Ralph Steyer

Programmierung in Python

Ein kompakter Einstieg für die Praxis

2. Auflage

CODE INSIDE

Springer Vieweg

Programmierung in Python

Ralph Steyer

Programmierung in Python

Ein kompakter Einstieg für die Praxis

2. Auflage

Springer Vieweg

Ralph Steyer
Bodenheim, Deutschland

ISBN 978-3-658-44285-9 ISBN 978-3-658-44286-6 (eBook)
https://doi.org/10.1007/978-3-658-44286-6

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://portal.dnb.de abrufbar.

Planung/Lektorat: David Imgrund
Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.
Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Vorwort

„And now for something completely different.“ Was ist der Sinn des Lebens? Wie geht es Brian? Was machen Ritter mit den Kokosnüssen? Ist die Schwerkraft wirklich so wunderbar? Was macht man mit diesem lebensgefährlichen Witz? Und was haben diese Fragen mit dem Buch zu tun? Fragen Sie für die Antworten einfach einmal den Erfinder von Python – Guido van Rossum. Denn der war bei der Entwicklung der Sprache ein großer Fan der englischen Komikertruppe Monty Python und deren Show Monty Python's Flying Circus. Und nun fragen Sie sich immer noch, woher der Name „Python“ stammt? Ich überlasse die Antwort Ihrer Phantasie oder intensiven Recherchen im Internet. Oder Ihrer Aufmerksamkeit beim Lesen dieses Buchs, denn ich nehme mir die Freiheit und lasse immer wieder Zitate von Monty Python einfließen – denn auch ich bin Fan der Truppe.

Die Programmiersprache Python hat sich in den vergangenen Jahren zu einem Schwergewicht in der Programmiererszene entwickelt. Offensichtlich trifft das Konzept von Python den Nerv der Zeit. Oder genauer gesagt: Das Konzept stellt Ansätze, Lösungen und Vorgehensweisen für Probleme bereit, die andere Sprachen so nicht bieten und viele Leute interessant finden. Insbesondere auch die KI (künstliche Intelligenz) oder engl. AI („artificial intelligence“) entwickelt sich seit einiger Zeit als wesentliches Einsatzgebiet für Python.

Python gilt aktuell als eine der beliebtesten Einsteigersprachen überhaupt, denn Python wurde mit dem Ziel größter Einfachheit und Übersichtlichkeit entworfen. Zentrales Ziel bei der Entwicklung der Sprache war die Förderung eines gut lesbaren, knappen Programmierstils. In vielen Ländern hat Python an Universitäten bei Anfängerkursen in informatikbezogenen Studiengängen Java abgelöst, das über viele Jahre die Szene beherrscht hatte und im professionellen Umfeld immer noch das Maß aller Dinge darstellt. Wobei man erwähnen muss, dass viele neue Python-Programmierer dann später zusätzlich Java oder eine andere OO-Sprache wie C# lernen oder auch darauf umsteigen. Aber auch für diesen Weg legt Python mit dem Zwang zu einem strukturierten, klaren Programmierstil eine hervorragende Grundlage – wenn man dessen Freiheiten nicht missbraucht.

Umgekehrt lässt sich Python sehr schnell erfassen, wenn man bereits Erfahrung mit anderen, weitgehend beliebigen Programmiersprachen hat. Denn Python unterstützt sowohl die objektorientierte, die aspektorientierte, die strukturierte als auch die funktionale Programmierung. Das bedeutet, Python zwingt den Programmierer nicht zu einem einzigen Programmierstil, sondern erlaubt, das für die jeweilige Aufgabe am besten geeignete Paradigma zu wählen. Und damit können ebenso Erfahrungen aus anderen Programmierkonzepten mehr oder weniger direkt weitergenutzt werden. Dieser universell mögliche Zugang ist neben der Einfachheit vermutlich eines der Erfolgsrezepte von Python.

Nun noch kurz zu meiner Person. Ich habe vor vielen Jahren in Frankfurt/Main an der Goethe-Universität Mathematik studiert (Diplom) und danach anfangs einen recht typischen Werdegang für Mathematiker genommen: Ich bin bei einer großen Versicherung gelandet – aber schon da mit IT-Schwerpunkt. Zuerst habe ich einige Jahre als Programmierer mit Turbo Pascal und später mit C und C++ gearbeitet. Nach vier Jahren habe ich in die fachliche Konzeption für eine Großrechnerdatenbank unter MVS gewechselt. Die Erfahrung war für meinen Schritt in die Selbstständigkeit sehr motivationsfördernd, denn mir wurde klar, dass ich das nicht auf Dauer machen wollte. Seit 1996 verdiene ich daher meinen Lebensunterhalt als Freelancer, wobei ich fliegend zwischen der Arbeit als Fachautor, Fachjournalist, EDV-Dozent, Consultant und Programmierer wechsele. Daneben referiere ich gelegentlich auf Web-Kongressen, unterrichte an verschiedenen Akademien und Fachhochschulen oder nehme Videotrainings auf. Das macht aus meiner Sicht einen guten Mix aus, bewahrt vor beruflicher Langeweile und hält mich in der Praxis wie auch am Puls der Entwicklung. Insbesondere habe ich das Vergnügen wie auch die Last, mich permanent über neue Entwicklungen auf dem Laufenden zu halten, denn die Halbwertszeit von Computerwissen ist ziemlich kurz. Dementsprechend ist mein Job zwar anstrengend, aber vor allem immer wieder spannend. Wenn Sie weitere und detaillierte Informationen zu meiner Person erhalten wollen, finden Sie mich natürlich im Internet. Ich bin in diversen sozialen Netzwerken und natürlich auch mit einer eigenen Webseite im weltweiten Web unterwegs. Etwa hier:

http://www.rjs.de
http://blog.rjs.de

Doch lassen Sie uns also jetzt mit Python beginnen.

Herbst und Winter
2023 und 2024

Ralph Steyer

Inhaltsverzeichnis

Einleitung und Grundlagen – Bevor es richtig losgeht

1

1.1 Was behandeln wir in dem einleitenden Kapitel?

Bevor es mit Python richtig losgeht, sollen in diesem einleitenden Kapitel einige Dinge geklärt werden, die Ihnen die folgende Arbeit mit diesem Buch und der Programmiersprache im Allgemeinen erleichtern werden. Insbesondere sorgen wir an der Stelle dafür, dass Ihnen Python überhaupt zur Verfügung steht.

1.2 Das Ziel des Buchs

Dieses Buch ist zum Lernen von Python gedacht, entweder in Form des Selbststudiums oder als Begleitmaterial in entsprechenden Kursen. Zuerst einmal lernen Sie die elementaren Grundlagen, um überhaupt Programme mit Python erstellen wie auch pflegen zu können. Danach werden erweiterte Techniken wie umfassendere Anwendung der Syntax, objektorientierte Programmierung mit Python, komplexere Anwendungen mit sequenziellen Datenstrukturen, Umgang mit Modulen, Ausnahmebehandlung, Dateizugriffe, Datenbankzugriffe und die Erstellung grafischer Oberflächen behandelt. Dabei wird über sämtliche Themen hinweg Wert auf die grundsätzliche Anwendung der verschiedenen Techniken und einfache Beispiele gelegt und nicht auf Vollständigkeit aller möglichen Anweisungen, Befehle oder Parameter. Ebenso wird immer wieder darauf hingewiesen werden, dass man gewisse Freiheiten, die Ihnen Python bietet, nicht unbedingt ausnutzen sollte.

R. Steyer, *Programmierung in Python,* https://doi.org/10.1007/978-3-658-44286-6_1

▶ Wir erstellen im Laufe des Buchs immer wieder praktische Beispiele. Die Quellcodes des Buchs finden Sie nach Kapiteln und darin erstellten Projekten sortiert auf den Webseiten des Verlags bzw. in einem Repository auf GitHub. Die Namen der jeweilig aktuellen Dateien beziehungsweise Projekte werden als Hinweise oder direkt im Text vor den jeweiligen Beispielen angegeben und bei Bedarf wiederholt. Ich empfehle allerdings, dass Sie die Beispiele unbedingt allesamt von Hand selbst erstellen. Das ist für das Verständnis und das Lernen eindeutig besser als ein reines Kopieren oder nur Anschauen.

1.3 Was sollten Sie bereits können?

Der Kurs ist als Einsteigerkurs konzipiert, in dem die Sprache Python von Grund auf erarbeitet wird. Dabei wird der Tatsache Rechnung getragen, dass Python mittlerweile oft als erste Programmiersprache überhaupt gelernt wird. Um nicht die einfachsten Programmiergrundlagen zu ausführlich erläutern zu müssen, wird aber zumindest ein Verständnis der Idee von Programmierung vorausgesetzt, obwohl Sie noch nicht zwingend programmieren müssen. Ebenso sollten Sie mit einem Texteditor umgehen können. Kenntnisse der wichtigsten Befehle Ihres Betriebssystems (Windows, Linux oder macOS) zum Umgang mit Dateien und Verzeichnissen in der Konsole sind hilfreich. Umsteiger aus anderen Sprachen sind aber auch explizit als Zielgruppe des Buchs einkalkuliert, und entsprechende Hinweise auf andere Programmiersprachen werden immer wieder Bezüge herstellen – insbesondere zu Sprachen der Webprogrammierung (PHP, JavaScript, …), Java und .NET-Sprachen.

1.4 Was ist Python?

Python ist eine universelle, höhere Programmiersprache, die üblicherweise interpretiert wird. Es gibt also in der Laufzeitumgebung von Python einen Interpreter samt notwendiger weiterer Ressourcen.

Hintergrundinformation
Nun ist der Begriff des Interpreters gefallen, und die Konzepte und Fachausdrücke zur Übersetzung von Quellcode sollen nicht einfach vorausgesetzt werden. Wenn Sie Python-Programme erstellen, schreiben Sie den sogenannten **Quellcode** oder **Quelltext,** der der englischen Sprache angelehnt ist. Aber zur Ausführung muss dieser übersetzt werden, und zwar in ein Format, das ein Computer versteht, und das ist **Maschinencode**. Dazu gibt es verschiedene Möglichkeiten.

- Einmal gibt es den **Interpreter**. Dieser sorgt für eine Übersetzung in Maschinencode während der Programmausführung. Interpreter übersetzen den Quellcode eines Programms zeilenweise.
- Die Alternative ist der **Compiler**, der den kompletten Quelltext vor der Laufzeit übersetzt.

- In den vergangenen Jahren hat sich auch eine **zweistufige Übersetzung** etabliert – die Übersetzung mit Zwischencode. Das wird in Java oder dem .NET-Framework gemacht. Denn ein Nachteil bei kompilierten Programmen ist, dass diese Programme maschinenabhängig sind und somit nicht auf jeder Computerplattform, zum Beispiel Linux und Windows, ausgeführt werden können. In Java oder bei der .NET-Plattform wird Quellcode nicht zu einem ausführbaren Programm, sondern in einen Zwischencode kompiliert (1. Schritt). Dieser Code ist für alle Plattformen gleich und kann mithilfe des entsprechenden plattformspezifischen Interpreters (2. Schritt) auf der jeweiligen Plattform ausgeführt werden.

1.4.1 Das Ziel von Python

Python ist den meisten gängigen Programmiersprachen verwandt,[1] wurde aber mit dem Ziel größter Einfachheit und Übersichtlichkeit entworfen. Zentrales Ziel bei der Entwicklung der Sprache ist die Förderung eines gut lesbaren, knappen Programmierstils. So wird beispielsweise der Code nicht durch geschweifte Klammern (wie in fast allen C-basierten Sprachen), sondern durch zwingende Einrückungen strukturiert.[2] Zudem ist die gesamte Syntax reduziert und auf Übersichtlichkeit optimiert.

Wegen dieser klaren und überschaubaren Syntax gilt Python als einfach zu erlernen, zumal die Sprache mit relativ wenig Schlüsselwörtern auskommt. Es wird immer wieder zu hören sein. dass sich Python-basierte Skripte deutlich knapper formulieren lassen als in anderen Sprachen.

1.4.2 Was umfasst Python?

Es gibt einmal die Sprache Python, die aus den üblichen Schlüsselworten, Operatoren, eingebauten Funktionalitäten etc. sowie einer eigenständigen Syntax besteht. Python besitzt zudem eine umfangreiche Standardbibliothek (API – Application Programming Interface, wörtlich „Anwendungsprogrammierschnittstelle“), und zahlreiche Pakete im Python Package Index, bei deren Entwicklung ebenfalls großer Wert auf Überschaubarkeit, aber auch eine leichte Erweiterbarkeit gelegt wurde.

Python-Programme lassen sich deshalb auch in anderen Sprachen als Module einbetten. Umgekehrt lassen sich mit Python Module[3] und Plug-ins[4] für andere Programme

---

[1] Die meisten aktuellen Programmiersprachen gehen von der Syntax her auf C zurück.

[2] Was allerdings viele erfahrene Programmierer mit C-Background eher verstört, denn die saubere Notation von geschweiften Klammern ist sehr übersichtlich.

[3] Zusammenfassungen von Quellcodestrukturen. Modul steht allgemein für einen Teil eines größeren Systems.

[4] Plug-ins sind allgemeine kleine Programme oder Programmpakete, mit denen sich Software nach den eigenen Bedürfnissen anpassen und erweitern lässt. Sie integrieren sich dazu in das System, das sie erweitern oder anpassen sollen.

schreiben, die die entsprechende Unterstützung bieten. Dies ist zum Beispiel bei Blender, Cinema 4D, GIMP, Maya, OpenOffice beziehungsweise LibreOffice, PyMOL, SPSS, QGIS oder KiCad der Fall.

- Ganz wichtig ist, dass Python über Pakete (packages) organisiert wird und dazu ein integriertes Paketverwaltungssystem mitbringt. Genaugenommen ist das **pip** genannte Tool (früher pyinstall) das de facto und empfohlene Paketverwaltungsprogramm für Python-Pakete aus dem Python Package Index (PyPI), aus dem diverse Erweiterungen von Python bezogen werden können. PyPI ist der zentrale Paketpool und umfasst mittlerweile weit über 100.000 Pakete. Entwickler können nach einer Registrierung dort Module hochladen und so anderen Benutzern zur Verfügung stellen.

1.4.3 Die verschiedenen Python-Paradigma

Unter der Oberfläche ist Python streng objektorientiert. Aber die Art, wie man mit Python umgeht bzw. programmiert, ist flexibel. Python unterstützt sowohl die objektorientierte, die aspektorientierte, die strukturierte als auch die funktionale Programmierung. Das bedeutet, Python zwingt den Programmierer nicht zu einem einzigen Programmierstil, sondern erlaubt, das für die jeweilige Aufgabe am besten geeignete Paradigma zu wählen. Objektorientierte und strukturierte Programmierung werden vollständig, funktionale und aspektorientierte Programmierung werden zumindest durch einzelne Elemente der Sprache unterstützt. Ein zentrales Feature ist in Python die dynamische Typisierung samt dynamischer Speicherbereinigung. Damit kann man Python auch als reine Skriptsprache nutzen.

1.5 Was benötigen Sie zum Arbeiten mit dem Buch?

Python steht auf verschiedenen Plattformen und in verschiedenen Versionen zur Verfügung. Wir schauen uns erst einmal an, mit welchen Voraussetzungen Sie an das Buch gehen können.

1.5.1 Hardware und Betriebssystem

Python gibt es für Windows und macOS sowie Linux. Damit sind die drei Betriebssysteme genannt, mit denen wohl die meisten Leser arbeiten werden. Aber es gibt Python auch für viele weitere, teils ziemlich alte beziehungsweise exotische Betriebssysteme (unter anderem AS/400 [OS/400], BeOS, MorphOS, MS-DOS, OS/2, RISC OS, Series 60, Solaris oder HP-UX), wobei Sie beachten sollten, dass für ältere

Betriebssysteme die neuesten Python-Versionen oft nicht mehr zur Verfügung stehen. Ab der Version Python 3.5 wird etwa Windows XP nicht mehr unterstützt. Aber auch viele vorinstallierte Versionen von Python – wenn es denn solche gibt – sind nicht auf dem aktuellen Stand.

Letztendlich sind aber die Voraussetzungen für die Arbeit mit Python ziemlich niedrig. Sie benötigen für die Arbeit mit dem Buch und Python im Grunde nur einen Computer mit einem halbwegs modernen Betriebssystem. Als Basis für die Unterlagen wird explizit ein PC vorausgesetzt.

1.5.2 Die Python-Version

Die **Referenzbetriebssysteme** in diesem Buch werden Windows 10 und 11 sowie eine aktuelle Linux-Distribution sein.[5] Aber auch macOS wird beachtet. Überwiegend wird in den Unterlagen mit Windows 10/11 gearbeitet, aber die Ausführungen und Beispiele sind auf andere Systeme übertragbar oder nicht von der Plattform abhängig. Sollten Spezifika interessant sein, wird das hervorgehoben.

1.5.2.1 Die Evolution von Python

- Python wurde Anfang der 1990er-Jahre von Guido van Rossum am Centrum Wiskunde & Informatica in Amsterdam als Nachfolger für die Sprache ABC entwickelt.
- Ursprüngliche Zielplattform war das verteilte Betriebssystem Amoeba.
- Die erste Vollversion erschien im Januar 1994.
- Python 2.0 erschien Oktober 2000.
- Python 3.0 (auch Python 3000) erschien im Dezember 2008, und die Serie 3 ist 2024 immer noch aktuell.

1.5.3 Python laden und installieren

Zum Erstellen von Python-Programmen braucht man im Grunde nur einen Klartexteditor. Aber um Python-Programme oder -Skripte auszuführen, braucht man mehr – eine Umgebung für Python. Deshalb muss für die weiteren Schritte mit dem Buch Python auf Ihrem Rechner installiert sein, oder Sie sollten das zu Beginn machen. Dabei müssen Sie beachten, dass es aktuell zwei „Schienen“ bei den Versionen von Python gibt. Zwar ist Python 3.0 wie erwähnt bereits im Dezember 2008 erschienen. Da Python 3.0 jedoch teilweise inkompatibel zu früheren Versionen ist, wurde Python 2.7 lange Zeit parallel zu Python 3.x weiter durch Updates unterstützt, wobei die Weitentwicklung der 2.x-Schiene

[5] Konkret wird an einigen Stellen Mint Linux 21 verwendet.

mittlerweile ausgelaufen ist. Dennoch ist diese Version nicht unwichtig, denn Python in der Version 2.x wird immer noch als Standard bei verschiedenen Betriebssystemen bereitgestellt.

- Die Besonderheiten der 2er-Versionen von Python werden im Buch nicht mehr berücksichtigt.

Die **Referenzversion** von Python für dieses Buch ist die Version 3.12.x, wobei unsere Ausführungen im Grunde für alle Versionen 3.x gelten. Die Unterschiede sind für dieses Buch in der Regel nicht wirklich relevant, außer für einige neuere Features in Python, die jeweils mit spezifischen Versionen von 3.x erst verfügbar sind. Bei den betroffenen Features wird im Buch explizit angegeben, ab welcher Version diese genutzt werden können. Sie sollten auch beachten, dass es für unsere Referenzbetriebssysteme kleinere Abweichungen in der verfügbaren Version von Python geben kann. Oder genauer: Es kann sich zeitlich etwas unterscheiden, wann etwa eine Version für Windows, Linux oder macOS veröffentlicht wird.

1.5.3.1 Das zentrale Webportal und die Python Software Foundation (PSF)

Wenn Sie erstmals Kontakt zu Python aufnehmen wollen, gehen Sie am besten zuerst auf die Webseite https://www.python.org/ (Abb. 1.1). Das ist das zentrale Webportal von Python. Hier finden Sie die wichtigsten Informationen zu Python und auch alle notwendigen Ressourcen.

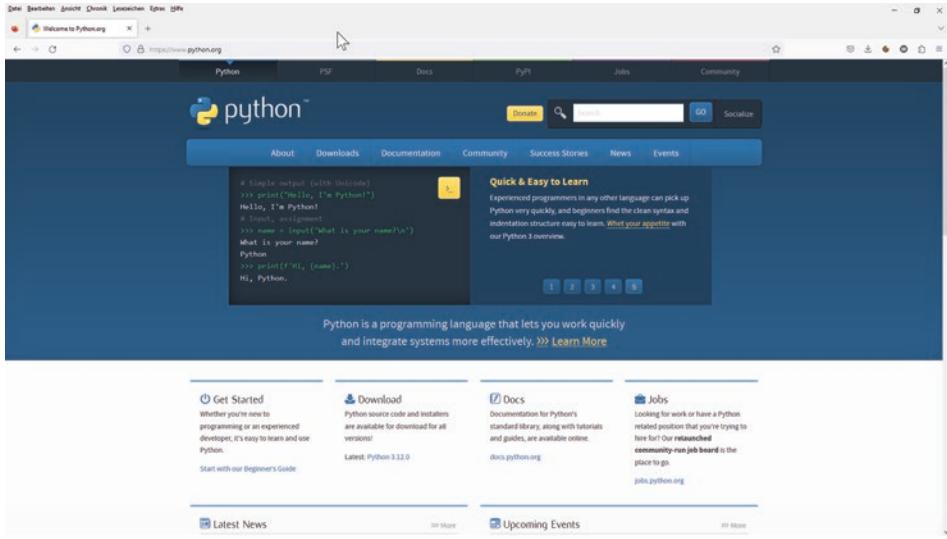

Abb. 1.1 Das „offizielle“ Webportal zu Python

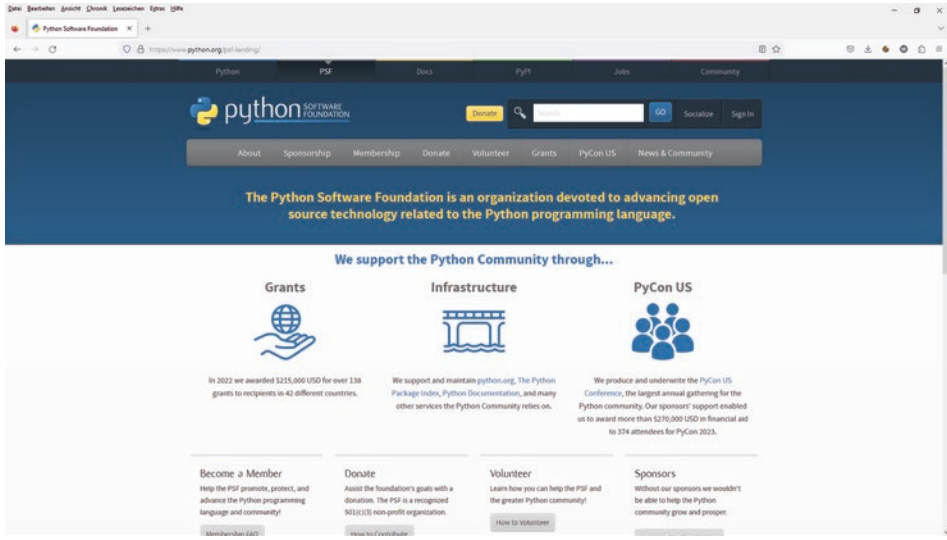

Abb. 1.2 Die Webseite der PSF

Von da gelangen Sie über einen Link oben in der Navigation oder direkt über den Link https://www.python.org/psf/ zur Python Software Foundation (PSF, Abb. 1.2).

PSF ist eine Non-Profit-Organisation, die hinter dem Open-Source-Projekt der Programmiersprache steht. Mitglieder der PSF sind sowohl relevante Einzelpersonen aus dem Python-Umfeld als auch Firmen wie Google, Microsoft, Redhat und Canonical, die als Sponsoren agieren. Die PSF ist Herausgeber und Rechteinhaber der Python-Software-Foundation-Lizenz (https://docs.python.org/3/license.html Abb. 1.3) und besitzt die Markenrechte an Python. Die Python-Software-Foundation-Lizenz ist ähnlich der BSD-Lizenz und kompatibel mit der GNU General Public License.

1.5.3.2 Die Dokumentation

Im Python-Webportal finden Sie unter dem Menüpunkt Documentation und dort Docs zahlreiche Informationen inklusive einer kompletten Dokumentation, FAQs und verschiedenen Tutorials (Abb. 1.4).

Von da aus kann man diese Materialen herunterladen, aber das ist im Grunde gar nicht notwendig. Zum einen ist man ja sowieso meist online, und zudem ist die Dokumentation bereits im Installationspaket von Python enthalten beziehungsweise wird mit installiert, wenn Sie das bei der Installation auswählen.

1.5.3.3 Der Download von Python

Bereits im Dokumentationsbereich steht Ihnen Python zum Download bereit. Aber es gibt auch auf der Einstiegsseite des Webportals einen eigenen Menüpunkt Downloads. Darüber erhalten Sie die aktuellen, aber auch bei Bedarf frühere Versionen für verschiedene Betriebssysteme (Abb. 1.5).

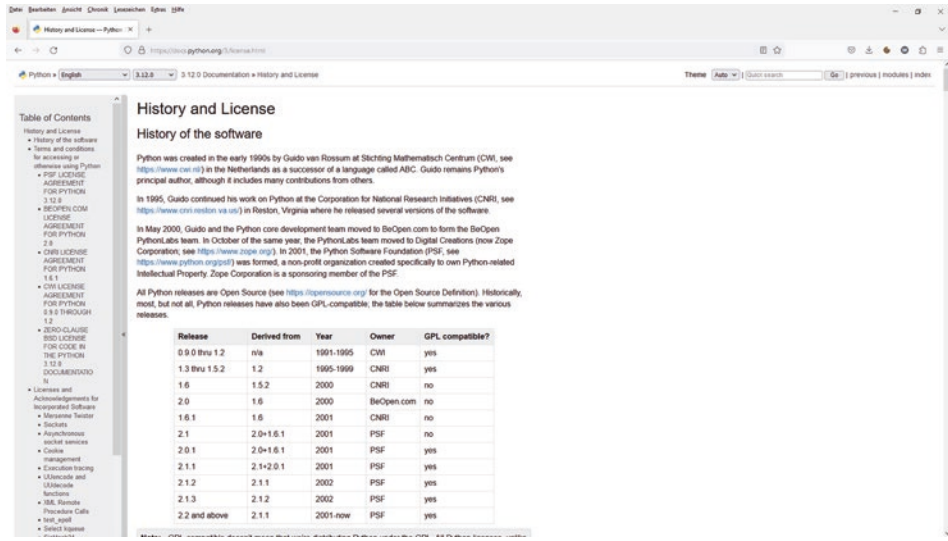

Abb. 1.3 Details zur Python-Lizenz

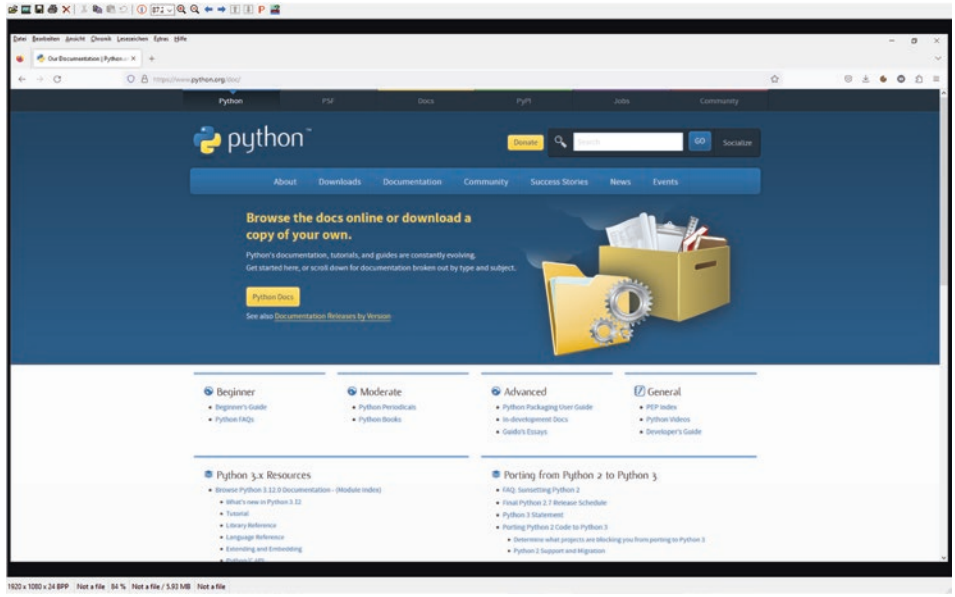

Abb. 1.4 Informationen, Quellen und Tutorials

- Laden Sie sich am besten immer die neueste Version von Python für Ihr Betriebssystem herunter.

1.5.3.4 Die konkrete Installation

Wir gehen nun die Installation für Windows, Linux und macOS einzeln durch, und Sie werden sehen, dass diese in der Regel geradezu trivial ist, wenn es einen Installer gibt. Unter Linux und teils auch macOS kann es aber sein, dass Sie Python in einem Terminal installieren müssen bzw. können, wenn Sie die aktuelle Version haben wollen.

1.5.3.4.1 Die Installation unter Windows

Unser wichtigstes Referenzsystem im Buch ist Windows, und damit wollen wir beginnen. Unter Windows starten Sie einfach den Installer, den Sie auf Ihren PC geladen haben. Dieser startet wiederum mit einem Auswahldialog mit allen wichtigen Informationen und Vorgabeeinstellungen (Abb. 1.6).

Wenn Sie in dem Dialog bereits den Installationsbefehl geben (INSTALL NOW), läuft die Installation ohne weitere Interaktion mit Ihnen durch – abgesehen von eventuellen Rückfragen von Windows, zum Beispiel, ob Sie dem Installationsprogramm wirklich erlauben wollen, Dinge auf dem PC zu verändern. Auf dem Rechner hat der Python-Installer danach alle Dateien im voreingestellten Verzeichnis installiert, die man für die Ausführung von Python-Programmen benötigt.

Nun gibt es in dem Dialog u. a. drei optionale Einstellungsmöglichkeiten, die Sie auswählen sollten.

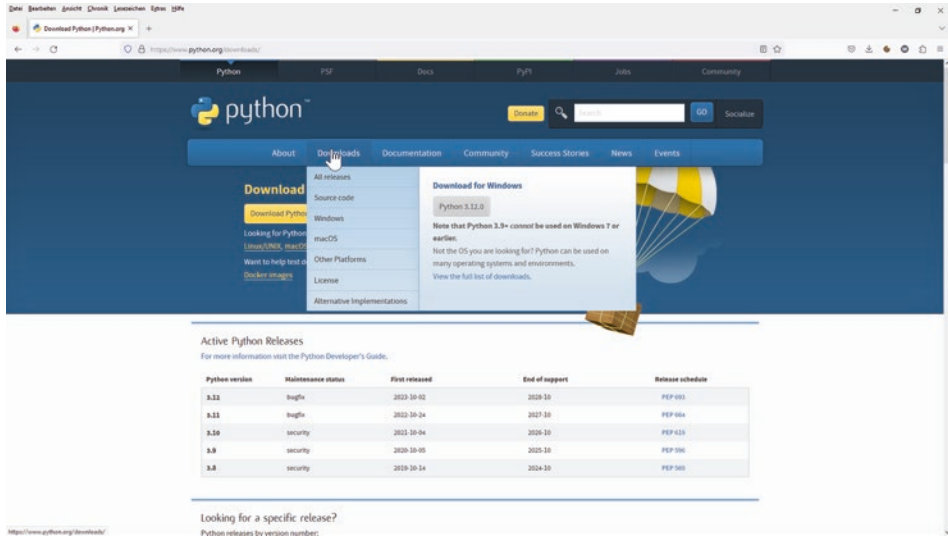

Abb. 1.5 Python laden

Abb. 1.6 Start des Setupassistenten unter Windows 11

1. Die erste Option stellt allen Benutzern auf dem Rechner den sogenannten Launcher für Python zur Verfügung. Das bedeutet, dass diese Anwender dann Python-Dateien direkt ausführen können. Oder anders ausgedrückt: Die Python-Dateierweiterung wird mit dem Python-Interpreter verknüpft (Abb. 1.7).
2. Besonders zu empfehlen ist das Hinzufügen von Python zur *Path*-Angabe – dem Suchpfad von Windows (Abb. 1.6). Dazu klicken Sie in dem Dialog die entsprechende Option an – bei der im Buch verwendeten Version von Python nennt die sich ADD PYTHON.EXE TO PATH. Wenn Sie die entsprechende Option ausgewählt haben, wird das Installationsverzeichnis in den Suchpfad von Windows aufgenommen, und Sie können Python in der Konsole von jedem Verzeichnis aus aufrufen. Das ist zwar nicht zwingend notwendig, aber sehr bequem.
3. Achten Sie darauf, dass Sie pip installieren (Abb. 1.7), damit Sie Pakete bequem installieren und verwalten können. Das Paketverwaltungssystem wird mittlerweile auch über reines Python hinaus verwendet.

▶ Sie können die Installationsverzeichnisse unter Windows selbstverständlich nachträglich dem Suchpfad hinzufügen. Das erfolgt in den Einstellungen zum System unter ERWEITERTE UMGEBUNGS-VARIABLEN. Aber wenn das vom Setup-Assistent schon geleistet wird, ist das natürlich angenehm. Wenn Sie nicht mehr wissen, wo der Installationsordner von Python zu finden ist, können Sie den Befehl *where* verwenden. Geben Sie einfach in der Eingabeaufforderung *where python* an.

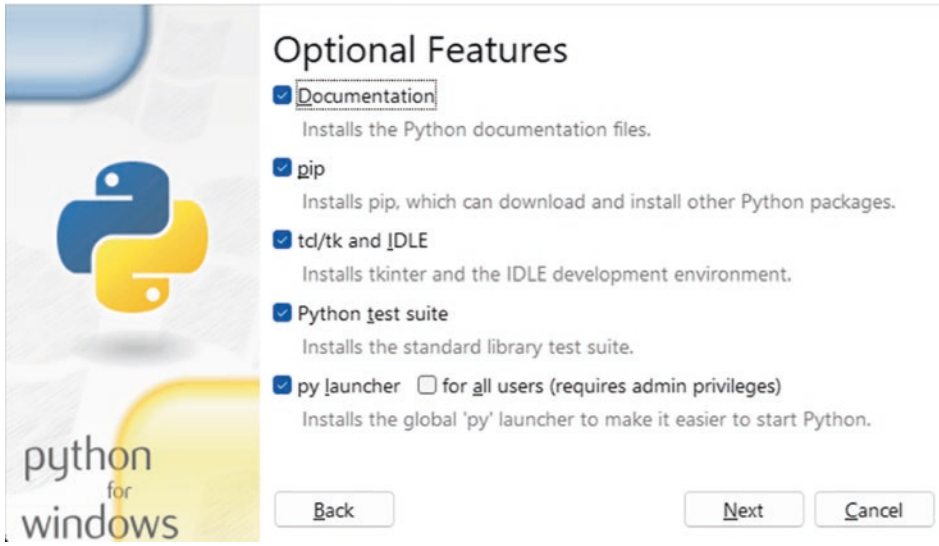

Abb. 1.7 Optionale Features

Nach der Installation gibt es in Windows insbesondere zwei Programme, die von zentraler Bedeutung sind für Python:

- Die **Python-Kommandozeile** – auch Python-Interpreter[6] oder Python-Shell genannt,
- die Python-GUI, die man mit **IDLE** (Python's Integrated Development Environment) abkürzt.

Auf die Details dazu gehen wir später ein.

Wenn Sie wollen, können Sie vom Einstiegsdialog aus die Arbeit des Assistenten auch individuell anpassen.

Zum einen können Sie optionale Features festlegen (Abb. 1.7). Diese Einstellungen können aber meist in der Voreinstellung bleiben.

Aber sehr interessant sind die erweiterten Optionen im darauffolgenden Anpassungsdialog (Abb. 1.8). Denn insbesondere die Änderung des Installationsorts von Python kann wichtig sein. Ich verwende etwa bei meinem Windows-System aktuell eine Workstation mit einer recht kleinen SSD, aber einer sehr großen HDD. Die Daten und nicht wirklich performancehungrige Programme lagere ich deshalb grundsätzlich auf die größere HDD aus. Aber standardmäßig werden Sie Python meist auf Laufwerk C: installieren.

[6] Was eigentlich etwas ungenau ist, denn der Interpreter an sich ist keine Konsole in dem Sinn. Aber man benutzt den Begriff dennoch im Python-Umfeld.

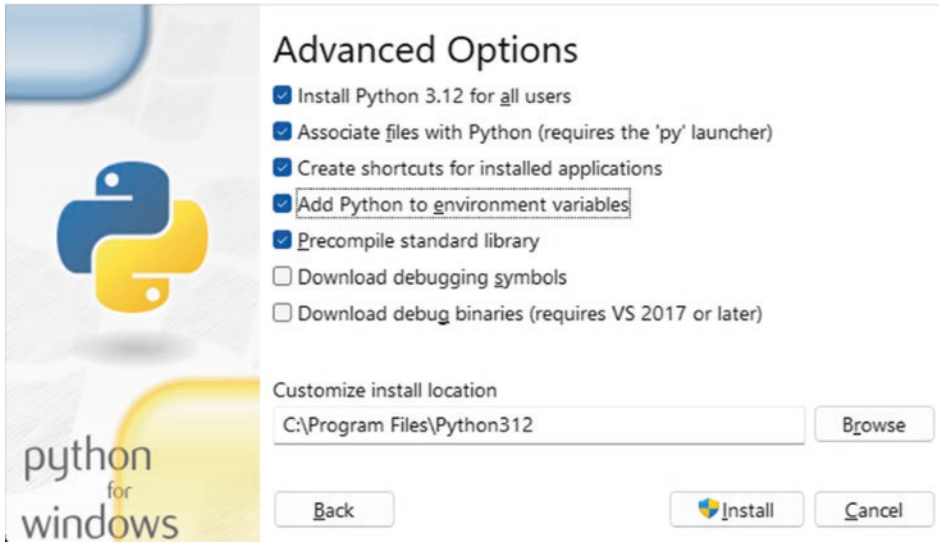

Abb. 1.8 Erweiterte Optionen

Wenn die Installation ohne Probleme beendet ist, erhalten Sie eine Erfolgsmeldung (Abb. 1.9).

Wenn ein Fehler passiert ist (was selbstverständlich weder zu hoffen noch zu erwarten ist), werden Sie natürlich auch über die Art der Probleme informiert.

1.5.3.4.2 Installieren auf einem Linux-System

Python gibt es wie gesagt auch für Linux. Dabei muss man einmal einschränken, dass es verschiedene Formate für die unterschiedlichen Linux-Distributionen gibt. Das macht die Sache etwas unübersichtlich. Aber dafür haben Sie auf der anderen Seite die Flexibilität und können Python sogar individuell angepasst aus Quellcodes für Ihr System erstellen.

Im Prinzip ist das meist gar nicht notwendig. Denn viele Linux-Distributionen installieren Python bereits automatisch mit. Oder sie stellen Python über die distributionseigene Anwendungsverwaltung zur Verfügung, womit die Installation genauso trivial wie unter Windows abläuft (Abb. 1.10).

Allerdings muss man beachten, dass in der Regel nicht die aktuelle Version von Python in den Installationspaketen beziehungsweise der standardmäßig installierten Version bereitsteht. Sogar halbwegs aktuelle Linux-Versionen (Abb. 1.11) oder auch macOS (Abb. 1.12) stellen noch Python in der Version 2.7.x bereit, und die Besonderheiten der 2er-Versionen werden wir wie gesagt explizit nicht mehr beachten. Wie erwähnt, liegt Python zum Zeitpunkt der Bucherstellung in der Version 3.12.x vor, und diese sollten Sie auch unter Linux zur Verfügung haben.

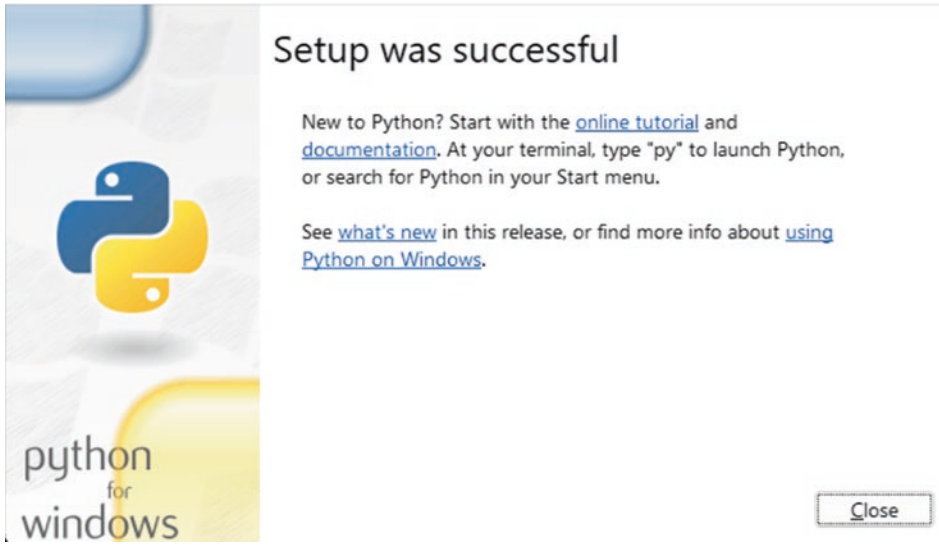

Abb. 1.9 Python wurde installiert

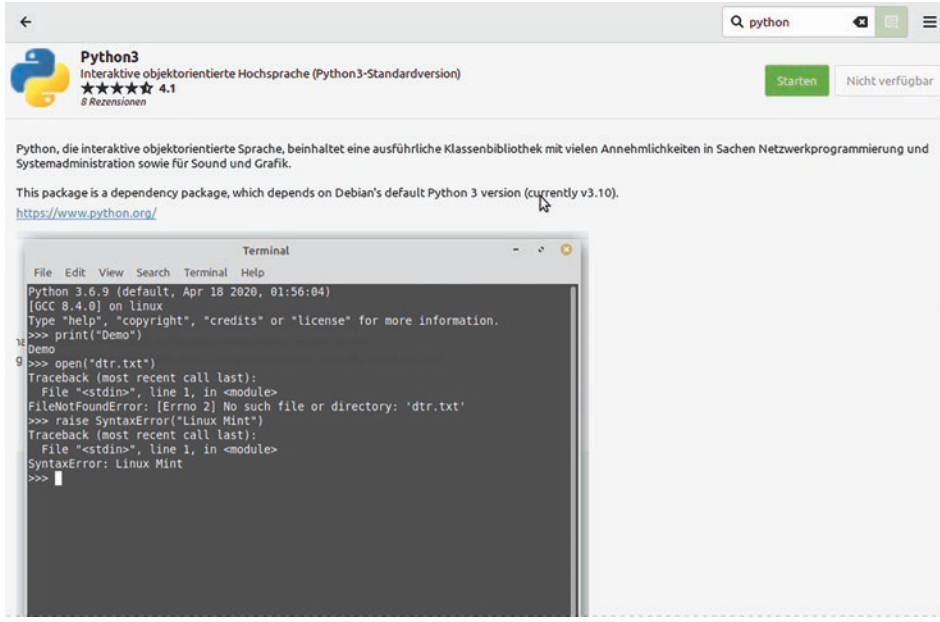

Abb. 1.10 Linux-Distributionen bringen Python eigentlich immer mit

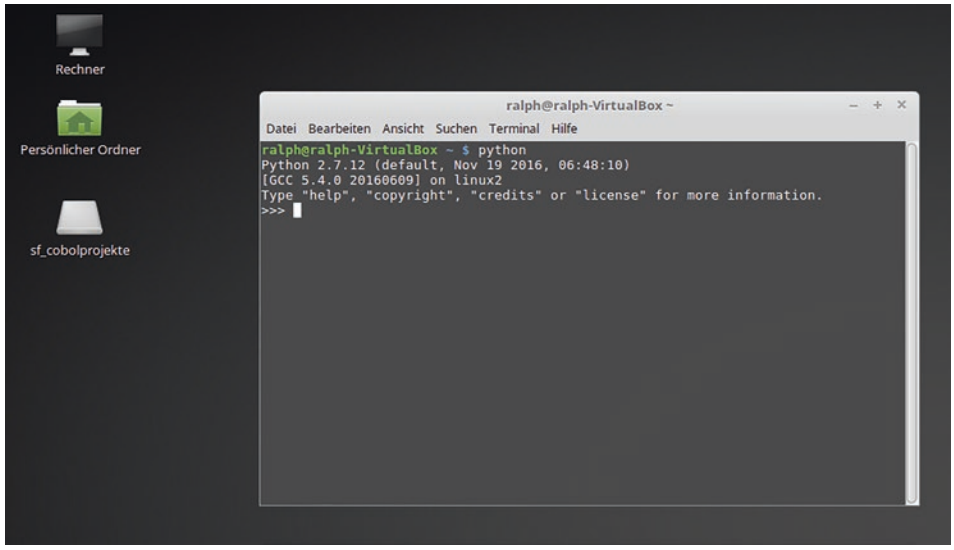

Abb. 1.11 Bei einigen halbwegs aktuellen Linux-Versionen ist standardmäßig noch die Version 2.x vorinstalliert

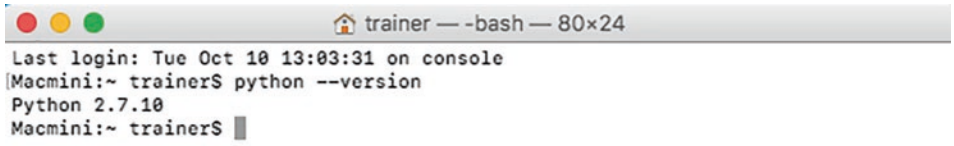

Abb. 1.12 Auch bei etwas älteren, aber immer noch verwendeten macOS findet man manchmal noch die Version 2.7

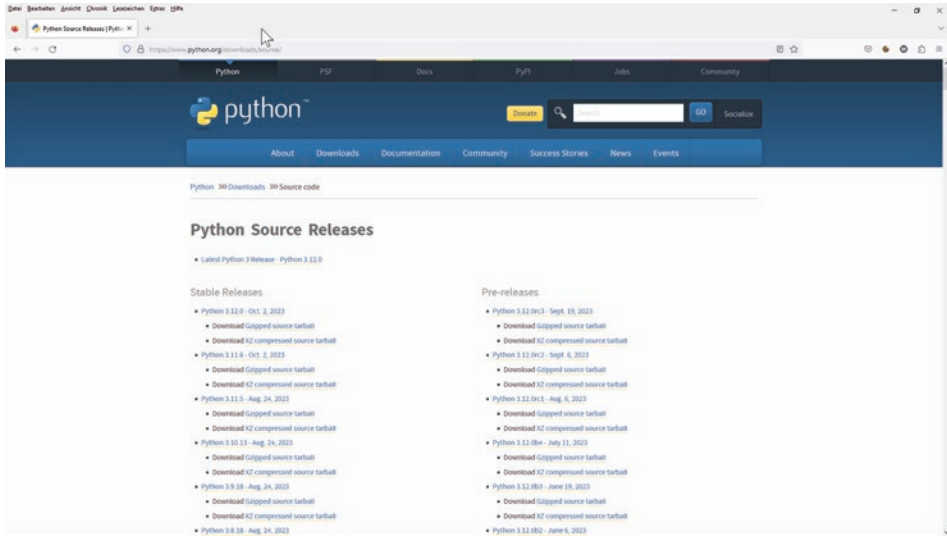

Abb. 1.13 Python für Linux laden

Falls Sie also die neueste Version oder zumindest eine Version der 3er-Schiene von Python nicht über die Anwendungsverwaltung Ihrer Linux-Distribution erhalten, können Sie eine individuelle Installationsdatei (wenn verfügbar) oder auch die Quellcodes laden und dann auf Ihrem Rechner konfigurieren, übersetzen und installieren (Abb. 1.13). Die letztgenannte Option ist meist besonders vielversprechend, um die aktuellste Version von Python zu erhalten.

Wie die dann aber konkret installiert wird, kann sich unterscheiden, und es muss für Details zu einer bestimmten Distribution auf die Dokumentation verwiesen werden – gerade im Fall von Problemen. Aber normalerweise erhalten Sie auf jeden Fall die Quelltextdateien von Python, die Sie auf die übliche Art unter Linux konfigurieren und installieren können. Die Ressourcen liegen dabei als Archiv vor, das Sie erst einmal extrahieren müssen (Abb. 1.14).

Danach öffnen Sie ein Terminal und gehen im allgemeinen Fall wie folgt vor (Abb. 1.15):

1. Wechseln Sie in das Verzeichnis, das bei der Extraktion der Python-Ressourcen angelegt wurde, etwa so:

Beispiel

cd Python-3.12.0 ◀

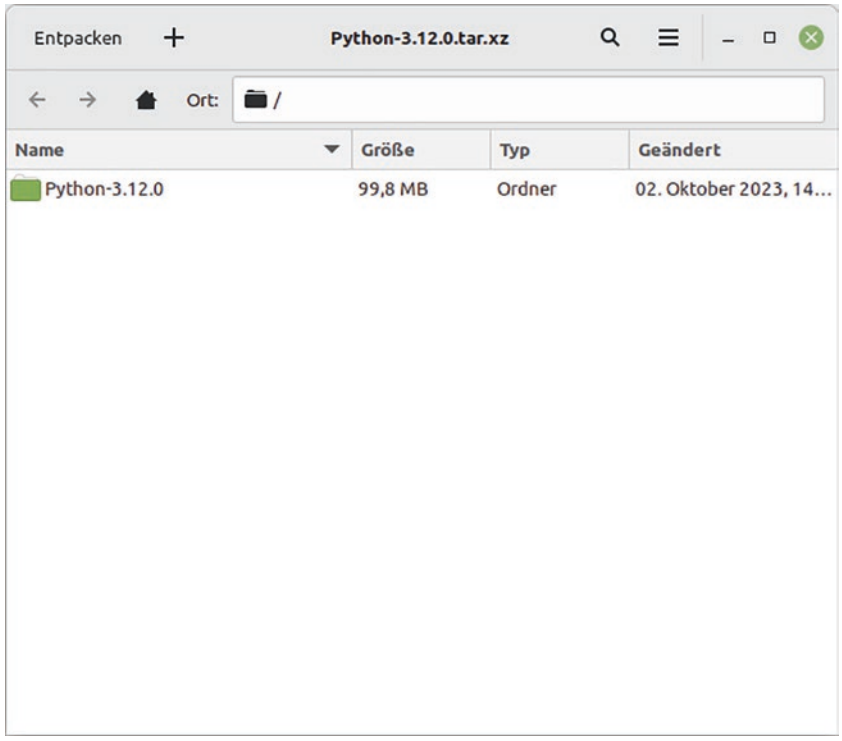

Abb. 1.14 Das Archiv muss extrahiert werden

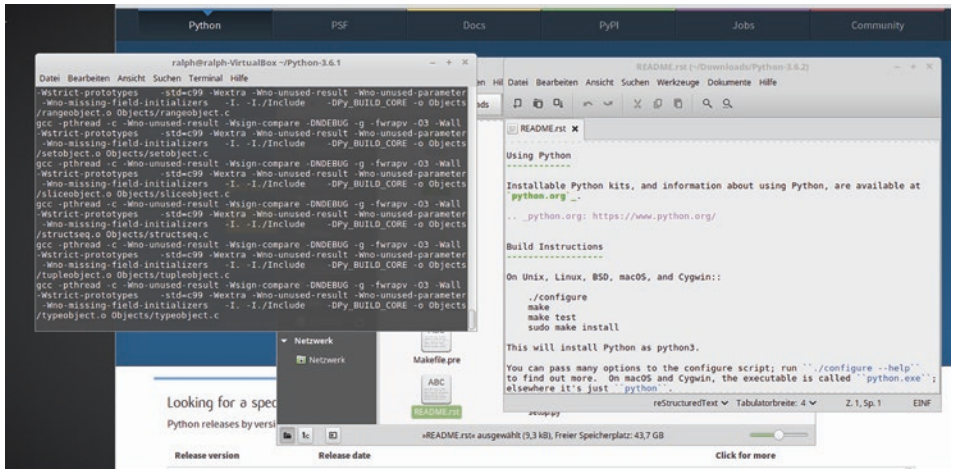

Abb. 1.15 Die Installation läuft

2. Konfigurieren Sie die Ressourcen für Ihren Rechner:

Beispiel

./configure ◀

3. Übersetzen Sie den Quellcode und erstellen Sie eine Make-Datei:

Beispiel

make ◀

4. Testen Sie das Make-File

Beispiel

make test ◀

5. Wenn der Test keine Fehler ergab, erfolgt die konkrete Installation. Das muss man in der Regel allerdings als root erledigen, also so, sofern Sie nicht schon in einem root-Terminal arbeiten oder sich vorher zum root gemacht hatten (dann kann *sudo* entfallen):

Beispiel

sudo make install ◀

Nach der erfolgreichen Installation sollte die neue Python-Version verfügbar sein (Abb. 1.16). Das können Sie etwa testen, indem Sie im Installationsverzeichnis von Python Folgendes eingeben:

Beispiel

./python ◀

Zu Beginn der nun aktivierten Python-Kommandozeile erkennen Sie die Python-Version (Abb. 1.16).

▶ Die Python-Kommandozeile schließen mit der Eingabe *exit()*. Beachten Sie die Klammern.

Auch unter Linux gilt, dass Python für eine bequemere Anwendung im Suchpfad eingetragen sein sollte. Wenn Python bei der Distribution schon installiert wurde oder Sie die Anwendungsverwaltung der Distribution ver-