测试过程中，我们经过发现脚本执行的时候展现出来的效果都是很快结束了，为了观察执行效果我们会增加一个等待时间来观察一下执行效果。这种等待时间我们只是为了我们便于观察，这种情况下是否包含等待时间不会影响我们的执行结果，但是有一种情况会直接影响我们的执行结果。在我们打开一个网站的时候需要由于环境的因素导致页面没有下载完成时，去定位元素此时无法找到元素，这个时候会影响到我们，这个时候我们增加一个等待时间就会显得万分重要。

selenium 主要提供Explicit Waits和Implicit Waits两种模式的等待，但是python time 模块也提供了一种非智能的sleep()等待，这个设置以后必须强制等待设置的时间，只有等待时间结束以后才会继续执行，这种模式我一般会用到观察执行的效果时候，而Explicit Waits和Implicit Waits 这两种我会在实际自动化测试中使用。

下面我们了解一下这种三种等待模式

1.强制等待

　　强制等待属于python time模块的一种等待，使用时需要引入time模块才可以使用，下面我们看一断代码

[复制代码](javascript:void(0);)

#-\*- coding:utf-8 -\*-

import time

from datetime import datetime

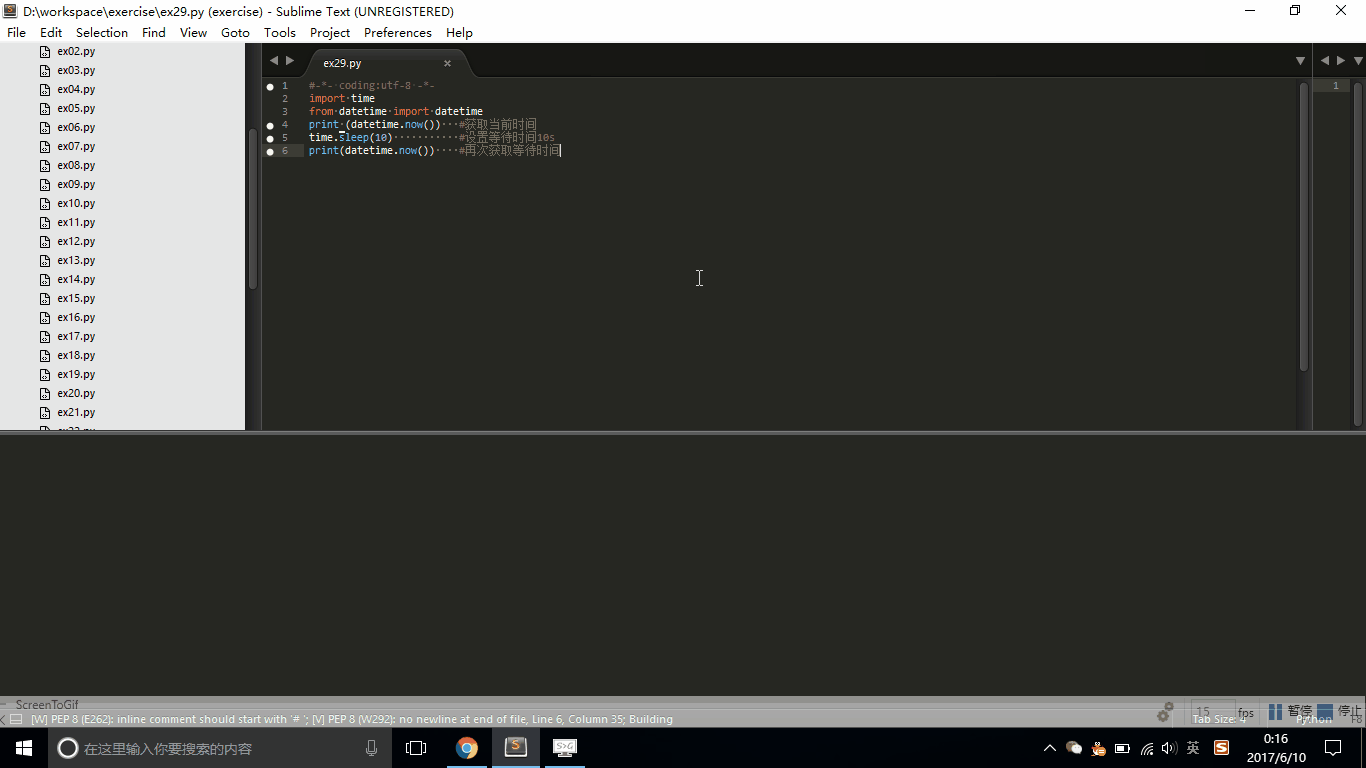
print (datetime.now()) #获取当前时间

time.sleep(10) #设置等待时间10s

print(datetime.now()) #再次获取当前时间

[复制代码](javascript:void(0);)

代码很简单，主要是获取当前时候后设置了一个等待时间，然后在等待时间后面设置一个事物，这个事物来查看等待时间结束后的当前时间。通过执行结果我们很好的看到只有执行完毕等待以后才会再次执行下一步骤。



 2.隐性等待 implicitly\_wait()

　　什么是隐性的等待呢？所谓的隐性的等待就是我们设置了一个等待时间范围，这个等待的时间是不固定的，最长的等待也莫过于我们设置的最大值。用高考来举个例子，高考的语文是6月7日上午9:00 - 11:30，我们只能在规定的时间内答题交卷，不能超过11:30 交卷，如果我们提前答完卷，那么我们可以提前交卷，那么这里规定的2.5小时是我们设置的最大等待时间，而我们可以再任何时间答完试卷均去交卷，但是我们不能超过11:30交卷，也就是说明最大的答卷时间就是2.5h（正好也是因为此时刚刚高考结束，所以我这边也趁下热度）。

下来我们看段代码

[复制代码](javascript:void(0);)

from selenium import webdriver

driver = webdriver.Chrome()

driver.implicitly\_wait(10) #这里设置智能等待10s

driver.get('https://www.google.com.hk')

print (driver.title)

driver.quit()

[复制代码](javascript:void(0);)

　　这里主要采用 implicitly\_wait(10)方法来设置等待时间，这里要比time.sleep(10)更加智能一些。

　　说明：首先Implicit Waits默认是等待时间是0，同时隐性等待是对driver起作用，所以只要设置一次即可，没有必要到处设置

3.显示的等待 WebDriverWait()

　WebDriverWait()会配合until()和until\_not()方法一起使用，根据判断条件而进行灵活进行处理时间等待问题，他会不断的根据你设定的条件去判断，直到超过你设置的等待时间，如果设置的条件满足，然后进行下一步操作，如果没有满足会报一个'selenium.common.exceptions.TimeoutException: Message: '错误，使用WebDriverWait首先需要导入from selenium.webdriver.support.ui import WebDriverWait模块，下面我们看一下实际的代码

[复制代码](javascript:void(0);)

#-\*- coding:utf-8 -\*-

from datetime import datetime

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

driver = webdriver.Chrome()

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

driver.maximize\_window()

try:

#为了更好的对比效果，首先我们设置了一个存在的元素，然后在去找一个不存在的元素，同样设置了10s的等待时间

#kw元素存在时

print(datetime.now()) #

element = WebDriverWait(driver,10).until( #until 也属于WebDriverWait,代表一直等待,直到某元素可见，until\_not与其相反，判断某个元素直到不存在

EC.presence\_of\_element\_located((By.ID, "kw")) #presence\_of\_element\_located主要判断页面元素kw在页面中存在。

)

#kw111元素不存在时

print(datetime.now())

element = WebDriverWait(driver,10).until(

EC.presence\_of\_element\_located((By.ID, "kw111"))

)

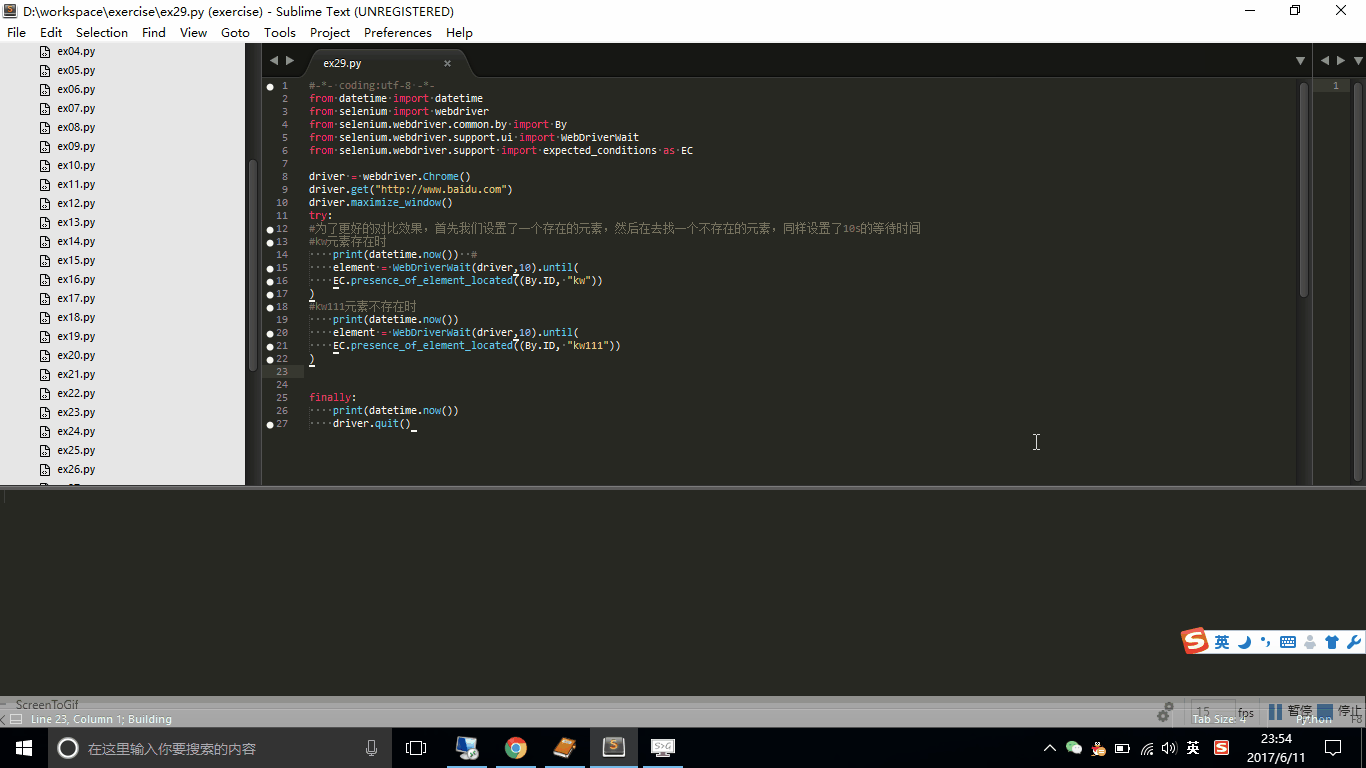
finally:

print(datetime.now())

driver.quit()

[复制代码](javascript:void(0);)

 看下刚才代码的效果



通过运行的结果我们可以看到,id为kw和kw111两个元素，均设置了10s等待时间，当id为kw的元素找到的时候很快就去执行下一步，,找id 为'kw111'元素时，发现无法找到一直等待下去,期间会不断的去巡查，直到超过我们设置的连接超时的时间报一个TimeoutException错误。