# [自动化测试po模式](https://www.cnblogs.com/csmashang/p/12878496.html)

PO模式最核心的思想是分层，实现松耦合！实现脚本重复使用，实现脚本易维护性！

**主要分三层：**

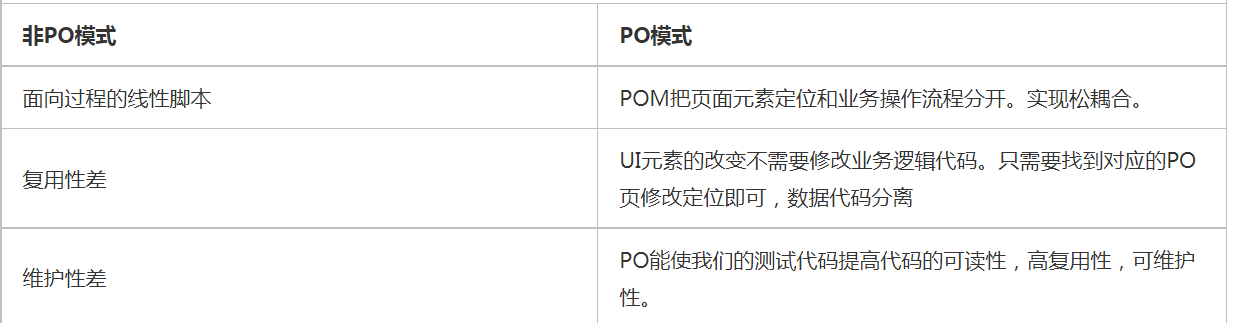
**1.基础层BasePage：封装一些最基础的selenium的原生的api方法，元素定位，框架跳转等。**

**2.PO层：元素定位、获得元素对象，页面动作**

**3.测试用例层：业务逻辑，数据驱动！**

**三者的关系：PO层继承继承层，测试用例层调用PO层！**

# **三、非PO模式和PO模式优缺点对比**



**非PO模式举个栗子：有如下百度搜索脚本：**

import unittestfrom selenium import webdriverfrom selenium.webdriver.common.by import By

class Test(unittest.TestCase):

def test01(self):

# 打开浏览器

driver = webdriver.Chrome()

# 加载百度首页

driver.get('http://www.baidu.com')

# 在百度搜索栏中输入软件测试

driver.find\_element(By.ID, 'kw').send\_keys('软件测试')

# 点击百度一下按钮

driver.find\_element(By.ID, 'su').click()

def test02(self):

# 打开浏览器

driver = webdriver.Chrome()

# 加载百度首页

driver.get('http://www.baidu.com')

# 在百度搜索栏中输入软件测试

driver.find\_element(By.ID, 'kw').send\_keys('硬件测试')

# 点击百度一下按钮

driver.find\_element(By.ID, 'su').click()

**如何把上述栗子改成PO模式呢？**

1、基础层BasePage

from selenium import webdriver

class BasePage:

#构造方法

def \_\_init\_\_(self):

# 打开浏览器

self.driver = webdriver.Chrome() # Alt+Enter

# 加载百度首页

self.driver.get('http://www.baidu.com')

#封装定位元素

def find\_ele(self,\*args):

ele = self.driver.find\_element(\*args)

return ele

2、PO层：封装百度页面元素定位，元素对象以及页面操作

from selenium.webdriver.common.by import Byfrom base.base\_page import BasePage

class BaiduPage(BasePage):

#元素定位，

baidu\_text\_loc = (By.ID, 'kw')

baidu\_submit\_loc = (By.ID, 'su')

#获得元素对象，

def get\_text\_obj(self):

ele = self.find\_ele(\*BaiduPage.baidu\_text\_loc)

return ele

def get\_submit\_obj(self):

ele = self.find\_ele(\*BaiduPage.baidu\_submit\_loc)

return ele

#页面操作

def search(self,search\_string):

self.get\_text\_obj().send\_keys(search\_string)

self.get\_submit\_obj().click()

3、测试用例层：业务逻辑和数据驱动

from ddt import ddt, datafrom po.baidu\_page import BaiduPage

@ddtclass BaiduTest(unittest.TestCase):

@data('软件测试','硬件测试')

def test01(self,seaString):

BaiduPage().search(seaString)

time.sleep(5)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()