Inhalt

1 Einleitung 27

2 Die Programmiersprache Python 35

2.1 Historie, Konzepte, Einsatzgebiete 35
- 2.1.1 Geschichte und Entstehung 35
- 2.1.2 Grundlegende Konzepte 36
- 2.1.3 Einsatzmöglichkeiten und Stärken 37
- 2.1.4 Einsatzbeispiele 38

2.2 Die Installation von Python 38
- 2.2.1 Installation von Anaconda unter Windows 39
- 2.2.2 Installation von Anaconda unter Linux 39
- 2.2.3 Installation von Anaconda unter macOS 40

2.3 Die Verwendung von Python 41

TEIL I Einstieg in Python

3 Erste Schritte im interaktiven Modus 45

3.1 Ganze Zahlen 46

3.2 Gleitkommazahlen 47

3.3 Zeichenketten 48

3.4 Listen 49

3.5 Dictionarys 49

3.6 Variablen 50

3.7 Logische Ausdrücke 52

3.8 Funktionen und Methoden 54
- 3.8.1 Funktionen 54
- 3.8.2 Methoden 55

3.9 Bildschirmausgaben 56

4 Der Weg zum ersten Programm 59

4.1 Tippen, kompilieren, testen ... 59
4.1.1 Shebang ... 61
4.1.2 Interne Abläufe ... 61

4.2 Grundstruktur eines Python-Programms ... 63
4.2.1 Umbrechen langer Zeilen ... 65
4.2.2 Zusammenfügen mehrerer Zeilen ... 65

4.3 Das erste Programm ... 66

4.4 Kommentare ... 69

4.5 Der Fehlerfall ... 69

5 Kontrollstrukturen 71

5.1 Fallunterscheidungen ... 71
5.1.1 Die if-Anweisung ... 71
5.1.2 Bedingte Ausdrücke ... 75

5.2 Schleifen ... 76
5.2.1 Die while-Schleife ... 76
5.2.2 Abbruch einer Schleife ... 77
5.2.3 Erkennen eines Schleifenabbruchs ... 78
5.2.4 Abbruch eines Schleifendurchlaufs ... 79
5.2.5 Die for-Schleife ... 81
5.2.6 Die for-Schleife als Zählschleife ... 83

5.3 Die pass-Anweisung ... 84

6 Dateien 85

6.1 Datenströme ... 85

6.2 Daten aus einer Datei auslesen ... 86

6.3 Daten in eine Datei schreiben ... 90

6.4 Das Dateiobjekt erzeugen ... 91
6.4.1 open(filename, [mode, buffering, encoding, errors, newline]) ... 91
6.4.2 Attribute und Methoden eines Dateiobjekts ... 93
6.4.3 Die Schreib-/Leseposition verändern ... 94

7 Das Laufzeitmodell 97

7.1 Die Struktur von Instanzen ... 99
7.1.1 Datentyp ... 99
7.1.2 Wert ... 100
7.1.3 Identität ... 101

7.2 Referenzen und Instanzen freigeben ... 103

7.3 Mutable vs. immutable Datentypen ... 104
7.3.1 Mutable Datentypen und Seiteneffekte ... 105

8 Funktionen, Methoden und Attribute 109

8.1 Parameter von Funktionen und Methoden ... 109
8.1.1 Positionsbezogene Parameter ... 110
8.1.2 Schlüsselwortparameter ... 110
8.1.3 Optionale Parameter ... 111
8.1.4 Reine Schlüsselwortparameter ... 111

8.2 Attribute ... 112

9 Informationsquellen zu Python 113

9.1 Die Built-in Function help ... 113

9.2 Die Onlinedokumentation ... 114

9.3 PEPs ... 114

TEIL II Datentypen

10 Das Nichts – NoneType 119

11 Operatoren 121

12 Numerische Datentypen 125

12.1 Arithmetische Operatoren ... 125

12.2 Vergleichende Operatoren ... 127

12.3 Konvertierung zwischen numerischen Datentypen ... 128

12.4 Ganzzahlen – int ... 129

12.4.1 Zahlensysteme ... 129

12.4.2 Bit-Operationen ... 131

12.4.3 Methoden ... 134

12.5 Gleitkommazahlen – float ... 135

12.6 Boolesche Werte – bool ... 137

12.6.1 Logische Operatoren ... 137

12.6.2 Wahrheitswerte nicht-boolescher Datentypen ... 140

12.6.3 Auswertung logischer Operatoren ... 141

12.7 Komplexe Zahlen – complex ... 143

13 Sequenzielle Datentypen 147

13.1 Operationen auf Instanzen sequenzieller Datentypen ... 148

13.1.1 Ist ein Element vorhanden? – die Operatoren in und not in ... 149

13.1.2 Verkettung von Sequenzen – die Operatoren + und += ... 151

13.1.3 Wiederholung von Sequenzen – die Operatoren * und *= ... 152

13.1.4 Zugriff auf bestimmte Elemente einer Sequenz – der []-Operator ... 153

13.1.5 Länge einer Sequenz – die Built-in Function len ... 157

13.1.6 Das kleinste und das größte Element einer Sequenz – min und max 158

13.1.7 Die Position eines Elements in der Sequenz – s.index(x, [i, j]) ... 158

13.1.8 Anzahl der Vorkommen eines Elements der Sequenz – s.count(x) ... 159

13.2 Listen – list ... 159

13.2.1 Verändern eines Wertes innerhalb der Liste – Zuweisung mit [] ... 161

13.2.2 Ersetzen von Teillisten und Einfügen neuer Elemente – Zuweisung mit [] ... 161

13.2.3 Elemente und Teillisten löschen – del zusammen mit [] ... 162

13.2.4 Methoden von list-Instanzen ... 162

13.2.5 Weitere Eigenschaften von Listen ... 168

13.3 Unveränderliche Listen – tuple ... 171
13.3.1 Tuple Packing/Unpacking und Sequence Unpacking ... 172
13.3.2 Immutabel heißt nicht zwingend unveränderlich! ... 174

13.4 Strings – str, bytes, bytearray ... 174
13.4.1 Steuerzeichen ... 177
13.4.2 String-Methoden ... 179
13.4.3 Formatierung von Strings ... 189
13.4.4 Zeichensätze und Sonderzeichen ... 198

14 Zuordnungen 207

14.1 Dictionary – dict ... 207
14.1.1 Operatoren ... 210
14.1.2 Methoden ... 212

15 Mengen 219

15.1 Die Datentypen set und frozenset ... 219
15.1.1 Operatoren ... 220
15.1.2 Methoden ... 226

15.2 Veränderliche Mengen – set ... 227

15.3 Unveränderliche Mengen – frozenset ... 229

16 Collections 231

16.1 Verkettete Dictionarys ... 231

16.2 Zählen von Häufigkeiten ... 232

16.3 Dictionarys mit Standardwerten ... 235

16.4 Doppelt verkettete Listen ... 235

16.5 Benannte Tupel ... 237

16.6 Sortierte Dictionarys ... 239

17 Datum und Zeit 241

17.1 Elementare Zeitfunktionen – time ... 241
17.1.1 Attribute ... 243
17.1.2 Funktionen ... 244

17.2 Objektorientierte Datumsverwaltung – datetime ... 249
17.2.1 datetime.date ... 250
17.2.2 datetime.time ... 251
17.2.3 datetime.datetime ... 252
17.2.4 datetime.timedelta ... 254
17.2.5 Operationen für datetime.datetime und datetime.date ... 257
17.2.6 Bemerkung zum Umgang mit Zeitzonen ... 259

18 Aufzählungstypen – Enum 261

18.1 Aufzählungstyp für Bitmuster – Flag ... 263

18.2 Ganzzahlige Aufzählungstypen – IntEnum ... 264

TEIL III Fortgeschrittene Programmiertechniken

19 Funktionen 267

19.1 Schreiben einer Funktion ... 269

19.2 Funktionsobjekte ... 272

19.3 Funktionsparameter ... 273
19.3.1 Optionale Parameter ... 273
19.3.2 Schlüsselwortparameter ... 274
19.3.3 Beliebige Anzahl von Parametern ... 275
19.3.4 Reine Schlüsselwortparameter ... 277
19.3.5 Entpacken einer Parameterliste ... 278
19.3.6 Seiteneffekte ... 280

19.4 Namensräume ... 283
19.4.1 Zugriff auf globale Variablen – global ... 283
19.4.2 Zugriff auf den globalen Namensraum ... 284
19.4.3 Lokale Funktionen ... 285
19.4.4 Zugriff auf übergeordnete Namensräume – nonlocal ... 286

19.5 Anonyme Funktionen ... 288

19.6 Annotationen ... 289

19.7 Rekursion ... 291

19.8 Eingebaute Funktionen ... 291

19.8.1 abs(x) ... 295
19.8.2 all(iterable) ... 295
19.8.3 any(iterable) ... 295
19.8.4 ascii(object) ... 296
19.8.5 bin(x) ... 296
19.8.6 bool([x]) ... 296
19.8.7 bytearray([source, encoding, errors]) ... 297
19.8.8 bytes([source, encoding, errors]) ... 298
19.8.9 chr(i) ... 298
19.8.10 complex([real, imag]) ... 298
19.8.11 dict([source]) ... 299
19.8.12 divmod(a, b) ... 300
19.8.13 enumerate(iterable) ... 300
19.8.14 eval(expression, [globals, locals]) ... 300
19.8.15 exec(object, [globals, locals]) ... 301
19.8.16 filter(function, iterable) ... 301
19.8.17 float([x]) ... 302
19.8.18 format(value, [format_spec]) ... 302
19.8.19 frozenset([iterable]) ... 303
19.8.20 globals() ... 303
19.8.21 hash(object) ... 303
19.8.22 help([object]) ... 304
19.8.23 hex(x) ... 304
19.8.24 id(object) ... 305
19.8.25 input([prompt]) ... 305
19.8.26 int([x, base]) ... 305
19.8.27 len(s) ... 306
19.8.28 list([sequence]) ... 306
19.8.29 locals() ... 307
19.8.30 map(function, [*iterable]) ... 307
19.8.31 max(iterable, {default, key}), max(arg1, arg2, [*args], {key}) ... 308
19.8.32 min(iterable, {default, key}), min(arg1, arg2, [*args], {key}) ... 309
19.8.33 oct(x) ... 309
19.8.34 ord(c) ... 309
19.8.35 pow(x, y, [z]) ... 310
19.8.36 print([*objects], {sep, end, file, flush]) ... 310

19.8.37 range([start], stop, [step]) ... 311
19.8.38 repr(object) ... 312
19.8.39 reversed(sequence) ... 312
19.8.40 round(x, [n]) ... 312
19.8.41 set([iterable]) ... 313
19.8.42 sorted(iterable, [key, reverse]) ... 313
19.8.43 str([object, encoding, errors]) ... 313
19.8.44 sum(iterable, [start]) ... 314
19.8.45 tuple([iterable]) ... 315
19.8.46 type(object) ... 315
19.8.47 zip([*iterables]) ... 315

20 Modularisierung 317

20.1 Einbinden globaler Module ... 317

20.2 Lokale Module ... 320
20.2.1 Namenskonflikte ... 321
20.2.2 Modulinterne Referenzen ... 322
20.2.3 Module ausführen ... 322

20.3 Pakete ... 323
20.3.1 Importieren aller Module eines Pakets ... 325
20.3.2 Namespace Packages ... 326
20.3.3 Relative Import-Anweisungen ... 326

20.4 Das Paket importlib ... 327
20.4.1 Einbinden von Modulen und Paketen ... 328
20.4.2 Verändern des Import-Verhaltens ... 328

21 Objektorientierung 333

21.1 Klassen ... 338
21.1.1 Definieren von Methoden ... 339
21.1.2 Der Konstruktor und die Erzeugung von Attributen ... 340

21.2 Vererbung ... 343
21.2.1 Technische Grundlagen ... 344
21.2.2 Die Klasse GirokontoMitTagesumsatz ... 347
21.2.3 Mögliche Erweiterungen der Klasse Konto ... 352

21.2.4 Ausblick ... 356
21.2.5 Mehrfachvererbung ... 356

21.3 Setter und Getter und Property Attributes ... 358
21.3.1 Setter und Getter ... 358
21.3.2 Property-Attribute ... 359

21.4 Klassenattribute und Klassenmethoden sowie statische Methoden ... 360
21.4.1 Statische Methoden ... 361
21.4.2 Klassenmethoden ... 362
21.4.3 Klassenattribute ... 363

21.5 Built-in Functions für Objektorientierung ... 363
21.5.1 Funktionen für die Verwaltung der Attribute einer Instanz ... 364
21.5.2 Funktionen für Informationen über die Klassenhierarchie ... 365

21.6 Objektphilosophie ... 367

21.7 Magic Methods und Magic Attributes ... 369
21.7.1 Allgemeine Magic Methods ... 369
21.7.2 Operatoren überladen ... 376
21.7.3 Datentypen emulieren ... 383

22 Ausnahmebehandlung 389

22.1 Exceptions ... 389
22.1.1 Eingebaute Exceptions ... 390
22.1.2 Werfen einer Exception ... 391
22.1.3 Abfangen einer Exception ... 392
22.1.4 Eigene Exceptions ... 396
22.1.5 Erneutes Werfen einer Exception ... 398
22.1.6 Exception Chaining ... 400

22.2 Zusicherungen – assert ... 402

23 Iteratoren 403

23.1 Comprehensions ... 403
23.1.1 List Comprehensions ... 403
23.1.2 Dict Comprehensions ... 405
23.1.3 Set Comprehensions ... 406

23.2 Generatoren ... 407
23.2.1 Subgeneratoren ... 409
23.2.2 Generator Expressions ... 413

23.3 Iteratoren ... 414
23.3.1 Verwendung von Iteratoren ... 417
23.3.2 Mehrere Iteratoren für dieselbe Instanz ... 420
23.3.3 Nachteile von Iteratoren gegenüber dem direkten Zugriff über Indizes 422
23.3.4 Alternative Definition für iterierbare Objekte ... 422
23.3.5 Funktionsiteratoren ... 423

23.4 Spezielle Generatoren – itertools ... 424
23.4.1 accumulate(iterable, [func]) ... 426
23.4.2 chain([*iterables]) ... 426
23.4.3 combinations(iterable, r) ... 426
23.4.4 combinations_with_replacement(iterable, r) ... 427
23.4.5 compress(data, selectors) ... 427
23.4.6 count([start, step]) ... 428
23.4.7 cycle(iterable) ... 428
23.4.8 dropwhile(predicate, iterable) ... 429
23.4.9 filterfalse(predicate, iterable) ... 429
23.4.10 groupby(iterable, [key]) ... 429
23.4.11 islice(iterable, [start], stop, [step]) ... 430
23.4.12 permutations(iterable, [r]) ... 430
23.4.13 product([*iterables], [repeat]) ... 431
23.4.14 repeat(object, [times]) ... 431
23.4.15 starmap(function, iterable) ... 432
23.4.16 takewhile(predicate, iterable) ... 432
23.4.17 tee(iterable, [n]) ... 432
23.4.18 zip_longest([*iterables], [fillvalue]) ... 433

24 Kontextobjekte 435

24.1 Die with-Anweisung ... 435

24.2 Hilfsfunktionen für with-Kontexte – contextlib ... 438
24.2.1 Einfache Funktionen als Kontext-Manager ... 438
24.2.2 Bestimmte Exception-Typen unterdrücken ... 439
24.2.3 Den Standard-Ausgabestrom umleiten ... 440

25 Manipulation von Funktionen und Methoden 441

25.1 Decorator ... 441

25.2 Das Modul functools ... 444

25.2.1 Funktionsschnittstellen vereinfachen ... 444

25.2.2 Methodenschnittstellen vereinfachen ... 446

25.2.3 Caches ... 447

25.2.4 Ordnungsrelationen vervollständigen ... 448

25.2.5 Überladen von Funktionen ... 449

TEIL IV Die Standardbibliothek

26 Mathematik 455

26.1 Mathematische Funktionen – math, cmath ... 455

26.1.1 Zahlentheoretische Funktionen ... 456

26.1.2 Exponential- und Logarithmusfunktionen ... 458

26.1.3 Trigonometrische und hyperbolische Funktionen ... 459

26.1.4 Umrechnen von Winkeln ... 459

26.1.5 Darstellungsformen komplexer Zahlen ... 459

26.2 Zufallszahlengenerator – random ... 460

26.2.1 Den Status speichern und laden ... 461

26.2.2 Zufällige ganze Zahlen erzeugen ... 461

26.2.3 Zufällige Gleitkommazahlen erzeugen ... 462

26.2.4 Zufallsgesteuerte Operationen auf Sequenzen ... 463

26.2.5 SystemRandom([seed]) ... 464

26.3 Präzise Dezimalzahlen – decimal ... 464

26.3.1 Verwendung des Datentyps ... 465

26.3.2 Nichtnumerische Werte ... 468

26.3.3 Das Context-Objekt ... 469

27 Kryptografie 471

27.1 Hash-Funktionen – hashlib ... 471

27.1.1 Verwendung des Moduls ... 473

27.1.2 Weitere Algorithmen ... 474
27.1.3 Vergleich großer Dateien ... 474
27.1.4 Passwörter ... 475

27.2 Verschlüsselung – PyCrypto ... 476
27.2.1 Symmetrische Verschlüsselungsverfahren ... 477
27.2.2 Asymmetrische Verschlüsselungsverfahren ... 480

28 Reguläre Ausdrücke 485

28.1 Syntax regulärer Ausdrücke ... 485
28.1.1 Beliebige Zeichen ... 486
28.1.2 Zeichenklassen ... 486
28.1.3 Quantoren ... 487
28.1.4 Vordefinierte Zeichenklassen ... 489
28.1.5 Weitere Sonderzeichen ... 491
28.1.6 Genügsame Quantoren ... 492
28.1.7 Gruppen ... 493
28.1.8 Alternativen ... 493
28.1.9 Extensions ... 494

28.2 Verwendung des Moduls ... 496
28.2.1 Searching ... 497
28.2.2 Matching ... 498
28.2.3 Einen String aufspalten ... 498
28.2.4 Teile eines Strings ersetzen ... 498
28.2.5 Problematische Zeichen ersetzen ... 499
28.2.6 Einen regulären Ausdruck kompilieren ... 500
28.2.7 Flags ... 500
28.2.8 Das Match-Objekt ... 502

28.3 Ein einfaches Beispielprogramm – Searching ... 503

28.4 Ein komplexeres Beispielprogramm – Matching ... 504

29 Schnittstelle zu Betriebssystem und Laufzeitumgebung 509

29.1 Funktionen des Betriebssystems – os ... 509
29.1.1 environ ... 510

29.1.2 getpid() ... 510
29.1.3 cpu_count() ... 510
29.1.4 system(cmd) ... 511
29.1.5 popen(command, [mode, buffering]) ... 511

29.2 Zugriff auf die Laufzeitumgebung – sys ... 512
29.2.1 Kommandozeilenparameter ... 512
29.2.2 Standardpfade ... 512
29.2.3 Standard-Ein-/Ausgabeströme ... 513
29.2.4 Das Programm beenden ... 513
29.2.5 Details zur Python-Version ... 513
29.2.6 Details zum Betriebssystem ... 514
29.2.7 Hooks ... 516

30 Kommandozeilenparameter 519

30.1 Taschenrechner – ein einfaches Beispiel ... 520

30.2 Ein weiteres Beispiel ... 524

31 Dateisystem 527

31.1 Zugriff auf das Dateisystem mit os ... 527

31.2 Dateipfade – os.path ... 534

31.3 Zugriff auf das Dateisystem – shutil ... 538
31.3.1 Verzeichnis- und Dateioperationen ... 540
31.3.2 Archivoperationen ... 542

31.4 Temporäre Dateien – tempfile ... 544

32 Parallele Programmierung 547

32.1 Prozesse, Multitasking und Threads ... 547
32.1.1 Die Leichtgewichte unter den Prozessen – Threads ... 548
32.1.2 Threads oder Prozesse? ... 550

32.2 Pythons Schnittstellen zur Parallelisierung ... 550

32.3 Parallelisierung von Funktionsaufrufen ... 551
32.3.1 Ein Beispiel mit einem futures.ThreadPoolExecutor ... 552
32.3.2 Executor-Instanzen als Kontext-Manager ... 554
32.3.3 Die Verwendung von futures.ProcessPoolExecutor ... 555
32.3.4 Die Verwaltung der Aufgaben eines Executors ... 556
32.4 Die Module threading und multiprocessing ... 562
32.5 Die Thread-Unterstützung in Python ... 563
32.5.1 Kritische Bereiche mit Lock-Objekten absichern ... 565
32.5.2 Datenaustausch zwischen Threads mit Critical Sections ... 567
32.5.3 Gefahren von Critical Sections – Deadlocks ... 571
32.6 Einblick in das Modul multiprocessing ... 572
32.7 Ausblick ... 574

33 Datenspeicherung 575

33.1 Komprimierte Dateien lesen und schreiben – gzip ... 575
33.2 XML ... 577
33.2.1 ElementTree ... 579
33.2.2 SAX – Simple API for XML ... 586
33.3 Datenbanken ... 590
33.3.1 Pythons eingebaute Datenbank – sqlite3 ... 594
33.4 Serialisierung von Instanzen – pickle ... 611
33.4.1 Funktionale Schnittstelle ... 612
33.4.2 Objektorientierte Schnittstelle ... 613
33.5 Das Datenaustauschformat JSON – json ... 614
33.6 Das Tabellenformat CSV – csv ... 615
33.6.1 reader-Objekte – Daten aus einer CSV-Datei lesen ... 616
33.6.2 Dialect-Objekte – eigene Dialekte verwenden ... 619

34 Netzwerkkommunikation 623

34.1 Socket API ... 624
34.1.1 Client-Server-Systeme ... 625
34.1.2 UDP ... 628
34.1.3 TCP ... 630

34.1.4 Blockierende und nicht-blockierende Sockets ... 632
34.1.5 Erzeugen eines Sockets ... 633
34.1.6 Die Socket-Klasse ... 634
34.1.7 Netzwerk-Byte-Order ... 638
34.1.8 Multiplexende Server – selectors ... 639
34.1.9 Objektorientierte Serverentwicklung – socketserver ... 641

34.2 URLs – urllib ... 643
34.2.1 Zugriff auf entfernte Ressourcen – urllib.request ... 644
34.2.2 Einlesen und Verarbeiten von URLs – urllib.parse ... 648

34.3 FTP – ftplib ... 652
34.3.1 Mit einem FTP-Server verbinden ... 653
34.3.2 FTP-Kommandos ausführen ... 654
34.3.3 Mit Dateien und Verzeichnissen arbeiten ... 654
34.3.4 Übertragen von Dateien ... 655

34.4 E-Mail ... 659
34.4.1 SMTP – smtplib ... 659
34.4.2 POP3 – poplib ... 662
34.4.3 IMAP4 – imaplib ... 667
34.4.4 Erstellen komplexer E-Mails – email ... 672

34.5 Telnet – telnetlib ... 677
34.5.1 Die Klasse Telnet ... 677
34.5.2 Beispiel ... 678

34.6 XML-RPC ... 680
34.6.1 Der Server ... 681
34.6.2 Der Client ... 685
34.6.3 Multicall ... 687
34.6.4 Einschränkungen ... 688

35 Debugging und Qualitätssicherung ... 691

35.1 Der Debugger ... 691

35.2 Formatierte Bildschirmausgabe – pprint ... 694

35.3 Logdateien – logging ... 696
35.3.1 Das Meldungsformat anpassen ... 698
35.3.2 Logging Handler ... 700

35.4 Automatisiertes Testen ... 702
35.4.1 Testfälle in Docstrings – doctest ... 702

35.4.2 Unit Tests – unittest ... 706

35.5 Analyse des Laufzeitverhaltens ... 710

35.5.1 Laufzeitmessung – timeit ... 710

35.5.2 Profiling – cProfile ... 713

35.5.3 Tracing – trace ... 717

35.6 Optimierung ... 720

35.6.1 Die Optimize-Option ... 721

35.6.2 Mutabel vs. immutabel ... 721

35.6.3 Schleifen ... 722

35.6.4 Funktionsaufrufe ... 723

35.6.5 C ... 723

35.6.6 Lookups ... 723

35.6.7 Exceptions ... 724

35.6.8 Keyword Arguments ... 724

35.6.9 Alternative Interpreter: PyPy ... 725

36 Dokumentation 727

36.1 Docstrings ... 727

36.2 Automatisches Erstellen einer Dokumentation – pydoc ... 729

TEIL V Weiterführende Themen

37 Anbindung an andere Programmiersprachen 733

37.1 Dynamisch ladbare Bibliotheken – ctypes ... 734

37.1.1 Ein einfaches Beispiel ... 734

37.1.2 Die eigene Bibliothek ... 735

37.1.3 Datentypen ... 737

37.1.4 Schnittstellenbeschreibung ... 739

37.1.5 Pointer ... 741

37.1.6 Strings ... 742

37.2 Schreiben von Extensions ... 742

37.2.1 Ein einfaches Beispiel ... 743

37.2.2 Exceptions ... 748
37.2.3 Erzeugen der Extension ... 749
37.2.4 Reference Counting ... 750

37.3 Python als eingebettete Skriptsprache ... 752
37.3.1 Ein einfaches Beispiel ... 752
37.3.2 Ein komplexeres Beispiel ... 754

37.4 Alternative Interpreter ... 757
37.4.1 Interoperabilität mit der Java Runtime Environment – Jython ... 758
37.4.2 Interoperabilität mit .NET – IronPython ... 763

38 Distribution von Python-Projekten 769

38.1 Eine Geschichte der Distributionen in Python ... 769
38.1.1 Der klassische Ansatz – distutils ... 770
38.1.2 Der neue Standard – setuptools ... 770
38.1.3 Der Paketindex – PyPI und pip ... 771

38.2 Erstellen von Distributionen – setuptools ... 771
38.2.1 Schreiben des Moduls ... 772
38.2.2 Das Installationsskript ... 773
38.2.3 Erstellen einer Quellcodedistribution ... 778
38.2.4 Erstellen einer Binärdistribution ... 778
38.2.5 Distributionen installieren ... 780
38.2.6 Eigenständige Distributionen erstellen ... 780
38.2.7 Erstellen von EXE-Dateien – cx_Freeze ... 781

38.3 Der Python-Paketmanager – pip ... 782

38.4 Der Paketmanager conda ... 783

38.5 Lokalisierung von Programmen – gettext ... 786
38.5.1 Beispiel für die Verwendung von gettext ... 787
38.5.2 Erstellen des Sprachkompilats ... 789

39 Grafische Benutzeroberflächen 791

39.1 Toolkits ... 791

39.2 Einführung in tkinter ... 794
39.2.1 Ein einfaches Beispiel ... 794

39.2.2 Steuerelementvariablen ... 796
39.2.3 Der Packer ... 798
39.2.4 Events ... 802
39.2.5 Steuerelemente ... 809
39.2.6 Zeichnungen – das Canvas-Widget ... 828
39.2.7 Weitere Module ... 836

39.3 Einführung in PyQt ... 839
39.3.1 Installation ... 840
39.3.2 Grundlegende Konzepte von Qt ... 841
39.3.3 Entwicklungsprozess ... 843

39.4 Signale und Slots ... 849

39.5 Wichtige Widgets ... 853
39.5.1 QCheckBox ... 853
39.5.2 QComboBox ... 853
39.5.3 QDateEdit, QTimeEdit, QDateTimeEdit ... 854
39.5.4 QDialog ... 855
39.5.5 QLineEdit ... 856
39.5.6 QListWidget, QListView ... 856
39.5.7 QProgressBar ... 857
39.5.8 QPushButton ... 857
39.5.9 QRadioButton ... 857
39.5.10 QSlider, QDial ... 858
39.5.11 QTextEdit ... 859
39.5.12 QWidget ... 859

39.6 Zeichenfunktionalität ... 860
39.6.1 Werkzeuge ... 861
39.6.2 Koordinatensystem ... 863
39.6.3 Einfache Formen ... 863
39.6.4 Grafiken ... 866
39.6.5 Text ... 867
39.6.6 Eye Candy ... 868

39.7 Model-View-Architektur ... 872
39.7.1 Beispielprojekt: ein Adressbuch ... 873
39.7.2 Auswählen von Einträgen ... 882
39.7.3 Bearbeiten von Einträgen ... 884

40 Python als serverseitige Programmiersprache im WWW – ein Einstieg in Django 889

40.1 Konzepte und Besonderheiten von Django ... 890

40.2 Installation von Django ... 891

40.2.1 Installation mit Anaconda ... 892

40.2.2 Für Leser, die Anaconda nicht verwenden ... 892

40.3 Erstellen eines neuen Django-Projekts ... 894

40.3.1 Der Entwicklungswebserver ... 895

40.3.2 Konfiguration des Projekts ... 896

40.4 Erstellung einer Applikation ... 898

40.4.1 Die Applikation in das Projekt einbinden ... 899

40.4.2 Ein Model definieren ... 900

40.4.3 Beziehungen zwischen Modellen ... 901

40.4.4 Übertragung des Modells in die Datenbank ... 901

40.4.5 Das Model-API ... 903

40.4.6 Unser Projekt bekommt ein Gesicht ... 909

40.4.7 Djangos Template-System ... 916

40.4.8 Verarbeitung von Formulardaten ... 929

40.4.9 Djangos Administrationsoberfläche ... 932

41 Wissenschaftliches Rechnen 939

41.1 Installation ... 940

41.2 Das Modellprogramm ... 941

41.2.1 Der Import von numpy, scipy und matplotlib ... 942

41.2.2 Vektorisierung und der Datentyp numpy.ndarray ... 943

41.2.3 Visualisieren von Daten mit matplotlib.pyplot ... 946

41.3 Überblick über die Module numpy und scipy ... 949

41.3.1 Überblick über den Datentyp numpy.ndarray ... 949

41.3.2 Überblick über scipy ... 957

42 Insiderwissen 961

42.1 URLs im Standardbrowser öffnen – webbrowser ... 961

42.2 Interpretieren von Binärdaten – struct ... 961

42.3 Versteckte Passworteingabe ... 964

42.4 Kommandozeilen-Interpreter ... 965

42.5 Dateiinterface für Strings – io.StringIO ... 967

42.6 Generatoren als Konsumenten ... 968

42.6.1 Ein Decorator für konsumierende Generatorfunktionen ... 970

42.6.2 Auslösen von Exceptions in einem Generator ... 971

42.6.3 Eine Pipeline als Verkettung konsumierender Generatorfunktionen ... 972

42.7 Kopieren von Instanzen – copy ... 973

42.8 Die interaktive Python-Shell – IPython ... 977

42.8.1 Die interaktive Shell ... 977

42.9 Das Jupyter Notebook ... 980

42.10 Bildverarbeitung – Pillow ... 983

42.10.1 Bilddateien laden und speichern ... 984

42.10.2 Zugriff auf einzelne Pixel ... 985

42.10.3 Teilbereiche eines Bildes ausschneiden ... 986

42.10.4 Bilder zusammenfügen ... 986

42.10.5 Geometrische Bildtransformationen ... 987

42.10.6 Vordefinierte Bildfilter ... 989

42.10.7 Eigene Pixeloperationen ... 990

42.10.8 Bildverbesserungen ... 991

42.10.9 Zeichenoperationen ... 991

42.10.10 Interoperabilität ... 993

43 Von Python 2 nach Python 3 995

43.1 Die wichtigsten Unterschiede ... 998

43.1.1 Ein-/Ausgabe ... 998

43.1.2 Iteratoren ... 999

43.1.3 Strings ... 1000

43.1.4 Ganze Zahlen ... 1001

43.1.5 Exception Handling ... 1002

43.1.6 Standardbibliothek 1002
43.1.7 Neue Sprachelemente in Python 3 1003
43.2 Automatische Konvertierung 1004
43.3 Geplante Sprachelemente 1007

Anhang 1009

A.1 Reservierte Wörter **1009**
A.2 Eingebaute Funktionen **1009**
A.3 Eingebaute Exceptions **1013**
A.4 Python IDEs **1017**

Index 1025