## 为什么用selenium?

* 免费开源
* 支持多语言（java,python,c#,ruby等）
* 支持多平台（Windows，linux, MAC）
* 支持多浏览器（ie,ff,safari,opera,chrome）
* 分布式，可以把测试用例分布到不同的测试机器的执行，相当于分发机的功能。
* 技术支持（成熟的社区，大量的文档支持）



## 功能

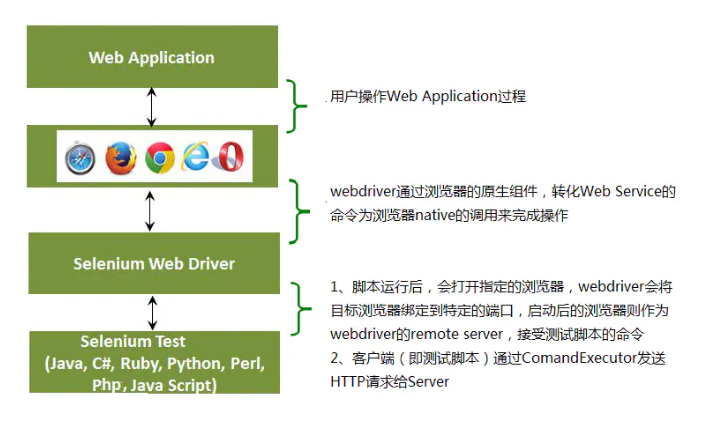
* 框架底层使用JavaScript模拟真实用户对浏览器进行操作。测试脚本执行时，浏览器自动按照脚本代码做出点击，输入，打开，验证等操作，就像真实用户所做的一样，从终端用户的角度测试应用程序。
* 使浏览器兼容性测试自动化成为可能，尽管在不同的浏览器上依然有细微的差别。
* 使用简单，可使用Java，[Python](https://baike.baidu.com/item/Python/407313" \t "_blank)等多种语言编写用例脚本。

## 环境搭建

* Python ：官网下载地址<https://www.python.org/downloads/>
* python编辑器pycharm
* selenium包：pip install Selenium
* 浏览器driver：下载对应的浏览器驱动
* 配置环境变量 将浏览器驱动的路径添加到系统变量Path中。

## 工作原理

1. 脚本启动driver
2. Driver驱动浏览器作为远程服务器
3. 执行脚本发送请求
4. 服务器解析请求作出响应，并把响应返回给客户端（脚本）
   1. selenium client(python等语言编写的自动化测试脚本)初始化一个service服务，通过Webdriver启动浏览器驱动程序chromedriver.exe
   2. 通过RemoteWebDriver向浏览器驱动程序发送HTTP请求，浏览器驱动程序解析请求，打开浏览器，并获得sessionid，如果再次对浏览器操作需携带此id
   3. 打开浏览器，绑定特定的端口，把启动后的浏览器作为webdriver的remote server
   4. 打开浏览器后，所有的selenium的操作(访问地址，查找元素等)均通过RemoteConnection链接到remote server，然后使用execute方法调用\_request方法通过urlib3向remote server发送请求
   5. 浏览器通过请求的内容执行对应动作
   6. 浏览器再把执行的动作结果通过浏览器驱动程序返回给测试脚本

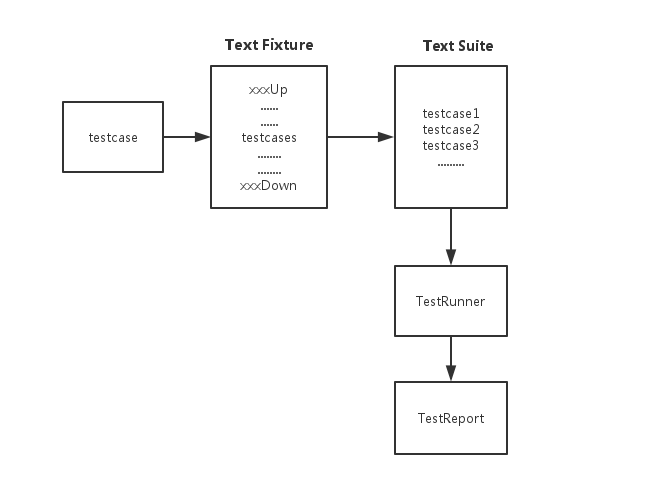


## selenium定位元素的方式



## selenium常用事件

## Unittest



### unittest.TestCase(用例)

**setUpClass()**:必须使用@classmethod 装饰器，所有case运行之前只运行一次  
**tearDownClass()**:必须使用@classmethod装饰器,所有case运行完之后只运行一次

**setUp()**:用于测试用例执行前的初始化工作，如实例化浏览器。每个以test开头的case运行前都会执行

**tearDown()**:用于测试用例执行之后的善后工作，如关闭浏览器。每个以test开头的case运行后都会执行

**assert\*():**断言，在执行测试用例的过程中，最终用例是否执行通过，是通过判断测试得到的实际结果和预期结果是否相等决定的。

### unittest.TestSuite（用例集）

* 一般通过addTest()或者addTests()向suite中添加。
* case的执行顺序与添加到Suite中的顺序是一致的

### unittest.TestLoader（把用例添加到用例集）

* TestLoadder用来加载TestCase到TestSuite中。
* loadTestsFrom\*()方法从各个地方寻找testcase，创建实例。

### unittest.TextTestRunner（执行用例集）



其他：

用例数据存储,读取

Unittest.ddt

HtmltestRunner报告

PageObjet模式

Pytest

参考链接：

<https://blog.csdn.net/weixin_36279318/article/details/79475388>

<https://www.jianshu.com/p/e7cf427468c8>

<https://blog.csdn.net/weixin_42139375/article/details/83216648>

## pytest

安装：pip install pytest

用例编写要求：

* 测试文件以test\_开头（以\_test结尾也可以）
* 测试类以Test开头，并且不能带有 init 方法
* 测试函数以test\_开头
* 断言使用基本的assert即可

Unittest & pytest 对比

* 测试用例编写规范不同

Unittest 测试用例类需要继承unittest, 测试用例需要以test开头

Pytest 测试用例类需要以Test开头，测试用例需要以test\_开头，测试文件以test\_开头或以\_test结尾

* 断言：

Unittest用assertFalse, assertTrue; pytest直接用assert

* 生成报告

Unittest生成报告用HTMLTestRunner或python库 HTMLReport

Pytest 生成报告用pytest-html或allure

* 第三方插件：

Pytest :

pytest-xdist(多进程运行case) e.g pytest -n 5 #5为启动的进程数

pytest-html 生成报告 e.g pytest --html=report.html

pytest-retunfailures 失败重试 e.g pytest -reruns 1 # 1为失败重试次数

### 运行

pytest -x test\_se.py 遇到错误时停止测试

pytest --maxfail=1 当用例个数达到指定数量时，停止测试