

Der Inhalt (im Überblick)

	Einführung	xxi
1	Der Anfang des Programmierens: <i>Orientierungshilfe</i>	1
2	Textdaten: <i>Jeder Text zu seiner Zeit</i>	37
3	Funktionen: <i>Ordnung schaffen</i>	77
4	Daten in Dateien und Listen: <i>Sortieren</i>	113
5	Abbildungen und Datenbanken: <i>Daten ihren Platz zuweisen</i>	145
6	Modulare Programmierung: <i>In der Spur bleiben</i>	177
7	Grafische Benutzeroberflächen: <i>Visuell-Werdung</i>	215
8	GUIs und Daten: <i>Grafisch Dateneingabe</i>	257
8 1/2	Ausnahmen und Dialogfenster: <i>Nachricht angekommen?</i>	293
9	Elemente grafischer Oberflächen: <i>Das richtige Werkzeug wählen</i>	313
10	Eigene Widgets und Klassen: <i>Objekte im Sinn</i>	349
Anhang	Was übrig bleibt: <i>Die Top-Ten der Dinge (die wir nicht behandelt haben)</i>	385
	Index	397

Der Inhalt (jetzt ausführlich)

Einführung

Ihr Gehirn und das Programmieren. Sie versuchen, etwas zu *lernen*, und Ihr *Hirn* tut sein Bestes, damit das Gelernte nicht *hängen bleibt*. Es denkt nämlich: »Wir sollten lieber ordentlich Platz für wichtigere Dinge lassen, z.B. für das Wissen, welche Tiere einem gefährlich werden könnten, oder dass es eine ganz schlechte Idee ist, nackt Snowboard zu fahren.« Tja, wie schaffen wir es nun, Ihr Gehirn davon zu überzeugen, dass Ihr Leben davon abhängt, etwas über das Programmieren zu wissen?

Für wen ist dieses Buch?	xxii
Wir wissen, was Sie jetzt denken	xxiii
Metakognition	xxv
So machen Sie sich Ihr Gehirn Untertan	xxvii
Lies mich	xxviii
Die technischen Gutachter	xxx
Danksagungen	xxxi

Der Anfang des Programmierens

Orientierungshilfe

1

Eigene Programme verleihen Ihnen die Macht, Ihren PC zu steuern.

Fast jeder weiß, wie man mit einem Computer *arbeitet*, aber nur wenige machen den nächsten Schritt und lernen, wie man ihn *steuert*. Wenn Sie Software nutzen, die von anderen entwickelt wurde, sind Sie immer auf das beschränkt, was Sie *deren* Meinung nach tun möchten. Schreiben Sie Ihre eigenen Programme, und nur noch Ihre eigene Vorstellungskraft kann Sie einschränken. Das Programmieren macht Sie kreativer, lässt Sie präziser denken und lehrt Sie, Probleme logisch zu analysieren und zu lösen.

Wollen Sie lieber programmiert oder Programmierer sein?

Mit Programmieren können Sie mehr erreichen	2
Und wie führt man das Programm aus?	5
Eine neue Programmdatei erstellen	6
Code vorbereiten und ausführen	7
Ein Programm ist mehr als eine einfache Folge von Befehlen	12
Codeville: Ihr Programm ist wie ein Straßennetz	13
Verzweigungen sind Codeschnittstellen	14
if/else-Verzweigungen	15
Der Python-Code braucht verknüpfte Verzweigungen	20
In Python werden Pfade durch Einrückung verbunden	21
Mit Schleifen können Sie Code immer wieder ausführen lassen	28
Pythons while-Schleife	29
Ihr Programmierwerkzeugkasten	35

Textdaten

Jeder Text zu seiner Zeit

2

Stellen Sie sich vor, Sie müssten ohne Worte kommunizieren.

Alle Programme verarbeiten Daten – und eine der wichtigsten Arten von Daten ist **Text**.

In diesem Kapitel werden Sie das Basiswissen zu **Textdaten** erhalten. Sie werden Text

suchen und automatisch genau das bekommen, **was Sie suchten**. Und unterwegs

werden Sie grundlegende Konzepte der Programmierung wie **Methoden** aufsammeln

und erfahren, wie Sie sie einsetzen können, **um sich Ihre Daten gefügig zu machen**.

Und schließlich werden Sie Ihren Programmen mithilfe von **Codebibliotheken** im

Handumdrehen **Superkräfte geben**.

Der neue Sternback-Auftrag	38
Hier ist der aktuelle Sternback-Code	39
Der Preis ist in das HTML eingebettet	41
Ein String ist eine Folge von Zeichen	41
Zeichen in Text finden	42
Aber wie erhält man mehr als ein Zeichen?	43
Beans'R'Us belohnt Stammkunden	50
Suchen ist nicht leicht	52
Python-Daten sind schlau	54
Strings und Zahlen sind unterschiedlich	64
Das Programm hat den Beans'R'Us Server überfordert	67
Zeit ... hätten wir nur mehr davon	68
Sie nutzen bereits eine Bibliothek	69
Die Ordnung ist wiederhergestellt	74
Ihr Programmierwerkzeugkasten	75

Funktionen

3

Ordnung schaffen

Wenn Programme wachsen, wird Code unübersichtlich.

Unübersichtlicher und komplexer Code kann schwer zu lesen und noch schwerer zu warten sein. Eine Möglichkeit, diese Komplexität in den Griff zu bekommen, ist der Einsatz von **Funktionen**. Funktionen sind **Codeeinheiten**, die Sie nach Bedarf in Ihren Programmen einsetzen. Sie ermöglichen Ihnen, **gemeinsam genutzte Aktionen auszulagern**, und das bedeutet, dass Sie Ihren Code **besser lesbar** und **leichter wartbar** machen. In diesem Kapitel werden Sie entdecken, wie Ihnen etwas Wissen zu Funktionen **das Programmiererleben erheblich vereinfachen kann**.

Sternback gehen die Bohnen aus!	78
Was muss das neue Programm leisten?	79
Vermeiden Sie Codeverdopplung ...	81
Codewiederverwendung mit Funktionen	82
Bringen Sie die Dinge in die richtige Reihenfolge	84
Mit dem Befehl return Daten zurückliefern	87
Nutze das Web, Luke	93
Die Funktion sendet immer die gleiche Nachricht	94
Funktionsverdopplung mit Parametern vermeiden	96
Jemand hat an Ihrem Code herumgepfuscht	102
Der Rest des Programms kann die Variable password nicht sehen	104
Wenn Sie eine Funktion aufrufen, erstellt der Computer eine neue Variablenliste	105
Wenn Sie eine Funktion verlassen, werden Ihre Variablen weggeworfen	106
Die Sternback-Lager sind gefüllt!	110
Ihr Programmierwerkzeugkasten	111

Daten in Dateien und Listen

Sortieren

4

Mit Ihren Programmen entwickeln sich auch die Datenverarbeitungsanforderungen.

Und wenn Sie mit vielen Daten arbeiten müssen, wird es schnell recht mühselig, einzelne Variablen für jedes einzelne Datenelement zu verwenden. Programmierer nutzen deswegen ziemlich beeindruckende Behälter oder Container (die als **Datenstrukturen** bezeichnet werden), um sich die Arbeit mit vielen Daten zu vereinfachen. Häufig kommen alle Daten aus Dateien, die auf der Festplatte gespeichert sind. Aber wie arbeitet man mit Daten in Dateien? Es wird sich herausstellen, dass es ein Kinderspiel ist. Blättern Sie um und erfahren Sie, warum!

Starke Brandung in Codeville	114
Die höchste Punktzahl in der Ergebnisdatei finden	115
Dateien mit dem Öffnen-Lesen-Schließen-Muster durchlaufen	116
Die Datei enthält nicht nur Zahlen ...	120
Die Zeilen beim Lesen spalten	121
Die Methode <code>split()</code> zerlegt den String	122
Aber Sie benötigen mehr Ergebnisse	126
3 Highscores machen den Code komplexer	127
Eine geordnete Liste macht den Code erheblich einfacher	128
Im Speicher sortieren ist einfacher	129
Sie können unmöglich eine eigene Variable für jede Datenzeile nutzen	130
Mit Listen können Sie ganze Datenzüge verwalten	131
Listen in Python	132
Die Liste vor Anzeige der Ergebnisse sortieren	136
Die Punktzahlen in absteigender Folge sortieren	139
Und wer ist der Sieger?	142
Wir haben die Namen der Surfer vergessen	143
Ihr Programmierwerkzeugkasten	144

Abbildungen und Datenbanken

5

Daten ihren Platz zuweisen

Listen sind nicht der einzige Datencontainer.

Programmiersprachen bieten weitere Spielzeuge zum Festhalten von Daten, und unser erwähltes Werkzeug, Python, macht da keine Ausnahme. In diesem Kapitel werden Sie Werte mit Namen **verknüpfen** und dazu eine Datenstruktur nutzen, die unter vielen Namen bekannt ist, Hash, Tabelle, **Abbildung**, und unter Python *Dictionary*. Und bei der Speicherung von Daten wollen wir uns jetzt nicht mehr auf Textdaten in Dateien beschränken, sondern auf Daten in einem *externen Datenbanksystem zugreifen*. **Information** ist alles, und da es die ohne Daten nicht gibt, blättern Sie um und beginnen damit, Ihre stetig wachsenden Programmierfertigkeiten auf einige coole Datenverarbeitungsaufgaben anzuwenden.

Wer hat den Wettbewerb gewonnen?	146
Punkte und Namen verbinden	150
Einen Schlüssel und einen Wert verbinden	153
Abbildungen mit for durchlaufen	154
Die Daten sind nicht sortiert	158
Wenn Daten komplexer werden	160
Aus einer Funktion eine Datenstruktur liefern	164
Und hier: Ihr neues Brett!	168
Unterdessen im Studio ...	169
Der Code bleibt gleich; die Funktion ändert sich	170
Die vKbF-TV-Daten bringen Geld!	174
Ihr Programmierwerkzeugkasten	175

Modulare Programmierung

In der Spur bleiben

6

Ihr Code geht in viele Programme ein.

Und obgleich **Teilen** eine gute Sache ist, müssen Sie *aufpassen*. Es könnte sein, dass Programmierer Ihren Code nehmen und auf **unerwartete** Weise nutzen oder andere ihn einfach ändern, ohne Sie darüber zu informieren. Eventuell möchten Sie eine Funktion in all Ihren Programmen nutzen, und mit der Zeit **ändert** sich der Code dieser Funktion, damit er weiterhin Ihre Anforderungen erfüllt. Schlaue Programmierer nutzen *modulare Programmiertechniken*, um ihre Arbeitsbelastung unter Kontrolle zu halten. Finden Sie auf den folgenden Seiten heraus, wie man das anstellt ...

Fit von Kopf bis Fuß aktualisiert seine Systeme	178
Das Programm muss eine Transaktionsdatei erstellen	179
Strings mit Strings formatieren	180
Spät in der Nacht schneit eine Mail rein	187
50000 ... für einen Donut?!	188
Nur die Geschäfte aus Ihrem Programm sind betroffen	189
Die neue Bank nutzt ein neues Format	190
Das Programm in der Bar nutzt immer noch das alte Format	191
Aktualisieren Sie nicht einfach Ihren Code	192
Und wie erstellen wir ein Modul?	193
Auch die Transaktionsdatei funktioniert	199
Der Fitnessclub hat einen neuen Wunsch	200
Der Sternback-Code	205
Die beiden Rabattfunktionen haben den gleichen Namen	206
Vollständigqualifizierte Namen verhindern Verwirrungen	207
Der Rabatt lässt die Kunden herbeiströmen	213
Ihr Programmierwerkzeugkasten	214

Grafische Benutzeroberflächen

7

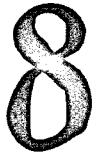
Augen- und Ohrenschmaus

Ihre Fertigkeiten als Programmierer nehmen stetig zu.

Aber es ist ein Jammer, dass Ihre Programme nicht *ansehnlicher* sind. Fragen und Antworten auf der Konsole anzeigen zu können, ist ja ganz hübsch, aber doch recht retro, oder nicht? Fehlt eigentlich nur noch grüner Text auf schwarzem Hintergrund zum wahren IT-Steinzeitgefühl. Es muss doch eine *bessere* Möglichkeiten geben, mit Benutzern zu kommunizieren – und die gibt es natürlich: **grafische Benutzeroberflächen** oder **GUIs**. Klingt cool und komplex und kann beides sein. Aber keine Angst, mit ein zwei Tricks haben Sie Ihren grafischen Code in null Komma nichts laufen.

Von Kopf bis Fuß-TV produziert auch Game-Shows	216
pygame ist plattformübergreifend	220
0... 2... 1... 9... abheben!	230
tkinter schenkt Ihnen die Ereignisschleife	234
tkinter hat massenhaft Optionen	235
Das GUI funktioniert, macht aber nichts	238
Code mit Button-Ereignissen verbinden	239
Jetzt ist das GUI-Programm für ein Casting bereit	244
Aber noch ist der Moderator nicht zufrieden	246
Labeln Sie!	249
Ihr Programmierwerkzeugkasten	255

GUIs und Daten



Grafische Dateneingabe

GUIs verarbeiten nicht nur Ereignisse, sondern auch Daten.

Fast alle GUI-Anwendungen müssen Benutzerdaten lesen, und die Wahl der richtigen Eingabeelemente kann entscheiden, ob Ihre Benutzeroberfläche eine *Dateneingabehölle* oder der siebte *Benutzerhimmel* ist. Widgets können einfachen Text akzeptieren oder nur eine Optionsliste präsentieren. Es gibt viele verschiedene Widgets, und das heißt, dass Sie viele Optionen haben. Welche Sie davon wählen, kann natürlich ganz entscheidende Auswirkungen haben. Es ist an der Zeit, dass Sie Ihre GUI-Programme **weiterentwickeln**.

PPD braucht ein neues Versandsystem	258
Es gibt bereits einen Entwurf für die Benutzeroberfläche	259
Datenerfassung im GUI	260
Mit Entry- und Text-Widgets kann Ihr GUI Textdaten aufnehmen	261
Daten in Textfeldern lesen und schreiben	262
Text-Felder sind komplizierter	263
Eine der Sendungen ging in die Irre	270
In die Felder können beliebige Werte eingegeben werden	271
Radiobuttons beschränken die Eingabe	272
Radiobuttons in tkinter erstellen	273
Die Radiobuttons sollten zusammenarbeiten	275
Die Radiobuttons können ein Modell teilen	276
Das System sagt den anderen Widgets, wenn sich das Modell ändert	277
Wie man in tkinter Modelle nutzt	278
PPD expandiert	282
Es gibt zu viele Depots im GUI	283
Ein OptionMenu bietet Ihnen so viele Optionen, wie Sie benötigen	284
Das Modell bleibt gleich	285
Ihr Programmierwerkzeugkasten	292

8¹/₂

Ausnahmen und Dialogfenster

Nachricht angekommen?

Manchmal geht etwas schief, und Sie müssen sich darum kümmern.

Es gibt immer Dinge, die Sie nicht im Griff haben. Netzwerke fallen aus. Dateien verschwinden. Schlaue Programmierer wissen, wie sie mit derartigen **Fehlern** umgehen, und bieten in ihren Programmen **Ausweichlösungen**. Gute Software informiert den Nutzer, wenn etwas Übles passiert, und sagt ihm, was erforderlich ist, um die Angelegenheit zu regeln. Wenn Sie lernen, wie Sie **Ausnahmen** und **Dialogfenster** einsetzen, können Sie die Zuverlässigkeit und Qualität Ihrer Software steigern.

Was ist das für ein Geruch?	294
Jemand hat die Dateiberechtigungen geändert	295
Als es die Datei nicht schreiben konnte, löste Ihr Programm eine Ausnahme aus	296
Die Ausnahme abfangen	297
Ausnahmen überwachen mit try/except	298
Es gibt ein Problem mit der Ausnahmebehandlung	302
Ein Dialogfenster verlangt Aufmerksamkeit	303
Dialogfenster in Python	304
Ihr Programmierwerkzeugkasten	311

Elemente grafischer Oberflächen

Das richtige Werkzeug wählen

9

Gut verwendbare Programme zu schreiben, ist nicht schwer..

Bei GUI-Anwendungen gibt es einen ganz entscheidenden Unterschied zwischen *funktionierenden* Schnittstellen und **nützlichen** sowie **effektiven** Schnittstellen. Wie man das richtige Werkzeug für eine Aufgabe wählt, lernt man durch Erfahrung, und die erwirbt man nur, indem man mit den verfügbaren Werkzeugen arbeitet. In diesem Kapitel werden Sie Ihre Fertigkeiten im Aufbau von GUI-Anwendungen erweitern. Es gibt noch eine Menge nützlicher Widgets, die auf Sie warten. Blättern Sie also um und lassen Sie uns fortfahren.

Zeit zu mixen	314
Die Musik spielte einfach weiter ...	318
Nicht alle Ereignisse werden von Buttons erzeugt	319
Das Ereignis abfangen reicht nicht	326
Zwei Buttons oder nicht zwei Buttons? Das ist die Frage ...	328
Eine Checkbox ist ein An/Aus-Schalter	331
Checkboxes in tkinter	332
Die Lautstärke anpassen	336
Einen Schieber mit Skala modellieren	337
Die Lautstärke mit pygame anpassen	339
Mit tkinter den Rest erledigen	340
Der DJ ist happy!	347
Ihr Programmierwerkzeugkasten	348

10

Eigene Widgets und Klassen

Objekte im Sinn

Anforderungen können komplex sein, Programme müssen es nicht sein.

Indem Sie Objektorientierung nutzen, können Sie Ihre Programme sehr **leistungsfähig** machen, ohne dazu Unmengen an Code zusätzlich schreiben zu müssen. Lesen Sie weiter und erfahren Sie, wie Sie **eigene Widgets** erstellen, die genau das machen, was Sie *wollen*, und Ihnen dazu verhelfen, **Ihre Programmierfertigkeiten weiterzuentwickeln**.

Der DJ möchte mehrere Stücke spielen	350
Den Code für die Stücke als Funktion speichern	351
Die neue Funktion enthält andere Funktionen	356
Die Funktion muss Widgets und Ereignis-Handler erstellen	357
Der DJ ist verwirrt	362
Widgets gruppieren	363
Frame-Widgets enthalten andere Widgets	364
Klassen sind Maschinen zur Erstellung von Objekten	366
Klassen haben Methoden, die Verhalten definieren	367
Aber wie rufen Objekte Methoden auf?	369
Die Klasse MixerPanel hat große Ähnlichkeit mit der Funktion gui_erstellen()	370
Klassen = Methoden + Daten	372
Der DJ hat ein ganzes Verzeichnis voll mit Musik	378
It's party time!	382
Ihr Programmierwerkzeugkasten	383
Der Aufbruch ...	384
Es war nett mit Ihnen hier in Codeville!	384

Anhang: Was übrig bleibt



Die Top-Ten der Dinge (die wir nicht behandelt haben)

Sie haben einen weiten Weg zurückgelegt.

Doch programmieren zu lernen ist eine Aufgabe, die nie abgeschlossen ist. Je mehr Sie programmieren, umso wichtiger wird es, dass Sie **neue Wege erforschen, um bestimmte Dinge zu tun**. Sie müssen sich mit **neuen Werkzeugen** und **neuen Techniken** vertraut machen. Leider bietet dieses Buch einfach nicht genügend Raum dafür, Ihnen all das zu zeigen, was eventuell nützlich für Sie werden könnte. Deswegen finden Sie hier eine Liste der zehn wichtigsten Dinge, die wir nicht behandelt haben, die Sie vielleicht aber als Nächstes lernen sollten.

1. »Den Python-Weg« wählen	386
2. Mit Python 2 arbeiten	387
3. Andere Programmiersprachen	388
4. Automatisiertes Testen	389
5. Debuggen	390
6. Kommandozeilenausführung	391
7. OOP ist etwas zu kurz gekommen	392
8. Algorithmen	393
9. Fortgeschrittene Programmierthemen	394
10. Andere IDEs, Shells und Texteditoren	395