[**Python**](http://c.biancheng.net/python/)**类实例方法**

通常情况下，在类中定义的方法默认都是实例方法。前面章节中，我们已经定义了不只一个实例方法。不仅如此，类的构造方法理论上也属于实例方法，只不过它比较特殊。  
  
比如，下面的类中就用到了实例方法：

1. **class** CLanguage:
2. #类构造方法，也属于实例方法
3. **def** \_\_init\_\_(self):
4. self.name = "C语言中文网"
5. self.add = "http://c.biancheng.net"
6. # 下面定义了一个say实例方法
7. **def** say(self):
8. **print**("正在调用 say() 实例方法")

实例方法最大的特点就是，它最少也要包含一个 self 参数，用于绑定调用此方法的实例对象（Python 会自动完成绑定）。实例方法通常会用类对象直接调用，例如：

1. clang = CLanguage()
2. clang.say()

运行结果：

正在调用 say() 实例方法

当然，Python 也支持使用类名调用实例方法，但此方式需要手动给 self 参数传值。例如：

1. #类名调用实例方法，需手动给 self 参数传值
2. clang = CLanguage()
3. CLanguage.say(clang)

运行结果为：

正在调用 say() 实例方法

有关使用类名直接调用实例方法的更多介绍，可阅读《[Python类调用实例方法](http://c.biancheng.net/view/2267.html)》一节。

**Python类方法**

Python 类方法和实例方法相似，它最少也要包含一个参数，只不过类方法中通常将其命名为 cls，Python 会自动将类本身绑定给 cls 参数（注意，绑定的不是类对象）。也就是说，我们在调用类方法时，无需显式为 cls 参数传参。

和 self 一样，cls 参数的命名也不是规定的（可以随意命名），只是 Python 程序员约定俗称的习惯而已。

和实例方法最大的不同在于，类方法需要使用＠classmethod修饰符进行修饰，例如：

1. **class** CLanguage:
2. #类构造方法，也属于实例方法
3. **def** \_\_init\_\_(self):
4. self.name = "C语言中文网"
5. self.add = "http://c.biancheng.net"
6. #下面定义了一个类方法
7. @classmethod
8. **def** info(cls):
9. **print**("正在调用类方法",cls)

注意，如果没有 ＠classmethod，则 Python 解释器会将 fly() 方法认定为实例方法，而不是类方法。

类方法推荐使用类名直接调用，当然也可以使用实例对象来调用（不推荐）。例如，在上面 CLanguage 类的基础上，在该类外部添加如下代码：

1. #使用类名直接调用类方法
2. CLanguage.info()
3. #使用类对象调用类方法
4. clang = CLanguage()
5. clang.info()

运行结果为：

正在调用类方法 <class '\_\_main\_\_.CLanguage'>  
正在调用类方法 <class '\_\_main\_\_.CLanguage'>

**Python类静态方法**

静态方法，其实就是我们学过的函数，和函数唯一的区别是，静态方法定义在类这个空间（类命名空间）中，而函数则定义在程序所在的空间（全局命名空间）中。  
  
静态方法没有类似 self、cls 这样的特殊参数，因此 Python 解释器不会对它包含的参数做任何类或对象的绑定。也正因为如此，类的静态方法中无法调用任何类属性和类方法。  
  
静态方法需要使用＠staticmethod修饰，例如：

1. **class** CLanguage:
2. @staticmethod
3. **def** info(name,add):
4. **print**(name,add)

静态方法的调用，既可以使用类名，也可以使用类对象，例如：

1. #使用类名直接调用静态方法
2. CLanguage.info("C语言中文网","http://c.biancheng.net")
3. #使用类对象调用静态方法
4. clang = CLanguage()
5. clang.info("Python教程","http://c.biancheng.net/python")

运行结果为：

C语言中文网 http://c.biancheng.net  
Python教程 http://c.biancheng.net/python

在实际编程中，几乎不会用到类方法和静态方法，因为我们完全可以使用函数代替它们实现想要的功能，但在一些特殊的场景中（例如工厂模式中），使用类方法和静态方法也是很不错的选择。