# PYTHON 结程基础

# 动态扩展类与实例 和\_/lot/\_\_变量

Python作为一种动态语言,除了可以在定义类时定义属性和方法外,还可以动态地为已经创建的对象绑定新的属性和方法。

在给对象绑定方法时,需要使用types模块中的MethodType方法, 其第一个参数是要绑定的函数名,第二个参数是绑定的对象名。

例:绑定新方法示例。



#### 动态扩展类与实例

- from types import MethodType #从types模块中导入MethodType方法
  class Student: #定义学生类
  pass
- 4 def SetName(self,name): #定义SetName函数
- 5 self.name=name
- 6 def SetSno(self,sno): #定义SetSno函数
- 7 self.sno=sno



```
if name ==' main ':
       stu1=Student() #定义Student类对象stu1
       stu2=Student() #定义Student类对象stu2
10
       stu1.SetName=MethodType(SetName,stu1)
11
                       #为stu1对象绑SetName方法
       Student.SetSno=SetSno #为Student类绑定SetSno方法
12
       stu1.SetName('李晓明')
13
       stu1.SetSno('1810100')
14
       #stu2.SetName('张刚') #取消注释则会报错
15
       stu2.SetSno('1810101')
16
```



#### 提示

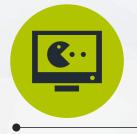
给一个对象绑定方法后,只能通过该对象调用该方法,其他未绑定该方法的对象则不能调用。例如,没有为stu2对象绑定SetName方法,因此,如果将第15行前面的注释符取消,则程序运行时系统会报错。

而第12行为Student类绑定了SetSno方法,则Student类中的所有实例对象都有该方法。

#### \_\_slots\_\_

- 一 在定义类时, Python提供了\_\_slots\_\_变量以限制可动态扩展的属性。
- \_\_slots\_\_中所做的动态扩展属性限制只对\_\_slots\_\_所在类的实例对象有效。
- 如果子类中没有\_\_slots\_\_定义,则子类的实例对象可以进行任意属性的动态扩展。
- 如果子类中有\_\_slots\_\_定义,则子类的实例对象可动态扩展的属性包括子类中通过\_\_slots\_\_定义的属性和其父类中通过\_\_slots\_\_定义的属性。
- ▶ 例:\_\_slots\_\_使用示例。

#### \_\_slots\_\_



- 1 class Person: #定义Person类
- 2 \_\_slots\_\_ = ('name') #定义允许动态扩展的属性
- 3 class Student(Person): #以Person类作为父类定义子类Student类
- 4 \_\_slots\_\_ = ('sno') #定义允许动态扩展的属性
- 5 class Postgraduate(Student): #以Student类作为父类定义子类Postgraduate类
- 6 pass

#### \_slots\_



\_\_slots\_\_

12	pg=Postgraduate() #定义Postgraduate类对象pg
13	pg.sno='1810101' #为pg对象动态扩展属性sno
14	pg.name='张刚' #为pg对象动态扩展属性name
15	pg.tutor='马红' #为pg对象动态扩展属性tutor