



Farid Hajji

Das Python Praxisbuch

Der große Profi-Leitfaden für Programmierer



Inhaltsverzeichnis

| | |
|--|-------------|
| Einführung | xvii |
| | |
| Teil I Die Python-Umgebung | 1 |
| | |
| 1 Python installieren | 3 |
| 1.1 Python auf Unix | 3 |
| 1.1.1 Python ohne root-Rechte installieren | 5 |
| 1.2 Python auf Windows | 6 |
| 1.3 Mehrere Python unter einem Dach | 9 |
| 1.4 Drittanbietermodule installieren | 11 |
| 1.4.1 Einfache .py Dateien | 11 |
| 1.4.2 distutils-kompatible Packages | 12 |
| 1.4.3 Packages ohne root-Rechte installieren | 12 |
| 1.5 setuptools, Eggs und ein Käse-Shop | 14 |
| 1.5.1 setuptools installieren | 14 |
| 1.5.2 Ein Käse-Shop mit vielen Leckereien | 20 |
| 1.5.3 ZODB3 mit easy_install hinzufügen | 20 |
| 1.5.4 easy_install benutzen | 23 |
| 1.5.5 Was sind Eggs? | 24 |
| 1.6 Zusammenfassung | 27 |
| | |
| 2 Die Python-Shell | 29 |
| 2.1 Die Qual der (Python-Shell-) Wahl | 29 |
| 2.1.1 Python-Shells unter Unix | 30 |
| 2.1.2 Python-Shells unter Windows | 33 |
| 2.2 Ein Taschenrechner | 34 |

| | | |
|----------|---|------------|
| 2.3 | Auf Erkundungstour | 39 |
| 2.3.1 | Introspektion mit <code>dir</code> , <code>type</code> und <code>__doc__</code> | 40 |
| 2.3.2 | Das Hilfesystem <code>help</code> | 44 |
| 2.3.3 | Das Dokumentationstool <code>pydoc</code> | 52 |
| 2.4 | Zusammenfassung | 56 |
| 3 | Hello, World! | 59 |
| 3.1 | Das Hello, World!-Programm | 59 |
| 3.1.1 | <code>hello.py</code> verstehen | 60 |
| 3.1.2 | <code>hello.py</code> unter Unix ausführen | 61 |
| 3.1.3 | <code>hello.py</code> unter Windows ausführen | 67 |
| 3.1.4 | <code>hello.py</code> in IDLE ausführen | 68 |
| 3.2 | <code>hello2.py</code> mit <code>sys.argv</code> | 69 |
| 3.2.1 | <code>hello2.py</code> verstehen | 70 |
| 3.2.2 | <code>hello2.py</code> ausführen | 72 |
| 3.2.3 | <code>hello2.py</code> unterm Debugger | 73 |
| 3.3 | <code>tkhello.py</code> mit Tkinter | 77 |
| 3.4 | Zusammenfassung | 84 |
| | Teil II Python-Grundlagen | 87 |
| 4 | Zahlen | 89 |
| 4.1 | Die Grundzahlentypen | 89 |
| 4.2 | Dezimalzahlen mit dem <code>decimal</code> -Modul | 98 |
| 4.3 | Zufallszahlen mit dem <code>random</code> -Modul | 107 |
| 4.3.1 | Zufallszahlen aus dem Mersenne Twister | 109 |
| 4.3.2 | Zufallszahlen aus einer Entropiequelle | 113 |
| 4.4 | Zusammenfassung | 115 |
| 5 | Strings | 117 |
| 5.1 | Einfache Bytestrings | 117 |
| 5.1.1 | String-Literale | 118 |
| 5.1.2 | String Interpolation | 119 |
| 5.1.3 | String Slices | 123 |
| 5.1.4 | String-Operatoren | 124 |
| 5.1.5 | String-Methoden | 125 |
| 5.2 | Unicode-Strings | 127 |
| 5.2.1 | Warum Unicode? | 127 |

- 5.2.2 Unicode-Encodings 128
- 5.2.3 Der unicode-Datentyp 129
- 5.2.4 Codieren und Decodieren von Unicode 130
- 5.2.5 Codecs 132
- 5.3 Reguläre Ausdrücke 134
 - 5.3.1 Was sind reguläre Ausdrücke? 134
 - 5.3.2 re.search und re.sub 135
 - 5.3.3 Kompilierte reguläre Ausdrücke 137
 - 5.3.4 Das Match-Objekt 139
 - 5.3.5 Die Flags 140
 - 5.3.6 findall und finditer 141
- 5.4 Anwendungen 142
 - 5.4.1 Suchen in Strings 142
 - 5.4.2 Strings effizient aufbauen 143
- 5.5 Zusammenfassung 145

- 6 Listen 149**
 - 6.1 Zugriff auf Listenelemente 149
 - 6.2 Listen-Slices 152
 - 6.3 Memberfunktionen von Listen 153
 - 6.4 Built-in-Funktionen für Listen 160
 - 6.5 Anwendungen 164
 - 6.5.1 Listen durchlaufen 164
 - 6.5.2 Listen kopieren und vergleichen 166
 - 6.5.3 Listen sortieren 170
 - 6.5.4 List comprehensions 175
 - 6.5.5 Das DSU-Idiom 177
 - 6.5.6 Stabiles Sortieren 178
 - 6.6 Zusammenfassung 180

- 7 Dictionarys 183**
 - 7.1 Dictionary-Literale 183
 - 7.2 Dictionarys abfragen und verändern 186
 - 7.2.1 Was tun bei nicht-existierenden Einträgen? 189
 - 7.2.2 Wie verändert man Dictionarys? 190
 - 7.2.3 Dictionarys mit Default-Initialwerten (setdefault) 194
 - 7.3 Dictionarys durchlaufen 196
 - 7.3.1 Dictionarys effizient durchlaufen 200

| | | |
|----------|--|------------|
| 7.4 | Dictionaries kopieren | 202 |
| 7.5 | Anwendung: word count | 205 |
| 7.6 | Iteratoren und Generatoren | 211 |
| 7.6.1 | Was ist ein Iterator? | 212 |
| 7.6.2 | Einen eigenen Iterator schreiben | 213 |
| 7.6.3 | Generatoren | 216 |
| 7.7 | Zusammenfassung | 221 |
| 8 | Funktionen | 223 |
| 8.1 | Funktionen aufrufen | 224 |
| 8.2 | Funktionsparameter | 226 |
| 8.3 | Rückgabewerte | 230 |
| 8.4 | Scope | 230 |
| 8.4.1 | Lokale und globale Namensräume verstehen | 232 |
| 8.5 | Ein Blick unter die Haube | 234 |
| 8.6 | Factory-Funktionen und -Closures | 240 |
| 8.7 | Dekoratoren | 241 |
| 8.7.1 | Dekoratoren manuell erstellen | 242 |
| 8.7.2 | Das decorator-Modul | 248 |
| 8.8 | Zusammenfassung | 257 |
| 9 | Dateien und das Dateisystem | 259 |
| 9.1 | Dateien | 259 |
| 9.1.1 | Die Funktion open | 260 |
| 9.1.2 | Die Funktion close | 266 |
| 9.1.3 | Textdateien lesen und schreiben | 268 |
| 9.1.4 | Binärdateien lesen und schreiben | 275 |
| 9.1.5 | RAM-Dateien mit StringIO | 283 |
| 9.1.6 | Memory mapped Dateien mit mmap | 286 |
| 9.1.7 | Spezielle Dateiformate | 291 |
| 9.2 | Das Dateisystem | 300 |
| 9.2.1 | Dateien verschieben oder löschen | 301 |
| 9.2.2 | Metadaten einer Datei | 301 |
| 9.2.3 | Das Dateisystem durchlaufen | 307 |
| 9.2.4 | Das shutil-Modul | 315 |
| 9.3 | Zusammenfassung | 316 |

| | | |
|-----------|--|------------|
| 10 | Klassen und Objekte | 319 |
| 10.1 | Hello, OO-World! | 319 |
| 10.2 | Klassen verwenden | 320 |
| 10.2.1 | Objekte instanziiieren | 320 |
| 10.2.2 | Objektattribute | 322 |
| 10.2.3 | Objektmethoden (Memberfunktionen) | 324 |
| 10.3 | Klassen schreiben | 327 |
| 10.3.1 | Klassenmethoden und statische Methoden | 333 |
| 10.3.2 | Klassenvererbung | 336 |
| 10.4 | Hooks | 339 |
| 10.4.1 | Eine Einführung in Hooks | 339 |
| 10.4.2 | Eine Tour der object-Hooks | 340 |
| 10.4.3 | Ein Dictionary mit case-insensitive Schlüsseln | 344 |
| 10.4.4 | Ein Dictionary mit Default-Werten | 346 |
| 10.4.5 | Ein aufrufbares Objekt | 348 |
| 10.4.6 | Propertyts | 350 |
| 10.4.7 | Deskriptoren | 353 |
| 10.4.8 | <code>__slots__</code> | 360 |
| 10.5 | Metaklassen | 364 |
| 10.5.1 | Klassen sind Instanzen von Metaklassen | 364 |
| 10.5.2 | Das <code>__metaclass__</code> -Attribut | 368 |
| 10.5.3 | Anwendungen von Metaklassen | 370 |
| 10.6 | Zusammenfassung | 374 |
| 11 | Python und C/C++ | 377 |
| 11.1 | ctypes | 378 |
| 11.1.1 | Die ctypes-Datentypwrapper | 379 |
| 11.1.2 | Komplexe Datentypen | 382 |
| 11.1.3 | C-Funktionen aufrufen | 387 |
| 11.2 | SWIG | 400 |
| 11.2.1 | SWIG installieren | 401 |
| 11.2.2 | SWIG aufrufen | 403 |
| 11.2.3 | Konstanten und Variablen | 408 |
| 11.2.4 | Stringmanipulationen | 416 |
| 11.2.5 | Strukturen | 419 |
| 11.2.6 | C++-Klassen | 422 |

| | | |
|--------|--|-----|
| 11.2.7 | Unbekannte Datentypen sind Pointer | 426 |
| 11.2.8 | Wie geht's von hier aus weiter? | 430 |
| 11.3 | Boost.Python | 430 |
| 11.4 | Low-level Python/C-API | 431 |
| 11.5 | Zusammenfassung | 432 |

Teil III Anwendungen

435

| | | |
|-----------|--|------------|
| 12 | XML und XSLT | 437 |
| 12.1 | Eine XML-Datei | 438 |
| 12.2 | xml.etree.ElementTree | 440 |
| 12.3 | 4Suite-XML | 447 |
| 12.3.1 | 4Suite-XML installieren | 448 |
| 12.3.2 | Die 4Suite-XML-Skripte | 450 |
| 12.3.3 | Ft.Xml.InputSource-Eingabequellen | 456 |
| 12.3.4 | DOM | 457 |
| 12.3.5 | SAX | 466 |
| 12.4 | Transformationen mit XSLT | 469 |
| 12.5 | Zusammenfassung | 471 |
| 13 | Persistenz und Datenbanken | 473 |
| 13.1 | Serialisieren und Deserialisieren | 474 |
| 13.1.1 | Ein naiver Versuch mit str und eval | 474 |
| 13.1.2 | Die richtige Lösung mit pickle | 476 |
| 13.2 | Persistente Dictionaries mit anydbm | 479 |
| 13.2.1 | Eine naive suboptimale Lösung | 480 |
| 13.2.2 | Die richtige Lösung mit anydbm | 481 |
| 13.2.3 | Besonderheiten von anydbm-Dictionaries | 483 |
| 13.2.4 | Die anydbm-Architektur | 485 |
| 13.3 | Mehr Flexibilität mit bsddb | 486 |
| 13.4 | Persistente Datenstrukturen mit shelve | 490 |
| 13.4.1 | Eine umständliche Lösung | 491 |
| 13.4.2 | Die shelve-Lösung | 493 |
| 13.5 | Die ZODB-objektorientierte Datenbank | 498 |
| 13.5.1 | ZODB installieren | 498 |
| 13.5.2 | Die ZODB benutzen | 499 |
| 13.6 | Ein Blogs-Backend mit ZODB | 508 |

| | | |
|-----------|--|------------|
| 13.6.1 | Comment, Article und Blog | 508 |
| 13.6.2 | Das Blog-Backend BlogDB | 513 |
| 13.7 | DB-API 2.0 SQL-Anbindungen | 517 |
| 13.7.1 | Eine kurze DB-API 2.0-Einführung | 518 |
| 13.8 | SQLite-Anbindung mit sqlite3 | 521 |
| 13.8.1 | sqlite3 benutzen | 522 |
| 13.8.2 | Das sqlite3-Tool | 522 |
| 13.8.3 | Das sqlite3-Modul | 526 |
| 13.8.4 | Anwendung: Verwaltung von MP3-Metadaten | 531 |
| 13.9 | PostgreSQL-Anbindung mit psycopg2 | 540 |
| 13.9.1 | Was ist PostgreSQL? | 540 |
| 13.9.2 | PostgreSQL installieren | 541 |
| 13.9.3 | psycopg2 installieren | 555 |
| 13.9.4 | psycopg2 benutzen | 559 |
| 13.9.5 | Anwendung: MP3-Metadaten unter PostgreSQL | 563 |
| 13.10 | MySQL-Anbindung mit MySQLdb | 566 |
| 13.10.1 | MySQL-Datenbank vorbereiten | 566 |
| 13.10.2 | MySQL-python installieren | 569 |
| 13.10.3 | MySQL-python benutzen | 571 |
| 13.10.4 | Anwendung: MP3-Metadaten unter MySQL | 574 |
| 13.11 | Der objektrelationale Mapper SQLAlchemy | 577 |
| 13.11.1 | Was sind objektrelationale Mapper? | 577 |
| 13.11.2 | SQLAlchemy installieren | 578 |
| 13.11.3 | SQLAlchemy benutzen | 580 |
| 13.11.4 | Das Blog-System mit SQLAlchemy | 588 |
| 13.11.5 | django.db, ein anderer ORM | 594 |
| 13.12 | Zusammenfassung | 595 |
| 14 | Netzwerkprogrammierung | 599 |
| 14.1 | Das Twisted Framework | 600 |
| 14.1.1 | Twisted installieren | 601 |
| 14.1.2 | Erste Schritte ins Twisted-Universum | 603 |
| 14.1.3 | Zeilenpufferung und ein einfacher Dialog | 613 |
| 14.1.4 | Anwendung: Ein Chat-Server | 620 |
| 14.1.5 | Deferred oder: Wenn ein Ergebnis auf sich warten lässt | 625 |
| 14.1.6 | Passwort-geschützte Bereiche mit cred | 635 |
| 14.1.7 | Twisted AMP (Asynchronous Messaging Protocol) | 646 |

| | | |
|-----------|---|------------|
| 14.1.8 | Ein Schnelldurchlauf durch die Twisted-Protokolle | 651 |
| 14.1.9 | Twisted und Datenbanken | 680 |
| 14.1.10 | Wie findet man sich in Twisted zurecht? | 680 |
| 14.2 | Module der Python Standard Library | 683 |
| 14.2.1 | Server mit SocketServer schreiben | 683 |
| 14.2.2 | FTP mit ftplib.FTP | 690 |
| 14.2.3 | Mails senden mit smtplib.SMTP | 692 |
| 14.2.4 | Mails abholen mit poplib.POP3 und imaplib.IMAP4 | 694 |
| 14.2.5 | Newsreader mit nntplib.NNTP | 697 |
| 14.2.6 | Telnet-Clients mit telnetlib.Telnet | 699 |
| 14.3 | Verteilte Programme | 700 |
| 14.3.1 | Twisted Perspective Broker | 701 |
| 14.3.2 | XML-RPC | 711 |
| 14.3.3 | asyncore/asynchat | 722 |
| 14.4 | Low-level-Programmierung mit Sockets | 730 |
| 14.5 | Zusammenfassung | 742 |
| 15 | Webprogrammierung und Web-Frameworks | 747 |
| 15.1 | Webserver in Python | 754 |
| 15.1.1 | Webserver aus der Python Standard Library | 754 |
| 15.1.2 | Webserver aus Drittanbietermodulen | 764 |
| 15.2 | Integration mit anderen Webservern | 800 |
| 15.2.1 | Lighttpd | 803 |
| 15.2.2 | Apache | 816 |
| 15.3 | WSGI | 832 |
| 15.3.1 | Was ist WSGI? | 833 |
| 15.3.2 | WSGI-Tools | 839 |
| 15.4 | Low-level-Programmierung mit CGI | 841 |
| 15.4.1 | Hello, CGI World! | 841 |
| 15.4.2 | CGI-Umgebungsvariablen | 842 |
| 15.4.3 | Anwendung: Ein Web-Taschenrechner | 844 |
| 15.4.4 | Ein Formular manuell auslesen | 847 |
| 15.4.5 | Das cgi-Modul | 851 |
| 15.4.6 | Anwendung: Dateien uploaden | 854 |
| 15.4.7 | Den Zustand clientseitig erhalten | 864 |
| 15.4.8 | Den Zustand serverseitig erhalten | 896 |
| 15.4.9 | Nachteile von CGI | 907 |

| | | |
|-----------|---|-------------|
| 15.5 | Webclients | 908 |
| 15.5.1 | Low-level HTTP Protokoll mit httplib | 909 |
| 15.5.2 | Einfache Webclients mit urllib | 914 |
| 15.5.3 | Flexiblere Webclients mit urllib2 | 915 |
| 15.5.4 | Webclients mit Twisted | 920 |
| 15.6 | Templating Engines | 921 |
| 15.6.1 | Templating für arme Leute | 922 |
| 15.6.2 | Text-basiertes Templating | 932 |
| 15.6.3 | XML-basiertes Templating | 961 |
| 15.6.4 | Weitere Templating-Systeme | 981 |
| 15.7 | Web-Frameworks | 982 |
| 15.7.1 | Django | 982 |
| 15.7.2 | Weitere Web-Frameworks | 1005 |
| 15.8 | Zope, Plone et al. | 1006 |
| 15.8.1 | Zope und Plone zusammen installieren | 1007 |
| 15.8.2 | Erste Schritte in Zope | 1008 |
| 15.8.3 | Macros in Plone | 1017 |
| 15.9 | Wikis | 1022 |
| 15.10 | Lose Enden | 1023 |
| 15.11 | Zusammenfassung | 1023 |
| 16 | GUI-Toolkits | 1031 |
| 16.1 | wxPython | 1035 |
| 16.1.1 | wxPython installieren | 1035 |
| 16.1.2 | Erste Schritte in wxPython | 1037 |
| 16.1.3 | Responsive GUIs | 1060 |
| 16.1.4 | Schnelle Entwicklung mit RAD-Tools | 1101 |
| 16.2 | PyQt4 | 1111 |
| 16.2.1 | PyQt4 installieren | 1112 |
| 16.2.2 | Erste Schritte in PyQt4 | 1118 |
| 16.2.3 | Responsive GUIs | 1126 |
| 16.2.4 | Schnelle Entwicklung mit RAD-Tools | 1134 |
| 16.2.5 | eric4, eine PyQt4-IDE | 1142 |
| 16.3 | Tkinter | 1143 |
| 16.3.1 | Erste Schritte mit Tkinter | 1144 |
| 16.3.2 | Wie findet man sich in Tkinter zurecht? | 1146 |

| | | |
|-----------|--|-------------|
| 16.4 | Text-basierte GUIs | 1146 |
| 16.4.1 | pythondialog | 1148 |
| 16.4.2 | Weitere NGUI | 1152 |
| 16.5 | Low-level APIs | 1153 |
| 16.5.1 | Ein Blick unter die Haube: Rohe Events | 1153 |
| 16.5.2 | GUI-Toolkits vereinfachen eine komplexe API | 1156 |
| 16.6 | Zusammenfassung | 1158 |
| 17 | Python für Wissenschaftler | 1163 |
| 17.1 | Das Computer Algebra System sympy | 1163 |
| 17.1.1 | sympy installieren | 1164 |
| 17.1.2 | Die Datentypen Rational, Real und Integer | 1166 |
| 17.1.3 | Ein paar Konstanten | 1173 |
| 17.1.4 | Ausdrücke | 1175 |
| 17.1.5 | Differenzial- und Integralrechnung | 1181 |
| 17.1.6 | Polynome | 1187 |
| 17.1.7 | Reihen | 1192 |
| 17.1.8 | Gleichungen lösen | 1196 |
| 17.1.9 | Pattern Matching | 1199 |
| 17.1.10 | Lineare Algebra | 1201 |
| 17.1.11 | Plotting | 1219 |
| 17.2 | Effiziente numerische Berechnungen mit numpy und scipy | 1220 |
| 17.2.1 | numpy | 1222 |
| 17.2.2 | scipy | 1244 |
| 17.3 | Plotten mit pylab (a.k.a. matplotlib) | 1249 |
| 17.4 | Zusammenfassung | 1253 |
| | Stichwortverzeichnis | 1255 |