高等学校网络空间安全专业“十三五”规划教材

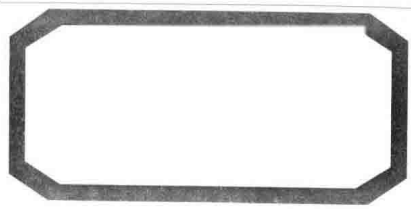
Python 安全实践

—— PythonHacking

主　编　胡建伟
副主编　崔艳鹏

西安电子科技大学出版社
http://www.xduph.com

高等学校网络空间安全专业“十三五”规划教材



Python 安全实践

——PythonHacking

主　编　胡建伟

副主编　崔艳鹏

西安电子科技大学出版社

内 容 简 介

Python 是一种魅力无限的编程语言，在网络安全、攻防渗透、大数据分析、人工智能和机器学习等几乎所有目前热门的领域里都得到了广泛的应用。Python 编程技能俨然已经成为现代信息技术人员的标配能力之一。本书结合网络攻防对抗的各个核心知识点，对 Python 易于编程、高效编程和自带丰富模块等特点进行讲解和展示，以帮助读者深度学习和深刻理解 Python 在网络渗透当中的强大功能。

本书可作为大中专院校计算机技术、网络工程、信息安全和信息对抗技术等相关专业的教材或参考书，也可供相关专业技术人员阅读与参考。

图书在版编目(CIP)数据

Python 安全实践：PythonHacking / 胡建伟主编. —西安：西安电子科技大学出版社，2019.8

ISBN 978-7-5606-5381-5

Ⅰ. ① P… Ⅱ. ① 胡… Ⅲ. ① 软件工具—程序设计 Ⅳ. ① TP311.561

中国版本图书馆 CIP 数据核字(2019)第 153108 号

策划编辑　马乐惠

责任编辑　曹　锦　阎　彬

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　西安日报社印务中心

版　　次　2019 年 8 月第 1 版　　2019 年 8 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印　张　17

字　　数　402 千字

印　　数　1～1000 册

定　　价　38.00 元

ISBN 978-7-5606-5381-5 / TP

XDUP 5683001-1

前　言

Python 作为一种常用的编程语言，不仅简单易学，而且拥有丰富的第三方支持库。特别是 Python 可与网络安全相结合，使得它成为一种黑客编程语言。不管在网络渗透测试还是网络安全运维，甚至各种 CTF 竞赛中，Python 都是主力军，其霸主地位无人能及。

近年来随着信息安全、人工智能和机器学习的快速发展，学习和使用 Python 的用户不断增加。本书的出版可在一定程度上满足读者对 Python 在网络攻防对抗领域的应用需求。

本书共八章，主要内容如下：

第一章：Python 基础，重点对 Python 的核心语法知识进行讲解，通过较为综合的 Python 实例，使读者对 Python 的基础语法有较为深刻的理解和掌握。

第二章：Python 网络编程，在介绍网络体系结构的基础上，对 Python 的 Socket 编程、面向对象的 SocketServer 编程以及功能强大的 Scapy 开发库三种网络编程方法进行详细分析和讲解。

第三章：Python 信息收集，从公开网络信息收集和主动的侦察及扫描技术两个维度，介绍 Python 在网络信息获取领域的关键技术和开发技能。

第四章：Python 协议攻击，综合应用网络协议知识、网络协议安全和 Python 基础知识，分别从协议体系分层的重点网络协议攻击方法进行讲解和分析，使读者可以对目前主流的网络协议安全有较为全面、深入的学习。

第五章：Python 运维，重点对系统的各类信息获取、核心参数指标采集以及日志的组成和分析进行初步的介绍，使读者具备一定的系统安全运维能力。

第六章：Python Web 渗透测试，重点对网站信息获取、口令暴力破解、服务端攻击技术进行分析和讨论，使读者对网站的渗透测试流程和方法有一定的了解。

第七章：Python 逆向，从软件逆向工程的角度出发，学习 Python 在 PE 文件、反汇编技术和 Hook 挂钩技术方面的技能。

第八章：Python 漏洞挖掘和利用，以实际的 FTP 服务端软件安全漏洞为例，介绍从漏洞的模糊测试发现到漏洞逆向分析，最后实现漏洞利用与开发等内容，使读者掌握基本的漏洞挖掘和利用手段。

本书由胡建伟任主编，胡门网络技术有限公司的核心团队参与编写，主要编写人员有崔艳鹏、赵伟、王坤、张玉、刘辰烁、冯璐铭、车欣、米泽宇、靳子玥、田浩帅和丁佳豪。全书由胡建伟统稿，崔艳鹏统校。

本书的出版得到了西安电子科技大学出版社的大力支持，在此对各位领导和编辑一并表示感谢。

本书的出版旨在给读者提供更多的学习机会和学习材料，也希望读者能在阅读本书的过程中有所受益。由于时间和水平有限，书中不足之处在所难免，敬请读者不吝指正，有任何问题可以与作者联系，作者 E-mail：99388073@qq.com。

不忘初心，继续前行，让我们一起开始有趣的 PythonHacking 之旅！

编　者

2019 年 4 月

目 录

第一章　Python 基础

1.1　Python 简介

Python 是一种简单易学却又功能十分强大的脚本语言。近年来随着信息安全、人工智能和机器学习的快速发展，学习和使用该语言的用户不断增加。Python 是一种解释型语言，其运行速度相对较慢，但是它拥有极高的开发效率和极其强大的内置与外置第三方库，利用 Python 可以在短时间内开发出满足要求的程序，因此 Python 在程序开发阶段节省的时间足以弥补其解释型语言运行速度低的缺陷。另外，Python 开发的程序通常都以源码形式发布，用户可以根据自己的需求修改代码，扩展性很强。

1.2　配 置 环 境

工欲善其事，必先利其器。本书大多数例子都选择网络安全领域常用的渗透平台 Kali 作为编程环境。作为目前流行的攻击渗透平台，Kali 自带 Python 开发环境，并且已经预安装了许多功能强大的第三方库，非常适合读者学习 Python 在网络渗透和安全防护方面的应用。

1.2.1　Kali 安装

Kali 目前的最新版本是 2.0，其官方下载网址为 https://www.kali.org/downloads/。

建议将 Kali 安装在虚拟机上，此时只需要在官方网站下载相应的虚拟机镜像文件。目前常用的 Kali 安装版本如图 1-1 所示，读者自行选择相应版本下载，然后在 VM(虚拟机软件)中导入就可以使用了。登录时使用 Kali 系统初始的用户名“root”和密码“toor”。

Image Name	Download	Size	Version	sha256sum
Kali Linux Light 64bit	HTTP|Torrent	867M	2018.4	ad63589f761a4344e930486e05e9d3652b8c8badb2e0f808951861d489db1f6
Kali Lunux Light Armhf	HTTP|Torrent	630M	2018.4	4b409b7f0650741400b2c3c9076333f6c52211205c4a2828d677f1099d3e5d64
Kali Linux Light 32bit	HTTP|Torrent	863M	2018.4	0659674f841d91b71bd2503e352ded588ec17d0e976c9fee4345dad35ace83b1
Kali Linux64 bit	HTTP|Torrent	3.0G	2018.4	7c65d6a319448efe4eelbe5b5a93d48ef30687d4e3f507896b46b9c2226a0ed0

图 1-1　常用的 Kali 安装版本

下载 Kali 后将其解压，并在安装完 VM 的情况下点击目录下后缀名为 vmx 的文件，即可运行 Kali 系统。打开虚拟机，出现图 1-2 所示的弹窗，点击“我已移动该虚拟机(M)”，即可继续开启虚拟机。

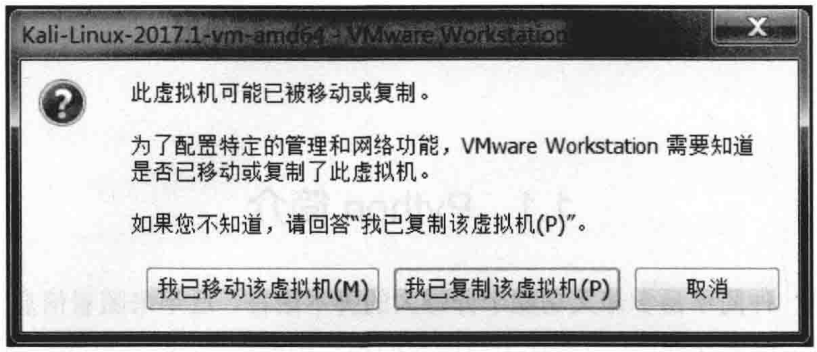


图 1-2　打开虚拟机

打开虚拟机后，以初始的用户名 root、密码 toor 登录进入 Kali 系统。如果需要对 Kali 进行更新以获得最新版本的软件及各种组件，可以使用 apt-get 命令对系统和软件包进行管理，参见图 1-3。

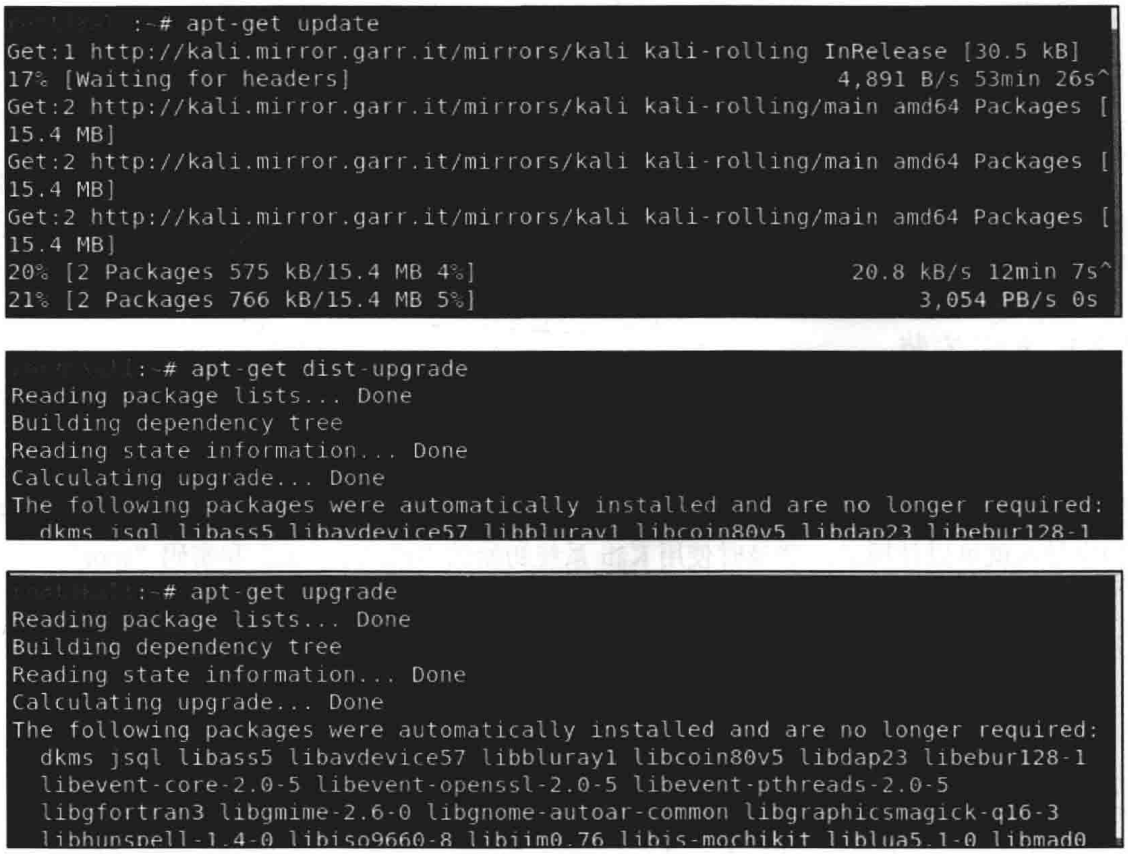


图 1-3　更新 Kali 系统中的软件

注：Kali 系统中软件包安装和管理命令 apt-get 的用法如下：

apt-get install package：安装包。

apt-get remove package：删除包。

apt-get update：更新源。

apt-get upgrade：更新已安装的包。

apt-get dist-upgrade：升级系统。

同时，为便于 Python 中各种第三方库的管理，建议安装 pip 工具。其操作步骤如下：

(1) 下载 pip 安装包：wget https://bootstrap.pypa.io/get-pip.py --no-check-certificate。

(2) 以 root 用户运行命令 python get-pip.py，即可完成安装。

(3) 如果执行 pip 命令后出现文件或者目录不存在的问题，可以通过建立符号链接来解决。

```
root@bogon:~# pip
bash: /usr/bin/pip: No such file or directory
root@bogon:~# which pip
/usr/local/bin/pip
root@bogon:~# ln -s /usr/local/bin/pip /usr/bin/pip
root@bogon:~# pip
Usage:
  pip <command> [options]
  Commands:
  install                     #安装包
  uninstall                   #卸载包
  freeze                      #按着一定格式输出已安装包列表
  list                        #列出已安装包
  show                        #显示包详细信息
  search                      #搜索包，类似 yum 里的 search
  ...
```

注：某些版本 Kali 自带的 pip 可能无法正常使用，建议先卸载 pip 包，命令如下：

root@bogon:~# apt-get remove python-pip

root@bogon:~# apt-get autoremove

1.2.2　WingIDE 安装

对于初学者，也可以选择付费软件 WingIDE 作为开发环境。WingIDE 本身使用 Python 语言开发且功能丰富、易于编程。在 Kali 中安装 WingIDE，首先需要下载其最新的 deb 安装包(下载网址为 http://wingware.com/pub/wingide)，下载完成后在相应目录下执行 dpkg -i wingide6_6.0.6-1_amd64.deb 即可完成安装，如图 1-4 所示。

```
:~# dpkg -i /root/Downloads/wingide6_6.0.6-1_amd64.deb
Selecting previously unselected package wingide6.
(Reading database ... 358871 files and directories currently installed.)
Preparing to unpack .../wingide6_6.0.6-1_amd64.deb ...
Unpacking wingide6 (6.0.6-1) ...
Setting up wingide6 (6.0.6-1) ...
Processing triggers for menu (2.1.47+b1) ...
```

图 1-4　安装 WingIDE

WingIDE 安装完成后，可以在虚拟机的“Applications”→“Usual applications”→“Programming”中找到安装好的 WingIDE，如图 1-5 所示。

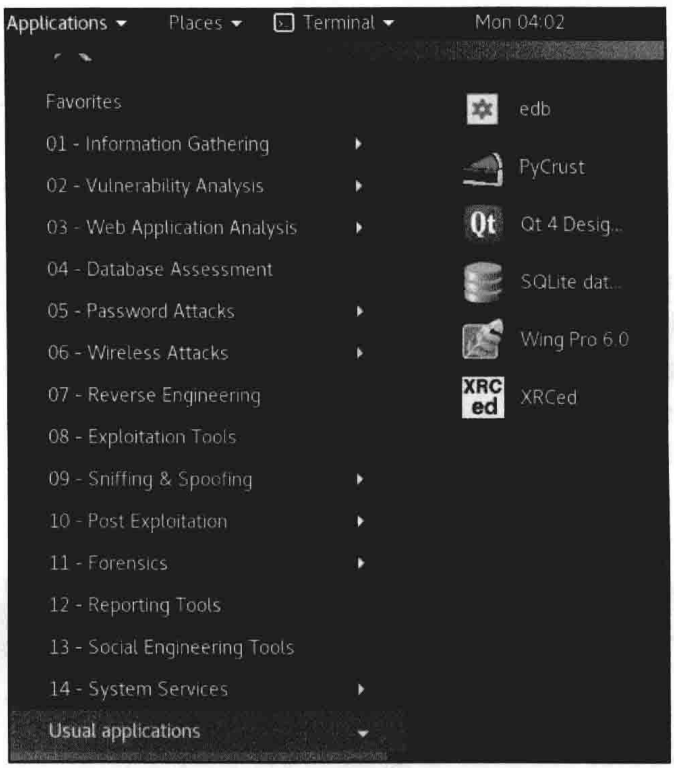


图 1-5　WingIDE 位置

在 Kali 中经常会出现 WingIDE 使用一段时间后在界面上方导航栏中找不到的情况。为了防止这种情况的出现，可以在其安装完成之后将启动快捷方式备份一份在桌面上。在“Files”→“Other Locations”→“Computer”中搜索“wingide”，将搜索结果中的“Wing Pro 6.0”置于桌面以备后用，如图 1-6 所示。

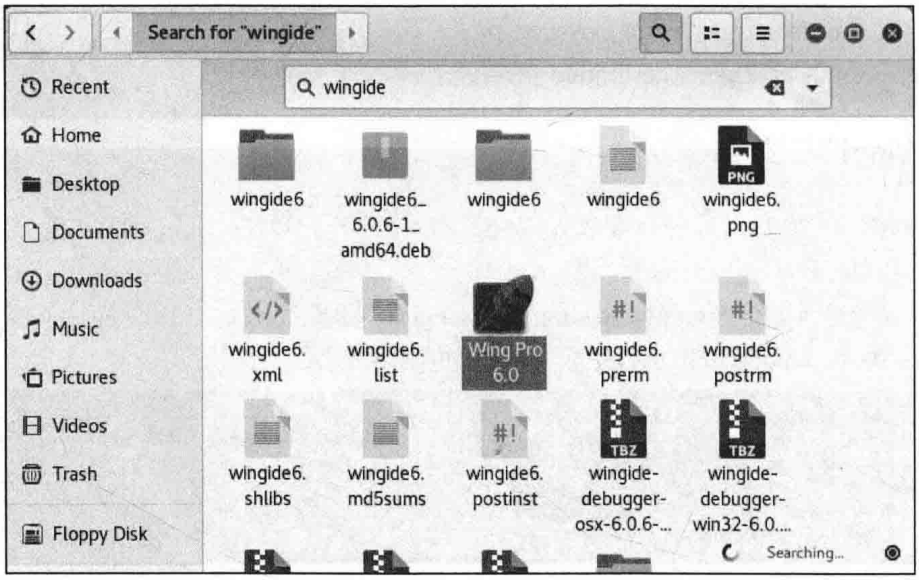


图 1-6　启动快捷方式

1.3　Python 基础语法

1.3.1　数据类型与变量

在计算机中，不同的数据需要用不同的数据类型来表示。Python 支持动态数据类型，程序员不需要提前声明数据类型，解释器会自动识别变量的数据类型。在 Python 中能够直接处理的数据类型包括整数、浮点数、字符串和布尔值，除此之外还有一些复杂的数据类型，比如列表、数组等。

注：本书演示代码采用 Python 2.7 编写。

在 Python 语言中可以使用内置函数 type 查看变量的类型，命令如下：

```
>>> age = 28
>>> type(age)
<type 'int'>                    #整型
>>> name = "hujianwei"
>>> type(name)
<type 'str'>                    #字符串
>>> age = 28.00
>>> type(age)
<type 'float'>                  #浮点数
>>> check = True
>>> type(check)
<type 'bool'>                   #布尔值
>>> nameList = ['port','banner','connect']
>>> type(nameList)
<type 'list'>                   #列表
```

type 命令用起来虽然简单，但是对于我们正确处理变量具有重要意义。例如，age 是整数时，对其用加运算(+)，代表正常的加、减、乘、除中的算术加法；但是如果 age 是字符串类型，那么加运算就意味着将两个字符串拼接在一起。因此，很多时候我们需要准确把握每次计算时变量的类型，以免出现语法或者功能错误。

程序中的变量都由一个名字来表示。变量名必须是大小写英文、数字和下划线的组合，且不能以数字开头。变量可以是任意的数据类型，它对应的数据存储在内存中，而内存中又可以存储不同类型的值。在下面的代码中，可通过 str( )函数将整数转换成字符串，然后把两个字符串连接成一个字符串。

```
>>> port = 22                           #定义一个整型变量
>>> print "this port is "+str(port)     #将变量连接在字符串中
this port is 22
```

在交互模式下，最后一个打印的表达式会赋予一个特殊变量'_'：

```
>>> ip = "8.8.8.8"
>>> ip.split('.')              #以点('.')作为分隔符来切分 ip 变量
['8','8','8','8']
>>>_
['8','8','8','8']
```

注意，变量 '_' 是只读的。

注：既然 Python 中已经有名字是 str 的函数了，就不要再用 str 作为变量的名字。

1.3.2 字符串

Python 中的字符串是以单引号 '、双引号 " 或者三引号(''' 、""")括起来的任意文本，如：'hello world'、"code" 等。单引号和双引号本质上是等价的，单、双引号都支持的好处在于字符串中一旦出现单引号或者双引号时无需用转义字符，而是用另一种引号括起来即可。单引号 ''定义字符串时，会认为字符串里面的双引号 " " 是普通字符，从而不需要转义；反之用双引号定义字符串时，就会认为字符串里面的单引号是普通字符无需转义。

```
>>>print 'hell\'o'
```

等价于

```
>>>print "hell'o"
```

三引号可以由多行组成，是编写多行文本的快捷语法，常用于文档字符串。

```
str1 = "hello, world"
```

如果要写成多行形式，可以使用反斜杠 “\ (连行符)”：

```
str2 = "hello,\
world"
```

str2 与 str1 是一样的。而对于三引号，多行可以直接书写(不用连行符)：

```
str3 = """hello,
world,
hahaha."""
```

str3 实际上就是 “hello,\nworld,\nhahaha.”，其中 “\n” 是转义字符。

使用三引号还可以在字符串中增加注释，具体如下：

```
str4 = """hello,        #在三引号的字符串内可以加注释
world,                  #这也是注释
hahaha."""
```

Python 的字符串模块提供了强大的字符串处理能力。下面通过举例来介绍一些常用的字符串处理功能：

```
>>> s = "i love code"
>>> len(s)                      #返回字符串长度
11
>>> s.find('code')              #返回匹配特定子串的起始下标
7
```

```
>>> s = "i love code"
>>> s.split( )                    #根据指定字符(默认是空格)分割字符串
['i', 'love', 'code']
>>> s.replace('code', 'study')    #将字符串 code 替换为 study
'i love study'
>>> s.upper()                     #将字符串中的小写字母转为大写字母
'I LOVE CODE'
>>> st = 'Student'
>>> st.lower()                    #将字符串中的大写字母转为小写字母
'student'
>>> st*3                          #字符串重复
'StudentStudentStudent'
```

1.3.3 列表

Python 语言不像 C 语言，并没有专用的数组类型，与其相似的概念有以下几个。

(1) list：普通的列表，初始化后可以通过特定方法动态增加元素。

定义方式：

arr = [元素]

(2) Tuple：元组，固定的数组，一旦定义后，其元素是不能修改的。

定义方式：

arr = (元素)

(3) Dictionary：字典类型，即 Hash 数组，采用键-值对的形式。

定义方式：

arr = {元素 key:values}

列表(list)是 Python 内置的一种数据类型，可以用来存储一组不同类型的数据。列表通过使用方括号括起来逗号隔开的不同的数据项即可，核心概念如图 1-7 所示。

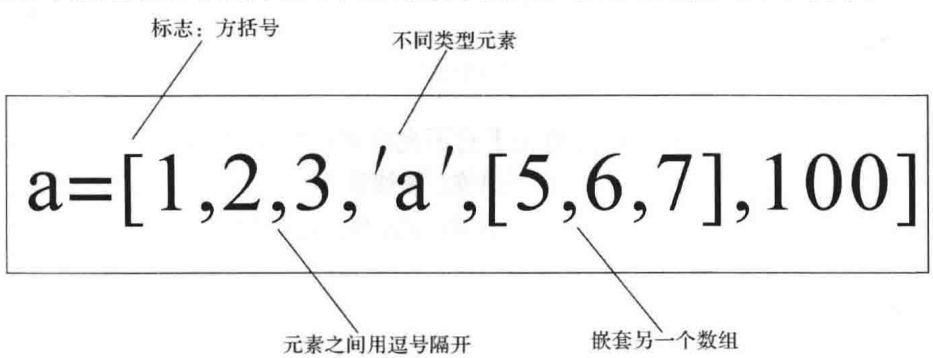


图 1-7　Python 中的 list 数组

与字符串的索引一样，列表索引从 0 开始。列表可以通过下标索引或者方括号进行截取、切片、组合等，如：

```
>>>a = [1, 2, 3]
```

那么

```
a[1]→2          #指定索引项，1 表示第二个元素
a[1:2]→[2]      #切片的范围是：包含开始位置到结束位置之前(不含结束位置)
a[1:]→[2,3]          a[:-1]→[1,2]      # -1 表示倒数第一个元素
```

Python 数组实际上是一个链表，因此定义后不能像 PHP 之类的语言一样，直接在后面追加元素，而是需要用操作链表的方法操作。常用的方法如表 1-1 所示。

表 1-1　Python 列表(list)操作方法

方　法	含　义
list[index]	index 为列表下标，从 0 开始，list[index]是指下标 index 对应的列表 list 中的元素。 list[0:3]是列表中下标 0 到下标 3(不含 3)对应的元素
del list[index] remove( )	删除列表 list 中下标为 index 的元素
list1+list2	可以用“+”直接将两个列表拼接在一起
append( )	在列表末尾添加新的元素
insert(index,str)	将 str 插入到列表 list 中下标为 index 的位置
cmp(list1,list2)	比较两个列表中的元素个数，若 list1 = list2，返回 0；若 list1 > list2，返回 1，否则返回 –1
count(str)	统计 str 在列表 list 中出现的次数
sort( )	对列表进行排序
extend(L)	将给定的列表 L 接到当前列表后面，等价于 a[len(a):] = L
index(x)	返回列表中第一个值为 x 的项的索引。如果没有匹配的项，则产生一个错误

在上述例子中，如果用 a[2] = 'c' 则可以改变第三个元素的值；但如果用 a[3] = 'd' 则增加一个元素是会出错的，此时应该用 a.append('d')或 a.insert(任意位置，'d')增加元素。

1.3.4　元组

元组与列表类似，最大的不同之处在于它不允许修改元组内的元素，如下所示：

```
>>> a = (1, 2, 3, 4)                    #注意：圆括号
>>> a[1] = 3                            #注意：无法修改元素的值
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

1.3.5　字典

除了列表外，字典也是 Python 的一种内置数据类型，用 { } 来表示，其元素为键-值形

式，通过键来找其对应的值，字典中没有索引。字典的有关语法点如图 1-8 所示。

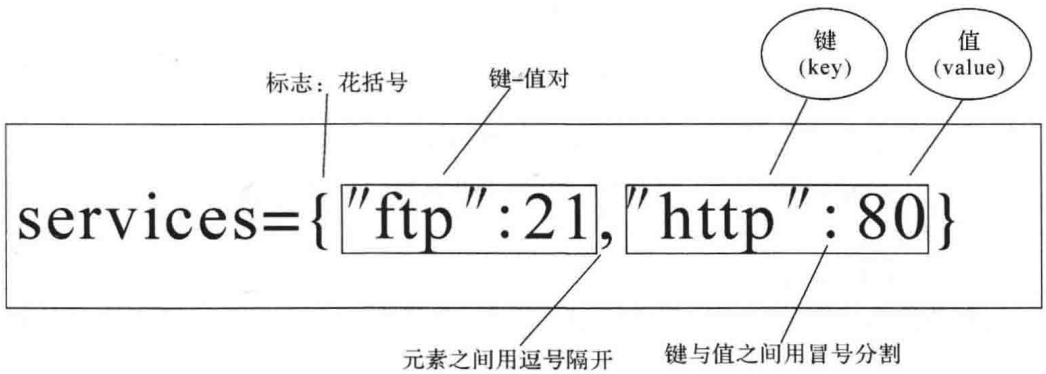


图 1-8　Python 中的字典

可通过以下面代码来学习字典的用法：

```
>>> services = {'ftp':21, 'ssh':22, 'smtp':25, 'http':80}    #创建字典，包含四对元素
>>> services.keys()                    #返回字典中所有的键的列表
['ftp', 'smtp', 'ssh', 'http']
>>> services.has_key('ftp')            #判断字典中是否包含键 ftp
True
>>> services['ftp']                    #查找键 ftp 所对应的值
21
>>> services.pop('ftp')                #删除键为 ftp 的元素
>>> services.clear()                   #清空字典中的内容
>>> print services
{}
```

注：由于 Python 语言是自动确定变量类型的，因此读者可以通过内置的 type 函数和 dir 函数获取变量类型信息以及该类型变量所支持的方法。

1.3.6　控制语句

和其他计算机语言一样，Python 语言的控制语句主要有分支语句和循环语句两种。

1. 分支语句

Python 中条件选择语句的关键字为 if、elif 和 else，其基本形式如下：

```
if 判断条件(condition):
    语句块(block)          # Python 中不能用括号来表示语句块，也不能用开始/结束标志符
                           #来表示，而是靠缩进来表示
elif condition:
    语句块(block)          #空白在 Python 中是重要的。事实上行首的空白是重要的
                           #行首的空白称为缩进。在逻辑行首的空白(空格和制表符)
                           #用来决定逻辑行的缩进层次，从而用来决定语句的分组
...
```