



# Python语言基础与应用

计算和控制流 / 引用扩展模块

陈斌 北京大学 [gischen@pku.edu.cn](mailto:gischen@pku.edu.cn)

# 引用扩展模块

- › 调用模块化的工具
- › Python语言标准库
- › 扩展命名空间

# 调用模块化的工具

## › 模块就是程序

每个扩展名为 `.py` 的 Python 程序都是一个独立的模块 (Module)

模块能定义函数、类和变量，让你能够有逻辑地组织你的 Python 代码段

## › 组织模块

包 (package) 是放在一个文件夹里的模块集合

# 调用模块化的工具

## › 模块引用方式

**import** <模块> [as <别名>]

将模块中的函数等名称导入当前程序

在调用模块中的函数的时候，需要加上模块的命名空间

可以给导入的命名空间替换一个新的名字

引用方法：<模块>.<名称>

**from** <模块> **import** <函数>

引入模块中的某个函数

调用时不需要再加上命名空间

# Python语言标准库

## › 标准库

在安装Python时就默认已经安装好的模块，  
Python具有功能强大的标准库

## › 数字和数学模块

numbers: 数字抽象基类

math: 数学函数

cmath: 复数的数学函数

decimal: 十进制定点和浮点算术

fractions: 有理数

random: 生成伪随机数

statistics: 数学统计功能

# Python语言标准库

## › 数据类型

**datetime:** 基本日期和时间类型

**calendar:** 与日历相关的一般功能

**collections:** 容器数据类型

**heapq:** 堆队列算法

**bisect:** 数组二分算法

**array:** 高效的数值数组

**weakref:** 弱引用

**types:** 动态类型创建和内置类型的名称

**copy:** 浅层和深层复制操作

**pprint:** 格式化输出

**reprlib:** 备用repr()实现

**enum :** 支持枚举

# Python语言标准库

## › 功能编程模块

`itertools`: 为高效循环创建迭代器的函数

`functools`: 可调用对象的高阶函数和操作

`operator`: 标准运算符作为函数

## › 数据持久化

`pickle`: Python对象序列化

`copyreg`: 注册`pickle`支持功能

`shelve`: Python对象持久化

`marshal`: 内部Python对象序列化

`dbm`: 与Unix“数据库”的接口

`sqlite3`: SQLite数据库的DB-API 2.0接口

# Python语言标准库

## › 数据压缩和存档

**zlib:** 与gzip兼容的压缩

**gzip/bz2:** 支持gzip/bzip2文件

**lzma:** 使用LZMA算法进行压缩

**zipfile:** 使用ZIP存档

**tarfile:** 读取和写入tar归档文件

## › 文件格式

**csv:** CSV文件读写

**configparser:** 配置文件解析器

**netrc:** netrc文件处理

**xdrllib:** 对XDR数据进行编码和解码

**plistlib:** 生成并解析Mac OS X.plist文件

# Python语言标准库

## › 文件和目录访问

**pathlib**: 面向对象的文件系统路径

**os.path**: 常见的路径名操作

**fileinput**: 迭代多个输入流中的行

**stat**: 解释stat()结果

**filecmp**: 文件和目录比较

**tempfile**: 生成临时文件和目录

**glob**: Unix样式路径名模式扩展

**fnmatch**: Unix文件名模式匹配

**linecache**: 随机访问文本行

**shutil**: 高级文件操作

**macpath**: Mac OS 9路径操作函数

# Python语言标准库

## › 通用操作系统服务

**os:** 其他操作系统接口

**io:** 用于处理流的核心工具

**time:** 时间访问和转换

**argparse:** 用于命令行选项，参数和子命令的解析器

**getopt:** 用于命令行选项的C风格解析器

**logging:** Python的日志记录工具

**getpass:** 便携式密码输入

**curses:** 字符单元格显示的终端处理

**platform:** 访问底层平台的标识数据

**errno:** 标准errno系统符号

**ctypes:** Python的外部函数库

# Python语言标准库

## › 并发执行

`threading`: 基于线程的并行性

`multiprocessing`: 基于进程的并行性

`concurrent.futures`: 启动并行任务

`subprocess`: 子流程管理

`sched`: 事件调度程序

`queue`: 同步的队列类

`_thread`: 低级线程API

## › 加密服务

`hashlib`: 安全哈希和消息摘要算法接口

`hmac`: 用于消息身份验证的密钥哈希算法

`secrets`: 生成用于管理机密的安全随机数

# Python语言标准库

## › 网络和进程间通信

**asyncio**: 异步I/O

**socket**: 低级网络接口

**ssl**: 套接字对象的TLS/SSL包装器

**select**: 等待I/O完成

**selectors**: 高级I/O复用

**asyncore**: 异步套接字处理程序

**asynchat**: 异步套接字命令/响应处理程序

**signal**: 设置异步事件的处理程序

**mmap**: 内存映射文件支持

# Python语言标准库

## › 互联网数据处理

**email:** 电子邮件和MIME处理包

**json:** JSON编码器和解码器

**mailcap:** Mailcap文件处理

**mailbox:** 以各种格式处理邮箱

**mimetypes:** 将文件名映射到MIME类型

**base64:** Base16/Base32/Base64/Base85数据编码

**binhex:** 对binhex4文件进行编码和解码

**binascii:** 在二进制和ASCII之间转换

**quopri:** 对MIME引用的可打印数据进行编码和解码

**uu:** 对uuencode文件进行编码和解码

# Python语言标准库

## › 互联网协议和支持

**webbrowser**: Web浏览器控制器

**cgi**: 通用网关接口支持

**cgitb**: CGI脚本的回溯管理器

**wsgiref**: WSGI实用程序和参考实现

**urllib**: URL处理模块

**http**: HTTP模块

**ftplib/poplib/imaplib/nntplib/smtplib**:

FTP/POP3/IMAP4/NNTP/SMTP协议客户端

**smtpd**: SMTP服务器

**telnetlib**: Telnet客户端

**socketserver**: 网络服务器的框架

**xmlrpc**: XMLRPC服务器和客户端模块

**ipaddress**: IPv4/IPv6操作库

# Python语言标准库

## › 多媒体服务

**audioop**: 处理原始音频数据

**aifc**: 读写AIFF和AIFC文件

**sunau**: 读取和写入Sun AU文件

**wave**: 读写WAV文件

**chunk**: 读取IFF分块数据

**colorsys**: 颜色系统之间的转换

**imghdr**: 确定图像的类型

**sndhdr**: 确定声音文件的类型

**ossaudiodev**: 访问兼容OSS的音频设备

# Python语言标准库

## › 结构化标记处理工具

html: 超文本标记语言支持

xml: XML处理模块

## › 程序框架

turtle – 海龟作图库

cmd –支持面向行的命令解释器

shlex –简单的词法分析

## › 图形用户界面

tkinter: Tcl/Tk的Python接口

# 扩展命名空间

## › 命名空间(namespace)

- 表示标识符(identifier)的可见范围
- 一个标识符可以在多个命名空间中定义，在不同命名空间中的含义互不相干

**dir(<名称>)函数：** 列出名称的属性

**help(<名称>)函数：** 显示参考手册

# 扩展命名空间

```
>>> import time
>>> dir(time)
['_STRUCT_TM_ITEMS', '__doc__', '__loader__', '__name__', '__package__', '__spec__', 'altzone', 'asctime', 'clock', 'ctime', 'daylight', 'get_clock_info', 'gmtime', 'localtime', 'mktime', 'monotonic', 'perf_counter', 'process_time', 'sleep', 'strptime', 'strftime', 'struct_time', 'time', 'timezone', 'tzname', 'tzset']
>>> time.tzname
('CST', 'CST')
>>> help(time.time)
Help on built-in function time in module time:

time(...)
    time() -> floating point number

    Return the current time in seconds since the Epoch.
    Fractions of a second may be present if the system
    clock provides them.

>>> print(time.time())
1490280256.450634
```