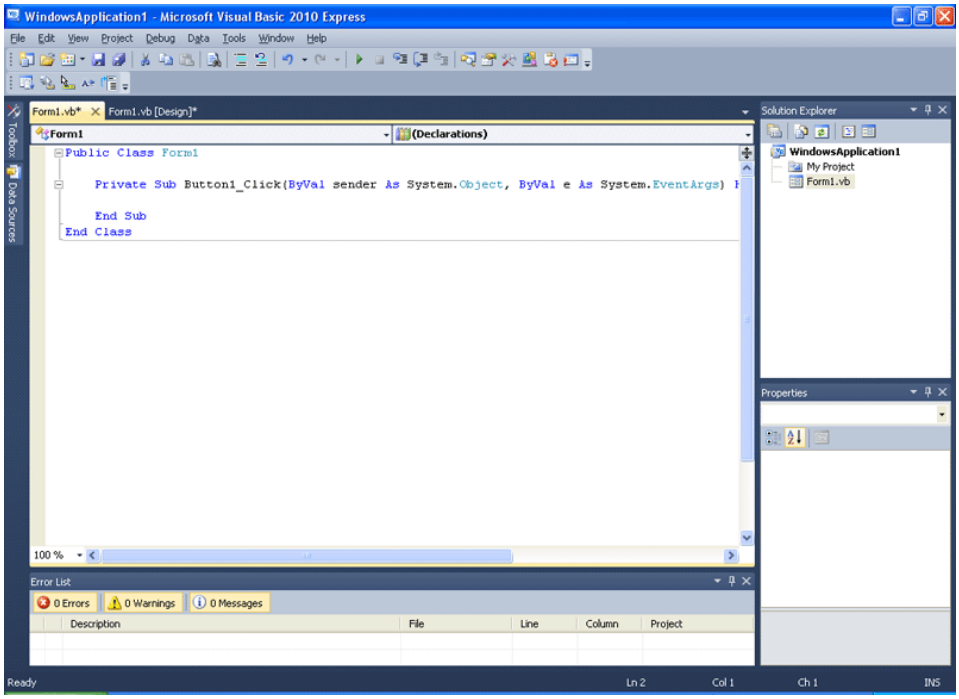
Το κείμενο του Button το κάνουμε "Δίνω μήνυμα"...



...ενώ το κείμενο της Label το σβήνουμε και το αφήνουμε κενό!



Πάμε να γράψουμε λίγο κώδικα τώρα!

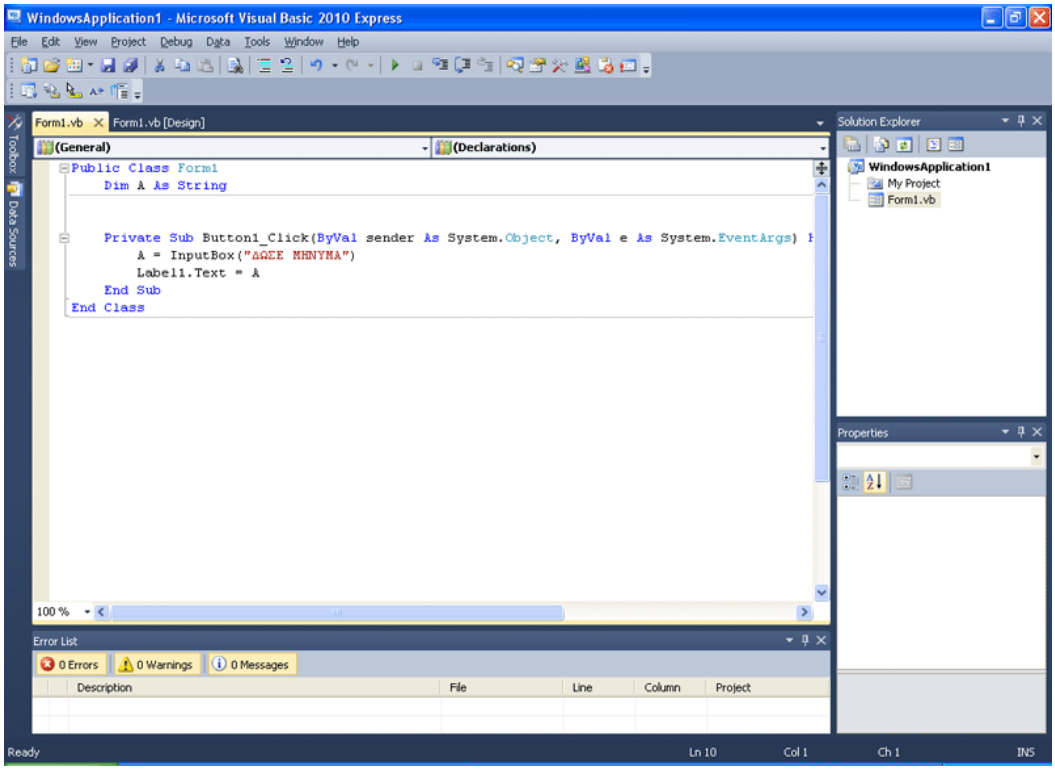


Πηγαίνουμε στην Public Class Form1 και γράφουμε την εντολή Dim A As String για να ορίσουμε μια μεταβλητή τύπου string, την οποία θα χρησιμοποιήσουμε στη συνέχεια.

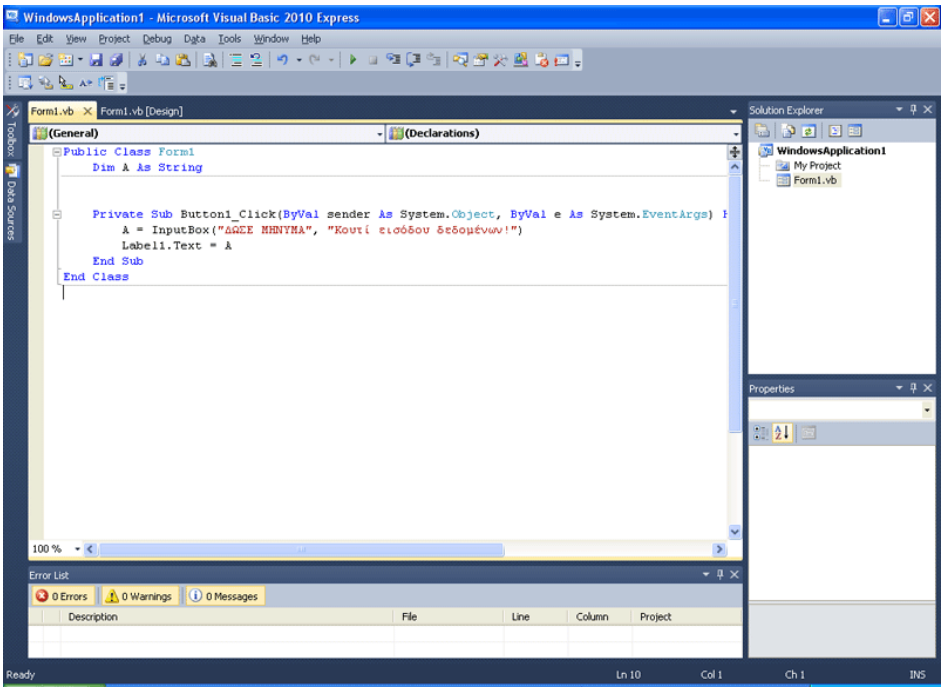
Έπειτα στο Private Sub Button1_Click γράφουμε
A = InputBox("ΔΩΣΕ ΜΗΝΥΜΑ")

Label1.Text = A

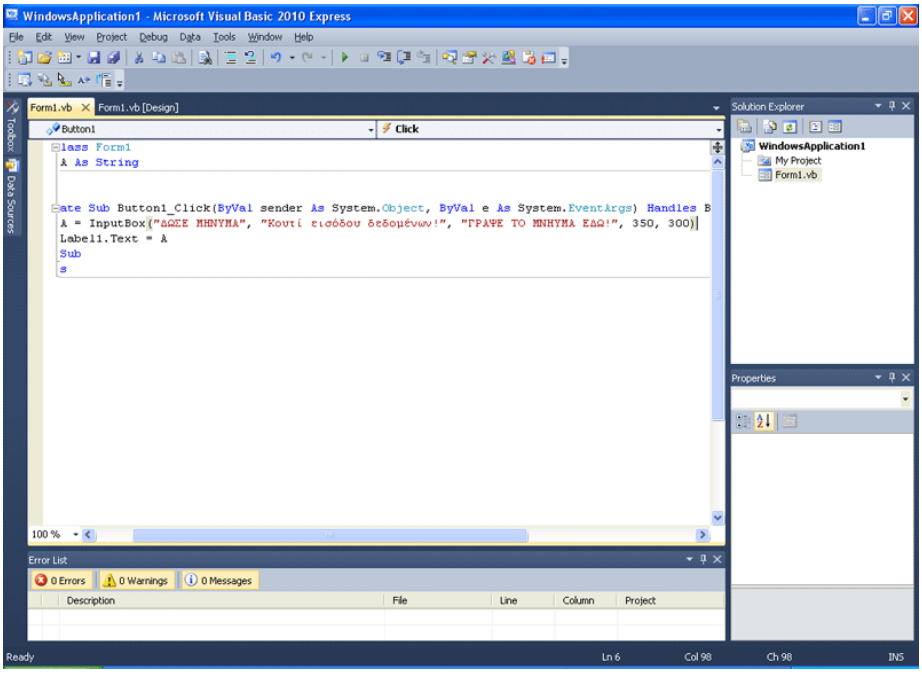
Η πρώτη εντολή θα εμφανίσει στον χρήστη ένα παράθυρο, στο οποίο θα μπορεί να πληκτρολογήσει κάτι, για να το εισάγει στο πρόγραμμα, στη μεταβλητή Α, ενώ μέσα στο παράθυρο, θα εμφανίζεται το μήνυμα "ΔΩΣΕ ΜΗΝΥΜΑ" Αυτό που θα πληκτρολογήσει ο χρήστης, αποθηκεύεται άμεσα από την εντολή, στη μεταβλητή Α, όταν πατήσει ο χρήστης "OK". Τι τύπος μεταβλητής είναι η μεταβλητή Α; String όπως την ορίσαμε.



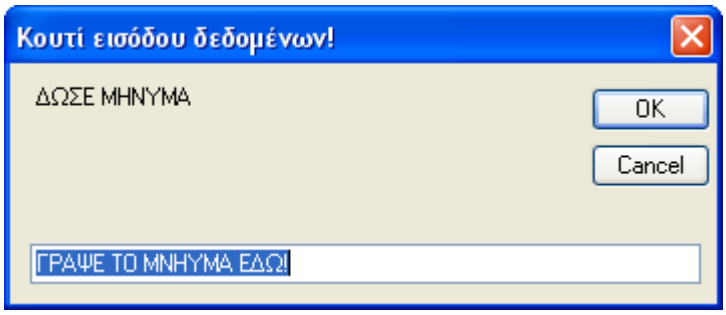
Επιπρόσθετα, μπορούμε σε αυτό το παράθυρο να ορίσουμε τον τίτλο του.
Εδώ τον ορίζουμε ως: "Κουτί εισόδου δεδομένων!"



Επίσης μπορούμε να ορίσουμε μια προκαθορισμένη τιμή... εδώ ορίσαμε την "ΓΡΑΨΕ ΤΟ ΜΗΝΥΜΑ ΕΔΩ!" Οι δύο τελευταίοι αριθμοί, χρησιμοποιούνται για να ορισθεί το που ακριβώς στην οθόνη, θα εμφανιστεί το παράθυρο αυτό. Οπότε έχετε πλέον έναν πλήρη τρόπο σύνταξης της InputBox()
Η εντολή Label1.Text = A απλώς παίρνει το περιεχόμενο της A και το τοποθετεί στο Label1.text!



Αν τρέξουμε το πρόγραμμα, θα εμφανιστεί το παρακάτω παράθυρο. Ανάλογα του τι πληκτρολογήσουμε, το πρόγραμμα αφού πατήσουμε Ok θα το τοποθετήσει στο text της Label1.

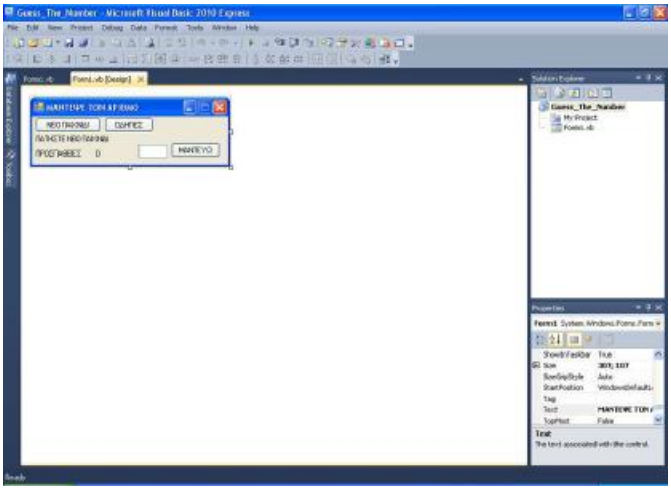


Καιρός να φτιάξουμε λοιπόν ένα παιχνίδι. Αυτό το παιχνίδι, σε πολλές γλώσσες προγραμματισμού, θεωρείται μια πολύ όμορφη μέθοδος, για την κατανόηση της IF. Πρόκειται για ένα παιχνίδι στο οποίο ένας παίκτης βάζει έναν αριθμό, όταν ο άλλος παίκτης δεν βλέπει και μετά καλείται ο δεύτερος παίκτης, να μαντέψει τον αριθμό. Ο αριθμός πρέπει να είναι από το 1 ως το 100 κι ακέραιος. Το πρόγραμμα θα στρογγυλοποιεί, πιθανό δεκαδικό. Επίσης θα εμφανίζει το πόσες προσπάθειες έκανε ο χρήστης, για να μαντέψει τον αριθμό, κι επίσης θα έχει ένα κουμπί, που θα δίνει οδηγίες στο χρήστη.

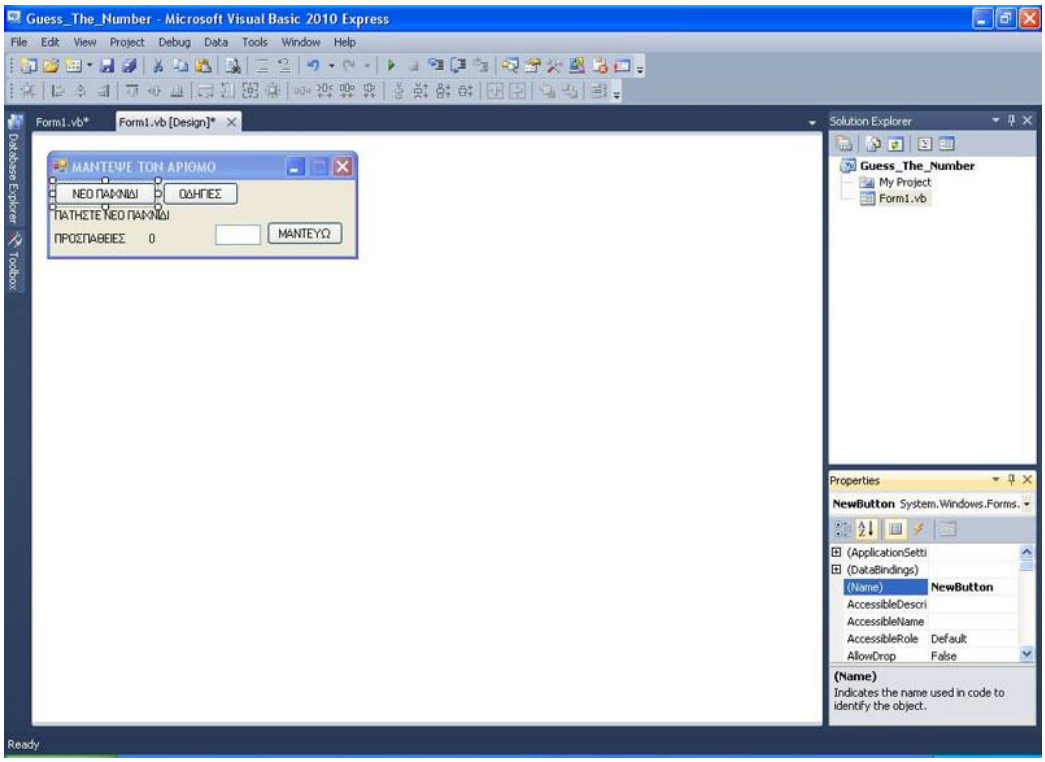
Σβήστε το Project και φτιάξτε ένα καινούργιο. Να η φόρμα μάς! Μικρή και πρακτική.

3 Button
3 Label
1 TextBox

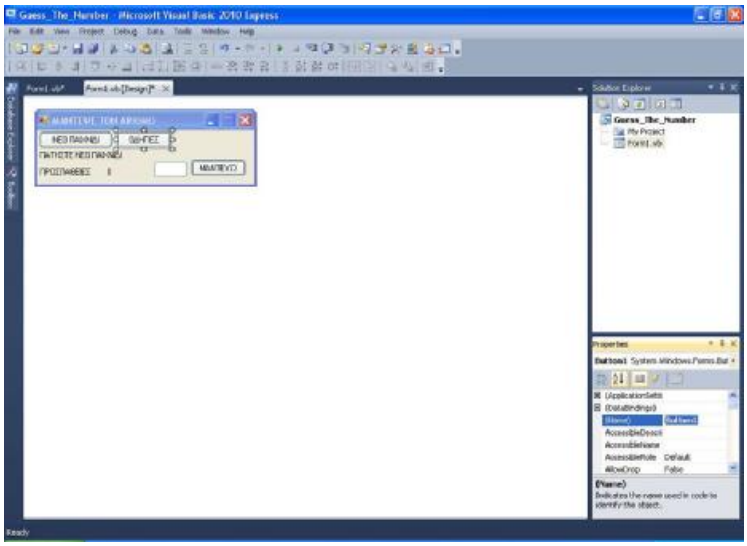
Φτιάξτε τη όπως τη βλέπετε στην εικόνα. Πλέον ξέρετε σίγουρα τον τρόπο, το έχετε κάνει ήδη πολλές φορές.



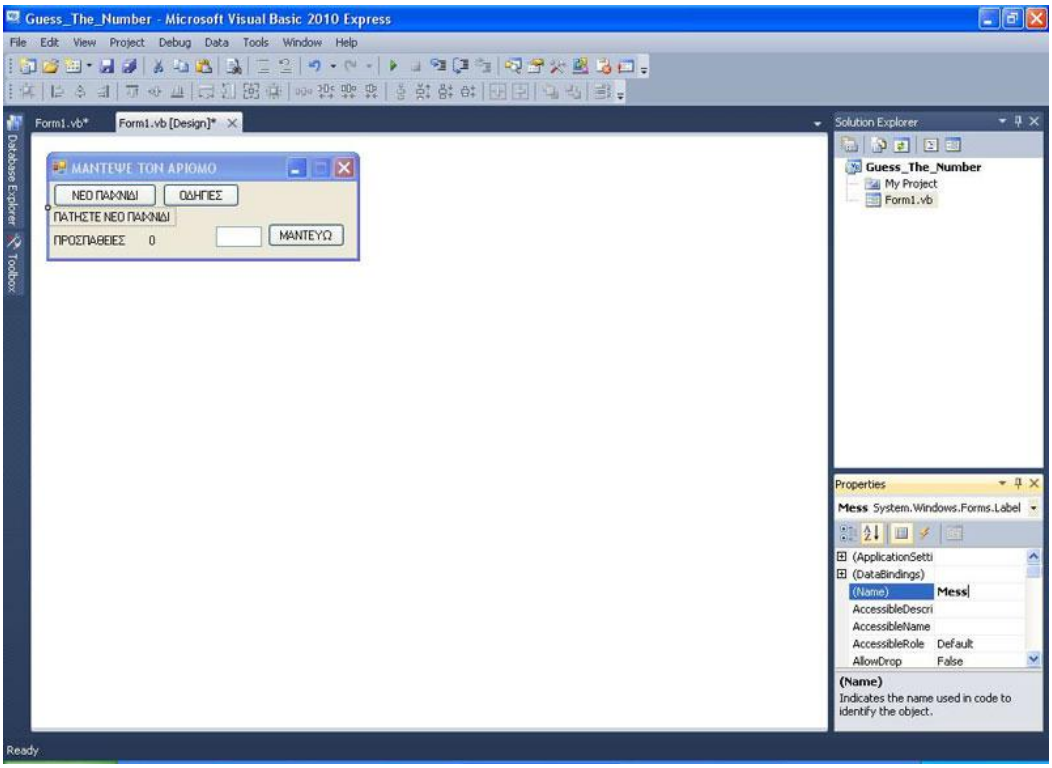
Το όνομα του κουμπιού αυτού, ονομάστε το NewButton.



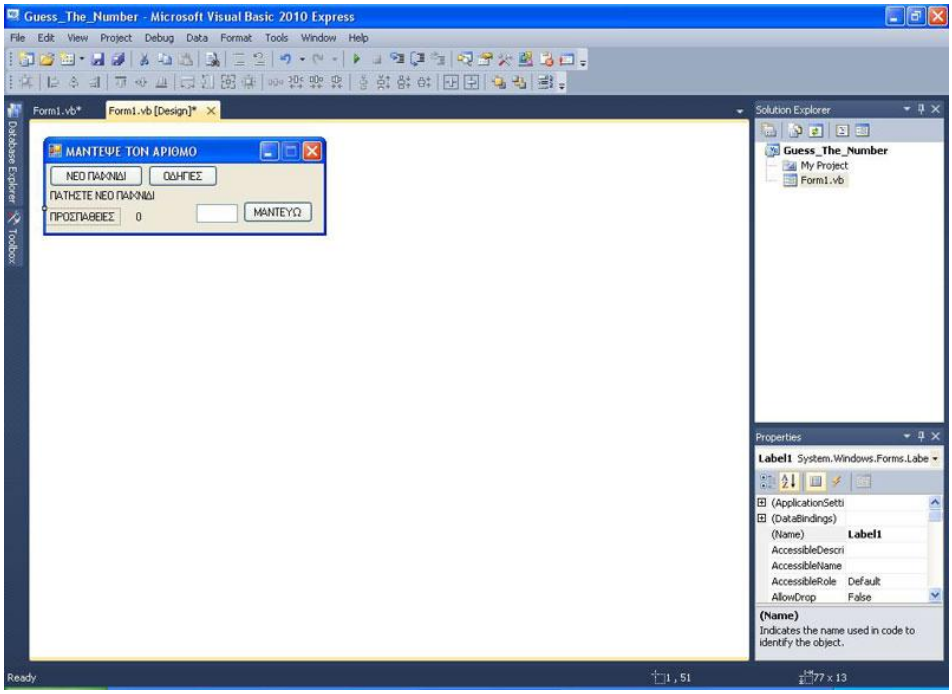
Ετούτο Button1.



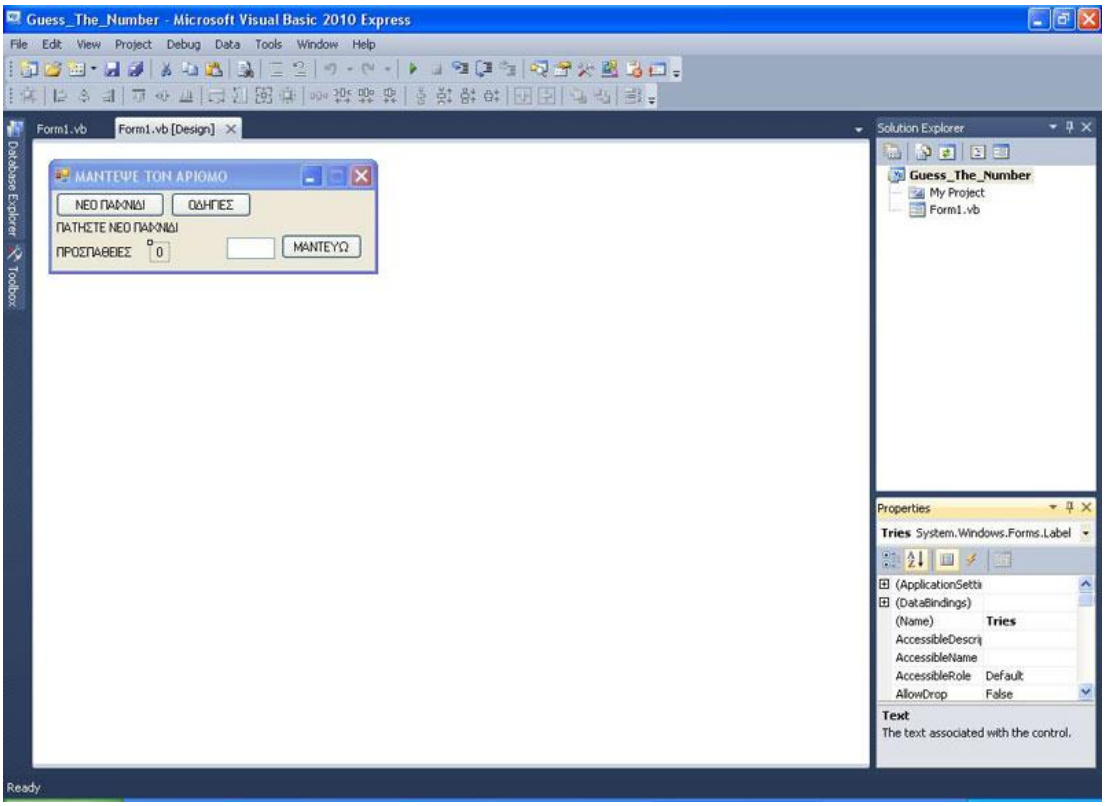
Τη Label αυτή ονομάστε τη Mess (από το Message)



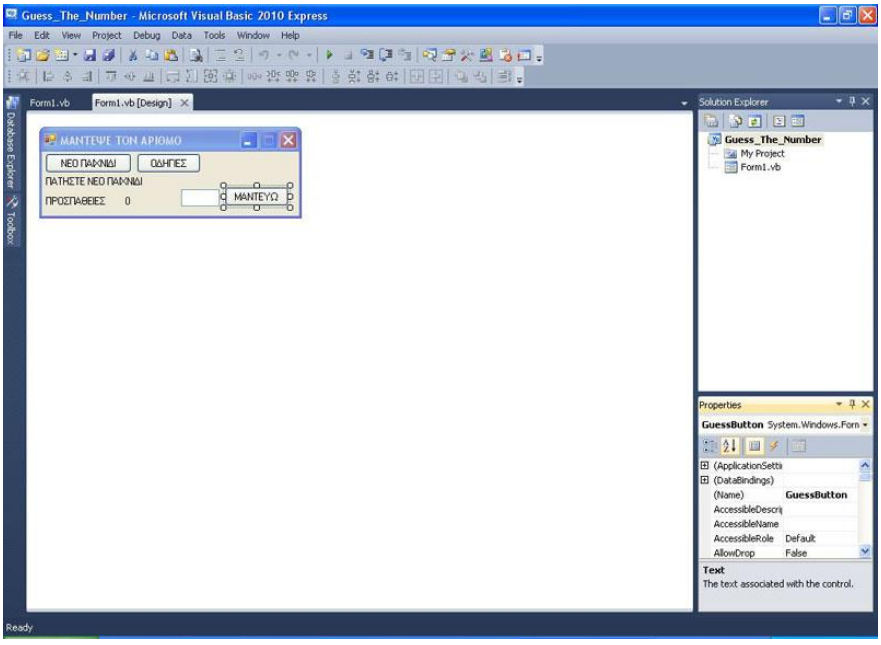
Αυτή ονομάστε τη Label1



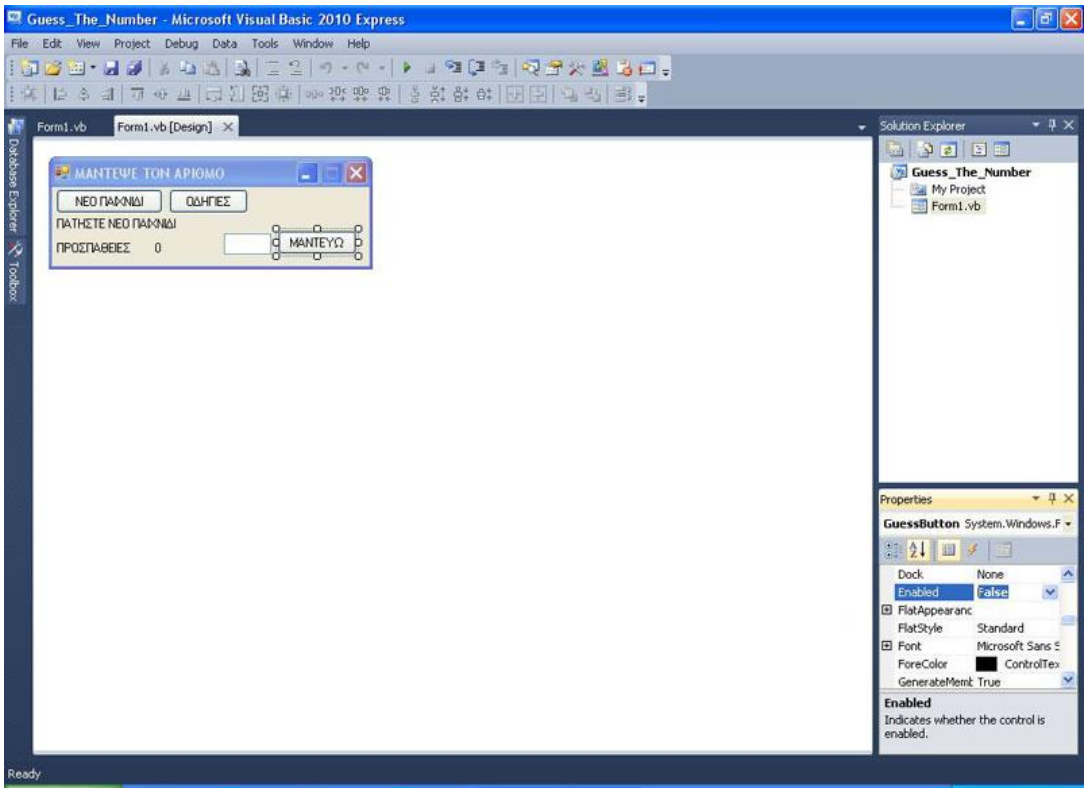
Ενώ τη Label αυτή, ονομάστε τη Tries. Αρχικό κείμενό της είναι το μηδέν.



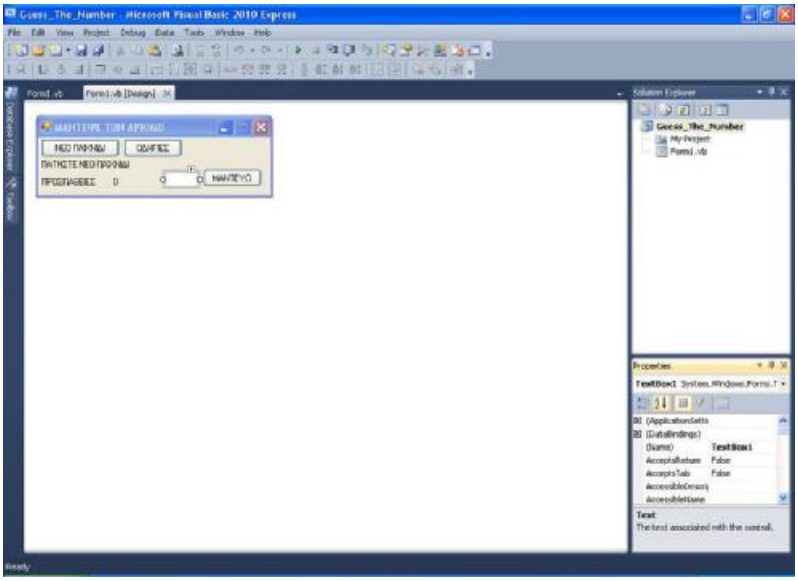
Αυτό το button το ονομάζουμε GuessButton.



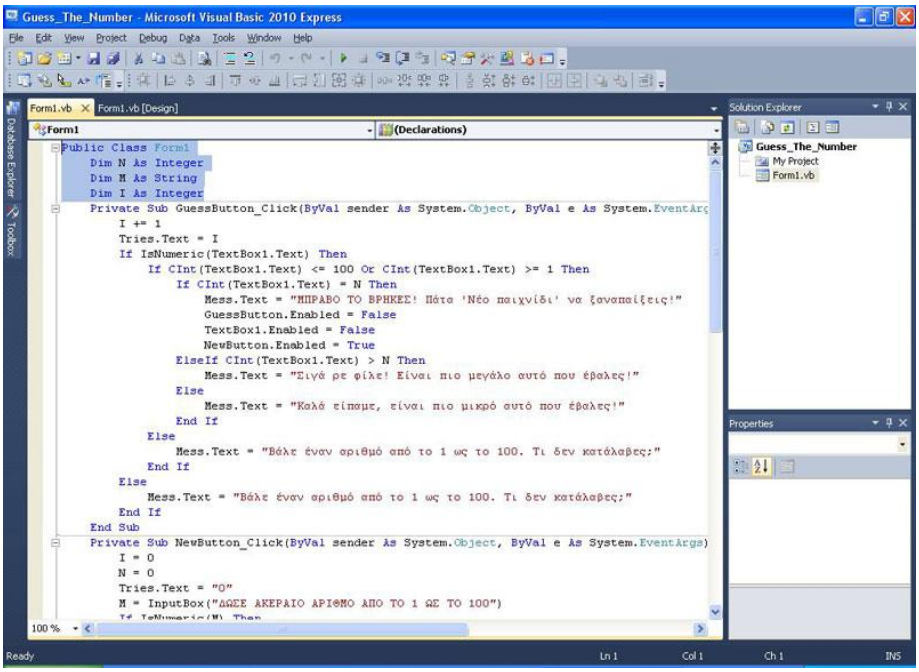
Πηγαίνετε τώρα στην ιδιότητα Enabled και επιλέξτε False. Τι κάναμε μόλις τώρα; Όταν ξεκινάει το πρόγραμμα, αυτό το κουμπί ΔΕ θα μπορεί να πατηθεί. Θα καταλάβετε αργότερα γιατί περιορίζουμε τον χρήστη σε αυτό.



Το Textbox1 το αφήνουμε ως έχει.



Πάμε στην Public Class Form_1
και πληκτρολογούμε:
Dim N As Integer
Dim M As String
Dim I As Integer
Δηλαδή φτιάχνουμε 2 μεταβλητές ακεραίων και μία μεταβλητή τύπου string.
I για τις προσπάθειες του χρήστη, N για τον αριθμό προς μάντεμα
M για το τι θα εισάγει ο χρήστης ως μυστικό αριθμό.



Θα ξεκινήσουμε λίγο ανάποδα. Έστω έχουμε βάλει έναν αριθμό και πρέπει ο χρήστης τώρα να τον μαντέψει. Πώς το κάνει αυτό; Στο TextBox1 βάζει έναν αριθμό και πατάει το κουμπί "ΜΑΝΤΕΥΩ". Αυτό είναι το GuessButton όπότε.... ο κώδικας του Private Sub GuessButton_Click ακολουθεί:

Private Sub GuessButton_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles GuessButton.Click

```
    I += 1
    Tries.Text = I
    If IsNumeric(TextBox1.Text) Then
      If CInt(TextBox1.Text) <= 100 Or CInt(TextBox1.Text) >= 1 Then
        If CInt(TextBox1.Text) = N Then
          Mess.Text = "ΜΠΡΑΒΟ ΤΟ ΒΡΗΚΕΣ! Πάτα 'Νέο παιχνίδι' να ξαναπαίξεις!"
          GuessButton.Enabled = False
          TextBox1.Enabled = False
          NewButton.Enabled = True
        ElseIf CInt(TextBox1.Text) > N Then
          Mess.Text = "Σιγά ρε φίλε! Είναι πιο μεγάλο αυτό που έβαλες!"
        Else
          Mess.Text = "Καλά είπαμε, είναι πιο μικρό αυτό που έβαλες!"
        End If
      Else
        Mess.Text = "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"
      End If
    Else
      Mess.Text = "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"
    End If
  End Sub
```

Επεξηγούμε:
I += 1 Θυμάστε τον τελεστή προσαύξησης; Εδώ τι κάνουμε;
Κάθε φορά που πατάμε το κουμπί, στο I θα προσθέτουμε 1.
Το I μετράει πόσες προσπάθειες χρειάστηκαν για να μαντέψουμε τον αριθμό.

Tries.Text = I
Το εμφανίζουμε και στη Label με όνομα Tries,
για να βλέπει κι ο χρήστης πόσες προσπάθειες χρειάστηκε.

If IsNumeric(TextBox1.Text) Then
Αν αυτό που ο χρήστης έγραψε στο TextBox1 της φόρμας μας, είναι αριθμός , τότε

If CInt(TextBox1.Text) <= 100 Or CInt(TextBox1.Text) >= 1 Then
Αν αφού μετατραπαεί σε ακέραιος, ειναι εντος του 1 και του 100, τότε

If CInt(TextBox1.Text) = N Then
Αν είναι ίσο με τον αριθμό που πρέπει να μαντέψουμε (το N), τότε

Mess.Text = "ΜΠΡΑΒΟ ΤΟ ΒΡΗΚΕΣ! Πάτα 'Νέο παιχνίδι' να ξαναπαίξεις!"
Δώσε μήνυμα στο χρήστη (πάνω στη φόρμα) πως το βρήκε.

GuessButton.Enabled = False
το παιχνίδι τελείωσε, οπότε απενεργοποιούμε το κουμπί μαντέματος
ώστε να μη μπορεί να πατηθεί.

TextBox1.Enabled = False
Και απενεργοποιούμε τη δυνατότητα να γράψει και στο TextBox1

NewButton.Enabled = True
Αλλά ενεργοποιούμε το κουμπί για να ξεκινήσει νέο παιχνίδι.

ElseIf CInt(TextBox1.Text) > N Then
Mess.Text = "Σιγά ρε φίλε! Είναι πιο μεγάλο αυτό που έβαλες!"

Αν δε το βρήκε κι έβαλε μεγαλύτερο αριθμό, του δίνουμε αντίστοιχο μήνυμα...

Else
Mess.Text = "Καλά είπαμε, είναι πιο μικρό αυτό που έβαλες!"
End If

Καλύψαμε το ενδεχόμενο ο χρήστης να μάντεψε σωστά.
Καλύψαμε το ενδεχόμενο ο χρήστης να έβαλε μεγαλύτερο αριθμό.
Τι μας έμεινε;
Αν τα δύο παραπάνω δεν αληθεύουν, λογικά βάλαμε μικρότερο αριθμό!
Με ένα Else λοιπόν, τοποθετούμε τι εδώ πέρα;
Κώδικα για το αν έβαλε ο χρήστης μικρότερο αριθμό.
Διότι αν δεν είναι μεγαλύτερο ή ίσο τι είναι; Μικρότερο!

```
Else
Mess.Text = "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"
End If
```

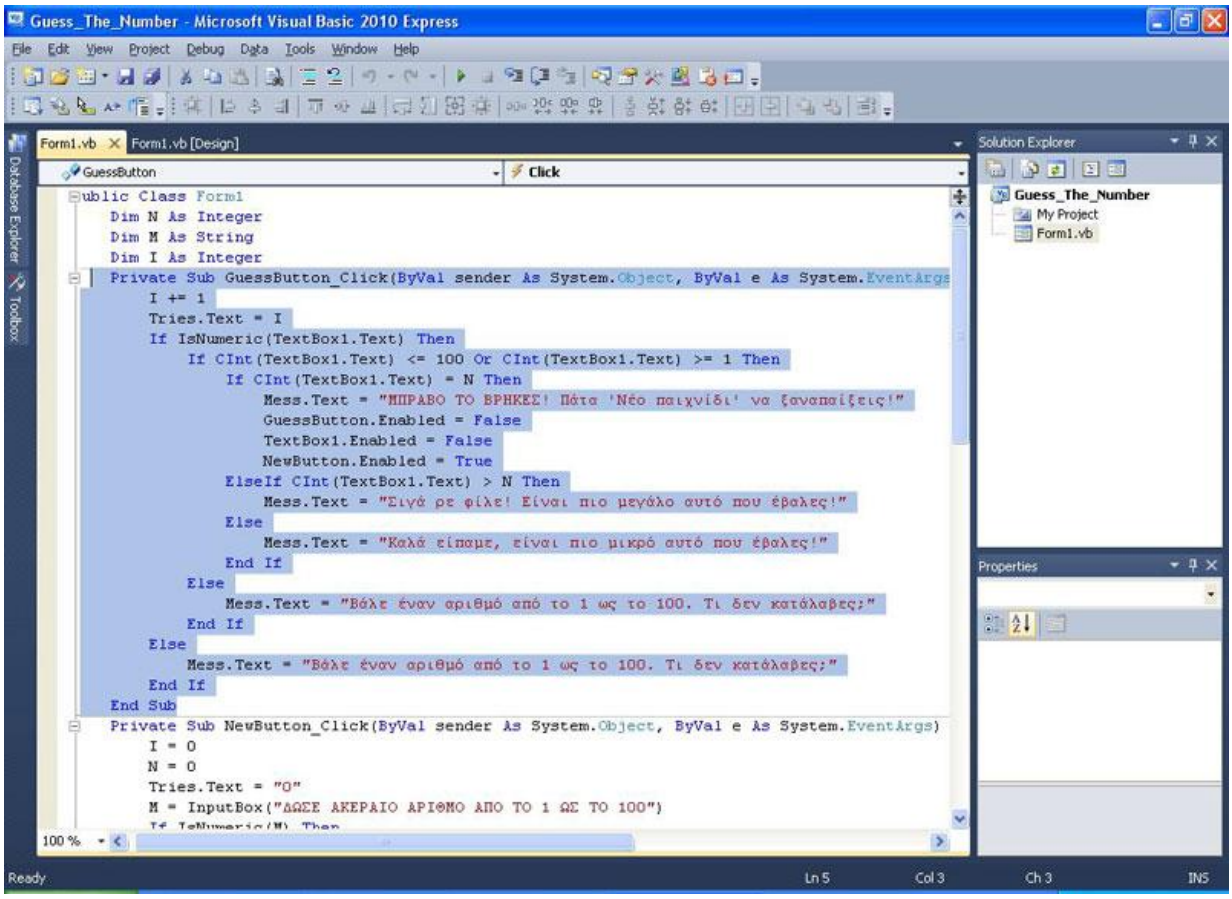
Το πρώτο αυτό Else αναφέρεται στο:
`If CInt(TextBox1.Text) <= 100 Or CInt(TextBox1.Text) >= 1 Then`
Αν δηλαδή δεν είναι μεταξύ 1 και 100 γράψε του μήνυμα
"Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"

```
Else
Mess.Text = "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"
End If
```

Έχουμε κι ένα ίδιο Else από κάτω λοιπόν.
Αυτό αναφέρεται στο
`If IsNumeric(TextBox1.Text) Then`
κοινώς αν δεν είναι αρθμός πες του να βάλει αριθμό κι όχι κείμενο.

ΠΑΡΑΤΗΡΗΣΗ:
Το πρόγραμμα αυτό είναι αγενές: «ΜΠΡΑΒΟ ΤΟ ΒΡΗΚΕΣ! Πάτα 'Νέο παιχνίδι' να ξαναπαίξεις! Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες; Σιγά ρε φίλε! Είναι πιο μεγάλο αυτό που έβαλες! Καλά είπαμε, είναι πιο μικρό αυτό που έβαλες!»
Απευθύνεται στο χρήστη στον ενικό... "ΜΠΡΑΒΟ ΤΟ ΒΡΗΚΕΣ!" Πάτα 'Νέο παιχνίδι' να ξαναπαίξεις! "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;"
Είναι απότομο, λες και είναι κάποιος εξυπνάκιας... "Σιγά ρε φίλε! Είναι πιο μεγάλο αυτό που έβαλες!" Απευθύνεται, λες κι είναι φίλος σου και μάλιστα με μεγάλη οικειότητα.

Γενικώς, το πρόγραμμα θα πρέπει να αντικατοπτρίζει ήθος και τυπικότητα. Αν φτιάχνετε κάποιο πρόγραμμα για δική σας χρήση, δεν θα υπάρξει πρόβλημα. Φανταστείτε όμως σε ένα σοβαρό πρόγραμμα, που μάλιστα πληρώσατε για να αποκτήσετε, να υπάρχει μια τέτοια συμπεριφορά. Κάποιους δε θα τους πείραζε. Όμως υπάρχουν αρκετοί άνθρωποι, που θα τους ενοχλούσε πάρα πολύ. Οπότε στα μηνύματα που θα δίνει το πρόγραμμα στο χρήστη, φροντίζετε να κρατάτε τυπικότητα κι αποστάσεις. Εδώ είναι ένα παιχνίδι. Όταν το τρέξετε, θα δείτε τι εννοούμε όμως. Επίτηδες το κάναμε αγενές, για να δώσουμε έμφαση σε αυτό το θέμα. Τα προγράμματα πρέπει να είναι ευγενικά λοιπόν, διότι απευθύνονται σε πολλούς χρήστες. Χρήστες, τους οποίους προσωπικά δε γνωρίζετε.
Επίσης να παρατηρήσουμε ότι το μήνυμα: "Βάλε έναν αριθμό από το 1 ως το 100. Τι δεν κατάλαβες;" πέραν της αγένειάς του, εμφανίζεται τόσο στην περίπτωση, που ο χρήσης έβαλε μεγαλύτερο του 100 ή μικρότερο του 1 αριθμό, όσο και στην περίπτωση που δεν έβαλε αριθμό, αλλά κάτι άλλο. Πώς όμως θα ξέρει ο χρήστης ποιο ήταν το λάθος του; Γενικώς, μηνύματα τύπου: "Συνέβη το λάθος Α ή το λάθος Β ή το λάθος Γ", που δε λένε συγκεκριμένα τι πρόβλημα παρουσιάστηκε, πρέπει να αποφεύγονται. Ο χρήστης πρέπει σχεδόν πάντοτε να ξέρει ποιό είναι το λάθος που κάνει. Ο μόνος που μπορεί να τον πληροφορήσει όμως είναι το ίδιο το πρόγραμμα. Έχετέ το κατά νου αυτό.
Παρακάτω βλέπετε τον κώδικα που μόλις αναλύσαμε.



Κάπως πρέπει να αρχίζει το παιχνίδι σωστά; Εδώ έχουμε λοιπόν τον κώδικα του NewButton "ΝΕΟ ΠΑΙΧΝΙΔΙ". Εδώ ξεκινάνε όλα. Ακολουθεί ο κώδικας του Private Sub NewButton_Click και φυσικά η επεξήγηση.

```
Private Sub NewButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
Handles NewButton.Click

    I = 0
    N = 0
    Tries.Text = "0"
    M = InputBox("ΔΩΣΕ ΑΚΕΡΑΙΟ ΑΡΙΘΜΟ ΑΠΟ ΤΟ 1 ΩΣ ΤΟ 100")
    If IsNumeric(M) Then
       If Len(M) <= 3 Then
          If CInt(M) > 0 And CInt(M) < 101 Then
             GuessButton.Enabled = True
             TextBox1.Enabled = True
             NewButton.Enabled = False
             N = CInt(M)
          Else
             Mess.Text = "Βάλε ακέραιο αριθμό από το 1 ως το 100!"
          End If
       Else
          Mess.Text = "ΔΩΣΕ ΑΚΕΡΑΙΟ ΑΡΙΘΜΟ ΑΠΟ ΤΟ 1 ΩΣ ΤΟ 100"
       End If
```

```
      Else
         Mess.Text = "Δεν έβαλες αριθμό!"
      End If
   End Sub

I = 0
N = 0
Tries.Text = "0"
```
Αρχικοποίηση τιμών (initialization)! Γιατί; Αν ήδη παίξαμε ένα παιχνίδι, το I και το Tries.text είναι ίσα με τις προηγούμενες προσπάθειες.
Το N επίσης, έχει την τιμή, που δόθηκε από την προηγούμενη παρτίδα.
Το N είναι ο αριθμός που πρέπει να μαντέψουμε. Όλα αυτά τα αρχικοποιώ σε 0 λοιπόν.

```
M = InputBox("ΔΩΣΕ ΑΚΕΡΑΙΟ ΑΡΙΘΜΟ ΑΠΟ ΤΟ 1 ΩΣ ΤΟ 100")
```
Να και η InputBox που αναλύσαμε πριν!
Πηγαίνετε να δείτε στην Public Class Form1 τι είχαμε ορίσει τη M.
Την είχαμε ορίσει ως String! Γιατί; Ο χρήστης μπορεί να βάλει σε ένα InputBox κείμενο.
Οπότε επειδή αποθηκεύεται το input άμεσα σε μια μεταβλητή, η μεταβλητή αυτή, στο παράδειγμα η M, πρέπει να είναι τύπου String.

```
If IsNumeric(M) Then
```
Αν λοιπόν αυτό που βάλαμε στο InputBox είναι όντως αριθμός τότε...

```
If Len(M) <= 3 Then
```
αν είναι όντως τα ψηφία του από ένα (1) ως τρία (100) τότε...

```
If CInt(M) > 0 And CInt(M) < 101 Then
```
αν το M αφού μετατραπεί σε ακέραιο είναι μεγαλύτερο του 0 και μικρότερο του 101 δηλαδή είναι μεταξύ 1 και 100, τότε...

```
GuessButton.Enabled = True
```
ενεργοποιείται κανονικά η διαδικασία του παιχνιδιου, κατά την οποία ο πάικτης μπορεί να μαντέψει, οπότε ενεργοποιείται το αντίστοιχο κουμπί.

```
TextBox1.Enabled = True
```
Ενεργοποιείται φυσικά και η δυνατότητα εισόδου δεδομένων στο TextBox1

```
NewButton.Enabled = False
```
κι απενεργοποιείται η δυνατότητα ΝΕΟ ΠΑΙΧΝΙΔΙ, μέχρις ώτου να τελειώσει η συγκεκριμένη παρτίδα.

```
N = CInt(M)
```
Το N παίρνει τη στρογγυλοποιημένη τιμή του M ακίνδυνα.

```
Else
Mess.Text = "Βάλε ακέραιο αριθμό από το 1 ως το 100!"
End If
```

Αυτό το Else, το πρώτο, αναφέρεται στην τελευταία If.
Δηλαδή την: If CInt(M) > 0 And CInt(M) < 101 Then

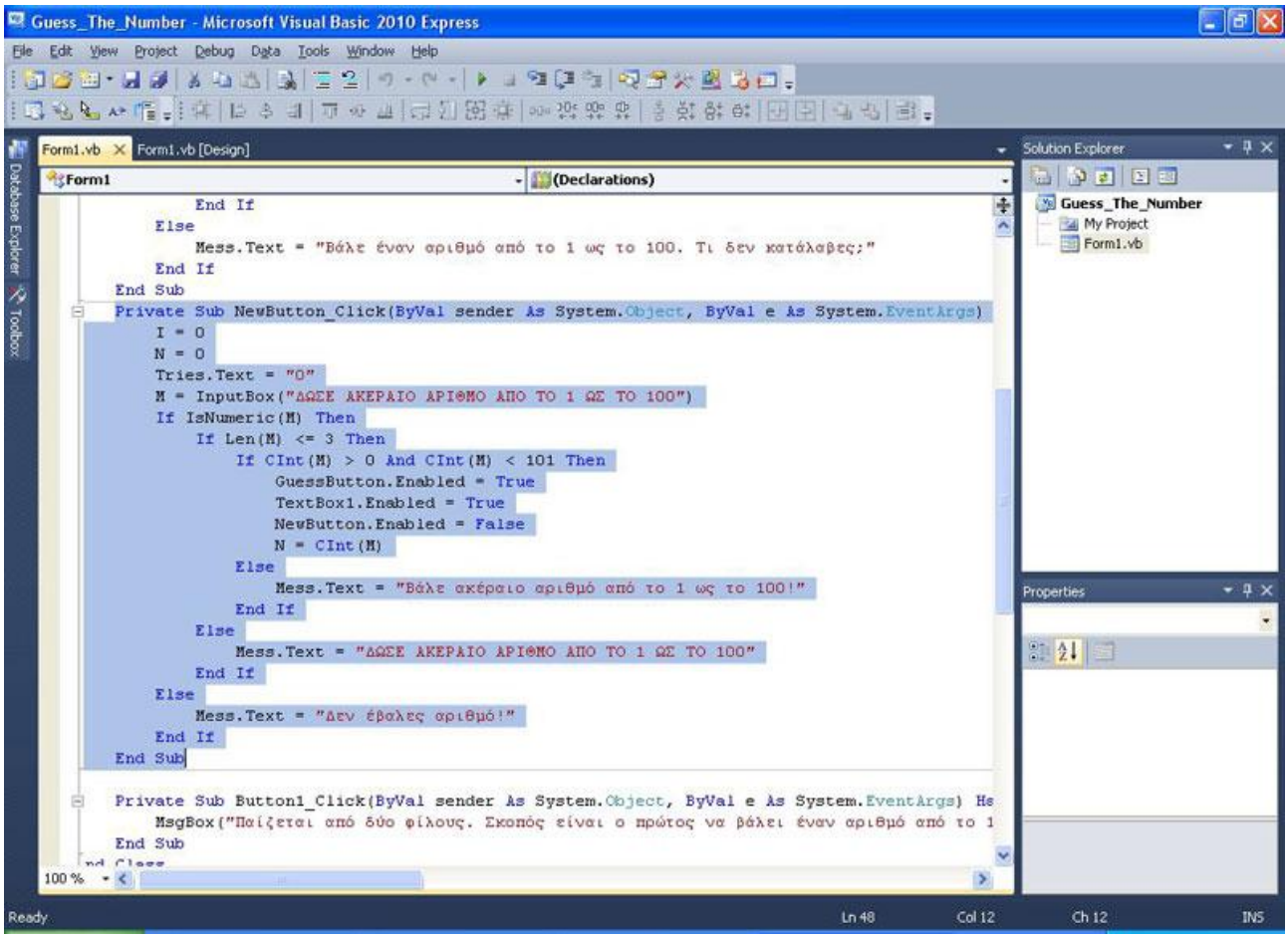
Οπότε τι γίνεται; Αν δε βάλαμε σωστό αριθμό, μας εμφανίζει ένα μήνυμα επάνω στη φόρμα, και δεν κάνει τίποτα άλλο.

```
Else
Mess.Text = "ΔΩΣΕ ΑΚΕΡΑΙΟ ΑΡΙΘΜΟ ΑΠΟ ΤΟ 1 ΩΣ ΤΟ 100"
End If
```

Αυτό το Else, αναφέρεται στην: If Len(M) <= 3 Then οπότε αν έβαλε κάποιον δεκαδικό τύπου, 12.36756786 το πρόγραμμα, αν και μπορεί να τον στρογγυλοποιήσει θα του εμφανίσει το μήνυμα που θα του υπενθυμίζει πως πρέπει να βάλει ακέραιο, και θα σταματήσει εκεί, περιμένοντας το χρήστη να επιχειρήσει να βάλει αριθμό πάλι.

```
Else
Mess.Text = "Δεν έβαλες αριθμό!"
End If
End Sub
```

Αυτό το Else, αναφέρεται στην: If IsNumeric(M) Then οπότε αν δεν έχει βάλει αριθμό ο χρήστης, αλλά κείμενο για παράδειγμα, θα πρέπει να εμφανιστεί αντίστοιχο μήνυμα και ομοίως με τις άλλες περιπτώσεις το πρόγραμμα θα σταματήσει εκεί, περιμένοντας το χρήστη να επιχειρήσει να βάλει αριθμό πάλι. Εδώ τελειώνει κι ο κώδικας με το End Sub.
Στην παρακάτω εικόνα φαίνεται ο κώδικας που μόλις αναλύσαμε.



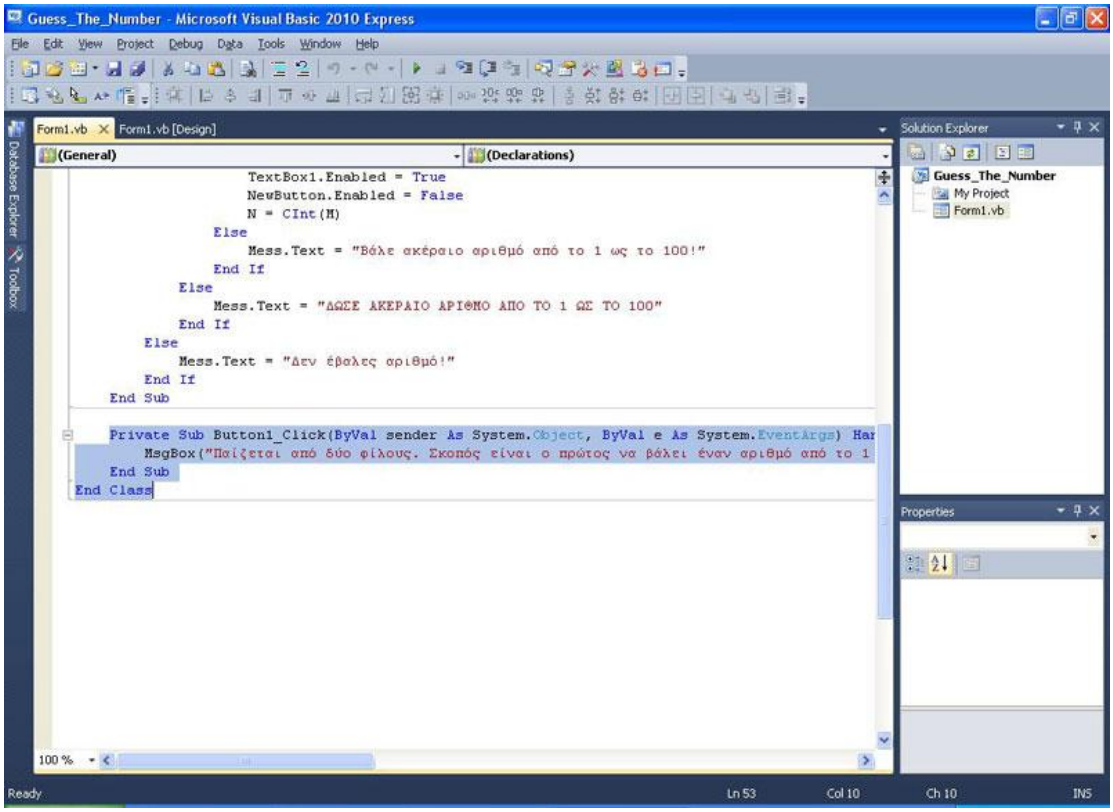
```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
 Handles Button1.Click

 MsgBox("Παίζεται από δύο φίλους. Σκοπός είναι.....")
   End Sub
```

Ουσιαστικά εδώ το κείμενο όλο είναι το παρακάτω:

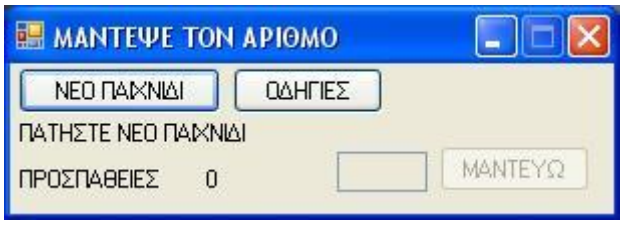
"Παίζεται από δύο φίλους.
Σκοπός είναι ο πρώτος να βάλει έναν αριθμό από το 1 ως το 100 χωρίς ο δεύτερος να βλέπει. Έπειτα ο δεύτερος καλείται με όσο το δυνατόν λιγότερες προσπάθειες, να μαντέψει τον αριθμό. Καλή διασκέδαση!"

Οπότε όλο αυτό, το γράφετε στην ίδια γραμμή της εντολής, αντί του "Παίζεται από δύο φίλους. Σκοπός είναι....." που έχουμε βάλει εδώ για λόγους απλότητας.

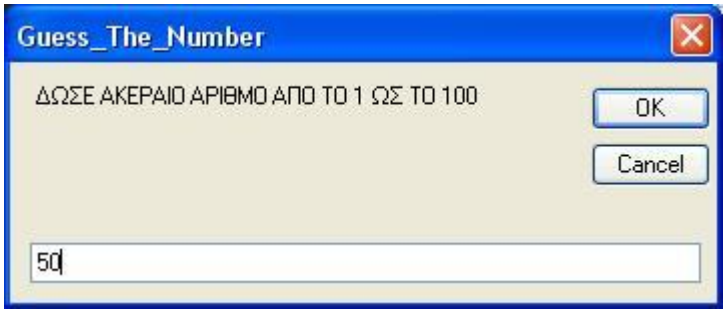
Το Button1 το αφήσαμε σαν ονομασία ως Button1 για τρεις λόγους. Πρώτον δε χρησιμοποιείται κάπου αλλού η ονομασία του, δεύτερον δεν κάνει κάτι κάπου αλλού με κώδικα, πέραν του να εμφανίζει ένα Inputbox, και τρίτον είναι πάντοτε Enabled.
Οπότε δε θα μας μπέρδευε ως Button1.



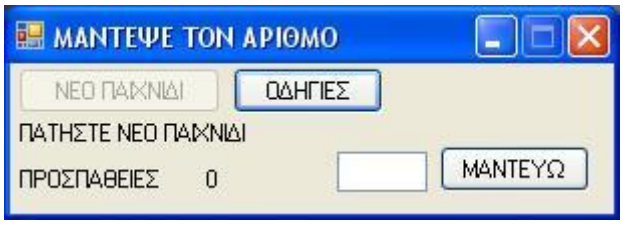
Η παρακάτω εικόνα μας δείχνει το πρόγραμμα με το που το τρέξουμε.



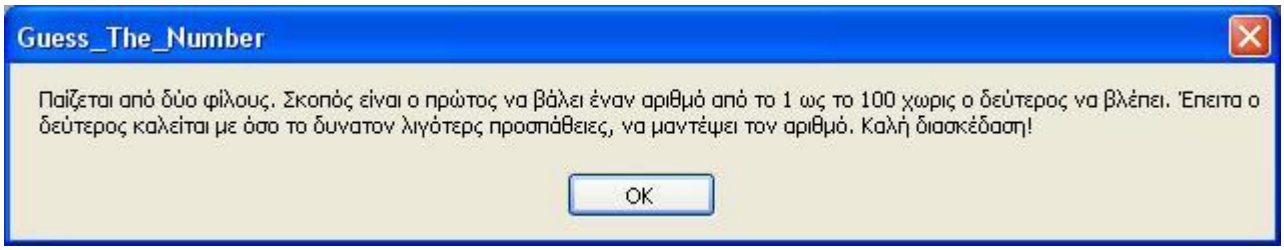
Πατήσαμε ΝΕΟ ΠΑΙΧΝΙΔΙ και βάλαμε τον αριθμό 50. Πατήστε OK.



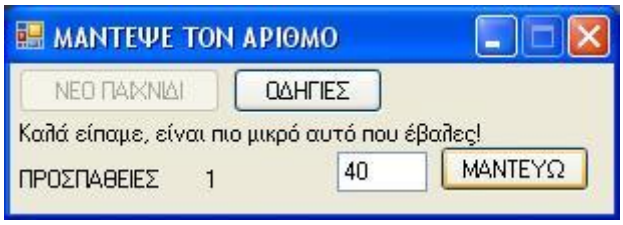
Το ΜΑΝΤΕΥΩ ενεργοποιείται μαζί με το TextBox ενώ το ΝΕΟ ΠΑΙΧΝΙΔΙ απενεργοποιήθηκε. Πατήστε το ΟΔΗΓΙΕΣ.



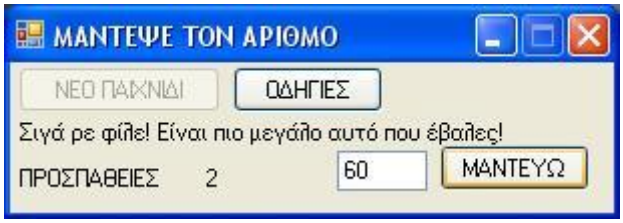
Εμφανίζεται το MessageBox.



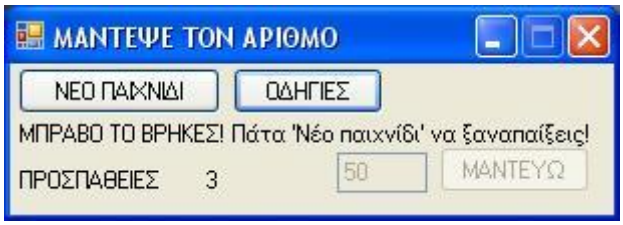
Πατήστε OK και βάλτε τον αριθμό 40 στο textbox. Πατήστε ΜΑΝΤΕΥΩ. Οι προσπάθειες αυξήθηκαν κατά 1 και σας δόθηκε μήνυμα πως βάλατε μικρότερο.



Βάλτε το 60 και πατήστε ΜΑΝΤΕΥΩ. Δύο οι προσπάθειες. Αντίστοιχο μήνυμα μεγαλύτερου εμφανίζεται. Βλέπετε πόσο άσχημα φαίνεται το μη τυπικό και αγενές ύφος του προγράμματος;



Τελειώνοντας, ας "μαντέψουμε" το 50 που βάλαμε. Βάλτε 50 και πατήστε ΜΑΝΤΕΥΩ.



Προτείνουμε σαν άσκηση, να:
1) Κάνετε ευγενικό και τυπικό το πρόγραμμα.
2) Μετατρέψετε τα μηνύματα λάθους σε πιο σαφή μηνύματα, ώστε να καταλαβαίνει ο χρήστης τι συμβαίνει. Μπορεί να χρειαστεί να μεγαλώσετε τη φόρμα βέβαια, για να χωρέσει το μήνυμα. Κάντε ότι χρειάζεται.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

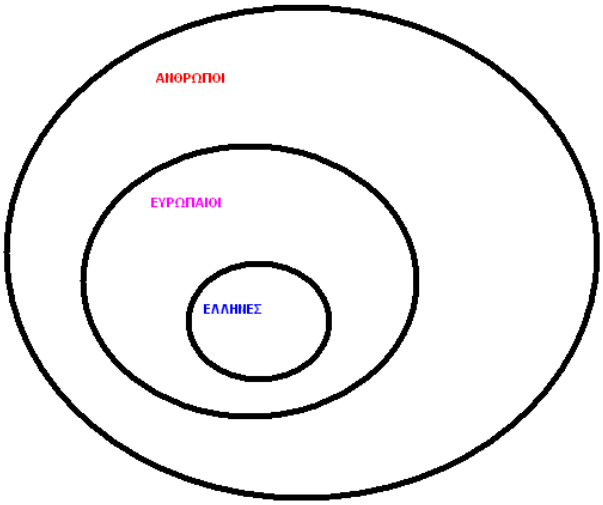
Σε αυτό το κεφάλαιο ο μαθητευόμενος εξοικειώνεται με τα message box και input box, ενώ παράλληλα εξασκεί τις γνώσεις του, φτιάχνοντας ένα παιχνίδι. Το παιχνίδι μαντέματος αριθμού, είναι ένα κλασικό παράδειγμα άσκησης προγραμματισμού. Ωστόσο το κάναμε αγενές, για να επεξηγήσουμε στην πράξη, πόσο σημαντικό είναι να είμαστε τυπικοί στη σύνταξη μηνυμάτων προς το χρήστη, ενός προγράμματος.

ΚΕΦΑΛΑΙΟ 12
ΚΑΤΑΝΟΩΝΤΑΣ ΤΑ ΑΝΤΙΚΕΙΜΕΝΑ, ΤΙΣ ΤΑΞΕΙΣ ΚΑΙ ΤΙΣ ΛΕΙΤΟΥΡΓΙΚΕΣ ΜΟΝΑΔΕΣ ΚΩΔΙΚΑ

Μέχρι τώρα, η διαδικασία που ακολουθούσαμε, για τη συγγραφή ενός προγράμματος, ήταν να φτιάξουμε μία φόρμα, να τη γεμίσουμε με εργαλεία με τα οποία ο χρήστης θα μπορεί να αλληλεπιδρά με το πρόγραμμα (κουμπιά, ετικέτες κτλ...) και έπειτα γράφαμε κώδικα. Λέγαμε πως η φόρμα είναι μία κλάση (τάξη) και πως το κουμπί είναι ένα αντικείμενο. Στην πραγματικότητα και η φόρμα είναι αντικείμενο. Πριν σας μπερδέψουμε όμως εντελώς, πρέπει να διαλευκάνουμε αυτούς τους όρους.

Υποθέστε πως ο αντικειμενοστραφής προγραμματισμός, επιχειρεί να παραστήσει οτιδήποτε υπάρχει στο φυσικό κόσμο με κάποιο τρόπο, έτσι ώστε να δοθεί λύση σε ένα πρόβλημα. Ένα κουμπί σε έναν υπολογιστή τσέπης (calculator), έχει φυσική υπόσταση. Βγάζουμε από την τσέπη μας το "κομπιουτεράκι", πατάμε τα κουμπιά, μας δίνει αποτέλεσμα. Τα κουμπιά αυτά, αναπαριστούνται λοιπόν στο προγραμματισμό, με το Button. Η οθόνη του calculator μπορεί να αναπαρασταθεί με μία Label. Έστω πως θέλω να αναπαραστήσω τη Σοφία Κομνηνού ή το Γιώργο Οικονομίδη ή τη Σοφία Ρωμανού. Πώς θα το κάνω;

Ας φανταστούμε έναν υποθετικό διάλογο μεταξύ εμάς και της γλώσσας προγραμματισμού. Θέλω να αναπαραστήσω το Γιώργο Οικονομίδη. Θα αρχίσω να λέω στη γλώσσα προγραμματισμού ότι είναι Έλληνας. Ώπα! Και τι είναι "Έλληνες"; Σωστή ερώτηση από μεριάς της γλώσσας. Έλληνες είναι υποσύνολο του πληθυσμού που ορίζουμε ως Ευρωπαίοι. Α μάλιστα, λέει η γλώσσα. Και τι είναι "Ευρωπαίοι"; Ευρωπαίοι είναι το υποσύνολο από όλους τους ανθρώπους στη γη, οι οποίοι ζουν στην Ευρώπη. Μάλιστα. Κατάλαβα. Και τι είναι άνθρωπος; Εδώ θα αρχίζατε να εκνευρίζεστε σαφώς. Ωστόσο, στον προγραμματισμό, πρέπει να ορίζονται σχεδόν τα πάντα. Φυσικά τα πράγματα είναι πολύ πιο απλά από όσο φαντάζουν. Επίσης όσο κι αν φαίνεται παράδοξο αρχικά, αυτός ο ορισμός βοηθάει στη δημιουργία καλύτερων κι αποτελεσματικότερων προγραμμάτων με όλο και πιο εύκολο τρόπο. Η διαδικασία δε γίνεται έτσι αλλά αντίστροφα.
Δείτε την παρακάτω εικόνα.



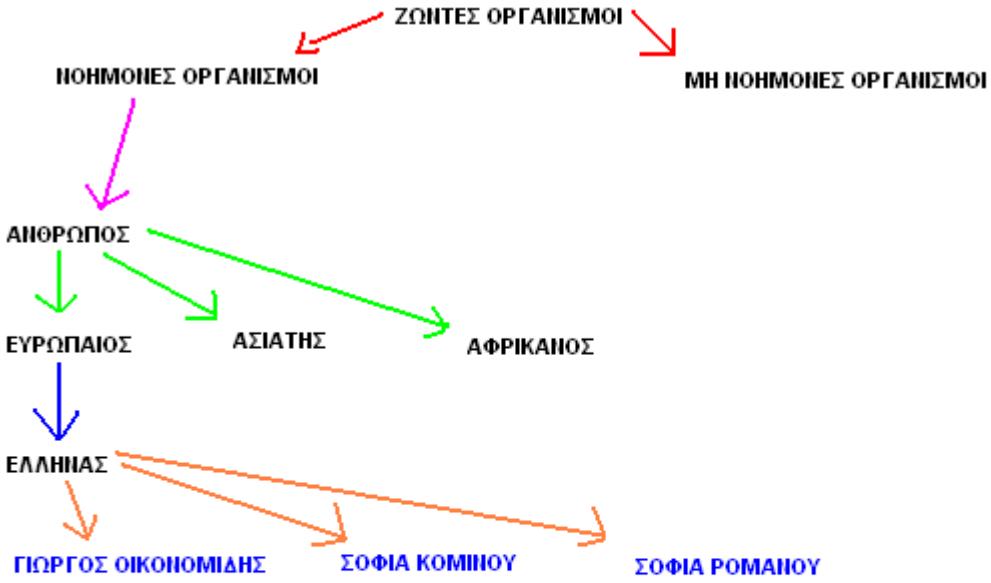
Τι έχουμε κάνει εδώ; Ορίζουμε τι είναι άνθρωπος. Κάθε άνθρωπος έχει κάποια κοινά χαρακτηριστικά. Έχει δύο μάτια, δύο πόδια, δύο χέρια, ο εγκέφαλος βρίσκεται μέσα στο κρανίο, το κρανίο ενώνεται με το σώμα μέσω του λαιμού, που καταλήγει στο βασικό κορμό που... φανταστείτε τέλος πάντων, πως περιγράφουμε τέλεια και εξαντλητικά την κοινή δομή ενός ανθρώπου. Είναι όλοι οι άνθρωποι ίδιοι; Σαφώς και όχι.

Ας υποθέσουμε πως το πρόβλημά μας απαιτεί να προσεγγίσουμε τον άνθρωπο τοπολογικά. Οι άνθρωποι χωρίζονται σε κατηγορίες βάσει του που μένουν λοιπόν εδώ. Ευρωπαίος λοιπόν, είναι ένα υποσύνολο όλων των ανθρώπων, όπως είδαμε στην εικόνα. Είναι ένας μικρότερος κύκλος μέσα στον κύκλο άνθρωποι. Ευρωπαίοι είναι οι άνθρωποι που ζουν στην Ευρώπη λοιπόν. Προσέξτε ότι όλοι οι Ευρωπαίοι είναι άνθρωποι, ωστόσο όλοι οι άνθρωποι ΔΕΝ είναι Ευρωπαίοι.

Υποθέστε ότι δεν υπάρχει πουθενά μετανάστευση. Υποσύνολο των Ευρωπαίων είναι οι Έλληνες. Όλοι οι Έλληνες είναι Ευρωπαίοι,αλλά όλοι οι Ευρωπαίοι ΔΕΝ είναι Έλληνες. Οπότε αν μου πει τώρα κάποιος ότι ο Γιώργος είναι Έλληνας, γνωρίζω αυτόματα ότι ο Γιώργος ζει στην Ελλάδα ΚΑΙ είναι Ευρωπαίος (οπότε ζει και στην Ευρώπη) ΚΑΙ είναι άνθρωπος (οπότε έχει δύο χέρια, δύο πόδια....)

Τα Έλληνας, Ευρωπαίος και άνθρωπος είναι τάξεις - κλάσεις. Φανταστείτε πως τάξη, είναι οδηγίες για την κατασκευή ενός πράγματος. Σαφώς και δεν είναι δόκιμο να πούμε πράγμα ή αντικείμενο τον άνθρωπο. Να μη ξεχνάμε πως αναφερόμαστε στη μοντελοποιήση της έννοιας και όχι στη καθ' αυτή έννοια. Συνεχίζουμε λοιπόν. Όσο πιο γενικευμένες είναι οι τάξεις, τόσο τις λέμε και αφηρημένες. Αφηρημένη Τάξη (Abstract Class) εδώ, είναι σαφώς η άνθρωπος. Από εκεί ξεκινάνε όλα. Ο Γιώργος, που είναι Έλληνας κληρονομεί όλα τα χαρακτηριστικά των αφηρημένων τάξεων που ανήκει η τάξη Έλληνας, δηλαδή των τάξεων Ευρωπαίος και άνθρωπος.

Οι τάξεις λοιπόν περιγράφουν με πολύ συγκεκριμένο τρόπο μια ομάδα αντικειμένων.
Η τάξη άνθρωπος περιγράφει όλους τους ανθρώπους. Η τάξη Ευρωπαίοι περιγράφει μια μικρότερη ομάδα ανθρώπων. Η Ευρωπαίοι κληρονομεί όλα τα χαρακτηριστικά της αφηρημένης τάξης άνθρωπος, στην οποία ανήκει και δίνει κάποια επιπλέον χαρακτηριστικά στα αντικείμενα που ανήκουν σε αυτή.
Δείτε την παρακάτω εικόνα.



Έχουμε μια αφηρημένη τάξη "Ζώντες Οργανισμοί". Τι χαρακτηριστικά έχουν τα "αντικείμενα" που ανήκουν στη τάξη αυτή; Έχουν ζωή. Στην τάξη "Ζώντες Οργανισμοί" ανήκουν οι τάξεις "Νοήμονες Οργανισμοί" και "Μη Νοήμονες Οργανισμοί". Ας δούμε την τάξη "Νοήμονες Οργανισμοί". Τι επιπλέον δίνει στα αντικείμενά της; Δίνει νόηση. Τι έχει κληρονομήσει; Την ιδιότητα "Έχουν ζωή".

Στην τάξη "Νοήμονες Οργανισμοί" ανήκει η τάξη "άνθρωπος". Αυτή η τάξη δίνει όλα τα ανθρώπινα χαρακτηριστικά. Βιολογικά, ψυχολογικά, ανατομικά και τα λοιπά. Για να μη γράψουμε τόμους περιγραφής, ας ονομάσουμε τα πλήρη χαρακτηριστικά του ανθρώπου ως "ΠΧΑ".

Οπότε η τάξη "άνθρωπος" δίνει επιπλέον στα αντικείμενά της "ΠΧΑ". Τι έχει κληρονομήσει από τις προηγούμενες; Νόηση και ζωή. Οι άνθρωποι που ανήκουν στην τάξη Ευρωπαίος, αντίστοιχα κληρονομούν τις ιδιότητες ζωή, νόηση, ΠΧΑ και αποκτούν μια νέα ιδιότητα. Ζει στην Ευρώπη. Αντίστοιχα στην τάξη Έλληνας, τα αντικείμενά της, έχουν κληρονομήσει τις ιδιότητες ζωή, νόηση, ΠΧΑ, Ζει στην Ευρώπη και παράλληλα αποκτούν την ιδιότητα Ζει στην Ελλάδα.

Ας υποθέσουμε πως δε χρειαζόμαστε άλλους διαχωρισμούς σε τάξεις.

Δείτε τι συμβαίνει εδώ.

Πάρτε τη Σοφία Κομηνού. Η Σοφιά Κομηνού ανήκει στην τάξη Έλληνας.
ΑΥΤΟΜΑΤΑ η Σοφία Κομηνού γνωρίζουμε πως:
είναι έμβιος οργανισμός,
είναι νοήμων οργανισμός,
έχει ΠΧΑ,
Ζει στην Ευρώπη,
Ζει στην Ελλάδα.

Αν θέλουμε πλέον να παραστήσουμε οποιονδήποτε Έλληνα στο μοντέλο, έχοντας έτοιμες τις τάξεις, δεν πρόκειται να ξαναπεριγράψουμε τίποτα. Έχουμε πλήρη γενική περιγραφή αυτόματα και για το Γιώργο Οικονομίδη και για τη Σοφία Ρωμανού. Ναι αλλά... ο Γιώργος είναι άντρας έτσι δεν είναι; Και οι Σοφίες είναι γυναίκες, σωστά;

Ας πάρουμε το Γιώργο Οικονομίδη. Λέμε πως είναι στιγμιότυπο του αντικειμένου Έλληνας. Κάθε αντικείμενο, ως στιγμιότυπο, έχει δικά του χαρακτηριστικά, που το κάνουν να διαφέρει από τα άλλα και μάλιστα το κάνει μοναδικό. Εδώ ας πούμε πως τα χαρακτηριστικά είναι τα εξής:

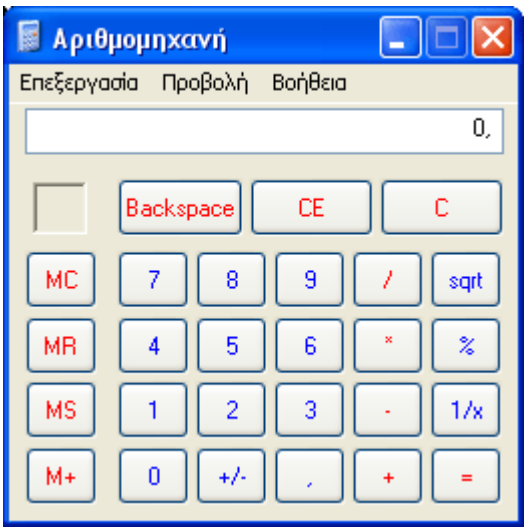
ΟΝΟΜΑ: Γιώργος
ΕΠΙΘΕΤΟ: Οικονομίδης
ΦΥΛΟ: Άνδρας
ΑΡΙΘΜΟΣ ΔΕΛΤΙΟΥ ΤΑΥΤΟΤΗΤΑΣ: ΩL 000000
Ο αριθμός δελτίου ταυτότητος είναι ξεκάθαρα χαρακτηριστικό του στιγμιότυπου του αντικειμένου.
ΣΤΙΓΜΙΟΤΥΠΟ (Instance) ενός αντικειμένου, είναι η εφαρμογή του μοντέλου των τάξεων, για να παραστήσουμε κάτι συγκεκριμένο στο μοντέλο μας.

Προσέξτε ότι το χαρακτηριστικό "ΑΡΙΘΜΟΣ ΔΕΛΤΙΟΥ ΤΑΥΤΟΤΗΤΑΣ" δεν κληρονομήθηκε από κάπου. Δόθηκε εξατομικευμένα. Οπότε κάθε αντικείμενο, έχει τα δικά του χαρακτηριστικά. Επίσης, κάθε αντικείμενο χαρακτηρίζεται εκτός από χαρακτηριστικά, κι από μεθόδους. Η μέθοδος όμως τι είναι; Αναρωτηθείτε τι κάνει ένας άνθρωπος.
Μιλάει, σκέφτεται, περπατάει, τρώει... Αυτά όλα τα έχει κληρονομήσει από την κλάση άνθρωπος. Κατανοεί την Ελληνική γλώσσα, αυτό το κληρονομεί από την κλάση Έλληνας.
Ας υποθέσουμε πως ο Γιώργος είναι και φοιτητής. Εξατομικευμένα λοιπόν, μελετάει και παρακολουθεί μαθήματα. Αυτές όλες οι πράξεις που κάνει, και πολλές άλλες, είναι οι μέθοδοί του.

Πιο πρακτικά, σε προγραμματιστική σκοπιά, έστω πως έχουμε ένα button.
Το button γίνεται click σωστά;
Αυτό το κλικ είναι μια μέθοδος, εκτός του ότι είναι συμβάν.
Πατιέται λέμε. Αρκετά απλό.

Ας επιστρέψουμε τώρα στο παράδειγμα με το calculator.
Στο φυσικό κόσμο το calculator μοιάζει με την παρακάτω εικόνα σε γενικές γραμμές.

Ας δούμε το calculator από τα Βοηθήματα του λειτουργικού συστήματος Windows XP της Microsoft.



Τι παρατηρούμε; Γνωστά χαρακτηριστικά. Ένα μεγάλο Textbox, μια φόρμα και 27 buttons! Έχει και μια μπάρα μενού, αλλά θα ασχοληθούμε παρακάτω με αυτό. Ας μελετήσουμε τα buttons. Καθένα έχει δικές του διαστάσεις, κείμενο χρώμα κειμένου και γνωρίζετε πως, αν και δε φαίνεται στο χρήστη, κάθε button έχει δική του μοναδική ονομασία.

Ο προγραμματιστής όμως, δεν έδινε πάντα εντολές στον ηλεκτρονικό υπολογιστή για το πως να σχεδιάσει buttons. Ήταν καλυμμένος από την ήδη κατασκευασμένη τάξη για τα button, που και εσείς απολαμβάνετε στη Visual Basic .NET! Φανταστείτε λοιπόν τις τάξεις, σαν οδηγίες προς ένα εργοστάσιο. Δίνεις στο εργοστάσιο οδηγίες να φτιάξει buttons κι έχεις όσα θες! Η Form1 λοιπόν, είναι αντικείμενο και μάλιστα ανήκει σε συγκεκριμένες τάξεις. Κάθε φόρμα σαν στιγμιότυπο έχει διαφορετικές διαστάσεις, αλλά όλες οι φόρμες έχουν κοινά χαρακτηριστικά. Σας θυμίζει κάτι; Φανταστείτε την τάξη λοιπόν σαν ένα πατρόν για να φτιαχτούν φουστάνια, σαν ένα καλούπι για να γεμίσει χυτοσίδηρο και να φτιαχτεί ένα αγαλματάκι.

Θυμόσαστε τι είπαμε στην αρχή;
"Υποθέστε πως ο αντικειμενοστραφής προγραμματισμός, επιχειρεί να παραστήσει οτιδήποτε υπάρχει στο φυσικό κόσμο με κάποιο τρόπο, έτσι ώστε να δοθεί λύση σε ένα πρόβλημα".

Σταματήστε να υποθέτετε. Είναι ακριβώς έτσι! Μοντελοποιούνται τα πάντα, αντικείμενα κι έννοιες. Κι έπειτα το μοντέλο τίθεται σε εφαρμογή, ώστε να δοθεί λύση.
Κοιτάξτε γύρω σας!
Κάθε τι στο φυσικό κόσμο είναι σε μία κατάσταση και μια συμπεριφορά.
Δείτε τη λάμπα ενός γραφείου. Είναι σε μία από τις δύο καταστάσεις που μπορεί να είναι. Αναμμένη ή σβηστή. Επίσης έχει δύο μόνο συμπεριφορές. Ανάβει και σβήνει.
Υπάρχουν πιο περίπλοκα αντικείμενα. Ένα ραδιόφωνο έχει περισσότερες καταστάσεις: On, Off , συγκεκριμένη συχνότητα λήψης (σταθμός), συγκεκριμένη ένταση ήχου.

Επίσης έχει περισσότερες συμπεριφορές: Ενεργοποίηση (Turn On), απενεργοποίηση (Turn Off) αύξηση έντασης ήχου, μείωση έντασης ήχου, αναζήτηση ραδιοφωνικού σταθμού, συντονισμός. Αυτό ισχύει και στον προγραμματισμό. Οι ιδιότητες του αντικειμένου είναι η κατάσταση που βρίσκεται. Οι μέθοδοί του, είναι οι συμπεριφορές του.

Οπότε μια συμπεριφορά ενός Button είναι το click.
Μια κατάσταση του Button1 είναι Enabled = True

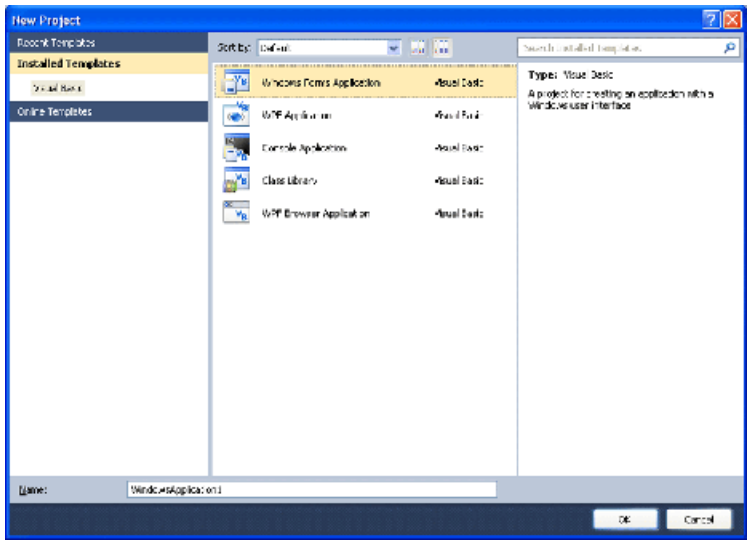
Φανταστείτε τώρα πιο απλά τις τάξεις, ως προσχέδια αντικειμένων. Κατανοητό; Ελπίζουμε πως ναι! Και γι αυτό το λόγο δώσαμε τόσα πολλά παραδείγματα.

Τώρα θα καταλάβετε και ένα προφανές πλεονέκτημα αυτού του τρόπου προγραμματισμού. Κάποτε κάποιοι προγραμματιστές έφτιαξαν κώδικα ο οποίος εμφανίζει ένα κουμπί, το οποίο ο χρήστης το κάνει κλικ κι αυτό εκτελεί κώδικα. Αυτή η τάξη είναι ΕΤΟΙΜΗ και μπορεί να χρησιμοποιηθεί από άλλους προγραμματιστές στα δικά τους προγράμματα. Καταλάβατε τώρα τι συμβαίνει όταν τοποθετείτε ένα Button σε μια φόρμα; Στην ουσία χρησιμοποιείτε έτοιμο κώδικα, που άλλοι έφτιαξαν για το δικό τους πρόγραμμα, αλλά μπορεί να χρησιμοποιηθεί και στο δικό σας! Έτσι δε χρειάζεται να βασανίζεστε για να φτιάξετε γραφικά ξανά και ξανά. Πάμε τώρα να δούμε ένα άλλο θέμα.

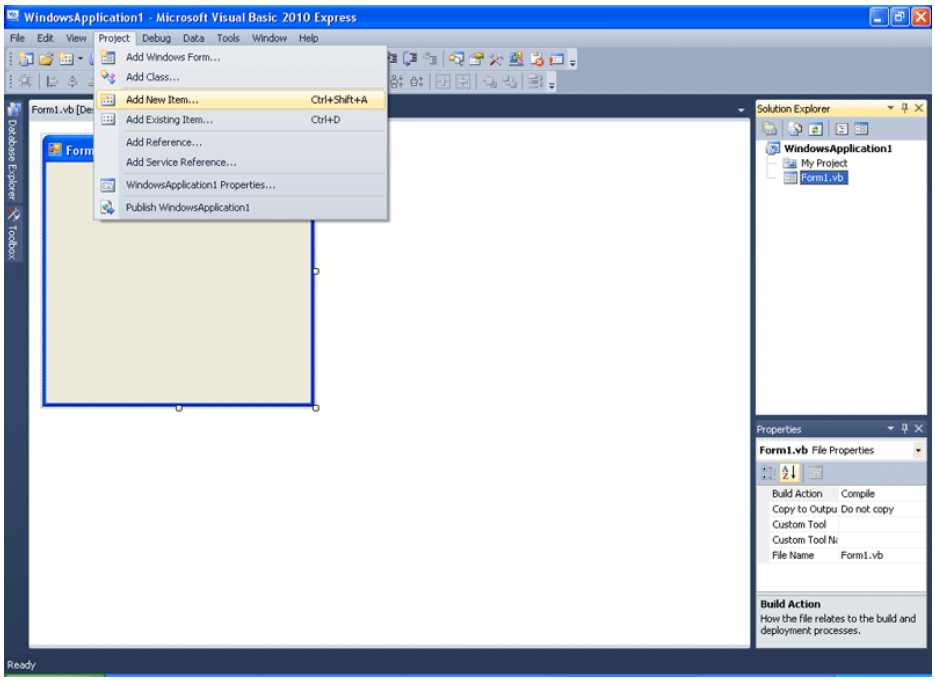
Τα προγράμματα που γράφαμε μέχρι τώρα, επειδή στόχο έχουν να σας βοηθήσουν στην κατανόηση κάποιων εννοιών, κάποιων εργαλείων ή κάποιων προγραμματιστικών εντολών, ήταν μικρά και γενικώς εύκολα κι απλά. Στην πραγματική ζωή, υπάρχουν πολύ πιο περίπλοκα προβλήματα, τα οποία πρέπει να λυθούν. Αν λοιπόν αρχίσουμε να γράφουμε τεράστια προγράμματα, πρέπει να έχουμε κατά νου, ένα πολύ βασικό κανόνα.
Θυμάστε τι συνέβαινε όταν δε χρησιμοποιούσαμε λογικούς τελεστές;
Else
Label5.Text = "ΜΕΤΕΞΕΤΑΣΤΕΟΣ"
End If
Πέντε φορές επανάληψη του ίδιου κώδικα.
Αυτό το φαινόμενο, σε μεγάλα προγράμματα, βαραίνει τόσο αυτόν που συντάσσει το πρόγραμμα, όσο και το μηχάνημα που θα το εκτελέσει. Αυτό το φαινόμενο της άδικης επανάληψης, μπορεί να συμβεί όχι μόνο σε μία κακογραμμένη δομικά IF δίχως λογικούς τελεστές, αλλά και σε ολόκληρα κομμάτια κώδικα.
Φανταστείτε λοιπόν, πως σε ένα πρόγραμμα, θα πρέπει να εκτελεστεί ένα κομμάτι κώδικα. Αυτό το κομμάτι κώδικα, εντελώς ίδιο, θα πρέπει να εκτελεστεί από δέκα διαφορετικές μεταξύ τους φόρμες.

Τι θα κάνουμε;
Το γράφουμε δέκα φορές; Αντιγραφή - επικόλληση; Εύκολη λύση αλλά όχι αποδοτική. Διότι αν το κάνουμε αυτό, αυξάνουμε άδικα το μέγεθος και τη βαρύτητα του προγράμματος. Κι αν πρέπει να μετατρέψουμε αυτό τον κώδικα στο μέλλον; Αν έχει κάποιο μη προφανές σφάλμα; Θα διορθώνουμε ΕΝΑ σφάλμα ΔΕΚΑ φορές; Κακή ιδέα, μη αποδοτικός προγραμματισμός.
Τι θα κάνουμε λοιπόν; Εδώ μας δίνουν λύση οι Λειτουργικές Μονάδες Κώδικα (Modules). Επίσης, όπως και στις τάξεις, τα modules μπορούν να γραφτούν μία φορά και να χρησιμοποιηθούν από πολλά προγράμματα στη συνέχεια. Καλό ε; Ας δούμε λοιπόν τι είναι και πως λειτουργούν!

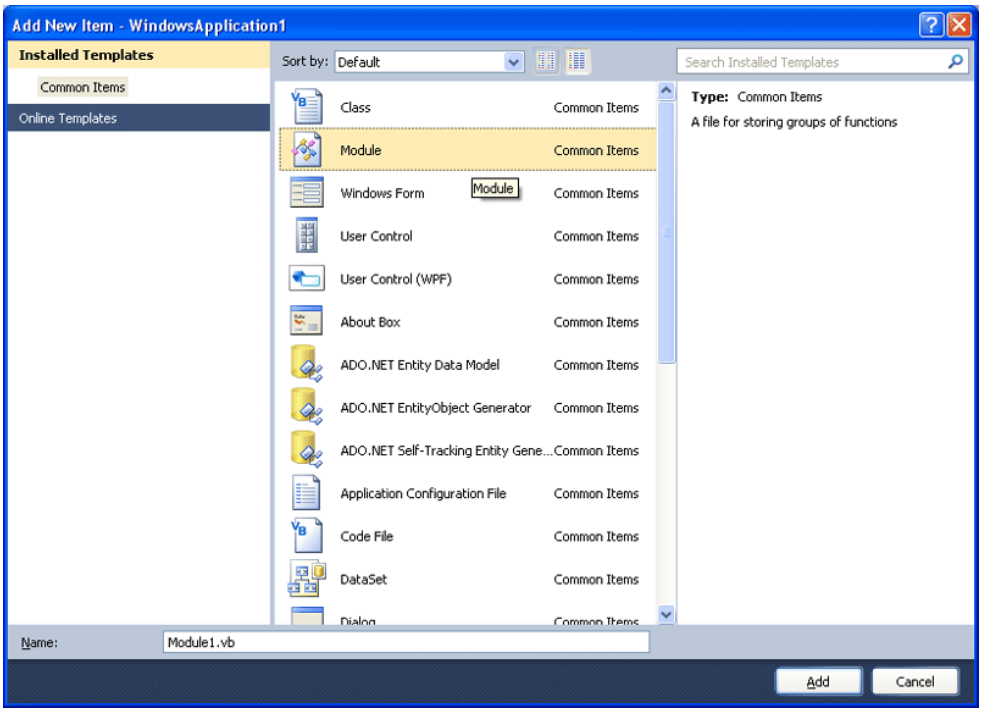
Φτιάχνουμε Windows Forms Application όπως ήδη γνωρίζουμε.



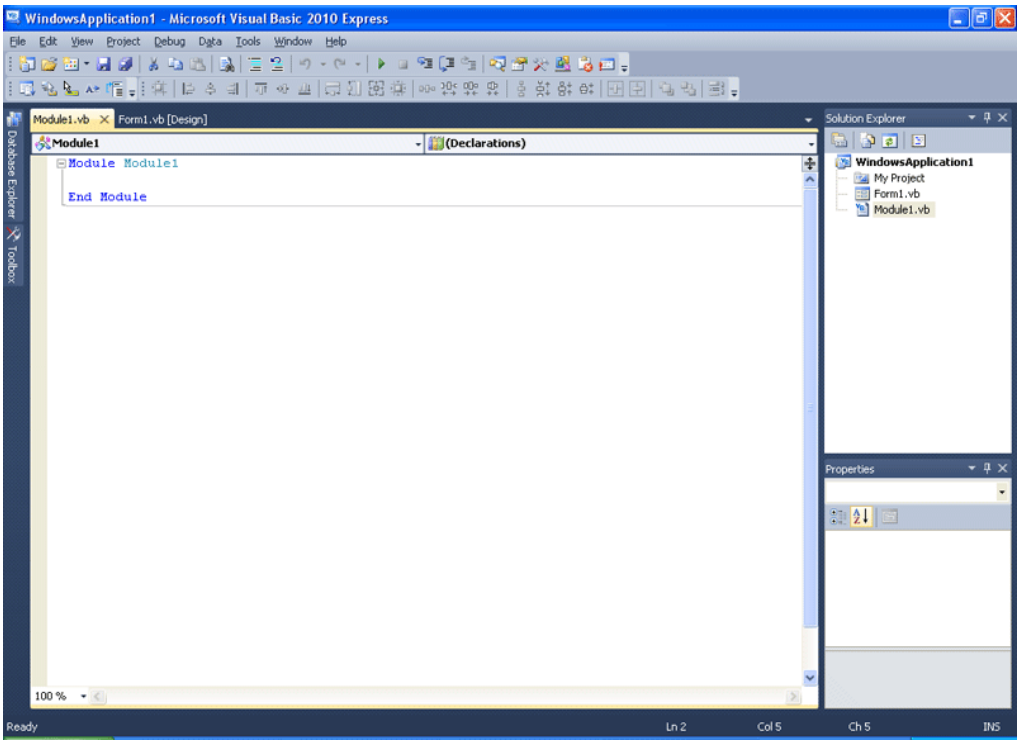
Πάμε στο μενού Project κι επιλέγουμε Add New Item.



Επιλέγουμε Module.



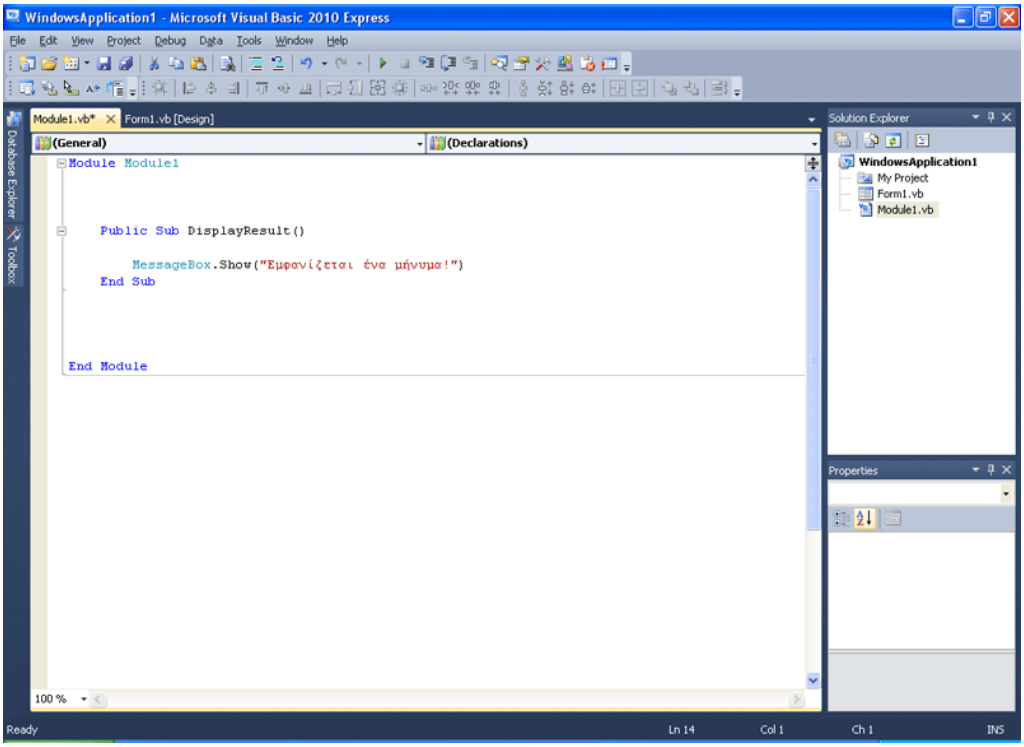
Η γλώσσα, θα φτιάξει την αρχή και το τέλος του Module.



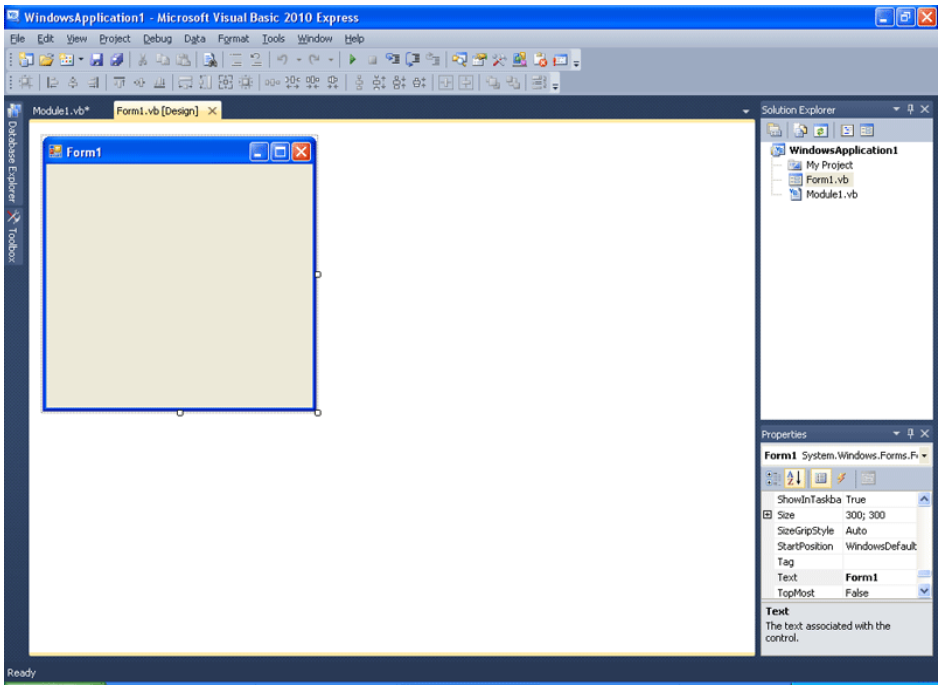
Ας του προσθέσουμε κάτι.
Γράφουμε:

```
Public Sub DisplayResult()
MessageBox.Show("Εμφανίζεται ένα μήνυμα")
End Sub
```

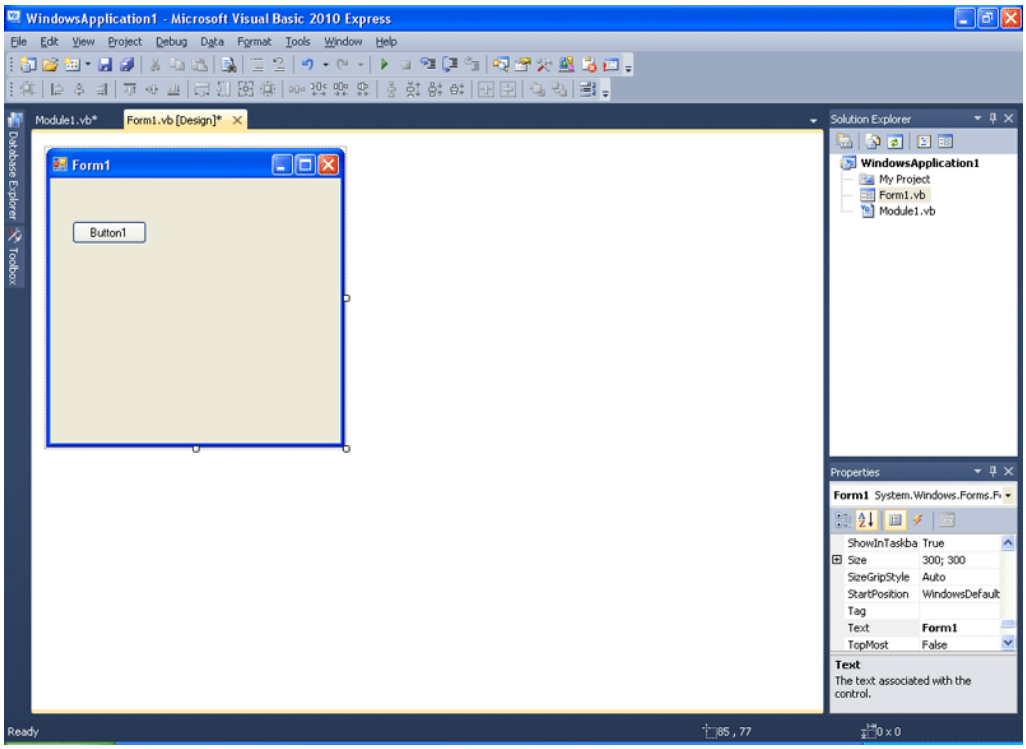
Δώστε πολλή προσοχή εδώ! Τα Buttons, όταν τα εκτελούσαμε, έτρεχαν τον κώδικά τους. Ο κώδικάς τους δεν εκτελείται από κάπου αλλού. Κοινώς δε χρησιμοποιούσε τον κώδικα αυτό άλλο αντικείμενο. Ξεκινούσαν με Private Sub. Εδώ όμως στο Module, επειδή ότι βάλουμε μέσα του, θέλουμε να μπορεί να χρησιμοποιηθεί από άλλα αντικείμενα, δε βάζουμε Private Sub αλλά Public Sub. Ο κώδικας ενός Public Sub είναι λοιπόν κοινόχρηστος. Πώς λέγεται αυτό το Sub; DisplayResult() λέγεται. Οι παρενθέσεις, αν και δεν περιέχουν κάτι είναι απαραίτητες. Θα καταλάβετε αργότερα το γιατί. Από τη στιγμή λοιπόν που ένα κομμάτι κώδικα έχει ονομασία, μπορώ να το καλέσω να εκτελεστεί.



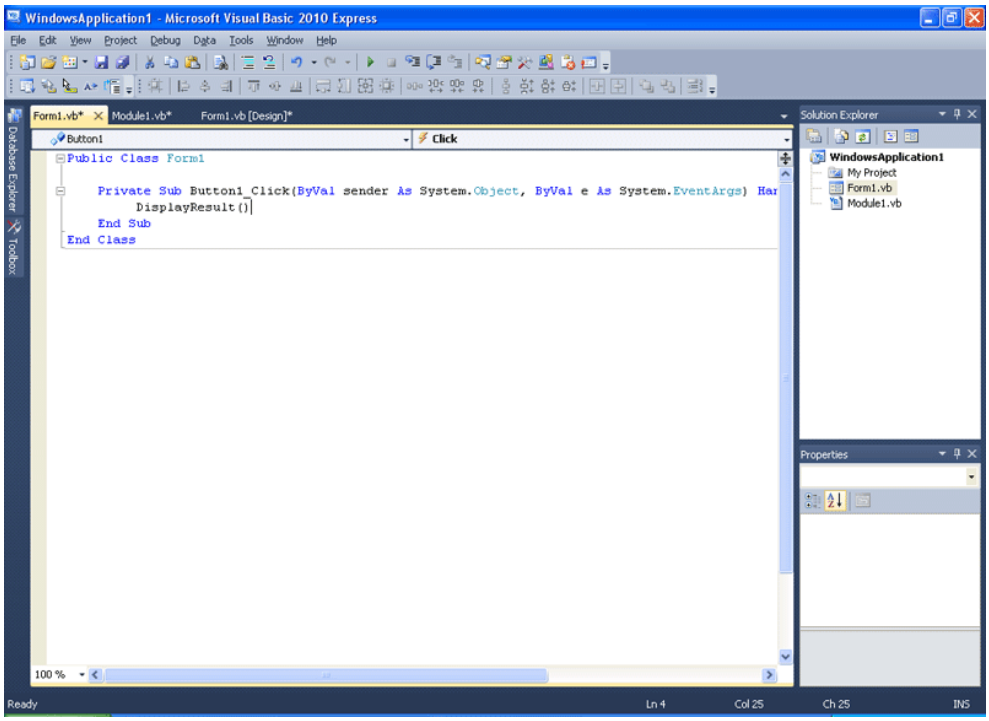
Πηγαίνετε στο Form1.vb[Design]



Βάλτε απλά ένα button.



Κάντε διπλό κλικ στο button και μέσα στον κώδικά του πληκτρολογήστε:
DisplayResult()
Αυτό ήταν! Σοβαρολογούμε! Τρέξτε το πρόγραμμα! Πατήστε το κουμπί.



Παρατηρείτε τι έγινε; Το κουμπί εκτέλεσε τον κώδικά του. Τι λέει ο κώδικας ουσιαστικά; Εκτέλεσε το DisplayResult() Πού βρίσκεται; Στο Module φυσικά! Το Button μπορεί να δει τι περιέχει το Module και μάλιστα μπορεί να δει το DisplayResult() διότι είναι Public Sub

DisplayResult() και όχι Private, που δε θα είχε και πολύ νόημα έτσι κι αλλιώς, μιας κι ως Private δε θα μπορούσε να το δει κανένα άλλο αντικείμενο. Τι έγινε λοιπόν εδώ; Το Button1 κάλεσε την DisplayResult(). Όταν λέμε "κάλεσε" εννοούμε ότι εκτέλεσε ένα κομμάτι κώδικα, που δε βρίσκεται μέσα του. Πράγματι, στο Button1 δεν υπάρχει η εντολή MessageBox.Show("Εμφανίζεται ένα μήνυμα"). Αυτή η εντολή, υπάρχει στη DisplayResult() που βρίσκεται μέσα στο Module1.
Οπότε αυτό που έγινε είναι:
Πατάμε το κουμπί.
Εκτελείται ο κώδικας στο κουμπί.
Ο κώδικας στέλνει το computer στο Module1 να εκτελέσει το Public Sub DisplayResult().
Αυτό εκτελείται κι αφού εκτελεστεί συνεχίζεται ο κώδικας στο κουμπί μας.
Ωστόσο εδώ δεν είχαμε κάτι άλλο να εκτελεστεί, οπότε το γεγονός κλικ λαμβάνει τέλος.

Πριν προχωρήσουμε, θα πρέπει να μιλήσουμε σε αυτό το σημείο λίγο για τις συναρτήσεις. Θυμόσαστε τη συνάρτηση CINT(); Θυμόσαστε τι έκανε;

Αν είχα μια μεταβλητή Α τύπου Double, με έναν δεκαδικό αριθμό μέσα της, η συνάρτηση CINT(A) θα μου έδινε ως αποτέλεσμα, δηλαδή θα μου επέστρεφε, όπως λέμε στον προγραμματισμό, ως στρογγυλοποιημένο ακέραιο, το περιεχόμενο της Α.
Κοινώς αν Α = 10,235645
CINT(A) επιστρέφει 10

Αυτή ήταν μια έτοιμη συνάρτηση. Στην ουσία όμως μπορούμε να δημιουργήσουμε και δικές μας συναρτήσεις. Ας θυμηθούμε λίγα μαθηματικά.
Λέγαμε F(x)= 2x Τι είναι αυτό;
Στο συμβολισμό F(x) ουσιαστικά, το γράμμα F βγαίνει, από το αρχικό F της λέξης Function, που σημαίνει λειτουργία. Οπότε το σύμβολο F(x) μας έλεγε πως έχουμε μια λειτουργία, μια διαδικασία, με μια μεταβλητή x.
Για να δούμε:
Εδώ ο τύπος, δηλαδή η λειτουργία της συνάρτησης F(x), είναι το 2x. Δηλαδή οτιδήποτε βάλω στην τιμή x θα πολλαπλασιαστεί επι του δύο. Στον προγραμματισμό θα λέγαμε, ότι η F(x) μας επιστρέφει το διπλάσιο της μεταβλητής x.

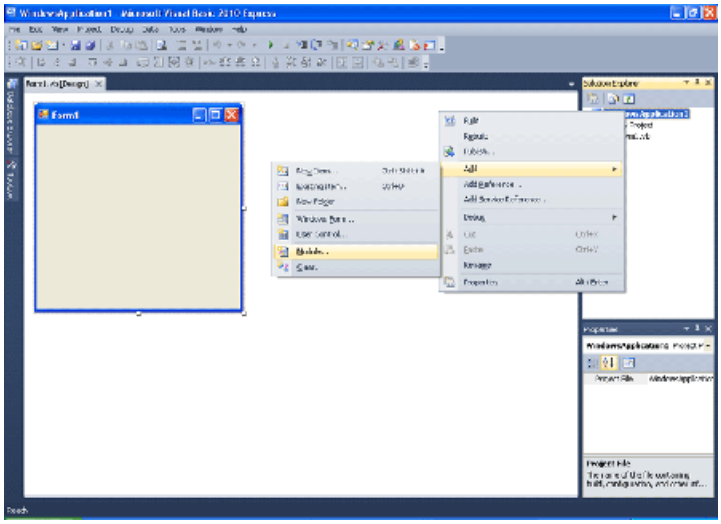
F(x)= 2x

F(0)= 0
F(1)= 2
F(2)= 4
F(3)= 6

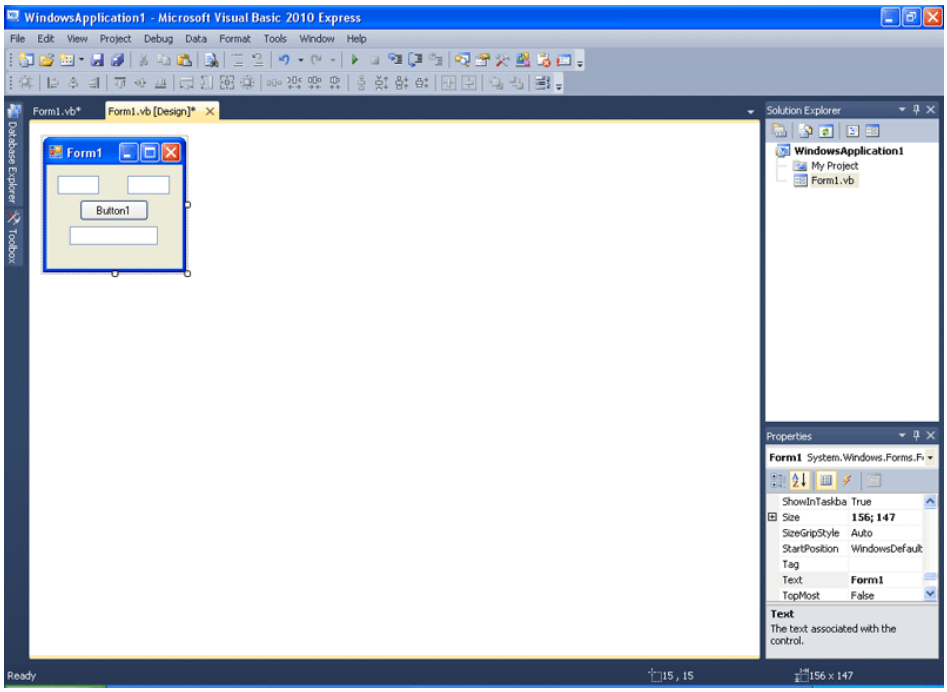
Μάλιστα. Οτιδήποτε βάλω στο x λοιπόν, η F(x) το διπλασιάζει. Αυτό φαίνεται και στον τύπο της όπως είδαμε. Ωστόσο, αυτό μπορούσαμε να το κάνουμε άμεσα στον προγραμματισμό. Δε χρειάζεται κόπο. X = X*2 και τελειώσαμε. Σιγά! Όμως φανταστείτε πως υπάρχουν συναρτήσεις, πολύ πιο πολύπλοκες, με περισσότερες πράξεις, περισσότερες από μία μεταβλητές, ακόμα και με περισσότερους από ένα τύπους. Εδώ όμως δεν έχουμε στόχο να διδάξουμε μαθηματικά και για αυτό το λόγο παίρνουμε απλά παραδείγματα. Φανταστείτε όμως ότι η χρησιμότητα των συναρτήσεων στα Modules, θα γίνεται σε μια πολύ συχνά επαναλαμβανόμενοι, όχι διαδοχικά, διαδικασία, μεγάλης ή μικρής πολυπλοκότητας.
Στην πραγματικότητα, οτιδήποτε επαναλαμβανόμενο μπορεί να γίνει Sub ή Function,
θα πρέπει να γίνεται!

Οσοδήποτε μικρό κι αν είναι. Σας διευκολύνει γιατί θα γράφετε λιγότερο κώδικα. Μπορείτε να μετατρέψετε όλη τη διαδικασία, παντού σε όλο το πρόγραμμά, από μόνο ένα σημείο, το Module, οπότε μπορείτε να βελτιώσετε, προσαρμόσετε, διορθώσετε ευκολότερα το πρόγραμμά σας. Φανταστείτε να γράφατε την ίδια συνάρτηση σε 100 σημεία στο πρόγραμμά σας και να σας πει ξαφνικά κάποιος ότι ήταν λάθος η συνάρτηση που σας δόθηκε. Αντί να γινόσασταν έξω φρενών λοιπόν για τον τόσο χαμένο χρόνο, και κόπο, η δόμηση του προγράμματος με modules θα μπορούσε να σας κάνει να τους απαντήσετε, πως δεν υπάρχει κανένα απολύτως πρόβλημα και πως θα το αλλάξετε άμεσα. Διότι η συνάρτηση είναι μία φορά γραμμένη σε ΕΝΑ σημείο.

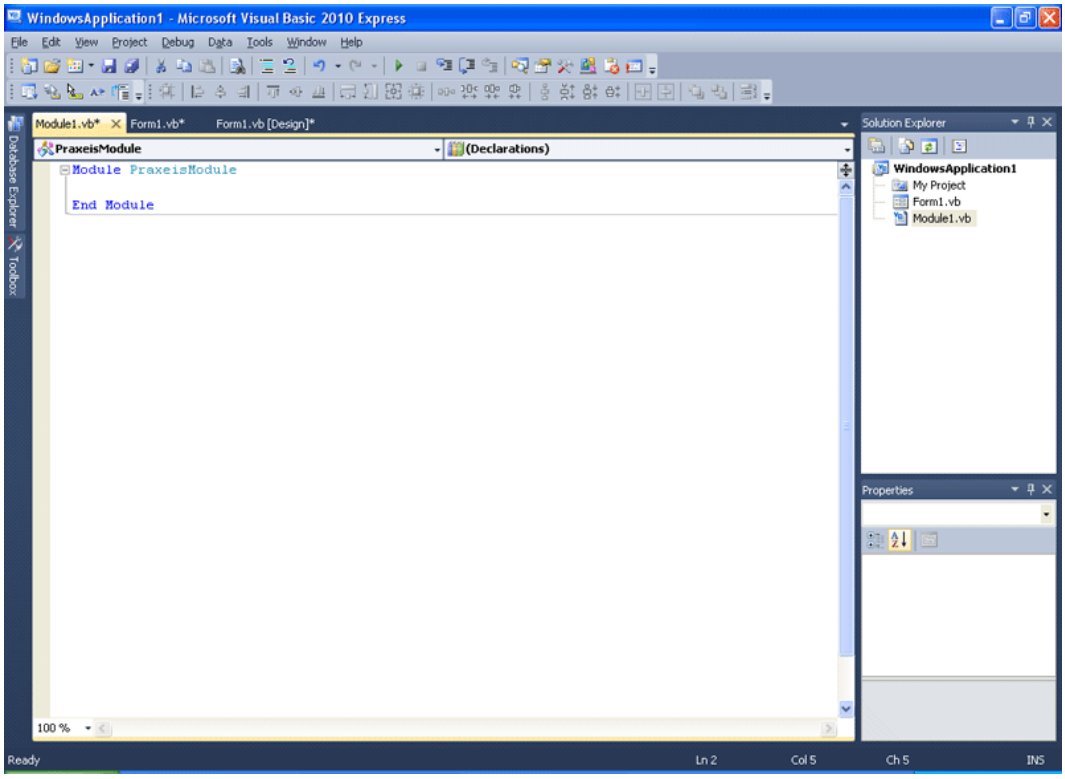
Σβήστε ότι κάνατε.
Ανοίξτε ξανά τη Visual Basic 2010.
Επιλέξτε πάλι Windows Forms Application .
Πηγαίνετε στο Solution Explorer, επιλέξτε το WindowsApplication1
κάντε δεξί κλικ, σύρετε το βελάκι στο μενού Add κι από το αναδυόμενο υπομενού, επιλέξτε το module.
Είναι ακριβώς το ίδιο πράγμα με το να πάτε στο μενού Project
και να κάνετε Add New Item για να επιλέξετε module, όπως κάναμε πριν.



Φτιάξτε τη φόρμα που βλέπετε στην παρακάτω εικόνα.
3 Textbox, 1 Button
Το πάνω αριστερά TextBox είναι το TextBox1. Δεξιά του είναι το TextBox2.
Κάτω από το Button1 είναι το TextBox3.
Θα φτιάξουμε έναν υπολογιστή, που πατώντας ένα κουμπί, προσθέτει δύο δοθέντες ακεραίους αριθμούς που έχουν δοθεί από το χρήστη μέσα στα TextBox1 και TextBox2.
Αυτό μάλιστα θα το κάνει καλώντας μια συνάρτηση από ένα module.
Το αποτέλεσμα θα το εμφανίζει στο TextBox3.
Πάμε!



Αλλάξτε το όνομα του Module1 σε PraxeisModule



Γράψτε τον παρακάτω κώδικα μέσα στο module. Μην ανησυχείτε, θα σας εξηγήσουμε τα πάντα κι αυτή τη φορά θα ασχοληθούμε και με την παρένθεση, διότι είναι ζωτικής σημασίας να καταλάβετε τι συμβαίνει εδώ!

```
Public Function addnumbers(Byval num1 As Integer, Byval num2 As Integer)
Return num1 + num2
End Function
```

Τι έχουμε εδώ;
Public Function addnumbers
Κοινόχρηστη συνάρτηση με όνομα addnumbers λοιπόν. Τι εννοούμε κοινόχρηστη; Άλλα αντικείμενα στο πρόγραμμα (όπως για παράδειγμα το Button1) μπορούν να την καλέσουν. Την παρένθεση θα την εξηγήσουμε στο τέλος.

Return num1 + num2
Τι λέμε εδώ; Επέστρεψε το άθροισμα των μεταβλητών num1 και num2.

End Function
Τελειώνει εδώ η συνάρτηση.

Πού βρέθηκαν όμως ξαφνικά οι μεταβλητές num1 και num2;
Τι γίνεται εδώ;

Ας το δούμε πρώτα μαθηματικά.

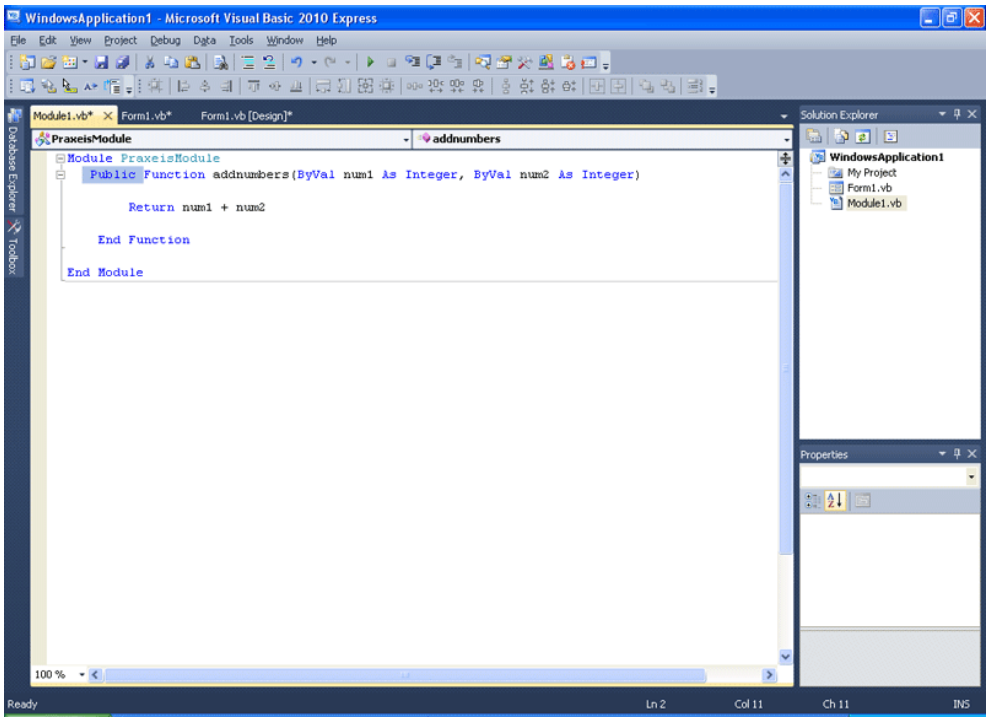
Τι θέλουμε να κάνουμε;
Θέλουμε να πάρουμε δύο οποιουσδήποτε αριθμούς και να τους προσθέσουμε. Η συνάρτησή μας δηλαδή θα λαμβάνει δύο αριθμούς και θα επιστρέφει το άθροισμά τους. Οι αριθμοί θα μπαίνουν λοιπόν σε δύο μεταβλητές, έστω x και y.
Έστω η συνάρτηση λέγεται F.
Έχουμε λοιπόν F(x,y) = x + y
Τι λέμε δηλαδή; Θα δώσουμε τιμή στο x και το y και η F θα μας επιστρέψει το άθροισμά τους. Τώρα αυτό στον προγραμματισμό πώς θα γίνει; Οι τιμές στη συνάρτησή μας, θα μπουν σε δύο μεταβλητές που θα χρησιμοποιεί η ίδια, τις num1 και num2. Το τι τιμές θα δέχεται η μεταβλητή μας, το ορίζουμε στην παρένθεσή της, με δύο ορισμούς:
Byval num1 As Integer και Byval num2 As Integer

(Byval num1 As Integer, Byval num2 As Integer)

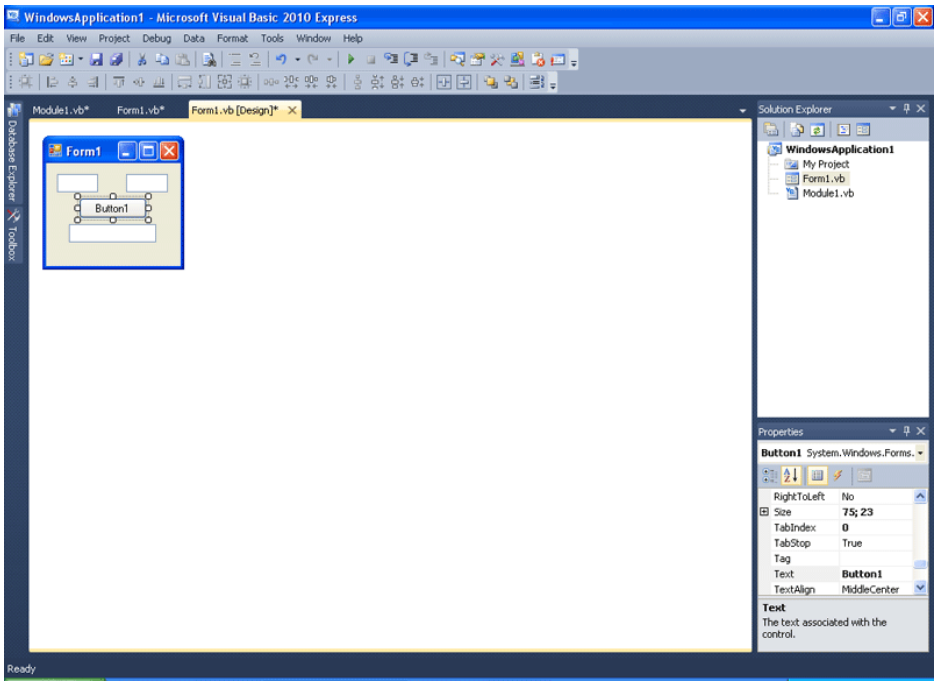
Κοινώς λέμε, θα δεχτείς την πρώτη τιμή και θα την τοποθετήσεις στη μεταβλητή num1 και θα δεχτείς και μια δεύτερη τιμή και θα την τοποθετήσεις στη μεταβλητή num2.

Οπότε τώρα καταλαβαίνουμε τι ακριβώς εννοούσαμε με τη γραμμή κώδικα
Return num1 + num2

Ωστόσο ο τρόπος που θα χρησιμοποιηθεί η συνάρτηση εξηγεί καλύτερα τη λειτουργία αυτή.



Κάντε διπλό κλικ στο Button1 στο Form[Design].



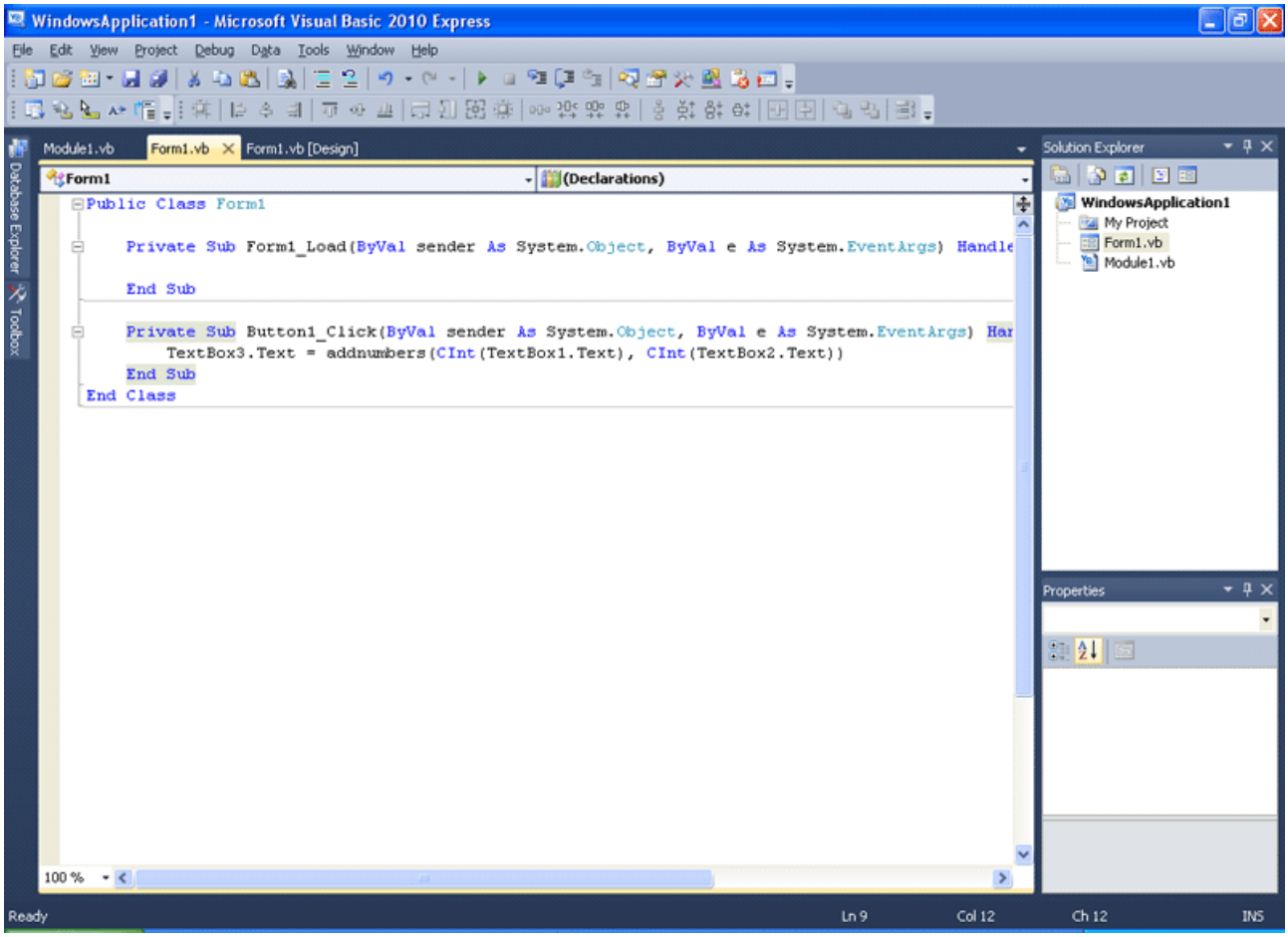
Γράψτε την εντολή:

TextBox3.Text = addnumbers(CINT(TextBox1.Text),CINT(TextBox2.Text))

Τι λέμε εδώ;
Το κείμενο του TextBox3 θα πάρει ως νέα τιμή, την τιμή που θα επιστρέψει η συνάρτηση addnumbers, στην οποία ως πρώτη τιμή δίνουμε το κείμενο του TextBox1 αφού γίνει ακέραιος και ως δεύτερη τιμή, δίνουμε το κείμενο του TextBox2 αφού γίνει ακέραιος.

Το CINT(TextBox1.Text) θα μπει στο num1
και το CINT(TextBox2.Text) θα μπεί στο num2

Το άθροισμά τους επιστρέφεται και αποθηκεύεται ως νέα τιμή του TextBox3.Text
διότι αυτό ακριβώς κάνει η Function addnumbers: Return num1 + num2



Τρέξτε το πρόγραμμα και βάλτε δύο ακεραίους. Πατώντας το κουμπί, θα σας δώσει στο TextBox3 το άθροισμά τους.
Το πρόγραμμα βέβαια, δεν είναι καθόλου σταθερό. Δεν έχουμε καν κάνει έλεγχο για το αν ο χρήστης θα βάλει κείμενο ή πολύ μεγάλο αριθμό στα textbox. Στόχο εδώ είχαμε να εξηγήσουμε τη λειτουργία των συναρτήσεων άλλωστε. Ως άσκηση, προσπαθήστε να κάνετε το πρόγραμμα, να ελέγχει το τι του τοποθετεί ο χρήστης, έτσι ώστε να γίνει σταθερό. Έχετε ήδη τη γνώση να το κάνετε, από τα προηγούμενα κεφάλαια. Εδώ κάπου, πρέπει να σας αναφέρουμε ένα τελευταίο θέμα.

Υπάρχουν διαδικασίες, συναρτήσεις και μέρη κώδικα, που όταν τρέχουν, δημιουργούν μεταβλητές. Αφού τελειώσουν την εκτέλεσή τους, οι μεταβλητές αυτές χάνονται. Κοινώς, σε μια διαδικασία, αφότου εκτελεστεί ο κώδικάς της, κάθε μεταβλητή που περιέχει, κι έχει οριστεί μέσα της, χάνεται. Για να μη συμβεί αυτό, θα πρέπει όταν την ορίζουμε να χρησιμοποιούμε τον όρο Static. Static varname As Integer.

Ορίζουμε ως Static, μια μεταβλητή τύπου Integer, με ονομασία varname. Με αυτό τον τρόπο η μεταβλητή διατηρείται στη μνήμη. Πιθανόν αυτό να σας ακούγεται λίγο ακατανόητο, ωστόσο όταν θα αρχίσετε να εμβαθύνετε στον προγραμματισμό, θα κατανοήσετε πολύ καλά τι εννοούμε. Για την ώρα να θυμάστε, πως οι μεταβλητές δημιουργούνται και κρατάνε χώρο στη μνήμη. Τα προγράμματα στη Visual Basic, όταν καλούν κώδικα που δημιουργεί μεταβλητές, δημιουργούν και τις μεταβλητές αυτές. Με τη λήξη αυτού του κώδικα όμως, απελευθερώνουν τη μνήμη απο τυχόν μεταβλητές που δημιούργησε ο κώδικας. Αυτό πολύ σωστά γίνεται, μιας και τέτοιες μεταβλητές, συνήθως δε χρειάζονται πια, κι άδικα πιάνουν χώρο στην μνήμη. Οπότε με τη Static δήλωση, εμποδίζετε να συμβεί αυτό, στη συγκεκριμένη μεταβλητή. Η Static δε φεύγει. μένει στη μνήμη μαζί με το περιεχόμενό της.

Εδώ κάπου τελειώνει το κεφάλαιο αυτό. Ελπίζουμε να μπορέσαμε να σας δώσουμε να καταλάβετε τις έννοιες που καταπιαστήκαμε, διότι είναι πολύ σπουδαίο να τις έχετε ξεκάθαρες στο μυαλό σας.

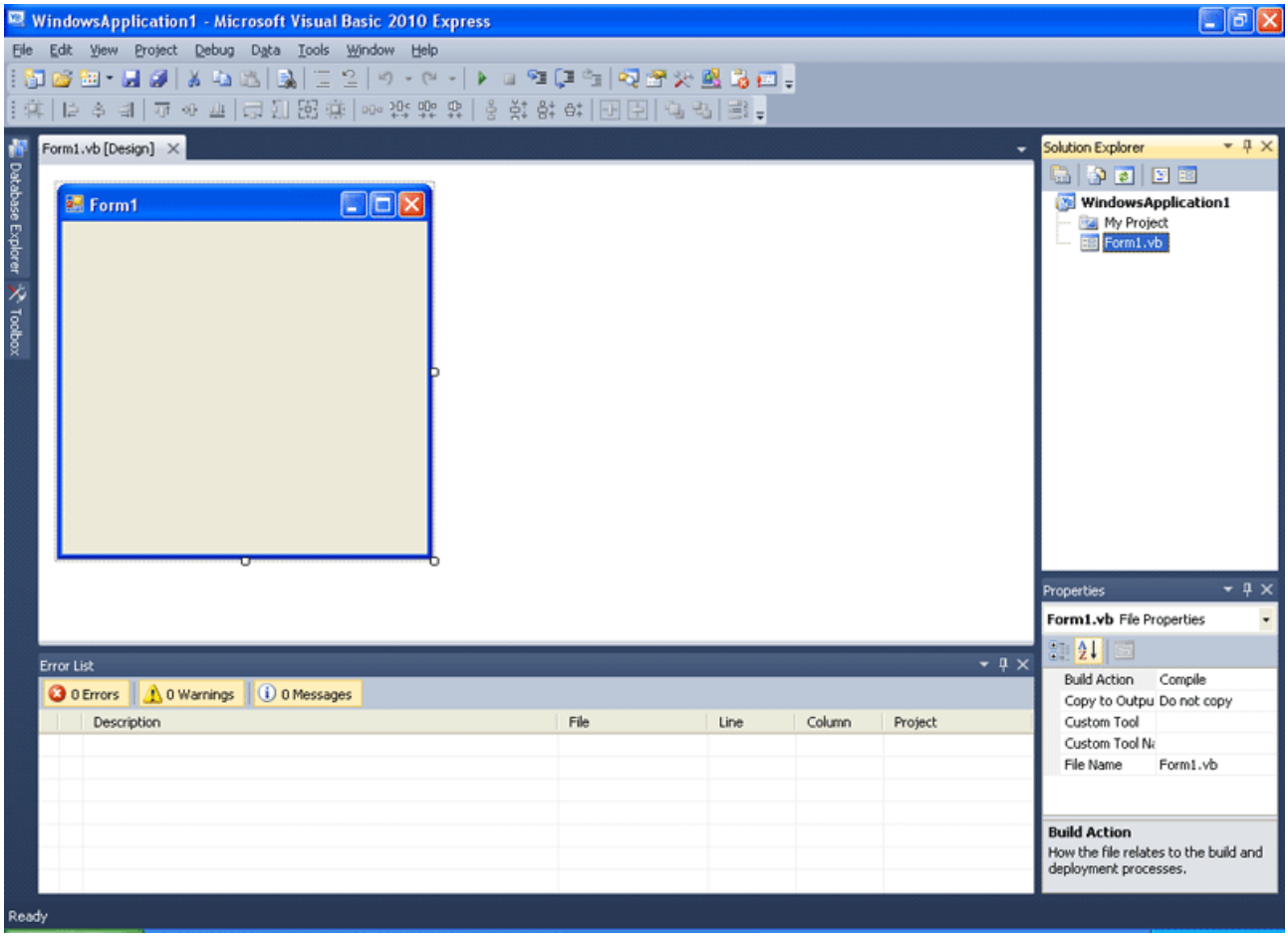
ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Σε αυτό το κεφάλαιο αναλύεται την έννοια της κλάσης ενώ εξοικειώνεται ο χρήστης με τις λειτουργικές μονάδες κώδικα. Χρησιμοποιήθηκαν πολλά παραδείγματα για το τι είναι κλάση, διότι θεωρήθηκε απαραίτητο να δοθεί έμφαση στο να κατανοηθεί ο όρος πάση θυσία. Κατόπιν αναλύθηκε και η λειτουργική μονάδα κώδικα, με επεξήγηση των κοινόχρηστων συναρτήσεων.

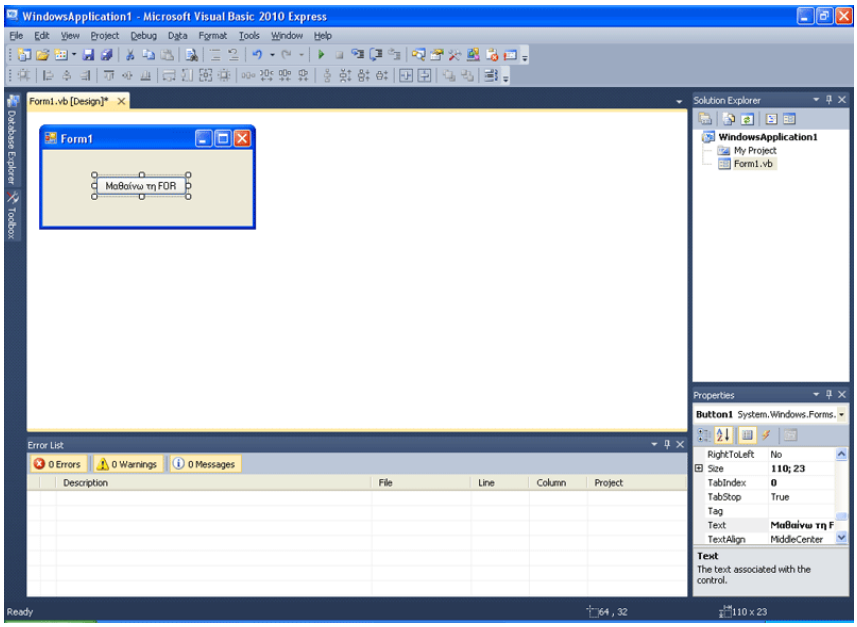
ΚΕΦΑΛΑΙΟ 13
ΕΠΑΝΑΛΗΨΕΙΣ ΜΕ ΤΗ FOR ΚΙ ΑΠΟΦΑΣΕΙΣ ΜΕ ΤΗΝ CASE

Το παρόν αλλά και το επόμενο κεφάλαιο, στόχο έχουν να σας εξηγήσουν και να σας εισάγουν ομαλά, σε νέα προγραμματιστικά εργαλεία. Στο παρόν κεφάλαιο θα ασχοληθούμε με την CASE και την FOR, δύο εντολές που σε κάποιες περιπτώσεις είναι τρομερά χρήσιμες. Σας τις εξηγούμε προκαταβολικά δε, διότι στη συνέχεια θα πρέπει να σας παρουσιάσουμε περισσότερα controls της Visual Basic .Net τα οποία όμως θα μπορείτε να χρησιμοποιήσετε, μόνο αν γνωρίζετε κάποιες από τις εντολές αυτές. Δε νομίζουμε να φανταστήκατε άλλωστε, πως το μόνο που θα μαθαίνατε εδώ, ήταν τα Buttons, Textbox, και Labels! Ας δούμε λοιπόν πρώτα τη FOR.

Φτιάξτε ένα νέο project, όπως ήδη γνωρίζετε.

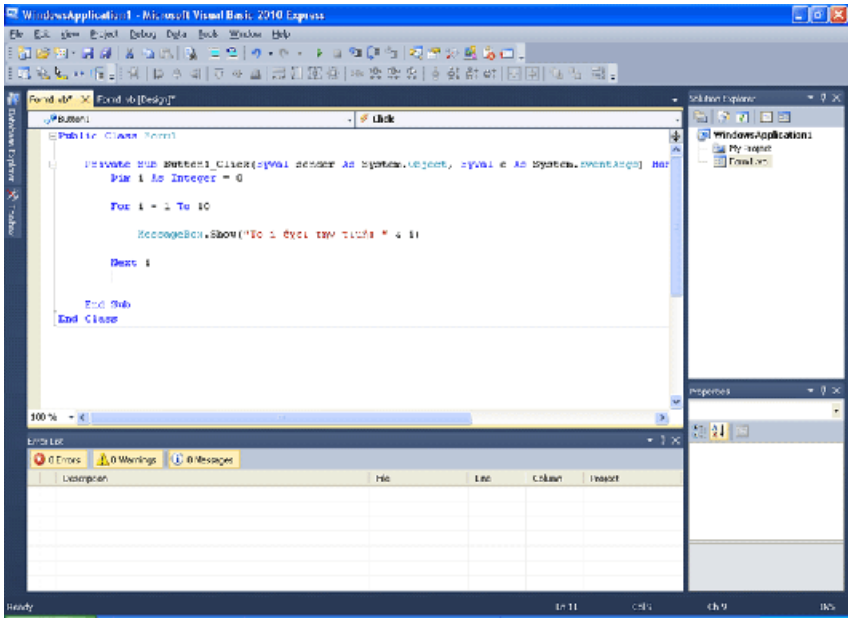


Μικραίνουμε τη φόρμα και φτιάχνουμε ένα Button.
Η ιδιότητα Text του Button θα περιέχει το κείμενο "Μαθαίνω τη FOR"



Κάντε διπλό κλικ στο Button1 και γράψτε μέσα στο Private Sub Button1_Click, τον παρακάτω κώδικα, που εκτενώς στη συνέχεια θα σας εξηγήσουμε τι κάνει.

```
Dim i As Integer = 0
 For i = 1 To 10
   MessageBox.Show("Το i έχει την τιμή: " & i)
 Next i
```



Τι έχουμε εδώ; Ουσιαστικά τρεις γραμμές κώδικα. Σε τι χρησιμεύει όμως η For; Τι κάνει; Λοιπόν, φανταστείτε πως είσαστε δρομέας στίβου. Ο προπονητής σας λοιπόν έχει μια παραξενιά. Φανταστείτε να σας δίνει την εντολή, "Κάνε ένα γύρο το γήπεδο" δέκα φορές. Δηλαδή, σας λέει "Κάνε ένα γύρο το γήπεδο", εσείς να κάνετε το γύρο, μετά έρχεται πάλι και λέει "Κάνε ένα γύρο το γήπεδο" κ.ο.κ. για δέκα φορές. Φανταστείτε ότι το κάνει για κάθε

αθλήτρια κι αθλητή. Θα μάλλιαζε η γλώσσα σας, αν ήσασταν στη θέση του σωστά. Είναι λογικό αν στο μυαλό του έχει τη θέληση να κάνετε δέκα γύρους το γήπεδο, να σας πει, "Κάνε δέκα φορές το γύρο του γηπέδου". Έτσι δεν είναι;

Ακριβώς αυτό κάνει η For. Πάμε να δούμε τον κώδικα.

Dim i As Integer = 0
Ορίζουμε μία μεταβλητή με όνομα "i" τύπου ακέραίου.
Της δίνουμε κι αρχική τιμή το μηδέν στην ίδια εντολή που την ορίζουμε.

MessageBox.Show("Το i έχει την τιμή: " & i)
Αυτή η εντολή εμφανίζει ένα μικρό παράθυρο με ένα μήνυμα κι ένα κουμπί OK.
Το μήνυμα είναι το εξής:
Εμφανίζει τη φράση:

"Το i έχει την τιμή: "

και με τον ειδικό τελεστή "&" εμφανίζει δίπλα από τη φράση την τιμή του i.

ΠΑΡΑΤΗΡΗΣΗ:
Εντός των εισαγωγικών, το i είναι χαρακτήρας κειμένου.
Εκτός των εισαγωγικών όμως, το i είναι σαφώς ισοδύναμο με την τρέχουσα τιμή του.

For i = 1 To 10
Next i
Τι λέει αυτή η δομή επανάληψης λοιπόν; Για τιμές του i που αρχίζουν από 1, μέχρι ΚΑΙ το 10 κάνε ότι βρίσκεται μεταξύ της FOR και του Next i. ενώ με το που φτάνεις στο Next i το i αυξάνεται κατά 1

Τρέξτε το πρόγραμμα. Σας εμφανίζεται ένα MessageBox που σας λέει την τρέχουσα τιμή του "μετρητή" i.
Για τιμές από 1 ως 10, οτιδήποτε περιέχεται μέσα στη For θα επαναλαμβάνεται. Με το που η τιμή γίνει 10 ξανατρέχει ο κώδικας που περιλαμβάνει η For κι έπειτα το Next i δίνει στο i την τιμή 11.

Οπότε επειδή η For λέει εδώ στο παράδειγμά μας:
Από 1 ως 10 (For i = 1 To 10) το πρόγραμμα συνεχίζεται στην επόμενη γραμμή κώδικα έξω από τη For μετά το Next i επειδή το i είναι 11 και άρα μεγαλύτερο του 10! Στο παράδειγμα δεν έχουμε όμως άλλες εντολές μετά τη For. Οπότε σταματάει και το πρόγραμμα.

Γιατί όμως στο Next i, το i αυξάνεται κατά 1; Το i αυξάνεται κατά 1 διότι δεν ορίσαμε εμείς κάτι διαφορετικό. Εξ ορισμού λοιπόν η γλώσσα δίνει τιμή αύξησης κατά 1.

Κι αν θέλω να αυξάνεται το i κατά 2; Η εντολή For παίρνει ένα όρισμα τότε, το STEP.
Το "βήμα" STEP ορίζει πόσο θα προσαυξάνεται το i σε κάθε επανάληψη.

Αλλάξτε τον κώδικα του προγράμματος ώς εξής:
For i = 1 To 10 Step 2
Αυτό ήταν. Τρέξτε το πρόγραμμα. 1, 3, 5, 7, 9!
Ξεκινάει από 1, αυξάνεται κατά 2, μεγαλύτερο του 10 σταματάει!

Κι αν ήθελα να είναι μέχρι το 100 οι επαναλήψεις;
For i = 1 To 100 Step 2
Απλό! Αυτό μη το δοκιμάσετε, θα βαρεθείτε να κάνετε κλικ. Υπάρχουν κίνδυνοι και για τενοντίτιδα άλλωστε από επαναλαμβανόμενες κινήσεις των δακτύλων.

Κι αν ήθελα να ξεκινάει από το 22 και να πηγαίνει ως το 100 αυξανόμενο κατά 7;
For i = 22 To 100 Step 7

Τέλος, αν δεν ήθελα να αυξάνει κατά ένα από το 1 στο 10 αλλά να μειώνει από το 10 στο 1 κατά 2 τι θα έκανα;
For i = 10 To 1 Step -2

Προσέχετε; το STEP μπορεί να είναι αρνητικός αριθμός, αλλά σε αυτή την περίπτωση η διαδικασία είναι αντίστροφη.

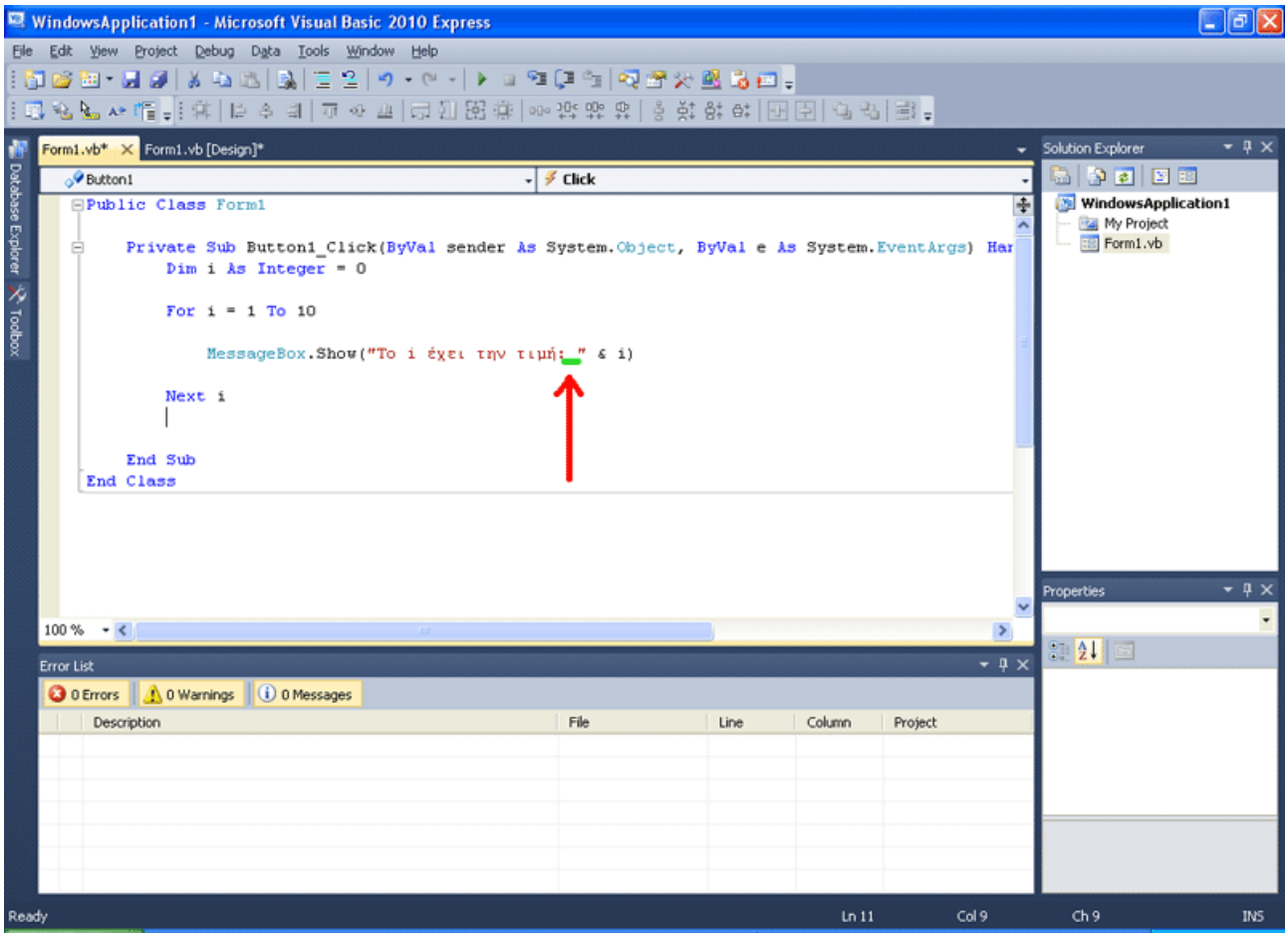
Συνοψίζοντας, είναι προτιμότερο αντί να γράφουμε κάτι τέτοιο...

```
  Dim i As Integer = 0
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
  i+=1
  MessageBox.Show("To i έχει την τιμή: " & i)
```

να χρησιμοποιούμε τη δομή επανάληψης For

```
Dim i As Integer = 0
 For i = 1 To 10
  MessageBox.Show("To i έχει την τιμή: " & i)
 Next i
```

Κοιτάξτε τώρα την παρακάτω εικόνα!



Με ένα κόκκινο βέλος, σας δείχνουμε μια πράσινη υπογράμμιση που κάναμε. Τι δείχνουμε εδώ; Ο τελεστής & προσθέτει το αλφαριθμητικό "Το i έχει την τιμή: ", μαζί με το περιεχόμενο της μεταβλητής i και τα αποδίδει ως κείμενο σε ένα MessageBox. Το κείμενο περιέχει ένα κενό μετά την άνω και κάτω τελεία. Γιατί όμως; Ο λόγος είναι για να μην εμφανιστεί άσχημα, κολλημένο στην άνω και κάτω τελεία, το περιεχόμενο του i, το οποίο είναι αριθμός. Βάζουμε ένα κενό, για να είναι πιο ευανάγνωστο από το χρήστη δηλαδή!

Επαναφέρετε τον κώδικα όπως ήταν στο αρχικό παράδειγμα, δηλαδή:

```
Dim i As Integer = 0
 For i = 1 To 10 Step 2
  MessageBox.Show("To i έχει την τιμή: " & i)
 Next i
```

κι από κάτω, αλλά εντός τών Private Sub Button1_Click() και End Sub γράψτε τον παρακάτω κώδικα, όπως φαίνεται και στην εικόνα:

```
For j = 1 To 5 Step 2
 MessageBox.Show(j)
Next j
```

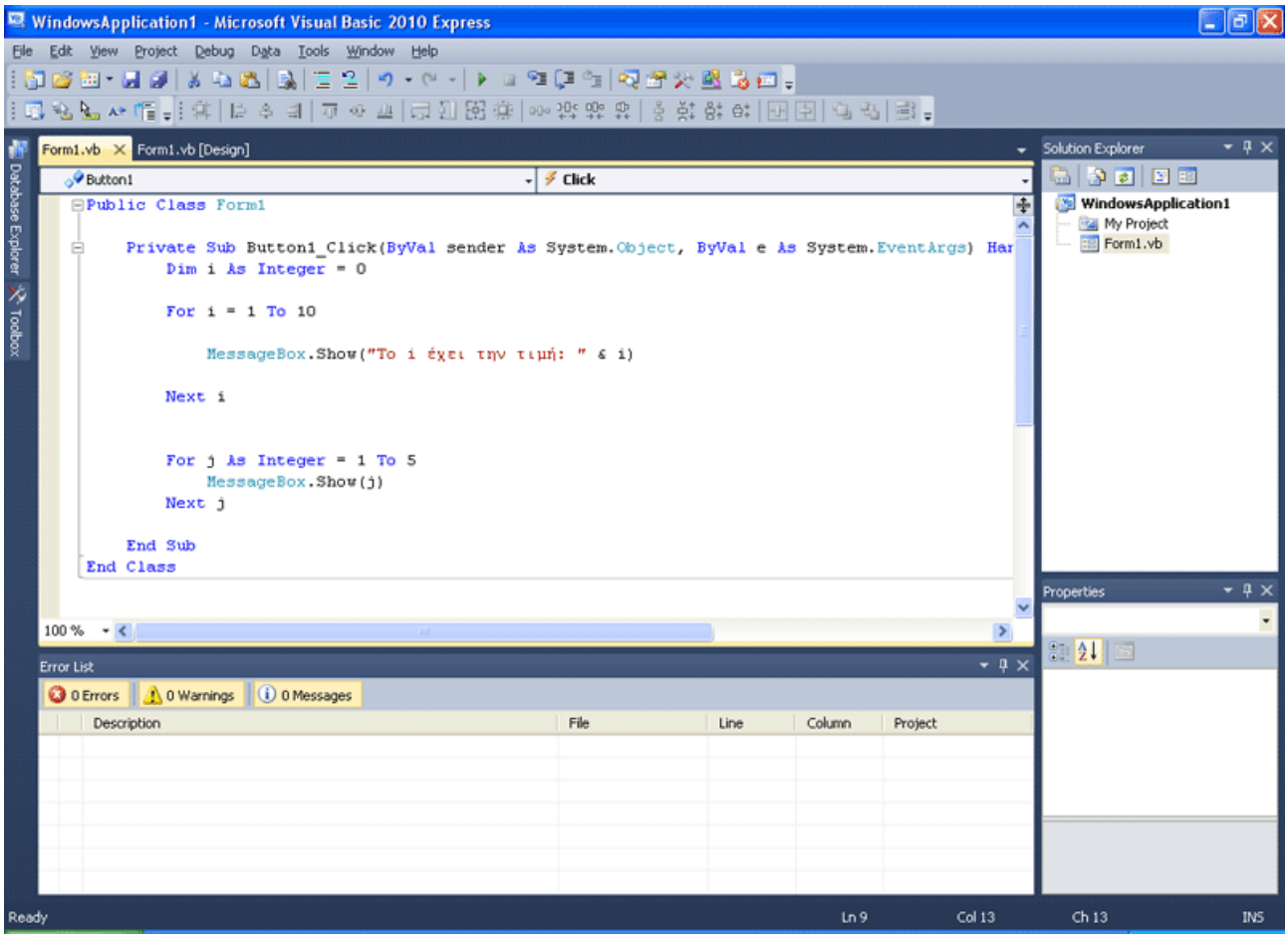
Αν πάτε να το τρέξετε θα σας εμφανίσει λάθος. Γιατί όμως;
Εκεί που ορίσατε το Dim i As Integer = 0 πρέπει να πάτε από κάτω του και να ορίσετε τη j μεταβλητή.
Dim j As Integer = 0
Προσέχετε! Τα i και j είναι μεταβλητές. Χρησιμοποιούνται ως μετρητές, αλλά δεν παύουν, ως μεταβλητές, να πρέπει να οριστούν με ανάλογη εντολή. Ο σωστός κώδικας λοιπόν είναι:

```
Dim i As Integer = 0
 Dim j As Integer = 0
   For i = 1 To 10 Step 2
    MessageBox.Show("Το i έχει την τιμή: " & i)
   Next i
     For j = 1 To 5 Step 2
       MessageBox.Show(j)
     Next j
```

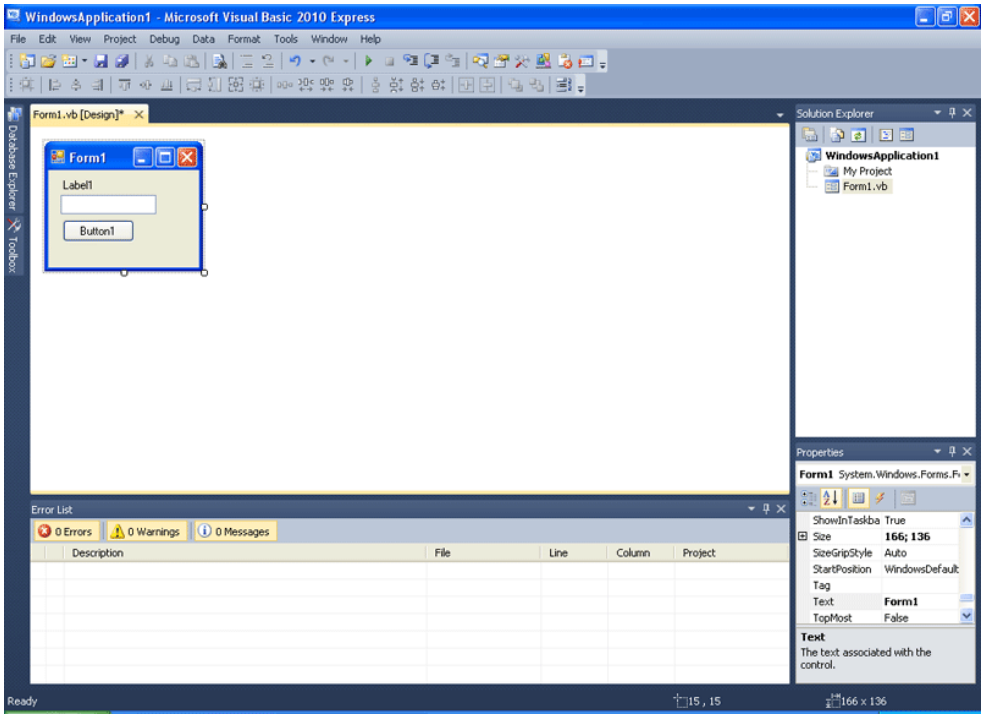
Τρέξτε το. Τι παρατηρείτε; ΠΡΩΤΑ τελειώνει η FOR i ΚΑΙ ΜΕΤΑ μπαίνουμε στη FOR j.

Υπάρχει εναλλακτική στη δήλωση μεταβλητής σε for;

Δοκιμάστε τον κώδικα της εικόνας



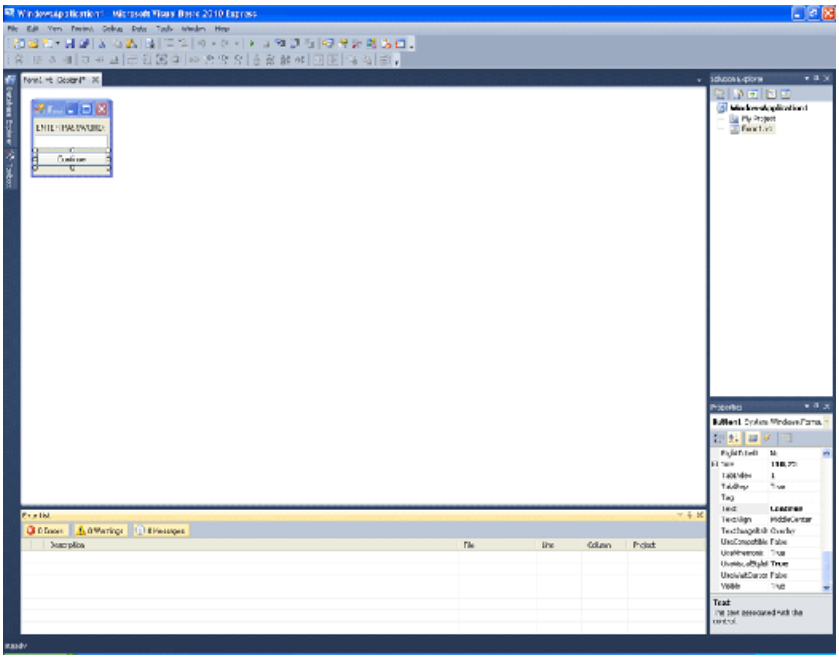
Τελευταίο ερώτημα που πρέπει να απαντηθεί για τη For, προτού προχωρήσουμε. Μπορεί να εκτελεστεί μια For μέσα σε μία άλλη; Η απάντηση είναι ένα τρανό ναι! Η For ενθυλακώνετε και μάλιστα είναι ο βασικός τρόπος διαχείρισης της δομής δεδομένων που αποκαλείται "πίνακας" και θα αναλυθεί σε μεταγενέστερο κεφάλαιο. Καιρός λοιπόν να δούμε μια δομή απόφασης τώρα. Μοιάζει κάπως με την If. Ωστόσο έχει ιδιαίτερη πρακτική εφαρμογή. Αναφερόμαστε στη SELECT CASE σαφώς!
Σβήστε το project που μελετούσαμε τη FOR
Δημιουργήστε ένα νέο.
Βάλτε στη φόρμα μια Label ένα TextBox κι ένα Button.
Στοιχίστε τα όπως στην παρακάτω εικόνα.



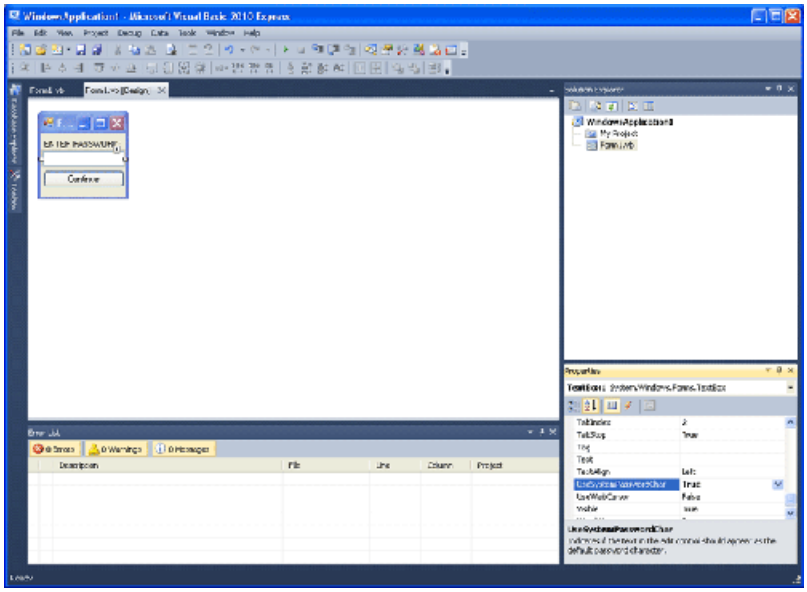
Το text της Label1 κάντε το "ENTER PASSWORD"
Το text του Button1 κάντε το "Continue" Φτιάξτε τη φόρμα και τα αντικείμενα ομοιόμορφα, όπως βλέπετε στην παρακάτω εικόνα. Φτιάχνουμε ένα πρόγραμμα, το οποίο θα δέχεται ένα συνθηματικό. Ανάλογα με το τι του δίνουμε, θα αποκρίνεται με αντίστοιχο τρόπο. Υπάρχουν μόνο δύο σωστά συνθηματικά, για δύο διαφορετικούς χρήστες. Οι χρήστες αυτοί δε μπορούν να αλλάξουν τον κωδικό τους. Είναι πάγιος κωδικός κοινώς.

Το συγκεκριμένο παράδειγμα, όπως καταλαβαίνετε δεν έχει στόχο να διδάξει ή να δείξει ένα τρόπο να προστατέψετε το πρόγραμμά σας από ανεπιθύμητη πρόσβαση.

Θέλουμε να δείξουμε τη SELECT CASE, κι όχι να κάνουμε μαθήματα ασφάλειας λογισμικού και κρυπτογράφησης. Απλά το παραθέτουμε σαν παράδειγμα, για να εξυπηρετήσει το στόχο μας, και να σας δείξουμε και κάτι νέο. Ωστόσο πρέπει να αναφέρουμε σαφώς, ότι η μέθοδος αυτή δεν προστατεύει κανένα πρόγραμμα και μπορεί να νικηθεί ΠΟΛΥ εύκολα.



Το νέο πράγμα που θέλουμε να σας δείξουμε λοιπόν είναι το εξής:
Πηγαίνετε και κάντε κλικ στο Textbox1.
Πηγαίνετε κάτω δεξιά στα Properties.
Βρείτε το UseSystemPasswordChar και δώστε του την τιμή True.
Πλέον ο χρήστης όταν πληκτρολογεί κάτι στο textbox θα εμφανίζονται κουκίδες όπως γίνεται σε κάθε textbox για κωδικό.

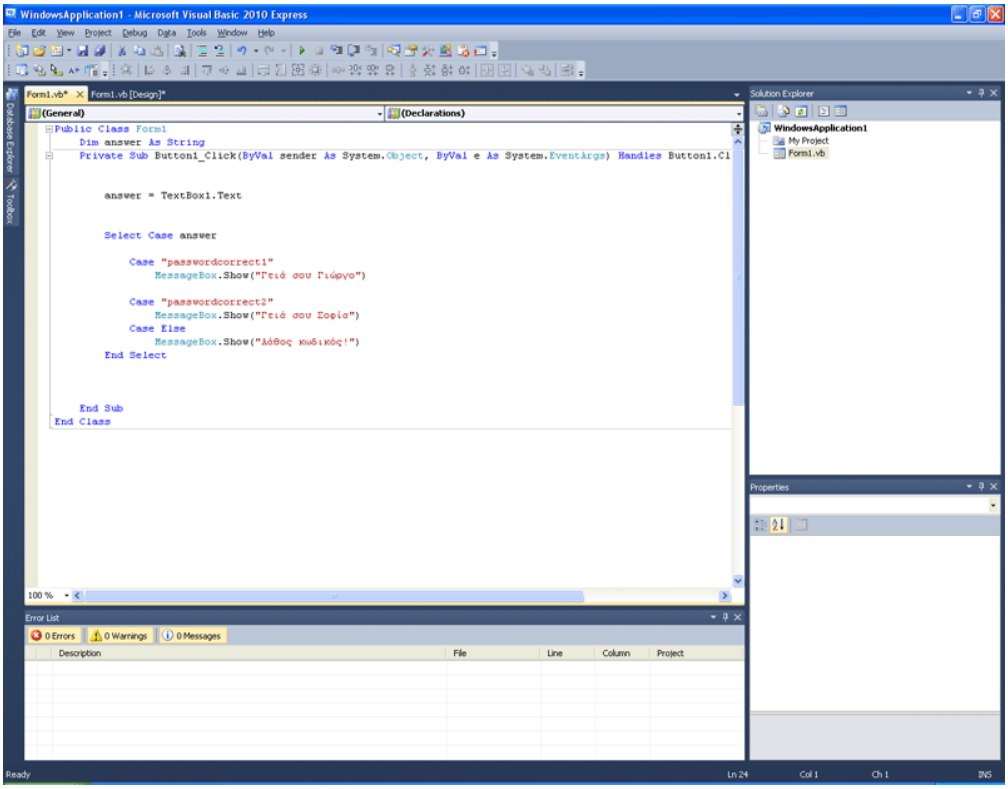


Ο κώδικας της Class Form1 και του Button1 ακολουθούν:

```
Public Class Form1
 Dim answer As String

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
 Handles Button1.Click
```

```
        answer = TextBox1.Text

        Select Case answer

            Case "passwordcorrect1"
                MessageBox.Show("Γειά σου Γιώργο")

            Case "passwordcorrect2"
                MessageBox.Show("Γειά σου Σοφία")

            Case Else
                MessageBox.Show("Λάθος κωδικός!")

        End Select

    End Sub

End Class
```

Στην παρακάτω εικόνα φαίνεται έτοιμο



Επεξηγούμε τώρα τον κώδικα.
Μέσα στη φόρμα ορίζουμε τη μεταβλητή answer.
Θέλουμε αν δώσει ο χρήστης "passwordcorrect1" να αποκριθεί το πρόγραμμα βγάζοντας ένα MessageBox με το μήνυμα "Γεια σου Γιώργο"
Θέλουμε αν δώσει ο χρήστης "passwordcorrect2" να αποκριθεί το πρόγραμμα βγάζοντας ένα MessageBox με το μήνυμα "Γεια σου Σοφία"
Θέλουμε αν δώσει ο χρήστης κάτι άλλο να αποκριθεί το πρόγραμμα βγάζοντας ένα

MessageBox με το μήνυμα "Λάθος κωδικός!"

Μπορεί να γίνει με την εντολή If;
Ναι. Αλλά θα είναι πιο περίπλοκο και πιο δύσκολο.

Συνήθως όταν έχουμε διακριτές περιπτώσεις, στο τι αποκρίσεις θέλουμε να δώσει το πρόγραμμα και ιδιαίτερα, διακριτές περιπτώσεις εισαγωγής δεδομένων από το χρήστη, αντί για την If μπορούμε να χρησιμοποιήσουμε τη SELECT CASE.

answer = TextBox1.Text
Η answer παίρνει ως τιμή, ότι έγραψε ο χρήστης.

Select Case answer
Ανάλογα με το τι περιέχεται στη μεταβλητή answer κάνε κάτι κατά περίπτωση.

Case "passwordcorrect1"
MessageBox.Show("Γειά σου Γιώργο")
Στην περίπτωση που ο χρήστης εισήγαγε "passwordcorrect1" εμφάνισέ του το MessageBox με μήνυμα "Γεια σου Γιώργο".

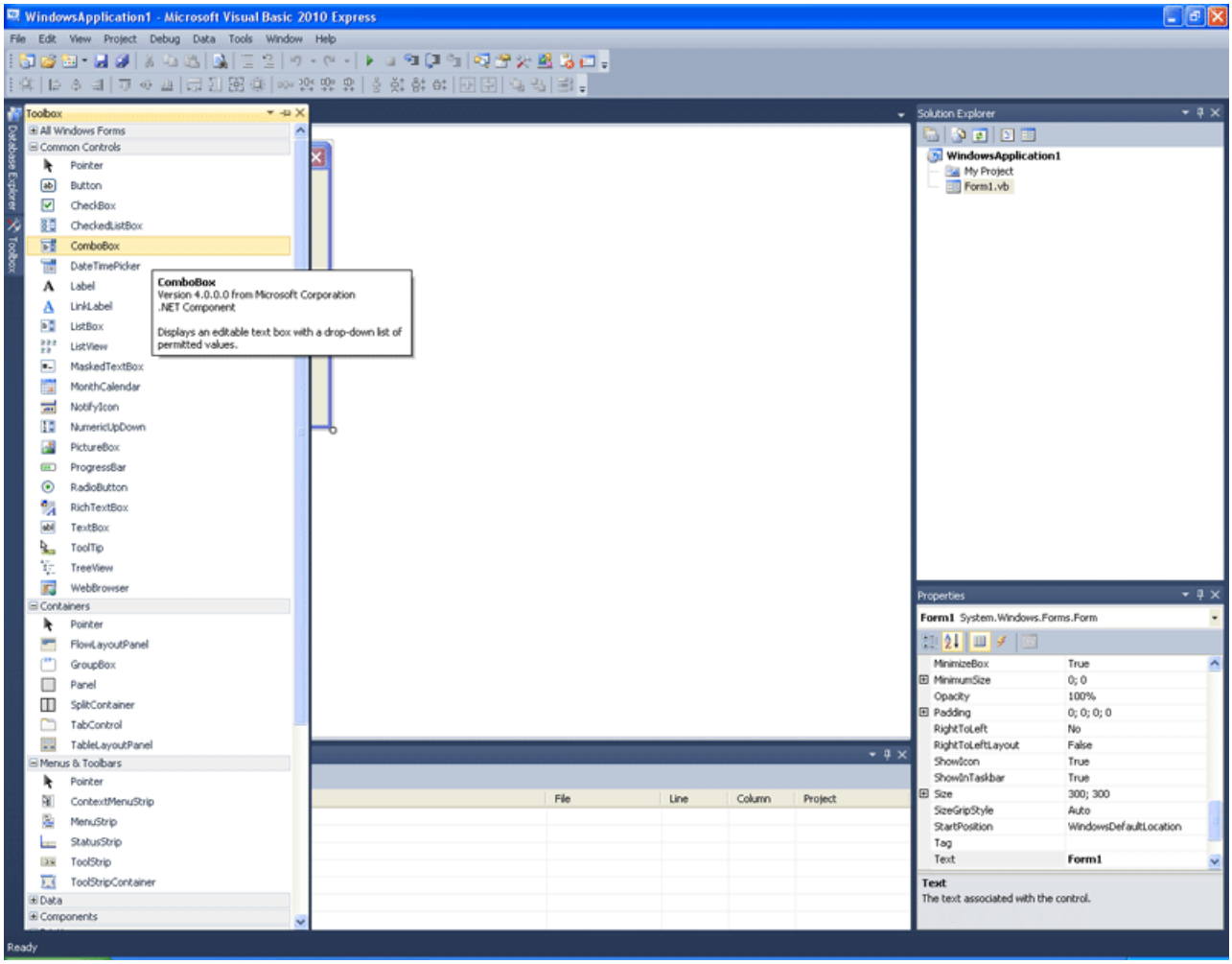
Case "passwordcorrect2"
MessageBox.Show("Γειά σου Σοφία")
Στην περίπτωση που ο χρήστης εισήγαγε "passwordcorrect2" εμφάνισέ του το MessageBox με μήνυμα "Γεια σου Σοφία".

Case Else
MessageBox.Show("Λάθος κωδικός!")
Σε κάθε άλλη περίπτωση εμφάνισε ένα MessageBox με το μήνυμα "Λάθος κωδικός!"

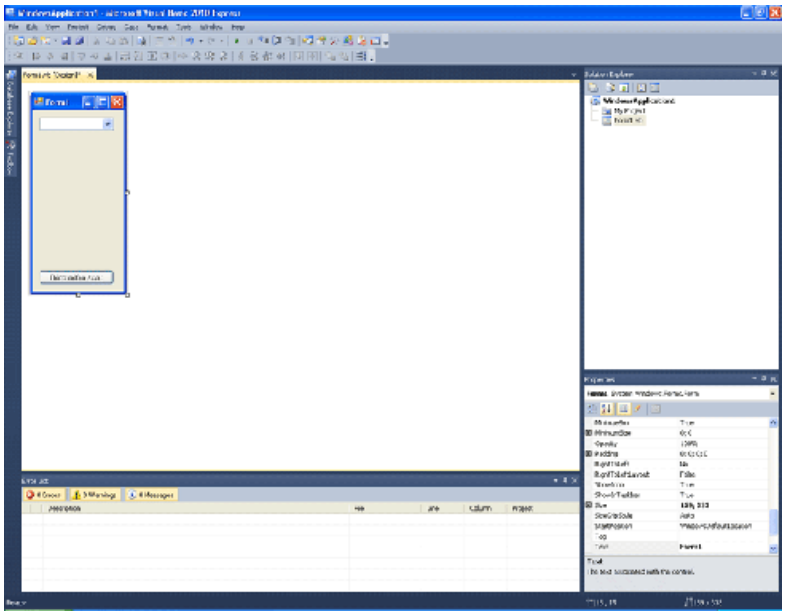
End Select
Εδώ τελειώνει η SELECT CASE.
Είναι το αντίστοιχο του End If στην If

Πότε όμως ΔΕ μπορείτε να χρησιμοποιείτε SELECT CASE και μόνη λύση είναι η IF;
Έχετε κατά νου ότι την SELECT CASE τη χρησιμοποιείτε μόνο όταν κάνει ευκολότερη τη ζωή σας. Οπότε αν με την IF και λογικούς τελεστές ο κώδικας γίνεται απλούστερος, ακολουθείτε δομή IF.

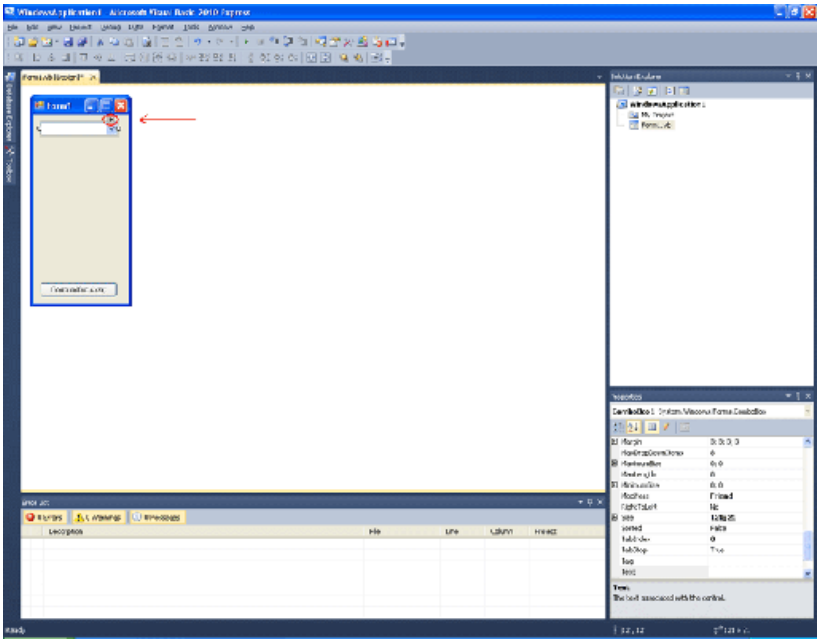
Κοιτάξτε όμως πότε θα μπορούσε να είναι χρήσιμη μια δομή SELECT CASE!
Φτιάξτε ένα νέο project.
Επιλέξτε από το toolbox, από το menu controls ένα ComboBox. Το ComboBox είναι TextBox που μπορούμε να του βάλουμε μια λίστα με προεπιλεγμένες απαντήσεις. ListBox θα ήταν απλώς μια λίστα. Το ComboBox (από το combination = συνδυασμός) συνδυάζει ένα TextBox που γράφουμε ό,τι θέλουμε, με ένα ListBox, που έχει προεπιλεγμένες, πεπερασμένες επιλογές.



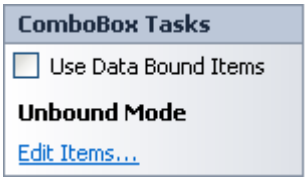
Επιλέξαμε λοιπόν το ComboBox για δύο λόγους. Για να δείτε ουσιαστικά τη λειτουργία του ListBox, μιας και δε διαφέρουν πολύ και συνάμα να έχει νόημα η περίπτωση CASE ELSE. Θα δείτε τι εννοούμε στη συνέχεια.
Φτιάξτε ένα Button και ως Text δώστε του το μήνυμα: "Πόσα πόδια έχει;"
Φτιάξτε μακρόστενη τη φόρμα, όπως βλέπετε στην εικόνα.
Το πρόγραμμα θα απαντάει, ανάλογα με το τι θα επιλέγουμε στο ComboBox.



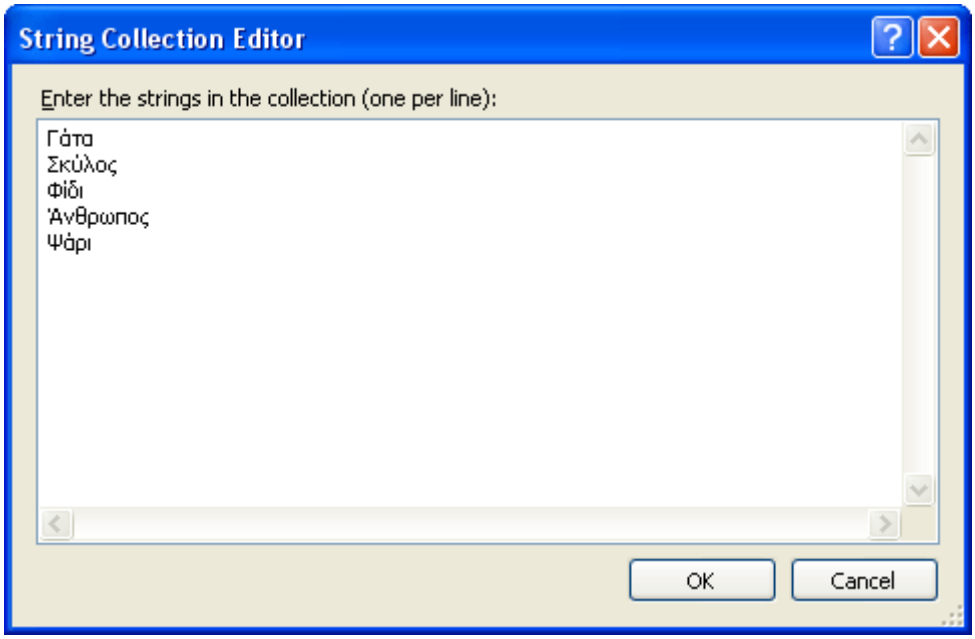
Κάντε κλικ στο ComboBox. Εμφανίζεται ένα τετραγωνάκι πάνω δεξιά, με ένα μαύρο τριγωνάκι μέσα του. Το τονίζουμε στη παρακάτω εικόνα με ένα κόκκινο βέλος.
Κάντε κλικ σε αυτό



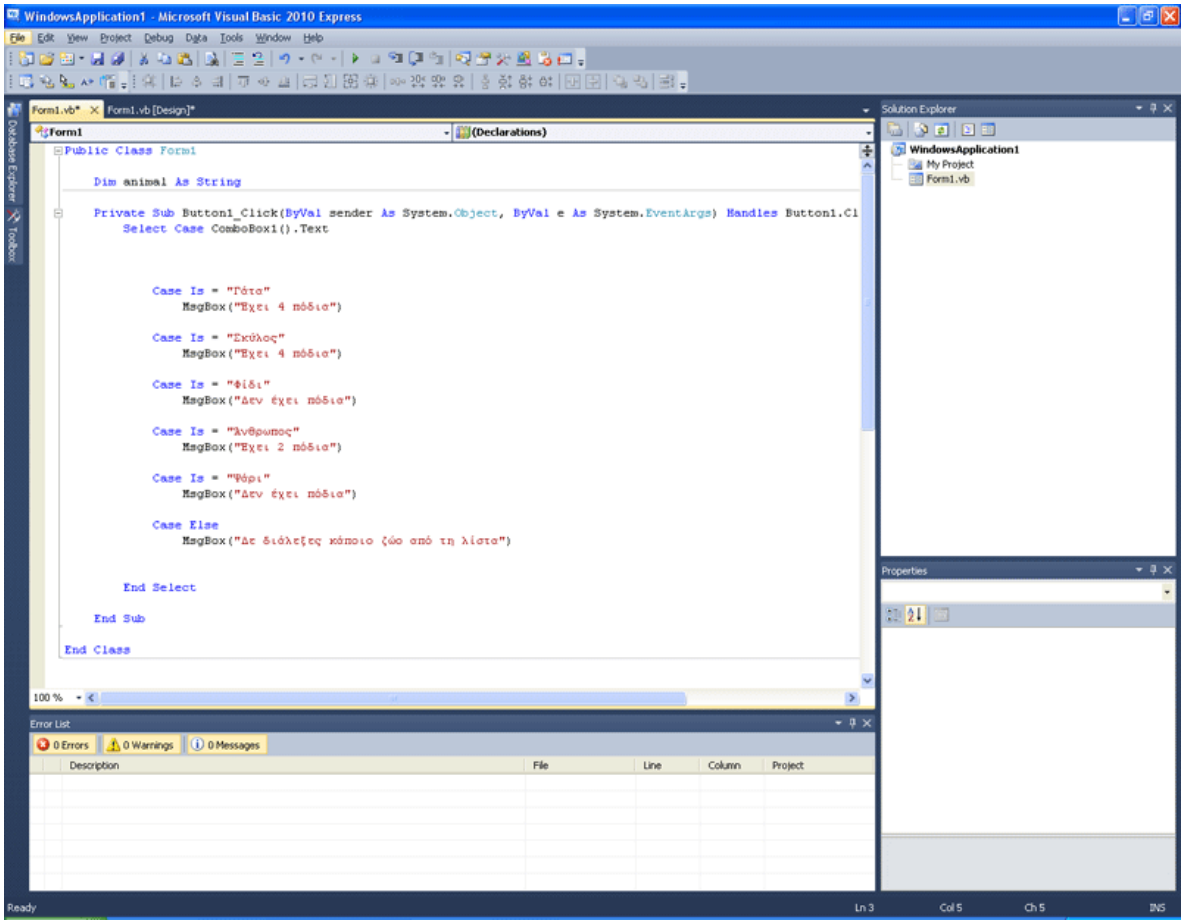
Θα εμφανιστεί αυτό το μενού.
Επιλέγετε Edit Items...



Κι εδώ γράφετε μία μία τις επιλογές, πατώντας κάθε φορά Enter.
Εδώ βάλαμε τις Γάτα, Σκύλος, Φίδι, Άνθρωπος, Ψάρι.



Να και η εικόνα με τον κώδικα!
Θα ακολουθήσει σαφώς εκτενής επεξήγηση



Στην Public Class Form1 προσθέσαμε τη γραμμή κώδικα:
Dim animal As String
Ορίζουμε δηλαδή, τη μεταβλητή με όνομα "animal", ως αλφαριθμητική.

Στο Private Sub Button1_Click() Προσθέτουμε τον κώδικα:

```
 Select Case ComboBox1().Text
   Case IS = "Γάτα"
     MsgBox("Έχει 4 πόδια")
   Case IS = "Σκύλος"
     MsgBox("Έχει 4 πόδια")
   Case IS = "Φίδι"
     MsgBox("Δεν έχει πόδια")
   Case IS = "Άνθρωπος"
     MsgBox("Έχει 2 πόδια")
   Case IS = "Ψάρι"
     MsgBox("Δεν έχει πόδια")
  Case Else
   MsgBox("Δε διάλεξες κάποιο ζώο από τη λίστα")
 End Select
```

Τι κάνουμε εδώ λοιπόν; Select Case ComboBox1().Text Παίρνουμε περιπτώσεις, κατ ευθείαν, ανάλογα το τι είναι το text του ComboBox.
Case IS = "Γάτα"
MsgBox("Έχει 4 πόδια") Αν το Text του ComboBox1 είναι "Γάτα" Ενημερώνουμε με MessageBox το χρήστη, ότι έχει 4 πόδια (το ζώο όχι ο χρήστης!) Ομοίως για κάθε επιλογή.
Αν όμως ο χρήστης γράψει κάτι άλλο;
Case Else
MsgBox("Δε διάλεξες κάποιο ζώο από τη λίστα")

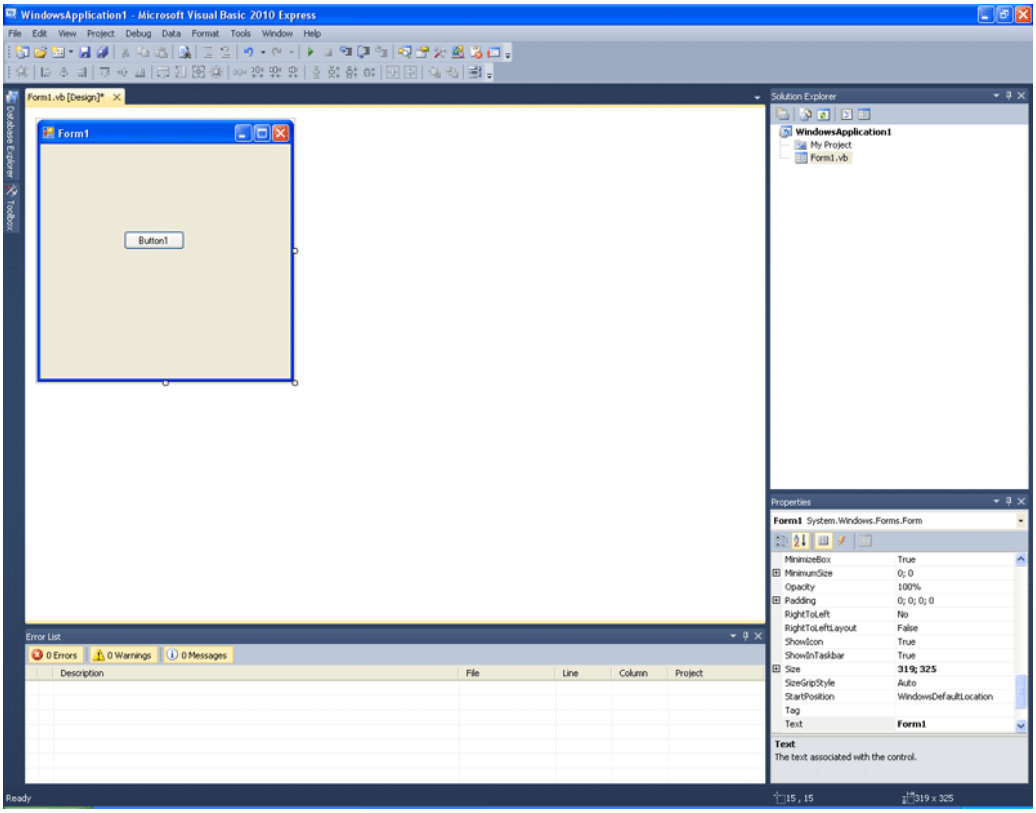
Σε κάθε άλλη περίπτωση, εκτός των επιλογών της λίστας, η απάντηση θα είναι: "Δε διάλεξες κάποιο ζώο από τη λίστα". Τρέξτε το πρόγραμμα και δείτε πως λειτουργεί. Ερώτηση: Πού χρησίμευσε η μεταβλητή animal; Απάντηση: Πουθενά!
Χρησιμοποιήσαμε έναν πολύ άμεσο τρόπο για τη SELECT CASE μιας και δε μας χρησιμεύει κάπου η επιλογή του χρήστη, εκτός του να απαντήσουμε άμεσα στο πόσα πόδια έχει το ζώο. Αν όμως πρέπει να γίνουν μαθηματικές πράξεις, ή πρέπει να αποθηκευτεί έστω και προσωρινά, αυτό που θα εισάγει ο χρήστης, τότε θα πρέπει να αποδοθεί η τιμή του Text του ComboBox στη μεταβλητη και μετά να γίνει η SELECT CASE πάνω στη μεταβλητή, όπως στα προηγούμενα παραδείγματα. Δηλαδή SELECT CASE animals. Εδώ όμως, η γραμμή κώδικα "Dim animal As String" απλώς καταναλώνει μνήμη, δίχως λόγο. Σβήστε τη και θα δείτε πως τίποτα δεν αλλάζει.
Να προσέχετε πόσο οικονομικό σε πόρους μπορεί να γίνει ένα πρόγραμμα.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Σε αυτό το κεφάλαιο αναλύεται η εντολή FOR και η εντολή SELECT CASE με αντίστοιχα παραδείγματα. Πλέον ο μαθητευόμενος πρέπει να αρχίσει να μαθαίνει τα «δομικά εργαλεία» της γλώσσας, έτσι ώστε με αυτά να μπορεί να φτιάξει ένα δικό του πρόγραμμα.

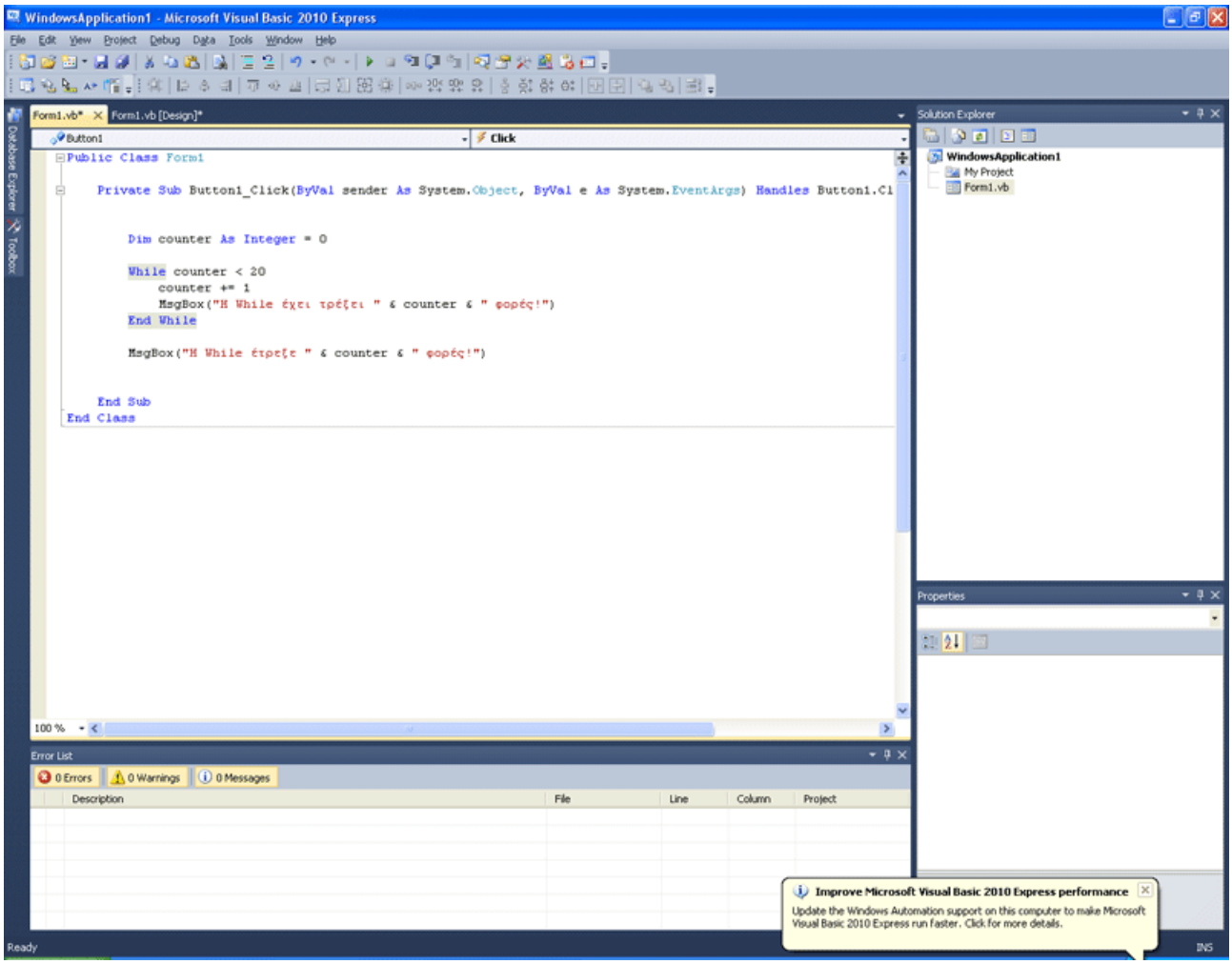
ΚΕΦΑΛΑΙΟ 14
ΕΠΑΝΑΛΗΨΕΙΣ ΜΕ DO LOOP WHILE UNTIL

Καιρός να δούμε μερικές ακόμα δομές επαναλήψεις.
Φτιάξτε ένα νέο Project.
Βάλτε στη φόρμα ένα Button.



Ο κώδικας του Button1 δίνεται παρακάτω!

```
Dim counter As Integer = 0
While counter < 20
    counter += 1
MsgBox("H While έχει τρέξει " & counter  & " φορές!")
End While
MsgBox ("H While έτρεξε " & counter  & " φορές!")
```

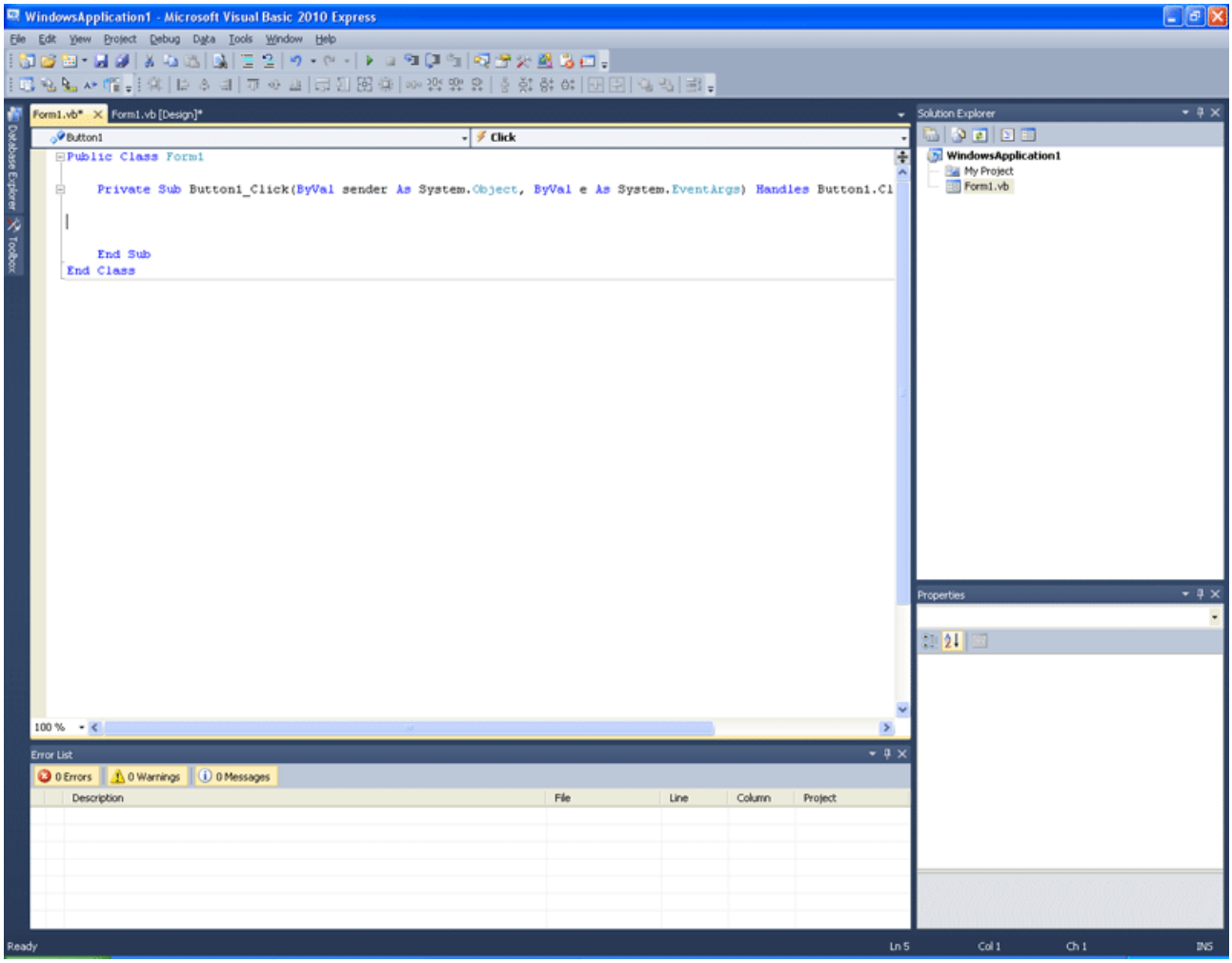


Στην εικόνα βλέπετε τον κώδικα, ενώ κάτω δεξιά η VB μας προτρέπει να κάνουμε κάποιες ρυθμίσεις προς όφελος του λογισμικού.

Πίσω στον κώδικά μας. Καιρός να τον εξηγήσουμε.
Dim counter As Integer = 0
Ορίζω μια μεταβλητή με όνομα counter ως τύπου ακέραιος και της δίνω αρχική τιμή το 0!
While counter < 20
Όσο η μεταβλητή counter είναι μικρότερη του 20
counter += 1
Η counter αυξάνεται κατά 1
MsgBox("Η While έχει τρέξει " & counter & " φορές!")
Εμφάνισε παράθυρο με μήνυμα.
End While
Εδώ τελειώνει η While!

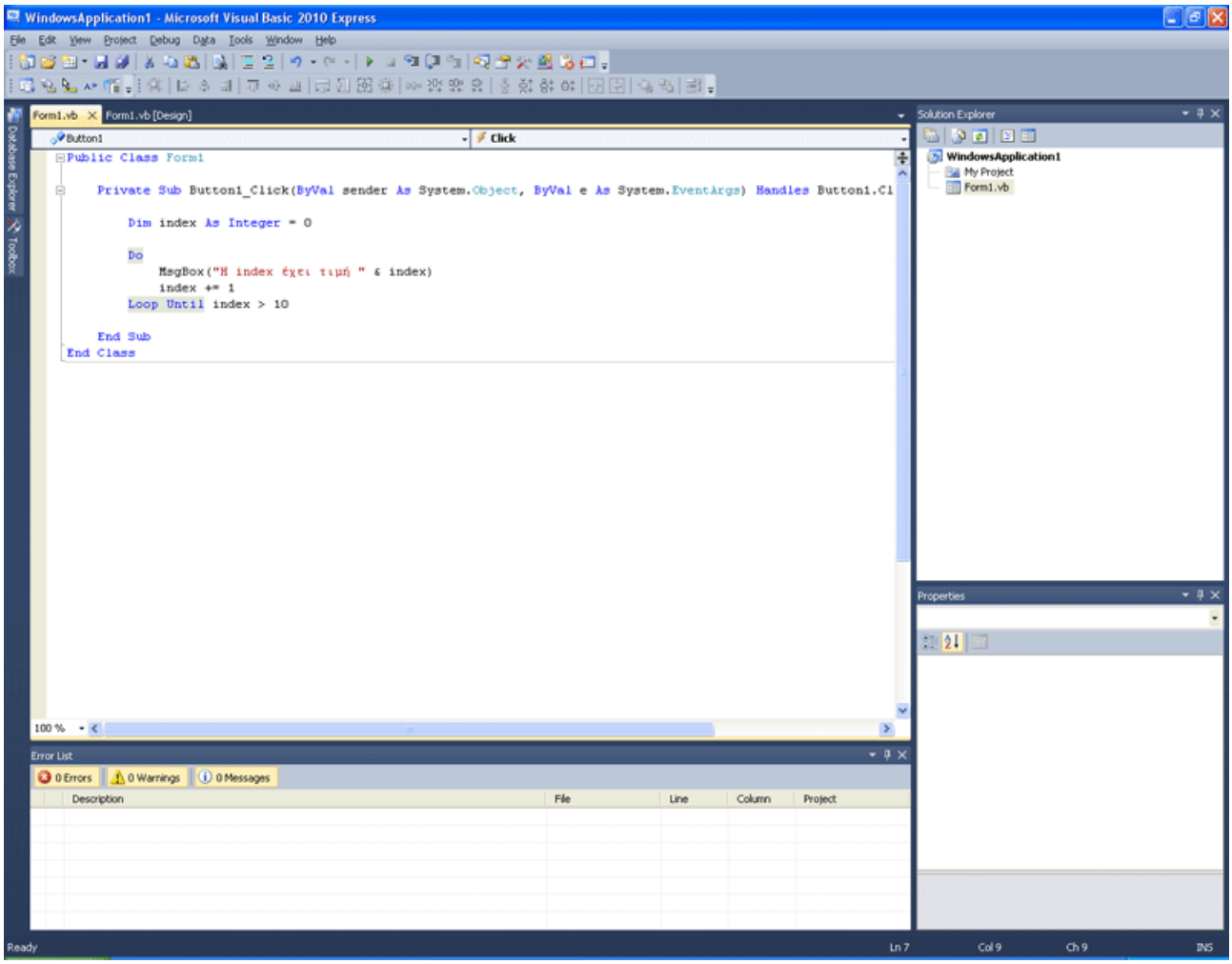
MsgBox ("Η While έτρεξε " & counter & " φορές!")
Μετά τη While, εμφανίζουμε με αυτή την εντολή ακόμα ένα μήνυμα.

Τρέξτε το πρόγραμμα και πατήστε το κουμπί.
Βλέπετε τη λειτουργία της επανάληψης;
Σβήστε τον κώδικα που γράψατε εδώ.
Θα δοκιμάσουμε τώρα κάτι άλλο.

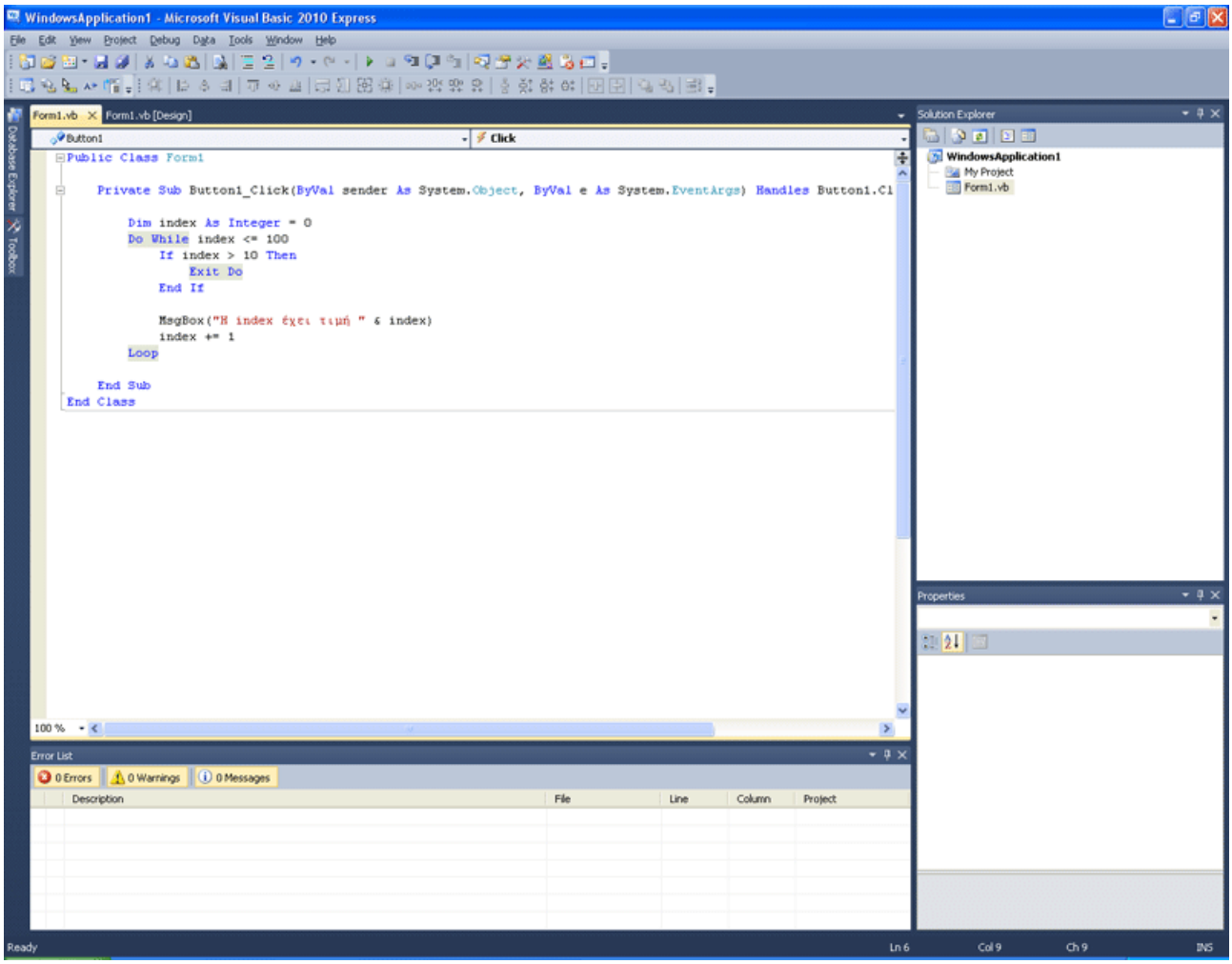


Γράψτε τον παρακάτω κώδικα.

```
Dim index As Integer = 0
  Do
    MsgBox ("H index έχει τιμή " & index)
    index += 1
  Loop Until index > 10
```



Εδώ τι έχουμε;
Dim index As Integer = 0
Ως μεταβλητή ακεραίων όρισε την index, με αρχική τιμή 0!
Do
Κάνε
MsgBox ("H index έχει τιμή " & index)
Εμφάνισε ένα MessageBox με το μήνυμα "H index έχει τιμή " και το περιεχόμενο της μεταβλητής index δίπλα.
index += 1
H index αυξάνεται κατά ένα.
Loop Until index > 10
Επέστρεψε στο Do κι επανέλαβε τα βήματα, μέχρι η index να γίνει μεγαλύτερη του 10 όπου προχωράς παρακάτω.
Τρέξτε το πρόγραμμα.
Το πρόγραμμα επαναλαμβάνει τη διαδικασία 11 φορές.
Οι τιμές της index είναι 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.
Σβήστε πάλι τον κώδικα.
Θα δοκιμάσουμε κάτι τελευταίο.



Ο κώδικας είναι ο παρακάτω:

```
Dim index As Integer = 0
Do While index <= 100
    If index > 10 Then
        Exit Do
    End If
    MsgBox ("H index έχει τιμή " & index)
    index += 1
Loop
```

Εδώ έχουμε κάτι νέο.
Dim index As Integer = 0
Ορίζουμε ώς μεταβλητή ακεραίων την index, με αρχική τιμή 0!
Do While index <= 100
Κάνε, όσο η index είναι μικρότερη ή ίση με 100
όλες τις εντολές που περιέχονται μεταξύ Do While και Loop.
If index > 10 Then
Exit Do
End If
Αν η index γίνει μεγαλύτερη από 10 τότε βγες από το βρόχο Do.
Βρόχος είναι μια δομή επανάληψης.

MsgBox ("Η index έχει τιμή " & index) Εμφάνισε ένα MessageBox με το μήνυμα "Η index έχει τιμή "
και το περιεχόμενο της μεταβλητής index δίπλα.
index += 1
Η index αυξάνεται κατά ένα.
Loop
Γύρνα πίσω στην αρχή, στη Do While index <= 100
κι επανέλαβε τα βήματα.
Τρέξτε το πρόγραμμα.

Ο Do While βρόχος εδώ, θα επαναλαμβάνεται μέχρι το 100. Ωστόσο, επειδή έχουμε μια If μέσα του, η οποία περιέχει ένα Exit Do, αυτή η Exit Do εντολή, θα μας βγάλει έξω από το βρόχο Do While, πριν φτάσουμε στο 100, διότι αναφέρεται στην ίδια μεταβλητή, την index. Με το που φτάσει η index στο 11, θα βγούμε από το βρόχο λόγω της If λοιπόν. Πρακτικά αυτό φαίνεται ανόητο. Και δίκιο έχετε να σας φαίνεται έτσι. Εδώ σας δείχνουμε τι μπορούμε να κάνουμε ουσιαστικά. Ο κώδικας δεν έχει κάποια πρακτική εφαρμογή. Ωστόσο, το Exit Do μπορεί να υλοποιείται, με κάποια άλλη μεταβλητή στην if, πέραν της index, η οποία θα παίρνει τιμές μέσα στο βρόχο, αν συμβαίνει κάτι. Για παράδειγμα, σε ένα παιχνίδι με ερωτήσεις πολλαπλής επιλογής, 100 γύρων, με 3 ευκαιρίες να κάνει λάθος ο παίκτης, όποτε κάνει λάθος ο παίκτης, θα αυξάνεται η μεταβλητή error κατά 1. Στα τρία λάθη σε στέλνει εκτός βρόχου, ειδάλλως τερματίζεται το παιχνίδι στον εκατοστό γύρο. Οι επιλογές είναι πολλές, εδώ στόχο έχουμε να σας δείξουμε πως λειτουργούν τα εργαλεία σας.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

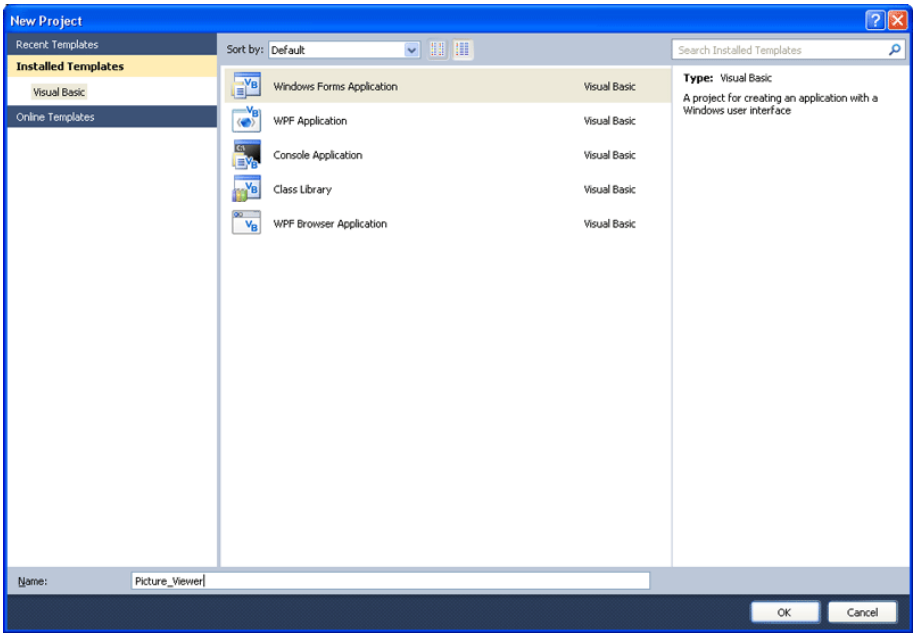
Το κεφάλαιο αυτό είναι ουσιαστικά προέκταση του προηγούμενου. Αναλύονται οι εντολές DO LOOP WHILE UNTIL με απλά παραδείγματα και όπως πάντα με ανάλυση του κώδικα.

ΚΕΦΑΛΑΙΟ 15
ΠΕΡΙΣΣΟΤΕΡΟ ΠΡΟΣΕΓΜΕΝΑ ΠΡΟΓΡΑΜΜΑΤΑ
ΑΣ ΦΤΙΑΞΟΥΜΕ ΕΝΑ ΠΡΟΓΡΑΜΜΑ ΕΜΦΑΝΙΣΗΣ ΕΙΚΟΝΩΝ!

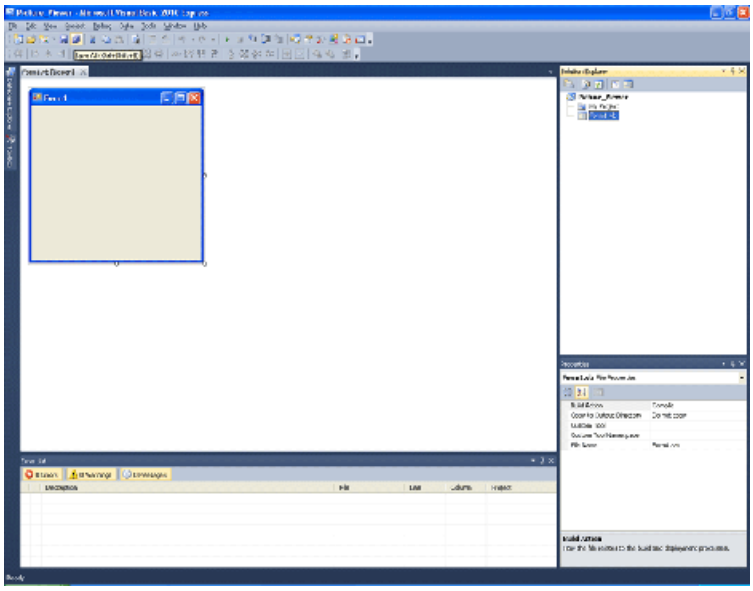
Είναι πλέον καιρός να πάμε ένα βήμα παραπέρα. Είναι καιρός να φτιάξουμε ένα πιο σοβαρό πρόγραμμα. Το παρόν κεφάλαιο βασίστηκε σε μία υπέροχη σειρά εκπαιδευτικών video της Kathleen McGrath, σε αντίστοιχο θέμα του msdn της Miscrosoft.

http://msdn.microsoft.com/en-us/vbasic/gg315352

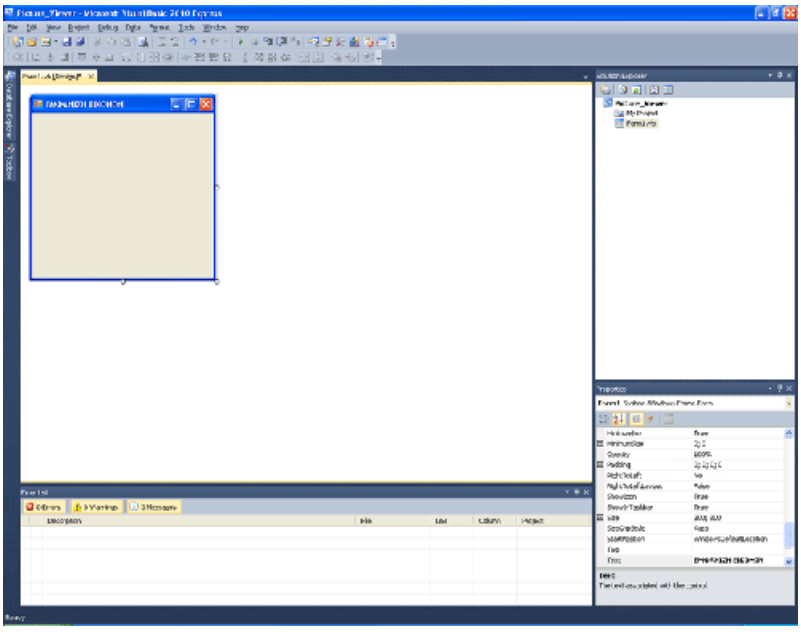
Εδώ θα μάθετε πολλά νέα πράγματα οπότε θα προχωράμε σιγά σιγά. Το κεφάλαιο, γι' αυτό το λόγο, είναι σχετικά μεγάλο, με πολλές εικόνες, έτσι ώστε να μπορέσετε να κατανοήσετε πλήρως κάθε τι καινούργιο που θα δείτε εδώ.
Για αρχή φτιάχνουμε ένα νέο project Windows Forms Application, με όνομα Picture_Viewer, όπως βλέπετε στην παρακάτω εικόνα.



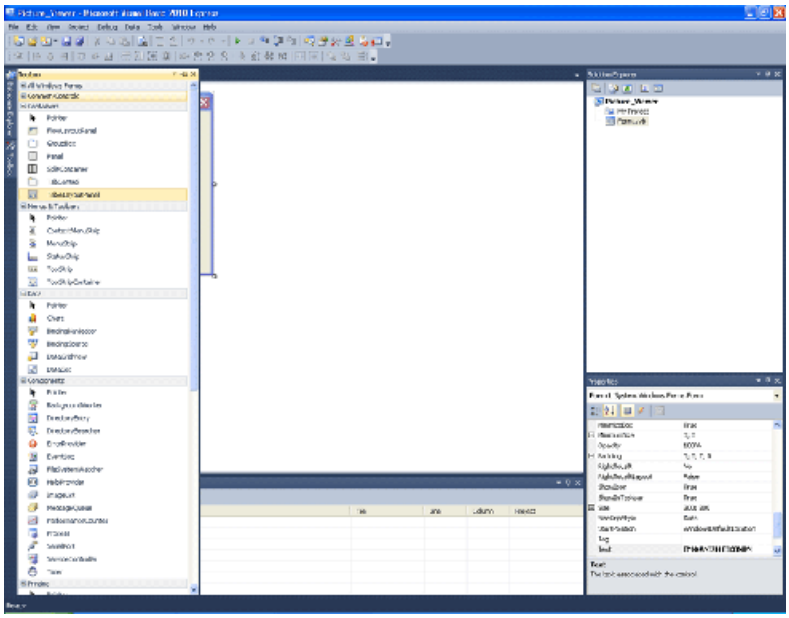
Κάνουμε Save all για να σωθεί το Project.



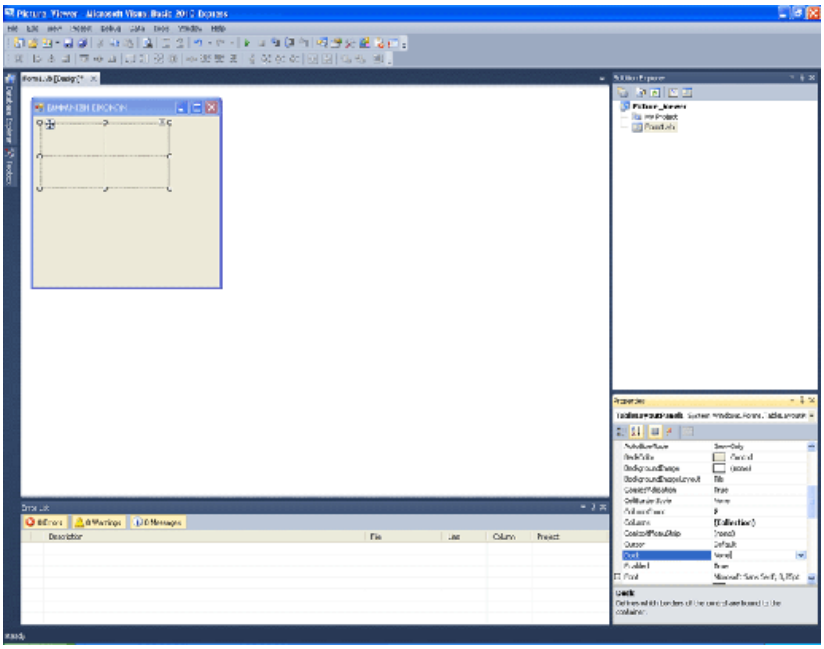
Ονομάζουμε τη φόρμα "ΕΜΦΑΝΙΣΗ ΕΙΚΟΝΩΝ", δίνοντας αυτή την τιμή στην ιδιότητα Text.



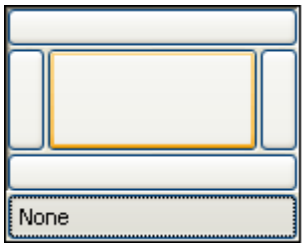
Πάμε τώτα στο toolbox, στο υπομενού Containers κι επιλέγουμε TableLayoutPanel.



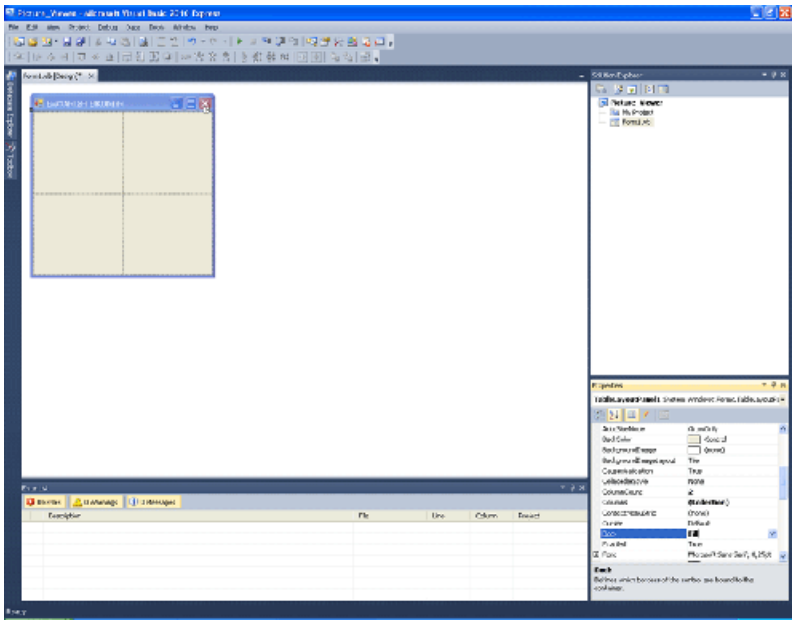
Εμφανίζεται αυτό το container. Το επιλέγουμε και βρίσκουμε την ιδιότητα Dock.



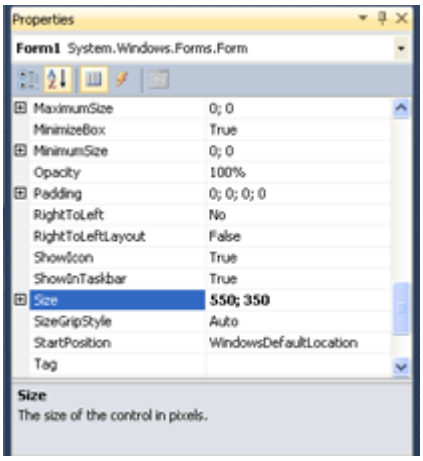
Είναι ComboBox. Όταν πατάμε σε αυτό, εμφανίζεται το παρακάτω παραθυράκι. Επιλέξτε το κεντρικό κιτρινισμένο πλαίσιο. Αυτό είναι η dock τιμή "Fill".



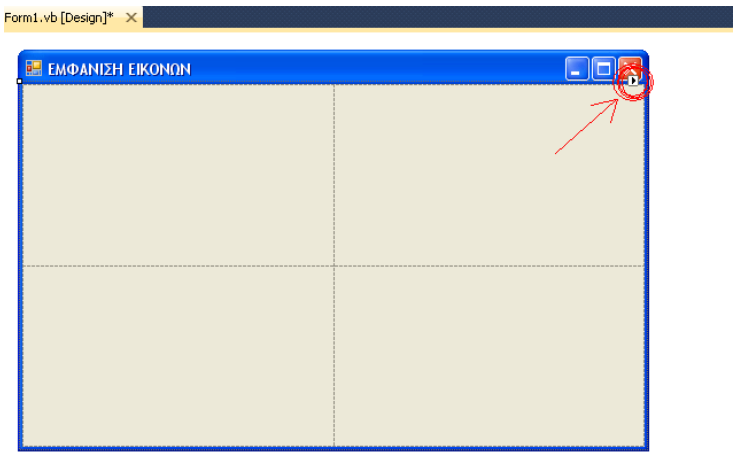
Το Container γεμίζει όλη τη φόρμα!



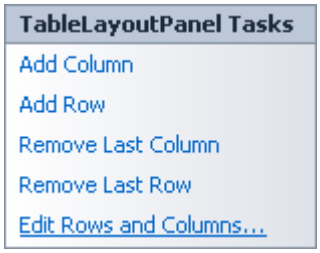
Επιλέξτε τη φόρμα κάνοντας πάνω της κλικ. Κάντε το Size της φόρμας τώρα 550;350



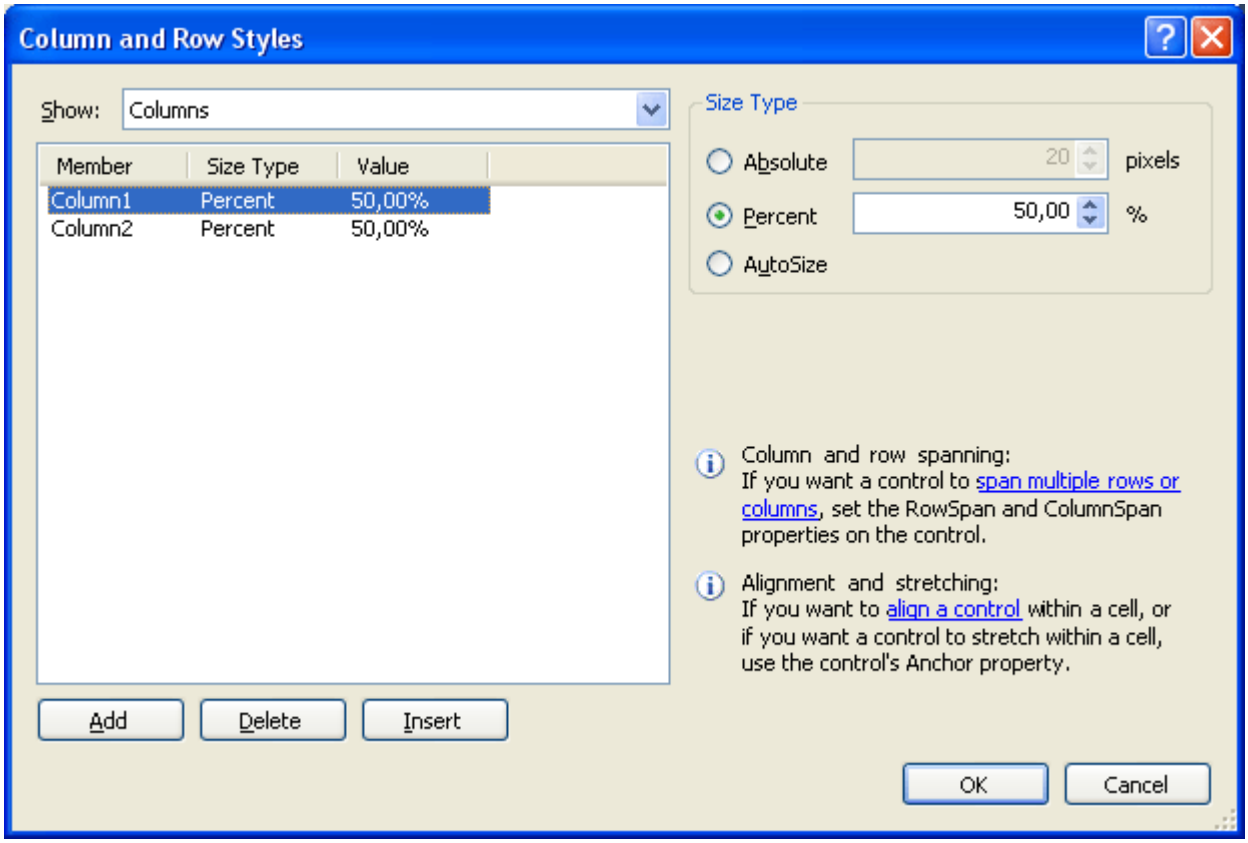
Κάντε τώρα κλικ σε αυτό το τετραγωνάκι, που βλέπετε στην παρακάτω εικόνα.



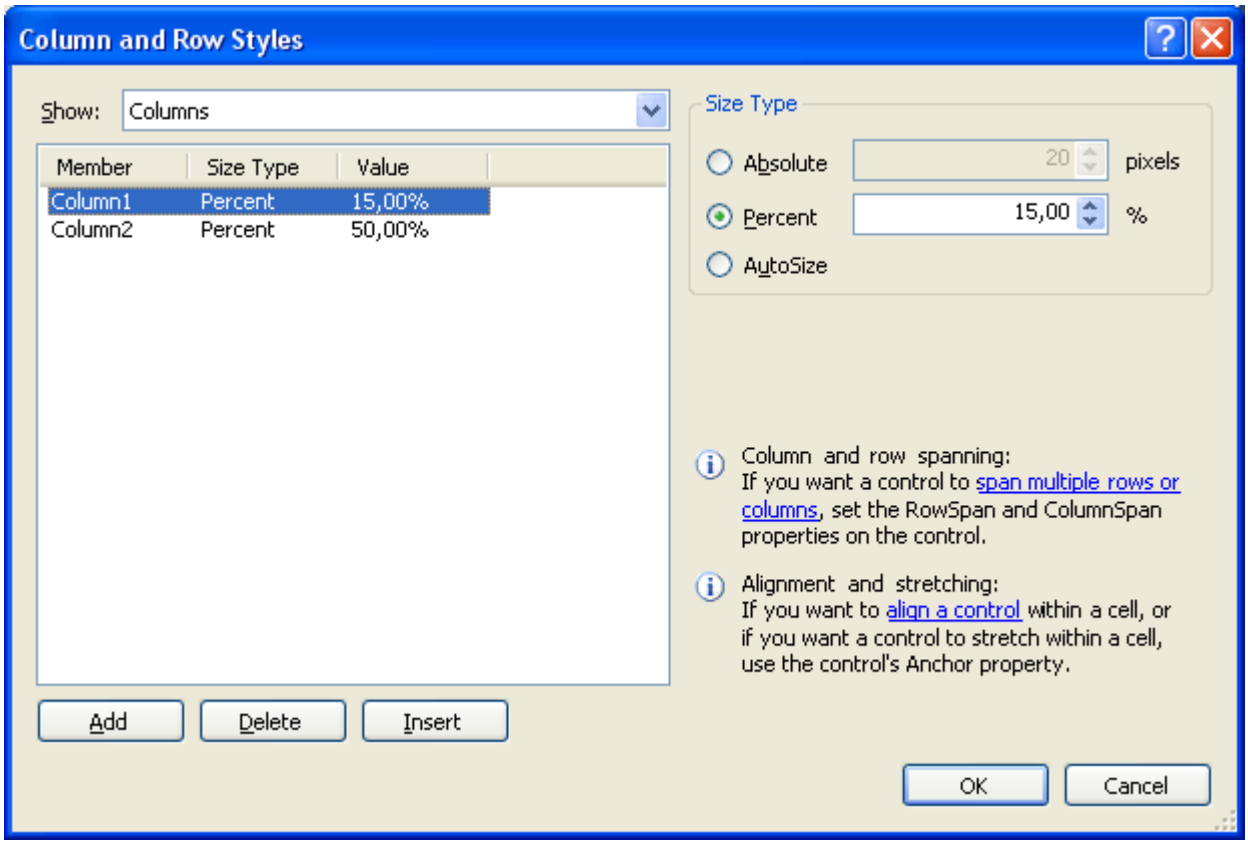
Επιλέγετε Edit Rows and Columns.



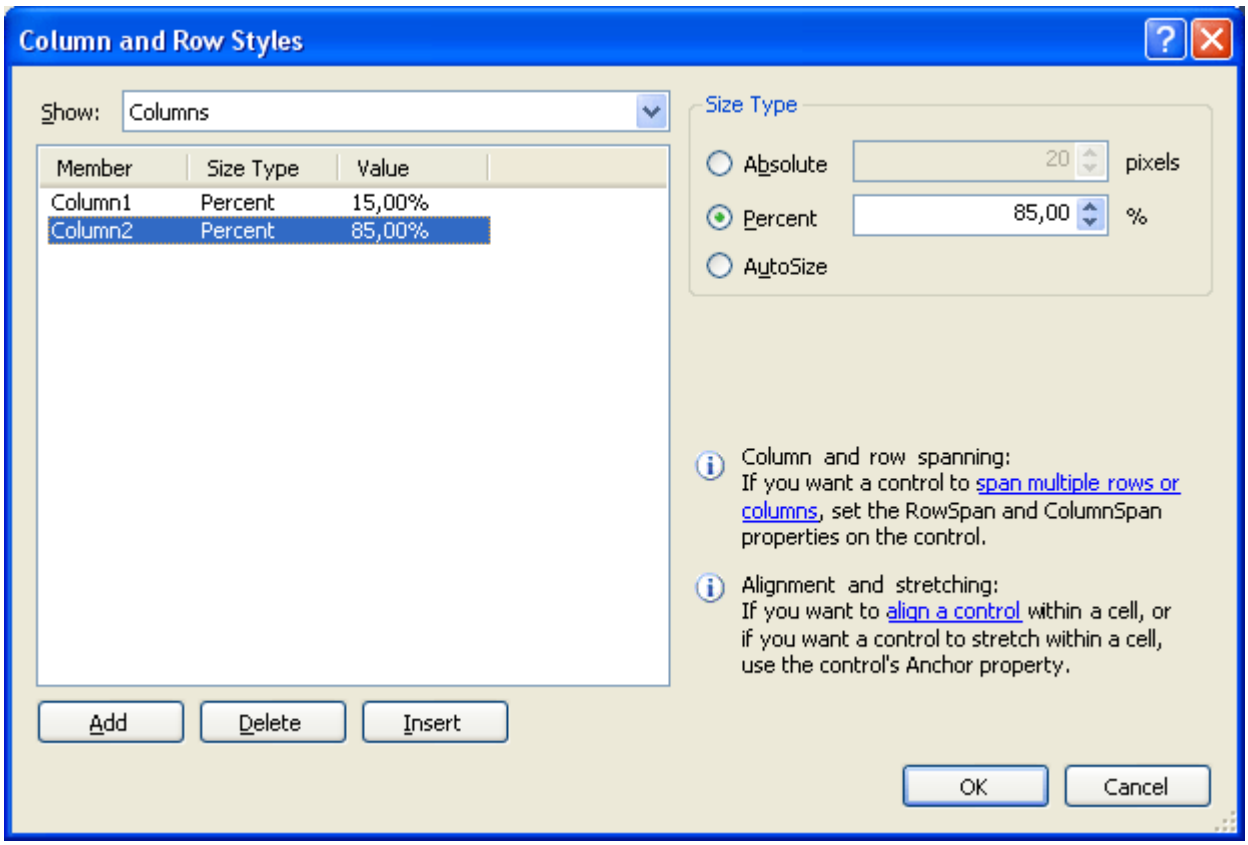
Επιλέξτε την Column1, όπως βλέπετε παρακάτω.



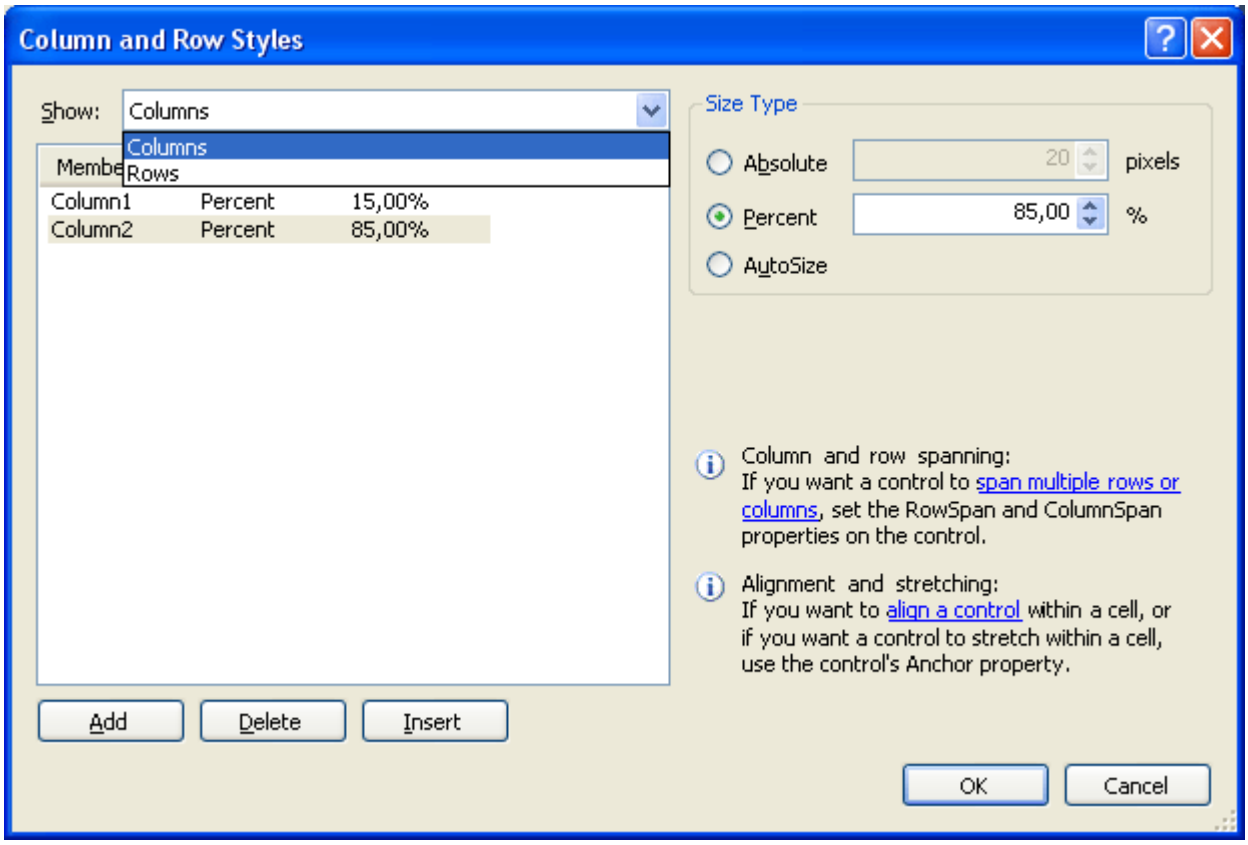
Ορίζετε το percent στο 15% όπως στην παρακάτω εικόνα.



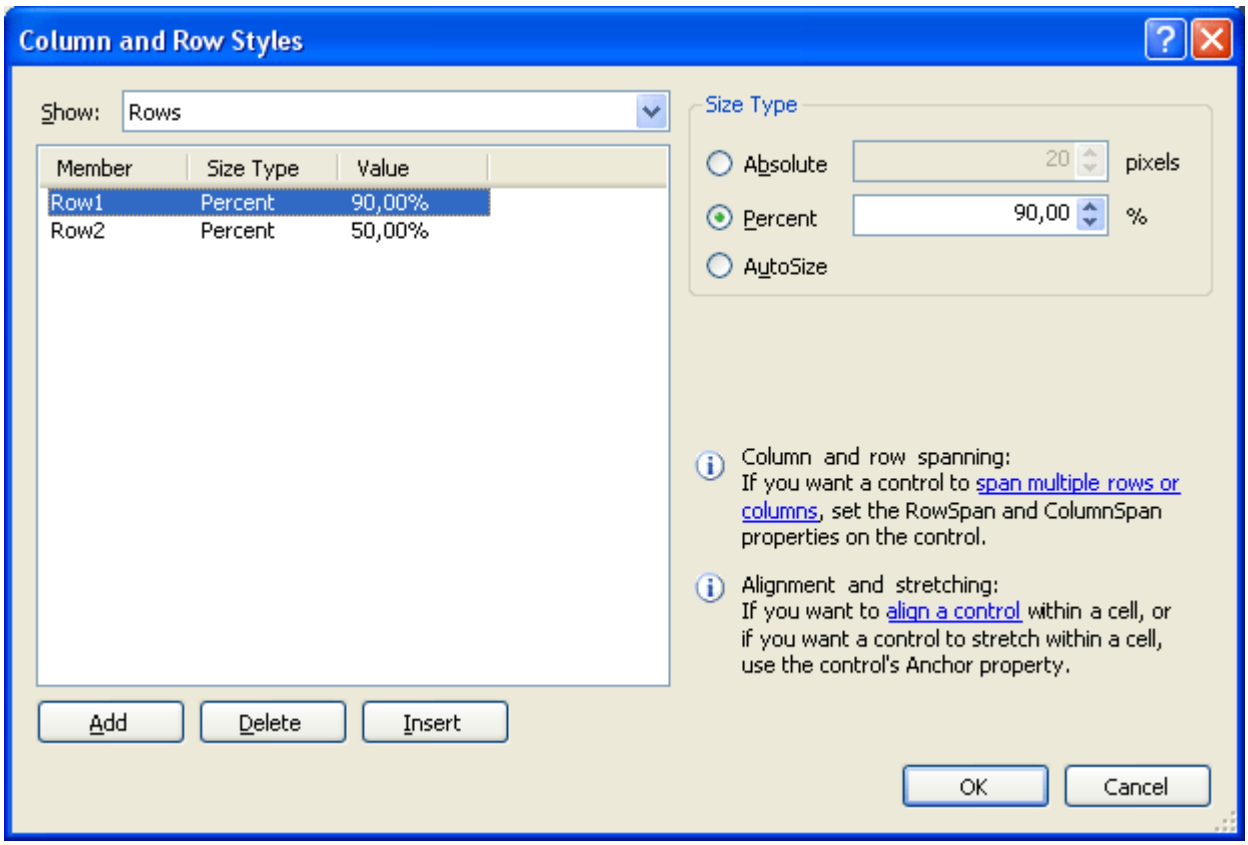
Της δεύτερης θα της δώσουμε τιμή 85% όπως καταλαβαίνετε.



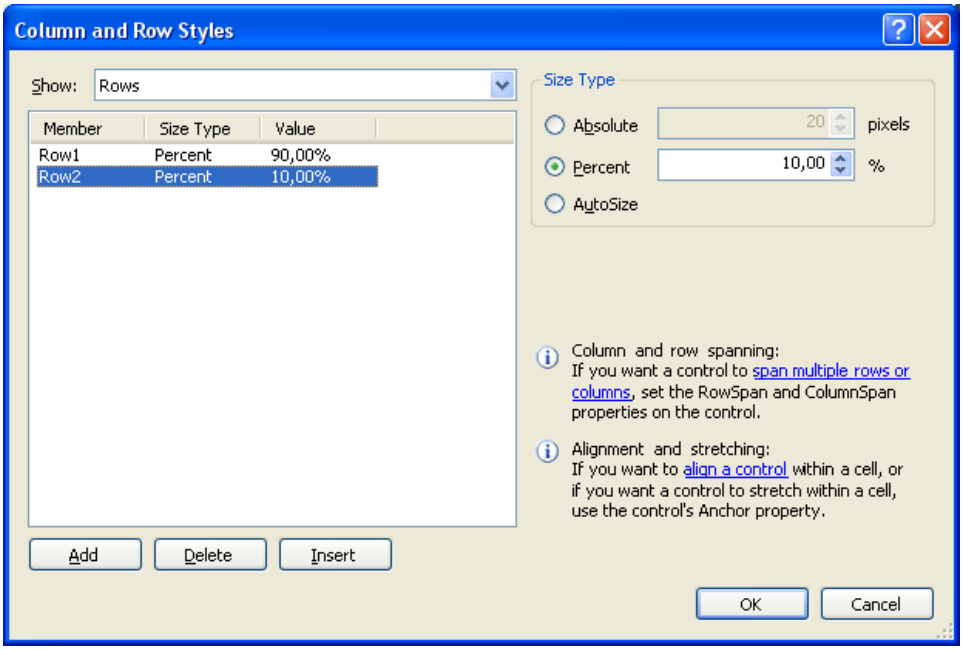
Πάμε στο Show εκεί που δείχνει Columns



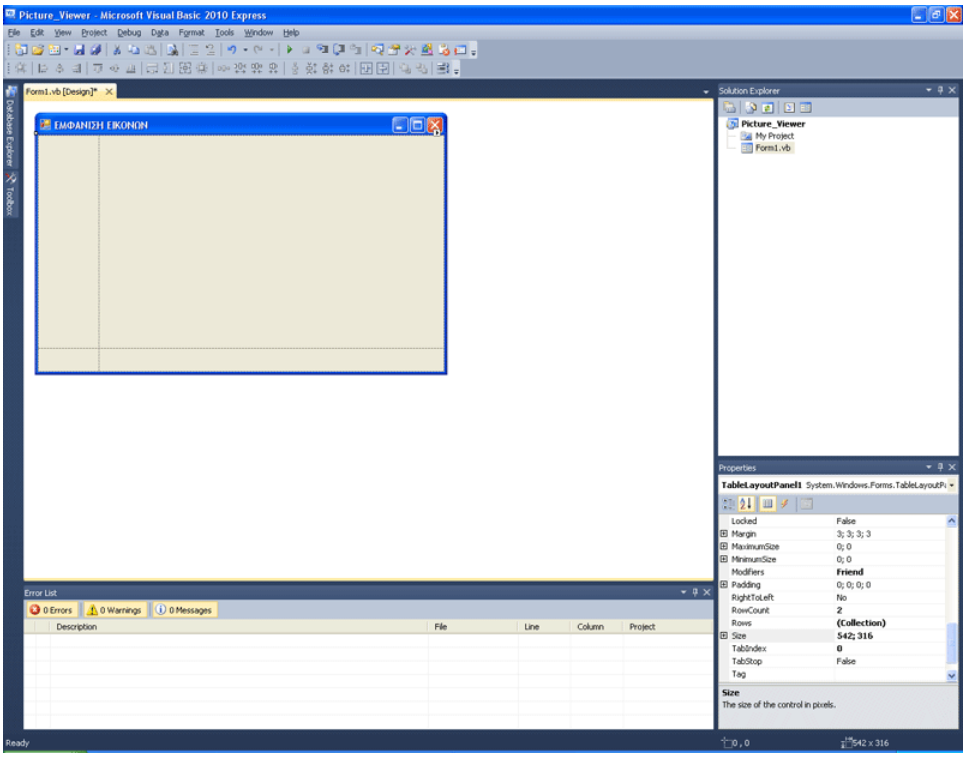
Το κάνουμε Rows - Της πρώτης Row την τιμή θα την κάνουμε 90%...



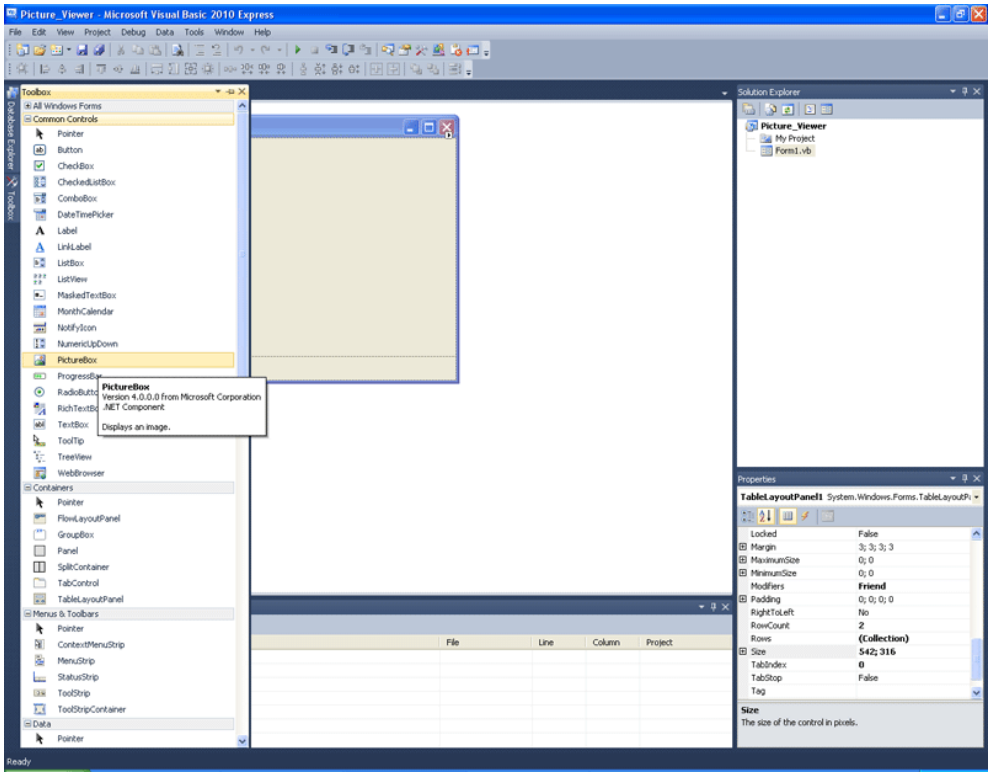
...ενώ τη δεύτερη 10%



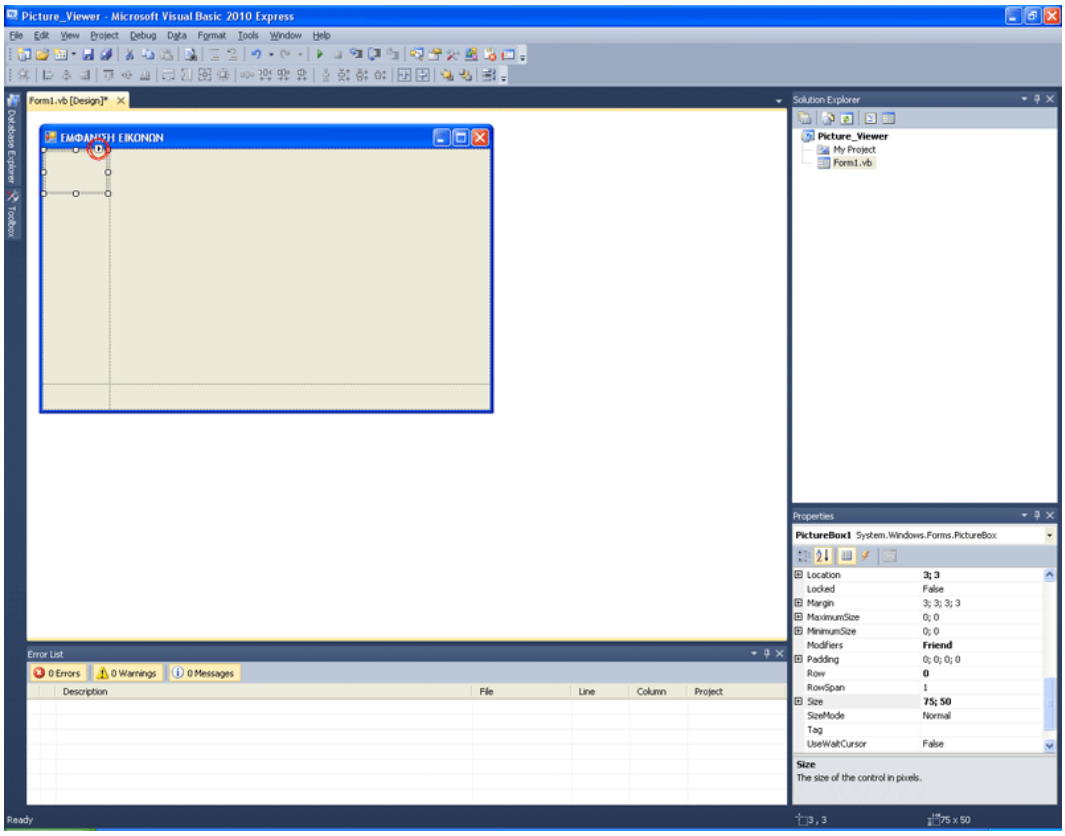
Ορίστε το αποτέλεσμα της διαδικασίας. Τι κάναμε όμως ακριβώς; Ορίσαμε αρχικά δύο στήλες (columns), με τιμές 15% και 85% αντίστοιχα. Μετά οι γραμμές (rows), από τα αριστερά προς τα δεξιά, θα καταλαμβάνουν 90% και 10% του χώρου αντίστοιχα! Παρακάτω φαίνεται και το τι κάναμε. Με αυτό τον τρόπο έχουμε χωρίσει τη φόρμα σε 4 μέρη. Κάθε μέρος χωράει μόνο ένα αντικείμενο.



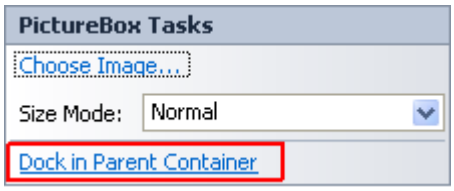
Επιλέξτε και βάλτε ένα PictureBox.Σε αυτό μπορεί να εμφανιστεί μια εικόνα



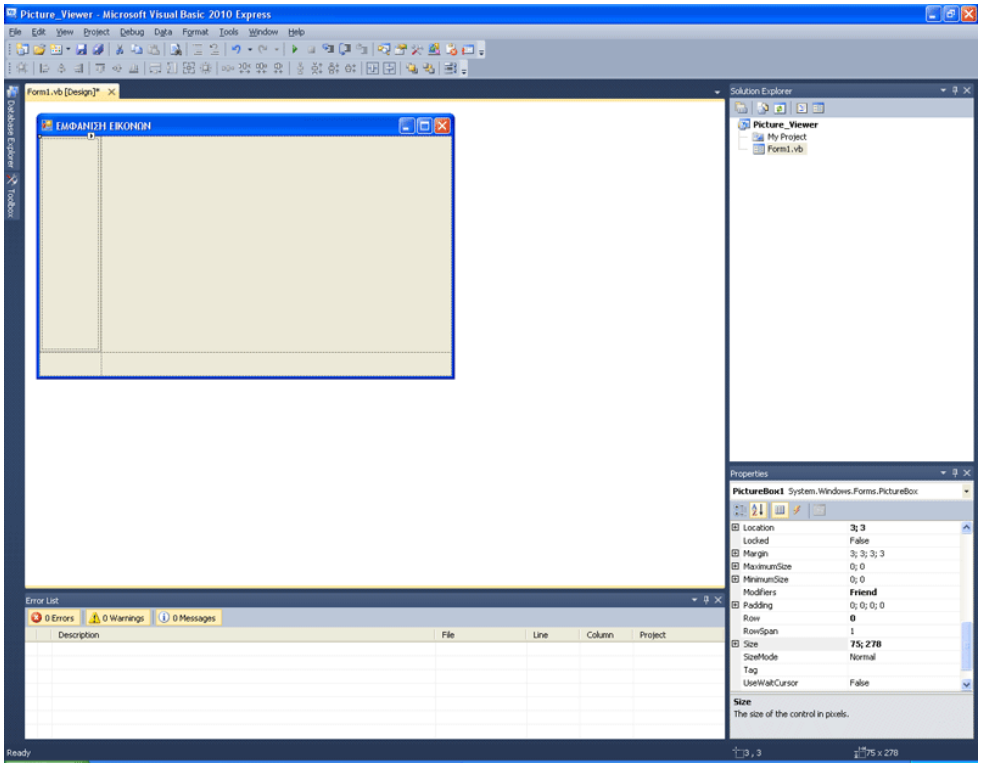
Κάντε κλικ στο μικρό αυτό τετραγωνάκι που έχουμε κυκλώσει στην εικόνα



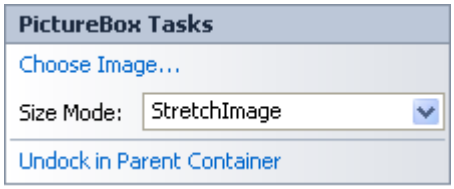
Επιλέγουμε Dock in parent Container



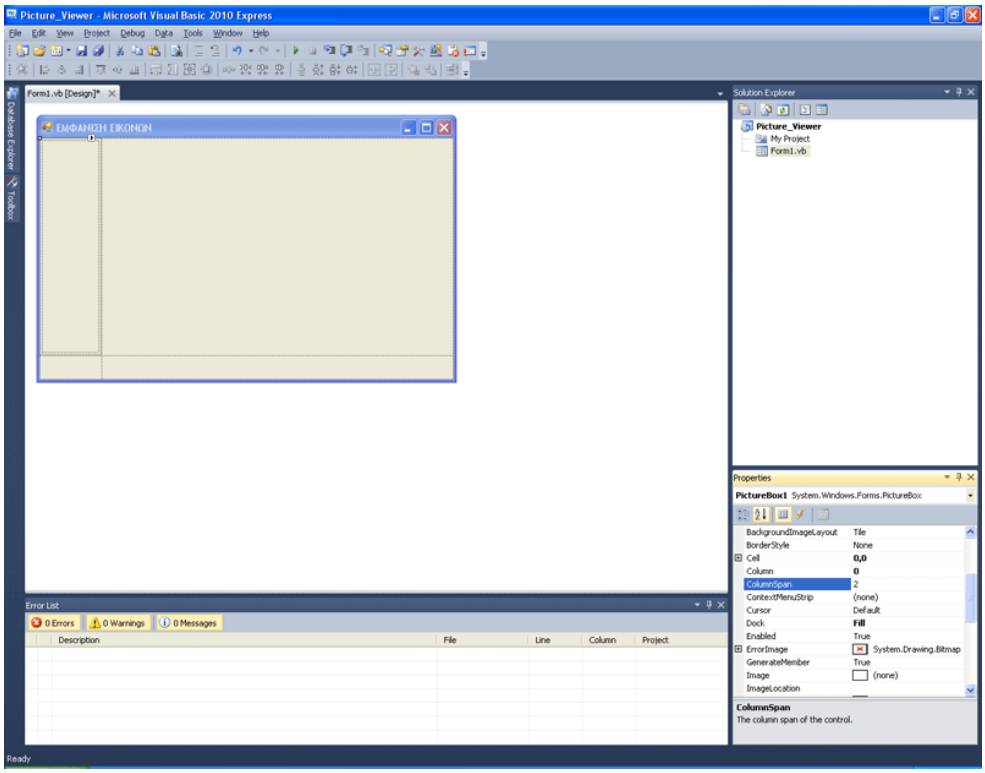
Κι έτσι γεμίζει το πλαίσιο όλο με το PivtureBox.
Ωστόσο θέλω να προσαρμόζει αυτόματα το μέγεθος της εικόνας, ανάλογα με το μέγεθος της φόρμας



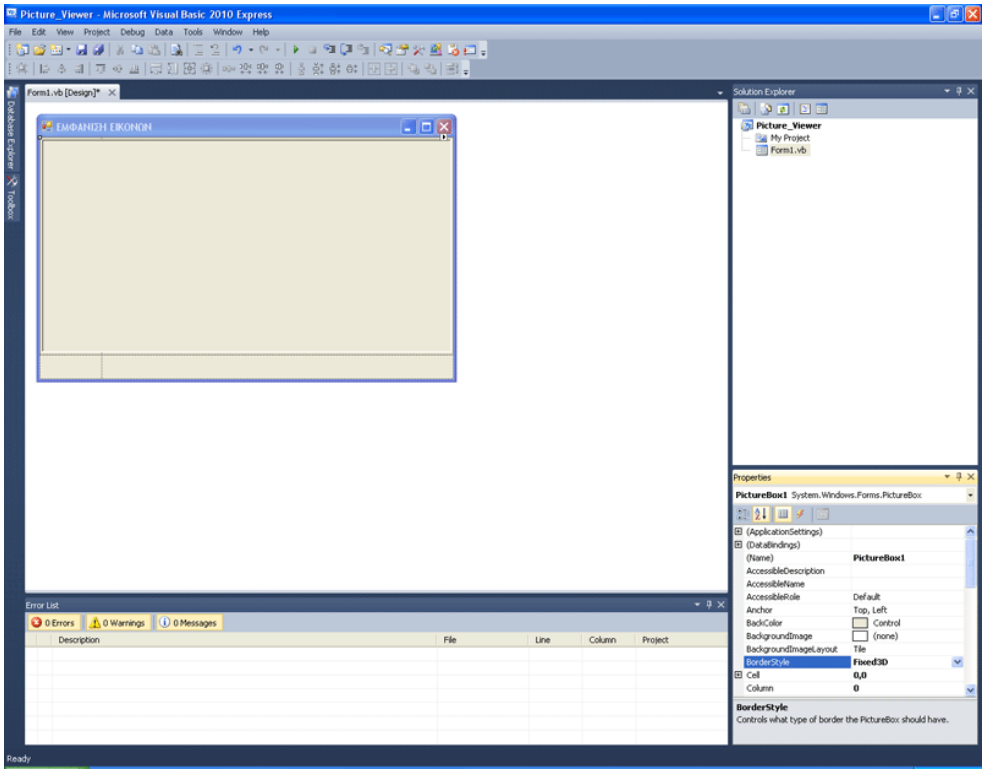
Στο Size mode επιλέγω StretchImage λοιπόν



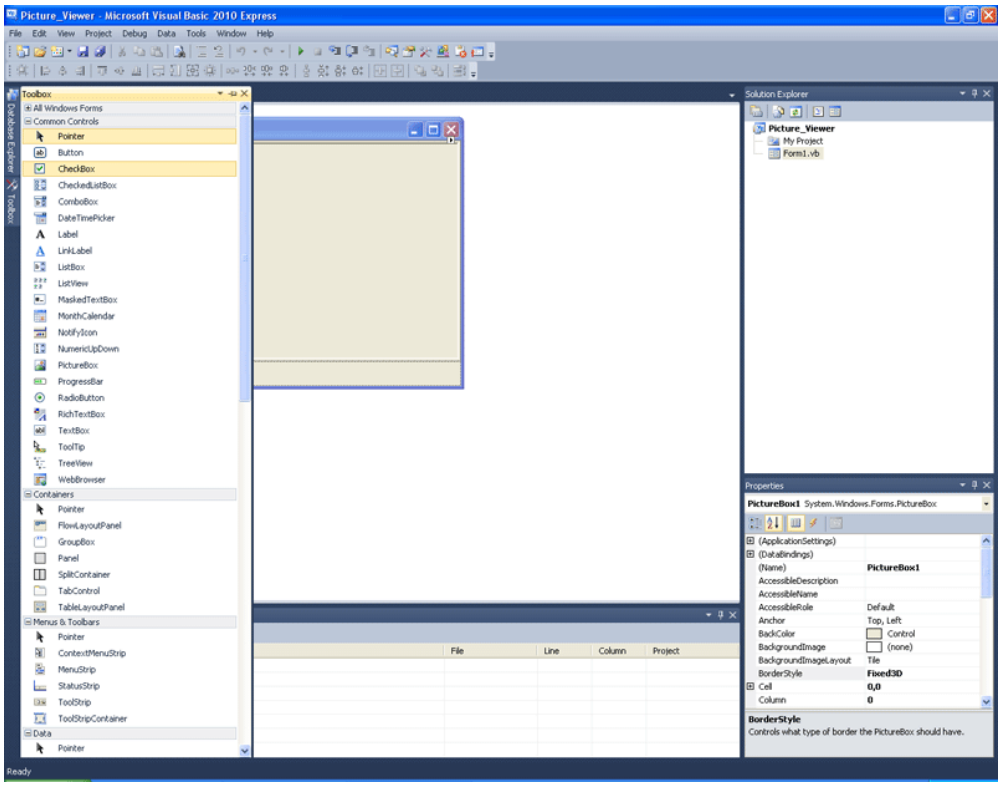
Έπειτα πάω στην ιδιότητα Columnspan και δίνω τιμή 2.
Τι κάναμε τώρα; Το PictureBox κατέλαβε και τις δύο στήλες!



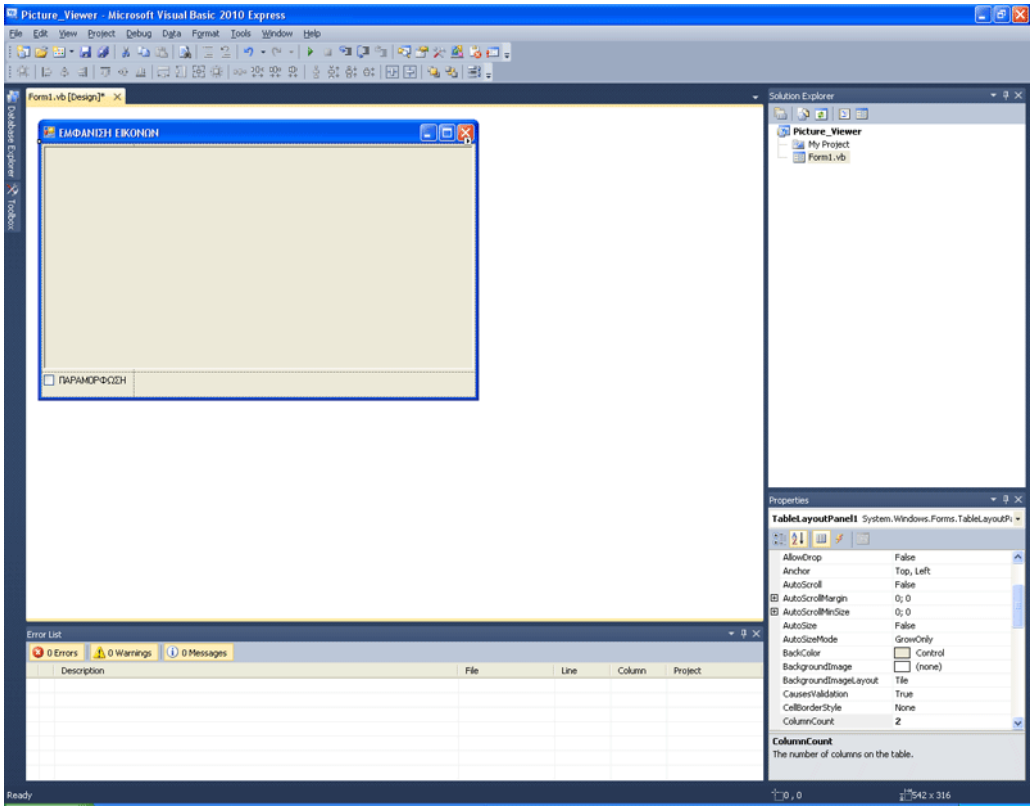
Ας κάνουμε και το BorderStyle ίσο με "Fixed 3D"
Έτσι για ομορφιά. Πως σας φαίνεται;



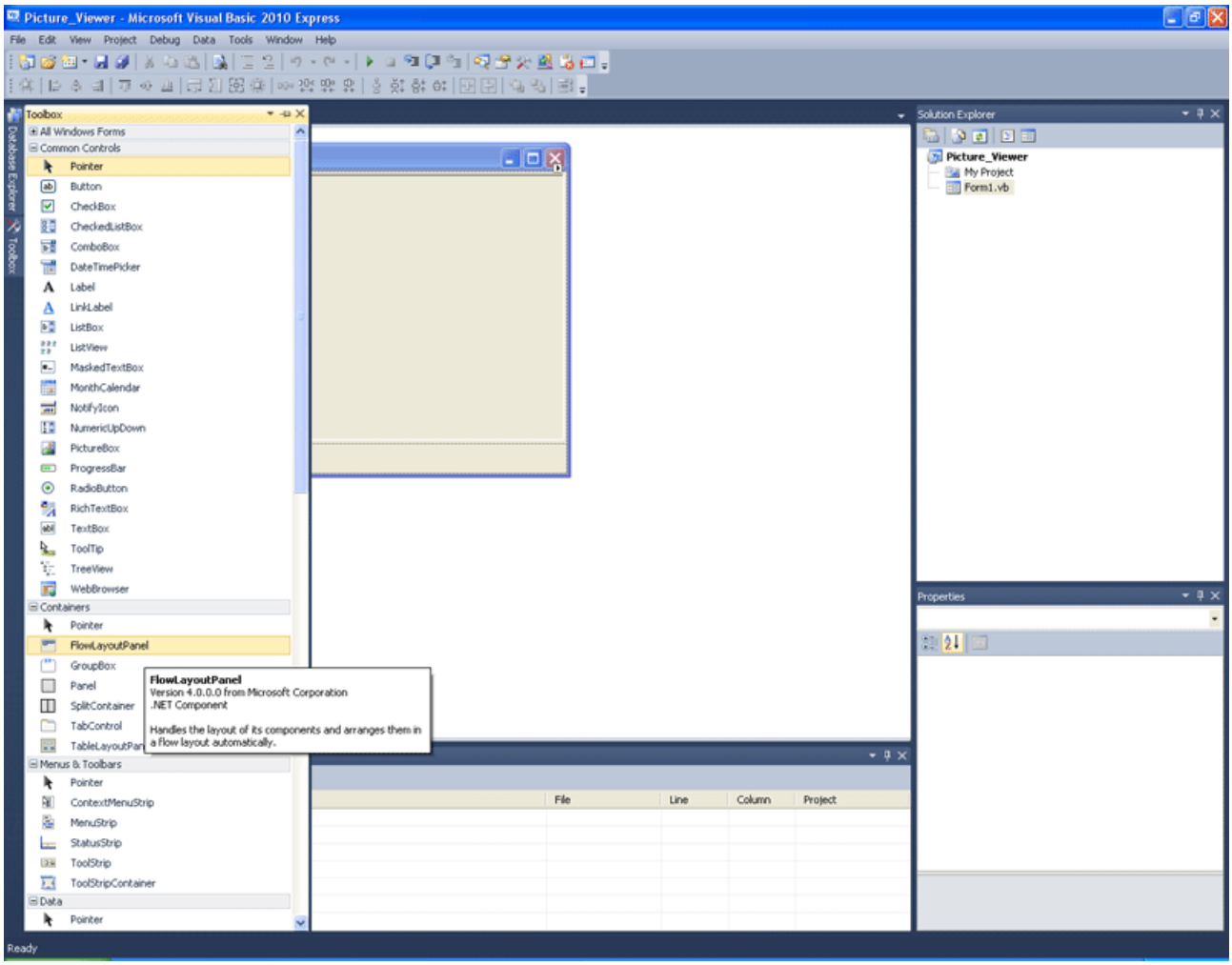
Στο επόμενο container βάζω CheckBox



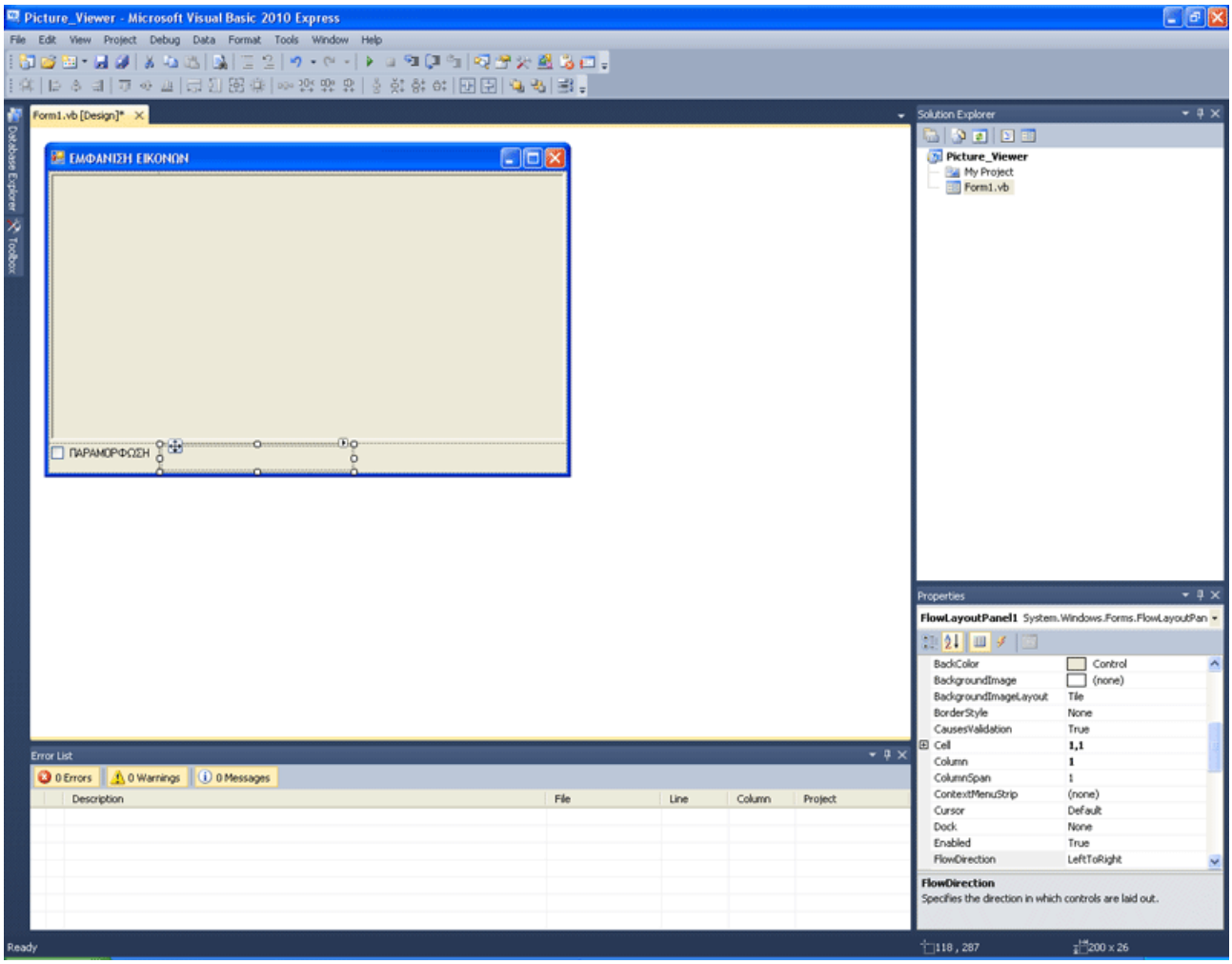
Το Text θα λέει "ΠΑΡΑΜΟΡΦΩΣΗ"



Εδώ θα πρέπει να αναφέρουμε το εξής. Παρατηρήσατε πως τόσο το PictureBox, όσο και το CheckBox, καταλαμβάνουν ένα ολόκληρο container; Στο επόμενο container όμως θα θέλω να βάλω 4 Buttons. Αυτό δε γίνεται χωρίς να προετοιμάσω το container να χωράει περισσότερα από ένα controls. FlowLayoutPanel είναι αυτό που προσθέτουμε πρώτα λοιπόν. Τι κάνει αυτό; Μέσα του χωράνε περισσότερα από ένα controls.

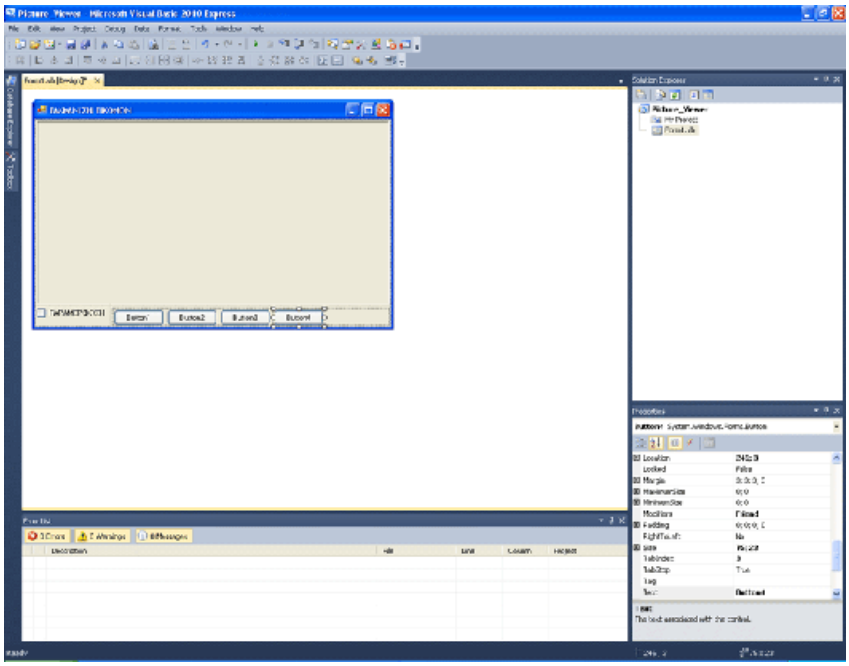


Βάλτε το στο τελευταίο container

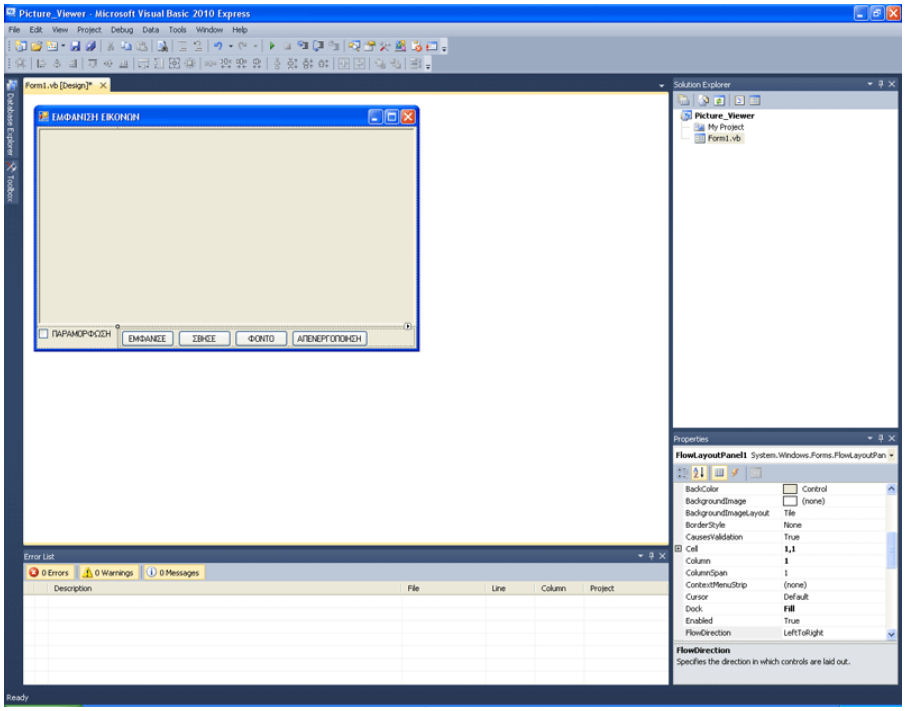


Πατήστε το τριγωνάκι, πάνω στο νέο αντικείμενο κι επιλέξτε Dock in Parent Container.

Έπειτα προσθέστε 4 Buttons

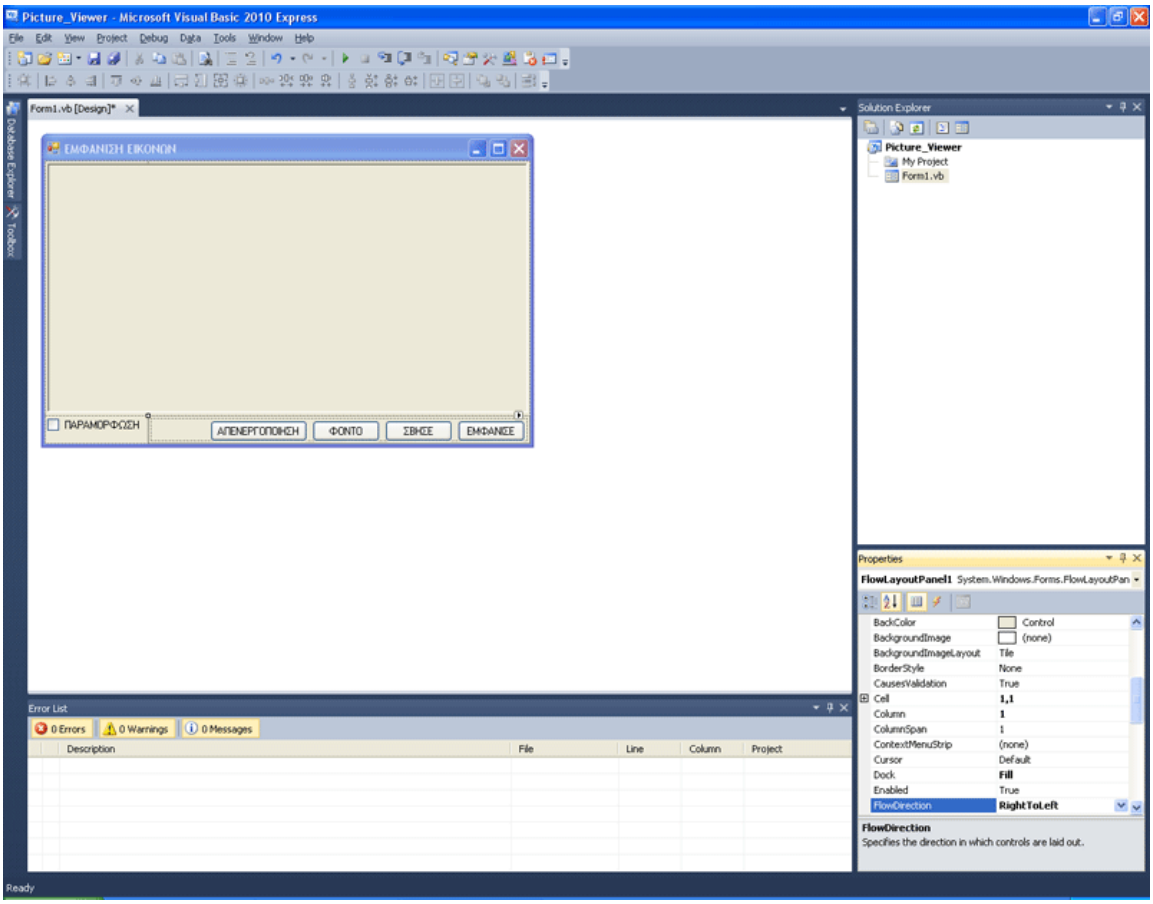


Ονομάστε τα (ιδιότητα text) με τη σειρά:
"ΕΜΦΑΝΙΣΕ", "ΣΒΗΣΕ", "ΦΟΝΤΟ", "ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ"

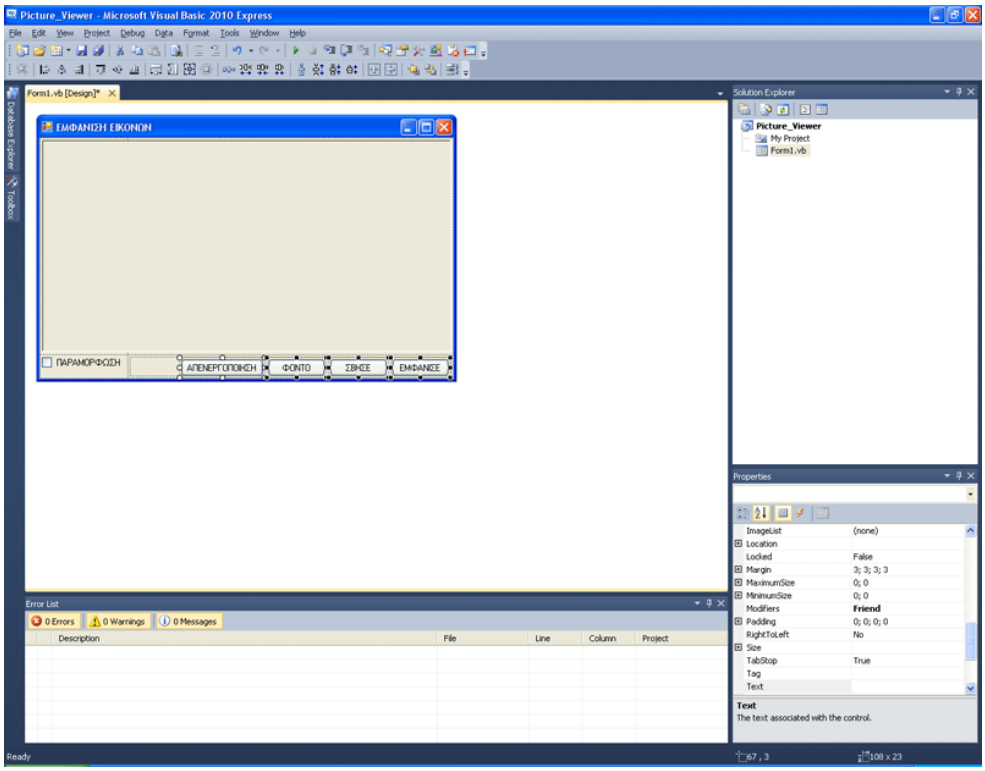


Επιλέξτε το FlowLayoutPanel.Βρείτε την ιδιότητα FlowDirection. Αυτή η ιδιότητα στοιχίζει με κάποιο τρόπο τα περιεχόμενά του FlowLayoutPanel1, δηλαδή εδώ, τα κουμπιά μας.

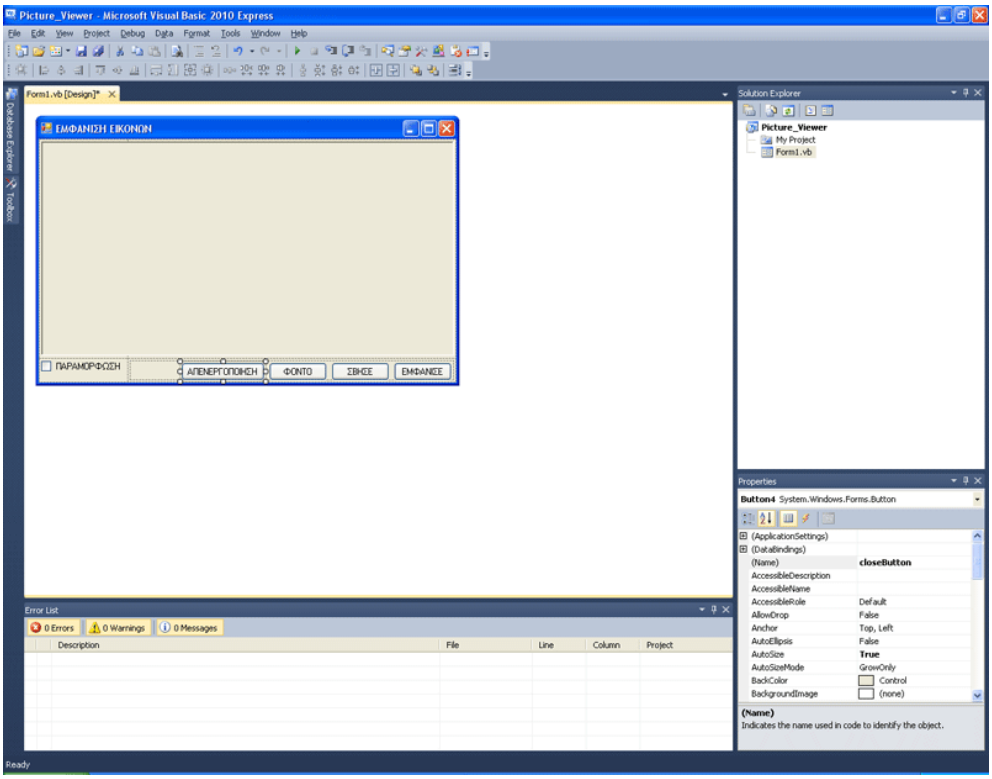
Επιλέξτε RightToLeft. Τα κουμπιά τοποθετούνται όπως θα δείτε από τα δεξιά προς τα αριστερά, σε διάταξη.



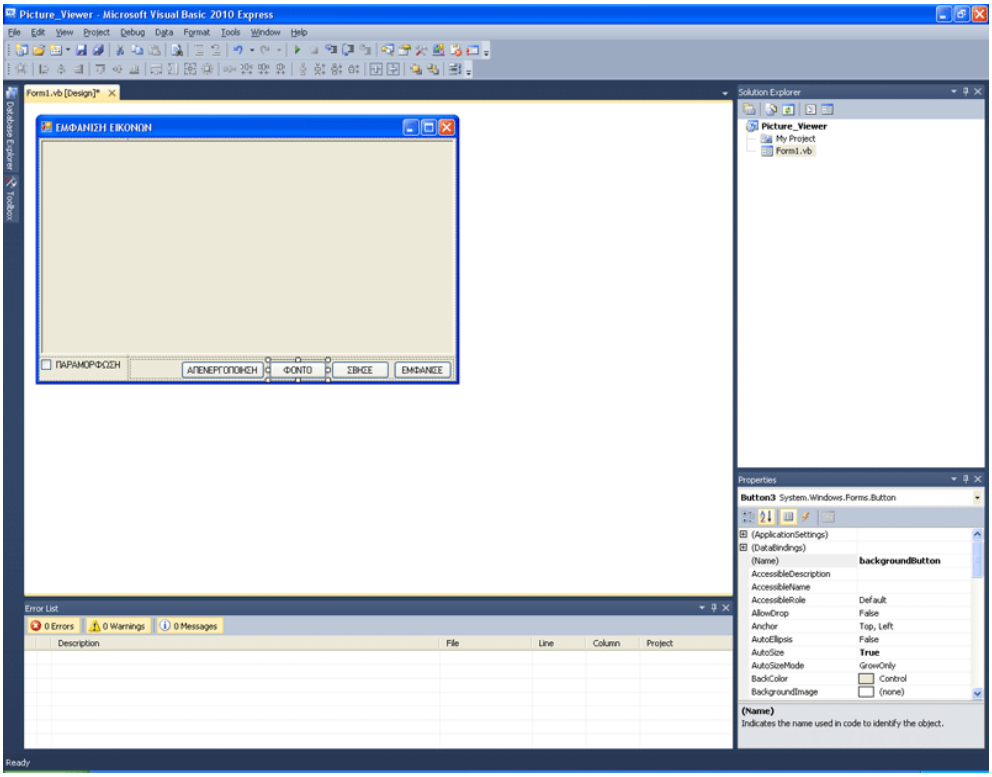
Κάνοντας κλικ σε ένα button, και κρατώντας το ctrl κάνοντας κλικ στα υπόλοιπα, καταφέρνετε να τα επιλέξετε όλα. Υπάρχον λοιπόν κάποιες ιδιότητες, που μπορείτε να αλλάξετε σε πολλά controls ταυτόχρονα, δίνοντας ίδια τιμή. Σαφώς αυτό δε γίνεται στην ιδιότητα name, και γι αυτό δε θα τη βρείτε. Κάθε αντικείμενο πρέπει να έχει το δικό του, μοναδικό προγραμματιστικό όνομα. Όμως ιδιότητες όπως AutoSize μπορούν να αλλάξουν. Βρείτε τη και θέστε τη ως True. Αν είχατε επιλέξει όλα τα buttons,τότε είναι true σε όλα. Αυτό σημαίνει πως το Button παίρνει μέγεθος, ανάλογα με το κείμενό του στην ιδιότητα text.



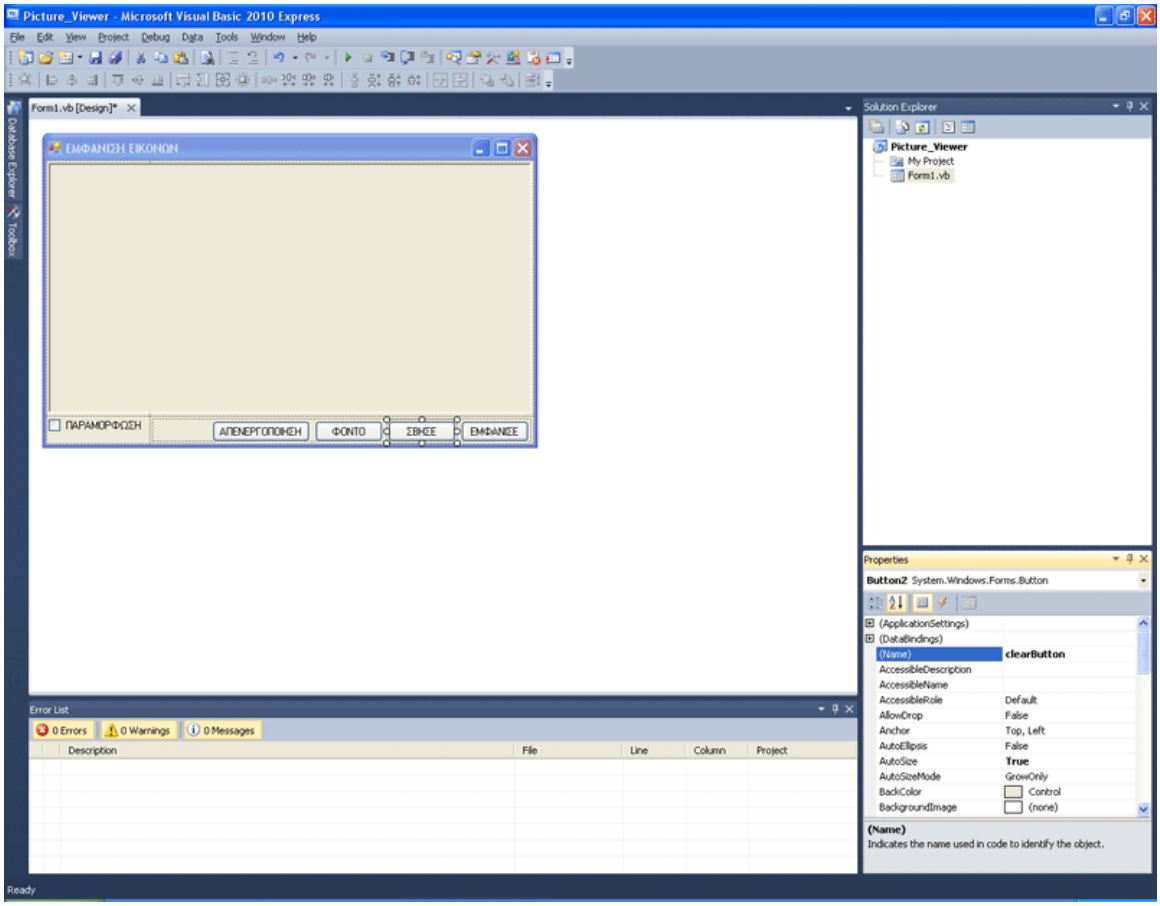
Επιλέξτε ΜΟΝΟ το κουμπί "ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ".
Βρείτε στα Properties στο κάτω δεξί menu την ιδιότητα (name). Είναι στις πρώτες επιλογές. Μετονομάστε τη σε closebutton. Γιατί το κάνουμε αυτό; Επειδή έχουμε τα Button1, Button2, Button3, Button4 κι επειδή σαν προγραμματιστές μπορεί να μπερδευτούμε με το τι κάνουν, παρεμπιπτόντως φανταστείτε αν είχαμε 40 αντί 4 τι θα γινόταν, τα ονομάζουμε με τρόπο που να καταλαβαίνουμε τη λειτουργία τους. Έτσι εδώ το closebutton συμπεραίνουμε άμεσα από το όνομα ότι θα απενεργοποιεί το πρόγραμμα.



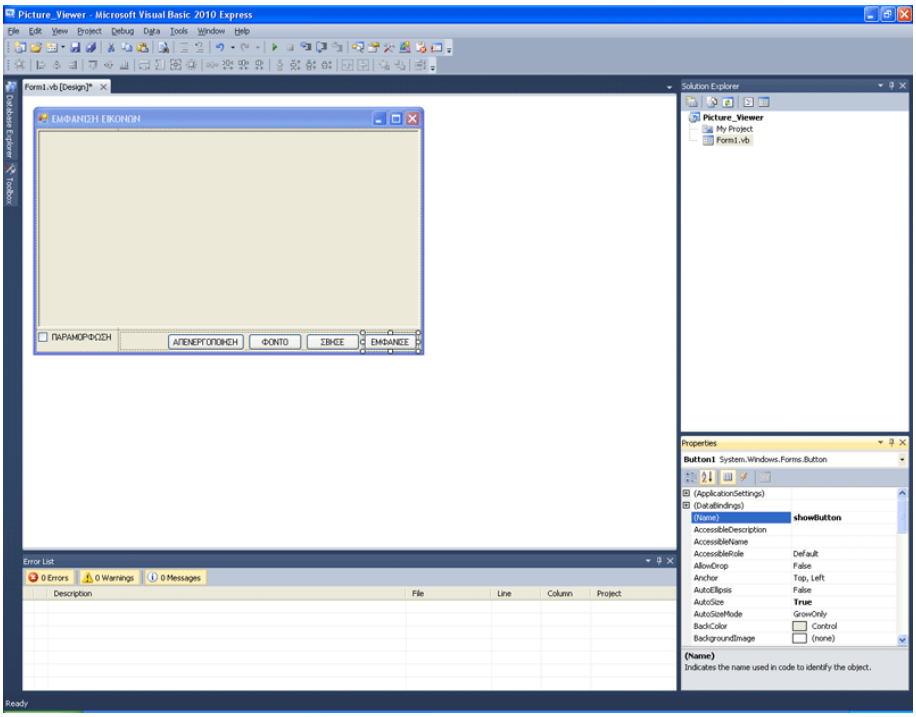
Το "ΦΟΝΤΟ" το ονομάζουμε backgroundbutton.



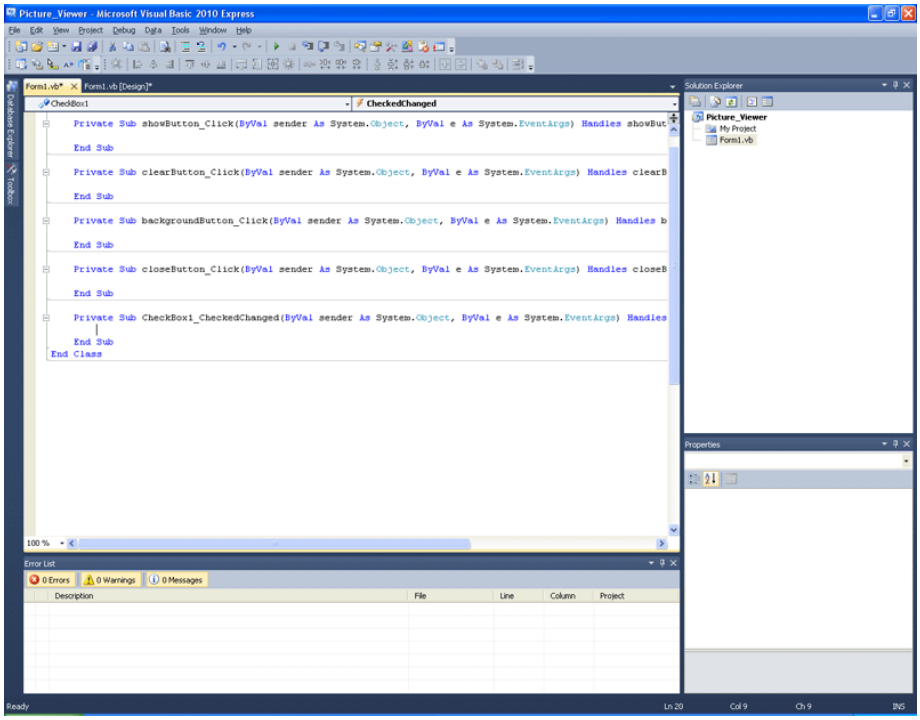
Το "ΣΒΗΣΕ" clearbutton



Το εμφάνισε, το ονομάζουμε showbutton

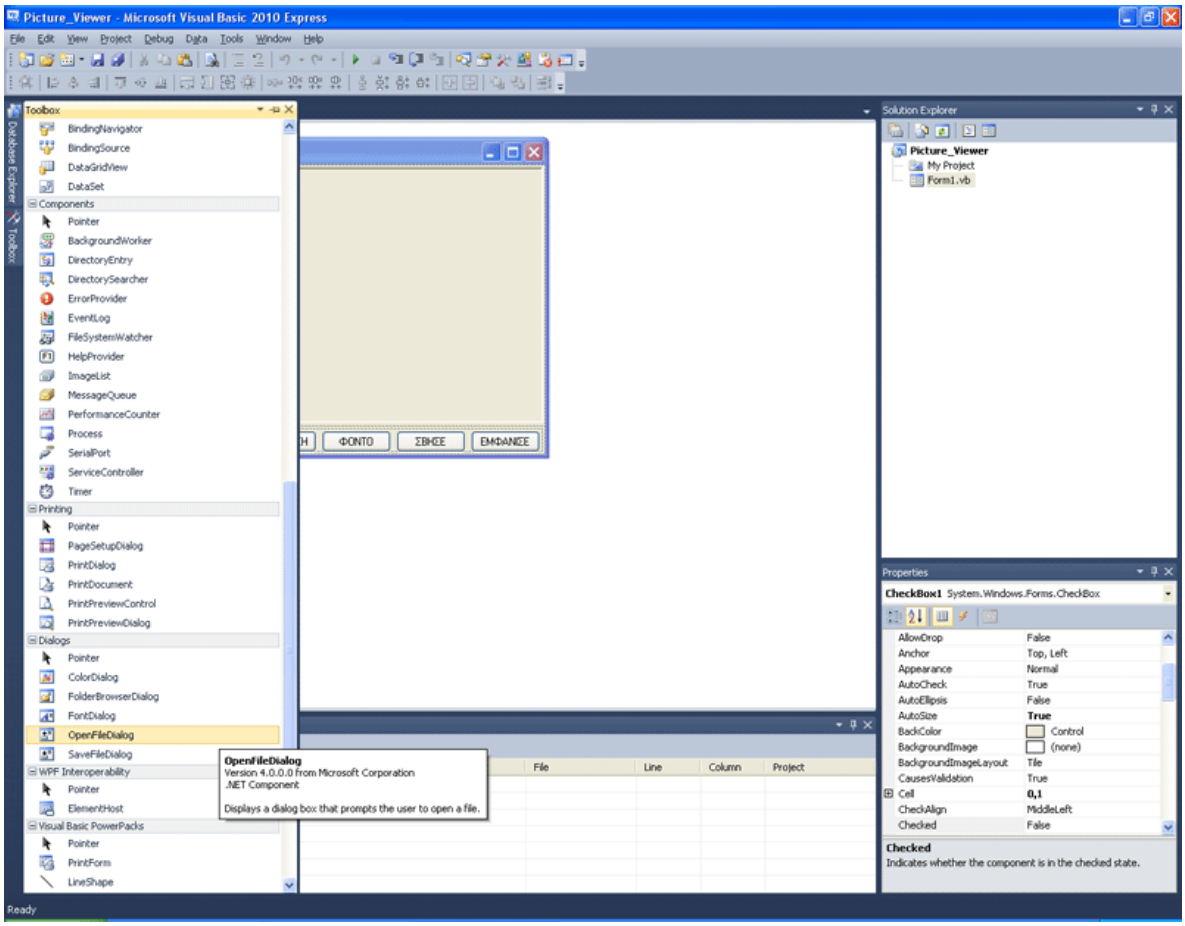


Κάντε διπλό κλικ σε όλα τα buttons και στο checkbox, για να φτιαχτούν τα αντίστοιχα Subs

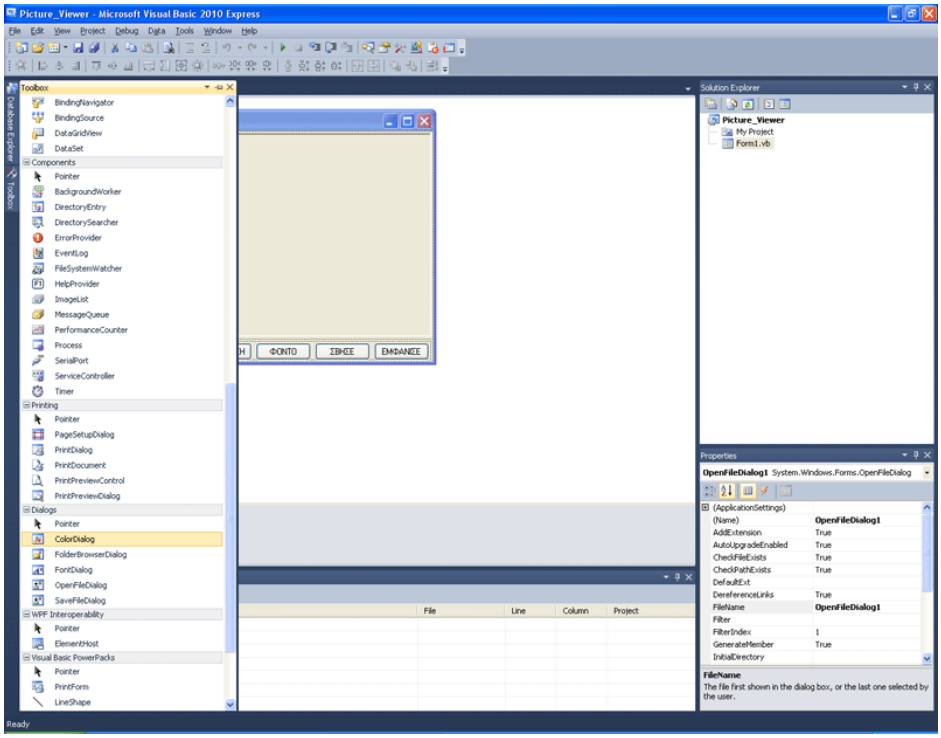


Ήρθε η ώρα να δείτε λίγη από τη δύναμη του .NET περιβάλλοντος. Το πρόγραμμα που θέλουμε να φτιάξουμε, επιθυμούμε να ανοίγει ένα αρχείο εικόνας. Πώς θα γίνει αυτό; Χρειάζεται πολύ περίπλοκος κώδικας; Χρειάζεται να δημιουργήσουμε γραφικά; Όχι. Είναι όλα έτοιμα από πριν, για εσάς.

Πηγαίνετε στο toolbox στο Dialogs κι επιλέξτε OpenFileDialog. Αυτό δε μπαίνει στη φόρμα, αλλά από κάτω σε μια γκρίζα λωρίδα



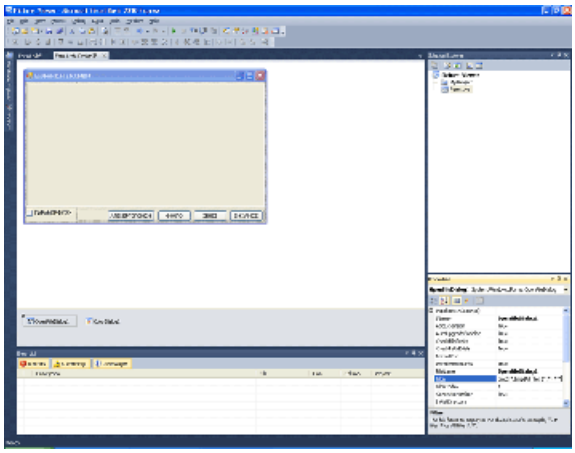
Από το ίδιο μενού επιλέξτε Colordialog. Θα εμφανιστεί κι αυτό στην ίδια λωρίδα



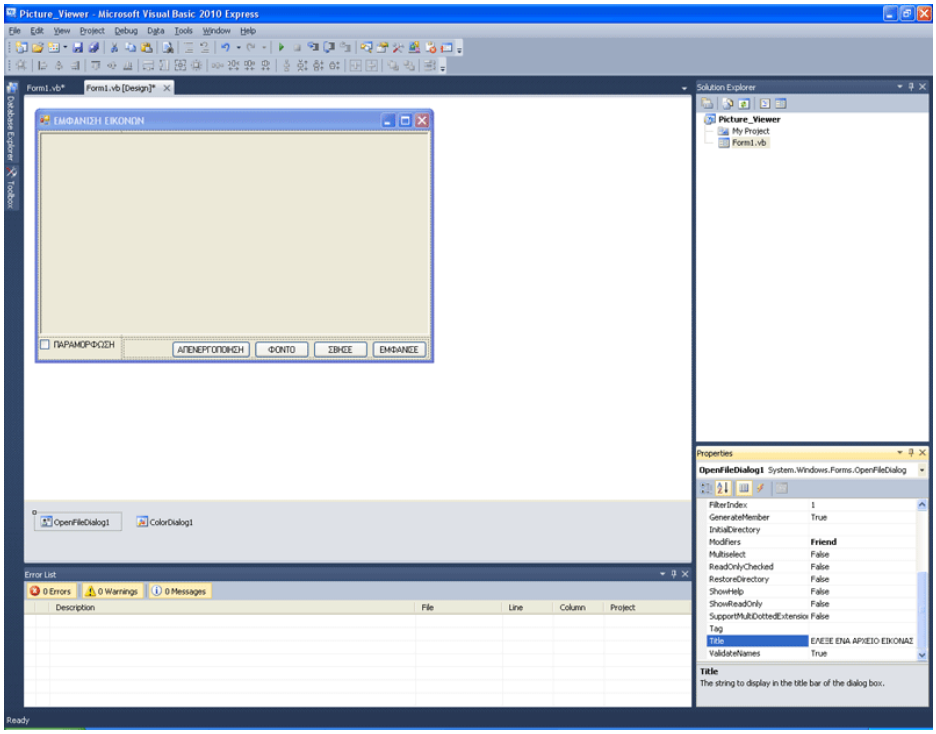
Πατήστε πάνω στο OpenFileDialog1.
Είναι αντικείμενο με ιδιότητες. Επιλέξτε την ιδιότητα Filter. Εδώ θα πρέπει να φτιάξετε ένα φίλτρο για το τι αρχεία θα φορτώνει η εφαρμογή σας. Ένα παράδειγμα φίλτρου, φαίνεται παρακάτω.

JPEG Files (*.jpg)|*.jpg|PNG Files (*.png)|*.png|BMP Files (*.bmp)|*.bmp|All files (*.*)|*.*

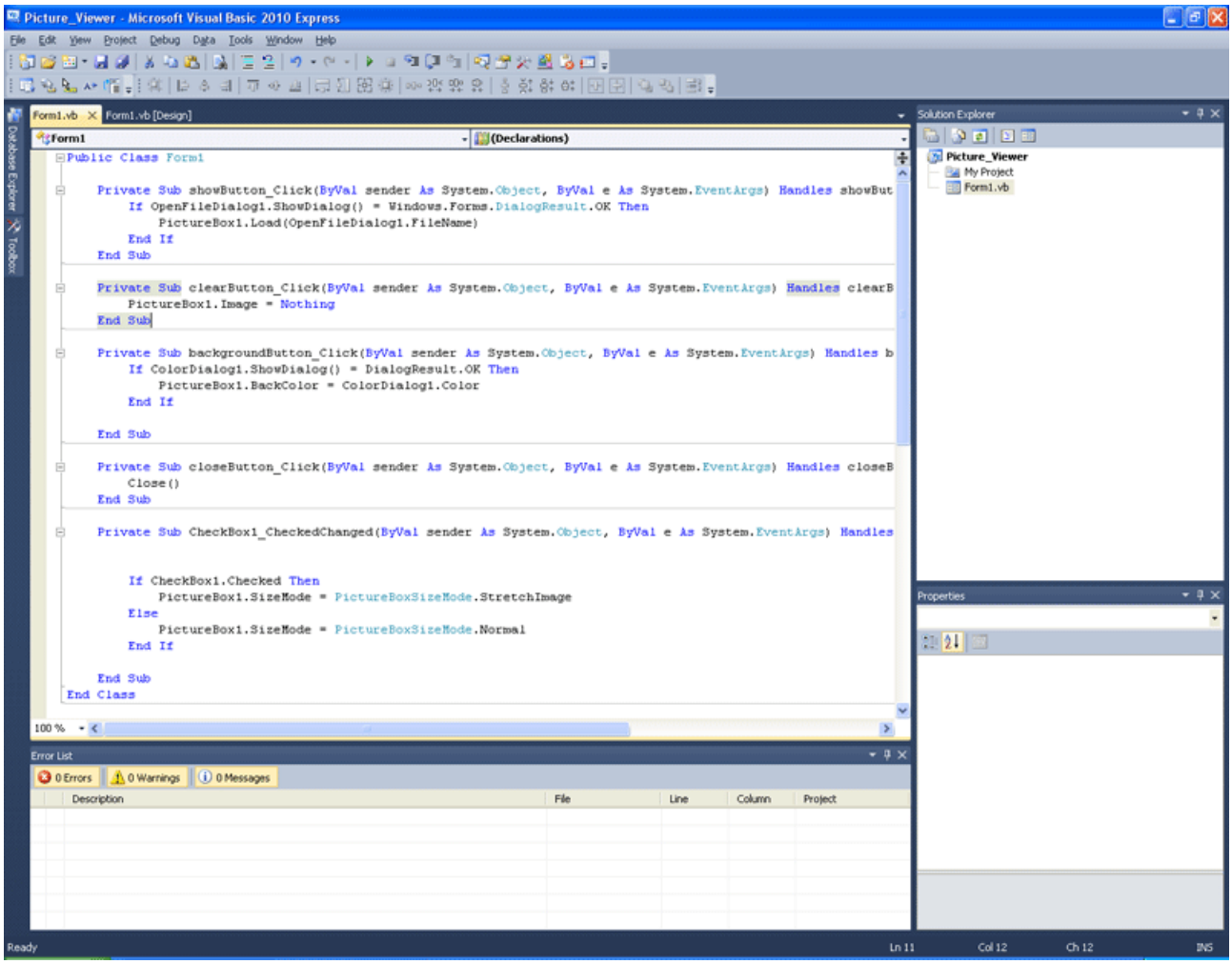
Το φίλτρο ακολουθεί μια τυποποίηση της μορφής: Περιγραφή είδους αρχείου1 (*.κατάληξη1)|*.κατάληξη1|Περιγραφή είδους αρχείου2 (*.κατάληξη2)|*.κατάληξη2| κτλ.



Επίσης ο Dialog μπορεί να ανοίξει για να βρει ο χρήστης ένα αρχείο εικόνας, έτσι ώστε να εμφανιστεί, από το πρόγραμμα. Ωστόσο πώς θα ξέρει ο χρήστης τι να κάνει; Οπότε πάμε στην ιδιότητα Title του OpenFileDialog1 και δίνουμε ως τιμή το κείμενο "ΕΠΙΛΕΞΤΕ ΕΝΑ ΑΡΧΕΙΟ ΕΙΚΟΝΑΣ".



Καιρός να σας δοθεί και ο κώδικας του προγράμματος, καθώς και η ανάλογη επεξήγηση.




```
Public Class Form1
    Private Sub showButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
            Handles showButton.Click
        If OpenFileDialog1.ShowDialog() = Windows.Forms.DialogResult.OK Then
            PictureBox1.Load(OpenFileDialog1.FileName)
        End If
    End Sub

    Private Sub clearButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
            Handles clearButton.Click
        PictureBox1.Image = Nothing
    End Sub
```

```
    Private Sub backgroundButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Handles backgroundButton.Click
      If ColorDialog1.ShowDialog() = DialogResult.OK Then
        PictureBox1.BackColor = ColorDialog1.Color
      End If
    End Sub

    Private Sub closeButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Handles closeButton.Click
      Close()
    End Sub

    Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Handles CheckBox1.CheckedChanged
      If CheckBox1.Checked Then
        PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
      Else
        PictureBox1.SizeMode = PictureBoxSizeMode.Normal
      End If
    End Sub
End Class
```

Κώδικας για το showButton:

```
If OpenFileDialog1.ShowDialog() = Windows.Forms.DialogResult.OK Then
Αν αφού ανοίξει το dialog και πατηθεί το κουμπί OK που βρίσκεται σε αυτό, τότε
PictureBox1.Load(OpenFileDialog1.FileName) το PictureBox φορτώνει μέσα του, το αρχείο
που επέλεξε ο χρήστης, από το OpenFileDialog1
End If
Τέλος της If
==================================
```

Κώδικας για το clearButton:

```
PictureBox1.Image = Nothing
Αδειάζει το PictureBox1. Απλά.
==================================
```

Κώδικας για το backgroundButton:

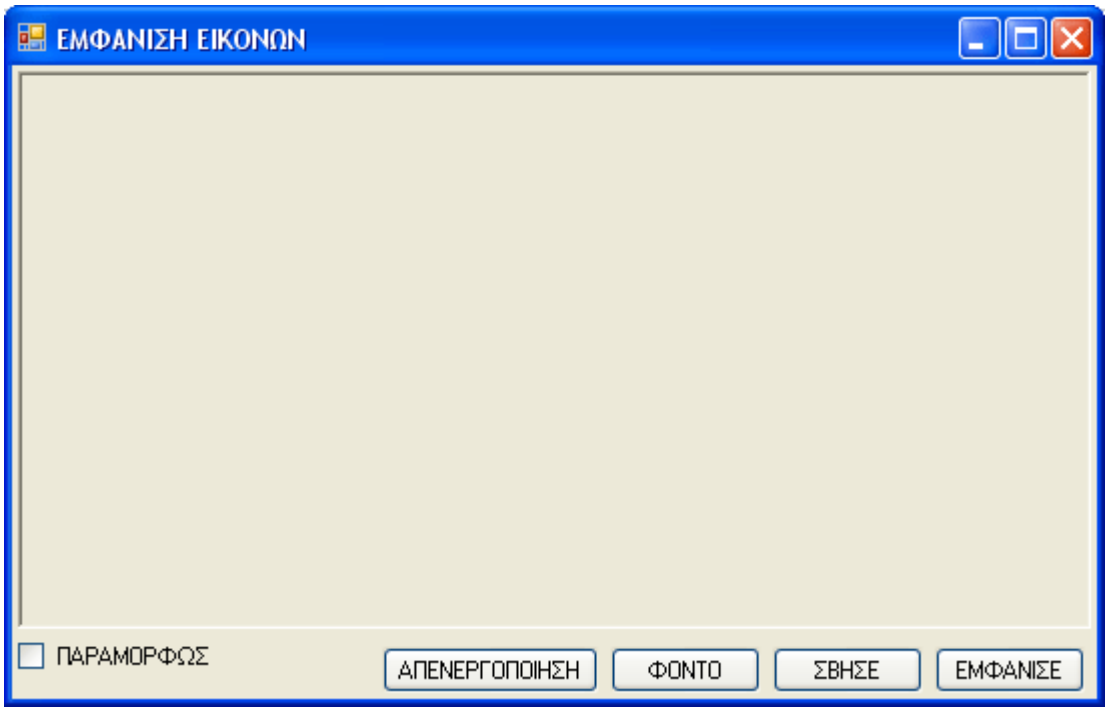
```
If ColorDialog1.ShowDialog() = DialogResult.OK Then
Αν στο ColorDialog1 πατηθεί το κουμπί OK τότε
PictureBox1.BackColor = ColorDialog1.Color
το χρώμα του φόντου γίνεται αυτό που επέλεξε ο χρήστης.

End If
Τέλος της If
==================================
```

Κώδικας για το closeButton:
Close()
Συνάρτηση που κλείνει το πρόγραμμα.
==========

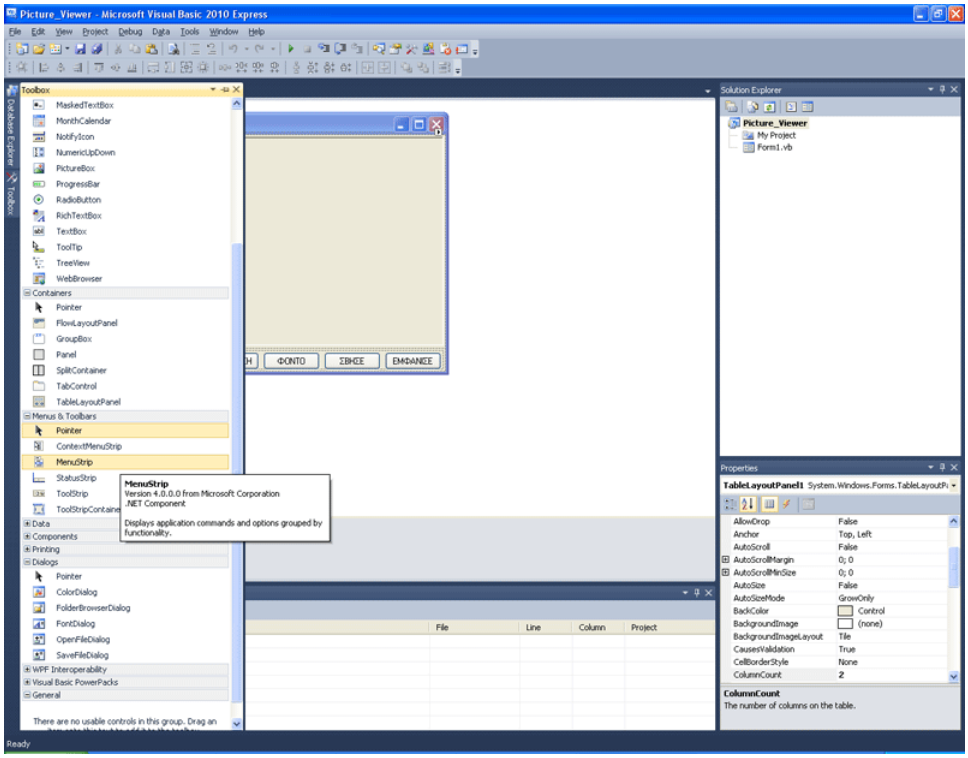
Κώδικας για το CheckBox1:
If CheckBox1.Checked Then
Άν έχει το ν στο checkbox τότε
PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
η εικόνα θα παραμορφώνει ώστε να γεμίζει το πλαίσιο.
Else
αλλιώς
PictureBox1.SizeMode = PictureBoxSizeMode.Normal
θα εμφανίζεται σε φυσιολογικό μέγεθος.
End If
Τέλος της If.
==========

Τρέξτε το πρόγραμμα! Όμορφο ε;

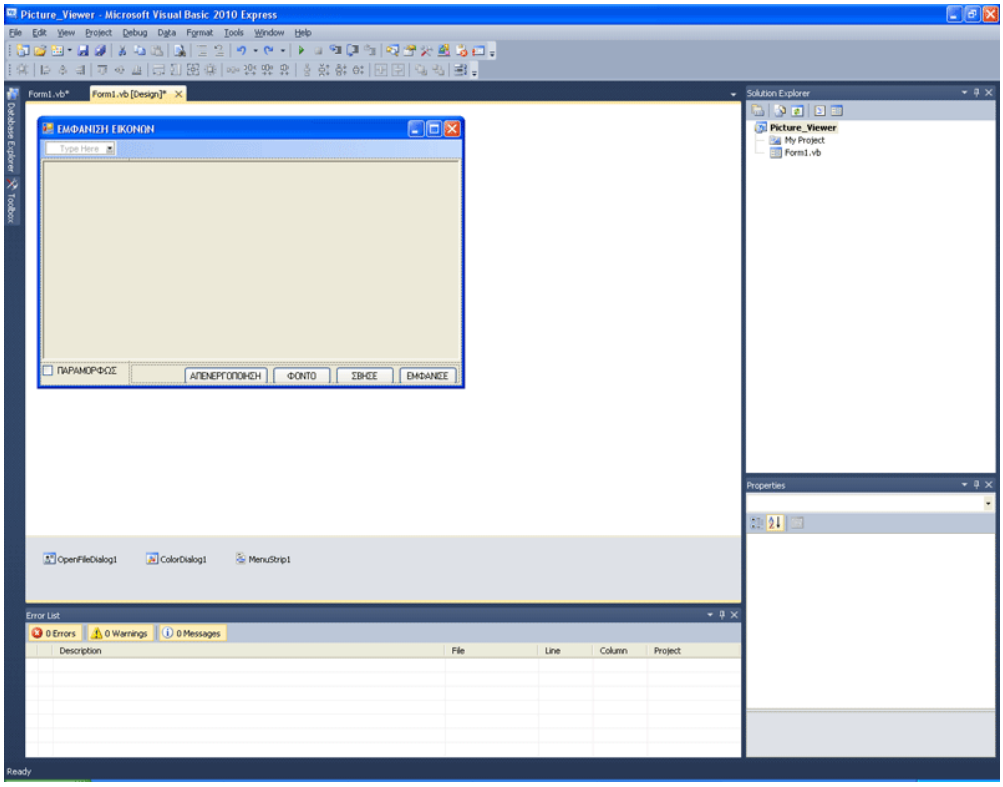
Τι θα λέγατε τώρα να σας μαθαίναμε να φτιάχνετε και μενού; Μοιάζει με τεράστιο άλμα γνώσεων, αλλά στην ουσία είναι πολύ απλό.



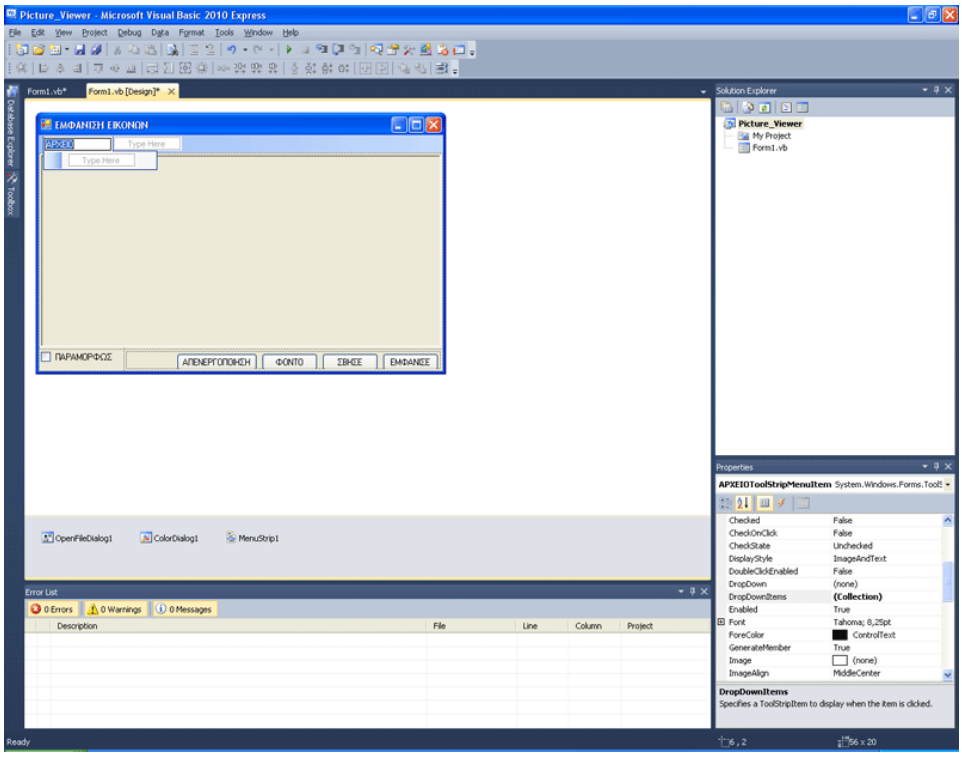
Όπως βλέπετε στην παρακάτω εικόνα, πάμε στο Toolbox, βρίσκουμε την ομάδα Menus & Toolbars κι επιλέγουμε να προσθέσουμε ένα MenuStrip. Βέλτιστα εμφανίζεται από μόνο του, πάνω από όλα τα containers σα μια μπλε λωρίδα.



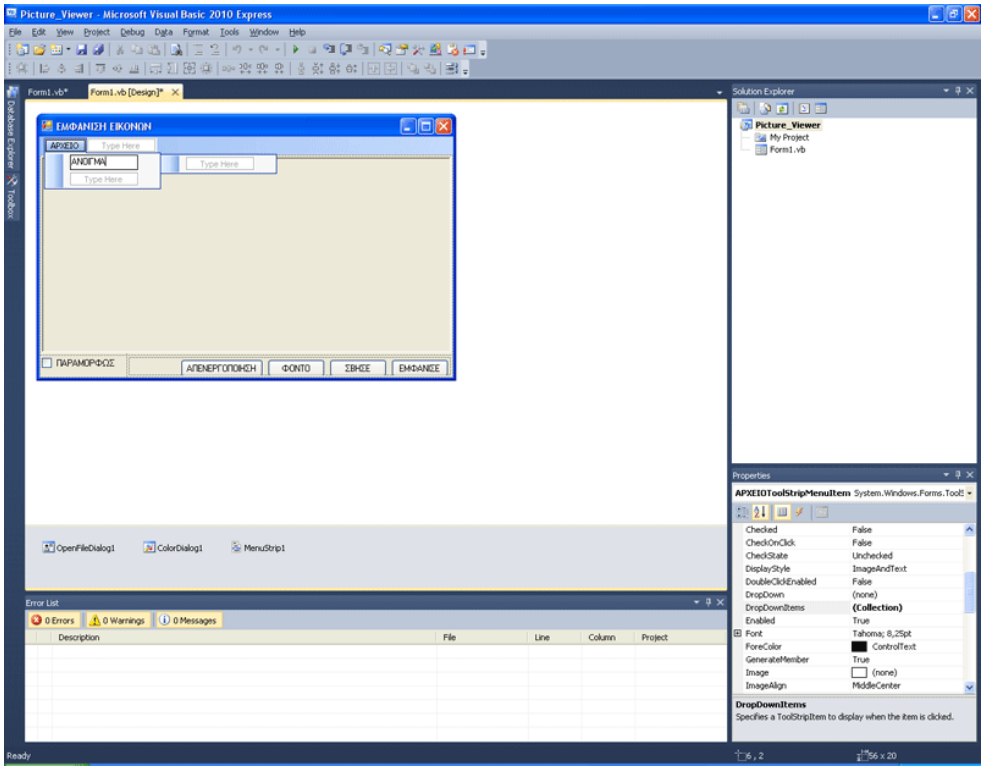
Κλικ στο Type Here, για να γράψουμε τίτλο του πρώτου μενού, δημιουργεί δίπλα αυτομάτως ακόμα ένα δυνητικά υπαρκτό. Κοινώς, αν δε γράψουμε στο διπλανό κάτι, δε θα υπάρξει.



Επίσης αναδύεται το μενού που φτιάχνουμε, ώστε να γράψουμε μέσα του, τις διάφορες επιλογές. Ονομάστε το πρώτο αυτό μενού, "ΑΡΧΕΙΟ"



Μέσα στο μενού πλέον, γράφετε την επιλογή "ΑΝΟΙΓΜΑ". Προσέξτε πως εμφανίζεται από κάτω μια ακόμα δυνητική επιλογή, για να σας διευκολύνει στη δημιουργία του μενού. Ονομάστε την επιλογή "ΑΝΟΙΓΜΑ"



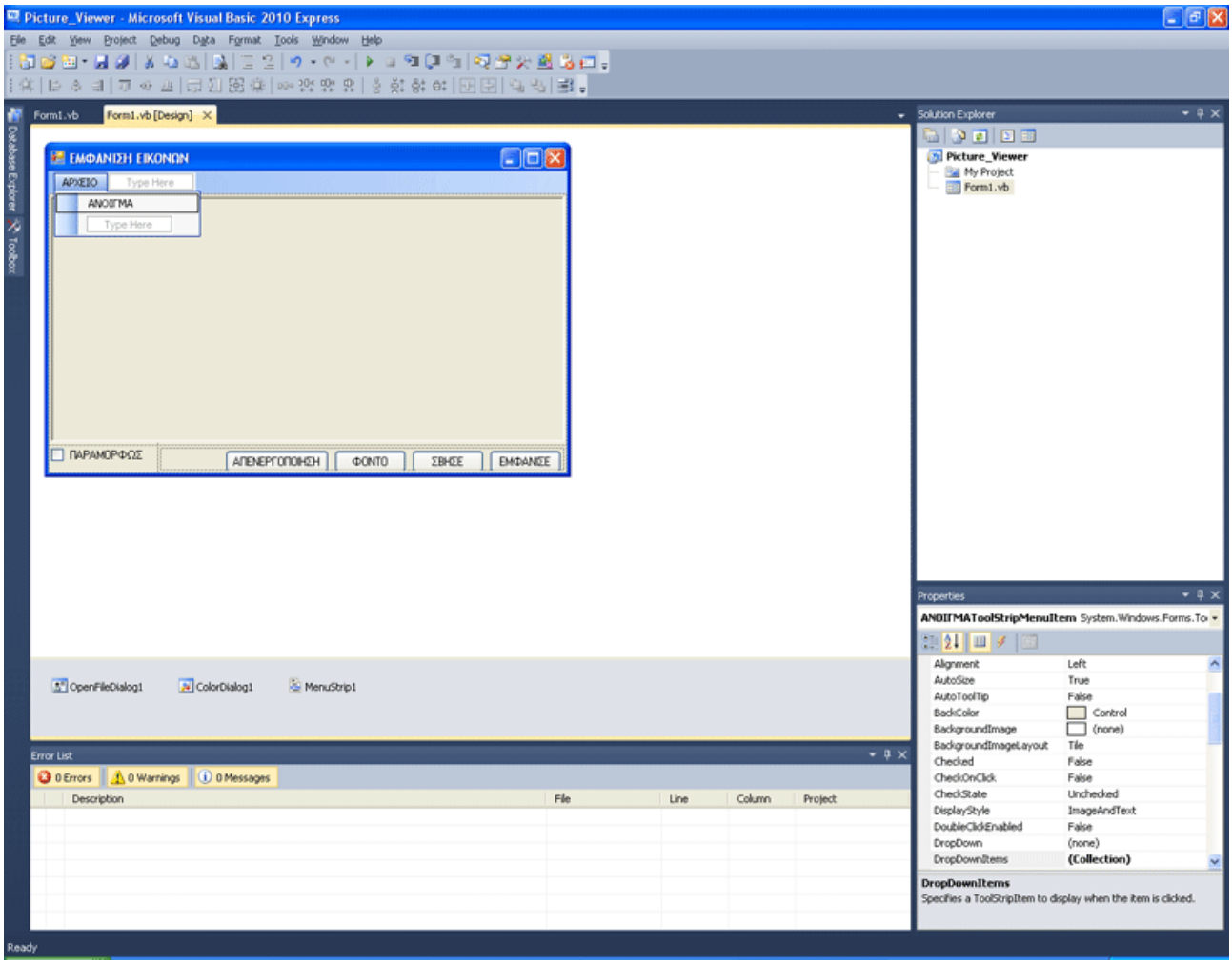
Τελειώσαμε προς το παρόν με τη σχεδίαση.
Κάντε πάνω στην επιλογή μενού "ΑΝΟΙΓΜΑ" διπλό κλικ. Παρατηρείτε τι έγινε;

Private Sub ΑΝΟΙΓΜΑToolStripMenuItem()
End Sub

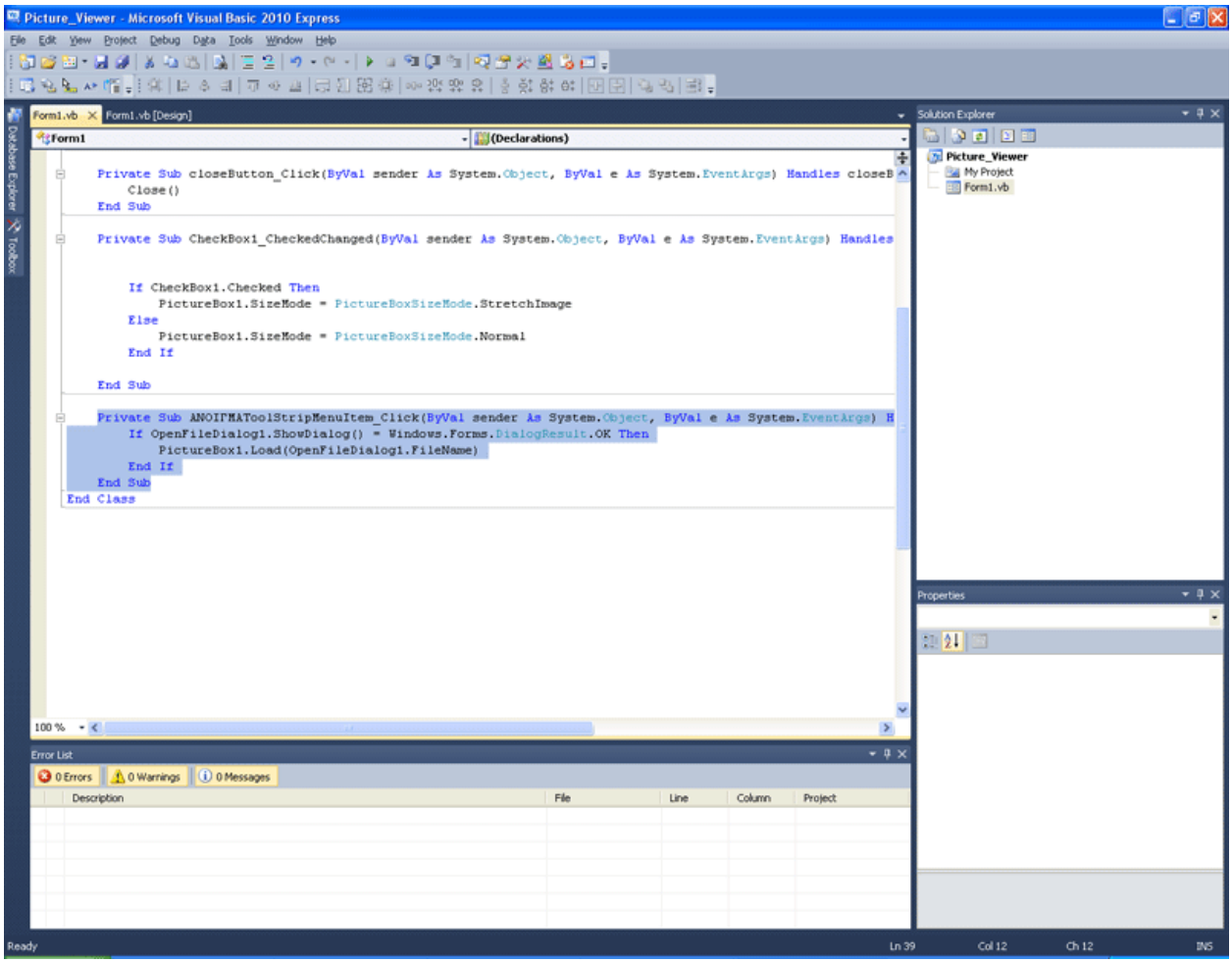
Βάζουμε τώρα μέσα του, τον κώδικα:

If OpenFileDialog1.ShowDialog() = Windows.Forms.DialogResult.OK Then
PictureBox1.Load(OpenFileDialog1.FileName)
End If

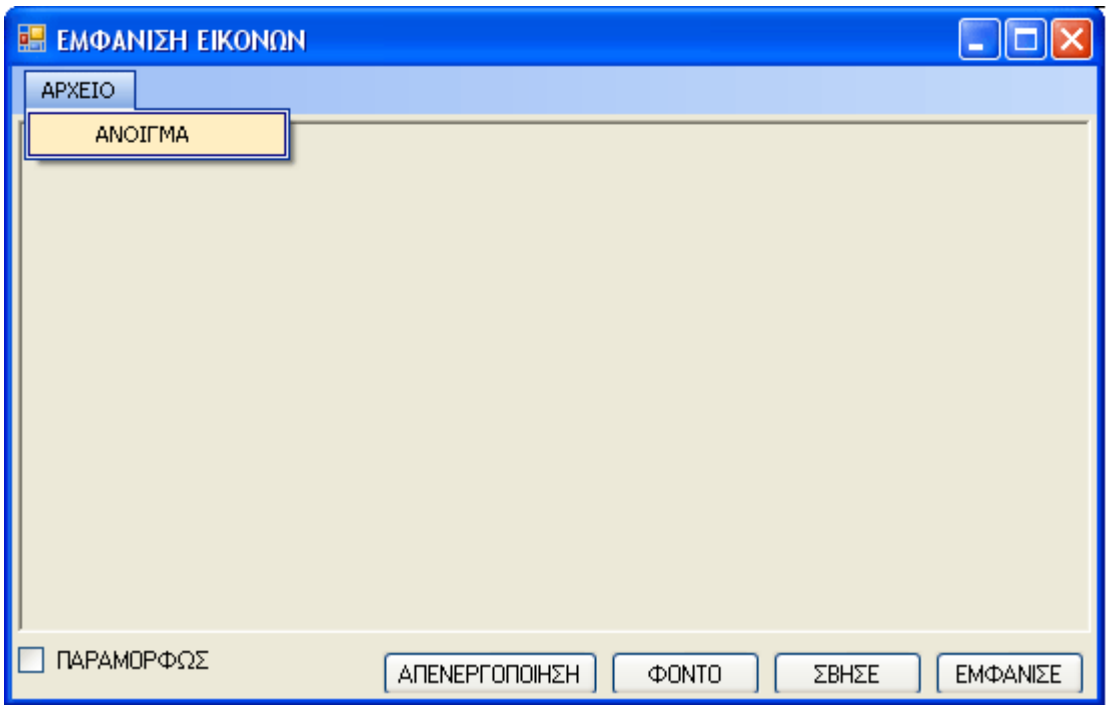
ΠΑΡΑΤΗΡΗΣΗ:
Είναι ακριβώς ο κώδικας του Private Sub showbutton_click ()



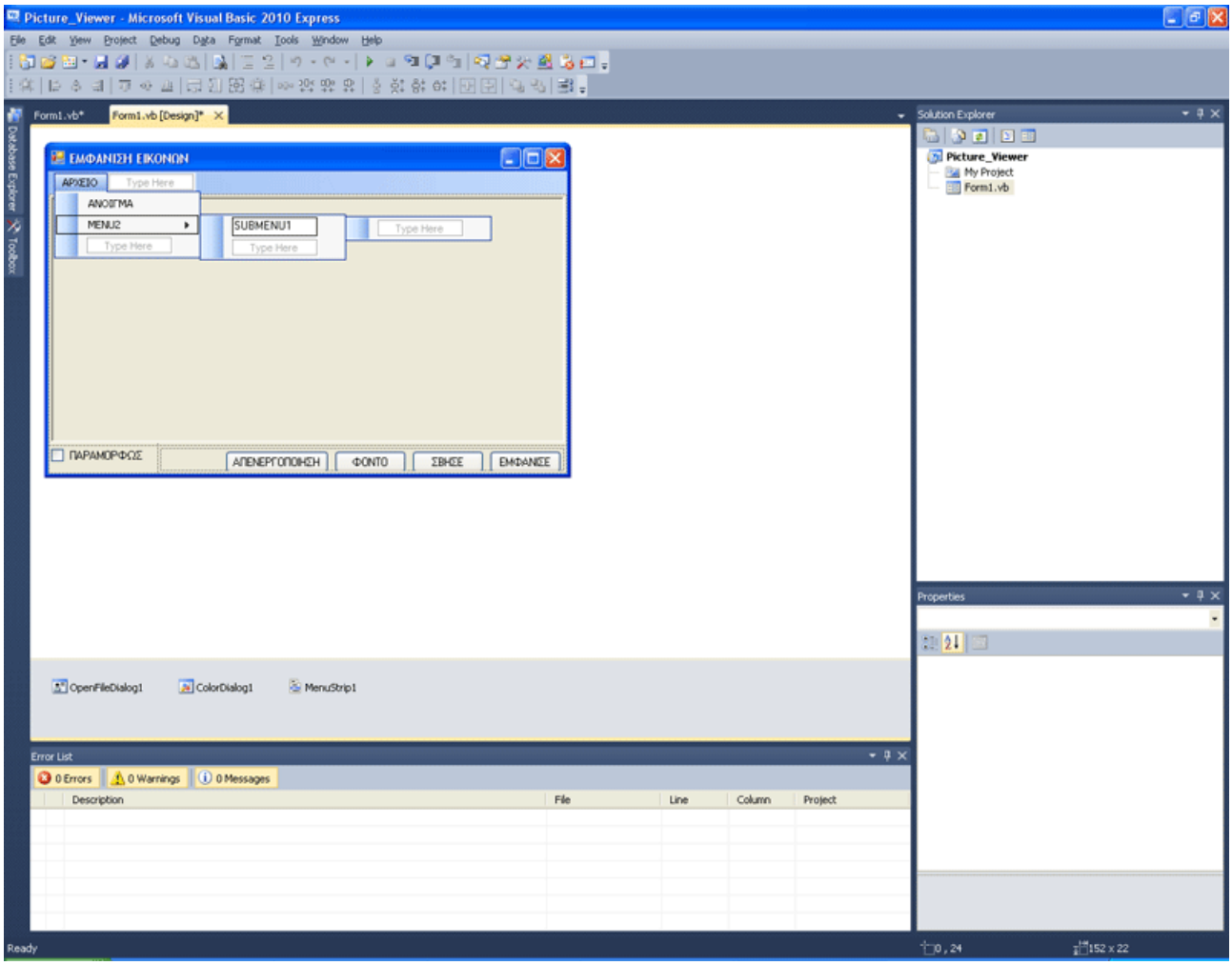
Έτσι πρέπει να φαίνεται ο κώδικας, αν το κάνατε σωστά



Τρέξτε το πρόγραμμα και δοκιμάστε το μενού



Δεν ξεχάσαμε να σας δείξουμε το πως φτιάχνονται τα υπομενού. Αν δεν το βρήκατε ήδη με δικό σας πειραματισμό, σας το δείχνουμε παρακάτω. Τα αντικείμενα μενού αυτόματα αποκτούν δυνητικά αντικείμενα, σε υπομενού. Το MENU2 είναι ένα αντικείμενο του μενού ΑΝΟΙΓΜΑ. Το SUBMENU1 είναι αντικείμενου ενός υπομενού που ανοίγει, όταν πάει ο κέρσορας στο MENU2.



ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

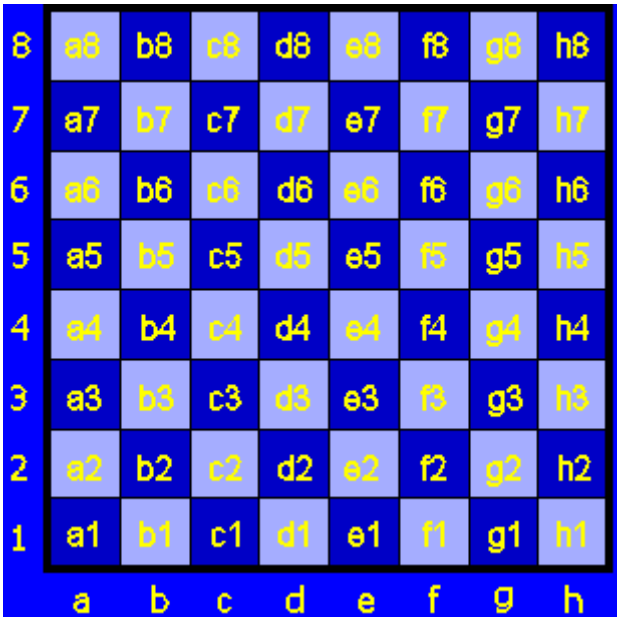
Το κεφάλαιο αυτό δίνει στο χρήστη νέες γνώσεις πάνω στο πώς να χειρίζεται και να τοποθετεί τα αντικείμενά του στις φόρμες, επεξηγεί τα containers καθώς και αναλύει εκτενώς τον τρόπο με τον οποίο μπορεί να φτιαχτεί ένα μενού σε μια εφαρμογή. Επίσης για πρώτη φορά μέχρι στιγμής στο βοήθημα, χρησιμοποιείται Dialog το οποίο ανοίγει αρχείο.

ΚΕΦΑΛΑΙΟ 16

ΠΙΝΑΚΕΣ, ΑΡΧΕΙΑ

Σε αυτό το κεφάλαιο, θα μιλήσουμε για δύο θέματα, κάπως πιο προχωρημένα στον προγραμματισμό. Για αρχή θα ασχοληθούμε εδώ με διαχείριση πινάκων. Αρχικά λοιπόν, ας δούμε τι είναι πίνακας. Πίνακας, είναι μια δομή δεδομένων. Θα αναρωτιέστε τώρα τι είναι δομή δεδομένων. Ας αφήσουμε λοιπόν στην άκρη για λίγο τον ορισμό του πίνακα κι ας δούμε τι είναι μια δομή δεδομένων.

Ας υποθέσουμε πως επισκέπτεστε για κάποιο λόγο μια Δημόσια Αρχή. Πρέπει να περιμένετε λοιπόν στην ουρά για να εξυπηρετηθείτε. Τι κάνετε; Παίρνετε ένα χαρτάκι, που έχει έναν αριθμό προτεραιότητας και περιμένετε να έρθει η σειρά σας. Όταν έρθει η σειρά σας, εξυπηρετείστε. Αν όλοι οι άνθρωποι που βρίσκονταν εκεί, δεν ακολουθούσαν αυτό τον απλό κανόνα, θα ήταν όλοι μπουλούκι, με αποτέλεσμα να μη διεξαχθεί τίποτα σωστά. Δε θα μπορούσε κανείς να εξυπηρετηθεί. Αντιθέτως, έχοντας έναν αριθμό προτεραιότητας, οι άνθρωποι δημιουργούν μια δομή που λέγεται ουρά. Φανταστείτε τώρα, ότι δεν έχουμε να κάνουμε με ανθρώπους, αλλά με δεδομένα. Μια δομή δεδομένων λοιπόν, είναι η ουρά. Μπαίνουν σε ένα δοχείο, για παράδειγμα αριθμοί, και αναλόγως τους κανόνες, βγαίνουν αντίστοιχα με κάποια σειρά. Υπάρχει λοιπόν, πέραν της ουράς, μία πολύ πρακτική δομή δεδομένων, που λέγεται πίνακας.
Φανταστείτε πως πρέπει να στοιχηθείτε εσείς και άλλοι δέκα άνθρωποι, σε δύο γραμμές των πέντε ατόμων. Είναι περίπου σαν στοίχιση για παρέλαση, ή στοίχιση για πρωινή μαθητική προσευχή. Φανταστείτε όμως, ότι αυτή η στοίχιση και τοποθέτηση, γίνεται ονομαστικά. Δηλαδή, δε στοιχίζετε όπως να' ναι, αλλά όπως θέλει ο στοιχιστής. Δεν αλλάζετε ποτέ θέση. Αυτό είναι μία δομή πίνακα, εφαρμοσμένη σε στοίχιση ανθρώπων.
Τι έχουμε δηλαδή σε μια δομή πίνακα; Δύο πράγματα: Γραμμές και Στήλες. Κάθε στοιχείο ενός πίνακα λοιπόν, ανήκει σε συγκεκριμένη γραμμή και στήλη. Κοινώς έχει συντεταγμένες. Κοιτάξτε την παρακάτω εικόνα.
Απεικονίζει μια σκακιέρα.



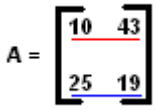
Στο σκάκι, οι παρτίδες γράφονται με συγκεκριμένο τρόπο. Αν κι έχει διάφορες παραλλαγές, ένα από τα κεντρικά στοιχεία γραφής σκάκι, είναι η αναφορά σε τετράγωνα της σκακιέρας. Η σκακιέρα έχει: 8 στήλες a,b,c,d,e,f,g,h και επίσης
8 γραμμές 1,2,3,4,5,6,7,8
Εδώ είναι λίγο ανάποδος ο συμβολισμός από ότι θα συμβολίσουμε εμείς τον πίνακα. Ωστόσο, είναι σαφές ότι έχουμε να κάνουμε με πίνακα. Κοινώς, η μόνη διαφορά είνα, ότι στις στήλες δε βάλαμε αριθμούς, αλλά γράμματα.

Οπότε αν η σκακιέρα συμβολίζεται με Σ κάθε τετράγωνο μπορεί να συμβολιστεί με τη μορφή Σ[στήλη,γραμμή]. Με αυτό τον τρόπο γίνεται στο σκάκι.
Εμείς όμως θα το εκφράσουμε διαφορετικά, με τη μορφή Σ[γραμμή,στήλη]. Γιατί όμως το κάνουμε αυτό; Στόχος μας είναι να σας κάνουμε στη συνέχεια μια μικρή και απλή εισαγωγή στο τι είναι πίνακας στα μαθηματικά. Αν κατανοήσετε την έννοια, το πως θα υλοποιήσουμε έναν πίνακα σε ένα πρόγραμμα και το πως θα τον χειριστούμε, θα γίνουν εύκολα κατανοητά.
Το τετράγωνο λοιπόν της σκακιέρας, που ανήκει στη γραμμή 3 και τη στήλη e είναι το Σ[3e]. Είναι ένα από τα 8 χ 8 = 64 τετράγωνα της σκακιέρας.

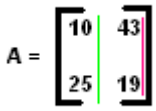
Τώρα φανταστείτε ότι δεν έχουμε γράμματα στις στήλες, αλλά αριθμούς πάλι.
Δηλαδή: a=1, b=2, c=3, d=4, e=5, f=6, g=7, h=8.
Οπότε οι συντεταγμένες του ίδιου τετραγώνου είναι Σ[3,5]
Εδώ βάζουμε κόμμα για να δηλώσουμε ότι έχουμε δύο αριθμούς κι όχι έναν.
Κοινώς έχουμε το 3 και το 5. Όχι το 35.

Και τώρα ας πάμε σε λίγα απλά μαθηματικά.
Ορισμός:
Πίνακας τύπου μχν ονομάζεται μία διάταξη αριθμών πλήθους μ επί ν, με μ γραμμές και ν στήλες. Οι αριθμοί λέγονται στοιχεία του πίνακα. Ο πίνακας συμβολίζεται με ένα κεφαλαίο γράμμα, (π.χ. Α).

Δείτε την παρακάτω εικόνα.
Ο Α είναι ένας πίνακας 2χ2 (δύο επί δύο).
Με κόκκινο και μπλε χρώμα δείχνουμε τις γραμμές του.



Με πράσινο και κόκκινο στην παρακάτω εικόνα δείχνουμε τις στήλες του.



Ο πίνακας 1χ1 με μία γραμμή και μία στήλη, έχει μόνο ένα στοιχείο, οπότε λέγεται και πίνακας στοιχείο. Ο πίνακας με πολλές στήλες και μία γραμμή 1 χ ν, λέγεται πίνακας γραμμή. Ο πίνακας με πολλές γραμμές και μία στήλη μ χ 1, λέγεται πίνακας στήλη. Αυτοί οι πίνακες, που το μ ή το ν τους, ισούται με 1, λέγονται και μονοδιάστατοι, μιας και η μία διάστασή τους, είναι σταθερή κι ίση με 1, κάνοντάς τον γραμμή ή στήλη. Οι πίνακες μ χ ν αντιθέτως, λέγονται πίνακες δύο διαστάσεων. Αναφερόμενοι σε έναν μ επί ν πίνακα Α (Αμχν) έστω ότι θέλουμε να συμβολίσουμε ένα στοιχείο του. Το στοιχείο αυτό θα ανήκει στη γραμμή i και τη στήλη j. Σαφώς ισχύει ότι:
i <= μ
και ότι
j <= ν
Επίσης μ,ν,i,j είναι ακέραιοι. Δε μπορείς να έχεις 1,345 γραμμές...

Οπότε συμβολίζουμε το στοιχείο της i γραμμής και j στήλης ως a[ij]

$$a_{ij}$$

Το a23 δηλαδή, είναι το στοιχείο του πίνακα Α
το οποίο ανήκει στη δεύτερη γραμμή και στην τρίτη στήλη.

$$a_{23}$$

Το b23 στον παρακάτω πίνακα Β, είναι πολύ απλά ο αριθμός 6.
Το β32 είναι το 8
Το β13 είναι το 9

$$\begin{bmatrix} 2 & 7 & 9 \\ 3 & 4 & 6 \\ 12 & 8 & -2 \end{bmatrix} = B$$

Κατανοητό; Ωστόσο πολλοί θα αναρωτιέστε...ποια είναι η χρησιμότητα του πίνακα; Λοιπόν, υπάρχει ένας βασικός κανόνας για τα δεδομένα. Δεδομένα δίχως τάξη, χύμα δηλαδή, δεν έχουν χρησιμότητα.Φανταστείτε σκόρπιες νότες μουσικής. Δεν έχουν νόημα. Τοποθετώντας τις στο πεντάγραμμο, δομούν άμεσα μια μελωδία. Οπότε, ο πίνακας, η ουρά, και άλλες δομές δεδομένων, δίνουν προστιθέμενη αξία στα δεδομένα μας. Δείτε την παρακάτω εικόνα. Εξαιρώντας τα τετράγωνα με τις μέρες και τις ώρες, έχουμε έναν πίνακα έτοιμο να συμπληρωθεί με το εβδομαδιαίο πρόγραμμά σπουδών ενός φοιτητή.

Ο πίνακας αυτός, έχει πέντε στήλες...

	ΔΕΥΤΕΡΑ	ΤΡΙΤΗ	ΤΕΤΑΡΤΗ	ΠΕΜΠΤΗ	ΠΑΡΑΣΚΕΥΗ
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					

...και 14 γραμμές! Είναι δηλαδή ένας 14 επί 5 πίνακας.
Αν γράφατε σε ένα τετράδιο σκόρπια τα μαθήματα που παρακολουθείτε, δίχως τις μέρες και ώρες, αυτό δε θα έβγαζε νόημα. Αν γράφατε σκόρπια κάθε μάθημα με μέρα και ώρα, αυτό δε θα ήταν και πολύ πρακτικό. Αντί να επαναλαμβάνεται η μέρα και ώρα σε κάθε εγγραφή, η δομή ορίζει μέρα και ώρα, ανάλογα που θα γράψουμε το μάθημα. Καταλάβατε τώρα σε τι μπορούν να χρησιμεύσουν οι πίνακες;

	ΔΕΥΤΕΡΑ	ΤΡΙΤΗ	ΤΕΤΑΡΤΗ	ΠΕΜΠΤΗ	ΠΑΡΑΣΚΕΥΗ
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					

Βέβαια η χρησιμότητά τους, αλλά και οι εφαρμογές τους δε σταματάνε εδώ. Ορίστε ένα μισθολόγιο. Έχει 12 γραμμές, μία για κάθε μήνα...

	ΜΙΣΘΟΙ ΥΠΑΛΛΗΛΩΝ	
	ΣΟΦΙΑ	ΓΙΩΡΓΟΣ
ΙΑΝΟΥΑΡΙΟΣ		
ΦΕΥΡΟΥΑΡΙΟΣ		
ΜΑΡΤΙΟΣ		
ΑΠΡΙΛΙΟΣ		
ΜΑΙΟΣ		
ΙΟΥΝΙΟΣ		
ΙΟΥΛΙΟΣ		
ΑΥΓΟΥΣΤΟΣ		
ΣΕΠΤΕΜΒΡΙΟΣ		
ΟΚΤΩΜΒΡΙΟΣ		
ΝΟΕΜΒΡΙΟΣ		
ΔΕΚΕΜΒΡΙΟΣ		

και δύο στήλες, μία για κάθε εργαζόμενο. Είναι κοινώς ένας 12 επί 2 πίνακας.

	ΜΙΣΘΟΙ ΥΠΑΛΛΗΛΩΝ	
	ΣΟΦΙΑ	ΓΙΩΡΓΟΣ
ΙΑΝΟΥΑΡΙΟΣ		
ΦΕΥΡΟΥΑΡΙΟΣ		
ΜΑΡΤΙΟΣ		
ΑΠΡΙΛΙΟΣ		
ΜΑΙΟΣ		
ΙΟΥΝΙΟΣ		
ΙΟΥΛΙΟΣ		
ΑΥΓΟΥΣΤΟΣ		
ΣΕΠΤΕΜΒΡΙΟΣ		
ΟΚΤΩΜΒΡΙΟΣ		
ΝΟΕΜΒΡΙΟΣ		
ΔΕΚΕΜΒΡΙΟΣ		

Τελειώνοντας με την επεξήγηση των πινάκων, θέλουμε να είμαστε σίγουροι ότι κατανοείτε τη δομή, αλλά και την αναφορά σε ένα στοιχείο.

Κοιτάξτε την παρακάτω εικόνα

1	2	a11	a12
3	9	a21	a22

Εδώ έχουμε έναν πίνακα Α2χ2

Το στοιχείο a11 είναι το 1
Το στοιχείο a12 είναι το 2
Το στοιχείο a21 είναι το 3
Το στοιχείο a22 είναι το 9

Κατανοητό; Κρατήστε την εικόνα της δομής λοιπόν κατά νου. Μία από τις μεγαλύτερες εφαρμογές των πινάκων είναι η κρυπτογράφηση. Επίσης τεράστιες εφαρμογές έχουν στις τεχνικές data mining, κατά τις οποίες μεγάλη μάζα δεδομένων, δομημένη σε πίνακες με ιδιαίτερο τρόπο, μπορούν να εξορύξουν επιπλέον και πολύ χρήσιμες πληροφορίες. Το να σας δείξουμε αυτές τις εφαρμογές όμως, προαπαιτεί πολύ καλή γνώση πινάκων, γραμμικής άλγεβρας και διαφόρων άλλων επιστημονικών γνώσεων. Καταλαβαίνετε λοιπόν πως αυτό θα απόκλινε από το στόχο του βοηθήματος αυτού, που δεν είναι άλλος από το να σας εισάγει και να σας διδάξει ομαλά τη Visual Basic 2010. Ωστόσο σας προσφέρουμε την άκρως απαραίτητη αρχική κατάρτιση, για να κατανοήσετε προγραμματιστικά το θέμα. Επιπλέον επιστημονικές εφαρμογές, που πιθανώς θα μελετήσετε, θα μπορέσετε να τις εφαρμόσετε άμεσα, γνωρίζοντας αυτά που ήδη σας παρουσιάσαμε και αυτά που θα σας δείξουμε στη συνέχεια.

Στη Visual Basic ο πίνακας, είναι μια συλλογή από μεταβλητές με δείκτες. Αρχικά να υπενθυμίσουμε, πως ο 1χ1 πίνακας, που έχει μόνο ένα στοιχείο, λέγεται και πίνακας στοιχείο. Όταν ορίζατε ουσιαστικά μεταβλητές μέχρι τώρα στη Visual Basic, αυτό κάνατε. Ορίζατε πίνακες στοιχεία. Για να δώσετε διαστάσεις σε ένα πίνακα, πέραν του 1χ1, κάνετε το εξής:

Dim Pinakasmou(9) As Integer
εναλλακτικά γράφετε το παρακάτω, το ίδιο κάνει:
Dim Pinakasmou(0 To 9) As Integer

Τι κάνει όμως αυτή η εντολή;
Δημιουργεί έναν μονοδιάστατο πίνακα, με όνομα pinakasmou, με δέκα μεταβλητές, που δέχονται ακεραίους αριθμούς.
Προσέξτε αυτό:
Ο δείκτης ενός πίνακα ξεκινάει από το μηδέν.
Οπότε 0,1,2,3,4,5,6,7,8,9 είναι δέκα δείκτες για δέκα μεταβλητές.
Προσέξτε και αυτό:
Τα στοιχεία του πίνακα στη Visual Basic, δεν είναι αριθμοί όπως στα μαθηματικά. Είναι μεταβλητές.
Θυμάστε το παράδειγμα που λέγαμε πως μια κανάτα είναι μεταβλητή τύπου Double; Φανταστείτε ένα μεγάλο κιβώτιο με στοιχισμένες πολλές κανάτες μέσα του! Αυτό είναι ένας πίνακας τύπου Double. Ωστόσο σε ένα πίνακα, οι μεταβλητές, είναι πάντοτε ιδίου τύπου. Απαιτείται μεγάλη λοιπόν προσοχή στον ορισμό τύπου δεδομένων στον πίνακα, όπως και στις μεταβλητές. Θα αναρωτιέστε τώρα, γίνεται να έχουμε πίνακα τύπου string; Σαφώς και γίνεται!
Ένας πίνακας δύο διαστάσεων ορίζεται πολύ απλά ως:
Dim twodimention(2,5) As Integer
2 γραμμές και 5 στήλες

Τριών διαστάσεων αντίστοιχα:
Dim triwndiastasewn(5,7,9) As Integer

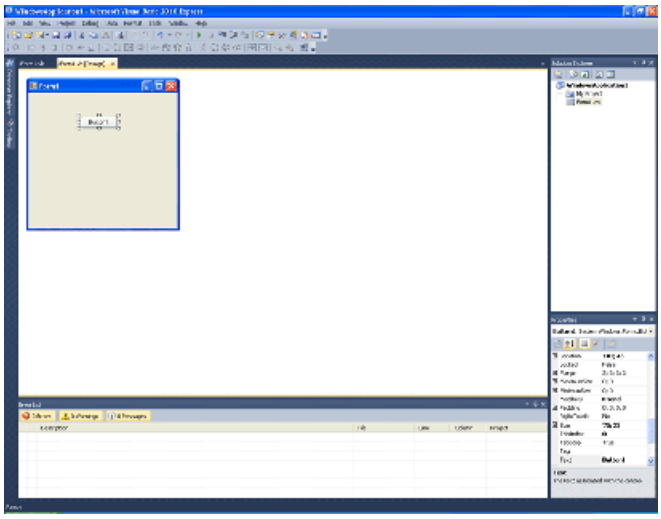
Τώρα ας δούμε όμως τον τρόπο που δίνουμε τιμές σε ένα στοιχείο του πίνακα.
Κοινώς σε μια μεταβλητή. Έστω ότι έχω τον μονοδιάστατο πίνακα Α(9) τύπου Integer, που περιέχει δέκα στοιχεία.
Αυτά είναι τα Α(0), Α(1), Α(2), Α(3), Α(4), Α(5), Α(6), Α(7), Α(8), Α(9).
Οπότε αν θέλω να δώσω τιμή σε κάποιο στοιχείο, θα κάνω ότι έκανα και για απλές μεταβλητές, αλλά επειδή εδώ έχουμε πίνακα, δηλαδή διατεταγμένες μεταβλητές, θα εξηγώ σε ποια μεταβλητή από όλες αναφέρομαι, δίνοντας και τις συντεταγμένες της.

Στο μονοδιάστατο πίνακα, έστω θέλω να αναφερθώ στην πρώτη μεταβλητή και να της δώσω τιμή 10.
Η εντολή είναι:
Α(0) = 10

Αν θέλω να αναφερθώ στο τρίτο στοιχείο, και να του δώσω τιμή 364 τότε:
Α(2) = 364

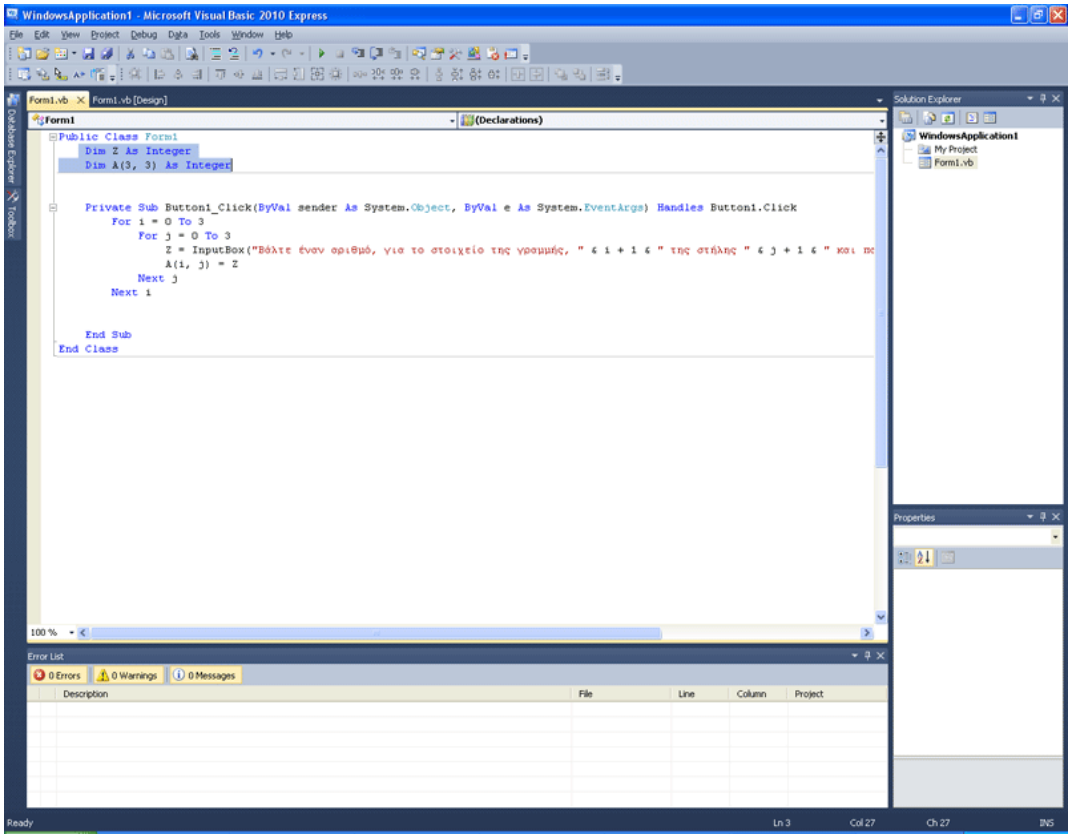
Το πρώτο στοιχείο, να το θυμόσαστε, έχει δείκτη μηδέν.
Οπότε το τρίτο έχει δείκτη 2.
Σε πίνακα δύο διαστάσεων, τύπου Integer,
αν θέλω να δώσω τιμή 5 στη μεταβλητή Α(3,7), τότε πολύ απλά δίνω την εντολή:
Α(3,7) = 5

Κατανοητό;
Έστω τώρα ότι έχω μια μεταβλητή Β τύπου Integer.
Θέλω να της δώσω την τιμή που έχει το στοιχειό Α(3,7).
Β = Α(3,7)
Πολύ απλά, πολύ εύκολα. Και τώρα έρχεται το ερώτημα. Πόσο πρακτικό είναι να χρησιμοποιώ πίνακες, πέραν της δομής δεδομένων και τα πλεονεκτήματά της.
Έστω πως θέλω να έχω ένα πίνακα 4 χ 4 τύπου Integer. Σε αυτόν θέλω να τοποθετήσω αριθμούς. Πώς το κάνω;



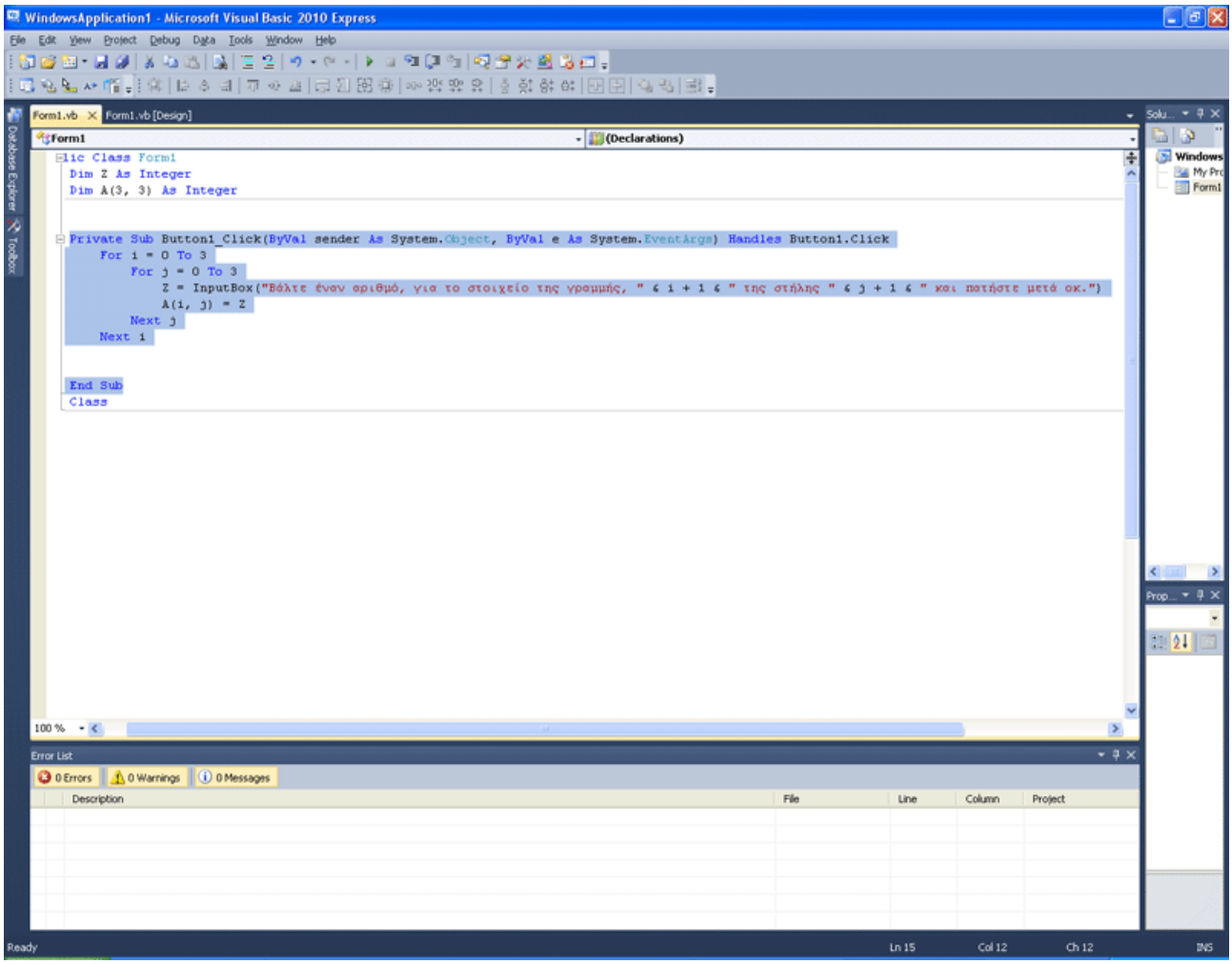
Θα φτιάξω μια φόρμα. Θα βάλω τώρα στη φόρμα ένα ένα κουμπί Button1.
Με το που φορτώνει η φόρμα, θα την κάνω να δημιουργεί τον πίνακά μου, με την εντολή:

Dim A(3,3) As Integer
Επίσης θα φτιάξω μια μεταβλητή τύπου Integer
Dim Z As Integer



Θα βάλω τώρα στη φόρμα ένα ένα κουμπί Button1.
Μέσα του θα γράψω τον κώδικα:

```
For i = 0 To 3
   For j = 0 To 3
   Z = InputBox("Βάλτε έναν αριθμό, για το στοιχείο της γραμμής, " & i + 1 & " της στήλης " & j + 1 & " και πατήστε μετά οκ.")
   A(i, j) = Z
   Next j
Next i
```



Τι κάνει αυτός ο κώδικας όμως; Κατ' αρχήν επισημαίνουμε πάλι ότι θέλουμε έναν 4 χ 4 πίνακα κι ότι η εντολή Dim A(3,3) As Integer, δε φτιάχνει έναν 3χ3 πίνακα, αλλά έναν 4χ4!
For i = 0 To 3
Για γραμμές από 0 ως 3 (0,1,2,3 οπότε έχω 4 γραμμές)
Βλέπετε; Ξεκινάμε από το δείκτη 0! Ωστόσο αυτό το προγραμματιστικό τερτίπι, πρέπει να το διορθώσουμε όταν θα εμφανίζεται στο χρήστη, διότι στα μαθηματικά δεν υπάρχει το στοιχείο α00 σε πίνακα, αλλά το α11. Θα δείτε πως θα γίνει.
For j = 0 To 3

Για στήλες από 0 ως 3 (0,1,2,3 οπότε έχω 4 στήλες)
Z = InputBox("Βάλτε έναν αριθμό, για το στοιχείο της γραμμής, " & i + 1 & " της στήλης " & j + 1 & " και πατήστε μετά οκ.")

Η Z παίρνει την τιμή που δίνει ο χρήστης στο Input Box, το οποίο του εμφανίζει ένα μήνυμα, με τα τρέχοντα i και j, προσαυξημένα κατά ένα. Γιατί αυτή η προσαύξηση; Γιατί στα μαθηματικά το πρώτο στοιχείο του πίνακα είναι το α11 ενώ προγραμματιστικά είναι το α00. Προσέξτε πώς ο τελεστής & συνενώνει τιμές μαθηματικών παραστάσεων με κείμενο. Προσέξτε και τα κενά που έχουμε βάλει σε αρχή και τέλος των κειμένων.

A(i, j) = Z
Το αντίστοιχο στοιχείο στον πίνακα, παίρνει την τιμή που του δόθηκε.

Next j
Επανέλαβε για όλα τα j

Next i
Επανέλαβε για όλα τα i

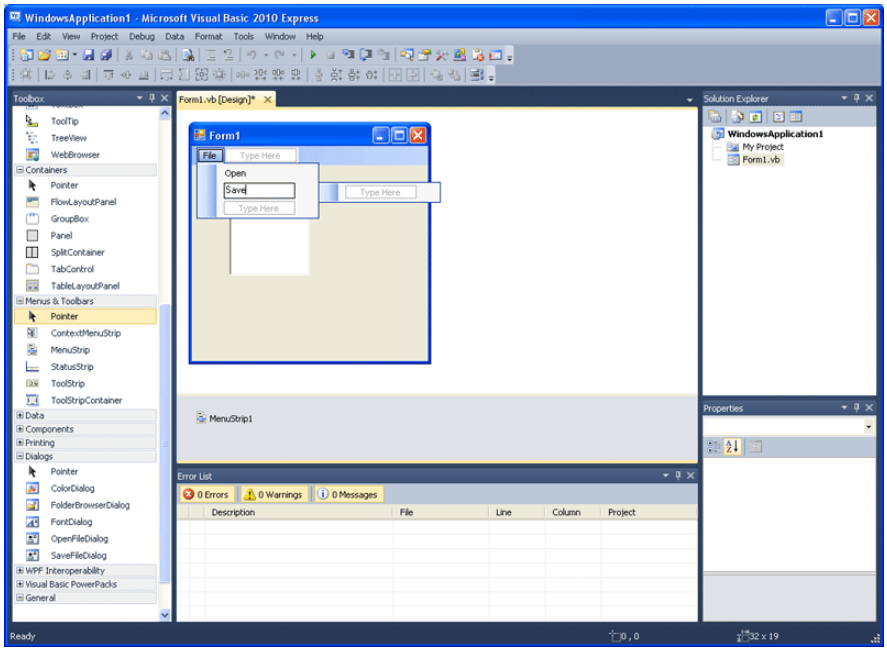
Τρέξτε το πρόγραμμα.
Βάλτε ακεραίους μόνο στο inputbox γιατί αλλιώς θα σας βγάλει σφάλμα. Δείτε το πως λειτουργεί. Είναι ίσως το καλύτερο παράδειγμα, για να δείτε τη δύναμη, δύο ενθυλακωμένων FOR.

Σαν άσκηση, βάλτε ακόμα ένα button, που θα έχει κώδικα με δύο For και θα εμφανίζει το περιεχόμενο των στοιχείων του πίνακα ένα προς ένα χρησιμοποιώντας Messagebox.

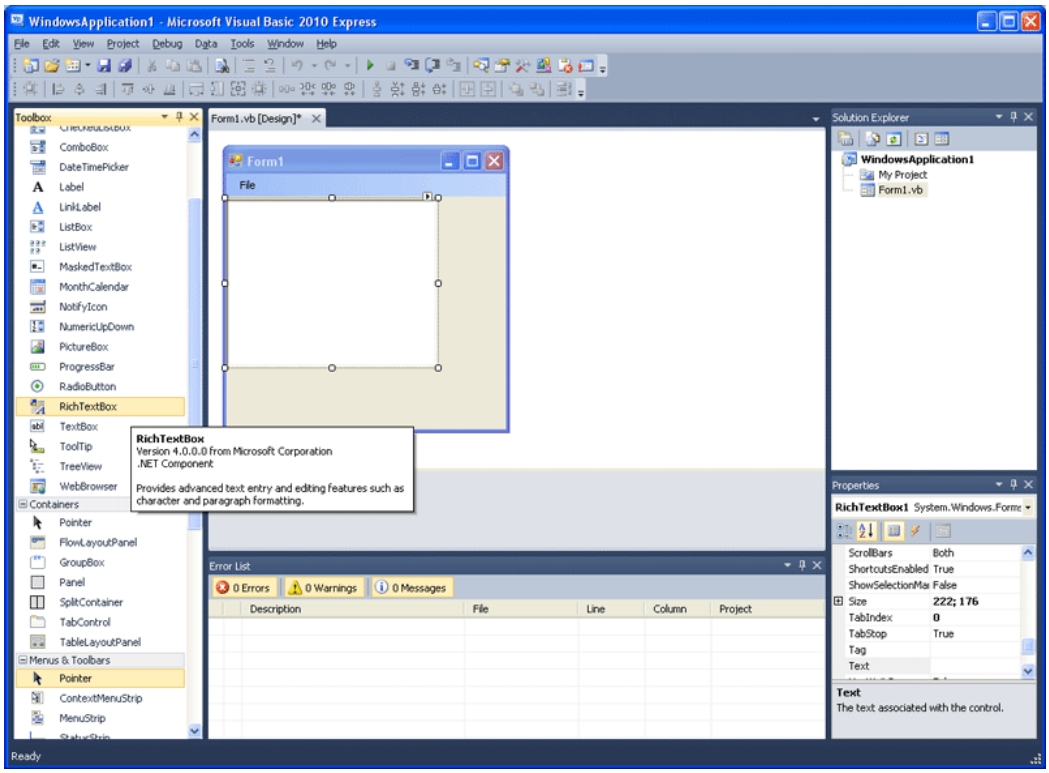
ΠΑΤΗΣΤΕ ΕΔΩ ΓΙΑ ΝΑ ΔΕΙΤΕ ΤΗ ΛΥΣΗ ΑΛΛΑ ΟΧΙ ΠΡΙΝ ΠΡΟΣΠΑΘΗΣΕΤΕ ΝΑ ΤΗ ΛΥΣΕΤΕ ΜΟΝΟΙ ΣΑΣ!

Και ήρθε ο καιρός να μιλήσουμε πλέον για αρχεία.
Μαθαίνοντας σε βάθος να τα χρησιμοποιείτε και να τα διαχειρίζεστε, μπορείτε να αποθηκεύετε πίνακες σε αυτά, έτσι ώστε να μη χάνονται τα δεδομένα σας. Σε ένα ηλεκτρονικό παιχνίδι, στον ηλεκτρονικό υπολογιστή, όταν διατηρούνται τα πρώτα δέκα καλύτερα score, τι νομίζετε πως συμβαίνει; Ενημερώνεται κι αποθηκεύεται ένας πίνακας 10x2 με πρώτη στήλη τα ονόματα και δεύτερη τα score. Αυτό απαιτεί αρκετή εξοικείωση με διάφορες μεθόδους. Ωστόσο εμείς θα περιοριστούμε στο να σας δείξουμε τη φιλοσοφία του όλου θέματος. Με τη σειρά σας εσείς έπειτα, θα μπορείτε να ανατρέξετε σε κάτι πιο εξειδικευμένο και να μπορέσετε να το κατανοήσετε.

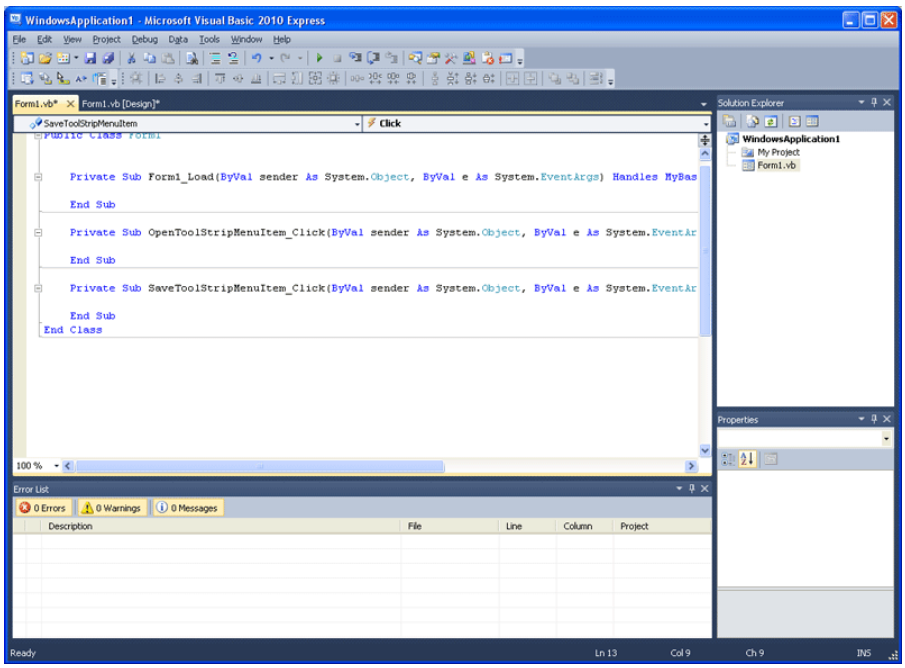
Φτιάχνουμε ένα μενού με τις επιλογές Open και Save



Μετά βάζουμε ένα Ritch TextBox. Το κάνουμε και Dock in parent container



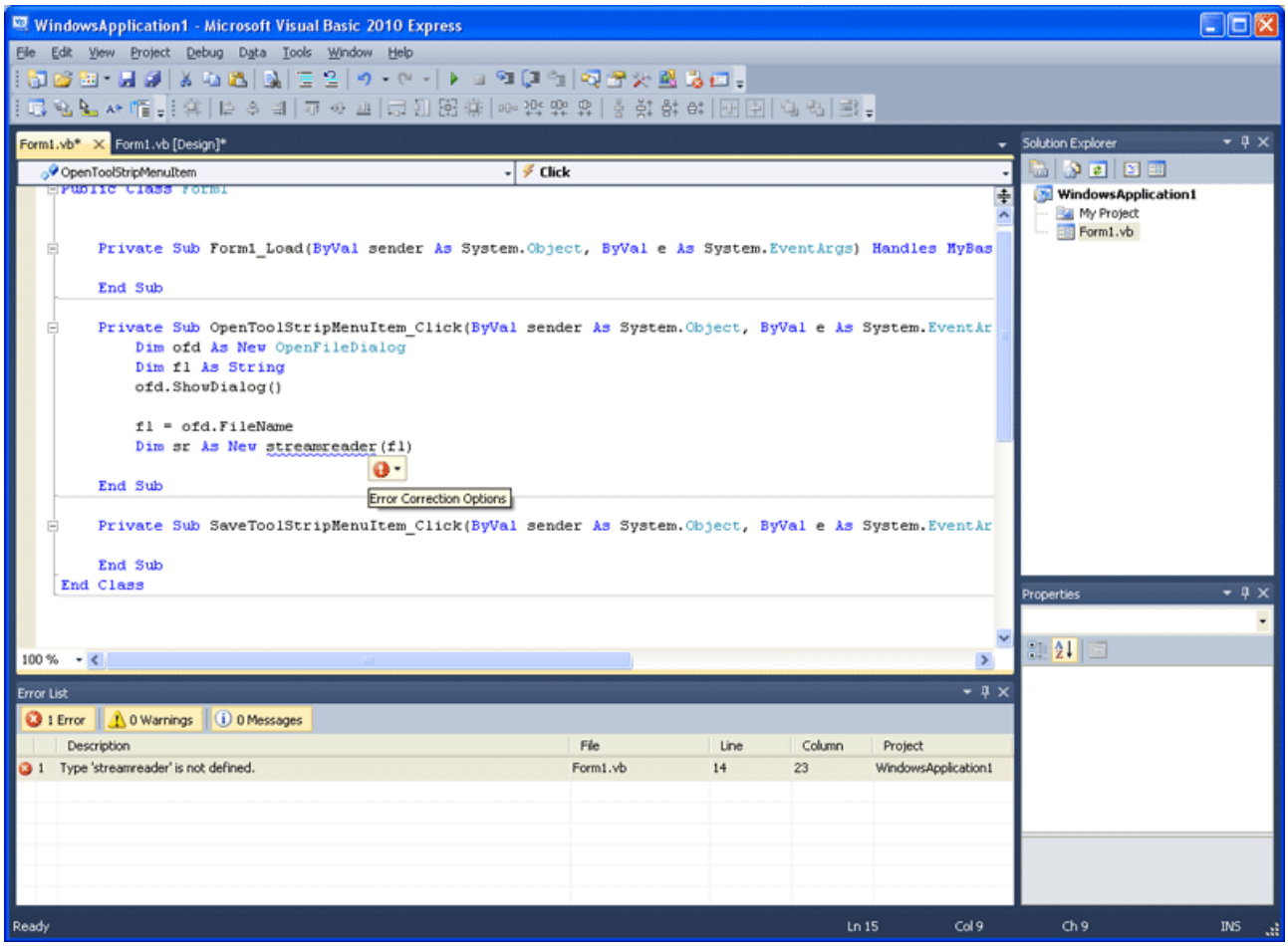
Κάνουμε διπλό κλικ πρώτα στην επιλογή Open του μενού που φτιάξαμε και μετά διπλό κλικ στη Save, για να δημιουργηθούν Subs κώδικα



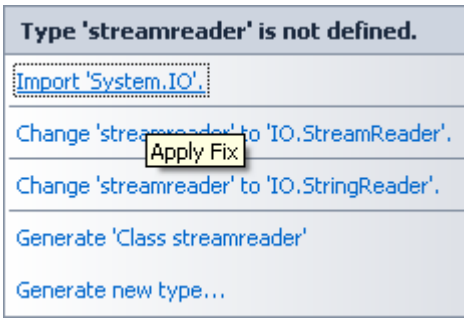
Αυτός είναι ο κώδικας για το OpenToolStripMenuItem_Click():

```
Dim ofd As New OpenFileDialog
Dim fl As String
ofd.ShowDialog()

fl = ofd.FileName
Dim sr As New StreamReader(fl)


RichTextBox1.Text = sr.ReadToEnd()
sr.Close()
```

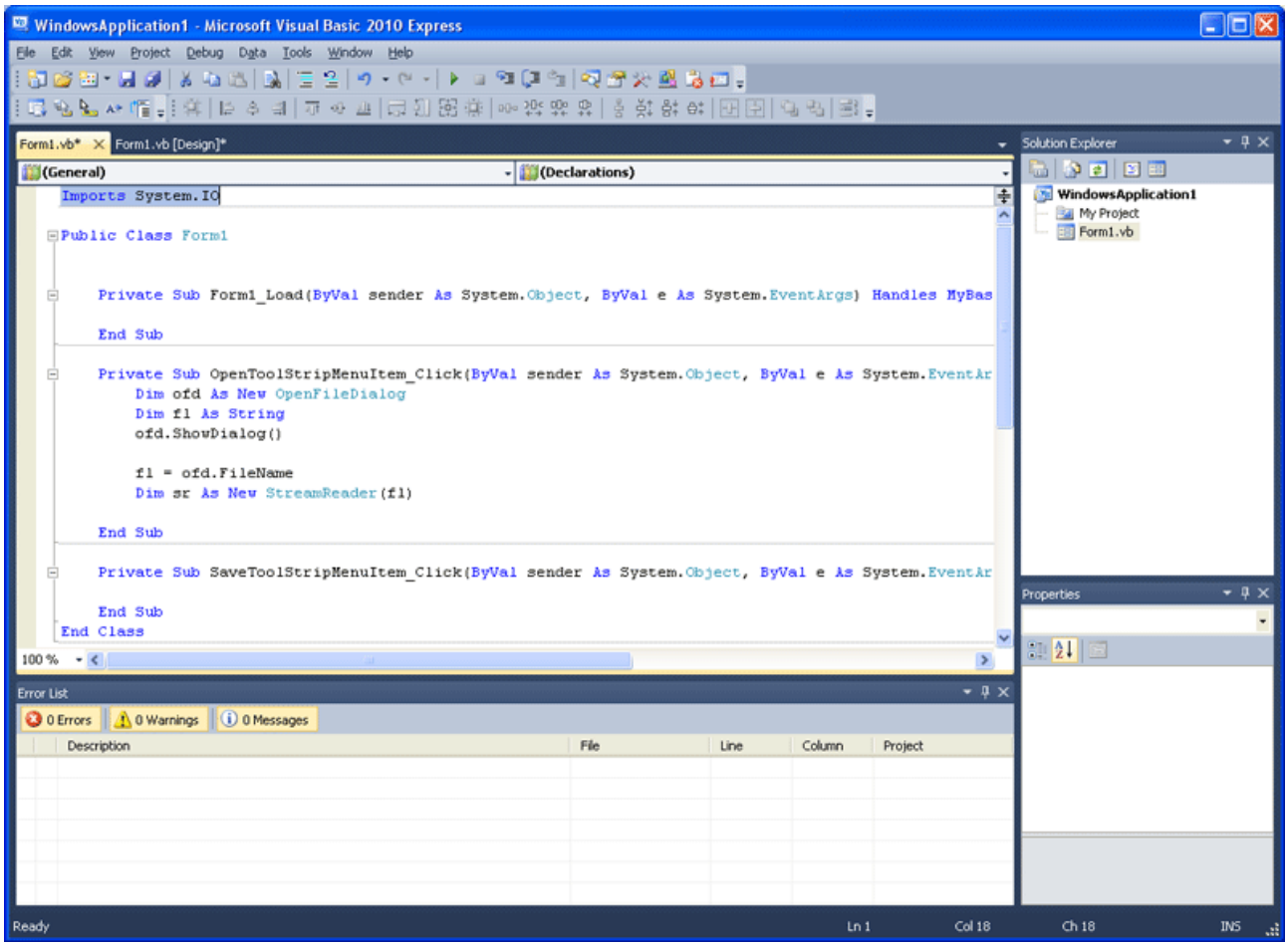
Αν το γράψετε θα δείτε, ότι η Visual Basic θα σας πει πως υπάρχει σφάλμα στο Dim sr As New StreamReader(fl). Το πρόβλημα έγκειται στο γεγονός, ότι πρέπει να γίνει included, η τάξη SystemIO στην οποία ανήκει το StreamReader(). Αλλιώς θα έπρεπε να το ορίζαμε μέσα στην εντολή. Επειδή όμως αυτό κάνει τις εντολές τεράστιες, είναι προτιμότερο στο πρόγραμμά μας, συμπεριληφθεί η class SystemIO. Φανταστείτε να είχαμε ένα menu με όλες τις εντολές διαχείρισης αρχείων. Εκεί θα ήταν απαραίτητο! Πάτε λοιπόν στο streamreader που έχει υπογραμμιστεί, και διαλέγετε από το εικονίδιο που εμφανίζεται κάτω δεξιά κάποια επιλογή.



Import 'System.IO' είναι η επιλογή μας!



Παρατηρήστε πως έξω ακόμα και από την class, στις γενικές δηλώσεις, πάνω πάνω, τοποθετείται ο κώδικας που έλειπε αυτόματα. Αυτό το εργαλείο επιδιόρθωσης τέτοιων σφαλμάτων, είναι σχετικά πρακτικό. Ωστόσο να θυμάστε πως εδώ είχαμε ένα βασικό δομικό πρόβλημα, που εύκολα εντοπίζεται. Σφάλματα λογικής είναι πολύ δύσκολο να εντοπιστούν. Γι' αυτό και προσχεδιάζουμε προγράμματα πριν γράψουμε κώδικα.



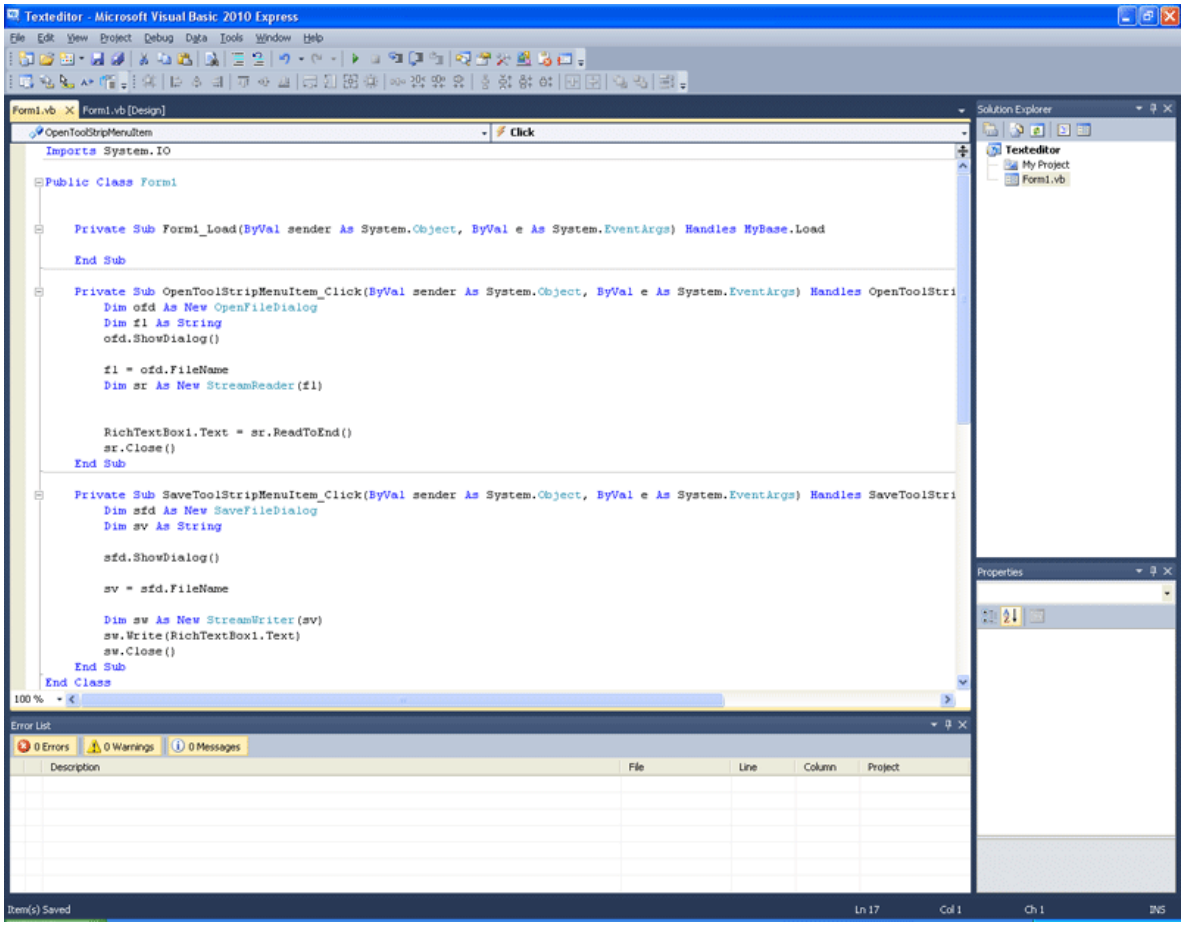
Ορίστε ο κώδικας λοιπόν και για τις δύο επιλογές menu:

Για το Open

```
Dim ofd As New OpenFileDialog
Dim fl As String
ofd.ShowDialog()

fl = ofd.FileName
Dim sr As New StreamReader(fl)


RichTextBox1.Text = sr.ReadToEnd()

sr.Close()
```

Για το Save

```
Dim sfd As New SaveFileDialog
 Dim sv As String
 sfd.ShowDialog()

 sv = sfd.FileName
Dim sw As New StreamWriter(sv)


sw.Write(RichTextBox1.Text)

sw.Close()
```



Dim ofd As New OpenFileDialog

Ορίζουμε μια μεταβλητή με όνομα ofd ως ένα OpenFileDialog. Πλέον όταν λέμε ofd εννοούμε ένα OpenFileDialog λοιπόν.

Dim fl As String
Ορίζουμε και μια μεταβλητή τύπου string με όνομα fl.

ofd.ShowDialog()
Εμφανίζουμε το Dialog στο οποίο ο χρήστης διαλέγει ένα text αρχείο.

fl = ofd.FileName
Η fl Παίρνει ως τιμή το όνομα και το path του αρχείου που διάλεξε ο χρήστης.

Dim sr As New StreamReader(fl)
Ορίζουμε μια μεταβλητή με όνομα sr ώς ένα StreamReader, που θα διαβάσει το αρχείο με όνομα το περιεχόμενο της fl.

RichTextBox1.Text = sr.ReadToEnd()

Στο RichTextBox1.Text προσθέτουμε μέχρι να τελειώσουν όλοι οι χαρακτήρες, ότι περιέχει ο sr ο οποίος περιέχει τι; Ότι περιέχει το αρχείο με όνομα και path, από την fl, η οποία πήρε τιμή από τι; Από την επιλογή αρχείου του χρήστη στο dialog.

sr.Close()
Τέλος κλείνουμε το stream Reader.

Στο Save ομοίως κάνουμε σχεδόν το ίδιο.
Dim sfd As New SaveFileDialog
Dim sv As String

sfd.ShowDialog()

sv = sfd.FileName

Dim sw As New StreamWriter(sv)
Εδώ είναι New StreamWriter κι όχι Reader. Απλά.

sw.Write(RichTextBox1.Text)
sw.Close()

Βασικό θέμα με το πρόγραμμα, είναι ότι ίσως ο χρήστης ανοίξει κάποιο αρχείο λάθος τύπου. Μπορούμε να τον βοηθήσουμε να μην κάνει τέτοιο σφάλμα, δίνοντας στο OpenFileDialog1 ένα φίλτρο. Η εντολή για αυτό είναι:

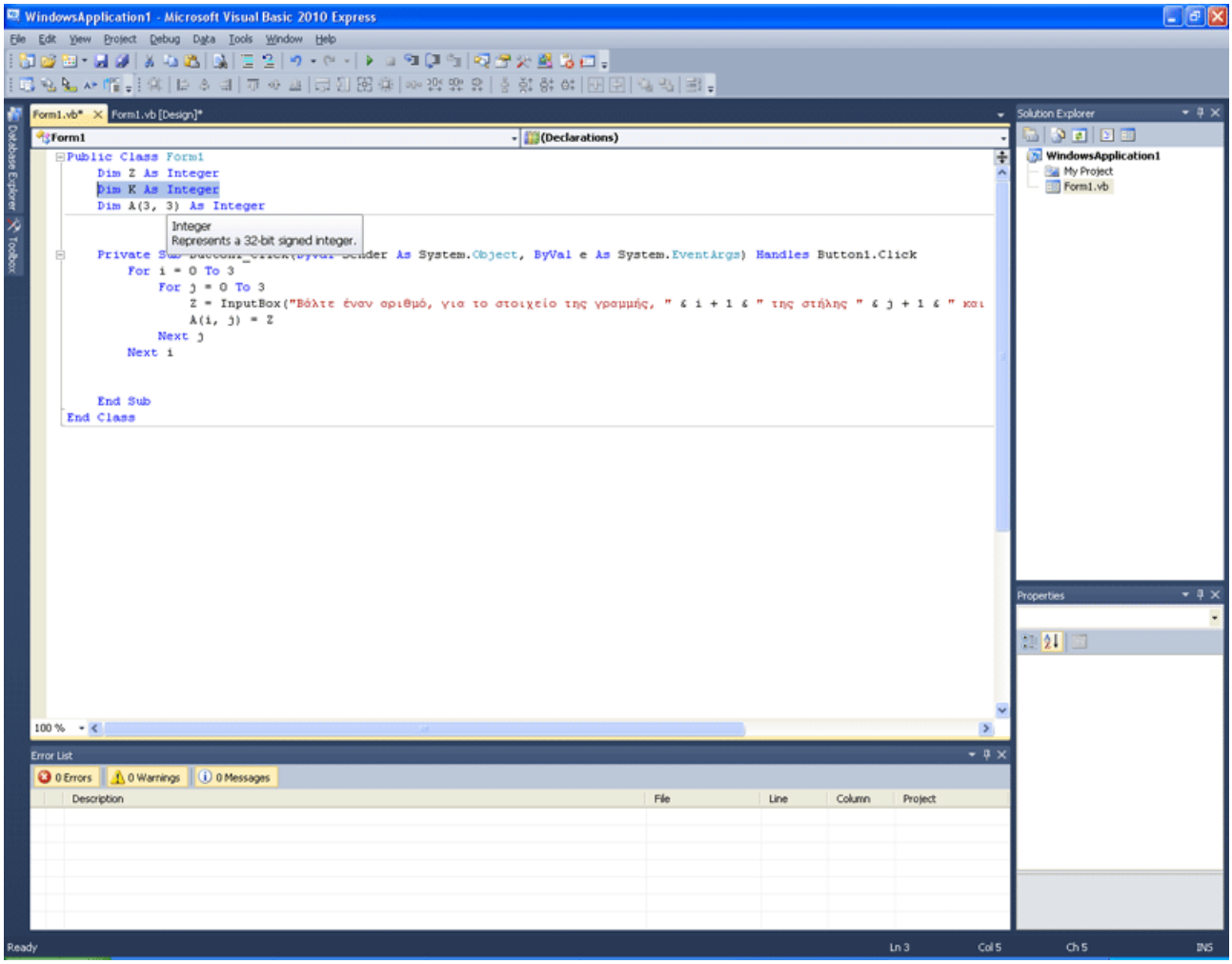
OpenFileDialog1.Filter = "Text Files|*.txt"

Και μπαίνει στο OpenToolStripMenuItem_Click() πριν εκτελεστεί οτιδήποτε άλλο.

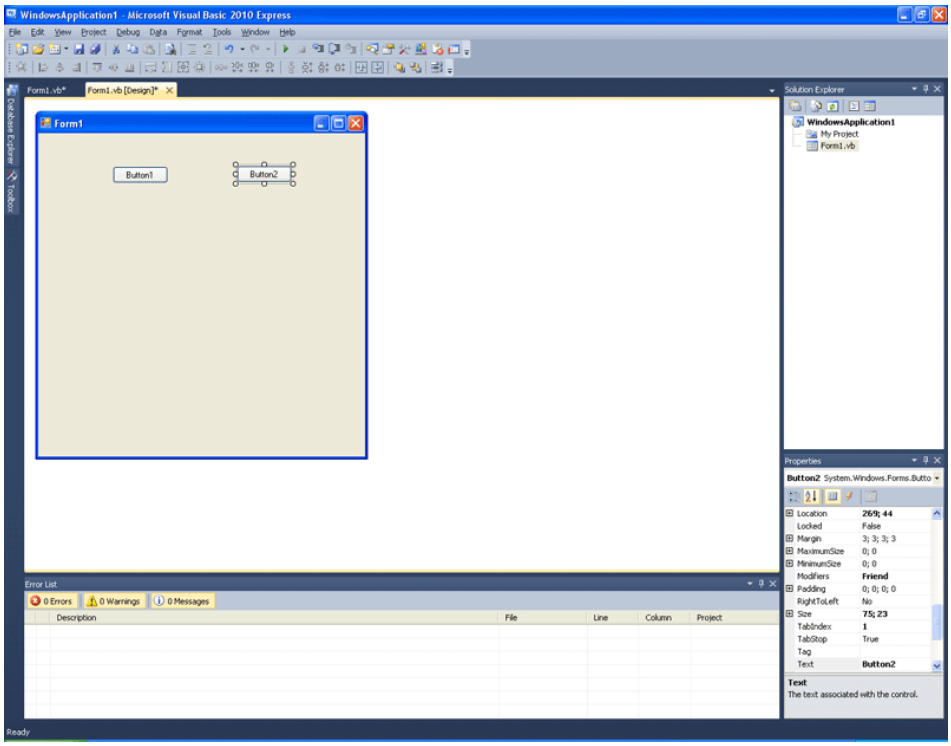
Επίσης εκτός της ReadToEnd, ο Stream Reader έχει και τη ReadLine. Η Readline χρησιμεύει πολύ σε αποθηκευμένους πίνακες κι όχι μόνο. Γενικώς μην αρκεστείτε μόνο στη γνώση που σας δίνουμε. Εμβαθύνετε, όπου επιθυμείτε, βάσει των αναγκών σας. Στο MSDN στο διαδίκτυο και στο Help της Visual Basic, υπάρχει τεράστιος όγκος πληροφοριών. Εμείς εδώ απλά σας μαθαίνουμε τη φιλοσοφία κάποιων θεμάτων, και τη λειτουργία. Για παράδειγμα, το RichTextbox δεν το έχουμε αναλύσει. Επειδή όμως γνωρίζετε το απλό TextBox, μπορείτε να το κατανοήσετε εύκολα. Αν δεν είχατε ήδη πειραματιστεί μαζί του, το κάνατε τώρα. Σας προτείνουμε, να πειραματιστείτε, με διάφορα controls και να αναζητήσετε πληροφορίες, τόσο, του τύπου "Τι είναι αυτό;" όσο και τύπου "Πώς κάνω αυτό", έτσι ώστε να καλύψετε, εξατομικευμένα πλέον, τις ανάγκες σας.

ΕΠΙΛΥΣΗ ΤΗΣ ΑΣΚΗΣΗΣ

Η άσκηση είναι πολύ απλή. Αρχικά θα χρησιμοποιήσουμε μια μεταβλητή τύπου ακεραίου με ονομασία Κ.
Dim K As Integer λοιπόν στη φόρμα μας!

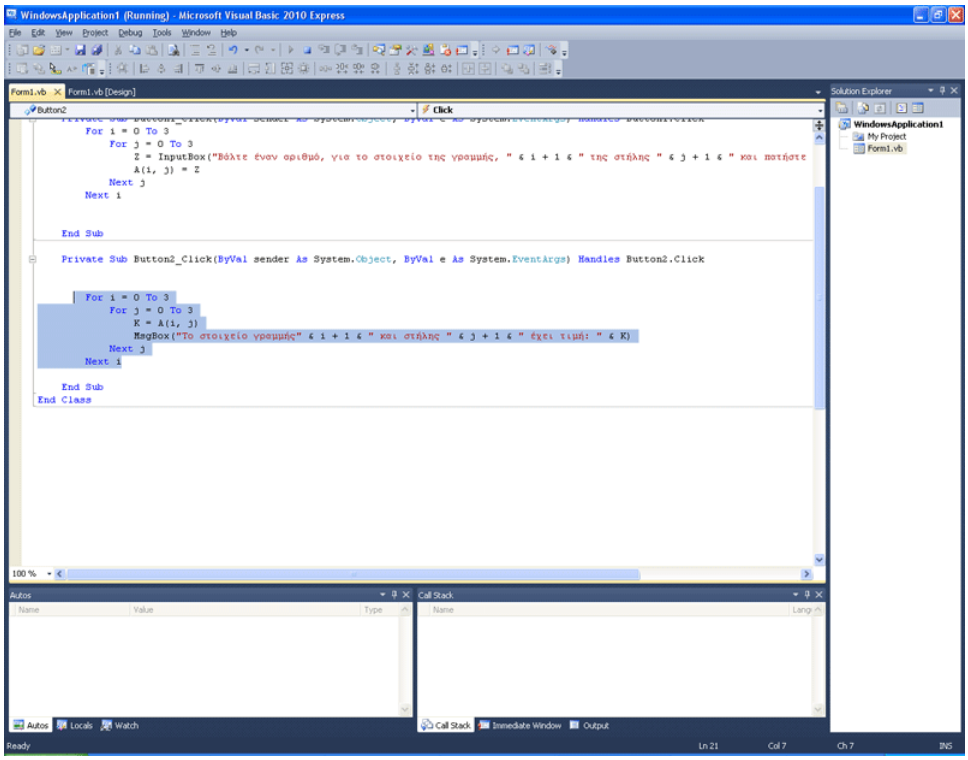


Δημιουργήστε ένα ακόμα button, όπως βλέπετε στην παρακάτω εικόνα



Στο button2 πληκτρολογούμε τον παρακάτω κώδικα:

```
For i = 0 To 3
    For j = 0 To 3
        K = A(i, j)
        MsgBox("Το στοιχείο γραμμής" & i + 1 & " και στήλης " & j + 1 & " έχει τιμή: " & K)
    Next j
Next i
```



Τρέξτε το πρόγραμμα.
Πατήστε το button1 και δώστε τιμές στις 16 μεταβλητές του πίνακα.
Πατήστε τώρα το button2! Θα δείτε ότι οι τιμές είναι όντως αποθηκευμένες στον πίνακα, μιας και τα messagebox, σας δείχνουν ένα ένα τα περιεχόμενά τους.

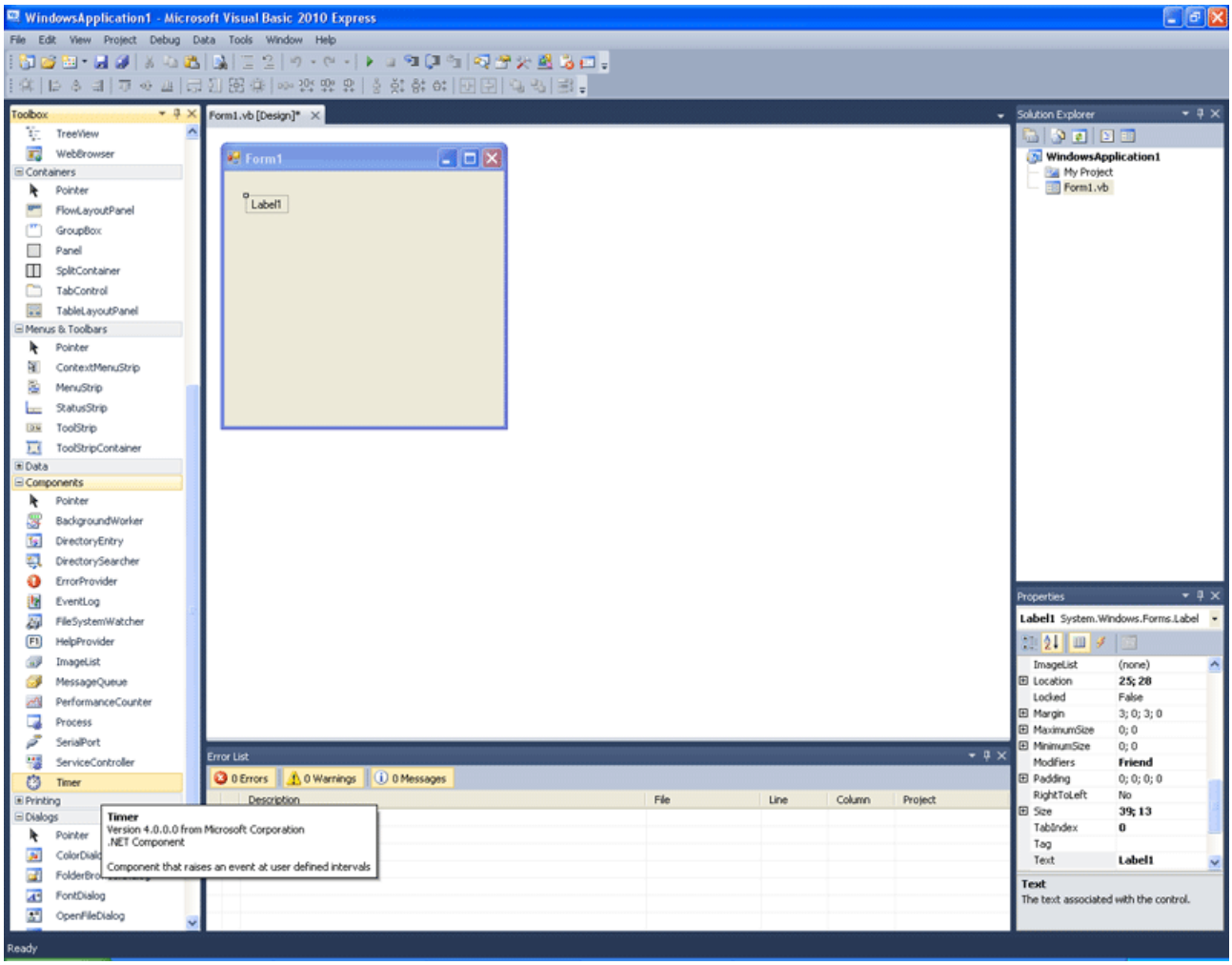
ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό δίνονται αρχικά, με απλό τρόπο, στοιχειώδεις γνώσεις μαθηματικών, όσον αφορά τους Πίνακες, αφότου επεξηγηθεί με καθημερινά παραδείγματα η έννοια του πίνακα. Κατόπιν δίνεται πρακτικό παράδειγμα καθώς και μία άσκηση με τη λύση της. Στη συνέχεια, γίνεται αναφορά στα αρχεία και τις λειτουργίες τους, ενώ δίνεται ως πρακτική άσκηση η δόμηση ενός τυπικού και στοιχειώδους επεξεργαστή κειμένου.

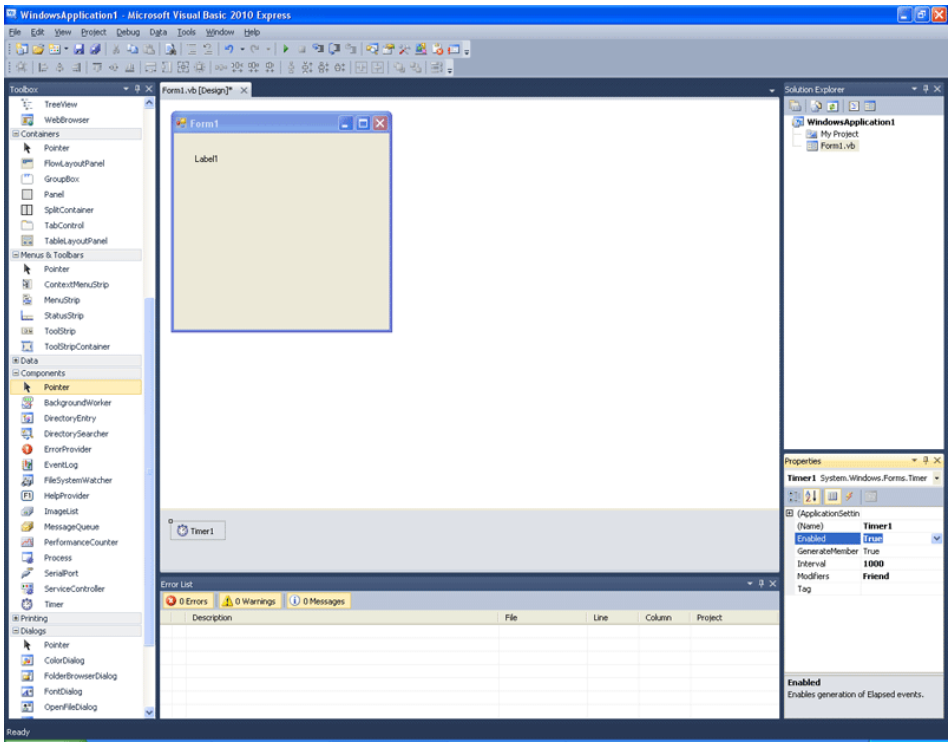
ΚΕΦΑΛΑΙΟ 17

Ο TIMER ΚΑΙ ΜΕΡΙΚΕΣ ΟΔΗΓΙΕΣ ΓΙΑ ΤΟ ΜΕΛΛΟΝ

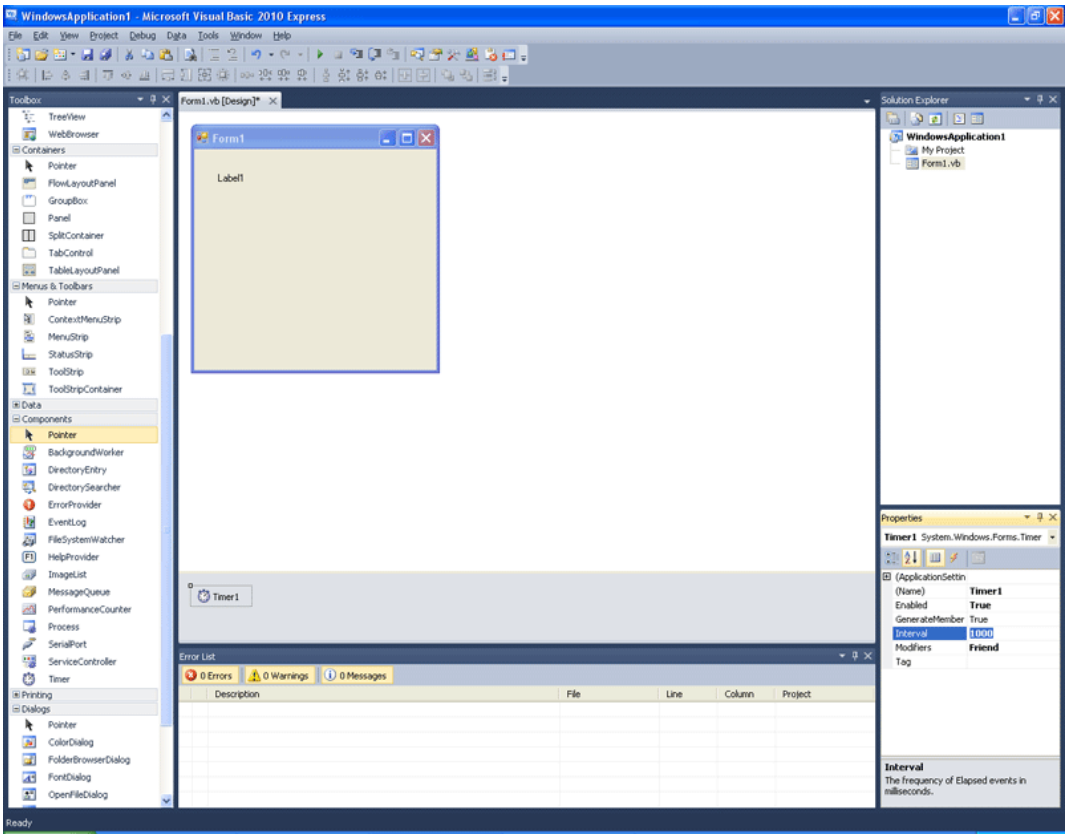
Φτάσαμε λοιπόν κοντά στο τέλος της σύντομης περιήγησης μας σε αυτή τη γλώσσα προγραμματισμού. Αφήσαμε λοιπόν για το τέλος κάτι καλό. Φτιάχνετε λοιπόν ένα νέο Project. Πάτε στο Toolbox, και βάζετε μια Label. Πάτε πάλι στο Toolbox, στα Components κι επιλέγετε Timer, όπως ακριβώς βλέπετε στην παρακάτω εικόνα.



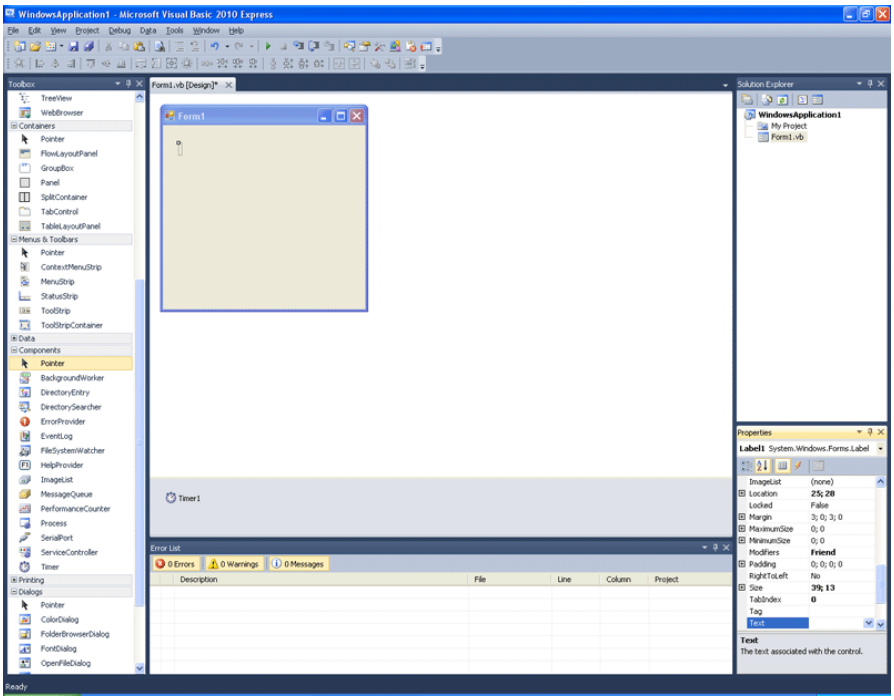
Με το που αρχίζει το πρόγραμμα, θέλουμε ο timer να ενεργοποιείται. Οπότε τον επιλέγουμε και στα Properties ορίζουμε Enabled = True...



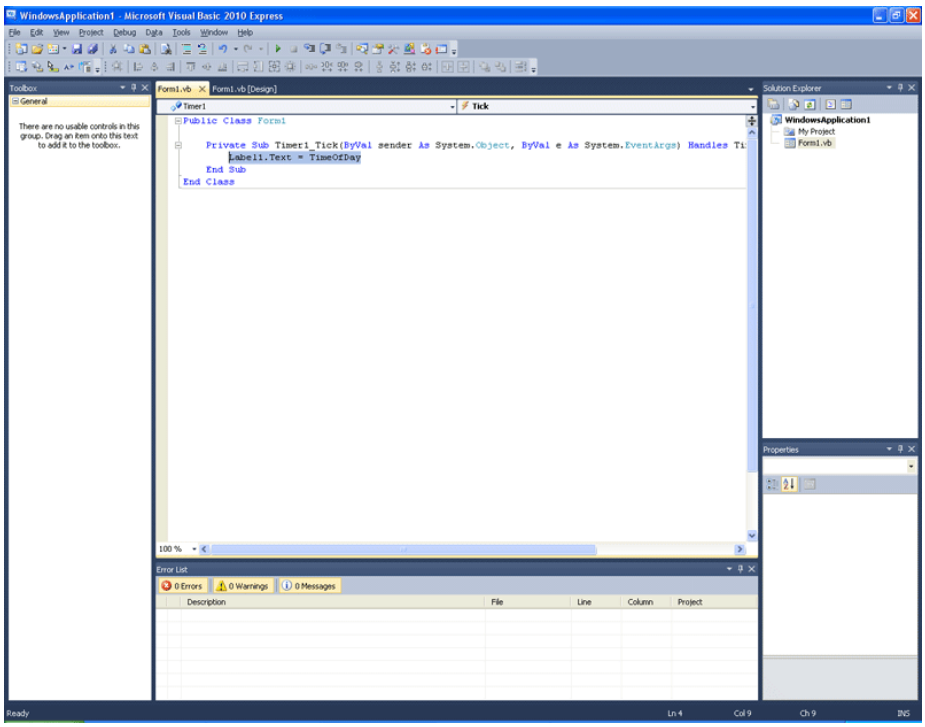
... και Interval = 1000



Σβήνουμε το περιεχόμενο του Label1.Text



Πατάμε διπλό κλικ στον timer για να μας εμφανίσει τον Sub κώδικα για το συμβάν Tick.
Προσθέτουμε την παρακάτω γραμμή κώδικα στο συμβάν Tick του Timer
Label1.Text = TimeOfDay



Τι κάναμε; Φτιάξαμε έναν Timer. Φροντίσαμε με το που ξεκινάει το πρόγραμμα, να ξεκινάει κι αυτός. Φανταστείτε ότι είναι κάτι σαν ωρολογιακός μηχανισμός, ο οποίος εκτελεί κώδικα, σε κάθε τικ. Το τικ μπορεί να γίνεται κάθε ένα δευτερόλεπτο, όπως εδώ.

Interval 1000 σημαίνει πως θα γίνεται κάθε 1000 milisecond.
Δηλαδή σε κάθε 1000/1000 = 1 δευτερόλεπτα.
Οπότε στο 1000 είναι κάθε δευτερόλεπτο.

Στο 500 είναι κάθε μισό δευτερόλεπτο.
500/1000 = 0.5 δευτερόλεπτα

Στο 10000 είναι κάθε δέκα δευτερόλεπτα.
10000/1000 = 10

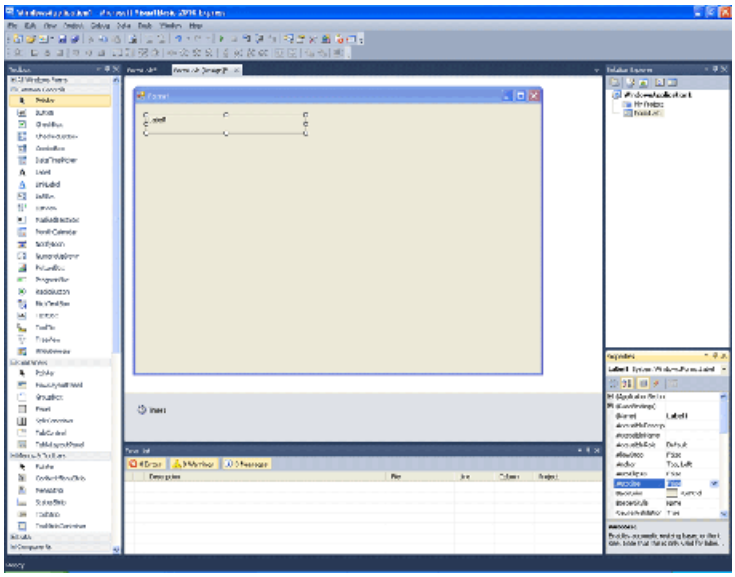
Μπορείτε λοιπόν να ορίσετε κάθε πότε θα εκτελείται ο κώδικας, με ακρίβεια milisecond.
Στο Τικ τι κώδικα εκτελεί;
Κάνε το Label1.Text να δείχνει την ώρα.

Κάθε πότε;
Κάθε δευτερόλεπτο.

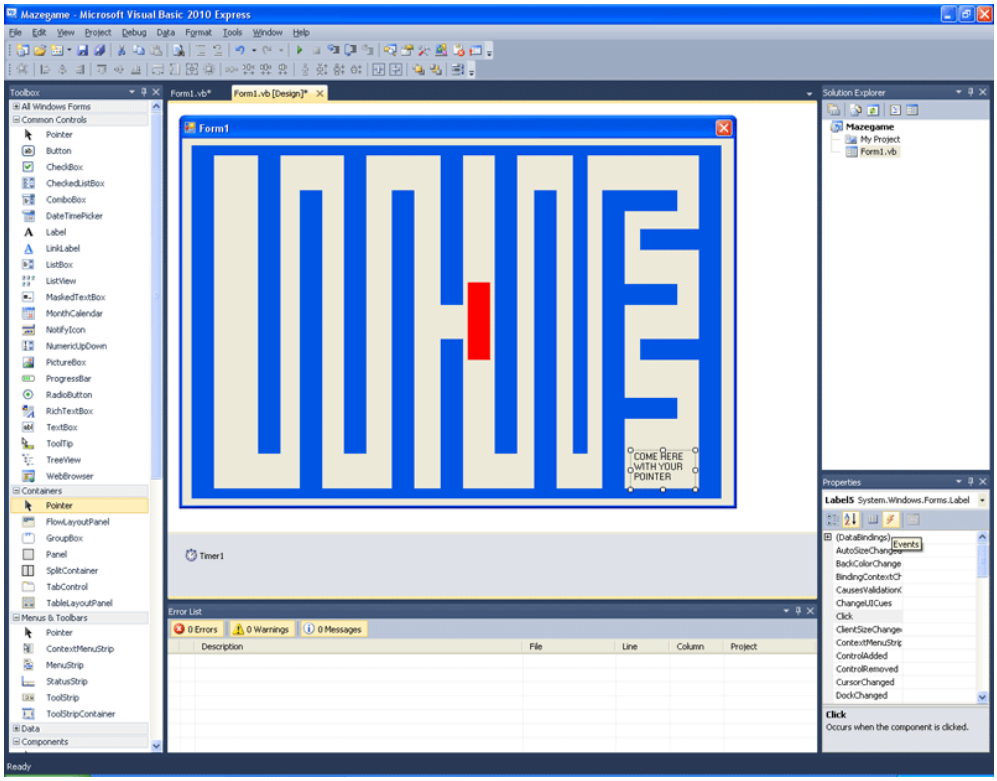
Ουσιαστικά δηλαδή φτιάξαμε ένα ρολόι!
Τρέξτε το πρόγραμμα!

Τι άλλο μπορείτε να κάνετε με τον Timer τώρα; Κάθε αντικείμενο έχει συντεταγμένες πάνω στη φόρμα. Αν κάθε κάμποσα milisecond αλλάζει θέση ένα αντικείμενο στην οθόνη... τότε μπορούμε να πούμε ότι δίνοντας κίνηση σε κάποια αντικείμενα, μπορούμε να φτιάξουμε ένα πολύ πιο άμεσο και διαδραστικό παιχνίδι! Βασισμένοι σε μια ιδέα του Mick Alberts και των διδακτικών videos του στο MSDN την πάμε ένα βήμα παρά πέρα, έτσι ώστε μαζί σας, να φτιάξουμε ένα διαδραστικό, παιχνίδι λαβύρινθο.

Νέο Project λοιπόν, φτιάχνουμε ένα timer, βάζουμε και μία label. Στη label την Auto Size Property την ορίζουμε FALSE. Τώρα μεγαλώστε τη label, όπως βλέπετε στην εικόνα.

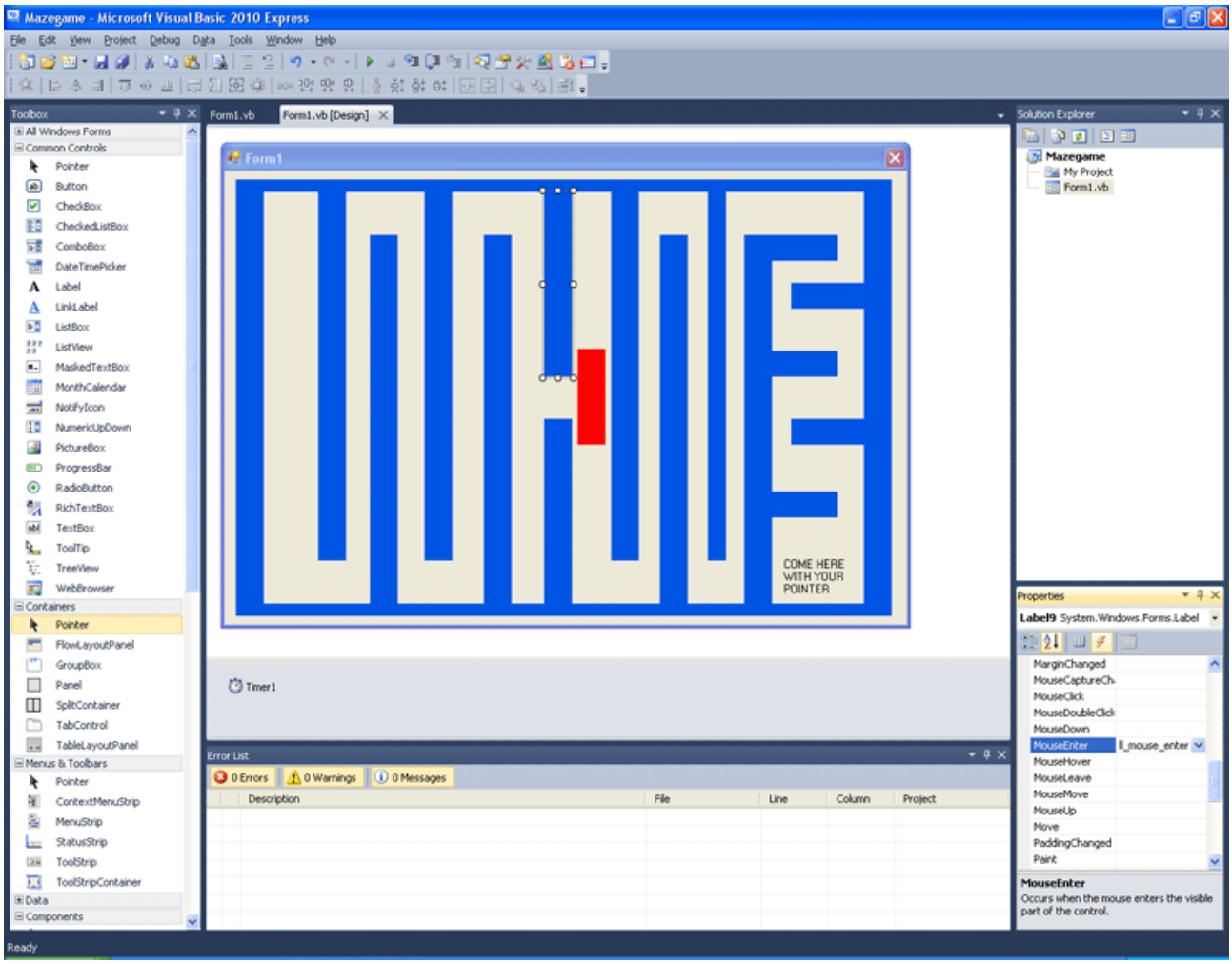


Με πολλές labels, των οποίων το Backcolor θα το κάνετε μπλε και το text θα το σβήσετε, φτιάχνετε κάτι σαν τη παρακάτω εικόνα. Μια καλή ιδέα, είναι την πρώτη label να την προετοιμάσετε. Δηλαδή να την κάνετε δίχως text και μπλε. Μετά με αντιγραφή - επικόλληση της, μπορείτε να φτιάξετε κι άλλες. Προσέξτε πως μία από αυτές, έχει κόκκινο χρώμα, ενώ η τελευταία, είναι άχρωμη, με text "COME HERE WITH YOUR POINTER". Επιλέξτε κάτω δεξιά στο Properties το εικονίδιο με τον κεραυνό. Μπορείτε να ξαναδείτε τα Properties βρίσκοντας το εικονίδιο δίπλα στον κεραυνό αυτό. Το εικονίδιο αυτό εμπεριέχει τα λεγόμενα events.



Επιλέξτε μία label. Επιλέγετε τα Events της. Βρείτε το event MouseEnter κι επιλέξτε το. Γράψτε στο δίπλα πλαίσιο wall_mouse_enter.

Κάντε διπλό κλικ πάνω στο MouseEnter και θα δημιουργηθεί κώδικας

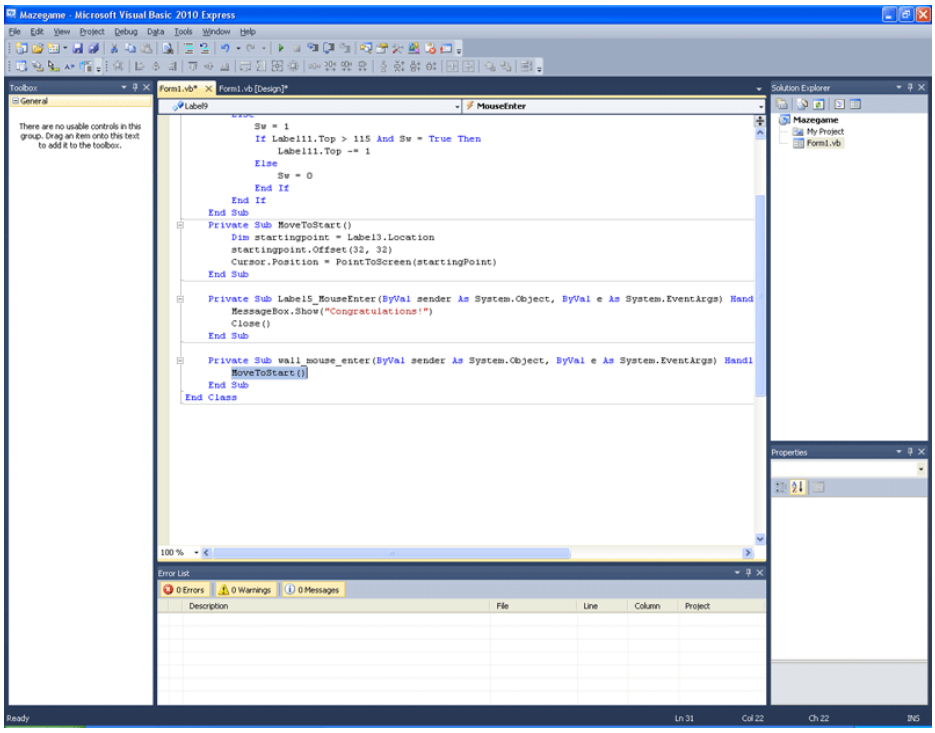


Γράφετε στον κώδικα...
MoveToStart()
..όπως θα δείτε στην παρακάτω εικόνα.

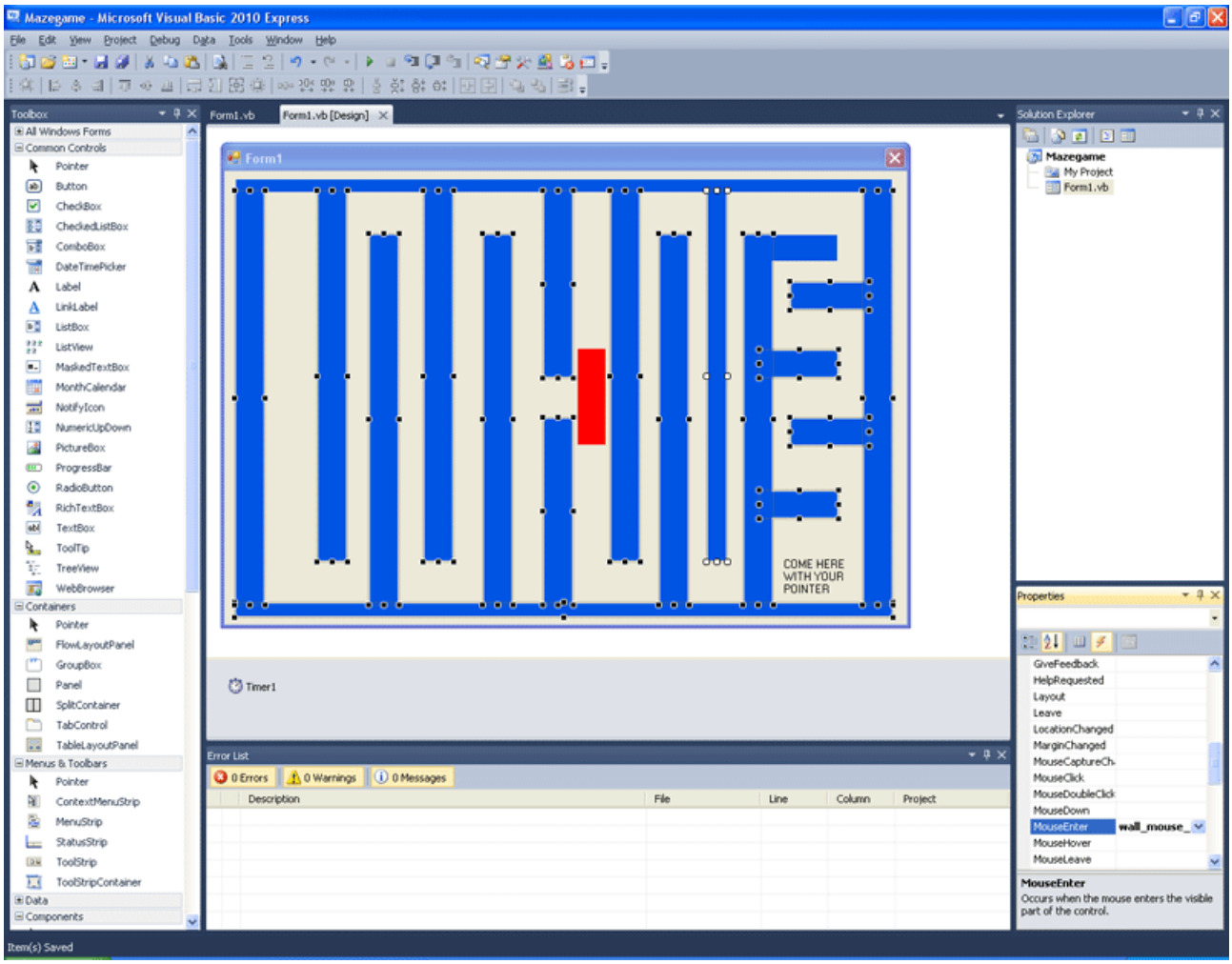
Η MoveToStart() είναι μέθοδος που δεν υπάρχει. Στην εικόνα έχουμε όλο τον κώδικα του project έτοιμο, οπότε από πάνω βλέπετε τον κώδικα της MoveToStart(). Μη τον γράψετε ακόμα όμως. Πρέπει να σας εξηγήσουμε εδώ τι συμβαίνει. Θέλουμε, όταν το βελάκι του ποντικιού δείξει το εσωτερικό ενός τοίχου, δηλαδή μπει στην επιφάνεια κάποιας μπλε label, να γυρνάει πίσω στην αρχική του θέση, πάνω αριστερά στη φόρμα. Αυτό δεν είναι έτοιμο σαν μέθοδος. Είναι ειδική μέθοδος, που καλύπτει πολύ συγκεκριμένη προγραμματιστική ανάγκη. Οπότε πρέπει να τη φτιάξουμε εμείς!

```
Private Sub MoveToStart()
        Dim startingpoint = Label3.Location
        startingpoint.Offset(32, 32)
        Cursor.Position = PointToScreen(startingPoint)
    End Sub
```

Εδώ να αναφέρουμε πως η Label3 είναι η πιο αριστερή Label στη φόρμα μας. Είναι κοινώς ο πιο αριστερός τείχος του λαβυρίνθου. Ορίζεται η θέση της στην ειδικού τύπου μεταβλητή startingpoint κι έπεται η εξής εντολή:
startingpoint.Offset(32, 32) αυξάνει τους αριθμούς αυτούς κατά 32.
Cursor.Position = PointToScreen(startingPoint)
Το βελάκι (κέρσορας) θα τοποθετηθεί λοιπόν 32 pixel δεξιότερα και κάτω από τη πιο γωνιακή Label. Αν η πιο αριστερή Label - τοίχος σας είναι άλλη αντί της Label3, για παράδειγμα Label12, κανένα πρόβλημα, βάζετε Label12 αντί Label3 στον κώδικα κι όλα καλά. Η Label5 στο παράδειγμά μας είναι η Label η οποία είναι η άχρωμη, με text "COME HERE WITH YOUR POINTER". Σε αυτή ο κώδικας είναι απλός.
Είναι δύο μόνο γραμμές.
MessageBox.Show("Congratulations!")
Close()
Εμφανίζει συγχαρητήριο μήνυμα, και μετά κλείνει η φόρμα.
Επειδή είναι η μόνη φόρμα του προγράμματος, κλείνει και το πρόγραμμα ουσιαστικά.



Πατώντας ctrl κάντε κλικ σε όλες τις Labels που έχουν χρώμα.



Πηγαίνετε στα Events MouseEnter κι επιέξτε στο combo box δίπλα του, τη wall_mouse_enter Τώρα αυτό το event κάνει handle όλες τις Labels αυτές. Δείτε τι γίνεται στον κώδικά του αυτόματα.

```
Private Sub wall_mouse_enter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.MouseEnter,
 Label4.MouseEnter, Label3.MouseEnter, Label21.MouseEnter, Label20.MouseEnter,
Label19.MouseEnter, Label18.MouseEnter,
 Label17.MouseEnter, Label16.MouseEnter, Label15.MouseEnter, Label14.MouseEnter,
Label13.MouseEnter, Label1.MouseEnter,
 Label9.MouseEnter, Label8.MouseEnter, Label7.MouseEnter, Label6.MouseEnter,
Label2.MouseEnter, Label12.MouseEnter,
 Label10.MouseEnter, Label11.MouseEnter
            MoveToStart()
  End Sub
```

Τεράστια δουλειά να ορίσετε με το χέρι τι κάνει handle. Καλύτερα όπως το κάνατε, δε νομίζετε; Η Sub wall_mouse_enter λοιπόν, αφορά όλες τις labels αυτές και εφαρμόζεται σε όλες.

Πάνω από όλες τις Sub, αλλά κάτω από τη Public Class Form1
γράψτε τη παρακάτω μέθοδο:

```
Public Sub New()
        InitializeComponent()
        MoveToStart()
    End Sub
```

Αυτή η μέθοδος στέλνει τον κέρσορα στην αρχική θέση που θέλουμε, με τη μέθοδο MoveToStart() όταν ξεκινάει το πρόγραμμα.

Κάντε κλικ στον Timer.
Στα Properties ορίζουμε

Enabled σε True
Interval ίσο με 10

Ξεκινάει δηλαδή με την έναρξη του προγράμματος και κάθε 10 χιλιοστά του δευτερολέπτου, επαναλαμβάνει τον κώδικά του.

Διπλό κλικ στον timer τώρα!
Κάτω από το Public Class Form1ορίζουμε μια μεταβλητή Dim Sw As Boolean = False

Και τώρα ο κώδικας του timer.

```
 Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        If Label11.Top < 259 And Sw = False Then
            Label11.Top += 1
        Else
            Sw = 1
            If Label11.Top > 115 And Sw = True Then
                Label11.Top -= 1
            Else
                Sw = 0
            End If
        End If
    End Sub
```

Θα παρατηρήσατε ότι μία label είναι κόκκινη. Στο παράδειγμά μας είναι η Label11
Στο δικό σας βρείτε ποια είναι κι απλά προσαρμόστε αναλόγως τον κώδικα.
Είναι μια ετικέτα που λειτουργεί σαν τοίχος, με τη μόνη διαφορά, ότι θα την κάνουμε να κινείται, χρησιμοποιώντας τον timer. Πως το κάνουμε. Πήραμε τη Label πάνω στη φόρμα, και την μετακινήσαμε εκεί που θέλαμε να πάει, με το χέρι. Κρατήσαμε τις συντεταγμένες, γράφοντάς τες σε ένα χαρτί. Ουσιαστικά Label.Top 259 ήταν το σημείο που θέλουμε να φτάσει. Με το φτάσει εκεί, θέλουμε να κινηθεί αντίθετα ως το Label.Top 115. Αυτούς τους αριθμούς τους βρήκαμε τοποθετώντας την κόκκινη Label με το ποντίκι μας, εκεί που θέλουμε να πηγαίνει οριακά και βλέποντας και τις δύο φορές στο Properties τις συντεταγμένες της.

Σε εσάς, σαφώς είναι πολύ πιθανόν να είναι διαφορετικοί.
Θυμόσαστε τη μεταβλητή Sw που ορίσαμε ως Boolean (παίρνει τιμές 0 και 1);
Εδώ θα τη χρειαστούμε, ως σημαία αλλαγής κίνησης.

If Label11.Top < 259 And Sw = False Then
Αν δεν έχεις φτάσει μέχρι το 259 ΚΑΙ η Sw είναι False (0 δηλαδή), τότε

Label11.Top += 1
προσθέτεις ένα pixel στην απόσταση της Label από την κορυφή της φόρμας (κίνηση προς τα κάτω),

Else
ειδάλλως

Sw = 1
Πρώτα ορίζεις την Sw True (1)
κι έπειτα ελέγχεις:

If Label11.Top > 115 And Sw = True Then
Αν απέχει περισσότερο του 115 ΚΑΙ η Sw έιναι True Τότε

Label11.Top -= 1
αφαιρείς ένα pixel στην απόσταση της Label από την κορυφή της φόρμας (κίνηση προς τα πάνω),

Else
ειδάλλως

Κάνε την Sw = 0
End If
Τέλος Αν
End If
Τέλος Αν

Τι κάναμε δηλαδή;
Με απλά λόγια, Αν η σημαία Sw είναι πεσμένη και είμαστε σε θέση, μικρότερη του 259 τότε πλησιάζουμε το 259
Αν όμως φτάσουμε στο 259 η σημαία ανεβαίνει. Αν δεν έχουμε επιστρέψει στο 115 και η σημαία είναι όρθια επιστρέφουμε πίσω προς το 115.
Αλλιώς
(αν δηλαδή επιστέψαμε τελικά με όρθια τη σημαία) Κατεβάζουμε τη σημαία και συνεχίζουμε όπως πριν, χωρίς η λούπα να σταματάει.

Χρησιμοποιήσαμε λοιπόν έναν timer για να θέσουμε σε κίνηση ένα αντικείμενο (Label).
Ωστόσο στο Interval βάλαμε 10 miliseconds. Αν βάλουμε 1 δεν υπάρχει μεγάλη διαφορά στην ταχύτητα της κίνησησς του αντικειμένου. Είναι δυνατόν να δημιουργήσουμε γρηγορότερη κίνηση; Ναι! Αυξάνοντας την αλλαγή στην απόσταση κάθε 10 miliseconds
Αυτό γίνεται αν γράψουμε Label11.Top += 2 αντί για 1!

Συνολικά ο κώδικας του παιχνιδιού:

```
Public Class Form1
   Dim Sw As Boolean = False
   Public Sub New()
      InitializeComponent()
      MoveToStart()
   End Sub
   Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
      If Label11.Top < 259 And Sw = False Then
         Label11.Top += 1
      Else
         Sw = 1
         If Label11.Top > 115 And Sw = True Then
            Label11.Top -= 1
         Else
            Sw = 0
         End If
      End If
   End Sub
   Private Sub MoveToStart()
      Dim startingpoint = Label3.Location
      startingpoint.Offset(32, 32)
      Cursor.Position = PointToScreen(startingPoint)
   End Sub

   Private Sub Label5_MouseEnter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label5.MouseEnter
      MessageBox.Show("Congratulations!")
      Close()
   End Sub

   Private Sub wall_mouse_enter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.MouseEnter, Label4.MouseEnter, Label3.MouseEnter,
Label21.MouseEnter, Label20.MouseEnter, Label19.MouseEnter, Label18.MouseEnter,
Label17.MouseEnter, Label16.MouseEnter, Label15.MouseEnter, Label14.MouseEnter,
Label13.MouseEnter, Label1.MouseEnter, Label9.MouseEnter, Label8.MouseEnter,
Label7.MouseEnter, Label6.MouseEnter, Label2.MouseEnter, Label12.MouseEnter,
Label10.MouseEnter, Label11.MouseEnter
      MoveToStart()
   End Sub
End Class
```

Φτάσαμε λοιπόν στο τέλος αυτού του βοηθήματος. Το βοήθημα όμως δε θα ήταν πλήρες, αν δε σας έδινε οδηγίες για το μέλλον, για την περαιτέρω διεύρυνση των γνώσεών σας.

Προτείνουμε αρχικά να εξερευνήσετε το Help της Visual Basic .NET 2010 Express, καθώς και τον ιστότοπο MSDN της Microsoft. Εκεί θα βρείτε αρκετές πληροφορίες στα Αγγλικά. Καλή ιδέα είναι να διαβάσετε κάποιο αναλυτικό βιβλίο για τη Visual Basic. Υπάρχουν πολλά στο εμπόριο, μεταφρασμένα στα Ελληνικά ή και γραμμένα από Έλληνες. Είναι καλό να εστιάσετε πρώτα στη χρήση των υπόλοιπων controls, έπειτα να ασχοληθείτε με διάφορες ενδιαφέρουσες έτοιμες συναρτήσεις της Visual Basic και τέλος να αναζητήσετε τρόπους να υλοποιήσετε προγράμματα, που να φέρνουν εις πέρας εξατομικευμένα δικές σας ανάγκες.

Πρώτο πράγμα που πρέπει να έχετε κατά νου όμως, είναι η ποιότητα αυτού που θέλετε να φτιάξετε. Να θυμάστε πάντοτε να αφήνετε σχόλια στον κώδικα χρησιμοποιώντας το ειδικό σύμβολο [‘]. Επίσης να αναλύετε το πρόβλημα, να το προσχεδιάζετε, να γράφετε τον κώδικα πρώτα σε ψευδοκώδικα και μετά να επιχειρείτε την υλοποίηση.
Έχετε κατά νου να γράφετε τον κώδικα, έτσι ώστε να είναι εύκολα επεκτάσιμος κι αναβαθμίσιμος. Το ότιι εργάζεστε με αντιμενοστραφή γλώσσα βοηθάει πολύ, αλλά θα πρέπει πάντοτε να εργάζεστε σκεπτόμενοι, ότι κάποιος τρίτος αν ανοίξει να δει τον κώδικα, θα πρέπει να μπορεί εύκολα να καταλάβει τι του γίνεται.

Όσον αφορά τη διεύρυνση των γνώσεών σας, έχετε πολλές επιλογές, ανάλογα με τις ανάγκες που απαιτούνται να καλυφθούν:
Μελετήστε δομές δεδομένων και θεωρία βάσεων δεδομένων, αν θέλετε να φτιάξετε προγράμματα που να διαχειρίζονται πληροφορίες μέσω βάσεων δεδομένων. Να έχετε κατά νου ότι η Microsoft έχει εναν SQL Server αλλά και την Access, όπου και τα δύο πακέτα λογισμικού, συνεργάζονται με τη Visual Basic περίφημα. Σε κάτι τέτοιο, αλλά και γενικότερα στον προγραμματισμό, θα σας βοηθήσουν γνώσεις, γλώσσας μοντελοποίησης UML. Πρόκειται για ένα πρότυπο, κατά το οποίο σχεδιάζονται λεπτομερώς πολύπλοκα συστήματα. Γνωρίζοντας τα εργαλεία του (διαγράμματα) θα μπορέσετε να εργαστείτε πιο εύκολα και πιο επαγγελματικά.

Είναι καλή ιδέα, τώρα που εξοικειωθήκατε με το Visual περιβάλλον, να μάθετε πιο "δημοφιλείς" στην αγορά, γλώσσες προγραμματισμού. Το Visual Studio περιέχει αρκετές τέτοιες. Υπάρχουν βέβαια και αντικειμενοστραφείς γλώσσες ανοιχτού κώδικα. Δώστε προσοχή κι εκεί. Επίσης αν είσαστε φοιτητής καλό είναι να εμημερωθείτε για τον ιστότοπο Dreamspark της Microsoft.

Ψάξτε γενικά στο internet για ιστοτόπους προγραμματισμού, αλλά και forums. Πριν χρησιμοποιήσετε όμως κάπου κώδικα, προσέξτε μήπως υπόκειται σε δικαιώματα πνευματικής ιδιοκτησίας. Στα forums υπάρχουν πολλοί πρόθυμοι χρήστες να μοιραστούν τη χαρά της γνώσης μαζί σας. Σας ευχόμαστε καλή συνέχεια στο υπέροχο ταξίδι γνώσεων που ξεκινήσατε, τιμώντας μας, από εδώ.

ΤΕΚΜΗΡΙΩΣΗ ΚΕΦΑΛΑΙΟΥ

Το τελευταίο κεφάλαιο, επεξηγεί τη χρησιμότητα του timer με δύο παραδείγματα, ενώ δίνονται και μερικές ακόμα προγραμματιστικές υποδείξεις. Εν κατακλείδι δίνονται οδηγίες για τη μελλοντική αναζήτηση επιπρόσθετων πληροφοριών πάνω στη Visual Basic και όχι μόνο, ανάλογα με το που θέλει να προσανατολιστεί προγραμματιστικά ο μαθητευόμενος.

ΑΝΤΙ ΕΠΙΛΟΓΟΥ

Η εργασία αυτή κάλυψε ένα ευρύ φάσμα θεμάτων, δίνοντας πολλά βασικά εφόδια στο μαθητευόμενο. Ξεκινώντας από βασική ορολογία και τις γενικές γνώσεις για τη γλώσσα, προχώρησε στον τρόπο εγκατάστασης, επεξήγησε το περιβάλλον της και διάφορα έτοιμα εργαλεία της, έδωσε θεμελιώδεις γνώσεις προγραμματισμού, έτσι ώστε ο μαθητευόμενος να καταλαβαίνει στη συνέχεια τι κάνει (δυαδικό σύστημα και μεταβλητές), προχώρησε στις συναρτήσεις, στις δομές αποφάσεων και τους τελεστές, τις δομές επανάληψης, επεξήγησε βασικές έννοιες αντικειμενοστραφών γλωσσών, όπως αντικείμενο και κλάση, παρουσίασε το module, ανέλυσε εννοιολογικά βασικές γνώσεις για τους πίνακες τόσο μαθηματικά, όσο και προγραμματιστικά, ενώ παράλληλα, επεξήγησε τη σημαντικότητα θεμάτων όπως αρχικοποίηση, ανάλυσης και σχεδιάσης του προγράμματος πριν την υλοποίηση, κι όλα αυτά με όσο το δυνατόν απλούστερο τρόπο, παραδείγματα, και σε μερικά κεφάλαια ασκήσεις. Επείτηδες ωστόσο παραλήφθηκε η εντολή GOTO κι είναι η πρώτη φορά που την αναφέρουμε, εδώ, στον επίλογο, η οποία μπορεί μεν κάποτε να βοηθούσε αρχάριους να φτιάχνουν προγράμματα εύκολα, όμως κατέληξε για τους αρχάριους αυτούς να γίνει τροχοπέδη στην ανάπτυξη σκέψης δομημένου προγραμματισμού.

ΕΥΧΑΡΙΣΤΙΕΣ

Τον φοιτητή του τμήματος Πληροφορικής του Πανεπιστημίου Πειραία, Κοτσώνη Αλέξανδρο, για την πολύτιμη γνώμη του και το beta testing του.
Τον πρωτοετή φοιτητή του τμήματος Πληροφορικής του Πανεπιστημίου Πειραία, Λεωνίδα Μαρκοδημητράκη, για την πολύτιμη γνώμη του και το beta testing του.
Το Διευθυντή και την Υποδιευθύντρια του 1ου ΕΠΑΛ Πειραιά, για τη συνεργασία τους.

Τους Καθηγητές:
Βιδάλη Ιωάννη (ΠΕ 19)
Νικολουδάκη Παρασκευή (ΠΕ 19)
Καρελιά Βασιλική (ΠΕ 19)
Αραπάκη Στυλιανή (ΠΕ 19)

για την πολύτιμη γνωμάτευσή τους, πάνω στο βοήθημα, καθώς και για την ανατροφοδότιση που μας παρείχαν, από την εμπειρία χρήσης του βοηθήματος αυτού.

Ευχαριστούμε επίσης τους μαθητές του ΒΠ του 1ου ΕΠΑΛ Πειραιά για την προφορική τους ανατροφοδότιση που μας παρείχαν, από την εμπειρία χρήσης του βοηθήματος αυτού.

Ευχαριστούμε όλους τους καθηγητές του Τ.Ε.Ι. μας, για τις γνώσεις που μας προσέφεραν.

Την εταιρία Microsoft, που προσφέρει η γλώσσα Visual Basic .NET 2010 Express δωρεάν για διδακτικούς σκοπούς.

Και δε μπορούμε να παραλείψουμε να ευχαριστήσουμε εσάς, για την επιλογή σας να μας εμπιστευτείτε στα πρώτα σας βήματα, στο να μάθετε τη γλώσσα Visual Basic .NET 2010 Express.

ΒΙΒΛΙΟΓΡΑΦΙΑ ΚΑΙ ΠΗΓΕΣ

ΒΙΒΛΙΑ:

ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ
ΑΘ. Γ. ΤΣΟΥΡΟΠΛΗΣ
ΚΩΝ. Σ. ΚΛΗΜΟΠΟΥΛΟΣ
5Η ΕΚΔΟΣΗ
ΕΚΔΟΣΕΙΣ ΝΕΩΝ ΤΕΧΝΟΛΟΓΙΩΝ
ISBN 960-8105-84-6

Μάθετε τη Visual Basic.NET
Foxall, James
Εκδόσεις Γκιούρδας Μ.
ISBN 978-960-512-321-5

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΜΕ ΤΗ VISUAL BASIC 6
ΑΓΓΕΛΗ ΧΡΥΣΑΝΘΗ
Εκδόσεις ΣΥΓΧΡΟΝΗ ΕΚΔΟΤΙΚΗ
ISBN 960-8165-05-9

VB.NET LANGUAGE IN A NUTSHELL A Desktop Quick Reference
Steven Roman
Petrusha Ron
Paul Lomax
ISBN: 978-0-596-00092-9
Εκδόσεις: O'Reilly Media

ΔΙΚΤΥΑ ΚΑΙ ΔΙΑΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ και εφαρμογές τους στο Internet
DOUGLAS E. COMER
CD- ROM από τον RALPH DROMS
Τέταρτη Αμερικάνικη έκδοση
Εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ
ISBN 978-960-461-040-2

Σχεδιασμός πετυχημένων ιστοσελίδων
Σπύρος Δ. Αρσένης
ISBN 978-960-461-041-9
Εκδόσεις: ΚΛΕΙΔΑΡΙΘΜΟΣ

Αντικειμενοστρεφής Ανάπτυξη Λογισμικού με τη UML
Βασίλης Γερογιάννης
Γιώργος Κακαρόντζας
Αχιλλέας Καμέας
Γιάννης Σταμέλος
Πάνος Φιτσιλής
ISBN 960-209-913-5
Εκδόσεις: ΚΛΕΙΔΑΡΙΘΜΟΣ

ΓΕΝΙΚΑ ΜΑΘΗΜΑΤΙΚΑ Ι
ΧΑΡΑΛΑΜΠΟΣ Γ. ΖΑΓΟΥΡΑΣ
ΔΗΜΗΤΡΙΟΣ Ν. ΓΕΩΡΓΙΟΥ
ISBN 960-406-468-1
ΕΚΔΟΣΕΙΣ: ΕΛΛΗΝΙΚΑ ΓΡΑΜΜΑΤΑ

Μαθηματικά 4ης Δέσμης γραμμική άλγεβρα - πιθανότητες
(σύμφωνα με το αναλυτικό πρόγραμμα 1993-1994)
Μιχάλη Ανιτσάκη
ISBN 960-01-0472-7
ΕΚΔΟΣΕΙΣ: GUTENBERG

ΟΡΘΟΛΟΓΙΚΗ ΛΗΨΗ ΑΠΟΦΑΣΕΩΝ ΜΙΑ ΟΛΟΚΛΗΡΩΜΕΝΗ ΠΡΟΣΕΓΓΙΣΗ
Andrew Lang Golub
Επιστημονική επιμέλεια Μητρόπουλος Ιωάννης
ISBN 960-87710-0-5
Εκδόσεις: Γκότσης

ΔΙΟΙΚΗΤΙΚΗ ΕΠΙΣΤΗΜΗ ΛΗΨΗ ΕΠΙΧΕΙΡΗΣΙΑΚΩΝ ΑΠΟΦΑΣΕΩΝ ΣΤΗΝ ΚΟΙΝΩΝΙΑ ΤΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
Γρηγόρης Πραστάκος
ISBN 960-351-501-9
ΕΚΔΟΣΕΙΣ: ΑΘ. ΣΤΑΜΟΥΛΗ

ΔΙΟΙΚΗΣΗ – ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΟΥ
Αντώνης Δημητριάδης
ISBN 960-8105-71-4
ΕΚΔΟΣΕΙΣ ΝΕΩΝ ΤΕΧΝΟΛΟΓΙΩΝ

ΜΑΡΚΕΤΙΝΓΚ ΜΑΝΑΤΖΜΕΝΤ Η ΕΛΛΗΝΙΚΗ ΠΡΟΣΕΓΓΙΣΗ
Γιάννης Πετρώφ
Κώστας Τζωρτζάκης
Αλεξία Τζωρτζάκη
ISBN 960-85749-5-1
ΕΚΔΟΣΕΙΣ: Rosili

Εισαγωγή στις Πανεπιστημιακές Σπουδές
Δ.Γ. Τσαούσης
ISBN 960-01-0648 – 7
ΕΚΔΟΣΕΙΣ: GUTENBERG

ΑΛΛΕΣ ΠΗΓΕΣ:

Σημειώσεις:

Α.Τ.Ε.Ι. ΠΑΤΡΑΣ
ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ
ΤΜΗΜΑ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΥ ΣΧΕΔΙΑΣΜΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΟΡΓΑΝΩΣΗ ΑΡΧΕΙΩΝ
Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ PASCAL
Δρ. ΑΡΗΣ Ν. ΜΠΑΚΑΛΗΣ
ΠΑΤΡΑ 2009

Περιοδικά:

Total Hacker

Ιστοσελίδες στο Internet που συμβουλευτήκαμε, πήραμε εικόνες ή χρησιμοποιήσαμε προγράμματα:

http://msdn.microsoft.com/en-us/vbasic/default
http://blogs.msdn.com/b/kathleen/archive/2010/10/27/getting-started-tutorials.aspx
https://www.dreamspark.com/
http://www.youtube.com/watch?v=JJJtsnmkzXY
http://el.wikipedia.org
http://en.wikipedia.org
http://www.thefreedictionary.com
http://www.homeandlearn.co.uk/net/vbNet.html
http://www.functionx.com/
http://users.teicrete.gr/taxd/02/notes/arithmetics/00.arithmetic%20systems.htm
http://download.oracle.com/javase/tutorial/java/concepts/index.html
http://www.theasciicode.com
http://notepad-plus-plus.org/
http://www.gimp.org/windows/
http://www.adobe.com/
http://el.openoffice.org/
http://www.openoffice.org/
http://www.schach-lernen.de/seiten/schachbrett-bezeichnung-der-felder.html
http://www.dropbox.com/