

Python核心编程



[Python核心编程_下载链接1](#)

著者:

出版者:机械工业出版社

出版时间:2001-8-1

装帧:

isbn:9787111089834

Python是一种不复杂但很健全的编程语言。它不光具备传统编译型程序设计语言强大的功能和复杂性，还在某种程度上具备比较简单的脚本和解释型程序设计语言的易用性。该书向读者介绍了这种语言的核心内容，并展示了Python语言可以完成哪些任务。其主要内容包括：语法和编程风格、Python语言的对象、Web程序设计、执行环境等。该书条理清晰、通俗易懂，是学习Python语言的最好教材及参考手册。

所附光盘包括Python语言最新的三个版本及书中示例代码。

作者介绍:

目录: 前言
第一部分 程序设计语言Python
第1章 欢迎使用Python 1
1.1 什么是Python语言 1
1.2 Python语言的历史 1
1.3 Python语言的特色 1
1.4 获得Python语言 5
1.5 安装Python语言 6
1.6 运行Python语言 7
1.6.1 命令行上的交互式解释器 7

1.6.2 命令行上的脚本程序	9
1.6.3 集成开发环境	10
1.7 Python语言的文档	13
1.8 Python与其他语言的比较	13
1.9 JPython的特色	14
1.10 练习	15
第2章 快速入门	16
2.1 程序输出、print语句和“Hello World!”	16
2.2 程序输入和raw_input() 内建函数	17
2.3 程序注释	17
2.4 操作符	17
2.5 变量和赋值	19
2.6 数字	19
2.7 字符串	20
2.8 列表和表列	20
2.9 字典	21
2.10 代码段使用缩进	22
2.11 if语句	22
2.12 while循环	23
2.13 for循环和range()内建函数	23
2.14 文件和open()内建函数	25
2.15 错误和例外	26
2.16 函数	26
2.17 类	27
2.18 模块	29
2.19 练习	31
第3章 语法和程序设计风格	34
3.1 语句和语法	34
3.2 变量分配	36
3.3 标识符	38
3.4 程序设计风格准则	39
3.4.1 模块的结构和布局	40
3.4.2 在主体部分里加上测试方面的内容	42
3.5 内存管理	43
3.5.1 变量定义	43
3.5.2 动态确定变量的类型	43
3.5.3 内存分配	43
3.5.4 废弃物回收	44
3.5.5 引用计数	44
3.5.6 del语句	44
3.5.7 减少引用计数	45
3.6 第一个Python应用程序	45
3.7 练习	49
第4章 对象	50
4.1 Python语言中的对象	50
4.2 标准数据类型	51
4.3 其他内建的数据类型	51
4.3.1 type类型和type()内建函数	52
4.3.2 None空类型	52
4.4 内部数据类型	52
4.4.1 代码对象	53
4.4.2 框架对象	53
4.4.3 跟踪记录对象	53
4.4.4 序列切片对象	53
4.4.5 Ellipsis对象	54
4.4.6 Xrange对象	54
4.5 与数据类型有关的标准操作符	55
4.5.1 值的比较	55
4.5.2 对象实体的比较	56
4.5.3 布尔表达式	58
4.6 与数据类型有关的标准内建函数	59
4.6.1 cmp()	59
4.6.2 str()和repr()	59

- 4.6.3 深入type() 60
- 4.7 标准数据类型的分类 63
 - 4.7.1 存储模型 64
 - 4.7.2 修改模型 64
 - 4.7.3 访问模型 66
- 4.8 Python语言不支持的数据类型 67
- 4.9 练习 68
- 第5章 数字 69
 - 5.1 数字简介 69
 - 5.2 整数 69
 - 5.2.1 (普通) 整数 70
 - 5.2.2 长整数 70
 - 5.3 浮点实数 70
 - 5.4 复数 71
 - 5.5 操作符 72
 - 5.5.1 混状态操作符 72
 - 5.5.2 标准类型的操作符 74
 - 5.5.3 数值类型操作符 74
 - 5.5.4 *位操作符 76
 - 5.6 内建函数 77
 - 5.6.1 标准类型函数 77
 - 5.6.2 数值类型函数 78
 - 5.6.3 只适用于整数的函数 82
 - 5.7 相关模块 83
 - 5.8 练习 85
- 第6章 序列：字符串、列表和表列 88
 - 6.1 序列 88
 - 6.1.1 操作符 88
 - 6.1.2 内建函数 92
 - 6.2 字符串 93
 - 6.3 字符串和操作符 94
 - 6.3.1 标准类型操作符 94
 - 6.3.2 序列操作符 95
 - 6.4 只作用于字符串的操作符 99
 - 6.4.1 格式操作符 (%) 99
 - 6.4.2 生字符串操作符 (r/R) 102
 - 6.4.3 Unicode字符串操作符 (u/U) 103
 - 6.5 内建函数 103
 - 6.5.1 标准类型函数 103
 - 6.5.2 序列类型函数 104
 - 6.5.3 字符串类型函数 104
 - 6.6 字符串的内建方法 104
 - 6.7 字符串的特性 107
 - 6.7.1 特殊或控制字符 107
 - 6.7.2 三引号 108
 - 6.7.3 字符串的不可变性 109
 - 6.7.4 Unicode支持 111
 - 6.7.5 Python语言没有字符或数组 112
 - 6.8 相关模块 113
 - 6.9 字符串总结 114
 - 6.10 列表 115
 - 6.11 操作符 117
 - 6.11.1 标准类型操作符 117
 - 6.11.2 序列类型操作符 117
 - 6.11.3 列表类型操作符 120
 - 6.12 内建函数 120
 - 6.12.1 标准类型函数 120
 - 6.12.2 序列类型函数 121
 - 6.12.3 列表类型内建函数 123
 - 6.13 列表类型的内建方法 123
 - 6.14 列表的特性 125
 - 6.14.1 利用列表创建其他数据结构 125
 - 6.14.2 列表的子类 130

- 6.15 表列 131
- 6.16 表列的操作符和内建函数 132
 - 6.16.1 标准和序列操作符与内建函数 132
 - 6.16.2 表列类型操作符和内建函数及方法 133
- 6.17 表列的特性 133
 - 6.17.1 不可变性对表列有何影响 133
 - 6.17.2 表列也不是绝对“不可变的” 133
 - 6.17.3 括号的作用 134
 - 6.17.4 单元素表列 135
- 6.18 相关模块 136
- 6.19 *浅拷贝与深拷贝 137
- 6.20 练习 141
- 第7章 字典数据类型 144
 - 7.1 字典简介 144
 - 7.2 操作符 147
 - 7.2.1 标准类型操作符 147
 - 7.2.2 字典的键字检索操作符[] 147
 - 7.3 内建函数 148
 - 7.3.1 标准类型函数type()、str()和cmp() 148
 - 7.3.2 映射类型函数len() 150
 - 7.4 内建方法 150
 - 7.5 字典键字 152
 - 7.5.1 不允许一个键字对应一个以上的数据项 153
 - 7.5.2 键字必须是不可变的 153
 - 7.6 练习 156
- 第8章 条件语句和循环语句 159
 - 8.1 if语句 159
 - 8.1.1 多重条件表达式 159
 - 8.1.2 单语句子句 159
 - 8.2 else语句 159
 - 8.3 elif语句 161
 - 8.4 while语句 161
 - 8.4.1 一般语法 161
 - 8.4.2 计数循环 162
 - 8.4.3 无限循环 162
 - 8.4.4 单语句子句 163
 - 8.5 for语句 163
 - 8.5.1 一般语法 163
 - 8.5.2 与序列类型一起使用 163
 - 8.5.3 switch/case语句的代理 164
 - 8.5.4 range()内建函数 165
 - 8.6 break语句 167
 - 8.7 continue语句 167
 - 8.8 pass语句 168
 - 8.9 else语句之二 169
 - 8.10 练习 170
- 第9章 文件和输入/输出操作 173
 - 9.1 文件对象 173
 - 9.2 文件的内建函数 173
 - 9.3 文件的内建方法 175
 - 9.3.1 输入 175
 - 9.3.2 输出 175
 - 9.3.3 文件内移动 175
 - 9.3.4 其他 176
 - 9.3.5 其他各种文件方法 176
 - 9.4 文件的内建属性 179
 - 9.5 标准文件 179
 - 9.6 命令行参数 180
 - 9.7 文件系统 181
 - 9.8 文件的执行 186
 - 9.9 永久性存储模块 186

- 9.9.1 pickle和marshal模块 187
- 9.9.2 DBM风格的模块 187
- 9.9.3 shelve模块 187
- 9.10 相关模块 189
- 9.11 练习 190
- 第10章 错误和例外处理 193
 - 10.1 什么是例外 193
 - 10.1.1 错误 193
 - 10.1.2 例外 194
 - 10.2 Python语言中的例外 194
 - 10.3 检测和处理例外 196
 - 10.3.1 try-except语句 196
 - 10.3.2 打包一个内建函数 197
 - 10.3.3 带多个except的try语句 199
 - 10.3.4 处理多个例外的except语句 200
 - 10.3.5 不带例外名参数的try-except语句 201
 - 10.3.6 例外参数 202
 - 10.3.7 把打过包的函数用在一个应用程序里 204
 - 10.3.8 else从句 206
 - 10.3.9 try-except语句用法总结 206
 - 10.3.10 try-finally语句 207
 - 10.4 *例外的字符串形式 209
 - 10.5 *例外的类形式 210
 - 10.5.1 通过对象的标识符进行挑选 211
 - 10.5.2 例外之间的关系 211
 - 10.6 引发例外 212
 - 10.7 确认 214
 - 10.8 标准例外 215
 - 10.9 *创建例外 216
 - 10.10 为什么会发生例外 222
 - 10.11 为什么要有例外 222
 - 10.12 例外和sys模块 223
 - 10.13 相关模块 224
 - 10.14 练习 224
- 第11章 函数 226
 - 11.1 什么是函数 226
 - 11.1.1 函数与过程的比较 226
 - 11.1.2 返回值和函数类型 226
 - 11.2 函数的调用 228
 - 11.2.1 函数操作符 228
 - 11.2.2 关键字参数 228
 - 11.2.3 缺省参数 228
 - 11.3 函数的创建 229
 - 11.3.1 def语句 229
 - 11.3.2 函数声明与函数定义的比较 229
 - 11.3.3 向前引用 229
 - 11.4 函数可以用做其他函数的参数 230
 - 11.5 正式参数 231
 - 11.5.1 位置参数 232
 - 11.5.2 缺省参数 232
 - 11.6 可变长参数 235
 - 11.6.1 非关键字可变长参数 235
 - 11.6.2 关键字可变长参数 236
 - 11.6.3 调用带有可变长参数对象的函数 238
 - 11.7 函数化的程序设计 239
 - 11.7.1 匿名函数和lambda 239
 - 11.7.2 内建函数: apply()、filter()、map()、reduce() 241
 - 11.8 变量的作用范围 253
 - 11.8.1 全局变量和局部变量的比较 254
 - 11.8.2 global语句 255
 - 11.8.3 作用范围到底有几个 255
 - 11.8.4 作用范围的其他特性 256

- 11.9 *递归 257
- 11.10 练习 258
- 第12章 模块 261
 - 12.1 什么是模块 261
 - 12.2 模块和文件 261
 - 12.2.1 名字空间基本概念 261
 - 12.2.2 搜索路径和路径搜索 262
 - 12.3 名字空间 263
 - 12.3.1 名字空间与变量作用范围的比较 264
 - 12.3.2 名字的查找、确定作用范围和覆盖 264
 - 12.4 导入模块 265
 - 12.4.1 模块加载时的执行情况 265
 - 12.4.2 导入与加载的比较 266
 - 12.5 导入模块属性 266
 - 12.5.1 把名字导入当前名字空间 266
 - 12.5.2 被导入到导入者作用范围的名字 266
 - 12.6 模块的内建函数 267
 - 12.6.1 `__import__()` 267
 - 12.6.2 `globals()`和`locals()` 268
 - 12.6.3 `reload()` 268
 - 12.7 软件包 269
 - 12.7.1 目录结构 269
 - 12.7.2 软件包的`from-import`语句操作 270
 - 12.8 模块的其他特性 270
 - 12.8.1 自动加载模块 270
 - 12.8.2 阻止某个属性的导入 271
 - 12.9 练习 271
- 第13章 类和OOP 272
 - 13.1 简介 272
 - 13.2 面向对象的程序设计 277
 - 13.2.1 OOD和OOP之间的联系 278
 - 13.2.2 现实世界中的问题 278
 - 13.2.3 抽象世界里的模型 279
 - 13.3 类 281
 - 13.3.1 类的创建 282
 - 13.3.2 声明和定义的比较 282
 - 13.4 类的属性 282
 - 13.4.1 *类的数据属性 283
 - 13.4.2 方法 283
 - 13.4.3 确定类的属性 284
 - 13.4.4 类的特殊属性 285
 - 13.5 实例 286
 - 13.5.1 实例化：调用类对象创建实例 287
 - 13.5.2 `__init__()` 构造器方法 287
 - 13.5.3 `__del__()` 拆除器方法 288
 - 13.6 实例的属性 289
 - 13.6.1 “实例化”实例的属性 289
 - 13.6.2 确定实例的属性 291
 - 13.6.3 实例的特殊属性 292
 - 13.6.4 内建类型的属性 293
 - 13.6.5 实例属性和类属性的比较 293
 - 13.7 绑定和方法的调用 295
 - 13.7.1 调用绑定方法 296
 - 13.7.2 调用未绑定方法 297
 - 13.8 构造 298
 - 13.9 子类的分离和推导 299
 - 13.10 继承性 300
 - 13.10.1 类属性 `__bases__` 301
 - 13.10.2 通过继承覆盖掉方法 302
 - 13.10.3 对标准类型进行推导 303
 - 13.10.4 多重继承 304
 - 13.11 类、实例和其他对象的内建函数 304
 - 13.11.1 `issubclass()` 304

- 13.11.2 isinstance() 305
- 13.11.3 hasattr()、getattr()、setattr()、delattr() 307
- 13.11.4 dir() 308
- 13.11.5 vars() 308
- 13.12 类型和类/实例的比较 309
- 13.13 用特殊方法对类进行定制 310
- 13.13.1 对类进行简单定制的例子 312
- 13.13.2 *对类进行较复杂定制的例子 314
- 13.14 私密性 319
- 13.15 对类型进行打包 319
- 13.15.1 打包 319
- 13.15.2 实现对类型进行的打包 320
- 13.16 相关模块和文档 326
- 13.17 练习 328
- 第14章 执行环境 333
- 14.1 可调用对象 333
- 14.1.1 函数 334
- 14.1.2 方法 335
- 14.1.3 类 337
- 14.1.4 类的实例 338
- 14.2 代码对象 338
- 14.3 可执行对象语句和内建函数 339
- 14.3.1 callable() 339
- 14.3.2 compile() 340
- 14.3.3 eval() 341
- 14.3.4 exec 341
- 14.3.5 input() 346
- 14.3.6 内置字符串和intern() 347
- 14.4 执行其他Python程序 348
- 14.4.1 导入 348
- 14.4.2 execfile() 349
- 14.5 执行其他非Python程序 349
- 14.5.1 os.system() 351
- 14.5.2 os.popen()只适用于UNIX和Windows 351
- 14.5.3 os.fork()、os.exec*()、os.wait*()只适用于UNIX 352
- 14.5.4 os.spawn*()只适用于Windows 354
- 14.5.5 其他函数 354
- 14.6 受限执行环境 354
- 14.7 中断程序的执行 357
- 14.7.1 sys.exit()和SystemExit 357
- 14.7.2 sys.exitfunc() 358
- 14.7.3 os._exit()函数 359
- 14.8 相关模块 359
- 14.9 练习 360
- 第二部分 高级论题
- 第15章 规则表达式 361
- 15.1 介绍与动机 361
- 15.2 规则表达式使用的特殊符号和字符 363
- 15.3 规则表达式和Python语言 368
- 15.3.1 re模块的核心函数和方法 368
- 15.3.2 re模块的其他函数和方法 375
- 15.4 规则表达式的使用示例 379
- 15.5 练习 385
- 第16章 网络程序设计 388
- 16.1 介绍 388
- 16.1.1 什么是客户-服务器体系结构 388
- 16.1.2 客户-服务器网络程序设计 390
- 16.2 套接字：通信端点 391
- 16.2.1 什么是套接字 391
- 16.2.2 套接字地址：主机加端口 391

- 16.2.3 面向连接方式和无连接方式 392
- 16.3 使用Python语言进行网络程序设计 393
 - 16.3.1 socket()模块函数 393
 - 16.3.2 套接字对象的内建方法 393
 - 16.3.3 创建一个TCP服务器 394
 - 16.3.4 创建一个TCP客户 396
 - 16.3.5 执行TCP客户-服务器应用程序 397
 - 16.3.6 创建一个UDP服务器 398
 - 16.3.7 创建一个UDP客户 399
 - 16.3.8 执行UDP客户-服务器应用程序 400
 - 16.3.9 其他socket模块函数 401
- 16.4 相关模块 401
- 16.5 练习 402
- 第17章 多线程程序设计 404
 - 17.1 介绍 404
 - 17.2 线程和进程 405
 - 17.2.1 什么是进程 405
 - 17.2.2 什么是线程 405
 - 17.3 线程和Python 406
 - 17.3.1 全局性解释器锁 406
 - 17.3.2 退出线程 407
 - 17.3.3 从Python访问线程 407
 - 17.3.4 不使用线程时的程序设计情况 407
 - 17.3.5 Python语言中的线程化模块 408
 - 17.4 thread模块 409
 - 17.5 threading模块 412
 - 17.5.1 Thread类 413
 - 17.5.2 菲波那契数列、阶乘、连加和 418
 - 17.5.3 制造商-消费者问题和Queue模块 419
 - 17.6 练习 422
- 第18章 使用Tkinter进行GUI程序设计 423
 - 18.1 介绍 423
 - 18.1.1 什么是Tcl、Tk和Tkinter 423
 - 18.1.2 安装Tkinter并使它工作 423
 - 18.1.3 再论客户-服务器体系结构 424
 - 18.2 Tkinter和Python程序设计 424
 - 18.2.1 Tkinter模块：把Tk添加到应用程序中去 424
 - 18.2.2 GUI程序设计简介 425
 - 18.2.3 顶层窗口：Tkinter.Tk() 426
 - 18.2.4 Tk素材 426
 - 18.3 Tkinter程序示例 427
 - 18.3.1 Label素材 427
 - 18.3.2 Button素材 428
 - 18.3.3 Label和Button素材 429
 - 18.3.4 Label、Button和Scale素材 430
 - 18.3.5 中规模Tkinter程序示例 431
 - 18.4 相关模块和其他GUI 436
 - 18.5 练习 436
- 第19章 Web程序设计 438
 - 19.1 介绍 438
 - 19.1.1 网上冲浪：客户-服务器计算 438
 - 19.1.2 因特网 439
 - 19.2 用Python网上冲浪：编写简单的Web客户 441
 - 19.2.1 统一资源定位器 441
 - 19.2.2 urlparse模块 442
 - 19.2.3 urllib模块 443
 - 19.3 高级Web客户 446
 - 19.4 CGI：帮助Web服务器处理客户数据 451
 - 19.4.1 CGI简介 451
 - 19.4.2 CGI应用程序 452
 - 19.4.3 cgi模块 452

- 19.5 建立CGI应用程序 453
 - 19.5.1 制作结果网页 453
 - 19.5.2 制作表单和结果主页 456
 - 19.5.3 完全以交互方式运行的Web站点 459
- 19.6 高级CGI 464
 - 19.6.1 包含多个组成部分的表单的提交操作和文件的上传 464
 - 19.6.2 多取值输入域 465
 - 19.6.3 cookie 465
 - 19.6.4 高级CGI实战 466
- 19.7 Web服务器 474
- 19.8 相关模块 477
- 19.9 练习 478
- 第20章 扩展Python语言 483
 - 20.1 介绍 483
 - 20.1.1 什么是扩展 483
 - 20.1.2 为什么要扩展Python语言 483
 - 20.2 用编写扩展的办法扩展Python语言 484
 - 20.2.1 编写应用程序代码 484
 - 20.2.2 给代码加上程序接口 486
 - 20.2.3 编译 490
 - 20.2.4 引用的计数 494
 - 20.2.5 线程化和GIL方面的考虑 495
 - 20.3 相关论题 495
 - 20.4 练习 496
- 第三部分 附录
 - 附录A 部分练习答案 497
 - 附录B 参考信息 503
 - 附录C Python操作符汇总 511
 - 附录D Python版本2.0的新增功能 513
 - • • • • [\(收起\)](#)

[Python核心编程 下载链接1](#)

标签

python

编程

计算机

程序语言

入门

程序设计

计算机科学

评论

囧～ 第一版是01年出的 实在太老了 对于C/C++ 这还可以接受 对于python 绝对不行…… =。 =

工具书，用到翻翻

本书对一些概念讲述得很深刻，透彻，非常不错。

神一樣的书.

不错，就是比不上编程金典

针对python1.5和2.1吧，老了一点；

是我公司的同事买的，我借过来花了两周看完了，因为有前面2个月的Python基础，在看的过程中没遇到什么困难，觉得Python只是巩固了不少。

写的有点太详细了，并且很基础。作为入门级的书应该还算不错。不过就是python的语法其实本来就是比较简单的那种，所以看起来有点无趣。我是跳着看了一些章节，对于新手而言，可以看看吧。

[Python核心编程_下载链接1](#)

书评

第一眼看到这本书的时候，天哪，又是一本砖头一样的书，顿时信心少了一半，随便翻了下书，对于当时的我来说，过于深，此书并不太适合初学者来看，尤其我这种基础差的人。而且，当初看了豆瓣

的评价，立马打退堂鼓，把此书丢书架上去了。在我看完了think Python和pyth...

Python核心编程2ed 中文翻译工程成员自述 行者无疆
<http://code.google.com/p/openbookproject/wiki/CorePyZhPeople>

刚开始接触Python的时候看过几本用于入门的纸质书籍，比如《Python基础教程》、《A Byte of Python》，也在网路上上看了几个教程，比如w3cschool的《Python 基础教程》、《A Byte of Python》的译文《简明 Python 教程》，这些都是用于入门很不错的读物，当然也仅限于入门。我...

冲着CPyUG的名气买了这本书，但是这本书的质量却令我很失望。
首先，这本书的原书的勘误表就有够长的(<http://starship.python.net/crew/wesc/cpp/errata2.htm> 看看原书有多少错误！)。翻译时却几乎没有参考勘误表，把原书的所有低级错误都搬进去了。
其次，这本书的原书质量...

想买第二版，看到出了第三版，果断下单，发现买错！我仔细看了第二版和第三版的英文名，前者是Core Python Programming后者是Core Python Application Programming，显然后者注重应用的开发，同一个中文译名真是误人啊！虽然误购此书，但总体看下来，觉得这书其实不错：几乎涵...

我看的中文版，感觉太垃圾了，看到第十三章实在看不下去了。也不知道是不是翻译的问题，只能算是错别字少，行文拖沓，语言费解，挂一漏万。13章专门讲类的，居然只能在第一小节概述里面的示范代码中能看出如何实现__init__，专门讲__init__的部分居然没有细讲如何实现__init__...

这周没什么事情，索性把python继续研究下，以前读过Python基础教程，觉得那本书很一般，或者不是自己喜欢的风格，这次选了这本书，通过几天时间把第一部分自己阅读了一下，还是蛮有收获的，对python中的OO编程有了更深的理解，还有新式类方面的知识也增加了不少，第二部分只读...

对于初学者，尤其是以前对编程没什么经验的人来说，这不是一本很适合的书。书的章节内容划分不好，内容常常相互交叉。我在看第一遍的时候很多地方都是不明觉厉，只好浅尝辄止，主要是快速浏览主要内容。在使用python一两个月后，再次回过来又看了一遍，其实书里很可以...

第四页正则表达式[A-Za-z]\w+ 的含义介绍。1 You can use regular expressions, such as the one here, which recognizes valid Python identifiers. [A-Za-z]\w+ means the first character should be alphabetic, that is, either A-Z or a-z, followed by at least one (...

期待下一版可以包括更新的python版本。这本书其实读起来还是有难度。不过，对于有程序背景的同

学来说。会有更好的感悟。遇到一本书，是需要时机的。个人认为在有了好的基础的情况下才更能有所斩获。如果纯为了体验，推荐head first。如果想全面了解，官方文档应该帮助更大。再...

智普教育python周末班推荐参考教材

这本教材适合有点编程基础的python初学者，智普教育（jeapedu.com）开设的针对在职人员的python周末班推荐使用这个教材。是很好的一本由浅入深的系列教材。

对Python基础数据结构讲得很好,比如对象深拷贝、浅拷贝，list,set,dic,等，还有面向对象那块讲得很细致，尽管Python面向对象比较怪，但是面向对象那块讲得还有欠缺的地方，比如对一些内置方法没有细致地讲（PHP管这类方法叫魔术方法）。现在列出10点本人觉得比较欠缺的地方： ...

十分推荐的一本书。本书的内容实际上就是大致介绍了一下部分python标准库里的模块和一些第三方模块，并且主要是网络方向。书中讲解的都不深，可以说只是对这些模块做了简单介绍并给出一些小例子（基本上每个模块都有一个或几个配套的实例）。如果真正想做开发肯定还要单独去深...

看了一百多页了，主要有以下特点 1、内容讲的是不错，比较详细。 2、结构比较乱，不是常规入门编程书籍顺序，有点东拉西扯的感觉，不过对有基础的童鞋来说也不是坏事。 3、印刷和翻译的低级错误也比较多，许多例子的代码就是错的，少印刷几行的情况时有发生。 4、但是实在...

在百度工作的第一个月，需要用到Python来做项目。我用三天时间把这本书看完，并马上写代码付诸实践，并得到不错的效果。虽然这本书存在着这样或那样的BUG（翻译不好），但是对于快速入门的新手还是很不错的一本书的。因此，我极力推荐想在几天内快速入门Python的看这本书~

对于新手来说选择这本书做为Python语言学习的入门书籍还不错，书中从基础的语法说起，对Python的基础知识作了比较全面的介绍，不过因为是翻译过来的书，还是存在不少翻译错误的，学习过程中要自己慢慢体会啊，找其他的途径去校正错误。。。学习过程中最好还是根据书中...

这本书，严重不推荐！非常不推荐！谁推荐你看这本书，就是在坑你！

1，这本书是2016年6月第一次印刷的，现在是2020年，python都是3的版本了，书里的内容相当一部分内容还是python2的，有些库和函数在python3里已经被废弃了的。

有些示例，也不清楚地告诉你是python2还是python3...

[Python核心编程_下载链接1](#)