ΕΚΔΟΣΕΙΣ ΚΛΕΙΔΑΡΙΘΜΟΣ

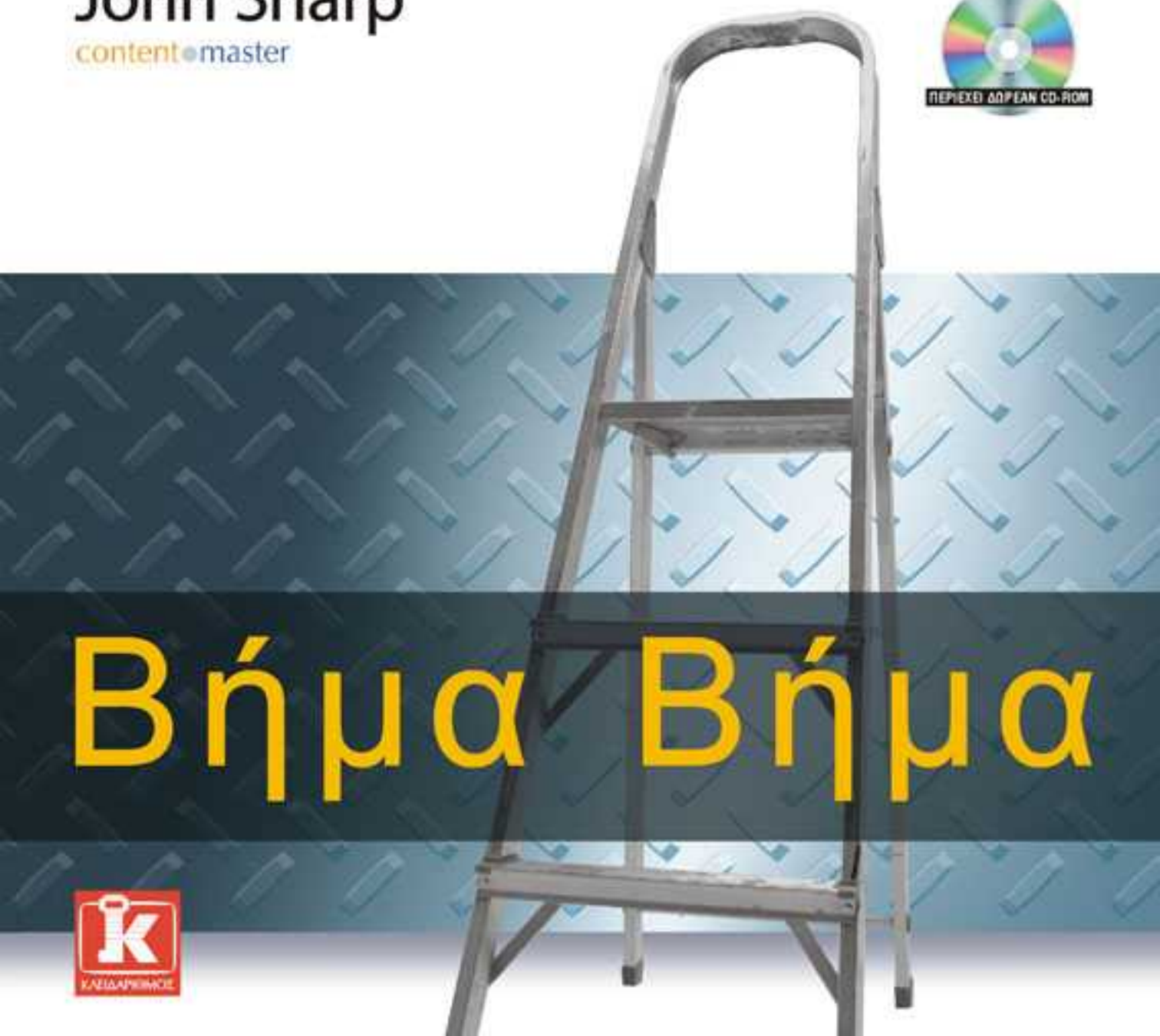
Ένα βιβλίο από τη Microsoft

Microsoft

Visual C# 2008

John Sharp
content master

ΠΕΡΙΕΧΕΙ ΔΩΡΕΑΝ CD-ROM



Βήμα Βήμα

ΚΛΕΙΔΑΡΙΘΜΟΣ

Περιεχόμενα

Ευχαριστίες **19**

Εισαγωγή **21**

Μέρος I Εισαγωγή στη Microsoft Visual C# και το Microsoft Visual Studio 2008

1 Καλωσορίσατε στη C# **35**

Ξεκίνημα στον προγραμματισμό στο περιβάλλον του Visual Studio 2008 35
Γραφή του πρώτου σας προγράμματος 40
Χρήση χώρων ονομάτων 46
Δημιουργία μιας εφαρμογής γραφικών 49

2 Χρήση μεταβλητών, τελεστών, και παραστάσεων **61**

Οι εντολές 61
Χρήση αναγνωριστικών 62
Αναγνώριση λέξεων-κλειδιών 62
Χρήση μεταβλητών 63
Ονομασία μεταβλητών 64
Δήλωση μεταβλητών 64
Πρωτογενείς τύποι δεδομένων 65
Εμφάνιση τιμών πρωτογενών τύπων δεδομένων 66
Χρήση αριθμητικών τελεστών 70
Τελεστές και τύποι δεδομένων 71
Εξέταση των αριθμητικών τελεστών 72
Έλεγχος της προτεραιότητας 75
Χρήση προσεταιριστικότητας για τον υπολογισμό παραστάσεων 76
Η προσεταιριστικότητα και ο τελεστής ανάθεσης τιμής 77
Βηματική αύξηση και μείωση μεταβλητών 77
Προθεματική και επιθεματική μορφή 78
Δήλωση τοπικών μεταβλητών υπονοούμενου τύπου 79
Γρήγορη αναφορά Κεφαλαίου 2 80

3 Γραφή μεθόδων και εφαρμογή εμβέλειας..... 81

Δήλωση μεθόδων 81
Καθορισμός της σύνταξης δήλωσης μιας μεθόδου 82
Γραφή εντολών *return* 83
Κλήση μεθόδων..... 85
Καθορισμός της σύνταξης κλήσης μιας μεθόδου 85
Εφαρμογή εμβέλειας..... 88
Ορισμός τοπικής εμβέλειας 88
Ορισμός εμβέλειας κλάσης..... 88
Υπερφόρτωση μεθόδων 89
Γραφή μεθόδων 90
Γρήγορη αναφορά Κεφαλαίου 3..... 98

4 Χρήση εντολών λήψης αποφάσεων 99

Δήλωση λογικών μεταβλητών..... 99
Χρήση λογικών τελεστών 100
Τελεστές ισότητας και συγκριτικοί τελεστές 100
Λογικοί τελεστές συνθήκης.....101
Σύνοψη προτεραιότητας και προσεταιριστικότητας τελεστών 102
Χρήση εντολών *if* για τη λήψη αποφάσεων.....103
Η σύνταξη της εντολής *if* 103
Χρήση μπλοκ για ομαδοποίηση εντολών 105
Διαδοχικές εντολές *if*.....105
Χρήση εντολών *switch*.....110
Η σύνταξη της εντολής *switch*.....111
Οι κανόνες της εντολής *switch*.....112
Γρήγορη αναφορά Κεφαλαίου 4.....116

5 Χρήση σύνθετων εντολών ανάθεσης τιμής και επανάληψης..... 117

Χρήση σύνθετων τελεστών ανάθεσης τιμής.....117
Γραφή εντολών *while* 119
Γραφή εντολών *for*.....123
Η εμβέλεια μιας εντολής *for*.....124
Γραφή εντολών *do* 125
Γρήγορη αναφορά Κεφαλαίου 5.....134

6 Διαχείριση σφαλμάτων και εξαιρέσεων..... 135

Αντιμετώπιση των σφαλμάτων 135
Δοκιμή κώδικα και σύλληψη εξαιρέσεων.....136
Χειρισμός μιας εξαίρεσης.....137
Ανεπίλυτες εξαιρέσεις.....137

Χρήση πολλών χειριστών *catch*....138
Σύλληψη πολλών εξαιρέσεων....138
Χρήση αριθμητικής ακεραίων *checked* και *unchecked*....143
Γραφή εντολών *checked*....144
Γραφή παραστάσεων *checked*....145
Μεταβίβαση εξαιρέσεων....146
Χρήση ενός μπλοκ *finally*....150
Γρήγορη αναφορά Κεφαλαίου 6....152

Μέρος II Η γλώσσα προγραμματισμού C#

7 Δημιουργία και διαχείριση κλάσεων και αντικειμένων....155

Τι είναι η ταξινόμηση....155
Σκοπός της ενθυλάκωσης....156
Ορισμός και χρήση μιας κλάσης....156
Έλεγχος της προσπελασιμότητας....158
Χρήση μεθόδων κατασκευής....159
Υπερφόρτωση μεθόδων κατασκευής....160
Στατικές μέθοδοι και δεδομένα....168
Δημιουργία κοινόχρηστου πεδίου....169
Δημιουργία στατικού πεδίου με τη λέξη-κλειδί *const*....169
Γρήγορη αναφορά Κεφαλαίου 7....174

8 Τι είναι οι τιμές και οι αναφορές....177

Αντιγραφή μεταβλητών τύπων τιμής και κλάσεων....177
Κενές τιμές και οιονεί κενοί τύποι....182
Χρήση οιονεί κενών τύπων....183
Οι ιδιότητες των οιονεί κενών τύπων....184
Χρήση παραμέτρων *ref* και *out*....184
Δημιουργία παραμέτρων *ref*....185
Δημιουργία παραμέτρων *out*....186
Οργάνωση της μνήμης του υπολογιστή....188
Χρήση της στοίβας και του σωρού....189
Η κλάση System.Object....190
Πλαισίωση....191
Κατάργηση πλαισίωσης....191
Ασφαλής μετατροπή τύπου δεδομένων....193
Ο τελεστής *is*....193
Ο τελεστής *as*....194
Γρήγορη αναφορά Κεφαλαίου 8....196

9 Δημιουργία τύπων τιμής με απαριθμήσεις και δομές 199

Απαριθμήσεις 199
Δήλωση μιας απαρίθμησης 199
Χρήση μιας απαρίθμησης.....200
Επιλογή κυριολεκτικών τιμών απαρίθμησης.....201
Επιλογή του υποκείμενου τύπου μιας απαρίθμησης.....202
Χρήση τύπων δομής 204
Δήλωση δομής 206
Οι διαφορές μεταξύ δομών και κλάσεων 207
Δήλωση μεταβλητών δομής.....208
Η απόδοση αρχικών τιμών σε δομή 209
Αντιγραφή μεταβλητών δομής 211
Γρήγορη αναφορά Κεφαλαίου 9 215

10 Χρήση πινάκων και συλλογών..... 217

Τι είναι ένας πίνακας; 217
Δήλωση μεταβλητών πίνακα 217
Δημιουργία παρουσιών πινάκων 218
Απόδοση αρχικών τιμών σε μεταβλητές πίνακα 219
Δημιουργία πίνακα υπονοούμενου τύπου 220
Πρόσβαση σε μεμονωμένα στοιχεία πίνακα 221
Επανάληψη μέσα σε πίνακα 222
Αντιγραφή πινάκων.....223
Τι είναι οι κλάσεις συλλογής; 224
Η κλάση συλλογής *ArrayList* 226
Η κλάση συλλογής *Queue* 228
Η κλάση συλλογής *Stack* 229
Η κλάση συλλογής *Hashtable* 230
Η κλάση συλλογής *SortedList* 231
Χρήση λιστών απόδοσης αρχικών τιμών σε συλλογές 232
Σύγκριση πινάκων και συλλογών 232
Χρήση κλάσεων συλλογής για χαρτοπαίγνια 233
Γρήγορη αναφορά Κεφαλαίου 10 238

11 Τι είναι οι πίνακες παραμέτρων 239

Χρήση ορισμάτων πινάκων 240
Δήλωση πινάκων παραμέτρων 241
Χρήση πινάκων παραμέτρων αντικειμένων 243
Χρήση πινάκων παραμέτρων 244
Γρήγορη αναφορά Κεφαλαίου 11 247

12 Χρήση της κληρονομικότητας 249

Τι είναι η κληρονομικότητα; 249
Εφαρμογή της κληρονομικότητας 250
Βασικές και παράγωγες κλάσεις 250
Κλήση μεθόδων κατασκευής βασικών κλάσεων 252
Ανάθεση κλάσεων 253
Δήλωση νέων μεθόδων 254
Δήλωση εικονικών μεθόδων 256
Υποσκελισμός μεθόδων 257
Προστασία πρόσβασης 259
Μέθοδοι επέκτασης 265
Γρήγορη αναφορά Κεφαλαίου 12 269

13 Δημιουργία διασυνδέσεων και ορισμός αφηρημένων κλάσεων 271

Τι είναι οι διασυνδέσεις; 271
Η σύνταξη των διασυνδέσεων 272
Περιορισμοί των διασυνδέσεων 273
Υλοποίηση μιας διασύνδεσης 273
Αναφορά σε μια κλάση μέσω της διασύνδεσής της 275
Χρήση πολλών διασυνδέσεων 276
Αφηρημένες κλάσεις 276
Αφηρημένες μέθοδοι 277
Σφραγισμένες κλάσεις 278
Σφραγισμένες μέθοδοι 278
Υλοποίηση επεκτάσιμου πλαισίου εργασίας 279
Σύνοψη των συνδυασμών λέξεων-κλειδιών 287
Γρήγορη αναφορά Κεφαλαίου 13 288

14 Συλλογή απορριμμάτων και διαχείριση πόρων 289
Έργα και ημέρες ενός αντικειμένου 289
Γραφή μεθόδων καταστροφής 290
Λόγοι χρήσης της συλλογής απορριμμάτων 292
Τρόπος λειτουργίας της συλλογής απορριμμάτων 293
Συστάσεις 294
Διαχείριση πόρων 294
Μέθοδοι αποκομιδής 295
Αποκομιδή ασφαλής από εξαιρέσεις 295
Η εντολή *using* 296
Κλήση της μεθόδου *Dispose* από μια μέθοδο καταστροφής 298
Δημιουργία κώδικα ασφαλούς από εξαιρέσεις 299
Γρήγορη αναφορά Κεφαλαίου 14 302

Μέρος III Δημιουργία συστατικών

15 Υλοποίηση ιδιοτήτων για την προσπέλαση πεδίων 307
Υλοποίηση της ενθυλάκωσης με τη χρήση μεθόδων 308
Τι είναι οι ιδιότητες; 310
Χρήση ιδιοτήτων 311
Ιδιότητες μόνο για ανάγνωση 312
Ιδιότητες μόνο για εγγραφή 312
Προσπελασιμότητα ιδιοτήτων 313
Οι περιορισμοί των ιδιοτήτων 314
Σωστή χρήση των ιδιοτήτων 315
Δήλωση ιδιοτήτων διασύνδεσης 316
Χρήση ιδιοτήτων σε μια εφαρμογή Windows 317
Δημιουργία αυτόματων ιδιοτήτων 319
Ανάθεση αρχικών τιμών σε αντικείμενα με τη χρήση ιδιοτήτων 320
Γρήγορη αναφορά Κεφαλαίου 15 324

16 Χρήση πινάκων αριθμοδεικτών 327
Τι είναι οι πίνακες αριθμοδεικτών; 327
Ένα παράδειγμα χωρίς πίνακες αριθμοδεικτών 327
Το ίδιο παράδειγμα με πίνακες αριθμοδεικτών 329
Μέθοδοι πρόσβασης πινάκων αριθμοδεικτών 331
Σύγκριση πινάκων αριθμοδεικτών και κοινών πινάκων 332
Πίνακες αριθμοδεικτών στις διασυνδέσεις 334
Χρήση πινάκων αριθμοδεικτών σε μια εφαρμογή Windows 335
Γρήγορη αναφορά Κεφαλαίου 16 340

17 Διακοπή ροής προγράμματος και χειρισμός συμβάντων 343
Δήλωση και χρήση εκπροσώπων 343
Το σενάριο του αυτοματοποιημένου εργοστασίου 344
Υλοποίηση του εργοστασίου χωρίς τη χρήση εκπροσώπων 344
Υλοποίηση του εργοστασίου με τη χρήση εκπροσώπου 345
Χρήση εκπροσώπων 348
Παραστάσεις λάμδα και εκπρόσωποι 351
Δημιουργία ενός προσαρμογέα μεθόδου 351
Χρήση μιας παράστασης λάμδα ως προσαρμογέα 352
Η μορφή των παραστάσεων λάμδα 353
Ενεργοποίηση γνωστοποίησης με συμβάντα 355
Δήλωση συμβάντος 355
Συνδρομή σε ένα συμβάν 356
Κατάργηση συνδρομής από ένα συμβάν 356
Ενεργοποίηση συμβάντος 357
Τι είναι τα συμβάντα διασύνδεσης WPF 357
Χρήση συμβάντων 359
Γρήγορη αναφορά Κεφαλαίου 17 361

18 Εισαγωγή στα Γενικά 365
Το πρόβλημα με τα αντικείμενα 365
Η λύση των Γενικών 367
Σύγκριση Γενικών και γενικευμένων κλάσεων 369
Γενικά και περιορισμοί 370
Δημιουργία μιας γενικής κλάσης 370
Θεωρία των δυαδικών δένδρων 370
Κατασκευή κλάσης δυαδικού δένδρου με τη χρήση Γενικών 373
Δημιουργία μιας γενικής μεθόδου 382
Ορισμός γενικής μεθόδου για τη δημιουργία δυαδικού δένδρου 383
Γρήγορη αναφορά Κεφαλαίου 18 386

19 Απαρίθμηση συλλογών 387
Απαρίθμηση των στοιχείων μιας συλλογής 387
Μη αυτόματη υλοποίηση απαριθμητή 389
Υλοποίηση της διασύνδεσης *IEnumerable* 393
Υλοποίηση απαριθμητή με τη χρήση επαναλήπτη 395
Ένας απλός επαναλήπτης 396
Ορισμός απαριθμητή για την κλάση *Tree<TItem>* με τη χρήση επαναλήπτη 398
Γρήγορη αναφορά Κεφαλαίου 19 400

20 Υποβολή ερωτημάτων σε δεδομένα της μνήμης με παραστάσεις ερωτημάτων 403

Τι είναι ένα Ερώτημα Ενσωματωμένο στη Γλώσσα (LINQ); 403
Χρήση LINQ σε μια εφαρμογή της C# 404
Επιλογή δεδομένων 406
Φιλτράρισμα δεδομένων.....409
Διάταξη, ομαδοποίηση, και συνάθροιση δεδομένων 409
Συνένωση δεδομένων 412
Χρήση τελεστών ερωτημάτων 413
Υποβολή ερωτημάτων σε αντικείμενα *Tree<TItem>*.....415
LINQ και καθυστερημένος υπολογισμός.....421
Γρήγορη αναφορά Κεφαλαίου 20 424

21 Υπερφόρτωση τελεστών 427

Ανάλυση των τελεστών.....427
Περιορισμοί τελεστών 428
Υπερφορτωμένοι τελεστές.....428
Δημιουργία συμμετρικών τελεστών 430
Σύνθετη ανάθεση τιμής.....432
Δήλωση τελεστών βηματικής αύξησης και βηματικής μείωσης.....433
Ορισμός ζευγών τελεστών.....435
Υλοποίηση ενός τελεστή 436
Οι τελεστές μετατροπής.....438
Παροχή ενσωματωμένων μετατροπών.....438
Υλοποίηση οριζόμενων από το χρήστη τελεστών μετατροπής.....439
Και πάλι η δημιουργία συμμετρικών τελεστών.....440
Προσθήκη τελεστή υπονοούμενης μετατροπής 441
Γρήγορη αναφορά Κεφαλαίου 21 443

Μέρος IV Εφαρμογές των Windows

22 Εισαγωγή στη λειτουργία Windows Presentation Foundation .. 447

Δημιουργία μιας εφαρμογής WPF.....447
Δημιουργία εφαρμογής Windows Presentation Foundation 448
Προσθήκη στοιχείων ελέγχου στη φόρμα 462
Χρήση στοιχείων ελέγχου WPF 462
Αλλαγή ιδιοτήτων δυναμικά.....471
Χειρισμός συμβάντων σε φόρμες WPF.....475
Επεξεργασία συμβάντων σε φόρμες WPF.....475
Γρήγορη αναφορά Κεφαλαίου 22 481

23 Μενού και πλαίσια διαλόγου 483
Οδηγίες και στυλ για μενού 483
Μενού και συμβάντα μενού 484
Δημιουργία μενού 484
Χειρισμός συμβάντων μενού 490
Μενού συντόμευσης 496
Δημιουργία μενού συντόμευσης 496
Στοιχεία ελέγχου κοινών πλαισίων διαλόγου 500
Χρήση της κλάσης *SaveFileDialog* 500
Γρήγορη αναφορά Κεφαλαίου 23 503

24 Επικύρωση 505
Επικύρωση δεδομένων 505
Στρατηγικές επικύρωσης καταχώρισης δεδομένων 505
Ένα παράδειγμα — πελατολόγιο 506
Επικύρωση με τη χρήση δέσμευσης δεδομένων 507
Αλλαγή της θέσης επικύρωσης 523
Γρήγορη αναφορά Κεφαλαίου 24 527

Μέρος V Διαχείριση δεδομένων

25 Αναζήτηση πληροφοριών σε μια βάση δεδομένων 531
Ερώτημα σε βάση δεδομένων με τη χρήση του ADO.NET 531
Η βάση δεδομένων Northwind 532
Δημιουργία της βάσης δεδομένων 532
Χρήση του ADO.NET για την αναζήτηση πληροφοριών παραγγελιών 535
Ερώτημα σε βάση δεδομένων με τη χρήση DLINQ 544
Ορισμός κλάσεων οντοτήτων 544
Δημιουργία και εκτέλεση ερωτήματος DLINQ 546
Καθυστερημένη και άμεση προσκόμιση 548
Συνένωση πινάκων και δημιουργία σχέσεων 549
Και πάλι η καθυστερημένη και η άμεση προσκόμιση 553
Ορισμός μιας προσαρμοσμένης κλάσης *DataContext* 554
Χρήση της DLINQ για την αναζήτηση πληροφοριών παραγγελιών 555
Γρήγορη αναφορά Κεφαλαίου 25 559

26 Εμφάνιση και επεξεργασία δεδομένων με τη χρήση δέσμευσης δεδομένων 561
Χρήση δέσμευσης δεδομένων με την DLINQ 561
Χρήση της DLINQ για την τροποποίηση δεδομένων 576

Ενημέρωση υπαρχόντων δεδομένων 576
Χειρισμός διενέξεων στις ενημερώσεις 577
Προσθήκη και διαγραφή δεδομένων 580
Γρήγορη αναφορά Κεφαλαίου 26 588

Μέρος VI Διαχείριση δεδομένων

27 Εισαγωγή στο ASP.NET 591

Το Διαδίκτυο ως υποδομή 592
Αιτήσεις και αποκρίσεις του διακομιστή Ιστού 592
Διαχείριση κατάστασης 593
Τι είναι το ASP.NET 593
Δημιουργία εφαρμογών Ιστού με το ASP.NET 595
Δόμηση εφαρμογής ASP.NET 596
Τα στοιχεία ελέγχου διακομιστή 607
Δημιουργία και χρήση θέματος 614
Γρήγορη αναφορά Κεφαλαίου 27 618

28 Τι είναι τα χειριστήρια επικύρωσης φορμών Ιστού 619

Σύγκριση επικύρωσης στο διακομιστή και στον πελάτη 619
Επικύρωση δεδομένων στο διακομιστή 620
Επικύρωση δεδομένων στο φυλλομετρητή Ιστού 620
Υλοποίηση της επικύρωσης στον πελάτη 621
Γρήγορη αναφορά Κεφαλαίου 28 628

29 Ασφάλεια τοποθεσίας Ιστού και προσπέλαση δεδομένων με φόρμες Ιστού 629

Διαχείριση της ασφάλειας 629
Ασφάλεια βασισμένη σε φόρμες 630
Υλοποίηση ασφάλειας βασισμένης σε φόρμες 630
Υποβολή ερωτημάτων και εμφάνιση δεδομένων 637
Το στοιχείο ελέγχου *GridView* των φορμών Ιστού 637
Εμφάνιση στοιχείων πελατών και ιστορικού παραγγελιών 638
Εμφάνιση δεδομένων ανά σελίδες 643
Επεξεργασία δεδομένων 644
Ενημέρωση γραμμών με ένα στοιχείο ελέγχου *GridView* 644
Μετακίνηση στις διάφορες φόρμες 646
Γρήγορη αναφορά Κεφαλαίου 29 653

30 Δημιουργία και χρήση υπηρεσίας Ιστού 655
Τι είναι μια υπηρεσία Ιστού; 655
Ο ρόλος του SOAP 656
Τι είναι η Γλώσσα Περιγραφής Υπηρεσιών Ιστού; 657
Μη λειτουργικές απαιτήσεις των υπηρεσιών Ιστού 657
Ο ρόλος της τεχνολογίας Windows Communication Foundation 659
Δημιουργία μιας υπηρεσίας Ιστού 659
Δημιουργία της υπηρεσίας Ιστού ProductService 660
Υπηρεσίες Ιστού, πελάτες, και μεσολαβητές 669
Επικοινωνία μέσω SOAP: ο δύσκολος τρόπος 669
Επικοινωνία μέσω SOAP: ο εύκολος τρόπος 669
Αξιοποίηση της υπηρεσίας Ιστού *ProductService* 670
Γρήγορη αναφορά Κεφαλαίου 30 676

Ευρετήριο 677

Κεφάλαιο 9

Δημιουργία τύπων τιμής με απαριθμήσεις και δομές

Μετά την ολοκλήρωση αυτού του κεφαλαίου, θα μπορείτε:

- Να δηλώνετε έναν τύπο απαρίθμησης.
- Να δημιουργείτε και να χρησιμοποιείτε έναν τύπο απαρίθμησης.
- Να δηλώνετε έναν τύπο δομής.
- Να δημιουργείτε και να χρησιμοποιείτε έναν τύπο δομής.

Στο Κεφάλαιο 8, "Τι είναι οι τιμές και οι αναφορές", μάθατε τα δύο θεμελιώδη είδη τύπων που υπάρχουν στη γλώσσα Microsoft Visual C#: τους *τύπους τιμής* (value types) και τους *τύπους αναφοράς* (reference types). Μια μεταβλητή τύπου τιμής διατηρεί την τιμή της απευθείας στη στοίβα, ενώ μια μεταβλητή τύπου αναφοράς περιέχει μια αναφορά προς ένα αντικείμενο που βρίσκεται στο σωρό. Στο Κεφάλαιο 7, "Δημιουργία και διαχείριση κλάσεων και αντικειμένων" μάθατε να γράφετε τις δικές σας κλάσεις, δημιουργώντας έτσι τους δικούς σας τύπους αναφοράς. Σε αυτό το κεφάλαιο θα μάθετε πώς να γράφετε τους δικούς σας τύπους τιμής.

Η C# υποστηρίζει δύο είδη τύπων τιμής: τις *απαριθμήσεις* και τις *δομές*. Θα τους εξετάσουμε με τη σειρά.

Απαριθμήσεις

Ας υποθέσουμε ότι θέλετε να αναπαραστήσετε τις εποχές του έτους σε ένα πρόγραμμα. Θα μπορούσατε να χρησιμοποιήσετε τους ακεραίους 0, 1, 2, και 3 για να αναπαραστήσετε την άνοιξη (spring), το καλοκαίρι (summer), το φθινόπωρο (fall), και το χειμώνα (winter) αντίστοιχα. Αυτό το σύστημα λειτουργεί αλλά δεν είναι και πολύ διαισθητικό. Αν χρησιμοποιούσατε την ακέραια τιμή 0 στον κώδικα, δε θα ήταν προφανές ότι κάποιο 0 αντιπροσωπεύει την άνοιξη. Αυτή η λύση δε θα ήταν επίσης πολύ ανθεκτική. Για παράδειγμα, δε θα σας εμπόδιζε τίποτε να χρησιμοποιήσετε οποιαδήποτε ακέραια τιμή αντί για τις 0, 1, 2, και 3. Η C# παρέχει μια καλύτερη λύση. Μπορείτε να δημιουργήσετε έναν τύπο απαρίθμησης (που ορισμένες φορές ονομάζεται τύπος *enum*), του οποίου οι τιμές περιορίζονται σε ένα σύνολο συμβολικών ονομάτων.

Δήλωση μιας απαρίθμησης

Για να δηλώσετε μια απαρίθμηση (enumeration), χρησιμοποιείτε τη λέξη-κλειδί *enum*, ακολουθούμενη από ένα σύνολο συμβόλων τα οποία καθορίζουν τις έγκυρες τιμές που μπορεί να πάρει ο τύ

πος, κλεισμένα μέσα σε άγκιστρα. Εδώ βλέπετε τον τρόπο δήλωσης μιας απαρίθμησης με το όνομα *Season* (Εποχή), του οποίου οι τιμές περιορίζονται στα συμβολικά ονόματα *Spring*, *Summer*, *Fall*, και *Winter*:

```
enum Season { Spring, Summer, Fall, Winter }
```

Χρήση μιας απαρίθμησης

Αφού δηλώσετε μια απαρίθμηση, μπορείτε να τη χρησιμοποιήσετε με τον ίδιο ακριβώς τρόπο όπως και τους άλλους τύπους. Αν το όνομα του τύπου απαρίθμησης είναι *Season*, μπορείτε να δημιουργήσετε μεταβλητές τύπου *Season*, πεδία τύπου *Season*, και παραμέτρους τύπου *Season*, όπως φαίνεται στο παρακάτω παράδειγμα:

```
enum Season { Spring, Summer, Fall, Winter }

class Example
{
   public void Method(Season parameter)
   {
      Season localVariable;
      ...
   }

   private Season currentSeason;
}
```

Για να χρησιμοποιήσετε την τιμή μιας μεταβλητής απαρίθμησης, πρέπει πρώτα να της αναθέσετε κάποια τιμή. Σε μια μεταβλητή απαρίθμησης μπορείτε να αναθέσετε μόνο μια τιμή που έχει οριστεί για την απαρίθμηση. Για παράδειγμα:

```
Season colorful = Season.Fall;
Console.WriteLine(colorful); // εμφανίζει 'Fall'
```

Σημείωση Όπως με όλους τους τύπους τιμής, μπορείτε να δημιουργήσετε μια οιονεί κενή (nullable) μιας μεταβλητής απαρίθμησης χρησιμοποιώντας τον τροποποιητή ?. Στη συνέχεια, μπορείτε να αναθέσετε στη μεταβλητή την τιμή *null*, καθώς και τις τιμές που ορίζονται στην απαρίθμηση:

```
Season? colorful = null;
```

Παρατηρήστε ότι πρέπει να γράψετε *Season.Fall* και όχι μόνο *Fall*. Όλα τα κυριολεκτικά ονόματα απαριθμήσεων έχουν εμβέλεια μέσα στους τύπους απαρίθμησης που δηλώνονται. Αυτό είναι πολύ χρήσιμο επειδή επιτρέπει σε διαφορετικούς τύπους απαρίθμησης να περιλαμβάνουν κυριολεκτικά (literals) με το ίδιο όνομα. Επίσης παρατηρήστε ότι, όταν εμφανίζετε τα περιεχόμενα μιας μεταβλητής απαρίθμησης με τη μέθοδο *Console.WriteLine*, ο μεταγλωττιστής παράγει κώδικα που γράφει στην κονσόλα το όνομα του κυριολεκτικού του οποίου η τιμή ταιριάζει με την τιμή της μεταβλητής.

Αν χρειαστεί, μπορείτε να μετατρέψετε ρητά μια μεταβλητή απαρίθμησης σε συμβολοσειρά που αντιπροσωπεύει την τρέχουσα τιμή της, με τη βοήθεια της ενσωματωμένης μεθόδου *ToString* την οποία διαθέτουν αυτόματα όλοι οι τύποι απαρίθμησης. Για παράδειγμα:

```
string name = colorful.ToString();
Console.WriteLine(name); // επίσης εμφανίζει 'Fall'
```

Πολλοί από τους βασικούς τελεστές που μπορείτε να χρησιμοποιήσετε σε μεταβλητές ακεραίων μπορούν επίσης να χρησιμοποιηθούν στις μεταβλητές απαρίθμησης (εκτός από τους τελεστές *bitwise* και *shift*, τους οποίους θα δούμε στο Κεφάλαιο 16, "Χρήση πινάκων αριθμοδεικτών"). Για παράδειγμα, με τον τελεστή == μπορείτε να ελέγξετε αν δύο μεταβλητές απαρίθμησης του ίδιου τύπου είναι ίσες, και μπορείτε επίσης να εκτελέσετε αριθμητικές πράξεις σε μια μεταβλητή απαρίθμησης (αν και το αποτέλεσμα μπορεί να μην έχει πάντοτε νόημα!).

Επιλογή κυριολεκτικών τιμών απαρίθμησης

Εσωτερικά, μια απαρίθμηση συσχετίζει μια ακέραια τιμή με κάθε στοιχείο. Εξ ορισμού, η αρίθμηση αρχίζει από το 0 για το πρώτο στοιχείο και αυξάνεται κατά 1. Μπορείτε να ανακτήσετε την υποκείμενη ακέραια τιμή μιας μεταβλητής απαρίθμησης. Για το σκοπό αυτόν, πρέπει να τη μετατρέψετε στον υποκείμενο τύπο της. Από την εξέταση της κατάργησης της πλαισίωσης στο Κεφάλαιο 8, θυμηθείτε ότι η μετατροπή τύπου (cast) μετατρέπει τα δεδομένα από έναν τύπο σε έναν άλλο, με την προϋπόθεση ότι η μετατροπή είναι έγκυρη και έχει νόημα. Για παράδειγμα, το επόμενο τμήμα κώδικα θα εμφανίσει την τιμή 2 και όχι τη λέξη *Fall* (η *Spring* είναι η τιμή 0, η *Summer* η 1, η *Fall* η 2, και η *Winter* η 3):

```
enum Season { Spring, Summer, Fall, Winter }
...
Season colorful = Season.Fall;
Console.WriteLine((int)colorful); // εμφανίζει '2'
```

Αν προτιμάτε, μπορείτε να συσχετίσετε μια συγκεκριμένη ακέραια σταθερά (όπως η τιμή 1) με ένα κυριολεκτικό (literal) απαρίθμησης (σαν τη μεταβλητή *Spring*), όπως στο παρακάτω παράδειγμα:

```
enum Season { Spring = 1, Summer, Fall, Winter }
```

Σημαντικό Η ακέραια τιμή την οποία δίνετε ως αρχική σε ένα κυριολεκτικό απαρίθμησης, πρέπει να είναι μια σταθερή τιμή κατά το χρόνο μεταγλώττισης (όπως η τιμή 1).

Αν δε δώσετε ρητά σε ένα κυριολεκτικό απαρίθμησης μια σταθερή ακέραια τιμή, ο μεταγλωττιστής τού δίνει μια τιμή μεγαλύτερη κατά 1 σε σχέση με την τιμή του προηγούμενου κυριολεκτικού της απαρίθμησης εκτός από το πρώτο, στο οποίο δίνει την προεπιλεγμένη τιμή *0*. Στο προηγούμενο παράδειγμα, οι υποκείμενες τιμές των κυριολεκτικών απαρίθμησης *Spring*, *Summer*, *Fall*, και *Winter* είναι αντίστοιχα οι τιμές 1, 2, 3, και 4.

Επιτρέπεται να δώσετε την ίδια υποκείμενη τιμή σε περισσότερα από ένα κυριολεκτικά απαρίθμησης. Για παράδειγμα, στην Αγγλία, το κυριολεκτικό απαρίθμησης *Fall* αναφέρεται ως *Autumn*. Μπορείτε να συμπεριλάβετε και τις δύο γλωσσικές συνήθειες ως εξής:

```
enum Season { Spring, Summer, Autumn = Fall, Winter }
```

Επιλογή του υποκείμενου τύπου μιας απαρίθμησης

Όταν δηλώνετε έναν τύπο απαρίθμησης, τα κυριολεκτικά της απαρίθμησης παίρνουν τιμές τύπου *int*. Μπορείτε να βασίσετε το δικό σας τύπο απαρίθμησης και σε κάποιο διαφορετικό υποκείμενο ακέραιο τύπο. Για παράδειγμα, για να δηλώσετε ότι ο υποκείμενος τύπος της απαρίθμησης *Season* είναι ο *short* αντί του *int*, μπορείτε να χρησιμοποιήσετε την παρακάτω εντολή:

```
enum Season : short { Spring, Summer, Fall, Winter }
```

Ο κύριος λόγος για να κάνετε κάτι τέτοιο είναι η οικονομία μνήμης· ο τύπος *int* καταλαμβάνει περισσότερη μνήμη από τον τύπο *short* και, αν δε χρειάζεστε ολόκληρο το εύρος των τιμών του τύπου *int*, ένας μικρότερος τύπος δεδομένων είναι πιο εύλογος.

Μπορείτε να βασίσετε μια απαρίθμηση σε οποιονδήποτε από τους επόμενους οκτώ ακέραιους τύπους: *byte*, *sbyte*, *short*, *ushort*, *int*, *uint*, *long*, ή *ulong*. Οι τιμές όλων των κυριολεκτικών της απαρίθμησης πρέπει να χωρούν στο εύρος του επιλεγμένου βασικού τύπου. Για παράδειγμα, αν βασίσετε μια απαρίθμηση στον τύπο δεδομένων *byte*, μπορείτε να έχετε ένα μέγιστο αριθμό 256 κυριολεκτικών (ξεκινώντας από το 0).

Τώρα που γνωρίζετε πώς δηλώνετε μια απαρίθμηση, το επόμενο βήμα είναι να τη χρησιμοποιήσετε. Στην επόμενη άσκηση, θα δουλέψετε με μια εφαρμογή κονσόλας για να δηλώσετε και να χρησιμοποιήσετε μια κλάση απαρίθμησης που αναπαριστά τους μήνες του έτους.

Δημιουργήστε και χρησιμοποιήστε έναν τύπο απαρίθμησης

1. Ξεκινήστε το Microsoft Visual Studio 2008 αν δεν είναι ήδη ανοιχτό.
2. Ανοίξτε το έργο *StructsAndEnums* από το φάκελο \Microsoft Press\Visual CSharp Step By Step\Chapter 9\StructsAndEnums του φακέλου Documents του λογαριασμού σας.
3. Στο παράθυρο *Διορθωτή κώδικα και κειμένου*, εμφανίστε το αρχείο Month.cs.

   Το πηγαίο αρχείο περιέχει έναν κενό χώρο ονομάτων με το όνομα *StructsAndEnums* (δομές και απαριθμήσεις).
4. Στο χώρο ονομάτων *StructsAndEnums*, προσθέστε έναν τύπο απαρίθμησης με το όνομα *Month* για να αναπαραστήσετε τους μήνες του έτους, όπως βλέπετε εδώ με έντονα γράμματα. Τα 12 κυριολεκτικά της απαρίθμησης *Month* είναι οι μήνες January (Ιανουάριος) έως December (Δεκέμβριος).

```
namespace StructsAndEnums
{
   enum Month
   {
      January, February, March, April,
      May, June, July, August,
      September, October, November, December
   }
}
```

5. Εμφανίστε το αρχείο Program.cs στο παράθυρο *Διορθωτή κώδικα και κειμένου*.

Όπως στις ασκήσεις των προηγούμενων κεφαλαίων, η μέθοδος *Main* καλεί τη μέθοδο *Entrance* και παγιδεύει τυχόν εξαιρέσεις που προκύπτουν.

6. Στο παράθυρο *Διορθωτή κώδικα και κειμένου*, προσθέστε στη μέθοδο *Entrance* μια εντολή για να δηλώσετε μια μεταβλητή τύπου *Month* με το όνομα *first* και να της δώσετε ως αρχική τιμή την τιμή *Month.January*. Προσθέστε μία ακόμη εντολή που θα εμφανίζει την τιμή της πρώτης μεταβλητής στην κονσόλα.

Η μέθοδος *Entrance* θα πρέπει να μοιάζει με αυτή:

```
static void Entrance()
{
   Month first = Month.January;
   Console.WriteLine(first);
}
```

Σημείωση Όταν πληκτρολογήσετε την τελεία μετά από το *Month*, η λειτουργία Intellisense θα εμφανίσει αυτόματα όλες τις τιμές της απαρίθμησης *Month*.

7. Από το μενού *Debug*, επιλέξτε τη διαταγή *Start Without Debugging*.

Το Visual Studio 2008 μεταγλωττίζει και εκτελεί το πρόγραμμα. Βεβαιωθείτε ότι στην κονσόλα εμφανίζεται η λέξη *January*.

8. Πατήστε το πλήκτρο Enter για να κλείσετε το πρόγραμμα και να επιστρέψετε στο περιβάλλον προγραμματισμού του Visual Studio 2008.

9. Προσθέστε δύο ακόμη εντολές στη μέθοδο *Entrance*, για να αυξήσετε τη μεταβλητή *first* και να εμφανίσετε τη νέα της τιμή στην κονσόλα, όπως βλέπετε εδώ με έντονα γράμματα:

```
static void Entrance()
{
   Month first = Month.January;
   Console.WriteLine(first);
   first++;
   Console.WriteLine(first);
}
```

10. Από το μενού *Debug*, επιλέξτε τη διαταγή *Start Without Debugging*.

Το Visual Studio 2008 μεταγλωττίζει και εκτελεί το πρόγραμμα. Βεβαιωθείτε ότι στην κονσόλα εμφανίζονται οι λέξεις *January* και *February*.

Παρατηρήστε ότι η εκτέλεση μιας μαθηματικής πράξης (όπως η αύξηση της τιμής) σε μια μεταβλητή απαρίθμησης αλλάζει την εσωτερική ακέραια τιμή της μεταβλητής. Στην οθόνη εμφανίζεται η αντίστοιχη τιμή της απαρίθμησης.

11. Πατήστε το πλήκτρο Enter για να κλείσετε το πρόγραμμα και να επιστρέψετε στο περιβάλλον προγραμματισμού του Visual Studio 2008.

12. Τροποποιήστε την πρώτη εντολή της μεθόδου *Entrance* ώστε η μεταβλητής *first* να παίρνει ως αρχική την τιμή *Month.December*, όπως βλέπετε εδώ με έντονα γράμματα:

```
static void Entrance()
{
   Month first = Month.December;
   Console.WriteLine(first);
   first++;
   Console.WriteLine(first);
}
```

13. Από το μενού *Debug*, επιλέξτε τη διαταγή *Start Without Debugging*.

Το Visual Studio 2008 μεταγλωττίζει και εκτελεί το πρόγραμμα. Αυτή τη φορά στην κονσόλα εμφανίζεται η λέξη *December*, ακολουθούμενη από τον αριθμό *12*. Αν και μπορείτε να εκτελέσετε αριθμητικές πράξεις σε μια απαρίθμηση, αν τα αποτελέσματα της πράξης βρεθούν έξω από το εύρος τιμών που ορίστηκε για την απαρίθμηση, το μόνο που μπορεί να κάνει το σύστημα χρόνου εκτέλεσης είναι να χειριστεί τη μεταβλητή ως την αντίστοιχη ακέραια τιμή.

14. Πατήστε το πλήκτρο Enter για να κλείσετε το πρόγραμμα και να επιστρέψετε στο περιβάλλον προγραμματισμού του Visual Studio 2008.

Χρήση τύπων δομής

Στο Κεφάλαιο 8, είδαμε ότι οι κλάσεις ορίζουν τύπους αναφοράς οι οποίοι δημιουργούνται πάντοτε στο σωρό. Σε μερικές περιπτώσεις, η κλάση μπορεί να περιέχει τόσο λίγα δεδομένα ώστε η επιβάρυνση από τη διαχείριση του σωρού καταλήγει δυσανάλογα μεγάλη. Σε αυτές τις περιπτώσεις, είναι καλύτερα να ορίσετε τον τύπο ως δομή (structure). Επειδή οι δομές αποθηκεύονται στη στοίβα, η επιβάρυνση από τη διαχείριση της μνήμης συχνά μειώνεται, με την προϋπόθεση ότι η δομή είναι αρκετά μικρή.

Μια δομή μπορεί να έχει τα δικά της πεδία, μεθόδους, και μεθόδους κατασκευής όπως ακριβώς μια κλάση και αντίθετα με μια απαρίθμηση.

Συνήθεις τύποι δομών

Ίσως δεν το έχετε αντιληφθεί, αλλά έχετε ήδη χρησιμοποιήσει δομές σε προηγούμενες ασκήσεις αυτού του βιβλίου. Στη γλώσσα C#, οι πρωτογενείς αριθμητικοί τύποι *int*, *long*, και *float* είναι ψευδώνυμα των δομών *System.Int32*, *System.Int64*, και *System.Single* αντίστοιχα. Αυτές οι δομές διαθέτουν πεδία και μεθόδους, και μπορείτε να καλέσετε μεθόδους για μεταβλητές και κυριολεκτικά αυτών των τύπων. Για παράδειγμα, όλες αυτές οι δομές διαθέτουν μια μέθοδο *ToString*, που μπορεί να μετατρέψει μια αριθμητική τιμή στην ισοδύναμη αναπαράσταση συμβολοσειράς. Οι παρακάτω εντολές είναι όλες έγκυρες στη γλώσσα C#:

```
int i = 99;
Console.WriteLine(i.ToString());
Console.WriteLine(55.ToString());
float f = 98.765F;
Console.WriteLine(f.ToString());
Console.WriteLine(98.765F.ToString());
```

Δε θα δείτε αυτή τη χρήση της μεθόδου *ToString* και πολύ συχνά, επειδή η μέθοδος *Console.WriteLine* την καλεί αυτόματα κάθε φορά που χρειάζεται. Η χρήση των στατικών μεθόδων που αποκαλύπτουν αυτές οι δομές είναι πολύ πιο συνηθισμένη. Για παράδειγμα, σε προηγούμενα κεφάλαια χρησιμοποιήσαμε τη στατική μέθοδο *int.Parse* για τη μετατροπή μιας συμβολοσειράς στην αντίστοιχη ακέραια τιμή της. Αυτό που κάνατε στην πραγματικότητα είναι να καλείτε τη μέθοδο *Parse* της δομής *Int32*:

```
string s = "42";
int i = int.Parse(s);  // Ακριβώς ίδιο με το Int32.Parse
```

Οι δομές αυτές περιλαμβάνουν επίσης μερικά χρήσιμα στατικά πεδία. Για παράδειγμα, η τιμή *Int32.MaxValue* είναι η μέγιστη τιμή η οποία μπορεί να αποθηκευτεί στον τύπο *int*, και η *Int32.MinValue* είναι η μικρότερη τιμή που μπορείτε να αποθηκεύσετε στον τύπο *int*.

Ο παρακάτω πίνακας δείχνει τους πρωτογενείς τύπους της γλώσσας C#, τους αντίστοιχους ισοδύναμους τύπους στο Πλαίσιο Εφαρμογών .NET, και το αν κάθε τύπος είναι κλάση ή δομή.

Λέξη-κλειδί	Ισοδύναμος τύπος	Κλάση ή δομή
bool	System.Boolean	Δομή
byte	System.Byte	Δομή
decimal;	System.Decimal	Δομή
double	System.Double	Δομή
float	System.Single	Δομή
int	System.Int32	Δομή
long	System.Int64	Δομή
object	System.Object	Κλάση
sbyte	System.Sbyte	Δομή
short	System.Int16	Δομή
string	System.String	Κλάση
uint	System.UInt32	Δομή
ulong	System.UInt64	Δομή
ushort	System.UInt16	Δομή

Δήλωση δομής

Για να δηλώσετε το δικό σας τύπο τιμής δομής, χρησιμοποιείτε τη λέξη-κλειδί *struct* ακολουθούμενη από το όνομα του τύπου, και μετά από το σώμα της δομής ανάμεσα σε αριστερό και δεξιό άγκιστρο. Για παράδειγμα, ακολουθεί μια *δομή* με το όνομα *Time* (χρόνος), που περιέχει τρία δημόσια πεδία τύπου *int* με τα ονόματα *hours* (ώρες), *minutes* (λεπτά), και *seconds* (δευτερόλεπτα):

```
struct Time
{
   public int hours, minutes, seconds;
}
```

Όπως στις κλάσεις, στις περισσότερες περιπτώσεις δε συνιστάται ο ορισμός των πεδίων μιας δομής ως δημοσίων (*public*)· δεν υπάρχει τρόπος να εξασφαλίσετε ότι τα δημόσια πεδία θα περιέχουν έγκυρες τιμές. Για παράδειγμα, ο καθένας θα μπορούσε να δώσει στα πεδία *minutes* ή *seconds* τιμές μεγαλύτερες από 60. Μια καλύτερη ιδέα είναι να ορίσετε τα πεδία ως ιδιωτικά (*private*) και να δημιουργήσετε τη δική σας δομή με μεθόδους κατασκευής και μεθόδους απόδοσης αρχικών τιμών και χειρισμού των πεδίων αυτών, όπως φαίνεται στο παρακάτω παράδειγμα:

```
struct Time
{
   public Time(int hh, int mm, int ss)
   {
      hours = hh % 24;
      minutes = mm % 60;
      seconds = ss % 60;
   }

   public int Hours()
   {
      return hours;
   }
   ...
   private int hours, minutes, seconds;
}
```

Σημείωση Εξ ορισμού, δεν μπορείτε να χρησιμοποιήσετε πολλούς από τους συνήθεις τελεστές στους δικούς σας τύπους δομών. Για παράδειγμα, δεν μπορείτε να χρησιμοποιήσετε τελεστές όπως ο τελεστής ισότητας (==) και ο τελεστής "διάφορο" (!=) στις δικές σας μεταβλητές δομών. Όμως, μπορείτε ρητά να δηλώσετε και να υλοποιήσετε τελεστές για τους δικούς τους τύπους δομών. Τη σύνταξη γι' αυτό θα τη δούμε στο Κεφάλαιο 21, "Υπερφόρτωση τελεστών".

Χρησιμοποιήστε δομές για να υλοποιήσετε απλές έννοιες των οποίων το βασικό γνώρισμα είναι η τιμή τους. Για παράδειγμα, ο τύπος *int* είναι ένας τύπος τιμής επειδή το βασικό του γνώρισμα είναι η τιμή του. Αν έχετε δύο μεταβλητές τύπου *int* που περιέχουν την ίδια τιμή (όπως η τιμή 42), αυτές είναι ισοδύναμες. Όταν αντιγράφετε μια μεταβλητή τύπου τιμής, παίρνετε δύο αντίγραφα της ίδιας τιμής. Όταν όμως αντιγράφετε μια μεταβλητή τύπου αναφοράς, παίρνετε δύο αναφορές προς το ίδιο αντικείμενο. Συμπερασματικά, χρησιμοποιήστε δομές για μικρές τιμές δεδομένων όπου η αντι-

γραφή της τιμής είναι το ίδιο ή σχεδόν το ίδιο αποδοτική με την αντιγραφή μιας διεύθυνσης. Χρησιμοποιήστε κλάσεις για πιο σύνθετα δεδομένα ώστε να έχετε τη δυνατότητα να επιλέξετε την αντιγραφή μόνο της διεύθυνσης της πραγματικής τιμής όταν θέλετε να βελτιώσετε την απόδοση του κώδικά σας.

Οι διαφορές μεταξύ δομών και κλάσεων

Οι δομές και οι κλάσεις είναι συντακτικά παρόμοιες, αλλά παρουσιάζουν μερικές σημαντικές διαφορές. Ας δούμε μερικές από αυτές.

- Δεν μπορείτε να δηλώσετε μια προεπιλεγμένη μέθοδο κατασκευής (μια μέθοδο κατασκευής χωρίς παραμέτρους) για μια δομή. Το παρακάτω παράδειγμα θα μεταγλωττιζόταν αν η *Time* ήταν κλάση, επειδή όμως η *Time* είναι δομή δεν πρόκειται να μεταγλωττιστεί:

```
struct Time
{
   public Time() { ... }  // σφάλμα χρόνου μεταγλώττισης
   ...
{
```

  Ο λόγος για τον οποίο δεν μπορείτε να δηλώσετε τη δική σας προεπιλεγμένη μέθοδο κατασκευής σε μια δομή είναι επειδή ο μεταγλωττιστής δημιουργεί *πάντοτε* μια τέτοια. Σε μια κλάση, ο μεταγλωττιστής δημιουργεί την προεπιλεγμένη μέθοδο κατασκευής μόνο στην περίπτωση που δε θα τη γράψετε σεις. Η προεπιλεγμένη μέθοδος κατασκευής που δημιουργεί ο μεταγλωττιστής για μια δομή δίνει πάντοτε στα πεδία τις τιμές *0*, *false*, ή *null* — όπως ακριβώς για μια κλάση. Επομένως, πρέπει να εξασφαλίσετε ότι μια τιμή δομής η οποία δημιουργείται από την προεπιλεγμένη μέθοδο κατασκευής θα συμπεριφέρεται εύλογα και θα έχει έννοια γι' αυτές τις προεπιλεγμένες τιμές. Αν δε θέλετε να χρησιμοποιήσετε αυτές τις προεπιλεγμένες τιμές, μπορείτε να δώσετε στα πεδία διαφορετικές αρχικές τιμές, δημιουργώντας μια μη προεπιλεγμένη μέθοδο κατασκευής. Αν όμως δε δώσετε αρχικές τιμές σε κάποιο πεδίο της μεθόδου κατασκευής, ο μεταγλωττιστής δεν πρόκειται να το κάνει για λογαριασμό σας. Αυτό σημαίνει ότι πρέπει να δώσετε ρητά αρχικές τιμές σε όλα τα πεδία όλων των μεθόδων κατασκευής της δομής, διαφορετικά θα πάρετε ένα σφάλμα χρόνου μεταγλώττισης. Για παράδειγμα, παρόλο που αν η *Time* ήταν κλάση το παρακάτω παράδειγμα θα μεταγλωττιζόταν και θα έδινε αρχική τιμή 0 στο πεδίο *seconds*, επειδή η *Time* είναι δομή το πρόγραμμα δε θα μεταγλωττιστεί:

```
struct Time
{
   public Time(int hh, int mm)
   {
      hours = hh;
      minutes = mm;
   } // σφάλμα μεταγλώττισης: η seconds δεν πήρε αρχική τιμή
   ...
   private int hours, minutes, seconds;
}
```

- Σε μια κλάση, μπορείτε να δώσετε αρχικές τιμές σε πεδία παρουσιών στη θέση που δηλώνονται. Αυτό είναι αδύνατον για μια δομή. Για παράδειγμα, ο παρακάτω κώδικας θα μεταγλωττιζόταν αν η *Time* ήταν κλάση, επειδή όμως είναι δομή προκαλείται ένα σφάλμα χρόνου μεταγλώττισης:

```
struct Time
{

   ...
   private int hours = 0;   // σφάλμα χρόνου μεταγλώττισης
   private int minutes;
   private int seconds;
}
```

Ο παρακάτω πίνακας συνοψίζει τις βασικές διαφορές μεταξύ δομών και κλάσεων.

Ερώτηση	Δομή	Κλάση
Είναι τύπος τιμής ή τύπος αναφοράς;	Η δομή είναι τύπος τιμής.	Η κλάση είναι τύπος αναφοράς.
Οι παρουσίες βρίσκονται στο σωρό ή στη στοίβα;	Οι παρουσίες των δομών ονομάζονται *τιμές* και βρίσκονται στη στοίβα.	Οι παρουσίες των κλάσεων ονομάζονται *αντικείμενα* και βρίσκονται στο σωρό.
Μπορείτε να δηλώσετε μια προεπιλεγμένη μέθοδο κατασκευής;	Όχι	Ναι
Αν δηλώσετε τη δική σας μέθοδο κατασκευής, ο μεταγλωττιστής θα εξακολουθήσει να δημιουργεί την προεπιλεγμένη μέθοδο κατασκευής;	Ναι	Όχι
Αν δε δώσετε αρχική τιμή σε ένα πεδίο της δικής σας μεθόδου κατασκευής, ο μεταγλωττιστής θα του δώσει αυτόματα αρχική τιμή;	Όχι	Ναι
Επιτρέπεται να δώσετε αρχικές τιμές στα πεδία παρουσιών κατά τη δήλωσή τους;	Όχι	Ναι

Εκτός από αυτές, υπάρχουν και άλλες διαφορές μεταξύ κλάσεων και δομών που έχουν σχέση με την κληρονομικότητα. Αυτές οι διαφορές εξετάζονται στο Κεφάλαιο 12, "Χρήση της κληρονομικότητας". Τώρα που γνωρίζετε πώς να δηλώνετε δομές, το επόμενο βήμα σας είναι να τις χρησιμοποιήσετε για να δημιουργήσετε τιμές.

Δήλωση μεταβλητών δομής

Αφού ορίσετε έναν τύπο δομής, μπορείτε να τον χρησιμοποιήσετε με τον ίδιο ακριβώς τρόπο όπως και οποιονδήποτε άλλο. Για παράδειγμα, αν έχετε ορίσει τη δομή *Time*, μπορείτε να δημιουργήσετε μεταβλητές, πεδία, και παραμέτρους τύπου *Time*, όπως φαίνεται στο επόμενο παράδειγμα:

```
struct Time
{
   ...
   private int hours, minutes, seconds;
```

```
}

class Example
{
   public void Method(Time parameter)
   {
      Time localVariable;
      ...
   }

   private Time currentTime;
}
```

Σημείωση Μπορείτε να δημιουργήσετε μια οιονεί κενή (nullable) έκδοση μιας δομής χρησιμοποιώντας τον τροποποιητή ?. Στη συνέχεια, μπορείτε να αναθέσετε την τιμή *null* στη μεταβλητή:

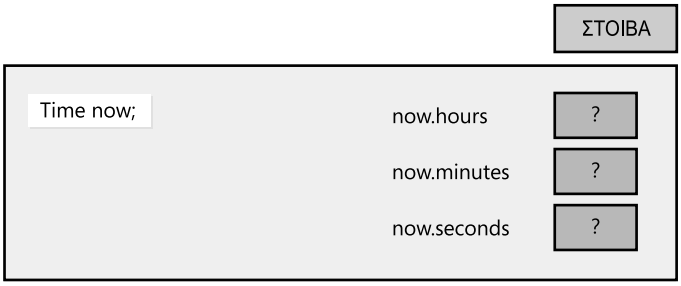
```
Time? currentTime = null;
```

Η απόδοση αρχικών τιμών σε δομή

Σε προηγούμενη ενότητα περιγράψαμε τον τρόπο απόδοσης αρχικών τιμών στα πεδία δομής με τη χρήση μιας μεθόδου κατασκευής. Επειδή όμως οι δομές είναι τύποι τιμής, μπορείτε να δημιουργήσετε μεταβλητές τύπου δομής χωρίς να καλέσετε κάποια μέθοδο κατασκευής, όπως στο παρακάτω παράδειγμα:

```
Time now;
```

Στο παράδειγμα αυτό, δημιουργείται η μεταβλητή *now* αλλά δε δίνονται αρχικές τιμές στα πεδία της. Κάθε προσπάθεια πρόσβασης στις τιμές των πεδίων θα έχει ως αποτέλεσμα ένα σφάλμα του μεταγλωττιστή. Στην επόμενη εικόνα φαίνεται η κατάσταση των πεδίων της μεταβλητής *now*:



Αν καλέσετε μια μέθοδο κατασκευής, οι κανόνες των μεθόδων κατασκευής δομών που περιγράψαμε προηγουμένως εγγυώνται ότι όλα τα πεδία της δομής θα πάρουν αρχικές τιμές:

```
Time now = new Time();
```

Microsoft

Visual C# 2008

Βήμα Βήμα

Ο πρακτικός οδηγός για να μάθετε βήμα-βήμα πώς να δημιουργείτε εφαρμογές με τη Visual C#.

Μάθετε τη Visual C# 2008 — ένα βήμα τη φορά. Αυτό το πρακτικό βοήθημα, ιδανικό για προγραμματιστές με βασικές ικανότητες στον προγραμματισμό υπολογιστών, περιέχει όλα τα πρακτικά μαθήματα που χρειάζεστε για να δημιουργήσετε συστατικά της C# και εφαρμογές Windows®.

Θα μάθετε τα εξής:

- Να δηλώνετε μεταβλητές, να δημιουργείτε τελεστές, και να καλείτε μεθόδους
- Να συλλαμβάνετε και να χειρίζεστε σφάλματα εξαιρέσεων
- Να διαχειρίζεστε πόρους χρησιμοποιώντας μεθόδους καταστροφής (destructors) και αποκομιδή απορριμμάτων
- Να ορίζετε ιδιότητες και πίνακες αριθμοδεικτών (indexers), και να χειρίζεστε συμβάντα (events)
- Να χρησιμοποιείτε γενικά (generic) για να ορίζετε κλάσεις και συλλογές με ασφάλεια τύπων
- Να χειρίζεστε δεδομένα χρησιμοποιώντας το Microsoft ADO.NET και τη γλώσσα LINQ (Language Integrated Query)
- Να δημιουργείτε αλληλεπιδραστικές εφαρμογές Ίστού και υπηρεσίες Ίστού
- Να δημιουργείτε εμπλουτισμένες διασυνδέσεις με το χρήστη μέσω της τεχνολογίας Windows Presentation Foundation (WPF)

Το CD περιλαμβάνει:

- Αρχεία εξάσκησης και παραδείγματα κώδικα
- Ηλεκτρονική έκδοση του βιβλίου (στα αγγλικά) με δυνατότητα αναζήτησης κειμένου

Για πληροφορίες σχετικά με τις ***απαιτήσεις συστήματος*** *δείτε την ενότητα "Εισαγωγή" του βιβλίου.*

Ο συγγραφέας

Ο **John Sharp** είναι Τεχνολόγος Μηχανικός στην Content Master, μέλος του ομίλου εταιρειών CM Group Ltd, μια εταιρεία τεχνικών εκδόσεων και συμβούλων. Ο John είναι ειδικός στην ανάπτυξη εφαρμογών με το Microsoft .NET Framework και τη διαλειτουργικότητα, και έχει δημιουργήσει εκπαιδευτικά προγράμματα, τεχνικές προδιαγραφές, και παρουσιάσεις σχετικά με τα κατανεμημένα συστήματα, τις υπηρεσίες Ίστού, και τη γλώσσα C#. Έχει γράψει αρκετά δημοφιλή βιβλία, όπως τα *Microsoft Windows Communication Foundation Step by Step* και *Microsoft Visual J# Core Reference*.

ΣΕΙΡΕΣ ΒΙΒΛΙΩΝ ΤΗΣ MICROSOFT PRESS

Σειρά Βήμα-Βήμα

- Πρακτικές ασκήσεις που καλύπτουν θεμελιώδεις τεχνικές και λειτουργίες
- Περιλαμβάνουν αρχεία εξάσκησης σε CD
- Προετοιμάζουν και πληροφορούν νέους προγραμματιστές

Developer Reference (στα αγγλικά)

- Κάλυψη σε βάθος των βασικών θεμάτων
- Εκτεταμένα ρεαλιστικά παραδείγματα κώδικα
- Παρέχουν επάρκεια γνώσεων επαγγελματικού επιπέδου στις τεχνολογίες της Microsoft

Focused Topics (στα αγγλικά)

- Κάλυψη σε βάθος προχωρημένων τεχνικών και δυνατοτήτων
- Εκτεταμένα, προσαρμόσιμα παραδείγματα κώδικα
- Παρέχουν το μέγιστο επίπεδο γνώσεων στις τεχνολογίες της Microsoft

Microsoft® Visual Studio® 2008

Ένα βιβλίο από τη Microsoft®

Επισκεφθείτε μας στο Internet
www.klidarithmos.gr

ISBN 978-960-461-205-5

εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ

Δομοκού 4, Σταθμός Λαρίσης, 10440 ΑΘΗΝΑ, Τηλ. 210-5227535