

Inside **C#**

Tom Archer

Microsoft®

IT Press

Inside C#

Kirjoittanut	Tom Archer
Kääntäjä	Jussi Arola
Kansi	Frank Chaumont
Kustantaja	Edita Oyj IT Press PL 760 00043 EDITA
	Sähköpostiosoite palvelu@itpress.fi Internet www.itpress.fi
Painopaikka	Edita Oyj, Helsinki 2001

Copyright © 2001 by Microsoft Corporation.

All rights published by arrangement with the original publisher, Microsoft Press, a division of Microsoft Corporation, Redmond, Washington, U.S.A.

Finnish language edition published by IT Press Copyright © 2001.

Kaikki oikeudet pidätetään. Tämän julkaisun tai sen osan jäljentäminen ilman tekijän kirjallista lupaa painamalla, monistamalla, äänittämällä tai muulla tavoin on tekijänoikeuslain mukaisesti kielletty.

Suomenkielisen version on julkaissut IT Press Copyright © 2001.

Alkuperäisen teoksen nimi on *Inside C#*

ISBN 951-826-455-4

Kirjan luvut

Osa I	Pohjan luominen	
1	Olioperusteisen ohjelmoinnin teoria	3
2	Johdanto Microsoft .NETiin	23
3	Hello, C#	35
Osa II	C#-luokkien perusteet	
4	Tyyppijärjestelmä	57
5	Luokat	69
6	Metodit	103
7	Ominaisuudet, taulukot ja indeksoijat	123
8	Attribuutit	143
9	Rajapinnat	161
Osa III	Koodin kirjoittaminen	
10	Lausekkeet ja operaattorit	189
11	Ohjausrakenteet	215
12	Virheenkäsittely poikkeusten avulla	245
13	Operaattorin ylikuormitus ja käyttäjän muunnokset	267
14	Delegaatit ja tapahtumakäsittelijät	281
Osa IV	Vaativampi C#	
15	Monisäikeinen ohjelmointi	303
16	Metadatan kyseleminen Reflection-metodeilla	327
17	Yhteistoiminta hallitsemattoman koodin kanssa	345
18	Koosteet	369

Sisällysluettelo

	Esipuhe	xiii
	Johdanto	xvii
Osa I	Pohjan luominen	
1	Olioperusteisen ohjelmoinnin teoria	3
	Kaikki ovat objekteja	5
	Objektit ja luokat	9
	Instantiointi	9
	Olioperusteisen ohjelmointikielen kolme perusominaisuutta	11
	Kapselointi	11
	Periytyminen	14
	Monimuotoisuus	17
2	Johdanto Microsoft .NETiin	23
	Microsoft .NET Platform	23
	.NET Framework	24
	Windows DNA ja .NET	24
	Common Language Runtime	25
	.NET Framework -luokkakirjastot	26
	Microsoft Intermediate Language ja JITterit	28
	Yleinen tyyppijärjestelmä	30
	Metadata ja reflection-menetelmä	30
	Turvallisuus	31
	Ohjelmien jakelu	32
	Yhteistoiminta hallitsemattoman koodin kanssa	32
3	Hello, C#	35
	Ensimmäisen C#-sovelluksen kirjoittaminen	35
	Editorin valinta	35
	Hello, World	38
	Komentorivikäntäjän käyttäminen	38
	Sovelluksen käynnistäminen	39
	Koodin tutkiminen	40

Sisällysluettelo

	Yhden pysäyksen ohjelmointi	40
	Luokat ja jäsenet	41
	<i>Main</i> -metodi	41
	<i>System.Console.WriteLine</i> -metodi	42
	Nimiavaruudet ja <i>using</i> -määre	42
	Koodin runko	43
	Jotain meni pieleen!	45
	Kääntäjän ilmoittamat virheet	45
	Tutkiminen ILDASM-ohjelmalla	46
	Hello, World MSIL-koodina	47
	C# ohjelmointiohjeet	50
	Milloin määrittelet oman nimiavaruuden	50
	Nimeämisohjeet	50
	Nimeämisstandardit	51
Osa II	C#-luokkien perusteet	
4	Tyypijärjestelmä	57
	Kaikki ovat objekteja	57
	Arvotyypit ja viittaustyyppit	58
	Arvotyypit	58
	Viittaustyyppit	59
	Paketointi ja purkaminen	59
	Kaikkien tyyppien äiti: <i>System.Object</i>	60
	Tyypit ja peitenimet	61
	Tyyppien väliset muunnokset	62
	Nimiavaruudet	64
	<i>using</i> -määre	65
	CTS:n edut	66
	Kielten yhteistoiminta	66
	Yksikantainen objektihierarkkia	67
	Tyypiturvallisuus	67

5	Luokat	69
	Luokkien määrittäminen	69
	Luokan jäsenet	70
	Käsittelymääreet	71
	<i>Main</i> -metodi	72
	Komentorivin parametrit	73
	Paluuarvot	74
	Useita <i>Main</i> -metodeja	74
	Muodostimet	75
	Staattiset jäsenet ja instanssijäsenet	77
	Muodostimen alustajat	79
	Vakiot ja vain-luku-tyyppiset kentät	82
	Vakiot	82
	Vain-luku-tyyppiset kentät	83
	Objektin tyhjennys ja resurssien hallinta	85
	Palanen historiaa	86
	Deterministinen lopetus	87
	Suorituskyky	88
	Täydellinen ratkaisu	94
	(Melkein) täydellinen ratkaisu	95
	IDispose-rajapinnan suunnitteluperiaatteet	96
	Periytyminen	97
	Monta rajapintaa	99
	Sinetöidyt luokat	100
6	Metodit	103
	<i>ref</i> ja <i>out</i> -tyyppiset parametrit	103
	Metodin ylikuormitus	108
	Muuttuva määrä parametreja	111
	Virtuaaliset metodit	112
	Metodin korvaaminen	112
	Monimuotoisuus	114
	Staattiset metodit	120
	Luokan jäsenten käsittely	122

7	Ominaisuudet, taulukot ja indeksoijat	123
	Ominaisuudet ovat älykkäitä kenttiä	123
	Ominaisuuksien määrittely ja käyttäminen	124
	Mitä kääntäjä itse asiassa tekee	126
	Vain-luku-ominaisuudet	128
	Periytyvät ominaisuudet	128
	Ominaisuuksien erikoiskäyttö	129
	Taulukot	130
	Taulukon määrittely	130
	Esimerkki yksiulotteisen taulukon käytöstä	131
	Moniulotteiset taulukot	132
	Ulottuvuuksien määrän selvittäminen	134
	Sisäkkäiset taulukot	135
	Objektien käsitteleminen taulukon tavoin indeksoijien avulla	136
	Indeksoijan määrittely	137
	Indeksoijaesimerkki	138
	Suunnitteluohjeita	140
8	Attribuutit	143
	Johdanto attribuutteihin	144
	Attribuutin määrittely	145
	Attribuuttien kyseleminen	146
	Luokan attribuutit	146
	Metodin attribuutit	149
	Kentän attribuutit	151
	Attribuutin parametrit	153
	Sijaintiparametrit ja nimetyt parametrit	153
	Nimettyihin parametreihin liittyviä virheitä	155
	Attribuutin kelvolliset parametrityypit	155
	<i>AttributeUsage</i> -attribuutti	156
	Attribuutin kohteen määrääminen	156
	Yksikäyttöiset ja monikäyttöiset attribuutit	158
	Attribuutin periytymisen määrittely	159
	Attribuutin tunniste	159

9	Rajapinnat	161
	Rajapinnan käyttäminen	162
	Rajapintojen määrittäminen	163
	Rajapintojen toteuttaminen	164
	Toteutuksen kyseleminen <i>/s</i> -operaattorilla	166
	Toteutuksen kyseleminen <i>as</i> -operaattorilla	170
	Explisiittinen rajapinnan jäsenen nimen määrittäminen	173
	Nimen piilottaminen rajapinnalla	173
	Nimiristiriitojen välttäminen	176
	Rajapinnat ja periytyminen	180
	Rajapintojen yhdistäminen	183
Osa III	Koodin kirjoittaminen	
10	Lausekkeet ja operaattorit	189
	Määritellyt operaattorit	189
	Operaattorien suoritusjärjestys	190
	Miten C# määrittelee suoritusjärjestyksen	191
	Vasen ja oikea liittyvyys	191
	Käytännön suoritusjärjestyksestä	192
	C#:n operaattorit	193
	Lausekkeen perusoperaattorit	193
	Matemaattiset operaattorit	198
	Suhteelliset operaattorit	206
	Yksinkertaiset sijoitusoperaattorit	209
11	Ohjausrakenteet	215
	Valintakäskyt	215
	<i>if</i> -käsky	215
	<i>switch</i> -käsky	220
	Toistokäskyt	225
	<i>while</i> -käsky	225
	<i>do/while</i> -käsky	227
	<i>for</i> -käsky	229
	<i>foreach</i> -käsky	232

Sisällysluettelo

Haarautuminen hyppykäskyillä	234
<i>break</i> -käsky	234
<i>continue</i> -käsky	237
Epäsuositettu <i>goto</i> -käsky	238
<i>return</i> -käsky	243
12 Virheen käsittely poikkeusten avulla	245
Johdanto poikkeusten käsittelyyn	245
Normaalin poikkeuksen käsittelyn syntaksi	247
Poikkeuksen aiheuttaminen	247
Poikkeuksen kiinniottaminen	247
Poikkeuksen jatkaminen	248
Siivoaminen <i>finally</i> -lohkossa	249
Virheen käsittelytekniikoiden vertailua	250
Poikkeusten käsittelyn edut paluuarvoon verrattuna	251
Virheiden käsitteleminen oikeassa ympäristössä	253
Koodin luettavuuden paraneminen	254
Poikkeuksen aiheuttaminen muodostimissa	256
<i>System.Exception</i> -luokan käyttäminen	256
<i>Exception</i> -objektin luominen	256
<i>StackTrace</i> -ominaisuuden käyttäminen	259
Useiden poikkeustyyppien kiinniottaminen	260
Omien <i>Exception</i> -luokkien periyttäminen	261
Poikkeusten käsittelyn suunnittelu	263
<i>try</i> -lohkon suunnitteluperiaatteet	263
<i>catch</i> -lohkon suunnitteluperiaatteet	265
13 Operaattorin ylikuormitus ja käyttäjän muunnokset	267
Operaattorin ylikuormitus	267
Syntaksi ja esimerkki	268
Ylikuormitettavat operaattorit	271
Rajoitukset operaattorin ylikuormituksessa	271
Suunnitteluohjeita	272
Käyttäjän muunnokset	272
Syntaksi ja esimerkki	273

14	Delegaatit ja tapahtumakäsittelijät	281
	Delegaattien käyttö takaisinkutsumetodeina	281
	Delegaattien määrittelevä staattisiksi jäseniksi	285
	Delegaattien luominen vain tarpeen vaatiessa	287
	Delegaattikooste	289
	Tapahtumien määrittely delegaateilla	295
Osa IV Vaativampi C#		
15	Monisäikeinen ohjelmointi	303
	Säikeistysten perusteet	303
	Säikeet ja moniajo	304
	Kontekstin vaihto	304
	Monisäikeinen sovellus	305
	Työskentely säikeillä	306
	<i>AppDomain</i>	306
	<i>Thread</i> -luokka	307
	Säikeiden ajoitus	310
	Säieturvallisuus ja synkronointi	314
	Koodin suojaaminen <i>Monitor</i> -luokan avulla	315
	Monitorilukkojen käyttäminen C#:n <i>lock</i> -käskyllä	319
	Koodin synkronisointi käyttämällä <i>Mutex</i> -luokkaa	321
	Säieturvallisuus ja .NET-luokat	323
	Säikeistysohjeita	323
	Milloin säikeitä tulee käyttää	323
	Milloin säikeitä ei tule käyttää	324
16	Metadatan kyseleminen Reflection-metodeilla	327
	Reflection-API:n rakenne	327
	<i>Type</i> -luokka	328
	Instanssin tyylin selvittäminen	328
	<i>Type</i> -objektin hakeminen nimen perusteella	329
	Tyyppien tulkitseminen	329
	Työskentely koosteilla ja moduleilla	332
	Koosteen tyyppien selvittäminen	332
	Koosteen modulien luettelo	335
	Myöhäinen sidonta Reflection-menetelmän avulla	337
	Koodin luominen ja ajaminen suorituksen aikana	340

Sisällysluettelo

17	Yhteistoiminta hallitsemattoman koodin kanssa	345
	Platform Invocation -palvelut	346
	Käytettävän DLL-funktion määrittelemine	346
	Takaisinkutsufunktioiden käyttäminen C#:ssa	349
	Muotoilu ja <i>PInvoke</i>	350
	Turvattoman koodin kirjoittaminen	351
	Osoittimien käyttäminen C#:ssa	352
	<i>fixed</i> -käsky	353
	Yhteistyö COMin kanssa	355
	Uljas uusi maailma	355
	Perusta	356
	Metadatan generointi COMin tyyppikirjastosta	357
	Aikainen sidonta COM-komponentteihin	360
	COM-rajapinnan valitseminen dynaamisesti	362
	Myöhäinen sidonta COM-komponentteihin	363
	COMin säikeistysmallit	365
18	Koosteet	369
	Johdanto koosteisiin	369
	Luettelon tiedot	370
	Koosteiden edut	371
	Koosteen pakkaaminen	371
	Koosteen jakelu	371
	Koosteen versiointi	372
	Koosteiden tekeminen	372
	Useita moduleja sisältävän koosteen tekeminen	373
	Jaetun koosteen tekeminen	375
	Yleisen koostevaraston käsittely	377
	Koostevaraston tarkastelu	377
	Koosteiden versiointi	379
	QFE ja oletusversiointikäytäntö	382
	Safe Mode -asetustiedoston tekeminen	382
	Hakemisto	385

Esipuhe

Olen ollut koko urani ajan Microsoftilla parantamassa ohjelmoiden mahdollisuuksia, yleensä keskittyen ohjelmoinnin tuottavuuden nostoon. Työni on käsittänyt suuren määrän eri tuotteita ja tekniikoita, mutta en koskaan ole ollut näin innostunut työstäni kuin nyt. Se mahdollisuuksien laajuus, jonka Microsoft .NET tarjoaa, on uskomaton. Tarjoamme uuden ohjelmointikielen rikkoen perinteisen ohjelmoiden jaon erillisiin mutta erilaisiin kielimaailmisiin ja antaen Web-sivustojen osaltaan auttavan vastaamaan käyttäjien tarpeisiin. Jokainen näistä olisi jo yksinään mielenkiintoinen, mutta niiden yhdistelmä on todella mukaansatempaava.

Katsotaan .NETin peruspalikoita ja niihin liittyviä tekniikoita:

- **C#, uusi kieli** C# on ensimmäinen komponenttisuuntautunut kieli C ja C++ -kieliperheessä. Se on yksinkertainen, nykyaikainen, olioperusteinen ja tyyppiturvallinen ohjelmointikieli, joka periytyy C:stä ja C++:sta. C# yhdistää Microsoft Visual Basicin tuottavuuden ja C++:n tehon.
- **Common Language Runtime** Tehokas Common Language Runtime (CLR) sisältää suoritusmoottorin, roskienkeruun, täsmäkääntäjän ja laajan luokkakirjaston (.NET Framework). CLR suunniteltiin alusta asti tukemaan useita ohjelmointikieliä.
- **Common Language Specification** Common Language Specification (CLS) kuvaa kielen toiminnallisuuden yleisen tason. CLS:n suhteellisen tiukka vaatimustaso mahdollistaa CLS-sopivien kielten joukon kehittämisen. Sen kukin jäsen nauttii kahdesta edusta: täydestä .NET Frameworkin toiminnallisuudesta ja yhteistyöstä muiden CLS-sopivien kielten kanssa. Esimerkiksi Visual Basic voi periä luokan C#:sta ja korvata sen virtuaalisen metodin.
- **Laaja joukko CLR:ään perustuvia kieliä** Microsoftin toteuttamia CLR-sopivia kieliä ovat Visual Basic, Visual C++ Managed Extensions -laajennuksin, Visual C# ja JScript. Kolmannet osapuolet ovat tuottamassa monia muita kieliä. Kieliä on niin monta, että en voi edes luetella niitä tässä!
- **Web services -palvelut** Nykyään World Wide Web koostuu pääosin yksittäisistä sivustoista. Vaikka käyttäjä voi vierailla useilla sivustoilla suorittaakseen määrätyn tehtävän, kuten esimerkiksi tehdä ryhmän

matkajärjestelyt, nämä sivustot eivät yleensä toimi yhteistyössä. Webin seuraava sukupolvi perustuu Web-sivustojen yhteistyöverkostoon. Syy on yksinkertainen: yhteistoiminnassa Web-sivustot täyttävät paremmin käyttäjien vaatimukset. Microsoftin Web services -palvelut edistävät Web-sivustojen yhteistyötä mahdollistamalla tietojen siirron XML-perusteisilla protokollilla, jotka ovat sekä kieliriippumattomia että alustariippumattomia. Monet tärkeät Web-palvelut tulevat perustumaan C#:lle ja Windows-käyttöjärjestelmässä pyörivälle CLR:lle mutta arkkitehtuuri on aidosti avoin.

- **Visual Studio.NET** Visual Studio.NET sitoo nämä kaikki palat yhteen ja tekee helpoksi julkaista erilaisia komponentteja, sovelluksia ja palveluja, jotka on tehty eri ohjelmointikielillä.

Nyt kun olen maininnut muutamista tärkeistä C#-kieleen liittyvistä tekniikoista, katsotaan tarkemmin itse C#-kieltä. Ohjelmoijat ovat investoineet paljon valitsemaansa kieleen ja siten on uuden kielen velvollisuus todistaa arvonsa yhdistämällä hyväksi koettuja ratkaisuja, lisäparannuksia ja syvällisiä keksintöjä.

Hyväksi koetut ratkaisut

Hippocrates sanoi, "Tee kahdesta asiasta tapa: auta tai älä ainakaan aiheuta haittaa." "Älä aihauta haittaa"-osa näytteli merkittävää osaa C#:n suunnittelussa. Jos C:n tai C++:n ominaisuus ratkaisi ongelman hyvin, emme muuttaneet sitä. Oleellisimmailta osiltaan C# lainaan C:stä ja C++:sta ydinalueilla, kuten lausekkeissa, käskyissä ja kokonaisrakenteessa. Koska normaalista ohjelmasta suuri osa koostuu näistä osista, C ja C++-ohjelmoijat tuntevat heti olonsa kotoisaksi C#:n kanssa.

Lisäparannuksia

Paljon erilaisia parannuksia on tehty, liian paljon, jotta niitä voisi luetella esipuheessa. Mutta seuraavassa mainitsen muutamia, jotka ovat aiheuttaneet yleisiä ja aikaa vieviä ongelmia C ja C++ -ohjelmissa:

- Muuttujat pitää alustaa ennen käyttöä, joten alustamattoman muuttujan käytöstä aiheutuneet virheet jäävät pois.
- *if* ja *while* -käskyt tarvitsevat Boolean-arvon, joten ohjelmoija, joka vahingossa käyttää sijoitusoperaattoria (=) vertailuoperaattorin (==) sijasta, löytää virheen jo käännöksessä.
- "Läpijuoksu" *switch*-käskyssä on estetty, joten ohjelmoija, joka epähuomiossa unohtaa *break*-käskyn, löytää virheen jo käännöksessä.

Syvällisiä keksintöjä

Syvällisempiä keksintöjä on löydettävissä C#:n tyyppijärjestelmästä. Se sisältää seuraavia etuja:

- C#:n tyyppijärjestelmä suorittaa automaattisen muistinhallinnan ja vapauttaa siten ohjelmoijan aikaavievältä ja virheherkältä muistin hallinnalta. Toisin kuin useimmat tyyppijärjestelmät, C#:n tyyppijärjestelmä mahdollista myös osoitintyyppien ja objektiosoitteiden suoran käsittelyn. (Nämä toimenpiteet sallitaan vain määrättyssä turvaympäristössä.)
- C#:n tyyppijärjestelmä on yhtenäinen, sillä kaikki ovat objekteja. Paketointi (boxing) ja purkaminen (unboxing) -tekniikoiden avulla C# yhdistää arvotyyppin ja viittaustyyppin erot ja salliin kaiken tiedon käsittelyn objektien tapaan.
- Ominaisuudet, metodit ja tapahtumat ovat C#:n perusjuttuja. Monet kielet jättävät pois luontaisen tuen ominaisuuksille ja tapahtumille ja aiheuttavat tarpeettoman epäyhteensopivuuden kielen ja siihen liittyvän kehyksen välille. Jos esimerkiksi kehys tukee ominaisuuksia ja kieli ei, ominaisuuden lisääminen on kömpelöä (esimerkiksi `o.SetValue(o.GetValue() + 1)`). Jos myös kieli tukee ominaisuuksia, toiminto on yksinkertainen (`o.Value++`).
- C# tukee attribuutteja, joiden avulla määritellään komponentteja ja liitetään selventävää informaatiota niihin. Mahdollisuus määritellä uuden tyyppistä informaatiota on aina ollut tehokas työkalu kielen suunnittelijoille. Nyt kaikilla C#-hjelmoijilla on tämä mahdollisuus.

Inside C#

Inside C#-kirjassa Tom Archer luo pohjan esittelemällä .NETin ja CLR:n, selvittää C#:n perusteet, ja sukeltaa sitten vaativampiin C#-kielen rakenteisiin. Hänen syvälinen kokemuksensa C++:sta, J++:sta ja Microsoft Windowsista, sekä ohjelmoijana että kirjoittajana, antaa mahdollisuuden selvittää C# tavalla, joka on sekä miellyttävä että informatiivinen.

Lukijat, toivon, että nautitte ensimmäisen C#-ohjelmanne kirjoittamisesta ja toivon myös, että se on ensimmäinen niistä monista, joita tulevana vuosina tulette kirjoittamaan.

Scott Wiltamuth
C#-suunnitteluryhmän jäsen
Microsoft Corporation

Johdanto

Miksi kirjoitin tämän kirjan

Olen tehnyt ohjelmia 20 vuotta (tunnen oloni vanhemmaksi aina kun ajattelen sitä!) ja olen tullut tilanteeseen, jossa ohjelmointi on alkanut tuntua vanhan toistolta. Älä käsitä minua väärin: jos olisin monimiljonääri eikä minun tarvitsisi tehdä työtä, luultavasti silti jatkaisin ohjelmien kirjoittamista, koska todella nautin siitä paljon. Olen kuitenkin ruvennut ajattelemaan, "Kaikki on tehty!" Sitten tulivat Microsoft .NET ja C# ja avautui kokonaan uusi mahdollisuuksien maailma. Olen puhnut muutamien kaverien kanssa, joille on myös tullut tämä jonkinasteinen uudelleenherääminen .NETin julkaisun myötä. Meillä on uusi, jännittävä tekniikka, joka lopulta ratkaisee asiat, joiden parissa olemme painineet vuosia (esimerkiksi monikieliset kehitysympäristöt, suurten järjestelmien jakelu- ja versiointiongelmien ja niin edelleen.) Kirjoitin tämän kirjan, koska koodin kirjoittaminen on taas jännittävää. Koska jälleen nousen aamuisin ylös ajatellen uutta ja mielenkiintoista asiaa, jonka tulen oppimaan. Toivon, että samalla kun opit tämän uuden kielen, jaat innostukseni.

Jokainen, joka kirjoittaa kirjaa C#:sta tämän kirjoittamisen aikaan, pitää opetella itse kieli samaan aikaan. Jos olet tehnyt sovellusta ja samaan aikaan opetellut käytettävää SDK:ta tai kieltä (kukapa ei olisi?) tietää, että se on hankala tilanne. Yritä sitten kuvitella, että kymmenet tuhannet ihmiset tulevat arvostelemaan sen, kun se on valmis! Suurin ongelma on se, että puoliväissä projektia, kun olet selvittänyt, mitä sinun tulee tehdä, palat halusta suunnitella ja tehdä koko asia uudelleen! Kirjan valmistumisaikataulun takia se on tietenkin mahdotonta toteuttaa. Uskon kuitenkin, että tämä kirja on hyvä apuväline C#:n opiskeluun. Koska opettelin itse samalla kuin kirjoitin tätä, kirjaan jäi ristiriitaisuuksia ja asioita, joita olisin voinut tehdä paljon paremmin. Mutta jos saan mahdollisuuden tehdä kirjasta toisen painoksen, voin luvata, että sinä ja minä tulemme hyötymään omasta oppimisestani, jonka olen tätä kirjaa tehdessäni hankkinut.

Lopksi haluan sanoa, että kaikki tätä kirjaa koskeva palaute on tervetullutta. En ole yksi noista "Olen niin hyvä, koska kirjoitan kirjoja" -kavereista. Olen aivan tavallinen kaveri, joka on ollut onnekas saadessaan mahdollisuuden tämän kirjan kirjoittamiseen. Olen aina valmis oppimaan muilta ja itse asiassa pidän siitä kovasti. Saat minuun yhteyden osoitteessa <http://www.TheCodeChannel.com>.

Kenen tulisi lukea tämä kirja

Tämä kirja on henkilöille, jotka suunnittelevat C# ja .NET-sovelluskehityksen aloittamista. Kuten mainitsin, tämä on yksi mielenkiintoinen alusta ja tulee olemaan tulevaisuudessa niin kauan kuin hajautettua Microsoft Windows -ohjelmistokehitystä tehdään. Tämä kirja olettaa, että sinulla on taustaa jostakin C-perheen kielestä: C:stä, C++:sta tai Javasta. Ainoa muu tarvittava vaatimus on mielestäni päätös oppia ja hankkia uusia ulottuvuuksia sovellusten kirjoittamiseen. Ja koska pidät tätä kirjaa kädessäsi, uskon, että sinä luultavasti olet sen päätöksen tehnyt!

Kirjan rakenne

Tämä kirja on järjestetty huolellisesti neljään loogisesti peräkkäiseen osaan, joista kukuin koostuu joukosta lukuja. Kukin luku käsittelee määrätyn alueen C# tai .NET-ohjelmoinnista.

Kirja alkaa osalla I, "Pohjan luominen," joka on tarkoitettu aloitteleville C#-ohjelmoijille ja niille, joille .NET ei ole ennestään tuttu. Osan luvut tarjoavat johdannon .NETiin ja näyttävät, miten luot ja testaat ensimmäisen C#-sovelluksesi.

Osassa II, "C#-luokkien perusteet," esittelen C#-luokkien määrittelyn ja käytön perusteet. Osan luvut antavat sinulle vankan tietopohjan siihen, miten C# tukee jäseniä (luetellut tyypit, ominaisuudet, taulukot a niin edelleen) ja miten määrittelet ja käytät niitä C#-sovelluksessa.

Vaikka olet kirjoittanut koodia useissa luvuissa määritellesäsi luokan jäseniä, osassa III, "Koodin kirjoittaminen," alat nähdä erilaisia tapoja eri tehtävien tekemiseen, kuten ohjauskäskyt, virheiden käsittely (poikkeusten avulla) ja tapahtumaksäittelijöiden kirjoittaminen delegaattien avulla.

Kirja päättyy osaan IV, "Vaativampi C#." Tyypillisenä nörttinä nautin eniten tämän osan kirjoittamisesta. Se sisältää luvut monisäikeisestä ohjelmoinnista, reflection-menetelmästä, työskentelystä hallitsemattoman koodin kanssa (mukaanlukien COM-yhteistoiminta) ja versioinnista.

Kirjan mukana tuleva cd

Tämän kirjan mukana tulee cd. Jos Windowsissasi on Autorun-toiminto päällä, saat aloitusruudun eteesi, kun asetat cd:n cd-rom-asemaan. Aloitusruudulta voit valita asennusvalinnat. Voit käynnistää aloitusruudun myös cd-juurihakemistosta käskyllä StartCD. StartCD-ohjelma sisältää linkin cd:llä olevaan eKirjaan, asennusohjelman kirjan esimerkkitiedostoihin ja linkin MSDN:ään, josta voit ladata Microsoft .NET Framework SDK:n viimeisimmän version, jota tarvitset kääntääksesi ja ajaaksesi esimerkkiohjelmat.

Kirjan esimerkkiohjelmat sijaitsevat Code-kansiossa. Voit selata esimerkkejä cd:ltä tai voit asentaa ne koneellesi käyttämällä StartCD:n asennusohjelmaa.

Huomaa Jos et pysty selaamaan Samples-kansion tiedostoja, sinulla saattaa olla vanhempi cd-ohjain, joka ei tue pitkiä tiedostonimiä. Tällöin sinun pitää asentaa esimerkkitiedostot kiintolevyillesi asennusohjelmalla, jotta voisit tutkia niitä.

Järjestelmävaatimukset

Jos haluat saada tästä kirjasta mahdollisimman paljon irti, suosittelen, että käyt esimerkkisovellukset läpi samalla kuin luet tekstiä. Siksi sinun pitää asentaa viimeisin .NET Framework SDK. Tämän kirjoitushetkellä se sisältää .NETin ajonaikaisen ympäristön ja C#-kääntäjän. Olen lisäksi tietoisesti välttänyt Visual Studio.NET-tuotteiden käyttöä ja keskittynyt itse kieleen ja ajonaikaiseen ympäristöön, jotta en rajoittaisi sinua määrättyyn kehitysympäristöön. Siksi kirjan kaikki esimerkit käännetään ja suoritetaan komentorivikäskyillä.

Kiitokset

Ensimmäiseksi haluan kiittää avusta toimittajaani, Devon Musgravea. En teeskentele olevani "kirjoittaja." Olen ohjelmoija, joha haluaa auttaa muita ja kirjojen kirjoittaminen on yksi tapa tehdä se. Ilman Devonin taivuttelua ja muotoilua se mitä sanoin eroaisi huomattavasti siitä, mitä halusin sanoa ja kirjan teksti ei olisi lähimainkaan niin luettavaa kuin se nyt on. Kiitokset, Devon!

Haluan kiittää myös Brian Johnsonia, kirjan teknistä toimittajaa. Brian oli korvaamaton, kun yritimme täsmätä kirjan tekstiä ja esimerkkiohjelmia. Hän oli myös suureksi avuksi selvitellessämme kääntäjään tehtyjä viime hetken muutoksia, kun kirja oli menossa painoon. Kirjoitin tämän kirjan käyttäen C#:n beta-versiota. Kun kääntäjä aikanaan julkaistaan, muutama kirjan esimerkeistä ei ehkä toimi. Yksikään niistä ei ole kuitenkaan Brianin vika, sillä hän oli äärimmäisen huolellinen testatessaan jokaisen esimerkkiohjelman.

Myös kaksi muuta Microsoft Pressin henkilökunnasta ansaitsee kiitokset: Anne Hamilton ja Danielle Bird. He molemmat olivat vastuussa tämän kirjan projektin käynnistymisestä ja he antoivat minulle mahdollisuuden tämän kirjan kirjoittamiseen. Heidän uskonsa minuun auttoi muutamien vaikeiden hetkien ohii ja arvostan paljon heidän tukeaa. Kiitokset!

Haluan myös kiittää seuraavia Microsoftin työntekijöitä, jotka suhtautuivat kärsivällisesti kysymyksiini, kun opiskelin C#-kieltä ja .NET BLC:tä: Joe Nalewabau (joka vastasi muutamiiin aikaisiin C#-kysymyksiini), Brian Harry (joka tarjosi tietoja deterministisestä lopetuksesta) ja Steven Pratschner (joka auttoi yleiseen koostevastoon ja koosteen versiontiin liittyvissä asioissa). Kiitokset myös Scott Wiltamuthille hänen erinomaisesta esipuheestaan.

Lopuksi haluan kiittää Aravind Coreraa, joka auttoi valtavasti luvussa, joka käsittelee hallitsematonta koodia. Hänen erinomaisen COM-kappaleensa ansiosta luku sisältää jotain erikoista. Hienoa työtä, Aravind. Toivon, että voimme tehdä yhteistyötä toisenkin kerran!