__new__和__init__方法

```
class A(object):
    def __init__(self):
        print("这是 init 方法")

    def __new__(cls):
        print("这是 new 方法")
        return object.__new__(cls)

A()
```

总结

- `__new__` 至少要有一个参数cls，代表要实例化的类，此参数在实例化时由Python解释器自动提供
- `__new__` 必须要有返回值，返回实例化出来的实例，这点在自己实现 `__new__` 时要特别注意，可以return父类 `__new__` 出来的实例，或者直接是object的 `__new__` 出来的实例
- `__init__` 有一个参数self，就是这个 `__new__` 返回的实例， `__init__` 在 `__new__` 的基础上可以完成一些其它初始化的动作， `__init__` 不需要返回值

单例设计模式

举个常见的单例模式例子，我们日常使用的电脑上都有一个回收站，在整个操作系统中，回收站只能有一个实例，整个系统都使用这个唯一的实例，而且回收站自行提供自己的实例。因此回收站是单例模式的应用。

确保某一个类只有一个实例，而且自行实例化并向整个系统提供这个实例，这个类称为单例类，单例模式是一种对象创建型模式。

```python
# 实例化一个单例
class Singleton(object):
    __instance = None
    __is_first = True

    def __new__(cls, age, name):
        if not cls.__instance:
            cls.__instance = object.__new__(cls)
        return cls.__instance

    def __init__(self, age, name):
        if self. __is_first: # 不会再创建第二个对象
            self.age = age
            self.name = name
            Singleton. __is_first = False


a = Singleton(18, "张三")
b = Singleton(28, "张三")

print(id(a))
print(id(b))

print(a.age) # 18
print(b.age) # 18

a.age = 19
print(b.age)
```