# 类的封装

OOP编程（Object Oriented Programming，面向对象编程）是一种组织程序的方向，它是一种代码重用的思想。大部分流行的编程语言都支持OOP。面向对象编程的三个特征：封装，继承和多态。



### 定义一个类

#!/usr/bin/python3 #类定义

class People:

#定义基本属性

name = ''

age = 0

#定义私有属性,私有属性在类外部无法直接进行访问

\_\_weight = 0

#定义构造方法

def \_\_init\_\_(self,n,a,w):

self.name = n

self.age = a

self.\_\_weight = w

def speak(self):

print("%s 说: 我 %d 岁。" %(self.name,self.age))

# 实例化类

p = people('runoob',10,30)

p.speak()

**构造函数**

\_\_init\_\_是构造函数，创建对象时会自动调用该函数。

**类的公有属性**

不以下划线开头的变量是公有的，任何代码都可以访问。

print(p.name)

**类的私有属性**

**\_\_private\_attrs**：以两个下划线开头，声明该属性为私有，不能在类地外部被使用或直接访问。在类内部的方法中使用时 **self.\_\_private\_attrs**。

p = People('孙悟空',10,30)

print(p.name)

print(p.\_\_weight) # 报错

**类的属性也可以不写，但不直观。**

#name = ''

#age = 0

#\_\_weight = 0

**程序可以照常执行。**

**类的方法**

在类的内部，使用 def 关键字来定义一个方法，与一般函数定义不同，类方法必须包含参数 **self**，且为第一个参数，**self** 代表的是类的实例。

调用类的方法：

p.speak()

**类的私有方法**

**\_\_private\_method**：两个下划线开头，声明该方法为私有方法，只能在类的内部调用 ，不能在类地外部调用。**self.\_\_private\_methods**。这个和类的私有属性是一样的原理。

### 类的专有方法

**类的专有方法有：**

**\_\_init\_\_ :** 构造函数，在生成对象时调用

**\_\_del\_\_ :** 析构函数，释放对象时使用

**\_\_repr\_\_ :** 打印，转换

\_\_str\_\_: 输出对象信息

**\_\_setitem\_\_ :** 按照索引赋值

**\_\_getitem\_\_:** 按照索引获取值

**\_\_len\_\_:** 获得长度

**\_\_cmp\_\_:** 比较运算

**\_\_call\_\_:** 函数调用

**\_\_add\_\_:** 加运算

**\_\_sub\_\_:** 减运算

**\_\_mul\_\_:** 乘运算

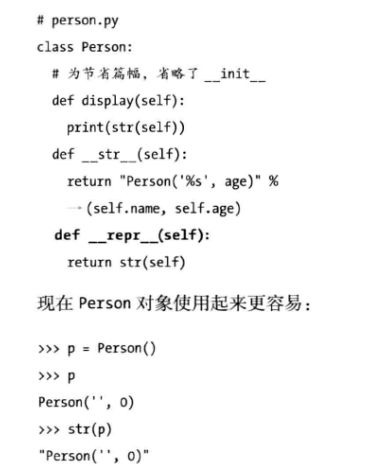
**\_\_div\_\_:** 除运算

**\_\_mod\_\_:** 求余运算

**\_\_pow\_\_:** 乘方

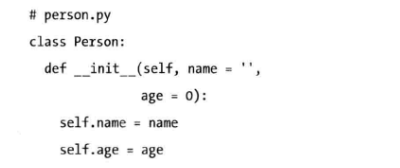
**覆写专用方法：**

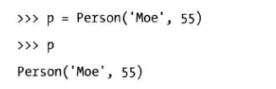
**\_\_repr\_\_函数和\_\_str\_\_函数例子**



**直接输入对象名，会调用\_\_repr\_\_函数。**

### 构造函数有默认参数

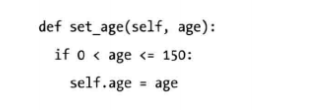




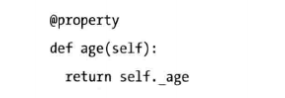
这样，就可以有多种创建对象的形式。

### 设置对象属性

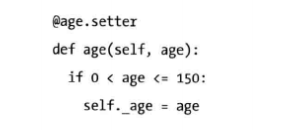
（1）、通过成员函数设置属性

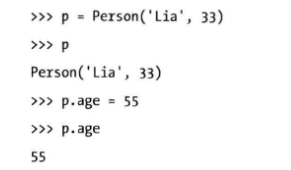


（2）、通过@property**获取**对象的属性(不能设置)



（3）、通过@setter**设置**对象的属性





**实例**：游戏人生程序

1、创建三个游戏人物，分别是：

* 武媚娘，女，18，初始战斗力1000
* 孙悟空，男，20，初始战斗力1800
* 朱元元，女，19，初始战斗力2500

2、游戏场景，分别：

* 草丛战斗，消耗200战斗力
* 自我修炼，增长100战斗力
* 多人游戏，消耗500战斗力

# -\*- coding:utf-8 -\*-

# -\*- coding:utf-8 -\*-

# #####################  定义实现功能的类  #####################

class Person:

def \_\_init\_\_(self,na, gen, age, fig):

self.name = na

self.gender = gen

self.age = age

self.fight = fig

def grassland(self):

"""注释：草丛战斗，消耗200战斗力"""

self.fight = self.fight - 200

def practice(self):

"""注释：自我修炼，增长100战斗力"""

self.fight = self.fight + 200

def incest(self):

"""注释：多人游戏，消耗500战斗力"""

self.fight = self.fight - 500

def detail(self):

"""注释：当前对象的详细情况"""

temp = "姓名:%s ; 性别:%s ; 年龄:%s ; 战斗力:%s" % (self.name, self.gender, self.age, self.fight)

print(temp)

# #####################  开始游戏  #####################

cang = Person('武媚娘','女',18,1000) # 创建武媚娘角色

dong = Person('孙悟空','男',20,1800) # 创建孙悟空角色

bo = Person('朱元元','女',19,2500) # 创建朱元元角色

cang.incest() #武媚娘参加一次多人游戏

dong.practice() #孙悟空自我修炼了一次

bo.grassland() #朱元元参加一次草丛战斗

#输出当前所有人的详细情况

cang.detail()

dong.detail()

bo.detail()

cang.incest() #武媚娘又参加一次多人游戏

dong.incest() #孙悟空也参加了一个多人游戏

bo.practice() #朱元元自我修炼了一次

#输出当前所有人的详细情况

cang.detail()

dong.detail()

bo.detail()