

# Python语言基础与应用

高级特性/面向对象：类定义中的特殊方法

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



# 面向对象：类定义中的特殊方法

- › 基本概念
- › 构造与解构
- › 算术运算
- › 其他特殊方法

# 基本概念

## › 特殊方法(special method)

也被称作魔术方法(magic method)

在类定义中实现一些特殊方法，可以方便地使用python中一些内置操作

所有特殊方法的名称以两个下划线(\_\_)开始和结束

# 构造与解构

## › 对象构造器

`__init__(self, [...])`

对象的构造器，实例化对象时调用

## › 析构器

`__del__(self, [...])`

销毁对象时调用



# 构造与解构

```
from os.path import join

class FileObject:
    '''给文件对象进行包装从而确认在删除时文件流关闭'''

    def __init__(self, filepath='~', filename='sample.txt'):
        #读写模式打开一个文件
        self.file = open(join(filepath, filename), 'r+')

    def __del__(self):
        self.file.close()
        del self.file
```

# 算术运算

## › 算术操作符

`__add__(self, other)`: 使用+操作符

`__sub__(self, other)`: 使用-操作符

`__mul__(self, other)`: 使用\*操作符

`__div__(self, other)`: 使用/操作符

## › 反运算

当左操作数不支持相应的操作时被调用

`__radd__(self, other)`, `__rsub__(self, other)`

`__rmul__(self, other)`, `__rdiv__(self, other)`

# 算术运算

## › 大小比较

`__eq__(self, other)`: 使用`==`操作符

`__ne__(self, other)`: 使用`!=`操作符

`__lt__(self, other)`: 使用`<`操作符

`__gt__(self, other)`: 使用`>`操作符

`__le__(self, other)`: 使用`<=`操作符

`__ge__(self, other)`: 使用`>=`操作符

# 算术运算

```
13     __add__ = add
14
15     def __str__(self):
16         return "F<%s,%s>" % (self.fx, self.fy)
17
18     def __mul__(self, n):
19         x, y = self.fx * n, self.fy * n
20         return Force(x, y)
21
22     def __eq__(self, force2):
23         return (self.fx == force2.fx) and \
24             (self.fy == force2.fy)
```



# 算术运算

```
37 # 操作符使用
38 f3 = f1 + f2
39 print("Fadd=%s" % (f3,))
40 f3 = f1 * 4.5
41 print("Fmul=%s" % (f3,))
42 print("%s==%s? -> %s" % (f1, f2, f1 == f2))
```

Fadd=F<3,5>

Fmul=F<0.0,4.5>

F<0,1>==F<3,4>? -> False

# 其他特殊方法

## › 字符串操作

不仅数字类型可以使用像+(\_\_add\_\_())和-(\_\_sub\_\_())的数学运算符，例如字符串类型可以使用+进行拼接，使用\*进行复制

`__str__(self)`: 自动转换为字符串

`__repr__(self)`: 返回一个用来表示对象的字符串

`__len__(self)`: 返回元素个数

## › 其它特殊方法参见课程网站

<http://gis4g.pku.edu.cn/python-magic-method/>