

Stručný obsah

1. Rychlý úvod do procedurálního programování.....	19
2. Datové typy	57
3. Datové typy představující kolekce	109
4. Řídící struktury a funkce	159
5. Moduly.....	193
6. Objektově orientované programování.....	229
7. Práce se soubory.....	279
8. Pokročilé techniky programování	329
9. Ladění, testování a profilování	399
10. Procesy a vlákna.....	423
11. Propojení v síti	439
12. Programování databází	455
13. Regulární výrazy	469
14. Úvod do syntaktické analýzy.....	491
15. Seznámení s programováním grafického uživatelského rozhraní	543

Obsah

Úvod 13

Uspořádání knihy.....	15
Získání a instalace Pythonu 3.....	16
Poděkování.....	17

Lekce 1

Rychlý úvod do procedurálního programování..... 19

Tvorba a spuštění programů napsaných v jazyku Python.....	20
Nádherné srdce jazyka Python.....	24
Oblast č. 1: Datové typy.....	25
Oblast č. 2: Odkazy na objekty.....	26
Oblast č. 3: Datové typy pro kolekce.....	28
Oblast č. 4: Logické operátory.....	31
Oblast č. 5: Příkazy pro řízení toku programu.....	35
Oblast č. 6: Aritmetické operátory.....	39
Oblast č. 7: Vstup a výstup.....	42
Oblast č. 8: Tvorba a volání funkcí.....	44
Příklady.....	46
Program bigdigits.py.....	46
Program generate_grid.py.....	49
Shrnutí.....	51
Cvičení.....	53

Lekce 2

Datové typy..... 57

Identifikátory a klíčová slova.....	58
Celočíselné typy.....	60
Celá čísla.....	61
Logické hodnoty.....	64
Typy s pohyblivou řádovou čárkou.....	64
Čísla s pohyblivou řádovou čárkou.....	65
Komplexní čísla.....	68
Desetinná čísla.....	69

Řetězce	71
Porovnávání řetězců	73
Řezání a krokování řetězců	74
Řetězcové operátory a metody	76
Formátování řetězců metodou str.format()	83
Kódování znaků.....	95
Příklady.....	98
Program quadratic.py	98
Program csv2html.py	100
Shrnutí	105
Cvičení.....	107

Lekce 3

Datové typy představující kolekce..... 109

Typy představující posloupnost	110
N-tice.....	110
Pojmenované n-tice.....	113
Seznamy.....	115
Množinové typy.....	122
Množiny.....	123
Zmrazené množiny	126
Typy představující mapování	127
Slovníky	128
Výchozí slovníky.....	135
Uspořádané slovníky	136
Procházení a kopírování kolekcí	138
Operace a funkce pro iterátory a iterovatelné objekty.....	138
Kopírování kolekcí.....	146
Příklady	148
Program generate_usernames.py	148
Program statistics.py	151
Shrnutí	155
Cvičení.....	156

Lekce 4

Řídicí struktury a funkce..... 159

Řídicí struktury.....	160
Podmíněné větvení.....	160
Cykly	161

Zpracování výjimek.....	163
Zachytávání a vyvolávání výjimek.....	163
Vlastní výjimky.....	167
Vlastní funkce.....	171
Jména a dokumentační řetězce.....	175
Rozbalení argumentů a parametrů.....	176
Přístup k proměnným v globálním oboru platnosti.....	178
Lambda funkce.....	180
Tvrzení.....	181
Příklad: <code>make_html_skeleton.py</code>	183
Shrnutí.....	188
Cvičení.....	189

Lekce 5

Moduly 193

Moduly a balíčky.....	194
Balíčky.....	197
Vlastní moduly.....	200
Přehled standardní knihovny Pythonu.....	209
Práce s řetězci.....	210
Programování na příkazovém řádku.....	211
Matematika a čísla.....	212
Datum a čas.....	212
Algoritmy a datové kolekce představující kolekce.....	214
Souborové formáty, kódování a perzistence dat.....	215
Práce se soubory, adresáři a procesy.....	218
Sítě a Internet.....	220
XML.....	222
Další moduly.....	224
Shrnutí.....	225
Cvičení.....	226

Lekce 6

Objektově orientované programování..... 229

Objektově orientovaný přístup.....	230
Objektově orientované principy a terminologie.....	231
Vlastní třídy.....	234
Atributy a metody.....	234
Dědičnost a polymorfismus.....	239

Řízení přístupu k atributům pomocí vlastností.....	241
Tvorba kompletních, plně integrovaných datových typů.....	243
Vlastní třídy představující kolekce.....	255
Tvorba tříd agregujících kolekce.....	256
Tvorba tříd představujících kolekce pomocí agregace.....	262
Tvorba tříd představujících kolekce pomocí dědičnosti.....	269
Shrnutí	275
Cvičení.....	277

Lekce 7

Práce se soubory..... 279

Zapisování a čtení binárních dat.....	284
Naložené objekty s volitelnou kompresí.....	285
Holá binární data s volitelnou kompresí.....	288
Zapisování a analyzování textových souborů.....	297
Zapisování textu	297
Analyzování textu.....	298
Analyzování textu pomocí regulárních výrazů.....	301
Zapisování a analyzování souborů XML.....	303
Stromy elementů	304
Model DOM (Document Object Model).....	307
Ruční zápis kódu jazyka XML	310
Analýza kódu jazyka XML pomocí rozhraní SAX (Simple API for XML)	311
Binární soubory s náhodným přístupem	314
Generická třída BinaryRecordFile	314
Příklad: Třídy modulu BikeStock.....	322
Shrnutí	326
Cvičení.....	327

Lekce 8

Pokročilé techniky programování 329

Další techniky procedurálního programování.....	330
Větvení pomocí slovníků	331
Generátorové výrazy a funkce	332
Dynamické provádění kódu a dynamické importy	334
Lokální a rekurzivní funkce	341
Dekorátory funkcí a metod.....	345
Anotace funkcí.....	349

Další objektově orientované programování.....	351
Řízení přístupu k atributům.....	352
Funktory.....	355
Správce kontextu.....	357
Deskriptory.....	360
Dekorátory tříd.....	365
Abstraktní báze třídy.....	368
Vícenásobná dědičnost.....	375
Metatřídy.....	377
Funkcionální styl programování.....	381
Částečná aplikace funkce.....	384
Korutiny.....	385
Příklad: Valid.py.....	393
Shrnutí.....	395
Cvičení.....	396

Lekce 9

Ladění, testování a profilování 399

Ladění.....	400
Syntaktické chyby.....	401
Chyby za běhu programu.....	402
Vědecké ladění.....	406
Testování jednotek.....	410
Profilování.....	416
Shrnutí.....	420

Lekce 10

Procesy a vlákna 423

Modul pro práci s více procesy.....	424
Modul pro práci s vlákny.....	428
Příklad: Vícevláknový program pro hledání slova.....	429
Příklad: Vícevláknový program pro hledání duplicitních souborů.....	432
Shrnutí.....	437
Cvičení.....	438

Lekce 11

Propojení v síti	439
Tvorba klienta TCP	441
Tvorba serveru TCP.....	446
Shrnutí	452
Cvičení.....	453

Lekce 12

Programování databází	455
Databáze DBM	456
Databáze SQL.....	460
Shrnutí	467
Cvičení.....	468

Lekce 13

Regulární výrazy	469
Jazyk Pythonu pro regulární výrazy	471
Znaky a třídy znaků.....	471
Kvantifikátory	472
Seskupování a zachytávání	474
Aserce a příznaky	475
Modul pro regulární výrazy	479
Shrnutí	488
Cvičení.....	489

Lekce 14

Úvod do syntaktické analýzy	491
Terminologie formy BNF a syntaktické analýzy.....	493
Ruční tvorba analyzátorů.....	497
Analyzování jednoduchých dat ve tvaru klíč-hodnota	497
Analyzování seznamu skladeb	500
Analýza bloků jakožto doménově specifického jazyka.....	502
Syntaktická analýza ve stylu jazyka Python pomocí nástroje PyParsing.....	511
Stručné seznámení s nástrojem PyParsing	511
Jednoduchá analýza dat ve tvaru klíč-hodnota.....	515
Analyzování seznamu skladeb	516

Analýza bloků jakožto doménově specifického jazyka.....	518
Syntaktická analýza logiky prvního řádu.....	523
Syntaktická analýza s nástrojem PLY podle nástrojů Lex a Yacc.....	528
Analýza jednoduchých dat ve tvaru klíč-hodnota	530
Analýza seznamu skladeb	532
Analýza bloků jakožto doménově specifického jazyka.....	534
Syntaktická analýza logiky prvního řádu.....	536
Shrnutí	540
Cvičení.....	541

Lekce 15

Seznámení s programováním grafického uživatelského rozhraní543

Programy ve stylu dialogových oken.....	547
Programy s hlavním oknem.....	552
Vytvoření hlavního okna.....	552
Vytvoření vlastního dialogového okna	563
Shrnutí	565
Cvičení.....	566

Závěrem.....569

Rejstřík571

Úvod

Python je pravděpodobně nejsnadněji osvojitelný programovací jazyk, který se nejkrásněji používá. Kód jazyka Python je srozumitelný pro čtení i zápis a k tomu je stručný bez jakéhokoli nádechu tajemna. Python je velmi expresivní jazyk, což znamená, že obvykle stačí napsat daleko méně řádků kódu jazyka Python, než kolik by jich bylo zapotřebí pro ekvivalentní aplikace napsanou třeba v jazyku C++ nebo Java.

Python je multiplatformní jazyk. Obecně lze tedy říci, že program napsaný v jazyku Python lze spustit ve Windows i v unixových systémech, jako je Linux, BSD a Mac OS X, pouhým zkopírováním souboru či souborů, které tvoří daný program, na cílový stroj, aniž by jej bylo nutné „sestavovat“ nebo kompilovat. Je možné vytvářet programy napsané v Pythonu, které používají funkčnost specifickou pro určitou platformu. To ale jen zřídka nezbytné, protože téměř celá standardní knihovna Pythonu a většina knihoven třetích stran jsou plně a transparentně multiplatformní.

Jednou z opravdu silných stránek Pythonu je, že se dodává se skutečně kompletní standardní knihovnou, díky čemuž můžeme provádět třeba stahování souboru z Internetu, rozbalování zkomprimovaného archivního souboru nebo vytváření webového serveru jen pomocí jediného nebo několika málo řádků kódu. A kromě standardní knihovny je k dispozici tisíce knihoven třetích stran, z nichž některé poskytují ve srovnání se standardní knihovnou výkonnější a sofistikovanější možnosti (např. síťová knihovna Twisted nebo numerická knihovna NumPy), zatímco jiné poskytují funkčnost, která je příliš specializovaná na to, aby byla zahrnuta do standardní knihovny (např. simulační balíček SimPy). Většina knihoven třetích stran je k dispozici v seznamu balíčků pro jazyk Python (pypi.python.org/pypi).

V jazyku Python lze programovat v procedurálním, objektově orientovaném a v menší míře též funkcionálním stylu, i když v jádru je Pythonu objektově orientovaným jazykem. V této knize si ukážeme, jak psát procedurální a objektově orientované programy, a osvojíme si též prvky funkcionálního programování v jazyku Python.

Účelem této knihy je prezentovat způsob, jakým psát programy ve správném stylu Pythonu 3, a po přečtení se stát užitečnou příručkou pro jazyk Python 3. Přestože Python 3 je spíše evolučním nežli revolučním pokračováním Pythonu 2, nejsou u něj starší postupy již vhodné nebo nezbytné, přičemž se objevilo několik nových, využívajících přednosti Pythonu 3. Python 3 je lepší jazyk než Python 2 – je totiž postaven na mnohaleté zkušenosti s Pythonem 2 a přidává spoustu nových možností (a současně vypouští ty, které se v Pythonu 2 neosvědčily), díky nimž je programování ještě příjemnější, pohodlnější, snazší a konzistentnější.

Cílem knihy je naučit jazyk Python, a přestože se v ní seznámíte s množstvím standardních knihoven Pythonu, nesetkáte se se všemi. To ale není žádný problém, protože po přečtení knihy budete mít o Pythonu dost znalostí na to, abyste použili jakoukoli ze standardních knihoven nebo z knihoven třetích stran, a také na to, abyste byly schopni vytvářet své vlastní knihovní moduly.

Kniha je navržena tak, aby byla užitečná pro různé skupiny čtenářů, mezi něž patří samouci a amatérští programátoři, studenti, vědci, inženýři a všichni ostatní, kteří potřebují v rámci své práce něco naprogramovat, a samozřejmě také profesionální vývojáři a počítačový odborníci. Ovšem k tomu, aby byla kniha použitelná pro tak široké spektrum čtenářů, aniž by přitom znudila nebo méně zkušené ztrácela, musí předpokládat alespoň nějaké zkušenosti s programováním (v libovolném jazyku). Především předpokládá základní znalosti v oblasti datových typů (jako jsou čísla a řetězce), datových typů představujících kolekce (jako jsou množiny a seznamy), řídicích struktur (jako jsou příkazy `if` a `while`) a funkcí. Kromě toho některé příklady a cvičení předpokládají základní znalost značkovacího jazyka HTML a některé ze specializovanějších lekcí na konci vyžadují alespoň základní orientaci v probíraném tématu. Například Lekce o databázích předpokládá základní znalost jazyka SQL.

Kniha je uspořádána s ohledem na maximální možnou produktivitu a rychlost. Na konci první lekce budete schopni psát v jazyku Python malé, ale užitečné programy. V každé další lekci se seznámíte s novými tématy a zároveň témata probíraná v předchozích lekcích budete často rozšiřovat a prohlubovat. To znamená, že při postupném pročítání jednotlivých lekcí můžeme kdykoliv přestat – a s dosud získanými znalostmi budete schopni psát ucelené programy. Potom se můžete samozřejmě pustit do dalšího čtení a naučit se pokročilejší a sofistikovanější techniky. Z tohoto důvodu se s některými tématy seznámíte v jedné lekci a pak je blíže prozkoumáte v další či v několika pozdějších lekcích.

Při výuce nového programovacího jazyka se objevují dva hlavní problémy. Prvním je, že někdy, když je nutné se naučit nějaký nový princip, tento princip závisí na jiném, který zase přímo či nepřímo závisí na tom prvním. Druhý problém tkví v tom, že na začátku může čtenář o jazyku vědět jen něco málo neb vůbec nic, takže je velice obtížné prezentovat zajímavé nebo užitečné příklady či cvičení. V této knize se budeme snažit vyřešit oba problémy. První předpokládáním nějakým předchozích zkušeností s programováním a druhý představením „nádherného srdce“ jazyka Python v lekci 1, což je osm klíčových oblastí jazyka Python, které jsou samy o sobě dostatečné pro tvorbu ucházejících programů. Důsledkem tohoto přístupu je, že v prvních lekcích jsou některé příklady v trošičku umělém stylu, poněvadž používají pouze to, co jsme se do místa jejich prezentace naučili. Tento vliv se s každou další lekcí zmenšuje, a to až do konce lekce 7, kde jsou všechny příklady zapsány stylem, který je pro Python 3 naprosto přirozený.

Přístup knihy je veskrze praktický, takže budete vyzýváni, abyste si příklady a cvičení sami vyzkoušeli a získali tak určitou praxi. Kdykoliv to bude možné, použijeme pro příklady kompletní programy a moduly představující realistické případy užití. Příklady, řešení pro cvičení a errata ke knize jsou k dispozici na stránce <http://knihy.cpress.cz/K1747>.

I když je nejlepší používat nejnovější verzi Pythonu 3, nemusí to být vždy možné, pokud uživatelé nemohou nebo nechtějí svoji verzi Pythonu modernizovat. Každý příklad v této knize funguje s Pythonem 3.0, přičemž příklady a funkční prvky specifické pro Python 3.1 jsou výslovně uvedeny.

Přestože je možné tuto knihu použít pro vývoj softwaru, který používá pouze Python 3.0, měli by všichni, kteří chtějí vytvářet software, který se bude používat řadu let a který by měl být kompatibilní s pozdějšími vydáními Pythonu 3.x, používat Python ve verzi 3.1 a podporovat tuto verzi jako nejstarší verzi Pythonu 3. To je dáno zčásti tím, že Python 3.1 nabízí několik velice pěkných nových možností, ale především tím, že vývojáři Pythonu důrazně doporučují používat Python 3.1 (nebo

novější). Vývojáři se rozhodli, že Python 3.0.1 bude posledním vydáním v řadě 3.0.y a že již žádná další vydání v této řadě nebudou, a to ani tehdy, pokud se objeví nějaké chyby či bezpečnostní problémy. Chtějí totiž, aby všichni uživatelé Pythonu 3 přešli k Pythonu 3.1 (nebo k novější verzi), který bude mít běžná vydání s opravami chyb a bezpečnostních problémů.

Uspořádání knihy

Lekce 1 prezentuje osm klíčových oblastí jazyka Python, které jsou dostatečné pro psaní kompletních programů. Dále popisuje některá z dostupných programovacích prostředí Pythonu a prezentuje dva malinké programy sestavené s využitím osmi klíčových oblastí jazyka Python probíraných v dřívější části lekce.

Lekce 2 až 5 představují prvky procedurálního programování jazyka Python, včetně jeho základních datových typů, datových typů představujících kolekce a řady užitečných vestavěných funkcí a řídicích struktur společně s velmi jednoduchou prací se soubory. Lekce 5 ukazuje, jak vytvářet vlastní moduly a balíčky, a poskytuje přehled standardní knihovny Pythonu, abyste měli dobrou představu o funkcích, které jsou v Pythonu ihned k dispozici – a díky kterým nemusíte znovu objevovat kolo.

Lekce 6 poskytuje důkladné seznámení s objektově orientovaným programováním v jazyku Python. Veškerá látka týkající se procedurálního programování, kterou jste se naučili v předchozích lekcích, i nadále platí, protože objektově orientované programování je postaveno na procedurálních základech. Využívá tak například stejné datové typy, datové typy představující kolekce a řídicí struktury.

Lekce 7 se věnuje zápisu a čtení souborů. V případě binárních souborů se navíc jedná o kompresi a náhodný přístup a u textových souborů o syntaktickou analýzu prováděnou ručně a pomocí regulárních výrazů. Tato Lekce dále ukazuje, jak zapisovat a číst soubory XML, včetně použití stromů elementů, modelu DOM (Document Object Model – objektový model dokumentu) a rozhraní SAX (Simple API for XML – jednoduché aplikační rozhraní pro XML).

Lekce 8 reviduje látku probíranou v několika předchozích lekcích a prozkoumává řadu pokročilejších prvků jazyka Python v oblasti datových typů a datových typů představujících kolekce, řídicích struktur, funkcí a objektově orientovaného programování. Tato Lekce dále představuje spoustu nových funkcí, tříd a pokročilých technologií, včetně funkcionálního stylu programování a použití korutin. Probíraná témata jsou sice náročná, ale zato velice užitečná.

Lekce 9 se od všech předchozích lekcí liší v tom, že místo představování nových prvků jazyka Python probírá techniky a knihovny pro ladění, testování a profilování programů.

Zbývající lekce se věnují nejrůznějším pokročilým tématům. Lekce 10 ukazuje techniky pro rozložení pracovní zátěže programu do více procesů nebo vláken. Lekce 11 ukazuje, jak pomocí standardní podpory Pythonu pro komunikace přes síť vytvářet aplikace s architekturou klient-server. Lekce 12 se věnuje databázovému programování (jednoduché soubory DBM s daty ve tvaru klíč-hodnota i databáze SQL).

Lekce 13 vysvětluje a demonstuje minijazyk regulárních výrazů v Pythonu a věnuje se modulu pro regulární výrazy. Lekce 14 pokračuje dále a ukazuje základní techniky syntaktické analýzy pomocí regulárních výrazů a také použití dvou modulů třetích stran, PyParsing a PLY. Nakonec Lekce 15 představuje programování grafického uživatelského rozhraní (Graphical User Interface neboli GUI)

pomocí modulu `tkinter`, který je součástí standardní knihovny Pythonu. Kniha má dále velmi stručný závěr a samozřejmě rejstřík.

Mnohé lekce jsou pro udržení související látky na jednom místě docela dlouhé. Nicméně lekce jsou rozděleny na části, oddíly a někdy i pododdíly, takže je lze číst takovým tempem, které vám nejlépe vyhovuje – třeba přečtením jedné části nebo jednoho oddílu najednou.

Získání a instalace Pythonu 3

Máte-li moderní a aktualizovaný unixový systém nebo Mac, pak již máte Python 3 nejspíše nainstalovaný, což ověříte zapsáním příkazu `python -V` (jedná se o velké písmeno V) do konzoly (`Terminal.app` v systému Mac OS X). Jedná-li se o verzi 3.x, pak je Python 3 již přítomen, takže nemusíte nic instalovat. Pokud Python nebyl vůbec nalezen, může to být tím, že má název, který obsahuje číslo verze. Zkuste napsat `python3 -V`, a pokud ani to nefunguje, tak `python3.0 -V` nebo `python3.1 -V`. Pokud některá z těchto možností funguje, pak víte, že již máte Python nainstalovaný, a znáte jeho verzi i název. (V této knize používáme název `python3`, můžeme ale používat takový název, který u vás funguje, například `python3.1`.) Pokud nemáte nainstalovanou žádnou verzi Pythonu 3, čtěte dále.

Pro systémy Windows a Mac OS X jsou k dispozici snadno použitelné grafické instalační balíčky, které vás provedou instalačním procesem krok za krokem. Můžete je stáhnout na adrese www.python.org/download. Pro Windows stáhněte balíček „Windows x86 MSI Installer“, pokud si ovšem nejste jisti, že váš stroj má jiný procesor, pro který je dodáván jiný instalátor. Máte-li například AMD64, sáhněte po balíčku „Windows X86-64 MSI Installer“. Jakmile instalační balíček získáte, stačí jej už jen spustit a řídit se pokyny na obrazovce.

Pro Linux, BSD a další unixové systémy (kromě systému Mac OS X, pro nějž je k dispozici instalační soubor `.dmg`) spočívá nejjednodušší způsob instalace Pythonu v použití systému pro správu balíčků vašeho operačního systému. Ve většině případů je Python k dispozici v několika samostatných balíčcích. Například v systému Ubuntu (od verze 8) existuje `python3.0` pro Python, `idle-python3.0` pro editor IDLE (jednoduché vývojové prostředí) a `python3.0-doc` pro dokumentaci – společně se spoustou dalších balíčků, které vedle standardní knihovny poskytují doplňky s dalšími funkčními prvky. (Pro Python ve verzi 3.1 budou názvy balíčků samozřejmě začínat `python-3.1`.)

Pokud na vašem systému nejsou k dispozici žádné balíčky s Pythonem 3, pak musíte stáhnout zdrojový kód z adresy www.python.org/download a sestavit Python úplně od začátku. Stáhněte jeden z archivů `tarball` se zdroji a v případě komprese `gzip` jej rozbalte příkazem `tar xvzf Python-3.1.tgz` nebo v případě komprese `bzip2` příkazem `tar xvfj Python-3.1.tar.bz2`. (Číslo verze se může lišit, například `Python-3.1.1.tgz` nebo `Python-3.1.2.tar.bz2`, ale stačí jednoduše nahradit 3.1 skutečným číslem verze.) Konfigurace sestavení probíhá standardním způsobem. Nejdříve se přesuňte do nově vytvořeného adresáře `Python-3.1` a spusťte `./configure`. (Pro lokální instalaci můžete použít volbu `--prefix`.) Dále spusťte `make`.

Je možné, že na konci obdržíte několik zpráv oznamujících, že ne všechny moduly bylo možné sestavit. To obvykle znamená, že na svém počítači nemáte některé z požadovaných knihoven nebo hlaviček. Pokud například nelze sestavit modul `readline`, použijte systém pro správu balíčků pro nainstalování odpovídající vývojové knihovny – například `readline-devel` na systémech na bázi distribuci Fedora nebo `readline-dev` na systémech na bázi distribuce Debian, jako je například Ubuntu. Další

modul, který se nemusí ihned sestavit, je modul `tkinter`, který závisí na vývojových knihovnách `Tcl` a `Tk`, což jsou moduly `tk1-devel` a `tk-devel` na systémech na bázi distribuce Fedora a moduly `tk18.5-dev` a `tk8.5-dev` na systémech na bázi distribuce Debian (s tím, že vedlejší verze nemusí být 5). Naneštěstí nejsou názvy příslušných balíčků na první pohled zřejmé, a proto může být nutné obrátit se s žádostí o pomoc na diskuzní fórum Pythonu. Po nainstalování chybějících balíčků spusťte znovu `./configure` a `make`.

Po úspěšném provedení příkazu `make` se můžete spuštěním příkazu `make test` přesvědčit, zda je všechno v pořádku. Není to ale nezbytné a navíc může dokončení tohoto příkazu trvat spoustu minut.

Pokud použijete volbu `--prefix` pro lokální instalaci, pak stačí spustit `make install`. Pokud v případě Pythonu 3.1 instalujete třeba do adresáře `~/local/python31`, pak přidáním adresáře `~/local/python31/bin` do své proměnné prostředí `PATH` budete schopni spouštět Python příkazem `python3` a editor IDLE příkazem `idle3`. Pokud již máte lokální adresář pro spustitelné soubory, který se nachází v proměnné prostředí `PATH` (např. `~/bin`), pak můžete místo změny proměnné `PATH` přidat symbolické odkazy. Máte-li spustitelné soubory například v adresáři `~/bin` a Python jste nainstalovali do adresáře `~/local/python31`, pak můžete vytvořit vhodné odkazy spuštěním příkazů `ln -s ~/local/python31/bin/python3 ~/bin/python3` a `~/local/python31/bin/idle3 ~/bin/idle3`. Pro účely této knihy jsme v systémech Linux a Mac OS X přesně takto provedli lokální instalaci a přidali symbolické odkazy, přičemž ve Windows jsme použili binární instalátor.

Pokud nepoužijete volbu `--prefix` a máte přístup uživatele „root“, přihlaste se jako „root“ a proveďte příkaz `make install`. Na systémech podporujících příkaz `sudo`, jako je například Ubuntu, spusťte příkaz `sudo make install`. Je-li v systému Python 2, adresář `/usr/bin/python` se nezmění a Python 3 bude dostupný jako `python3.0` (nebo `python3.1` podle nainstalované verze) a od verze Python 3.1 také jako `python3`. Editor IDLE pro Python 3.0 se nainstaluje jako `idle`, takže pokud potřebujete i nadále přístup k editoru IDLE pro Python 2, musíte před provedením instalace starý editor IDLE přejmenovat (např. na `/usr/bin/idle2`). Python 3.1 nainstaluje editor IDLE jako `idle3`, takže k žádnému konfliktu s editorem IDLE pro Python 2 nedochází.

Poděkování

Nejdříve bych chtěl poděkovat za odezvu, kterou jsem obdržel od čtenářů první edice, kteří mi poskytli připomínky ohledně oprav, návrhů nebo obojího.

Mé další poděkování míří k odborným recenzentům knihy, počínaje Jasminem Blanchettem, který je počítačovým odborníkem, programátorem a spisovatelem, s nímž jsem spolupracoval na dvou knihách o C++ a knihovně Qt. Jeho zapojení do plánování lekcí, jeho rady, kritika všech příkladů i jeho pečlivé čtení významným způsobem zlepšily kvalitu této knihy.

Georg Brandl je přední vývojář a dokumentátor v oblasti Pythonu odpovědný za vytvoření nové sady dokumentačních nástrojů. Všiml si spousty zákeřných chyb a velice trpělivě a neústupně je vysvětloval, dokud nebyly pochopeny a opraveny. Dále provedl řadu zlepšení v rámci příkladů.

Phil Thompson je expertem na jazyk Python a tvůrcem knihovny PyQt, což je pravděpodobně nejlepší knihovna GUI pro Python. Jeho bystrozraká a podnětná odezva vedla k řadě vyjasnění a korekcí.

Trenton Schulz je hlavní softwarový inženýr ve společnosti Qt Software (před odkoupením společností Nokia známé jako Trolltech), který byl cenným recenzentem všech mých předchozích knih a který mi opět přišel na pomoc. Pozorně přečetl a množství jeho připomínek napomohlo k ujasnění řady problémů a vedlo k značným zlepšením v textu.

Kromě výše zmíněných recenzentů, z nichž každý přečetl celou knihu, nesmím zapomenout na Davida Boddieho, předního autora odborných titulů ve společnosti Qt Software, zkušeného odborníka na jazyk Python a vývojáře softwaru s otevřeným zdrojovým kódem, který přečetl a poskytl cennou odezvu na několik částí této knihy.

Pro tuto druhou edici bych také rád poděkoval Paulu McGuireovi (autorovi modulu PyParsing), který byl tak laskav a zkontroloval příklady využívající modul PyParsing, které se objevily v nové lekci věnované syntaktické analýze, a který mi poskytl spoustu uvážených a užitečných rad. A pro stejnou lekci zkontroloval David Beazley (autor modulu PLY) příklady využívající modul PLY a postaral se o cennou odezvu. Kromě toho Jasmin Blauche, Treon Schulz, Georg Braudla Phil Thompson přečetli většinu z nového materiálu této druhé edice a poskytli mi velice hodnotnou zpětnou vazbu.

Díky patří také Guidovi van Rossumovi, tvůrci jazyka Python, jakož i širší komunitě kolem Pythonu, která se významným způsobem podílela na tvorbě Pythonu a zvláště jeho knihoven, které jsou nesmírně užitečné a které je radost používat.

A jako vždy děkuji Jeffu Kingstonovi, tvůrci jazyka Lout pro sazbu písma, který používám již více než deset let.

Zvláštní díky patří mé redaktorce Debre Williams Cauley za její podporu a také za to, že se opět postarala, aby měl celý proces co nejhladší průběh. Děkuji též Anně Popick, která se tak dobře starala o produkční proces, a korektorovi Audrey Doyle, který opět odvedl naprosto skvělou práci. A v souvislosti s touto druhou edicí chci též poděkovat Jennifer Lindnerové za pomoc při udržování nového materiálu na srozumitelné úrovni a japonskému překladateli první edice Takahiro Nagaovi za odhalení zákeřných chyb, které jsem měl možnost v této edici opravit.

V neposlední řadě bych chtěl poděkovat své ženě Andree za to, že zvládla mé buzení ve čtyři hodiny ráno, kdy často přicházely nápady a opravy kódu, které se tu a tam dožadovaly poznamenání nebo otestování, a za její lásku, věrnost a podporu.