IN DIESEM KAPITEL

- Mit Ihrem Computer kommunizieren
- Programme erstellen, um mit Ihrem Computer zu sprechen
- Programme verstehen und lernen, sie zu erstellen
- Überlegen, warum Sie Python nutzen wollen

Kapitel 1
Die Kommunikation mit Ihrem Computer

Wenn es darum ging, mit einem Computer zu sprechen, handelte es sich vor gar nicht allzu langer Zeit regelmäßig noch um Drehbücher zu Science-Fiction-Filmen. Die Besatzung der *Enterprise* in *Star Trek* hat zum Beispiel regelmäßig mit ihrem Computer gesprochen. Und tatsächlich wurden sie von dem nicht nur verstanden, sondern sie erhielten vom Computer meist auch eine Antwort. Seit der Verbreitung von Apples Siri (`www.apple.com/de/ios/siri`), Amazons Echo (`https://www.amazon.com/dp/B00X4WHP5E/`) und anderer interaktiver Software sind derartige Gespräche heute allerdings durchaus möglich.

Den Computer nach Informationen zu fragen, ist eine Sache, ihn mit Befehlen zu füttern, eine andere. In diesem Kapitel wird diskutiert, warum Sie Ihrem Computer überhaupt etwas befehlen wollen und was Sie davon haben. Sie werden auch feststellen, dass für diese spezielle Kommunikation besondere Sprachen benötigt werden und warum Sie für derartige Aufgaben Python benutzen sollten. Das wichtigste Fazit, das Sie aus diesem Kapitel ziehen können, besteht darin, dass Programmieren einfach nur eine Art der Kommunikation ist, die anderen Formen der Kommunikation ähnelt, die Sie mit Ihrem Computer bereits pflegen.

Warum wollen Sie sich mit Ihrem Computer unterhalten?

Sich mit Maschinen zu unterhalten, scheint zunächst einmal ziemlich langweilig. Das ist aber nötig, da Computer ja keine Gedanken lesen können, auch wenn sie auch das vielleicht noch lernen können. Und selbst, wenn Computer tatsächlich Gedanken lesen könnten, würden Sie ja weiterhin mit Ihnen kommunizieren. Ohne den Austausch von Daten zwischen den Maschinen und Ihnen passiert erst einmal gar nichts. Aktivitäten wie

- ✔ das Lesen Ihrer E-Mails
- ✔ das Schreiben von Urlaubsberichten
- ✔ die Suche nach dem weltbesten Geschenk

sind alles Beispiele für Kommunikation zwischen Computern und Ihnen. Die grundlegende Idee besteht darin, dass nur dann Ergebnisse produziert werden können, wenn kommuniziert wird. Und das trifft natürlich auch auf jedwede Kommunikation zu, die zwischen Ihrem Computer und anderen Rechnern oder Menschen stattfindet.

Solange Sie nicht genauer darüber nachdenken, ist die Kommunikation für Sie meist transparent. Wenn Sie beispielsweise online einen Chatroom betreten, entsteht dabei der Eindruck, dass Sie direkt mit anderen Personen kommunizieren. Tatsächlich kommunizieren Sie aber mit Ihrem Computer, der über den Chatroom mit dem Computer Ihres Kommunikationspartners »spricht«, der wiederum mit seinem Besitzer kommuniziert. Abbildung 1.1 soll die Geschehnisse im Hintergrund verdeutlichen.

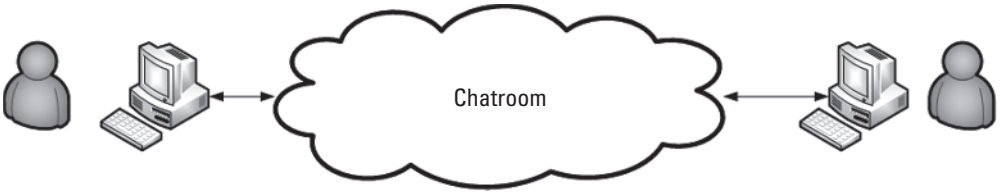

Abbildung 1.1: Die Kommunikation mit Ihrem Computer bleibt transparent, solange Sie nicht anfangen, genauer darüber nachdenken.

Betrachten Sie die Wolke in der Mitte von Abbildung 1.1. Sie könnte alles Mögliche enthalten. Wie Sie aber natürlich wissen, enthält sie auf jeden Fall andere Computer, die andere Anwendungen ausführen. Durch diese Computer können Sie und Ihre Freunde miteinander chatten. Und nun denken Sie einmal daran, wie einfach der gesamte Vorgang bei der Verwendung eines Chatprogramms ist. Trotz der komplexen Prozesse im Hintergrund kommt es Ihnen doch so vor, als würden Sie sich einfach nur mit Freunden unterhalten.

Anwendungen als eine Art von Kommunikation

Die Kommunikation mit dem Computer erfolgt durch Anwendungen. Sie nutzen zum Beispiel Anwendungen, um Ihre E-Mails zu beantworten, andere für Ihre Einkäufe und wieder andere zum Erstellen von Präsentationen. *Anwendungen* (*Programme*) die häufig nur *App* genannt werden, stellen eine Methode dar, um Computern menschliche Ideen so zu präsentieren, dass er sie verstehen kann. Außerdem werden durch Anwendungen die Werkzeuge bestimmt, mit denen die Daten spezifisch für die Kommunikation aufbereitet werden. Daten, die Präsentationsinhalte repräsentieren, unterscheiden sich von den Daten, die beim Einkauf von Geschenken für Ihre Mutter benötigt werden. Die Art und Weise der Betrachtung, Verwendung und Interpretation der Daten ist für die verschiedenen Aufgaben unterschiedlich. Also müssen Sie auch verschiedene Anwendungen (oder Teilanwendungen) benutzen, wenn Sie Daten derart verarbeiten wollen, dass sie von Computern und Ihnen selbst verstanden werden.

Sie können quasi für jede heute vorstellbare Aufgabenstellungen Anwendungen schreiben. Tatsächlich können Sie auf Anwendungen zugreifen, von denen Sie nicht einmal wissen, wofür sie überhaupt benötigt werden. Programmierer erstellen seit Jahren Abermillionen Anwendungen. Daher ist es vielleicht zunächst kaum nachvollziehbar, warum Sie selbst neue Anwendungen schreiben sollten, um mit Ihren Computern zu kommunizieren. Letztlich sollten Sie die Antwort finden, wenn Sie darüber nachdenken, welche Daten Ihnen vorliegen und wie diese interaktiv verarbeitet werden sollen. Einige Daten werden vielleicht zu selten benötigt, als dass es Programmierern bisher in den Sinn gekommen wäre, dafür Anwendungen zu schreiben. Vielleicht liegen die Daten auch in einer Form vor, die aktuell von keiner Anwendung unterstützt wird. Also können Sie Ihrem Computer die Daten (ohne jeweils übermäßigen Aufwand) kaum sinnvoll präsentieren, ohne eigene Anwendungen dafür zu schreiben.

In den folgenden Abschnitten werden Anwendungen unter dem Gesichtspunkt beschrieben, dass jeweils ganz spezifische Daten auf ganz spezielle Art und Weise weiterverarbeitet werden müssen. Zum Beispiel könnten Sie auf eine Videodatenbank zugreifen, ohne dabei sinnvoll mit den Daten arbeiten zu können. Sowohl die Daten als auch Ihre Zugriffsanforderungen sind sehr speziell. Deshalb reift in Ihnen der Gedanke heran, eine eigene Anwendung zu schreiben, die sowohl den Daten als auch Ihren Anforderungen gerecht wird.

Alltägliche Arbeitsabläufe

Bei *Arbeitsabläufen* oder *Prozessen* handelt es sich einfach um einen Satz von Schritten, die Sie bei der Ausführung einer Aufgabe befolgen. Um sich beispielsweise einen Toast zu machen, könnten Sie diese Schritte durchführen:

1. Sie holen Brot und Butter aus dem Kühlschrank.
2. Sie öffnen die Brottüte und nehmen zwei Scheiben Toast heraus.
3. Sie stecken das Kabel des Toasters in die Steckdose.

4. Sie geben je eine Scheibe Brot in einen Schlitz.
5. Sie drücken den Toasterhebel herunter, um den Toastvorgang zu starten.
6. Sie warten, bis der Toastvorgang abgeschlossen ist.
7. Sie nehmen den Toast aus dem Toaster.
8. Sie legen die Toastscheiben auf einen Teller.
9. Sie bestreichen den Toast mit Butter.

Vielleicht gehen Sie beim Toasten auch anders als hier vorgestellt vor, aber es ist recht unwahrscheinlich, dass Sie den Toast vor dem Toasten mit Butter bestreichen. Natürlich müssen Sie den Toast erst einmal aus der Verpackung holen, bevor Sie ihn toasten (den Toast samt Verpackung und allem Drum und Dran in den Toaster zu stecken, könnte zu unerwünschten Ergebnissen führen …). Die meisten Leute denken gar nicht über das Toastmachen nach. Trotzdem verwenden sie derartige Arbeitsabläufe, ohne darüber nachzudenken.

Ohne Ablaufplan können Computer keine Aufgaben ausführen. Sie müssen dem Computer sagen, welche Schritte er in welcher Reihenfolge ausführen soll, und Sie müssen ihn auf alle möglicherweise auftretenden Ausnahmen vorbereiten, die zu Fehlern führen können. Alle diese Angaben (und noch mehr) werden in Anwendungen beschrieben. Kurz gefasst sind Anwendungen einfach niedergeschriebene Arbeitsabläufe, die Sie verwenden, um Computern zu vermitteln, was sie *wann* und *wie* machen sollen. Da Sie bereits Ihr Leben lang Arbeitsabläufen folgen, müssen Sie dieses Wissen nur noch auf Dinge übertragen, die Computer für bestimmte Aufgaben wissen müssen.

Arbeitsabläufe aufschreiben

Als ich noch in der Grundschule war, erhielten wir von unserer Lehrerin die Aufgabe, etwas über das Toastmachen zu schreiben. Nachdem wir alle unsere Aufsätze abgegeben hatten, holte sie einen Toaster und ein paar Brote hervor. Jeder Aufsatz wurde dann vorgelesen und demonstriert. Zwar funktionierte keiner der von uns beschriebenen Abläufe wie vorgesehen, die produzierten Ergebnisse waren aber sehr lustig. Ich selbst hatte vergessen, die Lehrerin zu bitten, die Verpackung zu entfernen. Also versuchte sie pflichtbewusst, das Brot zusammen mit seiner Verpackung in den Toaster zu stecken. Diese Lektion ist bei mir hängen geblieben. Prozeduren aufzuschreiben, kann ziemlich schwer sein, da wir zwar genau wissen, was wir machen wollen, aber oft Schritte auslassen – wir nehmen einfach an, dass die andere Person auch genau weiß, was wir machen wollen.

Bei vielen alltäglichen Erfahrungen im Leben geht es einfach nur um die Kenntnis der erforderlichen Arbeitsabläufe. Denken Sie nur an die Checklisten, die von Piloten vor dem Start verwendet werden. Ohne standardisierte Abläufe könnte das Flugzeug abstürzen. Das Aufschreiben von Arbeitsabläufen kostet zwar Zeit, ist aber machbar. Vielleicht brauchen Sie mehrere Versuche, bis Sie Ihre Arbeitsabläufe voll funktionsfähig niedergeschrieben haben, wahrscheinlich werden Sie die Prozedur letztlich aber erstellen können. Es reicht aber nicht, den Arbeitsablauf nur aufzuschreiben, Sie müssen ihn auch von jemandem testen lassen,

der mit der zu erledigenden Aufgabe nicht vertraut ist. Arbeitet man mit Computern, sind diese natürlich die perfekten Testobjekte.

Anwendungen als gewöhnliche Arbeitsabläufe verstehen

Computer verhalten sich genauso wie die Grundschullehrerin im Beispiel des letzten Abschnitts. Wenn Sie eine Anwendung schreiben, schreiben Sie einen Prozess auf, der eine vom Computer auszuführende Schrittfolge definiert, um irgendeine von Ihnen vorgegebene Aufgabe zu erfüllen. Lassen Sie einen Schritt aus, werden die Ergebnisse nicht wie erwartet ausfallen. Der Computer versteht nicht, was Sie meinen, oder Sie selbst dachten, er würde irgendwas automatisch machen. Computer wissen nur, dass Sie ihnen einen Prozess vorgegeben haben und dass sie diesen ausführen sollen.

Computer nehmen alles wörtlich

Menschen gewöhnen sich an die Arbeitsabläufe, die Sie erstellen. Sie kompensieren automatisch Mängel in den Abläufen oder notieren sich jene Dinge, die Sie vergessen haben. Anders gesagt: Menschen kompensieren Probleme mit den von Ihnen erstellten Prozessen.

Wenn Sie mit dem Schreiben von Computerprogrammen beginnen, werden Sie wahrscheinlich erst einmal frustriert sein, weil Computer Aufgaben zwar äußerst präzise ausführen, dabei aber Ihre Anweisungen wörtlich nehmen. Sagen Sie dem Computer beispielsweise, dass ein bestimmter Wert gleich 5 sein soll, wird der Computer nach einem Wert suchen, der exakt 5 ist. Menschen könnten den Wert 4,9 sehen und dabei wissen, dass dieser noch gut genug ist. Computer sind anderer Ansicht. Computer sehen den Wert 4,9 und entscheiden, dass dieser nicht genau gleich 5 ist. Kurz gefasst sind Computer unflexibel, handeln wenig intuitiv und sind einfallslos. Wenn Sie einen Arbeitsablauf für einen Computer definieren, wird dieser jedes Mal präzise das tun, was Sie von ihm verlangen. Dabei wird er den Prozess nie eigenständig ändern und auch nie entscheiden, dass ihm eigentlich etwas ganz anderes aufgetragen wurde.

Was sind Anwendungen eigentlich?

Wie bereits erwähnt, können Anwendungen menschliche Ideen so ausdrücken, dass sie von Computern verstanden werden. Um dieses Ziel zu erreichen, benötigen Anwendungen einen oder mehrere Arbeitsabläufe, die dem Computer mitteilen, wie die Aufgaben zur Verarbeitung und Anzeige von Daten auszuführen sind. Sie sehen am Bildschirm zum Beispiel Text in Ihrem Textverarbeitungsprogramm. Damit Sie diese Daten sehen können, benötigt der Computer aber Vorgaben, wie er Daten von Festplatten lesen, sie in eine für Menschen verständliche Form bringen und schließlich am Bildschirm anzeigen kann. In den folgenden Abschnitten werden die Eigenschaften von Anwendungen detaillierter beschrieben.

Computer besitzen ihre eigene Sprache

Menschliche Sprachen sind komplex und schwer zu verstehen. Selbst Anwendungen wie Alexa oder Siri sind in ihren Möglichkeiten noch äußerst beschränkt, wenn es darum geht, das Gesagte zu verstehen. Mit der Zeit haben Computer zwar »gelernt«, menschliche Spracheingaben zu verarbeiten und bestimmte gesprochene Wörter als Befehle zu interpretieren. Computer verstehen die menschliche Sprache im Wesentlichen aber immer noch nicht wirklich. Und bei Dialekten ist schnell Schluss. (Das »Gennsefleisch« eines Sachsen wird deshalb von Nichtsachsen und erst recht von Computern wahrscheinlich kaum als »Können Sie vielleicht? verstanden werden.) Die Schwierigkeit der menschlichen Sprache lässt sich deshalb auch gut am Beispiel der Arbeit von Anwälten darstellen, die eine gewisse eigene Sprache pflegen. Wenn Sie sich Dokumente im Juristendeutsch anschauen, lesen sich diese oft wie das reinste Kauderwelsch. Ziel dieser Sprache ist es jedoch, keinerlei Spielraum für Interpretationen zu lassen. Anwälte haben schaffen es trotzdem nur selten, dieses Ziel vollständig zu erreichen, da die menschliche Sprache einfach zu ungenau ist und Juristen selbst häufig wieder Dinge voraussetzen, die für Normalsterbliche ganz und gar nicht selbstverständlich sind.

Wenn man sich nun ansieht, was Sie aus den vorigen Abschnitten dieses Kapitels wissen, könnten Computer sich niemals auf die menschliche Sprache verlassen, um die von Ihnen definierte Prozesse zu verstehen. Computer nehmen alles wörtlich und würden völlig unvorhersehbare Ergebnisse produzieren, wenn Anwendungen in menschlicher Sprache geschrieben würden. Darum müssen Menschen für die Kommunikation mit Computern spezielle *Programmiersprachen* verwenden. Mit diesen können Sie Prozesse präzise definieren, die dann auch für Computer und Menschen gleichermaßen verständlich sind.

Computer sprechen eigentlich gar keine Sprache. Sie verwenden Binärcodes, um Schalter intern umzulegen und mathematische Berechnungen durchzuführen. Computer verstehen nicht einmal Buchstaben, sondern nur Zahlen. Spezielle Anwendungen machen aus der computerspezifischen Sprache, die Sie zum Definieren von Arbeitsabläufen benötigen, Binärcodes. Aber für die Zwecke dieses Buches brauchen Sie sich keine besonderen Gedanken über derartige interne Details zu machen und müssen nicht wissen, wie Computer auf Binärebene funktionieren. Trotzdem ist es interessant zu wissen, dass die Kommunikation mit Computern über Mathematik und Zahlen erfolgt, also nicht wirklich mit einer Sprache.

Den Menschen helfen, sich mit dem Computer zu verständigen

Während des Programmierens ist es wichtig, dass Sie sich des Zwecks und der Zielsetzung der jeweiligen Anwendung immer bewusst sind. Anwendungen helfen Menschen, sich mit Computern auf bestimmte Weise zu unterhalten. Jede Anwendung benutzt irgendwelche Daten, die ihr als Eingabe übergeben, gespeichert, verändert und ausgegeben werden, sodass die menschlichen Benutzer der Anwendung ein gewünschtes Ergebnis erhalten. Ob es sich nun um ein Spiel oder eine Tabellenkalkulation handelt, die Grundidee ist dieselbe.

Computer verarbeiten Daten, die von Menschen (oder der Umwelt) bereitgestellt werden, um zu die erwünschten Ergebnisse zu erzielen.

Wenn Sie Anwendungen erstellen, stellen Sie Menschen neue Möglichkeiten für die Kommunikation mit Computern bereit. Mit dem von Ihnen neu entwickelten Ansatz können andere Menschen Daten auf neue Weise betrachten. Die Kommunikation zwischen Mensch und Computer sollte so einfach sein, dass die Programme eigentlich in den Hintergrund treten. Denken Sie an Programme, die Sie in der Vergangenheit verwendet haben. Die besten Anwendungen helfen Ihnen dabei, sich auf die zu verarbeitenden Daten zu konzentrieren. Zum Beispiel fesselt Sie ein Spiel nur dann, wenn Sie sich darauf konzentrieren können, einen Planeten zu retten oder ein Raumschiff zu fliegen, und nicht mit der Anwendung kämpfen müssen, die Ihnen das Spielen eigentlich nur ermöglichen soll.

Einer der besten Möglichkeiten, an die Entwicklung von Anwendungen heranzugehen, besteht darin, sich die Vorgehensweise anderer Programmierer anzusehen. Wenn Sie sich notieren, was Ihnen an anderen Programmen mehr oder weniger gefällt, können Sie besser herausfinden, wie Ihre eigenen Programme aussehen und funktionieren sollen. Folgende Dinge können Sie sich fragen, wenn Sie mit diesen Anwendungen arbeiten:

- ✔ Was finde ich an der Anwendung verwirrend?
- ✔ Welche Funktionen ließen sich einfach bedienen?
- ✔ Welche Funktionen waren schwierig zu benutzen?
- ✔ Wie hat mir die Anwendung geholfen, meine Daten zu verarbeiten?
- ✔ Wie ließe sich die Arbeit mit den Daten vereinfachen?
- ✔ Was will ich mit meiner Anwendung erreichen, was das vorhandene Programm nicht bietet?

Professionelle Entwickler stellen beim Programmieren von Anwendungen zwar noch viele andere Fragen, aber die vorgestellten Fragen sollten für den Anfang erst einmal reichen, um Ihnen Denkanstöße zu geben und deutlich zu machen, dass Anwendungen Menschen als Hilfsmittel bei der Kommunikation mit Computern dienen sollen. Wenn Sie sich jemals über irgendwelche Programme geärgert haben, wissen Sie schon, wie sich Ihre Anwender fühlen werden, wenn Sie sich vor der Programmierung nicht die richtigen Fragen gestellt haben. Kommunikation ist das wichtigste Element aller von Ihnen erstellten Anwendungen.

Sie können auch schon mal damit beginnen, über die Art und Weise nachzudenken, wie Sie arbeiten wollen. Dabei ist es sinnvoll, die jeweiligen Prozesse in einzelne Teilaufgaben zu zerlegen und für jeden Schritt alles aufzuschreiben, was Ihnen dazu einfällt. Wenn Sie damit fertig sind, bitten Sie einen Dritten, den Prozess probeweise durchzuspielen, um auszuprobieren, ob er wirklich funktioniert. Anfangs dürfte es Sie bestimmt überraschen, dass Sie trotz intensiver Bemühungen sehr schnell Schritte vergessen können.

Die schlechtesten Anwendungen der Welt beginnen normalerweise mit Programmierern, die nicht wissen, was die Anwendungen tun sollen, was sie so besonders macht oder welche Aufgabe sie für wen erfüllen sollen. Wenn Sie Anwendungen schreiben, überzeugen Sie sich davon, dass Sie wissen, warum diese erstellt werden und was damit erreicht werden soll. Einen Plan in der Tasche zu haben, hilft dabei, den Spaß am Programmieren nicht zu verlieren. Während Sie an Ihrem neuen Programm arbeiten, wird sich ein Ziel nach dem anderen erfüllen, bis die Anwendung fertig ist. Diese können Sie dann Ihren Freunden zeigen (die natürlich alle denken, wie cool es ist, dass Sie dieses Programm geschrieben haben).

Warum Python so cool ist

Heutzutage gibt es viele Programmiersprachen. Tatsächlich kann man ein ganzes Semester lang Computersprachen an Universitäten studieren und hat dann immer noch von allen existierenden Computersprachen wenigstens einmal gehört. Eigentlich sollte man meinen, dass sich Programmierer doch mit all diesen Programmiersprachen zufriedengeben und einfach eine auswählen könnten, um sich mit dem Computer zu verständigen. Aber trotzdem entwickeln sie immer wieder neue.

Programmierer entwickeln aus gutem Grund weiterhin neue Sprachen. Jede Sprache besitzt gewisse Besonderheiten, die sie außergewöhnlich gut beherrscht. Außerdem entwickeln sich auch Programmiersprachen weiter, um mit dem Fortschritt der Computertechnologie Schritt zu halten. Da es beim Schreiben von Anwendungen hauptsächlich um effiziente Kommunikation geht, beherrschen die meisten Programmierer mehrere Sprachen und können damit eine passende Sprache für bestimmte Aufgaben auswählen. Die eine Sprache eignet sich vielleicht besser für den Abruf von Daten aus Datenbanken, während sich mit einer anderen Benutzeroberflächen besonders gut erstellen lassen.

Wie andere Programmiersprachen kann Python einige Dinge außergewöhnlich gut und Sie sollten wissen, worum es sich dabei handelt, bevor Sie Python einsetzen. Sie werden vielleicht verblüfft sein, welche wirklich coolen Sachen Sie mit Python machen können. Beim Verwenden von Programmiersprachen ist es hilfreich, deren Stärken und Schwächen zu kennen. Außerdem können Sie Frust vermeiden, wenn Sie Sprachen nicht gerade für Dinge verwenden, für die sie sich kaum oder nur schlecht eignen. Die folgenden Abschnitte sollen Ihnen dabei helfen, derartige Entscheidungen für Python zu treffen.

Warum man Python verwenden sollte

Bei der Entwicklung der meisten Programmiersprachen stehen bestimmte Zielen im Vordergrund. Diese Ziele helfen dabei, die Charakteristika der Sprache zu definieren und festzulegen, was man mit der Sprache tun können soll. Es ist nicht wirklich möglich, eine Programmiersprache zu erfinden, die alles kann, da man es bei der Programmierung durchaus

mit widerstreitenden Zielen und Anforderungen zu tun haben kann. Die Hauptzielsetzung von Python bestand in der Entwicklung einer Programmiersprache, die Programmierer effizient und produktiv macht. Vor diesem Hintergrund zähle ich im Folgenden ein paar Gründe auf, die dafür sprechen, dass Sie Python zur Anwendungsentwicklung benutzen sollten:

- ✔ **Kürzere Entwicklungszeiten:** Python-Code ist normalerweise zwei bis zehn Mal kürzer als vergleichbarer Code in Sprachen wie C/C++ und Java. Dadurch benötigen Sie weniger Zeit für das Erstellen Ihrer Programme und es bleibt Ihnen mehr Zeit für deren Benutzung.
- ✔ **Leichtere Lesbarkeit:** Programmiersprachen sind wie beliebige andere Sprachen. Nur wenn sie leicht lesbar sind, können Sie verstehen, was sie machen. Python-Code ist tendenziell leichter lesbar als in anderen Sprachen geschriebener Code. Damit benötigen Sie weniger Zeit für das Interpretieren des Codes und haben mehr Zeit für die Durchführung wichtiger Änderungen.
- ✔ **Schnellere Erlernbarkeit:** Die Python-Entwickler wollten eine einfacher zu erlernende Programmiersprache mit weniger sonderbaren Regeln erschaffen. Im Grunde wollen Programmierer ja Anwendungen schreiben und nicht obskure und schwierige Sprachen lernen.

Python ist zwar beliebt, aber nicht immer auch unbedingt die beliebteste Sprache. Allerdings hat sie auf Seiten wie TIOBE (`https://www.tiobe.com/tiobe-index/`), einer Organisation, die (unter anderem) Nutzungsstatistiken verfolgt, aktuell (beim Schreiben dieses Buches) tatsächlich Rang 4 hinter Java, C und C++ erreicht. Wenn Sie Sprachen erlernen wollen, nur um einen Job zu ergattern, ist Python zwar eine gute Wahl, C/C++ oder Java sind dann aber noch bessere Alternativen. Visual Basic wäre ebenfalls eine gute Wahl, auch wenn es momentan weniger beliebt als Python ist. Wählen Sie eine Sprache aus, die Ihnen gefällt und Ihren Anwendungsentwicklungsanforderungen entspricht, aber wählen Sie sie auch danach aus, was Sie mit ihr erreichen wollen. Python war 2007 und 2010 die Sprache des Jahres und 2011 schon einmal die viertbeliebteste Sprache. Also ist sie gar nicht schlecht, wenn Sie einen Job suchen, aber nicht unbedingt die beste Wahl. Es wird Sie vielleicht überraschen, dass viele Universitäten in den USA Python verwenden, um Studenten das Programmieren beizubringen, und dass Python für diesen Zweck die beliebteste Sprache geworden ist. Wenn Sie mehr darüber erfahren wollen, lesen Sie am besten meinen (englischen) Blogeintrag `http://blog.johnmuellerbooks.com/2014/07/14/python-as-a-learning-tool`.

Wie Sie persönlich von Python profitieren können

Letztendlich können Sie jede Programmiersprache verwenden, um beliebige Anwendung zu schreiben. Setzen Sie aber bei einer Aufgabe auf die falsche Programmiersprache, wird Ihnen die Arbeit daran wahrscheinlich wenig Spaß machen und das Resultat wird schnell langsam und fehleranfällig, auch wenn Sie die Aufgabe sehr wahrscheinlich irgendwie lösen

können. Natürlich wollen die meisten von uns schreckliche und qualvolle Erfahrungen vermeiden. Also sollte man wissen, welche Anwendungen normalerweise mit Python geschrieben werden (auch wenn Python auch für weitere Zwecke eingesetzt wird):

- ✔ **Anwendungsbeispiele erstellen:** Entwickler müssen oft einen *Prototyp*, ein erstes Beispiel für eine Anwendung, erstellen, bevor sie Ressourcen für das Schreiben des eigentlichen Programms aufwenden. Da Produktivität bei Python besonders groß geschrieben wird, können Sie es gut verwenden, um schnell Prototypen für Anwendungen zu erstellen.

- ✔ **Browser-basierte Skripte programmieren:** Auch wenn JavaScript wahrscheinlich die beliebteste Sprache für Browser-basierte Anwendungsskripte ist, folgt ihr Python doch dicht auf dem zweiten Platz. Python bietet Funktionen an, die JavaScript nicht hat (für weitere Informationen finden Sie hier einen Vergleich: `https://blog.glyphobet.net/essay/2557`) und durch seine hohe Effizienz können Sie Browser-basierte Programme schneller erstellen (ein klarer Vorteil in der heutigen schnelllebigen Welt).

- ✔ **Mathematische, naturwissenschaftliche und ingenieurstechnische Anwendungen entwickeln:** Besonders interessant ist, dass Python Ihnen Zugriff auf ein paar wirklich coole Bibliotheken bietet, um mathematische, naturwissenschaftliche und ingenieurstechnische Programme einfacher zu schreiben. Die zwei beliebtesten Bibliotheken sind NumPy (`www.numpy.org`) und SciPy (`www.scipy.org`). Mit diesen Bibliotheken lassen sich derartige Anwendungen wesentlich schneller schreiben.

- ✔ **Mit XML arbeiten:** Die eXtensible Markup Language (XML) ist die Grundlage für die meisten Datenverwaltungslösungen vieler Internet- und Desktopanwendungen. Bei anderen Sprachen wurde XML eher dazugebastelt, während es bei Python einen Platz in der ersten Reihe hat. Wenn Sie beispielsweise mit Webdiensten arbeiten müssen, um Daten über das Internet (oder mit irgendeiner anderen XML-intensiven Anwendung) auszutauschen, ist Python eine gute Wahl.

- ✔ **Mit Datenbanken interagieren:** Die Industrie ist sehr von Datenbanken abhängig. Python ist nicht wirklich eine Abfragesprache wie die Structured Query Language (SQL) oder Language Integrated Query (LINQ), aber man kann mit ihr sehr gut mit Datenbanken arbeiten. Sie macht das Herstellen von Verbindungen zur Datenbank und die Arbeit mit den Daten ziemlich einfach.

- ✔ **Benutzeroberflächen erstellen:** Bei Python haben Sie nicht so wie bei anderen Sprachen wie C# einen integrierten Designer, in dem Sie Elemente aus einer Werkzeugpalette auf eine Benutzeroberfläche ziehen können. Python bietet jedoch eine große Auswahl an Frameworks für grafische Benutzeroberflächen (englisch: Graphical User Interfaces, GUIs) – Erweiterungen, die die Erstellung von Benutzeroberflächen viel einfacher machen (mehr dazu finden Sie unter `https://wiki.python.org/moin/GuiProgramming`). Manche dieser Frameworks liefern Designer mit, die die Erstellung von Benutzeroberflächen vereinfachen. Der Punkt ist, dass Python nicht nur eine Methode zur Erstellung einer Benutzeroberfläche anbietet – Sie können sich die Methode aussuchen, die Ihren Bedürfnissen am besten gerecht wird.

Anbieter	URL	Anwendungsart
Alice Educational Software – Carnegie Mellon University	`https://www.alice.org`	Lernsoftware
Fermilab	`https://www.fnal.gov`	Wissenschaftliche Anwendungen
Go.com	`http://go.com`	Browser-basierte Anwendungen
Google	`https://www.google.de`	Suchmaschine
Industrial Light & Magic	`http://www.ilm.com`	Alles, was so an Programmierung anfällt
Lawrence Livermore National Library	`https://www.llnl.gov`	Wissenschaftliche Anwendungen
National Space and Aeronautics Administration (NASA)	`http://www.nasa.gov`	Wissenschaftliche Anwendungen
New York Stock Exchange	`https://nyse.nyx.com`	Browser-basierte Anwendungen
Redhat	`http://www.redhat.com`	Installationswerkzeuge für Linux
Yahoo!	`http://www.yahoo.com`	Teile von Yahoo!-Mail
YouTube	`http://www.youtube.com`	Grafikengine
Zope – Digital Creations	`http://www.zope.org`	Publishing

Tabelle 1.1: Große Organisationen, die Python einsetzen

Welche Organisationen verwenden Python?

Python erfüllt die Aufgaben, für die es konzipiert wurde, ziemlich gut. Und das ist auch der Grund, warum viele große Firmen und Organisationen Python zur Programmierung von Anwendungen (allgemein zur Entwicklung) verwenden. Ein Grund mehr, Python zu nutzen, da diese Organisationen die Sprache unterstützen und Geld dafür ausgeben, sie zu verbessern. In Tabelle 1.1 finden Sie große Organisationen, die Python häufig einsetzen.

Dies sind nur ein paar der vielen Organisationen, die Python intensiv nutzen. Eine Liste mit weiteren Firmen finden Sie hier: `www.python.org/about/success`. Die Anzahl der Erfolgsgeschichten ist so riesig geworden, dass diese Liste wahrscheinlich auch nicht vollständig ist und man sie in Kategorien einteilen musste, damit sie übersichtlicher wird.

Nützliche Python-Programme finden

Vielleicht gibt es auf Ihrem Computer schon Anwendungen, die in Python geschrieben wurde, und Sie wissen es noch nicht einmal. Python wird in der Industrie in sehr vielen Anwendungsbereichen eingesetzt. Die Programme reichen von kleinen Konsolenanwendungen bis hin zu ausgewachsenen CAD/CAM-Plattformen. Manche Programme laufen auf mobilen Geräten, während andere auf großen Firmenservern ausgeführt werden. Zusammengefasst gibt es nichts, was Sie mit Python nicht tun können, aber es hilft, sich anzusehen, was andere so damit gemacht haben. Es gibt viele Seiten im Internet, wo Sie sich darüber informieren können, aber am besten schauen Sie hier nach: `https://wiki.python.org/moin/Applications`.

Als Python-Programmierer wollen Sie natürlich auch wissen, welche Entwicklungswerkzeuge es für Python gibt, mit denen Sie sich Ihre Arbeit erleichtern können. *Entwicklungswerkzeuge* bieten einen gewissen Grad an Automatisierung beim Schreiben von Programmen. Mit einem oder mehreren Entwicklungswerkzeugen müssen Sie weniger leisten, um funktionierende Anwendungen zu erstellen. Hier finden Sie eine umfangreiche Liste mit in Kategorien unterteilten Werkzeugen: `www.python.org/about/apps`.

Natürlich beschreibt auch dieses Kapitel ein paar Werkzeuge wie NumPy und SciPy (zwei naturwissenschaftliche Bibliotheken). Im weiteren Buch werden auch noch ein paar andere Tools erwähnt. Merken Sie sich am besten Ihre Lieblingswerkzeuge für später.

Python mit anderen Sprachen vergleichen

Es ist gefährlich, eine Sprache mit einer anderen zu vergleichen, da die Auswahl letztlich eine persönliche Geschmackssache und keine exakte Wissenschaft ist. Also, bevor ich von den vehementen Verteidigern anderer Sprachen attackiert werde, sollten Sie wissen, dass auch ich eine Vielzahl Sprachen verwende und es bei allen gewisse Überschneidungen gibt. Eine weltbeste Sprache gibt es nicht, allenfalls beste Sprachen für bestimmte Anwendungen. Denken Sie beim Lesen der folgenden Abschnitte daran. Diese bieten einen Überblicksvergleich zwischen Python und anderen Sprachen. (Vergleiche mit noch mehr Sprachen finden Sie unter `https://wiki.python.org/moin/LanguageComparisons`).

C#

Oft wird behauptet, dass Microsoft beim Entwickeln von C# einfach Java abgekupfert hat. Dennoch hat C# ein paar Vorteile gegenüber Java (und auch ein paar Nachteile). Die (unbestrittene) Hauptabsicht hinter C# ist, eine bessere C/C++-Sprache zu schaffen – eine, die leichter zu erlernen und zu verwenden ist. Vergleicht man Python mit C#, bietet Python folgende Vorteile:

- ✔ deutlich einfacher zu erlernen
- ✔ schlankerer (prägnanterer) Code
- ✔ vollständig quelloffen
- ✔ bessere Plattformunabhängigkeit
- ✔ Verwendung verschiedener Entwicklungsumgebungen leicht möglich
- ✔ einfacher mit Java und C/C++ zu erweitern
- ✔ bessere Unterstützung wissenschaftlicher Projekte und von Aufgaben aus den Ingenieurwissenschaften

Java

Über Jahre hinweg suchten Programmierer nach Sprachen, mit denen sie Programme nur einmal schreiben mussten, um sie dann überall ausführen zu können. Java soll auf allen Plattformen gut funktionieren. Um dieses Wunder zu ermöglichen, bedient es sich einiger Tricks, über die Sie weiter hinten in diesem Buch noch etwas erfahren werden. Momentan müssen Sie nur wissen, dass Java überall derart gut funktionierte, dass andere Sprachen ihm (mit unterschiedlichem Erfolg) nacheifern wollten. Trotzdem hat Python gegenüber Java ein paar wichtige Vorteile:

- ✔ deutlich leichter erlernbar
- ✔ schlankerer (prägnanterer) Der Code
- ✔ erweitertes Variablenmodell (Variablen sind temporäre Aufbewahrungsorte für Daten im Computerspeicher); basierend auf den Anwendungsbedürfnissen können Variablen während der Laufzeit verschiedene Arten von Daten enthalten (auch *Dynamische Typisierung* genannt)
- ✔ kürzere Entwicklungszeiten

Perl

Perl war ursprünglich ein Akronym für Practical Extraction and Report Language (was so viel bedeutet wie zweckmäßige Extraktions- und Berichtssprache). Heutzutage nennen die Leute sie einfach nur noch Perl – Punkt. Die Wurzeln von Perl erkennt man heute noch daran, dass es bei der Abfrage von Daten aus einer Datenbank und deren Präsentation in Berichtsform eine hervorragende Figur macht. Natürlich wurde Perl derart erweitert, dass es sehr viel mehr als nur das kann. Sie können mit Perl alle möglichen Anwendungen schreiben. (Ich habe es zum Beispiel zum Schreiben einer Webdienst-Anwendung verwendet.) Beim Vergleich mit Perl schneidet Python in folgenden Bereichen besser ab:

- ✔ leichter erlernbar
- ✔ leichter lesbar
- ✔ besserer Datenschutz
- ✔ bessere Java-Integration
- ✔ weniger plattformspezifische Eigenheiten

R

Datenwissenschaftler tun sich oft schwer, wenn sie sich zwischen R und Python entscheiden sollen, weil sich beide Sprachen sehr gut für statistische Analysen und die Art grafischer Darstellungen eignen, die von Datenwissenschaftlern benötigt werden, um Datenmuster verstehen zu können. Zudem sind beide Sprachen quelloffen und unterstützen eine

Vielzahl von Plattformen. R ist jedoch ein wenig stärker spezialisiert als Python und richtet sich tendenziell ein wenig stärker am akademischen Markt aus. Dadurch hat Python letztlich diese Vorteile gegenüber R:

- ✔ Betonung von Produktivität und Lesbarkeit des Codes
- ✔ auf den Einsatz im Unternehmensbereich zugeschnitten
- ✔ leichtere Fehlersuche
- ✔ verwendet konsistente Codiertechniken
- ✔ mehr Flexibilität
- ✔ leichter erlernbar