

Capítulo 1: Abrindo o apetite

Se você trabalha muito com computadores, acabará encontrando alguma tarefa que gostaria de automatizar. Por exemplo, você pode querer fazer busca-e-troca em um grande número de arquivos de texto, ou renomear e reorganizar um monte de arquivos de fotos de uma maneira complicada. Talvez você gostaria de escrever um pequeno banco de dados personalizado, ou um aplicativo GUI especializado, ou um jogo simples.

Se você é um desenvolvedor de software profissional, pode ter que trabalhar com várias bibliotecas C/C++/Java, mas o tradicional ciclo escrever/compilar/testar/recompilar é muito lento. Talvez você esteja escrevendo um conjunto de testes para uma biblioteca e está achando tedioso codificar os testes. Ou talvez você tenha escrito um programa que poderia utilizar uma linguagem de extensão, e você não quer conceber e implementar toda uma nova linguagem para sua aplicação.

Python é a linguagem para você.

Você poderia escrever um script para o shell do Unix ou arquivos em lote do Windows para algumas dessas tarefas, mas scripts shell são bons para mover arquivos e alterar textos, mas não adequados para aplicações GUI ou jogos. Você poderia escrever um programa em C/C++/Java, mas pode tomar tempo de desenvolvimento para chegar até um primeiro rascunho. Python é mais simples, está disponível em Windows, Mac OS X, e sistemas operacionais Unix, e vai ajudá-lo a fazer o trabalho mais rapidamente.

Python é fácil de usar, sem deixar de ser uma linguagem de programação de verdade, oferecendo muito mais estruturação e suporte para programas extensos do que shell scripts oferecem. Por outro lado, Python também oferece melhor verificação de erros do que C, e por ser uma linguagem de *muito alto nível*, ela possui tipos nativos de alto nível: dicionários e vetores (arrays) flexíveis. Devido ao suporte nativo a uma variedade de tipos de dados, Python é aplicável a um domínio de problemas muito mais vasto do que Awk ou até mesmo Perl, ainda assim muitas tarefas são pelo menos tão fáceis em Python quanto nessas linguagens.

Python permite que você organize seu programa em módulos que podem ser reutilizados em outros programas escritos em Python. A linguagem provê uma vasta coleção de módulos que podem ser utilizados como base para sua aplicação -- ou como exemplos para estudo e aprofundamento. Alguns desses módulos implementam manipulação de arquivos, chamadas do sistema, sockets, e até mesmo acesso a bibliotecas de construção de interfaces gráficas, como Tk.

Python é uma linguagem interpretada, por isso você pode economizar um tempo considerável durante o desenvolvimento, uma vez que não há necessidade de compilação e vinculação (*linking*). O interpretador pode ser usado interativamente, o que torna fácil experimentar diversas características da linguagem, escrever programas “descartáveis”, ou testar funções em um desenvolvimento bottom-up. É também uma útil calculadora de mesa.

Python permite a escrita de programas compactos e legíveis. Programas escritos em Python são tipicamente mais curtos do que seus equivalentes em C, C++ ou Java, por diversas razões:

- os tipos de alto nível permitem que você expresse operações complexas em um único comando;
- a definição de bloco é feita por indentação ao invés de marcadores de início e fim de bloco;
- não há necessidade de declaração de variáveis ou parâmetros formais;

Python é *extensível*: se você sabe como programar em C, é fácil adicionar funções ou módulos diretamente no interpretador, seja para desempenhar operações críticas em máxima velocidade, ou para vincular programas Python a bibliotecas que só estejam disponíveis em formato binário (como uma biblioteca gráfica de terceiros). Uma vez que você tenha sido fisgado, você pode vincular o interpretador Python a uma aplicação escrita em C e utilizá-la como linguagem de comandos ou extensão para esta aplicação.

A propósito, a linguagem foi batizada a partir do famoso show da BBC “Monty Python’s Flying Circus” e não tem nada a ver com répteis. Fazer referências a citações do show na documentação não é só permitido, como também é encorajado!

Agora que você está entusiasmado com Python, vai querer conhecê-la com mais detalhes. Partindo do princípio que a melhor maneira de aprender uma linguagem é usando-a, você está agora convidado a fazê-lo com este tutorial.

No próximo capítulo, a mecânica de utilização do interpretador é explicada. Essa informação, ainda que mundana, é essencial para a experimentação dos exemplos apresentados mais tarde.

O resto do tutorial introduz diversos aspectos do sistema e linguagem Python por intermédio de exemplos. Serão abordadas expressões simples, comandos, tipos, funções e módulos. Finalmente, serão explicados alguns conceitos avançados como exceções e classes definidas pelo usuário.