

---

---

# Stručný obsah

Úvod	29
<b>Část I: Jazyk C#</b>	
1. Architektura .NET	43
2. Základy jazyka C#	69
3. Objekty a typy	125
4. Dědění	157
5. Pole	181
6. Operátory a přetypování	201
7. Delegáty a události	239
8. Řetězce a regulární výrazy	269
9. Genericita	291
10. Kolekce	317
11. LINQ	367
12. Správa paměti a ukazatele	399
13. Reflexe	429
14. Chyby a výjimky	449

## Část II: Visual Studio

- |                        |     |
|------------------------|-----|
| 15. Visual Studio 2008 | 475 |
| 16. Nasazení aplikace  | 515 |

## Část III: Knihovny базových tříd

- |                                |     |
|--------------------------------|-----|
| 17. Sestavení                  | 543 |
| 18. Trasování a události       | 581 |
| 19. Podprocesy a synchronizace | 607 |
| 20. Zabezpečení                | 661 |
| 21. Lokalizace                 | 719 |
| 22. Transakce                  | 761 |
| 23. Služby systému Windows     | 799 |
| 24. Interoperabilita           | 835 |

## Část IV: Data

- |   |      |
|---|------|
| 25. Práce se soubory a systémovým registrem | 879  |
| 26. Přístup k datům                         | 935  |
| 27. LINQ pro SQL                            | 987  |
| 28. Manipulace s XML                        | 1013 |
| 29. LINQ pro XML                            | 1063 |
| 30. Programování pro .NET s SQL Serverem    | 1081 |
| Rejstřík                                    | 1113 |

---

---

# Obsah

<b>Úvod</b>	<b>29</b>
<b>Důležitost technologie .NET a jazyka C#</b>	<b>29</b>
Výhody technologie .NET	30
<b>Co je nového v platformě .NET Framework 3.5</b>	<b>31</b>
Implicitně typované proměnné	32
Automaticky implementované vlastnosti (properties)	32
Inicializátory kolekcí a objektů	33
Vestavěná podpora ASP.NET AJAX	33
.NET Language Integrated Query Framework (LINQ)	33
Podpora různých verzí .NET Framework	34
Podpora nejnovějších typů aplikací	34
<b>Porovnání jazyka C# s jinými jazyky</b>	<b>34</b>
<b>Požadavky na psaní a spouštění kódu v jazyce C#</b>	<b>36</b>
<b>Témata popsaná v této knize</b>	<b>36</b>
Část I: Jazyk C#	36
Část II: Visual Studio	36
Část III: Knihovny básových tříd (Base class libraries)	37
Část IV: Data	37
Část V: Prezentace	37
Část VI: Komunikace	37
Část VII: Přílohy	37
<b>Konvence</b>	<b>37</b>
<b>Zdrojový kód a přílohy</b>	<b>38</b>
<b>Errata</b>	<b>39</b>
<b>p2p.wrox.com</b>	<b>39</b>
<b>Poznámka redakce českého vydání</b>	<b>40</b>

## Část I: Jazyk C#

<b>1. Architektura .NET</b>	<b>43</b>
<b>Vztah jazyka C# k technologii .NET</b>	<b>44</b>
<b>Modul CLR</b>	<b>44</b>
Nezávislost na platformě	44
Zvýšení výkonu	45
Spolupráce mezi jazyky	45
<i>Visual Basic 2008</i>	46
<i>Visual C++ 2008</i>	46
<i>COM a COM+</i>	47
<b>Podrobná analýza jazyka IL</b>	<b>47</b>
Podpora objektové orientace a rozhraní	48
Odlišené hodnotové a referenční typy	49
Silná typová kontrola dat	49
<i>Význam silné typové kontroly dat pro spolupráci jazyků</i>	50
<i>Automatická správa paměti</i>	52
<i>Zabezpečení</i>	54
<i>Aplikační domény</i>	54
Ošetření chyb pomocí výjimek	56
Použití atributů	57
<b>Sestavení</b>	<b>57</b>
Soukromá sestavení	58
Sdílená sestavení	58
Reflexe	59
<b>Třídy platformy .NET Framework</b>	<b>59</b>
<b>Jmenné prostory</b>	<b>61</b>
<b>Tvorba aplikací pro .NET pomocí jazyka C#</b>	<b>61</b>
Tvorba aplikací ASP.NET	61
<i>Možnosti technologie ASP.NET</i>	62
<i>Webové formuláře</i>	63
<i>Webové služby založené na XML</i>	63
Tvorba formulářů pro Windows	64
Využití Windows Presentation Foundation (WPF)	64
Ovládací prvky pro Windows	64
Služby Windows	65
Windows Communication Foundation (WCF)	65
<b>Nástroje jazyka C# v podnikové architektuře .NET</b>	<b>65</b>
<b>Shrnutí</b>	<b>67</b>

<b>2. Základy jazyka C#</b>	<b>69</b>
<b>Úvodní informace</b>	<b>69</b>
<b>Váš první program v C#</b>	<b>70</b>
Kód	70
Kompilace a spuštění programu	71
Blíže pohled	71
<b>Proměnné</b>	<b>74</b>
Inicializace proměnných	74
Dovozený typ (Type Inference)	75
Obor proměnné (Variable Scope)	77
<i>Konflikty oborů lokálních proměnných</i>	77
<i>Konflikty oborů datových složek a lokálních proměnných</i>	78
Konstanty	79
<b>Předdefinované datové typy</b>	<b>80</b>
Hodnotové a referenční typy	80
Typy systému CTS	82
Předdefinované hodnotové typy	82
<i>Celočíselné typy</i>	82
<i>Typy s plovoucí řádovou čárkou</i>	84
<i>Typ decimal</i>	84
<i>Logický typ</i>	84
<i>Znakový typ</i>	85
Předdefinované referenční typy	86
<i>Typ object</i>	86
<i>Typ string</i>	86
<b>Řízení běhu programu</b>	<b>88</b>
Podmínkové příkazy	88
<i>Příkaz if</i>	88
<i>Příkaz switch</i>	90
Cykly	92
<i>Cyklus for</i>	93
<i>Cyklus while</i>	95
<i>Cyklus do...while</i>	95
<i>Cyklus foreach</i>	96
Příkazy skoku	97
<i>Příkaz goto (skok)</i>	97
<i>Příkaz break</i>	97
<i>Příkaz continue</i>	97
<i>Příkaz return</i>	97
<b>Výčty</b>	<b>97</b>
<b>Pole</b>	<b>99</b>
<b>Jmenné prostory</b>	<b>100</b>
Příkaz using	102
Alias jmenných prostorů	103

<b>Metoda Main()</b>	<b>104</b>
Více metod Main()	104
Předání argumentů funkci Main()	105
<b>Další informace o překladu souborů v C#</b>	<b>106</b>
<b>Konzolový vstup a výstup</b>	<b>107</b>
<b>Použití komentářů</b>	<b>109</b>
Interní komentáře ve zdrojových souborech	110
Dokumentace XML	110
<b>Direktivy preprocesoru C#</b>	<b>112</b>
#define a #undef	113
#if, #elif, #else a #endif	113
#warning a #error	115
#region a #endregion	115
#line	115
#pragma	116
<b>Doporučené zásady programování v C#</b>	<b>116</b>
Pravidla pro identifikátory	116
Konvence používání	117
<i>Konvence názvů</i>	118
<i>Velikost písmen v názvech</i>	119
<i>Styly názvů</i>	120
<i>Názvy jmenných prostorů</i>	120
<i>Názvy a klíčová slova</i>	120
Použití vlastností a metod	122
Použití datových složek	123
<b>Shrnutí</b>	<b>124</b>
<b>3. Objekty a typy</b>	<b>125</b>
<b>Třídy a struktury</b>	<b>126</b>
<b>Členy tříd</b>	<b>127</b>
<b>Datové členy</b>	<b>127</b>
<b>Funkční členy</b>	<b>127</b>
Metody	128
<i>Deklarace metod</i>	128
<i>Volání metod</i>	129
<i>Předávání parametrů metodám</i>	131
<i>Parametry ref</i>	133
<i>Parametry out</i>	133
<i>Přetěžování metod</i>	134
Vlastnosti	135
<i>Vlastnosti pouze pro čtení a pouze pro zápis</i>	136
<i>Přístupové modifikátory vlastností</i>	136
<i>Automaticky implementované vlastnosti</i>	137

---

Konstruktory	138
<i>Statické konstruktory</i>	139
<i>Volání konstruktorů z jiných konstruktorů</i>	141
Datové složky pouze pro čtení	143
<b>Anonymní typy</b>	<b>144</b>
<b>Struktury</b>	<b>145</b>
Struktury jsou hodnotové typy	147
Struktury a dědění	148
Konstruktory struktur	148
<b>Částečné třídy</b>	<b>148</b>
<b>Statické třídy</b>	<b>150</b>
<b>Třída Object</b>	<b>150</b>
Metody třídy System.Object	151
Metoda ToString()	152
<b>Rozšiřující metody (Extension methods)</b>	<b>154</b>
<b>Shrnutí</b>	<b>155</b>
<b>4. Dědění</b>	<b>157</b>

---

<b>Typy dědění</b>	<b>157</b>
Dědění implementace versus dědění rozhraní	157
Vicenásobné dědění	158
Struktury a třídy	158
<b>Dědění implementace</b>	<b>159</b>
Virtuální metody	160
Zastiňování metod	161
Volání funkcí ze základní třídy	162
Abstraktní třídy a funkce	163
Zapečetěné třídy a metody	163
Konstruktory odvozených tříd	165
<i>Přidáváme do hierarchie konstruktor bez parametrů</i>	166
<i>Přidáváme do hierarchie konstruktor s parametry</i>	168
<b>Modifikátory</b>	<b>170</b>
Modifikátory viditelnosti	170
Jiné modifikátory	171
<b>Rozhraní</b>	<b>172</b>
Definice a implementace rozhraní	173
Odvozená rozhraní	177
<b>Shrnutí</b>	<b>179</b>

---

<b>5. Pole</b>	<b>181</b>
<b>Jednoduchá pole</b>	<b>181</b>
Deklarace pole	181
Inicializace pole	182
Přístup k prvkům pole	183
Použití referenčních typů	183
<b>Vícerozměrná pole</b>	<b>185</b>
<b>Nepravidelná pole</b>	<b>186</b>
<b>Třída Array</b>	<b>187</b>
Vlastnosti	187
Vytváření polí	187
Kopírování polí	188
Třídění	189
<b>Rozhraní pro pole a kolekce</b>	<b>192</b>
IEnumerable	192
ICollection	192
IList	192
<b>Procházení pole (enumerátory)</b>	<b>193</b>
Rozhraní IEnumerator	193
Cyklus foreach	194
Příkaz yield	194
<b>Shrnutí</b>	<b>200</b>
<b>6. Operátory a přetypování</b>	<b>201</b>
<b>Operátory</b>	<b>201</b>
Složené operátory	203
Podmínkový operátor	204
Operátory checked a unchecked	205
Operátor is	206
Operátor as	206
Operátor sizeof	206
Operátor typeof	206
Nulovatelné typy a operátory	207
Operátor nulového sjednocení	207
Priorita operátorů	208
<b>Typová bezpečnost</b>	<b>208</b>
Převody typů	209
<i>Implicitní převody</i>	209
<i>Explicitní převody</i>	210
Automatické zabalování a vybalování	213
<b>Zjišťování rovnosti objektů</b>	<b>214</b>
Zjišťování rovnosti referenčních typů	214
<i>Metoda ReferenceEquals()</i>	214



<i>Virtuální metoda Equals()</i>	214
<i>Statická metoda Equals()</i>	215
<i>Operátor rovnosti (==)</i>	215
Rovnost hodnotových typů	215
<b>Přetěžování operátorů</b>	<b>216</b>
Fungování operátorů	217
Příklad přetěžování operátorů: Struktura Vector	218
<i>Přidání dalších přetížených operátorů</i>	221
<i>Přetěžování relačních operátorů</i>	223
Které operátory lze přetížit?	225
<b>Uživatelsky definovaná přetypování</b>	<b>226</b>
Implementace uživatelsky definovaných přetypování	227
<i>Přetypování mezi třídami</i>	231
<i>Přetypování mezi základními a odvozenými třídami</i>	232
<i>Automatické zabalení a vybalení přetypování</i>	233
Vicenasobné přetypování	234
<b>Shrnutí</b>	<b>238</b>
<b>7. Delegáty a události</b>	<b>239</b>
<b>Delegáty</b>	<b>239</b>
Deklarování delegátů v C#	240
Použití delegátů v C#	242
Příklad SimpleDelegate	245
Příklad BubbleSorter	247
Vicenasobné delegáty	250
Anonymní metody	254
Lambda výrazy	255
Kovariance a kontravariance	257
<i>Kovariance návratového typu</i>	257
<i>Kontravariance typu parametru</i>	258
<b>Události</b>	<b>259</b>
Události z pohledu příjemce	260
Vyvolávání událostí	262
<b>Shrnutí</b>	<b>267</b>
<b>8. Řetězce a regulární výrazy</b>	<b>269</b>
<b>System.String</b>	<b>270</b>
Vytváření řetězců	271
Členy třídy StringBuilder	274
Formátovací řetězce	275
<i>Princip formátování řetězce</i>	277
<i>Příklad: FormattableVector</i>	279
<b>Regulární výrazy</b>	<b>282</b>
Úvod do regulárních výrazů	282

Příklad RegularExpressionsPlayaround	283
Zobrazení výsledků	286
Shody, skupiny a záchyty	288
<b>Shrnutí</b>	<b>290</b>
<b>9. Genericita</b>	<b>291</b>
<b>Přehled</b>	<b>292</b>
Výkon	292
Typová bezpečnost	293
Opakované použití binárního kódu	294
Zvětšování kódu	294
Doporučení při pojmenování	294
<b>Vytváření generických tříd</b>	<b>295</b>
<b>Vlastnosti generických tříd</b>	<b>300</b>
Výchozí hodnoty	301
Omezení	301
Dědění	304
Statické členy	305
<b>Generická rozhraní</b>	<b>305</b>
<b>Generické metody</b>	<b>306</b>
<b>Generické delegáty</b>	<b>309</b>
Implementace metod volaných delegáty	309
Použití generických delegátů s třídou Array	311
<b>Další generické typy v .NET</b>	<b>313</b>
Nullable<T>	313
EventHandler<TEventArgs>	315
ArraySegment<T>	315
<b>Shrnutí</b>	<b>316</b>
<b>10. Kolekce</b>	<b>317</b>
<b>Rozhraní a typy kolekci</b>	<b>317</b>
<b>Seznamy</b>	<b>320</b>
Vytváření seznamů	322
<i>Inicializátory kolekce</i>	323
<i>Přidávání prvků</i>	323
<i>Vkládání prvků</i>	324
<i>Přístup k prvkům</i>	325
<i>Odstraňování prvků</i>	326
<i>Vyhledávání</i>	327
<i>Třídění</i>	329
<i>Přetypování</i>	331
Kolekce pouze pro čtení	332
<b>Fronty</b>	<b>332</b>

---

<b>Zásobníky</b>	<b>337</b>
<b>Spojové seznamy</b>	<b>338</b>
<b>Tříděné seznamy</b>	<b>346</b>
<b>Slovníky</b>	<b>348</b>
Typ klíče	349
Ukázka slovníku	350
Vyhledávání	354
Další třídy slovníků	355
<b>HashSet</b>	<b>356</b>
<b>Bitová pole</b>	<b>359</b>
Třída BitArray	359
BitVector32	362
<b>Výkon</b>	<b>364</b>
<b>Shrnutí</b>	<b>366</b>
<b>11. LINQ</b>	<b>367</b>

---

<b>Úvod do LINQ</b>	<b>367</b>
Dotaz s použitím třídy List<T>	368
Rozšiřující metody	374
Lambda výrazy	376
Dotazy s pomocí LINQ	376
Odložené provedení dotazu	377
<b>Standardní operátory dotazu</b>	<b>379</b>
Filtrování	381
Filtrování pomocí počítadla	381
Filtrování podle typu	382
Složené from	382
Řazení	383
Seskupování	384
Seskupování s vnořenými objekty	386
Spojování	387
Množinové operace	388
Rozdělování	389
Agregační operátory	391
Převod	392
Generující operátory	393
<b>Výrazové stromy</b>	<b>394</b>
<b>Poskytovatele LINQ</b>	<b>397</b>
<b>Shrnutí</b>	<b>398</b>
<b>12. Správa paměti a ukazatele</b>	<b>399</b>

---

<b>Technické principy správy paměti</b>	<b>399</b>
Hodnotové datové typy	400

Referenční datové typy	401
Úklid	403
<b>Uvolňování neřízených prostředků</b>	<b>404</b>
Destruktory	405
Rozhraní IDisposable	406
Implementace rozhraní IDisposable a destruktoru	407
<b>Nebezpečný kód</b>	<b>409</b>
Přímý přístup do paměti pomocí ukazatelů	410
<i>Psaní nebezpečného kódu s klíčovým slovem unsafe</i>	411
<i>Syntaxe ukazatelů</i>	412
<i>Přetypování ukazatelů na celočíselné typy</i>	414
<i>Přetypování mezi typy ukazatelů</i>	415
<i>Ukazatele typu void</i>	415
<i>Aritmetika ukazatelů</i>	415
<i>Operátor sizeof</i>	417
<i>Ukazatele na struktury: operátor nepřímého přístupu ke složkám</i>	417
<i>Ukazatele na složky třídy</i>	418
Příklad ukazatele: PointerPlayaround	420
Optimalizace výkonu pomocí ukazatelů	424
<i>Vytvoření polí fungujících v zásobníku</i>	425
<i>Příklad QuickArray</i>	427
<b>Shrnutí</b>	<b>428</b>
<b>13. Reflexe</b>	<b>429</b>
<b>Vlastní atributy</b>	<b>430</b>
Psaní vlastních atributů	430
<i>Atribut AttributeUsage</i>	431
<i>Parametry atributu</i>	433
<i>Volitelné parametry atributu</i>	433
Příklad vlastního atributu: WhatsNewAttributes	434
<i>Sestavení knihovny WhatsNewAttributes</i>	434
<i>Sestavení VectorClass</i>	436
<b>Reflexe</b>	<b>437</b>
Třída System.Type	437
<i>Vlastnosti třídy Type</i>	438
<i>Metody</i>	439
Příklad TypeView	440
Třída Assembly	442
<i>Zjištění informací o typech definovaných v sestavení</i>	443
Zjištění informací o vlastních atributech	443
Dokončení příkladu WhatsNewAttributes	444
<b>Shrnutí</b>	<b>448</b>

<b>14. Chyby a výjimky</b>	<b>449</b>
<b>Třídy výjimek</b>	<b>450</b>
<b>Zachycení výjimek</b>	<b>452</b>
Několik bloků catch	454
Zachycení výjimek z jiného kódu	458
Vlastnosti třídy System.Exception	459
Co se stane, když výjimka nebude obsloužena	459
Vnořené bloky try	460
<i>Změna typu výjimky</i>	461
<i>Zpracování různých výjimek na různých místech</i>	462
<b>Uživatelsky definované třídy výjimek</b>	<b>462</b>
Zachycení uživatelsky definovaných výjimek	463
Vyvolání uživatelsky definovaných výjimek	465
Definice uživatelských tříd výjimek	468
<b>Shrnutí</b>	<b>471</b>

## Část II: Visual Studio

<b>15. Visual Studio 2008</b>	<b>475</b>
<b>Práce s Visual Studiem 2008</b>	<b>475</b>
Vytvoření projektu	480
<i>Volba typu projektu</i>	481
<i>Vytvoření nového projektu konzolové aplikace</i>	484
<i>Ostatní vytvořené soubory</i>	485
Řešení a projekty	486
<i>Přidání dalšího projektu do řešení</i>	487
<i>Nastavení startovacího projektu</i>	489
Kód okenních aplikací	489
Načítání projektů z Visual Studia 6	489
Prozkoumávání a kódování projektu	490
<i>Skrývání částí kódu</i>	490
<i>Další okna</i>	492
<i>Tlačítka s ikonou špendlíku</i>	499
Sestavování projektu	500
<i>Překlad a co s ním souvisí</i>	500
<i>Ladicí a finální překlad</i>	500
<i>Volba konfigurace</i>	502
<i>Úprava konfigurace</i>	503
Ladění	504
<i>Body zastavení programu (zarážky)</i>	505
<i>Sledování proměnných</i>	506
<i>Výjimky</i>	507
<b>Refaktorování kódu</b>	<b>508</b>

<b>Práce s různými verzemi platformy .NET</b>	<b>510</b>
<b>WPF, WCF, WF a další</b>	<b>512</b>
Vývoj aplikací WPF ve Visual Studiu	512
Vývoj WF aplikací ve Visual Studiu	512
<b>Shrnutí</b>	<b>513</b>
<b>16. Nasazení aplikace</b>	<b>515</b>
<b>Návrh s ohledem na nasazování</b>	<b>515</b>
<b>Možnosti nasazování</b>	<b>516</b>
Xcopy	516
Nástroj Copy Web	516
Publikování webů	516
Projekty nasazování	516
Technologie ClickOnce	516
<b>Požadavky na nasazení</b>	<b>517</b>
<b>Nasazení prostředí .NET</b>	<b>518</b>
<b>Jednoduché nasazování</b>	<b>518</b>
Xcopy	519
Nasazování xcopy a webové aplikace	519
Nástroj Copy Web	519
Publikování webového portálu	520
<b>Projekty služby Installer</b>	<b>520</b>
Co to je služba Windows Installer	521
Vytváření instalačních balíčků	521
<i>Jednoduchá klientská aplikace</i>	522
<i>Projekt ve stejném řešení</i>	527
<i>Jednoduchá webová aplikace</i>	528
<i>Klient z webového serveru</i>	530
<i>Nasazování no-touch</i>	530
<b>Technologie ClickOnce</b>	<b>531</b>
Princip technologie ClickOnce	532
Publikování aplikace	532
Nastavení ClickOnce	533
Aplikační úložiště	533
Zabezpečení	534
Pokročilé možnosti	534
<i>Editor File System</i>	535
<i>Editor registru</i>	535
<i>Editor File Types</i>	535
<i>Editor User Interface</i>	536
<i>Editor Custom Actions</i>	538
<i>Editor Launch Conditions</i>	539
<b>Shrnutí</b>	<b>540</b>

## Část III: Knihovny bazových tříd

<b>17. Sestavení</b>	<b>543</b>
<b>Co jsou sestavení</b>	<b>543</b>
Vlastnosti sestavení	544
Struktura sestavení	544
Manifest sestavení	545
Jmenné prostory, sestavení a komponenty	546
Privátní a sdílená sestavení	546
Satelitní sestavení	546
Prohlížení sestavení	547
<b>Tvorba sestavení</b>	<b>547</b>
Tvorba modulů a sestavení	547
Atributy sestavení	549
<b>Dynamické zavádění a tvorba sestavení</b>	<b>551</b>
<b>Aplikační domény</b>	<b>554</b>
<b>Sdílená sestavení</b>	<b>558</b>
Silné názvy	558
Používání silných názvů a integrity	559
Globální mezipaměť sestavení	560
Tvorba sdíleného sestavení	561
Vytvoření silného názvu	562
Instalace sdíleného sestavení	564
Použití sdíleného sestavení	564
Odložené podepisování sestavení	565
Odkazy	566
Generátor nativních obrazů	567
<b>Konfigurace aplikací v .NET</b>	<b>568</b>
Kategorie konfigurací	568
Volba adresářů, ve kterých hledat sestavení	569
<codeBase>	570
<probing>	570
<b>Správa verzí</b>	<b>571</b>
Číslování verzí	572
Zjištění čísla verze v programu	572
Konfigurační soubory aplikace	573
Soubory se zásadami vydavatele	576
Vytvoření souboru se zásadami vydavatele	576
Vytvoření sestavení se zásadami vydavatele	577
Vložení sestavení se zásadami vydavatele do GAC	577
Obcházení zásad vydavatele	577
Verze běhového prostředí	578
<b>Shrnutí</b>	<b>579</b>

---

<b>18. Trasování a události</b>	<b>581</b>
<b>Trasování</b>	<b>581</b>
Trasovací zdroje	583
Trasovací přepínače	584
Trasovací přijímače	585
Filtry	587
Aserce	589
<b>Protokolování událostí (Event logging)</b>	<b>590</b>
Architektura protokolování událostí	590
Třídy pro protokolování událostí	592
Vytvoření zdroje událostí	593
Zápis do protokolu událostí	594
Soubory zdrojů	594
Přijímač protokolu událostí	599
<b>Monitorování výkonu</b>	<b>600</b>
Třídy monitorující výkon	601
Performance Counter Builder	601
Přidávání komponent pro sledování výkonu	602
Perfmon.exe	604
<b>Shrnutí</b>	<b>605</b>
<b>19. Podprocesy a synchronizace</b>	<b>607</b>
<b>Přehled</b>	<b>608</b>
<b>Asynchronní delegáty</b>	<b>609</b>
Cyklické dotazování (Polling)	610
Třída WaitHandle	610
Asynchronní zpětné volání (Callback)	611
<b>Třída Thread</b>	<b>613</b>
Předávání dat podprocesům	615
Podprocesy na pozadí	616
Priorita podprocesů	617
Řízení podprocesů	618
<b>Fondy podprocesů (Thread pools)</b>	<b>618</b>
<b>Problémy s podprocesy</b>	<b>620</b>
Konflikt časování	620
Uváznutí	623
<b>Synchronizace</b>	<b>625</b>
Příkaz lock a zabezpečení podprocesů	625
Interlocked	631
Monitor	633
Popisovač čekání (Wait Handle)	634
Mutex	635
Třída Semaphore	636



Události	639
ReaderWriterLockSlim	641
<b>Časovače (Timers)</b>	<b>645</b>
<b>COM apartmenty</b>	<b>647</b>
<b>Asynchronní vzory s využitím událostí</b>	<b>647</b>
BackgroundWorker	648
<i>Zrušení výpočtu</i>	651
<i>Zobrazení postupu výpočtu</i>	652
Tvorba asynchronní komponenty s využitím událostí	653
<b>Shrnutí</b>	<b>659</b>
<b>20. Zabezpečení</b>	<b>661</b>
<b>Autentizace a autorizace</b>	<b>661</b>
Identita a objekt zabezpečení (principal)	662
Role	663
Deklarativní zabezpečení založené na rolích	664
Klientské aplikační služby	665
<i>Aplikační služby</i>	665
<i>Klientská aplikace</i>	669
<b>Šifrování</b>	<b>670</b>
Podpis	673
Výměna klíčů a zabezpečený přenos	675
<b>Řízení přístupu k systémovým zdrojům</b>	<b>678</b>
<b>Přístupová práva aplikace</b>	<b>681</b>
Oprávnění	682
<i>Požadujeme oprávnění v programu</i>	684
<i>Deklarativní oprávnění</i>	685
<i>Žádost o oprávnění</i>	686
<i>Implicitní oprávnění</i>	689
<i>Odebírání oprávnění</i>	690
<i>Poskytování Prosazování oprávnění (Asserting permissions)</i>	691
Skupiny kódu	693
<i>caspol.exe – nástroj pro správu bezpečnostní politiky přístupu ke kódu</i>	694
<i>Zobrazení skupiny kódu sestavení</i>	696
Přístupová práva aplikace a sady oprávnění	698
Úrovně zásad: Machine (počítač), User (uživatel) a Enterprise (podnik)	701
<b>Správa bezpečnostních zásad</b>	<b>703</b>
Správa skupin kódu a oprávnění	706
Zapínání a vypínání zabezpečení	706
Tvorba skupiny kódu	707
Odstranění skupiny kódu	707
Změna oprávnění skupiny kódu	708
Tvorba a přidělení sady oprávnění	709
Distribuce kódu s využitím silného jména	711

Distribuce kódu s použitím certifikátů	712
<b>Shrnutí</b>	<b>718</b>
<b>21. Lokalizace</b>	<b>719</b>
<b>Jmenný prostor System.Globalization</b>	<b>719</b>
Problematika kódování Unicode	720
Kultury a regiony	721
<i>Specifické, neutrální a kulturně nezávislé jazykové verze</i>	721
<i>CurrentCulture a CurrentUICulture</i>	722
<i>Formátování čísel</i>	723
<i>Formátování kalendářního data</i>	724
Jazykové verze v praxi	725
Abecední řazení	730
<b>Prostředky</b>	<b>732</b>
Vytváření souborů prostředků	732
Nástroj Resource File Generator	732
Třída ResourceWriter	733
Použití souborů prostředků	734
Jmenný prostor System.Resources	739
<b>Lokalizace Windows Forms ve Visual Studiu</b>	<b>740</b>
Programová změna jazykové verze	744
Použití vlastních zpráv jako prostředků	746
Automatické použití standardních prostředků	747
Zadání překladů třetí firmě	747
<b>Lokalizace aplikací v ASP.NET</b>	<b>748</b>
<b>Lokalizace ve WPF</b>	<b>750</b>
Aplikace založená na WPF	751
Prostředky .NET	751
Lokalizace v XAML	752
<b>Vlastní nástroj pro čtení prostředků</b>	<b>754</b>
Vytvoření třídy DatabaseResourceReader	755
Vytvoření třídy DatabaseResourceSet	757
Vytvoření třídy DatabaseResourceManager	757
Klientská aplikace pro třídu DatabaseResourceReader	758
<b>Vytváření vlastních jazykových verzí</b>	<b>758</b>
<b>Shrnutí</b>	<b>760</b>
<b>22. Transakce</b>	<b>761</b>
<b>Přehled</b>	<b>761</b>
Fáze transakce	762
Vlastnosti ACID	763
<b>Třídy databází a entit</b>	<b>763</b>
<b>Tradiční transakce</b>	<b>765</b>
Transakce v ADO.NET	765

System.EnterpriseServices	767
<b>System.Transactions</b>	<b>768</b>
Transakce umožňující commit	770
Povyšování transakcí	773
Závislé transakce	775
Ambientní transakce	777
<i>Vnořování oborů a ambientní transakce</i>	778
<i>Podprocesy a ambientní transakce</i>	780
<b>Úroveň izolace</b>	<b>784</b>
<b>Uživatелеm definovaní správci zdrojů</b>	<b>786</b>
Zdroje v transakcích	787
<b>Transakce ve Windows Vista a Windows Server 2008</b>	<b>793</b>
<b>Shrnutí</b>	<b>797</b>
<b>23. Služby systému Windows</b>	<b>799</b>
<b>Co je to služba systému Windows?</b>	<b>799</b>
<b>Architektura služeb systému Windows</b>	<b>800</b>
Program služby	801
<i>Správce řízení služeb</i>	801
<i>Hlavní funkce, hlavní funkce služby a obslužné funkce</i>	801
Program pro řízení služby	802
Program pro konfiguraci služby	802
<b>Jmenný prostor System.ServiceProcess</b>	<b>803</b>
<b>Vytváření služby systému Windows</b>	<b>803</b>
Knihovna tříd používající sokety	803
Příklad použití třídy TcpClient	807
Projekt služby systému Windows	809
<i>Třída ServiceBase</i>	811
<i>Hlavní funkce</i>	812
<i>Spuštění služby</i>	813
<i>Obslužné metody</i>	813
Podprocesy a služby	814
Instalace služby	815
Instalační program	815
<i>Třída Installer</i>	815
<i>Třídy ServiceProcessInstaller a ServiceInstaller</i>	816
<i>Třída ServiceInstallerDialog</i>	819
<i>Nástroj installutil</i>	819
<i>Klient</i>	820
<b>Sledování a řízení služby</b>	<b>820</b>
Správa počítače prostřednictvím konzoly MMC	821
Pomocný program net.exe	821
Pomocný program sc.exe	821
Průzkumník serverů Visual Studio	822

Třída ServiceController	822
<i>Sledování služby</i>	824
<i>Řízení služby</i>	830
<b>Odstraňování problémů</b>	<b>831</b>
Interaktivní služby	832
Protokolování události	832
<b>Události napájení</b>	<b>833</b>
<b>Shrnutí</b>	<b>834</b>
<b>24. Interoperabilita</b>	<b>835</b>
<b>.NET a COM</b>	<b>836</b>
Metadata	836
Uvolňování paměti	837
Rozhraní	837
<i>Uživatelská rozhraní</i>	837
<i>Odesílací rozhraní</i>	837
<i>Duální rozhraní</i>	838
<i>Přetypování a QueryInterface</i>	838
Vazby metod	839
Datové typy	839
Registrace	839
Podprocesy	840
<i>Jednovláknové oddělení</i>	840
<i>Vícevláknové oddělení</i>	840
Zpracování chyb	841
Zpracování událostí	841
<b>Převod dat (Marshaling)</b>	<b>842</b>
<b>Využití komponenty COM klientem v .NET</b>	<b>842</b>
Vytvoření komponenty COM	843
Vytvoření volatelné obálky	848
Použití RCW	850
Primární interoperabilní sestavení	851
Problémy s podprocesy	851
Přidání přípojných bodů	852
Ovládací prvky ActiveX ve formulářích	855
<i>Import ovládacího prvku ActiveX</i>	855
<i>Vytvoření okenní aplikace</i>	855
Objekty COM v ASP.NET	857
<b>Komponenty .NET v klientech COM</b>	<b>858</b>
Volatelná obálka COM	858
Vytvoření komponenty v .NET	859
Vytvoření typové knihovny	860
Atributy pro interoperabilitu s COM	862
Registrace COM	866

Vytvoření klienta COM	867
Přidání přípojných bodů	868
Vytvoření klienta s objektem příjemce	870
Ovládací prvky formulářů Windows v Internet Exploreru	871
<b>Volání platformy</b>	<b>872</b>
<b>Shrnutí</b>	<b>876</b>

## Část IV: Data

### 25. Práce se soubory a systémovým registrem **879**

<b>Správa souborového systému</b>	<b>880</b>
Třídy .NET reprezentující soubory a složky	881
Třída Path	883
Příklad: prohlížeč souborů	884
<b>Přesouvání, kopírování a odstraňování souborů</b>	<b>889</b>
Příklad: FilePropertiesAndMovement	890
Kód příkladu FilePropertiesAndMovement	890
<b>Čtení ze souborů a zápis do nich</b>	<b>893</b>
Čtení souboru	894
Zápis do souboru	895
Proudy	896
Proudy s vyrovnávací pamětí	898
Čtení a zápis binárních souborů pomocí třídy FileStream	899
<i>Třída FileStream</i>	899
<i>Příklad: BinaryFileReader</i>	901
Čtení a zápis textových souborů	904
<i>Třída StreamReader</i>	905
<i>Třída StreamWriter</i>	907
<i>Příklad: ReadWriteText</i>	908
<b>Čtení informací o jednotkách</b>	<b>910</b>
<b>Zabezpečení souborů</b>	<b>913</b>
Čtení seznamů pro řízení přístupu k souboru	913
Čtení seznamů řízení přístupu ke složce	915
Přidávání a vyjímání seznamů řízení přístupu k souboru	916
<b>Práce se systémovým registrem</b>	<b>918</b>
Systémový registr	918
Třídy platformy .NET pro práci s registrem	921
Příklad: SelfPlacingWindow	923
<b>Čtení z izolovaného úložiště a zápis do něj</b>	<b>929</b>
<b>Shrnutí</b>	<b>934</b>

---

<b>26. Přístup k datům</b>	<b>935</b>
<b>Přehled technologie ADO.NET</b>	<b>936</b>
Jmenné prostory	936
Sdílené třídy	937
Třídy specifické pro různé databáze	937
<b>Použití databázových připojení</b>	<b>939</b>
Správa připojovacích řetězců	940
Efektivní práce s připojeními	942
<i>První možnost: try...catch...finally</i>	942
<i>Druhá možnost: příkaz using</i>	943
Transakce	944
<b>Příkazy</b>	<b>946</b>
Spouštění příkazů	946
<i>ExecuteNonQuery()</i>	947
<i>ExecuteReader()</i>	947
<i>ExecuteScalar()</i>	948
<i>ExecuteXmlReader()</i> (pouze poskytovatel SqlConnection)	949
Volání uložených procedur	950
<i>Volání uložené procedury, která nic nevrací</i>	951
<i>Volání uložené procedury, která vrací výstupní parametry</i>	952
<b>Rychlý přístup k datům: DataReader</b>	<b>953</b>
<b>Správa dat a relací: třída DataSet</b>	<b>956</b>
Datové tabulky	957
<i>Datové sloupce</i>	958
<i>Datové řádky</i>	959
<i>Generování schématu</i>	961
Datové relace	963
Datová omezení	965
<i>Nastavení primárního klíče</i>	965
<i>Nastavení cizího klíče</i>	966
<i>Nastavení omezení aktualizace a odstranění</i>	967
<b>Schémata XML: Generování kódu pomocí XSD</b>	<b>967</b>
<b>Zaplnění datové sady</b>	<b>974</b>
Zaplnění datové sady pomocí datového adaptéru	974
<i>Použití uložené procedury v datovém adaptéru</i>	975
Zaplnění objektu DataSet ze souboru XML	976
<b>Trvalé změny datové sady</b>	<b>976</b>
Aktualizace pomocí datových adaptérů	976
<i>Vložení nového řádku</i>	977
<i>Aktualizace existujícího řádku</i>	978
<i>Odstranění řádku</i>	979
Zápis výstupu do XML	979

<b>Práce s technologií ADO.NET</b>	<b>981</b>
Vývoj vrstev	981
Generování klíčů pomocí SQL Serveru	982
Konvence názvů	984
<i>Konvence pro databázové tabulky</i>	984
<i>Konvence pro databázové sloupce</i>	985
<i>Konvence pro omezení</i>	985
<i>Uložené procedury</i>	985
<b>Shrnutí</b>	<b>986</b>
<b>27. LINQ pro SQL</b>	<b>987</b>
<b>LINQ pro SQL a Visual Studio 2008</b>	<b>989</b>
Volání tabulky Products pomocí LINQ pro SQL: vytvoření konzolové aplikace	989
Přidání třídy LINQ pro SQL	990
Úvod do návrháře O/R	991
Vytvoření objektu Product	992
<b>Jak se objekty zobrazují na objekty LINQ pro SQL</b>	<b>994</b>
DataContext	995
<i>Příkaz ExecuteQuery</i>	995
<i>Vlastnost Connection</i>	996
<i>Transakce</i>	996
<i>Další metody a vlastnosti třídy DataContext</i>	997
Třída Table<TEntity>	998
<b>Práce bez návrháře O/R</b>	<b>999</b>
Vytvoření vlastního objektu	999
Dotazy pomocí vlastní třídy a LINQ	1000
Omezení sloupců v dotazu	1002
Práce s názvy sloupců	1002
Vytvoření vlastní třídy DataContext	1003
<b>Vlastní objekty a návrhář O/R</b>	<b>1004</b>
<b>Dotazy do databáze</b>	<b>1006</b>
Dotazovací výrazy	1006
Podrobnosti dotazovacích výrazů	1007
Filtrování pomocí výrazů	1007
Spojování tabulek	1008
Seskupování položek	1010
<b>Uložené procedury</b>	<b>1011</b>
<b>Shrnutí</b>	<b>1012</b>
<b>28. Manipulace s XML</b>	<b>1013</b>
Podpora standardů XML na platformě .NET	1014
Představujeme jmenný prostor System.Xml	1014
Použití tříd System.Xml	1015

<b>Čtení a zápis XML pomocí proudů</b>	<b>1016</b>
Použití třídy XmlReader	1017
<i>Metody pro čtení</i>	1017
<i>Načtení dat atributů</i>	1020
Ověřování pomocí třídy XmlReader	1021
Použití třídy XmlWriter	1023
<b>Použití modelu DOM na platformě .NET</b>	<b>1024</b>
Použití třídy XmlDocument	1025
<i>Vkládání uzlů</i>	1027
<b>Použití objektů typu XPathNavigator</b>	<b>1030</b>
Jmenný prostor System.Xml.XPath	1030
<i>XPathDocument</i>	1030
<i>XPathNavigator</i>	1031
<i>XPathNodeIterator</i>	1032
<i>Použití tříd ze jmenového prostoru XPath</i>	1032
Jmenný prostor System.Xml.Xsl	1035
<i>Transformace XML</i>	1036
<i>Ladění XSLT</i>	1040
<b>XML a ADO.NET</b>	<b>1042</b>
Převod databázových dat pomocí ADO.NET do XML	1042
<i>Převod relačních dat</i>	1046
Převod XML na databázová data	1048
<b>Serializace objektů v XML</b>	<b>1050</b>
Serializace bez přístupu ke zdrojovému kódu	1059
<b>Shrnutí</b>	<b>1062</b>
<b>29. LINQ pro XML</b>	<b>1063</b>
<b>LINQ pro XML a .NET 3.5</b>	<b>1064</b>
Nové objekty pro tvorbu dokumentů XML	1064
Visual Basic 2008 jde ještě dále	1064
Jmenné prostory a předpony	1064
<b>Nové třídy v .NET 3.5 pro práci s XML</b>	<b>1064</b>
XDocument	1065
XElement	1065
XNamespace	1067
XComment	1069
XAttribute	1070
<b>Dotazování nad dokumenty XML pomocí LINQ</b>	<b>1071</b>
Dotazy nad statickými dokumenty XML	1071
Dotazy nad dynamickými dokumenty XML	1072
<b>Práce s dokumentem XML</b>	<b>1074</b>
Čtení z dokumentu XML	1075
Zápis do dokumentu XML	1076



---

<b>Spolupráce LINQ pro SQL a LINQ pro XML</b>	<b>1078</b>
Nastavení komponent LINQ pro SQL	1078
Dotazy do databáze a výstup ve formátu XML	1078
<b>Shrnutí</b>	<b>1080</b>
<b>30. Programování pro .NET s SQL Serverem</b>	<b>1081</b>

---

<b>Hostitel běhového systému .NET</b>	<b>1082</b>
<b>Microsoft.SqlServer.Server</b>	<b>1083</b>
<b>Uživatelsky definované typy</b>	<b>1084</b>
Vytvoření typů UDT	1084
Použití typů UDT	1091
Použití typů UDT z kódu na straně klienta	1091
<i>Uživatelsky definované agregáty</i>	<i>1092</i>
Vytváření uživatelsky definovaných agregátů	1093
Použití uživatelsky definovaných agregátů	1095
<b>Uložené procedury</b>	<b>1095</b>
Vytváření uložených procedur	1096
Použití uložených procedur	1097
<b>Uživatelsky definované funkce</b>	<b>1098</b>
Vytváření uživatelsky definovaných funkcí	1098
Použití uživatelsky definovaných funkcí	1098
<b>Spouště</b>	<b>1099</b>
Vytváření spouští	1099
Použití spouští	1101
<b>Datový typ XML</b>	<b>1101</b>
Tabulky s daty XML	1101
Čtení hodnot XML	1103
Dotaz na data	1106
<b>Jazyk XML DML</b>	<b>1108</b>
<b>Indexy XML</b>	<b>1109</b>
XML se silnou typovou kontrolou	1110
<b>Shrnutí</b>	<b>1112</b>
<b>Rejstřík</b>	<b>1113</b>

---



Kdybychom označili jazyk C# a související prostředí .NET Framework za nejdůležitější novou technologii pro vývojáře za posledních mnoho let, rozhodně bychom nepřeháněli. Návrh technologie .NET poskytuje nové prostředí, ve kterém lze vyvíjet téměř libovolné aplikace určené pro systém Windows, a C# je nový programovací jazyk, který byl navržen zejména pro spolupráci s technologií .NET. Pomocí jazyka C# můžete například napsat dynamickou webovou stránku, webovou službu XML, komponentu distribuované aplikace, komponentu pro přístup k databázi, klasickou aplikaci pro pracovní plochu systému Windows, nebo dokonce novou inteligentní klientskou aplikaci s podporou režimů online i offline. Tato kniha se zabývá platformou .NET Framework 3.5. Programujete-li s verzí 1.0, 1.1, 2.0 nebo i 3.0, je možné, že některé části knihy nebudete moci využít. Pokusíme se upozorňovat na funkce, které se objevily až v platformě .NET Framework 3.5.

Nenechte se zmást označením .NET. NET jako součást názvu má zdůraznit přesvědčení společnosti Microsoft, že cestu vpřed představují distribuované aplikace, kde je výpočetní zpracování rozděleno mezi klienta a server. Jazyk C# však neslouží pouze k psaní internetových nebo síťových aplikací. Nabízí možnosti pro tvorbu téměř libovolných typů programů nebo komponent, které můžete v systému Windows požadovat. Jazyk C# spolu s technologií .NET mají zásadně změnit způsob psaní programů a usnadnit programování v systému Windows jako nikdy předtím.

To je poměrně odvážné tvrzení, které je nutné obhájit. Všichni nakonec víme, jak rychle se počítačové technologie mění. Každý rok přichází společnost Microsoft s novými programy, programovacími nástroji nebo verzemi systému Windows a prohlašuje přitom, že budou mimořádným přínosem pro vývojáře. V čem jsou tedy technologie .NET a jazyk C# jiné?

## Důležitost technologie .NET a jazyka C#

Abychom porozuměli významu technologie .NET, je vhodné si připomenout podstatu několika technologií pro systém Windows, které se objevily v posledních asi deseti letech. Na první pohled se sice mohou značně lišit, ale všechny operační systémy Windows od verze Windows 3.1 (uvedena na trh roku 1992) až po Windows Server 2008 mají ve svém jádru stejné, důvěrně známé rozhraní Windows API. S novými verzemi systému Windows přibývalo v rozhraní API mnoho nových funkcí, ale jednalo se o vývoj a rozšiřování rozhraní, nikoli o jeho nahrazení.

Totéž lze říci o mnoha technologiích a platformách, pomocí nichž jsme vyvíjeli software pro systém Windows. Například model **COM (Component Object Model)** vycházel ze služeb **OLE (Object Linking and Embedding – propojování a vkládání objektů)**. V té době se do značné míry jednalo pouze o metodu, jak propojit různé typy dokumentů sady Office, abyste mohli například vložit malou tabulku v Excelu do dokumentu ve Wordu. Dalším vývojem vznikl model **COM, DCOM (Distributed COM)** a nakonec **COM+**: pokračíla technologie, která se stala základem komunikace téměř všech komponent, stejně jako implementace transakcí, služeb zasílání zpráv a sdružování objektů.

Společnost Microsoft zvolila tento evoluční přístup k softwaru samozřejmě proto, že jí záleží na zpětné kompatibilitě. V průběhu let vyvinuli jiní dodavatelé rozsáhlou programovou základnu pro systém Windows. Kdyby v každé nové verzi společnost Microsoft zavedla novou technologii, která by neumožnila používat existující programy, nedosáhl by její systém takového úspěchu.

Zpětná kompatibilita sice byla klíčovou vlastností technologií Windows a jednou ze silných stránek této platformy, avšak má i velkou nevýhodu. Pokaždé, když se nějaká technologie vyvíjí a získává nové funkce, postupně se přitom komplikuje.

Bylo jasné, že něco se musí změnit. Společnost Microsoft nemohla věčně rozšiřovat stejné vývojové nástroje a jazyky a neustále zvyšovat jejich složitost, aby uspokojila konfliktní požadavky podpory nejnovějšího hardwaru a udržování zpětné kompatibility s technologiemi, které se používaly v době, kdy se systém Windows začátkem 90. let poprvé masově rozšířil. Chcete-li přijít s jednoduchou, ale přesto pokročilou sadou jazyků, prostředí a vývojových nástrojů, které vývojářům umožní snadnou tvorbu špičkových programů, musíte dříve nebo později začít s čistým stolem.

Přesně takový nový začátek představuje jazyk C# a technologie .NET. Zjednodušeně řečeno je .NET platforma (aplikační rozhraní, API) k programování pro Windows. Spolu s platformou .NET Framework se objevuje jazyk C#, který byl od základů navržen tak, aby spolupracoval s technologií .NET a zároveň využil veškerý pokrok ve vývojových prostředích a koncepcích objektově orientovaného programování, ke kterému došlo během posledních dvaceti let.

Na tomto místě bychom měli zdůraznit, že zpětná kompatibilita zůstala zachována. Existující programy budou fungovat i nadále a technologie .NET byla navržena tak, aby se stávajícím softwarem spolupracovala. Komunikace mezi softwarovými komponentami v systému Windows je v současnosti téměř výhradně založena na modelu COM. S ohledem na tento fakt může technologie .NET poskytnout obálky (wrapper) kolem existujících komponent COM, aby s nimi mohly komunikovat komponenty .NET.

Je pravda, že chcete-li psát kód pro technologii .NET, nemusíte se naučit jazyk C#. Společnost Microsoft rozšířila jazyk C++, nabídla jiný nový jazyk s názvem J# a zásadně přepracovala jazyk Visual Basic, který se změnil na výkonnější jazyk Visual Basic .NET. Tím umožnila uplatnit kód napsaný v libovolném z těchto jazyků v prostředí .NET. Tyto jazyky jsou však zatíženy dědictvím mnohaletého vývoje, neboť při jejich vývoji ještě neexistovaly dnešní technologie.

Tato kniha vás vybaví znalostmi programování v jazyku C# a současně poskytne potřebné základní informace o fungování architektury .NET. Nebudeme se zabývat pouze principy jazyka C#, ale uvedeme si také příklady aplikací, které používají různé související technologie, včetně přístupu k databázím, dynamických webových stránek, pokročilé grafiky a přístupu k adresářům. Předpokládáme pouze znalost minimálně jednoho jazyka vyšší úrovně, který se používá v systému Windows: může to být C++, Visual Basic nebo J++.

## Výhody technologie .NET

Obecně jsme se zmínili o tom, jak je technologie .NET skvělá, ale zatím jsme příliš nevysvětlili, jak může vývojářům usnadnit život. V této části si stručně rozebereme některé vylepšené funkce této technologie.

- **Objektově orientované programování:** Platforma .NET Framework a jazyk C# jsou již od začátku kompletně založeny na objektově orientovaných principech.

- **Dobrý návrh:** Knihovna základních tříd je od základu navržena vysoce intuitivním způsobem.
- **Jazyková nezávislost:** V technologii .NET se všechny jazyky, tj. Visual Basic .NET, C#, J# a spravované (managed) C++, překládají do společného *zprostředkujícího jazyka* (Intermediate Language). To znamená, že jazyky mohou spolupracovat způsobem, který dříve nebyl k dispozici.
- **Lepší podpora dynamických webových stránek:** Prostředí ASP sice poskytovalo značnou pružnost, ale zároveň nebylo příliš efektivní, protože používalo interpretované skriptovací jazyky. Nedostatečný objektově orientovaný návrh také často vedl ke vzniku nepřehledného kódu ASP. Technologie .NET nabízí integrovanou podporu webových stránek pomocí nové technologie ASP.NET. V rámci ASP.NET je kód stránek překládán a lze jej psát v jazyce vysoké úrovně kompatibilním s technologií .NET, jako je např. C#, J# nebo Visual Basic 2008.
- **Účinný přístup k datům:** Sada komponent .NET souhrnně označovaná jako ADO.NET zajišťuje efektivní přístup k relačním databázím a různým zdrojům dat. Jsou také k dispozici komponenty, které poskytují přístup k systému souborů a k adresářům. Zejména je do technologie .NET integrována podpora jazyka XML, což umožňuje manipulaci s daty, která lze importovat nebo exportovat na jiné platformy než Windows.
- **Sdílení kódu:** Technologie .NET zcela mění způsob sdílení kódu mezi aplikacemi. Zavádí koncepci **sestavení** (assembly), která nahrazují tradiční knihovny DLL. Sestavení mají formální funkce pro správu verzí a je možné, aby vedle sebe existovaly různé verze sestavení.
- **Zlepšené zabezpečení:** Každé sestavení může také obsahovat integrované informace o zabezpečení, jež mohou přesně určovat, který uživatel nebo která kategorie uživatelů či procesů smí volat určité metody definovaných tříd. Tak získáváte velmi podrobnou kontrolu nad tím, jak lze zaváděná sestavení používat.
- **Instalace s nulovým dopadem:** Existují dva typy sestavení: sdílená (shared) a soukromá (private). Sdílená sestavení jsou společně knihovny dostupné všem programům, zatímco soukromá sestavení jsou určena pouze k použití s konkrétním softwarem. Soukromé sestavení je zcela soběstačné, takže se zjednodušuje proces instalace. Nevznikají žádné položky registru. Stačí umístit příslušné soubory do správné složky v systému souborů.
- **Podpora webových služeb:** Technologie .NET zahrnuje plně integrovanou podporu vývoje webových služeb, které lze vytvářet stejně snadno jako libovolný jiný typ aplikací.
- **Visual Studio 2008:** Pro technologii .NET se dodává vývojové prostředí Visual Studio 2005, které umožňuje stejně snadno pracovat s jazyky C++, C#, J# a Visual Basic 2005 i s kódem ASP.NET. Visual Studio 2008 integruje nejlepší funkce příslušných jazykově specifických prostředí Visual Studia .NET 2002/2003/2005 a Visual Studia 6.
- **C#:** C# je nový objektově orientovaný jazyk, který je určen pro spolupráci s technologií .NET.

V kapitole 1 se podrobněji zaměříme na výhody architektury .NET.

## Co je nového v platformě .NET Framework 3.5

Vydání první verze platformy .NET Framework (1.0) roku 2002 se setvalo s velkým nadšením. .NET Framework 2.0 byla představena roku 2005 a byla považována za klíčové vydání této platformy. Ačkoli .NET Framework 3.5 nepřináší tak rozsáhlé změny jako verze 2.0, je stále považována za významnou verzi produktu, která obsahuje mnoho podstatných nových vlastností.

Při každém vydání této platformy se společnost Microsoft snažila zajistit, aby byly dopady na vyvinutý kód co nejmenší. Dosud se jí to velmi dobře dařilo.

Nezapomeňte instalovat testovací server, abyste mohli upgrade svých aplikací na platformu .NET Framework 3.5 podrobně vyzkoušet. Není vhodné přímo aktualizovat aktivně používanou aplikaci.

Následující podkapitoly popisují některé novinky v platformě .NET Framework 3.5, C# 2008 a také nové funkce vývojového prostředí Visual Studio 2008, které je pro tuto platformu určeno.

## Implicitně typované proměnné

C# 2008 nyní umožňuje deklarovat proměnnou s tím, že její typ určí implicitně překladač. Zjistíte, že LINQ tuto možnost využívá při práci s dotazy. Využití této funkcionality lze s pomocí klíčového slova `var`:

```
var x = 5;
```

Překladač se u takového příkazu pokusí určit typ proměnné s pomocí hodnoty 5. V tomto případě bude příkaz interpretován takto:

```
int x = 5;
```

## Automaticky implementované vlastnosti (properties)

V C# 2008 je často opakovaný proces deklarování vlastností zase o něco snazší. Před touto verzí byly vlastnosti deklarovány například takto:

```
private int _myItem;
public int MyItem
{
    get {
        return myItem;
    }

    set {
        myItem = value;
    }
}
```

Nyní již můžete tuto práci přenechat překladači. Místo neustále opakovaného vkládání výše uvedeného kódu je teď možné využít následující konstrukci:

```
public int MyItem { get; set; }
```

Tento zápis vygeneruje výsledek zcela shodný s výše uvedeným (a značně delším) kódem. Překladač provede konverzi zkráceného zápisu na původní zápis vlastností za vás a umožní vám tak psát kód přehledněji a rychleji než kdy dřív.

## Inicializátory kolekcí a objektů

C# 2008 nyní umožňuje přiřadit hodnotu vlastnostem (property) objektu v okamžiku, kdy je tato vlastnost inicializována. Mějme například následující kód:

```
public class MyStructure
{
    public int MyProperty1 { get; set; }
    public int MyProperty2 { get; set; }
}
```

V C# 2008 pak můžeme objekt MyStructure inicializovat následovně:

```
MyStructure myStructure = new MyStructure() { MyProperty1 = 5, MyProperty2 = 10 };
```

Tato funkcionalita nám také umožňuje deklarovat několik prvků kolekce zároveň:

```
List<int> myInts = new List<int> () { 5, 10, 15, 20, 25 };
```

Tímto způsobem jsou všechna zadaná čísla přidána do kolekce stejně, jako bychom je přidali metodou Add().

## Vestavěná podpora ASP.NET AJAX

Ačkoli bylo možné vytvářet stránky využívající technologii ASP.NET Ajax již s pomocí .NET Framework 2.0, vyžadovalo to instalaci dalších komponent. Nyní je podpora ASP.NET AJAX zabudována přímo do ASP.NET 3.5 a Visual Studia 2008.

Každá stránka, kterou s pomocí ASP.NET s .NET Frameworkem 3.5 vytvoříte, nyní může využívat Ajax (veškerá nastavení Ajaxu jsou v souboru Web.config). V sadě nástrojů pro ASP.NET teď také najdete nové serverové ovládací prvky, které vám umožní doplnit do vašich webových aplikací funkcionalitu založenou na technologii AJAX. Více informací o ASP.NET Ajaxu je v kapitole 39.

## .NET Language Integrated Query Framework (LINQ)

LINQ je jednou z nejpříjemnějších – a nejočekávanějších – vlastností nového .NET. Microsoft poskytuje LINQ jako silně typované rozhraní pro přístup k externím zdrojům dat. LINQ dává vývojářům možnost pracovat s externími daty v rámci prostředí, na které jsou zvyklí, s pomocí objektů, které spolupracují s IDE, IntelliSense, a dokonce i s ladicími nástroji.

S použitím LINQ lze provádět dotazy vůči objektům, kolekcím, SQL databázím, XML souborům a mnoha dalším datovým zdrojům. Příjemné je, že bez ohledu na to, z jakého zdroje jsou data načítána, se k nim přistupuje vždy stejným způsobem, protože LINQ k nim poskytuje univerzální strukturovaný přístup.

Následující příklad ukazuje jak lze z XML dokumentu získat jména zákazníků:

```
XDocument xdoc = XDocument.Load(@"C:\Customers.xml");
var query = from people in xdoc.Descendants("CustomerName")
            select people.Value;
```

```
Console.WriteLine("{0} Customers Found", query.Count());
```

```
Console.WriteLine();  
foreach (var item in query)  
{  
    Console.WriteLine(item);  
}
```

Další aspekty využití LINQ jsou popsány v kapitolách 11, 27 a 29.

## Podpora různých verzí .NET Framework

Vývojáři .NET dnes v mnoha případech pracují s několika aplikacemi, které jsou určeny pro různé verze .NET Frameworku (2.0, 3.0, nyní 3.5). Bylo by těžkopádné udržovat na počítači souběžně několik verzí Visual Studia jen proto, aby bylo možné s těmito aplikacemi pracovat. Nejnovější verze Visual Studia 2008 proto umožňuje zvolit verzi platformy, se kterou chcete pracovat. Při vytváření nové aplikace tak máte možnost rozhodnout se, zda má být určena pro .NET Framework verze 2.0, 3.0 nebo 3.5.

## Podpora nejnovějších typů aplikací

Nedávno uvedený .NET Framework 3.0 přinesl některé zcela zásadní možnosti – mezi nimi i přístup k aplikacím využívajícím Windows Presentation Foundation (WPF) či knihovny Windows Communication Foundation (WCF).

Ve Visual Studiu 2008 naleznete prostředky pro sestavování těchto aplikací – všechny jsou k dispozici jako samostatný projekt, včetně nových nastavení, průvodců (wizard) a specifické podpory ze strany vývojového prostředí.

## Porovnání jazyka C# s jinými jazyky

Z určitého hlediska lze jazyk C# považovat za stejnou revoluci mezi programovacími jazyky, jakou v prostředí Windows představuje technologie .NET. Současně s tím, jak společnost Microsoft přidávala v posledním desetiletí další a další funkce do systému Windows a do rozhraní Windows API, rozrostly se i jazyky Visual Basic 2008 a C++. Jazyky Visual Basic a C++ sice díky tomu získaly mimořádné možnosti, ale oba jazyky také kvůli způsobu svého vývoje trpí různými problémy.

V případě jazyka Visual Basic 6 a předchozích verzí spočívala hlavní síla jazyka v tom, že jej bylo možné snadno pochopit a že usnadňoval mnoho programátorských úkolů, protože před vývojářem z větší části skrýval podrobnosti rozhraní Windows API a infrastruktury komponent COM. Kvůli tomu však jazyk Visual Basic nikdy nebyl plně objektově orientovaný a kód velkých aplikací proto brzy ztrácel přehlednost a obtížně se udržoval. Syntaxe jazyka Visual Basic byla navíc převzata ze starších verzí jazyka BASIC (který byl opět navržen tak, aby se jej mohli intuitivně naučit začínající programátoři, nikoli aby se v něm psaly velké komerční aplikace). Proto se naprosto nehodil k tvorbě kvalitně strukturovaných či objektově orientovaných programů.

Jazyk C++ má na druhé straně své kořeny v definici standardu ISO jazyka C++. Nesplňuje normu ISO úplně z toho důvodu, že společnost Microsoft napsala svůj první kompilátor C++ ještě před oficiálním vydáním standardu ISO. Velmi se však této normě blíží. Naneštěstí to vedlo ke dvěma problémům. Za prvé má norma ISO C++ své kořeny v technologii staré desítky let, což se projevuje v nedostatečné podpoře moderních koncepcí (např. řetězců v kódování Unicode a generování



dokumentace v XML) a v určitých zastaralých strukturách určených pro dřívější kompilátory (jako je oddělení deklarace od definice členských funkcí). Za druhé se společnost Microsoft současně snažila vyvinout C++ jako jazyk, který je určen k náročným úkolům v systému Windows. Aby toho dosáhla, musela jazyk rozšířit o velké množství specifických klíčových slov a také různých knihoven. Výsledek je, že v systému Windows se z tohoto jazyka stal naprostý „guláš“. Stačí se zeptat vývojářů v C++, na kolik definic řetězce si dokážou vzpomenout: `char*`, `LPTSTR`, `string`, `CString` (verze MFC), `CString` (verze WTL), `wchar_t*`, `OLECHAR*` atd.

V této fázi na scénu vstupuje zcela nové prostředí .NET, které bude zahrnovat nová rozšíření obou jazyků. Společnost Microsoft tento problém vyřešila přidáním dalších vlastních klíčových slov do jazyka C++ a kompletním přepracováním jazyka Visual Basic v prostředí Visual Basic .NET na jazyk Visual Basic 2008. Tento jazyk si zachovává některé základní syntaktické prvky jazyka VB, ale jeho návrh se natolik liší, že jej z praktického hlediska můžeme považovat za nový jazyk.

Za této situace se společnost Microsoft rozhodla poskytnout vývojářům alternativu – jazyk navržený speciálně pro technologii .NET a vytvořený zcela nově. Výsledkem je C#. Společnost Microsoft oficiálně popisuje jazyk C# jako „jednoduchý, moderní, objektově orientovaný a typově bezpečný programovací jazyk, který je odvozen od jazyků C a C++“. Většina nezávislých pozorovatelů by tento popis pravděpodobně opravila na „odvozený od jazyků C, C++ a Java“. Tyto definice jsou technicky přesné, ale příliš nezachycují krásu či eleganci jazyka. Syntakticky se jazyk C# velmi podobá jazykům C++ i Java do takové míry, že mnohá klíčová slova jsou shodná. C# také sdílí stejnou strukturu bloků se závorkami ({} ) pro označení bloků kódu a odděluje příkazy pomocí středníků. Výpis kódu C# na první pohled velmi připomíná kód v jazycích C++ nebo Java. Navzdory této prvotní podobnosti se však jazyk C# lze naučit mnohem snáze než C++ a jeho obtížnost je srovnatelná s jazykem Java. Jeho struktura lépe vyhovuje moderním vývojovým prostředím než oba starší jazyky a byl navržen tak, aby programátorům současně poskytl jednoduchost používání jazyka Visual Basic a v případě potřeby vysoký výkon a nízkoúrovňový přístup k paměti jako jazyk C++. Uvedme si některé vlastnosti jazyka C#:

- plná podpora tříd a objektově orientovaného programování, včetně dědičnosti rozhraní i implementace, virtuálních funkcí a přetížení operátorů,
- konzistentní a vhodně definovaná sada základních typů,
- integrovaná podpora automatického generování dokumentace ve formátu XML,
- automatické čištění dynamicky přidělované paměti,
- možnost označit třídy nebo metody uživatelsky definovanými atributy. Tato funkce může být užitečná pro dokumentaci a někdy má určitý vliv na překlad (např. při označení metod, aby se překládaly pouze v ladicích sestaveních),
- plný přístup ke knihovně základních tříd .NET a také snadná dostupnost rozhraní Windows API (pokud ho skutečně potřebujete, k čemuž nedochází příliš často),
- v případě potřeby jsou dostupné ukazatele a přímý přístup do paměti, ale jazyk je navržen takovým způsobem, že lze bez nich pracovat téměř ve všech situacích,
- podpora vlastností a událostí ve stylu jazyka Visual Basic,
- pouhou změnou možností překladače můžete překládat buď spustitelný soubor, nebo knihovnu komponent .NET, kterou může volat jiný kód stejným způsobem jako ovládací prvky ActiveX (komponenty COM),
- pomocí jazyka C# lze psát dynamické stránky ASP.NET a webové služby založené na XML.

Je vhodné uvést, že většina výše zmíněných vlastností se vztahuje i na jazyky Visual Basic 2008 a spravované C++. Nicméně díky tomu, že jazyk C# je od začátku navržen pro spolupráci s technologií .NET, je jeho podpora funkcí této platformy kompletnější a dostupná na základě vhodnější syntaxe, než je tomu u výše uvedených jazyků. Jazyk C# je sice velmi podobný jazyku Java, ale poskytuje některá vylepšení. Jazyk Java zejména není určen ke spolupráci s prostředím .NET.

Na závěr tohoto tématu bychom měli uvést několik omezení jazyka C#. Jednou z oblastí, pro které tento jazyk není určen, je vývoj časově kritického nebo extrémně zatěžovaného kódu, tj. kódu, kde skutečně záleží na tom, zda průběh smyčky vyžaduje 1 000 nebo 1 050 cyklů procesoru, a potřebujete uvolnit prostředky v té milisekundě, kdy již nejsou potřeba. Mezi nízkourovňovými jazyky si v této oblasti nejspíš i nadále udrží výsadní postavení C++. Jazyk C# postrádá některé klíčové funkce, které jsou nutné pro aplikace s mimořádně vysokým výkonem, včetně možnosti specifikovat funkce s modifikátorem inline a destruktory, u kterých je zajištěno spuštění v definovaných bodech kódu. Do této kategorie však patří pouze malá skupina aplikací.

## Požadavky na psaní a spouštění kódu v jazyce C#

Platformu .NET Framework 3.5 lze spustit v systémech Windows XP, Windows Server 2003, Windows Vista a nejnovějším Windows Server 2008. Chcete-li psát kód pomocí technologie .NET, musíte si nejdříve nainstalovat sadu .NET 3.5. Jestliže navíc neuvažujete o tom, že byste psali kód C# v textovém editoru nebo vývojovém prostředí jiného dodavatele, budete téměř jistě potřebovat i prostředí Visual Studio 2008. Spravovaný kód lze spustit i bez plné sady SDK, je ale nutné běhové prostředí .NET. Někdy je vhodné běhové prostředí .NET distribuovat spolu s kódem kvůli těm klientům, kteří si ho zatím nainstalovali.

## Témata popsaná v této knize

Tuto knihu zahájíme v následující kapitole celkovým přehledem architektury .NET, abyste získali základní znalosti potřebné k psaní spravovaného kódu. Zbytek knihy je rozdělen na části, které se týkají jazyka C# a jeho aplikací v mnoha oblastech.

### Část I: Jazyk C#

Tato část představuje dostatečný úvod do vlastního jazyka C#. Nevyžaduje znalosti žádného konkrétního jazyka, ačkoli předpokládá, že jste zkušenými programátory. Začneme přehledem základní syntaxe a datových typů jazyka C# a potom si rozebereme objektově orientované funkce tohoto jazyka, abychom mohli přejít k pokročilejším tématům programování v C#.

### Část II: Visual Studio

Tato část se zabývá celosvětově nejvyužívanějším IDE pro vývojáře C# – Visual Studiem 2008. Dvě kapitoly této části se zabývají nejlepšími způsoby využití tohoto nástroje k vytváření aplikací založených na .NET Framework 2.0 nebo 3.0. Dále pak osvětlují možnosti nasazování (deployment) projektů.

## Část III: Knihovny bázových tříd (Base class libraries)

V této části se podíváme na principy programování v prostředí .NET. Zejména se zaměříme na Visual Studio .NET, zabezpečení, zavádění podprocesů do aplikací pro .NET a na metody vytváření vlastních knihoven a sestavení.

## Část IV: Data

Budeme se zabývat přístupem k databázím pomocí ADO.NET a LINQ a interakcí se soubory a adresáři. Podrobně také analyzujeme podporu jazyka XML v technologii .NET a na straně operačního systému Windows a funkce platformy .NET v SQL Serveru 2008. V rámci rozsáhlého prostoru věnového LINQ se budeme především soustředit na LINQ v SQL a LINQ v XML.

## Část V: Prezentace

V této části se soustředíme na tvorbu klasických okenních aplikací pro Windows, které se v technologii .NET označují jako formuláře (Windows Forms). Formuláře jsou aplikace typu silných klientů a v prostředí .NET lze tyto typy aplikací vytvářet snadno a rychle. Kromě informací o formulářích pro Windows se také zaměříme na GDI+. Pomocí této technologie budeme vytvářet aplikace, které zahrnují pokročilou grafiku.

V této části si také vysvětlíme psaní komponent, které jsou spouštěny na webových serverech a fungují jako webové stránky. Patří sem mimořádné množství nových funkcí, které poskytuje prostředí ASP.NET 3.5. Konečně zde naleznete také postupy pro vytváření aplikací založených na Windows Presentation Foundation a VSTO.

## Část VI: Komunikace

Tato část je kompletně věnována komunikaci. Zabýváme se zde webovými službami pro komunikaci nezávislou na platformě, vzdálenou komunikací v .NET (remoting) pro komunikaci mezi klienty a servery v .NET, službami Enterprise Services pro služby na pozadí a komunikací s DCOM. Ukážeme si asynchronní odpojenou komunikaci pomocí služby Message Queuing. Taktéž se zde zmíníme o využití Windows Communication Foundation a Windows Workflow Foundation.

## Část VII: Přílohy

Tato část zahrnuje tři přílohy, které podrobně analyzují principy objektově orientovaného programování a obsahují také specifické programátorské informace o jazyku C#.

## Konvence

V knize používáme několik různých stylů textu a grafické úpravy, které pomáhají rozlišit různé typy informací. Následují příklady používaných stylů spolu s vysvětlením, co znamenají:

Odrážky jsou odsazené a každá nová odrážka je označena takto:

- Důležitá slova jsou formátována *kurzivou*.
- Klávesy, které je nutné stisknout na klávesnici, jsou zobrazeny takto: Ctrl, Enter, Ctrl+Enter.

Kód je uveden několika různými způsoby. Pokud je diskutované slovo součástí textu (například při vysvětlení cyklu `if...else`), je označeno tímto písmem. Jestliže se jedná o blok kódu, který lze zadat jako program a spustit, naleznete ho také v šedém rámečku:

```
public static void Main()
{
    AFunc(1,2,"abc");
}
// Pokud jsme se nedostali na konec, vrať hodnotu true, jinak
// nastav neplatnou pozici a vrať hodnotu false.
pos++;
if (pos < 4)
    return true;
else {
    pos = -1;
    return false;
}
```

Takovéto panely s tučným písmem obsahují kriticky důležité informace, které přímo souvisí s okolním textem a které byste neměli zapomenout.

Poznámky psané kurzivou vyjadřují různé tipy, triky a doplňující informace k aktuálnímu výkladu.

Syntaxe zápisu metod a vlastností má následující formát.

```
Regsvcs BookDistributor.dll [NázevAplikaceCOM+] [KnihovnaTypů.tbl]
```

Kurziva zde označuje odkazy na objekty, proměnné nebo hodnoty parametrů, které je nutné zadat. Hranaté závorky představují volitelné parametry.

## Zdrojový kód a přílohy

Při procházení příkladů v knize můžete podle svého uvážení veškerý kód zadávat ručně, nebo můžete použít soubory se zdrojovými kódy, které jsou doplňkem knihy. Všechny zdrojové kódy uvedené v této knize jsou k dispozici ke stažení na webu [www.wrox.com](http://www.wrox.com). Po zobrazení úvodní stránky webu jednoduše vyhledejte název knihy (pomocí pole Search či jednoho ze seznamů titulů) a klepněte na odkaz Download Code na stránce podrobností o knize, abyste si stáhli kompletní zdrojový kód ke knize.

Hodně knih mívá podobné tituly, a proto může být jednodušší vyhledávat podle jejího čísla ISBN – tato kniha má v originálním anglickém vydání ISBN 978-0-470-19137-8.

Po stažení kódu jej stačí dekomprimovat vhodným komprimačním nástrojem. Můžete také přejít na hlavní stránku společnosti Wrox a stáhnout si ho na adrese [www.wrox.com/dynamic/books/download.aspx](http://www.wrox.com/dynamic/books/download.aspx), kde naleznete dostupné kódy této knihy a všech dalších knih nakladatelství Wrox.

Lokalizované zdrojové kódy všech příkladů jsou ke stažení na adrese <http://knihy.cpress.cz/1472>.

## Errata

Snažíme se v maximální možné míře zajistit, aby text ani kód neobsahovaly žádné chyby. Nikdo však není dokonalý a k chybám dochází. Naleznete-li v některé z našich knih chybu, jako je překlep nebo chybná část kódu, budeme vám velmi vděční, když nám to oznámíte. Odesláním opravy možná jiným čtenářům ušetříte hodiny frustrací a současně pomůžete vylepšit kvalitu našich budoucích knih.

Pokud chcete otevřít stránku s opravami k této knize, přejděte na adresu [www.wrox.com](http://www.wrox.com) a vyhledejte název knihy pomocí pole Search nebo jednoho ze seznamů titulů. Na stránce podrobností o knize pak klepněte na odkaz Book Errata. Tato stránka obsahuje seznam všech oprav, které redaktoři nakladatelství Wrox zadali a zveřejnili. Úplný seznam knih včetně odkazů na opravy každé z nich je také k dispozici na adrese <http://www.wrox.com/misc-pages/booklist.shtml>.

Pokud na stránce Book Errata „svou“ chybu nenajdete, přejděte na stránku <http://www.wrox.com/contact/techsupport.shtml> a vyplňte formulář pro odeslání nalezené chyby. Informaci zkontrolujeme, a jestliže bude správná, vystavíme zprávu na stránce oprav ke knize a vyřešíme problém v následujících vydáních knihy.

## p2p.wrox.com

Chcete-li se zúčastnit diskuse s autory a uživateli knihy, přihlaste se do fór P2P na adrese [p2p.wrox.com](http://p2p.wrox.com). Webový systém fór dovoluje odesílat zprávy týkající se knih nakladatelství Wrox a souvisejících technologií a komunikovat s jinými čtenáři a odborníky. Fóra umožňují, abyste přihlásili svou e-mailovou adresu k odběru zvolených témat, když se na fórech objeví nové příspěvky. Těchto fór se účastní autoři a redaktoři nakladatelství Wrox, jiní experti z oboru a další čtenáři.

Na adrese <http://p2p.wrox.com> naleznete mnoho různých fór, která můžete využít nejen při čtení této knihy, ale také při vývoji svých vlastních aplikací. Chcete-li se k fórum připojit, postupujte podle následujících kroků:

1. Přejděte na adresu [p2p.wrox.com](http://p2p.wrox.com) a klepněte na odkaz Register.
2. Přečtěte si podmínky používání a klepněte na tlačítko Agree.
3. Zadejte informace požadované pro přihlášení a případně také nepovinné informace, které o sobě chcete uvést, a klepněte na tlačítko Submit.

Obdržíte e-mail s popisem, jak potvrdit svůj účet a dokončit proces registrace.

Číst zprávy na fórech můžete i bez přihlášení k P2P, ale pro posílání vašich vlastních zpráv již musíte být jejich členem.

Po přihlášení můžete číst nové zprávy a odpovídat na příspěvky jiných uživatelů. Zprávy si můžete číst kdykoli, když máte přístup k webu. Pokud byste chtěli dostávat nové zprávy z určitého fóra e-mailem, klepněte na ikonu Subscribe to this Forum vedle názvu fóra v jejich seznamu.

Chcete-li získat další informace o práci se systémem Wrox P2P, nezapomeňte si na stránce často kladených otázek P2P FAQs přečíst odpovědi na dotazy o fungování softwaru fóra a také na mnoho běžných otázek, které se týkají systému P2P a knih nakladatelství Wrox. Tyto často kladené otázky naleznete po klepnutí na odkaz FAQ na libovolné stránce systému P2P.

## Poznámka redakce českého vydání

I nakladatelství Computer Press, které pro vás tuto knihu přeložilo, stojí o zpětnou vazbu a bude na vaše podněty a dotazy reagovat. Můžete se obrátit na následující adresy:

Computer Press  
redakce počítačové literatury  
náměstí 28. dubna 48  
635 00 Brno-Bystrc

nebo

[knihy@cpress.cz](mailto:knihy@cpress.cz).

Další informace a případné opravy českého vydání knihy najdete na internetové adrese <http://knihy.cpress.cz/k1472>. Prostřednictvím uvedené adresy můžete též naší redakci zaslat komentář nebo dotaz týkající se knihy. Na vaše reakce se srdečně těšíme.