一：selenium3 安装

1. pip install selenium (比较慢)

2. 压缩包安装

    1. https://pypi.org/project/selenium/ （较快）
    2. Download files => selenium-*.tar.gz
    3. 解压，进入文件夹。python setup.py install
    4. pip list 验证是否安装

二：验证使用

1. https://npm.taobao.org/mirrors/chromedriver/84.0.4147.30/ 安装 Chrome 驱动

2. 注意 Chrome 版本要对应，否则不会生效，如何查看 Chrome 版本：chrome://version/

3. 解压 => cd ~/DeskTop && cp ./chromedriver /usr/local/bin && cd /usr/local/bin && ls | grep chrome

三：自动化测试的优缺点

优点：1. 节省人力，避免做重复性的工作

2. 覆盖范围广，稳定性较高

缺点：1. 不适合需求变更较快的场景

2. 不适合周期短的项目

四：selenium 定位元素

1. find_element_by_id() => 定位 id
2. find_element_by_name() => tag name

3. find_element_by_class_name() => class

4. find_element_by_css_selector => css 选择器

五：selenium 常用 Api

1. clear() 清空输入

2. send_keys() 输入

3. back() 返回

4. click() 点击

5. maximize_window() 最大化窗口

6. getAttribute() 获取元素的属性

7. getCssValue() 获取 css 属性

8. getText() 获取文本

六：selenium 模拟用户操作 (比如鼠标悬停 hover 等等)

1. 引入 selenium 中用户操作模块

引用：from selenium.webdriver.common.action_chains import ActionChains

使用：ActionChains(driver).click(ele).perform() // perform 是指执行所有链式操作

2. 常见鼠标操作：

1. click(on_element=None) 单击鼠标左键

2. context_click(on_element=None) 点击鼠标左键

3. double_click(on_element=None) 双击左键

4. move_to_element(to_element) 鼠标移动到某个元素

3. 常见键盘操作 直接上网搜 selenium send_keys,就有各种详细的介绍

from selenium.webdriver.common.keys import Keys

1. 单击键盘

driver.find_element_by_id("kw").send_keys(Keys.SPACE)

2. 组合键

```
driver.find_element_by_id('kw').send_keys(Keys.CONTROL,'v')
driver.find_element_by_id("kw").send_keys(Keys.CONTROL,'a')
driver.find_element_by_id("kw").send_keys(Keys.CONTROL,'x')
```

七： unittest 单元测试框架

1. 用 import 语句引入 unittest 模块

2. 测试的类都继承于 TestCase 类

3. setUp() 测试前的初始化工作；tearDown()测试后的清除工作（每个测试用例都会执行）

注意：1. 所有类中的方法的入参为 self，定义方法的变量也要”self.变量“

2. 定义测试用例，以 test 开头命名的方法，方法的入参为 self

3. unittest.main()方法会搜索该模块下所有以 test 开头的测试用例方法，并自动指定它们

4. 注意文件不能用 unittest.py 命名，否则找不到 TestCase

成功输出. 失败输出 F

4. 常用断言 api

self.assertEqual('xxx','xxxx',msg='xxxx') // 判断是否相等,msg 是不满足条件的提示语

self.assertTrue('xxxx',msg='xxxx) // 判断输入的值是否为 true

self.assertFalse('xxxx',msg='xxxxx') // 判断输入的值是否为 false

八： 测试套件 TestSuite，用以控制测试用例的执行顺序

1. unittest.TestSuite() // 生成 suite 实例

2. suite.addTest() // 向其中添加测试用例

3. unittest.TextTestRunner(verbosity=1) // verbosity 参数控制报告的详细程度： 0 简单报告 1 一般报告 2 详细报告

4. runner.run(suite) // 执行测试用例

九：测试报告的生成，使用 HTMLTestRunner

当前的测试报告可读性非常差，需要有一个可视化的工具，方便我们阅读和分析测试结果。所以我们就需要这个工具.

0. 下载地址 http://tungwaiyip.info/software/HTMLTestRunner.html

1. python => import sys => sys.path => 确定 python 的安装目录，数组的第二个

2. 将 HTMLTestRunner.py 移动到这个安装目录下

3. import 验证