1. Um dich auf den Geschmack zu bringen

Wer viel am Computer arbeitet, kommt irgendwann zu dem Schluss, dass es Aufgaben gibt, die er gern automatisieren würde. Beispielsweise ein Suchen-und-Ersetzen für eine Vielzahl von Dateien oder eine Möglichkeit, einen Haufen Fotodateien auf komplizierte Art umzubenennen oder umzuräumen. Oder man hätte gerne eine kleine Datenbank nach Maß, eine spezialisierte GUI-Anwendung oder ein einfaches Spiel.

Als professioneller Softwareentwickler muss man vielleicht mit mehreren C/C++/Java-Bibliotheken arbeiten, findet aber den üblichen Schreiben/Kompilieren/Testen/Re-Kompilieren-Zyklus zu langsam. Wer eine Testsuite für solch eine Bibliothek schreibt, hält es vielleicht für eine ermüdende Aufgabe, den Testcode zu schreiben. Vielleicht hat der ein oder andere auch ein Programm geschrieben, das eine Erweiterungssprache gebrauchen könnte, will aber keine ganz neue Sprache für sein Programm entwerfen und implementieren.

Dann ist Python genau die richtige Sprache!

Man könnte natürlich Unix-Shellskripte oder Windows-Batchdateien für ein paar dieser Aufgaben schreiben. Mit Shellskripten lassen sich gut Dateien verschieben und Textdaten verändern, zur Entwicklung von GUI-Applikationen oder Spielen sind sie aber weniger geeignet. Man könnte ein entsprechendes C/C++/Java-Programm dafür schreiben, aber es kostet in der Regel bereits viel Entwicklungszeit, um überhaupt einen ersten Programmentwurf zu entwickeln. Python ist einfacher zu nutzen, verfügbar für Windows-, Mac OS X- und Unix-Betriebssysteme und hilft, die Aufgabe schneller zu erledigen.

Python ist einfach in der Anwendung, aber eine echte Programmiersprache, die viel mehr Struktur und Unterstützung für große Programme bietet, als Shellskripte oder Batchdateien es könnten. Auf der anderen Seite bietet Python auch mehr Fehlerüberprüfungen als C und hat, als stark abstrahierende Hochsprache, mehr abstrakte Datentypen wie flexible Arrays und Wörterbücher (Dictionaries) eingebaut. Aufgrund seiner allgemeineren Datentypen ist Python in ausgedehnteren Problembereichen einsetzbar als Awk oder sogar Perl, und dennoch sind viele Dinge in Python mindestens so einfach wie in diesen Sprachen.

Python erlaubt die Aufteilung von Programmen in Module, die in anderen Python-Programmen wiederverwendet werden können. Es kommt mit einer großen Sammlung von Standardmodulen, die als Grundlage für eigene Programme genutzt werden können; oder als Beispiele, um in Python Programmieren zu lernen. Manche der Module stellen Datei-I/O, Systemaufrufe, Sockets und sogar Schnittstellen zu GUI-Toolkits wie Tk bereit.

Python ist eine interpretierte Sprache, wodurch sich bei der Programmentwicklung erheblich Zeit sparen lässt, da Kompilieren und Linken nicht nötig sind. Der Interpreter kann interaktiv genutzt werden, so dass man einfach mit den Fähigkeiten der Sprache experimentieren, Wegwerf-Code schreiben oder Funktionen während der Bottom-Up-Programmentwicklung testen kann. Es ist auch ein praktischer Tischrechner.

Python ermöglicht die Entwicklung von kompakten und lesbaren Programmen. Programme, die in Python geschrieben sind, sind aus mehreren Gründen viel kürzer als C/C++/Java-Äquivalente:

- Die abstrakten Datentypen erlauben es, komplexe Operationen in einer einzigen Anweisung auszudrücken;

- Anweisungen werden durch Einrückungen und nicht durch öffnende und schließende Klammern gruppiert;

- Variablen- oder Argumentdeklarationen sind nicht nötig.

Python ist erweiterbar: Wer in C programmieren kann, kann einfach eine neue eingebaute Funktion oder ein Modul zum Interpreter hinzuzufügen. Entweder um zeitkritische Operationen mit maximaler Geschwindigkeit auszuführen oder um Python-Programme zu Bibliotheken zu linken, die nur in binärer Form (wie beispielsweise herstellerspezifische Grafikbibliotheken) verfügbar sind. Wenn man erst einmal mit Python vertraut ist, kann man den Python-Interpreter zu in C geschriebenen Applikationen linken und Python als Erweiterung oder Kommandosprache für diese Applikation nutzen.

So nebenbei: Die Sprache ist nach der BBC-Sendung “Monty Python’s Flying Circus” benannt und hat nichts mit Reptilien zu tun. Anspielungen auf Monty Python-Sketche in Dokumentation zu benutzen ist nicht nur erlaubt, sondern gern gesehen.

Jetzt, da du nun ganz heiß auf Python bist, wirst du mehr wissen und lernen wollen. Der beste Weg, um eine Sprache zu erlernen, ist ganz sicher der, sie einzusetzen. Darum lädt das Tutorial gerade dazu ein, während des Lesens mit dem Python-Interpreter zu experimentieren.

Im nächsten Kapitel wird der technische Teil der Nutzung des Interpreters erläutert. Das ist eher nüchterne Information, aber wichtig, um die später gezeigten Beispiele ausprobieren zu können.

Der Rest des Tutorials stellt anhand von Beispielen unterschiedliche Möglichkeiten und Sprachelemente von Python vor. Dies beginnt mit einfachen Ausdrücken (Expressions), Anweisungen (Statements) und Datentypen und geht weiter mit Funktionen und Modulen. Danach werden fortgeschrittene Konzepte wie Ausnahmen (Exceptions) und benutzerdefinierte Klassen behandelt.