

Inhalt

Vorwort — V

Teil I: Die Programmiersprache Python

1	Einleitung — 3
1.1	Konventionen in diesem Text — 3
1.2	Programmieren — 4
1.3	Der Python-Interpreter — 4
1.4	Kommentare — 5
1.5	Einrückung und Leerzeichen — 6
1.6	Fehlermeldungen — 6
1.7	Nichts tun: pass — 7
1.8	Eingebaute Funktionen — 7
1.9	Funktionen nachladen: import — 8
1.10	Variablen — 9
1.11	Gültige Variablennamen — 10
1.12	Ausgeben von Variablen — 11
1.13	Eingebaute Hilfen — 11
2	Eingebaute Objekttypen — 14
2.1	Zahlen — 14
2.1.1	Mathematische Operatoren — 15
2.1.2	Bit-Operatoren — 16
2.1.3	Vergleichsoperatoren — 17
2.1.4	Kurzschreibweisen von Operatoren — 17
2.2	Zeichenketten — 17
2.3	Listen — 20
2.4	Tupel — 21
2.5	Range — 22
2.6	Operationen auf Sequenztypen — 22
2.6.1	Index-Zugriff — 22
2.6.2	Slicing — 23
2.6.3	Addition, Multiplikation und andere Funktionen — 25
2.7	Dictionarys — 26
2.8	Set und Frozenset — 28
2.9	Erzeugung von Listen, Sets und Dictionarys — 30
2.10	File-Objekte — 31
2.11	Binärdaten — 32

VIII — Inhalt

2.11.1	Unveränderbare Byte-Folgen: bytes — 32
2.11.2	Veränderbare Byte-Folgen: bytearray — 34
2.11.3	Daten in bytes und bytearray manipulieren — 35

3 Fehlerbehandlung — 42

3.1	Fehler mit try ... except fangen — 42
3.2	Verschiedene Fehler fangen — 43
3.3	Alle Fehler fangen — 43
3.4	Weiter ohne Fehler: try ... except ... else — 43
3.5	Dinge auf jeden Fall ausführen: finally — 44
3.6	Mehr über den Fehler erfahren — 44
3.7	Einen Fehler auslösen — 45

4 Ein- und Ausgabe — 46

4.1	Eingabe mit input() — 46
4.2	Ausgabe — 48
4.2.1	Strings formatieren mit format() — 49
4.2.2	Formatierung mit Formatstrings (Old String Formatting) — 53
4.2.3	Ausgabe auf stdout oder stderr — 54
4.2.4	Ausgabe mit print in eine Datei — 55
4.3	Dateien — 55
4.3.1	Arbeiten mit Dateien — 55
4.3.2	Textdateien — 57
4.3.3	Binärdateien — 58
4.3.4	Daten in Objekte ausgeben — 60
4.3.5	Fehlerbehandlung bei der Arbeit mit Dateien — 61

5 Steuerung des Programmablaufs — 63

5.1	True, False, Leere Objekte und None — 63
5.2	Objektvergleich und Wertvergleich — 64
5.3	Boolesche Operatoren — 65
5.4	Bedingte Ausführung: if — 66
5.5	Mehrere Bedingungen: elif und else — 66
5.6	Objektabhängige Ausführung: with — 67
5.7	Bedingter Ausdruck — 68

6 Schleifen — 69

6.1	Zählschleife: for — 69
6.2	Bedingte Schleife: while — 71
6.3	Unterbrechung einer Schleife: break — 72
6.4	Neustart der Schleife: continue — 72

7	Funktionen — 73
7.1	Definieren und Aufrufen von Funktionen — 73
7.2	Sichtbarkeit von Variablen — 74
7.3	Funktionsparameter — 76
7.3.1	Positionsabhängige Parameter — 77
7.3.2	Variable Anzahl von Parametern — 78
7.3.3	Keyword Parameter — 80
7.4	Defaultwerte für Funktionsparameter — 81
7.5	Rückgabewert einer Funktion — 81
7.6	Manipulation von Argumenten — 82
7.7	Dokumentation im Programm: Docstring — 83
7.8	Geschachtelte Funktionen — 83
8	Funktionales — 85
8.1	Lambda-Funktionen — 85
8.2	Funktionen auf Sequenzen ausführen: <code>map()</code> — 86
8.3	Daten aus einer Sequenz extrahieren: <code>filter()</code> — 87
8.4	Das Modul <code>functools</code> — 87
9	Module — 88
9.1	Laden eines Moduls — 88
9.1.1	Selektives Laden aus einem Modul — 88
9.1.2	Umbenennen eines Moduls beim Import — 89
9.2	Ein Modul erstellen — 89
9.3	Wo werden Module gesucht? — 90
9.4	Fehler beim Importieren — 91
9.5	Ein Modul ist auch ein Programm — 91
9.6	Neu laden eines Moduls — 92
9.7	Mehrere Module – Ein Package — 92
10	Objekte — 94
10.1	Definition von Klassen — 94
10.2	Methoden — 95
10.3	Attribute — 95
10.4	Von der Klasse zum Objekt — 96
10.4.1	Ein Objekt initialisieren: Der Konstruktor — 96
10.4.2	Überladen von Funktionen — 98
10.5	Klassenvariablen — 98
10.6	Vererbung — 99
10.6.1	Einfache Vererbung — 100
10.6.2	Von eingebauten Klassen erben — 102
10.6.3	Mehrfachvererbung — 104

11	Objekte unter der Lupe — 106
11.1	Typ einer Variablen — 106
11.2	Attribute eines Objekts — 107
11.3	Standardfunktionen implementieren — 109
11.3.1	Vergleichsoperatoren — 111
11.3.2	Attributzugriff — 112
11.3.3	Verhalten von Containertypen — 115
11.3.4	Mathematische Operatoren — 116
11.3.5	Sonstige Operatoren und Konvertierungs-Funktionen — 117
11.4	Objekte aufrufen — 117
11.5	Darstellung eines Objekts als Zeichenkette — 119
11.6	Informationen über Objekte sammeln — 121
11.7	Attribute managen: Propertys — 122
11.8	Deskriptoren — 123
11.9	Dekoratoren — 126
11.9.1	Die zu dekorierende Funktion — 127
11.9.2	Eine Funktion als Dekorator — 128
11.9.3	Den Dekorator tarnen — 130
11.9.4	Objekt als Dekorator — 132
11.10	Iteratoren — 133
11.11	Generatoren — 135
11.12	Context Manager — 136
11.13	Exceptions — 139
12	Mehr zu Namensräumen — 140
12.1	Implizite Variablensuche — 141
12.2	Explizite Variablensuche: global und nonlocal — 141

Teil II: Batterien enthalten

13	Collections — 145
13.1	deque — 145
13.2	ChainMap — 147
13.3	Counter — 148
13.4	OrderedDict — 150
13.5	defaultDict — 151
13.6	UserDict, UserList und UserString — 152
13.7	namedtuple — 152
14	Datum und Uhrzeit — 154
14.1	UNIX-Zeit: time — 154

14.1.1	Ausgeben einer Zeit — 156
14.1.2	Einlesen einer Zeit — 157
14.1.3	Schlafen — 158
14.2	Das Modul datetime — 158
14.2.1	Datum: <code>datetime.date</code> — 159
14.2.2	Uhrzeit: <code>datetime.time</code> — 160
14.2.3	Zeitdifferenz/Zeitzone: <code>datetime.timedelta/datetime.timezone</code> — 162
14.2.4	Datum und Uhrzeit: <code>datetime.datetime</code> — 164
15	Dateien und Verzeichnisse — 169
15.1	Systemunabhängiger Zugriff mit <code>pathlib</code> — 169
15.1.1	Die Klasse <code>Path</code> — 170
15.1.2	Methoden von Pure-Pfad-Objekten — 173
15.1.3	Methoden von konkreten Pfad-Objekten — 174
15.2	OS-spezifischer Zugriff mit <code>os.path</code> — 179
15.3	Temporäre Dateien erzeugen — 181
16	Reguläre Ausdrücke — 183
16.1	Text beschreiben — 183
16.2	Pattern Matching — 185
16.2.1	Steuerung des Matching — 187
16.2.2	Individuelle Parameter für eine Gruppe — 187
16.3	Suchen und Ersetzen — 189
16.4	Referenzen auf gefundenen Text — 190
16.5	Regular Expression Objects — 191
16.6	Funktionen, Konstanten und Ausnahmen des Moduls — 191
17	Zufallszahlen — 195
17.1	Das Modul <code>random</code> — 195
17.2	Einfache Zufallszahlen — 196
17.3	Zahlen aus einem Bereich — 197
17.4	Auswahl aus einer Menge und Mischen — 198
17.5	Zufallszahlen mit der Klasse <code>SystemRandom</code> — 199
17.6	Verteilungsfunktionen — 199
18	Netzwerkprogrammierung mit Sockets — 202
18.1	Kommunikation im Internet — 202
18.1.1	Ein Socket-Server mit UDP — 203
18.1.2	Ein Socket-Client mit UDP — 204
18.1.3	UDP-Client und Server bei der Arbeit — 205
18.1.4	Ein Socket-Server mit TCP — 206

18.1.5	Ein Socket-Client mit TCP — 207
18.1.6	TCP-Client und Server bei der Arbeit — 207
18.2	UNIX-Sockets — 207
18.2.1	TCP UNIX-Socket — 208
18.2.2	UDP UNIX-Socket — 211
18.2.3	Client und Server mit UDP UNIX-Socket bei der Arbeit — 212
18.3	Mehrere Verbindungen über einen Socket — 212
19	Automatisches Testen mit doctest — 213
19.1	Doctest anwenden — 213
19.2	Doctest in ein Programm einfügen — 214
19.3	Umgang mit variablen Ausgaben — 216
19.4	Auslagern des Tests in eine Datei — 217
20	Iteratoren und funktionale Programmierung — 220
20.1	Erzeugung von Iteratoren mit <code>itertools</code> — 220
20.1.1	Endlosschleifen — 220
20.1.2	Sequenzfilter — 221
20.1.3	Permutationen — 227
20.2	Tools für Funktionen <code>functools</code> — 229
21	Binärdaten und Codierungen — 237
21.1	Binärstrukturen bearbeiten mit <code>struct</code> — 237
21.2	Transportcodierung mit <code>base64</code> — 242
21.3	Objekte Serialisieren — 243
21.3.1	Python-Objekte serialisieren mit <code>pickle</code> — 243
21.3.2	Daten serialisieren mit <code>json</code> — 247
22	Internetprotokolle — 251
22.1	Ein Webserver in Python — 251
22.1.1	Webserver für statische Seiten — 251
22.1.2	Server für dynamische Seiten — 253
22.1.3	Daten auf dem Webserver empfangen und verarbeiten — 255
22.2	Webseiten mit einem Programm holen — 258
22.2.1	Eine Webseite holen mit <code>urlopen()</code> — 259
22.2.2	Die Klasse <code>Request</code> — 261
22.3	Bearbeiten von URLs — 263
22.3.1	Aufbauen einer URL — 263
22.3.2	Zerlegen einer URL — 264
22.4	E-Mail senden mit <code>smtplib</code> — 265
22.4.1	Versenden einer Mail — 266
22.4.2	Header zu einer Nachricht hinzufügen — 269

22.4.3	Verschlüsselter Mailversand — 269
22.5	E-Mail erstellen mit dem Modul <code>email</code> — 272
22.5.1	Header zu einer Nachricht hinzufügen — 273
22.5.2	Charset des Nachrichtentextes — 273
22.5.3	Abfragen von Nachrichten-Headern — 274
22.5.4	Weitere Methoden für <code>email.message</code> — 277
22.5.5	Aufbau einer Multipart-Message — 279
22.6	Multipart-Nachrichten mit <code>email.mime</code> — 280
22.7	E-Mail aus einer Datei lesen — 283
22.8	E-Mail empfangen mit <code>poplib</code> — 286
22.8.1	Die Klasse <code>POP3</code> — 286
22.8.2	Weiterverarbeitung empfangener Nachrichten — 288
23	Multitasking — 290
23.1	Das Modul <code>threading</code> — 291
23.1.1	Die Klasse <code>Thread</code> — 292
23.1.2	Threads als Objekt erzeugen — 293
23.1.3	Threads identifizieren — 295
23.1.4	Threads als abgeleitete Klasse — 295
23.2	Größere Mengen von Threads — 298
23.3	Synchronisation von Threads — 298
23.3.1	<code>Lock</code> — 299
23.3.2	<code>RLock</code> — 301
23.3.3	<code>Condition</code> — 301
23.3.4	<code>Semaphore</code> — 306
23.3.5	<code>Event</code> — 307
23.3.6	<code>Timer</code> — 308
23.3.7	<code>Barrier</code> — 310
23.4	Datenaustausch zwischen Threads — 313
23.5	Das Modul <code>multiprocessing</code> — 318
23.6	Datenaustausch zwischen Prozessen — 323
23.6.1	Die Klasse <code>multiprocessing.Pipe</code> — 323
23.6.2	Die Klasse <code>multiprocessing.Queue</code> — 325
23.6.3	Shared Memory — 327
23.6.4	Die Funktion <code>Manager</code> — 331
23.6.5	Manager-Objekte erstellen und manipulieren — 334
23.6.6	Manager-Objekte zwischen lokalen Prozessen teilen — 336
23.6.7	<code>BaseManager</code> -Objekte — 337
23.6.8	Zugriff von Manager-Objekten über das Netzwerk — 338
24	Logging — 341
24.1	Das Modul <code>logging</code> — 341

24.1.1	Formatieren einer Logzeile — 342
24.1.2	Konfigurieren des Loggings mit <code>basicConfig()</code> — 345
24.1.3	Eigene Log-Level definieren — 346
24.1.4	Unterdrücken eines Log-Levels — 347
24.1.5	Funktionen im Modul <code>logging</code> — 348
24.2	Objektorientiertes Logging mit <code>logging</code> — 349
24.2.1	Logger-Objekte — 349
24.2.2	Die Klasse <code>Handler</code> — 353
24.2.3	Formatter-Objekte — 358
24.2.4	Filter-Objekte — 359
24.2.5	Ausgabe auf mehrere Kanäle gleichzeitig — 360
24.2.6	Konfigurationsmöglichkeiten für das Logging — 360
24.3	Logging mit <code>syslog</code> — 368
24.3.1	Log-Level im Modul <code>syslog</code> — 369
24.3.2	Berechnung der Bitmaske für <code>setlogmask()</code> — 369
24.3.3	Log-Kanäle — 370
24.3.4	Weitere Parameter für <code>openlog()</code> — 370
24.3.5	Beispiele — 371
25	Datenbanken — 372
25.1	Das DB-API 2.0 — 372
25.1.1	Attribute eines Datenbankmoduls — 373
25.1.2	<code>Connection</code> -Objekte — 373
25.1.3	<code>Cursor</code> -Objekte — 374
25.1.4	Exceptions — 375
25.2	Das Modul <code>sqlite3</code> — 376
25.2.1	Öffnen einer Datenbank — 377
25.2.2	Daten definieren und abfragen — 378
25.2.3	Transaktionen — 379
25.2.4	Transaktionen interaktiv — 381
25.2.5	Erweiterungen des <code>Connection</code> -Objekts — 384
25.3	PostgreSQL mit <code>psycopg2</code> — 385
25.3.1	Verbindung zur Datenbank — 385
25.3.2	Parameter an Anfragen übergeben — 387
25.3.3	Erweiterungen des <code>Connection</code> -Objekts — 388
25.3.4	Transaktionen und Isolation Level — 390
25.4	MySQL mit <code>mysqlclient</code> — 391
25.4.1	Verbindung zur Datenbank — 391
25.4.2	Parameter an Anfragen übergeben — 393
25.4.3	Erweiterungen des <code>Connection</code> -Objekts — 395
25.4.4	Cursor-Klassen — 396

26	Diverses — 397
26.1	math – Mathematische Funktionen — 397
26.2	hashlib – Hashfunktionen — 401
26.3	csv – CSV-Dateien — 403
26.4	getpass – Passwörter eingeben — 408
26.5	enum – Aufzählungstypen — 409
26.6	pprint – Variablen schöner ausgeben — 412
27	Verarbeiten von Startparametern — 414
27.1	Zugriff auf die Startparameter mit sys — 414
27.2	Startparameter mit argparse verarbeiten — 415
27.2.1	Die Klasse ArgumentParser — 416
27.2.2	Den Parser starten — 418
27.2.3	Argumente für den Parser definieren — 419
28	Python erweitern — 428
28.1	Module von Drittanbietern finden — 428
28.2	Pakete installieren mit pip — 428
28.3	virtualenv und pyvenv — 430
28.4	Interessante Module außerhalb der Standardlibrary — 431

Teil III: Größere Beispiele

29	Referenzzähler für Latex-Dokumente — 435
29.1	Anforderungen — 435
29.2	Eine Lösung — 435
29.3	Mögliche Erweiterungen — 437
30	Dateien und Verzeichnisse umbenennen — 438
30.1	Anforderungen — 438
30.2	Eine Lösung — 438
30.3	Mögliche Erweiterungen — 439
31	Verfügbarkeit von Webservern prüfen — 440
31.1	Einen Webrequest stellen — 440
31.2	Eine Klasse zum Abfragen einer URL — 441
31.3	Webseite in einem Thread abfragen — 442
31.4	Abfragen mehrerer Hosts — 443
31.5	Mögliche Erweiterungen — 443

32	Änderungen im Linux-Dateisystem überwachen – Inotify — 445
32.1	Beschränkungen von inotify — 445
32.2	Ereignisse im Dateisystem — 445
32.3	Einen Eventhandler implementieren — 446
32.4	Einrichten einer Überwachung — 447
32.5	Überwachen von /tmp — 448
32.6	Überwachung mehrerer Dateien/Verzeichnisse — 450
32.7	Mögliche Erweiterung — 451

Teil IV: Anhang

A	Keywords, Konstanten und Module — 455
----------	--

A.1	Keywords in Python — 455
A.2	Konstanten — 456
A.3	Eingebaute Funktionen — 457
A.4	Module — 460
A.4.1	Datentypen — 460
A.4.2	Mathematische Funktionen — 460
A.4.3	Verarbeitung von Text — 461
A.4.4	Dateien und Verzeichnisse — 461
A.4.5	Speicherung von Daten und Objekten — 462
A.4.6	Verschiedene Dateiformate — 462
A.4.7	Datenkomprimierung und Archivierung — 463
A.4.8	Grundlegende Funktionen — 464
A.4.9	Parallele Programmierung — 464
A.4.10	Kommunikation zwischen Prozessen — 465
A.4.11	Internet-Protokolle — 465
A.4.12	Internet-Datenformate — 466
A.4.13	XML, HTML — 467
A.4.14	Binärdaten — 467
A.4.15	Funktionale Programmierung — 467
A.4.16	Sonstiges — 468
A.4.17	Kryptografische Funktionen — 468
A.4.18	Internationalization — 469

B	Onlinequellen — 470
----------	----------------------------

Stichwortverzeichnis — 473
