Vorwort

Microsoft unterhält ein Feedback-Forum für Excel auf UserVoice[1], in dem jeder die Möglichkeit hat, neue Ideen einzureichen, über die andere abstimmen können. Die meistgenannte Funktionsanforderung, »Python als Excel-Skriptsprache«, hat ungefähr doppelt so viele Stimmen wie die zweithäufigste. Obwohl seit der Aufnahme der Idee im Jahr 2015 nicht wirklich etwas passiert ist, witterten Excel-Benutzer Ende 2020 Morgenluft, als Guido van Rossum, der Schöpfer von Python, twitterte (https://oreil.ly/N1_7N), dass sein »Ruhestand langweilig« sei und er zu Microsoft wechseln werde. Ob sein Wechsel einen Einfluss auf das Zusammenspiel von Excel und Python hat, weiß ich nicht. Aber ich weiß, was diese Kombination so überzeugend macht und wie Sie Excel und Python gemeinsam nutzen können – heute. Und das ist, kurz gesagt, das, worum es in diesem Buch geht.

Die treibende Kraft hinter der Geschichte von *Python für Excel* ist die Tatsache, dass wir in einer Welt voller Daten leben. Heutzutage sind riesige Datensätze für jeden und über alles verfügbar. Oftmals sind diese Datensätze so groß, dass sie nicht mehr in ein Tabellenblatt passen. Vor einigen Jahren hätte man dies vielleicht noch als *Big Data* bezeichnet, aber heutzutage ist ein Datensatz mit ein paar Millionen Zeilen wirklich nichts Besonderes mehr. Excel hat sich weiterentwickelt, um mit diesem Trend fertigzuwerden: Eingeführt wurde *Power Query*, um solche Datensätze zu laden und zu bereinigen, die nicht mehr in ein Tabellenblatt passen, und das Add-in *Power Pivot*, um Datenanalysen auf diesen Datensätzen durchzuführen und die Ergebnisse zu präsentieren. Power Query basiert auf der Formelsprache *Power Query M* (kurz M), während Power Pivot Formeln mithilfe von *Data Analysis Expressions* (DAX) definiert. Und wenn Sie einige Dinge in Ihrer Excel-Datei automatisieren möchten, verwenden Sie VBA (*Visual Basic for Applications*), die in Excel integrierte Automatisierungssprache. Für etwas ziemlich Einfaches haben Sie es letztlich mit VBA, M und DAX zu tun. Ein Problem dabei ist, dass Ihnen alle diese Sprachen nur in der Microsoft-Welt weiterhelfen – vor allem auf Excel und Power BI trifft dies zu. (Power BI werde ich kurz in Kapitel 1 vorstellen.)

1 *https://web.archive.org/web/20201209042635/https://excel.uservoice.com/forums/304921-excel-for-windows-desktop-application/filters/top*

Auf der anderen Seite ist Python eine Allzweckprogrammiersprache, die bei Analysten und Data Scientists zu einer der beliebtesten Sprache avanciert ist. Wenn Sie Python mit Excel verwenden, steht Ihnen eine Programmiersprache zur Verfügung, die für alle Aspekte der Geschichte geeignet ist, sei es, um Excel zu automatisieren, auf Datensätze zuzugreifen und sie vorzubereiten oder um Aufgaben der Datenanalyse und Visualisierung wahrzunehmen. Vor allem aber können Sie Ihre Python-Kenntnisse außerhalb von Excel wiederverwenden: Wenn Sie Ihre Rechenleistung erhöhen müssen, können Sie Ihr quantitatives Modell, Ihre Simulation oder Ihre Anwendung für maschinelles Lernen einfach in die Cloud verlagern, wo praktisch unbegrenzte Rechenressourcen auf Sie warten.

Warum ich dieses Buch geschrieben habe

Durch meine Arbeit an *xlwings*, dem Excel-Automatisierungspaket, das Sie in Teil IV dieses Buchs kennenlernen werden, stehe ich in engem Kontakt mit vielen Anwendern, die Python für Excel einsetzen – sei es über den *Issue Tracker* (*https://oreil.ly/ZJQkB*) auf GitHub, aufgrund einer Frage auf *StackOverflow* (*https://stackoverflow.com/*) oder bei einer Veranstaltung wie einem Treffen oder einer Konferenz.

Regelmäßig werde ich gebeten, Quellen für die ersten Schritte mit Python zu empfehlen. Obwohl es sicherlich keinen Mangel an Python-Einführungen gibt, sind sie oft entweder zu allgemein (ohne irgendwas über Datenanalyse) oder zu spezifisch (vollständig wissenschaftlich). Excel-Benutzer befinden sich jedoch eher in der Mitte: Sie arbeiten sicherlich mit Daten, aber eine vollständige wissenschaftliche Einführung mag ihnen zu technisch erscheinen. Außerdem haben sie oftmals spezielle Anforderungen und Fragen, die in den vorhandenen Materialien nicht beantwortet werden. Einige dieser Fragen lauten:

- Welches Python-Excel-Paket brauche ich für welche Aufgabe?
- Wie verschiebe ich meine Power-Query-Datenbankverbindung zu Python?
- Was ist in Python äquivalent zu den Features AutoFilter und Pivottabelle von Excel?

Ich habe dieses Buch geschrieben, damit Sie ohne vorherige Python-Kenntnisse in der Lage sind, Ihre Excel-orientierten Aufgaben zu automatisieren und die Python-Tools für Datenanalyse und wissenschaftliche Berechnungen ohne Umwege in Excel zu nutzen.

Für wen dieses Buch gedacht ist

Wenn Sie als fortgeschrittener Excel-Benutzer die Grenzen von Excel mit einer modernen Programmiersprache überwinden möchten, ist dieses Buch genau das richtige für Sie. Normalerweise bedeutet dies, dass Sie jeden Monat Stunden damit zubringen, große Datenmengen herunterzuladen, zu bereinigen, zu kopieren und in

unternehmenskritische Tabellen einzufügen. Es gibt zweifellos verschiedene Möglichkeiten, die Grenzen von Excel zu überwinden, doch dieses Buch konzentriert sich darauf, wie Sie für diese Aufgaben Python verwenden.

Es ist hilfreich, wenn Sie ein grundlegendes Verständnis von Programmierung mitbringen, denn wenn Sie schon einmal eine Funktion oder eine `for`-Schleife geschrieben haben (egal in welcher Programmiersprache), verfügen Sie schon über eine Vorstellung davon, was eine ganze Zahl oder ein String ist. Von diesem Buch können Sie möglicherweise auch profitieren, wenn Sie es gewohnt sind, komplexe Zellformeln zu schreiben, oder Erfahrung darin besitzen, aufgezeichnete VBA-Makros zu optimieren. Nicht erwartet werden Python-spezifische Kenntnisse, da Sie zu allen Tools, die wir hier verwenden, und zu Python selbst Einführungen bekommen.

Ein erfahrener VBA-Entwickler findet in diesem Buch regelmäßig Vergleiche zwischen Python und VBA, mit denen es möglich ist, die üblichen Probleme zu umschiffen und sofort loszulegen.

Dieses Buch kann auch hilfreich sein, wenn Sie als Python-Entwickler die verschiedenen Möglichkeiten kennenlernen müssen, die Python hat, um mit der Excel-Anwendung und den Excel-Dateien umgehen zu können, damit Sie je nach den Anforderungen Ihrer Geschäftskunden das richtige Paket auswählen können.

Wie dieses Buch aufgebaut ist

In diesem Buch zeige ich Ihnen alle Aspekte der Python-für-Excel-Geschichte, gegliedert in vier Teile:

Teil I: Einführung in Python
: Dieser Teil beleuchtet zunächst die Gründe dafür, dass Python ein so angenehmer Begleiter für Excel ist, und stellt dann die Tools vor, die wir in diesem Buch verwenden: die Anaconda-Python-Distribution, Visual Studio Code und Jupyter Notebooks. Außerdem erfahren Sie in diesem Teil genug über Python, um den Rest des Buchs zu meistern.

Teil II: Einführung in pandas
: Bei pandas handelt es sich um die Standardbibliothek von Python für die Datenanalyse. In diesem Teil lernen Sie, wie sich Excel-Arbeitsmappen durch eine Kombination aus Jupyter Notebooks und pandas ersetzen lassen. Normalerweise ist pandas-Code sowohl einfacher zu pflegen als auch effizienter als eine Excel-Arbeitsmappe. Und Sie können mit Datensätzen arbeiten, die für ein Excel-Tabellenblatt viel zu groß sind. Im Unterschied zu Excel können Sie mit pandas Ihren Code überall dort ausführen, wo Sie wollen, auch in der Cloud.

Teil III: Excel-Dateien ohne Excel lesen und schreiben

In diesem Teil geht es um die Bearbeitung von Excel-Dateien mit einem der folgenden Python-Pakete: pandas, OpenPyXL, XlsxWriter, pyxlsb, xlrd und xlwt. Diese Pakete sind in der Lage, Excel-Arbeitsmappen direkt vom Datenträger zu lesen und darauf zu schreiben, und ersetzen somit die Excel-Anwendung: Da Sie keine Installation von Excel benötigen, funktionieren sie auf jeder Plattform, die Python unterstützt, einschließlich Windows, macOS und Linux. So wendet man ein Reader-Paket typischerweise an, um Daten aus Excel-Dateien einzulesen, die man jeden Morgen von einem externen Unternehmen oder System erhält, und diese Daten in einer Datenbank zu speichern. Ein Writer-Paket liefert zum Beispiel die Funktionalität hinter der berühmten Schaltfläche *Nach Excel exportieren*, die man in fast jeder Anwendung findet.

Teil IV: Die Excel-Anwendung mit xlwings programmieren

In diesem Teil erfahren Sie, wie Sie Python mit dem Paket xlwings einsetzen können, um die Excel-Anwendung zu automatisieren, anstatt Excel-Dateien vom Datenträger zu lesen und darauf zu schreiben. Demzufolge ist für diesen Teil eine lokale Installation von Excel erforderlich. Sie lernen, wie Sie Excel-Arbeitsmappen öffnen und sie vor Ihren Augen bearbeiten. Neben dem Lesen und Schreiben von Dateien via Excel erstellen Sie interaktive Excel-Tools. Mit diesen ist es dann möglich, eine Schaltfläche anzuklicken, um Python etwas ausführen zu lassen, was Sie vielleicht schon mit VBA-Makros bewerkstelligt haben – beispielsweise eine umfangreiche Berechnung. Außerdem lernen Sie, wie Sie benutzerdefinierte Funktionen (*User-Defined Functions*, UDFs) in Python statt in VBA schreiben.

Es ist wichtig, den grundlegenden Unterschied zwischen Lesen und Schreiben von Excel-*Dateien* (Teil III) und dem Programmieren der Excel-*Anwendung* (Teil IV) zu verstehen, wie es Abbildung E-1 veranschaulicht.

Da Teil III keine Installation von Excel erfordert, funktioniert alles auf allen Plattformen, die Python unterstützen, namentlich Windows, macOS und Linux. Demgegenüber setzt Teil IV Plattformen voraus, die Microsoft Excel unterstützen, d. h. Windows und macOS, da sich der Code auf eine lokale Installation von Microsoft Excel stützt.

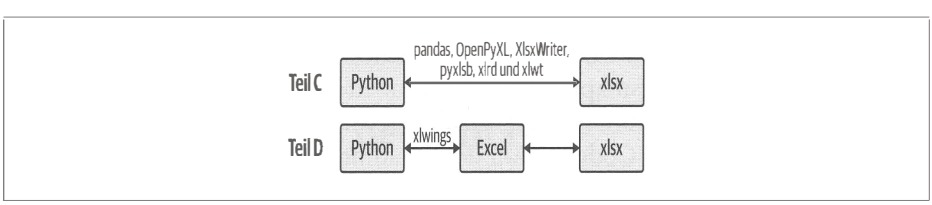

Abbildung E-1: Lesen und Schreiben von Excel-Dateien (Teil III) vs. Programmieren der Excel-Anwendung (Teil IV)

Python- und Excel-Versionen

Die Beispiele in diesem Buch sind mit der Anaconda-Python-Distribution 2020.11 (die Python 3.8 verwendet) und 2021.11 (die Python 3.9 verwendet) getestet worden. Wenn Sie eine neuere Version von Python verwenden möchten, beispielsweise Python 3.10, folgen Sie bitte den Anweisungen auf der Homepage des Buchs. Gelegentlich habe ich einen Kommentar hinterlassen, wenn sich etwas zwischen Python 3.8 und Python 3.9 geändert hat.

Es wird in diesem Buch davon ausgegangen, dass Sie eine moderne Version von Excel verwenden, d. h. mindestens Excel 2007 unter Windows und Excel 2016 unter macOS. Die lokal installierte Version von Excel, die zum Microsoft 365-Abonnement gehört, funktioniert ebenfalls perfekt – ich empfehle sie sogar, da sie über die neuesten Funktionen verfügt, die Sie in anderen Versionen von Excel nicht finden. Zudem ist es die Version, die ich für dieses Buch verwendet habe. Wenn Sie also mit einer anderen Version von Excel arbeiten, kann es sein, dass Sie kleine Unterschiede im Namen oder in der Position eines Menüelements feststellen.

In diesem Buch verwendete Konventionen

In diesem Buch gelten die folgenden typografischen Konventionen:

Kursiv
: Kennzeichnet neue Begriffe, URLs, E-Mail-Adressen, Dateinamen und Dateiendungen.

`Schreibmaschinenschrift`
: Wird in Programmlistings verwendet und im Fließtext für Programmelemente wie zum Beispiel Variablen- oder Funktionsnamen, Datenbanken, Datentypen, Umgebungsvariablen, Anweisungen und Schlüsselwörter.

`Schreibmaschinenschrift fett`
: Kennzeichnet Befehle oder andere Texte, die vom Benutzer buchstäblich eingegeben werden sollen.

`Schreibmaschinenschrift kursiv`
: Zeigt Text, der ersetzt werden soll durch Werte, die der Benutzer bereitstellt, oder Werte, die sich aus dem Kontext ergeben. Außerdem werden die in Tabellen kursiv gedruckten Transformationen derzeit von PyTorch nicht unterstützt.

Dieses Element kennzeichnet einen Tipp oder Vorschlag.

Dieses Element kennzeichnet einen allgemeinen Hinweis.

Dieses Element kennzeichnet eine Warnung oder einen Achtungshinweis.

Codebeispiele

Ich unterhalte eine Webseite (*https://xlwings.org/book*) mit ergänzenden Informationen, die Ihnen bei der Arbeit mit diesem Buch helfen sollen. Diese Seite sollten Sie sich unbedingt anschauen, vor allem dann, wenn Sie auf ein Problem stoßen.

Ergänzendes Material (Codebeispiele, Übungen usw.) stehen Ihnen unter *https://github.com/fzumstein/python-for-excel* zum Download zur Verfügung. Um dieses Begleitmaterial herunterzuladen, klicken Sie auf die grüne Schaltfläche *Code* und wählen dann *Download ZIP*. Nachdem Sie die ZIP-Datei heruntergeladen haben, klicken Sie unter Windows mit der rechten Maustaste auf diese Datei und wählen *Alle extrahieren*, um die im Archiv enthaltenen Dateien in einen Ordner zu entpacken. Unter macOS doppelklicken Sie hierzu einfach auf die Datei. Wenn Sie sich mit Git auskennen, können Sie das Archiv auch per Git auf Ihre lokale Festplatte klonen. Den Zielordner können Sie frei wählen, doch ich verweise im Buch gelegentlich auf folgenden Ordner:

```
C:\Users\username\python-for-excel
```

Wenn Sie die ZIP-Datei unter Windows einfach herunterladen und entpacken, erhalten Sie eine ähnliche Ordnerstruktur wie die folgende (wobei die sich wiederholenden Ordnernamen zu beachten sind):

```
C:\...\Downloads\python-for-excel-1st-edition\python-for-excel-1st-edition
```

Damit Sie Beispiele im Buch leichter verfolgen können, kopieren Sie diesen Ordner am besten in einen Ordner, den Sie unter *C:\Users\<Benutzername>\python-for-excel* anlegen. Das Gleiche gilt für macOS: Kopieren Sie die Dateien nach */Users/<Benutzername>/python-for-excel*.

Wenn Sie eine technische Frage oder ein Problem in Bezug auf die Codebeispiele haben, senden Sie bitte eine E-Mail an *kommentar@oreilly.de*.

Dieses Buch soll Ihnen bei Ihrer täglichen Arbeit helfen. Falls Beispielcode zum Buch angeboten wird, dürfen Sie ihn im Allgemeinen in Ihren Programmen und für

Dokumentationen verwenden. Sie müssen uns nicht um Erlaubnis bitten, es sei denn, Sie kopieren einen erheblichen Teil des Codes. Wenn Sie zum Beispiel ein Programm schreiben, das einige Codeblöcke aus diesem Buch verwendet, benötigen Sie keine Erlaubnis. Sollten Sie aber Beispiele aus O'Reilly-Büchern verkaufen oder verbreiten, ist eine Erlaubnis erforderlich. Wenn Sie eine Frage beantworten und dabei dieses Buch oder Beispielcode aus diesem Buch zitieren, brauchen Sie wiederum keine Erlaubnis. Aber wenn Sie erhebliche Teile des Beispielcodes aus diesem Buch in die Dokumentation Ihres Produkts einfließen lassen, ist eine Erlaubnis einzuholen.

Wir schätzen eine Quellenangabe, verlangen sie aber nicht. Eine Quellenangabe umfasst in der Regel Titel, Autor, Verlag und ISBN, zum Beispiel: »*Python für Excel*« von Felix Zumstein (O'Reilly). Copyright 2022 dpunkt.verlag, ISBN 978-3-96009-197-4.«

Sollten Sie der Meinung sind, dass Sie die Codebeispiele in einer Weise verwenden, die über die oben erteilte Erlaubnis hinausgeht, kontaktieren Sie uns bitte unter *kommentar@oreilly.de*.

Danksagung

Als Erstautor bin ich unglaublich dankbar für die Hilfe, die ich auf dem Weg bis zu diesem Buch von vielen Menschen bekommen habe – sie haben mir diese Reise sehr erleichtert!

Für ihre großartige Arbeit möchte mich bedanken bei meiner Lektorin Melissa Potter von O'Reilly – sie hat mich motiviert, im Zeitplan gehalten und mir geholfen, dieses Buch in eine lesbare Form zu bringen. Danke auch an Michelle Smith, die mit mir an dem ursprünglichen Buchvorschlag gearbeitet hat, und Daniel Elfanbaum, der nicht müde wurde, meine technischen Fragen zu beantworten.

Ein großes Dankeschön geht an alle meine Kollegen, Freunde und Kunden, die viele Stunden damit verbracht haben, meine ersten rudimentären Entwürfe zu lesen. Ihr Feedback war entscheidend, um das Buch verständlicher zu machen, und ein paar Fallstudien sind inspiriert worden von einigen realen Excel-Problemen, die sie mit mir geteilt haben. Mein Dank geht an Adam Rodriguez, Mano Beeslar, Simon Schiegg, Rui Da Costa, Jürg Nager und Christophe de Montrichard.

Hilfreiches Feedback habe ich auch von den Lesern der frühen Fassung erhalten, die auf der Onlinelernplattform von O'Reilly veröffentlicht wurde. Vielen Dank an Felipe Maion, Ray Doue, Kolyu Minevski, Scott Drummond, Volker Roth und David Ruggles!

Ich hatte das große Glück, dass hochqualifizierte technische Gutachter das Buch rezensiert haben, und ich weiß die harte Arbeit sehr zu schätzen, die sie unter großem Zeitdruck geleistet haben. Vielen Dank Jordan Goldmeier, George Mount, Andreas Clenow, Werner Brönnimann und Eric Moreira für all eure Hilfe!

Ein besonderer Dank geht an Björn Stiel, der nicht einfach technischer Rezensent war, sondern von dem ich auch viele der Dinge gelernt habe, über die ich in diesem Buch schreibe. Die schon mehrere Jahre dauernde Zusammenarbeit habe ich sehr genossen.

Nicht zuletzt möchte ich mich bei Eric Reynolds bedanken, der 2016 sein Excel-Python-Projekt in die xlwings-Codebasis integriert hat. Zudem hat er das gesamte Paket von Grund auf neu gestaltet, sodass meine schreckliche API aus den Anfangstagen der Vergangenheit angehört. Herzlichen Dank dafür.