**1.变量及函数**

**\_\_name\_\_**

每一个模块都有一个默认的属性\_\_name\_\_。

1、直接运行一个python的程序的时候，如python test.py, 则\_\_name\_\_ 的值是 \_\_main\_\_。

2、一个python模块被import的时候，则\_\_name\_\_的值为模块的文件名（不带路径及文件扩展名）。

\_\_name\_\_值的使用：用于处理初始化

**\_\_build\_class\_\_**

用类声明的内部帮助器函数

**\_\_builtins\_\_**

\_\_builtins\_\_即是引用，那么它内建模块有一个相同点：Python程序一旦启动，程序员所写的代码没有运行之前就已经被加载到内存中了。

**classmethod** 类，类方法

**isinstance 内建函数，**是否是指定对象或类型的实例

**object 类，**对象

**max 内建函数，**返回参数的最大值

**StopAsyncIteration 类，**停止异步迭代

**FloatingPointError 类，浮点错误**

**ZeroDivisionError 类，零除法错误**

**zip 类，打包解压元组**

**ConnectionRefusedError 类，连接被拒绝错误**

**set 类，集合**

**exit 退出**

**\_\_spec\_\_ 模块规格**

**ArithmeticError 类，算术错误**

**abs 内建函数，绝对值**

**FutureWarning 类，关于构造将来语义会有改变的警告**

**callable 内建函数，可调用?**

**OverflowError 类，溢出错误**

**AssertionError 类，断言语句失败**

**None None**

**license 许可证。若查看许可协议全文文本的类型 license()**

**GeneratorExit 类，生成器(generator)发生异常来通知退出**

**NameError类，未声明/初始化对象 (没有属性)**

**PermissionError 类，权限错误**

**SystemExit 类，解释器请求退出**

**FileNotFoundError 类，文件找不到错误**

**KeyError 类，映射键不存在**

**filter 类，过滤器**

**UnicodeError 类，Unicode 相关的错误**

**input 内建函数，输入**

**UnicodeTranslateError 类，Unicode转换错误**

**UnicodeEncodeError 类，Unicode编码错误**

**compile 内建函数，编译**

**TypeError 类，对类型无效的操作**

**LookupError 类，无效数据查询的基类**

**setattr 内建函数,赋值对象的属性**

**super 类，父类**

**SyntaxWarning 类，Python语法错误**

**NotImplementedError 类，尚未实现方法**

**round 内建函数，四舍五入**

**\_\_debug\_\_ True**

**ProcessLookupError 类，进程查询错误**

**TabError 类，Tab和空格混用**

**bin 内建函数，二进制数据**

**FileExistsError 类，文件已存在错误**

**InterruptedError 类，中断错误**

**str 类，字符串**

**locals 内建函数，局部变量字典**

**slice 类，切片**

**eval 内建函数，将字符串str当成有效的表达式来求值并返回计算结果。**

**Ellipsis 省略号**

**IndexError 类，索引超出序列范围**

**MemoryError 类，内存溢出错误**

**UnboundLocalError访问未初始化的本地变量**

**False False**

**complex 类，复数**

**hasattr 内建函数，对象属性是否存在**

**ChildProcessError 子进程错误**

**BrokenPipeError 类，管道破裂错误**

**tuple 类，元组**

**quit 退出**

**\_\_build\_class\_\_**

**frozenset 类，冻结的集合，不可变，hash ，易查找**

**LookupError 无效数据查询的基类**

**NotADirectoryError 非目录错误**

**divmod 内建函数，商和余数**

**len 返回字符串长度**

**dir 内建函数，收集参数信息**

**issubclass 内建函数，子类判断**

**AttributeError 属性错误**

**\_ None**

**hex 十六进制**

**oct 八进制**

**exec 执行字符串或文件语句**

**property 属性**

**staticmethod**

|  |  |
| --- | --- |
| **异常名称** | **描述** |
|  |  |
| BaseException | 所有异常的基类 |
| SystemExit | 解释器请求退出 |
| KeyboardInterrupt | 用户中断执行(通常是输入^C) |
| Exception | 常规错误的基类 |
| StopIteration | 迭代器没有更多的值 |
| GeneratorExit | 生成器(generator)发生异常来通知退出 |
| StandardError | 所有的内建标准异常的基类 |
| ArithmeticError | 所有数值计算错误的基类 |
| FloatingPointError | 浮点计算错误 |
| OverflowError | 数值运算超出最大限制 |
| ZeroDivisionError | 除(或取模)零 (所有数据类型) |
| AssertionError | 断言语句失败 |
| AttributeError | 对象没有这个属性 |
| EOFError | 没有内建输入,到达EOF 标记 |
| EnvironmentError | 操作系统错误的基类 |
| IOError | 输入/输出操作失败 |
| OSError | 操作系统错误 |
| WindowsError | 系统调用失败 |
| ImportError | 导入模块/对象失败 |
| LookupError | 无效数据查询的基类 |
| IndexError | 序列中没有此索引(index) |
| KeyError | 映射中没有这个键 |
| MemoryError | 内存溢出错误(对于Python 解释器不是致命的) |
| NameError | 未声明/初始化对象 (没有属性) |
| UnboundLocalError | 访问未初始化的本地变量 |
| ReferenceError | 弱引用(Weak reference)试图访问已经垃圾回收了的对象 |
| RuntimeError | 一般的运行时错误 |
| NotImplementedError | 尚未实现的方法 |
| SyntaxError | Python 语法错误 |
| IndentationError | 缩进错误 |
| TabError | Tab 和空格混用 |
| SystemError | 一般的解释器系统错误 |
| TypeError | 对类型无效的操作 |
| ValueError | 传入无效的参数 |
| UnicodeError | Unicode 相关的错误 |
| UnicodeDecodeError | Unicode 解码时的错误 |
| UnicodeEncodeError | Unicode 编码时错误 |
| UnicodeTranslateError | Unicode 转换时错误 |
| Warning | 警告的基类 |
| DeprecationWarning | 关于被弃用的特征的警告 |
| FutureWarning | 关于构造将来语义会有改变的警告 |
| OverflowWarning | 旧的关于自动提升为长整型(long)的警告 |
| PendingDeprecationWarning | 关于特性将会被废弃的警告 |
| RuntimeWarning | 可疑的运行时行为(runtime behavior)的警告 |
| SyntaxWarning | 可疑的语法的警告 |
| UserWarning | 用户代码生成的警告 |

**\_\_debug\_\_**

This constant is true if [**Python**](http://lib.csdn.net/base/python) was not started with an [*-O*](https://docs.python.org/2/using/cmdline.html#cmdoption-O) option.See also the[assert](https://docs.python.org/2/reference/simple_stmts.html#assert) statement.

Note

The names [None](https://docs.python.org/2/library/constants.html#None) and [\_\_debug\_\_](https://docs.python.org/2/library/constants.html#__debug__) cannot be reassigned(assignments to them, even as an attribute name, raise[SyntaxError](https://docs.python.org/2/library/exceptions.html#exceptions.SyntaxError)),so they can be considered “true” constants.

Changed in version 2.7:Assignments to \_\_debug\_\_ as an attribute became illegal.

**如果Python没有启用-O 默认为True。也可以参阅assert语句。**

**名字为None并且\_\_debug\_\_不能被重新设置（设置他们，甚至作为一个属性名字，抛出SyntaxError），因此他们被认为“True”常量。**

**\_\_doc\_\_**

模块本身是一个对象，而每个对象都会有一个\_\_doc\_\_属性。该属性用于描述该对象的作用。文档字符串，又称为DocStrings。用它可以为我们的模块、类、函数等添加说明性的文字，使程序易读易懂，更重要的是可以通过Python自带的标准方法将这些描述性文字信息输出。上面提到的自带的标准方法就是\_doc\_前后各两个下划线。注：当不是函数、方法、模块等调用doc时，而是具体对象调用时，会显示此对象从属的类型的构造函数的文档字符串。

**\_\_loader\_\_**

一个由其加载程序在导入的模块设置的属性。访问它应该返回loader对象本身。

**zip类**

zip([seql, ...])接受一系列可迭代对象作为参数，将对象中对应的元素打包成一个个tuple（元组），然后返回由这些tuples组成的list（列表）。若传入参数的长度不等，则返回list的长度和参数中长度最短的对象相同。

1 >>> z1=[1,2,3]

2 >>> z2=[4,5,6]

3 >>> result=zip(z1,z2)

4 >>> result

5 [(1, 4), (2, 5), (3, 6)]

6 >>> z3=[4,5,6,7]

7 >>> result=zip(z1,z3)

8 >>> result

9 [(1, 4), (2, 5), (3, 6)]

zip()配合\*号操作符,可以将已经zip过的列表对象解压。

1 >>> zip(\*result)

2 [(1, 2, 3), (4, 5, 6)]

**filter类**

filter(function or None, iterable) 🡪 过滤对象

返回一个迭代器得到iterable中执行function（item）是真的items。如果函数是没有的，返回是真的items。

**super类**

| super() -> same as super(\_\_class\_\_, <first argument>)

| super(type) -> unbound super object 不绑定父类对象

| super(type, obj) -> bound super object; requires isinstance(obj, type) 绑定父类对象，obj是type类型

| super(type, type2) -> bound super object; requires issubclass(type2, type) 绑定父类对象 要求type2是type的子类

| \_\_get\_\_(self, instance, owner, /)

| Return an attribute of instance, which is of type owner.

| 返回实例的一个属性，该类型是属性所有者的

|

| \_\_getattribute\_\_(self, name, /)

| Return getattr(self, name).

|

| \_\_init\_\_(self, /, \*args, \*\*kwargs)

| Initialize self. See help(type(self)) for accurate signature.初始化自己。查看type(self)帮助关于精准签名。

|

| \_\_new\_\_(\*args, \*\*kwargs) from builtins.type

| Create and return a new object. See help(type) for accurate signature.

|

| \_\_repr\_\_(self, /)

| Return repr(self).

**bin(number, /)**

Return the binary representation of an integer.

返回一个整数的二进制表示形式

**callable()**

检查对象object是否可调用。如果返回True，object仍然可能调用失败；但如果返回False，调用对象ojbect绝对不会成功。

类是可调用的，而类的实例实现了\_\_call\_\_()方法才可调用。

Return True if the object argument appears callable, False if not. If this returns true, it is still possible that a call fails, but if it is false, calling object will never succeed. Note that classes are callable (calling a class returns a new instance); class instances are callable if they have a \_\_call\_\_() method.

如果对象参数表现为可调用返回True，否则返回False。如果返回True，仍可能调用失败，但是如果False，调用对象将永不会成功。注意类是可调用的（调用一个类返回一个新的实例）；如果他们有一个\_\_call\_\_()方法，类的实例是可调用的。

**compile(**source, filename, mode[, flags[, dont\_inherit]**)**

将source编译为代码或者AST对象。代码对象能够通过exec语句来执行或者eval()进行求值。

参数source：字符串或者AST（Abstract Syntax Trees）对象。

参数 filename：代码文件名称，如果不是从文件读取代码则传递一些可辨认的值。

参数model：指定编译代码的种类。可以指定为 ‘exec’,’eval’,’single’。

AST (Abstract Syntax Tree) 抽象语法树

**complex([real[, imag]])**

创建一个值为real + imag \* j的复数或者转化一个字符串或数为复数。如果第一个参数为字符串，则不需要指定第二个参数。

参数real: int, long, float或字符串；

**dir()**

不带参数时，返回当前范围内的变量、方法和定义的类型列表；带参数时，返回参数的属性、方法列表。如果参数包含方法\_\_dir\_\_()，该方法将被调用。如果参数不包含\_\_dir\_\_()，该方法将最大限度地收集参数信息。

**参数object**： 对象、变量、类型。

**版本**：该函数在python各个版本中都有，但是每个版本中显示的属性细节有所不同。使用时注意区别。

Without arguments, return the list of names in the current local scope. With an argument, attempt to return a list of valid attributes for that object.

If the object has a method named \_\_dir\_\_(), this method will be called and must return the list of attributes. This allows objects that implement a custom \_\_getattr\_\_() or \_\_getattribute\_\_() function to customize the way dir() reports their attributes.

If the object does not provide \_\_dir\_\_(), the function tries its best to gather information from the object’s \_\_dict\_\_ attribute, if defined, and from its type object. The resulting list is not necessarily complete, and may be inaccurate when the object has a custom \_\_getattr\_\_().

The default dir() mechanism behaves differently with different types of objects, as it attempts to produce the most relevant, rather than complete, information:

If the object is a module object, the list contains the names of the module’s attributes.

If the object is a type or class object, the list contains the names of its attributes, and recursively of the attributes of its bases.

Otherwise, the list contains the object’s attributes’ names, the names of its class’s attributes, and recursively of the attributes of its class’s base classes.

The resulting list is sorted alphabetically.

如果没有参数，返回当前本地范围内的名称列表。用一个参数，试图返回该对象的有效属性列表。

如果对象有一个方法叫\_\_dir\_\_(),这个方法将被调用，必须返回属性列表。这允许实现一个自定义的\_\_getattr\_\_()或\_\_getatrribute\_\_()函数来定制dir()报告它们属性的方法。

如果对象没有提供\_\_dir\_\_()，函数尽量尝试从对象的\_\_dict\_\_属性收集信息，如果定义了，并从其类型的对象收集。这个结构并不一定完整，当对象有一个自定义的\_\_getatt\_\_()，可能就不正确了。

默认的dir()机制的行为不同于不同类型的对象，因为它是同产生最相关的但不完整的信息:

如果对象是一个模块对象，列表包含模块属性的名字。

如果对象是一个类型或类的对象，列表包含它的属性的名字，和以递归方式的它基类的属性。

否则，该列表包含对象属性的名字、其类的属性的名字和以递归方式的类的基类属性。

生成的列表是按照字母顺序排序的。

>>> import struct

>>> dir() # show the names in the module namespace

['\_\_builtins\_\_', '\_\_doc\_\_', '\_\_name\_\_', 'struct']

>>> dir(struct) # show the names in the struct module

['Struct', '\_\_builtins\_\_', '\_\_doc\_\_', '\_\_file\_\_', '\_\_name\_\_', '\_\_package\_\_', '\_clearcache', 'calcsize', 'error', 'pack', 'pack\_into', 'unpack', 'unpack\_from']

>>> class Shape(object):

def \_\_dir\_\_(self):

return ['area', 'perimeter', 'location']

>>> s = Shape()

>>> dir(s)

**eval()**

将字符串str当成有效的表达式来求值并返回计算结果。

**getattr(object, name[,default])**

获取对象object的属性或者方法，如果存在打印出来，如果不存在，打印出默认值，默认值可选。需要注意的是，如果是返回的对象的方法，返回的是方法的内存地址，如果需要运行这个方法，可以在后面添加一对括号。

**hasattr(object, name)**

本函数是用来判断对象object的属性（name表示）是否存在。如果属性（name表示）存在，则返回True，否则返回False。参数object是一个对象，参数name是一个属性的字符串表示。

**isinstance()**

isinstance是Python中的一个内建函数。

isinstance(object, classinfo)

如果参数object是classinfo的实例，或者object是classinfo类的子类的一个实例， 返回True。如果object不是一个给定类型的的对象， 则返回结果总是False。

如果classinfo不表示一个类（类型对象）, 那么它要么是一个类的元组, 或者递归地包含这样的（由数据类型构成的）元组.其他的序列类型是不被允许的。

如果classinfo不是一种数据类型或者由数据类型构成的元组，将引发一个TypeError异常。

**locals()**

Return a dictionary containing the current scope's local variables.

NOTE: Whether or not updates to this dictionary will affect name lookups in the local scope and vice-versa is \*implementation dependent\* and not covered by any backwards compatibility guarantees.

返回一个包含当前作用域的本地变量的字典。

注意︰更新到这本字典是否会影响名称查找在局部范围，反之亦然是\*执行依赖\*和不受任何向后兼容性保证。

**round()**

round( x [, n] )

返回X小数点四舍五入到n个数字。

**setattr(object, name, values)**

给对象的属性赋值，若属性不存在，先创建再赋值。