Introducere

De-a lungul anilor am avut norocul să proiectez şi să scriu programe pentru producţie în Microsoft Visual Basic, care se utilizează în prezent în cadrul mai multor companii nominalizate pe lista Fortune 500, din Statele Unite. De asemenea predau, angajez şi mă ocup de specialiştii în acest domeniu, fapt ce mi-a permis să-mi dau seama de modul de a gândi al multor programatori în Visual Basic. În orele mele de la universitate am realizat în mod repetat care sunt aspectele programării în Visual Basic care sunt înţelese de către programatori, ca şi domenii precum programarea orientată pe obiecte, la care unii programatori se mai încurcă. Cartea de faţă vă va urca la un nivel superior al programării, dezvăluind domenii ale acesteia pe care nu le cunoaşteţi prea bine şi, în acelaşi timp, îmbunătăţindu-vă cunoştinţele în domeniile în care deja lucraţi zilnic.

Cui i se adresează această carte

Această carte a fost scrisă pentru programatorii în Visual Basic de către un programator în Visual Basic. În expunerea mea cu privire la modul cum trebuie să lucraţi cu Visual Basic .NET, eu voi construi mai întâi un fundament, informându-vă despre schimbările în dezvoltarea tehnicii de calcul şi a programelor, care fac cunoaşterea Microsoft .NET Framework o necesitate practică şi un interes vital pentru programatori. Voi trata apoi aspectele esenţiale ale programării orientate pe obiecte în Visual Basic .NET şi voi explica modul cum să vă construiţi propriile dumneavoastră clase şi să lucraţi cu clasele din cadrul .NET, modul cum să lucraţi cu tablouri şi colecţii şi modul cum să depanaţi şi să trataţi erorile din programele dumneavoastră. De la bază, vom urca apoi la nivelul următor. Voi trata detaliile modului de lucru cu ansamblurile .NET, modul cum să lucraţi cu fişiere şi fluxuri de date şi cum să monitorizaţi fişierele pe reţea, incluzând şi modul de a construi o aplicaţie tip serviciu Windows care să ruleze pe un server. Pe cuprinsul a trei capitole întregi voi descrie modul cum Visual Basic .NET şi ADO .NET au modificat tehnicile de programare pentru accesul la date. Trecem apoi la lumea serviciilor de Web – programe şi componente destinate să ruleze pe Internet. În ultimul capitol voi aduna laolaltă tot ce s-a discutat în capitolele precedente. Veţi vedea pe parcurs o mulţime de exemple de cod utile şi interesante.

Să învăţăm Visual Basic .NET scriind programe funcţionale

În majoritatea cărţilor de calculatoare pe care le-am citit, indiferent de limbajul de programare utilizat, autorul oferă extrase de cod academice pentru a ilustra un anumit aspect sau construct. Această abordare este utilă, însă cititorii se întreabă cum acest extras de cod se potriveşte în schema generală a unui program complet funcţional. Am descoperit că cel mai bun mod de a învăţa un nou limbaj de programare, cum ar fi Visual Basic .NET, este de a scrie în acel limbaj programe complet funcţionale. Faptul de a avea în minte un rezultat – şi de a scrie un program care să rezolve o problemă – pune la lucru multe dimensiuni ale unui limbaj de programare şi de asemenea permite înţelegerea modului în

care piesele disparate se îmbină şi lucrează împreună. Aceasta este abordarea pe care am ales-o în această carte, aceea de a vă „purta prin" mai multe aplicaţii exemplu care ilustrează aspecte importante din Visual Basic .NET.

Dacă ajungeţi la Visual Basic .NET după ce aţi folosit un alt limbaj de programare precum C, C++, Java sau chiar COBOL – nu va dura mult până să vă familiarizaţi cu VB. Microsoft .NET Framework este „valul viitorului" iar programatorii în Visual Basic sunt cei mai pregătiţi să profite de această nouă tehnologie. Cartea de faţă vă va face experţi în fundamentele tehnologiei .NET şi sunt sigur că veţi vedea rapid forţa şi uşurinţa a ceea ce se poate realiza cu .NET şi veţi începe să consideraţi programarea într-un mod nou, captivant. În ultimul rând, dar nu în cel din urmă, vă veţi simţi bine în timpul lecturii.

Ce găsiţi în această carte

În lista care urmează voi descrie aspectele principale din cadrul fiecărui capitol sumarizând astfel ceea ce veţi învăţa, pe măsură ce veţi progresa în lectura cărţii.

- **Capitolul 1 – Visual Basic .NET din temelii.** Voi începe prin examinarea şi explicarea cadrului .NET şi a motivului pentru care acesta constituie o abordare revoluţionară (şi nu evolutivă) a programării în secolul 21. Unul din beneficiile majore ale cadrului de programare .NET este capacitatea sa de a scrie un program o singură dată, destinat fiind oricărui tip de hardware sau sistem de operare. Această flexibilitate este crucială în momentul în care programatorii trebuie să creeze aplicaţii atât pentru computerele de sine stătătoare cât şi pentru Internet. Voi trece în revistă evoluţia limbajului de programare Visual Basic de la limbajul care a propulsat în prim plan programarea în mediul Windows şi până la Visual Basic .NET. Voi explica la nivel înalt unele din facilităţile cheie ale cadrului de programare .NET, cum ar fi cadrul claselor, rutina de execuţie a limbajului comun, serviciile de Web, ansamblurile şi noul mediu interactiv de dezvoltare (IDE) .NET Visual Studio – punctul central de comandă pe care îl veţi utiliza în lucrul cu aceste noi capacităţi. Veţi vedea pentru prima dată unele exemple de cod în Visual Basic .NET, pentru a vă da seama de cerinţele scrierii unui program în Visual Basic .NET. După parcurgerea capitolului 1, veţi avea o bună înţelegere a ceea ce urmează şi a ceea ce este important.
- **Capitolul 2 – Programarea orientată pe obiecte în Visual Basic .NET** Visual Basic .NET este acum un limbaj complet orientat pe obiecte, astfel că, în cele din urmă, el se alătură aşa numitor limbaje sofisticate, cum ar fi C++ sau Java. Dacă sunteţi începător în programarea orientată pe obiecte, veţi descoperi că acest capitol este un mod simplu de a „vă lua avânt". Pentru a ilustra modul cum obiectele sunt generate din clase, voi utiliza un form simplu din Visual Basic .NET şi voi ilustra proprietăţile şi metodele, moştenirea şi spaţiile de nume. Voi prezenta de asemenea noţiuni precum variabilele partajate, supraîncărcarea, polimorfismul şi încapsularea. Deoarece o mare parte din forţa limbajului .NET provine din clasele de bază oferite de .NET Framework, vă voi arăta modul cum să utilizaţi spaţiile de nume şi modul cum să accesaţi această funcţionalitate integrată.
- **Capitolul 3 – Scrierea primei dumneavoastră clase.** În continuarea capitolului 2, în capitolul 3 vă voi explica modul cum să scrieţi prima dumneavoastră clasă şi apoi cum să creaţi o a doua clasă, moştenită din clasa, de bază. Veţi învăţa de asemenea despre

directiva *imports*, modul cum să adăugaţi un ansamblu unui proiect, modul cum să lucraţi cu variabile membru partajate şi când şi de ce să utilizaţi directivele *Option Strict* şi *Option Explicit*. În construirea unei clase veţi utiliza constructori supraîncărcaţi care vă permit să iniţializaţi un obiect la instanţiere. La încheierea capitolului 3 programarea orientată pe obiecte va fi demistificată şi veţi fi plăcut surprinşi de cât de atractivă poate fi.

- **Capitolul 4 – Tipuri de date şi facilităţi în Visual Basic .NET** Deşi înţelegerea tipurilor de date referinţă şi valoare (primitive) din versiunile de Visual Basic anterioare v-a fost de ajutor, înţelegerea acestor concepte în Visual Basic .NET este crucială din cauza modului în care obiectele – şi în .NET totul este obiect – sunt comparate şi iniţializate. Tipurile de date în .NET sunt tipuri strict definite (fiecare variabilă trebuie să aibă un tip de date specific) şi sunt de asemenea sigure (puteţi accesa o variabilă doar prin intermediul tipului ei de date). Atunci când o variabilă nu mai este necesară, ea este marcată pentru a fi ştearsă prin intermediul unui algoritm de finalizare nedeterminist, cunoscut sub numele eufemistic de colector de gunoaie. Dacă întotdeauna vă setaţi variabilele referinţă la valoarea Nothing, veţi fi interesaţi de modul în care Visual Basic .NET administrează eliberarea resurselor şi a memoriei. La încheierea capitolului 4 veţi avea o bună înţelegere a modului cum sunt create variabilele, iniţializate şi apoi înlăturate în Visual Basic .NET.
- **Capitolul 5 – Examinarea cadrului claselor .NET care utilizează fişiere şi şiruri de caractere.** Motorul cadrului orientat pe obiecte .NET îl constituie biblioteca de clase ierarhice a acestuia. Începând cu spaţiile de nume, voi descrie acest cadru de programare de la noţiunile de bază, explicând cum acesta este organizat şi utilizând unele exemple concrete pentru a examina exact modul cum să lucraţi cu el. Pentru aceia dintre dumneavoastră care nu aţi mai folosit programarea orientată pe obiecte, acest capitol vă va arăta precis cum să găsiţi şi să utilizaţi în interiorul cadrului de programare ceea ce aveţi nevoie. Utilizând un instrument integrat, Windows Class Viewer, veţi ajunge să stăpâniţi subtilităţile cadrului de programare şi să învăţaţi tehnici pentru a vă focaliza rapid asupra a ceea ce aveţi nevoie. Pe parcurs voi descrie notaţia de clasă C# utilizată pentru a desemna parametri, a supraîncărca constructori şi metode şi pentru tipurile de returnări. În exemplul principal din cadrul capitolului vă voi arăta modul cum sunt accesate în acest cadru de programare clasele pentru fişiere şi fluxuri de date şi cum sunt ele utilizate pentru a citi şi a scrie pe disc. Voi examina de asemenea şirurile de caractere şi natura lor nouă, fixă. Vor fi ilustrate şi explicate tehnicile pentru copierea, clonarea şi formatarea şirurilor de caractere.
- **Capitolul 6 – Tablouri şi colecţii de date în Visual Basic .NET.** Aşa cum probabil vă aşteptaţi, tablourile sunt tratate diferit în Visual Basic .NET decât în versiunile anterioare de Visual Basic. Clasa *System.Array* din .NET Framework constituie clasa de bază pentru toate tipurile de tablouri. Deoarece tablourile sunt obiecte (ce nu este?), fiecare tablou pe care îl creaţi va cunoaşte cât de multe elemente conţine, cât de multe dimensiuni include, limitele sale şi aşa mai departe. Şi ceea ce este cel mai bine, prin utilizarea metodei *System.Array Sort*, tablourile din Visual Basic pot fi automat sortate şi inversate. Şi nu numai atât; utilizând diferite abordări, cum ar fi metoda binară de căutare integrată, în tablourile .NET se pot efectua operaţiuni de căutare. Pentru a ilustra tablourile voi crea un program de calculator care transformă numerele din notaţia

arabă în cea romană. Deşi un tablou este în realitate o colecţie simplă, o colecţie reprezintă un grup de obiecte. O colecţie poate fi moştenită din spaţiul de nume *System.Collection* din cadrul .NET. Spaţiul de nume *Collection* conţine interfeţe şi clase pentru crearea de obiecte noi, cum ar fi liste de tablouri, tabele de codare, cozi, stive şi dicţionare. Unele din aceste structuri de date pot fi noi pentru programatorii în Visual Basic. Voi încheia capitolul 6 prin scrierea unui program care imită un psiholog non determinst Rogerian. În acest scop voi utiliza unele din facilităţile avansate din .NET. Utilizatorii pot rula acest proiect distractiv pentru a examina psihologia interacţiunii om/maşină prin intermediul Visual Basic .NET.

- **Capitolul 7 – Tratarea erorilor şi programe de depanare**. Nu erorile sunt cele care strică programele ci erorile netratate. Erorile (de rulare, de sintaxă sau de logică) pot apărea oricând şi atunci când apar se generează o excepţie. Visual Basic .NET utilizează un construct structurat, de tipul *Try...Catch...Finally,* pentru a-l înlocui pe cel nestructurat, *Goto ErrorHandler*, utilizat în versiunile anterioare de Visual Basic. Tratarea structurată a erorilor este integrată în inima cadrului de programare .NET astfel că forţa sa ne este imediat disponibilă. În acest capitol voi utiliza programul de calcul din capitolul 6 şi vă voi arăta tot ceea ce poate merge prost în program şi modul cum să utilizaţi tratarea structurată a erorilor în abordarea fiecărei potenţiale erori. Voi examina de asemenea şi instrumentul de depanare. Mediul de dezvoltare integrat Visual Studio .NET oferă programatorilor mai multe ferestre de depanare ce prezintă o viziune clară a tuturor aspectelor implicate, începând de la valorile variabilelor până la asamblarea codului într-un program în execuţie. Vom scrie o clasă generică de tratare a erorilor, ErrorTrace.vb, care oferă informaţii de execuţie înregistrate şi care poate fi adăugată oricărui program Visual Basic .NET. Voi încheia acest capitol arătându-vă modul cum să scrieţi dintr-un program Visual Basic .NET în jurnalele de evenimente din Windows NT sau Windows 2000.
- **Capitolul 8 – Ansamblurile în detaliu**. Ansamblurile sunt blocurile constructive ale programelor Visual Basic .NET. Ele constituie unitatea fundamentală de dezvoltare, control al versiunii, reutilizare şi securitate. În acest capitol voi construi un program numit AssemblySpy care examinează legăturile interne ale oricărui ansamblu .NET scris într-un limbaj compatibil .NET. Acest program va utiliza noi controale grafice .NET pentru interfaţa de utilizator şi vă va oferi informaţii despre câmpurile statice şi instanţe, proprietăţi, evenimente, metode şi constructori. Voi descrie beneficiile ansamblurilor private şi a celor partajate, ca şi motivele pentru crearea ansamblurilor cu nume de nivel înalt pentru atribuirea de numere de versiune şi pentru partajare. Ansamblurile cu nume de nivel înalt permit execuţia colaterală, astfel încât două ansambluri cu acelaşi nume pot rula în acelaşi director. Programele Microsoft .NET compilate cu un ansamblu cu nume de nivel înalt ştiu ce ansamblu să utilizeze, eliminând astfel conflictele de DLL cu care s-a confruntat programarea în Windows.
- **Capitolul 9 – Monitorizarea sistemului de fişiere**. Integrată în .NET Framework, în spaţiul de nume *System.IO* se găseşte clasa *FileSystemWatcher*. Această clasă declanşează evenimente atunci când fişierele sau directoarele sunt modificate, create, şterse sau redenumite. Voi examina în detaliu această clasă prin adăugarea sa la o clasă pe care o vom construi, clasă denumită *SystemObserver*, care moşteneşte din *FileSystemWatcher*.

Voi explica de asemenea noţiunea de delegaţi, care permite programatorilor să definească şi să reacţioneze la propriile lor evenimente. Voi adăuga clasei *SystemObserver* delegaţi care să răspundă la evenimentele din *FileSystemWatcher*. Vom construi de asemenea un program Windows denumit File Sentinel şi vom importa clasa *SystemObserver*. Acest program oferă o interfaţă care permite utilizatorului să selecteze fişiere sau directoare. File Sentinel poate monitoriza orice fişier sau director (cum ar fi cookie-urile) şi vă poate anunţa atunci când se produce ceva important. În încheierea acestui capitol vă voi arăta cum să dezvoltaţi o aplicaţie tip serviciu Windows (cunoscută anterior sub numele de serviciu NT). Vom transforma clasa *SystemObserver* într-un serviciu Windows, pentru a ilustra reutilizarea codului nostru.

- **Capitolul 10 – Accesul la date cu ADO.NET**. Componentele ADO.NET au fost reproiectate pentru a oferi un model de obiect mai consistent, oferind de asemenea o mai mare scalabilitate pentru programarea pentru Internet. Acest lucru se realizează prin utilizarea noului obiect deconectat *DataSet*, care oferă un mod comun de reprezentare şi manipulare a datelor la un client. Spre deosebire de programarea tradiţională pentru accesul la date, care presupunea o conexiune către depozitul de date, ADO.NET este complet deconectat. Această abordare utilizează ceea ce este cunoscut sub numele de furnizori de date administraţi (managed providers), care includ un obiect de conexiune, un obiect de comandă, un obiect *DataReader* şi diferite obiecte *DataAdapter*. Aspectul din ADO.NET cel mai atrăgător este faptul că setul de date din memorie îşi reprezintă conţinutul în format text XML care poate fi trecut în ambele direcţii printr-un firewall de pe portul HTTP 80, făcând o realitate din partajarea datelor între sisteme diferite sau sisteme non-Windows.
- **Capitolul 11 – Seturi de date în detaliu**. În acest capitol voi intra şi mai mult în detaliile modelului de obiect ADO.NET. Voi începe prin scrierea unui program care ilustrează modul cum seturile de date îşi reprezintă conţinutul în limbaj XML, examinând atât schema cât şi reprezentarea XML a datelor. Vă voi arăta modul cum datele sunt manipulate local şi cum modificările sunt apoi scrise din nou în sursă. Vă voi demonstra de asemenea cum să utilizaţi un instrument din cadrul Visual Studio .NET, Xsd.exe, care poate lua orice fişier XML (de exemplu de la un nou distribuitor) şi poate genera din el o definiţie de schemă XML (XSD). Fişierul XSD poate fi apoi utilizat pentru a crea o clasă Visual Basic .NET care ştie cum să adauge, să şteargă şi să modifice fişierul XML anterior necunoscut. În continuare vom construi din program un obiect *DataTable* într-un set de date şi vom adăuga dinamic o legătură între câmpuri pentru a construi o legătură de subordonare între tabele. Datele din cadrul setului de date vor fi salvate pe disc în format XML şi apoi se va utiliza un control de tip grilă de date pentru a reconstitui datele din fişierul XML. Grila de date nu ştie şi nici nu-i pasă dacă informaţia provine dintr-o bază de date sau un fişier XML – în fiecare caz ea are toate informaţiile de care are nevoie pentru a se reconstrui pe sine. O examinare a noului obiect *DataView* ilustrează modul cum puteţi crea două vederi separate ale unui singur tabel de date.
- **Capitolul 12 – Legarea datelor în ADO.NET**. Acest capitol încheie examinarea ADO.NET prin trecerea în revistă a obiectului *BindingContext*. Deoarece setul de înregistrări deconectat din ADO.NET nu conţine o implementare explicită a cursorului, vă voi descrie cum să navigaţi printre înregistrări cu ajutorul obiectului *BindingContext* şi a

obiectelor înrudite *CurrencyManager*. Pentru a vă arăta cum se face acest lucru, vă voi prezenta un program care navighează printre datele înregistrate într-un singur tabel. Voi încheia capitolul prin construirea pas cu pas a unui tabel şi prin adăugarea de înregistrări în el. Vom căuta în tabel anumite înregistrări şi le vom manipula.

- **Capitolul 13 – ASP.NET şi serviciile de Web**. În Visual Basic .NET puteţi construi un form pentru Web cu aceeaşi uşurinţă cu care construiţi un form pentru Windows. Se utilizează acelaşi model de obiect. Voi examina modul cum ASP.NET oferă o experienţă simplificată a dezvoltării, oferind în acelaşi timp o scalabilitate îmbunătăţită. Voi trece în revistă noile controale grafice de server ca şi „noul cod" de la baza acestui concept, care vă dă posibilitatea de a separa codul logic de codul interfeţei de utilizator. Vom construi în ASP.NET un program funcţional de calcul al împrumuturilor, pentru a trece complet în revistă form-urile de Web, starea obiectului, controalele de server, retransmiterea datelor, validatorii de câmp, memorarea variabilelor în cache, legătura de date către un tabel de date construit dinamic şi altele. Trecând mai departe la serviciile de Web, veţi vedea modul cum SOAP (Simple Object Access Protocol) înlătură barierele specifice protocolului prin trimiterea de text simplu care declanşează metode API pe serverele situate la distanţă. Serviciile de Web pot apoi să-şi prezinte API-urile şi tipurile de date prin utilizarea limbajului de descriere a serviciilor de Web (Web Services Description Languages, WSDL). Voi ilustra aceste concepte prin intermediul unui program exemplu, care utilizează un serviciu denumit MagicEightBall, şi un program Windows care să consume serviciul de Web MagicEightBall. Crearea şi consumarea serviciilor de Web constituie părţile mele favorite din Visual Basic .NET.
- **Capitolul 14 – Moştenirea vizuală şi controalele definite de utilizator**. Aşa cum aţi descoperit anterior, totul în Visual Basic .NET este un obiect, incluzând şi form-urile de Windows. Deoarece putem moşteni dintr-un obiect existent, în acest capitol vom construi un form standard care oferă un aspect standard şi apoi vom moşteni această bază pentru a construi form-uri derivate identice. Vom construi de asemenea un control Visual Basic .NET particularizat. Voi încheia cartea cu un proiect distractiv, care implică şi pune la lucru aproape tot ceea ce aţi învăţat în această carte – un program care plasează notiţe electronice informative pe ecranul computerului dumneavoastră. Acest program salvează automat conţinutul, mărimea şi locaţia unei notiţe şi reconstituie orice notiţe existente la rularea ulterioară a programului. Voi descrie modul cum să utilizaţi programele Visual Basic .NET şi să creaţi un proiect de lucru care va dezvolta proiectul nostru cu afişarea de notiţe informative pe orice sistem Windows 2000, Windows NT sau Windows XP, deşi bibliotecile comune pentru limbajul .NET nu au fost încă instalate.