

# Auf einen Blick

TEIL I Einstieg in Python .....	37
TEIL II Datentypen .....	113
TEIL III Fortgeschrittene Programmiertechniken .....	259
TEIL IV Die Standardbibliothek .....	449
TEIL V Weiterführende Themen .....	727

# Inhalt

---

<b>1</b>	<b>Einleitung</b>	25
<b>2</b>	<b>Die Programmiersprache Python</b>	31
<b>2.1</b>	<b>Historie, Konzepte, Einsatzgebiete</b>	31
2.1.1	Geschichte und Entstehung .....	31
2.1.2	Grundlegende Konzepte .....	32
2.1.3	Einsatzmöglichkeiten und Stärken .....	33
2.1.4	Einsatzbeispiele .....	34
<b>2.2</b>	<b>Die Verwendung von Python</b>	34
2.2.1	Windows .....	36
2.2.2	Linux .....	36
2.2.3	OS X .....	36

## TEIL I Einstieg in Python

---

<b>3</b>	<b>Erste Schritte im interaktiven Modus</b>	39
<b>3.1</b>	<b>Ganze Zahlen</b> .....	40
<b>3.2</b>	<b>Gleitkommazahlen</b> .....	41
<b>3.3</b>	<b>Zeichenketten</b> .....	42
<b>3.4</b>	<b>Listen</b> .....	42
<b>3.5</b>	<b>Dictionarys</b> .....	43
<b>3.6</b>	<b>Variablen</b> .....	44
<b>3.7</b>	<b>Logische Ausdrücke</b> .....	46
<b>3.8</b>	<b>Funktionen und Methoden</b> .....	47
3.8.1	Funktionen .....	48
3.8.2	Methoden .....	48
<b>3.9</b>	<b>Bildschirmausgaben</b> .....	49

---

<b>4</b>	<b>Der Weg zum ersten Programm</b>	53
<b>4.1</b>	<b>Tippen, kompilieren, testen</b> .....	53
4.1.1	Shebang .....	55
4.1.2	Interne Abläufe .....	55
<b>4.2</b>	<b>Grundstruktur eines Python-Programms</b> .....	57
4.2.1	Umbrechen langer Zeilen .....	59
4.2.2	Zusammenfügen mehrerer Zeilen .....	59
<b>4.3</b>	<b>Das erste Programm</b> .....	60
<b>4.4</b>	<b>Kommentare</b> .....	63
<b>4.5</b>	<b>Der Fehlerfall</b> .....	63
<b>5</b>	<b>Kontrollstrukturen</b>	65
<b>5.1</b>	<b>Fallunterscheidungen</b> .....	65
5.1.1	Die if-Anweisung .....	65
5.1.2	Bedingte Ausdrücke .....	69
<b>5.2</b>	<b>Schleifen</b> .....	70
5.2.1	Die while-Schleife .....	70
5.2.2	Abbruch einer Schleife .....	71
5.2.3	Erkennen eines Schleifenabbruchs .....	72
5.2.4	Abbruch eines Schleifendurchlaufs .....	73
5.2.5	Die for-Schleife .....	75
5.2.6	Die for-Schleife als Zählschleife .....	77
<b>5.3</b>	<b>Die pass-Anweisung</b> .....	78
<b>6</b>	<b>Dateien</b>	79
<b>6.1</b>	<b>Datenströme</b> .....	79
<b>6.2</b>	<b>Daten aus einer Datei auslesen</b> .....	80
<b>6.3</b>	<b>Daten in eine Datei schreiben</b> .....	84
<b>6.4</b>	<b>Das Dateiobjekt erzeugen</b> .....	85
6.4.1	open(filename, [mode, buffering, encoding, errors, newline]) .....	85
6.4.2	Attribute und Methoden eines Dateiobjekts .....	87
6.4.3	Die Schreib-/Leseposition verändern .....	88

---

<b>7 Das Laufzeitmodell</b>	91
<b>7.1 Die Struktur von Instanzen</b>	93
7.1.1 Datentyp .....	93
7.1.2 Wert .....	94
7.1.3 Identität .....	95
<b>7.2 Referenzen und Instanzen freigeben</b>	97
<b>7.3 Mutable vs. immutable Datentypen</b>	98
7.3.1 Mutable Datentypen und Seiteneffekte .....	100
<b>8 Funktionen, Methoden und Attribute</b>	103
<b>8.1 Parameter von Funktionen und Methoden</b>	103
8.1.1 Positionsbezogene Parameter .....	104
8.1.2 Schlüsselwortparameter .....	105
8.1.3 Optionale Parameter .....	105
8.1.4 Reine Schlüsselwortparameter .....	106
<b>8.2 Attribute</b> .....	106
<b>9 Informationsquellen zu Python</b>	109
<b>9.1 Die Built-in Function help</b> .....	109
<b>9.2 Die Onlinedokumentation</b> .....	110
<b>9.3 PEPs</b> .....	110
<b>TEIL II Datentypen</b>	
<b>10 Das Nichts – NoneType</b>	115
<b>11 Operatoren</b>	117

---

## 12 Numerische Datentypen

---

121

<b>12.1 Arithmetische Operatoren</b> .....	121
<b>12.2 Vergleichende Operatoren</b> .....	123
<b>12.3 Konvertierung zwischen numerischen Datentypen</b> .....	124
<b>12.4 Ganzzahlen – int</b> .....	125
12.4.1 Zahlensysteme .....	125
12.4.2 Bit-Operationen .....	127
12.4.3 Methoden .....	130
<b>12.5 Gleitkommazahlen – float</b> .....	130
<b>12.6 Boolesche Werte – bool</b> .....	133
12.6.1 Logische Operatoren .....	133
12.6.2 Wahrheitswerte nicht-boolescher Datentypen .....	136
12.6.3 Auswertung logischer Operatoren .....	137
<b>12.7 Komplexe Zahlen – complex</b> .....	138

## 13 Sequenzielle Datentypen

---

141

<b>13.1 Operationen auf Instanzen sequenzieller Datentypen</b> .....	142
13.1.1 Ist ein Element vorhanden? – die Operatoren in und not in .....	143
13.1.2 Verkettung von Sequenzen – die Operatoren + und += .....	145
13.1.3 Wiederholung von Sequenzen – die Operatoren * und *= .....	146
13.1.4 Zugriff auf bestimmte Elemente einer Sequenz – der []-Operator ....	147
13.1.5 Länge einer Sequenz – die Built-in Function len .....	151
13.1.6 Das kleinste und das größte Element einer Sequenz – min und max .....	152
13.1.7 Die Position eines Elements in der Sequenz – s.index(x, [i, j]) .....	152
13.1.8 Anzahl der Vorkommen eines Elements der Sequenz – s.count(x) ....	153
<b>13.2 Listen – list</b> .....	154
13.2.1 Verändern eines Wertes innerhalb der Liste – Zuweisung mit [] .....	155
13.2.2 Ersetzen von Teillisten und Einfügen neuer Elemente – Zuweisung mit [] .....	155
13.2.3 Elemente und Teillisten löschen – del zusammen mit [] .....	156
13.2.4 Methoden von list-Instanzen .....	156
13.2.5 Weitere Eigenschaften von Listen .....	163

<b>13.3 Unveränderliche Listen – tuple</b> .....	166
13.3.1 Tuple Packing/Unpacking und Sequence Unpacking .....	166
13.3.2 Immutabel heißt nicht zwingend unveränderlich! .....	168
<b>13.4 Strings – str, bytes, bytearray</b> .....	168
13.4.1 Steuerzeichen .....	171
13.4.2 String-Methoden .....	173
13.4.3 Formatierung von Strings .....	183
13.4.4 Zeichensätze und Sonderzeichen .....	192

---

## 14 Zuordnungen

---

<b>14.1 Dictionary – dict</b> .....	201
14.1.1 Operatoren .....	204
14.1.2 Methoden .....	206

---

## 15 Mengen

---

<b>15.1 Die Datentypen set und frozenset</b> .....	213
15.1.1 Operatoren .....	214
15.1.2 Methoden .....	220
<b>15.2 Veränderliche Mengen – set</b> .....	221
<b>15.3 Unveränderliche Mengen – frozenset</b> .....	223

---

## 16 Collections

---

<b>16.1 Verkettete Dictionarys</b> .....	225
<b>16.2 Zählen von Häufigkeiten</b> .....	226
<b>16.3 Dictionarys mit Standardwerten</b> .....	229
<b>16.4 Doppelt verkettete Listen</b> .....	230
<b>16.5 Benannte Tupel</b> .....	232
<b>16.6 Sortierte Dictionarys</b> .....	233

## 17 Datum und Zeit

235

---

<b>17.1 Elementare Zeitfunktionen – time</b> .....	235
17.1.1 Attribute .....	237
17.1.2 Funktionen .....	238
<b>17.2 Objektorientierte Datumsverwaltung – datetime</b> .....	243
17.2.1 datetime.date .....	244
17.2.2 datetime.time .....	245
17.2.3 datetime.datetime .....	246
17.2.4 datetime.timedelta .....	248
17.2.5 Operationen für datetime.datetime und datetime.date .....	251
17.2.6 Bemerkung zum Umgang mit Zeitzonen .....	253

## 18 Aufzählungstypen – Enum

255

---

## TEIL III Fortgeschrittene Programmietechniken

### 19 Funktionen

261

---

<b>19.1 Schreiben einer Funktion</b> .....	263
<b>19.2 Funktionsparameter</b> .....	267
19.2.1 Optionale Parameter .....	267
19.2.2 Schlüsselwortparameter .....	268
19.2.3 Beliebige Anzahl von Parametern .....	269
19.2.4 Reine Schlüsselwortparameter .....	271
19.2.5 Entpacken einer Parameterliste .....	272
19.2.6 Seiteneffekte .....	274
<b>19.3 Namensräume</b> .....	277
19.3.1 Zugriff auf globale Variablen – global .....	277
19.3.2 Zugriff auf den globalen Namensraum .....	278
19.3.3 Lokale Funktionen .....	279
19.3.4 Zugriff auf übergeordnete Namensräume – nonlocal .....	280
<b>19.4 Anonyme Funktionen</b> .....	281
<b>19.5 Annotationen</b> .....	282
<b>19.6 Rekursion</b> .....	284

<b>19.7 Eingebaute Funktionen .....</b>	<b>285</b>
19.7.1 abs(x) .....	288
19.7.2 all(iterable) .....	289
19.7.3 any(iterable) .....	289
19.7.4 ascii(object) .....	289
19.7.5 bin(x) .....	290
19.7.6 bool([x]) .....	290
19.7.7 bytearray([source, encoding, errors]) .....	290
19.7.8 bytes([source, encoding, errors]) .....	291
19.7.9 chr(i) .....	291
19.7.10 complex([real, imag]) .....	292
19.7.11 dict([source]) .....	292
19.7.12 divmod(a, b) .....	293
19.7.13 enumerate(iterable) .....	293
19.7.14 eval(expression, [globals, locals]) .....	294
19.7.15 exec(object, [globals, locals]) .....	294
19.7.16 filter(function, iterable) .....	295
19.7.17 float([x]) .....	295
19.7.18 format(value, [format_spec]) .....	296
19.7.19 frozenset([iterable]) .....	296
19.7.20 globals() .....	296
19.7.21 hash(object) .....	297
19.7.22 help([object]) .....	298
19.7.23 hex(x) .....	298
19.7.24 id(object) .....	298
19.7.25 input([prompt]) .....	298
19.7.26 int([x, base]) .....	299
19.7.27 len(s) .....	299
19.7.28 list([sequence]) .....	300
19.7.29 locals() .....	300
19.7.30 map(function, [*iterable]) .....	300
19.7.31 max(iterable, {default, key}) max(arg1, arg2, [*args], {key}) .....	302
19.7.32 min(iterable, {default, key}) min(arg1, arg2, [*args], {key}) .....	303
19.7.33 oct(x) .....	303
19.7.34 ord(c) .....	303
19.7.35 pow(x, y, [z]) .....	303
19.7.36 print([*objects], {sep, end, file, flush}) .....	304
19.7.37 range([start], stop, [step]) .....	304
19.7.38 repr(object) .....	305

---

19.7.39	reversed(sequence)	306
19.7.40	round(x, [n])	306
19.7.41	set([iterable])	306
19.7.42	sorted(iterable, [key, reverse])	307
19.7.43	str([object, encoding, errors])	307
19.7.44	sum(iterable, [start])	308
19.7.45	tuple([iterable])	309
19.7.46	type(object)	309
19.7.47	zip(*iterables)	309

## 20 Modularisierung

311

---

20.1	Einbinden globaler Module	311
20.2	Lokale Module	314
20.2.1	Namenskonflikte	315
20.2.2	Modulinterne Referenzen	316
20.2.3	Module ausführen	316
20.3	Pakete	317
20.3.1	Importieren aller Module eines Pakets	319
20.3.2	Namespace Packages	320
20.3.3	Relative Import-Anweisungen	320
20.4	Das Paket importlib	321
20.4.1	Einbinden von Modulen und Paketen	322
20.4.2	Verändern des Import-Verhaltens	322

## 21 Objektorientierung

327

---

21.1	Klassen	332
21.1.1	Definieren von Methoden	333
21.1.2	Der Konstruktor und die Erzeugung von Attributen	334
21.2	Vererbung	337
21.2.1	Technische Grundlagen	338
21.2.2	Die Klasse GirokontoMitTagesumsatz	341
21.2.3	Mögliche Erweiterungen der Klasse Konto	346
21.2.4	Ausblick	350
21.2.5	Mehrfachvererbung	351

<b>21.3</b>	<b>Setter und Getter und Property Attributes</b>	352
21.3.1	Setter und Getter .....	352
21.3.2	Property-Attribute .....	353
<b>21.4</b>	<b>Klassenattribute und Klassenmethoden sowie statische Methoden</b>	355
21.4.1	Statische Methoden .....	355
21.4.2	Klassenmethoden .....	356
21.4.3	Klassenattribute .....	357
<b>21.5</b>	<b>Built-in Functions für Objektorientierung</b>	358
21.5.1	Funktionen für die Verwaltung der Attribute einer Instanz .....	359
21.5.2	Funktionen für Informationen über die Klassenhierarchie .....	360
<b>21.6</b>	<b>Objektphilosophie</b>	361
<b>21.7</b>	<b>Magic Methods und Magic Attributes</b>	363
21.7.1	Allgemeine Magic Methods .....	364
21.7.2	Operatoren überladen .....	370
21.7.3	Datentypen emulieren .....	378

---

<b>22</b>	<b>Ausnahmebehandlung</b>	383
<b>22.1</b>	<b>Exceptions</b>	383
22.1.1	Eingebaute Exceptions .....	384
22.1.2	Werfen einer Exception .....	385
22.1.3	Abfangen einer Exception .....	386
22.1.4	Eigene Exceptions .....	390
22.1.5	Erneutes Werfen einer Exception .....	392
22.1.6	Exception Chaining .....	395
<b>22.2</b>	<b>Zusicherungen – assert</b>	396

---

<b>23</b>	<b>Iteratoren und Generatoren</b>	399
<b>23.1</b>	<b>Comprehensions</b>	399
23.1.1	List Comprehensions .....	399
23.1.2	Dict Comprehensions .....	402
23.1.3	Set Comprehensions .....	403
<b>23.2</b>	<b>Generatoren</b>	403
23.2.1	Subgeneratoren .....	406
23.2.2	Generator Expressions .....	409

<b>23.3 Iteratoren .....</b>	410
23.3.1 Verwendung von Iteratoren .....	413
23.3.2 Mehrere Iteratoren für dieselbe Instanz .....	416
23.3.3 Nachteile von Iteratoren gegenüber dem direkten Zugriff über Indizes .....	419
23.3.4 Alternative Definition für iterierbare Objekte .....	419
23.3.5 Funktionsiteratoren .....	420
<b>23.4 Spezielle Generatoren – itertools .....</b>	421

## 24 Kontextobjekte

---

<b>24.1 Die with-Anweisung .....</b>	431
<b>24.2 Hilfsfunktionen für with-Kontexte – contextlib .....</b>	434
24.2.1 Einfache Funktionen als Kontext-Manager .....	434
24.2.2 Bestimmte Exception-Typen unterdrücken .....	435
24.2.3 Den Standard-Ausgabestrom umleiten .....	436

## 25 Manipulation von Funktionen und Methoden

---

<b>25.1 Decorator .....</b>	437
<b>25.2 Das Modul functools .....</b>	440
25.2.1 Funktionsschnittstellen vereinfachen .....	440
25.2.2 Methodenschnittstellen vereinfachen .....	442
25.2.3 Caches .....	443
25.2.4 Ordnungsrelationen vervollständigen .....	444
25.2.5 Überladen von Funktionen .....	445

## TEIL IV Die Standardbibliothek

### 26 Mathematik

---

<b>26.1 Mathematische Funktionen – math, cmath .....</b>	451
26.1.1 Zahlentheoretische Funktionen .....	452
26.1.2 Exponential- und Logarithmusfunktionen .....	454
26.1.3 Trigonometrische und hyperbolische Funktionen .....	454

26.1.4	Umrechnen von Winkeln .....	455
26.1.5	Darstellungsformen komplexer Zahlen .....	455
<b>26.2</b>	<b>Zufallszahlengenerator – random .....</b>	<b>456</b>
26.2.1	Den Status speichern und laden .....	457
26.2.2	Zufällige ganze Zahlen erzeugen .....	457
26.2.3	Zufällige Gleitkommazahlen erzeugen .....	458
26.2.4	Zufallsgesteuerte Operationen auf Sequenzen .....	458
26.2.5	SystemRandom([seed]) .....	459
<b>26.3</b>	<b>Präzise Dezimalzahlen – decimal .....</b>	<b>460</b>
26.3.1	Verwendung des Datentyps .....	461
26.3.2	Nichtnumerische Werte .....	464
26.3.3	Das Context-Objekt .....	465

---

## 27 Kryptografie

---

<b>27.1</b>	<b>Hash-Funktionen – hashlib .....</b>	<b>467</b>
27.1.1	Verwendung des Moduls .....	469
27.1.2	Weitere Algorithmen .....	470
27.1.3	Vergleich großer Dateien .....	470
27.1.4	Passwörter .....	471
<b>27.2</b>	<b>Verschlüsselung – PyCrypto .....</b>	<b>472</b>
27.2.1	Symmetrische Verschlüsselungsverfahren .....	473
27.2.2	Asymmetrische Verschlüsselungsverfahren .....	476

---

## 28 Reguläre Ausdrücke

---

<b>28.1</b>	<b>Syntax regulärer Ausdrücke .....</b>	<b>481</b>
28.1.1	Beliebige Zeichen .....	482
28.1.2	Zeichenklassen .....	482
28.1.3	Quantoren .....	483
28.1.4	Vordefinierte Zeichenklassen .....	485
28.1.5	Weitere Sonderzeichen .....	487
28.1.6	Genügsame Quantoren .....	488
28.1.7	Gruppen .....	489
28.1.8	Alternativen .....	490
28.1.9	Extensions .....	490

<b>28.2 Verwendung des Moduls .....</b>	493
28.2.1 Searching .....	493
28.2.2 Matching .....	494
28.2.3 Einen String aufspalten .....	494
28.2.4 Teile eines Strings ersetzen .....	495
28.2.5 Problematische Zeichen ersetzen .....	496
28.2.6 Einen regulären Ausdruck kompilieren .....	496
28.2.7 Flags .....	496
28.2.8 Das Match-Objekt .....	498
<b>28.3 Ein einfaches Beispielprogramm – Searching .....</b>	499
<b>28.4 Ein komplexeres Beispielprogramm – Matching .....</b>	500

---

## 29 Schnittstelle zu Betriebssystem und Laufzeitumgebung

<b>29.1 Funktionen des Betriebssystems – os .....</b>	505
29.1.1 environ .....	506
29.1.2 getpid() .....	506
29.1.3 cpu_count() .....	506
29.1.4 system(cmd) .....	507
29.1.5 popen(command, [mode, buffering]) .....	507
<b>29.2 Zugriff auf die Laufzeitumgebung – sys .....</b>	508
29.2.1 Kommandozeilenparameter .....	508
29.2.2 Standardpfade .....	508
29.2.3 Standard-Ein-/Ausgabeströme .....	509
29.2.4 Das Programm beenden .....	509
29.2.5 Details zur Python-Version .....	510
29.2.6 Details zum Betriebssystem .....	511
29.2.7 Hooks .....	512

---

## 30 Kommandozeilenparameter

<b>30.1 Taschenrechner – ein einfaches Beispiel .....</b>	516
<b>30.2 Ein weiteres Beispiel .....</b>	520

<b>31 Dateisystem</b>	523
<b>31.1 Zugriff auf das Dateisystem mit os</b>	523
<b>31.2 Dateipfade – os.path</b>	530
<b>31.3 Zugriff auf das Dateisystem – shutil</b>	535
31.3.1 Verzeichnis- und Dateioperationen	537
31.3.2 Archivoperationen	538
<b>31.4 Temporäre Dateien – tempfile</b>	541
<b>32 Parallele Programmierung</b>	543
<b>32.1 Prozesse, Multitasking und Threads</b>	543
32.1.1 Die Leichtgewichte unter den Prozessen – Threads	544
32.1.2 Threads oder Prozesse?	546
<b>32.2 Python's Schnittstellen zur Parallelisierung</b>	546
<b>32.3 Parallelisierung von Funktionsaufrufen</b>	547
32.3.1 Ein Beispiel mit einem futures.ThreadPoolExecutor	548
32.3.2 Executor-Instanzen als Kontext-Manager	550
32.3.3 Die Verwendung von futures.ProcessPoolExecutor	550
32.3.4 Die Verwaltung der Aufgaben eines Executors	551
<b>32.4 Die Module threading und multiprocessing</b>	558
<b>32.5 Die Thread-Unterstützung in Python</b>	558
32.5.1 Kritische Bereiche mit Lock-Objekten absichern	560
32.5.2 Datenaustausch zwischen Threads mit Critical Sections	562
32.5.3 Gefahren von Critical Sections – Deadlocks	567
<b>32.6 Einblick in das Modul multiprocessing</b>	568
<b>32.7 Ausblick</b>	569
<b>33 Datenspeicherung</b>	571
<b>33.1 Komprimierte Dateien lesen und schreiben – gzip</b>	571
<b>33.2 XML</b>	573
33.2.1 ElementTree	575
33.2.2 SAX – Simple API for XML	583

<b>33.3 Datenbanken .....</b>	587
33.3.1 Pythons eingebaute Datenbank – sqlite3 .....	590
<b>33.4 Serialisierung von Instanzen – pickle .....</b>	607
33.4.1 Funktionale Schnittstelle .....	608
33.4.2 Objektorientierte Schnittstelle .....	609
<b>33.5 Das Datenaustauschformat JSON – json .....</b>	610
<b>33.6 Das Tabellenformat CSV – csv .....</b>	612
33.6.1 reader-Objekte – Daten aus einer CSV-Datei lesen .....	613
33.6.2 Dialect-Objekte – eigene Dialekte verwenden .....	615

## 34 Netzwerkkommunikation

619

---

<b>34.1 Socket API .....</b>	620
34.1.1 Client-Server-Systeme .....	621
34.1.2 UDP .....	624
34.1.3 TCP .....	626
34.1.4 Blockierende und nicht-blockierende Sockets .....	628
34.1.5 Erzeugen eines Sockets .....	629
34.1.6 Die Socket-Klasse .....	631
34.1.7 Netzwerk-Byte-Order .....	634
34.1.8 Multiplexende Server – selectors .....	635
34.1.9 Objektorientierte Serverentwicklung – socketserver .....	637
<b>34.2 URLs – urllib .....</b>	639
34.2.1 Zugriff auf entfernte Ressourcen – urllib.request .....	640
34.2.2 Einlesen und Verarbeiten von URLs – urllib.parse .....	644
<b>34.3 FTP – ftplib .....</b>	648
34.3.1 Mit einem FTP-Server verbinden .....	649
34.3.2 FTP-Kommandos ausführen .....	650
34.3.3 Mit Dateien und Verzeichnissen arbeiten .....	650
34.3.4 Übertragen von Dateien .....	652
<b>34.4 E-Mail .....</b>	654
34.4.1 SMTP – smtplib .....	655
34.4.2 POP3 – poplib .....	658
34.4.3 IMAP4 – imaplib .....	662
34.4.4 Erstellen komplexer E-Mails – email .....	668

<b>34.5 Telnet – telnetlib .....</b>	673
34.5.1 Die Klasse Telnet .....	673
34.5.2 Beispiel .....	674
<b>34.6 XML-RPC .....</b>	676
34.6.1 Der Server .....	677
34.6.2 Der Client .....	680
34.6.3 Multicall .....	682
34.6.4 Einschränkungen .....	683

---

## **35 Debugging und Qualitätssicherung**

---

<b>35.1 Der Debugger .....</b>	687
<b>35.2 Formatierte Bildschirmausgabe – pprint .....</b>	690
<b>35.3 Logdateien – logging .....</b>	691
35.3.1 Das Meldungsformat anpassen .....	694
35.3.2 Logging Handler .....	696
<b>35.4 Automatisiertes Testen .....</b>	698
35.4.1 Testfälle in Docstrings – doctest .....	698
35.4.2 Unit Tests – unittest .....	703
<b>35.5 Analyse des Laufzeitverhaltens .....</b>	706
35.5.1 Laufzeitmessung – timeit .....	707
35.5.2 Profiling – cProfile .....	710
35.5.3 Tracing – trace .....	713
<b>35.6 Optimierung .....</b>	716
35.6.1 Die Optimize-Option .....	717
35.6.2 Mutabel vs. immutabel .....	717
35.6.3 Schleifen .....	718
35.6.4 Funktionsaufrufe .....	719
35.6.5 C .....	719
35.6.6 Lookup .....	720
35.6.7 Exceptions .....	720
35.6.8 Keyword Arguments .....	721
35.6.9 Alternative Interpreter: PyPy .....	721

---

<b>36 Dokumentation</b>	723
<b>36.1 Docstrings</b> .....	723
<b>36.2 Automatisches Erstellen einer Dokumentation – pydoc</b> .....	725
<b>TEIL V Weiterführende Themen</b>	
<b>37 Anbindung an andere Programmiersprachen</b>	729
<b>37.1 Dynamisch ladbare Bibliotheken – ctypes</b> .....	730
37.1.1 Ein einfaches Beispiel .....	730
37.1.2 Die eigene Bibliothek .....	731
37.1.3 Datentypen .....	733
37.1.4 Schnittstellenbeschreibung .....	735
37.1.5 Pointer .....	737
37.1.6 Strings .....	738
<b>37.2 Schreiben von Extensions</b> .....	739
37.2.1 Ein einfaches Beispiel .....	739
37.2.2 Exceptions .....	744
37.2.3 Erzeugen der Extension .....	745
37.2.4 Reference Counting .....	747
<b>37.3 Python als eingebettete Skriptsprache</b> .....	748
37.3.1 Ein einfaches Beispiel .....	748
37.3.2 Ein komplexeres Beispiel .....	750
<b>37.4 Alternative Interpreter</b> .....	753
37.4.1 Interoperabilität mit der Java Runtime Environment – Jython .....	754
37.4.2 Interoperabilität mit .NET – IronPython .....	759
<b>38 Distribution von Python-Projekten</b>	765
<b>38.1 Eine Geschichte der Distributionen in Python</b> .....	765
38.1.1 Der klassische Ansatz – distutils .....	766
38.1.2 Der neue Standard – setuptools .....	766
38.1.3 Der Paketindex – PyPI und pip .....	767
<b>38.2 Erstellen von Distributionen – setuptools</b> .....	767
38.2.1 Schreiben des Moduls .....	768

38.2.2	Das Installationsskript .....	769
38.2.3	Erstellen einer Quellcodedistribution .....	774
38.2.4	Erstellen einer Binärdistribution .....	774
38.2.5	Distributionen installieren .....	776
38.2.6	Eigenständige Distributionen erstellen .....	776
38.2.7	Erstellen von EXE-Dateien – cx_Freeze .....	777
<b>38.3</b>	<b>Der Python-Paketmanager – pip .....</b>	<b>778</b>
<b>38.4</b>	<b>Lokalisierung von Programmen – gettext .....</b>	<b>779</b>
38.4.1	Beispiel für die Verwendung von gettext .....	780
38.4.2	Erstellen des Sprachkomplilats .....	781

## 39 Grafische Benutzeroberflächen

785

---

<b>39.1</b>	<b>Toolkits .....</b>	<b>785</b>
<b>39.2</b>	<b>Einführung in tkinter .....</b>	<b>788</b>
39.2.1	Ein einfaches Beispiel .....	788
39.2.2	Steuerelementvariablen .....	790
39.2.3	Der Packer .....	792
39.2.4	Events .....	796
39.2.5	Steuerelemente .....	803
39.2.6	Zeichnungen – das Canvas-Widget .....	823
39.2.7	Weitere Module .....	830
<b>39.3</b>	<b>Einführung in PyQt .....</b>	<b>834</b>
39.3.1	Installation .....	834
39.3.2	Grundlegende Konzepte von Qt .....	835
39.3.3	Entwicklungsprozess .....	837
<b>39.4</b>	<b>Signale und Slots .....</b>	<b>844</b>
<b>39.5</b>	<b>Wichtige Widgets .....</b>	<b>847</b>
39.5.1	QCheckBox .....	847
39.5.2	QComboBox .....	848
39.5.3	QDateEdit, QDateTimeEdit, QDateTimeEdit .....	849
39.5.4	QDialog .....	849
39.5.5	QLineEdit .....	850
39.5.6	QListWidget, QListView .....	850
39.5.7	QProgressBar .....	851
39.5.8	QPushButton .....	852
39.5.9	QRadioButton .....	852
39.5.10	QSlider, QDial .....	852

39.5.11	QTextEdit .....	853
39.5.12	QWidget .....	854
<b>39.6</b>	<b>Zeichenfunktionalität .....</b>	<b>855</b>
39.6.1	Werkzeuge .....	855
39.6.2	Koordinatensystem .....	857
39.6.3	Einfache Formen .....	858
39.6.4	Grafiken .....	860
39.6.5	Text .....	861
39.6.6	Eye Candy .....	863
<b>39.7</b>	<b>Model-View-Architektur .....</b>	<b>867</b>
39.7.1	Beispielprojekt: ein Adressbuch .....	868
39.7.2	Auswählen von Einträgen .....	877
39.7.3	Bearbeiten von Einträgen .....	879

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

---

<b>40.1</b>	<b>Konzepte und Besonderheiten von Django .....</b>	<b>884</b>
<b>40.2</b>	<b>Installation von Django .....</b>	<b>885</b>
40.2.1	Installation unter Linux und OS X .....	886
40.2.2	Installation unter Windows .....	887
<b>40.3</b>	<b>Erstellen eines neuen Django-Projekts .....</b>	<b>888</b>
40.3.1	Der Entwicklungswerkserver .....	889
40.3.2	Konfiguration des Projekts .....	890
<b>40.4</b>	<b>Erstellung einer Applikation .....</b>	<b>892</b>
40.4.1	Die Applikation in das Projekt einbinden .....	894
40.4.2	Ein Model definieren .....	894
40.4.3	Beziehungen zwischen Modellen .....	895
40.4.4	Übertragung des Modells in die Datenbank .....	896
40.4.5	Das Model-API .....	897
40.4.6	Unser Projekt bekommt ein Gesicht .....	903
40.4.7	Djangos Template-System .....	910
40.4.8	Verarbeitung von Formulardaten .....	923
40.4.9	Djangos Administrationsoberfläche .....	926

<b>41 Wissenschaftliches Rechnen</b>	933
<b>41.1 Installation</b>	934
<b>41.2 Das Modellprogramm</b>	934
41.2.1 Der Import von numpy, scipy und matplotlib	936
41.2.2 Vektorisierung und der Datentyp numpy.ndarray	936
41.2.3 Visualisieren von Daten mit matplotlib.pyplot	940
<b>41.3 Überblick über die Module numpy und scipy</b>	943
41.3.1 Überblick über den Datentyp numpy.ndarray	943
41.3.2 Überblick über scipy	951
<b>42 Insiderwissen</b>	955
<b>42.1 URLs im Standardbrowser öffnen – webbrowser</b>	955
<b>42.2 Interpretieren von Binärdaten – struct</b>	955
<b>42.3 Versteckte Passworteingabe</b>	958
<b>42.4 Kommandozeilen-Interpreter</b>	959
<b>42.5 Dateiinterface für Strings – io.StringIO</b>	961
<b>42.6 Generatoren als Konsumenten</b>	962
42.6.1 Ein Decorator für konsumierende Generatorfunktionen	964
42.6.2 Auslösen von Exceptions in einem Generator	965
42.6.3 Eine Pipeline als Verkettung konsumierender Generatorfunktionen	966
<b>42.7 Kopieren von Instanzen – copy</b>	967
<b>42.8 Die interaktive Python-Shell – IPython</b>	971
42.8.1 Die interaktive Shell	971
42.8.2 Das IPython-Notebook	974
<b>42.9 Bildverarbeitung – Pillow</b>	977
42.9.1 Bilddateien laden und speichern	978
42.9.2 Zugriff auf einzelne Pixel	979
42.9.3 Teilbereiche eines Bildes ausschneiden	979
42.9.4 Bilder zusammenfügen	980
42.9.5 Geometrische Bildtransformationen	981
42.9.6 Vordefinierte Bildfilter	982
42.9.7 Eigene Pixeloperationen	983
42.9.8 Bildverbesserungen	984

42.9.9 Zeichenoperationen .....	985
42.9.10 Interoperabilität .....	986

## 43 Von Python 2 nach Python 3

987

---

<b>43.1 Die wichtigsten Unterschiede .....</b>	990
43.1.1 Ein-/Ausgabe .....	990
43.1.2 Iteratoren .....	991
43.1.3 Strings .....	992
43.1.4 Ganze Zahlen .....	993
43.1.5 Exception Handling .....	994
43.1.6 Standardbibliothek .....	994
43.1.7 Neue Sprachelemente in Python 3 .....	995
<b>43.2 Automatische Konvertierung .....</b>	996
<b>43.3 Geplante Sprachelemente .....</b>	999

## A Anhang

1001

---

<b>A.1 Reservierte Wörter .....</b>	1001
<b>A.2 Eingebaute Funktionen .....</b>	1001
<b>A.3 Eingebaute Exceptions .....</b>	1005
<b>A.4 Python IDEs .....</b>	1009

---

<b>Index .....</b>	1017
--------------------	------