**Python新式类与经典类(旧式类)的区别**

**Python中类分两种：旧式类和新式类：**

➤新式类都从object继承，经典类不需要。

➤新式类的MRO(method resolution order 基类搜索顺序)算法采用C3算法广度优先搜索，而旧式类的MRO算法是采用深度优先搜索

➤新式类相同父类只执行一次构造函数，经典类重复执行多次。

**其中：**

➤截止到python2.1，只存在旧式类。旧式类中，类名和type是无关的：如果x是一个旧式类，那么x.\_\_class\_\_定义了x的类名，但是type(x)总是返回<type 'instance'>。这反映了所有的旧式类的实例是通过一个单一的叫做instance的内建类型来实现的，这是它和类不同的地方。

➤新式类是在python2.2为了统一类和实例引入的。一个新式类只能由用户自定义。如果x是一个新式类的实例，那么type(x)和x.\_\_class\_\_是一样的结果（尽管这不能得到保证，因为新式类的实例的\_\_class\_\_方法是允许被用户覆盖的）。

➤Python 2.x中默认都是经典类，只有显式继承了object才是新式类

➤Python 3.x中默认都是新式类,经典类被移除，不必显式的继承object

**Python2.x中：**

1. **class** **A**:
2. **pass**
3. **class** **B**:
4. **pass**
5. **class** **C**(B):
6. **pass**
7. **class** **D**(C,A):
8. **pass**

执行顺序为：D->C->B,->A

1. **class** **A**(object):
2. **pass**
3. **class** **B**(object):
4. **pass**
5. **class** **C**(object):
6. **pass**
7. **class** **D**(A,B,C):
8. **pass**

执行顺序为： D->A->B->C->Object