第六章节 自动化测试概念篇（C++方向）

【本节目标】

- 自动化测试
- Web自动化测试
- selenium

1. 自动化

1.1 自动化概念

自动的代替人的行为完成操作。

自动化在生活中处处可见，

自动洒水机，主要通上水就可以自动化洒水并且可以自动的旋转。

自动洗手液，免去了手动挤压可以自动感应出洗手液

超市自动闸门，不需要手动的开门关门

生活中的自动化案例有效的减少了人力的消耗，同时也提高了生活的质量。

软件中的自动化测试也是如此，通过自动化测试有效减少人力的消耗的同时也提高了测试的质量和效率。

> 自动化的主要目的就是用来进行回归测试。什么是回归测试?

1.1.1 回归测试

软件有多个版本需要进行功能的整体回归。

为了避免新增功能影响到历史的功能需要进行功能的回归。

✅ 常见面试题

1.自动化测试能够取代人工测试吗?

> 自动化测试不一定比人工测试更能保障系统的可靠性，自动化测试是测试人员手工编写，后续如果有功能的变更自动化也需要进行不定期的维护和更新。

2.自动化测试可以大幅度降低工作量?

> 错误
>
> 坑！！“一定程度上”和“大幅度”的表达方式也需要注意

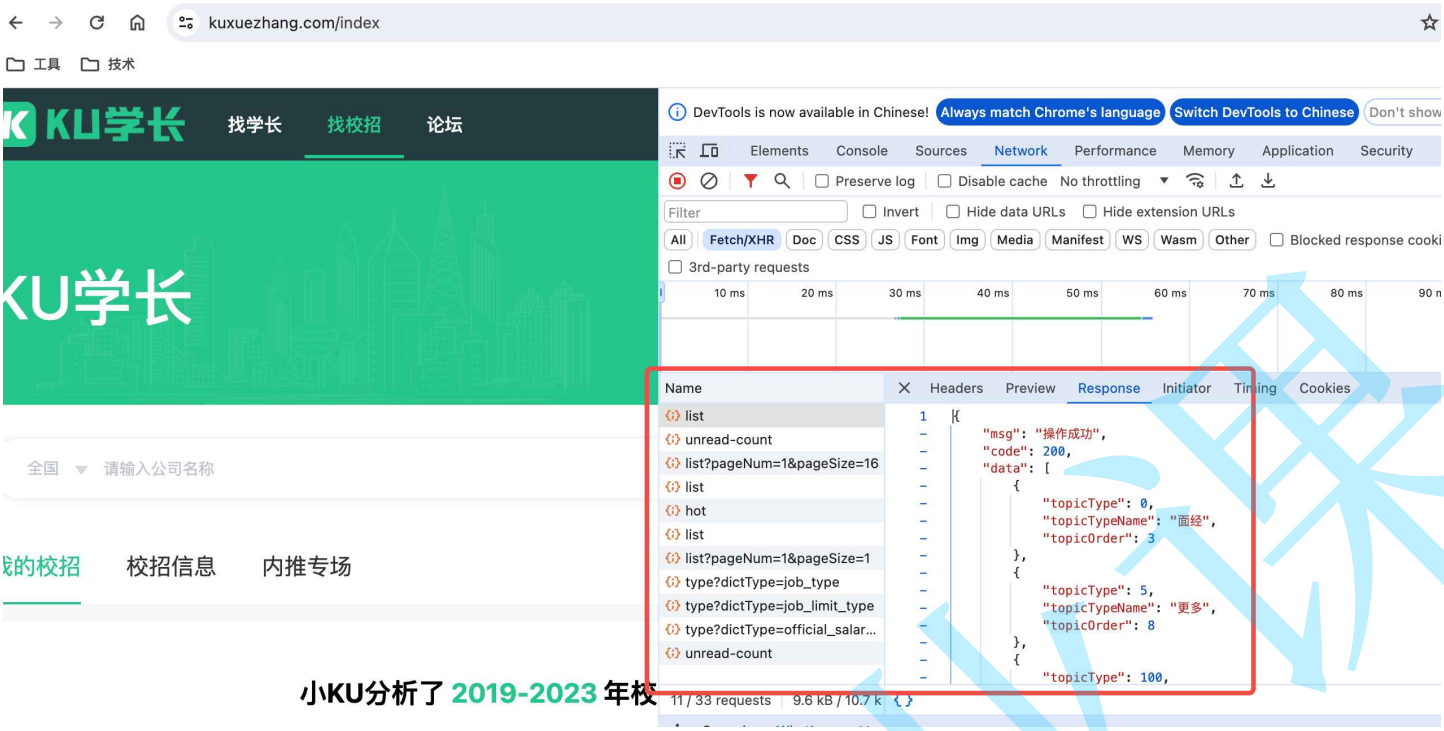
注意：测试笔试中的选择题尽量不要选择说的太死或者太绝对的选项~~~~

1.2 自动化分类

> - 讲自动化分类的目的是避免同学们将自动化混淆，同时避免讲不同的自动化如何实施造成的课时压力
> - 很多同学经常听到自动化这个词，但是很容易误以为自动化就是一个东西，自动化是个统称，同学们可以理解为我们常说的吃瓜这样的表达方式和自动化是一样的，吃瓜可以是吃西瓜，吃哈密瓜，吃香瓜，自动化也包含多种，如接口自动化，web自动化，移动端自动化等等...
> - 主要介绍各测试分类为什么需要实施自动化，目的和意义

1.2.1 接口自动化

> 什么是接口自动化测试?
>
> 为什么要做？解决了什么问题，目的是什么
>
> 测试接口



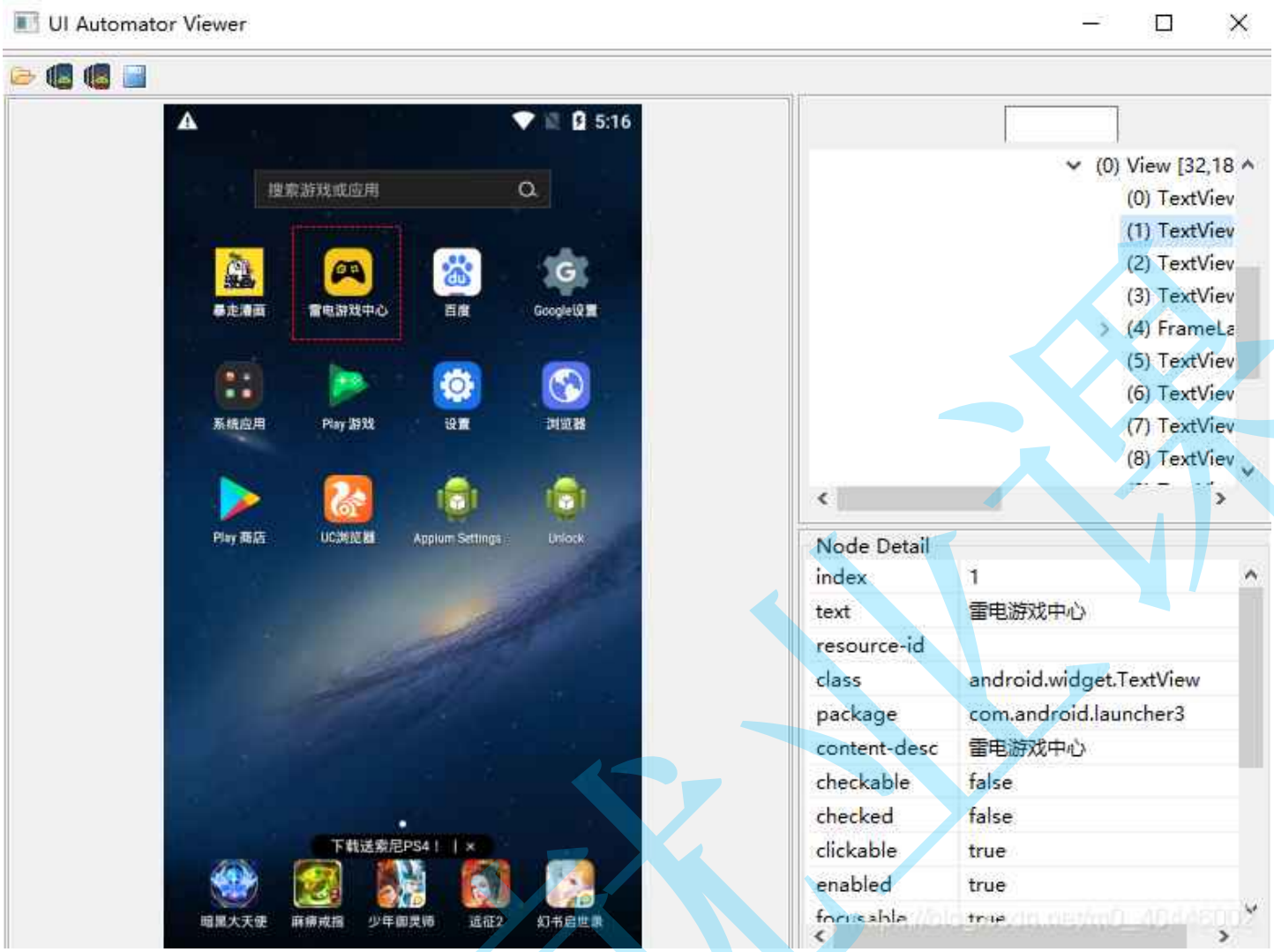
1.2.2 UI自动化

UI测试也称为界面测试，常见的UI自动化测试包含web自动化测试、移动端自动化测试等等。

- 移动端自动化测试

> 什么是移动端自动化测试?
>
> 为什么要做？解决了什么问题，目的是什么
>
> 测试移动端界面表现

移动端，顾名思义，就是部署在手机上的应用程序。移动端自动化测试主要测试程序部署在手机上能否按照预期的结果的进行操作。



移动端测试通常不是将程序部署在移动端上进行测试，而是需要安装模拟器，在电脑上编写自动化测试脚本代码对模拟器上的软件进行操作

移动端测试难度相对较大，移动端测试的稳定性要比接口自动化测试和web自动化测试的稳定性要差很多，原因主要是移动端测试收到的环境影响比较多

- web自动化测试

> 什么是web自动化测试?
>
> 为什么要做？解决了什么问题，目的是什么
>
> 测试web界面表现

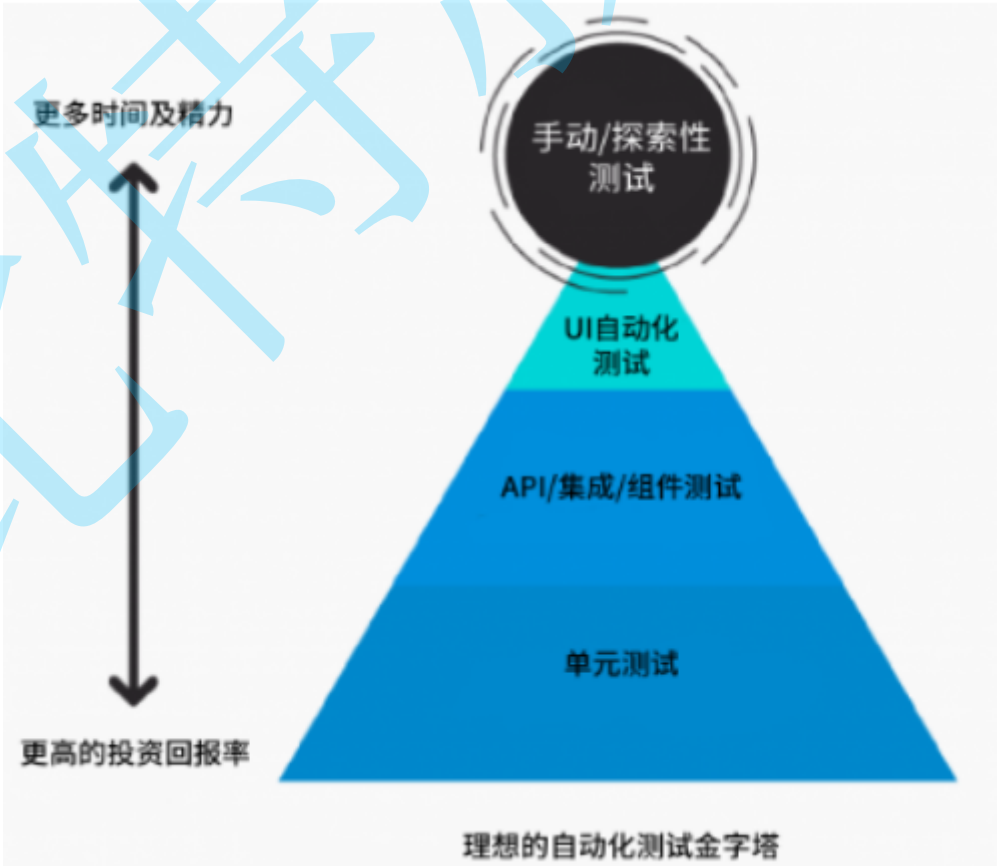
如何进行百度搜索?

通常来说我们手动进行百度搜索的步骤如下，web自动化能够代替我们自动的执行。

模拟人在浏览器上的操作行为，自动的打开浏览器，访问百度首页，并进行一系列的搜索和验证等行为。

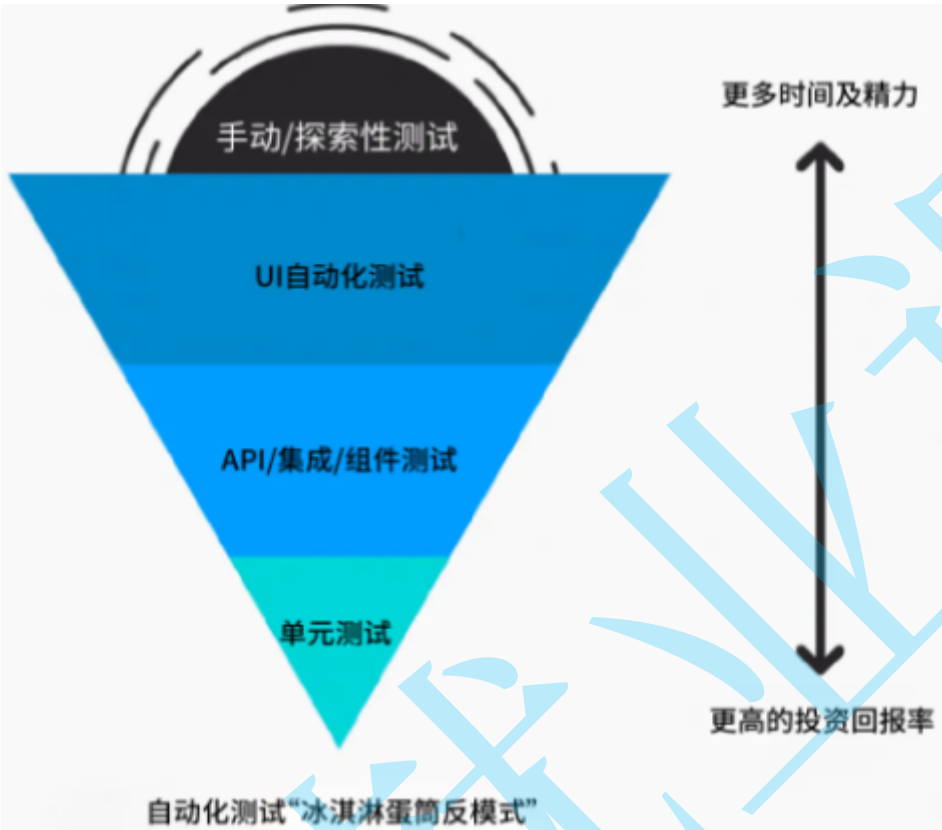
1.3 自动化测试金字塔

> 自动化的类型非常多，那么到底哪一种是最好的？哪一种测试的收益会更佳？
>
> 这里我们介绍测试圈内非常著名的自动化测试金字塔



理想的自动化测试金字塔表达了自动化测试的理想情况，利用较少的时间和精力在单元测试上就能够发现更多有效的问题。

然后实际上在企业中，自动化往往是“冰淇淋蛋筒反模式”



自动化需要大量的初始投资，找到“突破点”，与手动测试相比，我们开始看到它对长期成本产生的积极影响，也能够清楚，这两种测试活动都是完全兼容，产生短期和长期利益。

2. web自动化测试

> 学到这里，希望同学们对自动化测试能够有一定的了解。然后实际在企业中，我们需要对某一个特定的软件进行自动化的实施，包含但不限于软件所有界面的UI自动化测试、所有接口的接口自动化测试。
>
> 接下来我们将正式步入web自动化测试的学习，了解企业中是如何使用工具来统一编写和管理自动化用例。

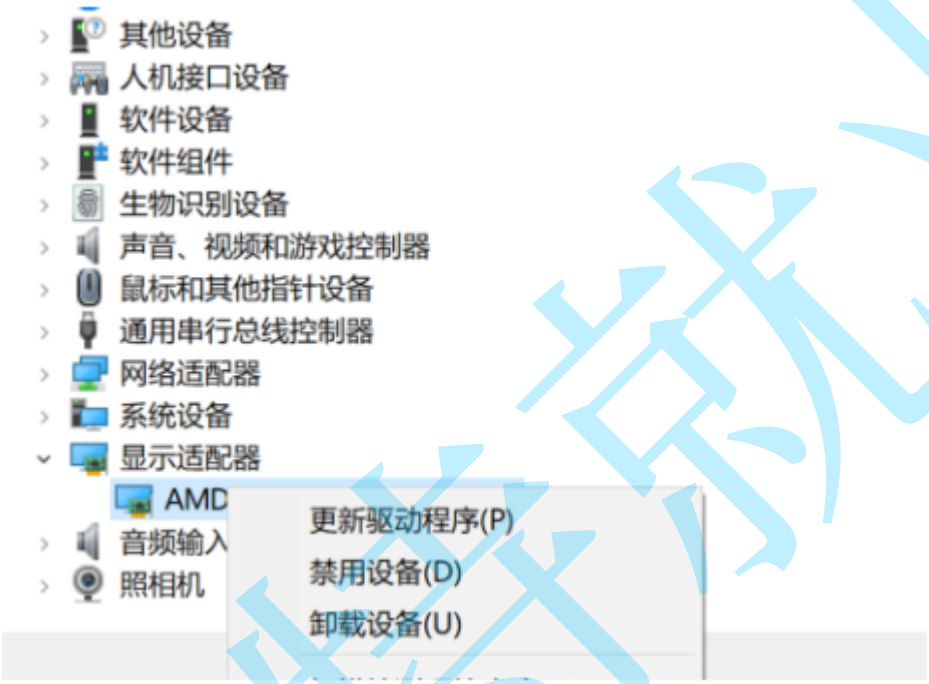
2.1 驱动

上面给大家已经介绍过，web系统的测试前提是需要打开浏览器，通过访问web服务器来对服务器界面进行一系列的操作。对于手工测试来说，这一系列的操作都需要测试人员手动的，一步一步的来执行测试。那么对于自动化程序来说，程序如何才能打开浏览器并执行我们预期的操作流程呢？

驱动一词应用广泛，同学们都不会陌生。

车有了驱动才能够让车跑起来。

计算机有了驱动程序就可以与设备（耳机，摄像头，麦克风，键盘，显示器等等设备）进行通信。



程序想要打开web浏览器就需要安装web驱动（即WebDriver)，WebDriver 以本地化方式驱动浏览器。

2.1.1 安装驱动管理

若通过安装驱动的方式来启动浏览器，每次浏览器更新后对应的驱动也需要更新，为了解决这个问题，selenium中提供了驱动管理工具webdriver-manager，有了webdriver-manager无需手动安装浏览器驱动，即使浏览器更新也不会影响自动化的执行。

> 命令：pip install webdriver-manager

```
C:\Users\mamian\AppData\Local\Programs\Python\Python39\Scripts>pip install webdriver-manager
Collecting webdriver-manager
  Downloading webdriver_manager-4.0.1-py2.py3-none-any.whl (27 kB)
Collecting python-dotenv
  Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Requirement already satisfied: requests in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (fro
m webdriver-manager) (2.31.0)
Requirement already satisfied: packaging in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (fr
om webdriver-manager) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\mamian\appdata\local\programs\python\python39\lib\si
te-packages (from packaging->webdriver-manager) (3.0.9)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-pac
kages (from requests->webdriver-manager) (2022.5.18.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages
(from requests->webdriver-manager) (3.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\mamian\appdata\local\programs\python\python39\lib\si
te-packages (from requests->webdriver-manager) (3.2.0)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-pac
kages (from requests->webdriver-manager) (1.26.9)
Installing collected packages: python-dotenv, webdriver-manager
Successfully installed python-dotenv-1.0.1 webdriver-manager-4.0.1
WARNING: You are using pip version 21.2.3; however, version 24.0 is available.
You should consider upgrading via the 'C:\Users\mamian\AppData\Local\Programs\Python\Python39\python.exe -m pip install
--upgrade pip' command.
```

驱动管理：

> webdriver-manager支持的python版本为：3.7~3.11

WebDriver Manager是一个开源的命令行工具，它可以自动下载和安装适用于不同浏览器的WebDriver。通过使用WebDriver Manager，我们可以确保浏览器驱动版本始终与浏览器版本保持一致，从而避免因版本不匹配而导致的各种问题。

2.1.2 selenium库

安装selenium库

> selenium版本很多，统一使用selenium 4.0.0版本
>
> pip install selenium==4.0.0

```
C:\Users\mamian\AppData\Local\Programs\Python\Python39\Scripts>pip install selenium==4.0.0
WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after connection broken by 'ReadTimeoutError("HTTPSConnectionPool(host='pypi.org', port=443): Read ti
med out. (read timeout=15)")': /simple/selenium/
Collecting selenium==4.0.0
  WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after connection broken by 'ConnectTimeoutError(<pip._vendor.urllib3.connection.HTTPSConnection obj
ect at 0x0000022C98AC0160>, 'Connection to files.pythonhosted.org timed out. (connect timeout=15)')': /packages/ad/24/39cab5fbaf425ff522ele51cce79f94f10f9523f015d2b2251e43f45e8a2/selenium-4
.0.0-py3-none-any.whl
  WARNING: Retrying (Retry(total=3, connect=None, read=None, redirect=None, status=None)) after connection broken by 'ConnectTimeoutError(<pip._vendor.urllib3.connection.HTTPSConnection obj
ect at 0x0000022C98AC03A0>, 'Connection to files.pythonhosted.org timed out. (connect timeout=15)')': /packages/ad/24/39cab5fbaf425ff522ele51cce79f94f10f9523f015d2b2251e43f45e8a2/selenium-4
.0.0-py3-none-any.whl
  Downloading selenium-4.0.0-py3-none-any.whl (954 kB)
     |████████████████████████████████| 954 kB 49 kB/s
Requirement already satisfied: trio~=0.17 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from selenium==4.0.0) (0.20.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from selenium==4.0.0) (0.9.2)
Requirement already satisfied: urllib3[secure]~=1.26 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from selenium==4.0.0) (1.26.9)
Requirement already satisfied: idna in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (3.3)
Requirement already satisfied: attrs>=19.2.0 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (21.4.0)
Requirement already satisfied: async-generator>=1.9 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (1.10)
Requirement already satisfied: sniffio in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (1.2.0)
Requirement already satisfied: outcome in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (1.1.0)
Requirement already satisfied: sortedcontainers in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (2.4.0)
Requirement already satisfied: cffi>=1.14 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio~=0.17->selenium==4.0.0) (1.15.0)
Requirement already satisfied: pycparser in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from cffi>=1.14->trio~=0.17->selenium==4.0.0) (2.21)
Requirement already satisfied: wsproto>=0.14 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from trio-websocket~=0.9->selenium==4.0.0) (1.1.0)
Requirement already satisfied: cryptography>=1.3.4 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from urllib3[secure]~=1.26->selenium==4.0.0) (37.0.2)
Requirement already satisfied: certifi in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from urllib3[secure]~=1.26->selenium==4.0.0) (2022.5.18.1)
Requirement already satisfied: pyOpenSSL>=0.14 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from urllib3[secure]~=1.26->selenium==4.0.0) (22.0.0)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\mamian\appdata\local\programs\python\python39\lib\site-packages (from wsproto>=0.14->trio-websocket~=0.9->selenium==4.0.0) (0.13.0)
Installing collected packages: selenium
  Attempting uninstall: selenium
    Found existing installation: selenium 4.1.5
    Uninstalling selenium-4.1.5:
      Successfully uninstalled selenium-4.1.5
Successfully installed selenium-4.0.0
WARNING: You are using pip version 21.2.3; however, version 24.0 is available.
You should consider upgrading via the 'C:\Users\mamian\AppData\Local\Programs\Python\Python39\python.exe -m pip install --upgrade pip' command.
```

3. Selenium

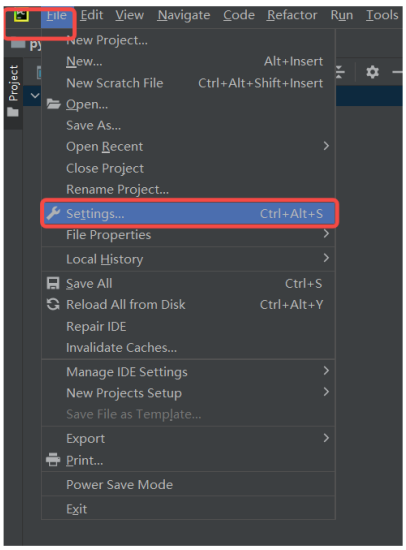
万事俱备，只欠东风。

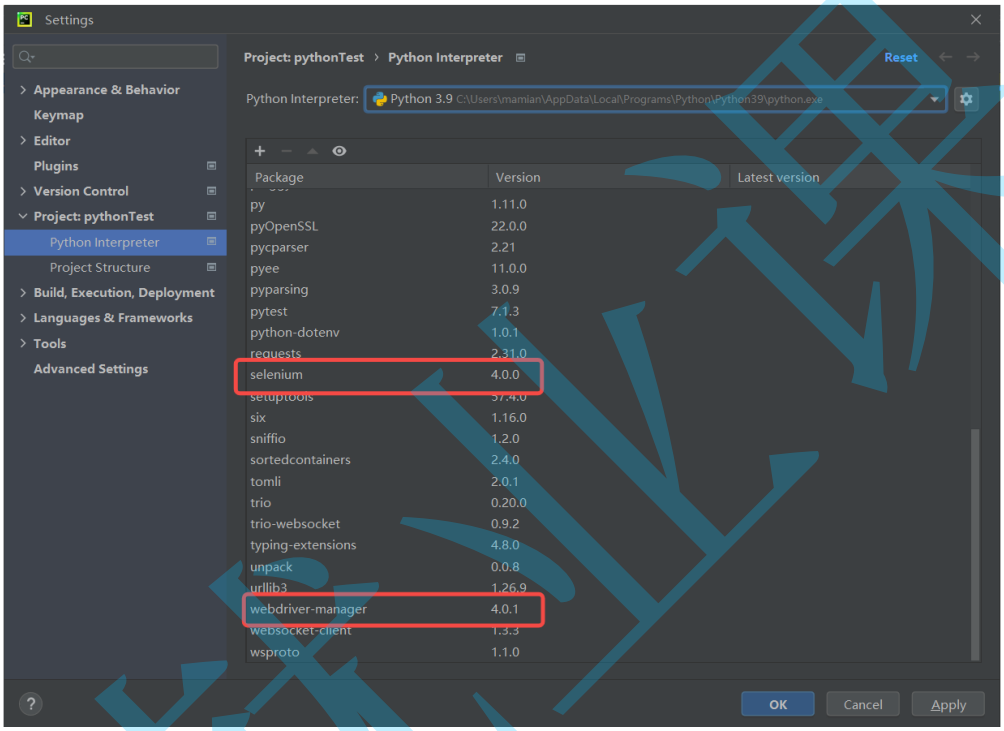
接下来就是使用selenium来编写web自动化测试脚本。

selenium是一个web自动化测试工具，selenium中提供了丰富的方法供给使用者进行web自动化测试。

3.1 一个简单的web自动化示例

1）检查python解释器，确定selenium库和WebDriverManager库都安装成功并加载到当前项目中





2）使用selenium编写代码

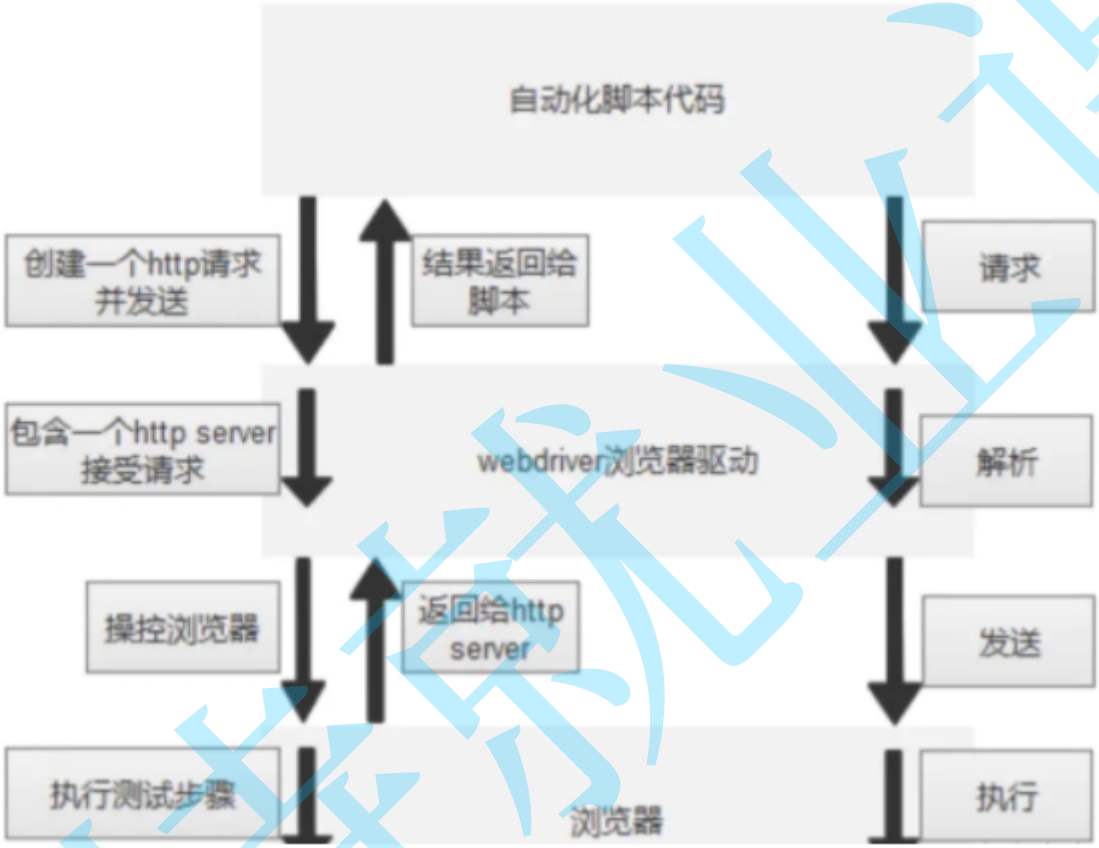
firstTest.py

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager

#驱动程序管理的自动化
#创建驱动对象
#1.打开浏览器
driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()))

#2.输入百度网址:https://www.baidu.com
driver.get("https://www.baidu.com")

#3、找到输入框并输入“迪丽热巴”
driver.find_element(By.CSS_SELECTOR,"#kw").send_keys("迪丽热巴")

#4、找到“百度一下”按钮并点击
driver.find_element(By.CSS_SELECTOR,"#su").click()
```

```
19
20 #5、关闭浏览器
21 driver.quit()
```

3.2 selenium+驱动+浏览器的工作原理

> 实现web自动化测试需要浏览器、浏览器驱动、selenium自动化脚本。这三者是如何交互最终实现web的自动化测试?



1. 通过selenium编写的自动化脚本代码中在ChromeDriverService中创建一个服务
2. 通过创建好的服务打开webdriver，安装在本地的驱动服务IP为localhost，PORT为ChromeDriverService中创建的端口号，该服务地址为selenium向webdriver发送请求的服务地址。
3. 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
4. 打开浏览器后，所有的selenium的操作(访问地址，查找元素等)均通过创建好的服务链接到webdriver，然后使用execute发送请求
5. 驱动收到请求并对请求进行解析，转成浏览器能够解析的脚本并发送给浏览器，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