**第23章 类的继承**

**第一部分**

对于面向对象的继承来说，其实就是将多个类共有的方法提取到父类中，子类仅需继承父类而不必一一实现每个方法。除了子类和父类的称谓，你可能看到过 派生类 和 基类 ，他们与子类和父类只是叫法不同而已。

**一、继承的概念**

一个类从另一个类中继承一些属性和方法，我们把这个类叫做子类(派生类)，另一个类叫做父类(基类)，其语法格式如下：

class DerivedClassName(BaseClassName1):

<statement-1> .

. .

<statement-N>

其中DerivedClassName就是子类，BaseClassName1是父类。

#encoding=utf-8

#定义一个类,类名是people

class people:

p\_address="北京昌平"

def getAddress(self):

return self.p\_address

#定义子类stu,使其继承父类people

class stu(people):

s\_name="czf"

def getName(self):

return self.s\_name

s1=stu()

print(s1.s\_name)

print(s1.getName())

#子类调用从父类继承下来的属性和方法

print(s1.p\_address)

print(s1.getAddress())

执行结果：

czf

czf

北京昌平

北京昌平

**二、子类不能继承父类的私有属性和私有方法**

#encoding=utf-8

#定义一个类,类名是stu

class people:

#父类公有的属性

p\_age=23

#父类公有的方法

def getAge(self):

return self.p\_age

#父类私有的属性

\_\_p\_address="北京昌平"

#父类私有的方法

def \_\_getAddress(self):

return self.\_\_p\_address

#定义子类stu,使其继承父类people

class stu(people):

s\_name="czf"

def getName(self):

return self.s\_name

s1=stu()

print(s1.s\_name)

print(s1.getName())

#子类调用从父类继承下来的公有属性和方法

print(s1.p\_age)

print(s1.getAge())

#子类不能继承父类私有属性和方法

#print(s1.\_\_p\_address)

#print(s1.\_\_getAddress())

**第二部分**

**三、方法重写**

如果父类方法的功能不能满足子类需求，可以在子类重写父类的方法。

#encoding=utf-8

#定义一个类,类名是stu

class people:

def show(self):

print("hello,everyone!")

class stu(people):

#子类重写父类的方法

def show(self):

print("hello,all student!")

p1=people()

#父类对象调用自己的方法

p1.show()

#子类对象调用重写后的方法

s1=stu()

s1.show()

执行结果：

hello,everyone!

hello,all student!

重写的几个必要条件：

1、存在继承关系

2、子类和父类的方法名相同

**四、多继承**

4.1 语法

Python同样有限的支持多继承形式。多继承的类定义形如下例:

class DerivedClassName(Base1, Base2, Base3):

<statement-1>

. . .

<statement-N>

需要注意圆括号中父类的顺序，若是父类中有相同的方法名，而在子类使用时未指定用哪一个父类的方法名，python从左至右搜索，即方法在子类中未找到时，从左到右查找父类中是否包含方法。

#encoding=utf-8

#定义一个类,类名是stu

class people:

p\_address="北京昌平"

def getAddress(self):

return self.p\_address

class stu:

s\_name="czf"

def getName(self):

return self.s\_name

#子类从多个父类中继承。每个父类用，隔开。

class pupil(people,stu):

s\_age=12

def getAge(self):

return self.s\_age

p1=pupil()

#子类pupil从父类people继承

print(p1.p\_address)

print(p1.getAddress())

#子类pupil从父类stu继承

print(p1.s\_name)

print(p1.getName())

#子类调用它独有的方法和属性

print(p1.s\_age)

print(p1.getAge())

除了直接使用类名作为基名，还可以用表达式，当基类定义在另一个模块中时这一点非常有用:

class DerivedClassName(modname.BaseClassName)。

4.2 继承的传递性

B类继承C类，A类继承B类，则A类继承C类。

#encoding=utf-8

#定义一个类,类名是people

class people:

p\_address="北京昌平"

def getAddress(self):

return self.p\_address

#定义一个类,类名是stu

class stu(people):

s\_name="czf"

def getName(self):

return self.s\_name

#B类继承C类，A类继承B类，则A类继承C类。

class pupil(stu):

s\_age=12

def getAge(self):

return self.s\_age

p1=pupil()

#子类pupil从父类people继承

print(p1.p\_address)

print(p1.getAddress())

#子类pupil从父类stu继承

print(p1.s\_name)

print(p1.getName())

#子类调用它独有的方法和属性

print(p1.s\_age)

print(p1.getAge())

**第三部分**

**五、继承静态方法和类方法**

从下面代码可以看出，如果子类继承父类的方法，子类覆盖了父类的静态方法，子类的实例继承了父类的static\_method静态方法，调用该方法，还是调用的父类的方法和类属性。

子类的实例继承了父类的class\_method类方法，调用该方法，调用的是子类的方法和子类的类属性。

# coding:utf-8  
class Foo(object):  
 X = 1  
 Y = 2  
 @staticmethod  
 def average(\*mixes):  
 return sum(mixes) / len(mixes)  
 @staticmethod  
 def static\_method():  
 return Foo.average(Foo.X, Foo.Y)  
 @classmethod  
 def class\_method(cls):  
 return cls.average(cls.X, cls.Y)  
class Son(Foo):  
 X = 3  
 Y = 5  
 @staticmethod  
 def average(\*mixes):  
 return sum(mixes) / 2  
p = Son()  
print(p.static\_method())  
print(p.class\_method())

1.5

4.0